START2IMPACT

Progetto Advanced Analytics

Studente: ***Alessandro Busà***

Indice

- Introduzione
- Primo Dataset Utilizzato
- Secondo Dataset Utilizzato
- Librerie Utilizzate
- Funzioni Utilizzate
- Vendite per Tipo di Cliente e Città + Grafico
- Vendite Totali per Tipo di Cliente + Grafico
- Totale Vendite per Città + Grafico
- Vendite Totali per Categoria di Prodotto + Grafico
- Vendite per Categoria di Prodotto e Genere + Grafici
- Vendite Totali per Genere + Grafico
- Vendite per Città, Genere e Categoria di prodotto + Grafico
- Classificazione della qualità delle mele + Grafico
- Mean Squared Error
- Trend delle Vendite Mensili nel Tempo (Grafico)
- GitHub

Introduzione

Questo script esegue un'analisi delle vendite suddivisa per città, genere e categoria di prodotto, insieme a una classificazione della qualità delle mele utilizzando un modello di machine learning.

- Obiettivi:
  - Analizzare i dati di vendita per identificare tendenze e differenze tra generi e categorie di prodotto.
  - Applicare il modello Random Forest per prevedere la qualità delle mele basandosi sulle caratteristiche.

- Metodologia:
  - Analisi delle vendite per genere e città.
  - Creazione di grafici a barre per visualizzare le vendite.
  - Classificazione delle mele e valutazione della performance del modello tramite matrice di confusione.
  - Visualizzazione del trend delle vendite mensili nel tempo.

Primo Dataset Utilizzato

Il dataset (supermarket_sales.csv) contiene circa 1.000 righe e 17 colonne. Ecco una breve descrizione delle colonne:

- ***Invoice ID***: Identificatore unico per ogni vendita.
- ***Branch***: Filiale del negozio.
- ***City***: Città in cui è avvenuta la vendita.
- ***Customer type***: Tipo di cliente (es. Membro, Normale).
- ***Gender***: Genere del cliente.
- ***Product line***: Categoria del prodotto (es. Salute e bellezza).
- ***Unit price***: Prezzo per unità del prodotto.
- ***Quantity***: Quantità di unità acquistate.
- ***Tax 5%***: Tassa applicata sull'acquisto.
- ***Total***: Costo totale comprensivo di tasse.
- ***Date***: Data della vendita.
- ***Time***: Orario della vendita.
- ***Payment***: Metodo di pagamento.
- ***cogs***: Costo del venduto.
- ***gross margin percentage***: Margine lordo come percentuale.
- ***gross income***: Guadagno lordo generato dalla vendita.
- ***Rating***: Valutazione del cliente per la vendita (numerico).

Questo dataset contiene informazioni dettagliate sulle vendite effettuate in un supermercato, utile per analisi di mercato, gestione delle vendite, o profilazione dei clienti.

Secondo Dataset Utilizzato

Il dataset (apple_quality.csv) contiene circa 4.000 righe e 9 colonne. Ecco una breve descrizione delle colonne:

- ***A_id***: ID unico di ogni mela (identificatore numerico).
- ***Size***: Misura della mela (numerico).
- ***Weight***: Peso della mela (numerico).
- ***Sweetness***: Livello di dolcezza (numerico).
- ***Crunchiness***: Livello di croccantezza (numerico).
- ***Juiciness***: Livello di succosità (numerico).
- ***Ripeness***: Livello di maturità (numerico).
- ***Acidity***: Livello di acidità (tipo oggetto, probabilmente valori numerici memorizzati come stringhe).
- ***Quality***: Classificazione della qualità della mela (categorico: es. "good" o "bad").

Questo dataset raccoglie dati dettagliati su diversi aspetti della qualità delle mele, probabilmente per un'analisi della qualità in un contesto di produzione o ricerca agricola.

Librerie Utilizzate

`from sklearn.model_selection import train_test_split`	Funzione per suddividere il dataset in training set e test set
`from sklearn.metrics import classification_report, mean_squared_error, confusion_matrix`	Funzioni per valutare le performance del modello di classificazione
`from matplotlib.ticker import FuncFormatter`	FuncFormatter per formattare gli assi dei grafici
`import matplotlib.pyplot as plt`	Matplotlib per la creazione di grafici
`import pandas as pd`	Pandas è utilizzato per la manipolazione e l'analisi dei dati
`import seaborn as sns`	Seaborn per la visualizzazione avanzata dei dati
`from sklearn.ensemble import RandomForestClassifier`	Classificatore Random Forest per problemi di classificazione
`from sklearn.linear_model import LinearRegression`	Modello di regressione lineare per problemi di regressione

Funzioni Utilizzate

```python
def thousand_separator(x):
    return f'{int(x):,}'.replace(',', '.')
```

Formatta il numero con il punto come separatore delle migliaia per l'output

Formatta il numero con il punto come separatore delle migliaia per i grafici

```python
def thousand_separator_for_plot(x, pos):
    return f'{int(x):,}'.replace(',', '.')
```

```python
def etichetta():
    for p in ax.patches:
        ax.annotate(f'{p.get_width():,.0f}'.replace(',', '.'),
                    (p.get_width(), p.get_y() + p.get_height() / 2),
                    ha='center', va='center',
                    color='black', fontsize=10, fontweight='bold')
```

Funzione per formattare l'etichetta dei grafici a barre

Vendite per Tipo di Cliente e Città

```
print("Vendite per città e tipo di cliente:")
for index, row in city_customer_sales.iterrows():
    print(f"{row['City']} ({row['Customer type']}): {thousand_separator(round(row['Total'], 0))}")
```

Stampa il nome della città, il tipo di cliente (Member o Normal) e il totale delle vendite formattato con separatore delle migliaia.

