POLITECNICO DI TORINO

DIPARTIMENTO DI SCIENZE MATEMATICHE

Corso di Laurea in Ingegneria Matematica

Metodologia per l'identificazione del Data Drift in una collezione di documenti testuali

RELATORE:
Prof. **Tania Cerquitelli**

CANDIDATO:
Elisa Grassi
Matr:255010

Anno Accademico 2019/2020

Indice

Introduzione

Nel contesto del Data Mining e del Machine Learning, esistono delle applicazioni in cui i dati evolvono nel tempo, e con essi cambiano le strutture e relazioni interne che li caratterizzano. Questo fenomeno, chiamato Concept Drift, determina la presenza di "concetti" diversi nei nuovi dati, cioè di strutture che non erano presenti nei dati iniziali. In questo contesto, i modelli di analisi creati sulla base dei dati di partenza, possono ad un certo punto diventare obsoleti e non adatti a descrivere i nuovi dati in ingresso. L'obiettivo sarebbe quindi quello di sviluppare dei modelli che siano in grado di riconoscere il drift dei dati e di adattarsi ad esso, in modo da imparare a riconoscere le nuove strutture interne che li caratterizzano.
La metodologia proposta in questa tesi mira alla creazione di un procedimento per l'identificazione del Concept Drift in dati testuali. In particolare si analizza il caso di un classificatore one-class, in grado inizialmente di identificare la classe dei dati di partenza. Si vuole determinare se, in presenza di nuovi dati, esso continua a funzionare correttamente o se si degrada. I dati nuovi infatti, se non appartengono alla stessa classe dei dati di partenza, verranno identificati come rumore o anomalie. Tramite algoritmi di clustering si analizzerà il rumore identificato, in particolare si determinerà se al suo interno si creano delle nuove classi. In questo caso il classificatore one-class non sarà più adatto a descrivere i nuovi dati in ingresso.
Il metodo è stato testato su articoli di Wikipedia di due diversi argomenti e su recensioni di film positive e negative. I dati testuali vengono trasfomati mediante la rete neurale del Word2vec in vettori numerici. Questa rete neurale trasforma le parole presenti nei documenti in vettori in modo che parole che si trovano spesso negli stessi contesti occuperanno le stesse regioni dello spazio.
Nel primo esperimento si considerano come dati storici i testi del primo argomento e come dati nuovi i testi del secondo. Il classificatore one-class è addestrato per imparare a riconoscere la classe dei dati storici, quindi i dati nuovi vengono classificati come rumore o anomalie. Tramite tecniche di clustering sul rumore riconosciuto, si identifica la presenza di nuove classi al suo interno, che in questo caso corrispondono

alla presenza di argomenti nuovi, inizialmente non conosciuti dal classificatore.
Nel secondo esperimento si considerano come dati storici le recensioni positive e come dati nuovi le negative. Il classificatore imparerà a riconoscere la classe delle recensioni positive, ma questa volta all'interno del rumore identificato non si riesce a distinguere la presenza di nuovi cluster. Ciò avviene a causa della trasformazione operata sulle parole, che pone nelle stesse regioni dello spazio parole che si trovano in contesti simili. Infatti in questo caso, l'argomento trattato nei documenti testuali (recensioni di film) non varia, e non si riescono ad identificare dei nuovi cluster nel rumore analizzato.
Sviluppi futuri della metodologia potrebbero essere la creazione di una metrica per l'identificazione quantitativa delle nuove classi formatesi e di strumenti specifici per la caratterizzazione delle nuove classi alternative alle word-cloud.

Capitolo 1

Data Mining

1.1 Introduzione

In un mondo in cui ogni giorno viene raccolta una grandissima quantità di dati è importante trovare degli strumenti per analizzare e interpretare tali dati. È a questa esigenza che risponde la nascita del Data Mining, ovvero dell'insieme di metodologie necessarie per trasformare i dati in informazioni e conoscenza.

Il Data mining può essere visto come l'evoluzione naturale della Tecnologia dell'Informazione. Infatti, già dalla metà degli anni '80 i sistemi di raccolta dati si sono spostati verso tecnologie molto più avanzate come i data-warehouse (sistemi in grado di raccogliere dati eterogenei sotto uno schema unificato) mentre negli anni '90 nascono Web-Based Database. Lo sviluppo di questi sistemi ha portato da un lato alla disponibilità di volumi di dati sempre più grandi, dall'altro alla necessità di strumenti sempre più sviluppati per poterli analizzare. Infatti questa enorme quantità di dati, eccedendo la capacità umana di comprensione, diviene inutile se non si dispone di metodologie che ci consentono di estrarre informazioni e significati da questi ultimi.

Questo capitolo è stato scritto sulla base del lavoro di Han, Kamber e Pei in [1], Tan, Steinbach e Kumar in [2] e Hastie, Tibshirani, Friedman [3]

1.2 Data Mining: definizione

Si definisce Data Mining il processo di estrazione di conoscenza e informazione da grosse quantità di dati. A volte con questo termine si indica l'intero processo di "scoperta di conoscenza", altre volte un solo passo di questo processo. Schematizziamo il procedimento dell'estrazione della conoscenza come segue:

1. **Pulizia dei dati** (rimozione del rumore e di dati inconsistenti)
2. **Integrazione dei dati** (dati provenienti da fonti diverse sono combinati insieme)
3. **Selezione dei dati** (vengono selezionati i dati interessanti per l'analisi d'interesse)
4. **Trasformazione dei dati** (i dati devono essere trasformati nel formato idoneo per svolgere le operazioni richieste dal tipo di analisi che si vuole effettuare)
5. **Data Mining** (si estraggono le strutture d'interesse presenti all'interno dei dati)
6. **Valutazione dei risultati** (si verifica che i risultati ottenuti siano d'interesse o meno)
7. **Rappresentazione dei risultati mediante tecniche di visualizzazione**

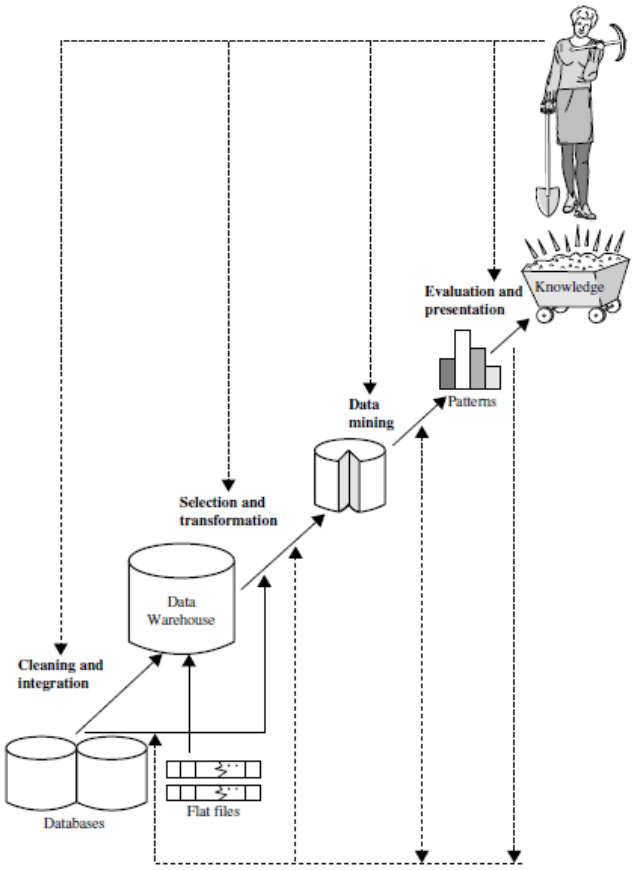
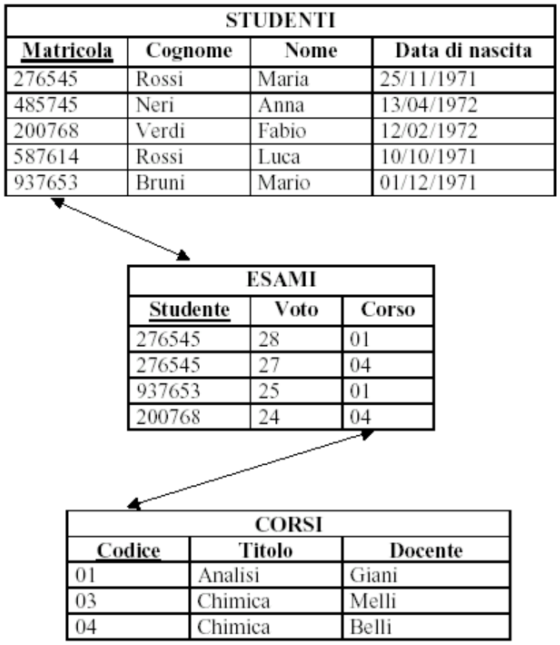


Figura 1.1:

1.3 Tipi di dati

Il data mining può essere applicato a tutte le tipologie di dati. Le tipologie di dati più semplici e più utilizzate sono quelle dei database, dei data ware-house e dei dati transazionali. Altre tipologie di dati possono essere dati testuali, immagini, grafi.

1. **Database data** Un database relazionale è una collezione di tabelle, ognuna delle quali consiste in un insieme di attributi (le colonne) e raccoglie una grande quantità di tuple (le righe). Ogni tupla in un tabella relazionale rappresenta un oggetto che è identificato da un codice univoco all'interno della tabella, detto chiave, ed è descritto dai valori degli attributi. Le relazione tra le tabelle sono poi esplicitate tramite i vincoli di integrità referenziale.



STUDENTI

Matricola	Cognome	Nome	Data di nascita
276545	Rossi	Maria	25/11/1971
485745	Neri	Anna	13/04/1972
200768	Verdi	Fabio	12/02/1972
587614	Rossi	Luca	10/10/1971
937653	Bruni	Mario	01/12/1971

ESAMI

Studente	Voto	Corso
276545	28	01
276545	27	04
937653	25	01
200768	24	04

CORSI

Codice	Titolo	Docente
01	Analisi	Giani
03	Chimica	Melli
04	Chimica	Belli

Figura 1.2:

2. **Data ware-house** Un data ware-house è un sistema in cui si raccolgono dati eterogenei sotto uno schema unificato. In genere in un data warehouse si raccologono informazioni storicizzate quindi si raccolgono dati aggregati, ottenuti tramite l'elaborazione di altri dati. Un data ware-house viene solitamente rappresentato tramite un cubo in cui ad ogni dimensione corrisponde un attributo (o un insieme di attributi) e ogni cella corrisponde a qualche valore aggregato (ad esempio il numero di articoli venduti in un determinato periodo di tempo).

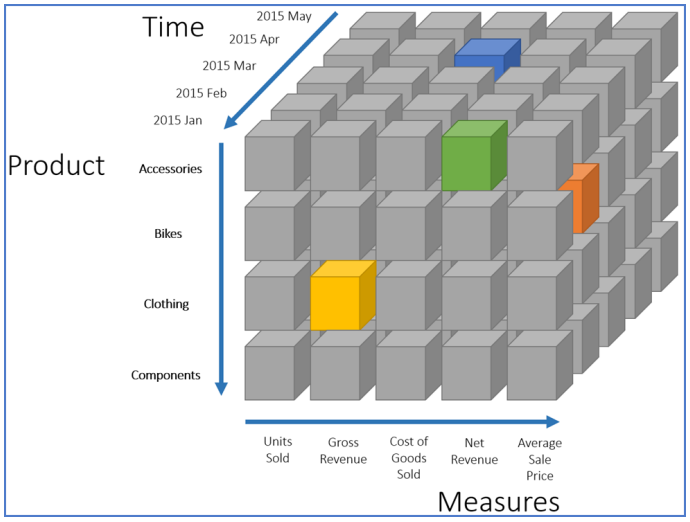


Figura 1.3:

3. **Dati transazionali** In generale in un database transazionale raccoglie dati che originano quando ha luogo una transazione (ad esempio l'acquisto di un biglietto aereo, un prelievo bancario, ...). Una transazione viene registrata con un codice univoco e con la lista degli elementi che concorrono alla realizzazione della transazione stessa.

trans_ID	*list_of_item_IDs*
T100	I1, I3, I8, I16
T200	I2, I8
...	...

Figura 1.4:

4. **Altri dati** Esistono molti altri tipi di dati che si può voler analizzare, ad esempio serie storiche, strem data (flussi continui di dati), dati spaziali (mappe), testi e dati multimediali. Queste tipologie di dati sono facili da raccogliere e conservare, ma difficili da analizzare e richiedono strumenti di Data Mining più avanzati.

1.4 Strumenti del Data Mining

Avendo come abiettivo quello di estrarre conoscenza dai dati, il Data Mining incorpora varie tecniche da molteplici discipline, come la statistica, il machine learning, il riconoscimento di pattern, sistemi di database e data warehouse, metodologie per la raccolta di informazioni, metodologie per la visualizzazione dei risultati.

1.4.1 Strumenti Statistici

L'utilizzo di modelli statistici è indispensabile per diversi scopi come ad esempio:

1. Per stimare parametri di interesse come la media o la varianza;
2. Per stimare la distribuzione di probabilità di un insieme di dati;.
3. Per modellizzare il rumore dei dati o inserire valori mancanti;
4. Per creare modelli predittivi;
5. Per descrivere una collezione di dati;
6. Per il riconoscimento di pattern nei dati;
7. Per convalidare i risultati mediante test d'ipotesi.

Utilizzare i modelli statistici può essere molto complicato, soprattutto quando essi vanno applicati a una grande quantità di dati. Infatti molti modelli statistici hanno un'alta complessità computazionale, quindi possono richiedere tempi molto lunghi se applicati a vasti datasets.

1.4.2 Machine Learning o Apprendimento automatico

Il machine Learning consiste in un insieme di tecniche di analisi dati che automatizzano alcuni processi. L'idea che sta dietro il machine learning è che i sistemi possono imparare dai dati, identificare modelli e patterns autonomamente e prendere decisioni con un intervento umano ridotto al minimo. Alcuni dei problemi classici del machine learning sono:

1. **Supervised Learning**: si vuole stimare una funzione che mappi un valore di input in un valore di output, sulla base di esempi costituiti da coppie di dati di input e output.
   Più formalmente possiamo definire un esempio come una coppia di valori

$(x, f(x))$, dove x è l'input, $f(x)$ è l'output ed f è la funzione che vogliamo stimare. Quindi un modello di Supervised learning, data una collezione di esempi di f, restituisce una sua stima h che approssima f. Esempi di Supervised Learning sono la Classificazione, la Regressione lineare e logistica,

2. **Unsupervised Learning**: consiste nel fornire al sistema una serie di input in cui che esso dovrà trovare classi o patterns sulla base di caratteristiche comuni, per poi effettuare previsioni sugli input successivi. In questo caso i dati non possiedono alcuna etichetta, pertanto non si conosce l'output di partenza. Esempi di Unsupervised learning sono il clustering o l'anomaly detection.

3. **Semi-supervised Learning**: utilizza come dati di input dati sia etichettati che non etichettati. Questo tipo di approccio è utilizzato principalmente quando si hanno a disposizione tanti dati ma è difficile ottenere le etichette. Una di queste tecniche consiste nell'utilizzare i dati etichettati per definire il modello e quelli non etichettati per raffinare i bordi tra le classi.

4. **Active Learning**: è una tecnica che richiede l'utilizzo attivo dell'utente nella fase di apprendimento in modo da ottimizzare la qualità del modello. Ad esempio si potrebbe chiedere all'utente di assegnare un'etichetta a un dato.

1.4.3 Database Systems e DataWarehouse

Molti compiti del data mining richiedono di utilizzare grandi quantità di dati. Pertanto l'utilizzo di tecnologie di database scalabili può agevolare il data mining nella ricerca di alta efficienza. Inoltre il Data Mining stesso può essere utilizzato per estendere la capacità di database esistenti di soddisfare dei requisiti più avanzati.

1.4.4 Information Retrieval (IR)

L'Information Retrieval è l'insieme di tecniche che si occupano di cercare documenti testuali o multimediali o di estrarre le informazioni contenute in essi. Gli strumenti adottati per analizzare questo tipo di documenti sono strumenti probabilistici. Per fornire un esempio si possono considerare dei dati testuali e stabilire se essi sono simili. A tale scopo si può associare ad ogni testo una distribuzione di probabilità sulle parole e verificare che essi abbiano delle distribuzioni di probabilità vicine. Il data mining su queste tipologie di dati presenta molte difficoltà in quanto necessita di strumenti molteplici ed eterogenei.

1.5 Tecniche di Data Pre-Processing

Il Pre-Processing consiste nell' insieme di tecniche che mirano a preparare i dati alla fase di analisi vera e propria, eliminando gli errori e il rumore, e trasformandoli nel formato più adatto per l'analisi. Il Pre-Processing mira a migliorare la qualità dei dati. Infatti una qualità bassa dei dati comprometterebbe il raggiungimento degli obiettivi dell'analisi, mentre una buona qualità la facilita e la rende più efficiente. Le caratteristiche che concorrono alla qualità di un dato sono:

1. Accuratezza: il suo valore deve corrispondere al dominio che rappresenta
2. Completezza: deve presentare tutti gli attributi necessari
3. Consistenza: deve rispettare le regole che sono state stabilite sull'insieme dei dati
4. Attualità: il dato deve essere elaborato e fornito nel tempo richiesto
5. Credibilità: proviene da una fonte verificata

Esistono molte tecniche che vengono utilizzate nella fase di Preprocessing. Il Data Cleaning è utilizzato per rimuovere rumore ed errori dai dati; il Data Integration serve a integrare sotto uno stesso schema dati che provengono da fonti diverse; il Data Reduction riduce la dimensione dei dati; il Data Trasformation trasforma i dati nel formato necessario per condurre le analisi.

Data Cleaning

I compiti principali del Data Cleaning sono il riempimento dei valori mancanti, la rimozione del rumore, identificazione degli outliers e delle inconsistenze dei dati. Per l' **eliminazione dei Valori Mancanti** si possono mettere in atto diverse strategie:

1. Ignorare il dato
2. Riempirlo con un valore costante
3. Riempire il valore manualmente
4. Utilizzare una grandezza come la media o la mediana
5. Utilizzare il valore più probabile

Il **Rumore (Noise) dei dati** è un errore randomico che a volte può essere descritto mediante una variabile aleatoria. Anche per la rimozione del rumore esistono diverse tecniche:

1. Binning: i dati vengono divisi in gruppi ("bins") e in ogni gruppo i valori dei singoli elementi vengono rimpiazzati con uno tra la media, la mediana , il valore massimo o il valore minimo del gruppo
2. Regressione: i dati vengono conformati a quelli della funzione di regressione che viene fittata su di essi.
3. Analisi degli outliers: si identificano gli outliers tramite diverse tecniche come ad esempio il clustering.

Data Integration

In questa fase si vogliono integrare dati provenienti da fonti diverse in uno stesso sistema. Una buona integrazione evita le ridondanze e le inconsistenze nei dati. I problemi principali del Data Integration sono:

1. L'Identificazione dell'Entità: quando si integrano dati provenienti da fonti diverse bisogna stabilire lo schema unico che essi devono avere.
2. La ridondanza: un attributo è ridondante se può essere ricavato da altri attributi. La ridondanze possono essere riconosciute mediante il test χ^2 o i coefficienti di correlazione o covarianza.
3. Conflitti tra dati: ad esempio dati che rappresentano la stessa grandezza possono avere unità di misura o rappresentazioni diverse.

Data Reduction

Si vuole ottenere una rappresentazione dei dati semplificata ma che al contempo mantenga le proprietà dei dati originari. Alcune tecniche fondamentali del Data Reduction sono:

1. Riduzione della dimensionalità: è il procedimento tramite cui si riduce il numero di variabili aletorie o di attributi. Include metodi come la trasformata wavelet o l'analisi delle componenti principali.

2. Riduzione della numerosità: rimpiazza il volume originario dei dati con una loro rappresentazione più piccola. Utilizza metodi parametrici (come la regressione lineare o logistica) o non parametrici (come il sampling o il clustering).

3. Compressione dei dati: Si applica una trasformazione per ottenere una versione compressa dei dati.

Data Trasformation

Con questa fase i dati vengono trasformati in una forma più adatta all'analisi che si vuole svolgere. I procedimenti fondamentali che vengono attuati sono:

1. La costruzione di nuovi attributi;

2. L'applicazione di operazioni di aggregazione come somme o conteggi;

3. La normalizzazione dei valori di alcuni attributi;

4. La discretizzazione di valori di attributi, passando ad esempio da valori continui a intervalli;

5. Introduzione di gerarchie tra gli attributi (ad esempio "strada" verrà generalizzato da un attributo più ad alto livello come "città").

1.6 Tecniche di Classificazione

La Classificazione è una forma di analisi dei dati che estrae dei modelli che descrivono delle classi. Questi modelli, chiamati Classificatori sono dei modelli predittivi e possono essere a valori categorici o quantitativi. Nel primo caso predicono il valore assunto da variabili categoriche che rappresentano le etichette delle classi. Nel secondo operano delle predizioni numeriche stimando una funzione a valori continui (come nel caso della Regressione).
In generale la Classificazione è descritta come un processo a due fasi.
Nella prima fase di **apprendimento** il classificatore è costruito sulla base dei dati di train. Questi ultimi sono costituiti da un insieme di tuple (X, l), ognuna delle quali è costituita da un vettore $X = (x_1, ..., x_n)$ e dal valore l dell'etichetta della classe a cui X corrisponde. Nel caso in cui si voglia costruire un predittore a valori numerici la fase di apprendimento corrisponde alla fase in cui viene realizzata la stima h della funzione f, sulla base di dati di train costituiti da tuple $(x, f(x))$.
Nella seconda fase di **test** il classificatore viene testato su una seconda parte di dati,

le tuple di test, e sulla base di indici come l'accuracy si valuta il suo funzionamento. Alcuni degli metodi di classificazione più utilizzati sono gli Alberi di Decisione e i Classificatori Bayesiani.

1.6.1 Alberi di Decisione

Questi algoritmi di classificazione costruiscono un albero sulla base delle tuple di training. Nella struttura ad albero ottenuta i nodi foglia rappresentano le classificazioni finali ottenute e i rami le proprietà che portano a quelle classificazioni. Ogni nodo interno rappresenta un test su un attributo che viene associato ad un predicato. Il predicato che si associa ad ogni nodo interno è chiamato condizione di split. Per evitare il problema dell'overfitting si adotta una strategia detta "pruning", che utilizza metodi statistici per eliminare i rami meno affidabili.

1.6.2 Classificatori Bayesiani

Questi classificatori utilizzano il teorema di Bayes per stimare la probabilità di un dato di appartenere ad una certa classe. Per diminuire il costo computazionale viene fatta l'assunzione che ci sia indipendenza condizionale ovvero che la probabilità del valore di un attributo di appartenere ad una classe è indipendente da quella degli altri attributi.

Teorema di Bayes

Consideriamo una tupla X e un'ipotesi H secondo cui la tupla appartiene ad una classe C. $P(H|X)$ è la probabilità che l'ipotesi sia vera. $P(X|H)$ è la probabilita a posteriori di X data H. $P(H)$ è la probabilità a priori. Il teorema di Bayes dice che:

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)} \tag{1.1}$$

1.6.3 Backpropagation

Il BackPropagation è un algoritmo per la classificazione che impiega il metodo della discesa del gradiente per determinare un insieme di pesi tramite cui si realizzano le predizioni delle classi. Nella fase di apprendimento, tramite un procedimento di iterazione sulle tuple di training, confronta il valore della classe predetto con quello vero. In questo caso come valore della classe si intende o l'etichetta della classe o, in caso di predizioni numeriche, di un valore continuo. I pesi nel network

sono inizializzati a piccoli valori randomici (ad esempio che vanno da -1 a 1). Per ognuna delle tuple di train i pesi sono modificati in modo da minimizzare il mean-squared-error tra le predizioni e i valori reali delle classi. Queste modifiche sono fatte iterando il procedimento all' indietro (da qui il nome Backpropagation) cioè dal layer di output a quello di input. In genere i pesi convergono verso dei valori (anche se la convergenza non è garantita) e il processo di apprendimento si ferma.

1.6.4 Support Vector Machine

La Support Vector Machine è un algoritmo per la classificazione per dati lineari e non lineari. Esso utilizza delle mappe non lineari per trasformare i dati di training in modo da aumentarne la dimensionalità. Dentro questo nuovo spazio creato l'SVM cerca di determinare l'iperpiano ottimale che separa le tuple di una classe da quelle dell'altra. Sfrutta quindi la proprietà secondo cui, data una mappa non lineare appropriata, che trasforma i dati in uno spazio di dimensione sufficientemente elevato, i dati di due classi possono sempre essere separati da un iperpiano.

1.7 Clustering

L'analisi dei gruppi (Clustering analysis) consiste nel processo di raggruppamento di una collezione di oggetti in sottinsiemi o "clusters". Ognuno di questi sottinsiemi viene creato in modo che gli elementi contenuti in uno stesso siano più simili tra loro rispetto che ad elementi di gruppi diversi. L'insieme di gruppi che viene creato prende il nome di "clustering". Diverse metodologie, se applicate allo stesso insieme di dati, possono portare a clustering diversi. Il clustering può anche essere inteso come un metodo di classificazione automatica, in quanto costruisce dei gruppi di dati ognuno con caratteristiche comuni. Il Clustering può anche essere utilizzato come metodo di outliers detection.
Comune a tutti i metodi di Clustering Analysis è la necessità di definire un grado di similarità o dissimilarità tra gli oggetti e tra i gruppi. Infatti proprio in base alla loro similarità due oggetti verranno assegnati allo stesso gruppo o a due diversi. Gli algoritmi di Clustering possono essere di tre tipi:

1. **Clustering Partizionale**: Dato un insieme di n oggetti, costruisce k partizioni dei dati dove ogni partizione rappresenta un cluster. I cluster iniziali possono essere inizializzati con diversi criteri (ad esempio randomicamente). Successivamente, tramite una tecnica di riallocazione iterativa, il clustering viene migliorato spostando gli elementi da un gruppo all'altro.

2. **Metodi Gerarchici**: In questo procedimento si cerca di ordinare i gruppi in una gerarchia. A tale scopo si attua un processo iterativo in cui ad ogni passo si fondono/separano alcuni clusters, in modo che i nuovi cluster presentino un alto valore di similarità o un basso valore di dissimilarità.

3. **Metodi basati sulla densità**: questi metodi si basano sull'idea di lasciare crescere un dato cluster fino a quando la densità di oggetti nell'intorno degli elementi del cluster eccede un dato valore soglia. Ad esempio, per ogni punto dentro un cluster, all'interno del suo vicinato ('neighborood') di un certo raggio ci deve essere almeno una certa quantità di elementi. Questi metodi possono essere utilizzati per filtrare il rumore o gli outliers, o per scoprire clusters di forma arbitraria.

4. **Metodi basati su griglia**: Questi metodi dividono lo spazio in un numero finito di celle che formano una struttura di giglia. In generale presentano tempi di elaborazione più brevi rispetto agli altri algoritmi.

1.7.1 Clustering Partizionale

Gli algoritmi di Clustering Partizionale sono i più È utilizzati. Prespecificato un numero di cluster $K < N$ assegnano ogni elemento del dataset ad un unico e solo cluster. La funzione:

$$C : \mathbb{R}^p \to \{1, ..., K\} \quad C(x_i) = j, j \in 1, ..., K \tag{1.2}$$

associa ogni osservazione ad un cluster. L'obiettivo dell'algoritmo sarà quindi di trovare la funzione C che assegna le osservazioni ai clusters in modo ottimale, ovvero quella che minimizza una funzione di perdita $W(C)$

Detta $d(x_i, x_j)$ la misura di dissimilarità tra le osservazioni x_i e x_j definiamo la funzione di perdita:

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{i:C(x_i)=k} \sum_{j:C(x_j)=k} d(x_i, x_j) \tag{1.3}$$

K-means Algorithm

È uno degli algoritmi di clustering più diffusi. È un metodo iterativo particolarmente adatto per dati quantitativi. Come misura di dissimilarità è scelta la metrica

euclidea:

$$d(x_i, x_j) = \sum_{k=1}^{p} (x_{ik} - x_{jk})^2 = \|x_i - x_j\|^2 \tag{1.4}$$

Data la funzione C che assegna ogni osservazione a un cluster, la funzione di perdita (2.31) può quindi essere definita come:

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{i:C(x_i)=k} \sum_{j:C(x_j)=k} \|x_i - x_j\|^2 = \sum_{k=1}^{K} N_k \sum_{j:C(x_j)=k} \|x_i - \overline{x_j}\|^2 \tag{1.5}$$

dove $N_k = \#$ elementi nel cluster k e $\overline{x_k}$ è la media degli elementi del cluster k. Quindi le N osservazioni vengono assegnate ai K clusters in modo da minimizzare la media delle dissimilarità dei punti dalla media dei punti del cluster.

Osservazione Dato il problema di minimo:

$$C^* = \min_{C} \quad \sum_{k=1}^{K} N_k \sum_{i:C(x_i)=k} \|x_i - \overline{x_k}\|^2$$

esso può essere risolto notando che

$$\operatorname*{argmin}_{m} \quad \overline{x_s} = \sum_{i \in S} N_i \|x_i - m\|^2$$

Quindi possiamo ottenere C^* risolvendo il seguente problema di minimizzazione:

$$\operatorname*{argmin}_{C, \{m_k\}_1^K} \quad \sum_{k=1}^{K} N_k \sum_{i:C(i)=k} \|x_i - m_k\|^2$$

Otteniamo quindi il seguente algoritmo:

Algoritmo 1 K-means Clustering

1. Si inizializzano i K cluster randomicamente

   (a) Si calcolano le medie $\{m_1, ..., m_k\}$

   (b) Si assegna ogni osservazione al cluster più vicino, ovvero:

   $$C(x_i) = \operatorname*{argmin}_{1<k<K} \quad \|x_i - m_k\|^2$$

2. Si ripetono gli step (a) e (b) finchè i cluster non si modificano più

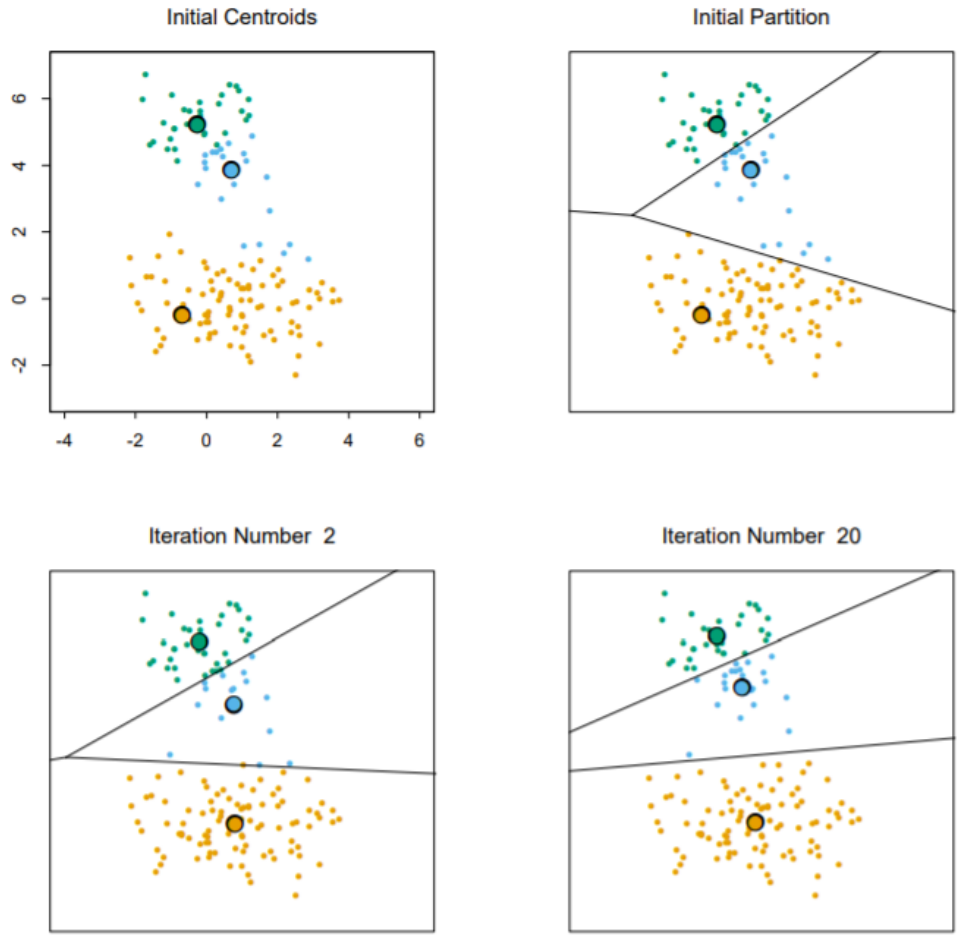


Figura 1.5: Esempio di iterazioni successive dell'agoritmo del K-means clustering

Le limitazioni di questo algoritmo sono date dal fatto che il risultato finale dipende sia dal numero K di clusters che si è predefinito, sia da come i cluster sono stati inizializzati. Infatti inizializzazioni diverse possono portare a risultati diversi e la scelta del parametro K^* ottimale può essere difficoltosa.

