Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Informatica

Tesi di Laurea Magistrale

Valutazione e miglioramento del sistema informativo produttivo di Automobili Lamborghini

Relatore

Prof. Antonio Vetrò

Candidato

Giacomo Cauda

Ottobre 2024

Sommario

In questa tesi vengono esaminati diversi aspetti del sistema informativo di produzione di Automobili Lamborghini, con particolare attenzione ai processi di assemblaggio nei due impianti principali: uno per i SUV e l'altro per le Super Sportive Car (SSC).
Entrambi gli impianti utilizzano un sistema automatizzato per verificare l'integrità e la correttezza delle etichette omologative sui componenti delle vetture. Queste etichette variano in base alla configurazione scelta dal cliente, al mercato di destinazione e alle normative vigenti (nel mercato di destinazione), implicando differenze anche per componenti identici.
Il sistema di controllo delle etichette utilizza una tecnologia avanzata di riconoscimento delle immagini per analizzare in tempo reale la conformità di ogni etichetta. La tesi presenta i risultati di un progetto di perfezionamento di questa applicazione, il progetto SurfDesk, analizzando i punti critici e i potenziali bias del modello matematico su cui si basa. Le analisi, condotte sui dati di produzione e sugli errori rilevati, sono state ottimizzate utilizzando piattaforme cloud, nello specifico la suite di applicazione di Amazon Web Services per migliorare l'efficienza dell'analisi.

Dopo la creazione dell'infrastruttura logica sul cloud, i dati di produzione raccolti sui modelli Urus e Revuelto fin dalla prima macchina prodotta sono stati copiati nel Datalake, normalizzati e analizzati. Per farlo, è stato necessario scrivere degli algoritmi appositi in Python, estraendo correlazioni di diverso tipo tra le prestazioni del sistema e determinate caratteristiche legate al processo di produzione del veicolo. Lo scopo è stato quello di trovare queste correlazioni al fine di intervenire per migliorare il sistema. Al termine dell'analisi, sono stati trovati molti piccoli fattori diversi che influenzano le prestazioni dell'algoritmo del sistema IR. Tuttavia, nessuno di questi è stato definito come la causa dei rallentamenti del sistema. Si è deciso

quindi di migliorare la formazione degli utilizzatori finali. Questo ha portato risultati concreti in termini di miglioramento delle prestazioni. Questo bias non sarebbe mai stato trovato senza l'impiego del progetto SurfDesk.

Indice

Capitolo 1

Il progetto SurfDesk

1.1 Il MES

L'ambiente di produzione di Lamborghini è interamente gestito da un avanzato sistema informativo chiamato MES (Manufacturing Execution System). Questo sistema registra e monitora tutte le operazioni eseguite durante le fasi di assemblaggio delle vetture Lamborghini, e non solo.

Il MES tiene traccia di tutte le operazioni di manifattura, inclusa la creazione di componenti specifici come i pezzi in fibra di carbonio, e il preassemblaggio. Quest'ultimo consiste nell'assemblare vari componenti che saranno successivamente installati sulla vettura, partendo dai singoli pezzi. Ad esempio, il preassemblaggio può includere il montaggio dei componenti su una portiera, che una volta completata, viene installata sulla vettura.

Inoltre, il MES permette di ottimizzare l'efficienza produttiva, garantendo che ogni fase del processo sia eseguita con precisione e qualità. Questo sistema non solo migliora la tracciabilità e la gestione delle operazioni, ma contribuisce anche a mantenere gli elevati standard di eccellenza per cui Lamborghini è rinomata.
Di seguito è riportato uno screenshot della pagina principale del MES. Come si può notare, sono presenti numerose piastrelle (tile) che, se cliccate, consentono di accedere a varie funzionalità del sistema MES.

Figura 1.1: Home page del MES

1.2 Image Recognition per le etichette Omologative

Il Sistema di Esecuzione della Produzione (MES) è organizzato in modo tale da includere diverse tile (blocchi logici) interattive. Queste tile sono cliccabili e selezionabili dagli operatori e sono associate alle varie stazioni di produzione. In ogni stazione, sono presenti funzioni specifiche che monitorano e registrano tutte le operazioni eseguite sui componenti o sui veicoli. Ad esempio, per ogni vite avvitata, viene registrata la coppia massima raggiunta dagli avvitatori, garantendo così un controllo preciso e dettagliato del processo produttivo. Oltre a queste tile funzionali, il MES offre una gamma di strumenti e applicazioni aggiuntive che possono essere utilizzate in qualsiasi stazione. Tra queste, spicca l'applicazione "Image Recognition Label", conosciuta dagli operatori come IR, che facilita il riconoscimento e la gestione delle etichette tramite il riconoscimento delle immagini.

1.2.1 Tracciabilità e conformità delle etichette omologative nei veicoli

L'obiettivo principale di questa applicazione è eseguire un controllo istantaneo e accurato sulle etichette omologative, che spesso sono complesse e ricche di dettagli difficili da verificare a occhio nudo. Questo sistema garantisce un'elevata precisione e rapidità nel controllo delle etichette. L'applicazione, accessibile dai tablet degli operatori, consente di scattare una foto delle etichette. L'immagine viene poi analizzata sul cloud di AWS, che estrae tutte le parole chiave presenti sull'etichetta. Queste parole chiave vengono confrontate con lo standard corretto per validare o meno l'etichetta. Se tutte le parole chiave sono corrette e corrispondono alla configurazione fisica del veicolo su cui è applicata l'etichetta, quest'ultima viene classificata come conforme. In caso contrario, ad esempio se una parola chiave è poco leggibile, errata o non corrisponde alla configurazione fisica del veicolo, il sistema rileva l'errore e notifica l'operatore, che dovrà sostituire l'etichetta omologativa con una corretta. Questo processo non solo migliora l'efficienza operativa, ma assicura anche che ogni veicolo rispetti gli standard di qualità e sicurezza, riducendo al minimo gli errori umani e garantendo una tracciabilità precisa e affidabile.

1.2.2 Integrazione di IR con il MES

Una volta avviata l'app IR dal MES, le pagine di ThingWorx, il middle layer su cui è installata e da cui preleva i dati l'applicazione di Image Recognition, si apriranno automaticamente grazie a un autenticatore personalizzato con il codice RFID del braccialetto per l'autenticazione in dotazione ad ogni operatore. Questo braccialetto RFID serve per autenticarsi al MES direttamente sugli Industrial PC appositamente posizionati lungo le linee di produzione di Lamborghini.
Grazie a questa integrazione, è possibile aprire automaticamente i difetti da IR, intendendo per difetti le etichette non conformi allo standard oppure rovinate dal punto di vista estetico, quindi non utilizzabili, e inviare allegati e immagini nella WPK. La WPK è un

documento legato ad ogni vettura prodotta in Lamborghini, che tiene traccia di qualsiasi operazione e componente montato sulla vettura. Di segutio, è disponibili un immagine che rappresente l'architettura del sistema sopra descritto.

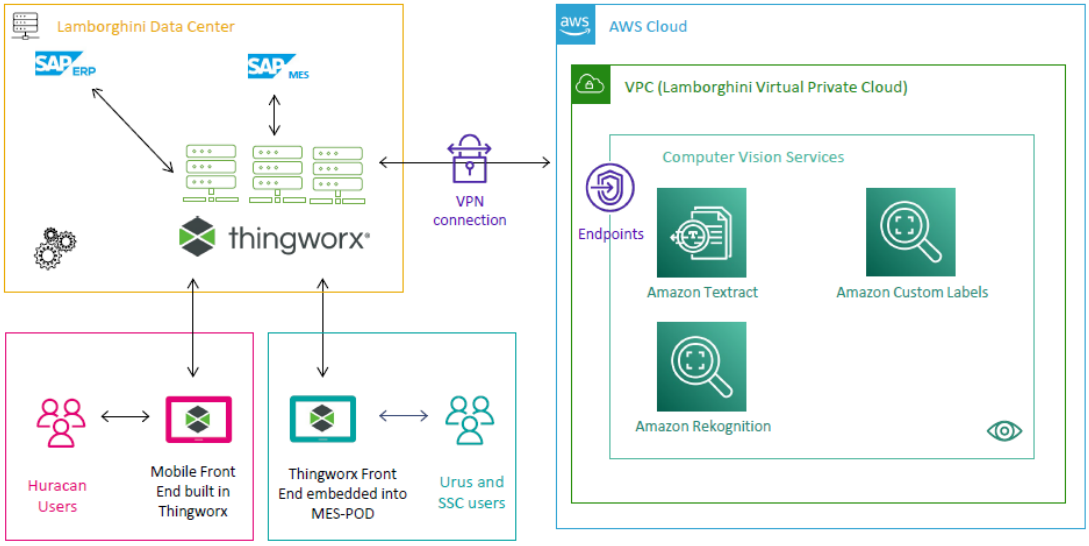

Figura 1.2: Architettura logica del sistema IR

1.2.3 descrizione dell'architettura

Il cuore pulsante dell'architettura dell'app IR è **Thingworx.**
ThingWorx è una piattaforma IoT (Internet of Things) sviluppata da PTC, progettata per abilitare lo sviluppo rapido di soluzioni IoT. Consente di connettere, monitorare, gestire e analizzare dispositivi e macchinari industriali. È molto usata in ambiti di Industria 4.0, per la gestione e l'ottimizzazione della produzione, manutenzione predittiva, e l'efficienza operativa.
In questo caso è necessaria perchè ci consente di accedere, tramite gli Industrial PC presenti sulla linee di produzione e i tablet su cui è installato il sistema MES, all' app IR. IR infatti sfrutta le fotocamere di questi dispositivi per scattare le foto alle etichette da analizzare sul cloud AWS (Come da schema riportato sopra).
Thingworx consente di effettuare queste operazioni dai dispositivi

mobili e di eseguire un'istanza di IR in ogni linea di Lamborghini, questo grazie alla sua predisposizione ed essere connesso a dispositivi e macchinari. Possiamo infatti analizzare le etichette omologative in linea Huracan, SSC (Super Sportive Car) dove vengono prodotte Revuelto e Temerario e nella linea URUS (che si trova in uno stabile completamente distaccato, sempre all'interno del plant produttivo di Sant'Agata Bolognese).

Per quanto riguarda la parte in cloud AWS, Lamborghini ha stipulato accordi specifici con AWS che gli consentono di mantenere una sezione privata della propria rete all'interno dell'infrastruttura Amazon. In questo modo, Lamborghini sfrutta servizi e capacità di storage di AWS garantendo sicurezza, prestazioni elevate e la possibilità di usufruire dei servizi cloud senza dover estendere il proprio data center on-premise.
Grazie a questa soluzione, Lamborghini può beneficiare di tutti i vantaggi del cloud, inclusa la scalabilità automatica e l'accesso a servizi avanzati come quelli per la computer vision, il tutto mantenendo la conformità con le normative di privacy e sicurezza richieste dall' ente (interno di Lamborghini) IT Security e Compliance . AWS, inoltre, garantisce una serie di prestazioni minime sempre disponibili, assicurando continuità operativa e qualità del servizio.
Il motore dell'app di Image Recognition (IR), così come lo storage delle foto delle etichette omologative, risiede nel cloud AWS. Le etichette presenti nelle immagini vengono analizzate tramite i seguenti servizi:

- **Custom Label:**
  Amazon Rekognition Custom Labels permette di addestrare modelli di visione artificiale personalizzati, consentendo di identificare oggetti o concetti specifici in base alle esigenze del Comune, come ad esempio le caratteristiche delle etichette omologative nelle foto.

- **Amazon Textract:**

Amazon Textract estrae automaticamente testo e dati strutturati da documenti e immagini, permettendo di identificare e digitalizzare informazioni presenti sulle etichette, come codici o descrizioni.

- **Amazon Rekognition:**
  Amazon Rekognition è un servizio di analisi delle immagini e dei video che utilizza l'intelligenza artificiale per rilevare, etichettare e analizzare oggetti, volti, testi e altre caratteristiche visive all'interno delle immagini, fornendo risultati in tempo reale e su larga scala.

Questa infrastruttura permette a Lamborghini di gestire efficacemente l'archiviazione e l'analisi delle etichette omologative, che contengono anche dati sensibili relativi ai mercati di destinazione delle vetture, in modo automatizzato e sicuro, sfruttando le risorse del cloud AWS per ottenere migliori performance e flessibilità.

1.2.4 Visualizzatore delle prestazioni

Un pannello di reportistica consente agli utenti con ruolo di amministratore di gestire e visualizzare i dati acquisiti dagli operatori durante il processo di riconoscimento delle etichette. I dati mappati sono:

- **Parametri di ciascun controllo:**
  Include dettagli specifici su ogni controllo effettuato, come impostazioni e condizioni operative.

- **Prestazioni di riconoscimento delle varie etichette:**
  Misura l'accuratezza e l'efficacia del sistema nel riconoscere correttamente le etichette omologative.

- **KPIs della soluzione:**
  Indicatori chiave di prestazione come la capacità di riconoscimento e il tempo richiesto per ogni veicolo.

Questo pannello fornisce una visione dettagliata e interattiva delle prestazioni del sistema, permettendo agli amministratori di identificare rapidamente aree di miglioramento e ottimizzare il processo di riconoscimento delle etichette.

1.3 Esempio di utilizzo di IR

Un caso d'uso specifico dell'app IR prevede l'elaborazione di un'etichetta omologativa, simile a quelle fornite in input sotto forma di foto. L'app è progettata per processare etichette che contengono testi e una geometria già nota, presente nel modello di riferimento. Questo perché la geometria dell'etichetta è uno dei principali elementi sottoposti ad analisi.

- **Analisi preliminare:**
  Una volta ricevuta l'etichetta in input, l'app effettua una prima analisi per identificare eventuali problemi, come graffi, macchie o altre imperfezioni visive. Questo passaggio assicura che l'etichetta sia leggibile e conforme prima di procedere con ulteriori controlli.
- **Controllo della geometria:**
  Se l'etichetta supera la fase di verifica preliminare, l'app confronta la geometria dell'etichetta con le specifiche standard previste. I dati estratti dall'immagine vengono confrontati con le tolleranze minime accettate dagli enti di certificazione e controllo delle etichette, verificando che le misure e la disposizione siano conformi alle normative.
- **Estrazione delle keyword:**
  Nell'ultima fase, l'app procede con l'estrazione delle parole chiave presenti sull'etichetta, che ne definiscono il contenuto. Queste informazioni vengono poi elaborate per garantire che il testo rispetti gli standard richiesti.

Questa sequenza di analisi consente di verificare la conformità delle etichette in modo automatizzato ed efficiente, migliorando l'accuratezza e la velocità del processo di controllo.