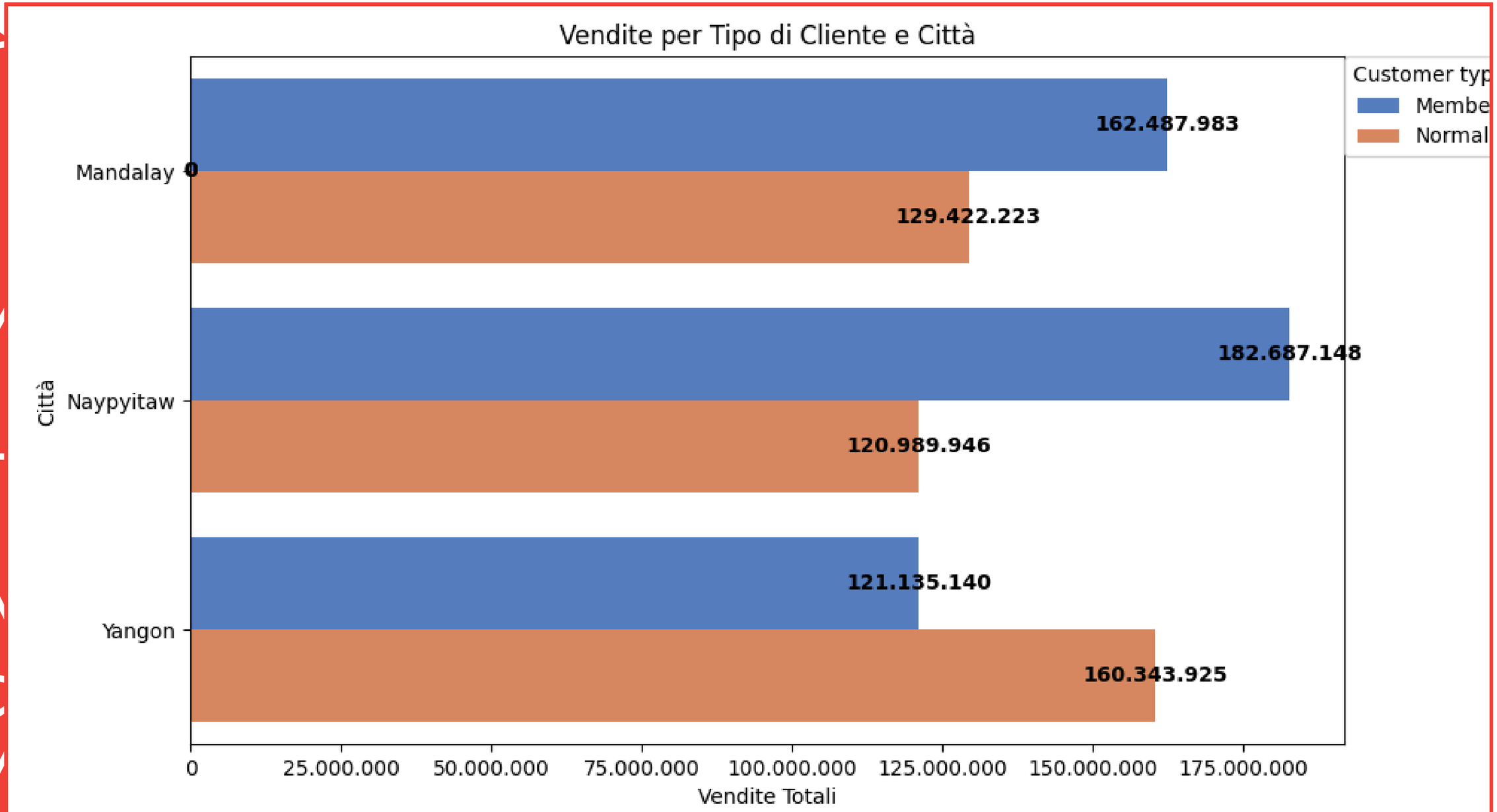
```
Vendite per città e tipo di cliente:
Mandalay (Member): 162.487.983
Mandalay (Normal): 129.422.223
Naypyitaw (Member): 182.687.148
Naypyitaw (Normal): 120.989.946
Yangon (Member): 121.135.140
Yangon (Normal): 160.343.925
```

*Mandalay mostra un totale di vendite di **162.487.983** per i membri, che è significativamente più alto rispetto alle vendite per i clienti normali, pari a **129.422.223**. Questo suggerisce che i clienti membri contribuiscono in modo più sostanziale alle vendite totali in questa città, il che potrebbe indicare l'efficacia di programmi di fidelizzazione o offerte specifiche per i membri.*

*Anche a Naypyitaw, le vendite per i membri (**182.687.148**) superano quelle per i normali clienti (**120.989.946**). Il divario tra le vendite dei membri e quelle dei normali clienti è evidente e potrebbe suggerire che le strategie di marketing e vendita siano particolarmente efficaci per attrarre membri in questa città.*

*A Yangon, la situazione è diversa. Le vendite per i clienti normali (**160.343.925**) superano quelle per i membri (**121.135.140**). Questo potrebbe indicare una base di clienti normali più forte o un interesse minore nei programmi di membership.*