1.7.2 Clustering gerarchico

Questi metodi superano alcune delle limitazioni del K-means, in quanto non necessitano di preimpostare un numero di cluster K e di inizializzare di questi ultimi. Essi creano una rappresentazione gerarchica dei cluster, in cui ad ogni livello i nuovi cluster sono creati unendo i cluster del livello precedente che sono più simili tra di loro. Al livello più basso ogni cluster contiene una singola osservazione. Al livello

più alto si ha un solo cluster che contiene tutte le osservazioni. Per realizzare questo procedimento sarà quindi necessario specificare una misura di dissimilarità tra i gruppi di osservazioni disgiunte.
Le strategie del clustering gerarchico sono due: agglomerativo (bottom up) e divisivo (bottom down).

I metodi agglomerativi partono dal basso (ovvero dalla configurazione iniziale in cui ogni cluster contiene un singolo elemento) e ad ogni livello seleziona le coppie di gruppi che presentano la dissimilarità più bassa e li unisce in un unico gruppo. Quindi ad ogni livello si avrà un numero di clusters inferiore rispetto al livello precedente.
I metodi divisivi iniziano dall'alto e ad ogni livello ricorsivamente dividono in due uno dei cluster esistenti. La divisione scelta è quella massimizza la dissimilarità tra i due nuovi gruppi creati. Con entrambi i metodi ci sono $N-1$ livelli nella gerarchia. In entrambi i metodi ogni livello della gerarchia rappresenta un modo in cui i dati sono stati raggruppati in un numero diverso di cluster disgiunti, quindi l'intera gerarchia rappresenta una sequenza ordinata di raggruppamenti. Starà a chi utilizza questi algoritmi determinare la scelta del livello (raggruppamento) più significativo per i dati considerati.
Il procedimento ricorsivo della divisione o agglomerazione dei clusters può essere rappresentato mediante l'utilizzo di un albero binario. Il nodo radice rappresenta l'intero dataset, le N foglie finali rappresentano le singole osservazioni, mentre gli altri nodi rapresentano i gruppi.
I metodi agglomerativi possiedono un'importante proprietà, ovvero la monotonicità. Infatti la dissimilarità tra i gruppi che vengono uniti è monotona crescente nel livello della gerarchia. Ciò vuol dire che l'albero binario può essere disegnato in modo che l'altezza di ogni nodo sia proporzionale alla dissimilarità dei due gruppi che corrispondono ai due nodi figli. Le foglie in basso, che rappresentano le singole osservazioni, sono poste ad altezza 0. Un tipo di grafico che ha queste proprietà prende il nome di dendrogramma.

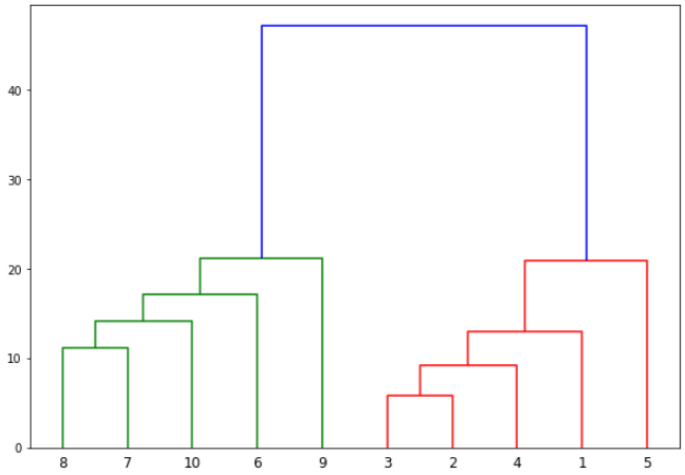


Figura 1.6: Esempio di dendrogramma

1.7.3 Metodi basati sulla Densità: DBscan

I metodi di Cluster Gerarchico e Partizionale, essendo stati ideati per trovare cluster di forma sferica, fanno difficoltà a trovare cluster di forme arbitrarie. Pertanto, forniti dati di questo tipo, idenficheranno comunque delle regioni convesse che includono rumore e outliers. Per modellare cluster di forma arbitraria sono più indicate le metodologie basate sulla densità che modellizzano i cluster come delle regioni dense nello spazio dei dati, separati da regioni in cui i dati sono sparsi.

DBscan

Il metodo del DBSCAN (Density-Based Spatial Clustering of Applications with Noise) cerca tra i dati degli oggetti detti "core objects", ovvero degli oggetti che hanno un vicinato denso. Una volta determinati questi ultimi li connette tra loro formando regioni dense che saranno i clusters. Il DBSCAN utilizza due parametri che devono essere specificati dall'utente:

1. ϵ **>0** è utilizzato per specificare il raggio del vicinato ('neighborood') che consideriamo per ogni oggetto. Quindi definiamo **ϵ-neighborood** di un oggetto o lo spazio racchiuso dentro un raggio di lunghezza ϵ e di centro o. Grazie alla sua grandezza fissa ϵ, la densità del vicinato può essere misurata con il solo numero di elementi al suo interno.

2. **MinPts**: specifica una soglia di densità minima. Un oggetto viene considerato un **core object** se nel suo ϵ**-neighborood** c'è un numero di oggetti almeno pari a **MinPts**.

Dato un insieme D di oggetti possiamo identificare tutti i **core objects**, rispetto ai parametri ϵ e MinPts. A partire dai **core objects** si identificheranno delle regioni dense di dati che costituiranno i clusters. Prima di analizzare il meccanismo di identificazione dei cluster è necessario fornire alcune definizioni:

Definizione 1. Dato un core-object q ed un oggetto p, si dice che p è **direttamente density-reachable** da q (ripetto ai parametri ϵ e MinPts) se p è nell' ϵ**-neighborood** di q.

Definizione 2. Dati due oggetti p e q di dice che p è **direttamente density-reachable** da q se e solo se q è un core-object e p è nell' ϵ**-neighborood** di q.

Definizione 3. Dati due oggetti p e q si dice che p è **density-reachable** da q (ripetto ai parametri ϵ e MinPts) se esiste una catena di oggetti $\{p_1,..., p_n\}$ tali che $p_1 = q$ e $p_n = p$ e $\forall i \in \{1, .., n\}$ p_{i+1} è **direttamente density-reachable** da p_i

Per connettere i vari core-objects e i loro vicinati in un'unica regiona di densità il DBSCAN utilizza la nozione di $density - connectedness$

Definizione 4. Dati due oggetti p_1 e $p_2 \in D$ si dice che essi sono **density-connected** (ripetto ai parametri ϵ e MinPts) se esiste un oggetto q tale che sia p_1 che p_2 sono density-reachable da q (rispetto ai parametri ϵ e MinPts)

A partire da questa nozione di **density-connectedness** possiamo tovare delle regioni connesse dense di dati ed identificare in queste i clusters. Allo scopo di determinare i cluster si attua un procedimento di diversi passi:

1. Tutti gli oggeti in D sono marcati come non-visitati

2. DBSCAN seleziona randomicamente un oggetto non visitato p, lo marca come visitato e verifica se è un core-object. Se non lo è, esso viene marcato come elemento di rumore, se invece lo è viene creato un cluster C per p.

3. Se si realizza la seconda opzione tutti gli elementi del vicinato di p sono aggiunti ad un sottinsiame N di candidati.

4. il DBSCAN considera ogni elemento p' in N e lo marca come visitato. Poi verifica se esso sia un core-object o meno. Se lo è anche tutti gli elementi del vicinato di p' vengono aggiunti a N.

5. se p' non appartiene ancora a nessun cluster, viene aggiunto a C
6. Questo procedimento viene iterato finchè C non si può più espandere, cioè finchè N non resta vuoto.

Una volta riempito il primo cluster C per trovare il cluster successivo il DBSCAN seleziona randomicamente un oggetto non ancora visitato dei restanti e ripete lo stesso procedimento. Il processo di clustering continua fino a quando tutti gli oggetti non sono stati visitati almeno una volta.

1.8 Analisi di dati testuali- Text Mining

Il text mining è l' insieme di tecniche utilizzate per estrarre conoscenza e informazioni da un documento di testuale. Si occupa della ricerca, dell'analisi e della classificazione delle informazioni contenute nei documenti, della ricerca di argomenti ricorrenti al loro interno. Queste tecniche considerano solo le parti rilevanti di un documento ed eliminano quelle non rilevanti (come le stop words). I dati testuali sono di solito strutturati o semi-strutturati (ad esempio quando sono organizzati in paragrafi o sezioni). Quando si ha a che fare con dati testuali come prima cosa essi vanno processati per ottenere una forma strutturata che possa essere analizzata

Document Processing La fase di Processing consiste nei seguenti passaggi:

1. Splitting: Il testo viene diviso in unità più piccole come frasi o paragrafi.
2. Tokenizzazione: Si divide il testo in frasi o in vettori di parole.
3. Normalizzazione: Si convertono le parole tutte in minuscolo.
4. Rimozione stopwords: Si rimuovono le stop-words, ovvero le parole più comuni della lingua e allo stesso tempo non utili per l'analisi, come articoli e preposizioni.
5. Stemming: Si riducono le parole alla loro radice.

Rappresentazioni dei testi Una volta processati i testi vanno trasformati in un formato adatto a svolgere le analisi successive. Esistono diversi tipi di trasformazioni che agiscono sui testi per ottenere una loro rappresentazione numerica che sia più facilmente analizzabile. Riportiamo alcune delle trasformazioni più utilizzate:

1. Bag of words: Questo tipo di trasformazione trasforma ogni documento testuale in un vettore. Ogni posizione di un vettore corrisponde ad una diversa parola e conterrà il numero di volte con cui quella parola compare nel documento.

	team	coach	play	ball	score	game	win	lost	timeout	season
Document 1	3	0	5	0	2	6	0	2	0	2
Document 2	0	7	0	2	1	0	0	3	0	0
Document 3	0	1	0	0	1	2	2	0	3	0

Figura 1.7: Esempio di trasformazione dei documenti tramite l'algoritmo Bag of Words

2. Tf-idf: La Term Frequence-Inverse Document Frequency determina l'importanza di una parola in base alla frequenza con cui essa appare in un insieme di documenti. In generale se una parola appare spesso in un documento essa dovrà avere un peso elevato. Tuttavia se al contempo essa compare spesso nell'intero corpus di documenti andrà penalizzata in quanto parola comune.
   t = termine
   d = documento
   N= numero di documenti nel corpus

$$tf(t,d) = \frac{\text{numero di volte t compare in d}}{\text{numero di termini in d}} \tag{1.6}$$

$$idf(t) = \log(\frac{N}{\text{numero di documenti in cui t è presente+1}}) \tag{1.7}$$

$$tf - idf(t,d) = tf(t,d) \cdot idf(t) \tag{1.8}$$

   Anche in questo caso ogni documento è trasformato in un vettore numerico ed ogni posizione del vettore corrisponde ad una parola. Tuttavia in questo caso la posizione sarà occupata dal valore tf-idf di quel termine nel documento considerato.

3. Reti neurali: Altre trasformazioni sono operate per mezzo di reti neurali. Ad esempio la rete neurale del Word2vec trasforma ogni parola in un vettore sulla

base del contesto in cui esse si trovano.

Una volta ottenuta la trasformazione dei testi, le nuove rappresentazioni dei dati possono essere utilizzate per diversi scopi di analisi: il clustering, la classificazione, il riconoscimento di argomenti trattati, la presenza di associazioni tra parole. Uno degli scopi del text mining è l'analisi del sentimento, ovvero la costruzione di sistemi per l'identificazione ed estrazione di opinioni delle opinioni che il testo vuole trasmettere.

1.8.1 Sentiment Analysis

La ricerca di tecniche per l'analisi del sentimento rispondono all'esigenza di catturare l'opinione pubblica riguardo ad eventi politici, sociali, campagne di marketing, preferenze e gusti. Lo sviluppo di queste tecniche non è semplice in quanto richiede strumenti in grado di riconoscere irregolarità, distinguere concetti impliciti ed espliciti, conoscere regole semantiche e grammaticali. Una delle tecniche essenziali dell'analisi semantica è la classificazione della polarità, ovvero la classificazione di parti del testo come appartenenti ad uno di due sentimenti contrapposti (positivo o negativo). Alcuni sistemi riescono ad identificare diversi gradi di positività o negatività ad un testo. Alcune tecniche utilizzano le parti del discorso (part of speech POS) ovvero delle sigle (tag) associate ad ogni parola che indicano la categoria a cui la parola appartiene, come aggettivi, avverbi, nomi. Ad esempio si possono identificare aggettivi che sono un buon indicatore del sentimento nascosto dietro ad un testo. In alcuni approcci si utilizzano dei dizionari semantici, ovvero delle liste di parole associate ad un determinato sentimento. [4]

Capitolo 2

Identificazione del Concept Drift

Gli algoritmi di machine learning spesso operano in ambienti dinamici che cambiano inaspettatamente. Quindi una proprietà che si desidera che questi algoritmi abbiano è la capacità di adattarsi rapidamente ad ambienti diversi incorporando nuovi dati. Nel caso di algoritmi di predizione ad esempio, se i dati non sono strettamente stazionari (come succede nella maggior parte delle applicazioni reali), il concetto oggetto della predizione può cambiare nel tempo. Questo cambiamento viene definito **concept drift**. Gli Algoritmi di Apprendimento Adattivi sono degli algoritmi in grado di adattarsi al processo di evoluzione dei dati nel tempo. [5]

2.1 Apprendimento online ed offline

Esistono due tipi di modalità di apprendimento che possono essere utilizzate da un algoritmo: online ed offline. Nell'apprendimento **offline** l'intero insieme dei dati di train deve essere disponibile nel momento in cui il modello viene addestrato. Solo una volta addestrato infatti, il modello può essere utilizzato per le predizioni. Al contrario, gli algoritmi online processano i dati sequenzialmente. Producono un modello e lo utilizzano senza avere disponibile l'intero insieme di dati di training fin dall'inizio.
I modelli predittivi che operano in queste condizioni hanno bisogni di meccanismi per riconoscere ed adattarsi all'evoluzione dei dati nel tempo, altrimenti la loro accuratezza degrada. Quindi al passare del tempo, i modelli decisionali potrebbero aver bisogno di essere aggiornati prendendo in considerazione i nuovi dati oppure di essere completamente rimpiazzati da nuovi modelli. Quindi i modelli predittivi devono sapere:

1. riconoscere il drift e adattarsi ad esso

2. distinguere il drift dal rumore

[6]

2.2 Concept Drift: una definizione formale

Il Concept Drift è il fenomeno in cui le proprietà statistiche di un dominio di dati cambiano nel tempo in modo arbitrario.
Dato un periodo di tempo $[0, t]$ indichiamo con $S_{0,t} = \{d_0, ..., d_t\}$ un insieme di dati raccolti in questo intervallo, dove $d_i = (X_i, y_i)$ rappresenta una singola osservazione, X_i è il vettore delle caratteristiche e y_i l'etichetta di classe. Indichiamo con $F_{0,t}(X, y)$ la distribuzione di $S_{0,t}$.
Si dice allora che si ha concept drift al tempo t+1 se

$$F_{0,t}(X, y) \neq F_{t+1,\infty}(X, y) \tag{2.1}$$

o altrimenti se $\exists$ t s.t.:

$$P_t(X, y) \neq P_{t+1}(X, y) \tag{2.2}$$

Secondo quest'ultima definizione il Concept Drift può essere definito come un cambiamento della probabilità congiunta di (X,y) al tempo t.
Dato che la probabilità congiunta può essere scritta come:

$$P_t(X, y) = P_t(X) \cdot P_t(y|X) \tag{2.3}$$

si può avere drift in tre diversi casi:

1. $P_t(X) \neq P_{t+1}(X)$ mentre $P_t y|(X) = P_{t+1}(y|X)$. Questo tipo di drift si definisce **drift virtuale** perchè non determina un cambiamento nei confini tra le classi.

2. $P_t(y|(X) \neq P_{t+1}(y|X)$ mentre $P_t(X) = P_{t+1}(X)$. Questo tipo di drift prende il nome di **drift reale** perchè determina un cambiamento nei bordi delle classi.

3. sia $P_t(y|(X) \neq P_{t+1}(y|X)$, sia $P_t(X) \neq P_{t+1}(X)$

La seguente figura mostra i vari tipi di drift. Notiamo che solo il drift reale modifica i bordi delle classi con conseguenza che il modello predittivo precedentemente trainato risulta obsoleto. Tuttavia quando il drift virtuale compare insieme al reale i confini tra le classi si modificano.

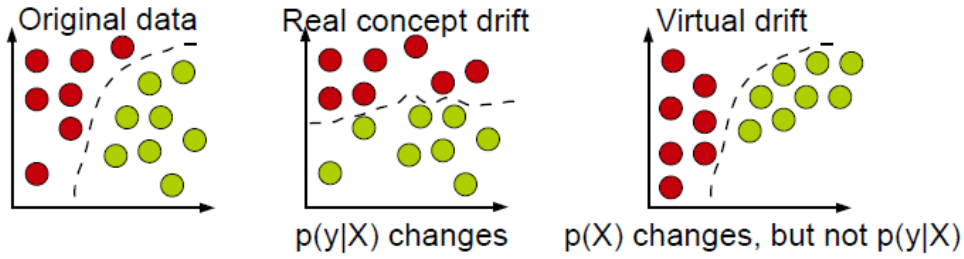


Figura 2.1: Tipi di drift:i cerchi rappresentano le istanze dei dati, i colori le diverse classi

[7] [6]

2.3 Tipi di Drift

Il Concept Drift può avvenire in diversi modi:

1. Improvvisamente: in questo caso la distribuzione dei dati cambia del tutto dal tempo t al tempo t+1
2. Gradualmente: in questo caso si hanno tanti drift intermedi tra un tempo t e un tempo t+n.
3. Può introdurre delle nuove classi o distribuzioni dei dati diverse o riportare a delle configurazioni o classi già viste in precedenza (reoccurring concepts)

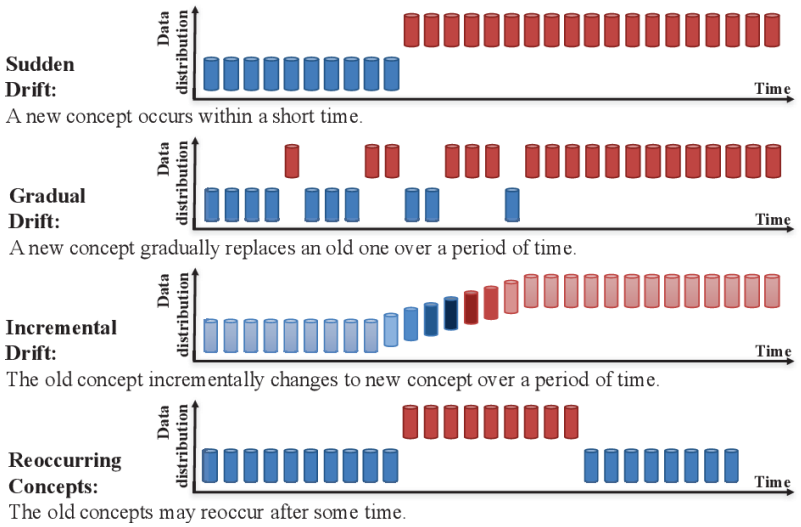


Figura 2.2: Tipi di drift:i cerchi rappresentano le istanze dei dati, i colori le diverse classi

Una delle difficoltà degli algoritmi di apprendimento adattivi è non confondere il drift con rumore o outliers. La maggior parte delle tecniche di drift detection si specializzano in una tipologia di drift. Molti di questi assumono un drift improvviso, ma nella realtà vari tipi di drift spesso si presentano insieme.
La ricerca per l'elaborazione di tecniche di apprendimento in presenza di drift si focalizza su tre fasi:

1. **Identificazione del Drift**: se esso avviene o meno
2. **Comprensione del Drift**: quando avviene e in che modalità
3. **Adattamento al Drift**: reazione alla presenza del drift

[7]

2.4 Identificazione del drift

Questa fase coinvolge le tecniche ed i meccanismi che vengono utilizzati per caratterizzare e quantificare il drift. Il procedimento che viene adottato solitamente è il seguente:

1. Recupero dei dati: I dati disponibili vengono suddivisi in blocchi. Infatti un singolo dato non è sufficiente a determinare la realizzazione del drift.
2. Modellizzazione dei dati: si estraggono le caratteristiche principali dei dati, ovvero quelle che hanno un maggiore impatto nel drift.
3. Calcoli statistici: i drift viene quantificato mediante il calcolo di indici o di metriche come ad esempio misure di similarità o di distanza. Questa fase è considerata la più difficoltosa nel riconoscimento del drift, in quanto non esiste un modo univoco per misurare la dissimilarità.
4. Test d'Ipotesi: utilizza test statistici per valutare i valori delle metriche utilizzate nella fase 3.

2.4.1 Alcuni algoritmi per il riconoscimento del drift

Riportiamo alcuni degli algoritmi più utilizzati per il riconoscimento del drift.

Drift Detection Method (DDM)

è un algoritmo che si basa sull'analisi dell'error rate del classificatore per determinare la presenza del drift. Nell'implementazione della fase 1 (il recuperodei dati) utilizza una finestra il cui punto iniziale è fisso mentre il punto finale varia inglobando i nuovi dati.Con l'inserimento dei dati nuovi il DDM calcola l'aumento dell'error rate. Poi, con un test d'ipotesi, si valuta se i due error rate sono significativamente diversi. In tal caso il DDM inizia a costruire un nuovo modello predittivo.

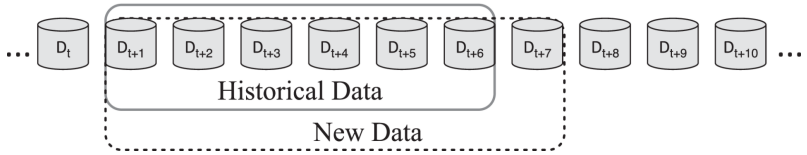


Figura 2.3: Landmark window: il punto iniziale è fisso, il punto finale variabile

Algoritmi basati sulla Distribuzione dei dati

Questi algoritmi utilizzano una funzione di distanza/ metrica per quantificare la dissimilarità tra la distribuzione dei dati storici e dei nuovi dati. Se si prova, con un test d'ipotesi, che la dissimilarità registrata è significativamente diversa il sistema attiverà un processo di aggiornamento del modello predittivo. Nell'implementazione della fase 1 (il recupero dei dati) si utilizzano due finestre di dati: La prima contiene i dati storici e resta fissa, la seconda scorre nei dati nuovi.

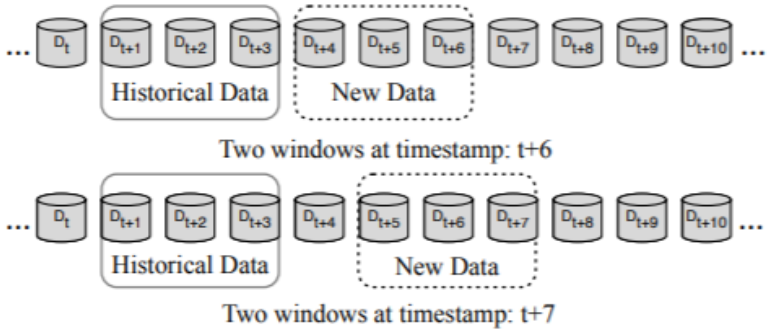


Figura 2.4: Landmark window: il punto iniziale è fisso, il punto finale variabile

La nozione di distanza tra le due distribuzioni è la variazione totale:

$$TV(P_1, P_2) = 2sup_{E \in \epsilon} |P_1(E) - P_2(E)| \tag{2.4}$$

oppure, se si conoscono le funzioni di densità delle due distribuzioni f_1 ed f_2

$$TV(f_1, f_2) = \int f_1(x) - f_2(x)\, dx \tag{2.5}$$

[7]

2.5 Comprensione del Drift

In questa fase si vuole rispondere alle domande: "Quando avviene il drift?", "In che modalità?", "Dove, in quali regioni dello spazio dei dati?".

2.5.1 L'istante di tempo a cui avviene il drift

Per rispondere alla prima domanda bisogna identificare l'istante di tempo t in cui si ha che:

$$P_t(X, y) \neq P_{t+1}(X, y) \tag{2.6}$$

In alcuni algoritmi viene inviato un segnale per indicare se ad un determinato istante di tempo t il drift è avvenuto o meno. Con questo segnale si avvia anche il procedimento di adattamento al drift del metodo predittivo. La corretta identificazione dell'istante di tempo in cui il drift avviene è indispensabile per il processo di adattamento. Infatti ritardi o falsi allarmi non consentirebbero all'algoritmo predittivo di apprendere le nuove strutture caratterizzanti i nuovi dati. In realtà,nella maggior parte degli algoritmi utilizzati il riconoscimento dell'istante di tempo in cui avviene il drift è quasi sempre ritardato, in quanto è necessaria una quantità minima di dati nuovi per determinare la realizzazione del drift.

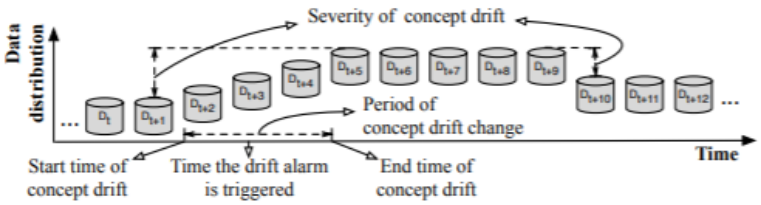


Figura 2.5:

2.5.2 La Severità: come avviene il drift

Nel Concept Drift la Severità indica la modalità con cui si quantifica la similarità tra i dati nuovi e storici. Formalmente essa può essere rappresentata:

$$\Delta = \delta(P_t(X, y), P_{t+1}(X, y)) \tag{2.7}$$

dove δ è una funzione che misura la dissimilarità tra due distribuzioni di dati, mentre Δ è un valore non negativo che indica la Severità del drift. Più Δ è grande maggiore sarà la dissimilarità tra le due distribuzioni.
La Severità può essere utilizzata come linea guida per determinare una strategia di adattamento al drift. Ad esempio nel caso di un modello di classificazione, un valore di Severità basso può indicare che nei nuovi dati i confini delle classi non si sono spostati di molto.
[7]

2.5.3 Le regioni in cui avviene il Drift

Sono le regioni di conflitto tra le due distribuzioni ovvero dove $P_t(X, y)$ e $P_{t+1}(X, y)$ sono significativamente diverse. [7] [8]

2.6 Adattamento al drift

La modalità più diretta di adattamento al drift è quella di riaddestrare il modello con gli ultimi dati e rimpiazzare il vecchio con quello nuovo. Adottando questa strategia bisogna stabilire un metodo per capire quando è necessario effettuare nuovamente l'addestramento del modello. Una delle strategie più utilizzate utilizza due modelli, uno stabile e uno reattivo. Se quello stabile classifica spesso i dati erroneamente mentre quello reattivo correttamente, allora viene individuato il drift e il classificatore stabile è rimpiazzato dal reattivo.

Nel caso in cui il drift porti a delle configurazioni di dati già viste in precedenza, potrebbe essere conveniente utilizzare dei modelli addestrati in tempi anteriori e poi abbandonati perchè risultati obsoleti. A questo scopo nascono degli algoritmi di adattamento al drift che contengono al loro interno di modelli predittivi diversi. Essi considerano un gruppo di classificatori che hanno diversi parametri e che sono stati trainati su dati diversi, e li utilizzano a seconda delle necessità. [7]

Capitolo 3

Metodologia

3.1 Strumenti Utilizzati

Iniziamo con illustrare gli strumenti che vengono sfruttati in questo metodo.

3.1.1 Word Embeddings

In generale con Embedding si intende una mappatura di una variabile discreta (categorica) in un vettore di valori numerici continui di lunghezza fissata. Quando un Embedding è generato tramite una rete neurale si parla di Neural Network Embedding. Word Embeddings sono dei particolari Neural Network Embeddings che permettono di rappresentare parole e frasi di file testuali attraverso vettori di numeri reali.
Si tratta di una funzione:

$$\begin{aligned} W &\to \mathbb{R}^D \\ w &\to \vec{v} \end{aligned} \tag{3.1}$$

che mappa una parola w del dizionario W in un vettore v in uno spazio di dimensione D.

3.1.2 Word to Vectors

Il Word to Vectors è una rete neurale a due strati che processa i documenti testuali mappando ogni parola in un unico vettore. Prende come input un corpus di documenti e dà in output un insieme di vettori. I vettori delle parole saranno disposti nello spazio in modo che parole con significati simili avranno dei vettori vicini tra di loro.

Data una finestra di dimensione 2D, l'addestramento di questa rete neurale può essere effettuato in due modi: il Continuous Bag-of-Words (CBOW) e lo Skip-Gram (SG).

Modello CBOW

La parola w_t predetta al centro della finestra è la parola con la probabilità più alta di comparire dopo le parole w_{t-1}, ... , w_{t-D} e prima delle parole w_{t+1}, ... , w_{t+D}. Il Word Embedding viene dunque addestrato attraverso l'ottimizzazione di una funzione obbiettivo che calcola la probabilità della parola centrale corrente date le parole contesto che la circondano.
Lo strato di input dello schema per una finestra di dimensione 2D è composto dai vettori one hot $x_{t-D}, ..., x_{t-1}, x_{t+1}, ..., x_{t+D}$ relativi alle parole contesto. Lo strato nascosto è composto da un vettore h di dimensione d. Lo strato di output è composto dal vettore y della parola centrale oggetto della predizione w_t.
Sia d la dimensione dei vettori e V la dimensione del dizionario. Lo strato di input è collegato allo strato nascosto tramite la matrice $W \in \mathbb{R}^{V \times d}$ composta dai vettori relativi alle parole contesto. Lo strato nascosto è collegato allo strato di output tramite la matrice $W^{'} \in \mathbb{R}^{d \times V}$, composta dai vettori relativi alle parole centrali. Data la matrice dei vettori delle parole contesto W, il valore del layer nascosto h viene calcolato come:

$$h = \frac{1}{2D} W \cdot \sum_{-D<j<D} x_{t-j} \tag{3.2}$$

Il valore u_i relativo alla i-esima parola u_i viene poi calcolati nel modo seguente:

$$u_i = (v_i')^T \cdot h \tag{3.3}$$

dove v_i' è l'i-esima colonna della matrice $W^{'}$.
Per la parola w_i l' output y_i viene calcolato tramite la funzione di Softmax:

$$y_i = P(w_i | w_{i-D}, ..., w_{i-1}, w_{i+1}, ..., w_{i+D}) = \frac{exp(u_i)}{\sum_j exp(u_j)} \tag{3.4}$$

L'addestramento dells rete neurale

L' addestramento dei pesi delle matrici W e $W^{'}$ avviene in un procedimento di tre fasi:

1. Si inizializzano W e W' con valori casuali, presi nell'intervallo [-1, 1].

2. In modo iterativo vengono forniti al modello le parole centrali e quelle di contesto e si calcolano le predizioni delle parole centrali.