Di seguito, alcuni esempi di etichette:

Figura 1.3: Foto per check omologazione portiera, scattata in linea Finizione

Figura 1.4: Foto per check omologazione pneomatici, scattata in linea Finizione

Capitolo 2

Obiettivi ed implementazione del progetto SurfDesk

Durante la fase di raccolta dei requisiti del progetto, sono stati identificati diversi punti di miglioramento per l'applicazione IR. Questi punti sono emersi grazie a una serie di domande rivolte agli operatori in produzione, che sono gli utilizzatori finali dell'applicazione. Di conseguenza, è emersa la necessità di condurre sia micro analisi che macro analisi per esaminare in dettaglio le seguenti aree e i dati relativi all' applicazione IR.

- **Performance di IR dipendenti dal colore:**
  L'analisi si concentra sulla verifica se le performance del sistema di riconoscimento delle immagini (IR) siano influenzate dal colore del telaio su cui vengono applicate le etichette. La domanda di ricerca indaga se le performance di IR dipendono dal colore, utilizzando la percentuale di confidenza associata all'immagine come metrica principale. L'obiettivo è aumentare la presenza dei colori con prestazioni attualmente basse nel dataset di training, in modo da non escludere nessun colore dalle analisi. Questo miglioramento mira a garantire che il sistema IR possa riconoscere accuratamente le etichette indipendentemente dal colore del telaio, migliorando così l'affidabilità complessiva del sistema.

- **Confidenza delle label errate:**
  Questa analisi esplora se l'applicazione IR tende a considerare come corrette certe etichette errate ma con una confidenza relativamente alta. La ricerca utilizza la percentuale di confidenza associata all'immagine e il numero di falsi positivi ottenuti per modulare la soglia di cut-off. L'obiettivo è diminuire la soglia di taglio a una confidenza più bassa, riducendo così i falsi positivi. Identificando il compromesso appropriato per la cutoff-treshold, si mira a migliorare l'affidabilità dei risultati mostrati all'utente, assicurando che le analisi con una confidenza superiore al 60 % siano sempre corrette.

- **Tempo di analisi variabile per WorkCenter:**
  Viene indagato se il tempo di analisi delle etichette varia in base ai WorkCenter. Utilizzando la percentuale di confidenza associata all'immagine e il numero di falsi positivi ottenuti, si analizza la correlazione tra le cartelle di lavoro (workcenter) Modello-Mercato e il tempo medio di analisi per etichetta. L'obiettivo è identificare i WorkCenter che causano rallentamenti e agire sul training del dataset per correggere questi comportamenti. Questo miglioramento mira a ottimizzare l'efficienza del sistema IR, eliminando i colli di bottiglia nei WorkCenter.

- **Tempo di analisi variabile per label uguali:**
  Viene monitorato se il control time per etichette uguali varia con il variare del WorkCenter. Utilizzando la percentuale di confidenza associata all'immagine e il numero di falsi positivi ottenuti, si cerca di trovare una correlazione tra WorkCenter e tempo di elaborazione. Ogni etichetta è controllata in due cartelle di lavoro diverse con due tempi di processo diversi. Sebbene l'analisi non abbia un obiettivo chiaro al momento, serve a dare visibilità a questioni attualmente sconosciute, come la correlazione tra cartella di lavoro, WorkCenter e tempo di elaborazione. Questo potrebbe rivelare problemi di rete o altre variabili che influenzano le performance.

- **Performance di IR dipendenti dai part number:**
  L'analisi si focalizza sul verificare se le performance di IR siano influenzate dai differenti part number nelle etichette. Utilizzando il numero di falsi positivi ottenuti, si analizzano i difetti che fanno riferimento ai part number sulle etichette. L'obiettivo è capire quali part number creano difetti inesistenti e se l'errore è nel training del modello o dipende da altri fattori. Questo miglioramento mira a migliorare l'accuratezza del sistema IR, identificando e correggendo i part number problematici.

- **Performance di IR dipendenti dal luogo di posizionamento delle label:**
  Questa analisi esplora se le performance di IR siano influenzate dal luogo di posizionamento delle etichette sulla vettura. Utilizzando la percentuale di confidenza associata all'immagine, si analizzano le location specifiche sulla vettura in cui vengono applicate le etichette. L'obiettivo è definire alcune location specifiche dove l'app IR non funziona bene e causa errori anche se le etichette sono corrette. L'obiettivo è migliorare l'affidabilità del sistema IR, identificando e correggendo le location problematiche.

- **Variazioni rapide delle performance di IR:**
  L'analisi si concentra sulla possibilità che le performance di IR possano subire rapide variazioni dovute a particolari codici presenti nelle etichette. Utilizzando la percentuale di confidenza associata all'immagine, si cerca di correlare il codice articolo dell'etichetta con le sue prestazioni di riconoscimento. L'obiettivo è notare rapidamente l'abbassamento delle prestazioni per una manutenzione più efficace del sistema. Questo miglioramento consente di monitorare e mantenere le performance del sistema IR, attivando allarmi in caso di diminuzione rapida delle prestazioni.

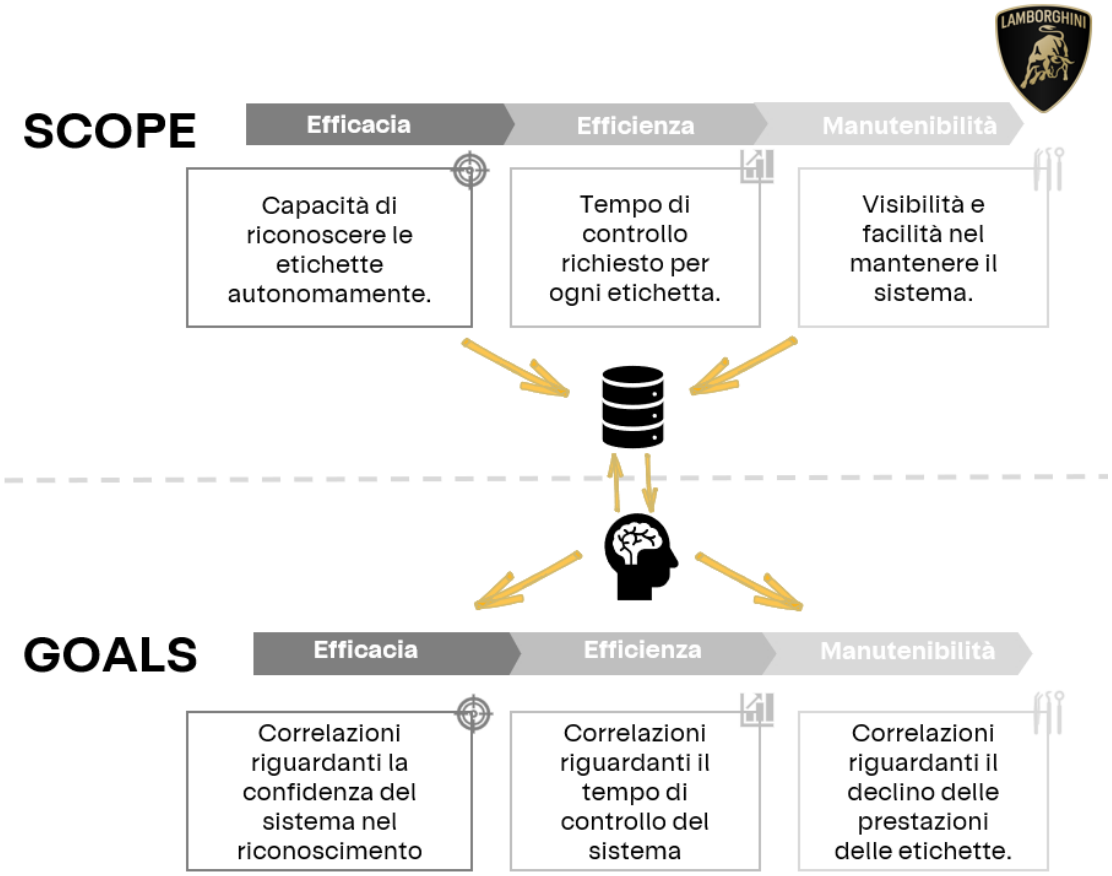

Figura 2.1: Definizione degli obiettivi del progetto Surfdesk

2.1 Miglioramento delle performance dei sistemi partendo dai dati: Nasce Surfdesk

SurfDesk, il nome del Proof of Concept (POC) che ho sviluppato per il lavoro relativo a questa tesi, ha eseguito una serie di analisi sui dati riguardanti le prestazioni e il funzionamento dell'applicazione IR. L'obiettivo è identificare le aree di intervento per aumentare la confidenza, ovvero la percentuale di corretto riconoscimento di una determinata etichetta, restituita come output dall'applicazione. Sono stati considerati diversi fattori e aree di progetto. Per ciascuna di esse è stata formulata una domanda di ricerca per determinare se

le performance di IR dipendessero da quell'area specifica. È stata condotta un'analisi per ognuna di queste aree, ottenendo risultati che hanno permesso di migliorare il sistema di riconoscimento delle etichette.

2.1.1 Identificazione dei dati di Sistema nel MES per il miglioramento delle prestazioni

Dopo aver definito gli obiettivi da raggiungere e le aree di maggiore interesse per il miglioramento delle prestazioni, abbiamo proceduto con l'identificazione dei dati di sistema presenti nel MES.

Partendo dagli output emessi dall'app IR relativi alle varie etichette analizzate, siamo risaliti ai processi di produzione che hanno portato all'applicazione di una determinata etichetta in un determinato punto.

2.1.2 Fattori Chiave e Aree di Interesse

Effettuando questa risalita nel database di produzione del MES, abbiamo identificato le seguenti aree di maggiore interesse:

- **Modello della vettura :**
  Questa informazione può fornire molti dati utili, poiché ogni modello è prodotto su una linea specifica e ha stazioni diverse rispetto ad altri modelli. Questo implica anche operatori differenti, reti internet interne diverse e hardware vari. Anche le architetture cloud differiscono, dato che alcune linee di produzione hanno i loro sistemi informatici che risiedono nel data center di Lamborghini, mentre altre sono completamente in cloud. Tutti questi fattori possono influire sulla percentuale di correttezza del riconoscimento dell'etichetta da parte dell'app IR.

- **Colore del telaio della vettura:**
  Gran parte delle etichette omologative è applicata direttamente sul telaio della vettura, che viene interamente colorato all'interno della linea di Verniciatura. Abbiamo ipotizzato che il colore della scocca potesse influire sulla capacità di riconoscimento

dell'etichetta. Avendo questi dati disponibili, e considerando che anche la linea di Verniciatura è totalmente monitorata e orchestrata dal MES, abbiamo deciso di mettere questi dati in relazione alle etichette per valutare se possano essere utili per la nostra analisi.

- **Part Number**
  Un part number identifica univocamente un determinato tipo di componente montato sopra una vettura. Su alcuni di essi, specialmente su quelli legati alla safety del guidatore, sono applicate alcune etichette omologative. Può essere molto interessante per la nostra analisi andare a mettere in correlazione la percentuale di correttezza dell'algoritmo di IR con i pezzi su cui sono state applicate le etichette. Questo perché alcuni pezzi, a causa dei materiali di cui sono fatti, o per la luce che riflettono possono provocare degli errori nelle fotografie scattate dagli operatori per il riconoscimento delle etichette e di conseguenza abbassare la confidenza dell'algoritmo.

- **Rework nell'area di applicazione dell'etichetta:**
  Questa analisi si concentra sulla correlazione tra i processi di rework e le prestazioni del sistema di riconoscimento delle etichette. I processi di rework includono lavorazioni effettuate per risolvere vari problemi, come graffi estetici, componenti danneggiati da sostituire o lavorazioni meccaniche. Durante un rework, è possibile che un operatore sostituisca dei componenti con altri, cambiando quindi alcuni seriali che non corrispondono a quelli indicati sull'etichetta. È quindi fondamentale aggiornare sempre le etichette per riflettere accuratamente i cambiamenti effettuati. Questo fattore può influire negativamente sulla confidenza dell'algoritmo di riconoscimento, riducendone l'accuratezza rispetto al processo standard. Le informazioni riguardanti questi dati sono sparsi su molte tabelle del sistema MES, in quanto i difetti possono essere aperti e rilavorati in ogni punto della fase di produzione della vettura. Un difetto su una vettura o su un

pezzo da lavorare, in Lamborghini, viene tracciato attraverso la tripletta: Place, Type e Location.

- Place : Indica il componente dove è stato riscontrato il difetto. Sul MES viene tracciato come NC_GROUP.
- Type : Indica il tipo di difetto. Sul MES viene tracciato come NC_CODE.
- Location : Indica una relazione tra Place e i componenti adiacenti, utilizzato per tracciare i casi in cui il difetto può compromettere il montaggio di altri componenti oltre a quello in cui il difetto è riscontrato.

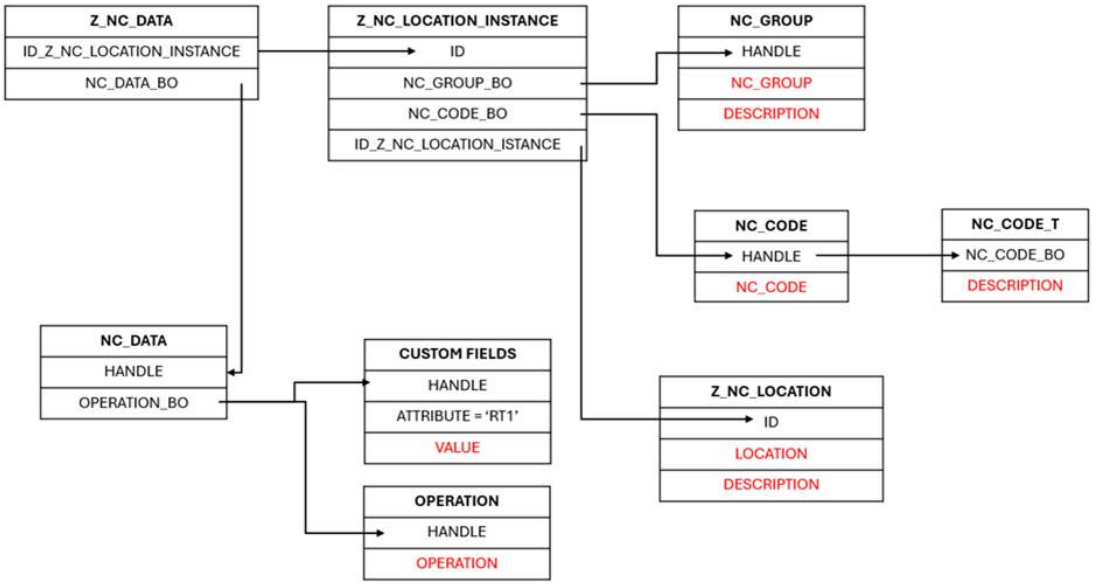

Figura 2.2: Schema illustrativo per il reperimento dei dati sui difetti

- **Dati riguardanti le performance di IR :**
  Il sistema IR genera dati in output derivanti dall'analisi delle immagini, come spiegato nel Capitolo 1. Uno dei valori fondamentali è la confidenza, ovvero la probabilità che un algoritmo assegna a una determinata classificazione o identificazione di un oggetto all'interno di un'immagine. In questo contesto, si considera la forma dell'etichetta e la correttezza delle parole in essa contenute.