Grafico delle Vendite per Tipo di Cliente e Città



Vendite Totali per Tipo di Cliente

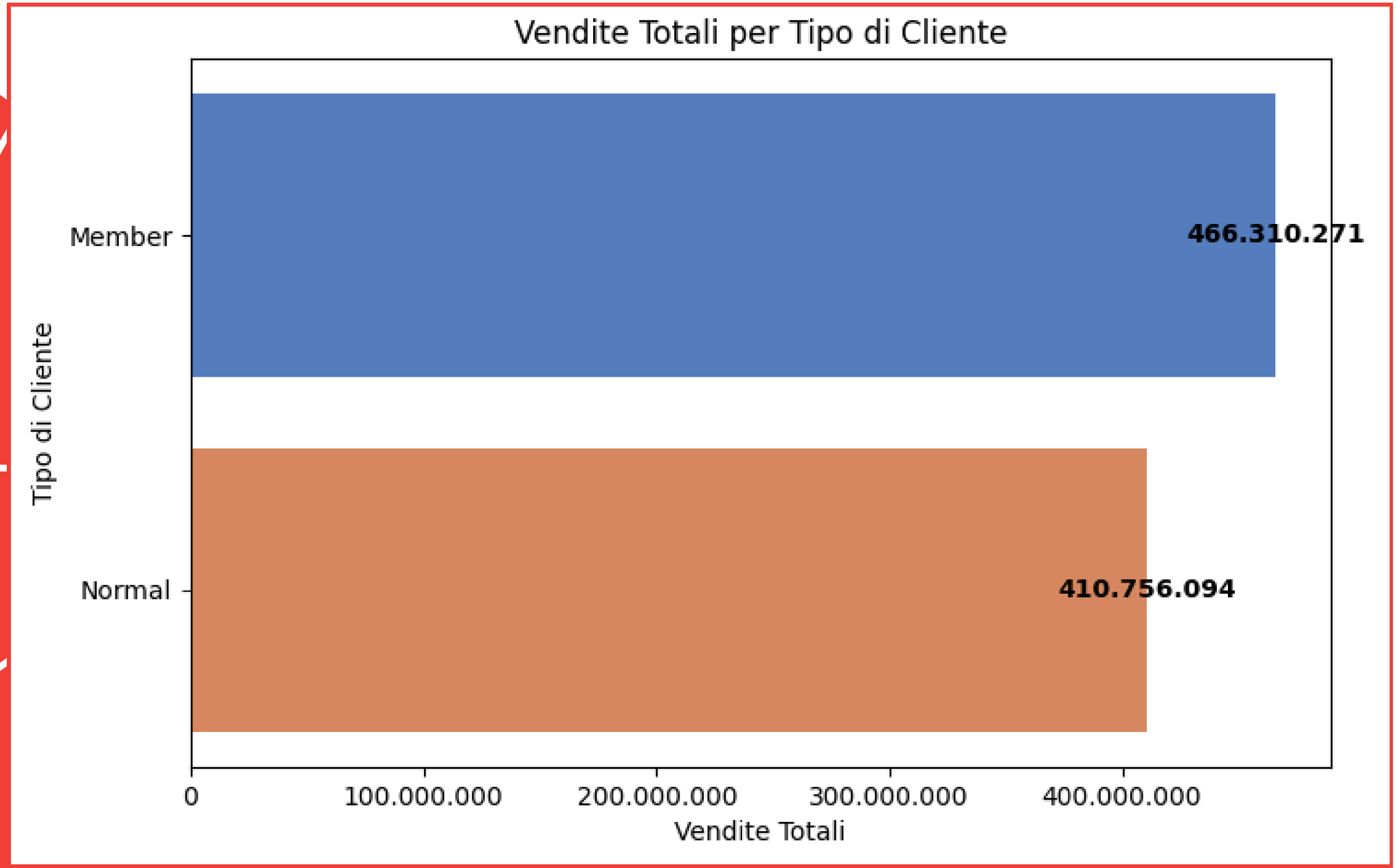
```
for index, row in customer_sales_total.iterrows():
    print(f"{row['Customer type']}: {thousand_separator(round(row['Total'], 0))}")
```

Stampa il tipo di cliente (Member o Normal) e il totale delle vendite formattato con separatore delle migliaia

```
Vendite totali per tipo di cliente:
Member: 466.310.271
Normal: 410.756.094
```

*Le vendite totali per tipo di cliente mostrano che i membri contribuiscono significativamente di più alle vendite complessive, con un totale di **466.310.271** contro **410.756.094** dei clienti normali. Questo è un dato positivo, in quanto suggerisce che l'implementazione di strategie di fidelizzazione ha portato a vendite maggiori. Tuttavia, è importante anche considerare la proporzione di clienti normali e trovare modi per convertirli in membri.*

