Università Politecnica delle Marche

Facoltà di Ingegneria

Corso di Laurea in Ingegneria Informatica e dell'Automazione

Analisi di un Dataset di un e-commerce attraverso linguaggio Python

Docenti
Prof. Ursino Domenico
Prof. Marchetti Michele

Studenti
Mori Nicola
Sospetti Mattia
Zitoli Francesca

Anno accademico 2021-2022

Indice

1 Dataset

La seguente trattazione, in ambito Data Science, si basa sull'analisi di un dataset riguardante i dati di vendita di una organizzazione di e-commerce degli Stati Uniti. Il dataset è disponibile al seguente link: `https://www.kaggle.com/roopeshbharatwajkr/ecommerce-dataset`.

I dati riportati nel dataset sono stati acquisiti dal 20 Settembre 2013 al 13 Gennaio 2014 e riguardano 12 tipologie di prodotti:

- Bag
- Books
- Cycle
- Faireness Cream
- Hair Band
- Jean
- Pen Drive
- Shirt
- Shoes
- Spectacles
- Vessels
- Wat New York Citys.

Ogni prodotto può appartenere a 8 categorie di prodotti differenti:

- Accessories
- Clothing
- Electoronics
- Fashion
- House Hold
- Stationaries

- Vehicle
- Wearables

Il file *.csv* acquisito inizialmente riportava le seguenti colonne:

ID	Descrizione
transaction_id	Id della transazione
customer_id	Id del cliente
date	Data in cui è avvenuta la transazione
product	Tipologia di prodotto ordinato
gender	Genere dell'acquirente
device_type	Indica il tipo di device da cui è stato effettuato l'ordine (Mobile, Web)
country	Paese da cui viene effettuato l'ordine
state	Stato da cui viene effettuato l'ordine
city	Città da cui viene effettuato l'ordine
category	Categoria a cui appartiene il prodotto ordinato
customer_login_type	Indica se l'acquirente è registrato al sito
delivery_type	Indica il tipo di consegna che viene effettuata (normale o in un giorno)
quantity	Quantità di prodotti ordinati
transaction_start	Indica lo stato della partenza della transazione, (sempre 1)
transaction_result	Assume valore booleano 0 o 1 a seconda del completamento della transazione (1 se completata)
amount_us	Prezzo totale pagato per l'ordine ($)
individual_price_us	Prezzo del singolo prodotto ($)
year_month	Anno e mese in cui è avvenuto l'ordine
time	Ora in cui è avvenuto l'ordine

1.1 ETL

Come prima cosa è stato impostato il campo *transaction_id* come Id del dataset.

È stato corretto un errore presente nel dataset poiché il nome della città di Los Angeles era scritto in modo errato, ovvero "Los Angles". In seguito è stata rimossa l'unica riga con valori nulli e sono state rimosse delle colonne che non apportavano informazioni significative al nostro studio. Le colonne eliminate sono state:

- *year_ month*: poiché è un'informazione ridondante dato l'attributo *date* già presente.
- *customer_ id*: poiché essendo un id dell'acquirente bastava l'id della transazione come indice.
- *country*: l'attributo era solamente valorizzato come "United States" per ogni tupla del dataset.
- *transaction_ start*: l'attributo era solamente valorizzato come "1" per ogni tupla del dataset.
- *transaction_ result*: poiché non lo abbiamo ritenuto utile ai fini dell'analisi.

È stato modificato il formato del campo 'Date' passandolo da object al formato *datetime*, così che possa essere usato nel modo corretto durante l'analisi delle serie temporali.

Nel dataset erano presenti campi in cui il valore di *individual_ price* era pari a '#VALUE!', poiché questo avrebbe comportato problemi per l'analisi dei dati tali righe sono state rimosse. Inoltre il campo *individual_ price* non era un campo numerico per cui è stata effettuata la conversione da *string* a *float*.

2 Analisi Dei Dati

È stata effettuata una prima analisi del dataset, si sono voluti da prima individuare quali sono i prodotti più ordinati dal sito e-commerce, in questo grafico si è preso in considerazione il numero di volte in cui i prodotti sono stati ordinati. Per maggior chiarezza i dati sono stati rappresentati sia con un diagramma a torta, Figura 2, sia con un grafico a barre, Figura 1.

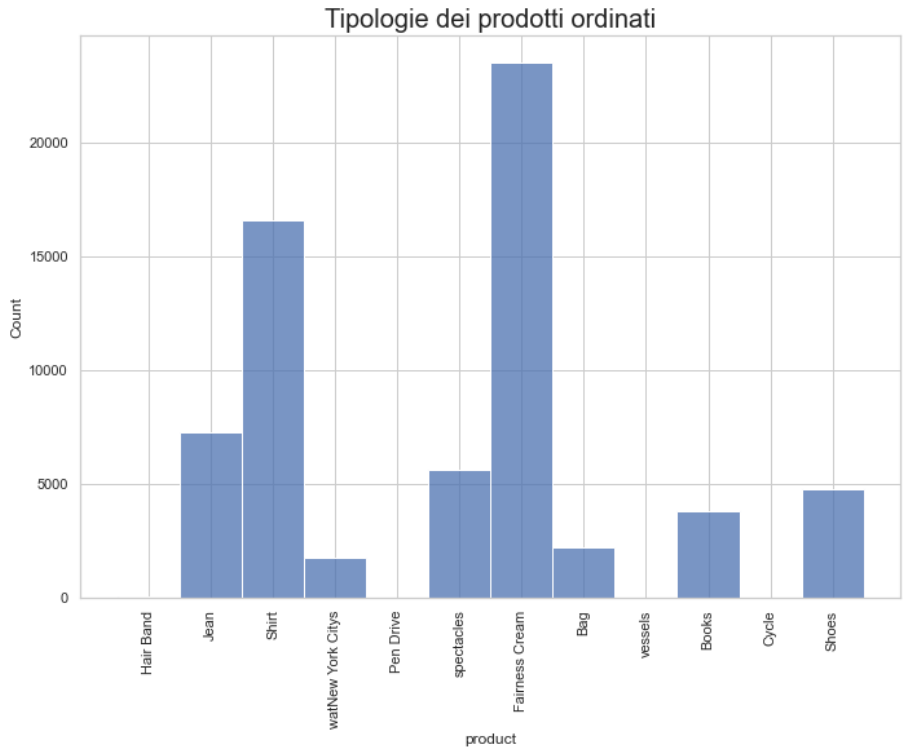

Figura 1: Istogramma mostra le quantità vendute per ogni prodotto

Come si può notare il prodotto maggiormente ordinato e preferito dagli acquirenti dell'e-commerce è Fairness Cream, ovvero le creme di bellezza, segue Shirt ovvero le camicie che da come si può notare dal grafico a torta, Fig. 2, vanno ben oltre il 50% del totale dei prodotti venduti del periodo.

In seguito è stato effettuato un raggruppamento per tipologia di prodotto e sono state sommate le quantità acquistate per ognuno, per osservare quali siano le quantità totali ordinate per ogni tipologia di prodotto, Figura 3. Come si può notare le quantità totali di Fairness Cream ordinate superano le 400000 vendite, seguite da Shirt che invece si aggira poco al di sotto 300000. Quindi segue una certa linearità tra numero di ordini e quantità ordinate,

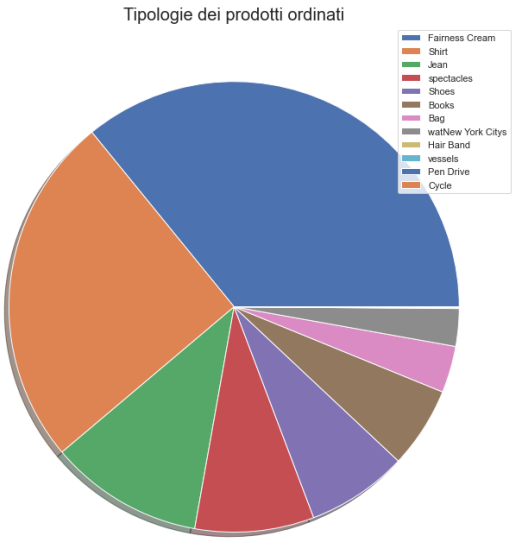

Figura 2: Grafo a torta che indica la quantità dei prodotti ordinati

il che ci suggerisce che all'incirca il le quantità ordinate sono più o meno le stesse per ogni ordine.

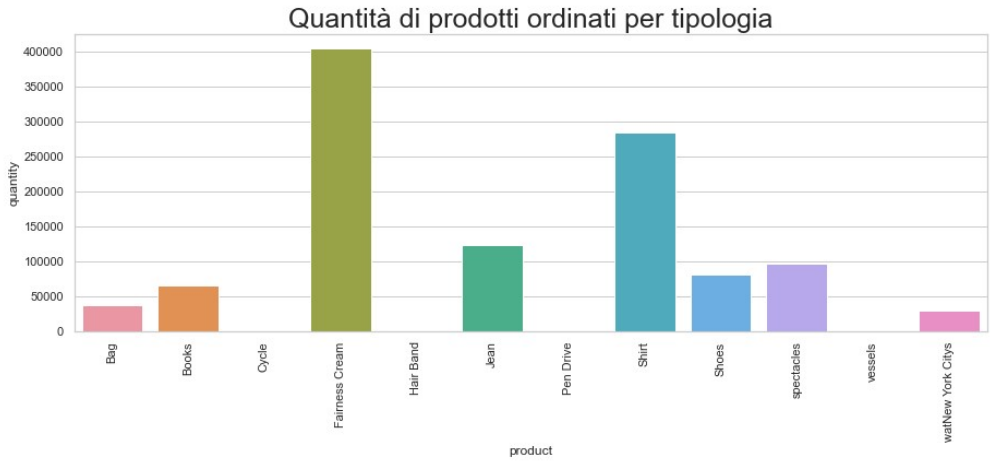

Figura 3: Quantità di prodotti venduti per ogni tipologia

Poi si è pensato di andare a studiare quale genere abbia effettuato più ordini nell'e-commerce, per comprendere se sia più orientato ad un cliente femminile o maschile. Come si può notare dalla Figura 4 nonostante si possa pensare che dato il prodotto più venduto sia Fairness Cream allora il genere femminile sia il più attivo in termini di ordini, i dati ci comunicano che il

risultato sia diverso ovvero notiamo come gli uomini abbiano effettuato il 56.91% di ordini che in termini assoluti equivale a 35000.

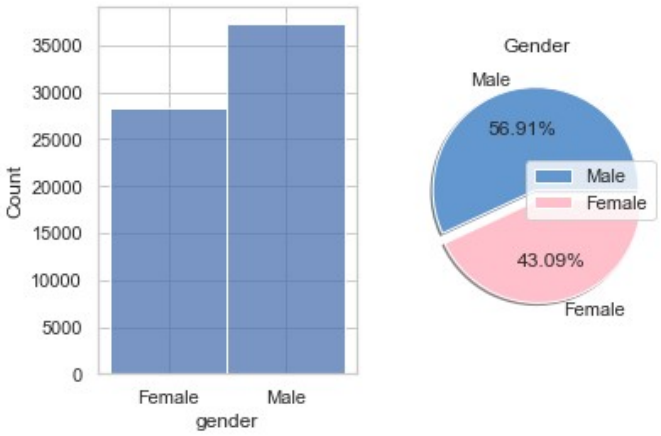

Figura 4: Quantità e percentuale di ordini per genere

Rimanendo sempre in termini di genere abbiamo pensato di confrontare se gli uomini, coloro che hanno effettuato il maggior numero di ordini, sono stati effettivamente quelli che hanno speso di più. Come mostra la Figura 5 c'è una prevalenza anche qui del genere maschile con un 56.04% del totale speso da quel genere.

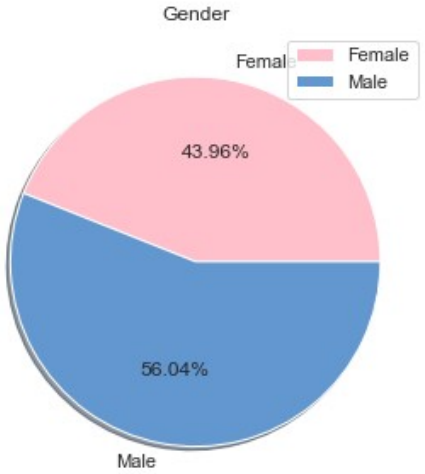

Figura 5: Grafico a torta del genere che ha speso di più

È stata effettuata un'analisi per visualizzare quali sono le categorie preferite dal genere maschile e dal genere femminile. Come si può notare dalla

Figura 6 notiamo che c'è un equilibrio per quanto riguarda la categoria Fashion. Questo significa che Fairness Cream non è prodotto esclusivo per le donne e la categoria Fashion comprende anche altre tipologie di prodotto che sono pensate anche per una clientela maschile. Nel campo Clothing più del doppio degli ordini viene effettuato dal genere maschile rispetto a quello femminile; ciò potrebbe giustificare la differenza di spesa. Possiamo notare come due categorie abbiano come acquirenti solo uomini: Electronics e Stationaries. Alla prima corrispondono oggetti come chiavette USB e sim telefoniche, per la seconda solo libri; questi prodotti non avendo dei generi siano stati associati di default al genere maschile. Discorso diverso vale per wearables che comprende calzature e qui possiamo affermare che con più sicurezza che siano dirette ad una clientela femminile.

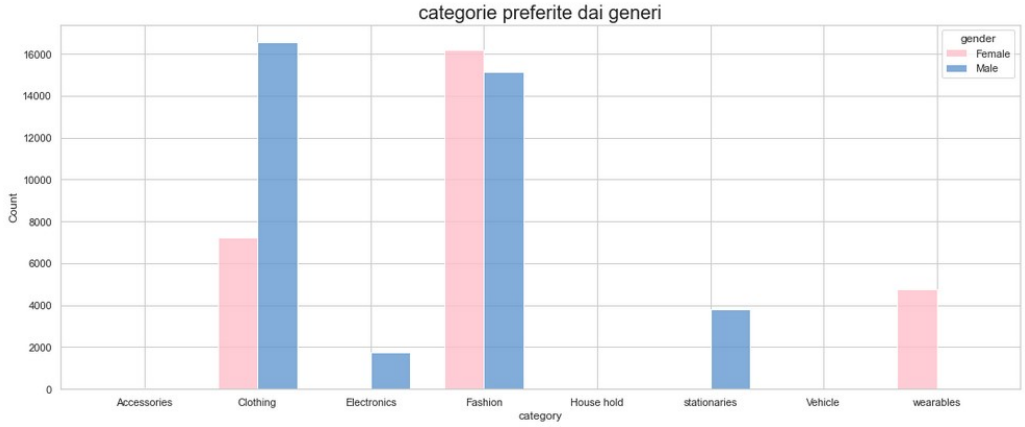

Figura 6: Grafico a barre delle categorie preferite dai due generi

Un'altra analisi effettuata è quella in cui si cerca di capire quali sono gli Stati che spendono di più, per fare questa analisi è stato creato un primo grafico in cui si contano gli ordini per ogni stato, poi è stato fatto un secondo grafico dove si mostra la spesa totale per ogni stato, Fig. 7. Si può notare come lo stato di Washington sia quello che abbia ordinato e speso di più, mentre lo stato di New York è quello che effettua meno ordini e spende meno.

Infine è stata analizzata la distribuzione del valore dei prezzi all'interno del dataset, Figura 8. Si nota subito che la maggior parte dei prodotti venduti hanno un prezzo basso, infatti notiamo un picco nella fascia di prezzo minore, mentre man mano che si sale di fascia di prezzo la quantità diminuisce ricalcando l'andamento della Power Law.