In altre parole, la confidenza misura quanto l'algoritmo "crede" che la sua previsione sia corretta. Questo valore è ottenuto attraverso un mix di informazioni, tra cui la custom-label, che verifica la correttezza geometrica e delle dimensioni, e le keyword, che riguardano il riconoscimento delle parole presenti nell'etichetta (come illustrato nello schema sottostante).

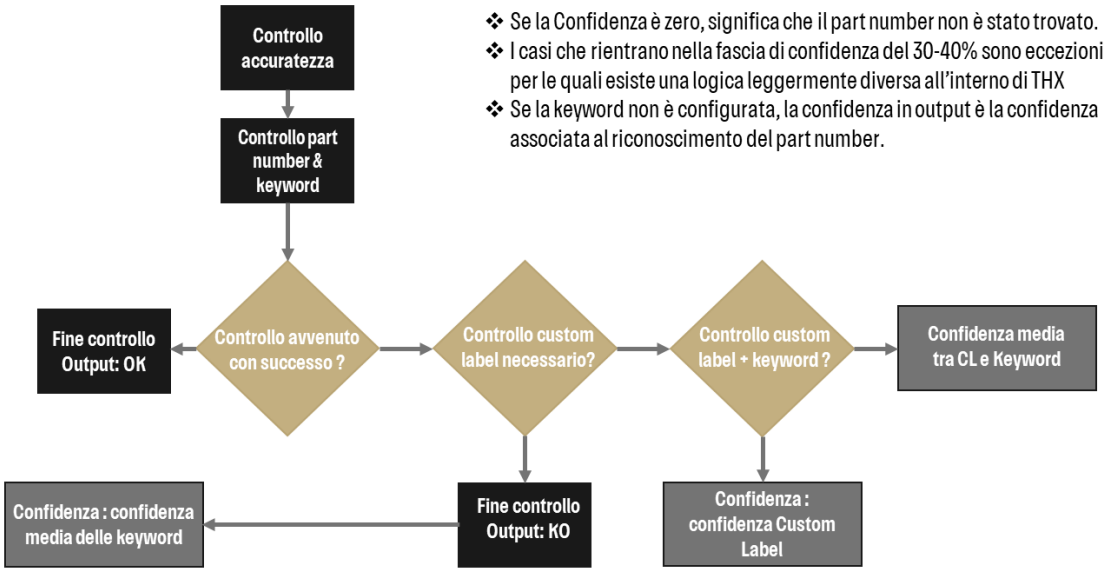

Figura 2.3: Schema illustrativo per la valutazione di un'etichetta da parte del sistema IR.

2.2 Infrastruttura del Datalake AWS

Una volta raccolte tutte le fonti dati dai vari sistemi, come MES e ThingWorks, è necessario trasferire i dati sul cloud, dove saranno analizzati da vari algoritmi di analisi. Il cloud scelto per questo lavoro è AWS, partner ufficiale di Lamborghini per tutte le infrastrutture cloud. Lamborghini, infatti, ha accesso diretto e crittografato ad alcuni server AWS, garantendo la segretezza dei dati depositati sul cloud. Dopo la selezione, i dati vengono integrati in un unico flusso che viene caricato nel bucket S3 di AWS. Questo passaggio è facilitato

grazie a un Datalake Integration Account, che assicura una gestione efficiente e sicura dei dati. Un **Data Lake Integration Account** è un servizio che permette di integrare e gestire grandi volumi di dati provenienti da diverse fonti in un unico repository centralizzato, spesso utilizzato per analisi avanzate e machine learning.

2.2.1 Datalake account

Per quanto riguarda il Data Lake Account, AWS fornisce tutti gli strumenti necessari per salvare i dati grezzi, catalogarli e dividerli in cluster. Utilizzando Amazon S3, è possibile archiviare grandi volumi di dati in modo sicuro e scalabile. Con AWS Glue, abbiamo creato un catalogo dei dati che facilita la ricerca e l'organizzazione delle informazioni.
Nell'ambiente AWS, è possibile scrivere codice in Python attraverso AWS SageMaker per sviluppare, addestrare e distribuire modelli di machine learning utilizzando i dati presenti nei bucket S3, sfruttando la potenza di calcolo dei server AWS.
SageMaker offre un ambiente integrato per eseguire analisi avanzate e machine learning sui dati, rendendo il processo di analisi più efficiente e scalabile. Per la parte di visualizzazione e querying dei dati è stato utilizzato AWS Athena. Questo servizio permette di eseguire query SQL direttamente sui dati archiviati in Amazon S3, senza necessità di configurare server o infrastrutture.
Athena è particolarmente utile per analisi ad-hoc e reportistica (funzionalità che centra in pieno il focus del nostro progetto di Data Analysis), offrendo una soluzione flessibile e potente per esplorare e analizzare i dati.

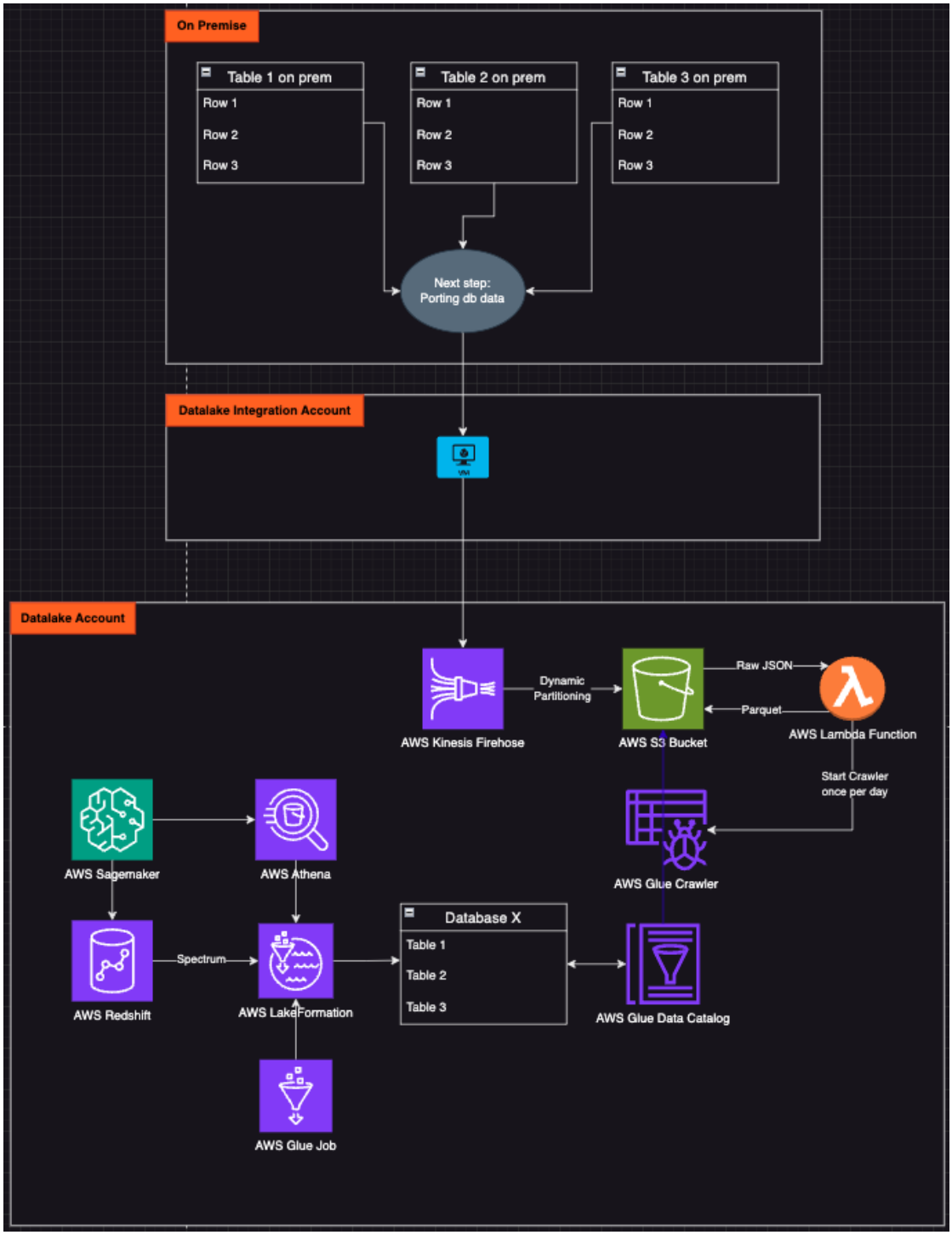

Figura 2.4: Schema illustrativo dell'archiettura e dei componenti utilizzati nel Datalake AWS

2.2.2 Struttura del bucket S3

Il bucket S3 è un contenitore di archiviazione su Amazon Web Services utilizzato per memorizzare e organizzare grandi quantità di dati. Nel progetto SurfDesk, il bucket S3 è strutturato in quattro tabelle principali, nelle quali sono memorizzati dati provenienti da diverse fonti. Ogni tabella contiene numerosi campi e memorizza grandi quantità di dati. Lo scopo del bucket S3 è aggregare i dati provenienti dal sistema IR, derivati da Thingworx, con i dati del MES relativi alle specifiche delle vetture.
Nel progetto Surfdesk, mettiamo in relazione i componenti e le vetture con i dati di IR. Questo ci aiuterà a comprendere se determinati malfunzionamenti di IR possano essere attribuiti a caratteristiche specifiche di un modello di autovettura o di un componente montato su di essa.
Il bucket S3, tuttavia, non ci fornisce una visione chiara dei dati, poiché questi sono molto aggregati. Sarà nostro compito creare query ad hoc per interrogare il bucket S3, estrarre informazioni e individuare possibili correlazioni che contribuiranno a migliorare il sistema IR.
Per fare queste operazioni di query, è stato utilizzato Amazon Athena ovvero un servizio di query interattivo e serverless di AWS che permette di analizzare dati direttamente in Amazon S3 utilizzando SQL standard, noto per la sua efficienza e versatilità, eliminia la necessità di complessi data warehouse.

Capitolo 3

Analisi e risultati

Di seguito, all'interno del terzo capitolo, sono riportati i punti più importanti del codice Python per l'analisi dei dati nel bucket S3, con l'obiettivo di trovare possibili correlazioni tra prestazioni di IR e i vari punti analizzati e descritti nel paragrafo 2.1 riguardante gli obiettivi delle analisi.
L'idea è di esaminare tre tipi diversi di correlazione:

- **Correlazione di Pearson:**
  La correlazione di Pearson, nota anche come coefficiente di correlazione lineare, misura la relazione lineare tra due variabili. Il coefficiente varia tra -1 e 1, dove 1 indica una correlazione positiva perfetta, -1 una correlazione negativa perfetta e 0 nessuna correlazione. È utile quando le variabili sono continue e la relazione è lineare1.

- **Correlazione di Kendall:**
  La correlazione di Kendall, o tau di Kendall, è una misura non parametrica che valuta la forza e la direzione dell'associazione tra due variabili ordinali. A differenza di Pearson, Kendall's tau misura la concordanza tra coppie di osservazioni, risultando particolarmente utile per dati ordinali o quando non si può assumere una relazione lineare2.

- **Correlazione di Spearman:**
  coefficiente di correlazione dei ranghi di Spearman, è un'altra

misura non parametrica che valuta la relazione monotona tra due variabili. Come Kendall, Spearman è adatto per dati ordinali o quando i presupposti per la correlazione di Pearson non sono soddisfatti. Il coefficiente varia tra -1 e 1, simile a Pearson.

L'analisi prevede inoltre la stampa della distribuzione delle etichette in relazione ad ognuno dei punti dell'analisi.

Python Code

Di seguito, alcuni punti fondamentali del codice python sviluppato per trovare le correlazioni tra il **colore esterno delle vetture**, e i **risultati del sistema IR**
Alcune informazioni sull'analisi

- Quest'analisi è stata fatta su un data dump totale di 7263 veicoli, divisi tra Revuelto (777) e Urus (6486).

- L'analisi è stata eseguita dividendo i dati per modello: Urus e Revuelto. I record analizzati (che corrispondono alle label) per Urus sono 145.831 mentre per Revuelto, modello molto più recente, sono 9.604

- Tutti i veicoli analizzati, sono stati prodotti tra il 05/04/2023 e il 18/04/2024

- Solamente il 72% possono essere considerati validi a causa di possibili errori di analisi che derivano da uno scatto errato della foto da analizzare

Lo scopo di quest'analisi è identificare e studiare alcuni possibili pattern che portano ad errori, in modo da definire una strategia di miglioramento per il training dei dati del modelli del sistema IR.
Le tabelle prese in considerazione dal bucket S3 sono la **Z_IR_DATAREGISTRY** e la **Z_SFC_INFO**

ID	AWSCorrettezza	CodiceEtichetta	CustomCorrettezzaDaApplicare	PartNumber
ZPBEC3ZLXPLA25176_Q_TLU_01_C_TPL*_9Y0010530_TP...	88,79	9Y0010530		9Y0.010.530
ZPBEC3ZLXPLA25176_Q_TLU_01_C_TPL*_4ML010515A_A...	91,00	4ML010515A	FindWord	4ML.010.515.A
ZPBEC3ZLXPLA25176_Q_TLU_01_C_TPL*_4ML010550C_S...	99,46	4ML010550C	FindMoreWords	4ML.010.550.C
ZPBEC3ZLXPLA25176_Q_TLU_01_C_TPL*_4ML010520A_S...	99,24	4ML010520A	FindWord	4ML.010.520.A
ZPBEC3ZLXPLA25176_Q_TLU_01_C_TPL*_4ML010525J_S...	0,74	4ML010525J	FindMoreWords	4ML.010.525.J
ZPBEC3ZLXPLA25176_Q_TLU_01_C_TPL*_4ML010502AC_...	83,68	4ML010502AC		4ML.010.502.AC
ZPBEC3ZLXPLA25176_Q_TLU_01_C_TPL*_470010001E_S...	99,76	470010001E	FindWord	470.010.001.E
ZPBEC3ZLXPLA25176_Q_TLU_01_C_TPL*_400010377T_S...	32,84	400010377T		400.010.377.T

Figura 3.1: Schema illustrativo della tabella principale di quest'analisi, la Z_IR_DATAREGISTRY

Un altro punto necessario da comprendere prima di entrare nel vivo dell'analisi del codice python è la scala colori Lamborghini. Il range dei colori di Lamborghini è molto ampio e declinato in tutte le forme. Per questo motivo è stato definito un mapping utilizzando tecniche di *Natural Language Processing (NLP)*

