Report di Data Analysis su Google Play Store

Domenico Antonio Izzo, Ciro Maccarone, Adelio Antonini

June 20, 2023

1 Introduzione

Il Google Play Store è una piattaforma di distribuzione digitale sviluppata da Google per dispositivi Android. Con milioni di applicazioni disponibili, offre agli utenti un'ampia gamma di giochi, app, libri, film e altro ancora. Il Play Store gioca un ruolo cruciale nell'ecosistema Android, consentendo agli sviluppatori di raggiungere un vasto pubblico e agli utenti di scoprire e scaricare applicazioni che soddisfano le loro esigenze.

Per il nostro progetto di data analysis, abbiamo utilizzato il dataset ottenuto dalla piattaforma Kaggle al url :

https://www.kaggle.com/datasets/gauthamp10/google-playstore-apps .

Tuttavia, il dataset si è rivelato poco adatto per le parti di clustering, autoregressione e classificazione. Nonostante questa sfida, ci si è impegnati a riadattare lo stesso dataset invece che scegliere differenti dataset già predisposti a questi task, cercando di ottenere i migliori risultati possibili, ottenendo buoni risultati nella parte di classificazione e autoregressione.

2 Descrizione del Dataset

Il dataset utilizzato è stato fornito in formato .csv e contiene i dati che vanno dal 28 gennaio del 2010 al 16 giugno del 2021. Il file principale, denominato "Google-Playstore.csv", ha una dimensione di 676,5 MB. Il dataset comprende un totale di 24 attributi, tra cui: "App Name", "App Id", "Category", "Rating", "Rating Count", "Installs", "Minimum Installs", "Maximum Installs", "Free", "Price", "Currency", "Size", "Minimum Android", "Developer Id", "Developer Website", "Developer Email", "Released", "Last Updated", "Content Rating", "Privacy Policy", "Ad Supported", "In App Purchases", "Editors Choice" e "Scraped Time".

Durante l'analisi, siamo stati particolarmente interessati a diversi attributi chiave. Questi includono "Category" (che fornisce informazioni sulla categoria dell'app), "Rating" (il punteggio di valutazione dell'app), "Rating Count" (il numero di valutazioni ricevute dall'app), "Ad Supported" (che indica se l'app supporta annunci pubblicitari), "In App Purchases" (che indica se l'app offre acquisti in-app), "Maximum Installs" (il numero massimo di installazioni dell'app), "Price" (il prezzo dell'app), "Size" (le dimensioni dell'app), e "Released" (la data di rilascio dell'app).

È importante notare che abbiamo scelto di utilizzare l'attributo "Maximum Installs" anziché "Installs" perché i dati sembravano più congruenti e precisi. Alcuni dati presenti nell'attributo "Installs" erano inesatti, mentre "Maximum Installs" sembrava fornire valori più affidabili e coerenti.

Inoltre il numero totale di app registrate nel dataset è di 2 312 945. Questo ampio numero di app può rendere computazionalmente dispendioso il calcolo di alcuni task. Di conseguenza, in alcuni casi abbiamo estratto campioni di dati rappresentativi dal dataset per ridurre il carico computazionale e garantire tempi di esecuzione accettabili per le analisi.

3 ETL (Extract, Transform, Load)

Durante la fase di ETL, abbiamo seguito diversi passaggi per preparare il dataset per l'analisi. Inizialmente, abbiamo estratto le colonne di interesse che volevamo utilizzare nella nostra analisi.

Successivamente, abbiamo calcolato il numero di valori nulli in ogni colonna del dataset. I risultati sono riportati nella tabella seguente:

	Valori mancanti	**Percentuale mancante**
App Name	2	0.000086
Category	0	0.000000
Rating	22,883	0.989345
Rating Count	22,883	0.989345
Ad Supported	0	0.000000
In App Purchases	0	0.000000
Maximum Installs	0	0.000000
Size	196	0.008474
Released	71,053	3.071972
Price	0	0.000000

Successivamente, abbiamo eliminato le righe che presentavano valori nulli in una o più colonne, poiché queste righe non erano utili per le nostre analisi.

Per quanto riguarda la colonna "Size", i dati erano rappresentati come stringhe utilizzando le metriche dei Byte. Abbiamo trasformato questi valori in notazione scientifica (ad esempio, da "10M" a "10e6") e successivamente li abbiamo convertiti da tipo (oggetto) String a tipo float64.

Le colonne "Ad Supported" e "In App Purchases" erano dei flag booleani. Abbiamo semplicemente trasformato questi flag in valori interi, assegnando 0 per indicare la mancanza del flag e 1 per indicare la presenza del flag.

Successivamente, abbiamo voluto semplificare la visualizzazione dei valori categorici della colonna "Category". Poiché le categorie originali erano numerose, le abbiamo raggruppate in quattro macrocategorie principali per scopi di visualizzazione: "Education", "Tools", "Entertainment" e "Lifestyle".

```
Education=['Education','Books & Reference','News & Magazines','Educational',
           'Art & Design','Word','Libraries & Demo','Parenting']
Tools=['Tools','Trivia','Personalization','Photography','Maps & Navigation',
       'Video Players & Editors','Weather','Communication','Finance','Business','Events']
Entertainment=['Entertainment','Music & Audio','Arcade','Puzzle','Casual','Action',
               'Simulation','Adventure','Auto & Vehicles','Board','Racing',
               'Role Playing','Strategy','Card','Casino','Music','Comics']
Lifestyle=['Lifestyle','Health & Fitness','Productivity','Shopping','Food & Drink',
           'Travel & Local','Sports','Medical','House & Home','Beauty','Dating','Social']
```

Figure 1: Macro-Categorie usate al posto delle categorie originali

Per ulteriori pulizie dei dati, abbiamo deciso di filtrare il dataset originale creando una copia del dataframe in cui abbiamo eliminato le app considerate "junk". Abbiamo definito come app "junk" quelle con un numero di installazioni inferiore a 10 e un conteggio delle valutazioni pari a 0.

Successivamente, abbiamo creato una nuova colonna chiamata "Appreciation Index" a fini di visualizzazione. Questa colonna è stata ottenuta moltiplicando il punteggio normalizzato attraverso la classe MinMaxScaler del Rating dell'app per il conteggio delle valutazioni ricevute.

Infine, sia per il Dataframe contenente le app "junk" sia per il Dataframe filtrato, abbiamo utilizzato la funzione "get_dummies" per trasformare i dati categorici della colonna "Category" in dati numerici. Questa operazione ha generato quattro nuove colonne, ciascuna rappresentante una macrocategoria di interesse ("Education", "Entertainment", "Lifestyle" e "Tools"). I valori 0 e 1 sono stati assegnati per

indicare l'appartenenza o meno a una specifica categoria. Il risultato finale del dataframe è mostrato nella tabella seguente:

Colonna	**Tipo di dati**
App Name	object
Rating	float64
Rating Count	int64
Ad Supported	int64
In App Purchases	int64
Installs	int64
Size	float64
Released	int64
Price	float64
Appreciation Index	float64
Category_Education	uint8
Category_Entertainment	uint8
Category_Lifestyle	uint8
Category_Tools	uint8

4 Visualizzazione dei dati

Nella fase di filtraggio del dataset, abbiamo ottenuto un numero ridotto di app per ciascuna categoria. La distribuzione delle app per categoria nel dataset filtrato è riportata nella tabella seguente:

Categoria	**Numero di App**
Education	249,073
Entertainment	356,054
Lifestyle	284,321
Tools	313,182

Complessivamente, il dataset filtrato contiene 882,230 righe, un numero significativamente inferiore rispetto alle oltre 2 milioni di app inizialmente considerate. Per visualizzare la distribuzione delle app per categoria, abbiamo creato un grafico a torta come mostrato di seguito:

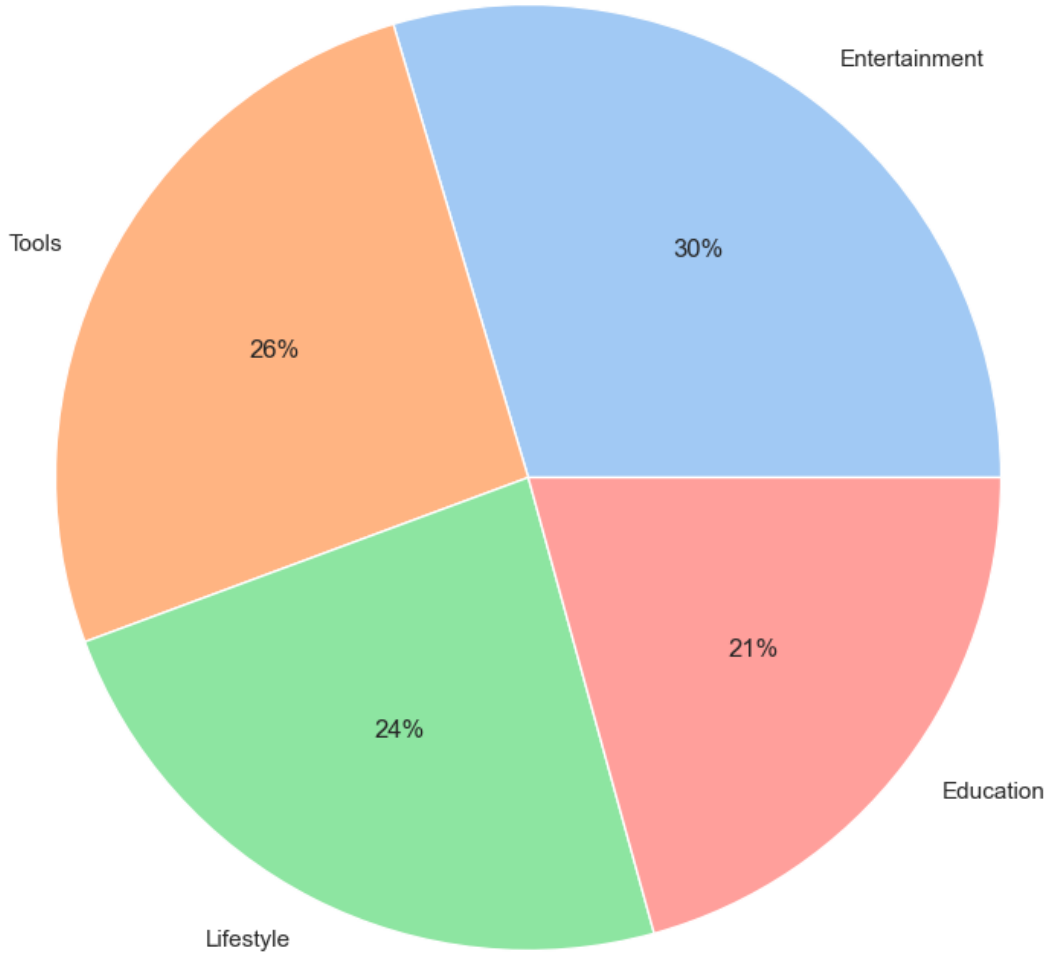

Figure 2: Distribuzione delle applicazioni dell'Google Play Store per macro-categoria

In secondo luogo, abbiamo esaminato le 10 migliori app in base al rating, al rating count e all'appreciation index (rating x rating count). Abbiamo notato che il rating da solo non è un indicatore affidabile, poiché ci sono molte app con un rating di 5/5 ma con un rating count molto basso. Pertanto, abbiamo applicato una doppia selezione: abbiamo prima filtrato le app con un rating di 5/5 e poi abbiamo selezionato le prime 10 app con il maggior rating count tra queste. La lista delle app selezionate è la seguente:

App	Categoria
STUDiLMU: Your One-stop Solution Learning Partner	Education
Crazy Fall	Entertainment
Calculator Plus	Lifestyle
白沙屯媽祖 GPS 即時定位	Tools
Жәннат Фирдаус - Намаз, Құран оқуды үйрену	Education
Last Bird Trip - Free Flappy Birdie Arcade Game	Entertainment
Hayalhanem	Education
21 Dias de Jejum (Jejum de Daniel)	Education
꾸매영 (꾸준하게 매일 빠세게 영어): 무료로 영어공부...	Education
Babel Novel - Fantastic Books & Webnovel Reading	Education

Come si può notare, queste app non sono molto conosciute, il che indica che il rating da solo non è un buon indice di popolarità. Invece, l'app con l'appreciation index più alto è al primo posto con un valore di 30964, mentre l'app con il rating count più alto ha un valore di 42,908, entrambi valori molto al di sotto di diversi ordini di grandezza rispetto ai valori dei primi posti per numero di valutazioni e indice di apprezzamento. Guardando poi le app per Indice di apprezzamento da noi introdotto e Rating Count possiamo notare come tali due tabelle mostrano una notevole sovrapposizione, con la maggior parte delle app presenti in entrambe le liste, seppur con posizioni differenti. WhatsApp Messenger, YouTube e Instagram sono le prime tre app comuni a entrambe le tabelle, YouTube e Instagram si contendono la seconda posizione. Le differenze significative tra le due tabelle si manifestano principalmente nell'ultima posizione, dove Candy Crush Saga appare solo nella tabella dell'Appreciation Index, mentre Google Play Services compare solo nella tabella del Rating Count. In generale, entrambe le tabelle forniscono un'idea della popolarità delle applicazioni.