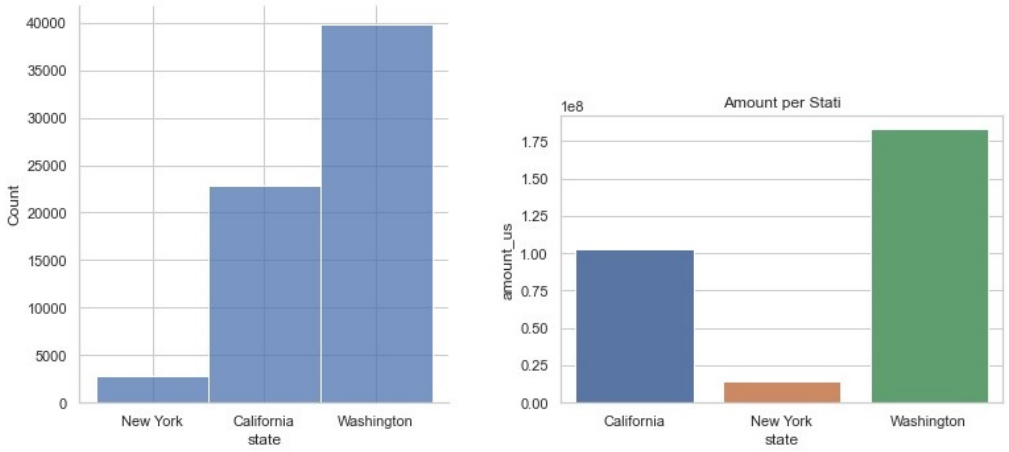

Figura 7: Grafico a barre del numero di ordini effettuati da ogni stato

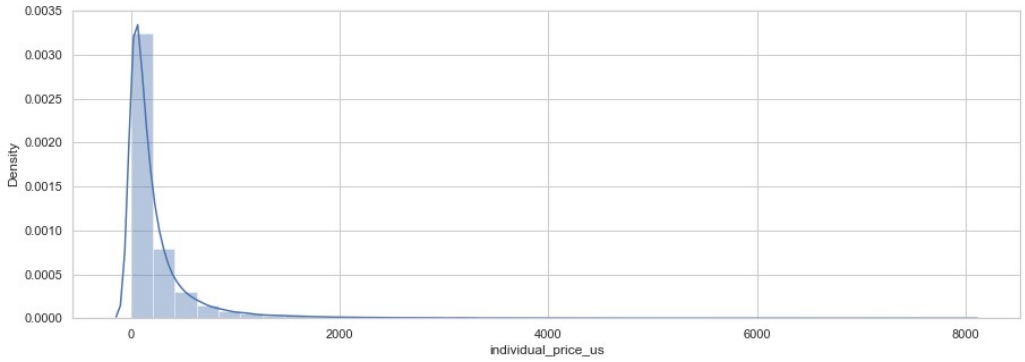

Figura 8: Distribuzione dei prezzi dei prodotti

2.1 Correlazione

Per terminare questa prima fase di analisi dei dati si è deciso di analizzare la correlazione tra gli attributi, ossia quanto le colonne del dataset siano dipendenti l'uno dell'altra. In primo luogo è stato effettuato un *pairplot*, Figura 9, seguito da una *heatmap* nella quale sono mostrati i valori delle correlazioni tra gli attributi, 10.

Notiamo una forte correlazione tra amount_us e individual_price, questo è ovvio poiché amount_us è ottenuto moltiplicando la quantità di prodotti acquistati per il prezzo del prodotto singolo (individual_price). Si può osservare, soprattutto dalla Heatmap, la scorrelazione tra le variabili quantity e amount_us, chiaramente questo ragionamento si estende ancheper la coppia quantity e individual_price_us essendo amount_us un multiplo di

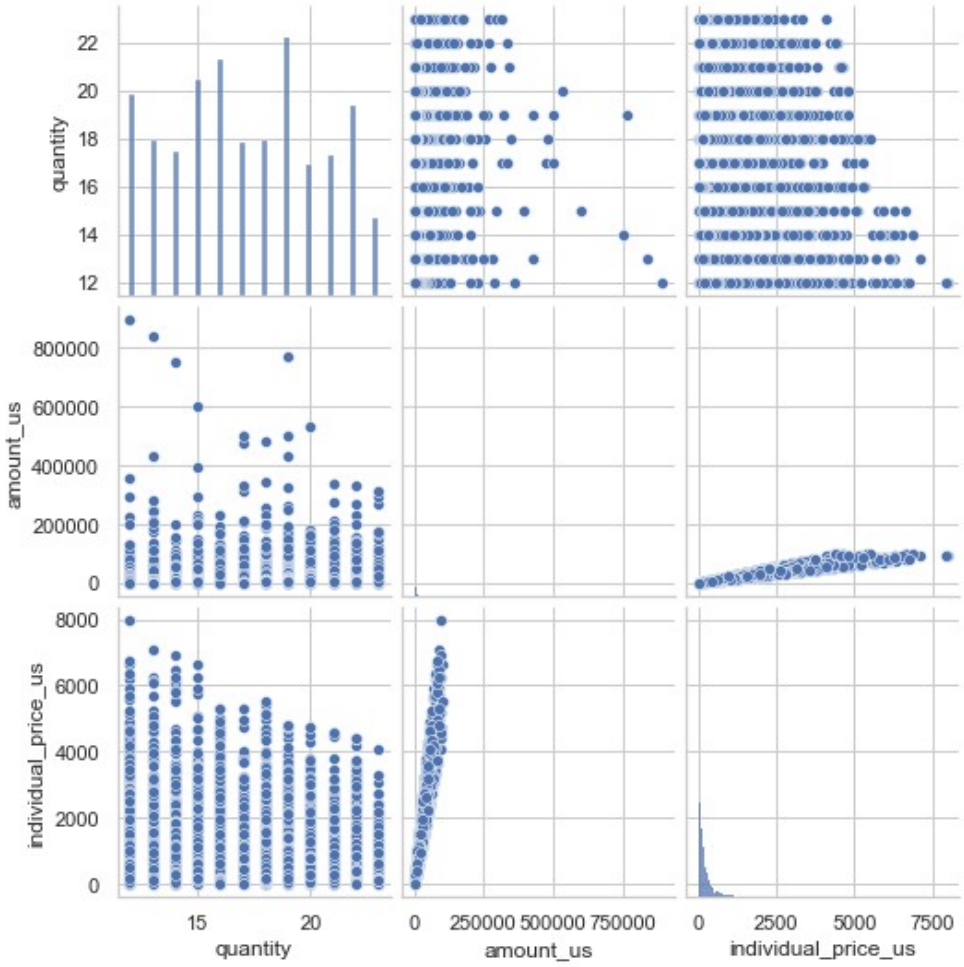

Figura 9: Pairplot con le variabili numeriche

individual_price_us.

Da questo possiamo capire che gli acquirenti molto probabilmente non guardano eccessivamente al prezzo di vendita dei prodotti per stabilire le quantità di acquisto.

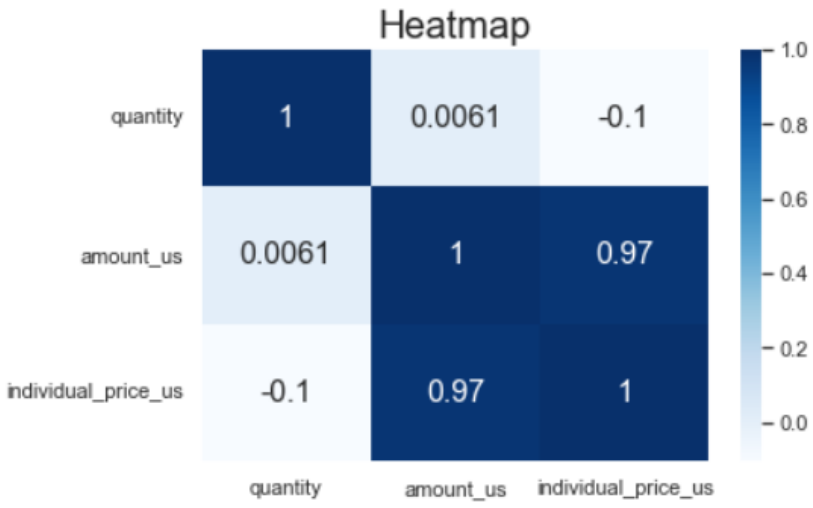

Figura 10: Heatmap della correlazione

3 Clustering

3.1 K-Means

Per questa analisi sono state rimosse alcune colonne che risultavano poco consone per lo studio da effettuare, poiché non sono attributi che hanno una valorizzazione numerica, quindi poco utili al processo di clusterizzazione attraverso K-Means. Le colonne rimosse sono le seguenti:

- date
- product
- device_type
- state
- city
- category
- costumer_login_type
- delivery_type
- time

In seguito a questa operazione abbiamo ottenuto un dataset ristretto come mostrato in figura 11

In questo caso per effettuare il clustering è stato usato l'algoritmo **K-Means**, questo è un algoritmo di apprendimento non supervisionato che trova un numero fisso di cluster in un insieme di dati. I cluster rappresentano i gruppi che dividono gli oggetti a seconda della presenza o meno di una certa somiglianza tra di loro, e vengono scelti a priori, prima dell'esecuzione dell'algoritmo. Ognuno di questi cluster raggruppa un particolare insieme di oggetti che vengono definiti data points. Per ogni cluster si definisce un centroide, ossia un punto al centro di un cluster. L'algoritmo è iterativo e si compone di tre macrofasi:

1. l'inizializzazione in cui si definiscono i parametri di input
2. l'assegnazione del cluster in cui ogni data points viene assegnato al centroide più vicino
3. l'aggiornamento della posizione del centroide in cui si ricalcola il punto esatto del centroide e di conseguenza si modifica la sua posizione

transaction_id	gender	quantity	amount_us	individual_price_us
40170	Female	12	6910.0	576
33374	Female	17	1699.0	100
14407	Female	23	4998.0	217
15472	Female	23	736.0	32
18709	Female	23	4389.0	191
...	...	...	...	...
67031	Female	20	5895.0	295
67043	Female	16	299.0	19
67045	Female	23	21990.0	956
67150	Female	22	8680.0	395
67181	Female	16	30901.0	1931

65375 rows × 4 columns

Figura 11: Dataset utilizzato per K-Means

La condizione di stop si ha quando nessun data points cambia cluster oppure la somma delle distanze è ridotta al minimo o infine quando viene raggiunto un numero massimo di iterazioni. Il K-Means è adatto per scenari in cui è possibile creare gruppi di oggetti simili da una collezione di oggetti distribuiti più o meno casualmente.

Abbiamo deciso di prendere come attributi su cui basare la clusterizzazione individual_price_us e quantity. Per agevolare l'algoritmo al calcolo dei cluster e poiché i due attributi hanno valori su scale diverse abbiamo deciso di normalizzare i dati utilizzando la metodologia *MinMax*, trasformandoli tutti in un intervallo compreso tra 0 e 1. Per capire quanti cluster prendere in considerazione abbiamo usato l'Elbow Method e dal grafico ottenuto, Figura 12, si prende il gomito più significativo che la curva crea come numero di cluster.

Come notiamo si ha un forte gomito corrispondente al valore 2, però abbiamo deciso di prendere come valore 3 per un maggiore contenuto informativo. Deciso il numero di cluster applichiamo l'algoritmo del K-Means e nel grafico, Figura 13, i colori rosso, azzurro e verde rappresentano ciò che l'algoritmo ha individuato, in viola sono rappresentati i centroidi dei rispettivi

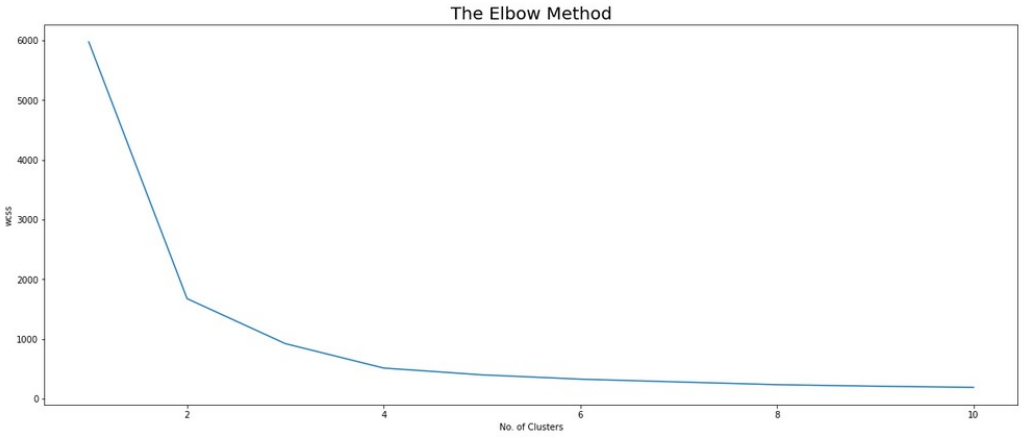

Figura 12: Elbow Method con individual_price_ e quantity

cluster.

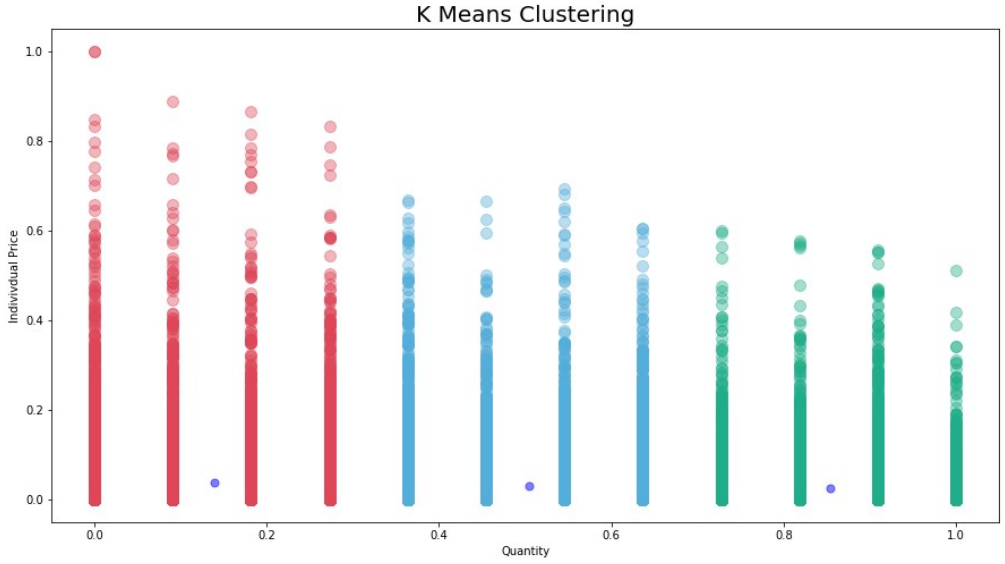

Figura 13: Clusterizzazione effettuata dal k-Means

Notiamo che il K-Means non fa altro che creare 3 cluster in base alle quantità vendute, quindi abbiamo un cluster, quello rosso, che indica che sono stati venduti pochi pezzi, il cluster blu indica che sono stati acquistati un numero medio di pezzi ed infine quello verde rappresenta un numero elevato di quantità ordinate.