EXT_COLOUR_LTXT	Mean of AWSCorrettezza	Median of AWSCorrettezza	Count of Records	Count of VIN
Nero Helene	87,16	95,94	14738	696
Nero Noctis	88,34	95,89	11986	512
Giallo Auge	85,06	95,12	10592	445
Grigio Keres	86,58	95,51	9560	424
Arancio Borealis	87,28	95,92	7318	310
Bianco Monocerus	88,86	96,11	7001	301
Nero Nemesis	87,44	95,62	6214	274
Bianco Icarus	89,22	96,24	4868	215
Grigio Telesto	87,99	96,14	4705	219
Giallo Inti	87,22	96,02	4289	205

MAIN_COLOR	Mean of AWSCorrettezza	Median of AWSCorrettezza	Count of Records	Count of VIN
Nero	87,64	95,86	33005	1486
Grigio	87,32	95,92	26603	1205
Giallo	85,98	95,50	17738	782
Bianco	89,05	96,16	16427	713
Blu	88,12	96,09	14330	664
Arancio	87,07	95,94	8937	381
Verde	86,98	95,98	8352	386
Militare	87,08	95,57	5990	265
Rosso	87,67	96,14	5015	243
Viola	87,68	96,09	4204	186

Figura 3.2: Schema illustrativo dei colori analizzati

```
# FUNCTIONS

# transforms categorical variables into numerics, the numbers will
    be associated in ascending order according to the number of
   occurrences
def redefine_categorical(df,col):
    dict_values=df[col].value_counts().rank(method='first',
   ascending=False).to_dict()
    if col in ['MAIN_COLOR','PartNumber','NomeModelloAWS']:
        print(dict_values)
    df[col] = df[col].map(dict_values)
    return df

```

```
# Correlations: Prints only cases where it detects significant correlations

# pearson correlation --> linear dependence between variables
from scipy.stats import pearsonr
# spearman correlation --> monotone dependence between variables
from scipy.stats import spearmanr
# kendall correlation --> monotone dependency between variables, similar to kendall but more robust in case of non-normal distributions
from scipy.stats import kendalltau
# spatial distance -->
from scipy.spatial.distance import correlation # values from 0 to 2; 0 perfect correlation, 2 perfect antibody
import traceback

def correlations(df, cols, variables, hue, title):
    correlation_found = False

    for col in cols:
        for var in variables:
            try:
                a = df[col]
                b = df[var]

                # Calcolo delle correlazioni
                corr_p, _ = pearsonr(a, b)
                corr_s, _ = spearmanr(a, b)
                corr_k, _ = kendalltau(a, b)

                if abs(corr_p) >= 0.4 or abs(corr_s) >= 0.4 or abs(corr_k) >= 0.4:
                    correlation_found = True
                    print(f'\n{col} - {var}')
                    print(f'Pearson: {corr_p:.3f}')
                    print(f'Spearman: {corr_s:.3f}')
                    print(f'Kendall: {corr_k:.3f}')

                distance = correlation(a, b)
                if distance >= 1.4 or distance <= 0.6:
                    correlation_found = True
                    print(f'\n{col} - {var}')
                    print(f'Distance: {distance:.3f}')

            except Exception as e:
                print(f"Error processing {col} and {var}: {e}")
                traceback.print_exc()

    if correlation_found:
```

```
55          fig = plt.figure(figsize=(9, 5))
56
57          correlation_matrix_p = df[cols + variables].corr(method='
    pearson')
58          ax1 = fig.add_subplot(221)
59          sns.heatmap(correlation_matrix_p, ax=ax1, xticklabels=
    False,  yticklabels=True, cmap='coolwarm', annot=True, fmt='.2f
    ', annot_kws={"size": 9})
60          ax1.set_title('Pearson Correlation')
61
62          correlation_matrix_s = df[cols + variables].corr(method='
    spearman')
63          ax2 = fig.add_subplot(222)
64          sns.heatmap(correlation_matrix_s, ax=ax2, xticklabels=True
    ,  yticklabels=False, cmap='coolwarm', annot=True, fmt='.2f',
    annot_kws={"size": 9})
65          ax2.set_title('Spearman Correlation')
66
67          correlation_matrix_k = df[cols + variables].corr(method='
    kendall')
68          ax3 = fig.add_subplot(223)
69          sns.heatmap(correlation_matrix_k, ax=ax3, xticklabels=True
    , yticklabels=True, cmap='coolwarm', annot=True, fmt='.2f',
    annot_kws={"size": 9})
70          ax3.set_title('Kendall Correlation')
71
72          df_subset = df[cols + variables]
73          g = sns.pairplot(df_subset, hue=hue)
74          g.fig.suptitle(title, y=1.02)
75
76      plt.show()
```

3.0.1 Analisi degli output dell'app IR

Per quanto riguarda le analisi dei dati prodotti dall'app IR, ci sono tre parametri fondamentali da considerare:

- **AWSCorrettezza:**
  Questo parametro rappresenta la confidenza con cui l'applicazione IR ha riconosciuto correttamente i parametri della label. È un indicatore della precisione iniziale del sistema.

- **AWSCorrettezzaEsito:**
  Questo valore è derivato dal parametro precedente, ma viene ulteriormente valutato rispetto a una soglia prefissata. Se il valore è sotto la soglia, viene considerato errato (false); se è sopra, viene considerato corretto (true).
- **AWSCorrettezzaEsitoUtente:**
  Questo parametro riflette l'esito fornito dall'utente dopo aver effettuato un controllo con il supporto dell'app IR. Può avere solo due valori: OK, se l'utente conferma la correttezza, e NOK, se l'utente rileva un errore.

Segue il codice che inserisce questi parametri nei tre tipi di correlazione analizzati precedentemente: Pearson, Spearman e Kendall

```
    # General

d=model_data[['AWSCorrettezza','AWSCorrettezzaEsito','
   AWSCorrettezzaEsitoUtente','wrongClassification','
   EXT_COLOUR_LTXT','MAIN_COLOR']]

fig = plt.figure(figsize=(9, 5))

correlation_matrix_p = d.corr(method='pearson')
ax1 = fig.add_subplot(221)
sns.heatmap(correlation_matrix_p, ax=ax1, xticklabels=False,
   yticklabels=True, cmap='coolwarm', annot=True, fmt='.2f',
   annot_kws={"size": 9})
ax1.set_title('Pearson Correlation')

correlation_matrix_s = d.corr(method='spearman')
ax2 = fig.add_subplot(222)
sns.heatmap(correlation_matrix_s, ax=ax2, xticklabels=True,
   yticklabels=False, cmap='coolwarm', annot=True, fmt='.2f',
   annot_kws={"size": 9})
ax2.set_title('Spearman Correlation')

correlation_matrix_k = d.corr(method='kendall')
ax3 = fig.add_subplot(223)
sns.heatmap(correlation_matrix_k, ax=ax3, xticklabels=True,
   yticklabels=True, cmap='coolwarm', annot=True, fmt='.2f',
   annot_kws={"size": 9})
ax3.set_title('Kendall Correlation')
```

3.1 Matrici di correlazione per URUS erevuelto

Per comprendere meglio la distribuzione degli errori del sistema IR, abbiamo definito un campo chiamato **WrongClassificationField**. Questo campo identifica i record in cui l'operatore di linea è intervenuto manualmente per correggere i risultati errati riportati nel campo **AWSCorrettezzaEsito**, che contiene i risultati del sistema IR.
Abbiamo analizzato le correlazioni tra i record contrassegnati dalla variabile WrongClassification e i campi *EXT_COLOR_LTXT* (il colore specifico della macchina secondo la nomenclatura Lamborghini) e *MAIN_COLOR*.
In generale, dall'analisi non emerge alcuna evidenza di correlazione lineare o non lineare tra i colori e un decremento delle prestazioni.
L'unico numero di parte (componente montato su URUS) che mostra una leggera correlazione lineare positiva tra WrongClassification e *EXT_COLOR_LTXT* è il componente **4ML.010.541.C.**

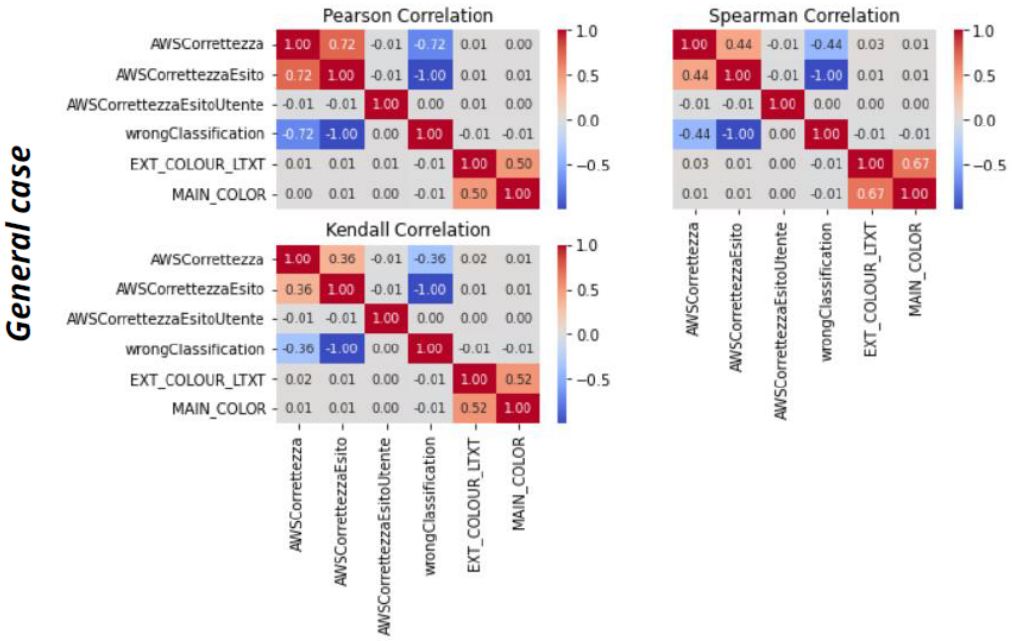

Figura 3.3: Correlazione General Case per URUS

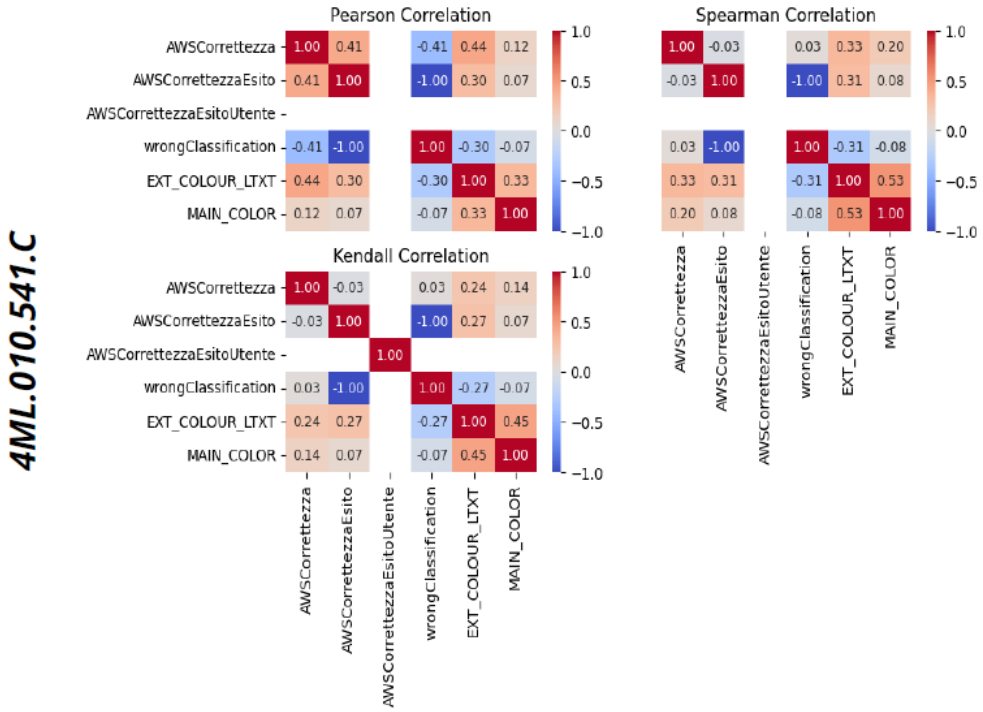

Figura 3.4: Correlazione per componente 4ML.010.541.C URUS

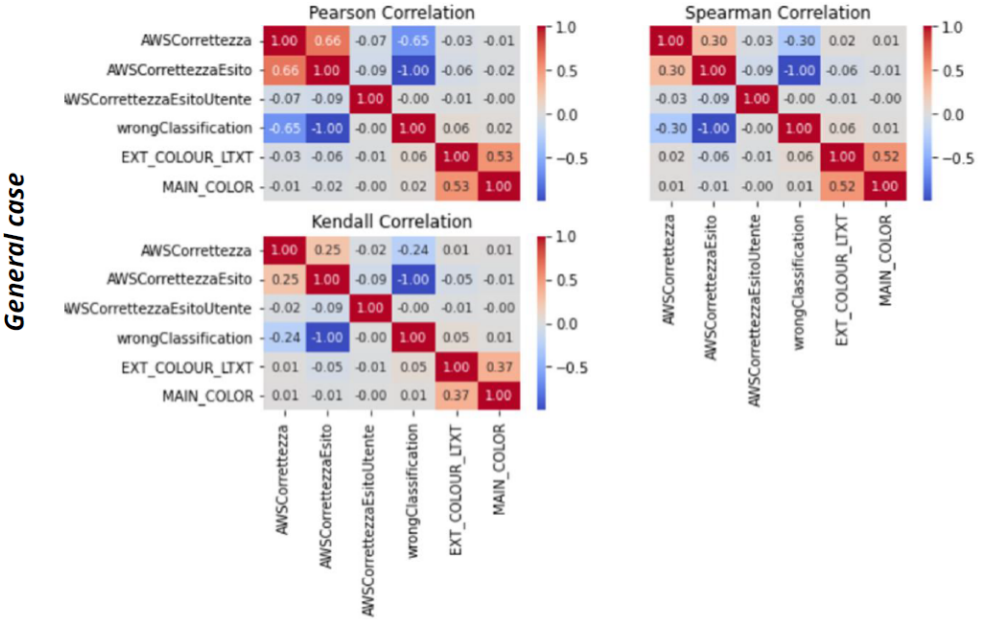

Figura 3.5: Correlazione General Case per modello Revuelto

3.2 Data Distribution

Le analisi condotte sulla correlazione tra i colori esterni dei veicoli e gli errori del sistema IR non hanno prodotto evidenze significative di una correlazione. Pertanto, abbiamo deciso di adottare un approccio alternativo per valutare le performance del sistema IR.
Considerando la distribuzione dei dati per una determinata categoria (ad esempio, famiglia di colori), se non esiste alcuna correlazione, la distribuzione degli errori per la stessa categoria dovrebbe rispecchiare la distribuzione dei dati originali. In altre parole, la proporzione di errori dovrebbe essere simile alla proporzione dei dati per ciascuna categoria.
Tuttavia, se emergono deviazioni significative tra le due distribuzioni, ciò potrebbe indicare la presenza di una correlazione. Questo approccio ci permette di identificare eventuali anomalie e di comprendere meglio le dinamiche tra il colore esterno dei veicoli e le performance del sistema IR. Le seguenti tabelle mostrano la distribuzione dei dati suddivisa non solo per colore, ma anche per part number e famiglia di etichette, confrontandola con la distribuzione dei casi di errore, evidenziandone le differenze. Seguendo questo approccio, possiamo osservare che esistono alcuni part number e famiglie di etichette in cui l'algoritmo di IA mostra una certa *pigrizia*. In particolare, analizzando l'etichetta **UR_PNEUMATICI_NERO**, è evidente come la distribuzione degli errori sia significativamente più alta rispetto alla distribuzione dei dati, presentando una notevole percentuale di differenza tra le due distribuzioni.
Tuttavia, non sono emerse evidenze significative riguardo al colore.