3. Si aggiornano le matrici W e W' in modo da minimizzare l'errore tra la predizione e il valore reale. A tale scopo la funzione obbiettivo da massimizzare è la probabilità condizionata della parola di output w_o date le parole contesto in input:

$$\begin{aligned} E = -\log P(w_o|w_{t-D}, ..., w_{t-1}, w_{t+1}, ..., w_{t+D}) = \\ -u_{w_o} - \log \sum_{j=1}^{V} exp(u_j) = \\ = (v'_{w_o})^T \cdot h - \log \sum_{j=1}^{V} exp((v'_j)^T \cdot h) \end{aligned} \tag{3.5}$$

[9]

Modello Skip-Gram

In questo caso le parole w_{t-D}, ... , w_{t-1} e w_{t+1}, ... , w_{t+D} predette sono quelle che massimizzano la probabilità di trovarsi attorno a w_t. Il Word Embedding viene addestrato attraverso l'ottimizzazione di una funzione obbiettivo che calcola la probabilità delle parole contesto w_{t-1}, ... , w_{t-D} e w_{t+1}, ... , w_{t+D} in una certa finestra di dimensione 2D data la parola centrale corrente w_t.
L' input del modello è il vettore one-hot x_t relativo alla parola centrale corrente w_t. L'output è costituito dai vettori y_{t-D}, ... ,y_{t-1},y_{t+1},..., y_{t+D} relativi alle parole contesto w_{t-D}, ... ,w_{t-1},w_{t+1}, ... , w_{t-D}. Anche in questo caso la matrice W di dimensione $V \times d$ è la matrice che collega lo strato di input allo strato nascosto ed ha per righe i vettori che corrispondono alle parole centrali, mentre la matrice W' di dimensione $d \times V$ contiene i pesi relativi alle parole contesto.

In questo caso il valore del layer nascosto h, data la parola centrale w_j viene calcolato come:

$$\begin{aligned} h &= x_j \\ W &= v_j \end{aligned} \tag{3.6}$$

essendo x_j un vettori one-hot. Di conseguenza data la parola centrale w_j nello strato nascosto per la parola di contesto c-esima si calcola:

$$u_{c,j} = (v'_j)^T \cdot h \tag{3.7}$$

Dunque datala parola centrale w_j l' output $y_{c,j}$ relativo alla c-esima parola di contorno sarà dato da:

$$y_{c,j} = P(w_{c,j}|w_j) = \frac{exp(u_{c,j})}{\sum_j^V exp(u_j)} \tag{3.8}$$

Anche in questo caso la fase di addestramento dei pesi delle matrici W e W' avviene in un procedimento di tre fasi:

1. Si inizializzano W e W' con valori casuali, presi nell'intervallo [-1, 1].

2. In modo iterativo vengono forniti al modello le parole centrali con quelle di contesto e si calcolano le predizioni delle parole di contesto.

3. Si aggiornano le matrici W e W' in modo da massimizzare la funzione obiettivo:

$$\begin{aligned} E &= -\log P(w_{o,t-D}, ..., w_{o,t-1}, w_{o,t+1}, ..., w_{o,t+D}|w_j) \\ &= -\log \prod_{j=1}^{V} \frac{exp(u_{c,j})}{\sum_j^V exp(u_j)} \\ &= -\sum_{j=1}^{V} u_{c,j} + 2D \sum_{j=1}^{V} u_j \end{aligned} \tag{3.9}$$

4.

[10]

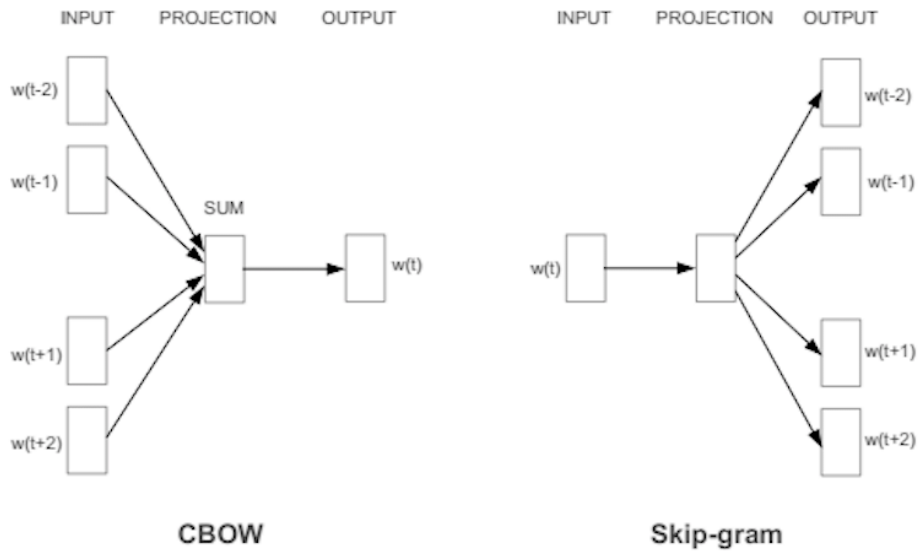


Figura 3.1:

3.1.3 Tecniche di riduzione della dimensione dei dati

I metodi per la riduzione della dimensionalità dei dati convertono dati di con molte dimensioni $X = x_1, x_2, ..., x_n$ in dati di due o tre dimensioni $Y = y_1, y_2, ..., y_n$ che di conseguenza sono più facilmente visualizzabili. Lo scopo della riduzione della dimensionalità è ottenere una rappresentazione dei dati a poche dimensioni preservando però il più possibile l'informatività e la significatività dei dati di partenza. In particolare in questa analisi verranno utilizzati i metodi della Principal Component Analysis (PCA) e del t-distributed Stochastic Neighbor Embedding (t-SNE)

Principal Component Analysis

Con Principal Component Analysis si intende il processo tramite cui si ottiene una riduzione della dimensionalità dello spazio dei dati proiettandoli lungo le componenti principali dei medesimi. Le componenti principali sono le direzioni lungo le quali i dati presentano la massima variabilitàe che al contempo definiscono sottospazi che minimizzano la distanza con i dati di partenza.
Supponiamo di avere a disposizione un dataset di n osservazioni ognuna delle quali ha p features (caratteristiche).

$$x_i = (x_{1i}, ..., xpi) \tag{3.10}$$

Indichiamo con

$$X_1, ..., X_p \tag{3.11}$$

le caratteristiche dei dati.
La Prima Componente principale di un insieme di features $X_1, ..., X_p$ è una combinazione lineare normalizzata delle features:

$$\begin{aligned} Z_1 &= \phi_{11}X_1, ..., \phi_{p1}X_p \\ &dove \sum_j \phi_{j1}^2 = 1 \end{aligned} \tag{3.12}$$

Il vettore dei pesi

$$\Phi_1 = (\phi_{11}, ..., \phi_{p1}) \tag{3.13}$$

viene scelto in modo da massimizzare la varianza dei dati, quindi deve risolvere il problema di otimizzazione:

$$\begin{aligned} \max_x \quad & \frac{1}{n}\sum_{i=1}^{n}(\sum_{j=1}^{p}\phi_{j1}x_{ij})^2 \\ \text{t.c.} \quad & \sum_{j=1}^{n}\phi_{j1}^2 = 1 \end{aligned}$$

Ottenuti i pesi ϕ_{j1} la proiezione di $x_i = x_{1i}, ..., xpi$ su Z_1 sarà data da:
$z_{i1} = \phi_{11}x_{i1}, ..., \phi_{p1}xip$

Una volta ottenuta la prima componente principale Z_1 possiamo determinare la seconda Z_2:

$$Z_2 = \phi_{12}X_1, ..., \phi_{p2}X_p \quad \text{dove} \quad \sum_j \phi_{j2}^2 = 1 \tag{3.14}$$

Il vettore dei pesi $\Phi_2 = (\phi_{11}, ..., \phi_{p1})$ dovrà massimizzare la varianza residua e al contempo essere perpendicolare a Φ_1:

$$\begin{aligned} \max_x \quad & \frac{1}{n}\sum_{i=1}^{n}(\sum_{j=1}^{p}\phi_{j2}(x_{ij} - z_{i1})^2 \\ \text{t.c.} \quad & \sum_{j=1}^{n}\phi_{j1}^2 = 1 \\ \text{e} \quad & < \Phi_1, \Phi_2 >= 0 \end{aligned}$$

La proiezione di x_i su Z_2 saràdata da $z_{i1} = \phi_{12}x_{i1}, ..., \phi_{p2}xip$

Reiterando questo procedimenti si può trovare il numero desiderato di componenti principali.
In generale al passo m si trova la componente principale:

$$Z_m = \phi_{1m}X_1, ..., \phi_{pm}X_p$$

dove $\Phi_m = (\phi_{1m}, ..., \phi_{pm})$ è il vettore dei pesi che viene determinato risolvendo il problema ottimizzazione:

$$\begin{aligned} \max_x \quad & \frac{1}{n}\sum_{i=1}^{n}(\sum_{j=1}^{p}\phi_j(x_{ij} - \sum_{l=1}^{m-1} z_{il})^2 \\ \text{t.c.} \quad & \sum_{j=1}^{n}\phi_{jm}^2 = 1 \\ \text{e} \quad & < \Phi_m, \Phi_1 >= 0, < \Phi_m, \Phi_2 >= 0, ... < \Phi_m, \Phi_{m-1} >= 0 \end{aligned}$$

t-SNE

La t-SNE è una tecnica di riduzione della dimensionalità dei dati non lineare. In generale i metodi non lineari sono maggiormente in grado di preservare la vicinanza dei dati iniziali a molte dimensioni. In particolare la t-SNE è in grado di catturare la struttura locale dei dati iniziali mostrando al contempo anche la presenza di alcune strutture globali, come la presenza di clusters.
In generale la t-SNE opera in modo tale da calcolare una misura di similarità tra le coppie di dati sia nello spazio a molte dimensioni che in quello a poche dimensioni e cerca di ottimizzare queste similarità minimizzando una funzione di costo. L'algoritmo si compone di tre fasi principali:

1. Si misura la similarità tra i punti nello spazio dei dati iniziali. In particolare per ogni dato x_i viene centrata una distribuzione Gaussiana e si misura la densità di punti x_j che cadono dentro la distribuzione. Come similarità tra i punti x_j e x_i si considera la funzione di densità di probabilità p_{ji} di x_j nella distribuzione Gaussiana centrata in x_i, ovvero:

$$p_{ji} = \frac{exp(-||x_j - x_i||^2/2\sigma^2)}{\sum_{i \neq j} exp(-||x_j - x_i||^2/2\sigma_i^2)} \tag{3.15}$$

   dove σ_i è la varianza della Gaussiana centrata in x_i.

2. Nello spazio a dimensioni inferiori la similarità tra i due punti y_j e y_i sarà calcolata sulla base di un a distribuzione t-Student con un grado di libertà:

$$q_{ij} = \frac{(1 + ||y_i - y_j||^{-}1)}{\sum_{i \neq l}(1 + ||y_i - y_l||^{-}1)} \tag{3.16}$$

La distribuzione t-Student è caratterizzata da code più pesanti rispetto alla distribuzione normale e ciò consente di modellare meglio la lontananza tra punti preservando tuttavia la vicinanza.

3. Nella creazione dello spazio di dimensione inferiore si vuole che le probabilità q_{ij} riflettano le p_{ij}. A tale scopo si misura la differenza tra le distribuzioni di probabilità nei due spazi tramite la divergenza di Kullback-Liebler:

$$C = \sum_i \sum_j p_{ij} \log p_{ij} - p_{ij} \log q_{ij} \tag{3.17}$$

Si utilizza il metodo della discesa del gradiente per minimizzare questa funzione di costo.

[11]

3.1.4 Tecniche di Outliers Detection - Isolation Forest

L'Isolation Forest è un metodo per il riconoscimento degli outliers il cui obiettivo è isolare le anomalie dei dati. Si definiscono anomalie gli elementi di un dataset che sono molto diversi dalla maggioranza, e che pertanto possono essere frutto di errori o di contaminazione.
L'Isolation Forest sfrutta due caratteristiche quantitative fondamentali delle anomalie, cioè:
1) consistono in una piccola parte dei dati
2) presentano valori che si discostano molto dal resto dei dati.
Queste due caratteristiche possono essere riassunte nell'espressione 'le anomalie sono poche e diversè. Per isolare le anomalie questo metodo utlizza la struttura degli alberi: pone vicino alla radice i dati che si discostano di più, e il resto via via più in profondità. Alcuni vantaggi di questo metodo sono:

- Per riconoscere le anomalie non viene utilizzata una funzione di distanza o di densità e ciò riduce il costo compuzionale
- ha una complessità lineare.

L'Isolation Forest sfrutta il principio secondo cui le anomalie essendo 'poche e diversè sono più facilmente isolabili dai resto dei dati. I dati vengono partizionati randomicamente per mezzo di random trees fino a quando tutti i dati non saranno stati isolati. I dati che hanno dei valori che si discostano di più vengono isolati con un numero più basso di partizionamenti rispetto al resto dei dati. Pertanto quando una foresta di random trees riesce ad isolare un gruppo di dati in pochi passaggi, questi saranno con molta probabilità delle anomalie.

Definition: Isolation Tree
Consideriamo un nodo T di un albero di isolamento binario. Consideriamo un attributo q e un valore soglia p tali che la condizione $q < p$ divide i punti in due rami T_l e T_r. Dato un insieme di n dati $X = x_1, ..., x_n$ per costruire un albero di isolamento i dati vengono divisi in modo ricorsivo selezionando randomicamente un attributo q e in valore soglia p fin quando o l'albero raggiunge un'altezza limite preimpostata oppure tutti i dati saranno separati tra loro in una diversa foglia dell'albero (a parte quelli che presentano valori uguali che occuperanno lo stesso ramo). Assumendo che tutti i dati siano distinti tra loro, ogni dato verrà isolato in un nodo diverso e il numero totale di nodi sarà $2n - 1$. Pertanto i requisiti di memoria sono limitati e crescono linearmente con n.

I dati vengono poi ordinati in base al numero crescente di split che sono stati necessari per isolarli e vengono classificati come anomalie quelli che si trovano in cima alla lista.

Definizione: Lunghezza del cammino
La lunghezza di un cammino $h(x)$ di un elemento x si misura con il numero minimo di archi necessari per raggiungere x dalla radice dell'albero.

Definizione: Indice di Anomalia
è necessario un indice che dato un elemento x ci dica se esso costituisce un'anomalia o meno. Si stima la lunghezza media dei cammini con la funzione:

$$c(n) = 2H(n-1) - (2(n-1)/n) \tag{3.18}$$

dove H (i) = ln(i) + 0.5772156649 è il numero armonico.

Definiamo l'indice di anomalia di un elemento x come:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{3.19}$$

Dove $E(h(x))$ è la media di$h(x)$ nei vari alberi.
Se $s \approx 1$ x è classificato come anomalia. [12]

3.1.5 Tecniche di Outliers Detection - Elliptic Envelope

L'Elliptic Evelope è un metodo di outliers detection in un dataset con distribuzione Gaussiana. Esso implementa l'algoritmo Fast-MCD (Rousseeuw e Driessen[13]) Supponendo che i dati considerati siano distribuiti secondo una normale multivariata, questo algoritmo vuole determinare la madia e la matrice di varianze e covarianze di questa distribuzione. A tale scopo cerca le h osservazioni la cui matrice delle covarianze ha il determinante più basso e calcola la media e la varianza di quest'ultime. L'algoritmo su cui si basa questo metodo sfrutta il seguente teorema e il seguente corollario:

Theorem 1. Consideriamo n osservazioni $\{X_1, ..., X_n\}$ provenienti da una distribuzione normale p-variata. Siano $H_1 \subset \{1,...,n\}, |H_1| = h$.
Si definisce

$$T_1 = \frac{1}{n}\sum\nolimits_{i \subset H_1} x_i \tag{3.20}$$

e

$$S_1 = \frac{1}{n}\sum\nolimits_{i \subset H_1} (x_i - T_1)(x_i - T_1)' \tag{3.21}$$

Se det(S_1) $\neq 0$ definiamo le distanze relative:

$$d_1(i) = (x_i - T_1)' S^{-1} (x_i - T_1) i \in (1, ..., n) \tag{3.22}$$

Adesso prendiamo una permutazione che ordini le distanze $d_1(i)$ in ordine crescente. Prendiamo H_2 tale che:

$$\{d_1(i); i \in H\_2\} = \{(d_1)_{(1:n)}, ..., (d\_1)_{(h:n)}\} \tag{3.23}$$

dove

$$\{(d_1)_{(1:n)} \leq (d_1)_{(2:n)} \leq ... \leq (d\_1)_{(n:n)}\} \tag{3.24}$$

e calcoliamo T_2 e S_2 sulla base di H_2
Allora $det(S_2) \leq$det(S_1)

Sulla base di questo Teorema definiamo la struttura principale di questo algoritmo, chiamata C-step. Essa consiste nei seguenti passaggi:

1. Dato H_{old} calcola le distanze $d_{old}(i)$ per i = 1,...,n
2. ordina le distanze con la permutazione π in modo che

$$(d_{old})(\pi(1)) \leq (d_{old})(\pi(2)) \leq ... \leq (d_{old})(\pi(n)) \tag{3.25}$$

3. $H_{new} = \pi(1), ..., \pi(h)$
4. $T_{new} = \text{ave}(H_{new})$ and $S_{new} = \text{cov}(H_{new})$

La sequenza $det(S_1) \leq det(S_2) \leq det(S_3)...$ è finita quindi converge.
Corollary 1.1. Il sottinsieme H per cui si ha il minimo determinante (MCD) dei dati X è separato da X/H da un ellissoide.

Infatti tutti gli $x_i \in H$ soddisfano:

$$(x_i - T_1)' S^{-1} (x_i - T_1) \leq m = \{(x_i - T_1)' S^{-1} (x_i - T_1)\}_{(h:n)} \tag{3.26}$$

mentre tutti gli $x_j \notin H$ soddisfano

$$(x_j - T_1)' S^{-1} (x_j - T_1) \geq m \tag{3.27}$$

Consideriamo l'Ellissoide

$$E = \{x \in R^p; (x - T_1)' S^{-1} (x - T_1)\} \leq m \tag{3.28}$$

Allora $H \subset E$ e $X_n/H \subset E^c$.
Si nota che c'è almeno un punto x_i sul bordo di E

Come creare il sottinsieme iniziale H_1
Consideriamo J un sottinsieme con (p+1) elementi presi randomicamente. Calcoliamo $T_0 =$ ave(J) ed $S_0 =$ cov(J). Se $det(S_0) = 0$ aggiungiamo elementi randomici

finchè $det(S_0) > 0$.
calcoliamo le distanze

$$d_0(i)^2 = (x_i - T_0)' S_0^{-1} (x_i - T_0) i \in (1, ..., n) \tag{3.29}$$

e le ordiniamo in

$$d_0(\pi(1)) \leq d_0(\pi(2)) \leq ... d_0(\pi(n)) \tag{3.30}$$

e poniamo $H_1 = \pi(1), ..., \pi(h)$

Quanti C-steps?
Si è visto che bastano pochi C-steps per ottenere una buone soluzioni, pertanto da ogni sottinsime iniziale H_1 si prendono solo due C-steps, si selezionano 10 diversi sottinsiemi H_3 con i determinanti più bassi e per questi continuiamo ad effettuare C-steps fino alla convergenza.

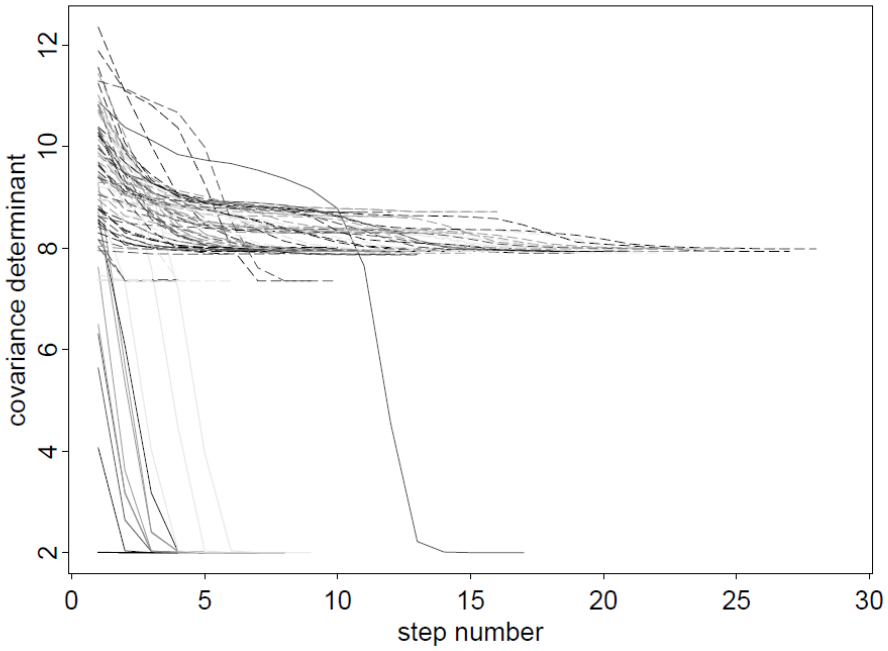


Figura 3.2:

L'algoritmo

1. Si pone inizialmente $h = \frac{(n+p+1)}{2}$

2. Se h = n l'algoritmo calcola la media e la matrice di varianze e covarianze di tutto il dataset e restituisce questi valori.

3. Se p =1, la distribuzione è una normale univariata e la stima del MCD viene realizzata tramite l'algoritmo Russeeuw e Leroy(1987).

4. Se h < n e p > 2:

   - Si ripete 500 volte
     - Si costruisce un sottinsieme H_1 iniziale col metodo precedentemente illustrato partendo da un sottinsieme randomico di (p+1) elementi.
     - Si realizzano 2 C-steps
   - Per i 10 risultati con $det(S_3)$ più basso
     - Si realizzano C-steps fino alla convergenza
   - Si restituisce la la coppia (T,S) con il det(S) più basso

5. Se n > 600:

   - Si divide il dataset randomicamente in 5 sottinsiemi disgiunti di grandezza n_{sub}
   - In ogni sottinsieme si ripete 100 volte:
     - Si costruisce un sottinsieme iniziale H_1 di grandezza $h_{sub} = n_{sub}(h/n)$
     - Si applicano due C-steps
     - Si prendono i migliori 10 risultati (T_{sub}, S_{sub})
   - Si mettono insieme gli insiemi fino ad ottenere un sottinsieme di grandezza massima $n_{merged} = 1500$ con un massimo di 50 soluzioni (T_{sub}, S_{sub})
   - Per ognuna delle 50 soluzioni (T_{sub}, S_{sub}):
     - Si realizzano due C-steps usando n_{merged} e $h_{merged} = n_{merged}(h/n)$
     - Si prendono i 10 migiori risultati (T_{merged}, S_{merged})
   - Si cinsidera l'intero dataset
     - Si realizzano motli C-steps con n e h
     - Si prende il risultato finale migliore (T^*, S^*)

[13] [14]

3.1.6 Tecniche di Clustering - Clustering Agglomerativo

Entriamo nel dettaglio dell'algoritmo di Clustering Agglomerativo. Allo scopo di definire un criterio di agglomerazione di due gruppi deve essere definita una funzione di dissimilarità tra due gruppi. Considerando due cluster G e H possiamo definire diversi tipi di funzioni di dissimilarità(o linkage) tra i due:

1. Single linkage (SL): Considera come dissimilarità tra due gruppi, la minima delle distanze tra gli elementi dei due gruppi.

$$d_{SL}(G,H) = \min_{x_i \in G, x_j \in H} \quad d(x_i, x_j)$$

2. Complete linkage (CL): Considera come dissimilarità tra due gruppi, la massima delle distanze tra gli elementi dei due gruppi.

$$d_{CL}(G,H) = \max_{x_i \in G, x_j \in H} \quad d(x_i, x_j)$$

3. Group average (GA): Considera come dissimilarità tra due gruppi, la media delle distanze tra gli elementi dei due gruppi.

$$d_{GA}(G,H) = \min_{x_i \in G, x_j \in H} \quad \frac{1}{N_G N_H} \sum_{x_i \in G} \sum_{x_j \in H} d(x_i, x_j)$$

4. Ward distance (WD): Considera come dissimilarità tra due gruppi, la media delle distanze al quadrato degli elementi dei due gruppi

$$d_{GA}(G,H) = \quad \frac{1}{N_G N_H} \sum_{x_i \in G} \sum_{x_j \in H} d(x_i, x_j)^2$$

[3]

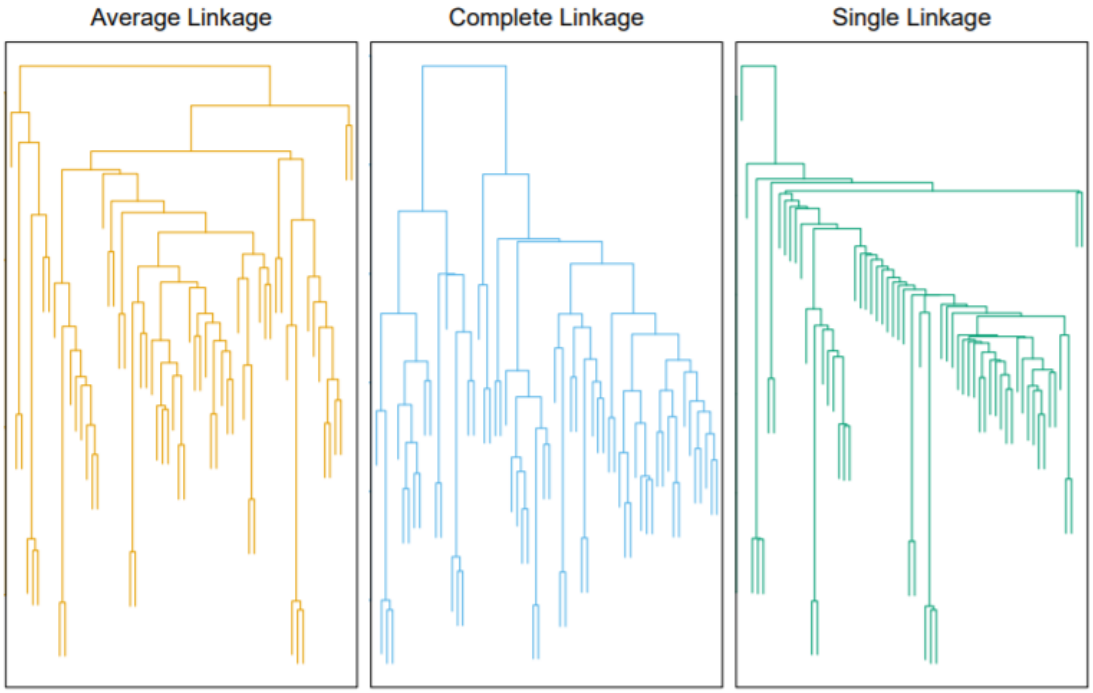


Figura 3.3: Esempi di dendrogrammi ottenuti dallo stesso dataset ma con diversi linkage

3.1.7 Tecniche di valutazione del Clustering: Silhouette

La Silhouette è un indice che misura quanto bene un oggetto è stato assegnato ad un determinato cluster. A tale scopo combina due grandezze: una misura di dissimilarità intra-cluster ($a(x)$) e una misura di dissimilarità inter-cluster ($b(x)$). Data l'osservazione $x_i \in A$, detto N_A il numero di osservazioni nel cluster A:

1. Definiamo:
$$a(x_i) = \frac{1}{N_A} \sum_{x_j \in A} d(x_i, x_j)$$
ovvero la media delle distanze di x_i dagli altri punti del cluster A.

2. Dato il cluster $C \neq A$ avente N_c osservazioni, definiamo la distanza di x_i da C come
$$d(x_i,) = \frac{1}{N_C} \sum_{x_j \in C} d(x_i, x_j)$$
ovvero la media delle distanze di x_i dai punti del cluster C.

3. Definiamo
$$b(x_i) = \min_{C \neq A} \quad d(x_i, C)$$

ovvero la minima tra le distanze di x_i dagli altri clusters.

A questo punto il valore della silhouette di x_i si ottiene combinando $a(x_i)$ e $b(x_i)$ come segue:

$$s(x_i) = \begin{cases} 1 - \frac{a(x_i)}{b(x_i)}, & \text{if } a(x_i) > b(x_i) \\ 0, & \text{if} a(x_i) = b(x_i) \\ \frac{a(x_i)}{b(x_i)} - 1 & \text{if } b(x_i) > a(x_i) \end{cases} \tag{3.31}$$

o più semplicemente:

$$s(x_i) = \frac{b(x_i) - a(x_i)}{\max\{a(x_i), b(x_i)\}} \tag{3.32}$$

quindi $s(x_i) \in [-1, 1] \forall i$.
Per capire il significato di quest'indice analizziamo tre casi:

1. $s(x_i)$ vicino a 1: Vuol dire che la misura di dissimilarità intra-cluster $a(x_i)$ è molto minore di quella inter-cluster $b(x_i)$, e che quindi x_i è stato assegnato al cluster in modo corretto.

2. $s(x_i)$ vicino a 0: Vuol dire che $a(x_i)$ e $b(x_i)$ hanno valori vicini, quindisi potrebbe assegnare x_i all'uno o all'altro gruppo indifferentemente.

3. $s(x_i)$ vicino a -1: Vuol dire che la misura di dissimilarità intra-cluster $a(x_i)$ è molto maggiore di quella inter-cluster $b(x_i)$, e che quindi x_i è molto più vicino ad un cluster diverso rispetto a quello a cui è stato assegnato, e che quindi è stato classificato erroneamente.

Oltre che per valutare se un determinato oggetto è stato assegnato al cluster corretto, la Silhouette può essere utilizzata in modo globale per analizzare il clustering ottenuto facendo la media delle silhouettes di tutti gli elementi del dataset. [15]

3.1.8 Tecniche per l'analisi semantica

In questa metodologia vengono utilizzati alcune tecniche dell'analisi semantica, in particolare per la caratterizzazione di frasi con accezione negativa e l'identificazione delle parti del discorso.

Caratterizzazione frasi negative:

1. Ogni testo viene diviso in frasi. Si considera frase l'insieme di parole che intercorrono tra due segni di punteggiatura successivi.

2. Se all'interno di una frase è presente una negazione allora tutte le parole successive alla negazione verranno marcate col suffisso "_neg".

Questo procedimento ci aiuta nell'identificazione di frasi con accezione negativa. Infatti optare per l'identificazione di singole parole associate ad un significato negativo potrebbe essere fuorviante, non riuscendo a catturare litoti o il significato che la parola ha in un determinato contesto.

Identificazione delle parti del discorso: Per l'identificazione delle parti del discorso (POS) si utilizza il database lessicale WordNet. In questo database le parole sono classificate in categorie, ad esempio nomi, verbi, aggettivi, avverbi... Essi sono poi raggruppati in insiemi di sinonimi (synsets) ognuno dei quali rappresenta un diverso concetto. Questi insiemi sono poi collegati tra loro da relazioni.
Nella metodologia proposta verranno utilizzati i POS-tag, ovvero le sigle associate alle parole che ci consentiranno di selezionare aggettivi, nomi, verbi e avverbi eliminando cosi' le parti dei testi che non ci interessano, come pronomi etc...
[16]

3.2 Fasi del Metodo

Nel contesto del riconoscimento del Concept Drift, con il metodo illustrato si vuole verificare se nei nuovi dati analizzati si presentano classi nuove che non erano presenti nei dati storici . In una prima fase si utilizzano algoritmi di outliers detection per la costruzione di un modello di classificazione one-class che sappia identificare la classe dei dati storici. Nella seconda fase si verifica che i nuovi dati non appartengono alla stessa dei classe dati storici, analizzando l'aumento del numero degli outliers riconosciuti testando il modello ottenuto sui dati nuovi. Infine, tramite algoritmi di clustering, si verifica se si creano nuove classi all'interno degli outliers riconosciuti.
Il metodo si articola nel seguente procedimento:

- Data Cleaning;
- Trasformazione dei dati;

- Visualizzazione dei dati con tecniche di riduzione;
- Creazione del metodo per l' outliers detection;
- Concept Drift Detection mediante il clustering degli outliers;
- Analisi del contenuto dei cluster identificati mediante word cloud.