Top 10 applicazioni per Indice di apprezzamento:

App	Categoria
WhatsApp Messenger	Tools
YouTube	Tools
Instagram	Lifestyle
Garena Free Fire - Rampage	Entertainment
Messenger – Text and Video Chat for Free	Tools
Clash of Clans	Entertainment
TikTok	Lifestyle
Google Photos	Tools
PUBG MOBILE - Traverse	Entertainment
Candy Crush Saga	Entertainment

Top 10 applicazioni per numero di valutazioni:

App	Categoria
WhatsApp Messenger	Tools
Instagram	Lifestyle
YouTube	Tools
Garena Free Fire - Rampage	Entertainment
Messenger – Text and Video Chat for Free	Tools
Clash of Clans	Entertainment
PUBG MOBILE - Traverse	Entertainment
TikTok	Lifestyle
Google Photos	Tools
Google Play services	Tools

Dopo aver analizzato la distribuzione delle applicazioni nel dataset, è emerso che alcuni attributi presentano una maggiore variabilità e possono essere di particolare interesse per le attività di clustering e classificazione. Tuttavia, è stato osservato che l'attributo "Price" ha una variazione limitata e risulta poco impattante nelle analisi. Sebbene le app a pagamento potrebbero avere un rating medio più elevato, rappresentano meno dell'1% dell'intero dataset. Pertanto, l'attributo "Price" verrà mantenuto, ma non avrà un ruolo significativo nelle nostre analisi.

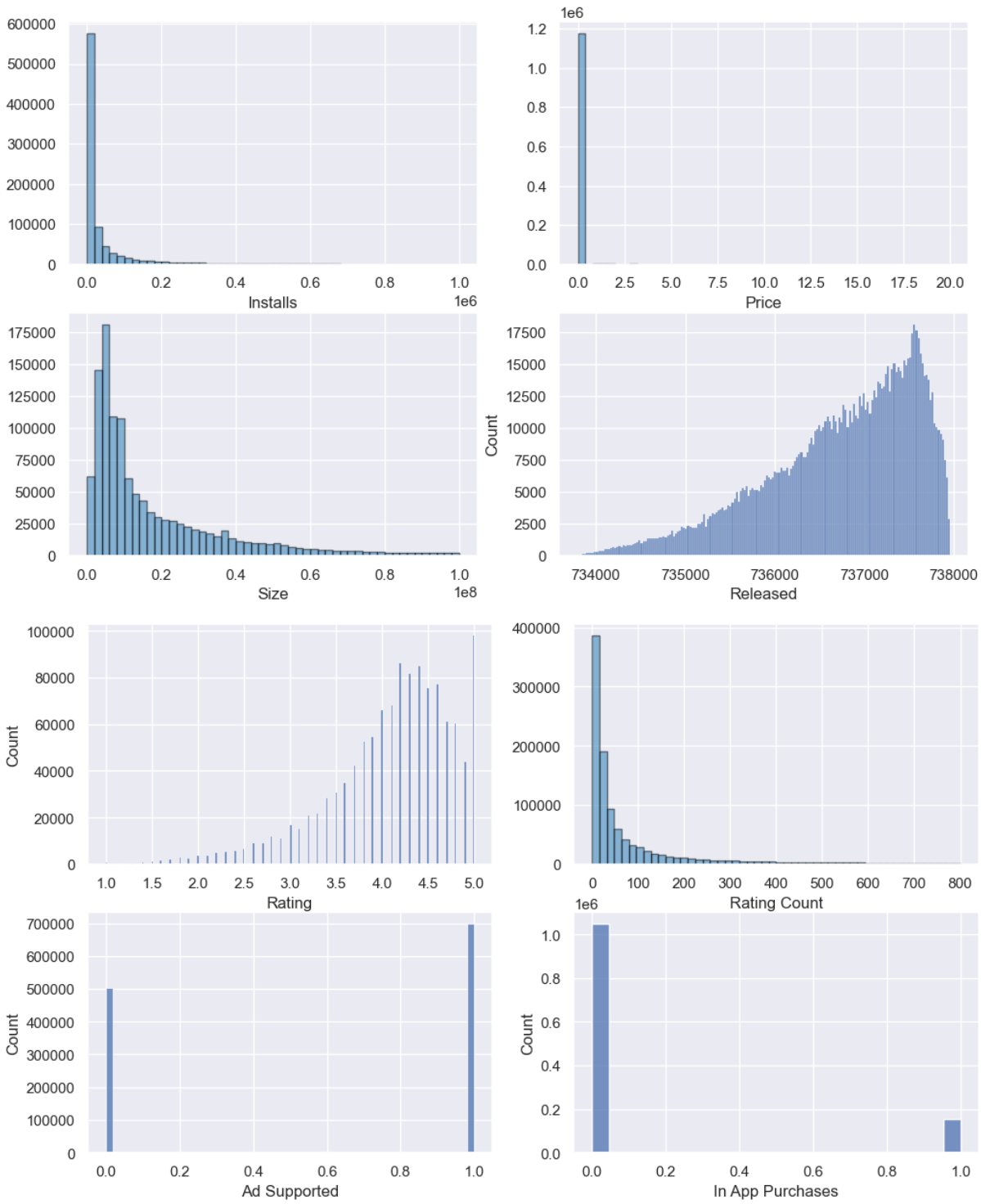

Figure 3: Distribuzione delle features

Al contrario, gli attributi "In App Purchases", "Ad Supported" e "Rating" mostrano una maggiore variabilità leggermente maggiore, rendendoli più rilevanti per le nostre analisi.
Successivamente, è stata calcolata la correlazione tra le colonne del dataset al fine di identificare eventuali ridondanze che potrebbero influire sullo studio di classificazione e clustering.

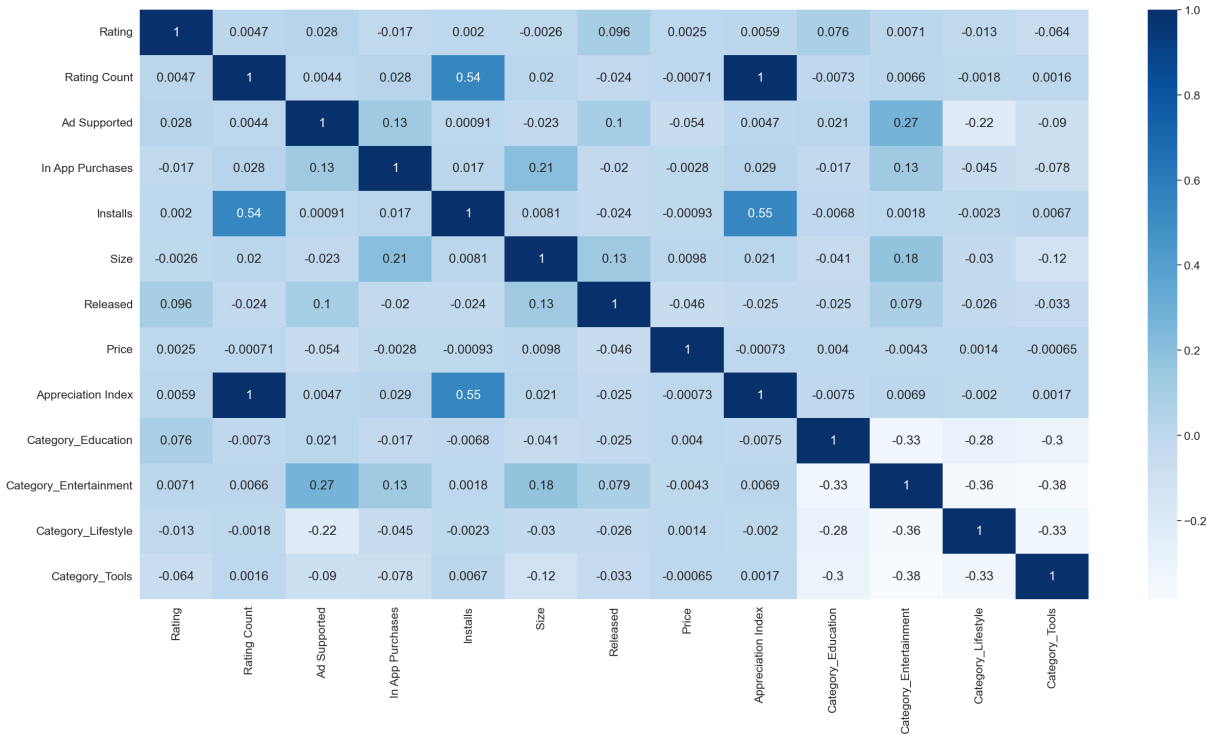

Figure 4: Correlazione per il Dataset pulito dalle app considerate "junk".

Si può osservare che il numero di installazioni ("Installs") presenta una correlazione del 54% con il numero di valutazioni ("Rating Count") e del 55% con l'Appreciation Index. Pertanto, abbiamo deciso di mantenere la colonna "Rating Count" per due motivi. In primo luogo, l'Appreciation Index è calcolato utilizzando il Rating, quindi non sarebbe corretto utilizzarlo come feature per determinare il rating durante la fase di classificazione. In secondo luogo, il numero di installazioni ("Installs") mostra una correlazione più debole con l'Appreciation Index e il Rating rispetto al numero di valutazioni ("Rating Count"), il che rende più logico mantenere quest'ultima come colonna di riferimento per la nostra classificazione basata sul Rating.