Part Number

PartNumber	Number of Records	Number of Errors	Data Distribution	Errors Distribution	Delta
4ML.010.502.AD	2086	1368	1,45	8,08	6,76
4ML.012.785.MA	4170	1452	2,90	8,71	5,81
4ML.010.502.AJ	3530	1306	2,45	7,84	5,38
4ML.010.525.J	1888	886	1,31	5,32	4,00
4ML.010.520.D	1614	557	1,12	3,34	2,22
4ML.010.525.L	1153	498	0,80	2,99	2,19

Label Families

NomeLabelModelloAWS	Number of Records	Number of Errors	Data Distribution	Errors Distribution	Delta
UR_PNEUMATICI_NERO	6385	3117	4,44	18,70	14,26
N.A.	2972	1308	2,07	7,85	5,78
UR_OTHER_MANUFACTURER	4212	1301	2,93	7,81	4,88
UR_US_MANUFACTURER	2521	951	1,75	5,71	3,95
4ML012785MA	3773	1056	2,62	6,34	3,71
UR_MATT	1371	515	0,95	3,09	2,14

Color

MAIN_COLOR	Number of Records	Number of Errors	Data Distribution	Errors Distribution	Delta
Nero	33005	3993	22,95	23,96	1,01
Giallo	17738	2207	12,33	13,24	0,91
Grigio	26603	3172	18,50	19,03	0,53

Figura 3.6: URUS | Distribuzione dati diviso per Part Number, Label Family, e Colore illustrativo dei colori analizzati

Dopo una prima analisi della distribuzione dei dati, abbiamo deciso di adottare un approccio più dettagliato. Abbiamo suddiviso le distribuzioni per famiglia e analizzato le distribuzioni dei dati e degli errori per colori differenti, considerando che non per tutte le famiglie l'algoritmo di Image Recognition è influenzato dal colore esterno della vettura. In alcune immagini, infatti, il colore non è presente.
Nella tabella in alto della seguente immagine, abbiamo riportato tutti i casi in cui il delta (la differenza tra le due distribuzioni) è maggiore di **5 punti percentuali**. I dati mostrano che per le famiglie **UR_4T0010556** e **UR_FAN**, quando il colore esterno della vettura è giallo, l'algoritmo di IR è particolarmente "pigro", aumentando il

numero di errori.
Un fenomeno simile è osservabile anche per il colore grigio, ma il numero di record considerato è troppo basso per classificare questo dato come significativo.

Di seguito, riporto il codice python per le analisi delle distribuzioni per famiglie diverse di Label

```
    # Distribution NomeLabelModelloAWS + MAIN_COLOR

# General
df_distr=df[['NomeLabelModelloAWS','MAIN_COLOR','
    wrongClassification']].groupby(['NomeLabelModelloAWS','
    MAIN_COLOR'])
df_distr=df_distr.value_counts().rename_axis().to_frame('
    count_record').sort_values(by=['NomeLabelModelloAWS','
    MAIN_COLOR'],ascending=False)


# color
df_distr.loc[:,'color_records']=df[['NomeLabelModelloAWS','
    MAIN_COLOR']].value_counts()


# label
df_distr.loc[:,'label_records']=df[['NomeLabelModelloAWS']].
    value_counts()
df_temp=df[['NomeLabelModelloAWS','wrongClassification']].
    value_counts().rename_axis().to_frame('
    label_count_classification')


df_distr=df_distr.join(df_temp,how='left').reset_index().set_index
    (['NomeLabelModelloAWS','MAIN_COLOR','wrongClassification']).
    sort_values(by=['NomeLabelModelloAWS','MAIN_COLOR','
    wrongClassification'],ascending=False)

# distribution
df_distr['distr_norm']=round(df_distr.color_records/df_distr.
    label_records*100,2)
df_distr['distr_error']=round(df_distr.count_record/df_distr.
    label_count_classification*100,2)

# delta
df_distr=df_distr.loc[:,:,1].rename(columns={'count_record':'
    color_error_records','label_count_classification':'
    label_error_recors'})
```

```
df_distr=df_distr[['color_error_records', 'color_records', '
    label_error_recors', 'label_records', 'distr_norm','distr_error
    ']]
df_distr['delta']=df_distr.distr_error-df_distr.distr_norm


# threshold
th_color=200
th_delta=4.5

# df_distr.sort_values(by=['delta'],ascending=False).loc['
    UR_7L0010872L',:]
df_distr.sort_values(by=['delta'],ascending=False)[(df_distr.
    color_records>th_color)&(df_distr.delta>=th_delta)].shape
df_distr.sort_values(by=['delta'],ascending=False)[(df_distr.
    color_records>th_color)&(df_distr.delta>=th_delta)]
```

NomeLabelModelloAWS	MAIN_COLOR	Number of Error Records for (Label, Color)	Number of Records for (Label, Color)	Number of Error Records for Label	Number of Records for Label	Data Distribution	Error Distribution	Delta
UR_4T0010556	Giallo	254	2041	461	5640	36,19	55,10	18,91
UR_7L0010872L	Grigio	6	585	20	3608	16,21	30,00	13,79
UR_FAN	Giallo	130	1298	600	10671	12,16	21,67	9,51
UR_4ML010500	Nero	38	622	122	2837	21,92	31,15	9,23
UR_4ML010538	Grigio	28	204	110	1098	18,58	25,45	6,87
UR_7L0010872L	Giallo	3	294	20	3608	8,15	15,00	6,85

UR_4T0010556

MAIN_COLOR	Number of Error Records for (Label, Color)	Number of Error Records for (Label, Color)	Number of Error Records for Label	Number of Records for Label	Data Distribution	Error Distribution	Delta
Giallo	254	2041	461	5640	36,19	55,10	18,91
Arancio	32	354	461	5640	6,28	6,94	0,66

UR_7L0010872L

MAIN_COLOR	Number of Error Records for (Label, Color)	Number of Records for (Label, Color)	Number of Error Records for Label	Number of Records for Label	Data Distribution	Error Distribution	Delta
Grigio	6	585	20	3608	16,21	30,0	13.79
Giallo	3	294	20	3608	8,15	15,0	6,85
Bianco	4	567	20	3608	15,72	20,0	4,28

UR_FAN

MAIN_COLOR	Number of Error Records for (Label, Color)	Number of Records for (Label, Color)	Number of Error Records for Label	Number of Records for Label	Data Distribution	Error Distribution	Delta
Giallo	130	1298	600	10671	12,16	21,67	9,51
Grigio	141	1975	600	10671	18,51	23,50	4,99

Figura 3.7: URUS | Data distribution per Label Family differente

NomeLabelModelloAWS	MAIN_COLOR	Number of Error Records for (Label, Color)	Number of Records for (Label, Color)	Number of Error Records for Label	Number of Records for Label	Data Distribution	Error Distribution	Delta
HU_REV_AC	Nero	2	127	3	752	16,89	66,67	49,78
REV_4KE010007	Grigio	5	110	9	704	15,62	55,56	39,94
REV_GASOLINE	Grigio	2	129	5	765	16,86	40,00	23,14
HU_REV_AC	Grigio	1	128	3	752	17,02	33,33	16,31
HU_REV_AC	Grigio	6	247	21	1503	16,43	28,57	12,14

Figura 3.8: REVUELTO | Data distribution per Label Family differente

Viene riportato il codice Python per le analisisi delle distribuzioni dei dati e degli errori. Nello specifico, il codice scritto nel box sottostante fa riferimento al modello URUS. Quello di Revuelto è analogo e per questo motivo non viene riportato

```
# DISTRIBUIONS

df=data_urus.copy()
rows=df.shape[0]

# Error by colour
error_count=df[df.wrongClassification==1].MAIN_COLOR.value_counts()
count=df.MAIN_COLOR.value_counts()
perc=count/rows*100
error_perc=error_count/df[df.wrongClassification==1].shape[0]*100
delta=error_perc-perc

color=pd.DataFrame({'count_record':count,'error_count':error_count, 'df_rows':rows, 'data_distribution':perc,'error_distribution':error_perc,'delta':delta})

# Error per prn
error_count=df[df.wrongClassification==1].PartNumber.value_counts()
count=df.PartNumber.value_counts()
perc=count/rows*100
error_perc=error_count/df[df.wrongClassification==1].shape[0]*100
delta=error_perc-perc

prn=pd.DataFrame({'count_record':count,'error_count':error_count, 'df_rows':rows, 'data_distribution':perc,'error_distribution':error_perc,'delta':delta})

# Error per family
error_count=df[df.wrongClassification==1].NomeLabelModelloAWS.value_counts()
count=df.NomeLabelModelloAWS.value_counts()
perc=count/rows*100
error_perc=error_count/df[df.wrongClassification==1].shape[0]*100
delta=error_perc-perc

fm = pd.DataFrame({'count_record':count,'error_count':error_count, 'df_rows':rows, 'data_distribution':perc,'error_distribution':error_perc,'delta':delta})


```

```
34 # Error per workcenter
35 error_count=df[df.wrongClassification==1].Workcenter.value_counts
       ()
36 count=df.Workcenter.value_counts()
37 perc=count/rows*100
38 error_perc=error_count/df[df.wrongClassification==1].shape[0]*100
39 delta=error_perc-perc
40
41 wc = pd.DataFrame({'count_record':count,'error_count':error_count,
        'df_rows':rows, 'data_distribution':perc,'error_distribution':
       error_perc,'delta':delta})
42
43
44 color.sort_values('delta', ascending=False)
45 prn.sort_values('delta', ascending=False)
46 fm.sort_values('delta', ascending=False)
47 wc.sort_values('delta', ascending=False)
```

3.3 Treshold Modulation

La correlazione tra **Modello**, **Mercato**, **Workcenter** e **Tempo Medio di Controllo** per Etichetta è stata analizzata utilizzando la tabella di input Z_IR_DATAREGISTRY. Dall'analisi dei Partnumber, sono emerse quattro configurazioni possibili, ciascuna corrispondente a controlli differenti. Tuttavia, a causa della mancanza di dati sufficienti per identificare eventuali errori nello step C1 (dello schema di analisi delle etichette mostrato sotto), l'ottimizzazione della soglia minima può essere effettuata solo per le configurazioni 3 e 4 presenti nella tabella sottostante.
Nota Importante: La configurazione 3 include un solo record per il modello Urus e nessun record per il modello Revuelto, rendendo impossibile la sua analisi.

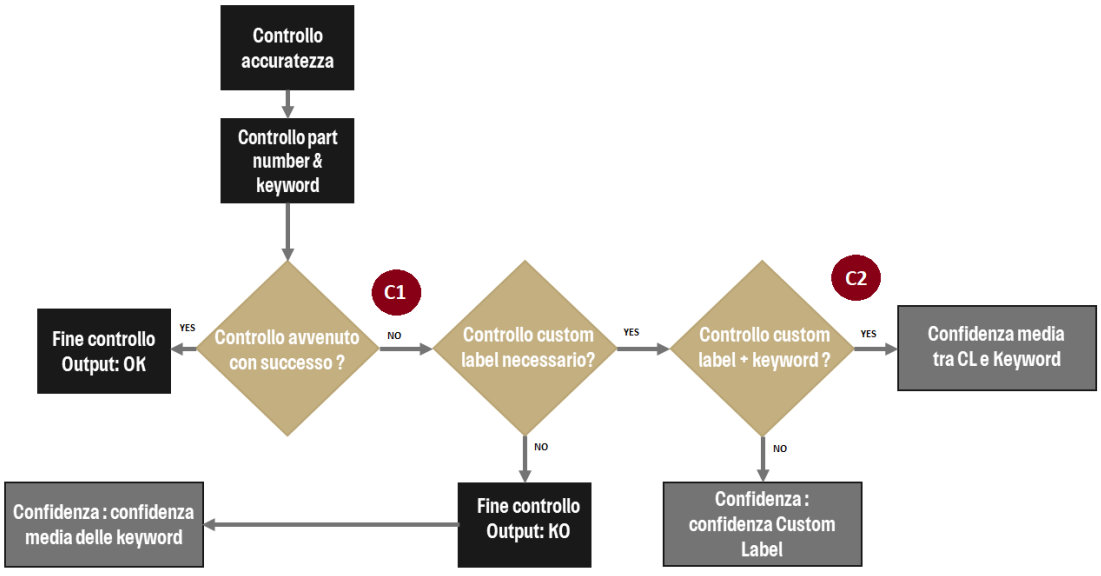

Figura 3.9: Schema di analisi delle label

Config ID	CheckCustomCorrettezza	CheckCustomLabelPerCorrettezza	AWSCorrettezza value
1	1	1	• Average between the confidences for keywords if C1 is OK • Avergage among the confidences for keywords and custom label if C1 is KO
2	0	1	• Confidence of the Part Number identification if C1 is OK • Confidence of Custom Label id C1 KO
3	1	0	• Average between the confidences for keywords if C1 is OK
4	0	0	• Confidence of the Part Number identification

Figura 3.10: Tabella di treshold modulation

Nel processo indicato dallo schema sopra riportato, AWSCorrettezzaEsito è uguale a 1 quando il punteggio ottenuto dal Riconoscimento, ovvero **AWSCorrettezza**, è superiore alla soglia data da **ConfidenzaAccettataCorrettezza**.
Analizzando i dati, classifichiamo i record secondo le seguenti definizioni:

- *AWSCorrettezzaEsito = UtenteCorrettezza = 1*: Vero Positivo (TP) → Etichetta corretta riconosciuta correttamente
- *AWSCorrettezzaEsito = UtenteCorrettezza = 0*: Vero Negativo (TN) → Etichetta sbagliata riconosciuta correttamente
- *AWSCorrettezzaEsito = 1, UtenteCorrettezza = 0*: Falso Positivo (FP) → Etichetta sbagliata riconosciuta come corretta
- *AWSCorrettezzaEsito = 0, UtenteCorrettezza = 1*: Falso Negativo (FN) → Etichetta corretta riconosciuta come sbagliata

Per valutare correttamente la previsione di un modello di classificazione, due metriche sono di particolare interesse:

- **Precisione: TP / (TP + FP)**
- **Richiamo: TP / (TP + FN)**

Le due metriche variano da 0 a 1; più alto è il valore, migliore è la performance del modello. La precisione del modello è 1 poiché non ci

sono Falsi Positivi.
Per quanto riguarda URUS, Il richiamo del modello è 0,9851. Questi valori eccellenti indicano che per questa configurazione il modello funziona molto bene.