3.2 PCA

Con la PCA si passa a clusterizzare prendendo in considerazione tutti i campi della tabella, non solo due. Come primo passo sono state reinserite le seguenti colonne: gender, city e category. Per prendere in considerazione anche questi i campi del dataset sono stati resi in variabili *dummy*[1]. Il dummie è stato effettuato per tutte le colonne ad eccezione delle seguenti: quantity, individual_price e amount_us che sono valori numerici quindi non necessitavano dell'operazione. Da questo processo è stata ottenuta la seguente tabella in figura 14.

transaction_id	date	product	gender	device_type	state	city	category	customer_login_type	delivery_type	quantity	amount_us
40170	14/11/2013	Hair Band	Female	Web	New York	New York City	Accessories	Member	one-day deliver	12	6910.0
33374	05/11/2013	Hair Band	Female	Web	California	Los Angeles	Accessories	Member	one-day deliver	17	1699.0
14407	01/10/2013	Hair Band	Female	Web	Washington	Seattle	Accessories	Member	Normal Delivery	23	4998.0
15472	04/10/2013	Hair Band	Female	Web	Washington	Seattle	Accessories	Member	Normal Delivery	23	736.0
18709	12/10/2013	Hair Band	Female	Web	Washington	Seattle	Accessories	Member	Normal Delivery	23	4389.0
...	...	...	...	...	...	...	...	...	...	...	...
67031	13/12/2013	Shoes	Female	Mobile	Washington	Seattle	wearables	Member	Normal Delivery	20	5895.0
67043	13/12/2013	Shoes	Female	Mobile	Washington	Seattle	wearables	Member	Normal Delivery	16	299.0
67045	13/12/2013	Shoes	Female	Mobile	Washington	Seattle	wearables	Member	Normal Delivery	23	21990.0
67150	13/12/2013	Shoes	Female	Mobile	Washington	Seattle	wearables	Member	Normal Delivery	22	8680.0
67181	13/12/2013	Shoes	Female	Mobile	Washington	Seattle	wearables	Member	Normal Delivery	16	30901.0

65534 rows × 13 columns

Figura 14: Tabella dopo operazione di dummy

È stata eliminata la colonna *data* poiché non utile per la clusterizzazione, insieme a tutte le colonne che esprimevano una grandezza temporale.

Successivamente è stato disegnato l'Elbow Method per capire quale fosse il numero di cluster da utilizzare nell'analisi.

Questa volta è possibile notare un gomito corrispondente al valore 3; scegliendo questo come numero di cluster otteniamo i risultati di figura, 16.

3.3 DBSCAN

Un altro test di clustering è stato effettuato con PCA e usando l'algoritmo DBSCAN. Per funzionare correttamente richiede in ingresso la conoscenza

[1] `https://pandas.pydata.org/docs/reference/api/pandas.get_dummies.html`

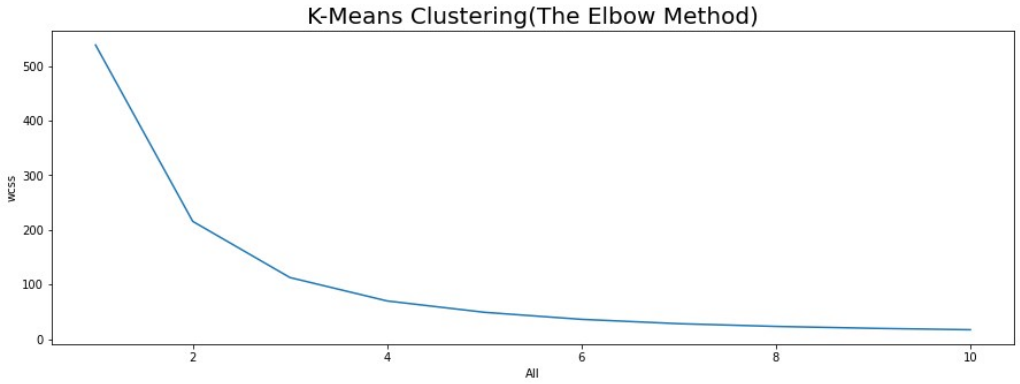

Figura 15: Elbow Method per la PCA

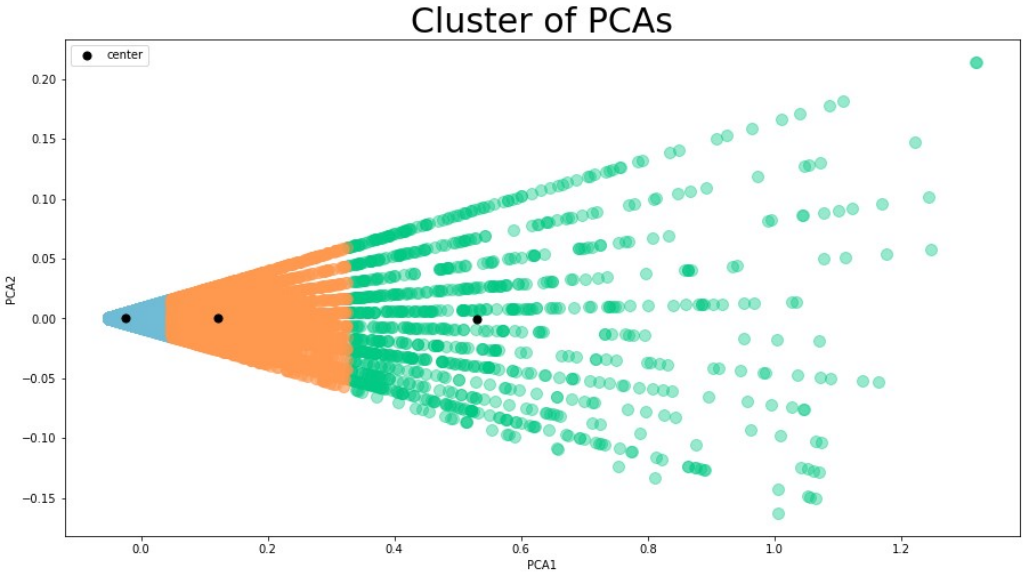

Figura 16: Clusterizzazione effettuata con la PCA

del parametro ε, che indica la distanza massima che due punti devono avere tra di loro affinché appartengano allo stesso cluster. Per trovare il valore "ottimo" di ε generiamo un grafico seguendo l'algoritmo di Nearest Neighbors per calcolare la distanza media tra ogni punto e i suoi n vicini. L'ε deve corrispondere ad un valore che ci permetta di passare tra un cluster e l'altro, dunque viene scelto il punto in cui il grafico aumenta la propria pendenza in maniera più repentina.

In base al grafico ottenuto, figura 17, abbiamo deciso di definire un ε pari a 0.01.

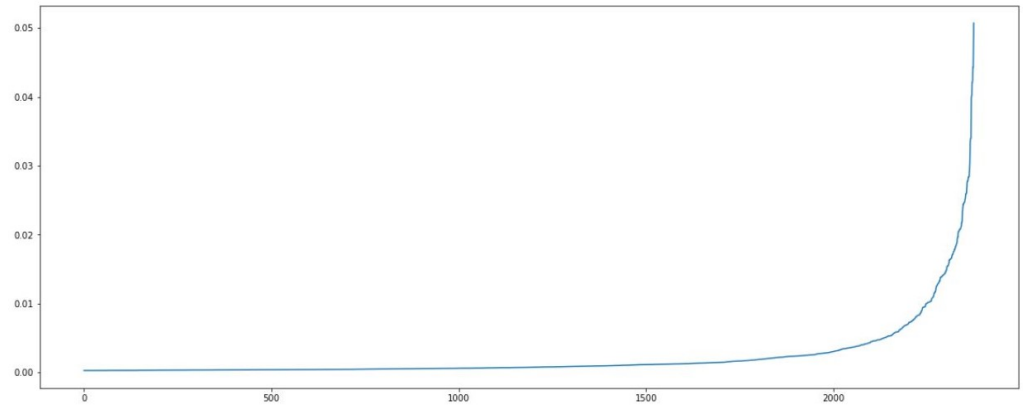

Figura 17: Grafico Nearest Neightbors

Al fine di fare clustering secondo l'algoritmo DBSCAN bisogna definire un ulteriore parametro *minSamples*, ovvero il minimo numero di punti richiesti per formare un cluster. Per questo parametro abbiamo deciso di selezionare 4 ovvero pari al doppio della dimensione della PCA passata all'algoritmo. Come si mostra in figura 18 il clustering non ha eseguito una buona separazione del dataset: del resto sono stati individuati solamente due cluster, di conseguenza sono poco descrittivi.

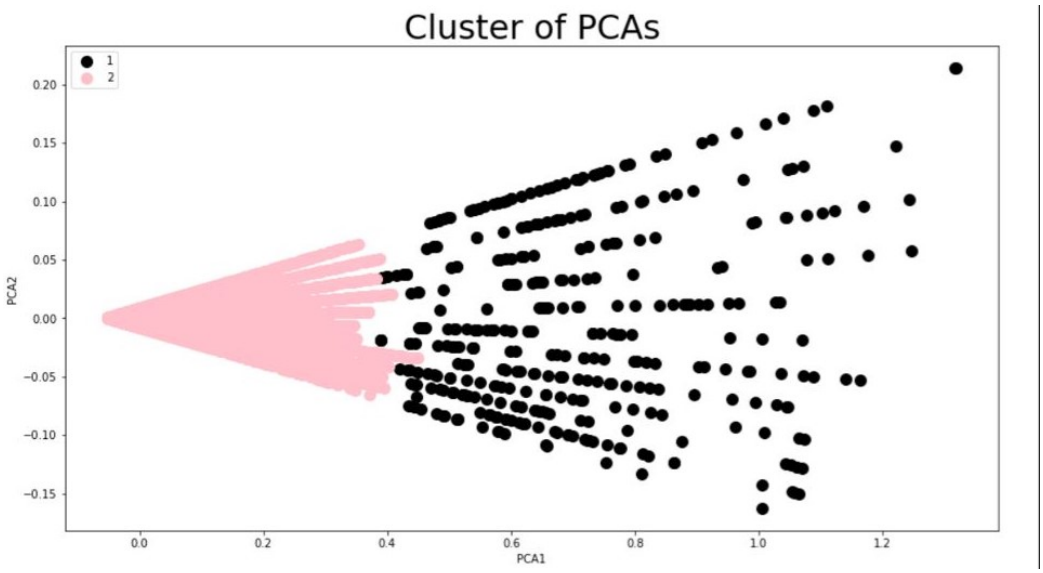

Figura 18: Clusterizzazione effettuata con DBSCAN

3.4 Yellowbrick

Infine si è provato ad effettuare la ricerca del numero ottimale dei cluster con la libreria di Python Yellobrick. In primo luogo abbiamo deciso di prendere il dataset ottenuto dalla PCA, lo abbiamo sottoposto all'Elbow Method con la funzione *KElbowVisualizer*. Questa funzione va alla ricerca del numero ottimale K di cluster sulla base del grado di distorsione tra gli elementi al loro interno. Passandogli il modello K-Means i risultati si vedono in figura 19.

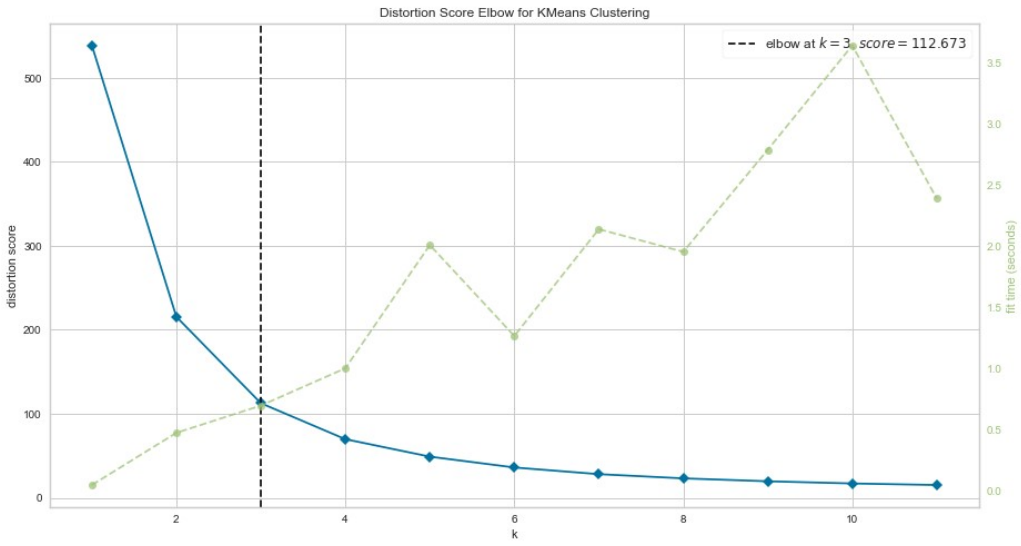

Figura 19: Elbow Method sulla PCA

Come si può notare il numero ottimale K è 3, sulla base di questo valore abbiamo deciso di adottare la *silhouette* come metrica per valutare la bontà dei cluster. I risultati che abbiamo ottenuto sono mostrati in figura 20; possiamo osservare che questo valore non produce un buon risultato dato che solamente un cluster contiene dei valori sopra l'indice *Average Silhouette Score*, oltre a questo la dimensione dei cluster è molto diversa tra di loro.

Utilizzando la stessa metodologia abbiamo deciso di rianalizzare il dataset della sezione K-Means che comprendeva l'individual_price_us e la quantity. Di seguito i risultati dell'Elbow Method, figura 21, e la Silhouette associata, figura 22.

Come possiamo notare in questo i risultati sono migliori dei precedenti; infatti con un $K = 3$ i tre cluster hanno diversi valori al di sopra dell'*Average Silhouette Score*, e le dimensioni sono molto più regolari.