3.2.1 Data Cleaning

Questa fase consiste nell'insieme di procedimenti che vengono messi in atto per pulire i file testuali da tutti gli elementi che non sono utili allo scopo dell'analisi.

- **Rimozione stopwords e caratteri speciali.**
  Si definiscono stopwords le parole che sono molto frequenti in una lingua e di conseguenza non sono considerate significative. Ad esempio sono stopwords le congiunzioni, gli articoli o le preposizioni. In questo metodo sono state considerate tra le stopwords anche le parole legate al contesto dei dati che vengono trattati. Insieme alle stopwords vengono rimossi anche i caratteri speciali.
- **Lemmatizzazione delle parole**
  Consiste nell'operazione di riportare le varie forme con cui le singole parole ricorrono nel testo analizzato alla loro forma di base. Ad esempio i verbi coniugati vengono riportati alla loro radice. Ad esempio 'loved' vine trasformato in 'lovè
- **Sostituizione degli aggettivi superlativi irregolari**
  Gli aggettivi superlativi irregolari vengono riportati al loro grado base.
- **Selezione di nomi, aggettivi, verbi e avverbi**
  Questa selezione ha come fine l'eliminazione di parole non utili per l'analisi, come i pronomi e gli articoli. Si attua tramite la funzione python pos_tag che associa ad ogni parola il corrispettivo Part of Speech tag del database WordNet. Si selezionano quindi solo le parole avente come tag uno tra: NOUN, ADJ, VERB, ADV.
- **Tokenizzazione**
  Ogni documento viene trasformato nel vettore di parole corrispondenti.

3.2.2 Trasformazione dei dati

Il processo di trasformazione di dati consiste in due fasi. Nella prima verrà creato un Word Embedding, ovvero uno spazio in cui ogni parola è rappresentata da un vettore. Nella seconda si opera la trasformazione di ogni recensione in un vettore.

1. **Creazione del Word Embedding**
   Il WordEmbedding viene creato tramite la rete neurale Word to Vectors. Questo algoritmo crea uno spazio di dimensione prefissata in cui ogni parola è trasformata in un vettore. Parole che si trovano in contesti simili verranno rappresentate da vettori vicini tra di loro. è possibile scaricare degli Embeddings già trainati con il Word to Vectors oppure trainare la rete neurale sui dati a disposizione. In questa analisi si sono effettuati esperimenti sia trainando il modello che scaricando modelli pretrainati. In particolare tra i pretrainati:

   - GoogleNews-vectors: questo modello è trainato su parte del dataset di Google News e contiene i vettori di 3 milioni di parole e frasi. Ogni parola viene trasformata in un vettore di 100 dimensioni.
   - Enwiki: questo modello è trainato su documenti di Wikipedia. Le parole sono lemmatizzate e trasformate in vettori di 100 dimensioni. L'ampiezza della finestra del train è 5.
   - Gigaword: questo modello è stato trainato su un archivio di notizie raccolto da diverse fonti. Le parole sono state lemmatizzate, la finestra del train ha ampiezza pari a due e i vettori hanno 300 dimensioni.

2. **Trasformazione dei documenti**
   La trasformazione dei documenti si basa sulla definizione di due funzioni.
   La funzione **get_most_common**(lista_documenti, n): prende in input una lista di file testuali e restituisce le n parole più frequenti.
   La funzione **weight_calc_improved**(documento, most_common): prende in input un documento e le n parole più frequenti della lista a cui il documento appartiene (restituite da **get_most_common**(lista_documenti, n) e restituisce un vettore secondo il procedimento seguente

3.2.3 Visualizzazione dei dati con tecniche di riduzione

Una volta applicata la trasformazione dei dati essi vengono scomposti tramite tecniche di riduzione della dimensionalità quali la t-SNE e la Principal Component Analysis. Queste tecniche permettono di proiettare i dati in spazi di dimensione

Algoritmo 2 weight_calc_improved(documento,most_common, word_embedding)

```
1: lista_vettori_parole = []
2: for parola in documento do
3:    if parola in most_common then
4:       vett_parola ← word_embedding[parola]
5:       aggiungi vett_parola a lista_parole
6:    end if
7: end for
8: return media(lista)
```

ridotta preservandone le proprietà, pertanto consentono di verificare la presenza di clusters o di pattern nei dati.

3.2.4 Train dell'Algoritmo di Outliers Detection: Elliptic Envelope

Il metodo Elliptic Envelope implementa l'algoritmo FAST-MCD per trovare la media e la matrice delle varianze e convarianze di una distribuzione normale multivariata.

Questo metodo è utilizzato in modo da riconoscere la classe dei dati storici. Pertanto va trainato su una parte di essi e poi testato sulla restante per valutare se esso è in grado di identificare la classe o meno. Nella fase di train determina un numero di outliers pari alla percentuale indicata dal parametro 'contamination'.
Per settare i parametri di questo modello è stata utilizzata una Grid Search che confronta i risultati di diversi parametri e sceglie i migliori.
Come dati di train si sceglie una quantità dei dati iniziali pari a **size_of_train**. Si dividono dunque i dati iniziali in due parti **X_train** e **X_0**. Si traina il modello su **X_train** e per testarne l'efficacia si testa su **X_0**.

3.2.5 Concept Drift Detection Mediante Clustering degli Outliers

In questa fase si vuole studiare il drift dei dati mediante l'analisi dei cluster presenti negli outliers. A tale scopo inizialmente si analizza se, testando il modello di outliers detection sui nuovi dati si verifica un aumento nel numero di outliers riconosciuti. Infatti ciò può essere interpretato come un segnale della possibile presenza di drift. Successivamente si vuole verificare se tra i nuovi outliers riconosciuti si ha la creazione di una nuova classe che inizialmente non era presente.
Per attuare questo procedimento si creano dei pacchetti di dati di test aventi una percentuale crescente di dati nuovi e si effettua l'analisi degli outliers che vengono riconosciuti. La valutazione dei clusters ottenuti avviene quantitativamente tramite

l'indice della Silhouette, che valuta la loro coesione, e qualitativamente mediante i word clous dei cluster ottenuti. Questi ultimi ci consentiranno di analizzare il contenuto dei documenti dei diversi clusters.

1. **Analisi dei clusters degli outliers dei dati di partenza**
   Questa analisi avviene considerando la parte dei dati iniziali che non sono stati utilizzati per il train, ovvero **X_0**. Si testa l'Elliptic Envelope su meta' di questi dati e si selezionano gli outliers. Per ottenere il clustering di questi dati si utilizza il metodo di clustering gerarchico.
   Si applicano agli outliers vari clustering ognuno corrispondente ad un diverso numero **k** di raggruppamenti. Si valuta il valore della sihouette per ogni **k** e si cerca il valore ottimale di **k**.

2. **Creazione delle finestre di test**
   In questa analisi sono state create delle finestre di dati di dimensione e composizione di dati iniziali e finali variabile. Sono state impostate 10 diverse dimensioni delle finestre e per ognuna delle equali sono state determinate 10 diverse composizioni con percentuali di dati nuovi via via crescenti.
   Detta **max_size** la dimensione della finestra più grande, si ottengono finestre di dimensione:
   window_size: $0.1 \cdot$ **max_size**, $0.2 \cdot$ **max_size**, ..., **max_size**.
   Per ognuna di queste dimensioni si ottengono 11 pacchetti di dati con un numero crescente di dati nuovi: 0 , $0.1 \cdot$ **window_size** , $0.2 \cdot$ **window_size** ..., **window_size**.
   Su ognuno di questi pacchetti di dati su testa l'elliptic envelope per vedere se effettivamente la pecentuale di outliers riconosciuti aumenta all'aumentare della percentuale di dati nuovi al loro interno. Infatti questo fenomeno sarebbe un segnale della possibile presenza di drift nei dati nuovi.

3. **Analisi dei clusters degli outliers delle finestre di test**
   Vengono create nuove finestre per l'analisi degli outliers ottenuti, in quanto, a causa del basso numero di dati a disposizione, non sarebbe utile l'analisi degli outliers su finestre di dati troppo piccole. Impostata con **size** la dimensione delle finestre si ottengono 11 diverse finestre di dati con composizione di dati nuovi via via crescenti dallo 0% al 100%:
   0 , $0.1 \cdot$ **size** , $0.2 \cdot$ **size** ..., **size**. Per ogni finestra di test si realizza il clu-

stering con diversi valori del numero di gruppi **k** e si confrontano i valori della silhouette per i vari **k**.
Per verificare la presenza di nuovi cluster negli outliers dei dati finali, si studia l'andamento delle silhouette al variare di k e all'aumentare della percentuale dei dati nuovi. In particolare si vuole vedere se per qualche **k** il valore della silhouette passa da valori molto bassi a valori alti, indicando la formazione di cluster sempre più coesi all'aumentare della percentuale dei dati nuovi. La coesione dei cluster viene visualizzata anche mediante le pca dei dati. Il contenuto dei cluster formatisi viene poi analizzati mediante i word cloud.

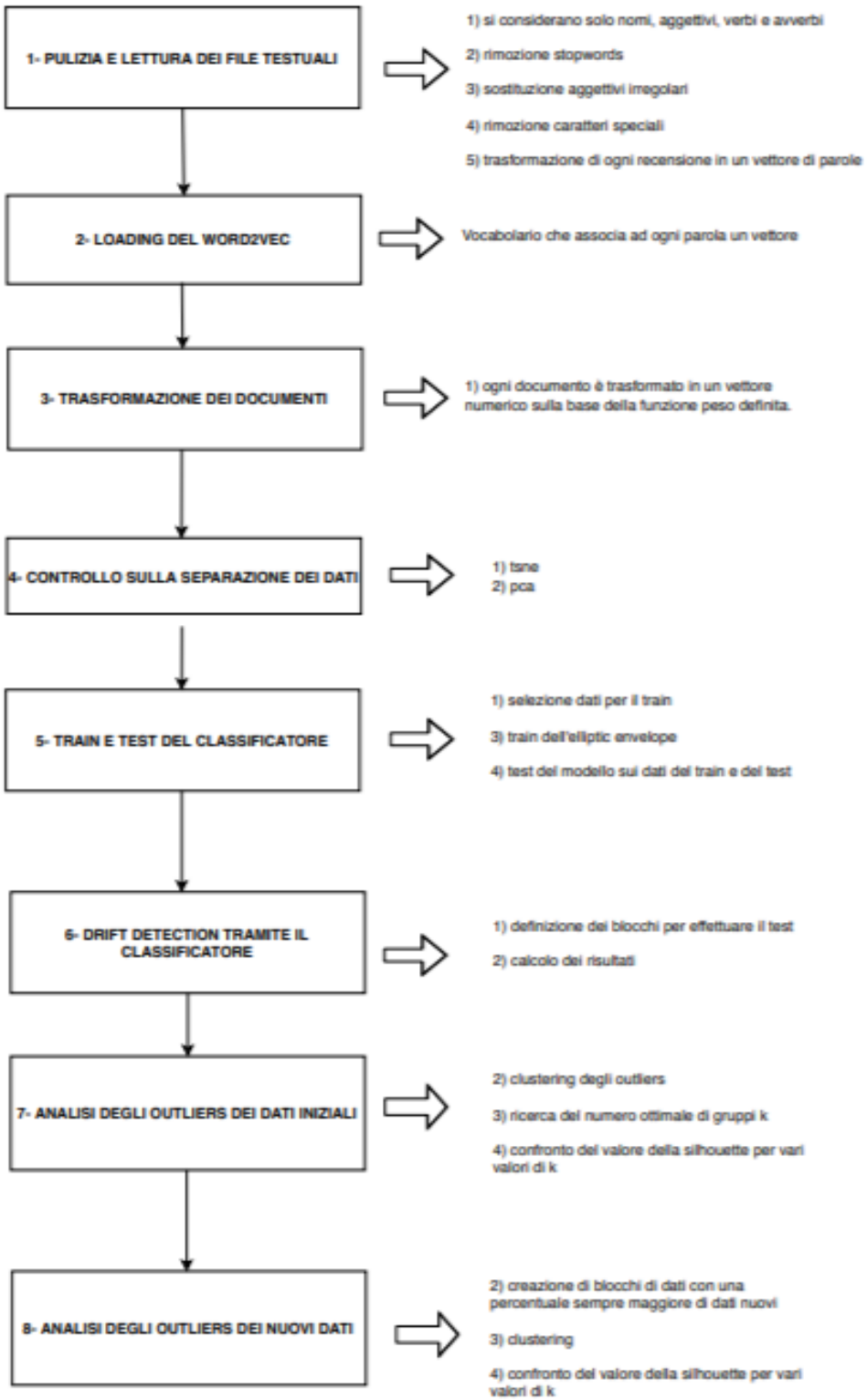


Figura 3.4: Diagramma riassuntivo della metodologia

Capitolo 4

Risultati sperimentali

4.1 Introduzione

In questa sezione vengono presentati gli esperimenti eseguiti e i risultati che sono stati ottenuti. Il procedimento illustrato nel capitolo 2 viene adattato al caso dei dati presi in considerazione.

4.2 Esperimento su Articoli di Wikipedia: Matematica e Tecnologia

Il dataset che consideriamo è costituito 5000 articoli di Wikipedia in lingua inglese di cinque argomenti: Matematica, Tecnologia, Letteratura, Biologia e Alimenti. Per ogni argomento si dispone di 1000 articoli. In questo esperimento considereremo gli articoli che trattano di Matematica e quelli che trattano di Tecnologia. Considereremo i primi come dati storici e i secondi come dati nuovi. Nella realizzazione dell'esperimento sarà necessario adattare il procedimento visto nel capitolo 2 al caso considerato, in particolare nella definizione delle stopwords. Si perde quindi generalità utilizzando dei metodi specifici per i dati considerati per ottenere dei risultati migliori.

4.2.1 Data Cleaning

Si elencano qui gli strumenti (intesi come metodi e variabili) che verranno utilizzati per la pulizia dei dati.

1. **definizione stopwords**: Come stopwords si considerano le stopwords inglesi a cui si aggiungono le parole legate al contesto degli articoli di wikipedia:

'encyclopedià, 'articlè, 'titl', 'http', 'pagè, 'badtitlè, 'wiki', 'request', 'containt', 'search', 'navigation', 'citation', 'quotè, 'disambiguation', 'redirect'.
Si aggiungono inoltre parole legate alla sintassi della creazione di pagine web: 'font', 'stylè, 'badtitlè, 'http', 'character'.
Si eliminano infine gli articoli relativi al messaggio di errore nella ricerca, ovvero quelli costituiti dalla sola stringa "Jump navigation Jump search requested page title contains unsupported characters Return Main Page Retrieved from https wikipedia wiki Special Badtitle"

2. **rimozione caratteri speciali**
   La funzione remove_string_special_characters(stringa): i caratteri speciali, la punteggiatura e i numeri dalla stringa passata in input.

3. **definizione di un dizionario per gli aggettivi irregolari**: servirà per trasformare gli aggettivi superlativi irregolari nella loro radice. Ad esempio per trasformare 'best' in 'good'. irr_adg = { 'worsè: 'bad', 'worst':'bad', 'better':'good' ...}

4. **trasformazione aggettivi**
   la funzione sub_adj(stringa): trasforma gli aggettivi superlativi nella loro forma base in due modi:

   - se l'aggettivo è nel dizionario degli irregolari lo trasforma nel suo corrispondente non superlativo. Esempio: 'best' diventa 'good'.
   - se invece l'aggettivo non è presente nel dizionario ma la parola finisce con 'est', allora la particella 'est' viene eliminata. Esempio: 'greatest' diventa 'great'.

5. **penn_to_wn(tag)**: riceve in input il tag PennTreebank e lo converte nel suo corrispettivo WordNet tag. Ad esempio per i nomi si avrà il tag 'NOUN', per gli aggettivi il tag 'ADJ', per i verbi il tag 'VERB', per gli avverbi il tag 'ADV'.

6. **filter_text(text)**: riceve un testo in input e ne seleziona gli aggettivi, i nomi, i verbi e gli avverbi utilizzando WordNet tag. Selezionerà quindi le parole aventi come tag uno tra: 'NOUN', 'ADJ', 'ADV', 'VERB'.

7. **process_lines(txt)**: riceve in input un documento di tsto e applica in ordine:

   - rimozione caratteri speciali

- trasformazione aggettivi superlativi
- selezione di nomi, verbi, avverbi, aggettivi
- rimozione stopwords
- riduzione delle parole alla loro radice

8. **lettura recensioni**: ogni recensione viene trasformata in un vettore di parole e aggiunta a una lista.
   Si mostra il risultato del procedimento di data cleaning su una recensione.

Gli articoli con argomento 'Matematicà, dopo la pulizia verranno salvati nella lista **lista_testi_0**, mentre quelli con argomento 'Tecnologià nella lista **lista_testi_1**.

Testo dell'articolo:
" From Wikipedia free encyclopedia Jump navigation Jump search other uses Abacus disambiguation Abaci Abacuses redirect here the Turkish surname Abac the medieval book Liber Abaci Calculating tool Chinese abacus Suanpan Calculating Table Gregor Reisch Margarita Philosophica woodcut shows Arithmetica instructing algorist abacist inaccurately represented Boethius Pythagoras There keen competition between two from introduction Algebra into Europe century until triumph The abacus plural abaci abacuses also called counting frame calculating tool that use Europe China Russia centuries before adoption written Hindu Arabic numeral system exact origin abacus still unknown Today abacuses often constructed bamboo frame with beads sliding wires originally they were beans stones moved grooves sand on tablets wood stone metal Abacuses come different designs Some designs like bead frame consisting beads divided into tens used mainly teach arithmetic although they remain popular post .."

Risultato del data cleaning:
'us', 'abacù, 'abacù, 'abacù, 'turkish', 'surnam', 'mediev', 'book', 'liber', 'calcul', 'tool', 'chinè, 'abacù, 'suanpan', 'calcul', 'tabl', 'gregor', 'reisch', 'margaritià, 'philosophicà, 'woodcut', 'show', 'arithmeticà, 'instruct', 'algorist', 'abacist', 'inaccur', 'repr', 'boethiù, 'pythagorà, 'keen', 'competit', 'introduct', 'algebrà, 'europ', 'centuri', 'abacù, 'plural', 'abacù, 'abacù, 'alsò, 'call', 'count', 'framè, 'calcul', 'tool', 'usè, 'europ', 'chinà, 'russià, 'centuri', 'adopt', 'writè, 'hindù, 'arab', 'numer', 'system', 'exact', 'origin', 'abacù, 'still', 'unknown', 'today', 'abacù, 'often', 'construct', 'bamboò, 'framè, 'bead', 'slidè, 'wirè, 'origin', 'bean', 'stonè, 'movè, 'groov', 'sand', 'tablet', 'wood', 'stonè, 'metal', 'abacù, 'comè, 'differ', 'design', 'design', 'bead', 'framè,

'consist', 'bead', 'divid', 'ten', 'usè, 'mainli', 'teach', 'arithmet', 'remain', 'popular', 'post', 'soviet', 'statè, 'tool', 'design', 'japan', 'soroban', 'usè, 'practic', 'calcul', 'even', 'involv', 'sever', 'digit', 'particular', 'abacù, 'design', 'usual', 'numer', 'differ', 'method', 'perform', 'certain', 'typè, 'calcul'

4.2.2 Creazione del Word Embedding

In questo caso si è scelto di utilizzare il word embedding realizzato con i documenti di Google News, in quanto è il più ricco di parole. Inoltre l'utilizzo di un word embedding già trainato consente di eliminare parole date da errori di battitura.

4.2.3 Trasformazione dei documenti

Si definisce n = 500 il numero il numero delle parole più comuni che si vogliono selezionare per ognuno dei due gruppi di dati, in quanto si è visto che è il numero di parole ottimale per identificare la classe dei dati storici.
Tramite la funzione **get_most_common**(lista_testi, n) si ottengono le 500 parole più comuni in **lista_testi_0** e in **lista_testi_1** e si salvano rispettivamente nei vettori **most_common_0** e **most_common_1**.
Con la funzione **weight_calc_improved**(lista_testi, word_embedding, most_common) si ottiene la trasformazione di ogni documento in un vettore numerico.

Trasformazione di un articolo

```
weight_calc_improved(review)
```

```
0    -0.437359
1    -0.415947
2    -1.183225
3    -1.464039
4     0.346146
        ...
95    0.274046
96   -0.902324
97   -0.348483
98    0.058999
99   -1.448795
Length: 100, dtype: float32
```

Figura 4.1:

4.2.4 Visualizzazione dei dati

Visualizziamo i dati tramite le tecniche t_SNE e PCA. Tramite queste tecniche possiamo notare se con la trasformazione realizzata i dati storici e quelli nuovi oc-

cupano le stesse regioni dello spazio o se si collocano in aree diverse. Notiamo che in questo caso i dati sembrano occupare zone distinte.

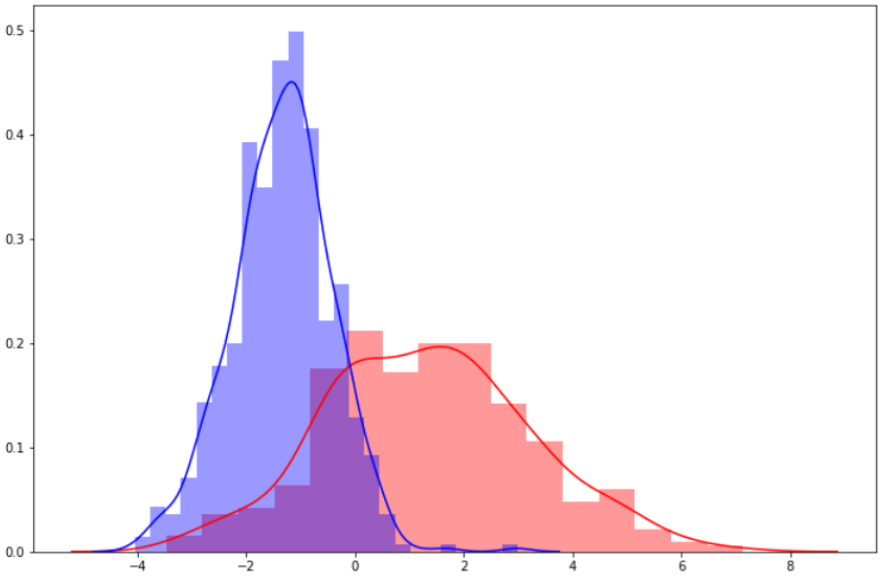


Figura 4.2: Plot delle prime due componenti principali dei dati. In particolare in rosso i dati storici e in blu i dati nuovi.

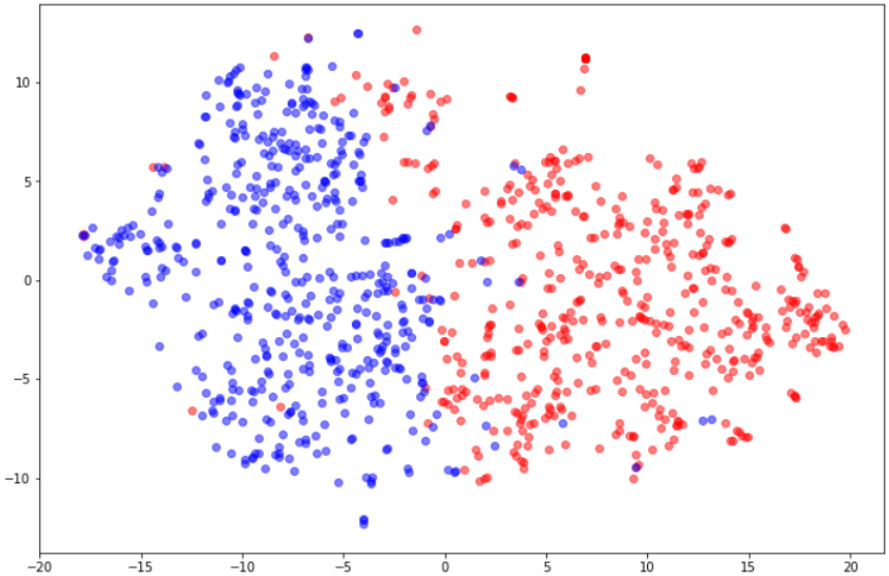


Figura 4.3: Plot della TSNE a due componenti. In particolare in rosso i dati storici e in blu i dati nuovi.

4.2.5 Train dell'Elliptic Envelope

I dati di **lista_testi_0** vengono divisi in due parti, la prima metà verrà utilizzata nella fase di training e l'altra nella fase di test. Si addestra quindi l'Elliptic Envelope sulla prima parte di dati e si testa sulla seconda per verificare l'efficacia dell'algoritmo.

4.2.6 Creazione delle finestre di dati di test

Per la creazione delle finestre su cui testare il modello si pone **max_size** = 500. Si ottengono quindi finestre di dimensione:
window_size: 0.1 · **max_size** = 50, 0.2 · **max_size** = 100, ..., **max_size** = **500**.
Per ognuna di queste dimensioni si ottengono 11 pacchetti di dati con un numero crescente di dati nuovi: 0 , 0.1 · **window_size** , 0.2 · **window_size** ..., **window_size**.

In totale avremo quindi 9 * 11 blocchi di dati e si testa l'Elliptic Envelope su ognuna di essi .
Notiamo che all'aumentare della percentuale di dati nuovi il numero di outliers riconosciuti aumenta. Ciò rappresenta un primo segnale della possibile presenza di drift. Infatti il fatto che la maggior parte dei dati nuovi venga identificata come costituita da outliers vuol dire che essa non viene riconosciuta come appartenente alla classe dei dati storici.

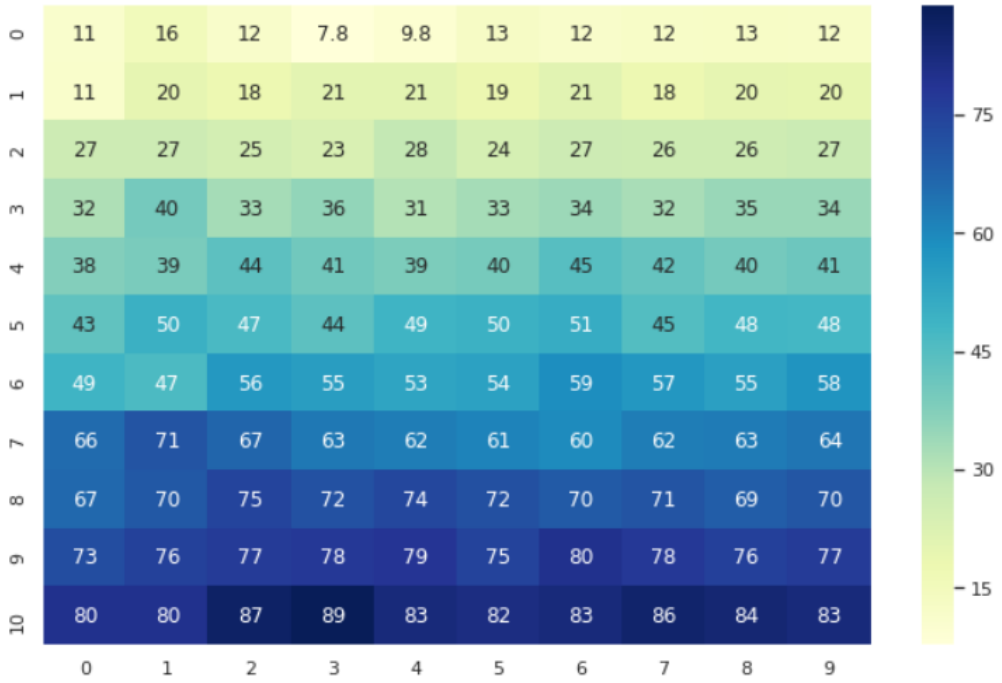


Figura 4.4:

4.2.7 Analisi degli Outliers per la Drift Detection

In questa fase si cercherà di identificare il drift dei dati. In particolare si vuole verificare se tra gli outliers dei dati nuovi si creano dei cluster o dei raggruppamenti coesi che non erano presenti nei dati originari, e che quindi essi possano identificare una nuova classe.

Analisi degli outliers dei dati iniziali

Consideriamo 200 degli elementi appartenenti ai dati iniziali che non sono stati utilizzati per il train dell'Elliptic Envelope e testiamo il modello di outliers detection su questi dati.
Salviamo nella lista **X_0** gli outliers ottenuti e applichiamo il clustering gerarchico su questi ultimi. Detto **k** il numero di clusters si itera il procedimento con **k** =1,...,8. Si calcola il valore della silhouette per ogni **k**.
Notiamo che i valori della silhouette sono tutti molto bassi, indice del fatto che non sono presenti cluster molto coesi.

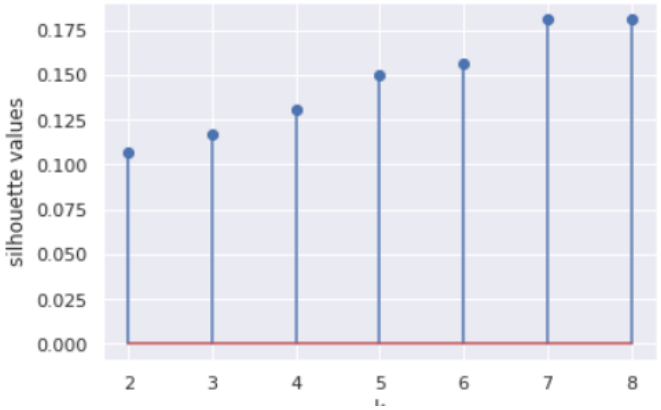


Figura 4.5: Valori della silhouette al variare di k

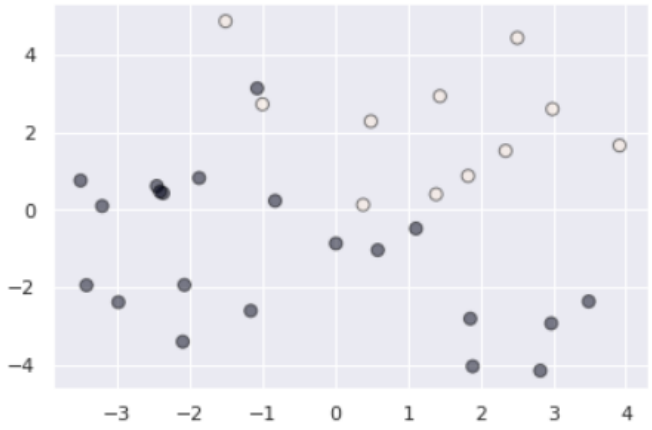


Figura 4.6: Visualizzazione mediante PCA dei cluster ottenuti. In nero gli elementi assegnati al primo cluster e in bianco quelli assegnati al secondo.

Analizziamo il contenuto dei cluster con $k = 2$ mediante l'analisi delle 20 parole più frequenti nei due cluster e dei WordCloud che essi generano. Notiamo che le parole più frequenti nei due cluster sono diverse, anche se appartengono allo stesso macroargomento di "Matematica".