Number of records	Number of TP	Number of TN	Number of FP	Number of FN
5.651	5.567	0	0	84

Figura 3.11: Tabella per analisi di precisione modello Urus

Per quanto riguarda Revuelto, la precisione del modello è 1 poiché non ci sono Falsi Positivi. Il richiamo del modello è 0,9987. Questi valori sono eccezionali, ancora migliori del caso Urus, e confermano come per questa configurazione il modello funzioni in modo eccellente.

Number of records	Number of TP	Number of TN	Number of FP	Number of FN*
2.277	2.274	0	0	3

Figura 3.12: Tabella per analisi di precisione modello Revuelto

Vediamo ora il cuore della parte di codice python sviluppata per questa analisi

```
#calculating TP, TN, FP, FN
tp = data_1['TP_ALL'].sum()
tn = data_1['TN_ALL'].sum()
fp =  data_1['FP_ALL'].sum()
fn = data_1['FN_ALL'].sum()

# Computations of the metrics
prec = round(tp/(tp+fp),4)
recall = round(tp/(tp+fn),4)
f1 = round(2 * (prec*recall)/(prec+recall),4)

print('Precision: ', prec)
print('Recall: ', recall)
print('F1-Score: ', f1)
```

Analizziamo ora la MACRO efficienza del sistema IR sulla linea URUS per quanto riguarda il tempo di controllo di ogni etichetta diviso per Part Number.

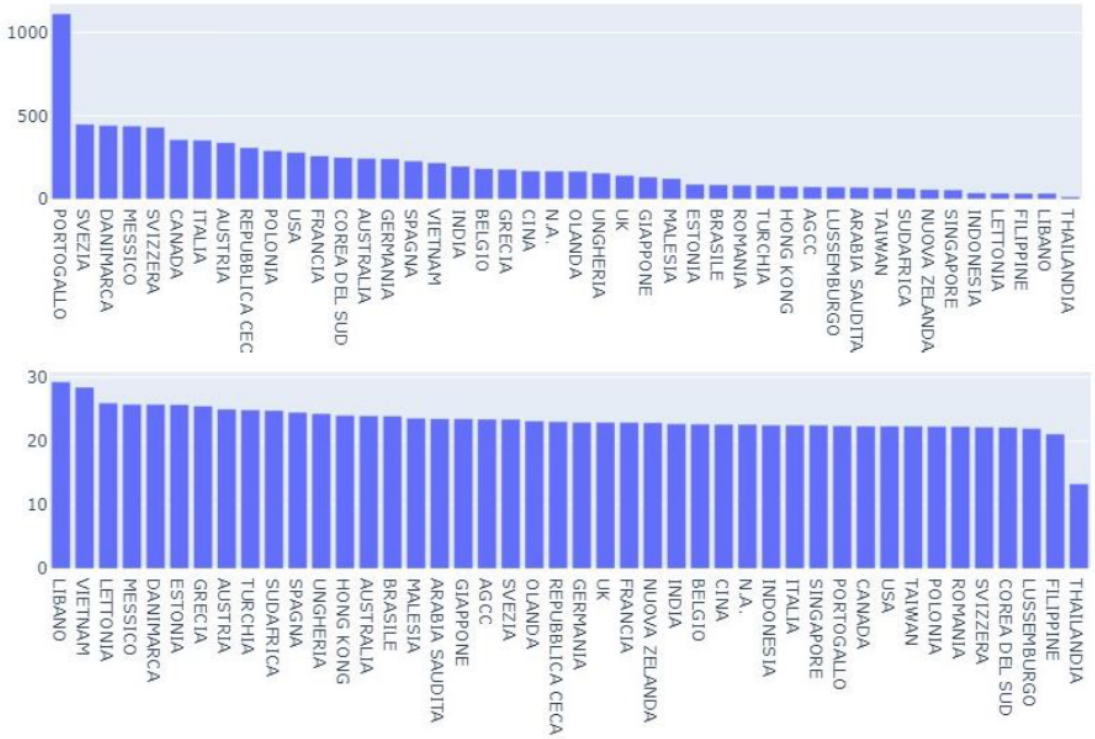

Figura 3.13: MACRO efficency per URUS. Media (in alto) e Mediana (in basso) del control Time delle etichette di un Part Number per Mercato di vendita

Dai tempi medi di controllo sembra che i controlli per il mercato portoghese siano molto più lenti rispetto ad altri paesi, ma questo effetto è dovuto a grandi valori anomali come si può vedere dai valori mediani (oltre 1400 secondi contro 20 secondi).
Dai valori mediani, i controlli per il mercato vietnamita sono i più lenti, ma per ogni paese il tempo mediano è approssimativamente inferiore ai 30 secondi.

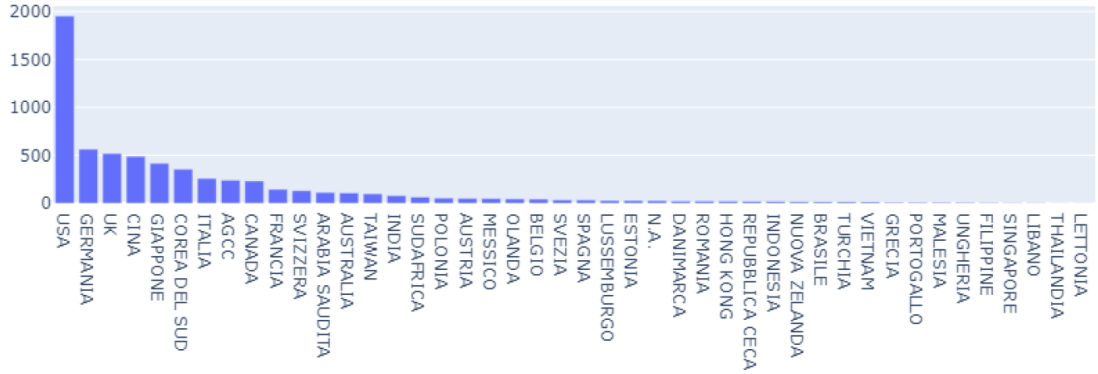

Figura 3.14: MACRO efficency del tempo il medio di controllo etichette analizzato per Mercato

Analizziamo ora alcuni aspetti relativi all'analisi del tempo medio di controllo per veicolo, in relazione all'analisi sul tempo medio di controllo per Part Number. Riusciamo a trovare delle similitudini nelle analisi.

- Analogamente all'analisi del numero di parte, anche il tempo medio per i controlli completi del veicolo è influenzato da valori anomali (principalmente per Canada e Portogallo).
- I valori mediani mostrano che controllare i veicoli per il mercato canadese e cinese richiede più tempo (quest'ultimo come previsto).

Questo tipo di analisi è stato condotto esclusivamente per il modello URUS e le sue varianti (UPHV, URSD, URSS, USTD), ovvero le diverse configurazioni del veicolo, poiché presentavano una base dati sufficientemente ampia per l'analisi.
Non è stato possibile effettuare la stessa analisi per il modello Revuelto, in quanto questo modello è stato introdotto nel 2023 ed è in produzione da pochi mesi al momento dell'analisi.

3.4 Analisi della correlazione tra Workcenter, Model type e Control Time

Per valutare l'efficienza del nostro algoritmo, eseguiamo due tipi di analisi: **Macro** e **Micro analisi**. Per la Macro analisi, prendiamo in considerazione i dati sia del modello URUS che del modello Revuelto. Per la Micro analisi, invece, ci concentriamo esclusivamente sul modello URUS, poiché non disponiamo di dati sufficienti per effettuare una microanalisi significativa sul modello Revuelto, che possa evidenziare eventuali correlazioni con i pochi dati disponibili.

3.4.1 Macro analisi

Nella tabella sottostante è riportata l'analisi delle correlazioni tra i diversi tipi di modello del URUS (inclusi tutte le varianti prodotte, come la URUS Sport, la URUS SE Hybrid, ecc.) suddivisi per centro di lavoro, ovvero le stazioni di analisi delle etichette, e il tempo medio di controllo di ciascuna etichetta.

Workcenter	Model type	Mean control time (s)	Median control time (s)	Part Numbers checked
FLU_REW_A	N.A.	48	43	9
FLU_REW_A	UPHV	42	21	293
FLU_REW_A	URSD	127	20	25210
FLU_REW_A	URSS	158	21	32464
FLU_REW_A	USTD	42	24	58
Q_TLU_01_C	N.A.	45	25	25
Q_TLU_01_C	UPHV	933	31	522
Q_TLU_01_C	URSD	327	25	31544
Q_TLU_01_C	URSS	293	25	40094

Figura 3.15: Analisi relativa ai model type derivati da URUS

- Escludendo i casi in cui il tipo di modello non è disponibile (tipo di modello = N.A.) e guardando ai valori mediani, il centro di lavoro FLU_REW_A impiega circa 20 secondi per ogni numero di parte. Il caso più lento è per il modello USTD, ma il numero di record è molto basso.

- centro di lavoro Q_TLU_01_C è più lento di FLU_REW_A, impiegando circa 25-30 secondi per numero di parte.
- Considerando i due tipi di modelli più rappresentati, si può vedere che FLU_REW_A richiede 20 secondi per eseguire i controlli mentre Q_TLU_01_C ne richiede cinque in più.

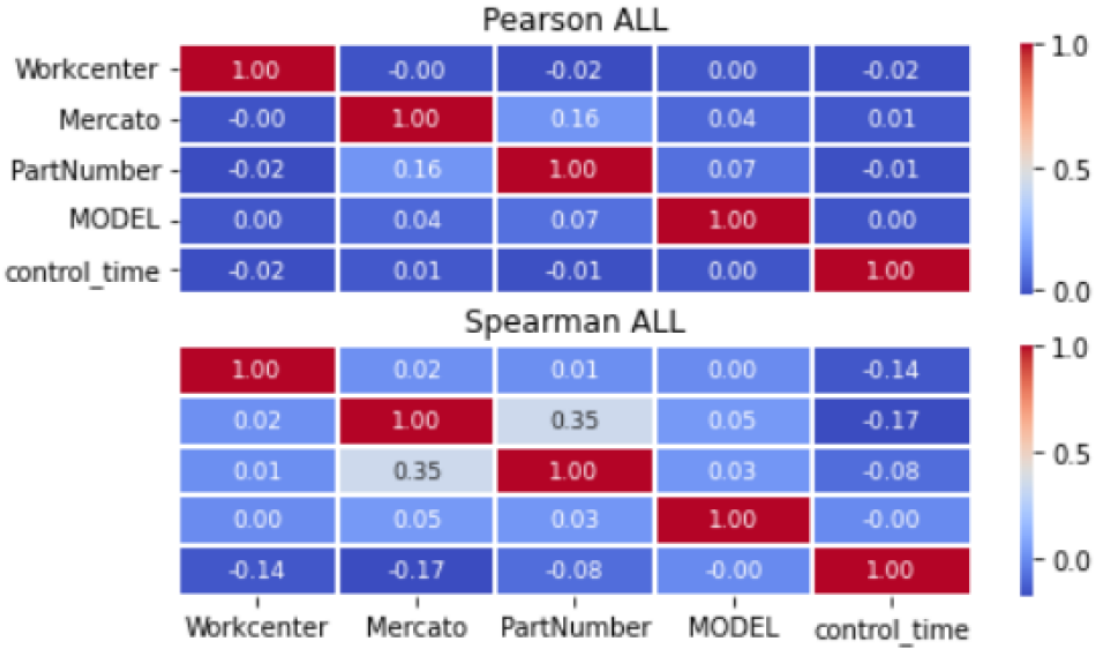

Figura 3.16: Correlazione di Pearson e Spearman per URUS

Dalle due matrici di correlazione possiamo vedere come sia evidente l'assenza di correlazioni tra le informazioni analizzate e dettagliate in precedenza.

Vediamo di seguito la stessa analisi per Revuelto

Come possiamo vedere dalla tabella sottostante, Revuelto, essendo un nuovo modello, non presenta ancora nessuna derivata di serie. L'unico modello in produzione è L744 che corrisponde alla prima versione di Revuelto. Analizziamo quindi la correlazione tra WorkCenter e l'unico model type presente.

Workcenter	Model type	Mean control time (s)	Median control time (s)	Part Numbers checked
Q_TL_01_A	L744	69	36	67
Q_TL_01_A	N.A.	182	41	49
Q_TL_01_C	L744	832	27	8647

Figura 3.17: Analisi relativa al model type Revuelto

- Escludendo i casi in cui il tipo di modello non è disponibile (tipo di modello = N.A.), guardando ai valori mediani, il centro di lavoro Q_TL_01_A impiega circa 35 secondi per ogni numero di parte, ma il numero di record è molto basso.
- Considerando il tipo di modello più rappresentato, si può vedere che Q_TL_01_C richiede 27 secondi per controllare i numeri di parte.