Grafico delle Vendite Totali per Tipo di Cliente



Totale Vendite per Città

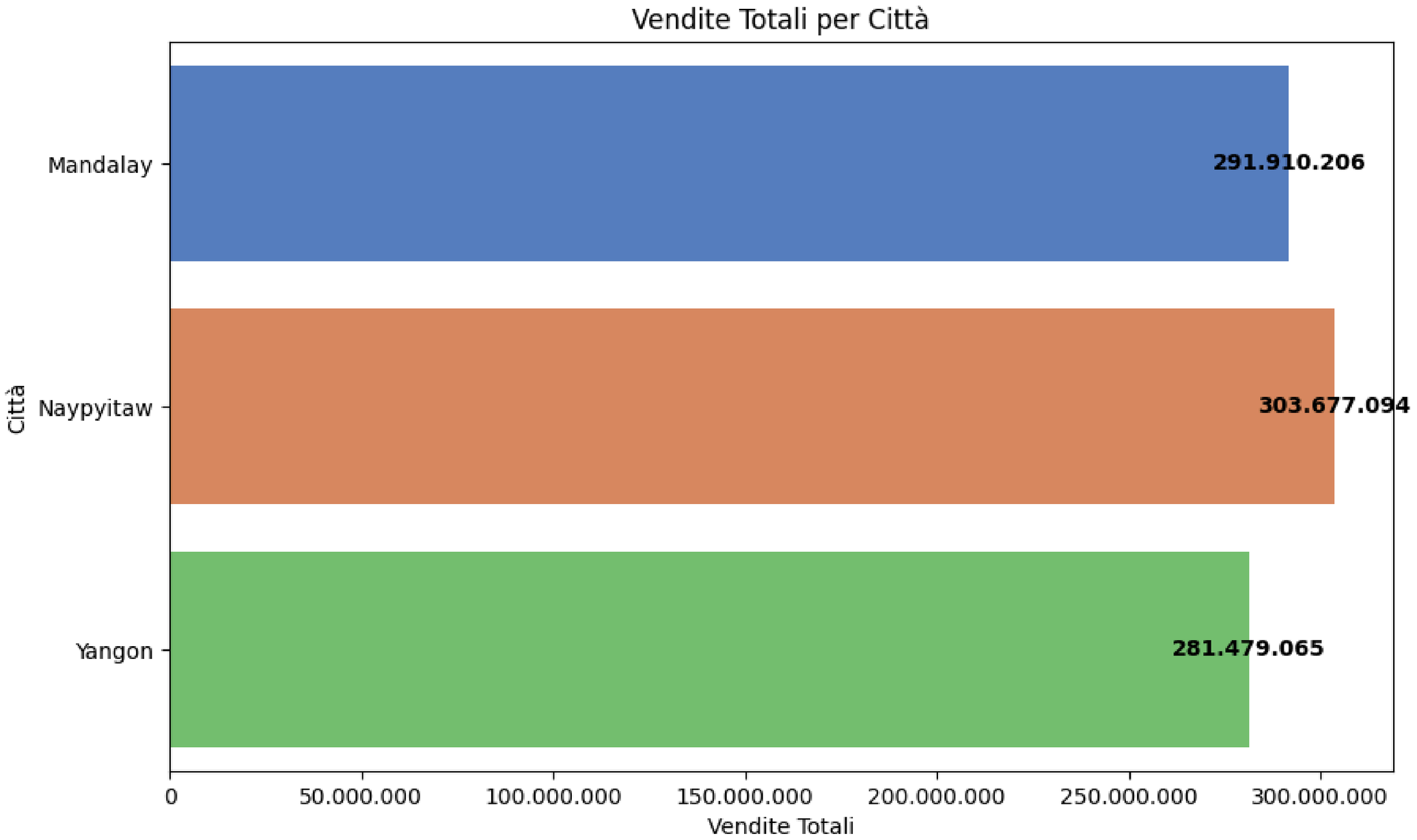
```
for index, row in city_total_sales.iterrows():
    print(f"{row['City']}: {thousand_separator(round(row['Total'], 0))}")
```

Stampa il nome della città e il totale delle vendite formattato con separatore delle migliaia.

```
TOTALE VENDITE PER CITTA'
Mandalay: 291.910.206
Naypyitaw: 303.677.094
Yangon: 281.479.065
```

*In termini di vendite totali per città, Naypyitaw ha il valore più alto con **303.677.094**, seguita da Mandalay (**291.910.206**) e Yangon (**281.479.065**). Questo potrebbe indicare che Naypyitaw sta performando meglio nel complesso, il che potrebbe essere dovuto a fattori come una migliore strategia di marketing, una maggiore popolazione di clienti o una combinazione di fattori favorevoli.*

Grafico sul Totale delle Vendite per Città



Vendite Totali per Categoria di Prodotto

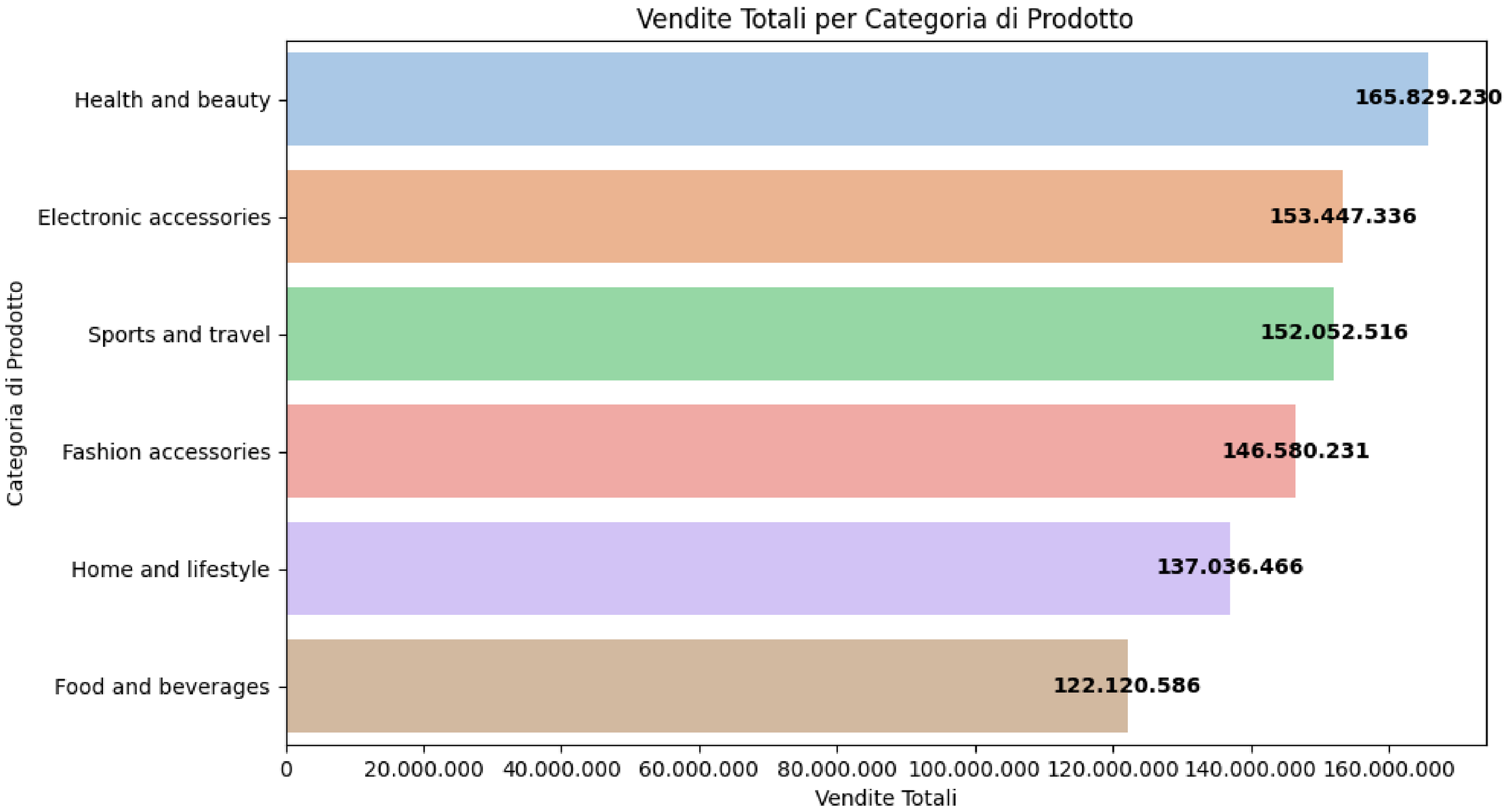
```
product_sales = sales_data.groupby('Product line')['Total'].sum().reset_index().sort_values(by='Total', ascending=False)
print("\nVendite per categoria di prodotto:")
for index, row in product_sales.iterrows():
    print(f"{row['Product line']}: {thousand_separator(round(row['Total'], 0))}")
```

```
Vendite per categoria di prodotto:
Health and beauty: 165.829.230
Electronic accessories: 153.447.336
Sports and travel: 152.052.516
Fashion accessories: 146.580.231
Home and lifestyle: 137.036.466
Food and beverages: 122.120.586
```