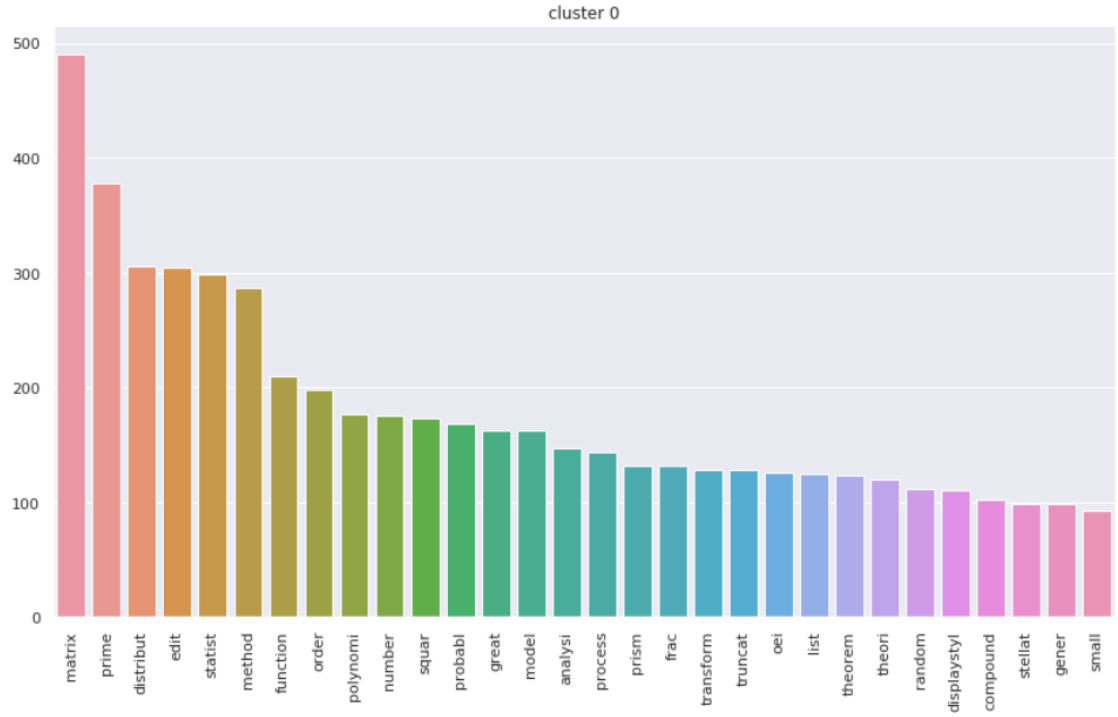


Figura 4.7: Le 20 parole più frequenti all'interno del primo cluster

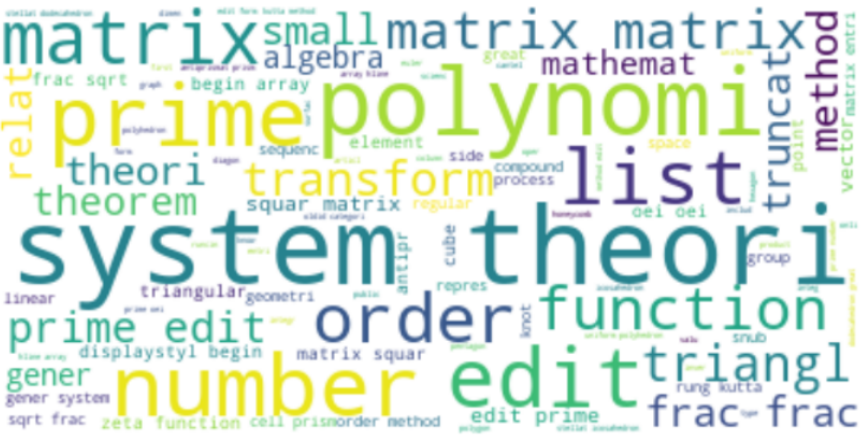


Figura 4.8: WordCloud delle 100 parole più frequenti nel primo cluster

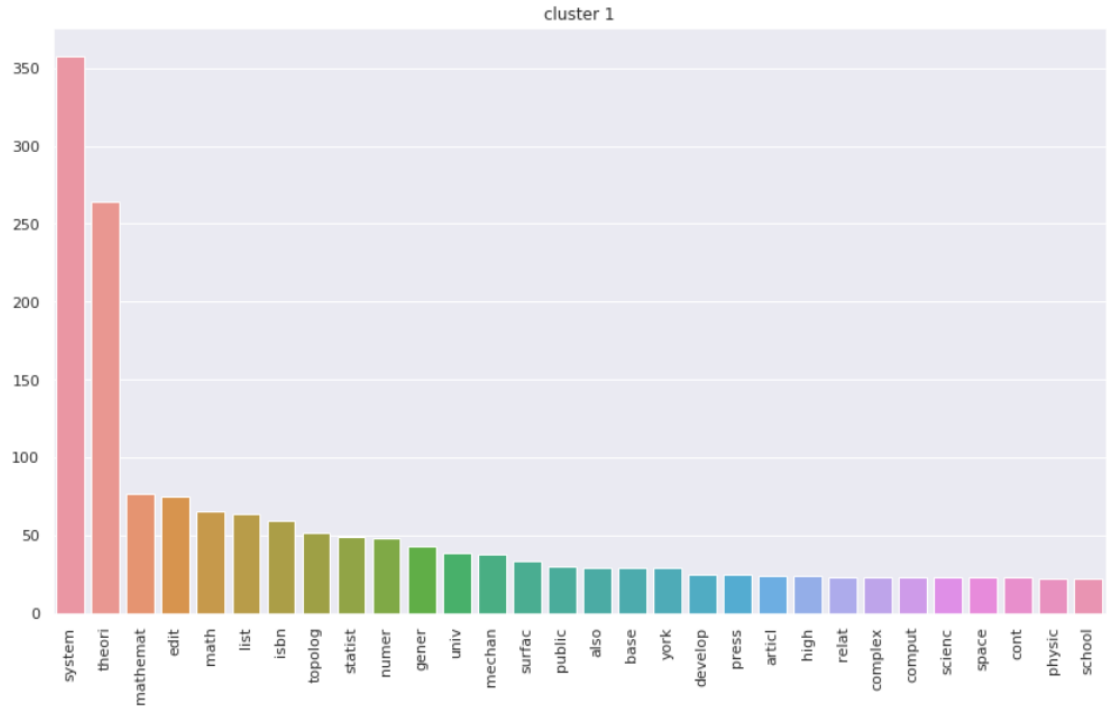


Figura 4.9: Le 20 parole più frequenti all'interno del secondo cluster

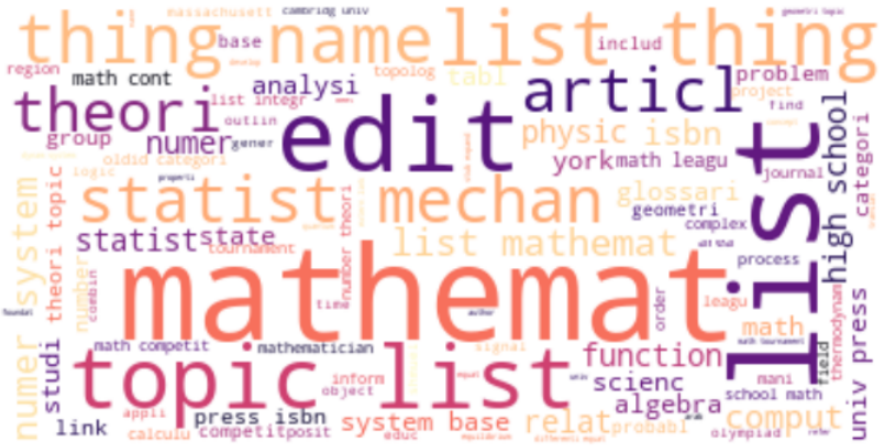


Figura 4.10: WordCloud delle 100 parole più frequenti nel secondo cluster

Analisi degli outliers delle finestre di test

Per la creazione delle finestre di test consideriamo la parte dei dati storici che non sono stati utilizzati nè per il train dell'Elliptic Envelope nè per lo studio degli

outliers della classe iniziale nel passaggio precedente (ovvero gli ultimi 300 articoli con argomento Matematica). Creiamo delle finestre di test di dimensione 300 con percentuale via via crescente di documenti con argomento "Tecnologia". Si vuole verificare la presenza di nuovi cluster all'interno dei nuovi dati. A tale scopo per ogni finestra di dati di test si realizzano i seguenti passaggi:

1. Si testa l'Elliptic Envelope sui dati del test considerato
2. Definiamo **X_test_outliers** come la lista contentente gli outliers del test. A questa lista viene aggiunta **X0** (la lista degli outliers precedentemente considerati)
3. Si applica il clustering a questo insieme di dati e si valuta il valore della silhouette per ogni k.

   Alla fine del procedimento analizziamo il valore delle silhouette al variare di k nei vari test. Notiamo che all'aumentare della percentuale di dati nuovi il valore della silhouette per k = 2 aumenta passando da 0.14 a 0.43, indice della presenza sempre più evidente di un cluster sempre più coeso negli outliers considerati.

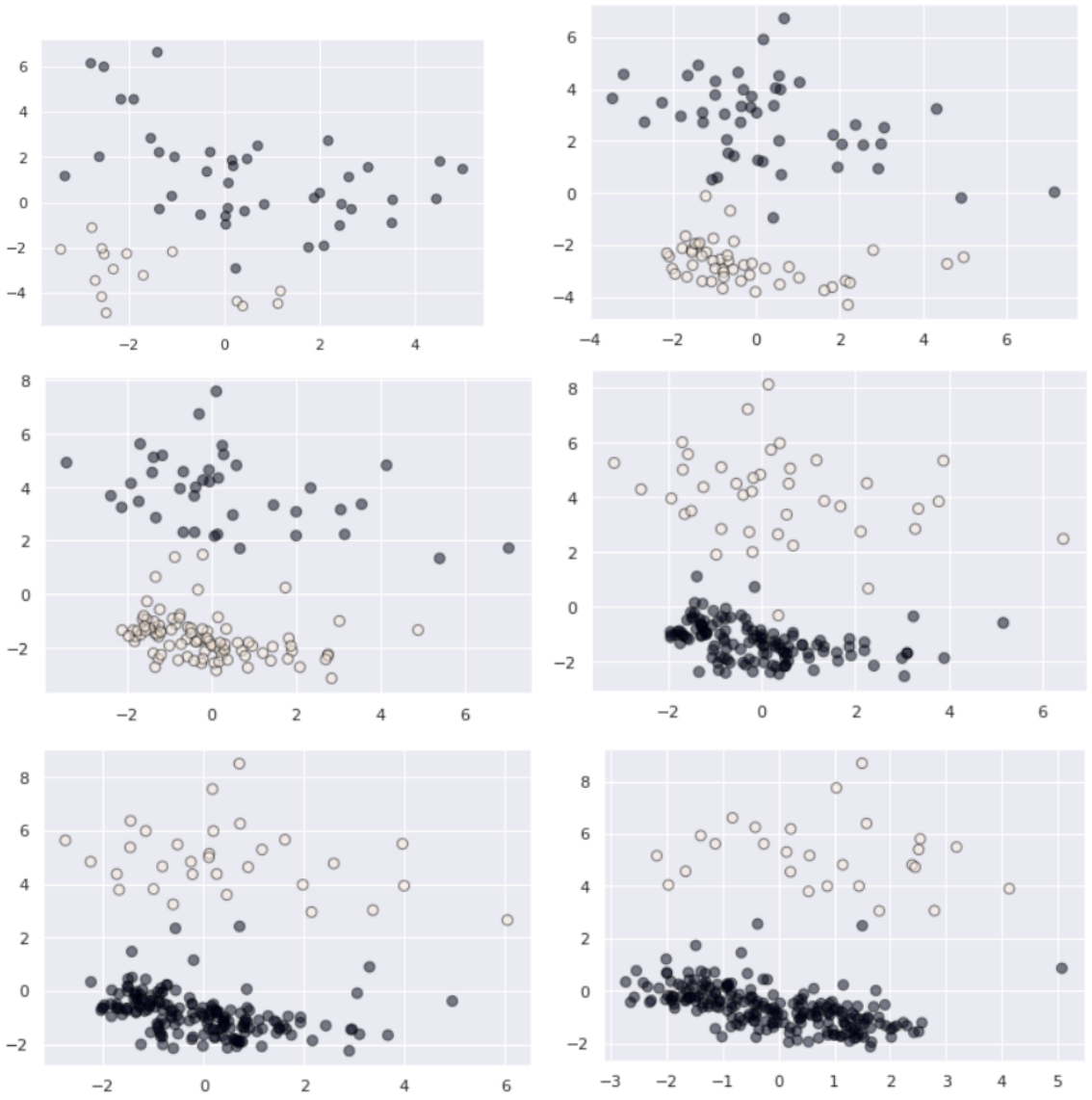


Figura 4.11: Visualizzazione mediante PCA dei cluster ottenuti nei vari test realizzati. In nero gli elementi assegnati al primo cluster e in bianco quelli assegnati al secondo.

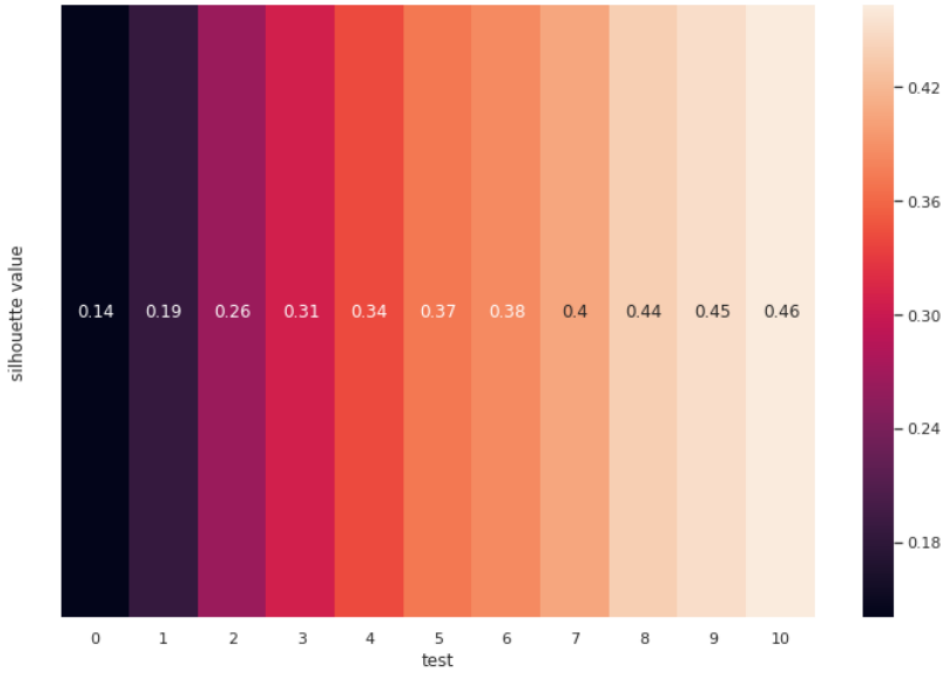


Figura 4.12: Valori della Silhouette per k =2 all'incrementare della percentuale dei dati nuovi all'interno delle finestre di test

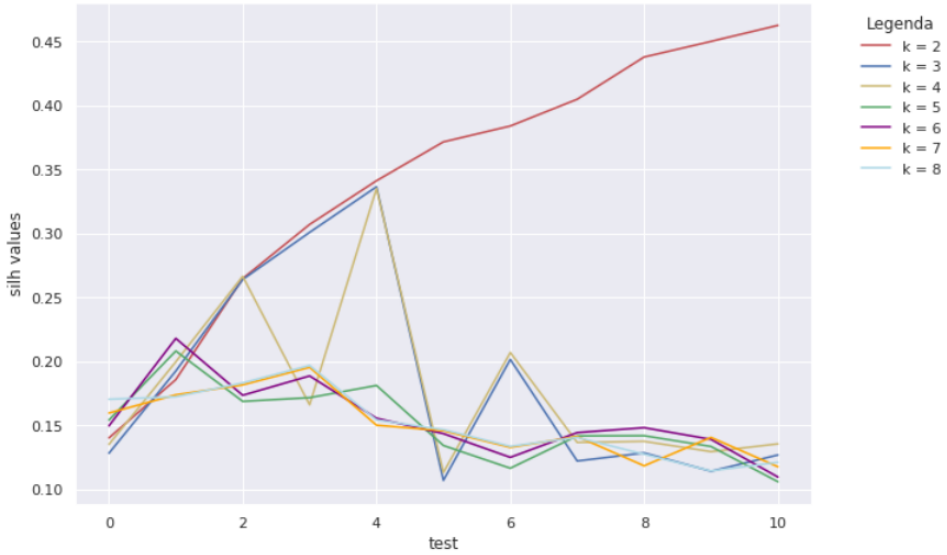


Figura 4.13: Valori della Silhouette al variare di k e al variare della finestra di test all'incrementare della percentuale dei dati nuovi

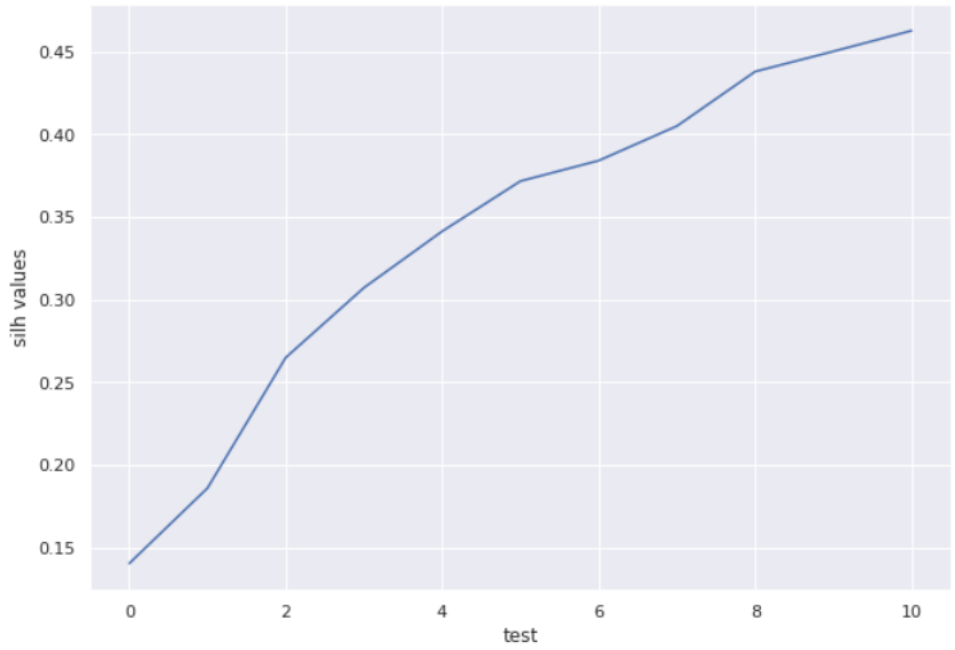


Figura 4.14: Valori della Silhouette per k =2 all'incrementare della percentuale dei dati nuovi all'interno delle finestre di test

Fissato k = 2 si vuole analizzare il contenuto dei cluster individuato considerando le 20 parole più frequenti al loro interno e dei WordCloud che essi generano.

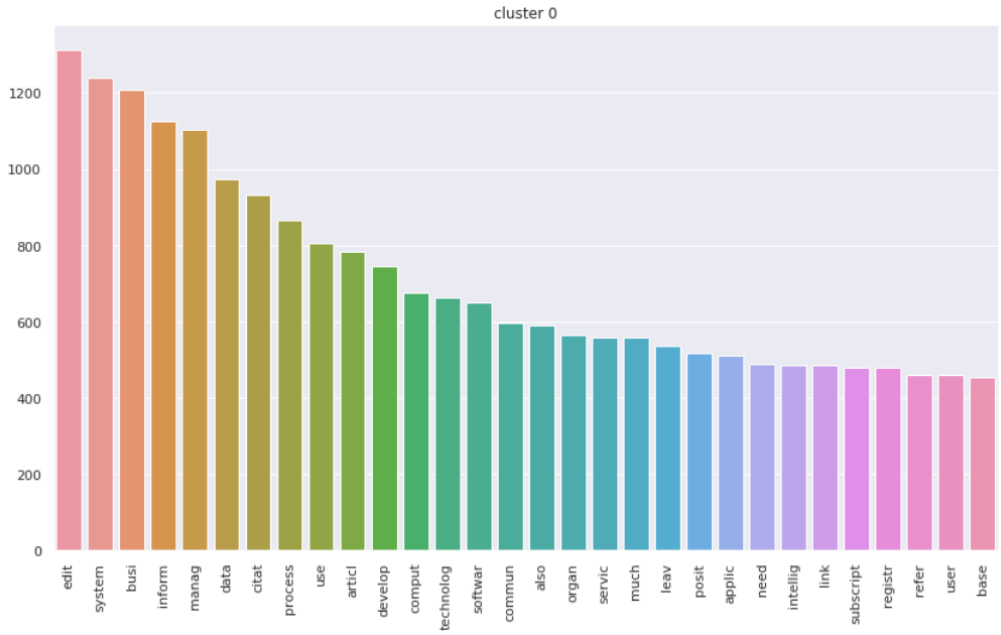


Figura 4.15: Le 20 parole più frequenti all'interno del primo cluster

Figura 4.16: WordCloud delle 100 parole più frequenti nel primo cluster

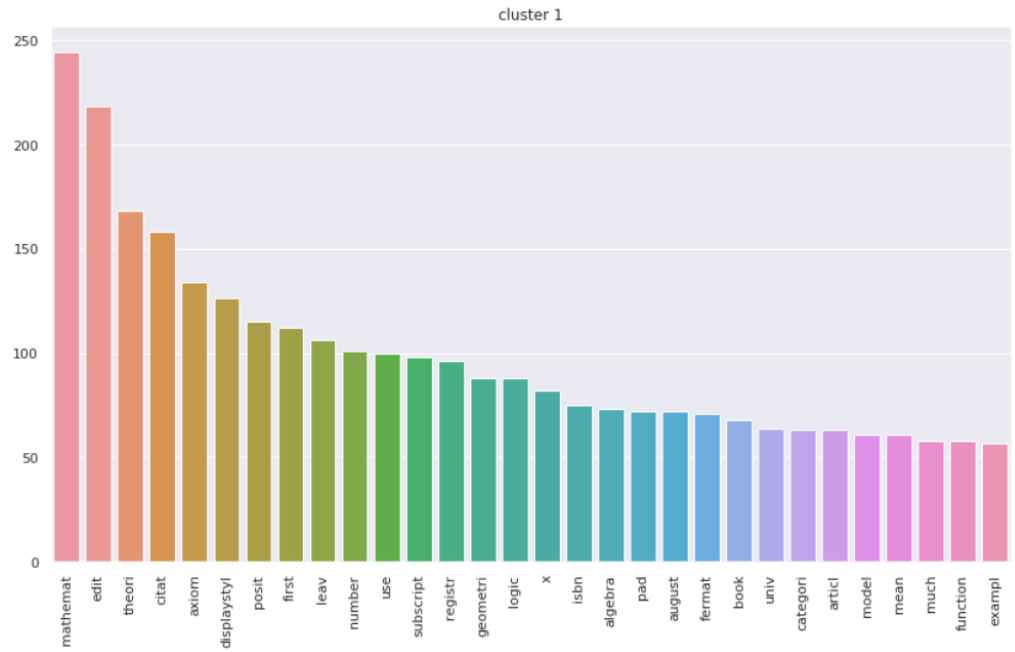


Figura 4.17: Le 20 parole più frequenti all'interno del secondo cluster

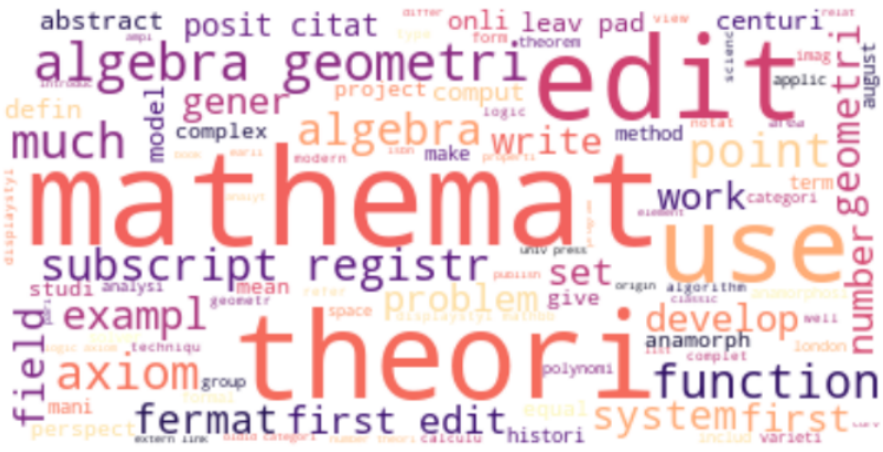


Figura 4.18: WordCloud delle 100 parole più frequenti nel secondo cluster

4.2.8 Conclusioni:

Dall'analisi degli outliers dei test emerge che all'aumentare della percentuali dei dati nuovi si definisce all'interno degli outliers una nuova classe sempre più coesa che non appariva nei dati originari. Analizzando il contenuto dei documenti di questo cluster appaiono delle parole nuove, legate a tematiche di tecnologia e informatica,

che non comparivano nei cluster degli outliers dei dati originari. Ciò è indice della presenza di un nuovo argomento nei nuovi dati che puo' essere associato ad una nuova classe.

4.3 Articoli di Wikipedia: altri risultati

In questa sezione consideriamo gli altri esperimenti che sono stati eseguiticon gli altri dati a disposizione nel dataset degli articoli di Wikipedia. Volta per volta considereremo dati appartenenti a due argomenti diversi e trattiamo da dati storici quelli del primo argomento e come dati nuovi quelli del secondo.

4.3.1 Data cleaning

Si mantengono le procedure utilizzate nell'esperimento precedente:

1. rimozione caratteri speciali
2. trasformazione aggettivi superlativi
3. selezione di nomi, verbi, avverbi, aggettivi
4. rimozione stopwords
5. riduzione delle parole alla loro radice

I dati storici, dopo la pulizia verranno salvati nella lista **lista_testi_0**, mentre quelli nuovi nella lista **lista_testi_1**.

4.3.2 Creazione del Word Embedding

Come nell'esperimento precedente anche in questi casi si è scelto di utilizzare il word embedding realizzato con i documenti di Google News.

4.3.3 Esperimento con i dati di Biologia e Letteratura

Si considerano in questo esperimento i 1000 aricoli di Biologia e i 1000 di letteratura.

Trasformazione dei documenti

Si definisce n = 500 il numero il numero delle parole più comuni che si vogliono selezionare per ognuno dei due gruppi di dati, in quanto si è visto che è il numero di parole ottimale per identificare la classe dei dati storici.
Tramite la funzione **get_most_common**(lista_testi, n) si ottengono le 300 parole più comuni in **lista_testi_0** e in **lista_testi_1** e si salvano rispettivamente nei vettori **most_common_0** e **most_common_1**.
Con la funzione **weight_calc_improved**(lista_testi, word_embedding, most_common) si ottiene la trasformazione di ogni documento in un vettore numerico.

Visualizzazione dei dati

Visualizziamo i dati tramite le tecniche t_SNE e PCA. Notiamo che anche in questo caso i dati storici e nuovi sembrano occupare zone diverse dello spazio.

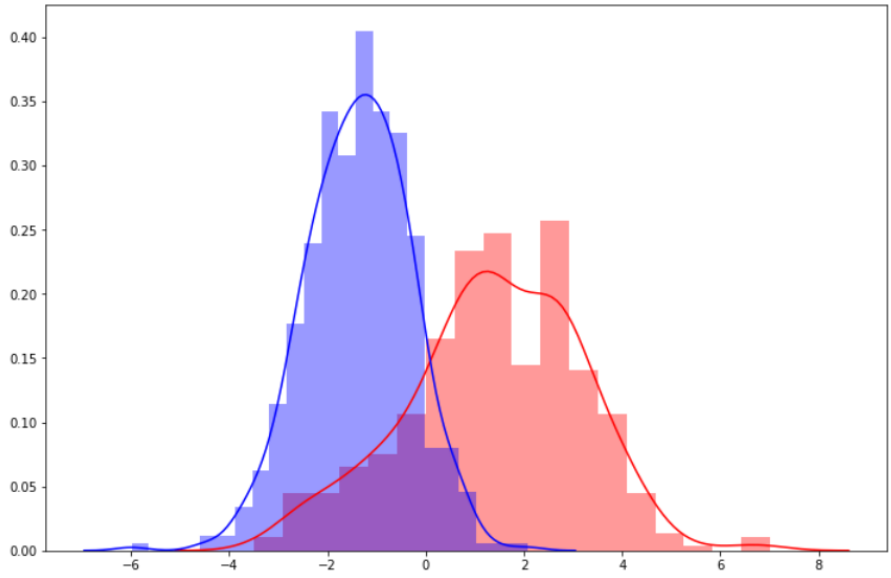


Figura 4.19: Plot delle prime due componenti principali dei dati. In particolare in rosso i dati storici e in blu i dati nuovi

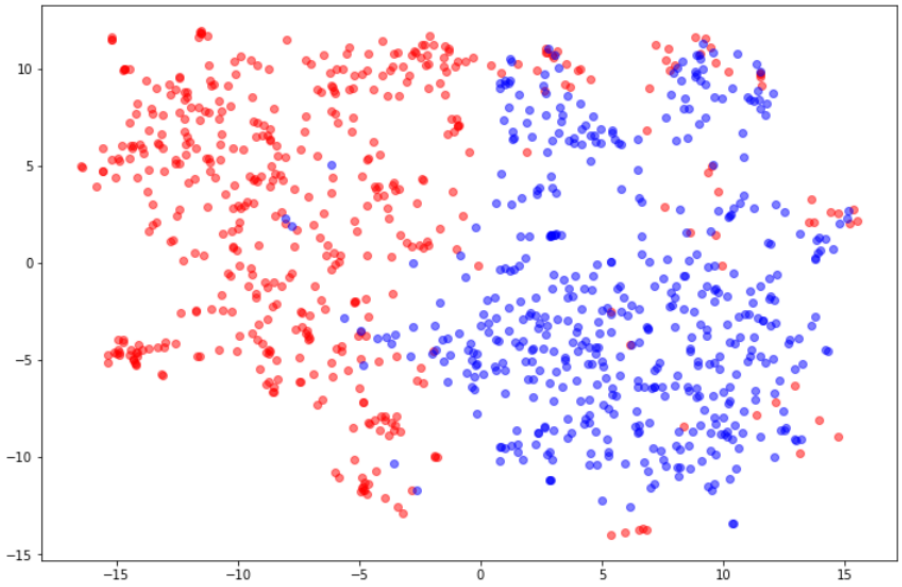


Figura 4.20: Plot della TSNE a due componenti. In particolare in rosso i dati storici e in blu i dati nuovi

Train dell'Elliptic Envelope

I dati storici di **lista_testi_0** vengono divisi in due parti di 500 dati, una verrà utilizzata nella fase di training e l'altra nella fase di test. Si addestra quindi l'Elliptic Envelope sulla prima parte di dati e si testa sulla seconda per verificare l'efficacia dell'algoritmo.

Creazione delle finestre di dati di test

Come nell'esperimento precedente vengono create finestre di test di dimensione variabile e con percentuale crescente di dati nuovi. Anche in questo caso notiamo che all'aumentare della percentuale di dati nuovi il numero di outliers riconosciuti aumenta, passando nei testi più numerosi (quelli corrispondenti all'ultima colonna) da percentuali di outliers tra il 17 e 19% a percentuali del 85 e 86%.

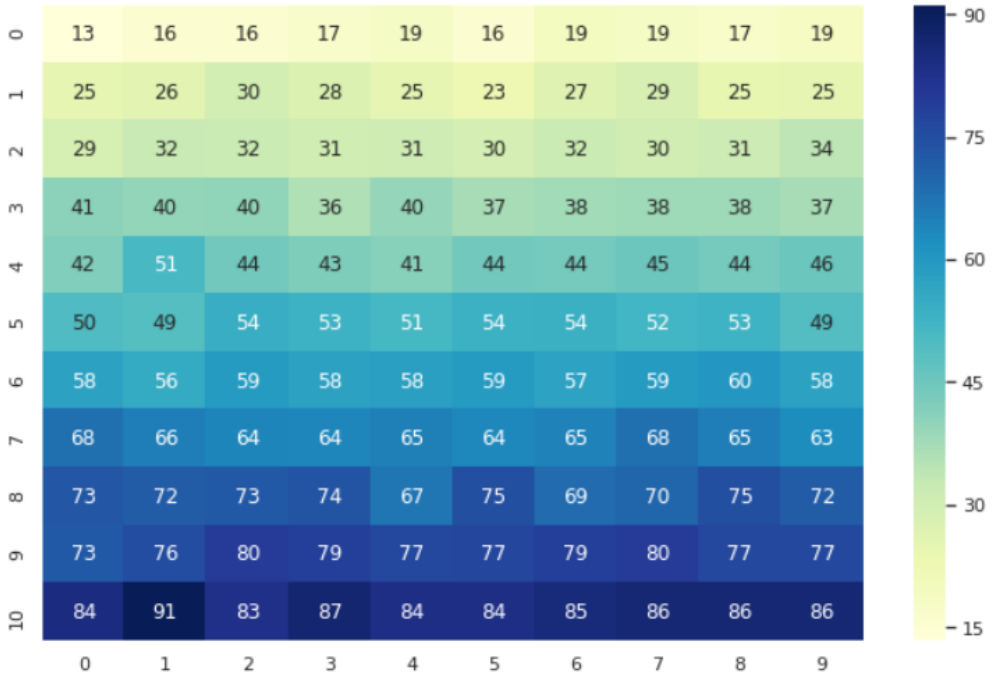


Figura 4.21: Percentuali di outliers riconosciute. Ogni elemento della tabella corrisponde al risultato su un test differente. Ogni colonna corrisponde ad una diversa dimensione della finestra di test e andando verso il basso si ha una percentuale sempre maggiore di dati nuovi, passando dallo 0% al 100%

Analisi degli outliers dei dati iniziali

Consideriamo la parte dei dati iniziali che non sono stati utilizzati per il train dell'Elliptic Envelope e testiamo il modello di outliers detection su 250 di questi dati.
Salviamo nella lista **X_0** gli outliers ottenuti e applichiamo il clustering gerarchico su questi ultimi. Detto **k** il numero di clusters si itera il procedimento con **k** =1,...,8. Si calcola il valore della silhouette per ogni **k**. Notiamo che i valori della silhouette si aggirano tutti intorno al 0.20, valori bassi che indicano che non sono presenti cluster molto coesi.