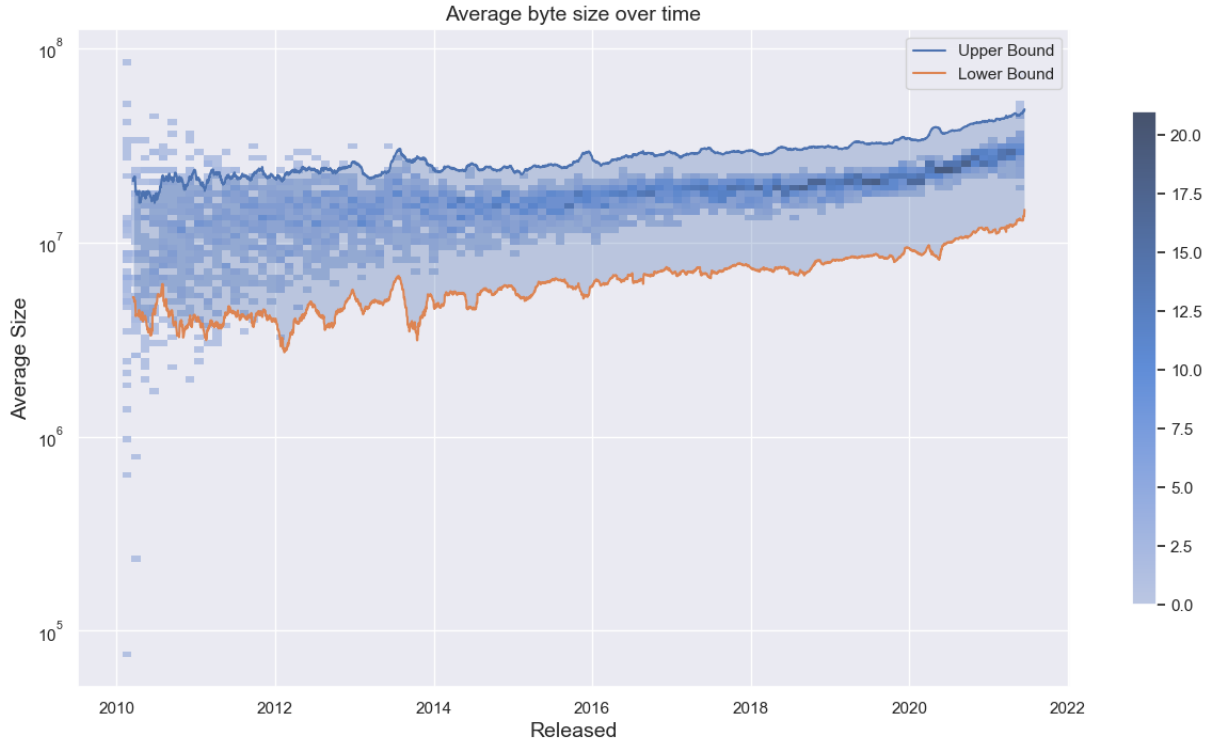

Figure 5: Trend nel tempo della dimensione delle App del Google Play Store.

Infine, per analisi autoregressiva le alternative prese in considerazione erano la variazione della dimensione delle app nel corso degli anni e la variazione del rating. Intuitivamente, si è scelto di considerare il parametro della dimensione delle app, poiché ci si aspetta che con l'aumento della capacità di memoria, le app diventino più complesse e pesanti, soprattutto nel settore del mobile gaming. Tuttavia, è importante notare che questo potrebbe essere in contrasto con un'aspettativa di maggiore efficienza in termini di memoria. In ogni caso, è possibile notare una maggiore convergenza nelle dimensioni delle nuove app rilasciate.

5 Modelli di Classificazione

Durante il processo di classificazione, sono stati adottati diversi modelli di machine learning e deep learning per analizzare il dataset. La versione del dataframe utilizzata comprendeva anche le app "junk" poiché l'obiettivo era classificare le app in base al rating, quindi è necessario mantenere anche la applicazioni con rating pessimo. Pertanto, il dataset è stato diviso in due parti: una contenente le applicazioni con un rating compreso tra 0 e 2 e l'altra contenente le applicazioni con un rating compreso tra 3 e 5, in tal modo è stato possibile trasformare il dato numerico del rating con un dato categorico, classificando le applicazioni come "popolari" (cioè con un rating di 3 o superiore) o "impopolari" (cioè con un rating di 2 o inferiore).

Successivamente, il dataset è stato bilanciato in modo che il dataset finale avesse lo stesso numero di applicazioni per entrambe le categorie "app popolari" e "impopolari", fortunatamente per queste due categorie il numero di applicazioni era pressoché uguale, mantenendo quindi circa 2 milioni di applicazioni nel dataset finale.

In seguito gli attributi "Installs" e "Appreciation Index" sono stati rimossi in quanto risultavano correlati con "rating count" come già descritto in precedenza. È stata eseguita quindi la normalizzazione del dataset utilizzando il MinMax scaler, garantendo che l'attributo "Rating count", che presentava una vasta variabilità, fosse normalizzato in un range di valori da 0 a 1e4 (altrimenti alcuni valori in un range più basso sembrerebbe dalle prove sperimentali essere azzerati del tutto, un altro tentativo sarebbe potuto essere trasformare il tipo di dato in float64).

I modelli di machine learning testati sono stati **Logistic Regression**, **Decision Tree Classifier**, **SVC**, **K Neighbors Classifier** e **Random Forest Classifier** della libreria scikit-learn. Inoltre, è stato utilizzato un modello di deep learning il classico **Multi-Layer Perceptron** (MLP).

Per valutare le prestazioni dei modelli, è stata effettuata una classificazione binaria e i risultati sono stati analizzati utilizzando diverse metriche, tra cui precisione, recall e F1-score, lo splitting del dataset è stato 80/10/10 rispettivamente per training, valutazione e testing.
Di seguito sono presentati i risultati ottenuti dai modelli di machine learning sul restante 10% del dataset di testing:

Logistic Regression Classification Report

	Precision	Recall	F1-Score	Support
0	0.68	0.95	0.79	108270
1	0.92	0.54	0.68	107775
Accuracy			0.75	216045
Macro Avg	0.80	0.75	0.74	216045
Weighted Avg	0.80	0.75	0.74	216045

Decision Tree Classifier Classification Report

	Precision	Recall	F1-Score	Support
0	0.98	0.96	0.97	108270
1	0.96	0.98	0.97	107775
Accuracy			0.97	216045
Macro Avg	0.97	0.97	0.97	216045
Weighted Avg	0.97	0.97	0.97	216045

SVC Classification Report

	Precision	Recall	F1-Score	Support
0	0.50	1.00	0.67	108270
1	1.00	0.01	0.02	107775
Accuracy			0.51	216045
Macro Avg	0.75	0.50	0.34	216045
Weighted Avg	0.75	0.51	0.34	216045

K Neighbors Classifier Classification Report

	Precision	Recall	F1-Score	Support
0	0.99	0.96	0.98	108270
1	0.96	0.99	0.98	107775
Accuracy			0.98	216045
Macro Avg	0.98	0.98	0.98	216045
Weighted Avg	0.98	0.98	0.98	216045

Random Forest Classifier Classification Report

	Precision	Recall	F1-Score	Support
0	0.99	0.96	0.98	108270
1	0.96	0.99	0.98	107775
Accuracy			0.98	216045
Macro Avg	0.98	0.98	0.98	216045
Weighted Avg	0.98	0.98	0.98	216045

Infine i risultati dal modello di deep learning:

MLPClassifier Classification Report

	Precision	Recall	F1-Score	Support
0	1.00	0.96	0.98	108270
1	0.96	1.00	0.98	107775
Accuracy			0.98	216045
Macro Avg	0.98	0.98	0.98	216045
Weighted Avg	0.98	0.98	0.98	216045

Complessivamente, i modelli di classificazione basati su algoritmi di machine learning hanno mostrato buone prestazioni nella classificazione delle applicazioni in base al rating. I modelli Decision Tree Classifier, K Neighbors Classifier, Random Forest Classifier e MLPClassifier hanno ottenuto particolarmente buoni risultati, con alti valori di precisione, recall e F1-score per entrambe le classi. Tuttavia, il modello Logistic Regression ha mostrato un'accuratezza inferiore e una recall relativamente bassa per la classe 1, suggerendo che potrebbe essere necessario implementare un algoritmo di cross validation per l'hyper-parameter tuning. Il modello SVC ha ottenuto prestazioni molto basse, evidenziando la necessità di ulteriori ottimizzazioni o esplorazioni di parametri per migliorare le sue capacità di classificazione (che tuttavia non saranno effettuate per motivi di potenza di calcolo e tempistiche).

Infine in base alla curva ROC e alle tabelle di confusione è stato selezionato il miglior modello:

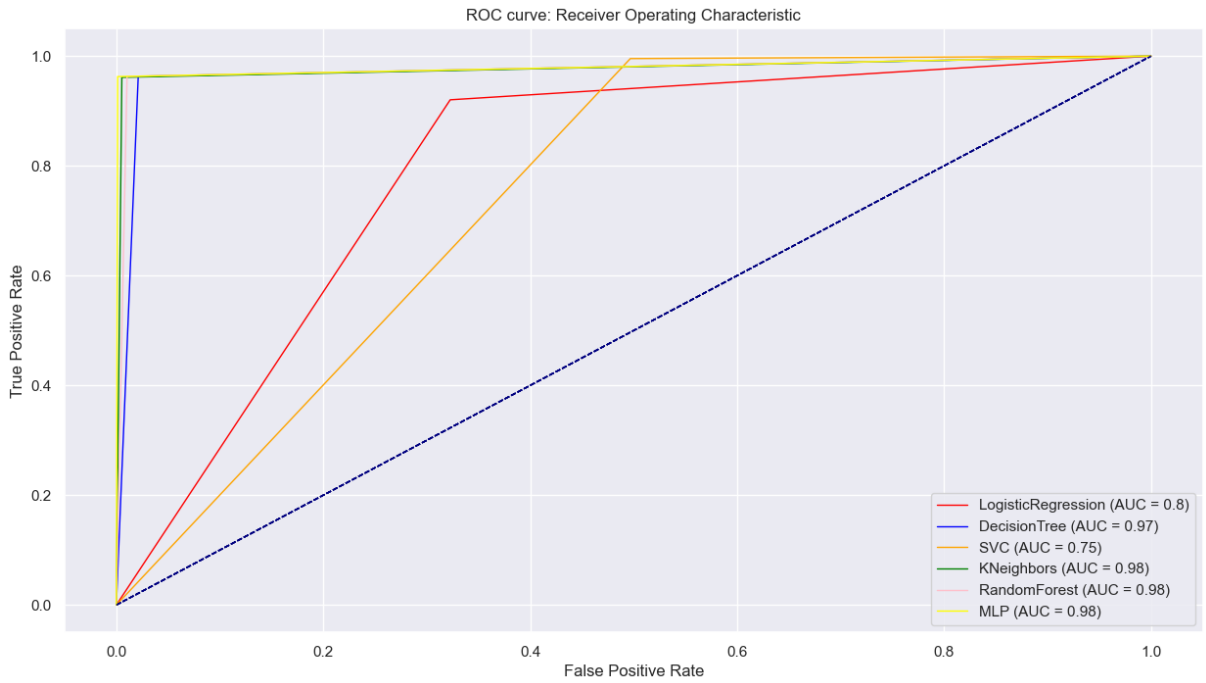

Figure 6: Comparazione della curva ROC per i vari modelli (dati riguardanti il set di testing).