Per ogni categoria di prodotto, stampa il nome della categoria e il totale delle vendite, formattato, 'thousand_separator(row['Total'])' applica la formattazione numerica definita per separare le migliaia e arrotonda al numero intero più vicino.

*Dominanza della categoria "**Health and Beauty**":*
*La categoria Health and Beauty registra il fatturato più alto con **165.829.230**. Questo potrebbe riflettere un crescente interesse e domanda per prodotti legati alla salute e alla bellezza, un trend amplificato anche dall'attenzione verso il benessere personale. È importante notare che i prodotti per la salute e la bellezza spesso hanno margini di profitto più elevati, il che potrebbe contribuire alla loro elevata performance.*
*Solidità degli **"Electronic Accessories"**:*
*Gli Electronic Accessories seguono da vicino con **153.447.336**. Questa categoria è in continua espansione grazie alla crescente digitalizzazione e all'aumento dell'uso di dispositivi tecnologici. La domanda di accessori per smartphone, computer e altri dispositivi elettronici è costante. La popolarità di gadget e accessori tecnologici suggerisce anche che ci siano opportunità per cross-selling e upselling con prodotti complementari.*
*Interesse per **"Sports and Travel"**:*
*Con **152.052.516**, la categoria Sports and Travel è anch'essa significativa. L'aumento dell'interesse per attività all'aperto e sport, soprattutto post-pandemia, può contribuire a questa cifra. Potrebbe essere utile promuovere prodotti legati a stili di vita attivi e viaggi, magari in sinergia con il marketing di esperienze e avventure.*
***"Fashion Accessories" e "Home and Lifestyle"**:*
*Le vendite di Fashion Accessories e Home and Lifestyle sono rispettivamente **146.580.231 e 137.036.466**. Queste categorie riflettono tendenze culturali e stili di vita. I clienti sono sempre più interessati a prodotti che migliorano la loro qualità della vita e il loro stile. Promozioni strategiche o collaborazioni con influencer possono incrementare l'interesse e le vendite in queste categorie.*
*Crescita di **"Food and Beverages"**:*
*Infine, la categoria Food and Beverages ha registrato vendite di **122.120.586**. Anche se è la categoria con le vendite più basse rispetto alle altre, essa può rappresentare un mercato in crescita, soprattutto se si considera l'interesse crescente per cibi sani e bevande artigianali. Si potrebbe considerare di diversificare l'offerta in questa categoria, introducendo nuovi prodotti o varianti per attrarre una clientela più ampia.*

Grafico delle Vendite Totali per Categoria di Prodotto



Vendite per Categoria di Prodotto e Genere

```
for index, row in category_gender_sales.iterrows():
    print(f"{row['Product line']} ({row['Gender']}): {thousand_separator(round(row['Total'], 0))}")
```

Per ogni riga, stampa la categoria di prodotto e il genere con il totale delle vendite formattato utilizzando la funzione 'thousand_separator' per formattare i numeri.

Electronic Accessories:
Le vendite di accessori elettronici sono relativamente equilibrate tra i due generi, con una leggera preferenza per gli acquisti da parte degli uomini.

Fashion Accessories:
La categoria degli accessori di moda mostra una forte predominanza delle donne come acquirenti, suggerendo che le donne sono più propense a spendere in questa categoria rispetto agli uomini.

Food and Beverages:
Anche qui, le donne mostrano una maggiore propensione a spendere in questa categoria, il che potrebbe riflettere le responsabilità tradizionali di acquisto di cibo e bevande da parte delle donne in molte culture.