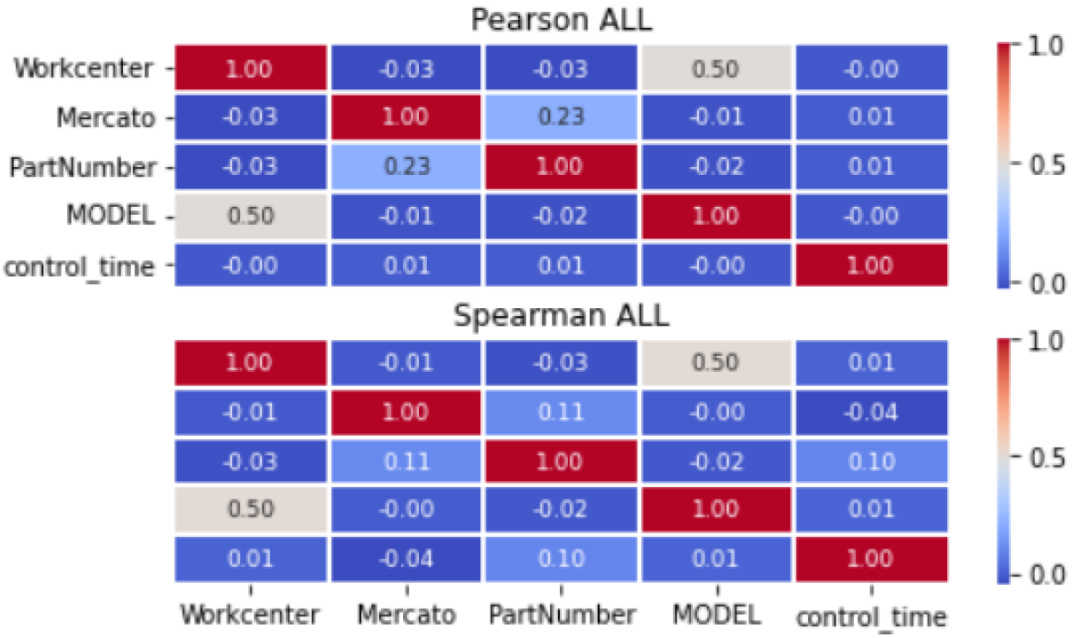

Figura 3.18: Analisi relativa al model type Revuelto

Anche in questo caso, dalle due matrici di correlazione, non sono presenti particolari correlazioni

Viene riportato di seguito il codice relativo alla Macro analisi per le correlazioni di Pearson, Spearman e Kendall.

```
#pearson correlation
from scipy.stats import pearsonr
#spearman correlation
from scipy.stats import spearmanr
#kendall correlation -->
from scipy.stats import kendalltau

from scipy.spatial.distance import correlation
import traceback

# Function to compute correlations
def correlations(df,cols, colors,hue, title):
    correlation_found=False
    for col in cols:
        for color in colors:
            try:
                a=df[col]
                b=df[color]
                corr_p, _ = pearsonr(a, b)
                corr_s, _ = spearmanr(a, b)
                corr_k, _ = kendalltau(a, b)

                if(abs(corr_p)>=0.4 or abs(corr_s)>=0.4 or abs(
   corr_k)>=0.4):

                    correlation_found=True

                    print('\n'+col+' - '+color)
                    print('Pearsons: %.3f' % corr_p)
                    print('Spearman: %.3f' % corr_s)
                    print('Kendall: %.3f' % corr_k)
            except:
                traceback.print_exc()
#
            distance = correlation(df[col], df[color])
            if(distance>=1.4 or distance<=0.6):

                correlation_found=True
                print('\n'+col+' - '+color)
                print('Distance: %.3f' %distance)

    #sezione relativa alla stampa dei grafici omessa
    return
```

3.4.2 Micro analisi

Correlazione tra centro di lavoro e tempo di elaborazione.
La tabella di origine è **z_irtimedataforAWSvelocity**. Il tempo di controllo è calcolato come la differenza tra il timestamp di inizio e fine di ogni controllo.

Workcenter	Average micro control time (seconds)	Median micro control time (seconds)
FLU_REW_A	4,13	3,59
Q_TLU_01_C	4,24	3,53

Figura 3.19: Micro analisi per tempo di controllo medio URUS

Viene rappresentato nel grafico sottostante, la percentuale del tempo di controllo totale, diviso per tempo di controllo per componente. Possiamo vedere che i due valori sono pressoché simili.
Questo ci conferma che le prestazioni di IR **non dipendono dal componente** analizzato.
Un'altra conferma di questa fatto è visibile dalla tabella dei tempi medi. Vediamo infatti che sia i tempi medi che mediani di controllo sono molto vicini per i due centri di lavoro.

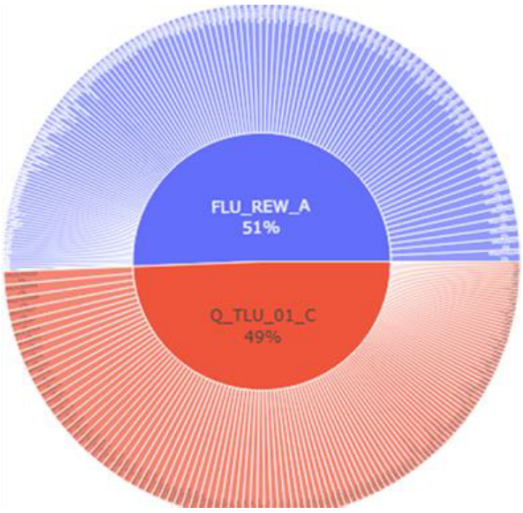

Figura 3.20: Micro analisi workcenter URUS

Andiamo ad analizzare, in modo ancora più dettagliato, ora per ora, quando le prestazioni dui due workcenter vanno a differire. Nello specifico, sono stati creati due grafici, rappresentanti la media e la mediana del tempo medio di controllo dei due Work center analizzati: **FLU_REW_A** e **Q_TLU_01_C**.
Dalle linee di tendenza si può vedere come ci siano due sovrapposizioni tra le curve, la prima alle 04:00 e la seconda alle 17:00.
In queste fasce orarie quindi il tempo di analisi è pressoché uguale.
Questo ci è servito per capire se alcuni cali di prestazioni in determinati workcenter, in determinate ore potessero derivare da alcuni cali prestazionali della rete internet interna all'azienda.

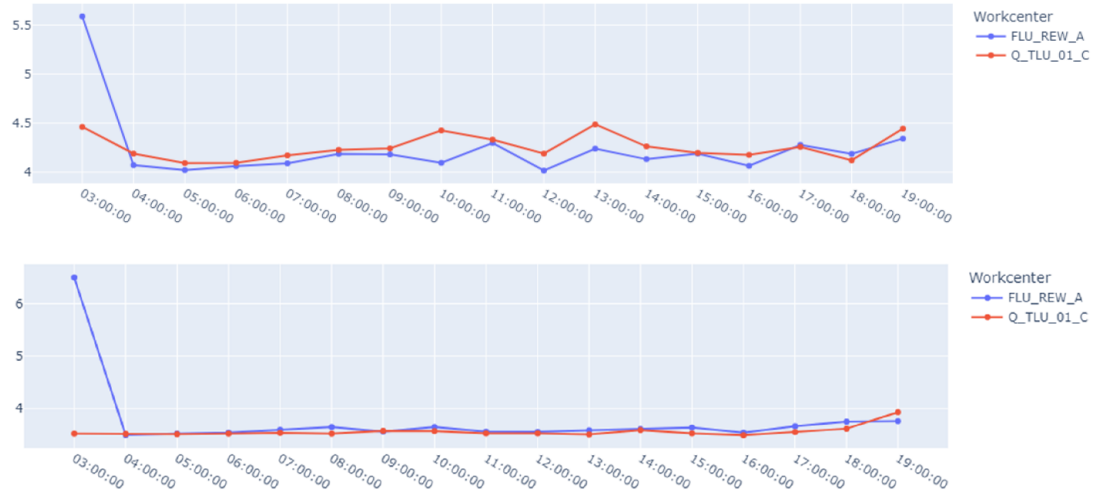

Figura 3.21: Micro analisi sulle prestazioni di IR per orario in linea URUS.

Vengono riportati di seguito alcune parti cruciali del codice python relativi alla MICRO analisi sulla linea URUS.

```
# Computation of control_time
velocity['control_time']=velocity.StopCheckAWSTime-velocity.
    StartCheckAWSTime
velocity['control_time']=velocity['control_time']/1000

# Convertions of Unix timestamp
velocity['StartCheckAWSTime'] = pd.to_datetime(velocity['
    StartCheckAWSTime'],unit='ms')
velocity['StopCheckAWSTime'] = pd.to_datetime(velocity['
    StopCheckAWSTime'],unit='ms')


velocity=velocity[['SFC','Workcenter','PartNumber','Mercato','
    TimeStamp','StartCheckAWSTime','StopCheckAWSTime','control_time
    ']]
velocity.head()

# Computaion of mean and median control times
ct_mean_velocity=velocity.groupby(['PartNumber','Workcenter']).
    mean('control_time')[['control_time']]
ct_mean_velocity=ct_mean_velocity.rename(columns={'control_time':'
    mean_control_time (s)'})

ct_median_velocity=velocity.groupby(['PartNumber','Workcenter']).
    median('control_time')[['control_time']]
ct_median_velocity=ct_median_velocity.rename(columns={'
    control_time':'median_control_time (s)'})

fig = px.bar(ct_mean_velocity.reset_index(), x="PartNumber", y="
    mean_control_time (s)",
             color='Workcenter', barmode='group',
             height=600)
fig.update_layout( xaxis={'categoryorder':'total descending'}).
    update_traces(hovertemplate='%{y}')

fig.update_layout(barmode='stack', xaxis={'categoryorder':'total
    descending'}).update_traces(hovertemplate='%{y}')

fig = px.sunburst(ct_mean_velocity.reset_index(), path=['
    PartNumber', 'Workcenter'], values='mean_control_time (s)')
fig.update_traces(textinfo="label+percent entry")
```

3.5 Difetti e Part Number

Quest'analisi, ha lo scopo di evidenziare le correlazioni tra i Difetti di produzione, registrati a MES e i Part Number su cui è stata applicata l' etichetta controllata dal sistema IR.
Questo serve per verificare se eventuali processi di rework per la correzione del difetto, possono aver compromesso in qualche modo le etichette applicate.
La tabella di origine per l'analisi è la Z_NC_IR_DATA, che contiene le informazioni sui difetti identificati dall'algoritmo di Riconoscimento Immagini, combinata con la z_ir_dataregistry per recuperare il modello dell'auto. Le due tabelle possono essere unite in join utilizzando il campo VIN (che identifica univocamente una vettura a livello globale).
L'analisi mira a scoprire i Part Number che presentano più difetti in relazione agli errori presentati dall'algoritmo IR per le etichette applicate su questi part number.

ID	VIN	Timestamp	Part Number	Model
1	ZPBEB4ZL1PLA25169	2023-04-05 09:52:36	TARGHETTA_CAMBIO_CINA	Urus
2	ZPBEB4ZL1PLA25169	2023-04-05 09:53:12	TARGHETTA_CAMBIO_CINA	Urus
3	ZPBEB4ZL1PLA25169	2023-04-05 09:54:12	400.010.530.A	Urus
4	ZPBEB4ZL1PLA25169	2023-04-05 09:54:51	CENTRALINA_AIRBAG_CINA	Urus
5	ZPBCC3ZLXPLA25167	2023-04-05 12:56:20	4ML.010.515.A	Urus

Figura 3.22: Tabella contenente i difetti associati ad un Part Number

La tabella sopra riportata ovviamente mostra solo una parte dei 3392 records estratti, ognuno di essi rappresenta un difetto riscontrato a MES corrispondente ad una label, che ha generato un errore sul sistema IR.
I record totali per questa tabella sono 3392. Dopodiché abbiamo proceduto raggruppando i dati per Part Number (quindi per componente), ordinando i record per maggior numero di difetti trovati su

ogni part number.
Abbiamo selezionato i primi 20 e ci siamo concentrati su di essi.

Part Number	Model	Number of records
4ML.012.785.MA	Urus	812
4F0.010.548	Urus	292
8E0.010.352.P	Urus	287
9Y0.010.539.A	Urus	252
4ML.010.515.A	Urus	214
470.010.001.E	Urus	158
4T0.010.556	Urus	95
7L0.010.872.L	Urus	94
AKL.437.040	Urus	69
4ML.010.500	Urus	67
4ML.010.502.AJ	Urus	65
4ML.010.560.A	Urus	65
4ML.010.533.E	Urus	60
4ML.010.525.J	Urus	57
4ML.010.525.G	Urus	56
4ML.010.502.AD	Urus	52
47B.010.129	Urus	52
4ML.010.525.L	Urus	45
4T0.010.012	Urus	39
4ML.010.541.A	Urus	37

Figura 3.23: Difetti raggruppati per Part Number

Dalla tabella sopra possiamo notare come :

- Ci sono sei Part number, che coprono il 59% di tutti i dati, con più di 100 record con errori

- Tutti i principali Numeri di Parte errati sono per Urus

3.5.1 Distribuzione dei difetti in relazione al Part Number

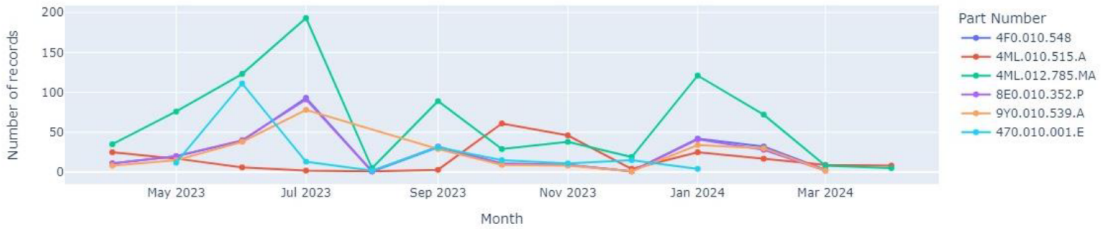

Figura 3.24: Distribuzione dei difetti nel tempo

La trend line rappresenta il numero di errori per ogni mese per i sei Part Number con più errori.
Guardando i grafici, per tutti i Part Number i grafici sono fluttuanti e non si possono trovare comportamenti particolari
Per il Part Number **470.010.001.E** non sono stati trovati errori di accuratezza dopo gennaio 2024.

Part Number	Model	Model Type	Market	Number of records	Percentage of records per PartNumber and Market
47B.010.525.E	Revuelto	L744	EUROPA	29	80,56%
4ML.012.785.MA	Urus	URSS	CINA	69	14,17%
4ML.012.785.MA	Urus	URSD	INDIA	10	12,82%
4T0.010.556	Urus	URSS	CINA	80	12,60%
4ML.012.785.MA	Urus	URSD	TAIWAN	12	12,50%
4ML.012.785.MA	Urus	URSS	FRANCIA	17	11,64%
4ML.012.785.MA	Urus	URSD	SVIZZERA	12	9,30%
4ML.012.785.MA	Urus	URSS	CANADA	21	9,29%
4F0.010.548	Urus	URSS	CANADA	21	9,21%
8E0.010.352.P	Urus	URSS	CANADA	21	9,21%
4ML.012.785.MA	Urus	URSD	GIAPPONE	35	8,45%
4ML.012.785.MA	Urus	URSS	GERMANIA	46	8,20%
4ML.012.785.MA	Urus	URSS	AGCC	19	7,98%
4ML.012.785.MA	Urus	URSD	CANADA	18	7,96%
8E0.010.352.P	Urus	URSD	CANADA	18	7,89%
4F0.010.548	Urus	URSD	CANADA	18	7,89%
4ML.012.785.MA	Urus	URSD	ITALIA	19	7,51%
4ML.010.525.J	Urus	URSD	GERMANIA	13	6,91%
4ML.012.785.MA	Urus	URSS	ITALIA	17	6,72%
4F0.010.548	Urus	URSS	USA	129	6,64%

Figura 3.25: Distribuzione dei difetti in relazione al mercato di vendita della vettura

Analizzando invece i 20 Part Number che che presentano più errori in relazione al Modello, Tipo di Modello, Mercato e numero di errori e Percentuale di errore rispetto al Part Number e Mercato, possiamo vedere come il Mercato che presente più errori è il mercato USA.

3.5.2 Analisi di correlazione Reowrk - Errori IR

Un aspetto molto interessante da analizzare è la correlazione tra i difetti rilevati sulle vetture, che hanno portato a un **rework** (rifacimento di un'operazione per correggere il difetto), e i difetti riscontrati dal sistema di Riconoscimento Immagini (IR) nell'analisi delle etichette.

Abbiamo messo in relazione questi due fattori per verificare quante volte un determinato rework è associato a un difetto di IR su un componente specifico. Questo ci può aiutare a comprendere se effettivamente un rework può danneggiare le etichette.

