POLITECNICO DI TORINO

Corso di laurea magistrale in Ingegneria Matematica

Tesi di laurea magistrale

Ottimizzazione dell'assortimento retail: confronto tra approcci stocastici e surrogati e l'impatto economico dei comportamenti d'acquisto

Relatore:
Prof. Paolo Brandimarte

Candidato:
Gaia Saroglia

Anno accademico 2025-2026

A nonno.

Abstract

Il lavoro presentato affronta il problema di ottimizzazione dell'assortimento e delle quantità da ordinare per un punto vendita di medie-piccole dimensioni sotto domanda stocastica e comportamento di acquisto dei clienti governato da sostituzioni a seguito delle scelte di primo livello.

Il modello di domanda considerato è un *Latent Class Choice Model* (LCCM) con sei classi di consumatori, calibrato su una categoria di prodotti costruita appositamente per questo studio (pasta secca, riso e gnocchi, per 111 opzioni totali). Gli arrivi a negozio seguono un processo di Poisson e la sostituzione è modellata dalla stessa dinamica fornita dal modello di scelta, con una variazione che valorizza una sostituzione vera rispetto che alla scelta della no-buy.

Sono implementati tre approcci di ottimizzazione basati su un problema stocastico a due stadi con ricorso. Il primo è il metodo *Sample Average Approximation, SAA*. Gli esperimenti condotti sono il risultato di 10 repliche indipendenti su 100 scenari, e forniscono la soluzione su cui verrano successivamente confrontati gli altri modelli: 1191 unità vendute in media su 300 scenari out-of-sample, livello di servizio rispetto alla domanda che arriva al negozio del 75.6% (che cala al 70% considerando anche i clienti che non entrano in negozio) al costo di 13 ore per replica. Il secondo tratta un modello surrogato che sfrutta l'apprendimento di un *Multi-Layer Perceptron* (MLP) rispetto alla funzione di ricorso: la qualità out-of-sample è tendenzialmente identica al SAA con 1182 unità vendute, pari a un -0.7% rispetto a SAA, con un tempo di ottimizzazione di pochi secondi per run, a fronte di un calcolo offline per generazione del dataset di addestramento e training della rete di circa 26 ore. Infine, il modello *Expected Value* deterministico con predittore Ridge ($R^2 = 0.51$) sulla domanda attesa, calcolato considerando due livelli di sostituzione, si rivela più in difficoltà in condizioni stocastiche: le vendite out-of-sample scendono a 1041 unità, ovvero una riduzione del 12.6% rispetto a SAA, e la no-buy complessiva sale (382

unità contro le 130 del SAA).

L'analisi quantitativa viene affiancata dall'interpretazione economica delle soluzioni. Negli scenari *out-of-sample* si identifica come collo di bottiglia il vincolo di capacità sullo scaffale Occhi e Alto. Infatti, essendo gli scaffali più ricercati, la preferenza dei clienti ricade su prodotti a quel livello di posizionamento. Tale fenomeno si riscontra in tutte le soluzioni proposte, e genera molte vendite perse. In scenari di stress test sulle distribuzioni dei segmenti di mercato, gli assortimenti proposti mostrano l'abilità di ciascuna soluzione di coprirsi o meno dal rischio. In particolare, per l'assortimento dell'ottimizzazione di SAA, abbiamo una diminuzione di vendite percentuale del 3.6%. Statistiche analoghe, leggermente degradate, le ritroviamo nel surrogato. Per il caso ibrido, invece, nello scenario più impattante, la diminuzione è del 5.1% e si nota come l'assortimento proposto tende a far aumentare il numero di no-buy option rispetto alla domanda insoddisfatta.

I risultati mostrano quindi l'inadeguatezza dell'approccio ibrido deterministico, mentre sottolineano l'ottima base di partenza proposta dal modello surrogato per ottimizzare l'assortimento in tempi ridotti. Inoltre, si evidenzia come lo studio economico sia utile nell'interpretazione finale dei dati matematici, al fine di poter inserire i commenti in un contesto reale.

Indice

Elenco delle figure **vii**

Elenco delle tabelle **ix**

1 Introduzione **1**

2 Fondamenti teorici **3**

2.1 Modelli di scelta discreti dei consumatori 3

2.2 Programmazione stocastica a due stadi 15

2.3 Il problema del valore atteso (EV) 20

2.4 Reti neurali artificiali 21

2.5 Il predittore lineare nel problema EV 30

3 Contesto e definizione del problema **32**

3.1 Il contesto di mercato 32

3.2 Il modello Latent Class utilizzato 33

3.3 Il dataset dei prodotti 41

3.4 Processo di arrivo dei clienti 45

3.5 Il problema di ottimizzazione 46

4 Dettagli di implementazione **59**

5 Analisi e commento ai risultati **62**

5.1 Modello SAA 62

5.1.1 Complessità computazionale e convergenza 62

5.1.2 Analisi dell'assortimento 66

5.1.3 Rimanenze, vendite perse e sostituzioni 68

5.2 Modello surrogato 70

5.2.1 Addestramento della rete neurale surrogata 70

5.2.2 Costo computazionale: surrogato vs SAA 72

5.2.3 Analisi dell'assortimento 73

5.3 Modello EV 75

5.3.1 Addestramento del predittore Ridge 75

5.3.2 Analsi dell'assortimento 77

5.4 Performance su scenari *out-of-sample* 79

5.4.1 Scenari di mercato 79

5.4.2 Scenari di stress 84

6 Conclusioni e sviluppi futuri **91**

Bibliografia **94**

Appendice A Documentazione dell'implementazione **96**

Elenco delle figure

2.1 Schema temporale programmazione stocastica a due stadi. 16

2.2 Schema di un perceptron. 22

2.3 Architettura di un MLP con due strati nascosti. 23

3.1 Struttura del modello Latent Class con $C = 6$ segmenti. 34

3.2 Test Drop Marca 40

3.4 Test Drop Prodotto 41

3.5 Test Drop Scaffale 42

3.6 Test Filtro Prezzo 42

3.7 Test Filtro Qualità e Drop Marca 43

3.8 Le due logiche di sostituzioni 56

5.1 Scenari di domanda. 63

5.2 Convergenza del gap Incumbent-BestBd nel tempo per due repliche SAA 65

5.3 Assortimento SAA 66

5.4 Vincoli strutturali di scaffalatura. 67

5.5 Esplorazione Optuna per il surrogato 72

5.6 Assortimento surrogato. 73

5.7 Distribuzione a scaffale dell'assortimento surrogato. 74

5.8 Curva di cross-validation per la selezione di α - modello Ridge. . . 76

5.9 Assortimento Hybrid EV. 77

5.10 Distribuzione a scaffale dell'assortimento ibrido. 78

5.11 Risultati assortimenti 81

5.13 Quali sono i prodotti più critici per ogni assortimento negli scenari di mercato out-of-sample. 84

5.15 Comportamneto dell'assortimento SAA in scenari di stress. 87

5.16 Comportamneto dell'assortimento Surrogato in scenari di stress. . . 89

5.17 Comportamneto dell'assortimento Ibrido in scenari di stress. 90

Elenco delle tabelle

3.1 Parametri del modello LCCM per i sei segmenti. 35
3.2 Capacità scaffali 44
3.3 Tassi di arrivo 45
3.4 Dimensioni del modello SAA. 54
3.5 Confronto modelli 58

5.1 Risultati SAA 64
5.2 Metriche assortimento SAA 69
5.3 Iperparametri ottimali 71
5.4 Confronto SAA-surrogato 73
5.5 Assortimento surrogato 74
5.6 Confronto surrogato e profitto reale 75
5.7 Addestramento Ridge 76
5.8 Risultati dell'ottimizzazione EV Ridge. 78
5.9 Performance out-of-sample 80
5.10 Performance stress-test 85

Capitolo 1

Introduzione

Consideriamo il contesto dell'ottimizzazione dell'assortimento nel settore retail, in particolare soffermiamoci sulla vendita di beni di largo consumo (ad esempio, prodotti alimentari molto amati nel nostro paese, come pasta, riso o gnocchi). In questo ambito è fondamentale valutare come il cliente prende le decisioni all'interno del negozio. Quando un cliente entra, lo fa con una preferenza rispetto a ciò ce vuole acquistare, l'esito della sua spesa è legato alla disponibilità del prodotto in questione. Se l'articolo è fisicamente presente sullo scaffale, l'acquisto si conclude e la domanda risulta soddisfatta a pieno. Diversamente, quindi se si dovesse verificare un out-of-stock, il consumatore si trova di fronte a un bivio: può decidere per la sostituzione dell'articolo mancante con un prodotto alternativo presente in assortimento, può posticipare l'acquisto nel medesimo negozio, oppure può abbandonare definitivamente lo store e andare a comprare la sua preferenza in un luogo diverso.

Ognuna di queste dinamiche del comportamento umano, in questo caso di un cliente, ha delle conseguenze, spesso critiche, sul rivenditore. In questo scenario, il problema dell'ottimizzazione dell'assortimento prende un ruolo centrale per dimiuire la domanda insoddisfatta e quindi eventi negativi per il negozio. Infatti, l'obiettivo del retailer è quello di determinare un insieme di prodotti e un livello di scorte ottimali, al fine di massimizzare il profitto e minimizzare i costi legati alle mancate vendite.

Per affrontare e risolvere tale livello di complessità, questa tesi propone un approccio risolutivo basato sull'ottimizzazione stocastica. Nello specifico, la domanda stocastica viene formalizzata attraverso un modello di scelta del consumatore. Tale

modello è costruito attraverso un Latent Class Choice Model che valuta sei segmenti di clientela, formati in modo specifico per catturare adeguatamente l'eterogeneità del mercato. A partire da questa struttura, verranno sviluppati tre modelli di ottimizzazione che andranno a definire l'assortimento ideale. La simulazione di questo scenario, modellerà matematicamente il processo decisionale del cliente, integrando le probabilità di sostituzione di primo grado in caso di out-of-stock. Infine, l'efficacia dei modelli proposti verrà valutata utilizzando i risultati sia in scenari di mercato normali, quindi quelli su cui sono ottimizati i risultati, sia in condizioni di stress della domanda. Si noti che tutte le soluzioni sono vincolate a parametri reali di limiti di capacità, la disposizione a scaffale e ulteriori dinamiche tipiche di un negozio.

Capitolo 2

Fondamenti teorici

In questo capitolo verranno introdotti e discussi, con diversi livelli di approfondimento, gli argomenti teorici alla base dello studio sperimenatale. In particolare, il capitolo è suddiviso in tre sezioni principali: la prima relativa ai modelli di scelta discreti, in cui verranno discusse le principali caratteristiche del modelli di *Random Utility* soffermandosi sui modelli *Multinomial Logit* e *Multinomial Nested Logit*. Nella sezione successiva, si tratterà la teoria relativa ai modelli di ottimizzazione stocastica. In particolare, tratteremo i modelli a due stadi caratterizzati da due livelli di decisione: in una prima fase si definiscono le scelte prima che si realizzi l'incertezza, e nella seconda si prendono le decisioni di ricorso, ovvero la risposta agli scenari che si realizzano. Infine, l'ultima sezione sarà dedicata alle reti neurali *feed-forward* e ai predittori lineari, con una introduzione teorica iniziale seguita dallo studio delle principali proprietà matematiche per giustificare il loro utilizzo successivo.

2.1 Modelli di scelta discreti dei consumatori

I modelli di scelta dei consumatori sono un elemento fondamentale per lo studio e l'ottimizzazione dell'assortimento. Essi ci permettono di formalizzare matematicamente il comportamento d'acquisto dei clienti, così da poter avere delle informazioni di base sulle dinamiche di merato. Come ogni modello matematico, tuttavia, anche la modellazione delle preferenze rappresenta un'approssimazione della realtà. La loro struttura si basa su specifiche assunzioni che ne facilitano l'interpretazione e la costruzione, ma portano con esse lo svantaggio di ridurre la fedeltà rispetto al

vero comportamento umano. Nonostante la consapevolezza dei limiti da cui sono caratterizzati, nelle analisi sperimentali che faremo nei successivi capitoli, useremo tali modelli come base teorica. Essi descrivono la probabilità che un individuo selezioni un'opzione da un insieme finito di alternative, a partire da un'utilità soggettiva associata a ciascuna di esse. Questo è esattamente ciò che serve nel caso di uno studio dell'assortimento: dato un catologo di prodotti, possiamo stabilire come la domanda di mercato si distribuisce tra essi, offrendo al rivenditore gli elementi per ottimizzare l'insieme dei prodotti proposti e massimizzare i profitti.

Il modello di Random Utility (RUM)

La base fondamentale dei modelli di scelta discreta proviene dagli studi di Daniel McFadden. [1] Il *modello di Random Utility* (RUM), è il risultato e la formalizzazione del suo lavoro. Ancora oggi è uno strumento molto usato non solo nell'ambito economico, ma in tutti quei contesti in cui poter modellare il comportamento delle scelte umane può avere un vantaggio.

Il RUM si fonda sull'assioma che, data un'alternativa, la sua utilità totale si divide in due componenti: una parte totalmente osservabile e una non osservabile. Quest'ultima modella l'imprevedibiltà del comportamento umano, gli errori che il cosumatore stesso fa quando vede un prodotto e sbaglia a fare le valutazioni rispetto alle sue caratteristiche e ingloba tutto ciò che l'anlista non riesce a includere esplicitamente nella parte deterministica (o sistematica).

Entrando nello specifico della formalizzazione matematica[2]:

Definizione 2.1.1 (Modello di Random Utility)**.** *Sia* $\mathscr{C} = \{1, 2, \ldots, J, 0\}$ *l'insieme delle alternative proposte al cliente n, dove* 0 *indica l'opzione di non acquisto (*no-buy*). Ciascuna alternativa* $j \in \mathscr{C}$ *ha un'utilità aleatoria associata*

$$U_{nj} = V_{nj} + \varepsilon_{nj}, \tag{2.1}$$

dove V_{nj} *è la componente* deterministica *(osservabile) e* ε_{nj} *è la perturbazione stocastica che cattura i fattori non osservabili. Il consumatore n seleziona l'alternativa* $j^* = \arg\max_{j \in \mathscr{C}} U_{nj}$.

La componente deterministica solitamente è definita come funzione lineare degli attributi del prodotto, ovvero quegli elementi che descrivono l'oggetto e che ne fanno

associare un valore da chi deve scegliere:

$$V_{nj} = \boldsymbol{\beta}^\top \mathbf{x}_{nj} = \sum_{k \in K} \beta_k x_{nj}^{(k)} \tag{2.2}$$

dove $\mathbf{x}_{nj} \in \mathbb{R}^K$ è il vettore degli attributi del prodotto $j \in \mathscr{C}$ che vengono valutati dal consumatore n (*ed esempio il prezzo, la qualità, ecc...*), con $x_{nj}^{(k)}$ componente k del vettore, e $\boldsymbol{\beta} \in \mathbb{R}^K$ è il vettore dei parametri da stimare, ovvero quanto pesa ogni elemento sulla scelta finale.

La probabilità che l'individuo n scelga l'opzione $j \in \mathscr{C}$ è quindi:

$$P_n(j) = P\big(U_{nj} \geq U_{nk},\ \forall k \in \mathscr{C}\big) = P\big(\varepsilon_{nk} - \varepsilon_{nj} \leq V_{nj} - V_{nk},\ \forall k \neq j\big)\,. \tag{2.3}$$

poichè il cliente andrà a scegliere l'opzione che gli fornisce l'utilità maggiore. L'individuo sceglierà il prodotto $j \in \mathscr{C}$ con una probabilità che dipende dalla realizzazione della componente stocastica; nello specifico equivale alla probabilità congiunta che la differenza tra i termini di errore di una alternativa $k \in \mathscr{C}$ rispetto ad un'altra possibilità $j \in \mathscr{C}$ non superi il delta tra le componenti deterministiche, ovvero quello che si guadagnerà per certo.

Possiamo quindi notare che (2.3) dipende dalla distribuzione congiunta delle perturbazioni $(\varepsilon_{nj})_{j \in \mathscr{C}}$. Questo ci porta necessariamente a introdurre delle assunzioni relativamente alla sua distribuzione. A seconda delle ipotesi fatte, si ottengono diversi modelli di scelta discreta.

Il modello Multinomial Logit (MNL)

Assumendo che le perturbazioni ε_{nj} siano indipendenti e identicamente distribuite secondo una Gumbel, si ottiene il modello *Multinomial Logit*

$$\varepsilon_{nj} \overset{\text{iid}}{\sim} \text{Gumbel}(0,1) \quad \Longrightarrow \quad P_n(j) = \frac{\exp(V_{nj})}{\sum_{k \in \mathscr{C}} \exp(V_{nk})}.$$

Definizione 2.1.2 (Distribuzione di Gumbel)**.** *Una variabile aleatoria ε è tale che $\varepsilon_{nj} \overset{\text{iid}}{\sim} \text{Gumbel}(\mu, \eta)$, ovvero segue una distribuzione Gumbel (nta anche come valore estremo di tipo I) con parametri (μ, η) con $\eta > 0$ se la sua funzione di distribuzione*

*cumulata (*CDF*) è*

$$F(\varepsilon) = \exp\left(-\exp\left(-\frac{\varepsilon-\mu}{\eta}\right)\right), \quad \varepsilon \in \mathbb{R}. \tag{2.4}$$

con μ che rappresenta la posizione, ovvero la mediana della distribuzione, e η che rappresenta la scala, ovvero la dispersione intorno alla mediana. All'aumentare della scala η le scelte tendono all'uniformità; al diminuire di $\eta \to 0$ la scelta tende al deterministico. Nel caso standard $\mu = 0$, $\eta = 1$, si ha $\mathbb{E}[\varepsilon] = \gamma_{\text{EM}} \approx 0.5772$ (costante di Euler-Mascheroni) e $\text{Var}[\varepsilon] = \pi^2/6$.

Teorema 2.1.1 (Derivazione del modello Multinomial Logit, McFadden)**.** *Nel modello* Random Utility (2.1)*, se le perturbazioni $\{\varepsilon_{nj}\}_{j\in\mathscr{C}}$ sono t.c. $\varepsilon_{nj} \overset{\text{iid}}{\sim}$* Gumbel$(\mu = 0, \eta = 1)$*, allora la probabilità che il consumatore n scelga l'alternativa $j \in \mathscr{C}$ è:*

$$P_n(j) = \frac{\exp(V_{nj})}{\sum_{k\in\mathscr{C}} \exp(V_{nk})}. \tag{2.5}$$

Dimostrazione. Per definizione del RUM, condizionando su $\varepsilon_{nj} = \varepsilon$:

$$\begin{aligned} P_n(j) &= \int_{-\infty}^{+\infty} P\big(\varepsilon_{nk} \le \varepsilon + V_{nj} - V_{nk},\ \forall k \neq j \;\big|\; \varepsilon_{nj} = \varepsilon\big)\, f(\varepsilon)\, d\varepsilon \\ &= \int_{-\infty}^{+\infty} \left(\prod_{k\neq j} F\big(\varepsilon + V_{nj} - V_{nk}\big)\right) f(\varepsilon)\, d\varepsilon. \end{aligned} \tag{2.6}$$

Con la distribuzione di Gumbel standard si ha $F(t) = e^{-e^{-t}}$ e $f(t) = e^{-t}e^{-e^{-t}}$. Da cui otteniamo:

$$\begin{aligned} \prod_{k\neq j} F\big(\varepsilon + V_{nj} - V_{nk}\big) &= \exp\left(-\sum_{k\neq j} e^{-(\varepsilon + V_{nj} - V_{nk})}\right) \\ &= \exp\left(-e^{-\varepsilon}\sum_{k\neq j} e^{V_{nk} - V_{nj}}\right). \end{aligned} \tag{2.7}$$

Includendo il termine $k = j$ nella somma:

$$\sum_{k\neq j} e^{V_{nk}-V_{nj}} + 1 = e^{-V_{nj}} \sum_{k\in\mathscr{C}} e^{V_{nk}} =: A \cdot e^{-V_{nj}},$$

dove $A \coloneqq \sum_{k\in\mathscr{C}} e^{V_{nk}}$. Quindi:

$$\begin{aligned} P_n(j) &= \int_{-\infty}^{+\infty} \exp\left(-e^{-\varepsilon} A e^{-V_{nj}}\right) e^{-\varepsilon} e^{-e^{-\varepsilon}} d\varepsilon \\ &= \int_{-\infty}^{+\infty} \exp\left(-e^{-\varepsilon}\left(A e^{-V_{nj}} + 1\right)\right) e^{-\varepsilon}\, d\varepsilon. \end{aligned} \tag{2.8}$$

Ponendo $t = e^{-\varepsilon} \cdot A e^{-V_{nj}}$, ovvero $d\varepsilon = -dt/(t)$ e $e^{-\varepsilon} = t\, e^{V_{nj}}/A$, abbiamo:

$$P_n(j) = \int_0^{+\infty} e^{-t} \cdot \frac{1}{A}\, dt = \frac{1}{A} = \frac{e^{V_{nj}}}{\sum\limits_{k\in\mathscr{C}} e^{V_{nk}}},$$

che conclude la dimostrazione. □

Proprietà fondamentali del modello MNL

Entrando nello specifico del modello MNL, questo è caratterizzato da alcune proprietà importanti, che portano con loro sia i suoi principali punti di forza sia i suoi limiti più impattanti.

Proprietà 2.1.1 (Indipendenza dalle Alternative Irrilevanti — IIA)**.** *Nel modello MNL, il rapporto tra le probabilità di due alternative j e k è indipendente dall'utilità di tutte le altre alternative $\ell \notin \{j,k\} \in \mathscr{C}$:*

$$\frac{P_n(j)}{P_n(k)} = \exp\left(V_{nj} - V_{nk}\right). \tag{2.9}$$

Dimostrazione. Segue direttamente da (2.5): il termine $\sum_{k\in\mathscr{C}} e^{V_{nk}}$ si semplifica, lasciando solo $e^{V_{nj}-V_{nk}}$. □

Questo significa che, se ad esempio si raddoppia l'utilità di un'alternativa j, allora raddoppia anche la sua probabilità, mentre le probabilità di tutte le altre alternative diminiuscono proporzionalmente in modo da mantenere la somma totale pari a 1. La proprietà IIA rappresenta contemporaneamente il punto di forza e il limite principale del modello MNL. Da un lato, ci dice che aggiungere o rimuovere un'alternativa modifica le probabilità di tutte le altre in modo proporzionale, e questo rende il modello facilmente interpretabile. Dall'altro, implica che le opzioni vengono trattate come *equally competitive*: due alternative con attributi simili (e quindi

potenzialmente dei sostituti stretti) competono nell'essere scelte dal consumatore nella stessa misura di due alternative con attributi distanti (quindi che potenzialmente un cliente con preferenza per una non ce l'ha per l'altra). L'esempio più esplicativo del grande limite che porta questa proprietà è il paradosso dell'autobus rosso e blu (*red bus/blue bus paradox*) [2].

Nel nostro modello di mercato, valutando il catalogo disponibile, questa proprietà è leggermente limitante, considerando la volontà di modellare le sostituzioni. Si mantiene però la scelta dell'utilizzo di tale modello di scelta per poter usufruire della sua semplicità. Una modellazione più completa avrebbe considerato i gruppi di sostituzione, valutando un *Nested Logit Model*: in questo caso la dinamica di sostituzione sarebbe stata più veritiera, ma la difficoltà nel definire la classi dei nidi era più svantaggiosa rispetto che il vantaggio di catturare perfettamente le scelte, data l'alta complessità dei legami tra prodotti presenti nel catalogo.

Un'altra caratteristica tipica del modello Multinomial Logit è quella della normalizzazione. Formalmente questa ci dice che:

Proprietà 2.1.2 (Normalizzazione)**.** *Nel modello MNL è necessario fissare un riferimento per tutte le utilità. Tipicamente si pone un'alternativa di base con* $V_{n,ref} = 0$.

La proprità della normalizzazione è una diretta conseguenza del fatto che la probabilità di scelta dipende solo dalla differenza tra le utilità. Se si aggiunge una costante c a tutte le parti deterministiche dell'utilità, si ha:

$$P_n(j) = \frac{e^{V_{nj}+c}}{\sum_k e^{V_{nk}+c}} = \frac{e^{V_{nj}}e^{c}}{e^{c}\sum_k e^{V_{nk}}} = \frac{e^{V_{nj}}}{\sum_k e^{V_{nk}}},$$

quindi la probabilità di scelta rimane invariata. Ciò ci porta a notare che i parametri stimati sono da interpretare come differenze di utilità rispetto al riferimento. Nel nostro caso, il modello rappresenta le scelte dei clienti, per questo motivo viene fissata come alternativa di riferimento l'opzione *no-buy*, essendo quella sempre presente.

Concludiamo con l'ultima proprietà tipica del modello MNL:

Proprietà 2.1.3 (Elasticità propria e incrociata)**.** *L'elasticità della probabilità di*

scelta $P_n(j)$ rispetto all'attributo $x_{nj}^{(m)}$ è:

$$e^{\frac{P_{nj}}{x_{nj}^{(m)}}} = \beta_m x_{nj}^{(m)} \left(1 - P_n(j)\right), \tag{2.10}$$

mentre l'elasticità incrociata rispetto all'attributo $x_{nk}^{(m)}$ di un'alternativa diversa $k \neq j \in \mathscr{C}$ è:

$$e^{\frac{P_{nj}}{x_{nk}^{(m)}}} = -\beta_m x_{nk}^{(m)} P_n(k). \tag{2.11}$$

Si nota che l'elasticità incrociata dipende solo da $P_n(k)$ e non dalle caratteristiche di j; questo deriva direttamente dalla proprietà IIA [3].

Dovendo dare un'interpretazione al significa dell'elasticità, questa misura quanto cambia la probabilità di scelta di un'alternativa j rispetto a una variazione dell'attributo $x_{nj}^{(m)}$. Se $\beta_m > 0$ (quindi l'attributo ha un impatto positivo sul cliente), allora è positiva, altrimenti è negativa. Questo ci dice che un miglioramento dell'attributo $x_{nj}^{(m)}$ aumenta la probabilità che j venga scelto, ma l'effetto diminuisce all'aumentare di $P_n(j)$ siccome la probabilità deve sempre sommare a 1. L'elasticità incrociata misura invece come viene influenzata la probabilità di scelta di j se si va a cambiare un attributo di un'altra alternativa k. Notiamo che questa risulta sempre negativa, dal momento che se l'alternativa k migliora, allora la probabilità di scegliere j si riduce. Questo implica che l'effetto di una variazione in un'alternativa k è lo stesso su tutte le altre alternative $j \neq k \in \mathscr{C}$, indipendentemente dalle loro caratteristiche. Questa proprietà ci permette di dare un'interpretazione economica del modello di scelta che andremo a costruire nei capitoli successivi e che useremo per le analisi sperimentali.

Stima per massima verosimiglianza

Come accennato nei paragrafi precedenti, il modello è caratterizzato dal vettore $\boldsymbol{\beta} \in \mathbb{R}^K$ dei parametri. Tali coefficienti possono essere stimati a partire dai dati disponibili. Nel modello MNL tale stima è fatta sfruttando il principio di *massima verosimiglianza* (MLE). Date N osservazioni, ciascuna relativa a un consumatore che sceglie $j_n \in \mathscr{C}$, la log-verosimiglianza è:

$$LL(\boldsymbol{\beta}) = \sum_{n=1}^{N} \log P_n(j_n; \boldsymbol{\beta}) = \sum_{n=1}^{N} \left[V_{nj_n} - \log \sum_{k \in \mathscr{C}} e^{V_{nk}} \right]. \tag{2.12}$$

Teorema 2.1.2 (Concavità della log-verosimiglianza MNL)**.** *La funzione $LL(\boldsymbol{\beta})$ definita in (2.12) è concava in $\boldsymbol{\beta}$.*

Dimostrazione (sketch). Si calcola la matrice Hessiana $H(\boldsymbol{\beta}) = \nabla^2_{\boldsymbol{\beta}} LL(\boldsymbol{\beta})$. Il termine negativo è

$$-\sum_{n=1}^{N}\left[\sum_k P_n(k)\,\mathbf{x}_{nk}\mathbf{x}_{nk}^\top - \left(\sum_k P_n(k)\,\mathbf{x}_{nk}\right)\left(\sum_k P_n(k)\,\mathbf{x}_{nk}\right)^\top\right],$$

che corrisponde a $-\sum_n \mathrm{Var}_n[\mathbf{x}_{nj}]$, ovvero la varianza del vettore degli attributi sotto la distribuzione di probabilità definita dal modelo. Il fatto che sia negativa, visto che l'Hessiana è semidefinita negativa, garantentisce la concavità di *LL* [2]. Potendo quindi stimare i parametri in modo univoco, possiamo assicurare l'esistenza di un unico massimo globale. □

Sotto condizioni di regolarità standard, con un numero suficiente di campioni, lo stimatore MLE è caratterizzato da proprietà asintotiche che garantiscono la sua ottimalità nell'ambito dei modelli di scelta discreta. Ai fini della tesi, non verrà ulteriormente approfondita la stima e le proprietà asintotiche del MLE, dal momento che non è centrale rispetto agli obiettivi dell'analisi condotta.

Il modello Latent Class Choice (LCCM)

Siccome soffermarsi semplicemente sul modello Multinomial Logit risulterebbe limitante per cercare di rappresentare, almeno approssimativamente, delle dinamiche di mercato reali, bisogna introdurre un modello di scelta discreta che consenta di catturare l'eterogeneità delle preferenze dei consumatori, ovvero il *Latent Class Choice Model* (LCCM). Questo modello è particolarmente interessante dal momento che sfrutta la segmentazione del mercato definendo dei gruppi omogenei caratterizzati da individui con comportamenti di scelta simili. Tale modello è stato scelto per essere implementato nella tesi, al fine di simulare scenari di mercato più realistici che permettessero di analizzare le diverse tipologie di preferenza.

Definizione 2.1.3 (Modello Latent Class Choice)**.** *Sia C il numero di segmenti (classi latenti). Per ogni segmento $c \in \{1,\dots,C\}$, sia $\boldsymbol{\beta}_c \in \mathbb{R}^K$ il vettore dei parametri*

specifico del segmento. Le probabilità di appartenere ad un segmento sono tali che $\pi_c \geq 0$ *e* $\sum_{c=1}^{C} \pi_c = 1$. *La probabilità che un consumatore scelga l'alternativa* j *è:*

$$P(j) = \sum_{c=1}^{C} \pi_c P_c(j), \tag{2.13}$$

dove $P_c(j) = \exp(V_{cj}) / \sum_{k \in \mathscr{C}} \exp(V_{ck})$ *è la probabilità MNL del segmento c.*

Quindi la scelta del prodotto è pesata attraverso la proporzione delle classi, rispetto alle scelte specifiche di ogni segmento disponibile.