Health and Beauty:
Le vendite nella categoria salute e bellezza sono più elevate tra gli uomini, il che potrebbe indicare un crescente interesse maschile verso i prodotti di bellezza e benessere, in linea con le tendenze di mercato recenti.

Home and Lifestyle:
Le donne sembrano dominare anche in questa categoria, suggerendo un forte interesse per l'arredamento e il miglioramento della casa.

Sports and Travel:
Qui, gli uomini mostrano una spesa superiore, il che può riflettere un maggiore coinvolgimento maschile in attività sportive e viaggi.

```
Vendite per categoria di prodotto e genere:
Electronic accessories (Female): 75.291.447
Electronic accessories (Male): 78.155.889
Fashion accessories (Female): 93.240.966
Fashion accessories (Male): 53.339.265
Food and beverages (Female): 75.420.639
Food and beverages (Male): 46.699.947
Health and beauty (Female): 79.495.626
Health and beauty (Male): 86.333.604
Home and lifestyle (Female): 82.081.902
Home and lifestyle (Male): 54.954.564
Sports and travel (Female): 68.898.060
Sports and travel (Male): 83.154.456
```

Grafico Vendite per Città e Genere “Electronic Accessories”

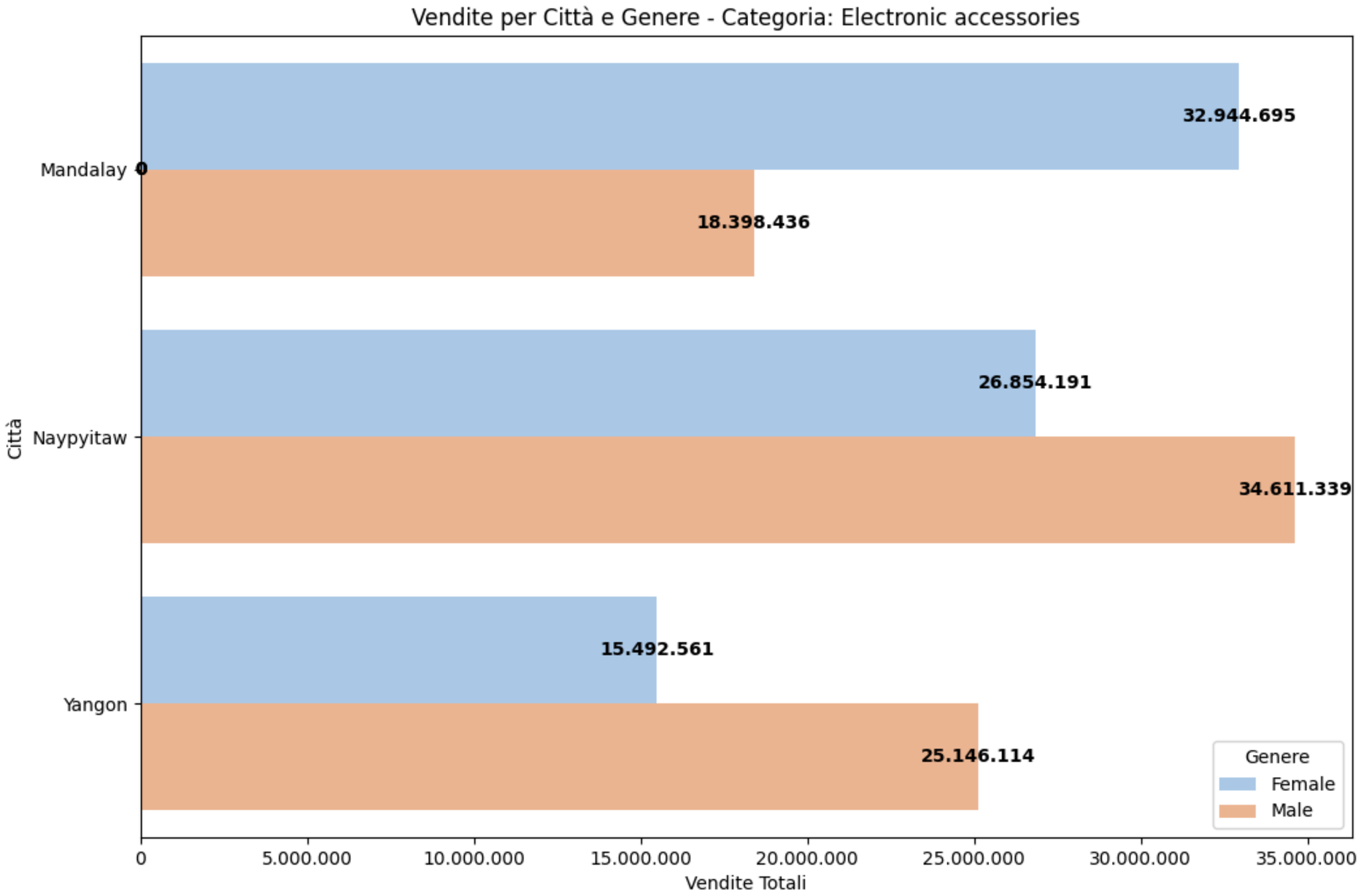


Grafico Vendite per Città e Genere “Fashion Accessories”