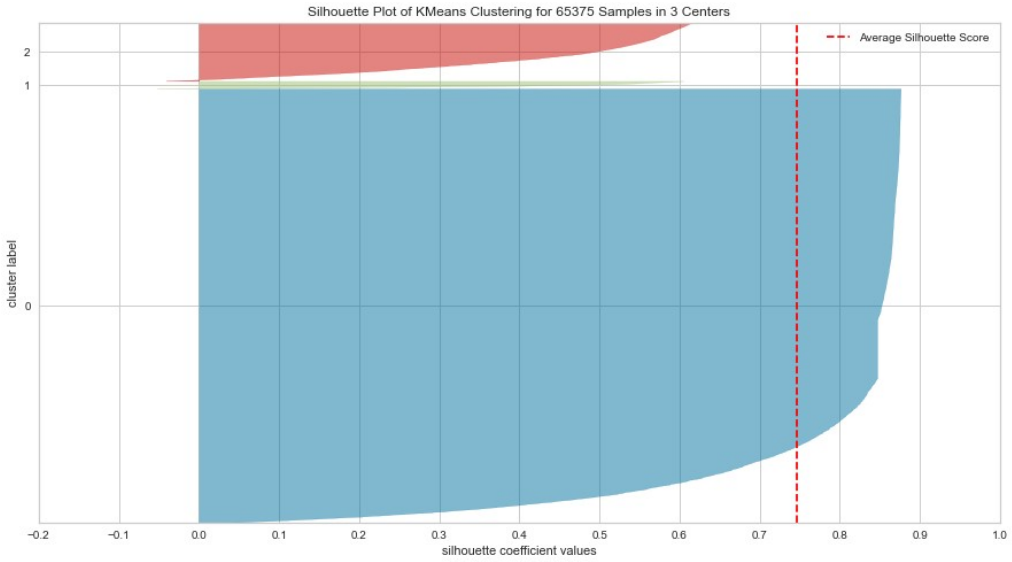

Figura 20: Silhouette del K-Means sulla PCA

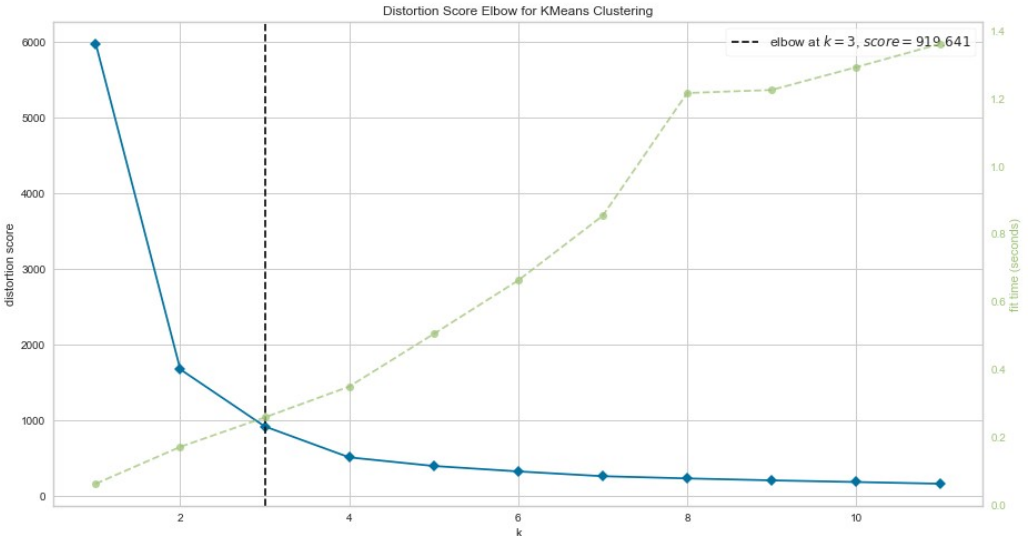

Figura 21: Elbow Method

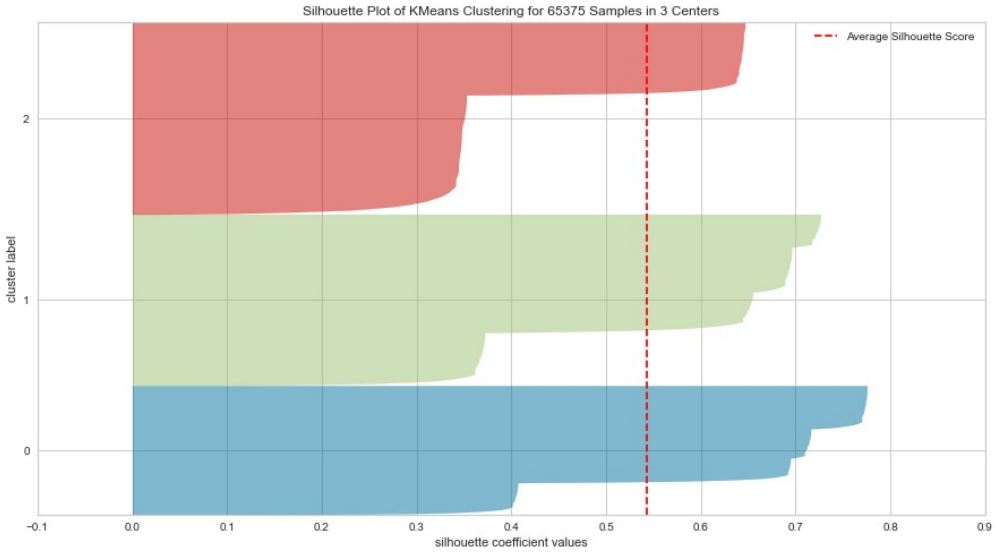

Figura 22: Silhouette del K-Means

4 Classificazione

In questa fase dello studio si va ed effettuare la classificazione usando diversi algoritmi e visualizzando i risultati per ognuno di questi. Per effettuare la classificazione sono stati prese in considerazione quattro tipologie di prodotti, ovvero quelli più venduti:

- Fairness Cream
- Jean
- Shirt
- Spectacles

Come primo passo è stato preso il dataset modificato per il clustering e sono state rimosse tutte le tuple dei prodotti che non ci interessavano. È stata calcolata la correlazione del dataset risultante che mostriamo attraverso la heatmap, figura 23.

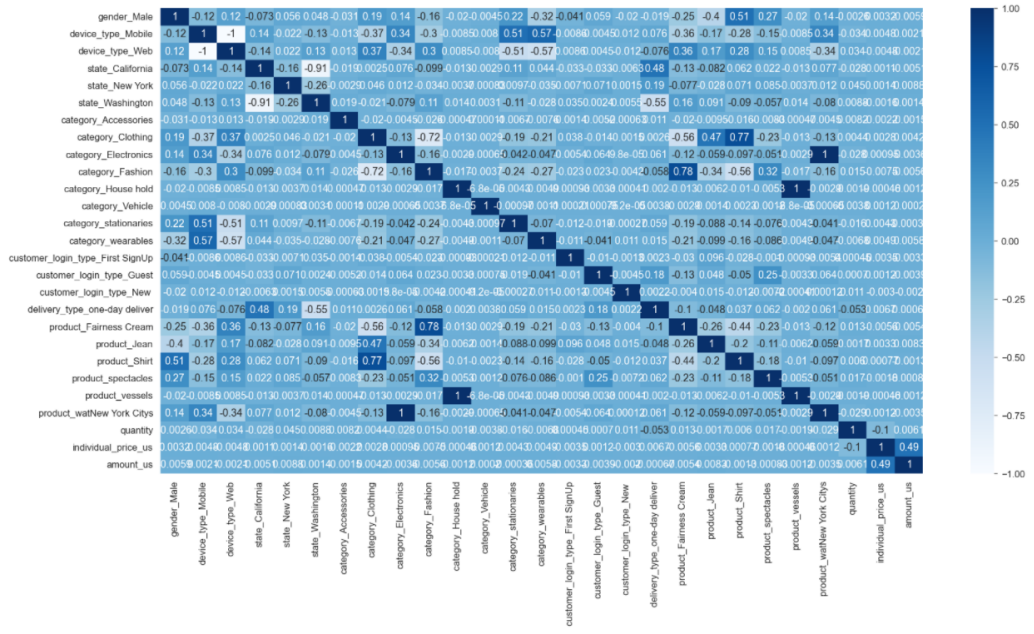

Figura 23: Heatmap del dataset

Sono state rimosse le colonne con una correlazione alta sia positivamente che negativamente così da alleggerire il dataset di colonne superflue ai fini della classificazione, ottenendo la figura 24.

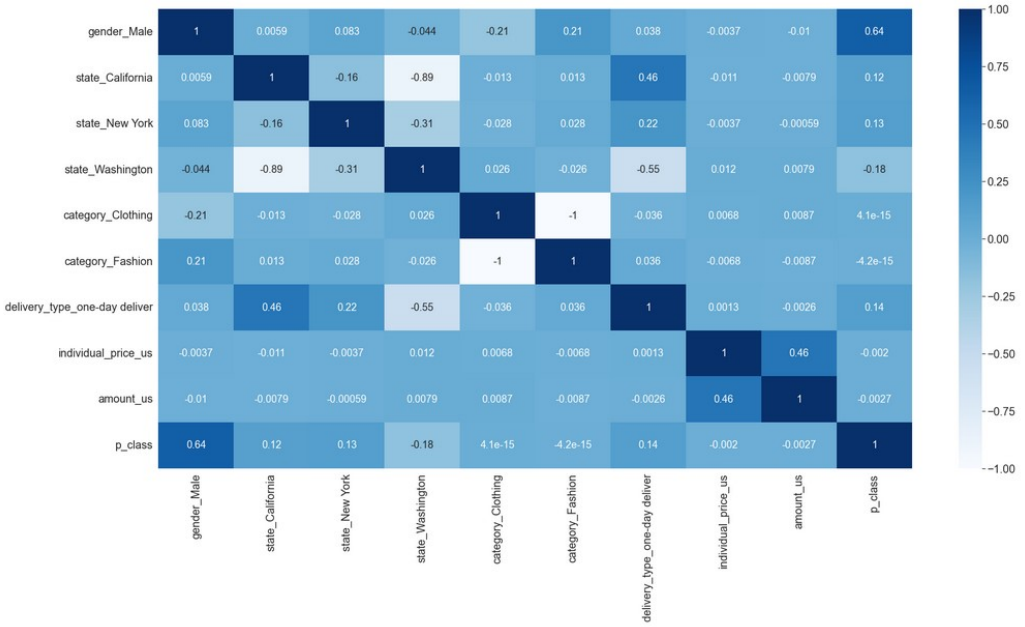

Figura 24: Heatmap aggiornata

Si è passati a bilanciare il dataset e successivamente lo si è normalizzato attraverso la funzione *MinMaxScaler()*.

Il dataset usato per il train è pari all'80% del dataset totale bilanciato e il restante 20% viene usato per la fase di testing. I prodotti Fairness Cream sono quelli indicati dal numero o, Jean dal numero 1, shirt dal numero 2 ed in fine spectacles è quello indicato con il numero 3.

Gli algoritmi usati per la classificazione sono i seguenti:

1. Logistic Regression
2. Decision Tree
3. SVC
4. Random Forest

Eseguiti gli algoritmi di classificazione sopra citati abbiamo ottenuto i risultati in termini di accuratezza riportati, in figura 25 ovvero il rapporto tra il numero di predizioni corrette ed il numero totale di casi.

I risultati dell'accuratezza dei modelli usati sono i seguenti:

- Logistic Regression: 0.89

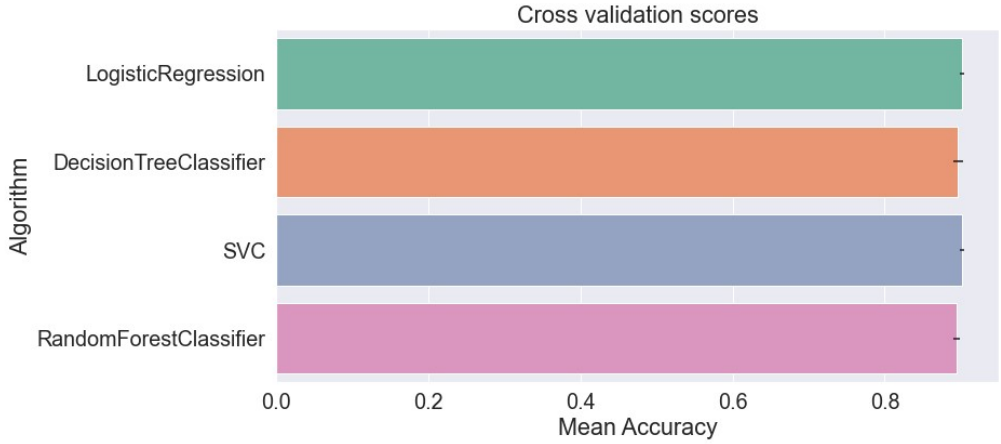

Figura 25: Accuratezza degli algoritmi di classificazione

- Decision Tree: 0.88
- Svc: 0.89
- Random Forest: 0.88

In seguito a questi risultati abbiamo deciso di effettuare un'analisi puntuale per ogni tipologia di prodotto valutando la matrice di confusione per ogni singolo algoritmo che mostriamo in figura 26

Possiamo notare come i classificatori vadano a predire le classi del dataset con un numero esiguo di errori, ad eccetto della classe 0, corrispondente alla Fairness Cream la quale viene confusa con la classe 3, equivalente alla tipologia spectacles; entrambe sono appartenenti alla categoria Fashion, quindi non è facile distinguerle sulla base delle colonne messe a disposizione.

Nel report di classificazione, figura 27, vengono mostrati i valori dei principali indici usati per valutare i classificatori.

Possiamo notare che in generale non ci sono grandi differenze tra i singoli algoritmi ad eccezione per la metrica *recall* la quale appare uguale 0.58 per quanto riguarda gli algoritmi di LogisticRegression e SVC, mentre migliora fino a 0.76 per DecisionTreeClassifier e RandomForestClassifier in merito alla tipologia FairnessCream.

A questo punto si è andati a valutare la presenza di eventuali correlazioni tra i modelli; come mostra la figura 28 notiamo una forte correlazione tra i modelli LogisticRegression e SVC, i quali ci restituiscono predizioni simili tra di loro. Questo grafico ci suggerisce che nell'operazione grid-search, ossia un algoritmo per andare testare tutte le possibili combinazioni di iperparametri,

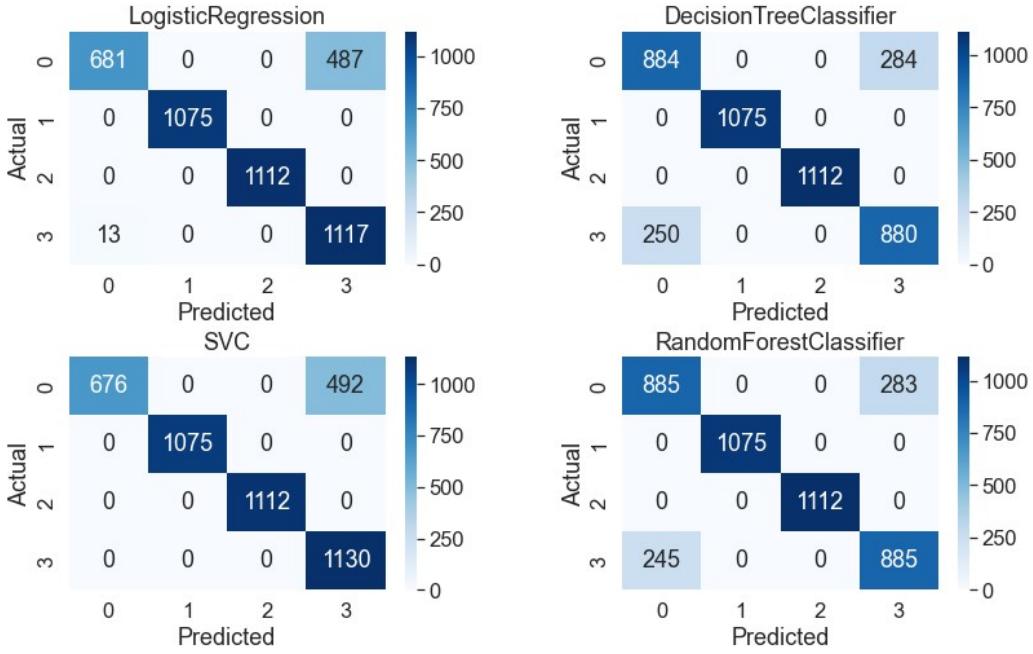

Figura 26: Heatmap degli algoritmi di classificazione

valuteremo i classificatori DecisionTree e RandomForest. I parametri per la grid-search sono mostrati in figura 29.

Eseguito l'addestramento la figura 30 valutiamo i modelli con e senza grid-search in termini di accuracy.

Successivamente abbiamo deciso di utilizzare il *VotingClassifier*, ovvero un classificatore che combinando tre classificatori differenti (LogisticRegression, RandomForest, DecisionTree) dove si vota sulla base delle predizioni degli altri modelli per decretare la classe di appartenenza. Valutando il risultato ottenuto abbiamo notato che non c'è un effettivo miglioramento dato che il risultato ottenuto è pari allo 0.89 quindi in linea con l'utilizzo del classificatore singolo.