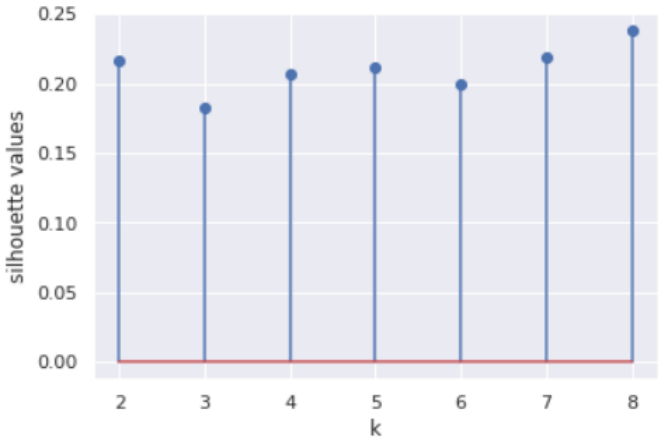


Figura 4.22: Valori della silhouette al variare di k

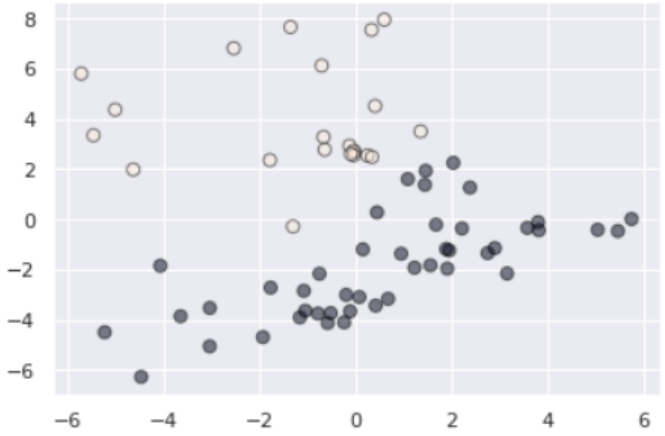


Figura 4.23: Visualizzazione mediante PCA dei cluster ottenuti. In nero gli elementi assegnati al primo cluster e in bianco quelli assegnati al secondo.

Poniamo $k = 2$ e analizziamo quindi le parole più frequenti dei primi due clusters.

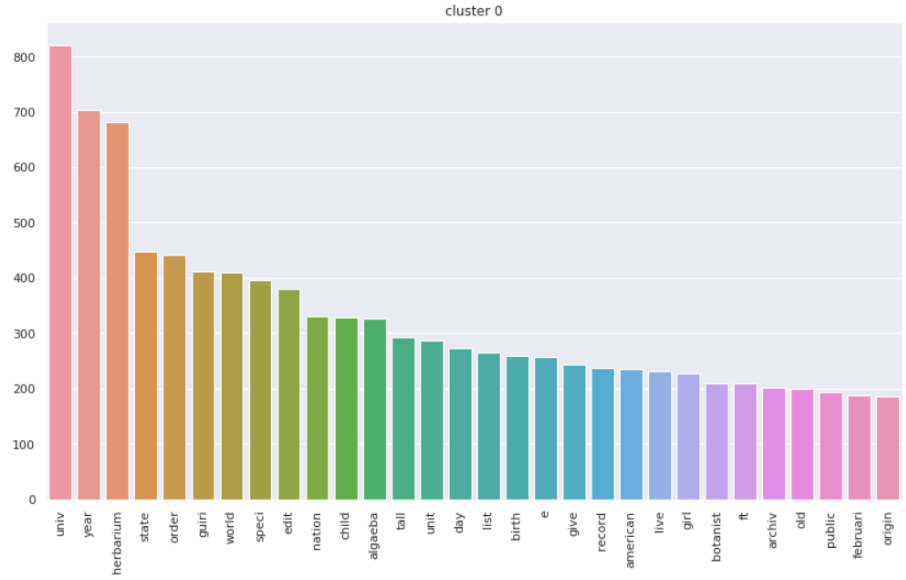


Figura 4.24: Le 20 parole più frequenti nel primo cluster

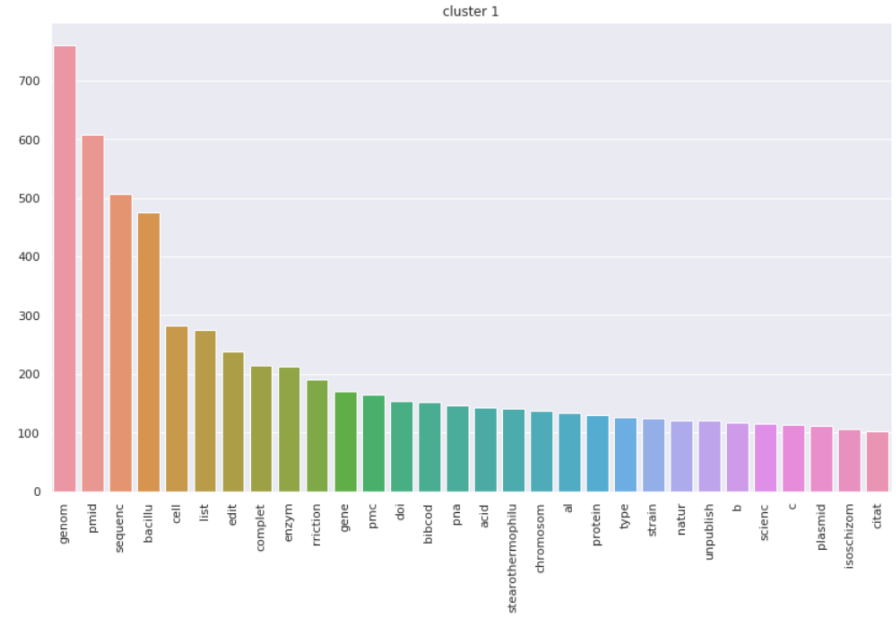


Figura 4.25: Le 20 parole più frequenti nel secondo cluster

Figura 4.26: WordCloud delle 100 parole più frequenti nel primo cluster

Figura 4.27: WordCloud delle 100 parole più frequenti del secondo cluster

Nel primo cluster compaiono parole inerenti alla biologia e alla chimica in senso stretto, cioè termini tecnici come nomi di molecole. Nel secondo cluster troviamo invece parole inerenti al mondo della biologia ma più generiche.

Analisi degli outliers delle finestre di test Si vuole verificare la presenza di nuovi cluster all'interno dei nuovi dati. A tale scopo consideriamo la parte dei dati storici che non sono stati utilizzati nè per il train dell'Elliptic Envelope nè per lo studio degli outliers della classe iniziale nel passaggio precedente (ovvero gli ultimi 250 articoli con argomento biologia). Creiamo 11 finestre di test di dimensione 250 con

percentuale via via crescente di documenti con argomento "letteratura", dallo 0 al 100%. Per ogni finestra di test:

1. Si testa l'Elliptic Envelope sui dati del test considerato
2. Definiamo **X_test_outliers** come la lista contentente gli outliers del test. A questa lista viene aggiunta **X_0** (la lista degli outliers precedentemente considerati)
3. Si applica il clustering a questo insieme di dati e si valuta il valore della silhouette per ogni k.

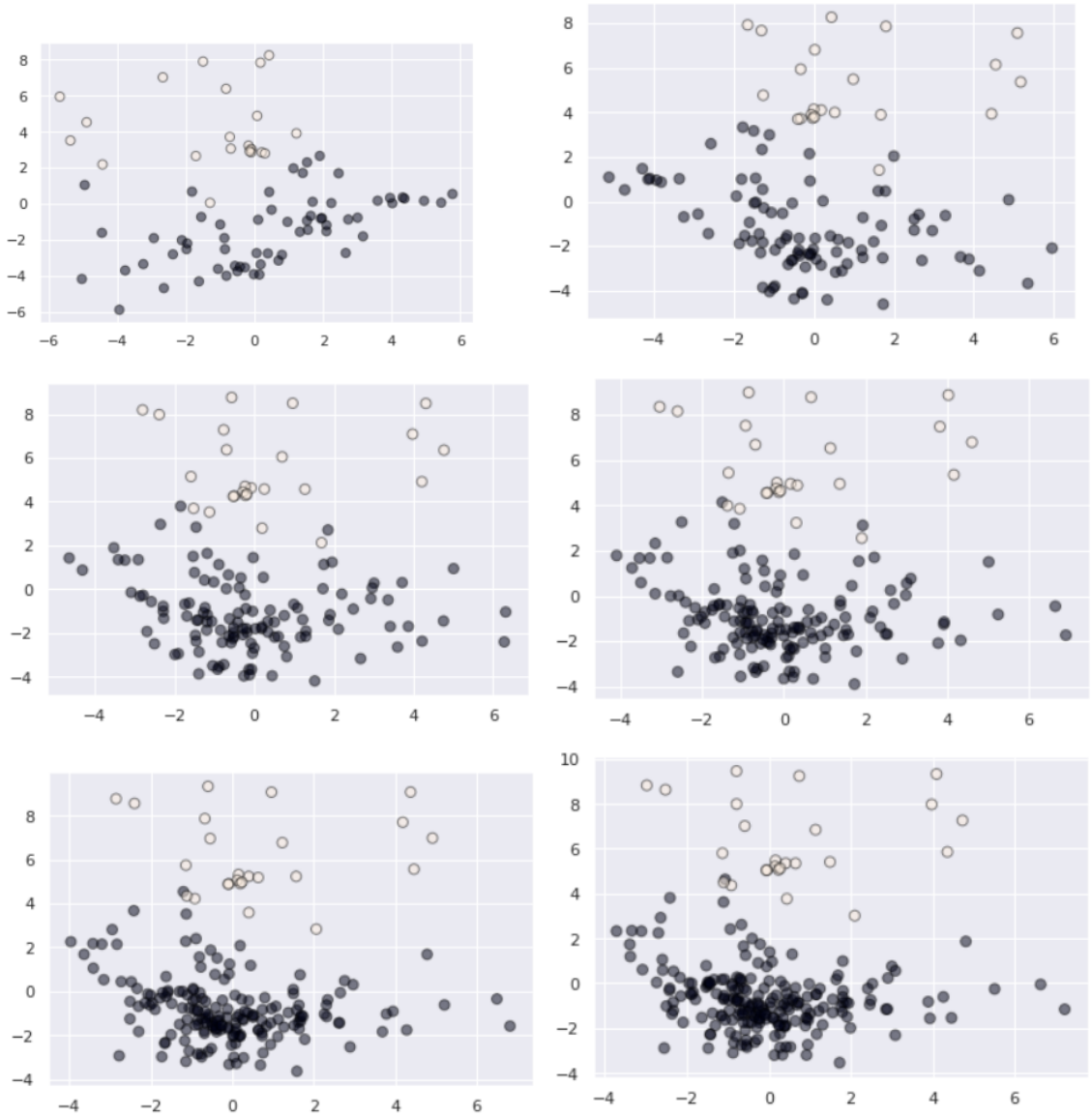


Figura 4.28: Visualizzazione tramite PCA dei cluster ottenuti nei vari test. In nero gli elementi del primo cluster e in bianco quelli del secondo.

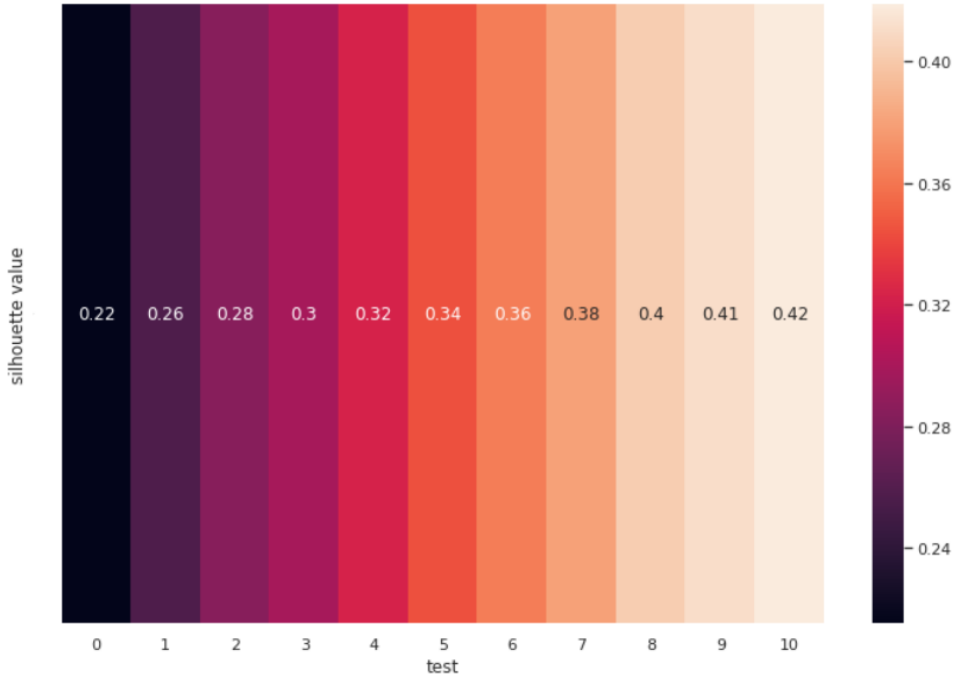


Figura 4.29: Valori della Silhouette per k =2 all'incrementare della percentuale dei dati nuovi all'interno delle finestre di test

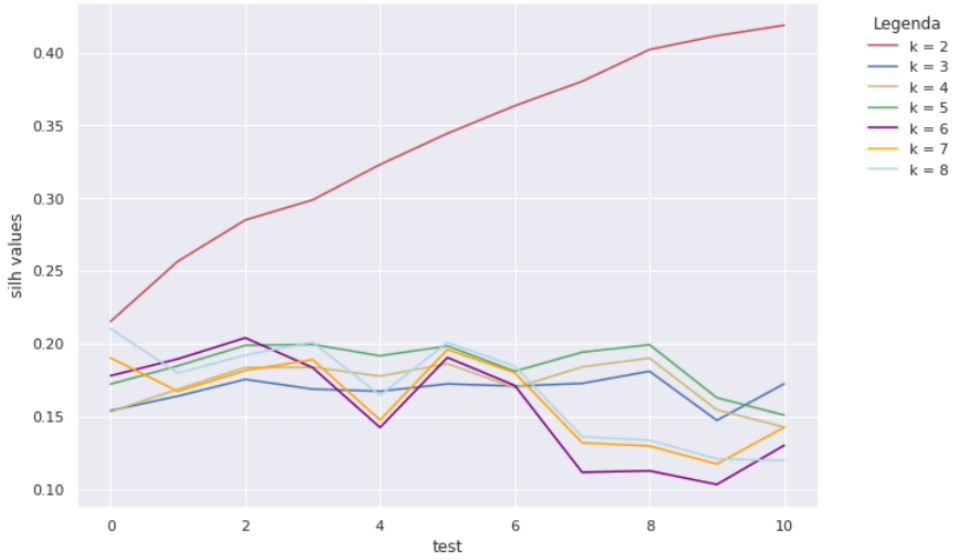


Figura 4.30: Valori delle Silhouettes al variare di k nei vari test

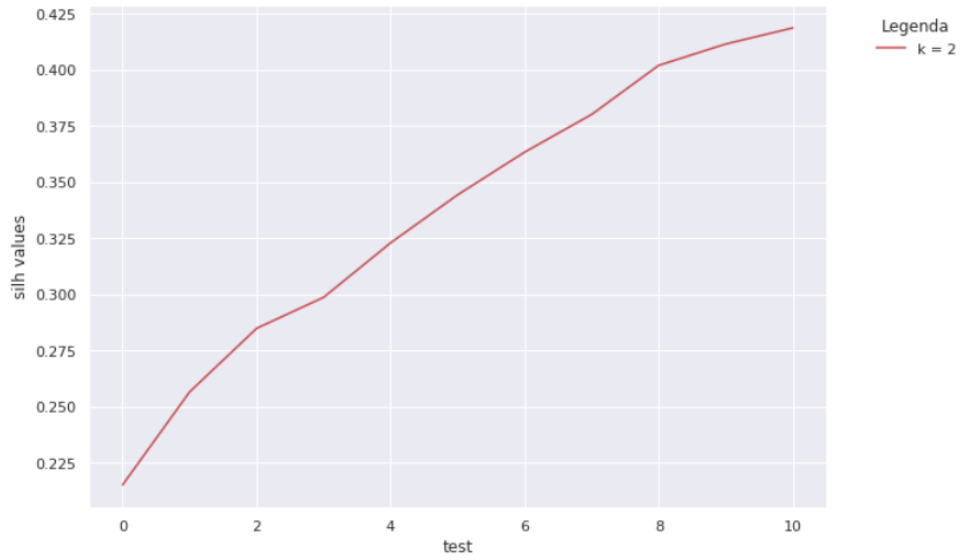


Figura 4.31: Valori della Silhouette per k =2 all'incrementare della percentuale dei dati nuovi all'interno delle finestre di test

Si nota quindi che all'aumentare dei dati nuovi nelle finestre di test si crea una classe sempre più coesa. Fissiamo dunque $k = 2$ e analizziamo il contenuto dei due cluster individuati considerando le 20 parole più frequenti al loro interno e dei WordCloud che essi generano.

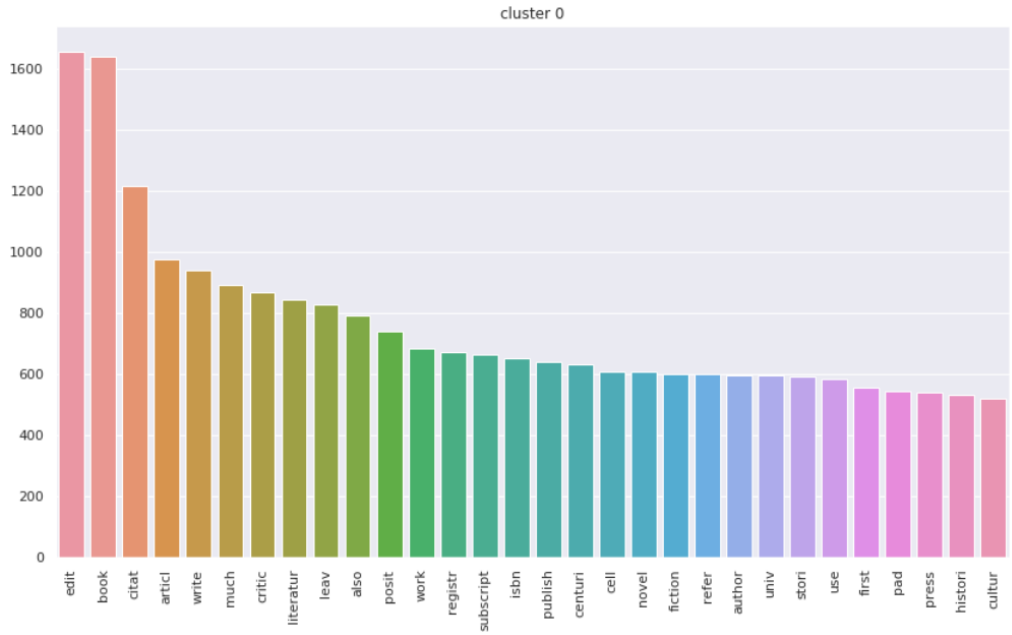


Figura 4.32: Le 20 parole più frequenti all'interno del primo cluster

Figura 4.33: WordCloud delle 100 parole più frequenti nel primo cluster

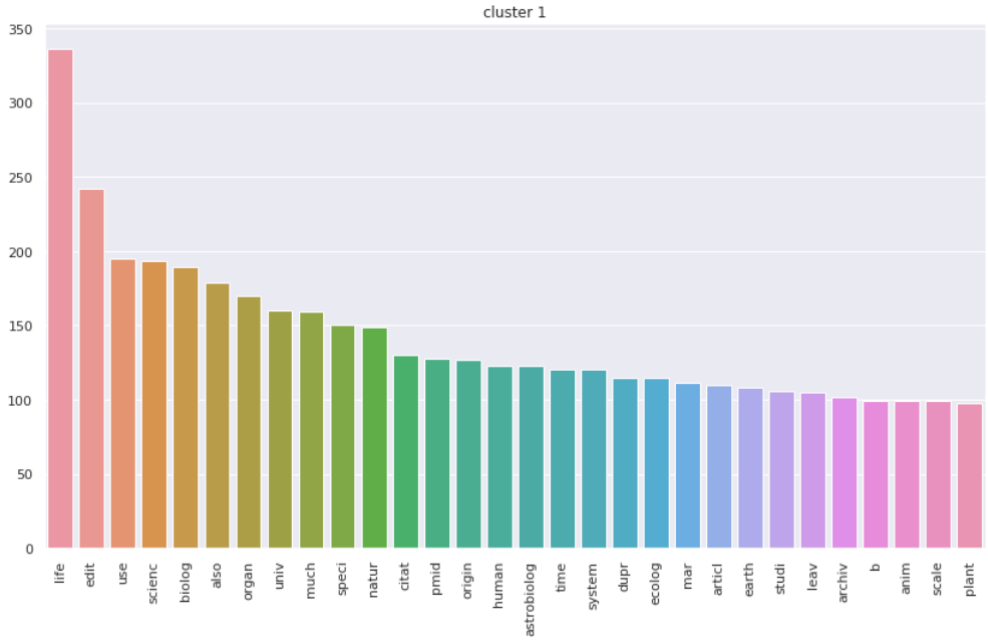


Figura 4.34: Le 20 parole più frequenti all'interno del secondo cluster

Figura 4.35: WordCloud delle 100 parole più frequenti nel secondo cluster

Nel primo cluster troviamo parole inerenti alla scienza e alla biologia mentre nel secondo a tematiche di letteratura e al mondo dell'editoria. Quindi si identifica un cluster con parole nuove, che prima non comparivano, ovvero il secondo.

Conclusioni: Dall'analisi degli outliers dei test emerge che all'aumentare della percentuali dei dati nuovi si definisce all'interno degli outliers di una classe che diviene sempre più coesa man mano che aumentano i dati nuovi. Ciò si può constatare sia visivamente nei plot realizzati che numericamente nell'aumento del valore della silhouette per $k = 2$. Inoltre notiamo che nel nuovo cluster compaiono parole inerenti a tematiche che non comparivano nei dati di partenza.

4.3.4 Esperimento con gli articoli di Biologia e Tecnologia

In questo esperimento consideriamo i 1000 articoli di Biologia e i 1000 di Tecnologia.

Trasformazione dei documenti Si definisce n = 300 il numero il numero delle parole più comuni che si vogliono selezionare per ognuno dei due gruppi di dati, in quanto si è visto che è il numero di parole ottimale per identificare la classe dei dati storici.
Tramite la funzione **get_most_common**(lista_testi, n) si ottengono le 300 parole più comuni in **lista_testi_0** e in **lista_testi_1** e si salvano rispettivamente nei vettori **most_common_0** e **most_common_1**.

Con la funzione **weight_calc_improved**(lista_testi, word_embedding, most_common) si ottiene la trasformazione di ogni documento in un vettore numerico.

Visualizzazione dei dati

Visualizziamo i dati tramite le tecniche t_SNE e PCA. Notiamo che anche in questo caso i dati storici e nuovi sembrano occupare zone diverse dello spazio.

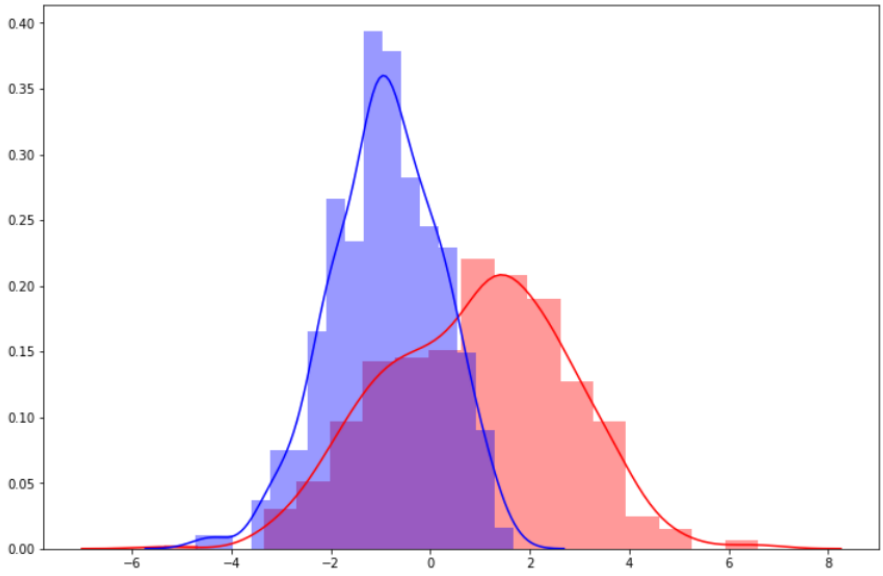


Figura 4.36: Plot delle prime due componenti principali dei dati. In particolare in rosso i dati storici e in blu i dati nuovi

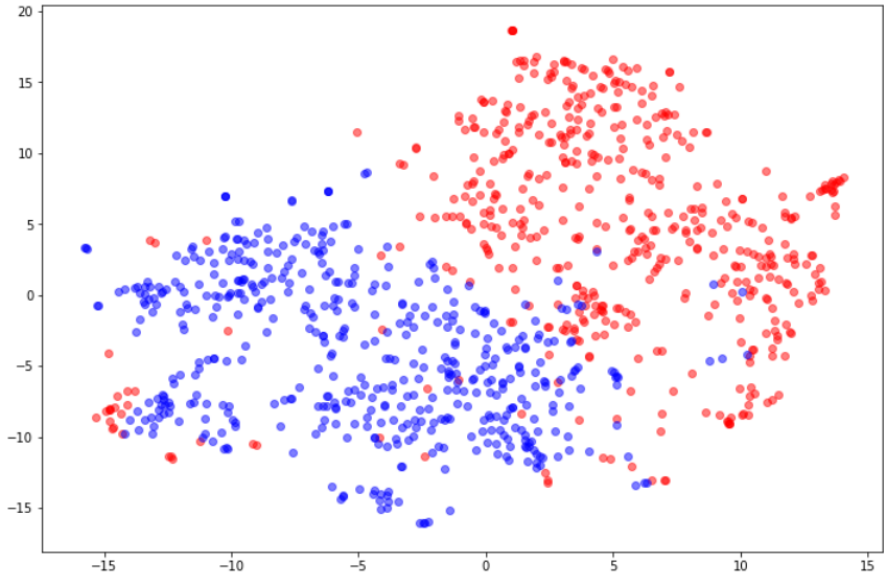


Figura 4.37: Plot della TSNE a due componenti. In particolare in rosso i dati storici e in blu i dati nuovi

Train dell'Elliptic Envelope

I dati storici di **lista_testi_0** vengono divisi in due parti, la prima meta' verrà utilizzata nella fase di training e mentre la seconda nella fase di test. Si addestra quindi l'Elliptic Envelope sulla prima parte di dati e si testa sulla seconda per verificare l'efficacia dell'algoritmo.

Creazione delle finestre di dati di test

Come nell'esperimento precedente vengono create finestre di test di dimensione variabile e con percentuale crescente di dati nuovi. Anche in questo caso notiamo che all'aumentare della percentuale di dati nuovi il numero di outliers riconosciuti aumenta, passando nei testi più numerosi (quelli corrispondenti all'ultima colonna) da percentuali di outliers del 15% a percentuali dell' 83%.

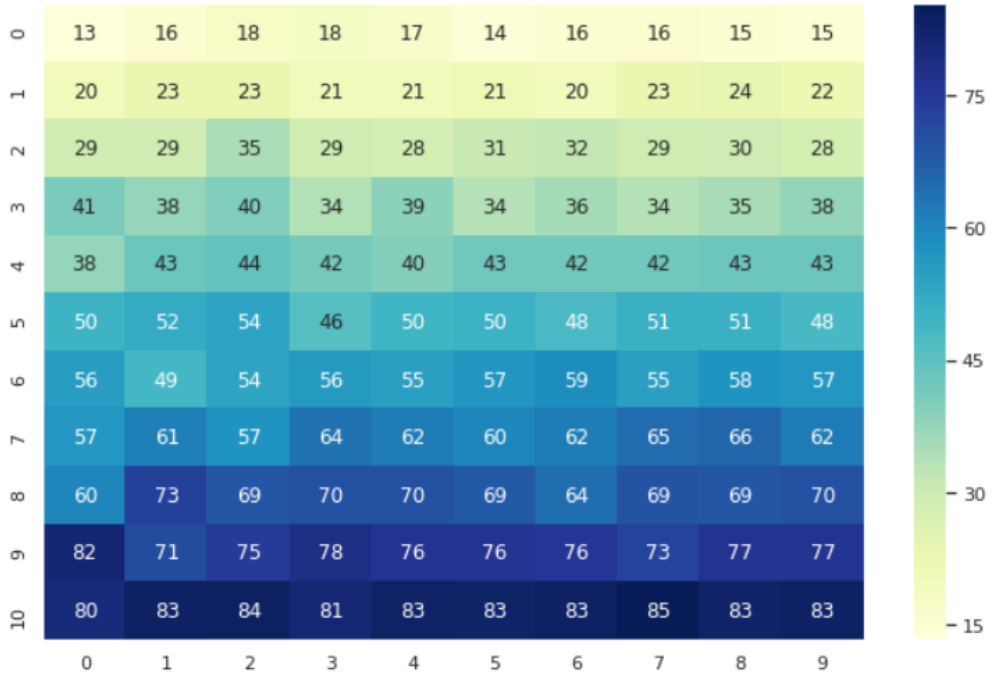


Figura 4.38: Percentuali di outliers riconosciute. Ogni elemento della tabella corrisponde al risultato su un test differente. Ogni colonna corrisponde ad una diversa dimensione della finestra di test e andando verso il basso si ha una percentuale sempre maggiore di dati nuovi, passando dallo 0% al 100%

Analisi degli outliers dei dati iniziali

Consideriamo la parte dei dati iniziali che non sono stati utilizzati per il train dell'Elliptic Envelope e testiamo il modello di outliers detection su 250 diquesti dati. Salviamo nella lista **X_0** gli outliers ottenuti e applichiamo il clustering gerarchico su questi ultimi. Detto **k** il numero di clusters si itera il procedimento con **k** =1,...,8 e si calcola il valore della silhouette per ogni **k**.
Notiamo che il valori della silhouette è basso per $k = 2$ e dal plot notiamo che i dati sono sparsi.