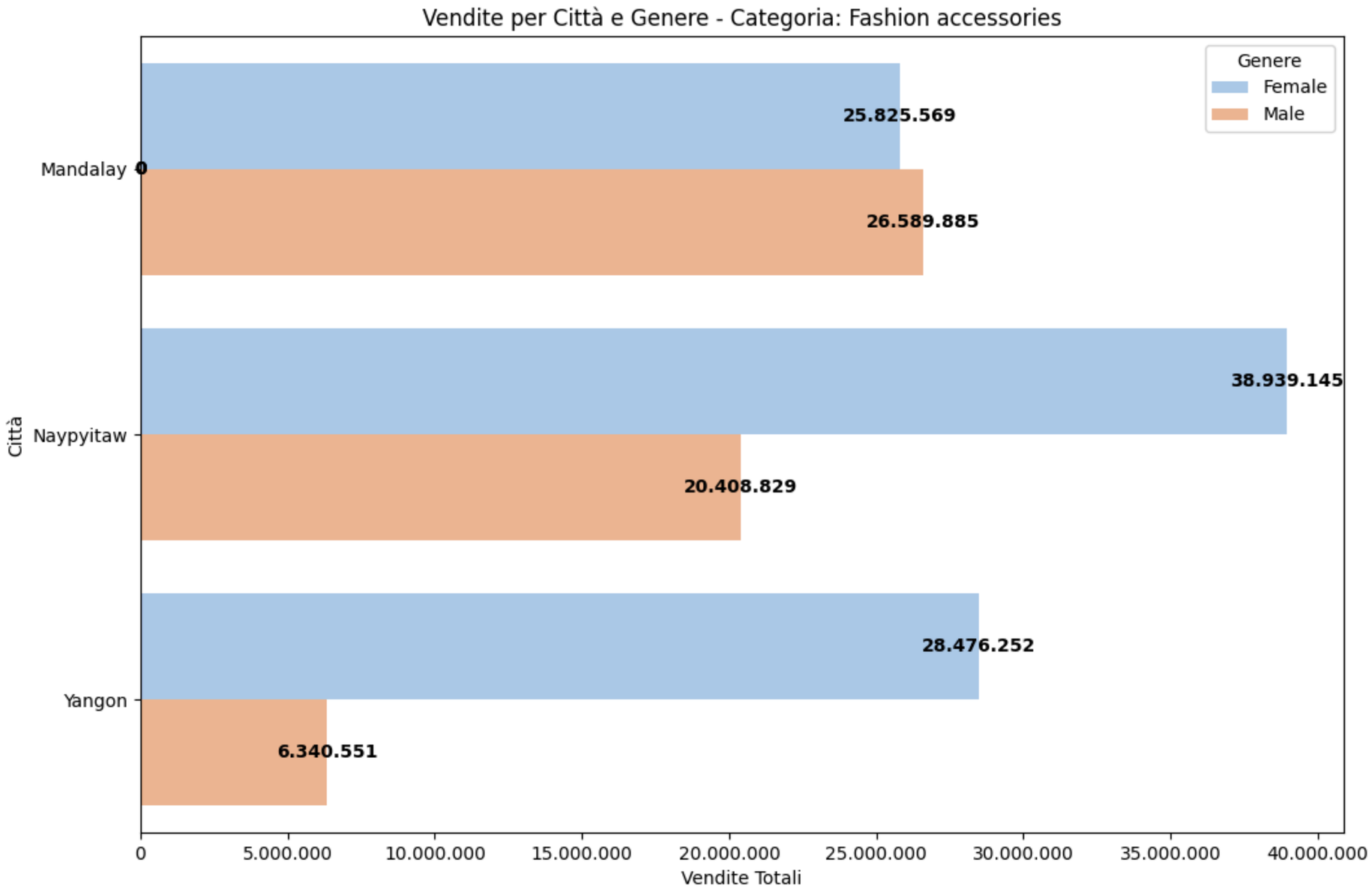


Grafico Vendite per Città e Genere “Food and Beverages”