Utilizzando il risultato ottenuto dal VotingClassifier abbiamo deciso di analizzare i risultati tracciando la curva ROC, figura 31. Come possiamo notare dal grafico le curve ROC della tipologia Jean e della tipologia Shirt non sono visualizzabili dato l'estrema precisione del nostro classificatore. Mentre per la Fairness Cream e spectacles abbiamo due curve che raggiungono presto il valore pari ad 1 del tasso di True Positive, sotto allo 0.2 rispetto ai False Positive. Questo è dovuto al fatto che le categorie sono fortemente discriminanti nella classificazione e combinate con le altre colonne permettono un basso valore di Falsi Positivi per raggiungere il grado pari ad 1 dei Veri

LogisticRegression Classification Report:

	precision	recall	f1-score	support
1.0	0.98	0.58	0.73	1168
2.0	1.00	1.00	1.00	1075
3.0	1.00	1.00	1.00	1112
4.0	0.70	0.99	0.82	1130
accuracy			0.89	4485
macro avg	0.92	0.89	0.89	4485
weighted avg	0.92	0.89	0.88	4485

DecisionTreeClassifier Classification Report:

	precision	recall	f1-score	support
1.0	0.78	0.76	0.77	1168
2.0	1.00	1.00	1.00	1075
3.0	1.00	1.00	1.00	1112
4.0	0.76	0.78	0.77	1130
accuracy			0.88	4485
macro avg	0.88	0.88	0.88	4485
weighted avg	0.88	0.88	0.88	4485

SVC Classification Report:

	precision	recall	f1-score	support
1.0	1.00	0.58	0.73	1168
2.0	1.00	1.00	1.00	1075
3.0	1.00	1.00	1.00	1112
4.0	0.70	1.00	0.82	1130
accuracy			0.89	4485
macro avg	0.92	0.89	0.89	4485
weighted avg	0.92	0.89	0.89	4485

RandomForestClassifier Classification Report:

	precision	recall	f1-score	support
1.0	0.78	0.76	0.77	1168
2.0	1.00	1.00	1.00	1075
3.0	1.00	1.00	1.00	1112
4.0	0.76	0.78	0.77	1130
accuracy			0.88	4485
macro avg	0.89	0.89	0.89	4485
weighted avg	0.88	0.88	0.88	4485

Figura 27: Metriche della classficazione

Positivi.

È stata tracciata successivamente la Precision Recall Curve, figura 32, sempre passando il modello ottenuto precedentemente. Dalla curva ottenu-

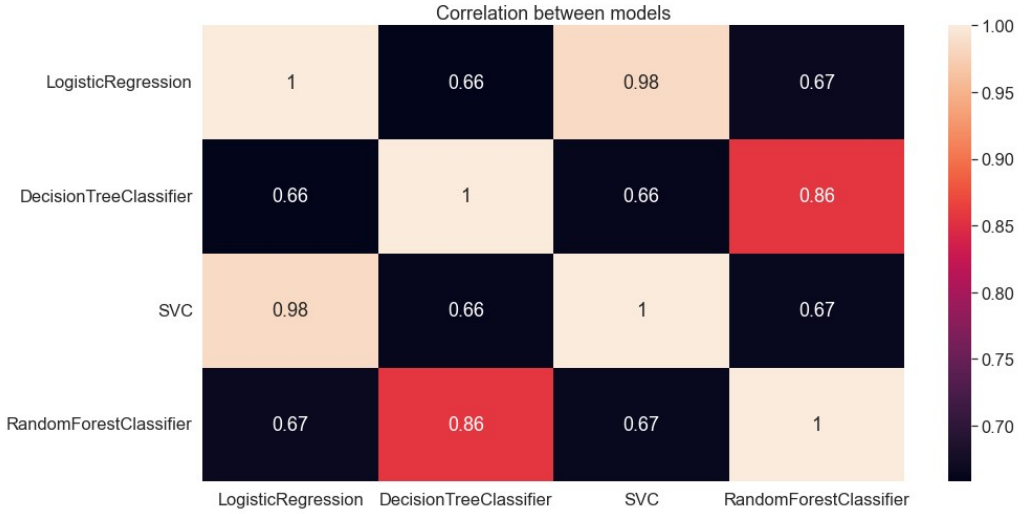

Figura 28: Heatmap che mostra la correlazione tra gli algoritmi di classificazione

```
DT_param = {"max_depth": [2,3,8,10],
            "max_features": [0.3, 0.7, 1],
            "min_samples_split": [2, 3, 10],
            "min_samples_leaf": [1, 3, 10],
            "criterion": ["gini"]}

RF_param = {"max_depth": [None],
            "max_features": [0.3, 0.7, 1],
            "min_samples_split": [2, 3, 10],
            "min_samples_leaf": [1, 3, 10],
            "bootstrap": [False],
            "n_estimators" :[100,300],
            "criterion": ["gini"]}
```

Figura 29: Parametri della grid-search

```
score without GridSearchCV:  0.896 0.894
score with GridSearchCV:  0.901 0.898
```

Figura 30: Risultati della grid-search

ta e dall'area sottostante alla curva come immaginavamo il tasso di Falsi Negativi è basso, ed abbiamo un basso tasso di Falsi Positivi.

Da questo grafo si può dedurre quali sono i punti ottimi in cui si ha sia una buona precision che una buona recall, questo punto coincide con il punto

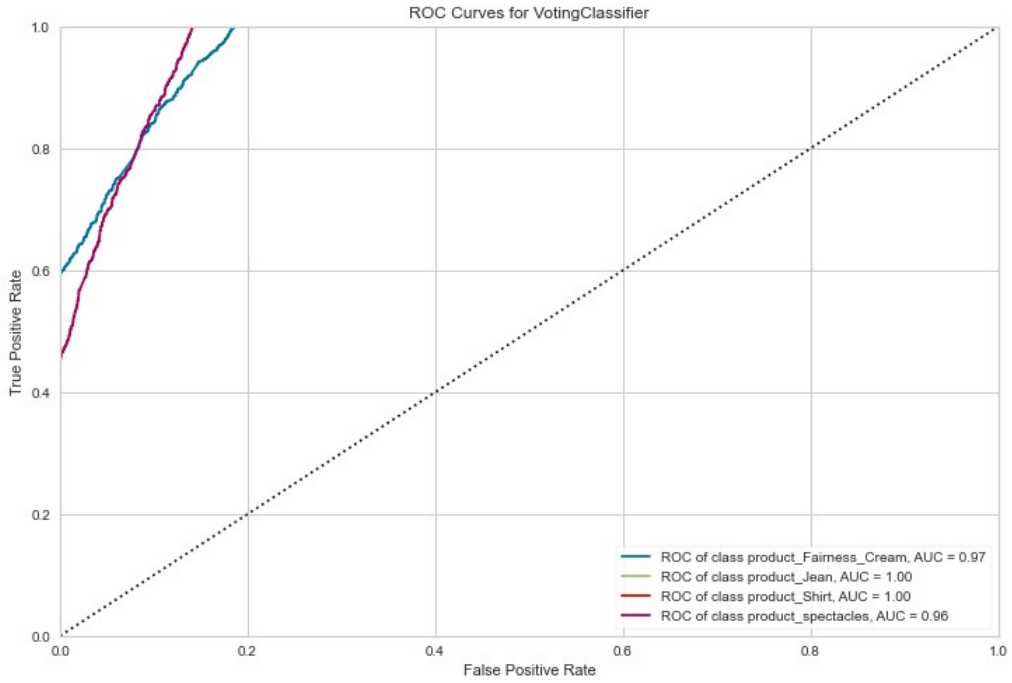

Figura 31: Curve ROC per le classi

di incontro della rette e della curva.

Infine è stato effettuato il plot della Learning Curve, figura 33, che mostra come il modello riesce ad apprendere e la variabilità della risposta man mano che vengono forniti più dati, la prima è rappresentata dalla retta blu mentre la seconda da quella verde.

Si nota che all'aumentare dei dati la curva della Cross Validation si avvicina alla curva del trainig, inoltre non c'è una varianza della risposta alta.

4.1 Considerazioni sulla Classificazione

In seguito ai risultati ottenuti abbiamo ritenuto interessante andare a valutare un'ulteriore classificazione nel caso in cui non si considerassero le categorie delle tipologie di interesse per vedere quanto incidessero effettivamente sul risultato, poiché secondo noi fossero determinanti ai fini dei risultati.

Per prima cosa abbiamo eliminato le due colonne che si riferivano alla categoria del prodotto ottenendo così una heatmap di correlazione mostrata in figura 34.

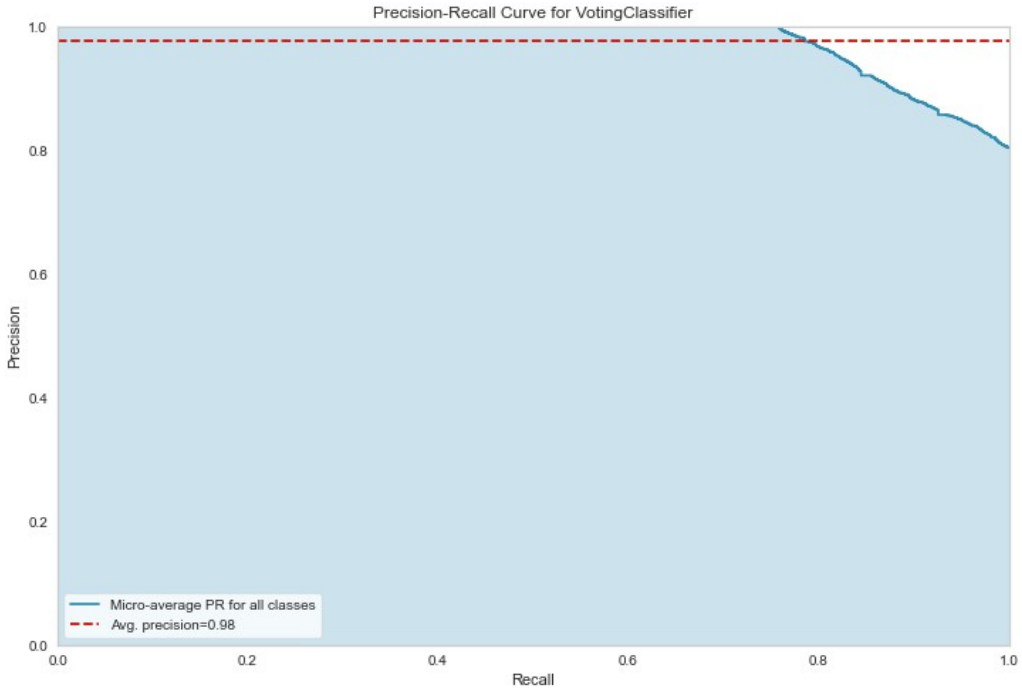

Figura 32: Curva di Precision-Recall

Applicando gli stessi algoritmi di classificazione otteniamo i risultati mostrati in figura 35, i quali mostrano un notevole cambiamento in negativo come ci aspettavamo.

Infatti possiamo vedere come i classificatori abbiano delle prestazioni inferiori ottenendo un risultato di accuratezza elencato di seguito:

- Logistic Regression: 0.51
- Decision Tree: 0.47
- Svc: 0.53
- Random Forest: 0.48

Quindi i risultati confermano le ipotesi che ci eravamo posti, sottolineando come le categorie siano determinanti per una buona classificazione. Nonostante questo notiamo come gli attributi rimanenti riescano comunque ad avere un risultato vicino al 50% basandosi comunque su attributi che apparentemente non sono significativi.

Seguendo i passi svolti nell'analisi precedente di seguito riportiamo la curva ROC, figura 36. Come mostra la figura abbiamo ottenuto delle curve

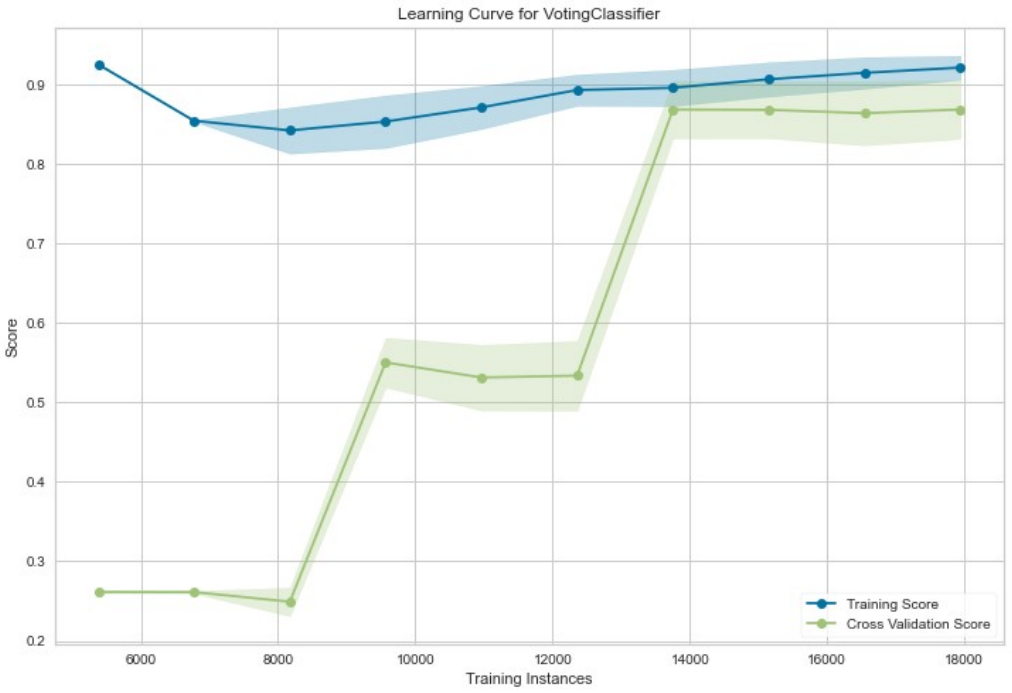

Figura 33: Learning Curve

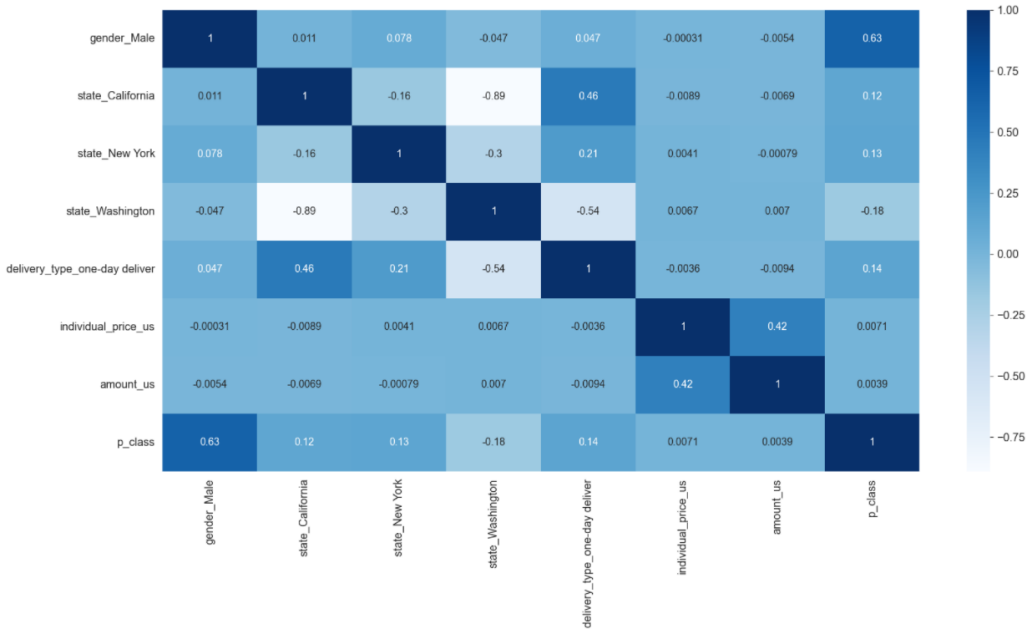

Figura 34: Heatmap Aggiornata

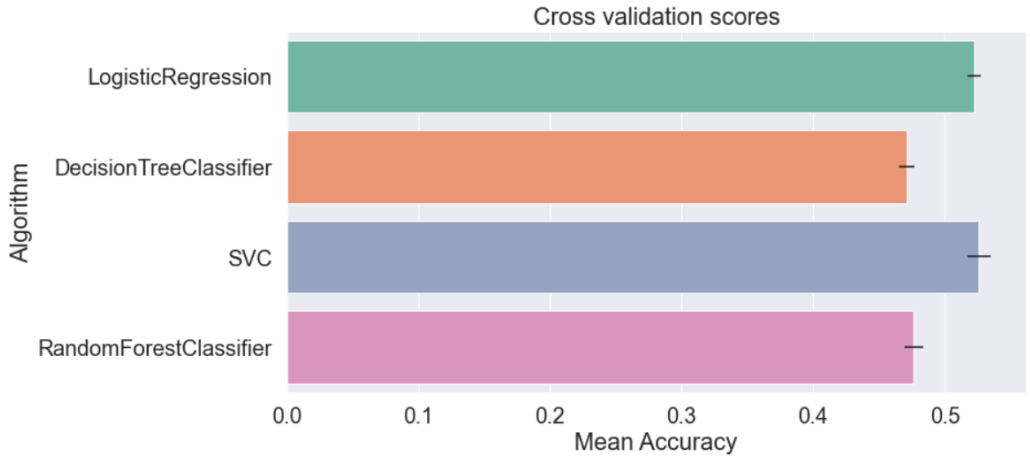

Figura 35: Risultati dei Classificatori aggiornati

decisamente diverse per ogni classe, dove la soglia per arrivare ad un True Positive Rate uguale 1 corrisponde ad una False Positive Rate maggiore. Nonostante questo Jean continua ad avere una soglia molto bassa di False Positive Rate per raggiungere il True Positive Rate pari ad 1.