Considerando principalmente i modelli di scelta discreta nell'ambito del retail, introduciamo di seguito alcuni degli elementi tipici che lo caratterizzano dal punto di vista economico e matematico:

La segmentazione del mercato

Nella letteratura di marketing e retail analytics, ci sono diversi studi relativi alla segmentazione del mercato. In questi, i ricercatori definiscono i segmenti di mercato sulla base di dati reali relativi a caratteristiche dei clienti [4]. Le macro categorie sono:

1. Aspetti geografici: dividono il mercato in unità geografiche come la nazione, la regione, la città. Questo tipo di *clusterizzazione*, permette di creare dei programmi di mercato *ad-hoc*, seguendo l'indicazione dei bisogni locali;
2. Aspetti demografici: i segmenti si ottengono in base all'età, la dimensione del nucleo familiare, il sesso, il patrimonio, l'educazione, l'occupazione, la religione, la classe sociale. Sono molto usati perchè sono facilmente misurabili e riconducibili a bisogni e necessità del consumatore;
3. Aspetti psicologici: si focalizzano sui tratti psicologi e di personalità, stile di vita o valori personali. In questo gruppo, si modellano le preferenze rispetto alle marche; la priorità al prezzo o alla qualità; e tutti quegli aspetti relativi.

Lo scopo della segmentazione è riuscire ad aggiunstare i programmi di marketing in modo tale da renderli il più adatti alle differenze che caratterizzano i consumatori. Soffermandoci principalmente sull'ultima classificazione, si riscontrano tipicamente segmenti riconducibili a tre categorie fondamentali:

1. **Fedeltà alla marca** (*brand loyalty*): consumatori con preferenza per uno specifico marchio, non tanto sensibili al prezzo e alla posizione sullo scaffale. Si possono definire più livelli di fedeltà: questa caratteristica è catturata da un coefficiente γ_{marca} molto elevato in positivo per la marca preferita e fortemente negativo per le concorrenti a seconda del grado di preferenza sulla marca definita.
2. **Sensibilità al prezzo** (*price sensitivity*): consumatori che riducono la loro utilità all'aumnetare del prezzo unitario. Il coefficiente θ associato al prezzo è in valore assoluto superiore rispetto agli altri segmenti.
3. **Sensibilità alla qualità** (*quality sensitivity*): consumatori che privilegiano attributi qualitativi del prodotto. Un modo per modellare la non linerità relativa a questi attributi è ad esempio, suddividere qualità bassa e qualità alta, e sfruttare due coefficienti diversi ν^- e ν^+ distinti, dove $|\nu^+| > |\nu^-|$.

L'opzione di non acquisto (*no-buy*)

Come accennato nel modello MNL 2.1, è importante definire nell'insieme delle alternative $\mathscr{C}$ un'opzione di non acquisto (opzione $j = 0$), che rappresenta la scelta di non acquistare nulla dall'assortimento disponibile. In letteratura questa opzione è detta *no-buy option* [2].

Dal punto di vista matematico, affinchè il modello di scelta sia valido, il *choice set* deve rispettare tre proprità: le opzioni devono essere *mutuamente esclusive, in numero finito ed esaustive*. Se l'opzione di non acquisto non fosse considerata, allora la probabilità di scelta non sommerebbe a 1.

Dal punto di vista economico, l'inclusione dell'opzione di non acquisto è fondamentale per spiegare le decisioni del consumatore. Quando un prodotto non è disponibile, il cliente può reagire sostituendo il prodotto con un'altra alternativa (Sezione 2.1) oppure decidere di uscire senza acquistare nulla. Chi compie questa scelta, viene assorbito nella *no-buy*. La letteratura legata alla psicologia e alla percezione del consumatore è estremamente vasta. Nel contesto del retail, l'ambiente gioca un ruolo fondamentale nelle risposte del cliente [5]. La definizione della *no-buy option* ci viene in aiuto per distinguere due situazioni qualitativamente diverse:

1. **Scelta prima di entrare nel negozio** (*channel choice*): il consumatore decide *se* entrare nel negozio e *quale prodotto* acquistare. In questo caso, la probabilità

di non acquisto include anche la probabilità di non visitare il negozio.

2. **Scelta all'interno del negozio** (*in-store choice*): il consumatore è *già all'interno del punto vendita* e sceglie tra i prodotti esposti sullo scaffale. In questo caso, la decisione di entrare è già stata presa: quindi il cliente all'interno ha maggiore predisposizione verso la scelta di acquistare qualcosa.

Questa distinzione ha importanti implicazioni per la modellazione delle *sostituzioni di prodotto*. Un consumatore già in negozio, di fronte all'assenza del suo prodotto preferito, è più propenso a prendere un prodotto che vada a sostituire la sua prima scelta. Un consumatore che deve ancora decidere se entrare nel negozio probabilmente è meno incline alla sostituzione se sa già da catalogo che non c'è il suo prodotto.

Definizione 2.1.4 (Utilità dell'opzione di non acquisto condizionata all'in-store)**.** *Sia* $\bar{V}_0 < 0$ *un valore di utilità ridotta per l'opzione di non acquisto nel contesto in-store. La probabilità di non acquisto condizionata alla presenza in negozio è:*

$$P_n^{\text{in-store}}(0) = \frac{e^{\bar{V}_0}}{e^{\bar{V}_0} + \sum_{j=1}^{J} e^{V_{nj}}}, \tag{2.14}$$

con $P_n^{\text{in-store}}(0) < P_n(0)$ *per ogni* $\bar{V}_0 < 0$.

Questa differenziazione è importante da segnalare nel calcolo della matrice delle sostituzioni, dove si modella precisamente il comportamento di un consumatore in-store che non trova il proprio prodotto preferito.

La Matrice di Sostituzione

Come detto, se un prodotto $j \in \mathscr{C}$ è esaurito o escluso dall'assortimento, la sua domanda viene ridistribuita: una quota di consumatori che preferivano j può acquistare un prodotto $k \neq j \in \mathscr{C}$ disponibile nell'assortimento.

Definizione 2.1.5 (Matrice delle sostituzioni)**.** *La matrice delle sostituzioni* $S \in [0,1]^{(J+1)\times(J+1)}$ *è definita da:*

$$S_{j\to k} = S_{jk} = P(\textit{il consumatore sceglie } k \mid j \notin \mathscr{A}) = \frac{e^{V_k}}{\sum_{i\in\mathscr{C}\setminus\{j\}} e^{V_i}}, \quad k \neq j, \quad (2.15)$$

dove $\mathscr{A}$ *è l'assortimento disponibile e* $\mathscr{C} \setminus \{j\}$ *indica l'insieme delle alternative senza il prodotto* j*.*

Ogni riga j della matrice S corrisponde a una distribuzione di probabilità su $\mathscr{C} \setminus \{j\}$: dato che il prodotto j è indisponibile, S_{jk} esprime la quota di consumatori che avevano scelto j e ora si spostano verso k.

Nel contesto del LCCM, la matrice delle sostituzioni aggregata è la media pesata delle matrici di sostituzione calcolate per ciascun segmento:

$$\bar{S}_{jk} = \sum_{c=1}^{C} \pi_c S_{jk}^{(c)}, \quad (2.16)$$

dove $S_{jk}^{(c)}$ è la probabilità di sostituzione del segmento c, calcolata a partire dalle probabilità MNL del segmento con il prodotto j rimosso dall'insieme delle alternative.

Nell'applicazione alla sostituzione dopo essere entrati in negozio, le utilità sono calcolate con l'utilità dell'opzione di non acquisto ridotta come descritto in 2.1.4.

Osservazione 2.1.1 (Connessione con la proprietà IIA)**.** *La proprietà IIA del modello MNL implica che la rimozione del prodotto* j *ridistribuisce la domanda di* j *proporzionalmente alle probabilità degli altri prodotti. Formalmente:*

$$S_{jk} = \frac{P_n(k)}{1 - P_n(j)}, \quad \forall k \neq j. \quad (2.17)$$

Questa proprietà, pur essendo una conseguenza diretta della struttura del modello, risulta particolarmente adatta alla modellazione della sostituzione in contesti dove è possibile comparare i prodotti.

2.2 Programmazione stocastica a due stadi

Il lavoro che presenteremo si colloca direttamente nel framework della *programmazione stocastica*: le decisioni devono essere prese prima che la domanda si realizzi, ma il loro valore è determinato dalla realizzazione degli scenari di domanda. Per ottimizzare un assortimento di un negozio e valutare la quantità da inserire a scaffale, bisogna esattamente valutare questa distinzione tra livelli temporali. Le scelte prese produrranno valore con la realizzazione dell'incertezza, successivamente si tratterà di valutare come gestire le risorse a scaffale al fine di massimizzare il profitto.

Formulazione generale

Definizione 2.2.1 (Programmazione stocastica a due stadi con ricorso)**.** *Sia* $(\Omega, \mathscr{F}, \mathbb{P})$ *uno spazio di probabilità e* $\boldsymbol{\xi} : \Omega \to \mathbb{R}^m$ *un vettore aleatorio che rappresenta l'incertezza. Un problema di* programmazione stocastica a due stadi con ricorso *è:*

$$\min_{\mathbf{x}\in\mathscr{X}} \left\{ f(\mathbf{x}) + \mathbb{E}_{\boldsymbol{\xi}}[Q(\mathbf{x}, \boldsymbol{\xi})] \right\}, \tag{2.18}$$

dove:

- $\mathscr{X} \subseteq \mathbb{R}^n$ *è l'* insieme delle soluzioni ammissibili del primo stadio*;*
- $f : \mathbb{R}^n \to \mathbb{R}$ *è il* costo del primo stadio*, noto al momento della decisione;*
- $Q(\mathbf{x}, \boldsymbol{\xi})$ *è il* valore ottimo del secondo stadio *(funzione di ricorso):*

$$Q(\mathbf{x}, \boldsymbol{\xi}) = \min_{\mathbf{y}\in\mathscr{Y}(\mathbf{x},\boldsymbol{\xi})} g(\mathbf{y}, \boldsymbol{\xi}), \tag{2.19}$$

$$\textit{con } \mathscr{Y}(\mathbf{x}, \boldsymbol{\xi}) = \{\mathbf{y} \in \mathbb{R}^{n_2} : W\mathbf{y} \leq h(\boldsymbol{\xi}) - T(\boldsymbol{\xi})\mathbf{x},\ \mathbf{y} \geq 0\} \textit{ e } g(\mathbf{y}, \boldsymbol{\xi}) = \boldsymbol{q}(\boldsymbol{\xi})^\top \mathbf{y}.$$

Le matrici $W \in \mathbb{R}^{m_2 \times n_2}$*,* $T(\boldsymbol{\xi}) \in \mathbb{R}^{m_2 \times n}$ *e* $h(\boldsymbol{\xi}) \in \mathbb{R}^{m_2}$ *definiscono i* vincoli di accoppiamento *tra primo e secondo stadio.*

Nei casi più complessi anche la matrice W potrebbe dipendere dalla realizzazione degli scenari di rischio, ma questa caso non è rilevante per il lavoro proposto.

Come detto, temporalmente il problema si divide come segue: la decisione $\mathbf{x}$ del primo stadio viene presa *prima* della realizzazione di $\boldsymbol{\xi}$; una volta osservato $\boldsymbol{\xi}$, la

decisione $\mathbf{y}$ del secondo stadio (dette variabili di *ricorso*) ottimizza la funzione dato $\mathbf{x}$ e $\boldsymbol{\xi}$.

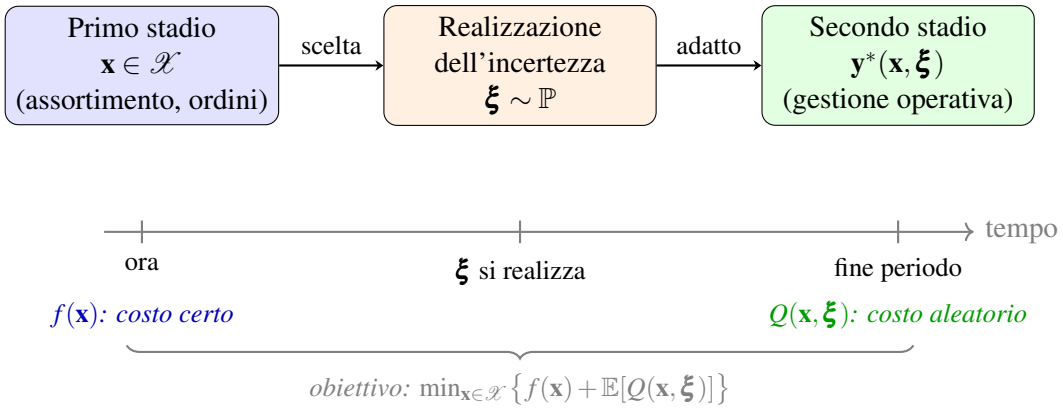


Figura 2.1 Schema temporale programmazione stocastica a due stadi.

Osservazione 2.2.1. *Il problema di secondo stadio* (3.13) *è detto* a ricorso completo *(*complete recourse*) se il problema del secondo stadio è ammissibile per ogni soluzione proposta de primo stadio. Questo vuol dire che la realizzazione della stocasticità non va a cambiare l'ammissibilità del secondo stadio. Per quanto riguarda il nostro problema, andando a considerare le variabili di lost-sales e di rimanenze, siamo certi del fatto che il ricorso sia completo, poichè avremo sempre una soluzione al secondo stadio, qualunque sia la scelta del primo.*

La funzione di ricorso $Q(\mathbf{x}, \boldsymbol{\xi})$ e il suo valore atteso $\mathscr{Q}(\mathbf{x}) := \mathbb{E}_{\boldsymbol{\xi}}[Q(\mathbf{x}, \boldsymbol{\xi})]$ sono funzioni convesse. Questa proprietà è fondamentale per la risoluzione del problema stocastico. In particolare:

Teorema 2.2.1 (Convessità della funzione di ricorso)**.** *Sia il problema di secondo stadio* (3.13) *un problema di programmazione lineare LP con* $\mathscr{Y}(\mathbf{x}, \boldsymbol{\xi}) = \{\mathbf{y} \geq 0 : W\mathbf{y} \leq h(\boldsymbol{\xi}) - T(\boldsymbol{\xi})\mathbf{x}\}$. *Per ogni realizzazione* $\boldsymbol{\xi}$ *della variabile aleatoria, la funzione* $Q(\cdot, \boldsymbol{\xi})$ *è* convessa *in* $\mathbf{x}$ *sul dominio* $\{\mathbf{x} : \mathscr{Y}(\mathbf{x}, \boldsymbol{\xi}) \neq \emptyset\}$.

Dimostrazione. Per ogni coppia $\mathbf{x}_1, \mathbf{x}_2$ e ogni $\lambda \in [0,1]$, siano $\mathbf{y}_1^*$ e $\mathbf{y}_2^*$ le soluzioni ottime corrispondenti. La soluzione $\mathbf{y}_\lambda = \lambda\mathbf{y}_1^* + (1-\lambda)\mathbf{y}_2^*$ è ammissibile considerando la combinazione convessa $\mathbf{x}_\lambda = \lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2$, poiché:

$$W\mathbf{y}_\lambda = \lambda W\mathbf{y}_1^* + (1-\lambda)W\mathbf{y}_2^* \leq \lambda(h - T\mathbf{x}_1) + (1-\lambda)(h - T\mathbf{x}_2) = h - T\mathbf{x}_\lambda.$$

Essendo $\mathbf{y}_\lambda$ subottimale:

$$Q(\mathbf{x}_\lambda, \boldsymbol{\xi}) \leq \boldsymbol{q}^\top \mathbf{y}_\lambda = \lambda \boldsymbol{q}^\top \mathbf{y}_1^* + (1-\lambda)\boldsymbol{q}^\top \mathbf{y}_2^* = \lambda Q(\mathbf{x}_1, \boldsymbol{\xi}) + (1-\lambda) Q(\mathbf{x}_2, \boldsymbol{\xi}). \quad \square$$

Corollario 2.2.2 (Convessità del valore atteso del ricorso)**.** *La funzione* $\mathscr{Q}(\mathbf{x}) = \mathbb{E}_{\boldsymbol{\xi}}[Q(\mathbf{x}, \boldsymbol{\xi})]$ *è convessa in* $\mathbf{x}$*.*

Dimostrazione. Segue immediatamente dalla convessità per ogni $\boldsymbol{\xi}$ (Teorema 2.2.1) e dalla linearità del valore atteso. $\square$

Osservazione 2.2.2 (Implicazioni per i problemi misti interi)**.** *Bisogna osservare però che in caso di variabili intere o binarie al primo stadio, come nel nostro caso, otteniamo una problema di* Programmazione Lineare Misto Intero Stocastico*.In questo caso, la convessità di* $\mathscr{Q}$ *non implica convessità della funzione obiettivo generale e questo rende il problema più difficie da risolvere rispetto alla formulazione continua, richiedendo tecniche di branch-and-bound o decomposizione. All'interno del software di Gurobi che viene utilizzato nel lavoro presentato, queste tecniche sono già implementate, per aiutare nella risoluzione dei problemi proposti. In ogni caso, la convessità è fondamentale nella risoluzione del problema rilassato, in cui tute le variabili vengono considerate continue, che è uno step fondamentale per ottenere la soluzione.*

Sample Average Approximation (SAA)

Solitamente, in questo tipo di problemi l'incertezza $\boldsymbol{\xi} : \Omega \to \mathbb{R}^m$ viene modellata tramite un albero di scenari, ovvero dicretizzando in un insieme finito il fattore di rischio. Si può quindi usare il metodo della *Sample Average Approximation* (SAA) che va a sostituire il valore atteso con la media campionaria su un insieme finito di N scenari $\{\boldsymbol{\xi}^{(s)}\}_{s=1}^N$ campionati i.i.d. da $\mathbb{P}$ dell'incertezza:

$$\min_{\mathbf{x}\in\mathscr{X}} \left\{ \hat{z}_N(\mathbf{x}) \coloneqq f(\mathbf{x}) + \frac{1}{N}\sum_{s=1}^{N} Q\left(\mathbf{x}, \boldsymbol{\xi}^{(s)}\right) \right\}. \tag{2.20}$$

Sia $z^* = \min_{\mathbf{x}\in\mathscr{X}}\{f(\mathbf{x}) + \mathscr{Q}(\mathbf{x})\}$ il valore ottimo del problema di partenza (2.18) e $\hat{z}_N^* = \min_{\mathbf{x}\in\mathscr{X}} \hat{z}_N(\mathbf{x})$ il valore ottimo SAA. Il teorema fondamentale della SAA [6], direttamente dalla Legge dei Grandi Numeri garantisce che:

Teorema 2.2.3. *Supponiamo che:*

(i) $\mathscr{X}$ *sia compatto;*
(ii) $Q(\mathbf{x}, \boldsymbol{\xi})$ *sia continua in* $\mathbf{x}$ *per quasi ogni* $\boldsymbol{\xi}$ *e misurabile in* $\boldsymbol{\xi}$ *per ogni* $\mathbf{x}$*;*
(iii) $\mathbb{E}[\sup_{\mathbf{x}\in\mathscr{X}} |Q(\mathbf{x}, \boldsymbol{\xi})|] < +\infty$.

Allora, con probabilità 1:

$$\lim_{N\to\infty} \hat{z}_N^* = z^*. \tag{2.21}$$

Inoltre, ogni punto di accumulazione della successione $\{\hat{\mathbf{x}}_N^*\}$ *di soluzioni ottime, SAA è soluzione ottima del problema originale.*

Teorema 2.2.4 (Tasso di convergenza SAA)**.** *Sotto le ipotesi del Teorema 2.2.3, e se* $Q(\mathbf{x}, \boldsymbol{\xi})$ *ha varianza finita* $\sigma^2(\mathbf{x}) \leq \sigma^2 < +\infty$ *per ogni* $\mathbf{x}$*, allora lo stimatore* $\hat{z}_N^*$ *è asintoticamente non distorto e:*

$$\mathrm{Var}[\hat{z}_N^*] = O\left(\frac{1}{N}\right), \qquad \hat{z}_N^* - z^* = O_p\left(\frac{1}{\sqrt{N}}\right). \tag{2.22}$$

Inoltre, per ogni $\varepsilon > 0$ *la probabilità che il gap SAA superi* ε *decresce* esponenzialmente *in N:*

$$\mathbb{P}(\hat{z}_N^* - z^* \geq \varepsilon) \leq \exp\left(-\frac{N\varepsilon^2}{2\sigma^2}\right). \tag{2.23}$$

Riduzione degli scenari tramite clustering

Dal momento che la convergenza della formulazione SAA alla soluzione ottima dipende dal numero di scenari che vengono considerati, si dovrebbe risolvere il problema sfruttando un numero estremamente elevato di scenari. Purtroppo, tanti scenari portano all'aumento della dimensione del problema di ottimizzazione. Un'idea utile, per poter puntare su una soluzione buona evitando di far esplodere la dimensione del problema, è quella di ridurre il numero di scenari andando a sceglierli in modo 'intelligente'. Attravero la *scenarios reduction* si seleziona un sottoinsieme rappresentativo di scenari, preservando in questo modo le proprietà statistiche.

Definizione 2.2.2 (Riduzione degli scenari)**.** *Dato un insieme di N scenari* $\{\boldsymbol{\xi}^{(s)}, p^{(s)}\}_{s=1}^N$ *con probabilità* $p^{(s)} > 0$ *e* $\sum_s p^{(s)} = 1$*, si trova un insieme di dimensione inferiore* $\{\tilde{\boldsymbol{\xi}}^{(k)}, \tilde{p}^{(k)}\}_{k=1}^K$*, con* $K \ll N$*, che minimizzi una misura di discrepanza tra la distribuzione originale e quella ridotta.*

Riduzione usando K-Means. Un approccio molto usato per la riduzione degli scenari è l'algoritmo *K-Means*. Dato l'insieme di scenari di partenza, questo viene suddiviso in gruppi, ognuno di questi viene rappresentato con il suo centroide e con una probabilità associata pari alla proporzione di scenari originali nel cluster.

Definizione 2.2.3 (K-Means)**.** *Dati N scenari* $\{\boldsymbol{\xi}^{(s)}\}_{s=1}^{N}$ *e un numero K di scenari finali, l'algoritmo K-Means:*

1. *Risolve il problema:*

$$\min_{\mathscr{C}_1,\ldots,\mathscr{C}_K} \sum_{k=1}^{K} \sum_{s\in\mathscr{C}_k} \left\| \boldsymbol{\xi}^{(s)} - \tilde{\boldsymbol{\xi}}^{(k)} \right\|_2^2,$$

   dove $\tilde{\boldsymbol{\xi}}^{(k)} = \frac{1}{|\mathscr{C}_k|} \sum_{s\in\mathscr{C}_k} \boldsymbol{\xi}^{(s)}$ *è il centroide del cluster k;*
2. *Assegna a ciascun centroide la probabilità* $\tilde{p}^{(k)} = |\mathscr{C}_k|/N$.

Il set ridotto è $\{(\tilde{\boldsymbol{\xi}}^{(k)}, \tilde{p}^{(k)})\}_{k=1}^{K}$.

Require: scenari $\{\boldsymbol{\xi}^{(s)}\}_{s=1}^{N}$, numero di cluster K
Ensure: scenari ridotti $\{(\tilde{\boldsymbol{\xi}}^{(k)}, \tilde{p}^{(k)})\}_{k=1}^{K}$
1: Inizializzazione dei centroidi $\tilde{\boldsymbol{\xi}}^{(1)}, \ldots, \tilde{\boldsymbol{\xi}}^{(K)}$
2: **repeat**
3: **Assegnazione**: $c(s) \leftarrow \arg\min_k \|\boldsymbol{\xi}^{(s)} - \tilde{\boldsymbol{\xi}}^{(k)}\|_2$ per ogni s
4: **Aggiornamento**: $\tilde{\boldsymbol{\xi}}^{(k)} \leftarrow \frac{1}{|\mathscr{C}_k|} \sum_{s:\, c(s)=k} \boldsymbol{\xi}^{(s)}$ per ogni k
5: **until** convergenza
6: $\tilde{p}^{(k)} \leftarrow |\mathscr{C}_k|/N$ per ogni k
7: **return** $\{(\lfloor\tilde{\boldsymbol{\xi}}^{(k)}\rceil, \tilde{p}^{(k)})\}_{k=1}^{K}$

Proprietà di convergenza. L'algoritmo K-Means converge in un numero finito di iterazioni, poiché ad ogni passo la funzione obiettivo non aumenta e il numero di possibili partizioni è finito. Bisogna però sottolineare il fatto che il minimo locale trovato dipende dall'inizializzazione dei centroidi, per cui non si ottiene sempre lo stesso risultati finale.

Teorema 2.2.5. *Sia* μ *la distribuzione degli scenari. Con probabilità 1, al crescere del numero di scenari originali* $N \to \infty$ *con K fisso, i centroidi K-Means convergono*

all'ottimo:

$$\min_{\tilde{\boldsymbol{\xi}}^{(1)},\ldots,\tilde{\boldsymbol{\xi}}^{(K)}} \mathbb{E}_{\boldsymbol{\xi}\sim\mu}\left[\min_{k} \|\boldsymbol{\xi} - \tilde{\boldsymbol{\xi}}^{(k)}\|_2^2\right]. \tag{2.24}$$

Inoltre, la distribuzione empirica dei centroidi con pesi $\tilde{p}^{(k)}$ converge debolmente a μ ristretta ai K punti ottimi [7].

2.3 Il problema del valore atteso (EV)

Facciamo una piccola parentesi su una semplificazione dei problemi stocastici a due stadi (Sezione 2.2). Quando si tratta questo argomento spesso si parte introducendo una semplificazione. Infatti, volendo ridurre la complessità del problema, un'idea abbastanza comune è quella di sostituire la distribuzione di probabilità della domanda ξ con il suo valore atteso $\bar{\xi} = \mathbb{E}[\xi]$, e risolvere il *problema del valore atteso* (Expected Value problem, EV):

$$z^{\mathrm{EV}} = \max_{\mathbf{a},\mathbf{y}} f(\mathbf{a},\mathbf{y}) + Q(\mathbf{a},\mathbf{y},\, \mathbb{E}[\boldsymbol{\xi}]), \tag{2.25}$$

invece del problema stocastico originale:

$$z^{*} = \max_{\mathbf{a},\mathbf{y}} f(\mathbf{a},\mathbf{y}) + \mathbb{E}_{\boldsymbol{\xi}}\left[Q(\mathbf{a},\mathbf{y},\, \boldsymbol{\xi})\right]. \tag{2.26}$$

La differenza $\mathrm{VSS} = z^{*} - z^{\mathrm{EV}}$ viene chiamata *Value of Stochastic Solution* (VSS), dove z^{EV} è il valore del problema stocastico originale quando la soluzione è fissata a quella ottima del problema EV. Questo numero mostra quanto sia fondamentale modellare espliciamente l'incertezza nel modello, dato che toglie la piattezza delle soluzione determnistica.

Nel capitolo successivo 3, introducendo i modelli che sono stati trattati, troveremo l'applicazione del problema EV nel *'Modello ibrido'*. La scelta è motivata dall'aumento della complessità della modellazione della realtà della dinamica di mercato: si è quindi deciso di ridurre la complessità del modello di ottimizzazione, per fornire una prospettiva diversa sul mercato.

2.4 Reti neurali artificiali

Quando si trattano funzioni di una certa complessità, uno strumento che si è sviluppato negli ultimi anni, grazie alla vasta disponibilità di dati di cui possiamo avvalerci, per ottenere un'approssimazione significativamente conveniente di tali funzioni sono le *reti neurali artificiali* (ANN, *Artificial Neural Networks*). Queste sono modelli ispirati al sistema nervoso biologico. Vediamo ora gli elementi principali delle reti neurali feed-forward e introduciamo le proprietà matematiche che ne giustificano l'impiego come approssimatori di funzioni complesse da calcolare nel contesto dell'ottimizzazione stocastica.

Una rete neurale feed-forward, anche detta Multi-Layer Perceptron (MLP), è caratterizzata da elementi principali:

- Il **perceptron**: o *neurone artificiale*, che è l'unità elementare della rete;
- La **funzione di attivazione**: ciò che porta la non-linearità nella rete;
- I **layers**: o *strati* della rete, che le danno profondità e aumentano le abilità di learning.

In particolare:

Il *perceptron* Il perceptron è stato introdotto da Rosenblatt [8] come elemento base delle reti neurali.

Definizione 2.4.1 (Perceptron)**.** *Un* perceptron *con d input è una funzione* $f : \mathbb{R}^d \to \mathbb{R}$ *definita da:*

$$f(\mathbf{x}) = \sigma\left(\boldsymbol{w}^\top \mathbf{x} + b\right) = \sigma\left(\sum_{k=1}^{d} w_k x_k + b\right), \tag{2.27}$$

dove $\boldsymbol{w} = (w_1, \ldots, w_d)^\top \in \mathbb{R}^d$ *è il vettore dei* pesi *(*weights*),* $b \in \mathbb{R}$ *è il* bias*, e* $\sigma : \mathbb{R} \to \mathbb{R}$ *è la* funzione di attivazione*.*

Operativamente il neurone calcola una combinazione lineare degli input su cui applica la funzione di attivazione non lineare σ. La non linearità è essenziale poichè, senza la sua presenza, combinando i neuroni otterremmo sempre una funzione lineare degli input, riducendo drasticamente la capacità espressiva del modello.

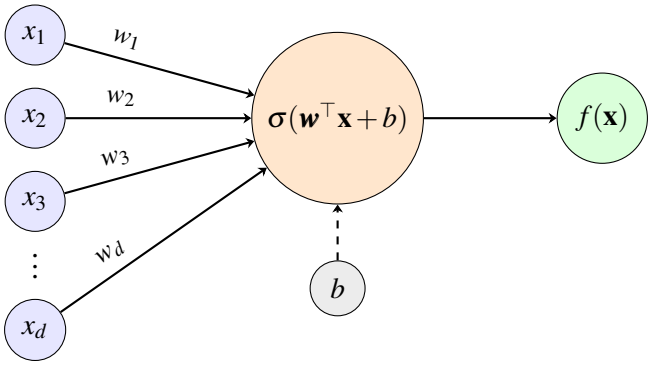


Figura 2.2 Schema di un perceptron.

Funzioni di attivazione Le funzioni di attivazione più rilevanti sono:

Sigmoide: $\sigma(z) = \dfrac{1}{1+e^{-z}} \in (0,1)$. Differenziabile ovunque, ma con lo svantaggio del *vanishing gradient*: per $|z| \gg 0$ si ha $\sigma'(z) \approx 0$, rallentando l'apprendimento negli strati profondi.

Tangente iperbolica: $\sigma(z) = \tanh(z) \in (-1,1)$. Simmetrica rispetto all'origine; anch'essa ha il problema del vanishing gradient.

ReLU (*Rectified Linear Unit*): $\sigma(z) = \max(0,z)$. Non è differenziabile in $z = 0$, ma è lineare e non satura per $z > 0$, questo accelera la convergenza dell'addestramento.