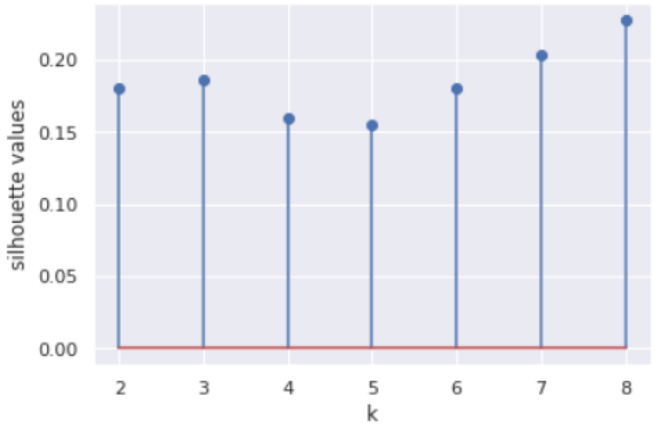


Figura 4.39: Valori della silhouette al variare di k

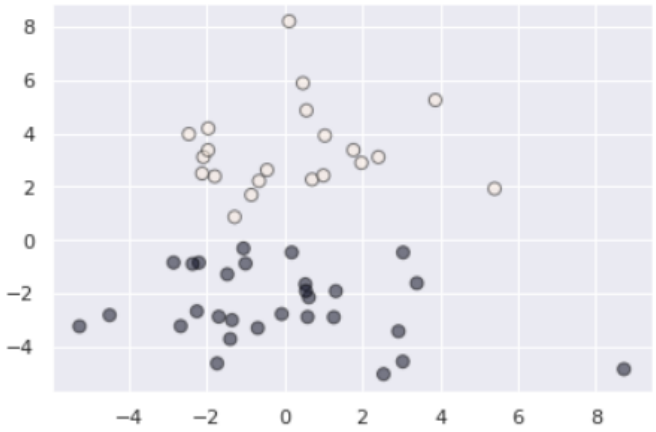


Figura 4.40: Visualizzazione mediante PCA dei cluster ottenuti. In nero gli elementi assegnati al primo cluster e in bianco quelli assegnati al secondo.

Le parole più frequenti dei due cluster dei dati con argomento "Biologia" potrebbero variare rispetto all'esperimento precedente in quanto questa volta per realizzare la trasformazione dei documenti testuali sono state prese $n = 300$ parole. Poniamo $k = 2$ e analizziamo quindi le parole più frequenti dei primi due clusters.

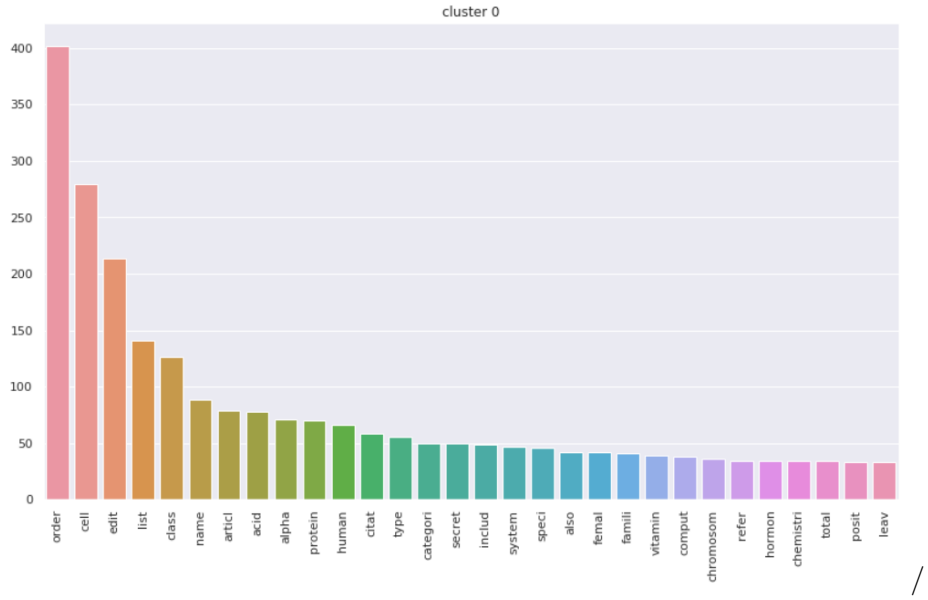


Figura 4.41: Le 20 parole più frequenti nel primo cluster

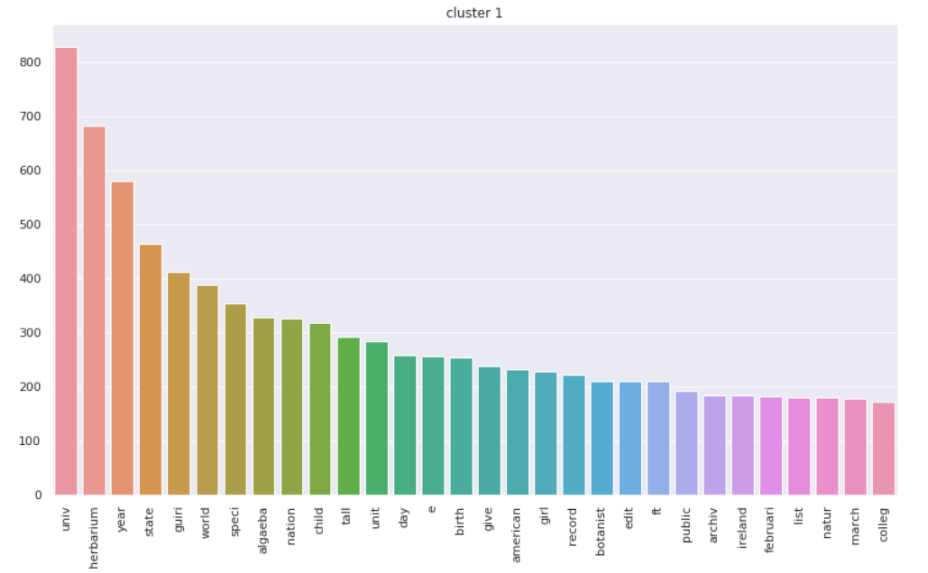


Figura 4.42: Le 20 parole più frequenti nel secondo cluster

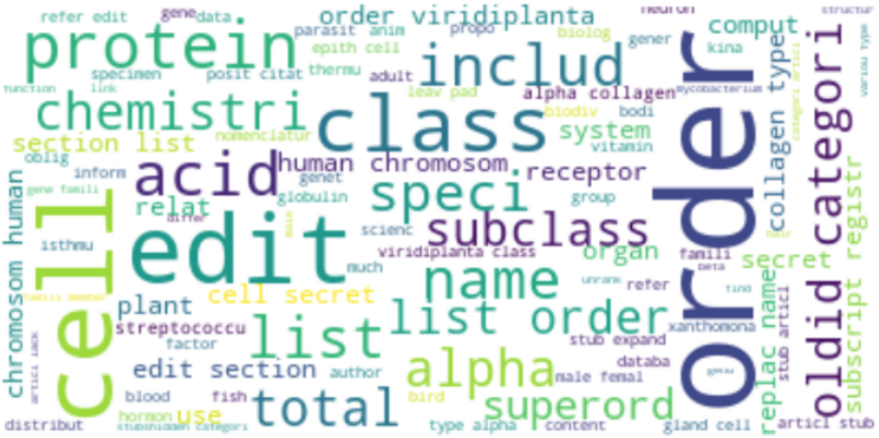


Figura 4.43: WordCloud delle 100 parole più frequenti nel primo cluster

Figura 4.44: WordCloud delle 100 parole più frequenti del secondo cluster

In questo caso nel primo cluster compaiono termini più specifici e tecnici di chimica e biologia, mentre nel secondo parole più generiche relative al mondo della ricerca.

Analisi degli outliers delle finestre di test

Consideriamo la parte dei dati storici che non sono stati utilizzati nè per il train dell'Elliptic Envelope nè per lo studio degli outliers della classe iniziale nel passaggio precedente (ovvero gli ultimi 250 articoli con argomento biologia). Come nei casi precedenti creiamo 11 finestre di test di dimensione 250 con percentuale via via crescente di documenti con argomento "Tecnologia", dallo 0 al 100% e attuiamo lo

stesso procedimento degli altri esperimenti eseguiti. Notiamo che all'aumentare della percentuale dei dati nuovi si identifica un cluster sempre più coeso, come mostrano i grafici delle PCA e i valori della silhouette crescenti per $k = 2$.

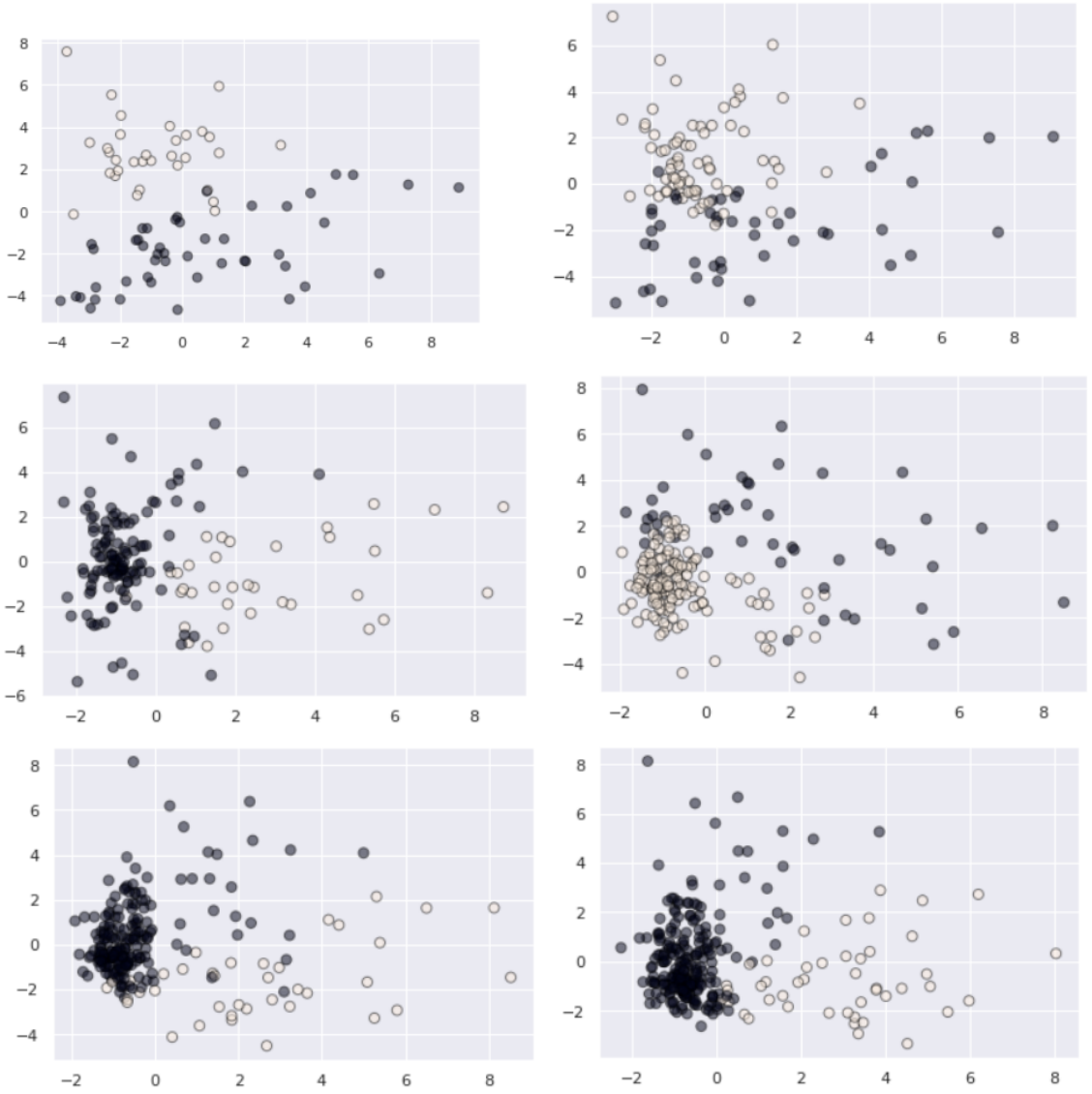


Figura 4.45: Visualizzazione tramite PCA dei cluster ottenuti nei vari test. In nero gli elementi del primo cluster e in bianco quelli del secondo.

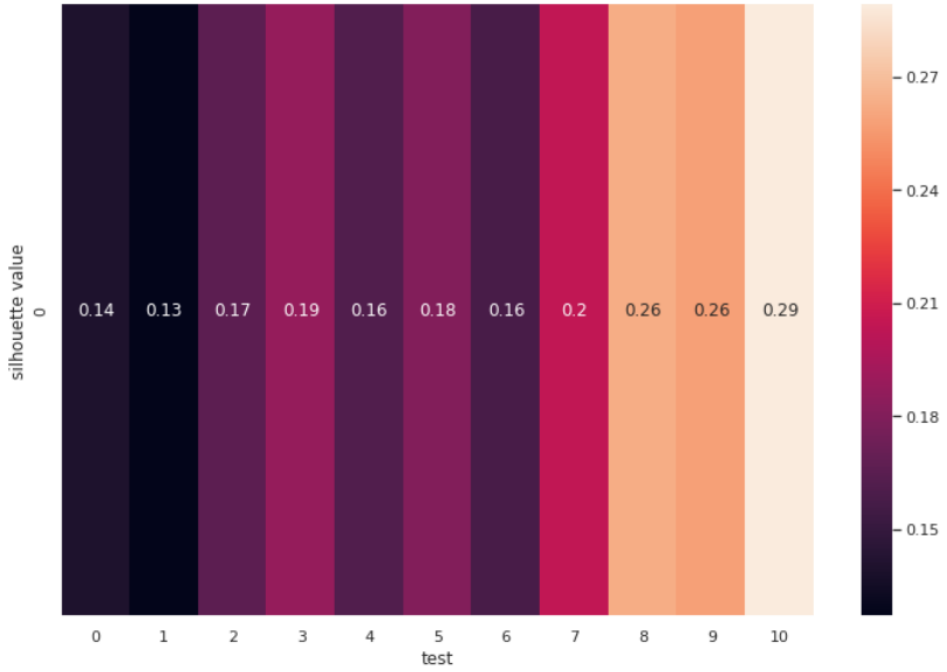


Figura 4.46: Valori della Silhouette per k =2 all'incrementare della percentuale dei dati nuovi all'interno delle finestre di test

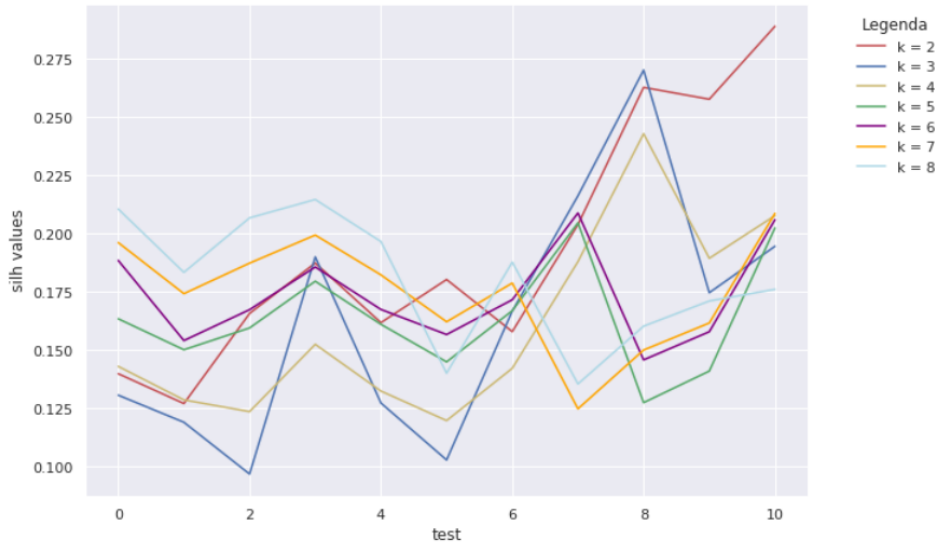


Figura 4.47: Valori delle Silhouettes al variare di k nei vari test

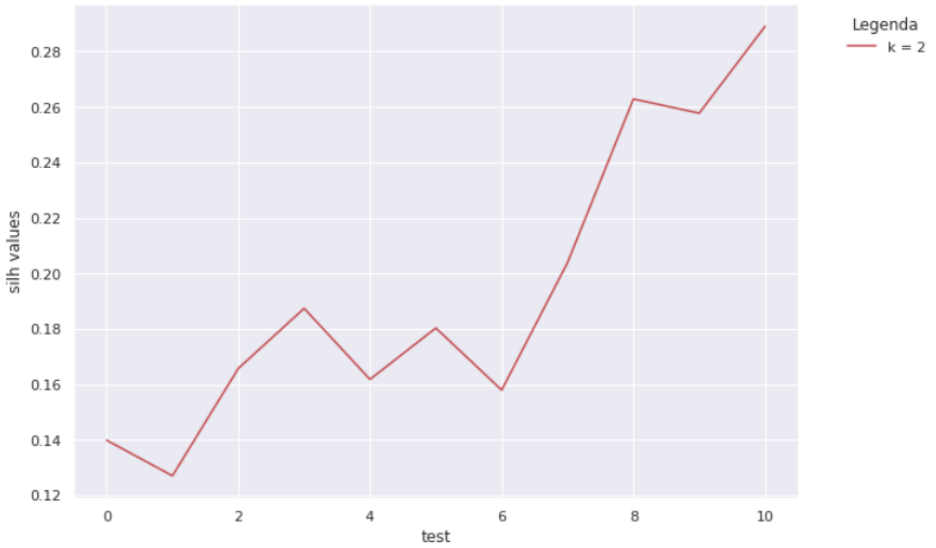


Figura 4.48: Valori della Silhouette per k =2 all'incrementare della percentuale dei dati nuovi all'interno delle finestre di test

Fissiamo $k = 2$ e analizziamo il contenuto dei due cluster individuati considerando le 20 parole più frequenti al loro interno e dei WordCloud che essi generano.

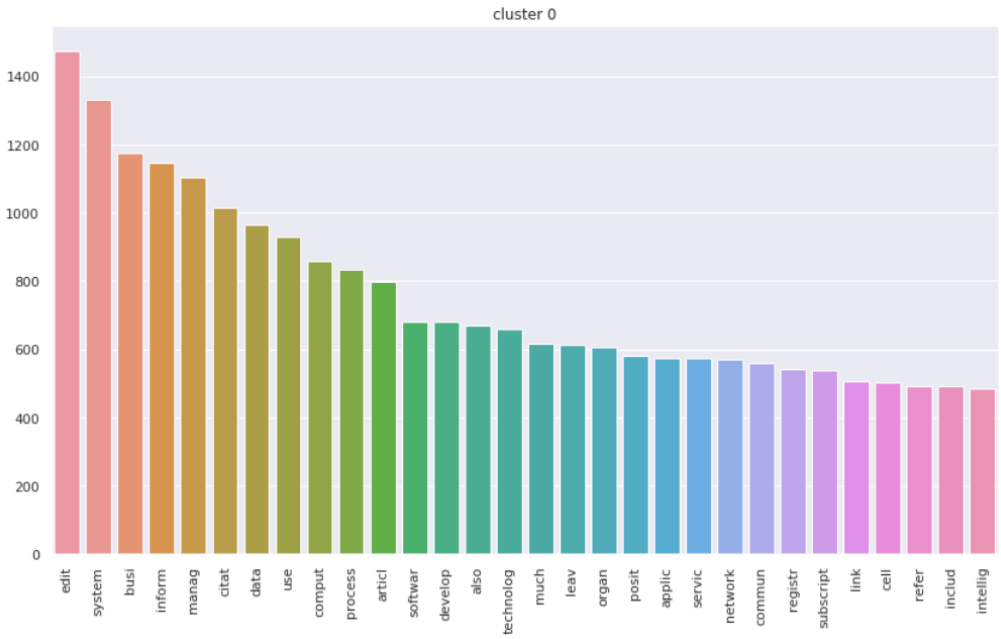


Figura 4.49: Le 20 parole più frequenti all'interno del primo cluster

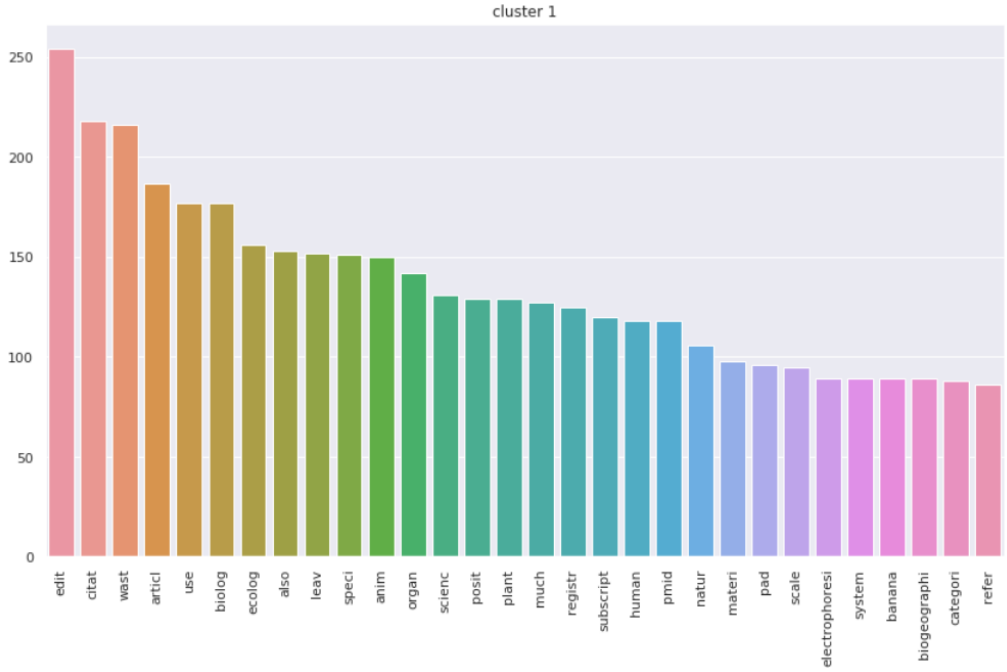


Figura 4.50: Le 20 parole più frequenti all'interno del secondo cluster

Figura 4.51: WordCloud delle 100 parole più frequenti nel primo cluster

Figura 4.52: WordCloud delle 100 parole più frequenti nel secondo cluster

Si identificano quindi due cluster, uno con parole inerenti alla tecnologia e all'informatica e l'altro con parole inerenti alla scienza e alla biologia. Si è quindi creato un cluster con parole nuove, che prima non comparivano, ovvero il primo.

4.3.5 Conclusioni:

Dall'analisi degli outliers dei test emerge che all'aumentare della percentuali dei dati nuovi si definisce all'interno degli outliers un cluster che diventa sempre più coeso all'aumentare della percentuale di dati nuovi nelle finestre di test. Ciò si può constatare sia visivamente, nei plot realizzati, che numericamente nell'aumento del valore della silhouette per $k = 2$. Questo nuovo cluster contiene parole nuove, che rimandano a concetti di tecnologia e innovazione e che non comparivano all'interno dei cluster dei dati storici. Ciò fa pensare alla presenza di un argomento diverso nei dati nuovi, rispetto a quello dei dati storici.
In questi esperimenti sui dati relativi agli articoli di Wikipedia, si e' visto il formarsi di una classe nuova all'aumentare dei dati nuovi inseriti. Tuttavia la scarsa quantità dei dati determina degli andamenti irregolari nella crescita del valore della silhouette.

4.4 Recensioni di film

Il dataset che consideriamo in questo esperimento è costituito da 25000 recensioni di film in lingua inglese, delle quali 12500 positive e 12500 negative. Il procedimento illustrato nel capitolo 2 viene adattato al caso di questi dati. A differenza dei casi precedenti, in cui la classe dei dati storici era caratterizzata da un unico argomento

comune a tutti i documenti, in questo caso il classificatore dovrà essere in grado di identificare la classe delle recensioni positive, e quindi imparare a distinguere l'accezione positiva o negativa che un testo può avere. A tale scopo sono state utilizzate alcune delle tecniche proprie della Sentiment Analysis che consentono di identificare frasi con accezione negativa.

4.4.1 Data Cleaning

Alle fasi di Data Cleaning descritte precedentemente si applicano alcune modifiche nella definizione delle stopwords e per l'identificazione di frasi negative.

Definizione stopwords: A quelle che si considerano solitamente per la lingua inglese togliamo i costrutti utilizzati per le negazioni, (ad esempio not, don't, didn't...) e aggiungiamo le parole legate al contesto (actor, film, movie, play, watch, show...).

Caratterizzazione frasi negative: Questo procedimento consiste in due fasi:

1. Ogni testo viene diviso in frasi. Si considera frase l'insieme di parole che intercorrono tra due segni di punteggiatura successivi.
2. Se all'interno di una frase è presente una negazione allora tutte le parole successive alla negazione verranno marcate col suffisso "_neg".

Questo procedimento è molto utile per l'identificazione dell'accezione di un testo. In un primo approccio si è provato a caratterizzare la positività o la negatività sulla base di insiemi di parole e aggettivi con significato positivo o negativo. Questo approccio non è risultato molto efficace in quanto all'interno di una frase parole con significato assoluto positivo possono essere utilizzate con accezione negativa e viceversa. Se ad esempio consideriamo la frase "the film was not bad", osserviamo la presenza dell'aggettivo "bad" con significato assoluto negativo, tuttavia la frase nel complesso ha un'accezione positiva.

I seguenti passaggi restano invariati:

- rimozione caratteri speciali
- trasformazione aggettivi superlativi
- selezione di nomi, verbi, avverbi, aggettivi
- riduzione delle parole alla loro radice

Le recensioni positive, dopo la pulizia verranno salvate nella lista **lista_testi_0**, mentre quelle negative nella lista **lista_testi_1**.

Testo della recensione:
"I went and saw this movie last night after being coaxed to by a few friends of mine. I'll admit that I was reluctant to see it because from what I knew of Ashton Kutcher he was only able to do comedy. I was wrong. Kutcher played the character of Jake Fischer very well, and Kevin Costner played Ben Randall with such professionalism. The sign of a good movie is that it can toy with our emotions. This one did exactly that. The entire theater (which was sold out) was overcome by laughter during the first half of the movie, and were moved to tears during the second half. While exiting the theater I not only saw many women in tears, but many full grown men as well, trying desperately not to let anyone see them crying. This movie was great, and I suggest that you go see it before you judge."

Risultato del data cleaning:
'gò', 'saw', 'last', 'night', 'coax', 'friend', 'minè, 'admit', 'reluct', 'know', 'ashton', 'kutcher', 'abl', 'comedi', 'wrong', 'kutcher', 'jakè, 'fischer', 'well', 'kevin', 'costner', 'randal', 'profession', 'sign', 'good', 'toy', 'emot', 'exactli', 'entir', 'theater', 'sell', 'overcom', 'laughter', 'first', 'movè, 'tear', 'second', 'half', 'exit', 'theater', 'not', 'saw_neg', 'many_neg', 'women_neg', 'tears_neg', '_neg', 'many_neg', 'full_neg', 'grown_neg', 'men_neg', 'well_neg', '_neg', 'trying_neg', 'desperately_neg', 'not_neg', 'let_neg', 'anyone_neg', 'crying_neg', 'great', 'sugg', 'gò, 'judg'

4.4.2 Creazione del Word Embedding

In questo caso si è scelto di non utilizzare un word embedding già addestrato, ma di ricavarlo addestrando il WordtoVec sui dati storici, ovvero le recensioni positive.

4.4.3 Trasformazione delle recensioni

Si definisce n = 100 il numero il numero delle parole più comuni che si vogliono selezionare per ognuno dei due gruppi di dati, in quanto si è visto che è il numero di parole ottimale per identificare la classe dei dati storici.
Tramite la funzione **get_most_common**(lista_testi, n) si ottengono le 100 parole più comuni in **lista_testi_0** e in **lista_testi_1** e si salvano rispettivamente nei vettori **most_common_0** e **most_common_1**.

Con la funzione **weight_calc_improved**(lista_testi, word_embedding, most_common) si ottiene la trasformazione di ogni documento in un vettore numerico.

4.4.4 Visualizzazione dei dati

Visualizziamo i dati tramite le tecniche t_SNE e PCA. Notiamo che in questo caso i dati storici e nuovi sembrano molto mescolati e non sembrano occupare zone distinte dello spazio.

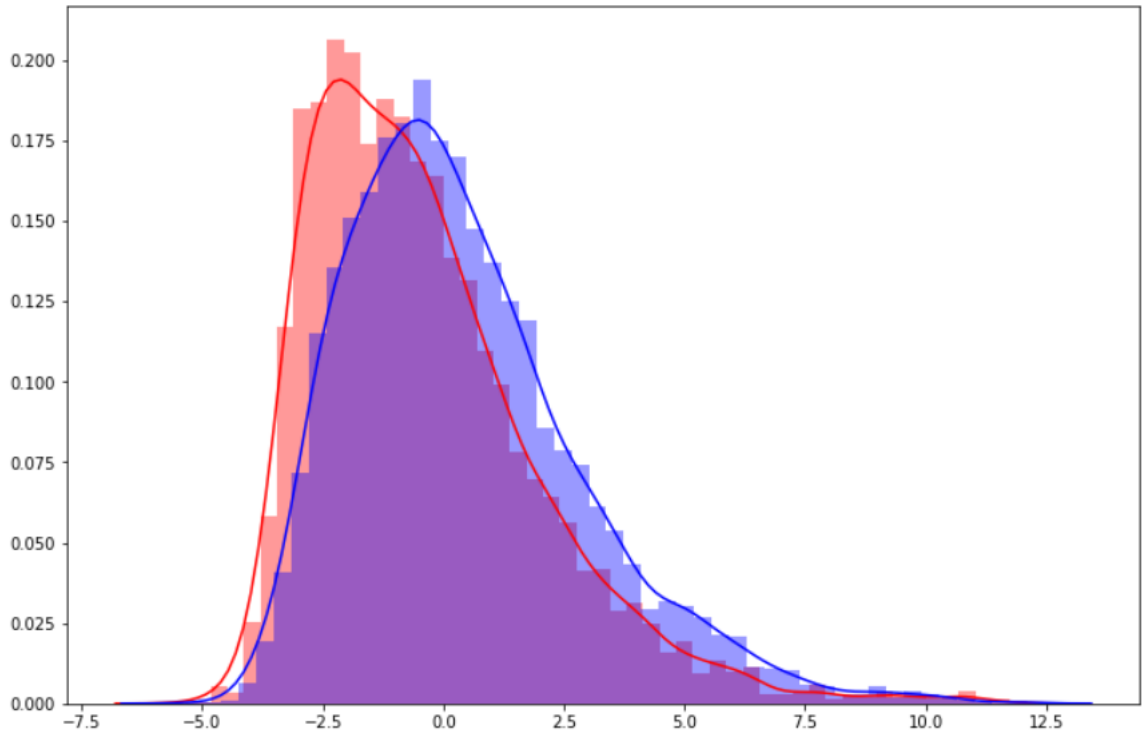


Figura 4.53: Plot delle prime due componenti principali dei dati. In particolare in rosso i dati storici e in blu i dati nuovi

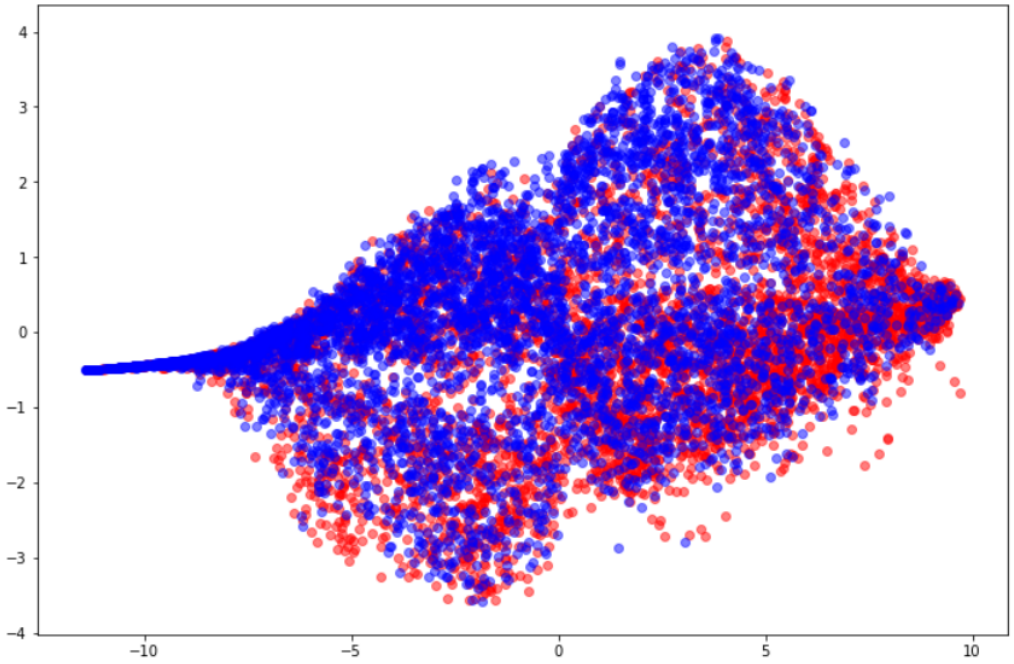


Figura 4.54: Plot della TSNE a due componenti. In particolare in rosso i dati storici e in blu i dati nuovi

4.4.5 Train dell'Elliptic Envelope

I dati di **lista_testi_0** vengono divisi in due parti. I primi 4000 dati verranno utilizzati nella fase di training e i restanti nella fase di test. Si addestra quindi l'Elliptic Envelope sulla prima parte di dati e si testa sulla seconda per verificare l'efficacia dell'algoritmo.