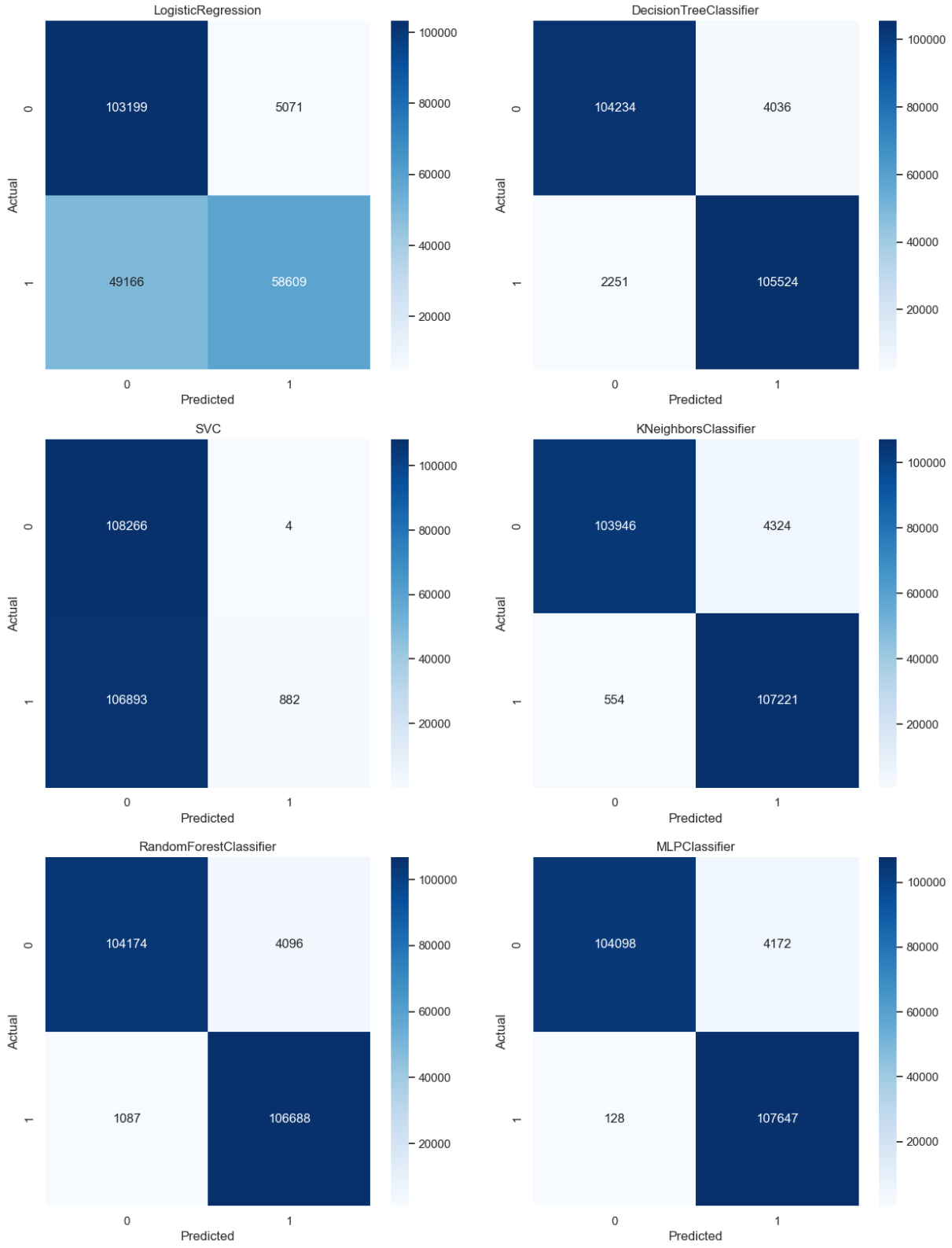

Figure 7: Matrice di confusione dei vari modelli (dati riguardanti il set di testing).

In basi a questi dati, in particolare considerando la matrice di confusione il MLP, sì può dedurre che sia il modello migliore per i nostri dati, avendo metriche simili al Random Forest Classifier, tempi di esecuzione più bassi data la possibilità di utilizzare l'early stopping, e avendo anche il numero minore di falsi negativi di tutti gli altri modelli. Per questo si è scelto di fare un ultimo test modificando il dataset in tre dati categorici (invece che due), aumentando la difficoltà del task per vedere se il modello riesce a classificare bene anche in caso di una classificazione multi-label (ternaria).

Quindi il dataset è stato modificato come segue:

- Rating da 0,1 sono stati categorizzate come applicazioni "junk"
- Rating da 2,3 sono state categorizzate come applicazioni "migliorabili"
- Rating da 4,5 sono state categorizzate come applicazioni "popolari"

I dopo il bilanciamento è risultato e il training, il MLP ha dato un accuratezza del 65% sul dataset di testing, producendo la seguente matrice di confusione:

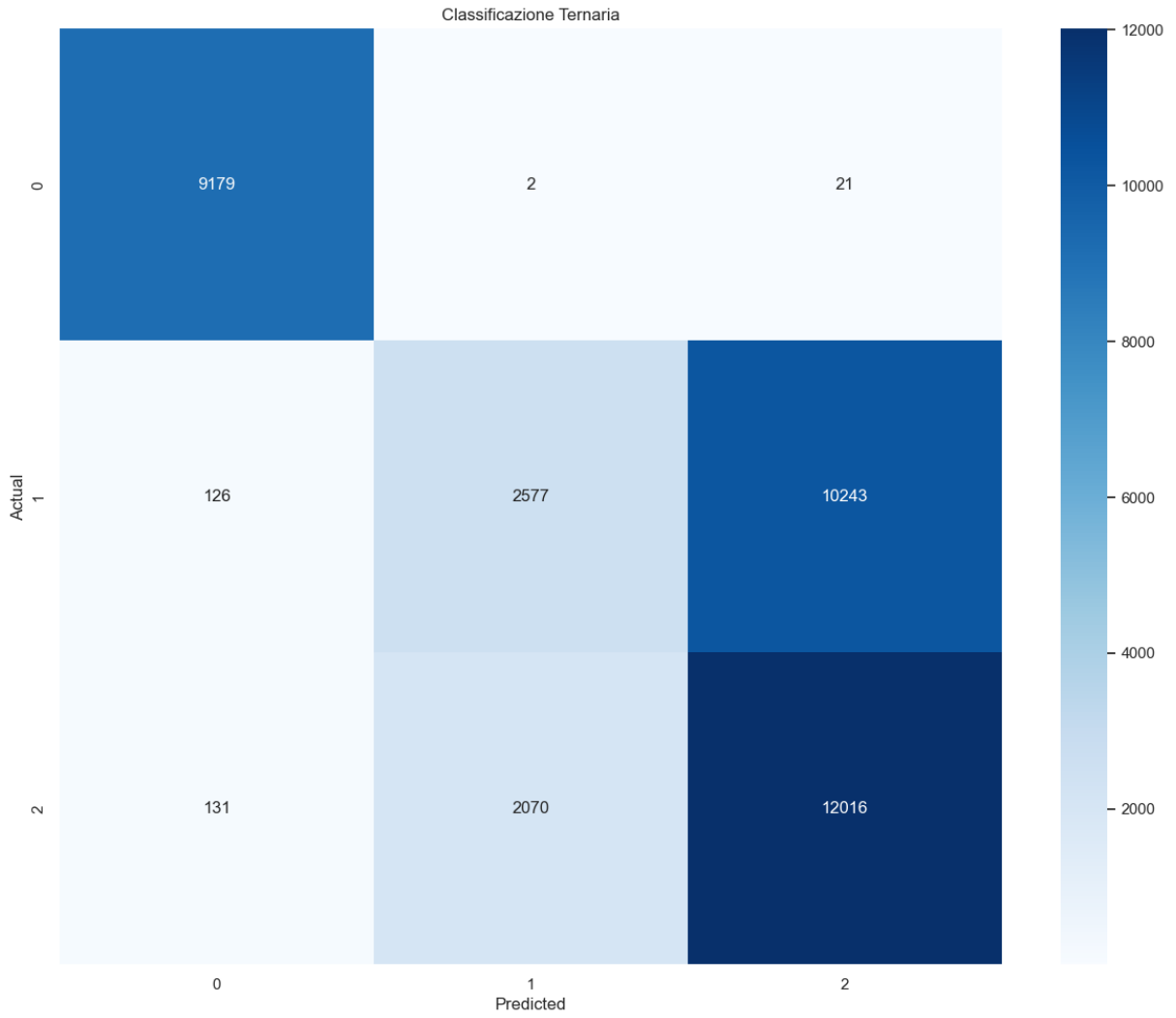

Figure 8: Matrice di confusione MLP in caso di classificazione ternaria (dati riguardanti il set di testing).

I risultati sono accettabili, sebbene è possibile notare come ci sia una certa sovrapposizione tra la classe 1, cioè le applicazioni "migliorabili" con un rating intermedio e la classe 2 cioè quelle con alto rating, sicuramente i risultati potrebbero essere migliorati con ulteriori test.

6 Modelli Autoregressivi e Regressivi

6.1 Analisi del Trend del numero di Apps rilasciate "day by day"

Per analizzare il trend delle applicazioni rilasciate nel tempo, sono state utilizzate tecniche di regressione. Per la regressione, è stato preso in considerazione il numero di applicazioni rilasciate in diversi periodi di tempo. L'obiettivo era comprendere se esistesse un andamento significativo nel numero di nuove applicazioni pubblicate sul Google PlayStore e valutare l'affidabilità di tale trend attraverso l'analisi del p-value e del coefficiente di determinazione (R-squared).

Sono stati testati due tipi di regressione utilizzando le librerie scipy e statsmodel in Python. La regressione

polinomiale di secondo grado ha prodotto risultati interessanti. L'indice di correlazione (Pearson's correlation coefficient) è stato calcolato a 0.9906, con un p-value molto basso di 1.8407e-112. Inoltre, il coefficiente R-squared ha raggiunto il valore di 0.9813. Questi risultati indicano che il modello di regressione polinomiale di secondo grado ha una buona capacità di adattarsi ai dati e di spiegare il trend delle nuove applicazioni rilasciate nel tempo.

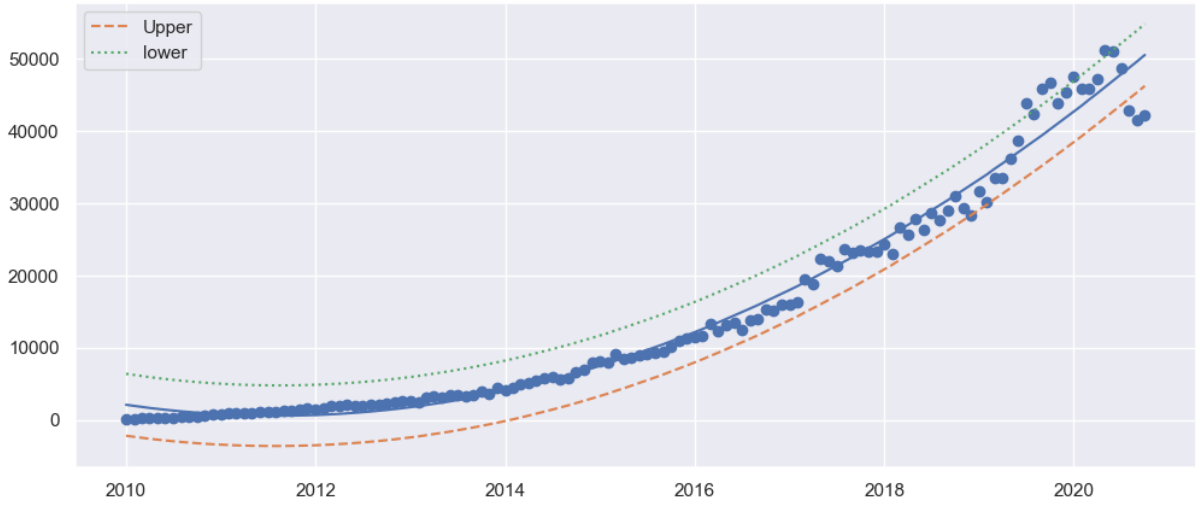

Figure 9: Modello polinomiale di secondo ordine.

È stata inoltre esaminata la regressione esponenziale, che ha mostrato anch'essa buoni risultati. L'indice di correlazione è stato calcolato a 0.9891, con un p-value molto basso di 1.8395e-108. Il coefficiente R-squared ha raggiunto il valore di 0.9784. Anche in questo caso, questi risultati indicano che il modello di regressione esponenziale riesce ad adattarsi bene ai dati e a spiegare il trend delle nuove applicazioni rilasciate.

Figure 10: Modello esponenziale.

Tuttavia, è importante notare che il modello polinomiale sembra adattarsi leggermente meglio ai dati. Questa differenza potrebbe essere attribuita alla presenza di outlier che si discostano dagli altri dati a partire dal 2020, causati dall'impatto della pandemia. È evidente che la situazione pandemica ha avuto un impatto significativo sul numero di nuove applicazioni pubblicate sul Google PlayStore, determinando un calo nel numero di rilasci.