Definizione 2.4.2 (Funzione ReLU)**.** *La* Rectified Linear Unit *è la funzione* ReLU : $\mathbb{R} \to \mathbb{R}_{\geq 0}$ *definita da:*

$$\mathrm{ReLU}(z) = \max(0,z) = \frac{z+|z|}{2} = \begin{cases} z & \text{se } z > 0, \\ 0 & \text{se } z \leq 0. \end{cases} \tag{2.28}$$

Il Multi-Layer Perceptron (MLP) Unendo più strati di un perceptron si ottiene un modello più complesso, la *rete neurale feed-forward* o *Multi-Layer Perceptron* (MLP).

Definizione 2.4.3 (Multi-Layer Perceptron)**.** *Un MLP con L strati nascosti (*hidden

layers*) è una funzione* $f_{\boldsymbol{\theta}} : \mathbb{R}^{d_0} \to \mathbb{R}^{d_{L+1}}$ *definita da:*

$$\begin{aligned} \mathbf{h}^{(0)} &= \mathbf{x} \in \mathbb{R}^{d_0}, \\ \mathbf{h}^{(\ell)} &= \sigma\Big(\mathbf{W}^{(\ell)}\mathbf{h}^{(\ell-1)} + \mathbf{b}^{(\ell)}\Big) \in \mathbb{R}^{d_\ell}, \quad \ell = 1, \dots, L, \\ f_{\boldsymbol{\theta}}(\mathbf{x}) &= \mathbf{W}^{(L+1)}\mathbf{h}^{(L)} + \mathbf{b}^{(L+1)} \in \mathbb{R}^{d_{L+1}}, \end{aligned} \tag{2.29}$$

dove $\mathbf{W}^{(\ell)} \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ *è la* matrice dei pesi *e* $\mathbf{b}^{(\ell)} \in \mathbb{R}^{d_\ell}$ *è il vettore dei* bias *dello strato* ℓ*. Il vettore di tutti i parametri è* $\boldsymbol{\theta} = \{(\mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})\}_{\ell=1}^{L+1}$.

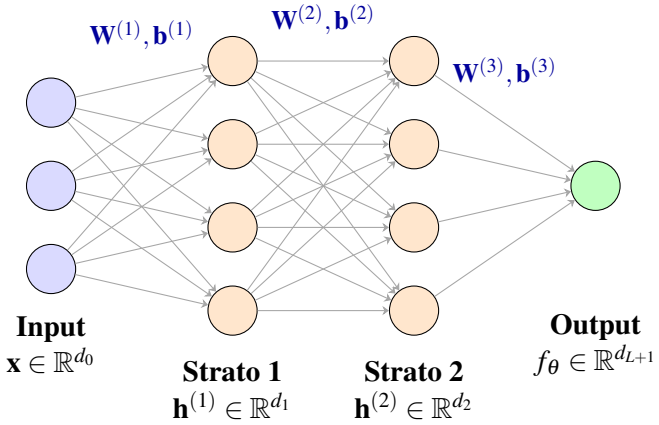


Figura 2.3 Architettura di un MLP con due strati nascosti.

Il teorema di approssimazione universale

Grazie al *teorema di approssimazione universale* (UAT) proposto da George Cybenko[9], possiamo giustificare l'uso delle reti neurali come approssimatori di funzioni.

Teorema 2.4.1 (Approssimazione Universale)**.** *Sia* $\sigma : \mathbb{R} \to \mathbb{R}$ *una qualsiasi funzione sigmoide. Per ogni* $f \in C(K)$ *(continua su un compatto* $K \subset \mathbb{R}^d$*) e per ogni* $\varepsilon > 0$*, esiste un intero* $M \in \mathbb{N}$ *e parametri* $\{(w_k, b_k, \alpha_k)\}_{k=1}^{M}$ *con* $w_k \in \mathbb{R}^d$*,* $b_k, \alpha_k \in \mathbb{R}$*, tali che la funzione*

$$g(\mathbf{x}) = \sum_{k=1}^{M} \alpha_k \, \sigma\Big(\boldsymbol{w}_k^\top \mathbf{x} + b_k\Big) \tag{2.30}$$

soddisfa $\sup_{\mathbf{x} \in K} |f(\mathbf{x}) - g(\mathbf{x})| < \varepsilon$.

Cybenko dimostra che una rete neurale con un singolo *hidden-layer* può approssimare qualsiasi funzione continua. In ogni caso il risultato è molto più forte poichè

altri studi hanno portato allo stesso risultato con una profondità maggiore della reta. Questo ci permette di usare le reti neurali per appossimare qualsiasi funzione continua nonostante la complessità.

Reti neurali come approssimatori del ricorso

Trattando il contesto della programmazione stocastica a due stadi (Sezione 2.2), notiamo subito che poter approssimare la funzione del ricorso atteso potrebbe portare ad un grande vantaggio in termini computazionali. Infati, il calcolo della funzione di $\mathscr{Q}(\mathbf{x}) = \mathbb{E}[Q(\mathbf{x}, \boldsymbol{\xi})]$ richiede, per ogni scelta $\mathbf{x}$, la risoluzione di un problema di ottimizzazione per ogni scenario. Trattando problemi a grande dimensionaltà, questo porta a costi molto elevati.

L'idea fondamentale del *modello surrogato* (*surrogate model*) è di approssimare $\mathscr{Q}$ con una funzione $f_{\boldsymbol{\theta}}$ ottenuta dal training su dati.

Definizione 2.4.4 (Modello surrogato Neural Network per il ricorso)**.** *Un* modello surrogato neurale *per la funzione di ricorso $\mathscr{Q}$ è un MLP $f_{\boldsymbol{\theta}} : \mathbb{R}^n \to \mathbb{R}$ addestrato su un dataset $\mathscr{D} = \{(\mathbf{x}^{(n)}, z^{(n)})\}_{n=1}^{N}$, dove $\mathbf{x}^{(n)}$ sono le decisioni di primo stadio e $z^{(n)} = Q_{\mathrm{SAA}}(\mathbf{x}^{(n)})$ è il corrispondente valore del ricorso SAA. La funzione $f_{\boldsymbol{\theta}}$ sostituisce $\mathscr{Q}$ nell'ottimizzazione:*

$$\mathbf{x}^* \approx \arg\max_{\mathbf{x} \in \mathscr{X}} \{ f_{\boldsymbol{\theta}}(\mathbf{x}) - \textit{costi primo stadio}(\mathbf{x}) \}. \tag{2.31}$$

Integrazione del surrogato nel MILP: linearizzazione della ReLU

Bisogna notare che, per sfruttare l'approccio del surrgato nella formulazione MILP, $f_{\boldsymbol{\theta}}$ deve integrarsi nell'ottimizzazione (2.31) come vincolo lineare. Poiché un MLP con attivazioni ReLU è una funzione lineare a tratti (*piecewise linear*), essa può essere rappresentata tramite un insieme di vincoli lineari [10].

Teorema 2.4.2 (Formulazione MIP della ReLU)**.** *Sia $y = \mathrm{ReLU}(z) = \max(0, z)$ con*

$l \leq z \leq u$, $l < 0 < u$. La formulazione MIP del vincolo $y = \mathrm{ReLU}(z)$ è:

$$y \geq 0, \tag{2.32}$$
$$y \geq z, \tag{2.33}$$
$$y \leq u\,\delta, \tag{2.34}$$
$$y \leq z - l\,(1-\delta), \tag{2.35}$$
$$\delta \in \{0,1\}, \tag{2.36}$$

con $\delta \in \{0,1\}$. Quando $\delta = 1$ (neurone attivo), i vincoli (2.34)-(2.35) *portano $y = z$; se $\delta = 0$, si ha $y = 0$.*

Dimostrazione. Suddividiamo i due casi:

- $\delta = 1$: (2.34) dà $y \leq u$; (2.35) dà $y \leq z$; con (2.33) ($y \geq z$) si ha $y = z$. Poiché $\delta = 1$ è selezionato solo quando $z \geq 0$ (altrimenti la soluzione $y = 0, \delta = 0$ risulterebbe migliore), si ha $y = z = \mathrm{ReLU}(z)$.
- $\delta = 0$: (2.34) dà $y \leq 0$; con (2.32) ($y \geq 0$) si ottiene $y = 0 = \mathrm{ReLU}(z)$ per $z \leq 0$.

□

Osservazione 2.4.1. *Per un MLP con L strati nascosti e d_ℓ neuroni per strato, l'integrazione nel MILP introduce $\sum_\ell d_\ell$ variabili binarie ausiliarie e $4\sum_\ell d_\ell$ vincoli lineari aggiuntivi.*

Addestramento della rete neurale

Abbiamo quindi introddotto il concetto di addestramento della rete su dei dati di training; infatti, per fare in modo che la rete neurale possa essere effettivamente in grado di approssimare una funzione, è necessario *addestrarla*. Il *training* consiste nel trovare i valori dei parametri $\boldsymbol{\theta}$ che minimizzano la *funzione di perdita* (*loss function*) $\mathscr{L}(\boldsymbol{\theta})$ misurata sul *training set* $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$.

Funzione di perdita

Per problemi il cui obiettivo è approssimare una funzione in $\mathbb{R}$, la funzione di perdita più utilizzata è l'*errore quadratico medio* (MSE, *Mean Squared Error*):

$$\mathscr{L}_{\text{MSE}}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N}\left(f_{\boldsymbol{\theta}}(\mathbf{x}^{(n)}) - y^{(n)}\right)^2. \tag{2.37}$$

Il MSE è non convesso rispetto all'insieme dei $\boldsymbol{\theta}$, a causa della non linearità degli strati nascosti. Il problema di addestramento è quindi un problema di ottimizzazione non convessa.

Algoritmo di Backpropagation

Per risolvere tale problema, si calcola il gradiente $\nabla_{\boldsymbol{\theta}}\mathscr{L}$ tramite l'algoritmo di *backpropagation*.

Per un MLP con *L hidden layers* e funzione di perdita $\mathscr{L}$, il gradiente rispetto ai pesi dello strato ℓ è:

$$\frac{\partial \mathscr{L}}{\partial \mathbf{W}^{(\ell)}} = \boldsymbol{\delta}^{(\ell)}\left(\mathbf{h}^{(\ell-1)}\right)^{\top}, \tag{2.38}$$

dove $\boldsymbol{\delta}^{(\ell)} = \left(\mathbf{W}^{(\ell+1)}\right)^{\top}\boldsymbol{\delta}^{(\ell+1)} \odot \sigma'\left(\mathbf{W}^{(\ell)}\mathbf{h}^{(\ell-1)} + \mathbf{b}^{(\ell)}\right)$ è l'errore del layer ℓ, calcolato a ritroso.

Ottimizzatore Adam

L'ottimizzatore più utilizzato per aggiornare i parametri in modo efficiente nei problemi ad alta dimensionalità è *Adam* (*Adaptive Moment Estimation*).

Definizione 2.4.5 (Algoritmo Adam)**.** *Dato il tasso di apprendimento* $\eta > 0$*, i fattori*

di decadimento $\beta_1, \beta_2 \in [0,1)$ *ed* $\varepsilon > 0$*, il parametro* θ *è aggiornato come segue:*

$$\begin{aligned}
g_t &= \nabla_\theta \mathscr{L}_t(\theta) \quad \textit{(gradiente sul mini-batch)}, \\
m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \quad \textit{(stima primo momento)}, \\
v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \quad \textit{(stima secondo momento)}, \\
\hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t} \quad \textit{(correzione bias)}, \\
\theta_t &= \theta_{t-1} - \eta \, \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon}.
\end{aligned} \tag{2.39}$$

Adam converge a un punto stazionario sotto condizioni di regolarità della funzione obiettivo.

Regolarizzazione L_2

Sempre con la finalità di ottenere un addestramento migliore della rete, spesso viene applicata la regolarizzazione. Questa è una tecnica per contrastare l'*overfitting*, ovvero quella tendenza della rete ad apprendere 'a memoria' ciò che le viene fornito dal training, peccando nella generalizzazione delle regole. La regolarizzazione L_2 (nota anche come *weight decay*) aggiunge alla funzione di perdita un termine proporzionale alla norma quadratica dei parametri:

$$\mathscr{L}_{\text{reg}}(\boldsymbol{\theta}) = \mathscr{L}_{\text{MSE}}(\boldsymbol{\theta}) + \frac{\alpha}{2} \|\boldsymbol{\theta}\|_2^2, \tag{2.40}$$

dove $\alpha > 0$ è il coefficiente di regolarizzazione (*penalità* L_2).

Osservazione 2.4.2 (Bias-variance tradeoff)**.** *Bisogna osservare che il termine di regolarizzazione introduce bias verso soluzioni con parametri piccoli, riducendo la varianza della stima. Sfruttare l'ottimizzazione degli iperparametri per trovare un valore ottimo di* α *bilancia questi due effetti (*bias-variance tradeoff*).*

Ottimizzazione degli iperparametri

Parlando appunto degli iperparamentri, come α nel caso della regolarizzazione, bisogna essere in grado di settarli nel modo corretto per ottenere una prestazione ottimale della rete neurale. Alcuni esempi di *iperparametri* sono: il numero di strati

L, il numero di neuroni per strato $\{d_\ell\}$, il tasso di apprendimento η, il coefficiente di regolarizzazione α, e altri. La scelta degli iperparametri porta ad un nuovo problema di ottimizzazione:

$$\lambda^* = \arg\min_{\lambda \in \Lambda} \mathscr{E}_{\text{val}}\left(f_{\boldsymbol{\theta}^*(\lambda)}\right), \tag{2.41}$$

dove $\lambda \in \Lambda$ è il vettore degli iperparametri in Λ, $\boldsymbol{\theta}^*(\lambda)$ è il vettore dei parametri ottimali addestrato con iperparametri λ, e $\mathscr{E}_{\text{val}}$ è l'errore sul validation set.

Il problema (2.41) è generalmente non convesso, non differenziabile e molto costoso da un punto di vista computazionale: ogni valutazione richiede l'intero ciclo di addestramento. Le strategie principali per risolvere questo problema sono:

- **Grid search**: esplorazione di una griglia prefissata di valori. Efficace in basse dimensionalità, ma poi scala esponenzialmente con il numero di iperparametri.
- **Random search**: campionamento casuale nello spazio Λ. Si dimostra che il random search è più efficiente del grid search quando solo pochi iperparametri influenzano l'addestramento.
- **Ottimizzazione bayesiana**: costruisce un modello probabilistico della funzione $\lambda \mapsto \mathscr{E}_{\text{val}}(\lambda)$ e lo usa per selezionare il prossimo punto da valutare.

Tree Parzen Estimator (TPE) Dal momento che nel Capitolo 4 relativa alla sperimentazione verrà citato il framework di *Optuna* per la ricerca ottimale degli iperparametri, introduciamo brevemente il *Tree Parzen Estimator* (TPE) implementato in *Optuna* [11]. Questo è un algoritmo di ottimizzazione bayesiana che modella la distribuzione degli iperparametri in modo non parametrico.

Il TPE modella separatamente due distribuzioni condizionali:

$$p(\lambda \mid y) = \begin{cases} \ell(\lambda) & \text{se } y \leq \text{soglia}, \\ g(\lambda) & \text{se } y > \text{soglia}, \end{cases} \tag{2.42}$$

dove $\ell(\lambda)$ è la densità degli iperparametri che hanno portato a prestazioni migliori e $g(\lambda)$ quella degli iperparametri che hanno portato a prestazioni peggiori. Il candidato successivo per la valutazione è scelto massimizzando il rapporto di verosimiglianza:

$$\lambda^{\text{next}} = \arg\max_{\lambda} \frac{\ell(\lambda)}{g(\lambda)}, \tag{2.43}$$

che corrisponde a selezionare gli iperparametri che massimizzano la probabilità di appartenere al gruppo dei *buoni*.

L'Active Learning

Dato che l'addestramento di un modello surrogato neurale (Sezione 2.4) su un dataset generato casualmente non garantisce una buona approssimazione di $\mathscr{Q}(\mathbf{x})$, si è utilizzata la metodoloiga di *active learning*. Questa affronta questo problema modificanto step by step il dataset di addestramento in base all'interazione tra il modello appreso e un *oracolo* esterno.

Introducendo brevemente il concetto, l'idea è quella di dare nuovi esempi, non ancora etichettati, con la propria etichetta al modello, affinchè possa ottenere un addestramento migliore. In particolare:

Definizione 2.4.6 (Setting dell'Active Learning)**.** *L'active learning è caratterizzato da:*

- *Un* insieme $\mathscr{U}$ *di elementi non etichettati che la rete non conosce;*
- *Un* oracolo $\mathscr{O} : \mathscr{X} \to \mathbb{R}$ *che, interrogato su un punto* $\mathbf{x} \in \mathscr{U}$*, fornisce l'etichetta vera* $y = \mathscr{O}(\mathbf{x})$ *a un certo costo* $c(\mathbf{x}) > 0$*;*
- *Un dataset etichettato* $\mathscr{D} = \{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{|\mathscr{D}|}$*;*
- *Una* strategia di query $q : \boldsymbol{\theta} \mapsto \mathbf{x}^* \in \mathscr{U}$ *che seleziona il prossimo punto da inserire nel dataset in base al modello corrente* $f_{\boldsymbol{\theta}}$*.*

L'obiettivo è ottenere $f_{\boldsymbol{\theta}}$ *ben addestrato cercando di fare il minor numero possibile di chiamate a* $\mathscr{O}$*.*

Per quanto riguarda le strategie di query, ne esistono di diverso tipo. Nel nostro caso, è stata usata la strategia *Model-Based Query*. Questa è determinata dall'ottimizzazione del modello corrente: dato il punto $\mathbf{x}^* = \arg\max_{\mathbf{x}} f_{\boldsymbol{\theta}}(\mathbf{x})$, soluzione attuale dato il surrogato, lo si dà in pasto all'oracolo che lo valuta per verificare e correggere la predizione. Questa strategia è particolarmente adatta agli scenari di *surrogate-based optimization*, in cui l'obiettivo finale è l'ottimizzazione piuttosto che la stima uniforme della funzione [12].

Nel capitolo sperimentale 4, andremo a trattare la generazione del dataset in modo più approfondito. Verrà discusso il processo di formazione dei dati a partire dal campionamento casuale guidato fino ad arrivare all'active learning.

2.5 Il predittore lineare nel problema EV

Nella sezione precedente 2.25 abbiamo introdotto il problema di EV, ovvero l'uso di una quantità deterministica, data dal valore atteso, a sostituire gli elementi di incertezza. Siccome esistono classi di predittori che possono portare a vantaggi relativamente all'integrazione nel modello MILP, pur portando con essi degli svantaggi, parliamo ora rapidamente di un predittore lineare che verrà usato nel *Modello Ibrido* 3.5.

Regressione Ridge come predittore

Definiamo la *regressione Ridge* (o regressione ℓ^2-regolarizzata) considerando un target scalare come:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^m} \sum_{k=1}^{N} (y^{(k)} - \boldsymbol{\beta}^\top \mathbf{x}^{(k)})^2 + \lambda \|\boldsymbol{\beta}\|_2^2, \tag{2.44}$$

dove $\lambda > 0$ è il parametro di regolarizzazione. La soluzione analitica è:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}, \tag{2.45}$$

con $\mathbf{X} \in \mathbb{R}^{N\times m}$ la matrice degli input. Il parametro $\lambda\|\boldsymbol{\beta}\|_2^2$ regolarizza per prevenire l'overfitting: per $\lambda \to \infty$ i coefficienti vanno a 0; per $\lambda \to 0$ si ha la regressione standard.

Trattando invece $\mathbf{y} \in \mathbb{R}^n$ con n componenti, il problema (2.44) diventano n problemi scalari indipendenti:

$$\hat{\boldsymbol{\beta}}_i = \arg\min_{\boldsymbol{\beta}_i} \sum_{k=1}^{N} (y_i^{(k)} - \boldsymbol{\beta}_i^\top \mathbf{x}^{(k)})^2 + \lambda \|\boldsymbol{\beta}_i\|_2^2, \qquad i = 1,\dots,n, \tag{2.46}$$

che porta a n vettori di coefficienti $\hat{\boldsymbol{\beta}}_1,\dots,\hat{\boldsymbol{\beta}}_n$. Ovvero, guardando tutto il vettore di input, andiamo a stimare indipendentemente la predizione sull'output. Ogni predizione è lineare nell'input: $\hat{y}_i(\mathbf{x}) = \hat{\boldsymbol{\beta}}_i^\top \mathbf{x}$.

Bisogna però notare che la regressione Ridge approssima in modo soddisfacente solo le funzioni lineari: non è in grado di catturare relazioni non lineari tra input e output. Per il problema che andremo a trattare, la funzione da approssimare

è $\mathbf{a} \mapsto \bar{\mathbf{d}}(\mathbf{a})$, dove $\bar{d_i}(\mathbf{a})$ è la domanda attesa del prodotto i dato l'assortimento $\mathbf{a}$. Questa funzione è non lineare a causa delle strutura intrinseca della generazione, per cui l'utilizzo di Ridge potrebbe non essere la soluzione ottimale, e nella sezione sperimentale si mostrerà proprio questa difficoltà. Per motivare la scelta da un punto di vista matematico [13], possiamo notare che, avendo come input un vettore binario, la nostra funzione ha la seguente struttura:

$$f : \{0,1\}^m \to \mathbb{R}^m \tag{2.47}$$

ovvero, il dominio della funzione è dato dai vertici di un ipercubo unitario. Possiamo quindi introdurre il seguente teorema:

Teorema 2.5.1 (*Fourier expansion theorem*)**.** *Ogni funzione $f : \{0,1\}^m \to \mathbb{R}$ può essere definita univocamente come un polinomio multilineare*

$$f(x) = \sum_{S \subset [m]} \hat{f}(S) x^S$$

Questa espressione è detta espansione di Fourier *di f, e $\hat{f}(S) \in \mathbb{R}$ è detto* coefficiente di Fourier *di f su S.*

La regressione Ridge è il troncamento di primo ordine di tale espansione. Come ogni approssimazione, essendo solo al primo ordine, non riuscirà a spiegare tutta l'informazione della vera funzione ma, avendo il potenziale della linearità, è un buon compromesso a cui potersi affidare.

Infatti, il vantaggio principale che si potrebbe riscontrare sfruttando il predittore lineare rispetto al MLP nel contesto dell'ottimizzazione dell'assotimento è l'integrazione nel modello MILP. Essendo che la predizione è già lineare, non è necessario introdurre variabili ausiliarie, big-M o linearizzazioni delle funzioni di attivazione.

Osservazione 2.5.1 (Complessità computazionale a confronto)**.** *Un MLP con L strati e H neuroni per strato aggiunge $O(L \cdot H)$ variabili binarie e vincoli big-M per l'inserimento delle ReLU (Teorema 2.4.2) nel modello. Introdurre n regressioni Ridge porta ad aggiungere n vincoli lineari per il problema in esame.*

Nel capitolo sperimenatale 4, andremo a trattare la qualità dell'approssimazione lineare. Verrà discusso ed intepretato il valore del coefficiente R^2 sul train set e sul test set.

Capitolo 3

Contesto e definizione del problema

Nel seguente capitolo introduciamo la struttura del problema affrontato, andando a vedere i modelli usati che verranno confrontati e discussi nel capitolo finale. Inizialmente si descrive la costruzione concreta del contesto su cui si fonda il lavoro sperimentale: la definizione del mercato, la calibrazione del modello di domanda, la costruzione del dataset dei prodotti. Successivamente ci sarà la formulazione del problema di ottimizzazione.

3.1 Il contesto di mercato

Come detto, il problema studiato è l'ottimizzazione dell'assortimento e delle quantità da ordinare per un punto vendita. Il contesto è basato sulla vendita di prodotti legati ai *primi piatti secchi e freschi*: pasta, riso e gnocchi. La scelta di questa categoria è motivata da alcune caratteristiche che la rendono interessante per sperimentare:

- È una categoria ad *alto volume* e *alta frequenza di acquisto*;
- Presenta una struttura di *segmentazione chiara*: ci sono consumatori fedeli a marchi storici italiani, consumatori price-sensitive, e consumatori attenti alla qualità;
- La varietà di formati (pasta, riso, gnocchi) introduce dinamiche di *sostituzione tra categorie*, non solo tra marchi della stessa categoria, rendendo il problema più interessante.

Il modello considera **4 marche**, **3 tipologie di prodotto** e **3 posizioni sullo scaffale**. Le marche rappresentano quattro tipologie principali presenti nel mercato:

Marca A Marchi leader di mercato per riconoscibilità, qualità media-alta e con prezzi nella fascia media. Sono la *categoria di riferimento* del modello di utilità.

Marca B Marchi discount orientati al prezzo e con qualità medio-bassa. Rappresentano la scelta del consumatore economicamente orientato.

Marca C Marchi artigianali caratterizzati da qualità alta, prezzi elevati, gamma ristretta. Attirano il segmento quality-sensitive.

Marca D Marchi percepiti come premium che hanno una forte identità di marca e prezzi medio-alti. Attraggono un segmento di consumatori con forte fedeltà.

Le tre posizioni sullo scaffale modellano l'effetto della *visibilità* sul comportamento d'acquisto ed entrano come caratteristiche pesate nella funzione di utilità deterministica:

Scaffale Occhi Posizione a livello degli occhi: massima visibilità, costo fisso più elevato ($f_{\text{Occhi}} = 8{,}00$ €/settimana).

Scaffale Alto Posizione sopra il livello degli occhi: buona visibilità, costo intermedio ($f_{\text{Alto}} = 6{,}00$ €/settimana).

Scaffale Basso Posizione bassa: visibilità ridotta, costo fisso inferiore ($f_{\text{Basso}} = 5{,}00$ €/settimana).

3.2 Il modello Latent Class utilizzato

Il modello di domanda adottato è il seguente *Latent Class Choice Model* (LCCM) a 6 segmenti.

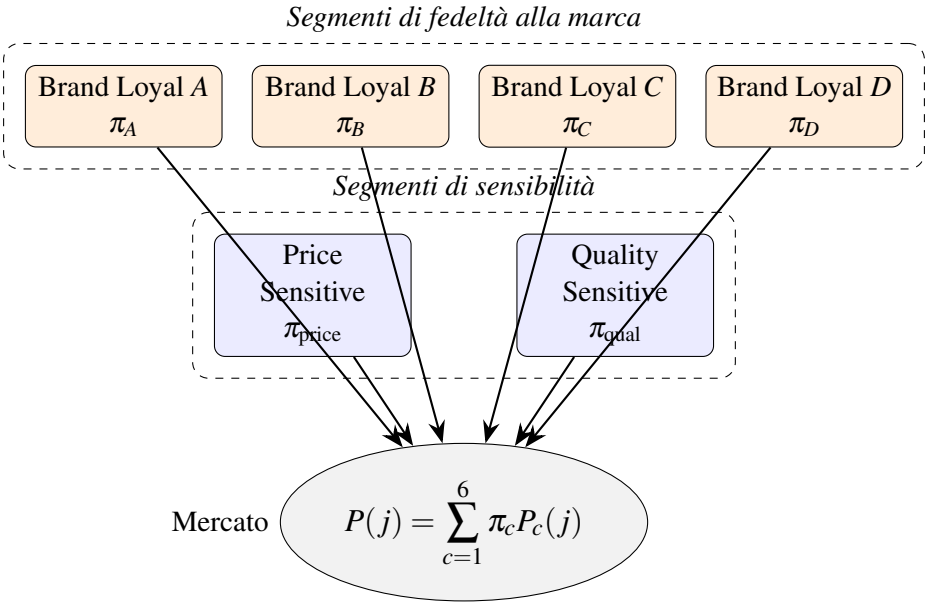


Figura 3.1 Struttura del modello Latent Class con $C = 6$ segmenti.

La funzione di utilità deterministica del segmento c per l'opzione (prodotto-scaffale) j è come segue:

$$\begin{aligned} V_{cj} = {} & \alpha_c + \nu_{1,c}\, q_j^- + \nu_{2,c}\, q_j^+ + \theta_c\, p_j + \beta_c^{\text{riso}}\, \mathbf{1}[\text{Riso}_j] + \beta_c^{\text{gno}}\, \mathbf{1}[\text{Gnocchi}_j] \\ & + \gamma_c^B\, \mathbf{1}[\text{Marca B}_j] + \gamma_c^C\, \mathbf{1}[\text{Marca C}_j] + \gamma_c^D\, \mathbf{1}[\text{Marca D}_j] \\ & + \delta_c^{\text{occ}}\, \mathbf{1}[\text{Occhi}_j] + \delta_c^{\text{bas}}\, \mathbf{1}[\text{Basso}_j], \end{aligned} \tag{3.1}$$

dove q_j^- indica il valore di qualità del prodotto j se $q_j^- < \bar{q} = 6{,}8$ (zero altrimenti), $q_j^+ \geq \bar{q}$ è il valore di qualità nell'altro caso, e p_j è il prezzo in €/kg. Le *categorie di riferimento* sono: **Pasta**, **Marca A**, **Scaffale Alto**.

La divisione in due rami della caratteristica relativa alla qualità è giustificata dalla volontà di modellare nel modo più coerente possibile il ragionamento fatto dal cliente: è diverso come viene percepita una scarsa qualità, che va logicamente a ridurre l'utilità, rispetto che una alta qualità che diversamente la aumenta.

I valori numerici dei parametri sono riportati nella tabella 3.1. Sono presenti anche le probabilità di appartenenza ai segmenti, scelte in modo tale che venga disegnata la struttura attesa del mercato.

Tabella 3.1 Parametri del modello LCCM per i sei segmenti.

Parametro	**Loyal A**	**Loyal B**	**Loyal C**	**Loyal D**	**Price**	**Quality**
π_c	0.06	0.06	0.06	0.06	0.40	0.36
α_c (intercetta)	$+3.6$	-2.4	-2.4	-3.4	-0.9	-0.5
$v_{1,c}$ (qualità < 6.8)	-0.1	-0.1	-0.1	-0.1	-0.3	-1.5
$v_{2,c}$ (qualità ≥ 6.8)	$+0.2$	$+0.2$	$+0.2$	$+0.2$	$+0.2$	$+0.6$
θ_c (prezzo)	-1.0	-1.0	-1.0	-1.0	-1.3	-1.0
β_c^{riso}	-0.4	-0.4	-0.4	-0.4	-0.5	$+0.2$
β_c^{gno}	-0.4	-0.4	-0.4	-0.4	-0.7	$+0.4$
γ_c^B	-6.0	$+6.0$	-1.0	0.0	$+0.6$	-0.4
γ_c^C	-6.0	-1.0	$+6.0$	0.0	-0.5	$+0.6$
γ_c^D	-5.0	0.0	0.0	$+7.0$	0.0	0.0
δ_c^{occ}	$+0.6$	$+0.6$	$+0.6$	$+0.6$	$+1.0$	$+0.2$
δ_c^{bas}	-0.5	-0.5	-0.5	-0.5	-0.9	-0.1

Analisi economica dei parametri

I segmenti di fedeltà alla marca:

I quattro segmenti di fedeltà alla marca (Loyal A, B, C, D) si differenziano nel valore numerico dei coefficienti di marca $\gamma_c^B, \gamma_c^C, \gamma_c^D$ e nell'intercetta α_c, siccome quest'ultima è strettamente legata al marchio A.