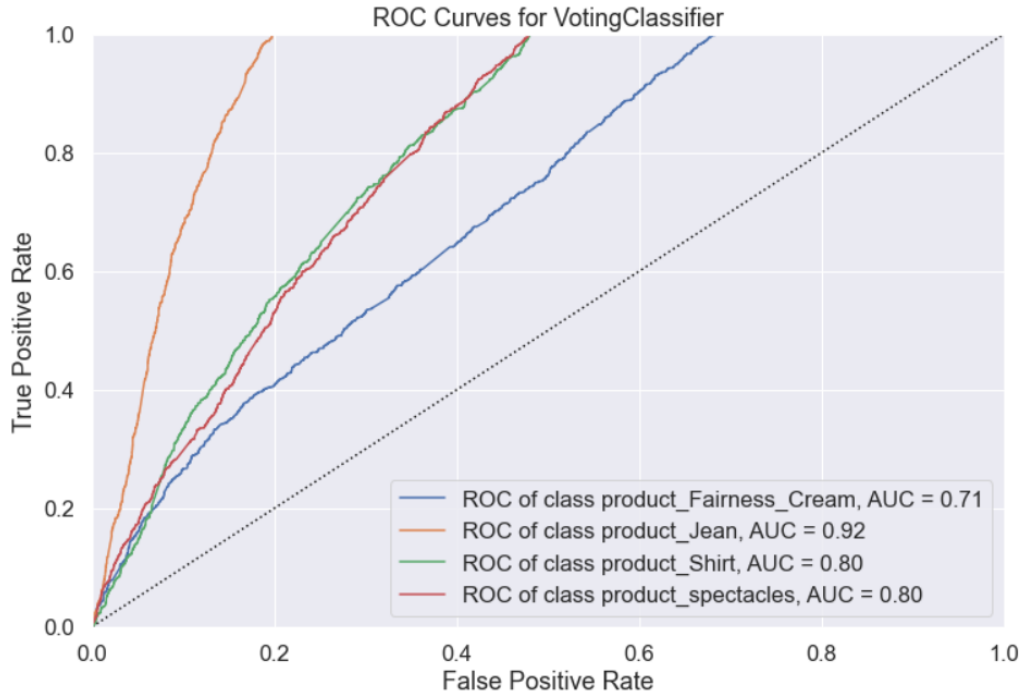

Figura 36: Curva ROC aggiornata

5 Serie Temporali

Ultima analisi fatta su questo dataset è quella riguardante le serie temporali. L'analisi è stata divisa in due fasi: in una prima fase si è andato ad analizzare l'andamento delle vendite giornaliere e poi si è passati ad una analisi oraria. In entrambi i casi è stata effettuata una prima analisi con ARIMA ed una successiva con SARIMAX.

Per questa analisi è stato ripreso il dataset originale ed è stata fatta una nuova fase di ETL, diversa a seconda dei due studi, per l'analisi giornaliera la sequenza di modifiche è la seguente:

1. Sono state eliminate quasi tutte le colonne ad eccezione di amount_us, date.
2. Poi abbiamo effettuato una *group by* andando a sommare i valori delle vendite per ogni singolo giorno presente nel dataset.
3. Abbiamo cambiato l'indice del dataset impostandolo sull'attributo *date*.
4. Abbiamo ordinato il dataset in base alla data, ovvero il nuovo indice.
5. Infine è stata notata la mancanza del valore di amount_us in data 2013-11-03, si è deciso di ovviare alla mancanza utilizzando il valore del giorno precedente.

Mentre per l'analisi oraria:

1. Sono state eliminate quasi tutte le colonne ad eccezione di amount_us, date, time.
2. Poi abbiamo effettuato una *group by* andando a sommare i valori delle vendite per ogni ora di ogni di ogni singolo giorno presenti nel dataset.
3. Abbiamo cambiato l'indice del dataset impostandolo su un nuovo attributo *date* dove al suo interno è presente sia il giorno che l'ora.
4. Infine abbiamo ordinato il dataset in base al nuovo attributo.

5.1 Serie Temporale Giornaliera

5.1.1 ARIMA

Come primo passo è stato effettuato il plot della serie e della corrispondente autocorrelazione con anche per i primi due ordini di derivazione, figura 37, ed in seguito è stato eseguito il test ADF per ognuna di esse.

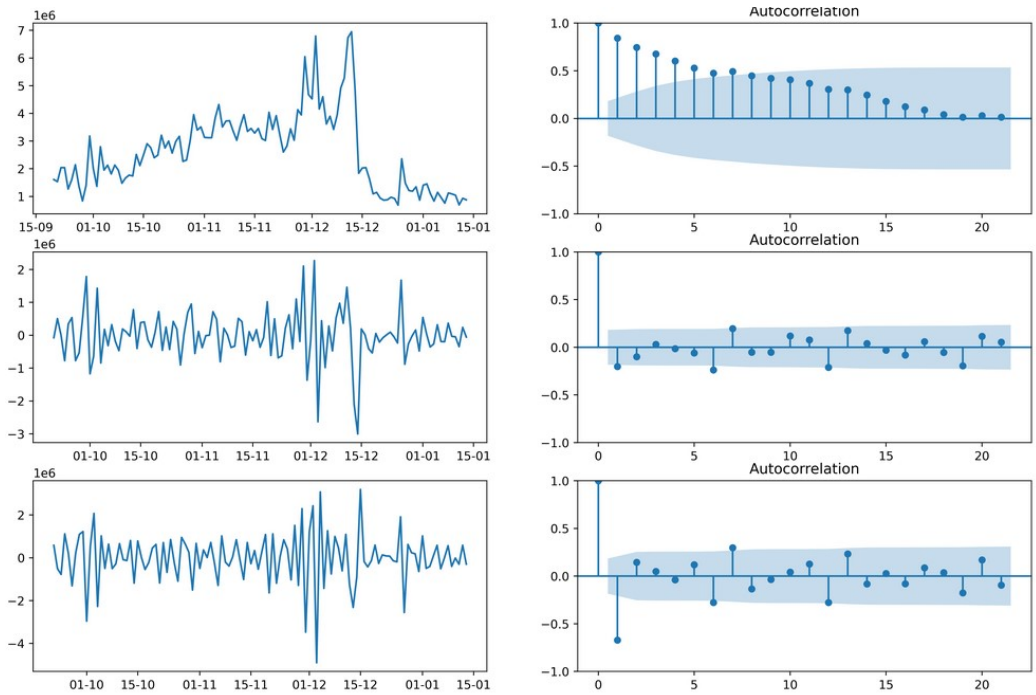

Figura 37: Serie e Autocorrelazione

	Serie originale	**Primo ordine di derivazione**	**Secondo ordine di derivazione**
p-value	0.136	2.750e-24	1.769e-07

Osservando i risultati ottenuti dall'ADF notiamo come la serie originale non sia stazionaria dato che il p-value ottenuto supera la soglia critica, a differenza del p-value del primo e del secondo ordine di derivazione. Quindi scegliamo di analizzare la serie con un ordine di derivazione scegliendo $d = 1$ per l'algoritmo di previsione ARIMA.

Successivamente abbiamo deciso di scegliere un fattore $p = 1$ poiché il primo lag è fuori dall'area di ammissibilità.

A seguire è stata effettuato uno studio per decidere la parte autoregressiva del modello, quindi per decidere quale valore far assumere al parametro q. Per effettuare questo studio sono stati generati i grafi sull'autocorrelazione parziale per il primo ordine, figura 38.

Dal grafico ottenuto abbiamo deciso di scegliere un valore di $q = 1$.

Da queste analisi si è arrivati alla conclusione che nel modello ARIMA verranno usati come parametri $p = 1$, $d = 1$ e $q = 1$.

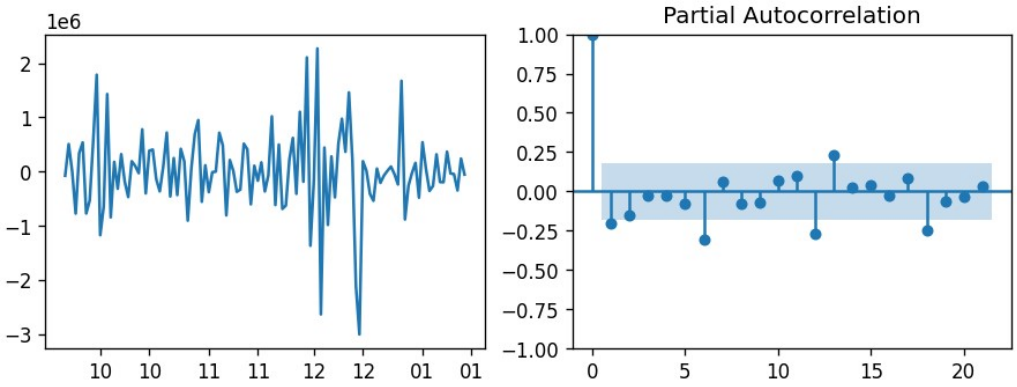

Figura 38: Primo ordine di derivazione e autocorrelazione parziale della serie

A questo punto una volta scelti i parametri per il modello lo abbiamo allenato ottenendo i seguenti risultati mostrati in figura 39.

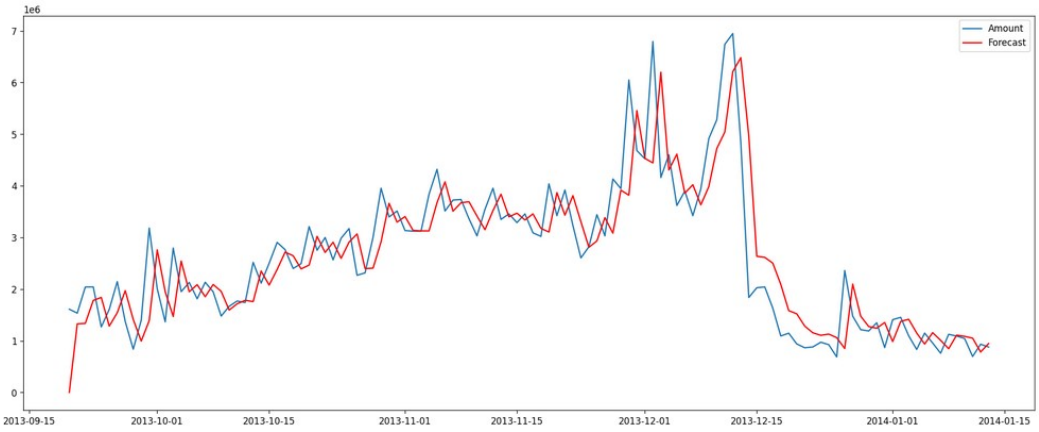

Figura 39: Modello con ARIMA

Notiamo come il modello segua generalmente l'andamento della serie ma non in modo ottimale come dimostrato dalle metriche.

MAE	518820.1216
MSE	543368510243.146
RMSE	737135.34
R-Squared	0.712

Il MAE indica la media delle differenze assolute tra i valori predetti e i valori reali, il valore ottenuto è alto influenzato anche dall'ordine di grandezza dei valori del dataset. L'MSE assume anch'esso un valore molto alto, questo

fornisce una misura sull'entità dell'errore, e questo comporta anche un elevato RMSE, dato che si ottiene dalla radice quadrata del MSE, probabilmente dovuto alle oscillazioni molto elevate. Infine è stato calcolato R-Squared, nel caso esaminato ha un valore pari a 0,712 questo indica che ARIMA non ha prodotto un buon modello.

Sono stati analizzati poi i residui, figura 40, dove non notiamo ci siano pattern particolari, considerazione possibile grazie al grafo a campana ottenuto.

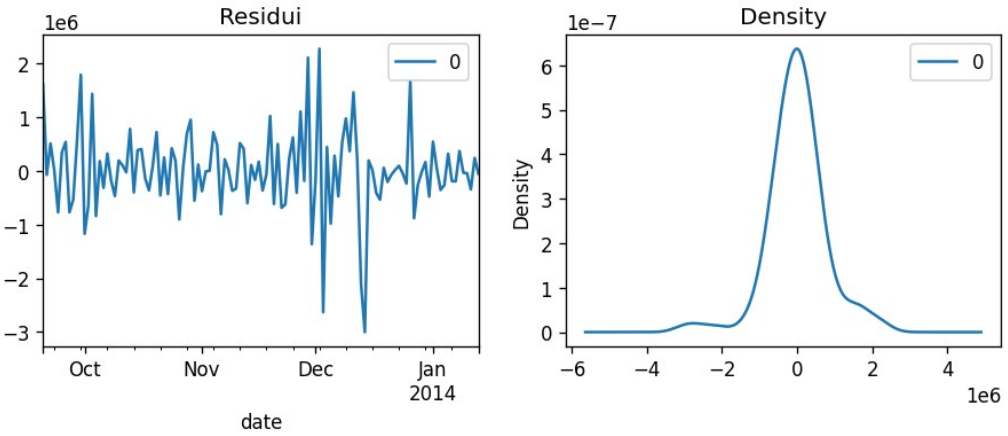

Figura 40: Andamento e densità dei residui

Infine si prova ad effettuare una predizione su quelle che potrebbero essere le possibili vendite nei futuri 14 giorni ottenendo il grafico in figura 41.

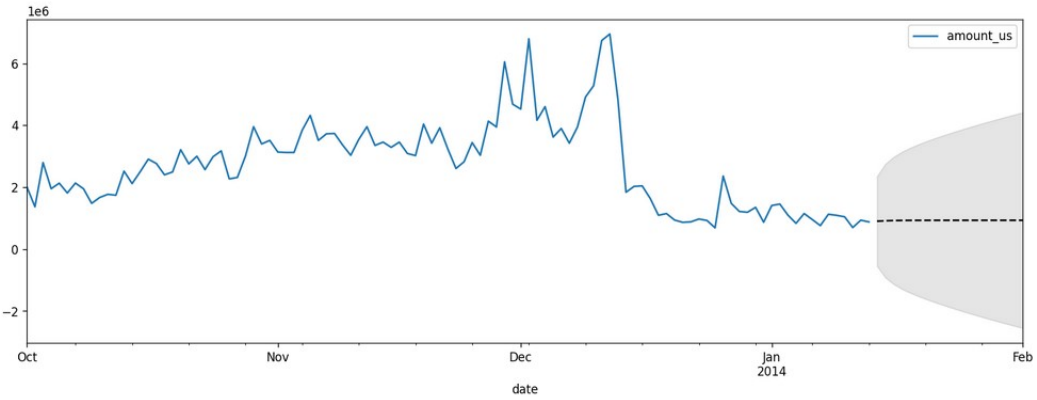

Figura 41: Previsioni con ARIMA

Si nota subito che il modello non riesce ad effettuare una predizione esaustiva, infatti il valore si assesta intono ad un valore medio dell'ultima parte del dataset.