6.2 Analisi Autoregressiva della dimensione delle applicazioni "day by day"

Per l'analisi autoregressiva, il parametro di riferimento considerato è stato la dimensione delle applicazioni, lo scopo è stato quindi di studiare se esiste un trend nella riduzione o incremento dello spazio necessario in memoria per le applicazioni. Per motivi di costi computazionali, avendo dati di oltre 10 anni, è stato utilizzato un campione contenente i dati da inizio 2014 a fine 2016 per il training, e un aggiuntivo 20% per il testing corrispondente a circa i successivi 8 mesi (quindi fino a metà 2017). Per determinare i parametri dei modelli, ARIMA e SARIMAX, sono state calcolate le funzioni di autocorrelazione e autocorrelazione parziale per identificare il numero di ritardi (lags) e determinare se è necessaria una differenziazione per rendere la serie stazionaria. Successivamente, è stato eseguito un test di stazionarietà (ADF) sulla serie temporale differenziata per valutarne la stazionarietà. In base ai risultati, il parametro "p" per l'autoregressione (modello arima o sarimax) può essere fissato a 1, come suggerito dall'autocorrelazione parziale dopo la differenziazione e dai valori del P-value e dell'ADF-Statistics post differeniazione.

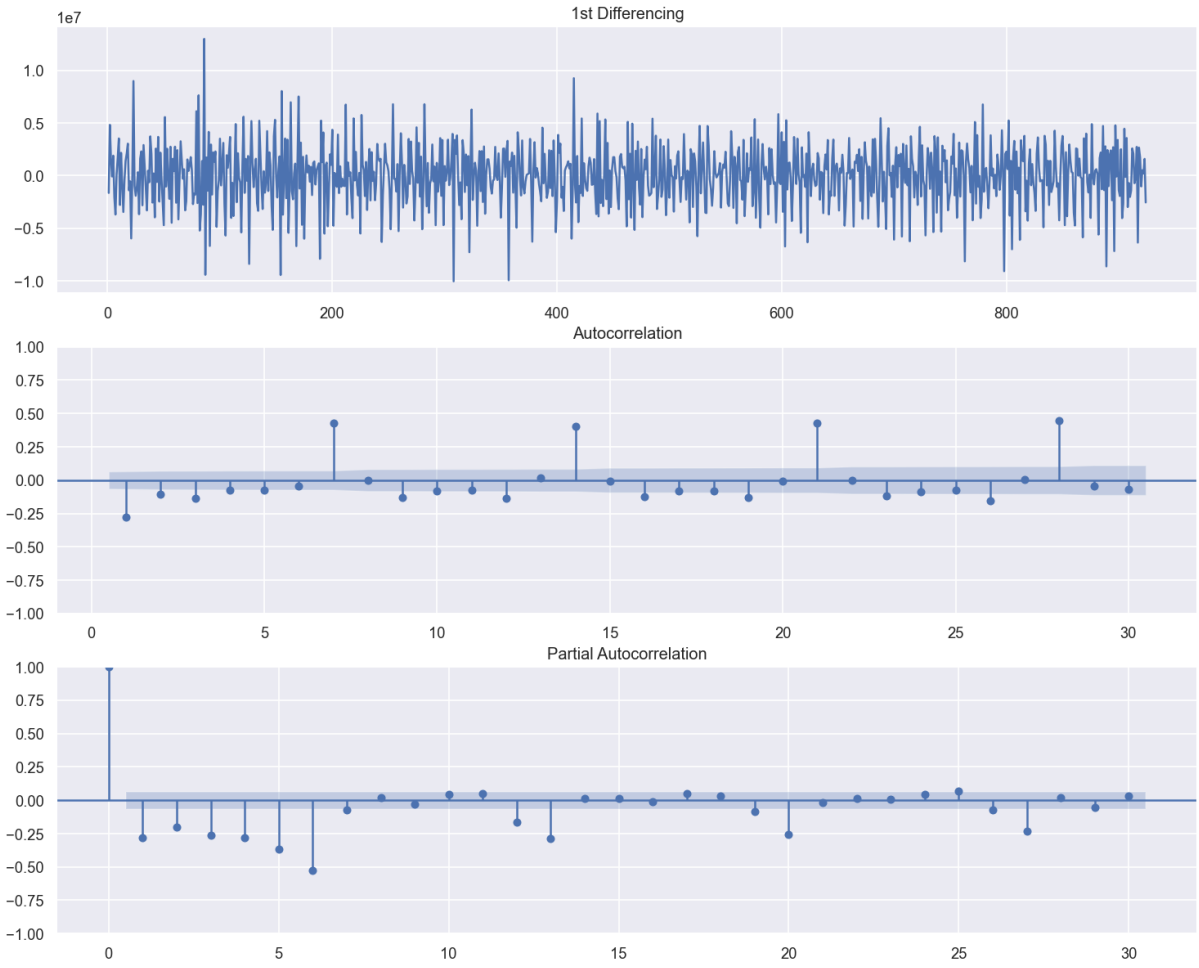

Figure 11: Autocorrelazione e Autocorrelazione parziale dopo la prima differenziazione.

Pre-differenziazione

ADF-Statistics	P-value
-2.209274470929484	0.20289659573034496

Post-differenziazione

ADF-Statistics	P-value
-12.107150571395064	1.9570164650228551e-22

Successivamente, è stato applicato il modello di decomposizione stagionale utilizzando la libreria statsmodel. Questo processo ha consentito di visualizzare graficamente la componente trend, stagionale e residua dei dati per valutarne la struttura.

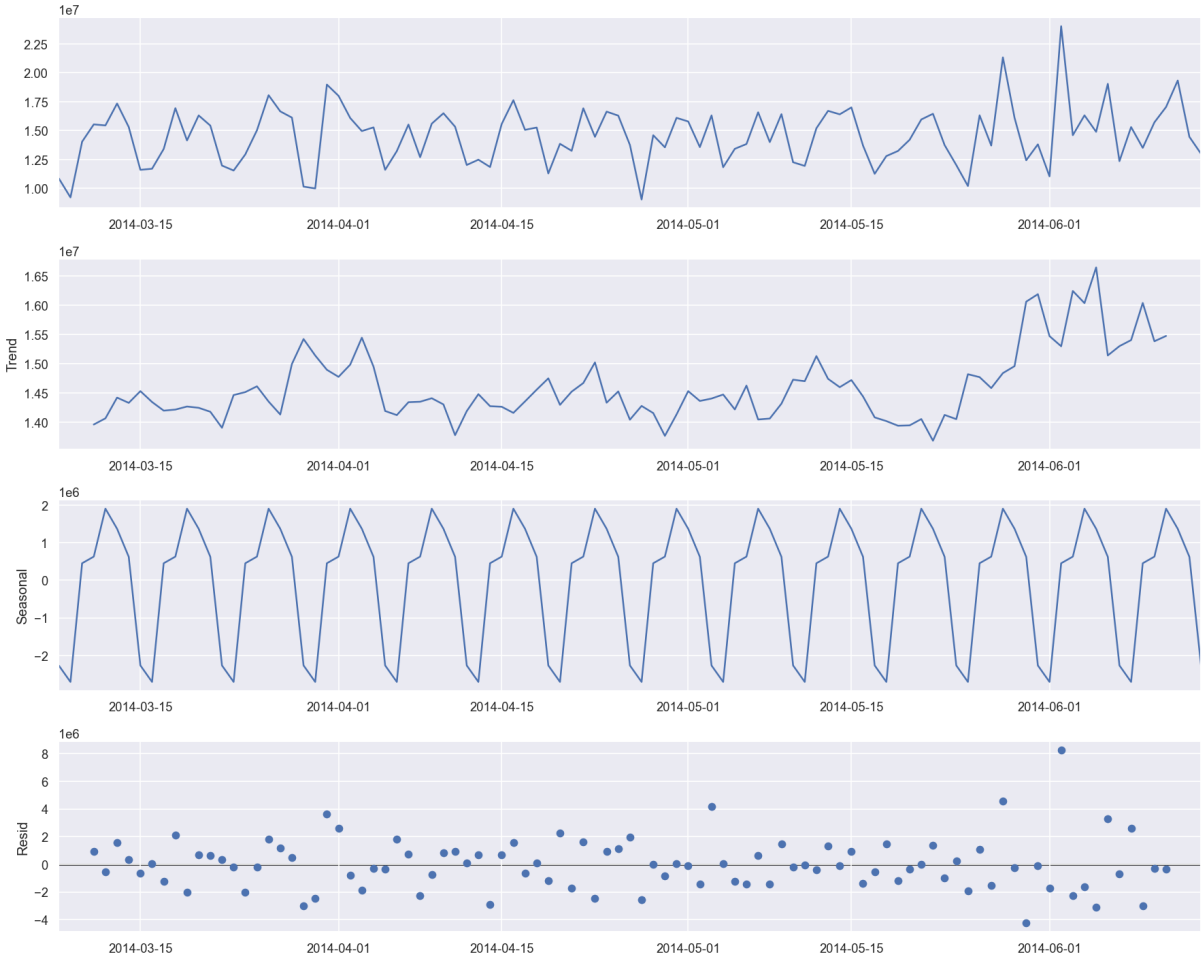

Figure 12: Decomposizione (segnale, trend, stagionalità e residuo).

Come sì può vedere la stagionalità (come potevamo aspettarci dal momento che non c'è una vera stagionalità sul rilascio delle applicazioni) è di pochi giorni ed è molto simile a rumore.

I modelli di machine learning utilizzati sono stati: ARIMA, SARIMAX, Prophet. Sono stati utilizzati diversi parametri per i modelli, utilizzando la stagionalità giornaliera e settimanale, in quanto più vicina a quella riportata dalla decomposizione. Alcuni dei risultati grafici:

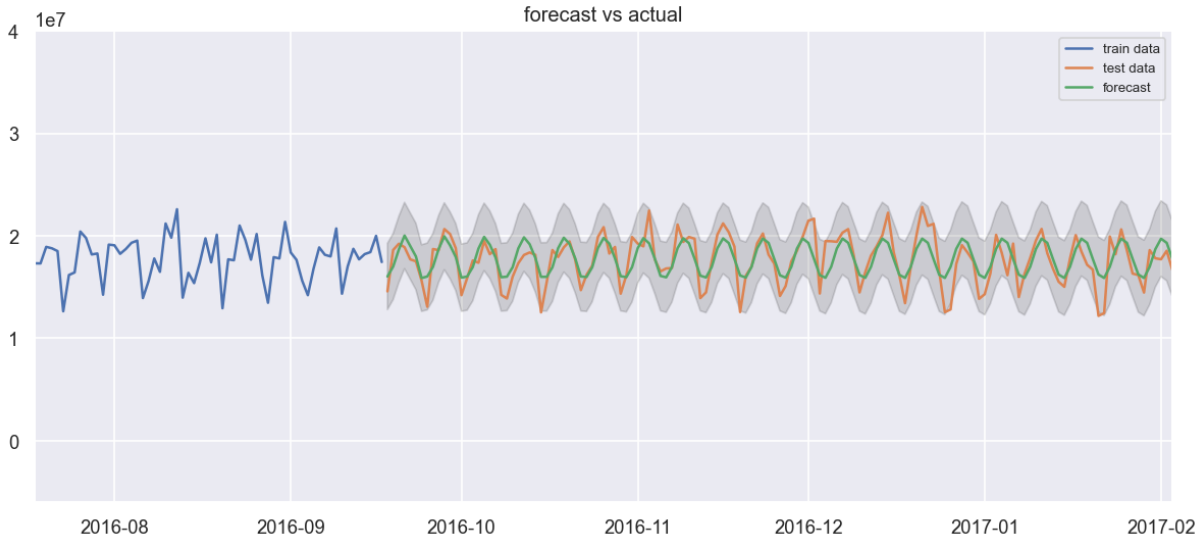

Figure 13: Arima (4,1,5)