L'intercetta. L'intercetta α_c chiarisce come il segmento percepisce la categoria di riferimento (Pasta, Marca A, Scaffale Alto). Il segmento Loyal A ha $\alpha_A = +3{,}6$: è il valore più alto tra i segmenti, poichè il suo prodotto preferito è proprio la Marca A che fa parte dell'intercetta. I segmenti Loyal B, C, D hanno intercette negative (rispettivamente $-2{,}4$, $-2{,}4$, $-3{,}4$), perché i loro prodotti preferiti sono le marche B, C o D.

La fedeltà alla marca. Dal momento che l'interpretazione del valore dei coefficienti γ_c^k ci dice quanta variazione positiva si ha scegliendo la marca k rispetto alla Marca A, vediamo che per un segmento Loyal k, il proprio coefficiente è molto alto

(+6 o +7), mentre i coefficienti delle altre marche sono molto negativi (−6 o −5). La differenza $\gamma_k^k - \gamma_{k'}^k$ indica quanto il segmento k preferisce la propria marca rispetto a una marca alternativa k':

$$\Delta V_{kk'}^{\text{brand}} = \gamma_k^k - \gamma_{k'}^k \in [7, 13]. \tag{3.2}$$

Andando a fare una valutazione monetaria di quanto incide una marca rispetto a quella di preferenza, paragoniamo il $\Delta V_{kk'}^{\text{brand}}$ con il coefficiente del prezzo. Se il prezzo cambia di 1 €/kg, la differenza indotta nell'utilità è $|\theta_c| = 1$. Quindi, visto che il gap di fedeltà varia tra 7 e 13, vuol dire che equivale a una differenza di prezzo compresa tra 7 e 13 €/kg: un valore molto alto se consideriamo i prezzi a catalogo nel nostro dataset (1–8 €/kg circa). Si nota quindi che la marca è l'elemento più importante per il segmento *brand-loyal*, il che implica che difficilmente una variazione di prezzo può indurre una sostituzione di marca.

Qualità e prezzo. I segmenti fedeli alla marca hanno $\nu_{1,c} = -0{,}1$, il che li rende poco avversi alla bassa qualità. L'effetto sul prezzo $\theta_c = -1{,}0$ è identico per tutti e quattro. Il consumatore brand loyal è già sicuro della sua scelta, per cui qualità e prezzo giocano un ruolo secondario.

Il segmento price-sensitive

Il segmento price-sensitive è il più numeroso ($\pi_{\text{price}} = 0{,}40$).

Sensibilità al prezzo. Il coefficiente di prezzo $\theta_{\text{price}} = -1{,}3$ è in valore assoluto meggiore rispetto agli altri segmenti ($-1{,}0$). Questo indica che le persone appartenenti a questa classe di clienti hanno una *maggiore elasticità al prezzo*. Per il modello MNL, l'elasticità della probabilità di scelta del prodotto j rispetto al prezzo è:

$$e_{jj}^{\text{price}} = \theta_c \cdot p_j \cdot (1 - P_c(j)). \tag{3.3}$$

Facendo un'analisi tra price-sensitve e loyal, fissando la probabilità di scelta $P_c(j)$ e il prezzo p_j, il rapporto tra l'elasticità è:

$$\frac{e_{jj}^{\text{price-sens}}}{e_{jj}^{\text{loyal}}} = \frac{\theta_{\text{price}}}{\theta_{\text{loyal}}} = \frac{1{,}3}{1{,}0} = 1{,}3. \tag{3.4}$$

Questo dato ci dice che, supponendo di ridurre i prezzi del 10%, allora ci sarebbe un aumento del segmento price-sensitive all'interno del mercato il 30% superiore rispetto all'aumento dei brand loyal. Questo implica che il segmento price-sensitive è **30% più sensibile al prezzo**.

Sensibilità alla visibilità sullo scaffale. Un elemento che incide molto sull'utilità finale percepita dal segmento price-sensitive è la posizione sullo scaffale: $\delta^{\text{occ}}_{\text{price}} = +1{,}0$ e $\delta^{\text{bas}}_{\text{price}} = -0{,}9$. La differenza tra scaffale Occhi e scaffale Basso in termini di ultilità è di $1{,}0-(-0{,}9) = 1{,}9$. Per confronto, consideriamo questo delta anche nei segmenti brand loyal, in cui risulta decisamente meno elevato: $0{,}6-(-0{,}5) = 1{,}1$. Questo significa che, se un prodotto a basso prezzo viene messo nello scaffale basso, la sua utilità è penalizzata di $-0{,}9$ unità, equivalente a un aumento di prezzo di $0{,}9/1{,}3 \approx 0{,}69$ €/kg. In questo modo andiamo a modellare il comportamento *impulsivo* di questa tipologia di clienti: il prodotto economico viene acquistato facilmente se è in bella vista, ma potrebbe ridursi molto la percentuale d'acquisto se lo stesso prodotto venisse messo in scaffali meno visibili.

Preferenza di categoria. Dal momento che nell'immaginario comune la pasta viene vista come un prodotto economico, soprattutto se rapportato a riso e gnocchi, i coefficienti $\beta^{\text{riso}}_{\text{price}} = -0{,}5$ e $\beta^{\text{gno}}_{\text{price}} = -0{,}7$ sono stati modellati in questo modo affinchè risulti che il segmento price-sensitive abbia una preferenza per la pasta.

Il segmento quality-sensitive.

Il segmento quality-sensitive ($\pi_{\text{qual}} = 0{,}36$) è caratterizzato da coefficiente legati alla qualità più impattanti rispetto che le altre classi.

Percezione della qualità. Il parametro $v_{1,c}$, collegato alla qualità bassa, vale $-1{,}5$ per il segmento quality-sensitive, contro $-0{,}1$ per i brand loyal e $-0{,}3$ per il price-sensitive. Il parametro della qualità alta $v_{2,c}$ vale $+0{,}6$, contro $+0{,}2$ per tutti gli altri segmenti. Vediamo che il consumatore quality-sensitive *penalizza la bassa qualità molto più di quanto premi l'alta qualità*: è estremamente contrario all'acquisto di materia scadente.

Vediamo chiaramente che la differenza di utilità tra un prodotto con qualità $q = 5$ e uno con qualità $q = 7{,}5$ per il segmento quality-sensitive è:

$$\Delta V^{\text{qual}} = \nu_2 \cdot 7{,}5 - \nu_1 \cdot 5{,}0 = 0{,}6 \cdot 7{,}5 - (-1{,}5) \cdot 5{,}0 = 4{,}5 + 7{,}5 = 12{,}0. \quad (3.5)$$

Lo stesso calcolo per un segmento brand loyal è:

$$\Delta V^{\text{loyal}} = 0{,}2 \cdot 7{,}5 - (-0{,}1) \cdot 5{,}0 = 1{,}5 + 0{,}5 = 2{,}0. \quad (3.6)$$

ovvero sei volte inferiore.

Preferenza per riso e gnocchi. Come commentato per il segmento price-sensitive, siccome riso e gnocchi sono, nell'immaginario comunque, prodotti più qualitativi, il segmento quality-sensitive è stato pensato in modo tale che questo si riflettesse nelle sue preferenze. Infatti, i coefficienti $\beta^{\text{riso}}_{\text{qual}} = +0{,}2$ e $\beta^{\text{gno}}_{\text{qual}} = +0{,}4$.

Preferenza di marca. I coefficienti $\gamma^{C}_{\text{qual}} = +0{,}6$, ovvero quello legata alla Marca C che comprendei prodotti artigianali, indicano una preferenza significativa per questi prodotti, percepiti come più qualitativi. La Marca D ($\gamma^{D}_{\text{qual}} = 0$) è neutrale, mentre la Marca B ($\gamma^{B}_{\text{qual}} = -0{,}4$) è penalizzata in quanto associata a prodotti discount.

La no-buy option e il mercato di riferimento

L'opzione di non acquisto viene normalizzata a zero ($V_0 = 0$) quando andiamo a considerare i clienti che devono scegliere se entrare o meno in negozio guardando il catalogo. Quando abbiamo un assortimento completo, la no-buy option ha un valore relativamente basso $6,25\%$, considerando che si tratta di prodotti ampiamente compranti, quindi è un valore abbastanza coerente.

Invece, quando andiamo a calcolare la matrice di sostituzione, consideriamo un valore di normalizzazione della no-buy di $\bar{V}_0 = -2{,}5$. Questo valore impone che la probabilità di uscire senza acquistare per un consumatore che si trova già di fronte allo scaffale è molto bassa. Questa scelta di modello è stata fatta per rispecchiare la psicologia del consumatore più predisposto a sostituire quando è già in negozio, poichè pensa: "Sono già dentro, anche se non c'è esattamente quello che voglio, provo a trovare qualcosa di analogo"

Inoltre, dato che la matrice S, visto che è calcolata su 111 opzioni, ha elementi molto piccoli che corrispondono a sostituzioni trascurabili e visto che una matrice densa, da un punto di vista pratico-computazionale è molto più pesante che una matrice sparsa, si sono posti a zero gli elementi di S al di sotto di un dato valore $\delta > 0$, redistribuendo la massa di probabilità eliminata proporzionalmente agli elementi rimanenti, per mantenere somma unitaria.

Validazione del modello: gli Stress Test

La validazione del modello di scelta è stata fatta attraverso la valutazione delle probabilità risultati dopo modifiche spefiche dell'assortimento proposto. Ogni stress test fatto, è stato valutato considernado la probabilità cumulata di acquisto, per ogni prosotto, e anche la probabilità suddivisa per segmento, in modo tale da verificare che il risultato fosse coerente con le previsioni teoriche della struttura dei segmenti.

I test effettuati sono i seguenti:

DROP MARCA Rimozione dall'assortimento di tutti i prodotti di una singola marca. Si effettua per ciascuna delle quattro marche per vedere come reagiscono i segmenti brand-loyal.

DROP TIPOLOGIA Rimozione di tutti i prodotti di una singola tipologia (Pasta, Riso o Gnocchi).

DROP SCAFFALE Rimozione di tutti i prodotti messi in una specifica posizione sullo scaffale (Alto, Occhi o Basso).

FILTRO PREZZO Rimozione dei prodotti al di sotto o al di sopra di una soglia di prezzo. Prevalentemente per vedere come reagisce il segmento price-sensitive.

FILTRO QUALITÀ Rimozione dei prodotti al di sotto o al di sopra di una soglia di qualità. Per osservare la reazione dei quality-sensitve.

Per ogni test, ci aspettiamo i seguenti risultati:

- **Drop Marca** k: noteremo che il segmento Loyal k sarà spinto verso l'opzione di no-buy, poichè non trova la sua marca del cuore. I segmenti price-sensitive e quality-sensitive, avendo anche altre alternativa nelle marche rimanenti, verrano influenzati meno da questo cambiamento.

- **Drop Pasta**: in questo caso sarà il segmento price-sensitive ad avere un aumento dell'opzione no-buy, siccome la pasta rappresenta il prodotto percepitop più conveniente. I segmenti brand loyal possono ancora scegliere altri prodotti della loro marca, quindi saranno meno influenzati dalla rimozione della pasta. Il segmento quality-sensitive è più predisposto all'acquisto di altre categorie, quindi non verrà particolarmente coinvolto.
- **Drop Scaffale Occhi**: in questo caso tutti i segmenti saranno penalizzati, i price-sensitive di più che gli altri, dal momento che lo scaffale a livello occhi è quello più blasonato.
- **Filtro Prezzo (rimozione fascia bassa)**: anche in questo caso sarà il segmento price-sensitive ad avere un aumento dell'opzione no-buy; per gli altri segmenti, ci aspettiamo un effetto marginale. Per esempio, se la marca preferita non viene eliminata, i brand-loyal non ne saranno particolarmente turbati.
- **Filtro Qualità (rimozione fascia bassa qualità)**: questa rimozione favorisce il segmento quality-sensitive, mentre penalizza il segmento price-sensitve, visto che si eliminano molti prodotti con prezzi favorevoli e anche i brand-loyal dei prosotti discount.

Nei seguenti grafici sono riportati alcuni esempi di stress test condotto. Viene rappresentata la distribuzione di probabilità si acquisto di ogni prodotto (compresa la no-buy option) relativamente ai singoli segmenti di mercato e al mercato aggregato.

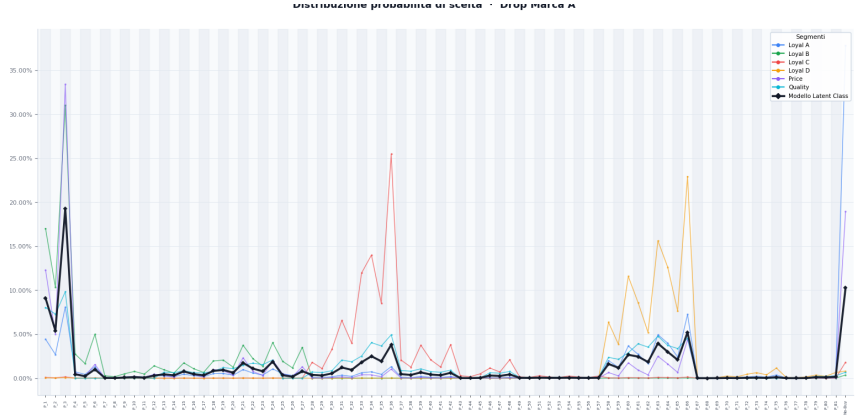


Figura 3.2 Stress test Drop Marca A: distribuzione della probabilità di scelta aggregata e per segmento. La rimozione della Marca A causa un forte aumento della probabilità di non acquisto nel segmento Loyal A.

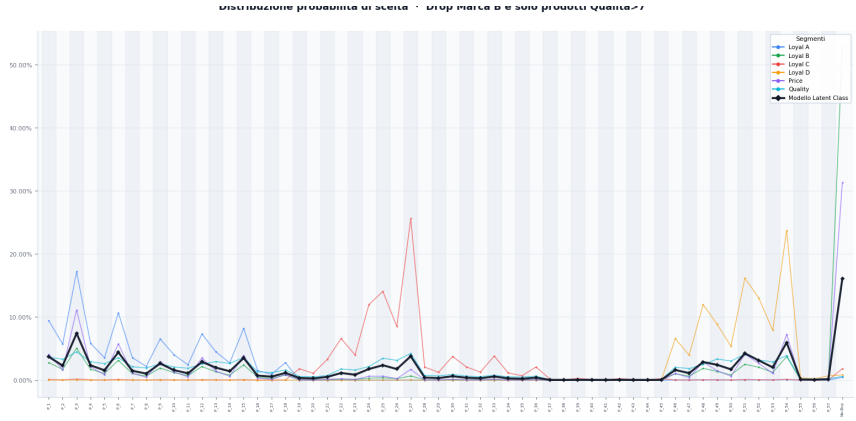


Figura 3.3 Stress test Drop Marca B. Penalizza fortemente i brand-loyal B e i price-sensitive

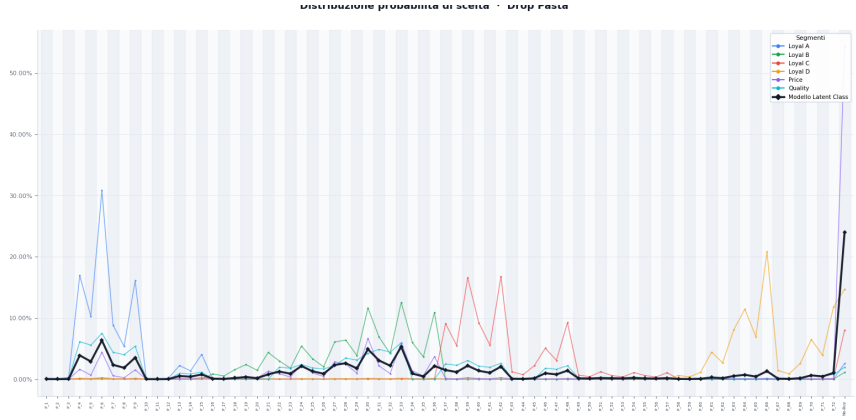


Figura 3.4 Stress test Drop Pasta: la rimozione dell'intera categoria pasta aumenta significativamente la no-buy nel segmento price-sensitive, mentre il segmento quality-sensitive è quasi indifferente (preferisce riso e gnocchi), come anche i brand-loyal, che trovano altri prodotti della loro marca.

3.3 Il dataset dei prodotti

Un elemento fondamentale nella costruzione del contesto è il dataset dei prodotti, che rappresenta l'input iniziale del modello di domanda e dell'ottimizzazione. Ciascun *prodotto* è identificato da (Marca, Tipologia, Prezzo, Scaffale), e il dataset contiene 111 record. Nello specifico, per ogni prodotto nel dataset abbiamo:

- **Prezzo** (p_j, €/kg): il prezzo di vendita al pubblico;
- **Qualità** ($q_j \in [0, 10]$): dà un rate al prodotto su una scala standard;
- **Costo** (c_j, €/kg): il costo di acquisto per il retailer;

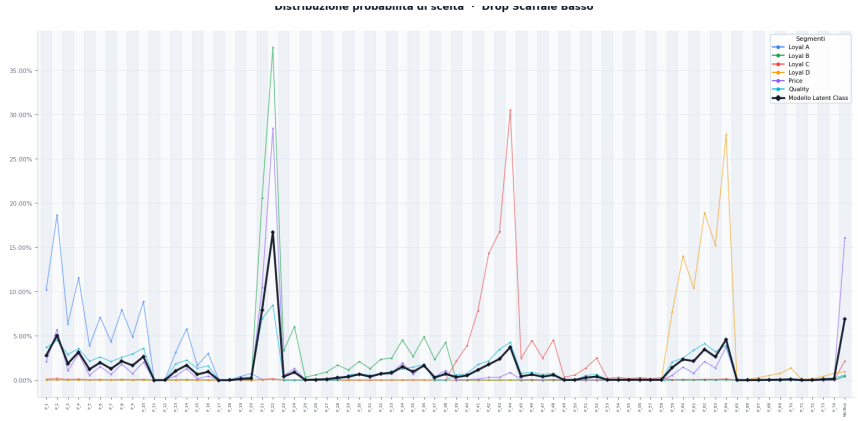


Figura 3.5 Stress test Drop Scaffale Basso: il segmento più penalizzato è quello price-sensitive.

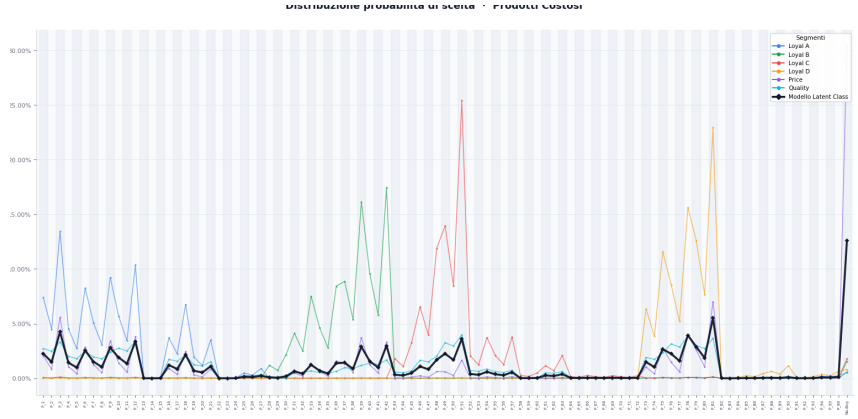


Figura 3.6 Filtro Prezzo: la rimozione dei prodotti a basso prezzo penalizza il segmento price-sensitive. Gli altri sono poco influenzati.

- **Holding cost** (h_j, €/kg): il costo di mantenimento del prdotto a scaffale per unità e per settimana.

Scelta dei valori

Tutti i valori numerici presenti nel dataset sono il risultato di una ricerca precisa. Ogni dato è reperibile online [14–19], e rappresenta la fotografia attuale del mercato:

Prezzi I prezzi (p_j) sono stati inseriti seguendo i listini del supermercati principali, che sono accessibili nei siti relativi. I valori sono espressi in €/kg per uniformare

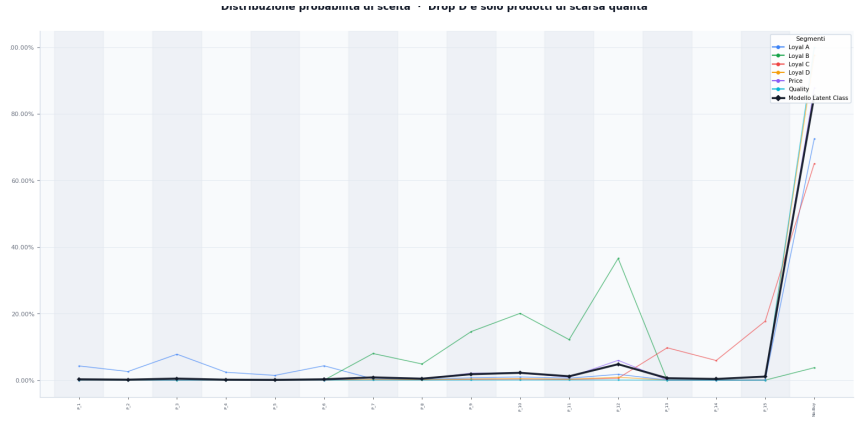


Figura 3.7 Filtro Qualità: la rimozione dei prodotti ad alta qualità penalizza il segmento Quality-Sensitive e anche i Loyal A e C vengono toccati da questo cambiamento, inoltre il Drop della Marca D penalizza i Loyal D.

il confronto tra formati diversi. Tendenzialmente si aggirano tra i 1,5 €/kg, per prodotti discount, fino a 8,0 €/kg, per gnocchi di marchi cari.

Qualità Il rate della qualità ($q_j \in [0, 10]$) è calcolato tenendo in considerazione:

1. Il rate fornito dai testi di qualità condotti da AltroConsumo [14], che assegna punteggi basati su test organolettici e analisi di laboratorio;
2. Elementi nell'etichetta del prodotto, che portano valore (presenza di denominazioni DOP/IGP o metodologie di lavorazione particolari).

La qualità presenta due picchi di frequenza per valori nell'intorno di 5 e di 8. Questa caratterizzazione dà rilevantza alla schelta di fissare la soglia per dividere qualità alta o bassa a $\bar{q} = 6{,}8$ nel modello LCCM (equazione (3.1)).

Costi per il retailer I costi unitari (c_j) sono stimati a partire dal costo delle materie prime sommato a costi di trasformazione e di produzione. Per la pasta secca, si tiene conto del costo attuale all'ingrosso del grano duro e si affinca a tale voce il costo di trasformazione, pesato in base alla qualità della lavorazione. Per il riso, si guarda il costo del Carnaroli. Infine, per gli gnocchi, il costo varia a seconda che siano definiti da materia prima fresca o da patate reidratate. Nel primo caso i costi sono superiori, siccome la materia fresca è più incline a deterioramento. I margini di profitto, ovvero

$m_j = p_j - c_j$, sono positivi per tutti i prodotti in assortimento, A seconda che si tratti di prodotti di discount o premium, il margine varia tra tra il 15% e il 45%.

Holding cost Il costo di mantenimento (h_j) incide maggiormente per i prodotti freschi per rispecchiare il maggiore dispendio di energia per il reparto frigo e per modellare il rischio di deterioramento dei prodotti freschi.

Capacità dello scaffale

Gli scaffali sono divisi in due aree: *scaffale secco*, dove verranno inseriti i prodotti Pasta e Riso e *scaffale frigo*, per gli Gnocchi. Le capacità sono espresse in kg, per facilità nella formulazione del modello di ottimizzazione, quindi 1 unità a scaffale equivale ad 1 kg di prodotto inseribile. I valori sono riportati nelle Tabelle 3.2.

Tabella 3.2 Capacità di scaffale per posizione e tipo di area (secco/frigo). Le capacità sono espresse in kg per posizione di scaffale.

Tipo scaffale	**Alto**	**Occhi**	**Basso**
Scaffale secco (kg)	480	480	360
Scaffale frigo (kg)	90	90	40

La differenza di capacità tra scaffale secco e frigo è abbastanza tipica nei negozi di medio-piccole dimensioni poichè l'estensione di scaffalatura refrigerata è nettamente inferiore alle stagere, anche per via dei costi di manutenzione.

Struttura del dataset: il ruolo dello scaffale

Il dataset è caratterizzato da una voce particolare: la tipologia dello scaffale. Si è deciso di inserire questo elemento per modellare la preferenza dei consumatori, fissato un prodotto, rispetto alla disposizione nello scaffale. Dovendo scegliere quali e quanti prodotti inserire, è naturale la scelta della posizione a scaffale. I parametri di utilità δ_c^{occ} e δ_c^{bas} (tabella 3.1) quantificano questo effetto di visibilità per ciascun segmento. Si noti che, nei vincoli di ottimizzazione, è stato scelto di limitare uno stesso prodotto a massimo due posizioni a scaffale contemporanemante.

3.4 Processo di arrivo dei clienti

Il processo di arrivo è modellato attraverso un processo di Poisson non omogeneo in cui il tasso è modellato diversamente per giorno della settimana.

Il tasso di arrivo settimanale

Nello specifico, il tasso medio è fissato a $\mu_0 = 330$ clienti/giorno. I coefficienti settimanali e i tassi giornalieri sono riportati nella tabella 3.3.

Tabella 3.3 Tassi di arrivo giornalieri per il processo di Poisson non omogeneo. κ_d è il coefficiente di modulazione e λ_d è il tasso giornaliero risultante.

Giorno	Lun	Mar	Mer	Gio	Ven	Sab	Dom
κ_d	0.80	0.70	1.00	0.40	0.80	1.50	0.00
λ_d (clienti/giorno)	264	231	330	132	264	495	0

Interpretazione economica dei coefficienti settimanali. Il profilo settimanale di κ_d riflette le seguenti osservazioni:

- Il **sabato** è il giorno con maggiore affluenza, che si discosta del 50% in aumento rispetto al valore standard ($\kappa_{\text{Sab}} = 1{,}5$, $\lambda_{\text{Sab}} = 495$)
- Il **mercoledì** è il secondo giorno più ricercato ($\kappa_{\text{Mer}} = 1{,}0$).
- Il **giovedì** è il giorno più scarso ($\kappa_{\text{Gio}} = 0{,}4$, $\lambda_{\text{Gio}} = 132$): coerentemente con il picco del giorno precedente e la vicinanza del weekend.
- La **domenica** ($\kappa_{\text{Dom}} = 0$): si assume chiusura domenicale.

Il numero totale atteso di clienti per settimana è:

$$\Lambda = \sum_{d=0}^{6} \lambda_d = 1{,}716 \text{ clienti/settimana.} \tag{3.7}$$

La domanda per singolo prodotto

La domanda settimanale totale per il prodotto j in uno scenario s è la somma delle domande giornaliere. Si estrae sfruttando la distribuzione multinomiale con èprobabilità data dal modello di scelta Latent Class e il processo degli arrivi poissoniani descritto precedentemente.

$$D_j^{(s)} = \sum_{d=1}^{7} D_{jd}^{(s)}, \tag{3.8}$$

$$\left(D_{1d}^{(s)}, \ldots, D_{Jd}^{(s)}, D_{0d}^{(s)}\right) \;\middle|\; N_d^{(s)} \sim \text{Multinomiale}\left(N_d^{(s)}, P(1), \ldots, P(J), P(0)\right), \tag{3.9}$$

dove $P(j)$ sono le probabilità di scelta calcolate con il modello LCCM (Sezione 2.1).

3.5 Il problema di ottimizzazione

In questa sezione si fornisce la formulazione matematica completa del problema di ottimizzazione dell'assortimento e delle quantità da ordinare. Presenteremo tutte le due variazioni considerate: il problema di programmazione stocastica a due stadi (Two-Stage Stochastic Program, TSP) e la semplificazione dello stesso consdierando il valore atteso della domanda (Expected Value, EV). Il TSP è presentato sotto due formulazioni per risolvere il secondo stadio: la prima segue la logica del SAA, come descritto in 2.2; il secondo prevede la formalizzazione del ricordo attraverso una rete MLP, quindi si descrive come modello surrogato 3.5. Il modello EV 3.5 è stato proposto per provare a considerare nella solzuione ottimale i due livelli di sostituzione: quella a catalogo, valutata direttamente dal modello di domanda, e quella in-store gestita come nella formulazione SAA.

Parametri

Economici sul prodotto Fissiamo $\mathscr{I}$ per definire l'insieme di tutti i prodotti, considerando prodotti diversi anche lo stesso prodotto ma non sullo stesso scaffale; $\mathscr{J}$ per l'insieme dei *prodotti unici* (ovvero senza guardare lo scaffale, quindi definito da marca, tipologia di prodotto e prezzo); $\mathscr{I}_l$ per l'insieme delle opzioni la cui

posizione è l; $\mathscr{I}(j) \subseteq \mathscr{I}$ per definire l'insieme delle tre opzioni scaffale associate a j e $\mathscr{L} = \{\text{Alto, Occhi, Basso}\}$ per l'insieme delle posizioni di scaffale disponibili.

$p_i \geq 0$ prezzo di vendita (€/kg) del prodotto $i \in \mathscr{J}$;

$c_i \geq 0$ costo di acquisto per il retailer (€/kg) di i;

m_i il margine per prodotto (€/kg) di i;

$h_i \geq 0$ costo di mantenimento a scaffale (€/kg per settimana) di i;

$f_l \geq 0$ costo fisso di inserimento in assortimento per la posizione $l \in \mathscr{L}$, con $f_{\text{Alto}} = 6{,}0$ €, $f_{\text{Occhi}} = 8{,}0$ €, $f_{\text{Basso}} = 5{,}0$ €.

Penalità

$\psi_i = m_i$ penalità per la domanda persa (lost sales) del prodotto i; pari al margine di profitto che viene perso non vendendo il prodotto;

$\rho_i = 0{,}2$ penalità per la sostituzione in uscita dal prodotto i. Usata per ridurre l'importanza della vendita sostituta e spingere il modello a preferire il soddisfacimento della domanda primaria;

$\phi = 2{,}0$ penalità associata alla violazione del livello di servizio, calcolato attraverso una variabile di slack, considerato come aggragato su tutti i prodotti;

$\eta = 0{,}95$ livello di servizio target. Questo significa che possiamo perdere al massimo il $(1-\eta) = 5\%$ della domanda aggregata senza ricevere una penalità aggiuntiva.