I dati emersi sono riportati nella tabella sottostante. È interessante notare come un rework sul sistema di avvitatura sia correlato con ben 333 errori del sistema IR, come evidenziato nel primo record della tabella.

ThingWorx Position	MES Position	Model	Number of records
Applicata su portiera lato guidatore	Sistema di Avvitatura	Revuelto	333
Zona porta inferiore SX	EQUIS	Revuelto	312
Zona porta inferiore SX	Sistema di Avvitatura	Revuelto	227
TBD	Sistema di Avvitatura	Revuelto	198
Zona porta inferiore SX	SFC-PREASSEMBLY	Revuelto	111
Zona porta inferiore SX	DATA COLLECTION	Revuelto	89
Zona porta inferiore SX	BAUGRUPPE	Revuelto	88
Applicata su portiera lato guidatore	EQUIS	Revuelto	85
Zona porta inferiore SX	RIFACIMENTO	Revuelto	65
TBD	EQUIS	Revuelto	56
Applicata su portiera lato guidatore	SFC-PREASSEMBLY	Revuelto	52
APRIRE COFANO POSTERIORE, APPLICATA INTERNAMENTE DX	Sistema di Avvitatura	Revuelto	43
APRIRE COFANO POSTERIORE, APPLICATA INTERNAMENTE SX	Sistema di Avvitatura	Revuelto	43
Aprire cofano posteriore	Sistema di Avvitatura	Revuelto	41
Zona porta inferiore SX	001_SETUP	Revuelto	40
Applicata su portiera lato guidatore	BAUGRUPPE	Revuelto	35
APRIRE COFANO POSTERIORE, APPLICATA INTERNAMENTE DX	SFC-PREASSEMBLY	Revuelto	33
APRIRE COFANO POSTERIORE, APPLICATA INTERNAMENTE SX	SFC-PREASSEMBLY	Revuelto	33
Applicata su portiera lato guidatore	RIFACIMENTO	Revuelto	31
Zona porta inferiore SX	000_DTC_CHECK	Revuelto	28

Figura 3.26: Rework associati agli errori rilevati da IR su thingworx

Riporto di seguit l'analisi in codice Python per trovare le correlazioni Rework - difetti IR

```
# define the filter to select only the damaged/wrong labels

filter_1 = (z_irdataregistry['UtenteCorrettezza'] == 0)
filter_2 = ((z_irdataregistry['UtenteCorrettezza'] == -1) & (
    z_irdataregistry['AWSCorrettezzaEsito'] == 0)))
z_irdataregistry_label_damaged = z_irdataregistry[filter_1 |
    filter_2][['SFC', 'DescrizionePosizione','Timestamp','
    VehicleFile']]
z_irdataregistry_label_damaged.head()

# select only the column of interest for the NC_GROUP_INFO
NC_GROUP_INFO_RED = NC_GROUP_INFO[['SFC','NC_GROUP_DESCRIPTION','
    PARTITION_DATE','CLOSED_WORKCENTER']].rename(columns= {'
    PARTITION_DATE' : 'ReworkTimestamp'})
NC_GROUP_INFO_RED['F_REWORK'] = 1

data_2 = z_irdataregistry_label_damaged.merge(NC_GROUP_INFO_RED,
    on = ['SFC'],how='inner')

# filter only the reworks made before the labels checks
data_2 = data_2[data_2.Timestamp>data_2.ReworkTimestamp]
data_2['DescrizionePosizione'].unique()

# uniforming the field DescrizionePosizione where the name is
    equivalent
data_2['DescrizionePosizione'] = np.where(data_2['
    DescrizionePosizione']=='Aprire cofano,lato dx','Aprire cofano,
     lato dx',data_2['DescrizionePosizione'])

## grouped distribution
data_2_gr_model = data_2.groupby(['DescrizionePosizione','
    NC_GROUP_DESCRIPTION', 'VehicleFile','CLOSED_WORKCENTER']).agg
    ({'F_REWORK' : 'sum'}).reset_index().sort_values(by='F_REWORK',
    ascending=False)
data_2_gr_model.head(20)
```

Capitolo 4

Conclusioni

Una volta completate tutte le analisi, i risultati sono stati raccolti e presentati in diverse sezioni, suddivise per area di analisi (colore, efficienza, tempo di controllo, ecc.). Ora, ripercorriamo gli obiettivi iniziali dell'analisi per verificare se sono stati raggiunti:

1. Determinare se le **performance del sistema IR dipendono dal colore** dei telai delle vetture.
2. Verificare se il sistema IR classifica come corrette alcune etichette "errate" o "scrap", con l'obiettivo di **aumentare la soglia di taglio (treshold)**, relativa alla confidenza dell'algoritmo IR.
3. Individuare eventuali **correlazioni tra workcenter e tempo di controllo**, ovvero il tempo di analisi di IR per ogni etichetta.
4. Analizzare se i **tempi di analisi variano** per lo stesso tipo di etichette e identificare i fattori che possono influire su queste variazioni.
5. Valutare se le performance di IR variano in base al **componente** su cui sono applicate.
6. Monitorare eventuali variazioni rapide nel tempo di controllo delle etichette, per **identificare possibili delta** significativi in determinate ore della produzione rispetto al tempo medio di analisi.

Questi obiettivi ci hanno guidato attraverso l'intero processo di analisi, permettendoci di ottenere una visione chiara e dettagliata delle performance del sistema IR e delle sue potenziali aree di miglioramento.
Come descritto in dettaglio nel capitolo 3, le analisi sono state condotte su due modelli attualmente in produzione: Urus e Revuelto. Tuttavia, alcune analisi sono state eseguite esclusivamente sul modello Urus, poiché non erano disponibili dati sufficienti per il modello Revuelto, essendo quest'ultimo appena entrato in produzione.
Questa distinzione è stata necessaria per garantire l'accuratezza e la rilevanza dei risultati ottenuti, permettendoci di trarre conclusioni affidabili e utili per il miglioramento del sistema IR, in quanto questo è il vero unico obiettivo di tutto questo progetto di analisi.

4.1 Correlazioni rilevate e Assenze di evidenze

Prima dell'inizio del progetto SurfDesk, il sistema IR già vantava ottime prestazioni. Tuttavia, in molti casi, l'utente doveva comunque fornire un input manuale per validare o meno il corretto controllo dell'etichetta omologativa. Questo processo veniva ripetuto in ogni fase di verifica e controllo della vettura, coinvolgendo quasi tutta la linea di montaggio.
L'obiettivo principale del progetto era determinare se questi errori del sistema IR fossero dovuti esclusivamente a foto scattate male dagli operatori, che causavano un controllo errato da parte degli algoritmi di computer vision, oppure se fossero il risultato di bug o malfunzionamenti dell'algoritmo IR stesso.
Abbiamo analizzato tutte le aree di interesse toccate da questo sistema. Ogni KPI e ogni prestazione dell'algoritmo sono stati esaminati, considerando però solo i dati provenienti da foto correttamente scattate. Abbiamo volutamente **escluso tutte le foto con errori di messa a fuoco o inquadratura**, concentrandoci esclusivamente sui casi in cui l'algoritmo ha ricevuto un input valido da processare senza difetti.

Questo approccio ci ha permesso di isolare le vere cause degli errori e di valutare con precisione le performance del sistema IR, fornendo una base solida per eventuali miglioramenti futuri.

4.1.1 Correlazione tra Colore e Performance del Sistema IR:

Le analisi non hanno evidenziato una correlazione significativa tra il colore del telaio delle vetture e le performance del sistema IR. La distribuzione degli errori non ha mostrato variazioni rilevanti in funzione del colore, suggerendo che il sistema IR mantiene un livello di accuratezza costante indipendentemente dal colore del telaio. I putni chiave che ci hanno permesso di poter confermare quanto appena scritto con certezza sono due:

- **Distribuzione dei dati e degli errori:** Considerando la distribuzione dei dati per una determinata categoria (ad esempio, famiglia di colori), se non esiste alcuna correlazione, la distribuzione degli errori per la stessa categoria dovrebbe rispecchiare la distribuzione dei dati originali. Le analisi hanno mostrato che, per la maggior parte delle categorie di colore, la proporzione di errori è simile alla proporzione dei dati, indicando l'assenza di una correlazione significativa tra colore e performance del sistema IR.

- **Eccezioni e anomalie:** Alcuni part number e famiglie di etichette hanno mostrato una distribuzione degli errori significativamente più alta rispetto alla distribuzione dei dati. Tuttavia, queste anomalie non sono risultate correlate al colore del telaio, ma piuttosto a specifiche caratteristiche dei componenti o delle etichette stesse.

In conclusione, per quanto riguarda il colore, possiamo dire che le performance del sistema IR non dipendono in modo dal colore del telaio delle vetture.

4.1.2 Correlazioni relative al Tempo di controllo

Andiamo ad analizzare tutti i fattori di correlazioni relativi al tempo di controllo.
Stando ai risultati della **Macro analisi**, osserviamo che

- L'analisi delle correlazioni tra i diversi tipi di modello del URUS e i workcenter ha mostrato che il tempo medio di controllo varia tra i workcenter. Ad esempio, il centro di lavoro FLU_REW_A impiega circa 20 secondi per ogni numero di parte, mentre il centro di lavoro Q_TLU_01_C impiega circa 25-30 secondi per Part Number

- Per il modello Revuelto, il centro di lavoro Q_TL_01_A impiega circa 35 secondi per ogni numero di parte, ma il numero di record è molto basso. Il centro di lavoro Q_TL_01_C richiede 27 secondi per controllare i numeri di parte.

Anche la **Micro analisi** ha confermato, fornendo esempi più dettagliati, quanto specificato nella Macro

- La micro analisi ha confermato che le prestazioni di IR non dipendono dal componente analizzato, ma ci sono variazioni nei tempi medi e mediani di controllo tra i workcenter. Ad esempio, i tempi medi e mediani di controllo sono molto vicini per i due centri di lavoro analizzati (FLU_REW_A e Q_TLU_01_C).

- Analizzando le prestazioni ora per ora, si è osservato che ci sono sovrapposizioni tra le curve dei tempi medi di controllo dei due workcenter in determinate fasce orarie, suggerendo che i cali di prestazioni potrebbero essere dovuti a problemi di rete o altre variabili. Tuttavia non sono emersi cali drastici delle prestazioni di IR, questo significa che, basandoci sullo storico di dati di più di un anno, la forma dell'etichetta o le keyword contenute all'interno non causano rallentamenti al sistema dato che non influiscono sul tempo di analisi dell'etichetta.

4.1.3 Dipendenza da altri fattori esterni

- **Mercato di Vendita:**
  I tempi medi di controllo per veicolo sono influenzati dal mercato di vendita. Ad esempio, i controlli per il mercato portoghese sono risultati più lenti rispetto ad altri paesi a causa di grandi valori anomali. I controlli per il mercato vietnamita sono i più lenti, ma per ogni paese il tempo mediano è approssimativamente inferiore ai 30 secondi.

- **Part Number:**
  Anche il part number può influenzare il tempo di controllo. Ad esempio, alcuni part number hanno mostrato una distribuzione degli errori significativamente più alta rispetto alla distribuzione dei dati, indicando che specifiche caratteristiche dei componenti possono influenzare le prestazioni del sistema IR.

In generale, il tempo di controllo del sistema IR dipende da vari fattori esterni, tra cui il **workcenter**, il **mercato di vendita** e il **part number**. Questi fattori possono influenzare le prestazioni del sistema e causare variazioni nei tempi di controllo. Le variazioni di prestazione però sono molto lievi. In generale non esiste un fattore che penalizzi in modo pesante le prestazioni di IR, ma sono tutti fattori con un basso impatto.

4.1.4 Livello attuale di precisione

La precisione del **modello Urus** è 1, poiché non ci sono falsi positivi, e il richiamo del modello è 0,98511. Questi valori indicano che il modello funziona molto bene.
Per il **modello Revuelto**, la precisione del modello è 1 e il richiamo del modello è 0,9987. Questi valori sono eccezionali, ancora migliori del caso Urus, confermando che il modello funziona in modo eccellente.
L'analisi ha mostrato che l'algoritmo IR ha già un livello di confidenza molto alto. La soglia attuale è stata ottimizzata per garantire che le

analisi con una confidenza superiore all' 85% siano sempre corrette. Modificare ulteriormente la soglia potrebbe non portare a miglioramenti significativi, dato che il sistema già raggiunge un'elevata precisione e richiamo. Alzare la soglia potrebbe ridurre il numero di falsi positivi, ma potrebbe anche aumentare il numero di falsi negativi, compromettendo l'efficacia complessiva del sistema.

Basandosi sulle analisi condotte nel progetto SurfDesk, emerge chiaramente che il sistema di riconoscimento delle immagini (IR) di Lamborghini già vanta ottime prestazioni. Tuttavia, l'unico motivo che causa errori nel riconoscimento o scarti delle immagini, portando alla necessità di una verifica manuale delle etichette, è dovuto a errori nello scatto delle foto da parte degli operatori. Pertanto, la soluzione che può portare i maggiori vantaggi è eseguire un training specifico agli operatori per migliorare l'utilizzo del sistema e dell'applicazione IR. Questo approccio consentirebbe di ridurre significativamente gli errori umani, migliorando l'efficienza operativa e garantendo che le etichette omologative siano sempre correttamente riconosciute dal sistema IR. Un training adeguato agli operatori non solo ottimizzerebbe l'accuratezza del sistema, ma ridurrebbe anche il tempo necessario per le verifiche manuali.

Elenco delle figure

Ringraziamenti

Con la conclusione di questa tesi si chiude anche il mio percorso accademico. Sono stati cinque anni ricchi di emozioni, sacrifici, gioie, tristezze, ma soprattutto persone. La cosa che più amo fare è circondarmi delle persone a cui tengo, e se oggi sei qui ad ascoltare queste poche parole, significa che fai parte di loro. Decidi tu se considerarlo una fortuna o meno.

Quanto ho amato e odiato l'università, lo sanno davvero in pochi. Tante volte ho pensato di mollare tutto. Eppure, se oggi ho la fortuna di scrivere questa tesi in una delle aziende più affascinanti e stimolanti al mondo, è proprio grazie all'impegno e al percorso universitario.

Grazie a tutti i miei amici, che avete sempre saputo calmarmi ed incoraggiarmi nei momenti di crisi in cui tutto mi sembrava impossibile. Grazie a tutti quelli che invece hanno festeggiato con me ogni piccolo traguardo raggiunto. Grazie a chi c'è sempre stato. Grazie Mamma e Papà per non avermi mai lasciato solo. È vero, non vi siete mai interessati troppo ai corsi, ai voti o ai tirocini, ma mi avete sempre fatto sentire al sicuro. Ho sempre percepito quella protezione che solo un genitore può offrire al proprio figlio, sia nei momenti difficili che in quelli di gioia, e per me questo era quello di cui ho avuto bisogno. Nulla di più.