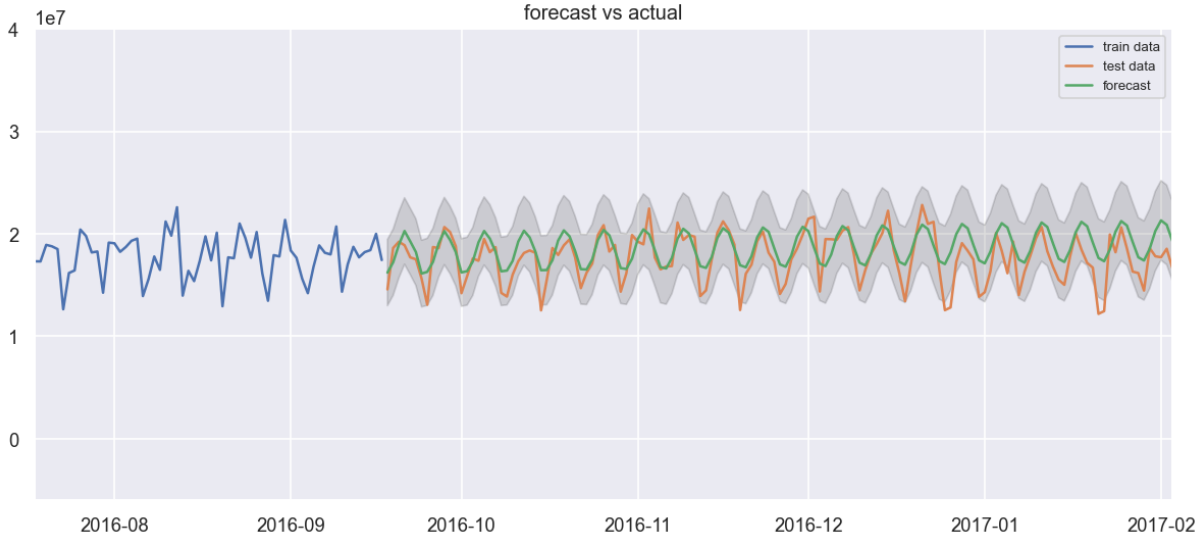

Figure 14: Sarimax (4,1,5)

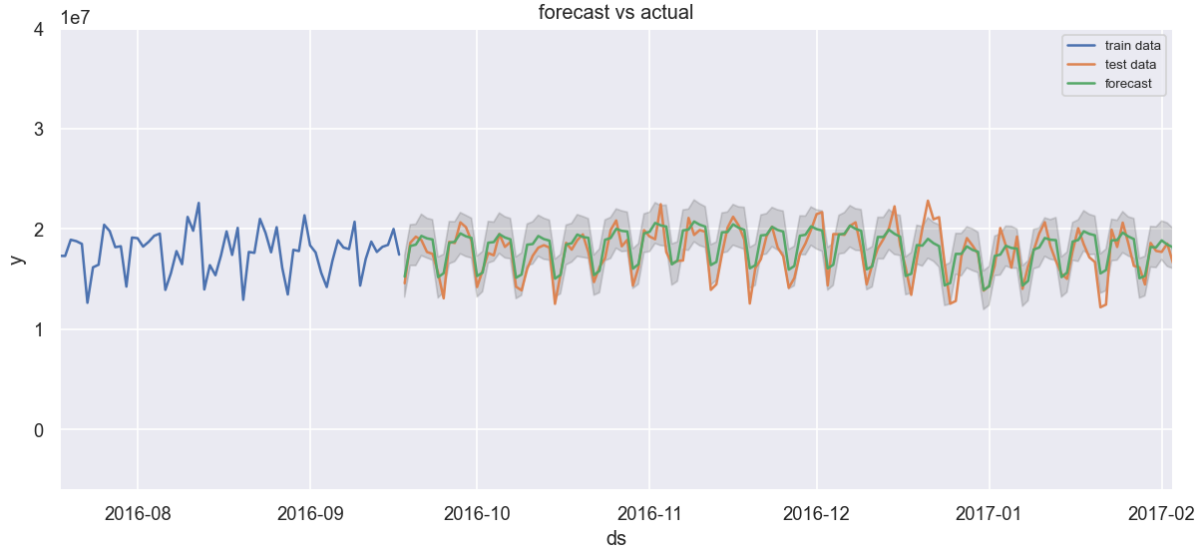

Figure 15: Prophet.

I dati quantitativi delle metriche sono stati i seguenti:

ARIMA:

MAPE	ME	MAE	MPE	RMSE	Corr	Min-Max	ACFE
0.0687	-6615.8672	1182106.0696	0.0086	1504575.9838	0.7404	0.0642	0.0949

SARIMAX:

MAPE	ME	MAE	MPE	RMSE	Corr	Min-Max	ACFE
0.1034	1345863.0794	1714228.1136	0.0854	2075923.7306	0.6993	0.0897	0.1811

Prophet:

MAPE	ME	MAE	MPE	RMSE	Corr	Min-Max	ACFE
0.067	406832.6331	1143362.1807	0.0294	1466783.0106	0.7715	0.0618	0.1736

Oltre ai modelli di machine learning, è stato utilizzato anche un modello misto, che utilizza sia componenti regressive che reti neurali, ovvero NeuralProphet che deriva dal modello Prophet, che ha avuto risultati quantitativi simili a prophet (notare che per NeuralProphet è stato utilizzato un 10% del dataset di training come validazione), anche se graficamente sono del tutto comparabili al modello Prophet.

NeuralProphet :

MAPE	ME	MAE	MPE	RMSE
0.06489676755074492,	-579174.8665197494	1170543.5906451396	-0.02659008713962434	1526972.8505607762

Come si può notare, Prophet e NeuralProphet hanno prestazioni migliori rispetto ai modelli classici, infine come confronto è stata visualizzata la correlazione tra i risultati dei modelli e i dati attuali:

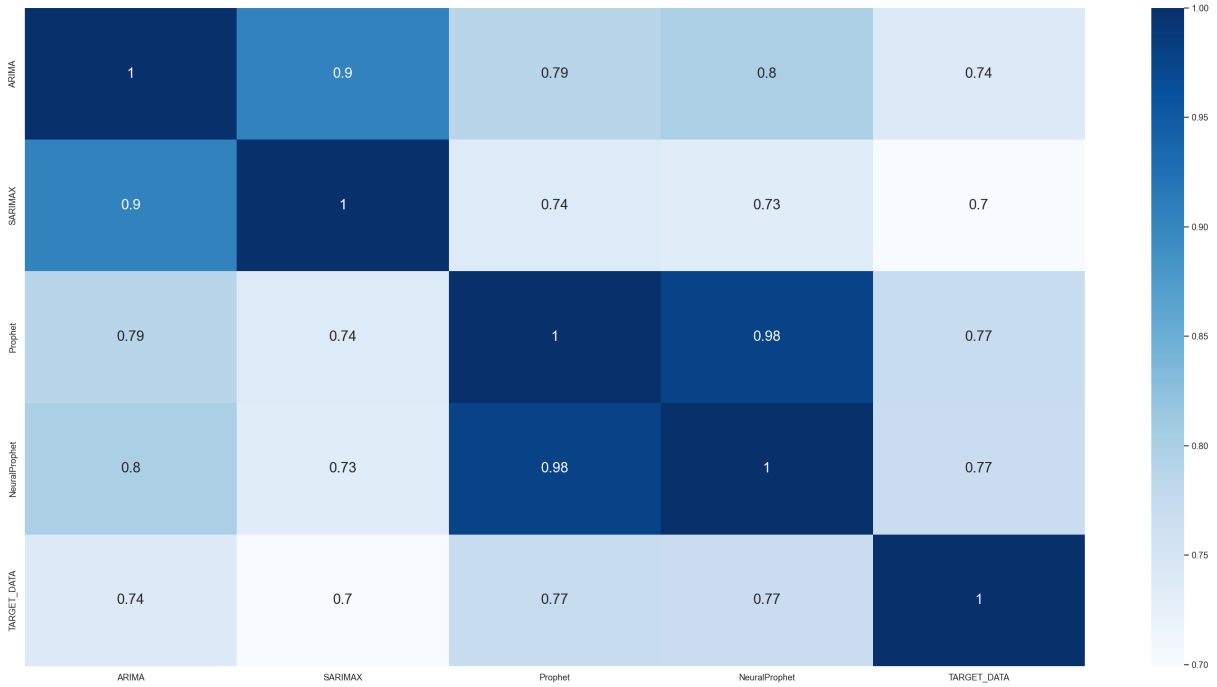

Figure 16: Correlazione tra le predizioni dei modelli regressivi allenati.

Come è possibile vedere, il forecast di NeuralProphet e NeuralProphet è meglio correlato ai dati di testing rispetto ad ARIMA e SARIMAX. Ma in ogni caso, andrebbe effettuata una Cross Validation per determinare i parametri ottimi per ogni modello attraverso algoritmi di "Grid Search" in modo da poter avere un confronto più accurato tra di essi.

7 Clustering Analysis

L'obiettivo principale dell'analisi è identificare possibili raggruppamenti o pattern all'interno dei dati, al fine di ottenere una migliore comprensione delle caratteristiche delle diverse app presenti nel dataset. Come primo passaggio, si è visualizzata la sovrapposizione tra diverse features, in modo che si potesse scegliere quali possano essere utilizzate per il clustering:

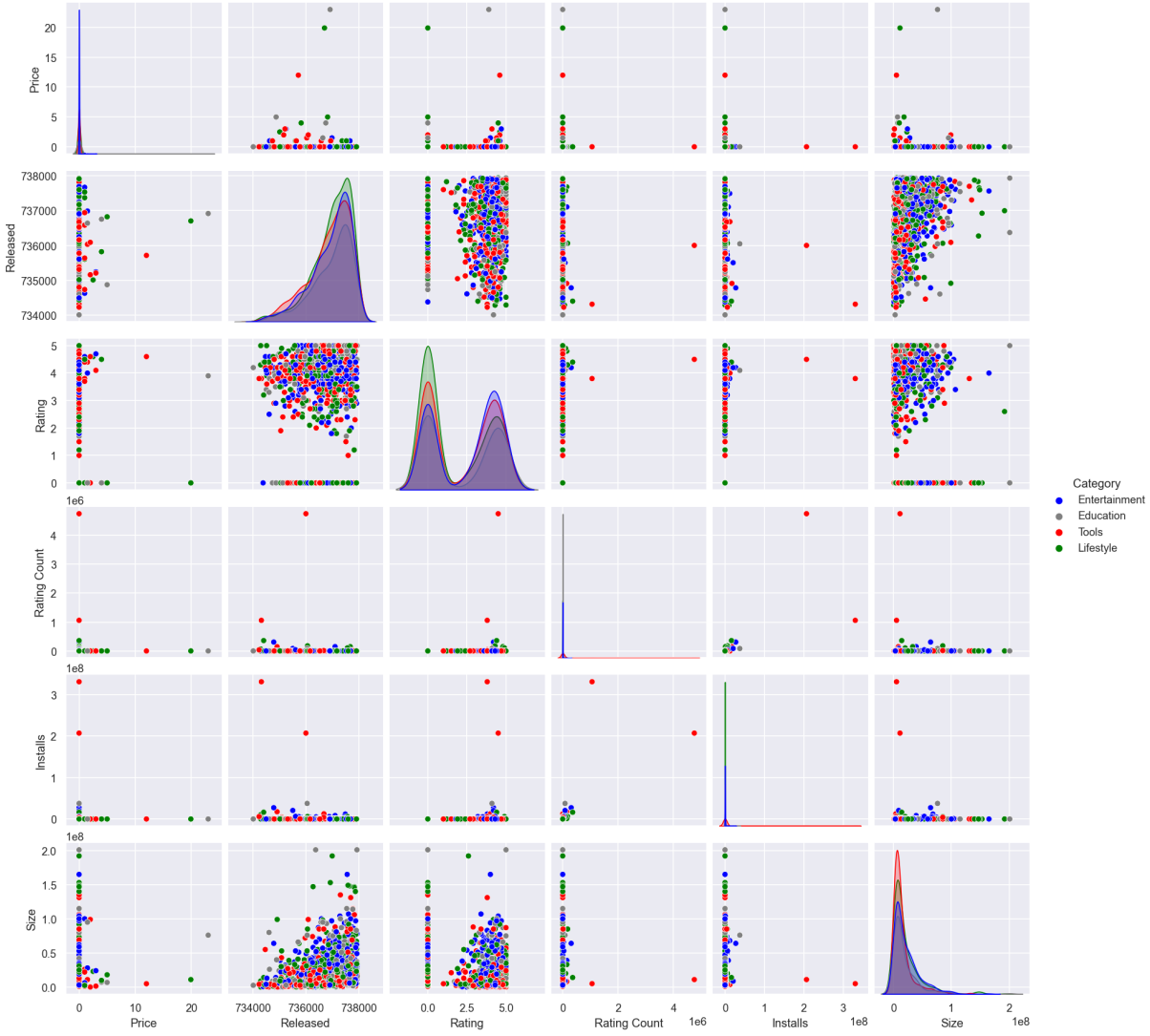

Figure 17: Distribuzione tra features

Come si può vedere, le distribuzioni più interessanti sono quelle tra la dimensioni delle applicazioni e il rating, quindi useremo queste due features come componenti principali dell'analisi per il clustering.