Capacità degli scaffali Fissiamo $\mathscr{I}^S$ per le opzioni prodotto-scaffale nello *scaffale secco* e $\mathscr{I}^F$ per *scaffale frigo* (gnocchi)

$$\mathscr{I}^S \cup \mathscr{I}^F = \mathscr{I}, \quad \mathscr{I}^S \cap \mathscr{I}^F = \emptyset$$

C_l^S capacità dello scaffale secco per la posizione l (kg);

C_l^F capacità dello scaffale frigo per la posizione l (kg);

Fattori stocastici Fissiamo $\mathscr{S}$ per l'insieme degli scenari considerati nel ricorso, nel nostro caso $|\mathscr{S}| = N_T = 100$.

$d_i^s \geq 0$ domanda settimanale (kg) per il prodotto $i \in \mathscr{I}$ nello scenario $s \in \mathscr{S}$;

$\pi^s > 0$ probabilità dello scenario s, con $\sum_{s\in\mathscr{S}} \pi^s = 1$;

S_{ik} probabilità di sostituzione dal prodotto i al prodotto k (compresa la no-buy).

Variabili decisionali

Di primo stadio. Le decisioni di primo stadio sono prese *prima* che il fattore di rischio si realizzi e rimangono fisse anche dopo la realizzazione di esso.

$a_i \in \{0,1\}$ variabile binaria per definire i prodotti in assortimento, con $i \in \mathscr{I}$;

$y_i \in \mathbb{N}_{\geq 0}$ variabile per definire la quantità ordinata per il prodotto $i \in \mathscr{I}$.

La variabile binaria rende il problema NP-hard, poichè si tratta di un MILP.

Di secondo stadio. Indichiamo $\mathscr{E}$ come insieme delle coppie in cui può avvenire la sostituzione. (i,k) con $i,k \in \mathscr{I}$, $S_{ik} > 0$, indica quanto della domanda i può andare in k.

Le variabili di secondo stadio vengono stabilite in successione alla realizzazione dell'incertezza, quindi il loro valore dipende direttamente dallo scenario $s \in \mathscr{S}$:

$x_{is} \in [0, d_i^s]$ indica quanta della domanda primaria di $i \in \mathscr{I}$ viene soddisfatta dallo stock di i stesso;

$\sigma_{(ik)s} \geq 0$ per $(i,k) \in \mathscr{E}$: quanto dei clienti che non trovano i e decidono di comprare k;

$r_{is} \geq 0$ rimanenza a magazzino del prodotto i;

$\ell_{is} \in [0, d_i^s]$ domanda persa del prodotto i;

$v_s \geq 0$ variabile di slack per il vincolo di livello di servizio. Si attiva quando superiamo il limite imposto e conta la quantità di prodotti extra.

Per considerare tutto il flusso di sostituzione che entra o che esce da un prodotto, definiamo:

$$\sigma_{is}^{\text{out}} = \sum_{k:\,(i,k)\in\mathscr{E}} \sigma_{(ik)s} \quad \text{(sostituzione in uscita da } i\text{)}, \tag{3.10}$$

$$\sigma_{is}^{\text{in}} = \sum_{k:\,(k,i)\in\mathscr{E}} \sigma_{(ki)s} \quad \text{(sostituzione in entrata verso } i\text{)}. \tag{3.11}$$

Funzione obiettivo

La funzione obiettivo, oltre a considerare il valore reale del profitto del retailer, che vuole massimizziare, cerca di tenere conto di quegli elementi che non hanno un effettivo valore di costo, ma che risultano come penalizzanti per il negozio.

$$\max_{a,y,x_s,\sigma_s,r_s,\ell_s,v_s} \quad \underbrace{\sum_{s\in\mathscr{S}} \pi^s\, Q(a,y,s)}_{\text{profitto atteso del secondo stadio}} \quad - \quad \underbrace{\sum_{i\in\mathscr{I}} \left(c_i\, y_i + f_{l(i)}\, a_i\right)}_{\text{costi certi del primo stadio}} \tag{3.12}$$

dove $l(i)$ denota la posizione di scaffale dell'opzione i, e il valore del ricorso $Q(a,y,s)$ è definito come:

$$Q(a,y,s) = \sum_{i\in\mathscr{I}} \left[p_i\left(x_{is} + \sigma_{is}^{\text{in}}\right) - \psi_i\, \ell_{is} - \rho_i\, \sigma_{is}^{\text{out}} - h_i\, r_{is}\right] - \phi\, v_s. \tag{3.13}$$

Nello specifico abbiamo:

- $p_i(x_{is} + \sigma_{is}^{\text{in}})$: il ricavo ottenuto dalle vendite del prdotto i, suddivise in vendite primarie e di sostituzione in entrata;
- $-\psi_i \ell_{is} = -m_i \ell_{is}$: la perdita del guadagno quando non riusciamo a soddisfare la domanda che arriva;
- $-\rho_i \sigma_{is}^{\text{out}} = -0{,}2\,\sigma_{is}^{\text{out}}$: per penalizzare il fatto che il cliente abbia dovuto sostituire la sua prima scelta;

- $-h_i r_{is}$: costo di mantenimento delle rimanenze di fine settimana;
- $-\phi v_s$: penalità per violazione del livello di servizio aggregato.

Vincoli

Primo Stadio:

Vincolo della posizione a scaffale. Ogni prodotto $j \in \mathscr{J}$ può essere collocato in al più in due posizioni di scaffale. La mutua esclusione è relativa allo Scaffale Alto e Scaffale Basso: non si può esporre lo stesso prodotto in entrambe le posizioni contemporaneamente. Proponiamo la notazione SOS1, usata su Gurobi:

$$\{a_{j,\text{Alto}}, a_{j,\text{Basso}}\} \in \text{SOS1}, \qquad \forall j \in \mathscr{J}, \tag{3.14}$$

questa formulazione è preferibile in Gurobi poichè consente strategie che velocizzano la risoluzione del MILP. [10].

Vincoli di capacità. Abbiamo i vincoli di capacità degli scaffali, quindi la quantità di prodotti che possiamo effettivamente esporre è limitata da questo valore. Per ogni posizione l e per ogni categoria della merce $\mathscr{I}_l$ associata a quella posizione, si impone:

$$\sum_{i \in \mathscr{I}_l^S} y_i \leq C_l^S, \qquad \forall l \in \mathscr{L}, \tag{3.15}$$

$$\sum_{i \in \mathscr{I}_l^F} y_i \leq C_l^F, \qquad \forall l \in \mathscr{L}. \tag{3.16}$$

Vincolo di collegamento assortimento-scorte (big-M). Per tradurre in una formulazione lineare il vincolo $y_i \cdot a_i = 0$, usiamo la tecnica della big-M:

$$y_i \leq M \cdot a_i, \qquad \forall i \in \mathscr{I}. \tag{3.17}$$

quindi se un prodotto non è in assortimento ($a_i = 0$), non può essere ordinato; altrimenti lo limitiamo con il valore di M, che deve essere stabilito in modo tale che non sia nè troppo grande nè troppo stringente.

Secondo Stadio (che valgono $\forall s \in \mathscr{S}$):

Vincolo di domanda primaria. Le vendite primarie non possono superare la domanda realizzata, né si possono vendere prodotti non in assortimento:

$$x_{is} \leq d_i^s \cdot a_i, \qquad \forall i \in \mathscr{I},\ s \in \mathscr{S}. \tag{3.18}$$

Vincoli di sostituzione. Per ogni coppia $(i,k) \in \mathscr{E}$, la domanda che transita da i a k deve tenere conto della domanda residua per i, quindi effettivamente la quantità che si deve spostare su altri prodotti, e anche della quantità di stock rimanente disponibile di k:

$$\sigma_{(ik)s} + S_{ik}\, x_{is} \leq S_{ik}\, d_i^s, \qquad \forall (i,k) \in \mathscr{E},\ s \in \mathscr{S}, \tag{3.19}$$

$$\sigma_{(ik)s} \leq M \cdot a_k, \qquad \forall (i,k) \in \mathscr{E},\ s \in \mathscr{S}. \tag{3.20}$$

Il vincolo (3.19) ci dice che la percentuale di clienti che preferivano il prodotto i ma che non lo hanno trovato, ovvero $d_i^s - x_{is}$, definita da S_{ik}, si può soddisfare con il prodotto sostituto k, ovvero $\sigma_{(ik)s}$. Il vincolo (3.20) limita la possibilità di andare verso un prodotto, quindi sceglierlo come sostituto, solo se esso si trova in assortimento.

Bilancio delle scorte. Per ogni prodotto i, lo stock in assortiento y_i deve tenere conto delle vendite primarie, delle vendite di sostituzione che arrivano a i e dell rimanenze:

$$x_{is} + \sigma_{is}^{\text{in}} + r_{is} = y_i, \qquad \forall i \in \mathscr{I},\ s \in \mathscr{S}. \tag{3.21}$$

Bilancio della domanda. La domanda del prodotto i si divide tra quanto sono riuscito a soddisfare con la quantità di prodotto stesso, più quanto si è spostato su altri prodotti, più quanto non sono riuscito a soddisfare:

$$x_{is} + \ell_{is} + \sigma_{is}^{\text{out}} = d_i^s, \qquad \forall i \in \mathscr{I},\ s \in \mathscr{S}. \tag{3.22}$$

Livello di servizio. Il *livello di servizio* ci dice quanto è il target di domanda che si vuole soddisfare. Quindi fissando il livello di servizio con $\eta \in (0,1)$ stiamo richiedendo che la domanda persa totale non superi il $(1-\eta)$ della domanda totale.

In questo caso calcoliamo il livello di servizio come vincolo *soft*, quindi non fissiamo esattamente che non venga superata la soglia di domanda, ma consideriamo la penalizzazione di tutte le unità che eccedono questo bound.:

$$\sum_{i \in \mathscr{I}} \ell_{is} - v_s \leq (1-\eta) \sum_{i \in \mathscr{I}} d_i^s, \qquad \forall s \in \mathscr{S}. \tag{3.23}$$

In questo modo andiamo a disincentivare scenari in cui si ottiene un livello di servizio troppo basso.

Formulazione completa SAA

La formulazione completa del modello SAA è:

$$\max \quad \sum_{s\in\mathscr{S}} \pi^s \left[\sum_{i\in\mathscr{I}} \Big(p_i(x_{is}+\sigma^{\text{in}}_{is}) - \psi_i \ell_{is} - \rho_i \sigma^{\text{out}}_{is} - h_i r_{is} \Big) - \phi\, v_s \right] - \sum_{i\in\mathscr{I}} (c_i y_i + f_{l(i)} a_i) \tag{3.24a}$$

$$\text{s.t.} \quad \{a_{j,\text{Alto}}, a_{j,\text{Basso}}\} \in \text{SOS1}, \qquad \forall j \in \mathscr{J}, \tag{3.24b}$$

$$\sum_{i\in\mathscr{I}^S_l} y_i \le C^D_l, \qquad \forall l \in \mathscr{L}, \tag{3.24c}$$

$$\sum_{i\in\mathscr{I}^F_l} y_i \le C^F_l, \qquad \forall l \in \mathscr{L}, \tag{3.24d}$$

$$y_i \le M a_i, \qquad \forall i \in \mathscr{I}, \tag{3.24e}$$

$$x_{is} \le d^s_i a_i, \qquad \forall i \in \mathscr{I},\ s \in \mathscr{S}, \tag{3.24f}$$

$$\sigma_{(ik)s} + S_{ik} x_{is} \le S_{ik} d^s_i, \qquad \forall (i,k) \in \mathscr{E},\ s \in \mathscr{S}, \tag{3.24g}$$

$$\sigma_{(ik)s} \le M a_k, \qquad \forall (i,k) \in \mathscr{E},\ s \in \mathscr{S}, \tag{3.24h}$$

$$x_{is} + \sigma^{\text{in}}_{is} + r_{is} = y_i, \qquad \forall i \in \mathscr{I},\ s \in \mathscr{S}, \tag{3.24i}$$

$$x_{is} + \ell_{is} + \sigma^{\text{out}}_{is} = d^s_i, \qquad \forall i \in \mathscr{I},\ s \in \mathscr{S}, \tag{3.24j}$$

$$\sum_{i\in\mathscr{I}} \ell_{is} - v_s \le (1-\eta) \sum_{i\in\mathscr{I}} d^s_i, \qquad \forall s \in \mathscr{S}, \tag{3.24k}$$

$$a_i \in \{0,1\},\ y_i \in \mathbb{N}_{\ge 0}, \qquad \forall i \in \mathscr{I}, \tag{3.24l}$$

$$x_{is}, r_{is}, \ell_{is}, v_s \ge 0, \qquad \forall i \in \mathscr{I},\ s \in \mathscr{S}, \tag{3.24m}$$

$$\sigma_{(ik)s} \ge 0, \qquad \forall (i,k) \in \mathscr{E},\ s \in \mathscr{S}. \tag{3.24n}$$

Analisi della struttura del modello

Dimensioni e complessità. Nella tabella 3.4 mostriamo la dimensione di ogni insieme considerato rispetto al dataset che abbiamo costruito. Questo ci permette di capire esattamente con quale struttura di complessità stiamo lavorando. Si noti che il valore di $|\mathscr{E}|$ è calcolato sul il numero di coppie di sostituzione della matrice S_{ik} sparsa, quindi quella definita fissando la soglia massima per considerare un elemento nullo.

Tabella 3.4 Dimensioni del modello SAA.

Elemento	**Valore**	
$\lvert\mathscr{I}\rvert$ (opzioni prodotto-scaffale)	111	marche $\times$ tipi $\times$ 3 scaffali
$\lvert\mathscr{J}\rvert$ (prodotti unici)	36	marche $\times$ tipi
$\lvert\mathscr{S}\rvert$ (scenari)	100	Scenari K-Means ridotti
Variabili binarie a_i	111	Primo stadio
Variabili intere y_i	111	Primo stadio
Variabili continue (x, r, ℓ)	$3 \times 111 \times 100$	Secondo stadio
Variabili di sostituzione $\sigma_{(ik)s}$	$\lvert\mathscr{E}\rvert \times 100$	Secondo stadio
Variabili di slack v_s	100	Secondo stadio
Vincoli SOS1	36	Uno per prodotto unico
Vincoli di capacità	$2 \times 3 = 6$	Secco + frigo per 3 posizioni
Vincoli big-M	$2 \times 111 = 222$	Link $a \leftrightarrow y$

Il modello surrogato

La formulazione matematica del modello surrogato, prendendo spunto dal lavoro presentato nel paper "*Solving Two-Stage Stochastic Programming Problems via Machine Learning*" [12] è strettamente collegata al modello SAA 3.5. In questo caso, la funzione di valore del ricorso $Q(a, y, s)$ è rimpiazzata dall'output $\hat{Q}_\theta(y)$ di una rete neurale surrogata Ψ, eliminando le variabili di secondo stadio e sostituendo i vincoli (3.24f)–(3.24k) con i vincoli MILP della rete neurale (vincoli ReLU linearizzati, Teorema 2.4.2). Mentre primo stadio rimane invariato, il secondo stadio scompare: questo implica che l'ottimizzatore non dovrà più considerare tutte quelle variabili del secondo stadio, ma conosce direttamente il profitto del secondo stadio guardando solamente alla quantità di prodotti in assortimento.

Quindi, dato come input $[y_1, y_2, ..., y_{111}]$ (questo vettore comprende la conoscenza dell'assortimento siccome $y > 0 \iff a > 0$), l'output $\hat{Q}_\theta(y)$ rappresenta la predizione del ricorso della rete Ψ. Questo valore viene sfruttato nel modello aggiungendolo come vincolo usando la funzion fornita da Gurobi Machine Learning [10].

La formulazione completa del modello Surrogato è:

$$\max \quad \hat{Q}_\theta(y) - \sum_{i \in \mathscr{I}} (c_i y_i + f_{l(i)} a_i) \tag{3.25a}$$

$$\text{s.t.} \quad \{a_{j,\text{Alto}}, a_{j,\text{Basso}}\} \in \text{SOS1}, \qquad \forall j \in \mathscr{J}, \tag{3.25b}$$

$$\sum_{i \in \mathscr{I}_l^D} y_i \leq C_l^D, \qquad \forall l \in \mathscr{L}, \tag{3.25c}$$

$$\sum_{i \in \mathscr{I}_l^F} y_i \leq C_l^F, \qquad \forall l \in \mathscr{L}, \tag{3.25d}$$

$$y_i \leq M a_i, \qquad \forall i \in \mathscr{I}, \tag{3.25e}$$

$$y_i \geq a_i, \qquad \forall i \in \mathscr{I}, \tag{3.25f}$$

$$add\_predictor\_constr(\Psi, y, \hat{Q}_\theta(y)) \tag{3.25g}$$

Il modello ibrido: deterministico

Sostituzione a livello di catalogo vs sostituzione a livello di stockout

Per analizzare correttamente la struttura del modello proposto, rispetto a quello precedente, è importante trattare due meccanismi di sostituzione che avvengono i due momenti temporali diversi:

Sostituzione a livello di catalogo (*assortment-level substitution*). Il consumatore osserva l'*assortimento offerto* (**a**) prima di effettuare la scelta, quindi la sua utilità considera *i prodotti disponibili in catalogo*. Questo modella la sostituzione *a priori*: il modello LCCM calcola le probabilità di scelta su tutti i prodotti i con $a_i = 1$, e la domanda attesa è:

$$\bar{d}_i(\mathbf{a}) = \Lambda \cdot P_i(\mathbf{a}) = \Lambda \cdot \frac{\sum_c \pi_c \exp(V_{ci}) \cdot a_i}{\sum_c \pi_c \sum_{j:a_j=1} \exp(V_{cj}) + \exp(V_{c0})}, \tag{3.26}$$

Sostituzione a livello di stock-out (*stock-out substitution*). Dopo che il consumatore entra in negozio può succedere che il proprio prodotto preferito non sia disponibile. A questo punto si può verificare la sostituzione per stock-out, quella descritta dalla matrice **S**.

La differenza è schematizzata nella figura seguent:.

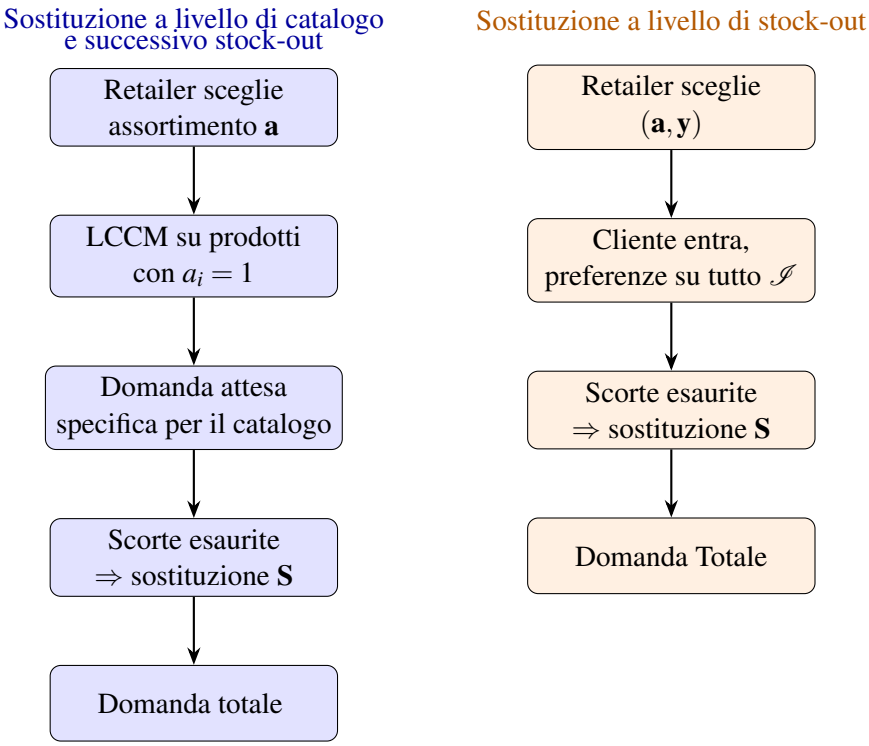


Figura 3.8 Le due logiche di sostituzioni

Un aspetto cruciale, che distingue il modello ibrido dal modello SAA, è che esso cerca di considerare entrambe le sostituzioni in sequenza. La sostituzione a catalogo definisce la domanda primaria che entra nel modello di ottimizzazione attracerso il predittore lineare, che è addestrato su dati generati applicando il modello LCCM, quindi la domanda che il modello impara già incorpora il fatto che i prodotti fuori catalogo non esistono. La sostituzione per stock-out è invece considerato all'interno del modello di ottimizzazione attraverso le variabili di sostituzione $\sigma_{(ik)}$, il bilancio delle scorte e il bilancio della domanda (vincoli (3.30f)–(3.30h)).

Formulazione del modello ibrido

Il modello di ottimizzazione sostituisce la domanda stocastica con la predizione deterministica del regressore lineare. Focalizzarsi solo sul valore medio dell'input stocastico, permette di ridurre la complessità degli scenari inserendo la complessità della doppia sostituzione, per passare al modello di ottimizzazione il comportamento dei clienti con un catalogo dell'assortimento disponibile a priori.

Dati $\hat{\boldsymbol{\beta}}_i \in \mathbb{R}^m$ che definiscono i vettori dei coefficienti appresi e $[y_{\min,i}, y_{\min,i} + \delta_i]$ l'intervallo in cui viene riscalato l'input e utilizzato come target del predittore, la domanda predetta è:

$$d_i^{\text{pred}} = \hat{\boldsymbol{\beta}}_i^{\top} \mathbf{a} \cdot \delta_i + y_{\min,i}, \qquad i = 1, \ldots, m, \tag{3.27}$$

La Δ_{no-buy} predetta, ovvero l'ultimo output del regressore, è:

$$\Delta_{no-buy}^{\text{pred}} = \hat{\boldsymbol{\beta}}_{no-buy}^{\top} \mathbf{a} \cdot \delta_{no-buy} + y_{\min,no-buy}. \tag{3.28}$$

Per permettere di penalizzare i clienti persi a causa del catalogo, inseriamo nella funzione obiettivo un elemento che vada a tracciare queste *lost*:

$$\psi_{mean} \Delta_{no-buy}^{\text{pred}}, \tag{3.29}$$

dove ψ_{mean} è la media dei valori di penalità dati alle vendite perse nel modello SAA.

La formulazione del modello ibrido deterministico è:

$$\max_{\mathbf{a},\mathbf{y},\mathbf{x},\boldsymbol{\sigma},\mathbf{r},\boldsymbol{\ell},v} \quad \sum_i p_i (x_i + \sigma_i^{\text{in}}) - \sum_i \psi_i \ell_i - \psi_{mean} \Delta_{no-buy}^{\text{pred}} - \sum_i h_i r_i - 0{,}2 \sum_i \sigma_i^{\text{out}} - \phi\, v - \sum_i (c_i y_i + f_{l(i)} a_i) \tag{3.30a}$$

$$\text{s.t.} \quad d_i^{\text{pred}} = \hat{\boldsymbol{\beta}}_i^{\top} \mathbf{a} \cdot \Delta_i + y_{\min,i}, \qquad \forall i, \tag{3.30b}$$

$$\Delta_{\text{no-buy}}^{\text{pred}} = \hat{\boldsymbol{\beta}}_{\text{no-buy}}^{\top} \mathbf{a} \cdot \delta_{\text{no-buy}} + y_{\min,\text{no-buy}}, \tag{3.30c}$$

[vincoli di scaffali, capacità e big-M come nel modello SAA], (3.30d)

$$x_i \le M \cdot a_i, \qquad \forall i, \tag{3.30e}$$

$$\sigma_{(ik)} + S_{ik} x_i \le S_{ik}\, d_i^{\text{pred}}, \qquad \forall (i,k) \in \mathscr{E}, \tag{3.30f}$$

$$x_i + \sigma_i^{\text{in}} + r_i = y_i, \qquad \forall i, \tag{3.30g}$$

$$x_i + \ell_i + \sigma_i^{\text{out}} = d_i^{\text{pred}}, \qquad \forall i, \tag{3.30h}$$

$$\sum_i \ell_i - v \le (1 - \eta)\, \bar{D}_{\text{tot}}, \tag{3.30i}$$

$$a_i \in \{0,1\},\ y_i \in \mathbb{N}_{\ge 0},\ x_i, r_i, \ell_i, v \ge 0. \tag{3.30j}$$

dove $\bar{D}_{\text{tot}}$ è la domanda totale media calcolata su 300 scenari, scelto come valore di

riferimento vero della domanda per poter calcolare facilmente il livello di servizio.

Quindi possiamo notare che, visto che la struttura del modello di ottimizzazione è molto simile al modello SAA, effettivamente il modello ibrido risolve:

$$\max_{\mathbf{a},\mathbf{y}} \; f(\mathbf{a},\mathbf{y}) + Q\big(\mathbf{a},\mathbf{y},\, \hat{\mathbf{d}}^{\text{lin}}(\mathbf{a})\big), \tag{3.31}$$

dove la funzione di ricorso Q è quella del modello SAA (3.13), ma $\boldsymbol{\xi}$ è sostituita dalla predizione lineare $\hat{\mathbf{d}}^{\text{lin}}(\mathbf{a})$.

La tabella 3.5 riassume le differenze tra il modello ibrido, il modello surrogato e il modello SAA.

Tabella 3.5 Confronto strutturale tra i modelli di ottimizzazione considerati.

Modello	**Predittore**	**Sostituzione**	**Scenari**	**Domanda**
SAA	—	Stockout	100	Stocastica
NN Surrogato	MLP $\mathbf{y} \to Q$	Stockout	0	Stocastica
Ibrido Deterministico	Ridge $\mathbf{a} \to \bar{\mathbf{d}}$	Catalogo + Stockout	0	Deterministica

Osservazione 3.5.1 (Comparabilità dei modelli)**.** *Il modello ibrido, il modello SAA e surrogato differiscono quindi per il modo in cui viene costruita la domanda. Questa differenza ha una conseguenza pratica: il confornto del modello ibrido deterministico con gli altri, non può essere fatto in modo completo. Nel capitolo di commento ai risultati, verranno comunque prosposte e analizzate le soluzioni di tutti e tre i modelli, ottenute valutando l'assortimento scelto su scenari di domanda 'normali' e di 'stress'.*

Capitolo 4

Dettagli di implementazione

Nel seguente capitolo verranno accennati i principali script che hanno prodotto i risultati del processo di ottimizzazione. Per una logica più dettagliata, leggere l'Appendice A. Il codice è implementato con Python.

Introduciamo i due file strutturali nella logica implementativa. La struttra della simulazione di mercato è presente nello script `models.py`. Per quanto riguarda i modelli di ottimizzazione, si fa riferimento a `optimization.py`.

In generale, tutto il flusso dell'architettura è pensata in modo da renderla estendibile ad altre varianti: sia per quanto riguarda la simulazione che per quanto riguarda l'ottimizzazione. Per cui, volendo aggiungere una nuova variante per l'ottimizzazione, basterebbe implementare un nuovo metodo nella classe `AssortmentOptimize`. Stessa cosa per quanto riguarda il mercato: l'idea è aggiungere una nuova classe relativa al modello di scelta che si preferisce usare.

Panoramica dell'implementazione L'implementazione ha una struttra a livelli diramata in tre rami principali: nel ramo centrale abbiamo un livello di simulazione del mercato (`models.py`), un livello di preparazione dei dati (`prepare_data.py`) e un livello di ottimizzazione (`optimization.py`).

Nel ramo di training: generazione dei dataset (`create_{nome_modello}_dataset.py`) e il training dei predittori (`train_{nome_modello}.py`).

Infine nel ramo di test abbiamo gli script relativi alla calibrazione del modello di domanda (`test_{nome_test_condotto}.py`) e quelli legati alla valutazione delle

soluzioni dell'ottimizzazione nei vari scenari di test considerati (`function_test_OOS.py`).

Nella tabella seguente facciamo un veloce recap delle principali classi e metodi che si trovano all'interno dei file di base:

`prepare_data.py`	Preparazione dati e generazione dataset	prepare_input_data(), prepare_NN_training_data(), prepare_arrivals_det()
`models.py`	Simulazione del mercato e della domanda	MarketSimulation, MNLChoiceModel, LatentChoiceModel, ClientArrivals
`optimization.py`	Formulazione e risoluzione dei modelli	AssortmentOptimizer (SAA, NN, Hybrid_det)
`function_test_OOS.py`	Script che presenta le funzioni usate nei test *out-of-sample*	run_out_of_sample(), simulation_results_file(), e grafici

Nei file di `main_{nome}.py` legati all'ottimizzazione si parte sempre dalla funzione prepare_input_data(), che carica i dati di prodotto, esegue la simulazione del mercato, genera gli scenari di domanda e restituisce un dizionario contenente tutti i parametri necessari. Successivamente costruiscono il modello di ottimizzazione corrispondente, lo risolvono e salvano i risultati del'ottimizzazione in un file di testo.

Librerie Python e Gurobi Per l'implementazione è necessario installare i seguenti pacchetti:

`gurobipy`	Formulazione e risoluzione dell'ottimizzazione	gp.Model, GRB, MVar, addConstr
`gurobi-ml`	Integrazione predittori ML in Gurobi	add_predictor_constr
`scikit-learn`	Costruzione dei predittori	MLPRegressor, RidgeCV, MinMaxScaler, StandardScaler
`optuna`	Ottimizzazione degli iperparametri	create_study, optimize
`numpy / pandas / scipy.sparse`	Usare dati in forma matriciale	np.dot, pd.read_excel, pd.DataFrame, coo_array
`joblib`	Salvataggio modelli	dump, load
`matplotlib`	Visualizzazione grafica	plt.plot, plt.show

Capitolo 5

Analisi e commento ai risultati

In questo capitolo si valutano i risultati numerici prodotti dai tre approcci implementati: il metodo SAA con 100 scenari ridotti (valutato su 10 repliche indipendenti), il surrogato che sfrutta la rete neurale MLP per apprendere il valore del ricorso e il modello EV deterministico con predittore della domanda Ridge. Per ciascun modello si analizzano prima le fasi di addestramento e ottimizzazione, poi si commentano le scelte di assortimento osservando le prestazioni out-of-sample sia in condizioni di mercato ordinarie sia sotto scenari di stress. In questa fase, l'analisi verrà condotta sottolineando l'interpretazione economica dei risultati.

5.1 Modello SAA

5.1.1 Complessità computazionale e convergenza

Il problema stocastico a due stadi, viene valutato su 100 scenari, ottenuti da una riduzione a partire da 200.000 scenari usando K-Means, e vengono fatte 10 repliche. L'utilizzo di K-Means è stato deciso poichè le dimensioni del rilassamento MILP erano molto elevate. Infatti grazie ai log di Gurobi, vediamo che il modello risultante presenta **355.722 coefficienti lineari** nella funzione obiettivo, con variabili sia binare che continue. Il rilassamento continuo richiede 64.000-68.000 iterazioni simplex solo alla radice dell'albero. Questo ha portato a capire che non si poteva valutare l'ottimizzazione su un numero elevato di scenari. In questo modo, pur tenendoli limitati a 100, si considerano degli scenari rappresentativi con pesi probabilistici

associati. Per ogni cluster, il centroide rappresenta lo scenario tipico e il suo peso è proporzionale al numero di scenari originali assegnati a quel cluster.