Dato che il pattern che abbiamo eseguito stima i parametri non garantisce che sia il miglior modello possibile. Dunque abbiamo deciso di utilizzare un algoritmo che testasse tutte le possibili combinazioni dei parametri con range prefissati (nel nostro caso $0 < (q, p) < 4$, e $0 < d < 3$), ed in seguito basandosi sul valore dell'AIC più basso ottenuto per le diverse casistiche abbiamo deciso di addestrare nuovamente il modello con ARIMA. Dal test effettuato si verifica che il miglior modello ottenuto lo si ha per valori di $p = 3, d = 2, q = 3$. Di seguito i grafici, figura 42 e figura 43 e le metriche.

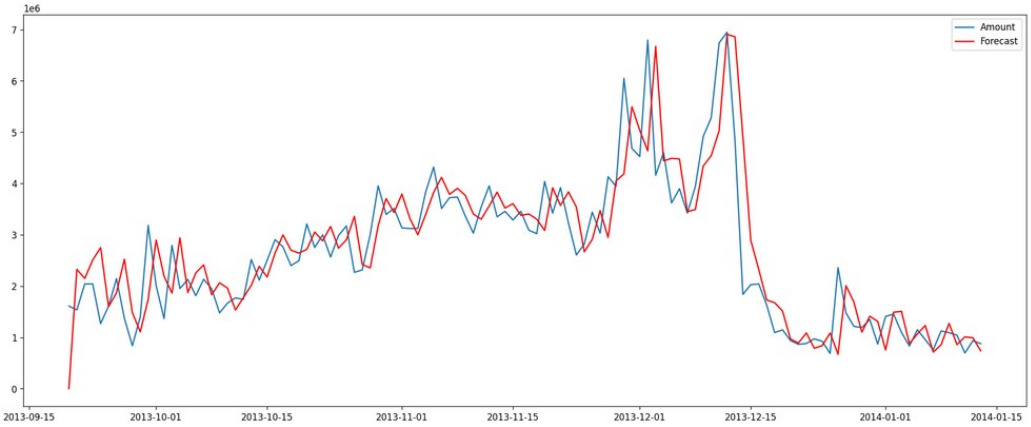

Figura 42: Modello con ARIMA

MAE	516549.837
MSE	556192953467.7549
RMSE	745783.44
R-Squared	0.705

Notiamo che le metriche non portano ad un sostanziale miglioramento quindi ARIMA non riesce a creare un buon modello per questo dataset e di conseguenza non riesce a predire al di fuori dell'andamento medio dell'ultima parte del dataset.

5.1.2 SARIMAX

Per l'analisi in SARIMAX è stato preso il dataset usato per l'analisi precedente. Come primo passo abbiamo deciso di effettuare una seasonal decompose per osservare la presenza di stagionalità, trend e andamento del rumore. Il significato dei tre elementi è il seguente:

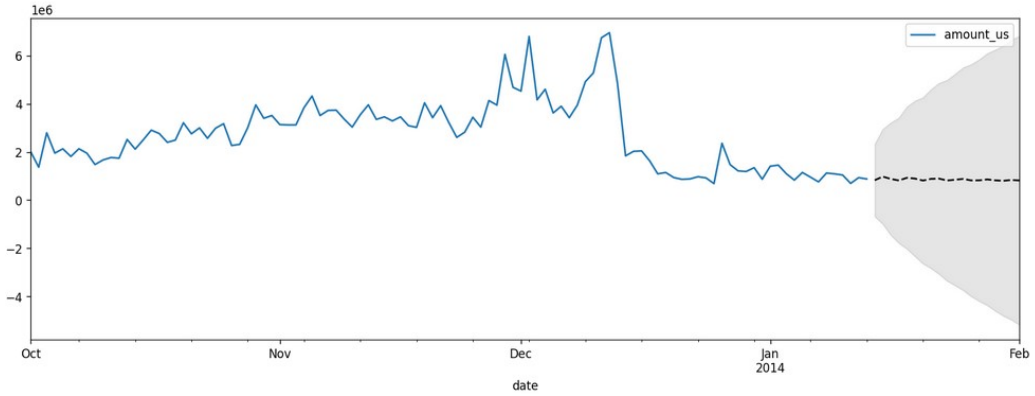

Figura 43: Previsioni con ARIMA

- Stagionalità: rappresenta un ciclo in una serie
- Trend: è una crescita o una decrescita della serie
- Residui o Rumore: variazioni random dei dati.

È stata usata la decomposizione stagionale automatica e sono stati tracciati i risultati, mostrati in figura 44, utilizzando il modello additivo. Mentre nella figura 45 abbiamo calcolato il modello moltiplicativo della seasonal decompose per mettere a confronto i residui. I residui per entrambi hanno un andamento casuale il che ci fa concludere che la serie può essere considerata sia additiva che moltiplicativa.

Nel primo grafico è rappresentato l'andamento della serie, nel secondo è mostrato il trend. Si nota che la serie non ha un trend ben definito dato che all'inizio cresce e poi è presente una brusca decrescita ad inizio 2014. Risalta all'occhio, invece, una forte stagionalità della serie in esame ogni 7 giorni. Il grafico dei residui mostra che c'è del rumore ma non c'è un andamento periodico.

È stato effettuato l'ADF test che ha restituito un p-value pari a 0.136494, quindi ben superiore alla soglia di 0.05, questo sta a significare che la serie non è stazionaria. Per questo motivo la serie è stata derivata di un ordine ed è stato rifatto L'ADF test che ha restituito un p-value pari a 0 e quindi ora si può affermare che la serie differenziata di un ordine è stazionaria. Per la scelta dei parametri p, d, q è stata seguita la stessa procedura della fase precedente, dunque si è scelta la terna $(1, 1, 1)$, a cui aggiungiamo il valore di $s = 7$ per la stagionalità presente nel dataset.

Di seguito mostriamo il modello, figura 46, le metriche, la previsione, figura 47.

Figura 44: Andamento della serie, trend, stagionalità e residui (Modello Additivo)

MAE	556440.24
MSE	577197936607.5
RMSE	759735.43
R-Squared	0.6943

Graficamente potremmo pensare che il modello abbia effettuato una predizione migliore rispetto all'ARIMA. Dando un'occhiata alle metriche però notiamo che questa cosa non è totalmente corretta: infatti l'R-Squared è simile a quello riscontrato in precedenza, se non leggermente peggiore, così come le altre metriche.

Come fatto per ARIMA abbiamo deciso di valutare le possibili combinazioni dei parametri da passare all'algoritmo SARIMAX, per capire qual fosse la combinazione migliore come nel caso precedente è stato valutata la metrica AIC. Dalla valutazione i parametri che esprimo il più basso valore di AIC sono $p = 2, d = 1, q = 2$ come parametri base, invece come parametri

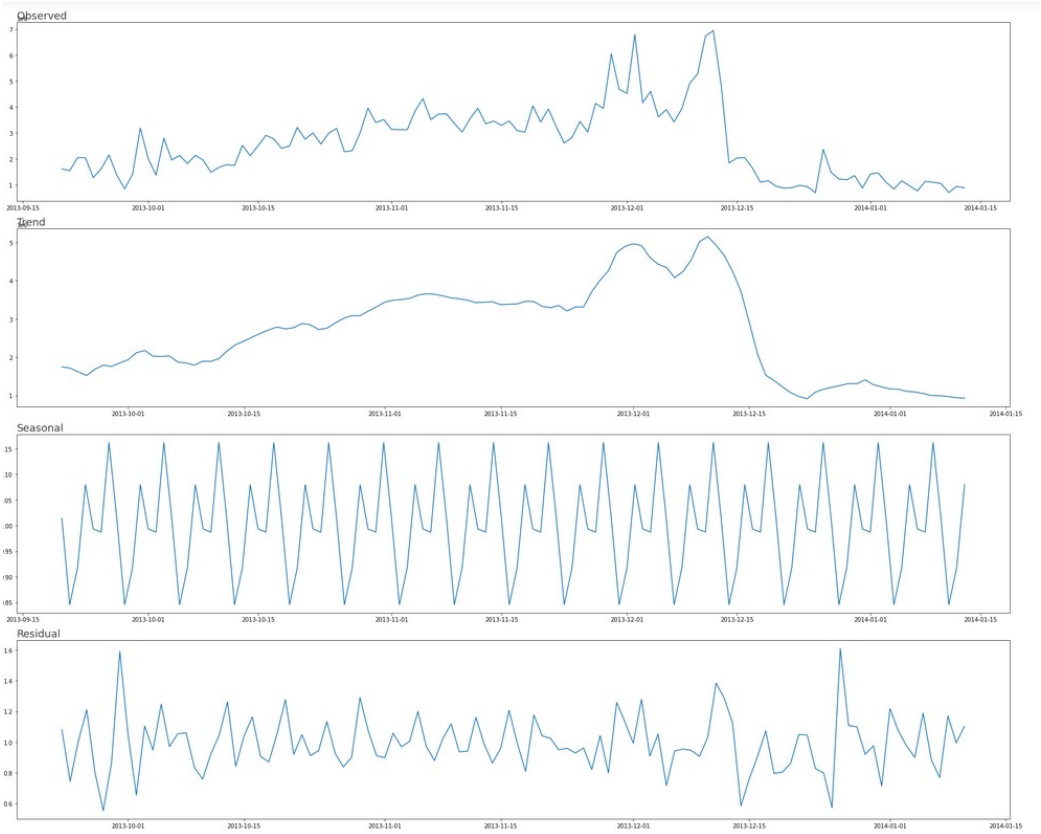

Figura 45: Andamento della serie, trend, stagionalità e residui (Modello Moltiplicativo)

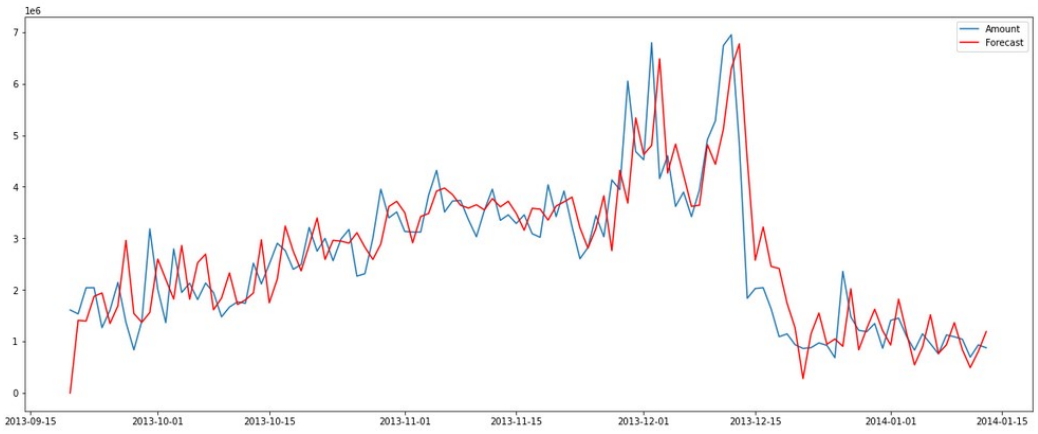

Figura 46: Modello con SARIMAX

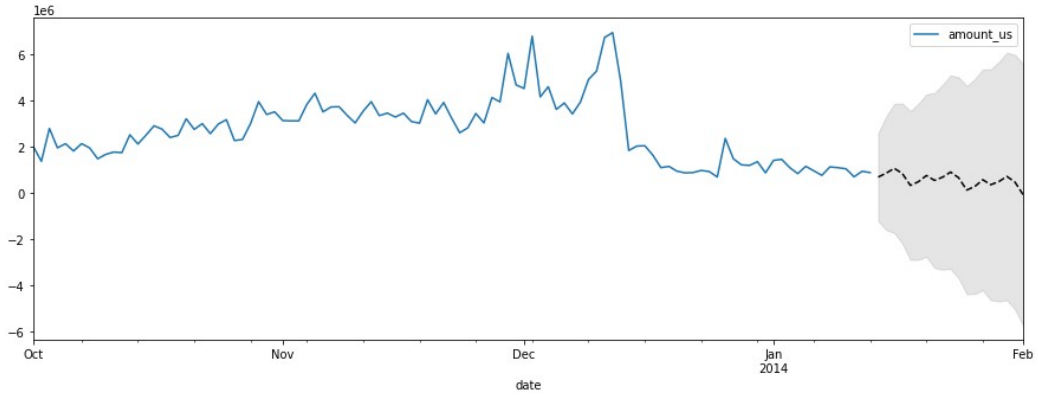

Figura 47: Previsione con SARIMAX

stagionali $p = 0, d = 1, q = 2, s = 7$. Nella figura 48 è raffigurato il modello, di seguito le metriche ed infine in figura 49 la previsione.

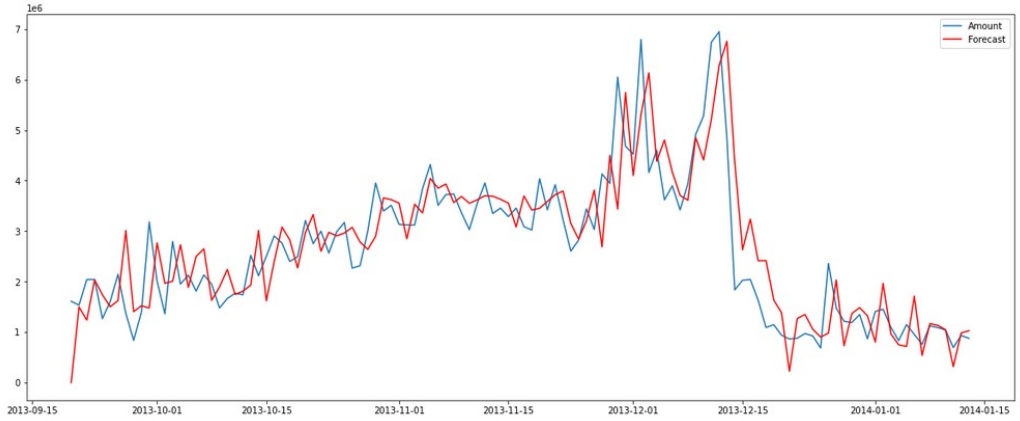

Figura 48: Modello Sarimax

MAE	549939.68
MSE	554220347410.8
RMSE	744459.76
R-Squared	0.70

Possiamo notare quindi come ci sia un leggero miglioramento anche se il risultato continua ad essere sempre sulla stessa linea più o meno, sia graficamente che nelle metriche.