Successivamente sono state effettuate alcune operazioni preliminari, è stato eseguito un filtraggio degli outliers per ogni colonna del dataframe, in modo da eliminare i valori che si discostino più di 3 volte il valore della deviazione standard dalla media ed è stata eseguita la normalizzazione tramite lo Standar Scaler. Il clustering è stato eseguito utilizzando l'algoritmo K-Means su un sottoinsieme dei dati. Ed è stato utilizzato l'Elbow method per determinare il numero ottimale di cluster, che rappresenta il numero di raggruppamenti significativi all'interno del dataset. Sono dunque stati identificati 4 Cluster significativi ovvero:

- Alto rating, Alta efficienza in termini di spazio.
- Alto rating, Bassa efficienza.
- Basso rating, Alta efficienza.

- Basso rating, bassa efficienza.

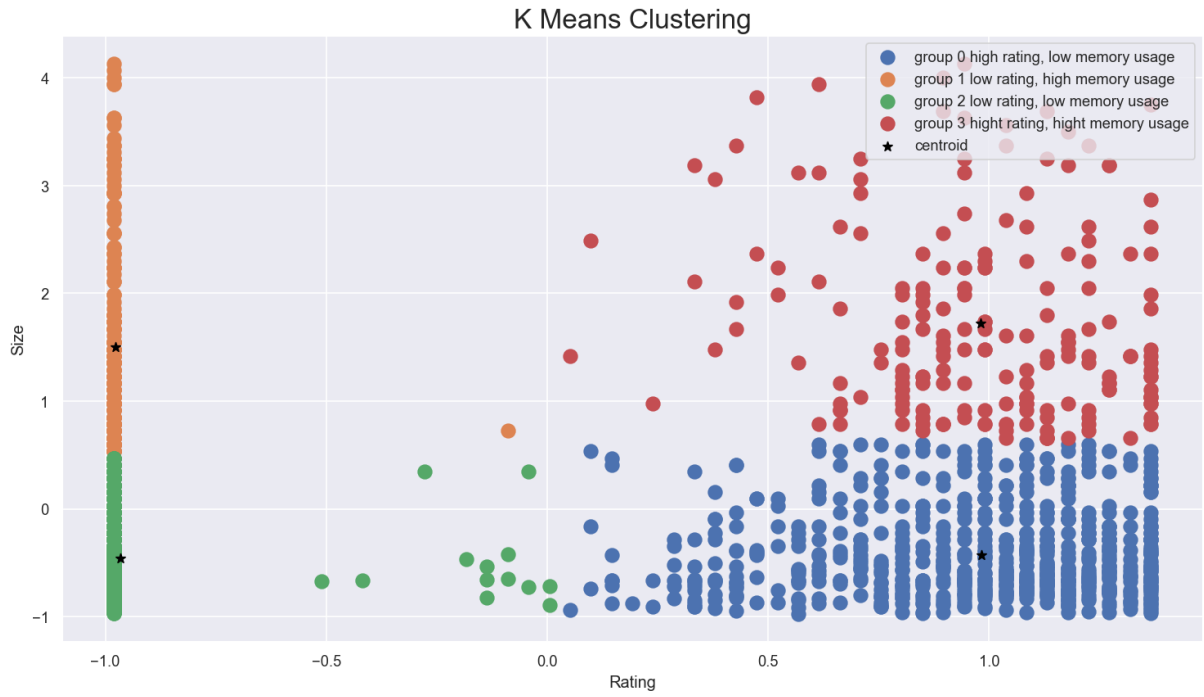

Figure 18: Clustering utilizzando l'algoritmo K-Means.

Inoltre, è stato eseguito un clustering gerarchico utilizzando l'algoritmo AgglomerativeClustering. Questo approccio costruisce gerarchie di cluster in modo incrementale, unendo iterativamente i cluster più simili fino a ottenere un'unica gerarchia. Anche in questo caso i cluster trovati sono simili ai precedenti come sì può vedere dall'immagine:

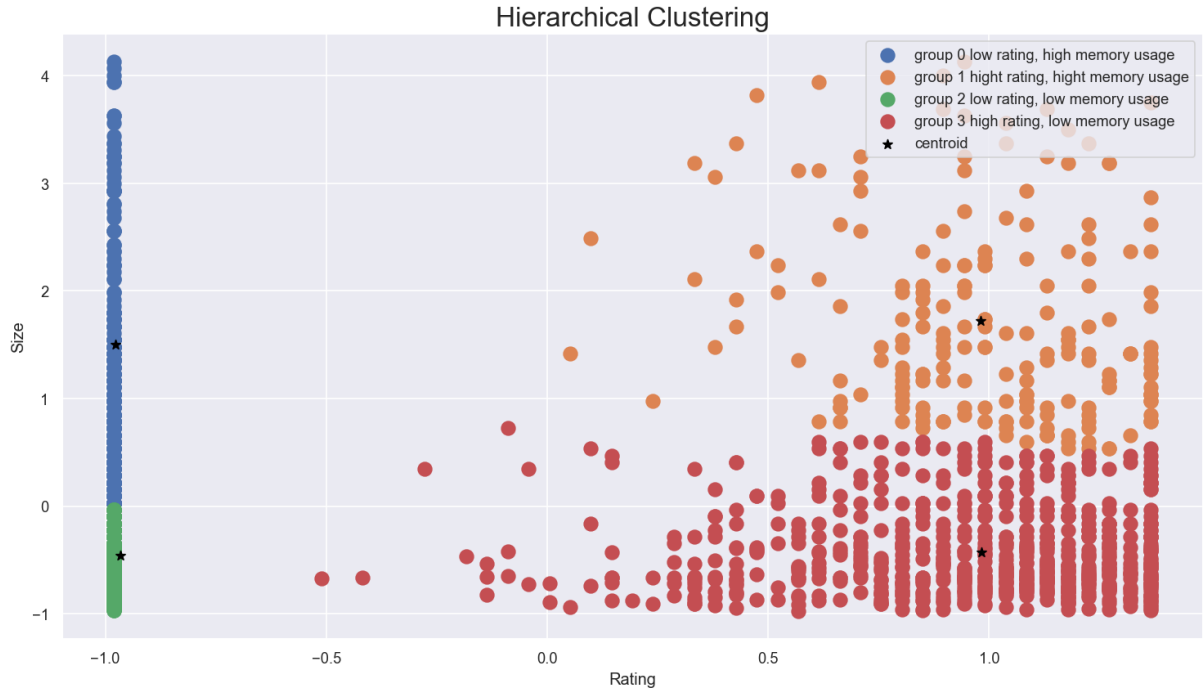

Figure 19: Clustering effettuato utilizzando l'algoritmo AgglomerativeClustering.

In seguito si è eseguita un'analisi delle componenti principali (PCA) per ridurre la dimensionalità dei dati mantenendo il contenuto informativo lungo gli assi. Viene mostrato un grafico della explained variance e della cumulative variance per valutare l'importanza delle componenti, oltre ai carichi delle componenti principali per comprendere quali variabili contribuiscono maggiormente a ciascuna componente. In questo caso invece di cercare relazioni tra Rating e Size, si è cercato di trovare relazioni tra Rating e Ad Supported, ovvero tra la popolarità dell'applicazioni e l'uso delle pubblicità all'interno delle stesse.

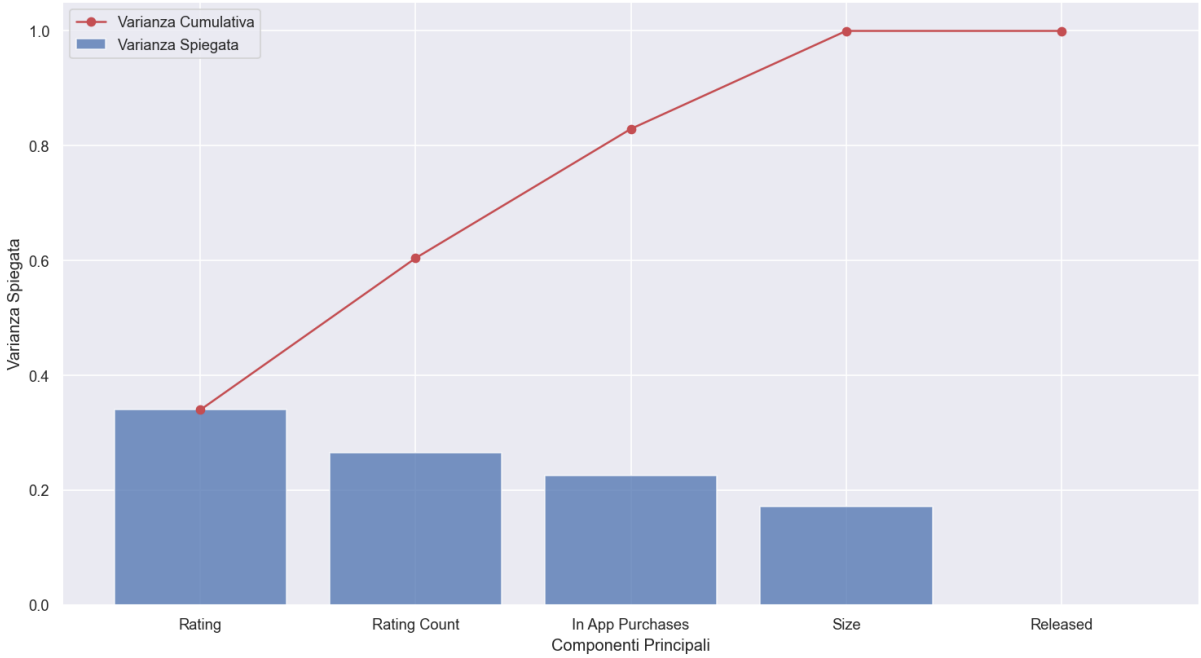

Figure 20: Explained variance delle componenti usate nella PCA

Come sì può vedere purtroppo, servono almeno 3 componenti per spiegare l'80% dei dati, La varianza cumulativa indica la quantità di varianza totale che viene spiegata da ciascuna componente principale. Nel caso in cui siano necessarie almeno tre variabili per spiegare l'80% dei dati, significa che le prime due componenti principali da sole non sono in grado di spiegare una percentuale sufficientemente alta di varianza nel dataset. Ciò può indicare che il dataset è complesso e richiede più componenti per catturare la variazione dei dati in modo significativo.
Quando si utilizzano solo due componenti principali, si perde parte della varianza complessiva del dataset e si rischia di ottenere una rappresentazione semplificata dei dati e ciò può riflettersi nelle successive analisi.

PCA Loadings

	Rating	Rating Count	In App Purchases	Size	Released
Componente 1	0.593113	0.421792	-2.220446e-16	-0.251347	-0.63807
Componente 2	0.357799	0.357129	1.554312e-15	0.828032	0.24249

Come si può vedere dai carichi invece, la componente uno della PCA è correlata positivamente con i parametri rating, rating count e molto negativamente con Released, cioè con applicazioni più vecchie, mentre la componente due della pca è correlata positivamente particolarmente con Size, Rating e Released (applicazioni più recenti).