Vediamo quindi che gli scenari di domanda vengono generati come segue:

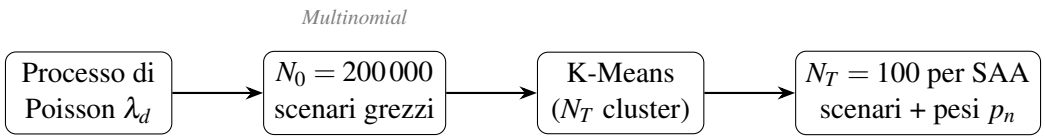


Figura 5.1 Come vengono generati e ridotti degli scenari di domanda settimanale.

Inoltre, per supportare il solutore a raggiungere la convergenza in tempi più contenuti, si sono settati i seguenti parametri:

`'Method'=4` Parametro che definisce il metodo con cui viene risolto il rilassamento continuo. Il valore 4 è utile per rilassamenti MIP grandi alla radice;

`'Heuristics'=0.3` Parametro che controlla la frazione di tempo di run che viene dedicato alla ricerca di soluzioni ammissibili attraverso euristiche;

`'MIPFocus'=2` Parametro che definisce il trade-off tra ricerca di una soluzione ammissibile e dimostrazione che la soluzione trovata è effettivamente ottima. Il valore 2 posiziona l'attenzione sul dimostrare che la soluzione trovata è quella ottimale;

`'Presolve'=2` Parametro per definire l'aggressività del presolve. Con un valore di 2 si è molto aggressivi;

`'Symmetry'=2` Parametro che valuta le simmetrie del problema, settandolo a 2 lo si rende più attento a tali presenze;

`'MIPGap'=0.1` Parametro di stop. Dice al solutore di fermarsi quando giunge ad un delta tra soluzione ammissibile trovata e bestBound inferiore al 10%;

'Cuts'=2 Parametro per definire l'aggressività dei tagli. Con valore 2, ci si aspetta una generazione di tagli pià aggressiva;

`'NoRelHeurTime'=300` Parametro che imposta un tempo, in secondi, di ricerca euristica di soluzioni fattibili e bestBd attuali, prima di entrare nella risoluzione del rilassamente della radice.

La tabella 5.1 riassume i risultati di ottimizzazione.

Tabella 5.1 Risultati di ottimizzazione per le 10 repliche SAA su 100 scenari. z^*: valore ottimo penalizzato; GAP: *optimality gap* a terminazione; t: tempo di calcolo.

Replica	z^* (€)	GAP (%)	t (h)	Tot. stock (kg)	Lost (unità)
1	−13.65	0.00	14.4	1204	357.85
2	−14.39	9.59	26.8	1204	358.32
3	−14.23	9.95	19.3	1204	358.17
4	−13.82	9.18	13.3	1204	357.77
5	−14.06	6.40	21.7	1204	358.09
6	−13.87	9.49	13.1	1204	357.98
7	−15.45	8.38	13.4	1204	358.57
8	−13.65	0.00	14.4	1204	357.85
9	−15.33	0.00	17.9	1204	358.52
10	−15.62	5.00	18.9	1204	359.15
Media	**−14.41**			1204	358.17
Dev. st.	**0.74**			—	0.42

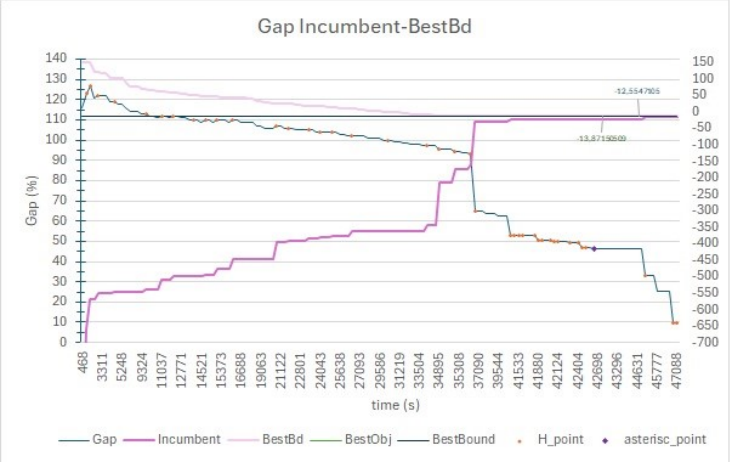
Tempo computazionale. L'ordine di grandezza è di **13–18 ore per replica** con `MIPGap = 0.1`.

Nella Figura 5.2 mostriamo, per due diverse repliche, come l'ottimizzatore giunge a convergenza rispetto al tempo.

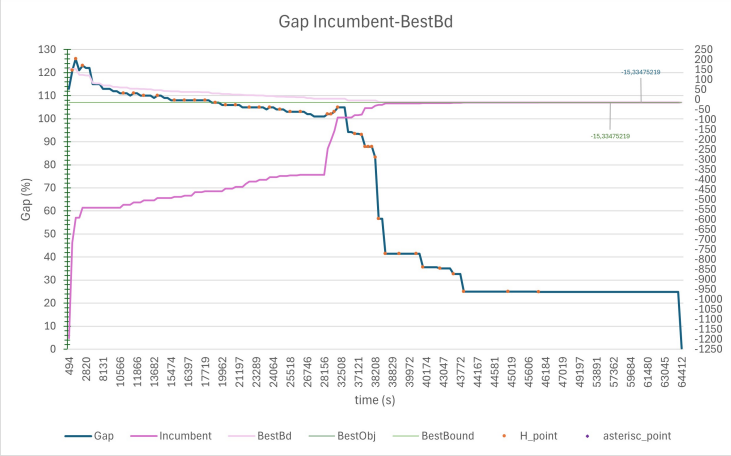
Dal grafico notiamo come i miglioramenti dell'incumbente sono pochi e concentrati, tendenzialmente si verificano quando Gurobi propone un'euristica. Questo va a confermare la struttura combinatoria difficile che caratterizza il problema, sia a causa della presenza delle variabili di assortimento, che sono binarie, sia per i vincoli di mutua esclusione delle stesse.

Valore della funzione obiettivo. Il valore ottimo oscillà tra -13.65 € e -15.62 €, con deviazione standard $\sigma = 0.74$ €. Vediamo che la variazione è inferiore al *15% del valore medio*. Questo valore ci dà conferma del fatto che, nonostange il Gap venga interrotto al 10%, l'approccio del SAA è abbastanza robusto dal punto di vista statistico. Inoltre, se consideriamo una valutazione economica, questa differenza si traduce in soli 2 € di differenza, ovvero non una grande perdita o guadagno per il rivenditore, mentre risulta un grande vantaggio per aiutare il solutore a finire.

É importante fare una precisazione sul segno della soluzione. Il valore $z^* \approx -14$ € è il *profitto penalizzato* e non il guadagno contabile. Infatti, come abbiamo



(a) Replica=6: 13 h, gap finale 9.49%.



(b) Replica=7: 18 h, gap finale 0%.

Figura 5.2 Convergenza del gap Incumbent-BestBd nel tempo per due repliche SAA (asse x: secondi) A sinistra: la curva blu è il gap%; la curva rosa (Incumbent) e quella azzurra (BestBd) mostrano rispettivamente la miglior soluzione ammissibile trovata e il lower bound corrente; i punti arancioni (*H_point*) segnalano nuove soluzioni incumbenti trovate dalle euristiche. A destra (seed = 6791): riesce a chiudere completamente il gap, trovando una soluzione ottima.

definito nella Sezione 3.5, la funzione obiettivo include dei termine di penalità per la domanda insoddisfatta, penalizza leggermente le sostituzioni e va a richiedere un certo livello di servizio. I costi di primo stadio superano il ricavo atteso per la sola ragione che il modello è costruito affinchè il rivenditore possa gestire l'incertezza della domanda e trovare un assortimento ottimale che vada a coprire il più possibile l'intera distribuzione della domanda, compresi gli scenari peggiori.

5.1.2 Analisi dell'assortimento

Tutte le repliche portano a una struttura di assortimento *quasi identica*. I prodotti scelti sono sempre gli stessi, quello che varia leggermente sono le quantità di alcuni. La Figura 5.10 mostra il confronto visivo inter-replica.

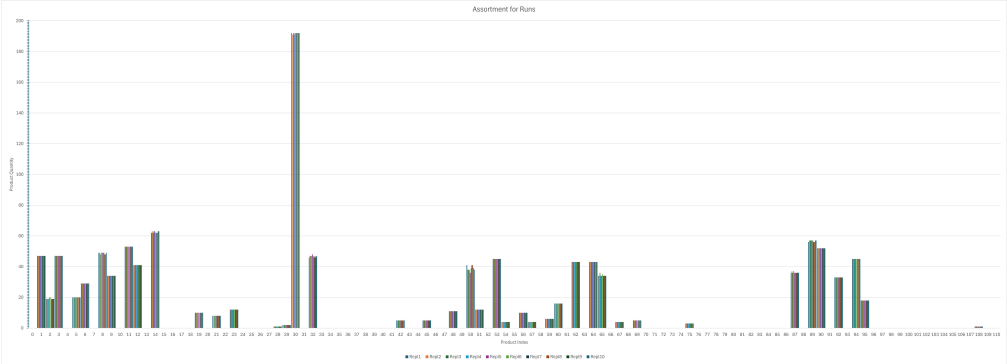
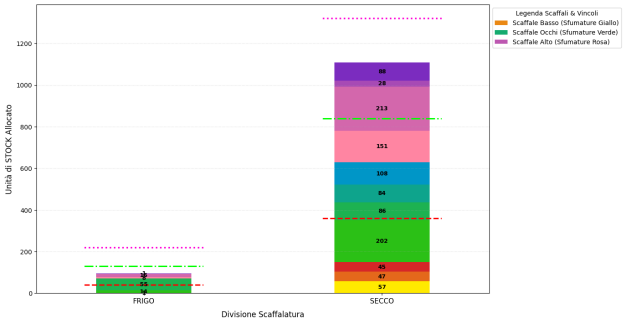


Figura 5.3 Confronto visivo delle scelte di assortimento (quantità per prodotto) nelle 10 repliche SAA.

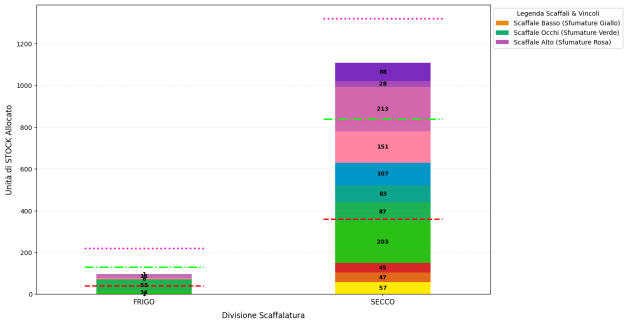
Nelle analisi successive, considerato che gli assortimenti per ogni replica sono abbastanza standardizzati, useremo come riferimento per la replica 7 (**seed = 1691**) e la replica 1 (**seed = 3761**), che presentano le due soluzioni più distanti in termini di quantità dei prodotti e di valore della funzione obiettivo.

Commento dell'assortimento. É interessante vedere come si distribuisce a scaffale l'assortimento che viene scelto. Tutte le repliche ordinano esattamente **1204 kg** di prodotti. All'incirca, vengono ordinati **1109 kg** di secco, vicino al limite di capienza (Alto + Occhi + Basso $= 480 + 480 + 160 = 1120$ kg) e una piccola parte di prodotti freschi, con un ordine di grandezza di **95 kg**, abbastanza inferiore alla capacità totale di 120 kg. Il fatto che tutte le repliche saturino lo stesso livello di stock

è una conferma della robustezza del metodo valutato su 100 scenari: il modello identifica univocamente il livello di riempimento ottimale, pur distribuendo le unità in modo leggermente diverso tra le opzioni. Nella Figura 5.4, possiamo apprezzare visivamente come il vincolo di scaffalatura venga sempre rispettato, ma mai saturato negli scaffali del frigo e in Basso. Questo è indice del fatto che, le preferenze dei clienti, si scontrano con i vincoli fisici del negozio: quindi, siccome c'è maggiore richiesta per articoli Secchi, soprattuto in scaffali Occhi e Alto, l'ottimizzazione porta a saturare tali vincoli. Invece, per lo scaffale Occhi e per tutti i prodotti freschi, c'è meno richiesta: non conviene comprare e allocare i prodotti se non verranno acquistati. Questa analisi si riflette anche negli scenari di test, guardando ai prodotti con maggior perdita di domanda 5.4.



(a) Replica=7: distribuzione sugli scaffali dei prodotti in assortimento.



(b) Replica=1: distribuzione sugli scaffali dei prodotti in assortimento.

Figura 5.4 I grafici rappresentano la suddivisione a scaffale dei prodotti impilandoli come i livelli visivi reali. Si evidenzianno con le sfumature la Marca di riferimento: la sfumatura inferiore è la Marca A fino a quella superiore che rappresenta la Marca D. Inoltre sono presenti i bound effettivi dei vincoli.

Notiamo in particolare che, per i prodotti a scaffale Basso in frigo, non viene inserita nessuna istanza in assortimento. Questa caratteristica evidenzia ulteriormente

come questa combinazione di elementi del prodotto, non vada a genio al consumatore modellato per il nostro mercato. Inoltre, per la scaffalatura Basso dei secchi, non ci sono prodotti relativi alla Marca B, che invece sono leader dello scaffale Alto, poichè sono quelli che portano più clientela e, in questo caso, è anche disposta a guardare sul secondo scaffale in termini di preferenza.

Possiamo fare ulteriori osservazioni relativamente alla struttura dell'assortimento, guardando fisicamente quali prodotti e in quale quantità vengono inseriti:

- **Dominanza della pasta a basso prezzo.** Il prodotto con la maggiore allocazione è Pasta B a 1.30 €/kg in posizione Alto (192 kg su un massimo di 480 kg a scaffale), seguito da diversi prodotti di Pasta A a diversi prezzi, in posizione Alto e Occhi (47-63 kg ciascuna). Il nostro mercato è molto sensibile al prezzo, visto che nelle percentuali di segmento il price-sensitive spicca, quindi questo risultato è coerente.

- **Riso in quantità limitata.** Sono incluse solo due varianti di Riso con quantità significative: Riso A 2.69 €/kg (10 kg, posizione Basso) e Riso B 2.82 €/kg (11 kg Alto + 41 kg Occhi). Il Riso A a 5.38 €/kg e tutte le possibilità C/D sono invece escluse: la loro domanda attesa non giustifica l'occupazione di scaffale.

- **Gnocchi in quantità marginale.** Come detto analizzando il livello di scaffale occupato, gli gnocchi non hanno grande scelta. Gnocchi B 2.78 €/kg (12 kg Alto + 45 kg Occhi) è l'unico prodotto con quantità rilevante; Gnocchi C e D (con prezzi unitari 7-8 €/kg) sono assenti, a causa dell'alto prezzo di acquisto e della preferenza instriseca di mercato.

- **Assenza dei prodotti premium.** Il modello esclude sistematicamente i prodotti con il maggiore prezzo unitario. Una domanda primaria bassa ed elevata variabilità rendono il rischio di rimanenze superiore al ricavo marginale atteso.

5.1.3 Rimanenze, vendite perse e sostituzioni

Sulle 10 repliche, vediamo che l'assortimento proposto, in media, fornisce un buon numero di vendite riuscendo a integrare bene anche la domanda arrivata dalla sostituzione in-store. Purtroppo, i vincoli fisici del negozio non riescono ad assorbire

e rispettare tutte le richieste del mercato: in particolare la struttura costruita delle richieste dei clienti è sbilanciata sui prodotti in vista, e le preferenze faticano a spostarsi verso il basso. In questo modo, pur non riempiendo il negozio, andiamo a saturare gli scaffali importanti. Vediamo infatti nella tabella 5.2 le principali metriche, mediate sugli scenari di ottimizzazione, che sottolineano le vendite perse a causa di questa caratteristica descritta:

Tabella 5.2 KPI valuatati su assortimento SAA (media su 100 scenari).

Metrica	Valore medio	Unità
Stock totale ordinato	1204.00	kg
Rimanenze medie	≈ 14	kg
Vendite perse	357.98	unità
Market share persa totale	27.92%	—
Market share persa a priori	6.41%	—
Clienti potenziali stimati	1664.67	clienti/settimana

Il 6.41% di *lost service* a priori rappresenta il no-buy iniziale: i clienti che, pur considerando la disponibilità completa di prodotti, non trovano nessun prodotto accettabile. Il restante 21.5% (circa) è domanda insoddisfatta per esaurimento scorte (*stock-out*) anche successivo alla dinamica di sostituzione. Le rimanenze medie sono limitate (≈ 14 kg su 1204 ordinati, pari all'1.2%), indice di un assortimento ben calibrato.

É molto interessante guardare le dinamiche di sostituzioni in-store: il modello di ottimizzazione riesce a catturare bene questi valori mostrando che sono numerose ma di entità contenuta. La struttura a scaffale influisce significativamente, come sulla domanda primaria, dal momento che la dinamica di mercato alla base è la stessa. I prodotti in posizione Alto e Occhi ricevono sia domanda primaria che di sostituzione, in modo elevato. Quando il prodotto va in stock-out, la sua domanda viene parzialmente assorbita anche dai prodotti presenti in assortimento ed allocati nello scaffale Basso. I generale, però, i prodotti che ricevono più sostituzione sono Pasta B a 1.30 €/kg nello scaffale Alto con circa 82 unità e nello scaffale Occhi, con circa 47.00 unità. Confermando la superiorità del prodotto nel mercato simulato. Da notare la presenza in assortimento dei prodotti Pasta D a 2.98€/kg in posizione Occhi e di Pasta C a 3.18 €/kg nello stesso scaffale, che vengono inserite quasi ed esclusivamente per rispettare la domanda giunta dalle sostituzioni.

5.2 Modello surrogato

5.2.1 Addestramento della rete neurale surrogata

Generazione del dataset e Active Learning

Il dataset di addestramento richiede la risoluzione di 210000 problemi di ricorso, ciascuno su 30 scenari, per associare l'assortimento estratto al valore vero della funzione di ricorso. Il tempo totale di generazione è stato di circa **23 ore**. Le fasi principali che hanno portato al dataset finale sono le seguenti:

Warm-Start si eseguono $N_{\text{warm}} = 10$ ottimizzazioni SAA a singolo scenario, una per ciascuno degli scenari di warm-start. Per ogni scenario si estrae la soluzione ottima $\mathbf{a}^{*(s)} \in \{0,1\}^m$ e si costruisce quindi il vettore di frequenza ottima:

$$\bar{f}_i = \frac{1}{N_{\text{warm}}} \sum_{s=1}^{N_{\text{warm}}} a_i^{*(s)}, \qquad i = 1, \ldots, m;$$

Punti estremi vengono generati $N_{\text{extr}} = 10000$ punti di estremi. Si inserisce il **caso nullo**: $\mathbf{a} = \mathbf{0}$, $\mathbf{y} = \mathbf{0}$. E successivamente i **casi a piena capacità**: gli assortimenti generati sono casuali $\gamma \sim \text{Uniform}(0{,}3, 1{,}0)$ con quantità $y_i \sim \text{Uniform}\{0, \lfloor 200\gamma \rfloor\}$. Ciascun punto candidato è proiettato nelle decisioni ammissibili tramite un problema di minimizzazione L1.

Punti casuali si costruiscono $N_{\text{train}} = 200000$ punti tramite un meccanismo di exploration-exploitation.

Nello specifico, per la generazione casuale, si segue il seguente flusso:

1. Si estrae $\alpha^{(k)} \sim \text{Uniform}(0,1)$.
2. Si definisce il vettore di probabilità di selezione per l'assortimento:

$$p_i^{(k)} = \alpha^{(k)} \cdot \bar{f}_i + (1 - \alpha^{(k)}) \cdot 0{,}5, \qquad i = 1, \ldots, m. \tag{5.1}$$

3. Si campiona il vettore obiettivo $a_i^{\text{tgt}} \sim \text{Bernoulli}(p_i^{(k)})$ indipendentemente per ogni i.

4. Si campiona l'intensità $\gamma^{(k)} \sim \text{Uniform}(0{,}1, 1{,}0)$ e le quantità finali $y_i^{\text{tgt}} \sim \text{Uniform}\{0, \lfloor 200\gamma^{(k)} \rfloor\}$, ponendo $y_i^{\text{tgt}} = 0$ se $a_i^{\text{tgt}} = 0$.

5. Con probabilità 0,2, un prodotto casuale viene portato alla capacità massima ($y_i \in [190, 200]$).

6. Il punto $(\mathbf{a}^{\text{tgt}}, \mathbf{y}^{\text{tgt}})$ è proiettato nella regione ammissibile tramite la soluzione della minimizzazione della distanza tra le varibili del modello base e i campionamenti con i vincoli di primo stadio del modello stocastico.

Successivamente è stata eseguita la fase di ottimizzazione degli iperparametri (Optuna, 30 trial), completata in **2 ore**, seguita da un affinamento tramite l'*active learning* di **13 iterazioni** (30 min), che ha aiutato il modello a migliorare la predizione nelle regioni di input prossime all'ottimo.

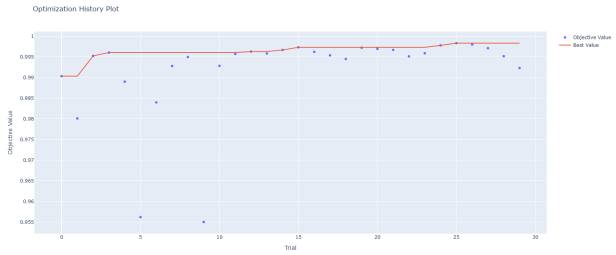
Relativamente al processo di Active Learning, si usa la rete già addestrata per trovare la soluzione ottimale attuale. Questa viene valutata con il modello SAA esatto, e se il gap tra il valore predetto e quello reale supera la soglia di tolleranza fissata a $= 100$ (euro), si aggiunge la nuova osservazione al training set e si riaddestra la rete. Inoltre, si aggiungono anche le perturbazioni attorno alla soluzione trovata. Per ogni iterazione, vengono generate 50 varianti perturbate, modificando le quantità di tre prodotti attivi di una percentuale tra il 5% e il 15%. Impostando il parametro di warm_start=True di scikit-learn MLPRegressor, si riparte dai pesi della rete precedente in modo tale da accelerare il training.

Iperparametri ottimali

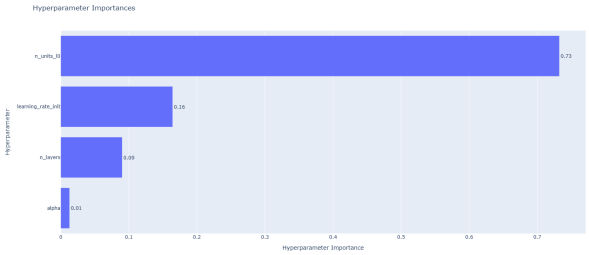
Tabella 5.3 Iperparametri migliori identificati da Optuna (30 trial, massimizzazione R^2 su validation set).

Parametro	Valore ottimale	Range esplorato
Numero di layer nascosti	2	$\{1, 2\}$
Neuroni layer 1	64	$\{16, 24, 32, 40, 48, 56, 64\}$
Neuroni layer 2	48	$\{16, 24, 32, 40, 48, 56, 64\}$
Regolarizzazione α	3.21×10^{-4}	$[10^{-5}, 10^{-2}]$
Learning rate iniziale	1.27×10^{-3}	$[10^{-4}, 10^{-2}]$
R^2 validation (migliore)	0.9983	—

Le Figure 5.5 mostrano la distribuzione dei punteggi R^2 al variare degli iperparametri e l'importanza relativa dei fattori.



(a) R^2 vs. numero di trial.



(b) Importanza degli iperparametri.

Figura 5.5 Esplorazione Optuna per il surrogato MLP. La convergenza avviene attorno ai trial 24-26.

Il valore di $R^2 = 0.9983$ sul set di validazione, numericamente indica un fitting eccellente della funzione di ricorso $Q(\mathbf{y})$. Bisogna però ragionare sul fatto che il dataset è stato costruito manualmente, includendo più tipologie di scenari, ma con una buona presenza di scenari che 'vedessero' l'ottimo. Questo dataset, quindi, porta ad interpretare l'elevato valore della metrica in modo più conservativo. Questa non rappresenta necessariamente la capacità di generalizzazione su tutto lo spazio ammissibile, ma solo su quello che è stato generato.

5.2.2 Costo computazionale: surrogato vs SAA

Trattiamo ora uno dei risultati più importanti, dal punto di vista matematico, che possiamo sostenere attraverso questo lavoro. Come commentato nela sezione 5.1.1, il problema formulato attraverso SAA è estremamente pesante dal punto di vista computazionale: volendo risolvere l'ottimizzazione sfruttando un numero superiore di scenari, o volendo provare più repliche, il tempo impiegato per ottenere una soluzione risulterebbe sempre più elevato, andando a più di 13 ore extra per ogni

replica su 100 scenari e ad un aumento esponenziale per ogni scenario aggiunto. Si consideri che l'aumento degli scenari, porta variabili e vincoli aggiuntivi al secondo stadio per ogni nuovo scenario considerato. In questo contensto si inserisce il punto di forza principale dell'approccio surrogato: l'utilizzo della rete neurale richiede solo un *investimento computazionale offline* di una certa importanza, dopodiché l'ottimizzazione su ogni nuova istanza è praticamente istantanea. La tabella 5.4 quantifica il confronto.

Tabella 5.4 Confronto del costo computazionale: SAA (10 repliche) vs. Surrogato NN. I tempi SAA si riferiscono alla singola replica con `MIPGap = 0.1`.

Fase	SAA	Surrogato NN
Generazione dataset	—	≈23 h (offline)
Ottimizzazione iperparametri	—	≈2 h (offline)
Active learning	—	≈30 min (offline)
Ottimizzazione per run	**13–18 h**	$<$ **1** min
Costo totale per 10 run	≈140 h	≈26 h

La riduzione è di 2-3 ordini di grandezza sul tempo totale. Il MILP del surrogato sostituisce le 355722 variabili del secondo stadio con un insieme di vincoli lineari associati ai pesi della rete neurale che vengono tradotti dal solutore. In questo modo, il solver chiude il gap, trovando la soluzione ottima, in pochi secondi anziché in ore.

Ovviamente, prima di festeggiare, bisogna valutare come si comporta nella realtà simulativa la soluzione ottenuta attraverso il modello surrogato.

5.2.3 Analisi dell'assortimento

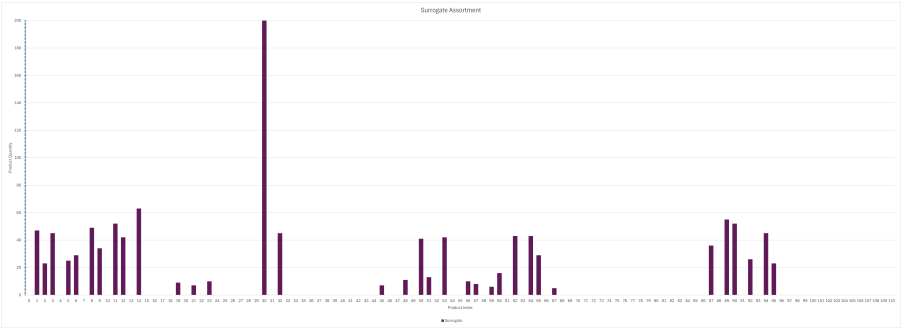


Figura 5.6 Assortimento surrogato.

Assortimento risultante. Innanzi tutto bisogna valutare come l'assortimento si distribuisce a scaffale, come fatto per il modello precedente. Dal momento che i prodotti scelti sono molto simili a quelli visti, possiamo notare che la distribuzione non cambia molto. Vengono ordinati **1191 kg** di prodotti di cui: **1109 kg** di secco, come per SAA, e **82 kg** di gnocchi. Anche in questo caso vengono saturati gli scaffali Alto e Occhi dei secchi. E la categoria a scaffale Basso non viene inserita nel frigo. Tutto esattamente allinneato al caso precedente.

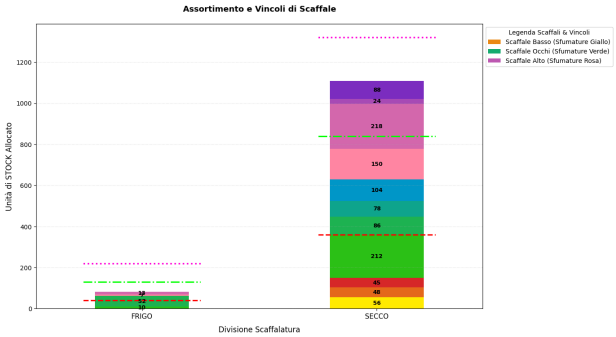


Figura 5.7 Distribuzione a scaffale dell'assortimento surrogato.

Nella tabella 5.5, possiamo vedere i risultati raggiunti con l'assortimento proposto comparati con l'assortimento ottimo definito dal modello SAA.

Tabella 5.5 Sintesi dell'assortimento surrogato NN vs. SAA (replica=6).

Metrica	SAA	Surrogato NN
N. prodotti in assortimento	41	34
Stock totale ordinato (kg)	1204	1191
Costo acquisto e fissi (1°stadio) (€)	2320.36	2208.37
Costi fissi scaffale (€)	274.00	231.00

Vediamo quindi che, anche in questo caso, abbiamo la dominanza della Pasta B a basso prezzo e l'esclusione dei prodotti premium. Le differenze principali riguardano alcune opzioni di fascia media, infatti il surrogato tende a essere meno "preciso" su prodotti con bassa domanda attesa, come Riso B 3.69 e alcune Pasta C, il che è fisiologico dato che il surrogato apprende la funzione di ricorso in modo aggregato e non distingue perfettamente contributi marginali piccoli. Questa difficoltà a gestire i prodotti che portano un impatto minimo, si riflette anche nel numero leggermente inferiore di prodotti in assortimento (34 vs. 41). Infatti, il surrogato tende a concentrare lo stock sui prodotti che sono più impattanti sul risultato finale. Ad esempio,

tutti quei prodotti che nell'assortimento finale di SAA entrano con valori in unità $\in \{1,2,3,4\}$, vengono esclusi dall'assortimento del surrogato.

Per concludere il confronto delle soluzioni di assortimento, notiamo che il valore della funzione obiettivo ottenuto con il surrogato è di $z^*_{\text{NN}} = +13.71\,€$. Questo numero non è da considerare come il vero prodfitto ottenuto dai prodotti selezionati, ma bensì è una stima ottimistica della performance reale.