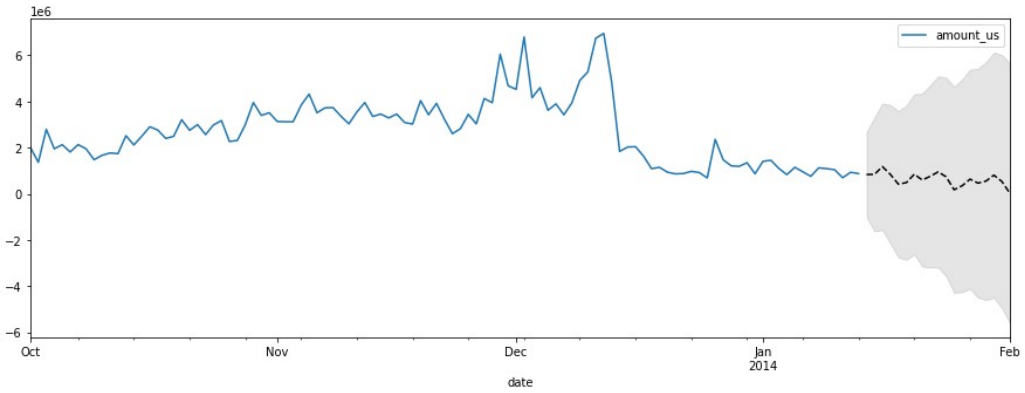

Figura 49: Previsione Sarimax

5.2 Serie Temporale Oraria

Data la disponibilità, come già detto, dell'orario ci siamo focalizzati su un'analisi oraria degli ordini, alla ricerca di eventuali orari di punta o pattern di acquisti nel corso delle giornate.

L'analisi che segue è stata realizzata sulla falsa riga di ciò che avevamo già illustrato a proposito della serie giornaliera, quindi prima un'analisi utilizzando l'algoritmo ARIMA per poi passare alla versione SARIMAX. [2]

5.2.1 ARIMA

Per prima cosa abbiamo plottato il nuovo grafico insieme alla sua autocorrelazione, stessa cosa fatta per il primo ed il secondo ordine di derivazione, come mostrato in figura 50.

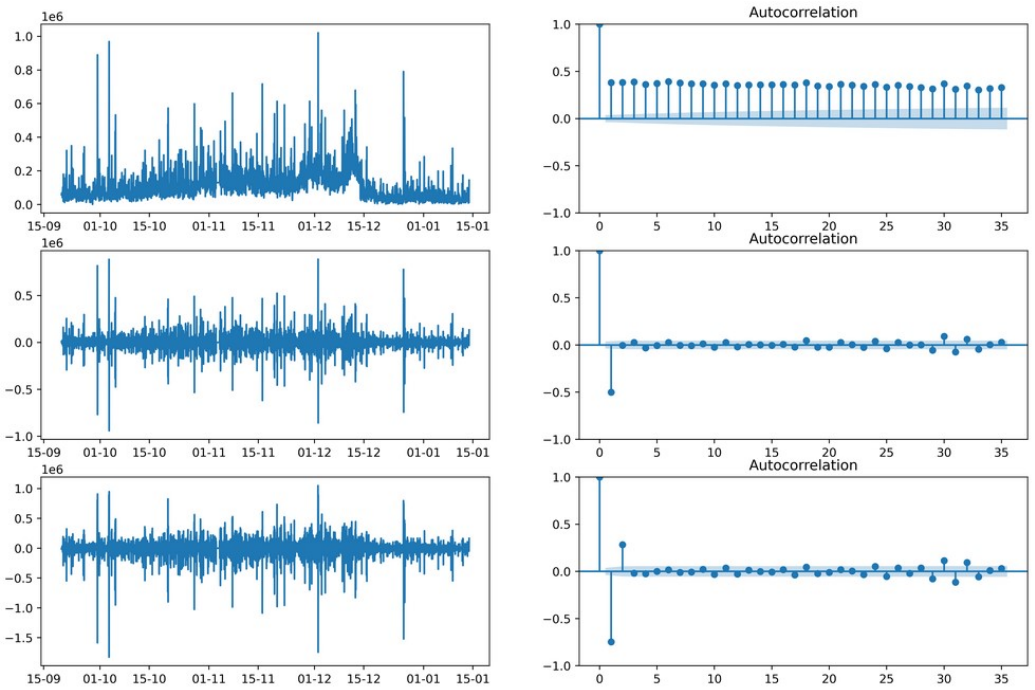

Figura 50: Plot Dataset Orario

Graficamente si può notare che per rendere stazionaria la serie necessitiamo del primo ordine di derivazione cosa confermata poi dall'ADF test,

[2]Non abbiamo utilizzato l'algoritmo per valutare la migliore combinazione dei parametri poiché avremo dovuto far riferimento ad un range di valori elevato, il che lo rendeva troppo pesante.

risultati mostrati di seguito. Nonostante la serie originale presenti un p-value pari a 0.037 abbiamo di scegliere la serie differenziata di un ordine quindi $d = 1$. Successivamente dal grafico dell'autocorrelazione al suo fianco possiamo scegliere un valore di $q = 1$, ed infine mostrando l'autocorrelazione parziale in figura 51 abbiamo scelto $p = 10$.

	Serie originale	**Primo ordine di derivazione**	**Secondo ordine di derivazione**
p-value	0.037	4.28e-28	0.00

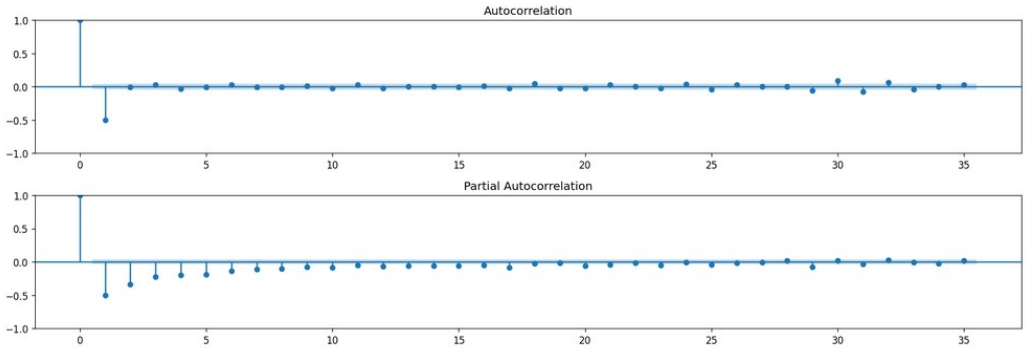

Figura 51: Plot Correlazione ed Autocorrelazione

Di conseguenza applicando la terza $(10, 1, 1)$ abbiamo ottenuto il seguente modello, mostrato in figura 52, le metriche di seguito, e la successiva previsione, figura 53.

MAE	43034.65
MSE	5280093189.8
RMSE	72664.24
R-Squared	0.34

Come possiamo notare sia dalle metriche sia dalla previsione i risultati ottenuti mostrano come ARIMA non riesca ad effettuare una buona previsione ed un buon modello rispetto al dataset orario.

5.2.2 SARIMAX

Come già fatto per il caso giornaliero abbiamo deciso prima di tutto di effettuare una seasonal decompose anche per il caso orario, figura 54.

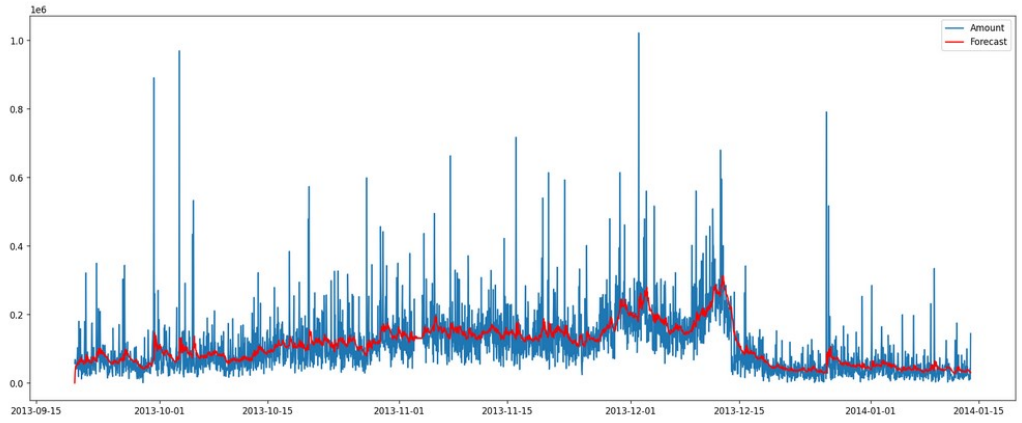

Figura 52: Modello Arima Orario

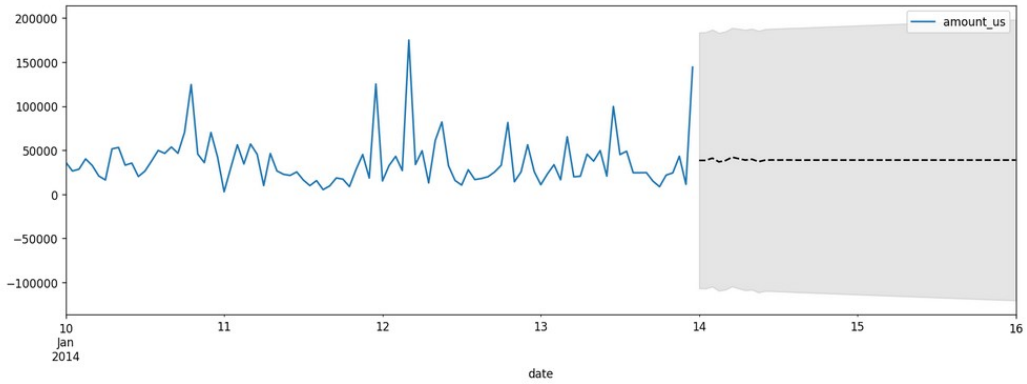

Figura 53: Previsione Arima Orario

Come si può notare non abbiamo un trend specifico, mentre la stagionalità è pari a 22 quindi il parametro $s = 22$ per l'algoritmo di SARIMAX. Per quanto riguarda i residui notiamo un andamento casuale. Successivamente è stato eseguito l'ADF come prima e quindi abbiamo settato $d = 1$, lasciando inalterati i parametri scelti nel caso ARIMA.

Di conseguenza abbiamo utilizzato la ternza $(10, 1, 1)$ con l'aggiunta di $s = 22$ per SARIMAX. Il modello ottenuto è mostrato in figura 55, le metriche di seguito e la previsione in figura 56.

MAE	44861.85
MSE	5343269188.10
RMSE	73097.66
R-Squared	0.33

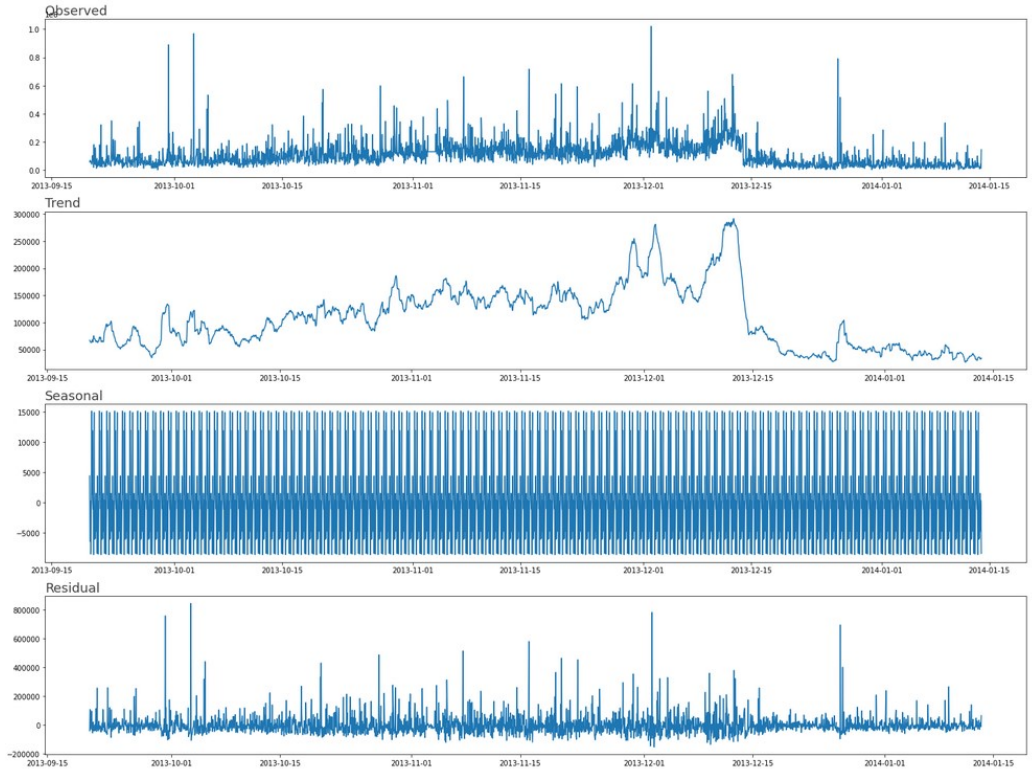

Figura 54: Seasonal Decompose

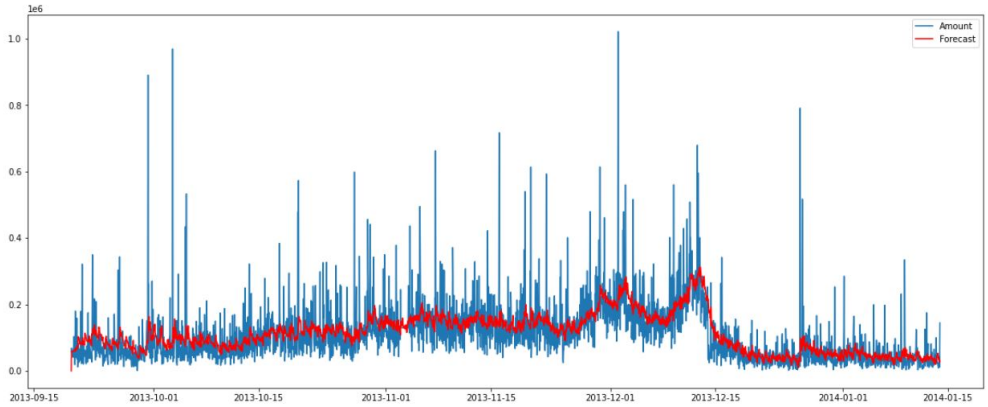

Figura 55: Modello Sarimax Orario

Come possiamo notare secondo la metrica R-Squared la validità del modello SARIMAX sia pressoché uguale con quello di ARIMA, come anche per il resto delle metriche. Nonostante le metriche non varino troppo in

realtà la previsione generata da SARIMAX appare più credibile rispetto alla controparte calcolata da ARIMA che si assestava attorno ad un valore medio.

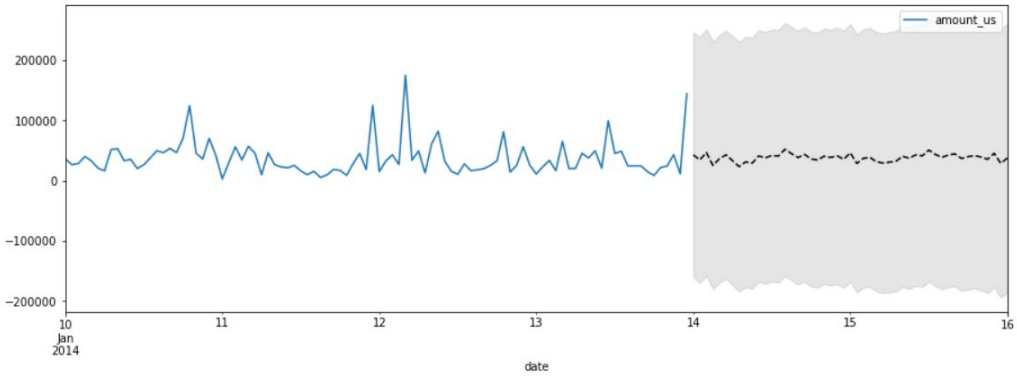

Figura 56: Previsione Sarimax Orario