4.4.6 Creazione delle finestre di dati di test

Come negli esperimenti precedenti vengono create finestre di test di dimensione variabile e con percentuale crescente di dati nuovi. Per la creazione delle finestre su cui testare il modello si pone **max_size** = 4000. Si ottengono quindi finestre di dimensione:
window_size: 0.1 · **max_size** = 50, 0.2 · **max_size** = 100, ..., **max_size** = **4000**.
Per ognuna di queste dimensioni si ottengono 11 pacchetti di dati con un numero crescente di dati nuovi: 0 , 0.1 · **window_size** , 0.2 · **window_size** ..., **window_size**.

In totale avremo quindi 9 * 11 blocchi di dati e si testa l'Elliptic Envelope su ognuna di essi .

Anche in questo caso notiamo che all'aumentare della percentuale di dati nuovi il numero di outliers riconosciuti aumenta, passando nei testi più numerosi (quelli corrispondenti all'ultima colonna) da percentuali di outliers tra il 10 e l'11% a percentuali 90%.

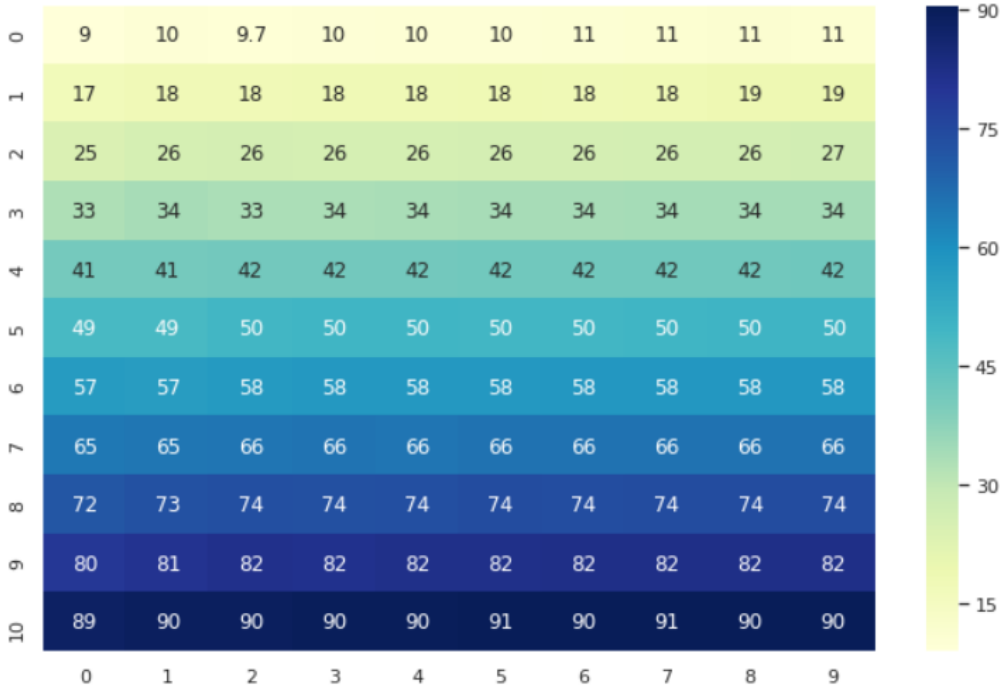


Figura 4.55:

4.4.7 Analisi degli Outliers per la Drift Detection

In questa fase si cercherà di identificare il drift dei dati. In particolare si vuole verificare se tra gli outliers dei dati nuovi si creano dei cluster o dei raggruppamenti che non erano presenti nei dati originari, e che quindi essi possono essere identificati da una nuova classe.

Analisi degli outliers dei dati iniziali

Consideriamo la parte dei dati iniziali che non sono stati utilizzati per il train dell'Elliptic Envelope e testiamo il modello di outliers detection su 4000 di questi

dati.
Salviamo nella lista **X_0** gli outliers ottenuti e applichiamo il clustering gerarchico su questi ultimi. Detto **k** il numero di clusters si itera il procedimento con **k** =1,...,8. Si calcola il valore della silhouette per ogni **k** e si analizza.
Notiamo che i valori della silhouette sono tutti molto bassi, quindi non sembrano identificarsi dei cluster coesi all'interno degli outliers.

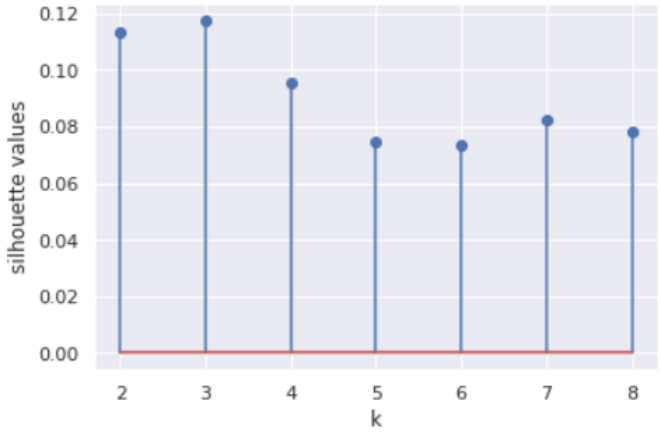


Figura 4.56: Valori della silhouette al variare di k

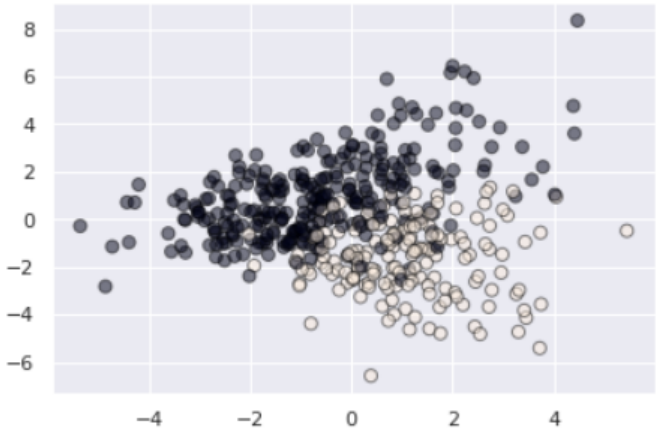


Figura 4.57: Visualizzazione mediante PCA dei cluster ottenuti. In nero gli elementi assegnati al primo cluster e in bianco quelli assegnati al secondo.

Poniamo $k = 2$ e analizziamo il contenuto dei cluster creati.

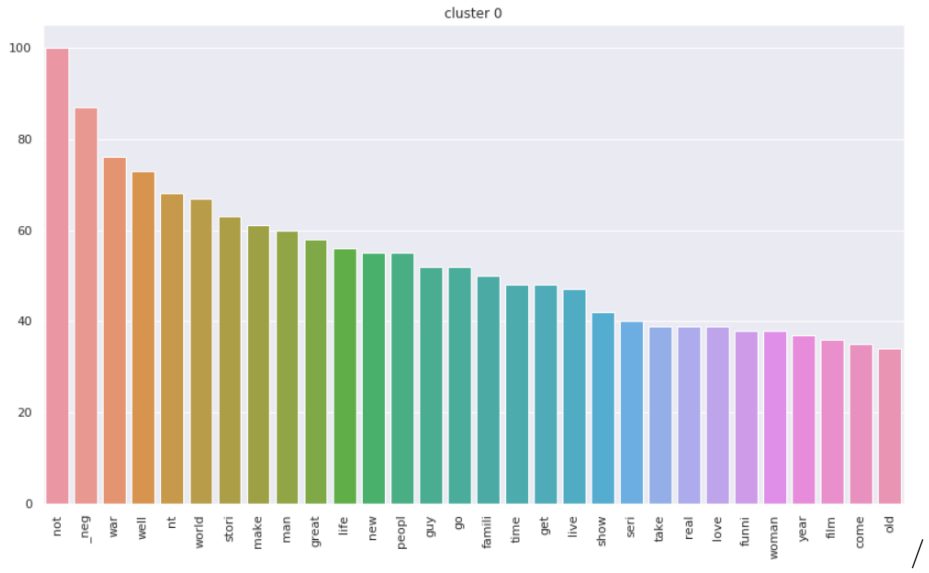


Figura 4.58: Le 20 parole più frequenti nel primo cluster

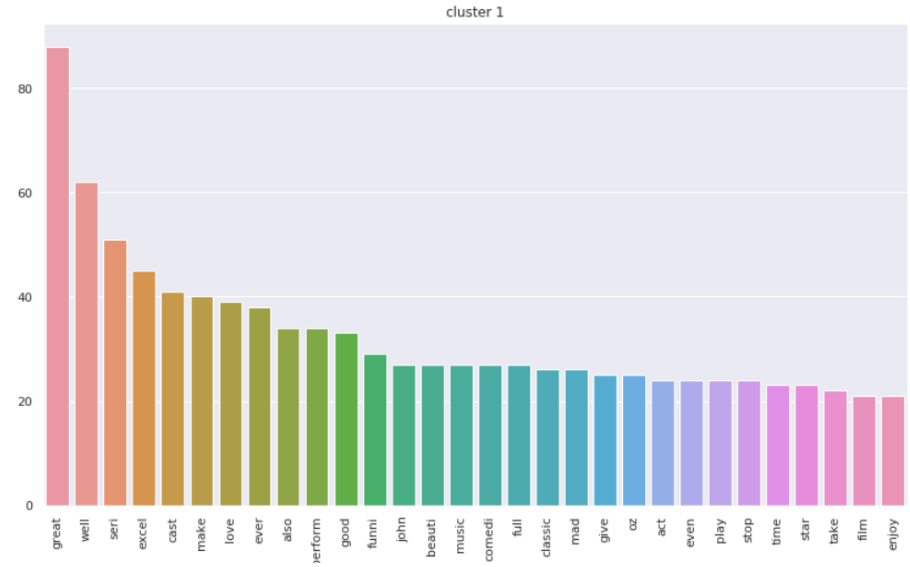


Figura 4.59: Le 20 parole più frequenti nel secondo cluster

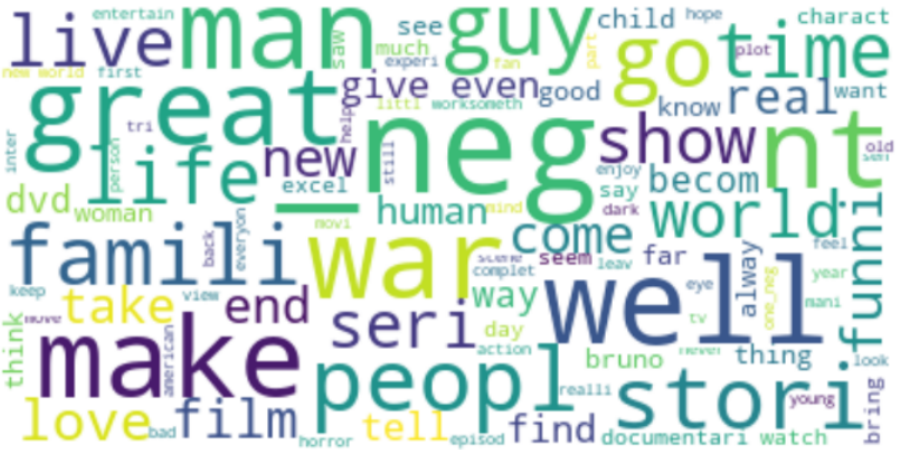


Figura 4.60: WordCloud delle 100 parole più frequenti nel primo cluster

Figura 4.61: WordCloud delle 100 parole più frequenti del secondo cluster

In questo caso non riusciamo a distinguere una differenza di argomenti nei due cluster creati, ma notiamo che nel secondo compaiono aggettivi positivi come "funny", "excellent", "well" che invece non compaiono nel primo.

Analisi degli outliers delle finestre di test

Si creano 11 finestre di dati di test per analizzare i cluster all'aumetare della percentuale di dati nuovi. Le finestre ottenute contengono tutte 4000 recensioni e una percentuale di elementi appartenenti ai dati nuovi via via crescente, dallo 0% al 100%. Notiamo che al variare di all'aumentare di recensioni negative all'interno

delle finestre di test i valori della silhouette aumentano di poco per $k = 2$ mentre restano più o meno costanti gli altri valori di k.

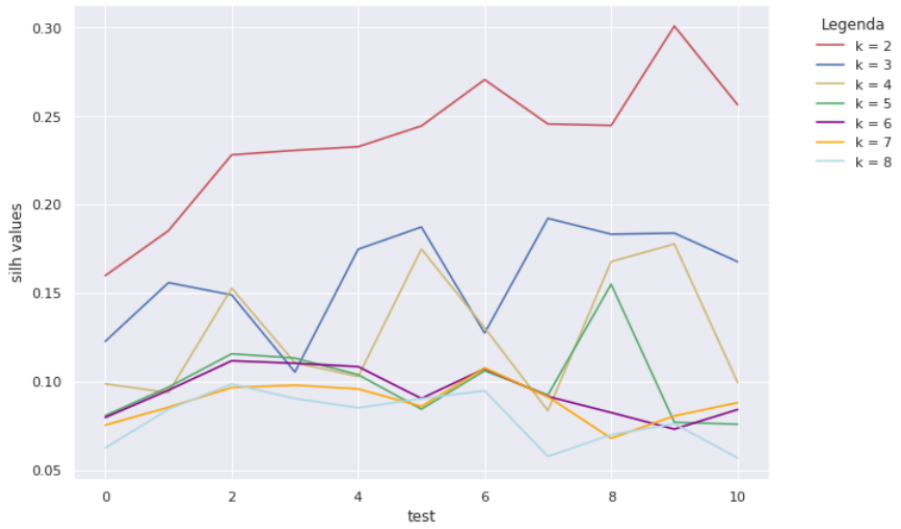


Figura 4.62: Valori delle Silhouettes al variare di k nei vari test

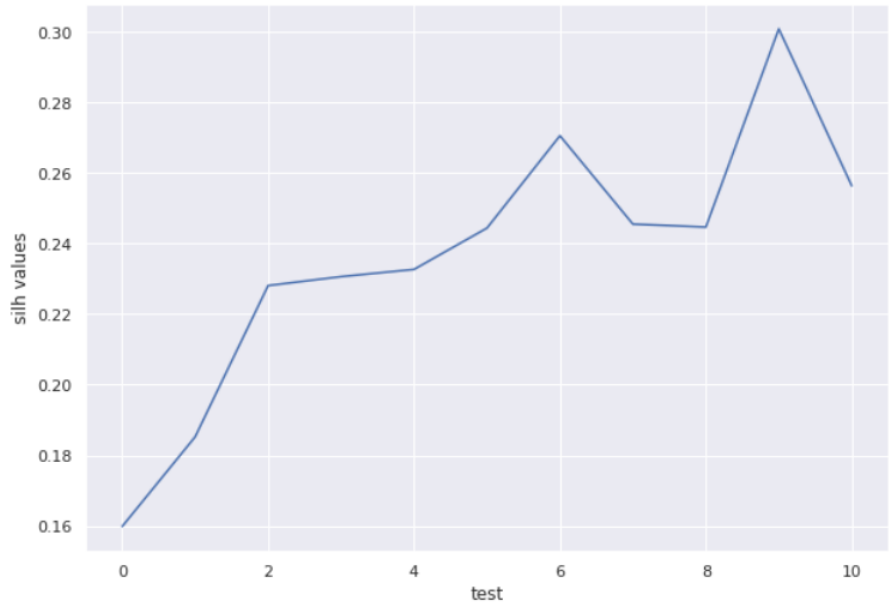


Figura 4.63: Valori della Silhouette per k =2 all'incrementare della percentuale dei dati nuovi all'interno delle finestre di test

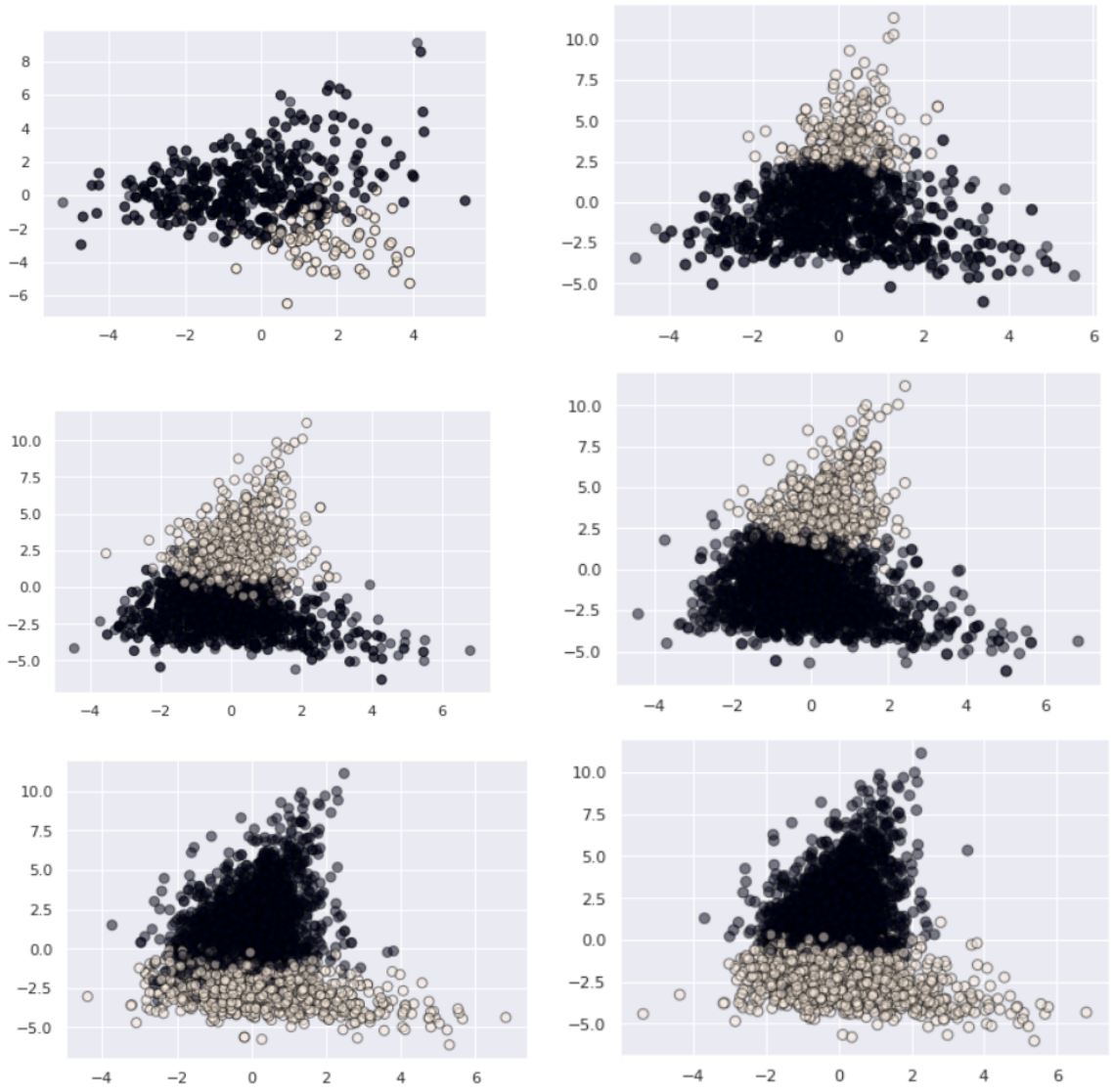


Figura 4.64: Visualizzazione tramite PCA dei cluster ottenuti nei vari test. In nero gli elementi del primo cluster e in bianco quelli del secondo.

Notiamo che dalla visualizzazione delle PCA i due cluster non sembrano molto definiti. Non si riesce a distinguere la presenza di una nuova classe rispetto ai dati di partenza.
Analizziamo il contenuto dei due cluster individuati considerando le parole più frequenti al loro interno.

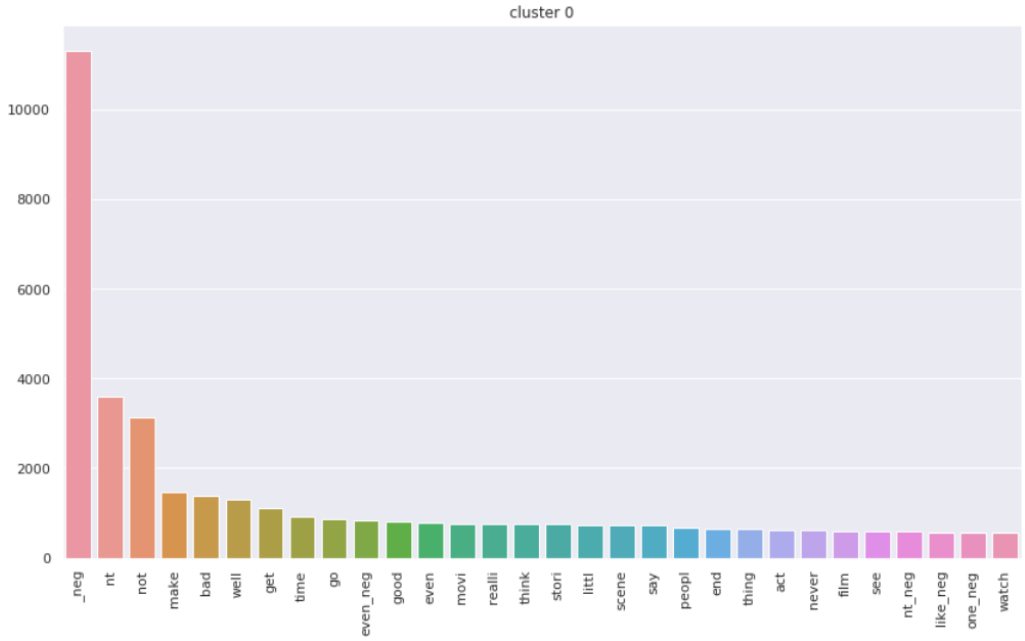


Figura 4.65: Le 20 parole più frequenti all'interno del primo cluster

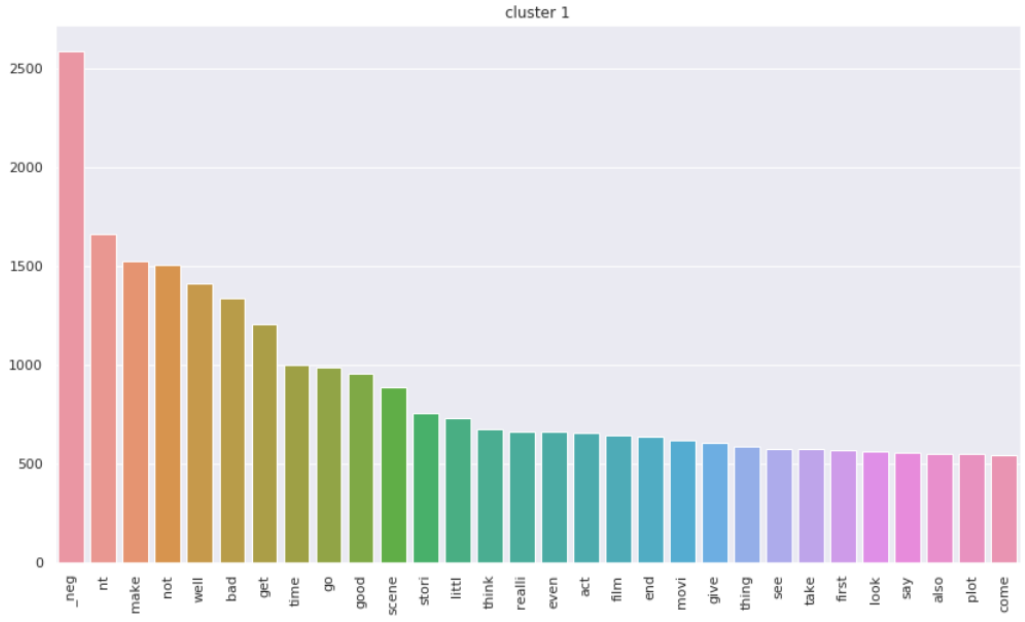


Figura 4.66: Le 20 parole più frequenti all'interno del secondo cluster

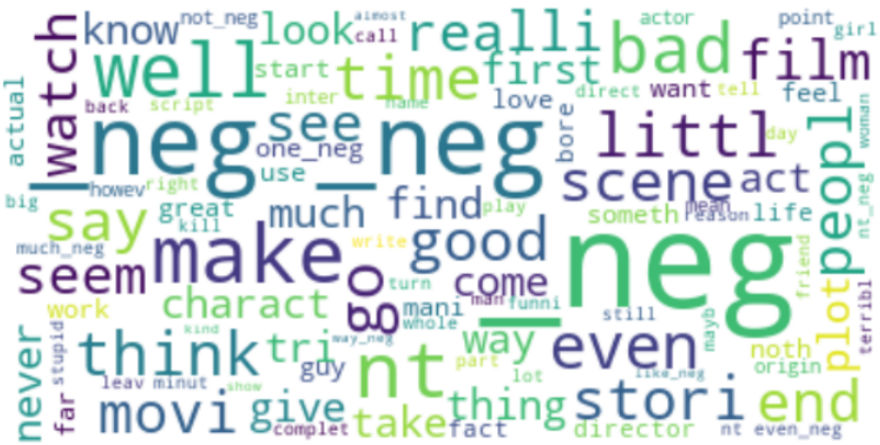


Figura 4.67: WordCloud delle 100 parole più frequenti nel primo cluster

Figura 4.68: WordCloud delle 100 parole più frequenti nel secondo cluster

Notiamo che nei due cluster finali non sono presenti parole molto diverse tra loro che possano far pensare a due diversi sottoargomenti.

4.4.8 Conclusioni:

In questo caso l'algoritmo di outliers detection è in grado di riconoscere che i nuovi dati non appartengono alla classe dei dati di partenza, tuttavia non si riescono ad individuare delle nuove classi all'interno del rumore identificato. Ciò potrebbe essere causa del tipo di trasformazione dei dati operata. Infatti tramite il WordtoVec si realizza una trasformazione delle parole in vettori numerici secondo cui parole che si trovano in contesti simili verranno associate a vettori che si trovano vicini tra loro.

In questo caso l'argomento trattato nelle recensioni positive e negative è lo stesso, pertanto i vettori che le rappresentano occuperanno posizioni vicine nello spazio e non si riusciranno ad identificare nuovi raggruppamenti.

Conclusioni

Dagli esperimenti eseguiti si evince che la metodologia proposta riesce ad identificare un nuovo cluster nel caso in cui quest'ultimo sia determinato dalla presenza di un nuovo argomento (inteso come un insieme di parole) nei dati nuovi, che nei dati storici non era presente, mentre non riesce ad identificare cluster di un'altra natura (ad esempio caratterizzati, come nell'esempio visto, da un'accezione semantica diversa da quella dei dati iniziali). Questo limite potrebbe essere dovuto alla tipologia di trasformazione delle parole eseguita. Infatti tramite il Word2vec le parole vengono trasformate in vettori disposti nello spazio in modo che parole che si trovano spesso in contesti simili occuperanno vettori vicini tra di loro. Trasformazioni diverse potrebbe mettere in luce cluster di tipo diverso rispetto a quelli determinati dalla presenza di un nuovo argomento.
Sarebbe anche interessante testare il metodo su tipi di dati diversi, non testuali. Questa metologia considera un classificatore one-class, quindi affinchè funzioni i dati storici devono appartenere tutti ad una singola classe. Sviluppi futuri potrebbero pertanto analizzare il comportamento di classificatori multiclasse all'ingresso di nuovi dati che presentano drift.
Un altro limite di questo metodo è l'incapacità di classificare l'appartenenza o meno alla classe iniziale di un singolo nuovo dato, senza che esso faccia parte di un gruppo di dati nuovi. Infatti per la trasformazione dei dati nuovi è necessaria l'estrazione delle n parole più frequenti all'interno dell'intero corpus.
Come indice quantitativo di valutazione della coesione dei cluster trovati è stata utilizzata la silhouette. Visivamente le PCA dei dati e i word cloud ci aiutano a visualizzare la struttura dei cluster trovati e il loro contenuto. Sviluppi futuri del metodo potrebbero focalizzarsi sulla ricerca di nuove metriche per l'identificazione della nuova classe che valutino quantitativamente, non solo la coesione dei cluster identificati, ma anche la dissimilarità rispetto ai cluster dei dati storici.

Ringraziamenti

Vorrei ringraziare tutti coloro che mi hanno supportato e aiutato nella realizzazione di questa tesi. In particolare la Prof.ssa Tania Cerquitelli per il supporto fornitomi e il dottorando Riccardo Callà per la disponibilità e la gentilezza con cui mi ha seguita. Inoltre ringrazio gli amici Deborah Dore, Marcello Lanciani e Gabriele Perrone per le consulenze e l'aiuto che mi hanno offerto.

Bibliografia

[1] J. Han, M. Kamber, e J. Pei, *Data Mining Concepts and Techniques*. 225 Wyman Street,Waltham, MA 02451 USA: Morgan Kaufmann, 2012.

[2] P.-N. Tan, M. Steinbach, e V. Kumar, *Introduction to Data Mining*. Pearson Addison-Wesley, 2006.

[3] T. Hastie, R. Tibshirani, e J. Friedman, *The Elements of Statistical Learning, Data Mining, Inference, and Prediction*. Springer, 2001.

[4] E. Cambria, B. Schuler, Y. Xia, e C. Havasi, "New avenues in opinion mining and sentiment analysis," *IEEE Intelligent Systems*, volume 28, numero 2, March-April 2013.

[5] G. Ditzler, M. Roveri, C. Alippi, e R. Polikar, "Learning in nonstationary environments: A survey," *IEEE Computational Intelligence Magazine*, 2015.

[6] I. Zliobaite, "Learning under concept drift: an overview," *Vilnius University*, volume 41, numero 3, 22 Oct 2010.

[7] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, e G. Zhang, "Learning under concept drift: A review," *IEEE Transactions on Knowledge and Data Engineering*, volume 31, numero 12, 13 Apr 2020.

[8] J. A. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, e A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, volume 1, numero 2, 2013.

[9] T. Mikolov, K. Chen, G. Corrado, e J. Dean, "Efficient estimation of word representations in vector space," 2013.

[10] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, e J. Dean, "Distributed representations of words and phrases and their compositionality," 2013.

[11] L. van der Maaten e G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, 2008.

[12] F. T. Liu, K. M. Ting, e Z.-H. Zhou, "Isolation forest," *Eighth IEEE International Conference on Data Mining*, 2008.

[13] P. J. Rousseeuw e K. V. Driessen, "A fast algorithm for the minimum covariance determinant estimator," *Technometrics*, 1999.

[14] M. Hubert e M. Debruyne, "Minimum covariance determinant," *John Wiley Sons*, 31 December 2009.

[15] P. J.Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, 1987.

[16] G. A. Miller, "Wordnet: A lexical database for english," *Communications of the ACM*, volume 38, numero 11, 1995.