La verifica della qualità della soluzione del surrogato viene eseguita risolvendo il problema di ricorso con l'*assortimento fissato* $\mathbf{y}^*_{\text{NN}}$ su 300 scenari indipendenti:

Tabella 5.6 Confronto tra stima del surrogato e profitto reale su 300 scenari.

Elemento di costo/profitto	Valore (€)	
Stima interna NN: z^*_{NN}	$+13.71$	Valore ottimizzato con surrogato
Profitto reale (300 scenari): z^*_{SAA}	-98.56	Ricorso LP, 300 scenari
Ricavo II stadio $\mathbb{E}[Q(\mathbf{y}^*)]$	$+2140.81$	Vero ricorso
Costi I stadio C_{I}	-2239.37	Acquisto + fissi
Gap surrogato-oracle	**112.27**	$z^*_{\text{NN}} - z_{\text{SAA}}$

Vediamo che il gap tra le soluzioni è di 112.27 € e si colloca di poco al di sopra della soglia di tolleranza $\Delta_{\text{tol}} = 110\,€$ fissata nell'active learning. Quindi, seppur il bias ottimista nell'area dell'ottimo, di cui soffre il surrogato, sia stato ridotto, non è completamente eliminato. Questo porta il modello surrogato a restituire valori di profitto leggermente superiori rispetto alla realtà, ma senza andare a creare problemi irrisolvibili per il retailer dal punto di vista economico: di fatto si stima una perdita aggiuntiva di 85 € con l'assortimento del surrogato, rispetto alla perdita valutata con l'assortimento ottimo del SAA.

5.3 Modello EV

5.3.1 Addestramento del predittore Ridge

Vengono addestrati **112 regressori Ridge indipendenti**, uno per ciascuna prodotto inseribile in assortimento e uno per imparare il valore di no-buy, escluso quello a priori ottenuto con l'assortimento completo. Quest'ultimo ha lo scopo di penalizzare

nella funzione obiettivo un assortimento che porta a tanta domanda persa prima dell'ingresso. La scelta del parametri di regolarizzazione α viene fatto tramite cross-validation, su cv=5, usando l'opzione interna nella funzione `CVRidge`, nella Figura5.8 possiamo vedere come il valore di α cambia l'errore quadratico medio per due prodotti selezionati.

(a) Prodotto 1

(b) Prodotto 36

Figura 5.8 Curva di cross-validation per la selezione di α - modello Ridge.

Il dataset di addestramento comprende 30000 record che vengono definiti a partire da un vettore di prodotti in assortimento: la formazione di tale vettore è fatta tenendo conto del vincolo di mutua esclusione dei prodotti nello scaffale Alto e Basso. Il target è definito dal valore della domanda associata a ciascun prodotto in assortimento, considerando la dinamica di sostituzione a catalogo.

Guardando le prestazioni del modello predittivo, purtroppo, ci si rende conto del fatto che la sua abilità di learning, non è adatta all'obiettivo che gli viene richiesto. La sola approssimazione lineare portata dal modello Ridge, è molto limitante per rappresentare la complessa dinamica del modello Latent Class.

Tabella 5.7 Metriche di addestramento del predittore Ridge (media sui 36 modelli).

Metrica	Valore	Interpretazione
MAE medio (test set)	5.96 unità	Errore assoluto medio per prodotto
R^2 medio (test set)	0.5088	Varianza spiegata media

Un $R^2 = 0.51$ è indicativo di un modello lineare che cattura circa la metà della varianza della domanda. L'errore assoluto medio per prodotto di 5.96 unità su una domanda che sta in un range di 20-100 unità per prodotto corrisponde a un errore relativo del 6-30% che, nei casi peggiori, potrebbe quindi essere molto alto. La scelta di valutare ugualmente l'ottimizzazione con questi predittori è dovuta alla difficoltà e all'aumento delle dimensioni del problema, supponendo di voler addestrare 112

MLP con lo stesso obiettivo: provando ad addestrare un singolo MLP multi-output, su un dataset più ampio, il valore di $R^2 = 0.35$ ottenuto risultava ancora peggiore rispetto al Ridge scelto.

5.3.2 Analsi dell'assortimento

L'assortimento ottenuto è strutturalmente diverso rispetto ai due modelli precedenti. Innanzi tutto, dato che il modello incorpora già nella domanda primaria la dinamica di sostituzione, abbiamo una frequenza di arrivi effettivi inferiore; inoltre, l'eliminazione della stocasticità, fa in modo che il solutore stabilisca un assortimento che lavori bene rispetto al valore deterministico di domanda ricevuto, andando a perdere tutta l'abilità di coprirsi da situazioni di rischio.

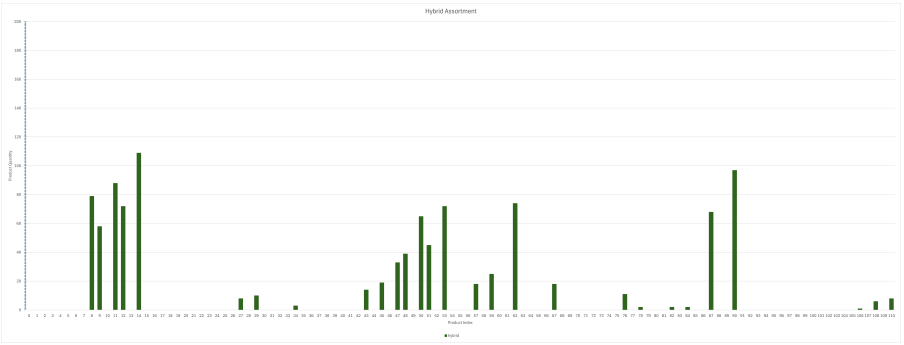


Figura 5.9 Assortimento Hybrid EV.

Assortimento risultante. In questo caso, la tendenza è cercare di inserire a scaffale tutte le possibilità, nel senso che si vanno a coprire tutte le fasce di prezzo, includendo prodotti della fascia medio-alta, molto meno presenti negli altri assortimento. Inoltre, anche tutti i vari livelli di scaffalatura, compresa la stagera bassa del frigo, ha dei prodotti all'interno. Notiamo che vengono ordinati **1046 kg** di prodotti di cui: **890 kg** di secco, e **156 kg** di gnocchi. Vediamo quindi che c'è un aumento significativo nel quantitivo di prodotti freschi, nonostante una diminuzione del totale acquistato. In questo caso non vengono saturati gli scaffali Alto e Occhi dei secchi.

Nella tabella 5.3.2 vediamo i risultati valutati rispetto al modello EV:

Osservando il valore $+705.65\,€$ della funzione obiettivo, notiamo subito che *non è comparabile* con i $-14\,€$ del SAA. Teniamo sempre in considerazione il fatto

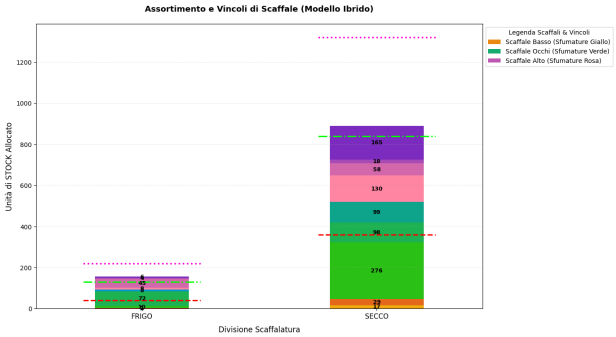


Figura 5.10 Distribuzione a scaffale dell'assortimento ibrido.

Tabella 5.8 Risultati dell'ottimizzazione EV Ridge.

Metrica	Valore
Valore funzione obiettivo	+705.65 €
Tempo di ottimizzazione	9181.64 s (≈2.5 h)
MIP gap a terminazione	35.82%
Stock totale allocato	1046.00 unità
Domanda totale predetta	1049.00 unità
Domanda media reale	1558.18 unità
Lost sales in-store	7.97 unità
Numero prodotti in assortimento	28

che la funzione obiettivo EV usa la domanda predetta come valore fisso, annullando praticamente il termine di penalità per lost service, poichè la domanda predetta viene soddisfatta per costruzione; inoltre, non avendo perdite, anche il livello di servizio sarà in linea con la soglia. In questo modo andiamo a calcolare un margine su ricavi certi, che non tengono conto della penalità che porta il modello stocastico. Un effetto che si mostra dalla soluzione, che è interessante da mostrare, è il valore delle sostituzioni all'interno del negozio: avere $\sigma_{ik} \approx 0$ è una conseguenza diretta della modellazione deterministica: se la domanda $\hat{d}_i(\mathbf{a})$ è fissa e il MILP assegna uno stock $y_i = \hat{d}_i(\mathbf{a})$ a ogni prodotto selezionato, non vi è mai eccesso di domanda da redistribuire tra prodotti. Purtroppo, si perde completamente l'interesse nel meccanismo di sostituzione.

Il gap di ottimizzazione. L'ottimizzatore è stato fermato al 35.82% di gap per due ragioni:

1. il solutore stava girando da troppo tempo: risolvere un problema deterministico in più di 2 h è decisamente esagerato. Il problema deriva dalla matrice densa dei coefficienti Ridge che indebolisce il rilassamento continuo e rallenta la chiusura del gap nell'albero B&B. Infatti, il valore dell'Incumbent è fermo alla soluzione finale già da dopo 140 secondi dall'inizio dell'ottimizzazione;

2. dal momento che il predittore Ridge è già intrinsecamente approssimato ($R^2 = 0.51$), è inutile inseguire un gap inferiore al 35%, perchè vorrebbe dire ottimizzare con estrema precisione un modello già distorto: il guadagno in qualità della soluzione finale sarebbe trascurabile rispetto al costo computazionale.

5.4 Performance su scenari *out-of-sample*

5.4.1 Scenari di mercato

Le soluzioni dei tre modelli sono testate su **300 scenari OOS** indipendenti, generati dallo stesso processo stocastico della fase di ottimizzazione, ma che non sono stati mai visti dal solutore.

Nella tabella 5.9 accompagnata dalla Figura 5.11 mostriamo i risultati principali del comportamente dell'assortimento scelto, simulando scenari in linea con il mercato, per vedere la robustezza della soluzione su avvenimenti tipici ma non visti direttamente dal solutore. Nei valori, viene inserito anche il livello di no-buy, che considera quanti clienti decidono di non entrare nel negozio a priori, dal momento che si tratta di un valore estremamente rappresentativo per dimostrare i limiti del modello deterministico.

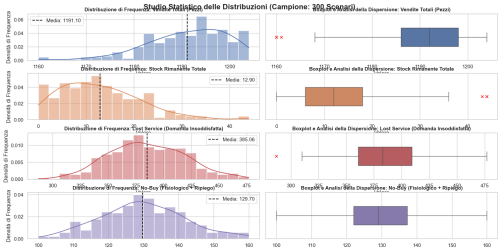
Tabella 5.9 Performance out-of-sample su 300 scenari. Tutti i valori sono aggregati su prodotti e medi sugli scenari.

Modello	Vendite medie (unità)	Lost service in-store medio (unità)	No-buy media (unità)	Worst-case lost (P_{95}, unità)	Worst-case sales (P_5, unità)
SAA (repl=1)	1191.10	385.06	129.70	445.00	1176.00
SAA (repl=6)	1191.13	385.16	129.70	445.25	1173.95
Surrogate NN	1182.24	395.82	131.20	460.00	1168.95
Ridge EV det	1041.49	277.87	302.25	277.87	1040.00

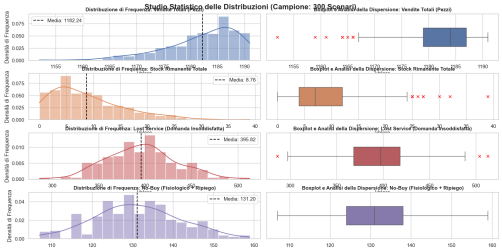
Dal momento che le due repliche SAA sono sostanzialmente identiche a livello OOS (differenza di 0.03 unità in vendite medie, 0.10 in lost), il che conferma la stabilità della soluzione, e sottolinea la buona riuscita della scelta del 100 scenari, nei prossimi commenti e nei grafici che presenteremo, considereremo solo i dati proveniente dall'assortimento ottenuto nella replica 1.

Dall'analisi dei dati possiamo affermare che, nell'assortimeto SAA e surrogato, le vendite si attestano su un buon livello, paragonabile al valore calcolato nell'ottimizzazione, scostandosi rispetto alla media negli scenari simulati di poche unità, per entrambi la deviazione standard si aggira su $\sigma = 7/8$ unità. Questo fattore è molto interessante: la simulazione è condotta considerando i due livelli di sostituzione, quindi quella a catalogo e quella in-store. Il fatto che l'ottimizzatore abbia stabilito l'assortimento vedendo la domanda a catalogo pieno, non ha influenzato il risultato valutato su scenari con informazione più completa poichè i valori di performance sono quasi uguali a quelli forniti dal modello.

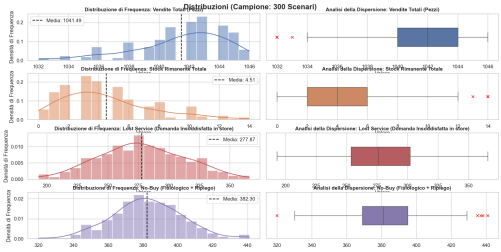
Per quanto riguarda l'assortimento fornito dal modello ibrido deterministico, ritroviamo lo stesso valore stabilito in fase di ottimizzazione di no-buy a priori, una quantità di stock venduto paragonabile allo stock allocato che porta ad avere poche rimanenze



(a) SAA: distribuzione vendite, rimanenze, lost-sales e no-buy.



(b) Surrogato: distribuzione vendite, rimanenze, lost-sales e no-buy.



(c) Ibrido deterministico: distribuzione vendite, rimanenze, lost-sales e no-buy.

Figura 5.11 I grafici rappresentano la distribuzione empirica dei valori di vednite effettuate, vendite perse, rimanenze a magazzino e no-buy ottenute dagli assortimenti dei tre modelli di ottimizzazione valutati su nuovi scenari di mercato.

a scaffale. É tuttavia interessante notare la distribuzione di queste ultime: sebbene tutti e tre gli assortimenti generino una coda estesa nel quantile svantaggioso, il caso ibrido ne risulta leggermente più penalizzato dal momento che presenta una coda anche più spessa rispetto agli altri casi. Questo comportamento è spiegato dal fatto che il modello deterministico non è in grado di modellare la variabilità degli scenari, quindi soffre maggiormente su scenari più estremi.

Vediamo subito l'importanza del tracciare la quantità di clienti che scelgono a priori la no-buy. Il livello di servizio medio calcolato come vendite su tentativi di acquisto è:

$$\mathrm{SL}_{\mathrm{SAA}} = \frac{1191}{1191+385} \approx 75.6\% \quad \mathrm{SL}_{\mathrm{surr}} = \frac{1182}{1182+396} \approx 74.9\%$$
$$\mathrm{SL}_{\mathrm{det}} = \frac{1041}{1041+278} \approx 78.9\%$$

mentre il livello di servizio medio totale (rispetto ai clienti potenziali medi):

$$\mathrm{SL}^{\mathrm{tot}}_{\mathrm{SAA}} \approx 70.0\% \quad \mathrm{SL}^{\mathrm{tot}}_{\mathrm{surr}} \approx 70.0\% \quad \mathrm{SL}^{\mathrm{tot}}_{\mathrm{det}} \approx 64\%.$$

Senza guardare la no-buy, l'impressione era che il modello deterministico riuscisse a gestire meglio di scenari di domanda, portando un assortimento in grado di coprire una buona parte delle richieste. In realtà, l'assortimento è forte solo per chi decide di entrare in negozio, ma buona parte del cliente viene tagliato fuori. La forza del modello SAA o del surrogato, è proprio quella di riuscire a definire un assortimento che valuti gli acquisti che entrano in store, ma che limiti anche chi decide di non entrare a priori: allocando più prodotti, riescono ad ampliare il bacino di utenza.

Prodotti critici. L'analisi per (scenario, prodotto) identifica i colli di bottiglia principali. Nell'assortimento di SAA, i prodotti con maggiore domanda persa media sui 300 scenari sono:

- **[032] Pasta B 1.30 €/kg Occhi**: 94.68 u. perse in media. Si nota chiaramente come la domanda superi sistematicamente lo stock allocato in quella posizione.
- **[002] Pasta A 1.98 €/kg Occhi**: 44.27 u.
- **[095] Pasta D 2.38 €/kg Occhi**: 36.18 u.
- **[005] Pasta A 2.58 €/kg Occhi**: 26.06 u.

Per quanto riguarda l'assortimento fornito dal surrogato:

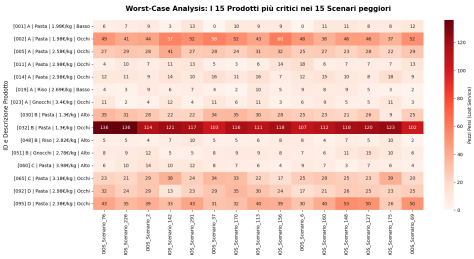
- **[032] Pasta B 1.30 €/kg Occhi**: 97.15 u (+2.47 rispetto a SAA).
- **[002] Pasta A 1.98 €/kg Occhi**: 45.54 u.
- **[095] Pasta D 2.38 €/kg Occhi**: 36.06 u.
- **[005] Pasta C 3.18 €/kg Occhi**: 30.21 u.

La classifica è praticamente identica a quella SAA, abbiamo solo due prodotti che si differenzaino nella top 15 dei peggiori: nella classifica del surrogato entra il prodotto Pasta A 2.58 €/kg Alto, che presenta 14 unità perse nello scenario peggiore, mentre esce il prodotto Riso B 2.82 €/kg Alto. Quindi notiamo che la rete neurale replica la stessa struttura di rischio e ciò conferma che l'assortimento surrogato presenta le stesse vulnerabilità.

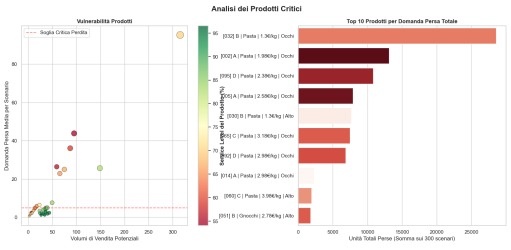
Infine, il risultato dell'assortimento ibrido porta ad una classifica abbastanza diversa, che nasce dal fatto che alcuni dei top peggiori prodotti degli altri assortimenti, in questo caso, non vengono neanche inseriti. Eliminando ogni tipo di possibile valutazione sulle vendite. La cosa interessante da notare per lo stock deterministico è il fatto che si esaurisce in modo quasi identico in ogni scenario, indipendentemente dalla domanda realizzata. Inoltre, i prodotti peggiori, pur mostrando delle differenze in termini di domanda persa, tendono ad avere un comportamento molto più uniforme rispetto che gli altri assortimenti.

Dall'analisi dei prodotti critici in Figura 5.13 notiamo un pattern ricorrente: la posizione Occhi è la più attrattiva per i clienti price-sensitive, e lo stock allocato in quella posizione viene esaurito più frequentemente rispetto ad Alto e Basso. Infatti, data la caratterizzazione del mercato con una maggiore prevalenza di clienti price-sensitive, questo è un comportamento che ci si poteva aspettare.

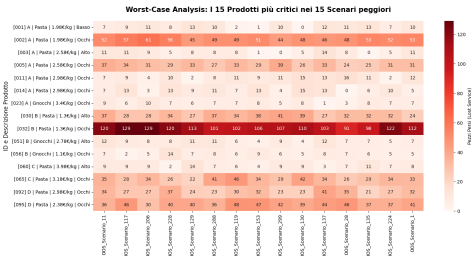
Un'osservazione aggiuntiva sul prodotto Pasta B 1.30 €/kg Occhi è che, nonostante la domanda sia molto alta, trattandosi di un prodotto estremamente attrattivo data la posizione a scaffale ottimale ed il prezzo competitivo, il fatto che venga limitata la sua presenza a scaffale è motivata dal vincolo di capacità. Lo scaffale 'Occhi', è quello più apprezzato in generale, per cui riceve domanda da prodotti che portano un margine di profitto maggiore, tanto da preferire una perdita di domanda così consistente per il prodotto considerato.



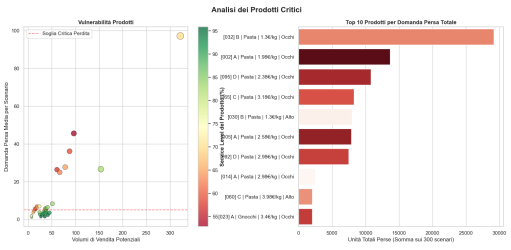
(a) SAA: prodotti in peggiori scenari.



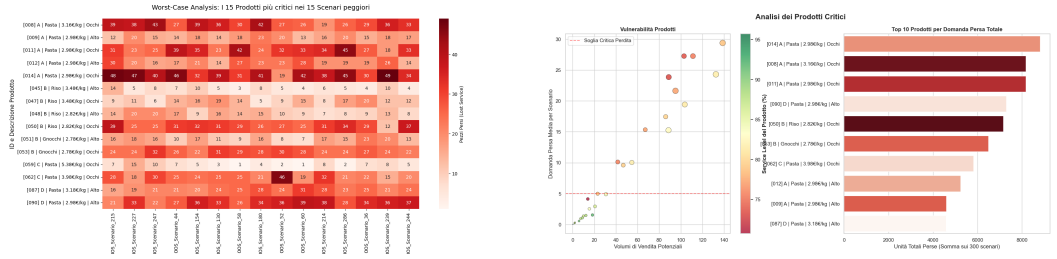
(b) SAA: analisi top 10 prodotti critici.



(c) Surrogato: prodotti in peggiori scenari.



(d) Surrogato: analisi top 10 prodotti critici.



(a) Ibrido: prodotti in peggiori scenari.

(b) Ibrido: analisi top 10 prodotti critici.

Figura 5.13 Quali sono i prodotti più critici per ogni assortimento negli scenari di mercato out-of-sample.

5.4.2 Scenari di stress

Per valutare la robustezza degli assortimenti scelti, testiamo il loro comportamento quando le probabilità dei segmenti di mercato si discostano dai valori di partenza.

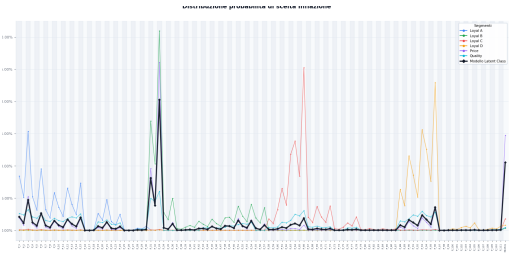
Sono stati considerati tre scenari di test:

Inflazione Segmento *price-sensitive* dominante ($p_5 = 0.70$): la clientela si concentra su prodotti economici.

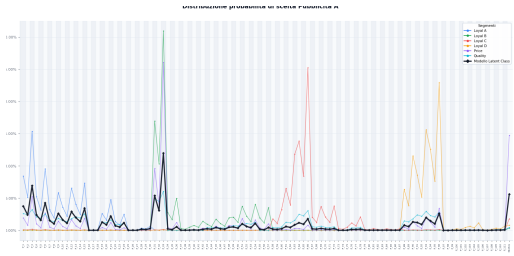
Pubblicità Marca A Segmento A potenziato ($p_1 = 0.26$): aumento della domanda per tutti i prodotti A.

Clientela premium Segmento sensibile al prezzo ridotto ($p_5 = 0.20$): preferenze verso prodotti costosi.

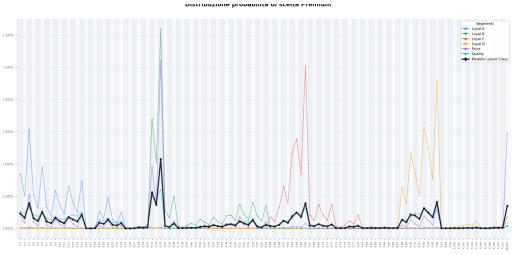
Nella Figura 5.4.2, sono mostrati come si distribuiscono le preferenze dei clienti sotto i nuovi valori di probabilità: vediamo come cambia la proporzione di scelta per ogni opzione e quindi come si sposteranno i numeri di domanda nelle nuove configurazioni. In particolare, nel primo scenario, come atteso, c'è un picco di richieste sul prodotto Pasta B 1.30 €/kg, che già rietrava tra i prodotti critici. Ci si aspetta quindi che l'assortimento patisca ulteriormente su questo prodotto, causando un buon numero di vendite perse. Nel secondo e nel terzo scenario, la preferenza per tale prodotto, seppur ancora alta, si riduce nettamente. Come anche la preferenza intrinseca per la no-buy. Per lo scenario di 'pubblicità' si ha, chiaramente, un incremento nelle preferenze dei prodotti Marca A. Mentre nell'ultimo scenario risultato attrattivi anche i prodotti più cari.



(a) Stress test: mercato sotto inflazione. Percentuale di clienti price-sensitive aumentata.



(b) Stress test: mercato successivamente a pubblicità per Marca A.



(c) Stress test: mercato con clientela premium.

Tabella 5.10 Performance sotto scenari di stress - SAA, modello EV deterministico e il surrogato.

Scenario	Metrica (media)	SAA	Surrogate NN	Ridge EV
Inflazione	Vendite (u.)	1113.06	1107.51	988.17
	Lost service (u.)	375.48	377.68	84.62
	No-buy (u.)	214.53	217.88	630.96
Adv Marca A	Vendite medie (u.)	1133.20	1128.42	1032.86
	Lost service (u.)	461.61	467.92	330.16
	No-buy (u.)	115.72	117.52	339.63
Premium	Vendite medie (u.)	1196.77	1184.93	1044.39
	Lost service (u.)	438.95	455.57	633.0
	No-buy (u.)	72.83	73.67	216.62

Analisi degli shock.

- **Inflazione.** SAA e surrogato vedono *ridursi* le vendite di circa 78 unità rispetto allo scenario standard (1191 $\rightarrow$ 1110), ma il lost service diminuisce (385 $\rightarrow$ 376) perché il mercato si concentra su prodotti economici che l'assortimento SAA già privilegia. Il modello deterministico migliora drasticamente il proprio lost (277.87 $\rightarrow$ 84.82): ciò dipende dal fatto che l'assortimento Ridge è orientato verso prodotti medio-alti che, in uno scenario price-sensitive, semplicemente *non sono cercati*, quindi la domanda rimanente si adatta meglio allo stock limitato disponibile. Questo è un falso segnale di superiorità del modello EV; infatti basta osservare il numero di no-buy che nascono in questo caso ben 630 per il modello deterministico contro meno di 220 per gli altri modelli.

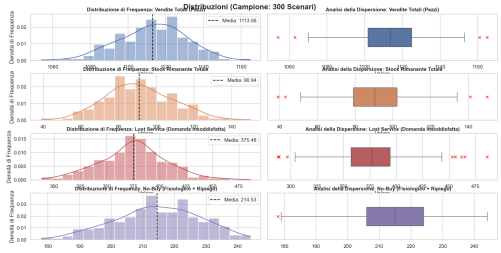
- **Pubblicità Marca A.** Sia il surrogato che il modello SAA registrano un aumento del lost service ($\approx +75$–$+80$ u. rispetto allo standard), perché la domanda aggiuntiva per i prodotti Marca A supera lo stock ottimizzato per la distribuzione di partenza. Anche il modello EV presenta un aumento consistente della lost service (277.87 $\rightarrow$ 330.16) poichè pur avendo in assortimento prodotti della Marca A, non riesce a gestire l'aumento dei clienti in quel segmento. Per tutti i modelli la no-buy è quasi inalterata.

- **Clientela premium.** La domanda si sposta verso le Marche, che gli assortimenti hanno, e i prodotti Riso e Gnocchi, che l'assortimento SAA e surrogato contengono in quantità minime. Il lost service cresce per SAA (da 396 a $\approx$440) e per il surrogato (da 385 a $\approx$455) ma rimane gestibile; per il modello deterministico le mancate vendite esplodono a 633 unità per effetto congiunto dello stock ridotto e dell'inadeguatezza del'assortimento in questo scenario.

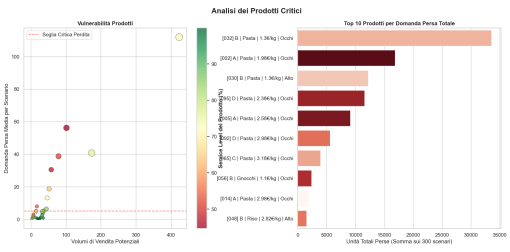
  Una nota interessante nasce dall'analisi del valore della no-buy: per tutti i modelli si riduce nettamente. Questo indica che l'assortimento disponibile risulta comunque attrattivo per il nuovo gruppo di clienti, che vedono la possibilità di trovare qualcosa in linea con le loro preferenze. Il dato nasce dal fatto che, diminuendo il segmento price-sensitive, sono aumentati i segmenti brand loyal e il quality-sensitive. In questo modo, essendoci in assortimento dei prodotti che rispecchiano le necessità di tutti i clienti, la no-buy option è drasticamente inferiore. Il problema strutturale, che genera l'elevato lost

service, risiede interamente nelle quantità fisiche: lo stock allocato non è sufficiente per coprire l'aumento dei volumi richiesti da questi clienti altamente propensi all'acquisto.

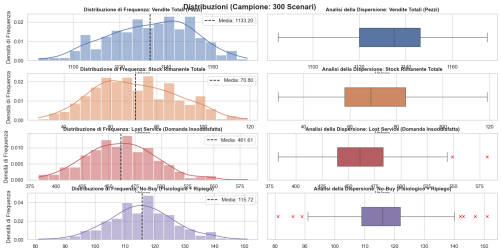
Modello SAA Nella Figura 5.15 mostriamo come si comporta l'assortimento proposto dal modello SAA nei tra scenari di stress. Vengono rappresentate le distribuzioni empiriche delle vendite totali, delle lost totali, delle rimanenze a scaffale e della no-buy. Inoltre, viene anche mostrata la classifica dei primi 10 prodotti più critici in termini di vendite perse.



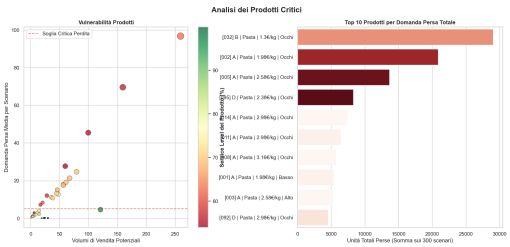
(a) Scenario inflazione - SAA.



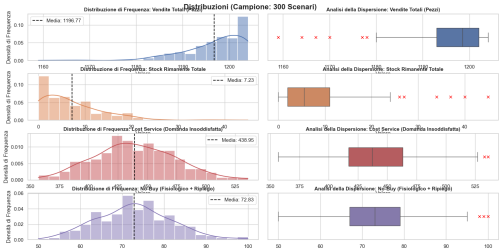
(b) 10 peggiori prodotti..