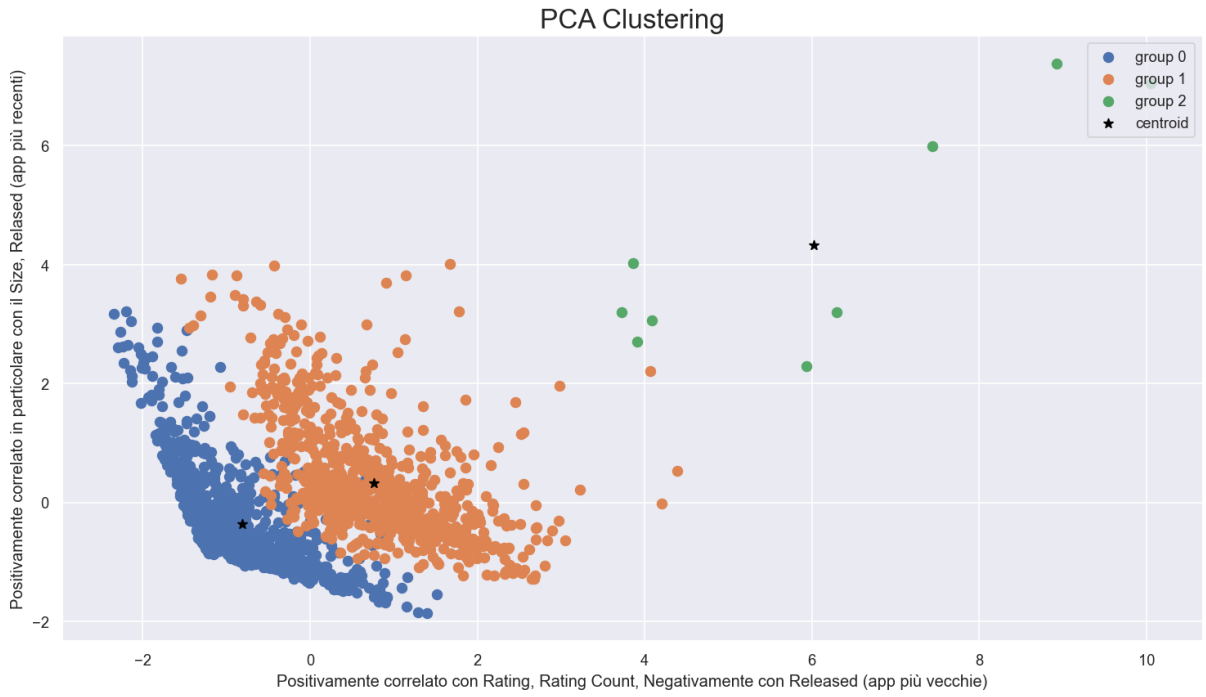

Figure 21: Algoritmo usato K-Means, numero di cluster scelto tramite la regola dell'Elbow Method.

Come è anche possibile vedere dalla shiluette c'è un po' di sovrapposizione tra il gruppo 0 e il gruppo 1, probabilmente perché mantenendo solo due componenti, può esserci una perdita di variabilità.

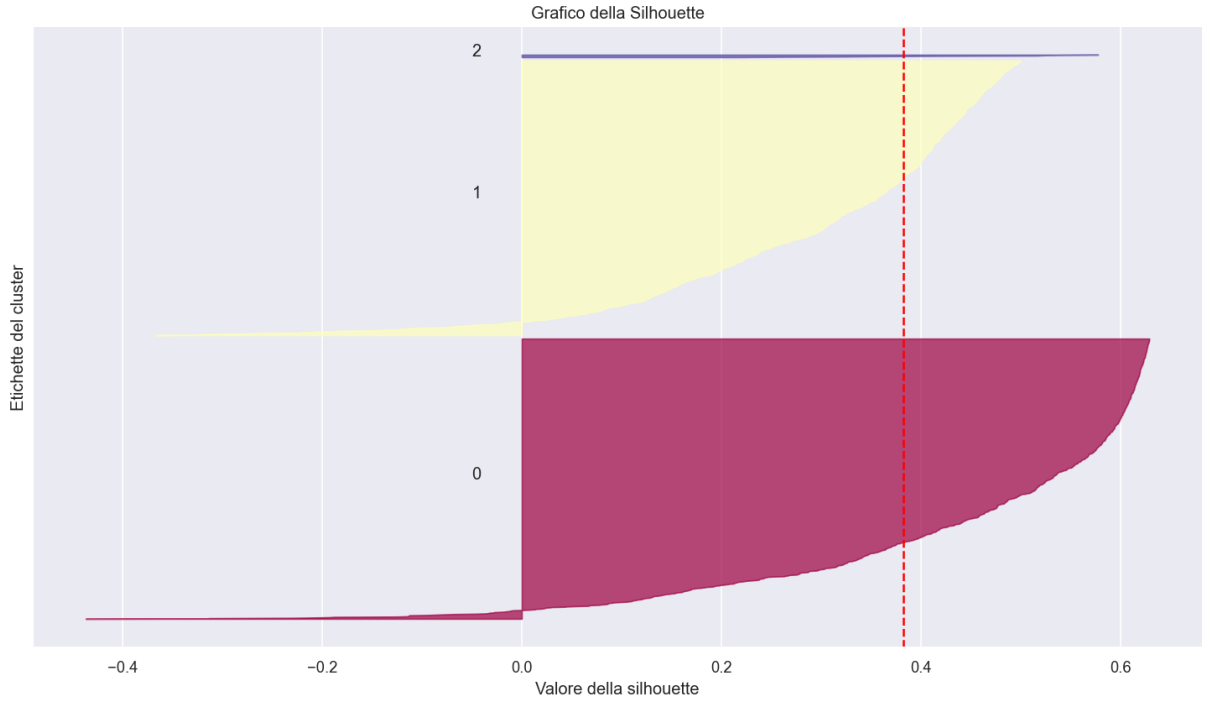

Figure 22: Shiluette del clustering

Tale problema potrebbe essere parzialmente risolto utilizzando tre componenti principali derivate dalla PCA, e quindi in tre dimensioni:

- Componente 1, correlata positivamente con rating, e molto negativamente con Released.
- Componente 2, correlata molto positivamente con Size.
- Componente 3, correlata molto positivamente con Rating Count.

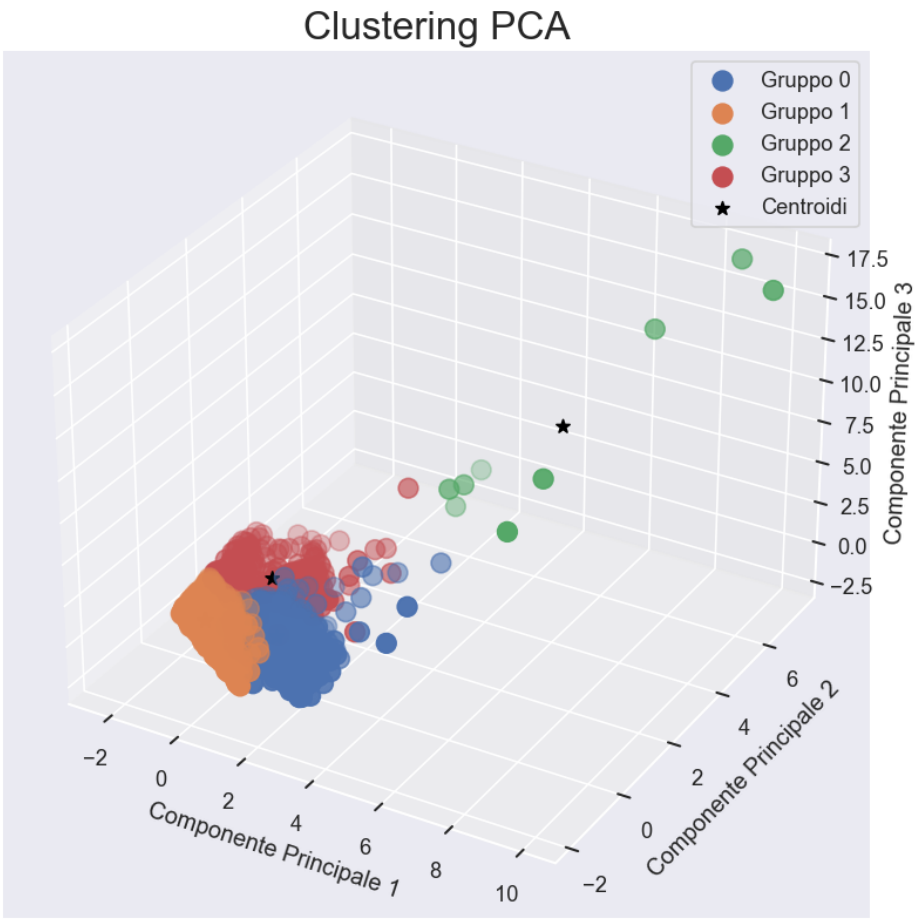

Figure 23: K-Means con PCA, mantenendo 3 componenti principali.

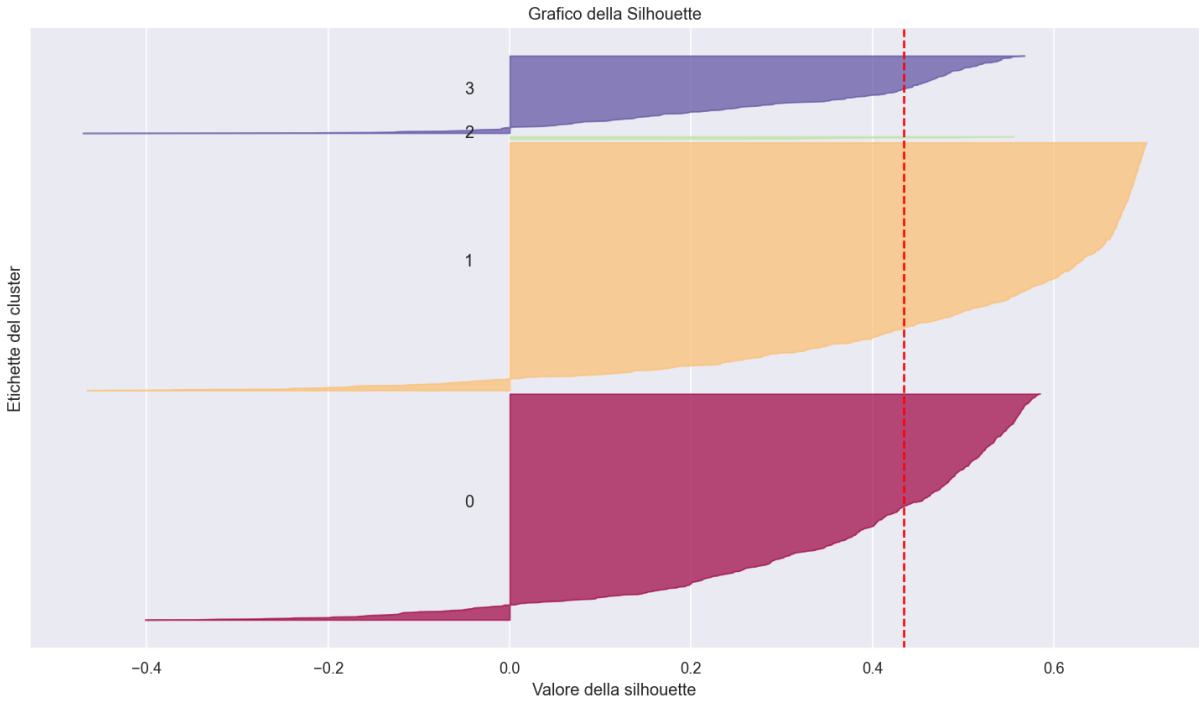

Figure 24: Shiluette utilizzando 4 clusters nel caso di Clustering PCA tridimensionale.

Sono stati anche utilizzati altri algoritmi di clustering, tra cui clustering come DBSCAN, tuttavia data la peggiore qualità dei risultati, questi ultimi sono stati omessi dall'analisi.