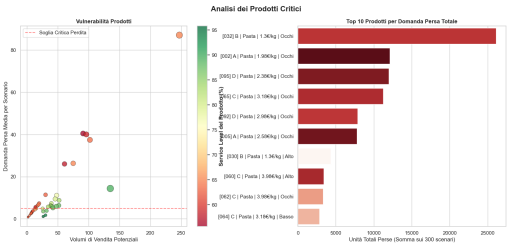
(c) Scenario pubblicità Marca A - SAA.



(d) 10 peggiori prodotti.



(e) Scenario clientela premium - SAA.



(f) 10 peggiori prodotti.

Figura 5.15 Comportamneto dell'assortimento SAA in scenari di stress.

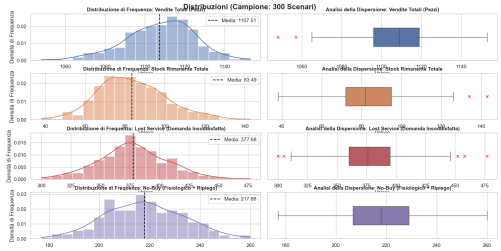
L'analisi dei grafici porta a conclusioni specifiche per ogni tipo di stress test simulato: i tre scenari si posizionano in tre fasce distinte di reazione dell'assortimento. La reazione allo scenario di inflazione è opposto a quello dei clienti premium, come è coerente che sia, mentre lo scenario che fortifica il segmento loyal Marca A si

posiziona a metà tra i due. Nel primo caso l'offerta è inadeguata alle preferenze, ma si mantiene ugualmente una discreta quantità di vendite. Nel secondo caso, l'assortimento è perfettamente in linea con le necessità del cliente, ma la quantità allocata non riesce a soddisfare tutta la domanda che arriva.

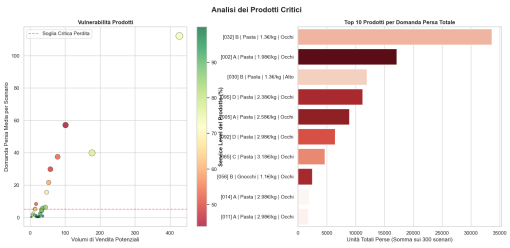
Per quanto riguarda i prodotti critici, come anche nello scenario di base, si verifica una tendenza a sottostimare la necessità di prodotto Pasta Marca B a 1.3€/kg nello scaffale Occhi. Questo prodotto, nonostante la variazione degli scenari, continua a rappresentare il prodotto più critico del sistema.

Confronto surrogato. Sotto i tre scenari di stress (Figura 5.16), il surrogato si comporta in modo pressoché identico al SAA in tutti gli scenari: la differenza massima è di ± 15 unità in lost service. Questo è un risultato incoraggiante poichè conferma che l'assortimento appreso dal surrogato è robusto e, come succede per il modello SAA, risponde in modo analogo agli scenari di stress.

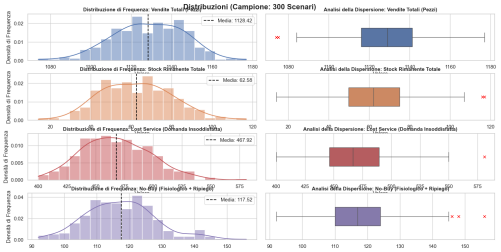
Confronto EV. Sotto i tre scenari di stress (Figura 5.17), possiamo notare che l'assortimento del modello ibrido va in difficoltà su ogni scenario. Mostra una scarsa capacità di adattamento agli shock, generando sempre grandi problematiche dal punto di vista economico. Nel primo stess test fa scappare tutti i clienti, le vendite crollano ed aumentano nettamente le rimanenze a scaffale. Nel secondo scenario, lavora al limite delle sue capacità, mostrando un elevato tasso di no-buy.Infine, nel terzo scenario, finisce la merce su tutti i prodotti, le rimanenze vanno tutte a zero ed aumenta nettamente la domanda persa. Questo sottolinea come gli approcci stocastici siano nettamente superiori, anche a livello di test di robustezza, rispetto al caso deterministico.



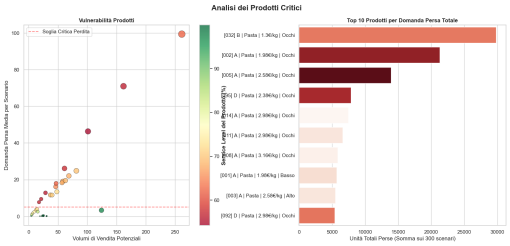
(a) Scenario inflazione - Surrogato.



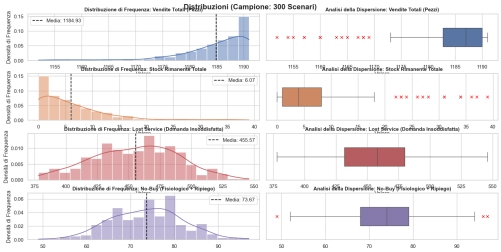
(b) 10 peggiori prodotti..



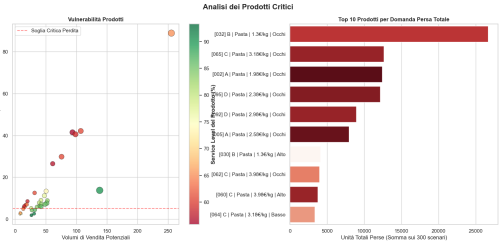
(c) Scenario pubblicità Marca A - Surrogato.



(d) 10 peggiori prodotti.

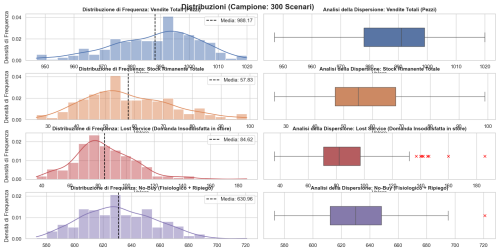


(e) Scenario clientela premium - Surrogato.

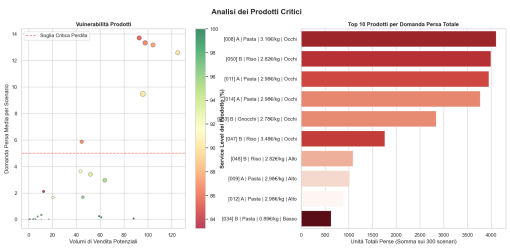


(f) 10 peggiori prodotti.

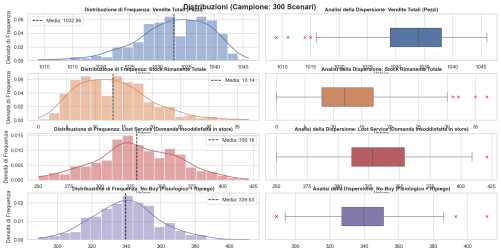
Figura 5.16 Comportamneto dell'assortimento Surrogato in scenari di stress.



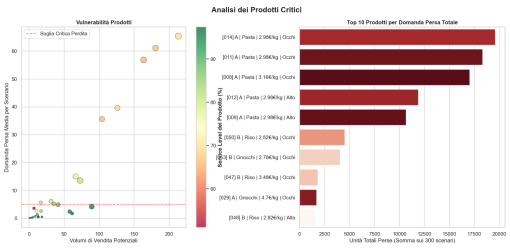
(a) Scenario inflazione - Ibrido.



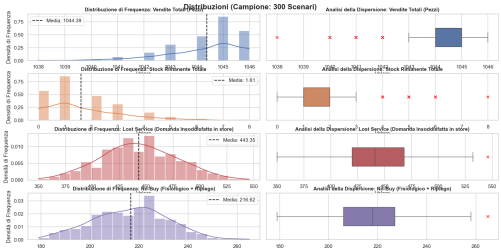
(b) 10 peggiori prodotti..



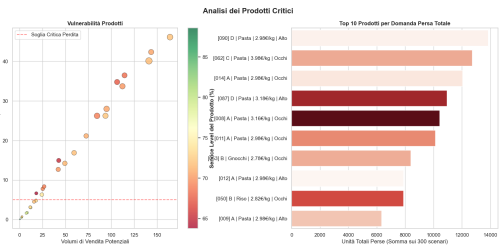
(c) Scenario pubblicità Marca A - Ibrido.



(d) 10 peggiori prodotti.



(e) Scenario clientela premium - Ibrido.



(f) 10 peggiori prodotti.

Figura 5.17 Comportamneto dell'assortimento Ibrido in scenari di stress.

Capitolo 6

Conclusioni e sviluppi futuri

In questo lavoro si è affrontato il problema di assortimento e delle quantità da ordinare per un rivenditore di medio-piccole dimensioni, in presenza di domanda stocastica e comportamento di acquisto dei clienti definito da un modello Latent Class (LCCM). É importante sottolineare che lo sviluppo del lavoro si è concentrato sia sulla componente matematica, legata al confronto tra gli approcci implementati, sia sulla componente economica, su cui è stata basata l'interpretazione dei risultati.

Sono stati implementati tre approcci: un modello di ottimizzazione stocastica a due stadi che sfrutta un numero finito di scenari; un modello surrogato con rete neurale per apprendere il valore del ricorso e un modello deterministico che itegra dei predittori Ridge come conoscitori delle quantità di domanda. Come visto, le soluzioni proposte hanno evidenziato differenze sia rispetto alla qualità e sia di costo computazionale. Il metodo di ottimizzazione stocastica a due stadi che sfrutta SAA con 100 scenari e 10 repliche produce soluzioni stabili: tutte le repliche portano a soluzioni di assortimento tendenzialmente identiche con una variabilità out-of-sample, valutata su 300 scenari, di sole 8.45 unità sulle 1191 vendute. La nota negativa risiede nel costo computazionale; infatti, per risolvere il problema di ottimizzazione, si sono impiegate $\approx 13h$ per ogni replica. L'approccio surrogato riesce a sopperire a questo problema riducendo il tempo di ottimizzazione per singola run a pochi secondi, senza perdere particolarmente precisione nei test out-of-sample. La degradazione è inferiore all'1% rispetto al SAA, portando un numero di 1182 unità venduta con deviazione standard pari a circa 7 unità. Il no-buy medio, aumenta leggermente rispetto alla soluzione SAA, ma si mantiene in un

intervallo più controllato, pur considerando che già nel caso stocastico di base contava una variabilità di poco più di 11 unità. Il costo di generazione offline (dataset, Optuna e active learning) si ammortizza già dalla seconda replica. Bisogna comunque sottolineare la difficoltà di questo approccio: la generazione del dataset e l'addestramento della rete risultano essere un notevole punto di debolezza. Riuscire ad esplorare lo spazio delle soluzioni, che è caratterizzato da altissima dimensionalità, è estremamente complesso. Da notare, comunque, che il surrogato è in grado di replicare anche la struttura di rischio relativa al sigolo prodotto, ed evidenzia lo stesso collo di bottiglia riscontrato dal modello SAA: Pasta B 1.30 €/kg nello scaffale ad altezza Occhi, con un sostanziale numero di unità perse. Questo risultato è molto interessante: una rete neurale ben addestrata, può far crollare i tempi di soluzione dell'ottimizzazione portando dei risultati ottimali di poco lontani dalla soluzione vera. Per concludere, si conferma che il predittore lineare Ridge ($R^2 = 0.51$) non riesce a catturare la non-linearità del modello di domanda. Il risultato è un assortimento strutturato diversamente, con prodotti che cercano di coprire l'intera differenziazione del catalogo. Negli scenari out-of-sample, produce in media un totale di circa 660 unità perse, considerando sia no-buy che lost sales (che corrispondono ad un livello di servizio del 64%). Contro le 515 dell'assortimento SAA o le 525 del surrogato (livello di servizio $\approx 70\%$), che si dimostrano essere, da un punto di vista economico, più cauti nei confronti del cliente: vedono il rischio nel non soddisfare le richieste. Inoltre, risponde male agli scenari di stress test, producendo in alcuni casi gravi mancanze di disponibilità a scaffale e in altri una scelta di assortimento completamente sbagliato, attestandosi su livelli di servizio di poco superiori al $\approx 55\%$.

Analizzando e confrontando le soluzioni si mette tuttavia in luce una caratteristica strutturale comune a tutti i metodi, che nasce direttamente dal modello di scelta costruito: la preferenza marcata per i prodotti presenti negli scaffali Alto e Occhi. Negli assortimenti proposti, questa porta a saturare il vincolo di capacità per cercare di ottenere più vendite possibili, lasciando invece meno assortimento negli altri scaffali e in particolare nel frigo, a causa di una domanda per tali prodotti ridotta. Questo si traduce in abbondanti perdite su prodotti particolarmente ricercati, L'esempio più lampante, riscontrato sia in SAA che nel surrogato, è proprio quello del prodotto Pasta B 1.30 €/kg in posizione Occhi, con ben 94-97 unità perse in media. Invece, nel modello ibrido, nonostante questo prodotto non venga neanche inserito nell'assortimento, i prodotti con maggiore perdita sono sempre quelli inseriti ad

assortimento che si trovano nello scaffale Occhi, confermando la criticità di questo vincolo fisico.

Il risultato centrale di questo lavoro è che *un surrogato neurale* ben addestrato permette di approssimare la soluzione SAA a una frazione del costo computazionale, con una differenza di valutazione OOS inferiore all'1%. É un risultato molto forte, che va a sottolineare come due strumenti matematici potenti, l'ottimizzazione e le reti neurali, possano supportarsi per ottenere soluzioni utili anche a livello di strategia economica, in tempi gestibili. Il risultato del modello EV deterministico conferma, al contrario, il valore dell'approccio stocastico: ignorare la variabilità della domanda rende l'assortimento fragile sia rispetto alla dinamica di mercato di base, sia ai cambiamenti della distribuzione, come mostrato dagli scenari out-of-sample e di stress.

Relativamente a sviluppi futuri del lavoro proposto, il modello di domanda adottato è costruito sulla base di esempi di mercato. La possibilità di *stima econometrica su dati reali*, porterebbe un miglioramento fondamentale nell'analisi e nell'interpretazione dei risultati. Invece, focalizzandoci sul costo computazionale del problema MILP risultante dall'approccio SAA (e in misura minore dal surrogato). Per migliorare questo aspetto, un approccio potrebbe essere quello di *ridurre a priori il numero di candidati* da considerare nell'ottimizzazione, sfruttando informazioni di mercato. Ad esempio, un'analisi di elasticità per segmento di consumatore permetterebbe di escludere tutti quei prodotti il cui margine atteso è dominato da altri. In questo caso, questo filtro avrebbe probabilmente eliminato a priori tutte le referenze premium (Riso A 5.38 €/kg, Gnocchi C/D) riducendo $|\mathscr{I}|$ e il numero di variabili binarie del MILP sarebbe diminuito. Inoltre, il LCCM fornisce le probabilità di acquisto per ogni prodotto in ogni configurazione di assortimento. Si potrebbe implementare una strategia che vada ad eliminare tutti quei prodotti con probabilità attesa sempre bassa rispetto al costo fisso di scaffale. Questa strategie farebbe diminuire la dimensione del problema, senza perdita di ottimalità. Altre euristiche basate su analisi economiche, che portano ad un ridimensionamento del problema, potrebbero essere una buona base di partenza per analisi future. Infine, un'estensione naturale e potenzialmente interessante è rendere *stocastico* il paradigma della versione ibrida, mantenendo l'architettura ibrida ma integrando la variabilità della domanda nel processo di ottimizzazione.

Bibliografia

[1] P. Zarembka (ed.). Conditional logit analysis of qualitative choice behavior. 1974.

[2] Kenneth E. Train. *Properties of Discrete Choice Models*. Cambridge University Press, 2009.

[3] M. Ben-Akiva and S. R. Lerman. Discrete choice analysis: Theory and application to travel demand. 1985.

[4] Keller K. L. Kotler, P. *Marketing Management*. Pearson Education, 2016.

[5] Gordon R Foxall and Gordon E Greenley. Consumers' emotional responses to service environments. *Journal of Business Research*, 46(2):149–158, 1999.

[6] A Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on stochastic programming. Modeling and theory*. 01 2009.

[7] D.Pollard. Strong consistency of k-means clustering, 1981.

[8] F.Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain, 1958.

[9] George Cybenko. Approximation by superpositions of a sigmoidal function, 1989.

[10] Gurobi optimizer reference manual.

[11] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[12] Xiaochen Chou, Enza Messina, and Stein W. Wallace. Solving two-stage stochastic programming problems via machine learning. In Giuseppe Nicosia, Varun Ojha, Sven Giesselbach, M. Panos Pardalos, and Renato Umeton, editors, *Machine Learning, Optimization, and Data Science*, pages 1–12, Cham, 2025. Springer Nature Switzerland.

[13] Ryan O'Donnell. Analysis of boolean functions, 2021.

[14] AltroConsumo. Test pasta, riso e gnocchi.

[15] CAAT - Centro Agroalimentare Torino. Listino prezzi all'ingrosso - mercuriale del 31-01-2025, 2025.

[16] Regione Campania - Direzione Generale Politiche Agricole. *Disciplinare di produzione della Indicazione Geografica Protetta «Pasta di Gragnano»*, 2020.

[17] Ente Nazionale Risi. Situazione di mercato al 29 gennaio 2026, 2026.

[18] Riso Italiano. Prezzi troppo bassi, 2026.

[19] Giuseppe Zeppa. *Tecnologia della pasta*, 2017.

[20] Nazrul Shaikh and Khaled N. Alqahtani. A two-stage stochastic programming model for assortment optimisation. *International Journal of Applied Management Science*, 11:185, 01 2019.

[21] Paolo Brandimarte. Lezioni del corso di business analytics, 2024.

[22] Jitka Dupačová, N. Groewe-Kuska, and Werner Roemisch. Scenario reduction in stochastic programming: An approach using probability metrics. *Mathematical Programming*, 95:493–, 01 2003.

[23] ISMEA Mercati. Prezzi medi nazionali dei prodotti agricoli e agroalimentari, 2026.

[24] Riso Italiano. Mercosur: prezzi all'export in salita, 2026.

Appendice A

Documentazione dell'implementazione

La struttura dell'implementazione si divide in tre percorsi principali a seconda dell'approccio di ottimizzazione scelto.

Percorsi di ottimizzazione

Percorso A — SAA

```
TabellaProdotti.xlsx

prepare_data.py::prepare_input_data(SAA_flag=True)
   Carica prodotti, simula mercato, genera N scenari di domanda

optimization.py::AssortmentOptimizer.build_model_SAA(...)
   Costruisce il modello MILP a due stadi in Gurobi

optimizer.solve(...)  e  optimizer.export_to_text_SAA(...)
   Risolve e scrive i risultati su file .txt
```

Percorso B — NN surrogata

```
[Prima fase: Training - offline]
create_NN_dataset.py : genera coppie (y, profitto_ricorso)
train_NN.py          : addestra MLP con Optuna,
                     salva surrogate_mlp.pkl
active_learning.py   : migliora la NN iterativamente
                     con nuovi punti

[Seconda fase: Ottimizzazione]
prepare_data.py::prepare_input_data(SAA_flag=False)
optimization.py::build_model_NN(...)  - la NN è integrata
                                        in Gurobi via gurobi_ml
optimizer.solve(...)
```

Percorso C — ibrido deterministico

```
[Prima fase: Training - offline]
create_hybrid_datasets.py -> genera coppie
                             (assortimento, domanda_media)
train_hybrid_det.py       -> addestra Ridge regression
                             per ogni prodotto,
                             salva best_linear_models_list.pkl

[Seconda fase: Ottimizzazione]
prepare_data.py::prepare_input_data()
optimization.py::build_model_hybrid_det(...) -> i modelli lineari
                                                sono inseriti
                                                nel modello Gurobi
optimizer.solve(...)
```

Moduli principali

Il modello di mercato — `models.py`

Questo modulo implementa la **simulazione del mercato** basata su un modello di scelta *latent class* MNL. Di seguito è riportata la gerarchia delle classi:

```
MNLChoiceModel                <- Modello di scelta per un singolo
                                 segmento
LatentChoiceModel             <- Combina più segmenti MNL con
                                 pesi
MarketSimulation              <- Simula il mercato e prepara
                                 le caratteritiche principali
  |-- QualityStressTest       <- Stress test: filtro sulla
                                 qualità
  |-- PriceStressTest         <- Stress test: filtro sul
                                 prezzo
  |-- MarcaDropStressTest     <- Stress test: rimozione marca
                                 o marche
  |-- TipoDropStressTest      <- Stress test: rimozione tipo/i
                                 prodotto
  |-- ScaffaleDropStressTest  <- Stress test: rimozione scaffale/i
ClientArrivals                <- Gestisce arrivi Poisson e scelte
                                 dei clienti
```

I modelli di ottimizzazione — `optimization.py`

Tutto contenuto nella classe `AssortmentOptimizer`. Essa contiene quattro formulazioni MILP diverse dello stesso problema, costruite con **Gurobi**. Struttura della classe:

```
AssortmentOptimizer
  |-- _prepare_mappings()          <- Strutture sparse
                                      (matrici si aggregazione per sostituzioni,
                                      indici per suddividere i prodotti)
```

```
|-- _mean_parameters()           <- Coeff. medi per la formulazione
                                  stocastica
|
|-- build_model_SAA()            <- Modello stocastico a due stadi
                                  completo
|-- build_model_NN()             <- Primo stadio + NN surrogata
|-- build_model_hybrid_det()     <- Ibrido: Ridge predice domanda
|-- build_model_hybrid_NN_stoc() <- [PER ESPANSIONE FUTURA]
                                  Ibrido_stocastico: NN predice
                                  domanda per scenario
|-- build_recurse_only()         <- Solo ricorso
                                  (per valutare la soluzione)
|
|-- solve()                      <- Lancia Gurobi
|-- solve_recurse()              <- Risolve il ricorso
                                  fissando (a,y)
|-- get_single_scenario_solution() <- Risolve SAA su un
                                  singolo scenario
|-- debug_valore_assortimento()  <- Debug: valuta un
                                  assortimento fisso
|
|-- export_to_text_SAA()         <- Report dettagliato
                                  su .txt (SAA)
|-- export_to_text_NN()          <- Report (Surrogato)
|-- export_to_text_hybrid_det()  <- Report (ibrido deterministico)
|-- export_to_text_hybrid_NN_stoc() <- [PER ESPANSIONE FUTURA]
                                       Report (ibrido NN stoc)
|-- close_model()                <- Libera la memoria del
                                  modello Gurobi
```

Preparazione dati — `prepare_data.py`

Questo modulo fa da collante tra i dati grezzi e i modelli. Contiene tutte le funzioni di preprocessing, che prendono il file .csv relativo al catalogo dei prodotti, e produce i dati parametrici per i modelli sia per l'ottimizzazione che per i test e crea i dataset

di addestramento. Le principali funzioni sono riassunte nella tabella A.

Funzione	Utilizzo
`import_dataset()`	Legge `TabellaProdotti.xlsx` e divide la qualità in due colonne (bassa/alta)
`market_sim_init()`	Inizializza `MarketSimulation`, `LatentChoiceModel`, `ClientArrivals`
`prepare_input_data()`	Genera i dati di input per l'ottimizzazione e salva/carica da `.pkl`
`prepare_test_input_data()`	Prepara i dati per il test OOS, con assortimento fissato
`prepare_NN_training_data()`	Genera dataset per la NN surrogata (exploration-exploitation con Gurobi)
`prepare_arrivals_det()`	Genera dataset per il modello ibrido deterministico
`prepare_NN_arrivals_stoc()`	[Per espansione futura] Versione stocastica del dataset ibrido

Funzioni principali del modulo `prepare_data.py`.

Di seguito è riportato un esempio della struttura dell'output generato da `prepare_input_data()`:

```
{
    "base": {
        "prodotti_tab": pd.DataFrame,        # catalogo
           prodotti
        "rate_arrival": list,                # tasso arrivi
            per giorno settimana
```

```
        "sub_matrix": np.ndarray,           # matrice di
            sostituzione
        "prezzi": np.ndarray,               # prezzi euro/
            kg per prodotto
        "cost": np.ndarray,                 # costi euro/
            kg per prodotto
        "holding": np.ndarray,              # costo di
            mantenimento a scaffale
        "capacita_scaffale_dict": dict,     # capacita
            scaffale per posizione
        "capacita_frigo_dict": dict,        # capacita
            frigo per posizione
        "costo_fisso_scaffale": dict,       # costo fisso
            inserimento assortimento
        "penalty_lost_sales": np.ndarray,   # penale per
            domanda persa (= margine)
        "penale_SL": float,                 # penale sul
            livello di servizio
        "SL_target": float                  # target
            livello di servizio (0.95)
    },
    "saa_input": {
        "demand_weekly": np.ndarray,        # scenari
            domanda [S x N_prodotti+1]
        "scenario_prob": np.ndarray,        # pesi degli
            scenari (sommano a 1)
        "warm_demand": np.ndarray,          # scenari
            aggiuntivi per dataset NN
        "max_revenue": np.ndarray           # upper bound
            ricavo per prodotto
    }
}
```

Training dei modelli ML (`train_ML/`)

NN surrogato

(`create_NN_dataset.py`, `train_NN.py`, `active_learning.py`)

Obiettivo: Addestrare una rete neurale che stima il valore del secondo stadio (profitto del ricorso) dato uno stock y.

1. **Generazione dataset** (`create_NN_dataset.py`): Utilizza un framework *exploration-exploitation*: campiona assortimenti vicini alla soluzione SAA e assortimenti casuali. Per ogni campione (a, y) risolve il modello di ricorso esatto per calcolare il target. L'output è `train_data_assortment.csv` con colonne: $[a_0, \ldots, a_N, y_0, \ldots, y_N, \text{target\_profit}]$.

2. **Training MLP** (`train_NN.py`): Input: colonne y_i del dataset; Target: `target_profit`. Usa **Optuna** per il *tuning* degli iperparametri (n_layers, n_units, alpha, lr).

   Salva: `surrogate_mlp.pkl`, `scaler_input.pkl`, `scaler_target.pkl`.

3. **Active Learning** (`active_learning.py`): Loop iterativo: 1) risolve il modello NN in Gurobi, 2) valuta la soluzione trovata con il ricorso esatto, 3) se il *gap* è grande aggiunge nuovi punti al dataset e riaddestra.

Modello ibrido deterministico

(`create_hybrid_dataset.py`, `train_hybrid_det.py`)

Obiettivo: Per ogni prodotto i, addestrare un modello lineare che predice la domanda media attesa dato l'assortimento.

1. **Generazione dataset** (`create_hybrid_dataset.py`): Campiona assortimenti casuali. Per ogni assortimento calcola la domanda media attesa considerando domanda primaria e sostituzione.

   L'output è `dataset_train_deterministico.csv` con colonne:

   $[a_0, \ldots, a_N, d_{0\_\text{mean}}, \ldots, d_{N\_\text{mean}}, \text{delta\_no\_buy}]$.

2. **Training Ridge Regression** (`train_hybrid_det.py`): Un modello per ogni prodotto (+ uno per `delta_no_buy`). Usa **RidgeCV** con *cross-validation* per scegliere il parametro di regolarizzazione. Salva: `best_linear_models_list.pkl`, `scaler_y.pkl`.

Test e validazione

Stress Test di mercato (`test_segment/`)

Ogni file `test_DROP_*.py` esegue uno stress test rimuovendo dal catalogo un sottoinsieme di prodotti e osservando come cambiano le probabilità di scelta dei clienti. I test disponibili sono riassunti nella tabella A.

```
prodotti_tab = import_dataset()
test = MarcaDropStressTest(prodotti_tab, marca_drop=["A",
   "B"])
res  = test.sim_MarcaDropStressTest(MNLChoiceModel,
   LatentChoiceModel)
plot_choice_probabilities(res, test_label="Drop Marche A e
    B", ...)
```

File	Cosa rimuove
`test_DROP_A/B/C/D.py`	Marca singola
`test_DROP_AD.py`, `test_DROP_BC.py`	Coppia di marche
`test_DROP_Alto/Occhi/Basso.py`	Posizione a scaffale
`test_DROP_Pasta/Riso/Gnocchi.py`	Tipo prodotto
`test_drop&filter_BqM7.py`	Drop marca B + filtro qualità
`test_filter_m65.py`, `test_filter_M8.py`	Filtri solo sul prezzo

Test di stress disponibili.

Test Out-of-Sample (`test_OOS/`)

Dato un assortimento e uno stock ottimale (trovato con uno dei tre approcci), valuta la performance reale simulando 300 settimane indipendenti. I vettori si trovano

nei risultati dell'ottimizzazione (file `test_res/assortimento.csv` generato da `export_to_text_SAA`).

```
assortment = [0, 1, 0, ...]  # vettore binario, lungh. =
   num. prodotti
quantity   = [0, 45, 0, ...] # vettore intero, lungh. =
   num. prodotti
main(assortment, quantity)
```

Output:

- Stampa a video: vendite medie, *lost service* medio, *worst-case percentile* 5%/95%.
- File CSV: `Risultati_(sxp)_...` dettaglio per scenario × prodotto e `KPI_scenari_...` dettaglio cumulato per scenario.
- Grafici: istogramma vendite, istogramma *lost service*, heatmap *worst case*.

Nota: Il modulo `main_test_shock.py` è identico ma permette di specificare `prob_scen` diversi (i pesi dei segmenti) per simulare shock di mercato.

Come Riprodurre i Risultati

Prerequisiti

Avere `TabellaProdotti.xlsx` nella cartella indicata in

`prepare_data.py::import_dataset()` (da modificare con il path locale)

e installare le dipendenze necessarie (4).

Esecuzione dei modelli

Approccio SAA (punto di partenza e baseline):

```
# Dalla cartella Codice/
python main_SAA.py
```

Produce: `analisi_risultati_*.txt` e `test_res/assortimento.csv`.

Approccio NN surrogato:

```
# Step 1: genera il dataset
python train_ML/create_NN_dataset.py

# Step 2: addestra la NN
python train_ML/train_NN.py

# Step 3 (opzionale): active learning
python train_ML/active_learning.py

# Step 4: ottimizzazione
python main_NN.py
```

Approccio ibrido deterministico:

```
# Step 1: genera il dataset
python train_ML/create_hybrid_datasets.py

# Step 2: addestra i modelli Ridge
python train_ML/train_hybrid_det.py

# Step 3: ottimizzazione
python main_hybrid.py
```

Validazione Out-of-Sample: Dopo aver ottenuto un assortimento ottimale, modificare i vettori `assortment` e `quantity` in `test_OOS/main_test_OOS.py` e lanciarlo.

Ogni script python associato presenta una descrizione iniziale delle classi, metodi e funzioni implementate. Per ognuno di questi, è presente un commento introduttivo che spiega gli input richiesti, gli output prodotti e in quale script main_{nome}.py viene richiamato.