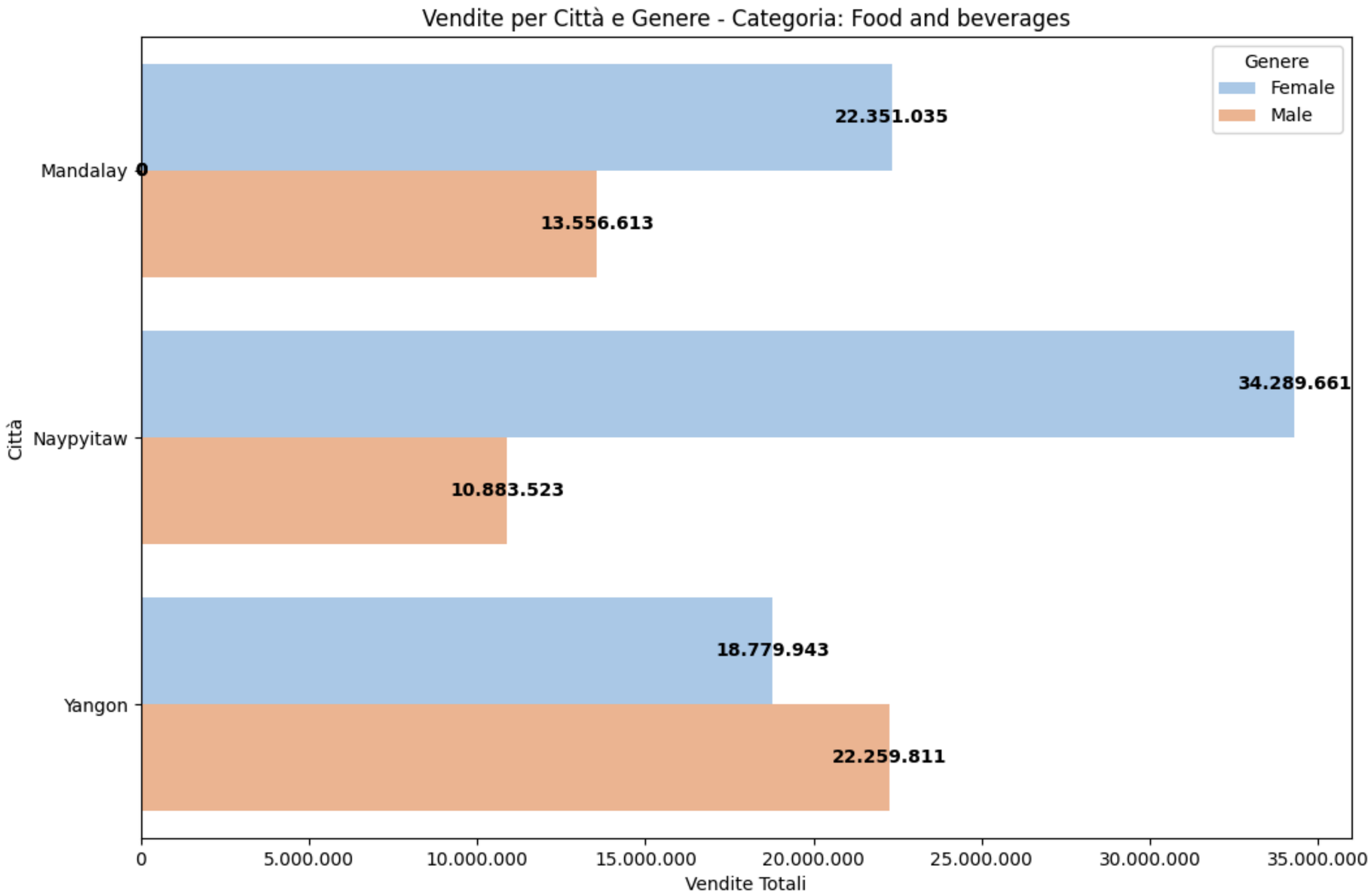


Grafico Vendite per Città e Genere “Health and Beauty”

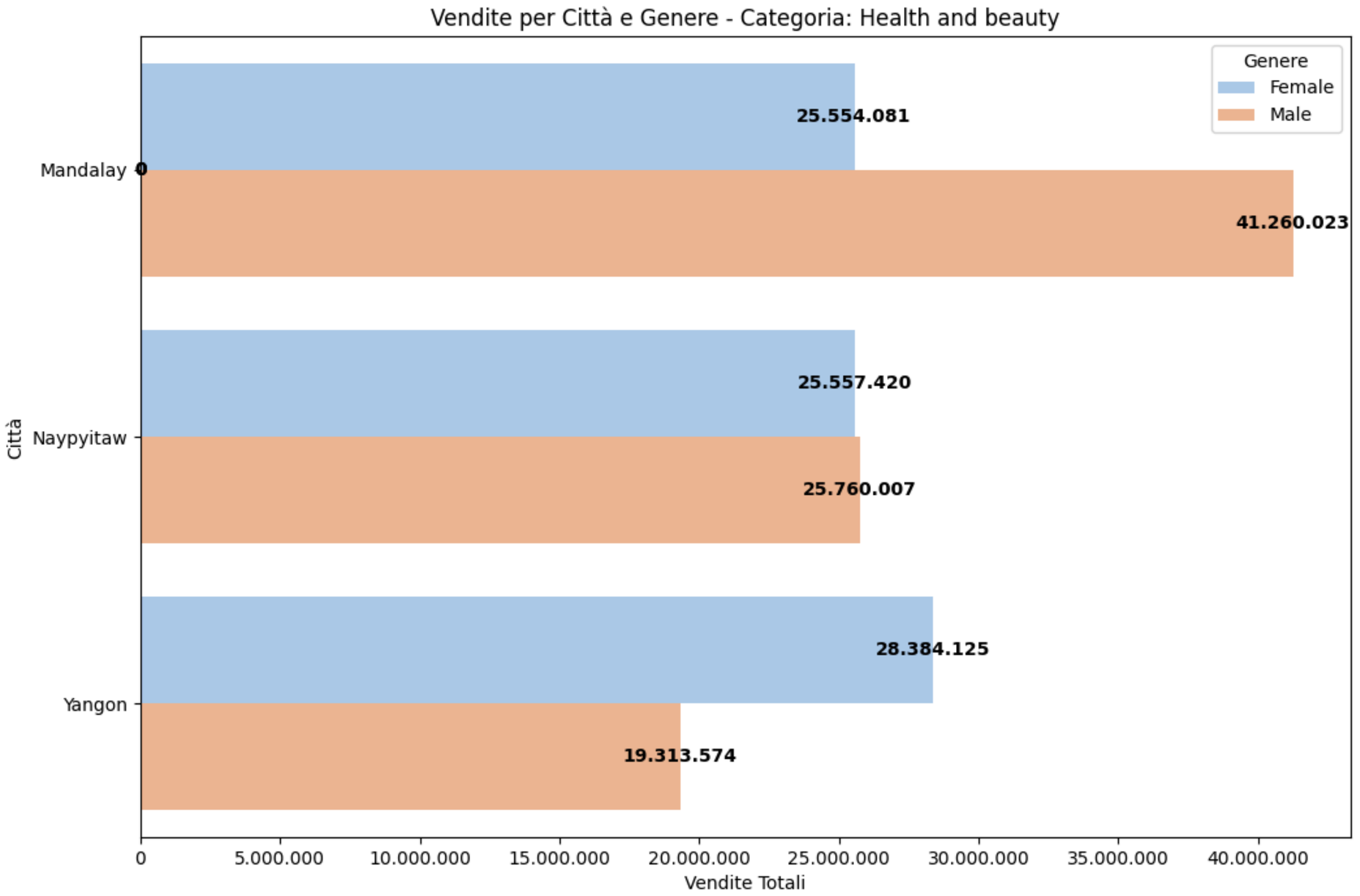


Grafico Vendite per Città e Genere “Home and Lifestyle”

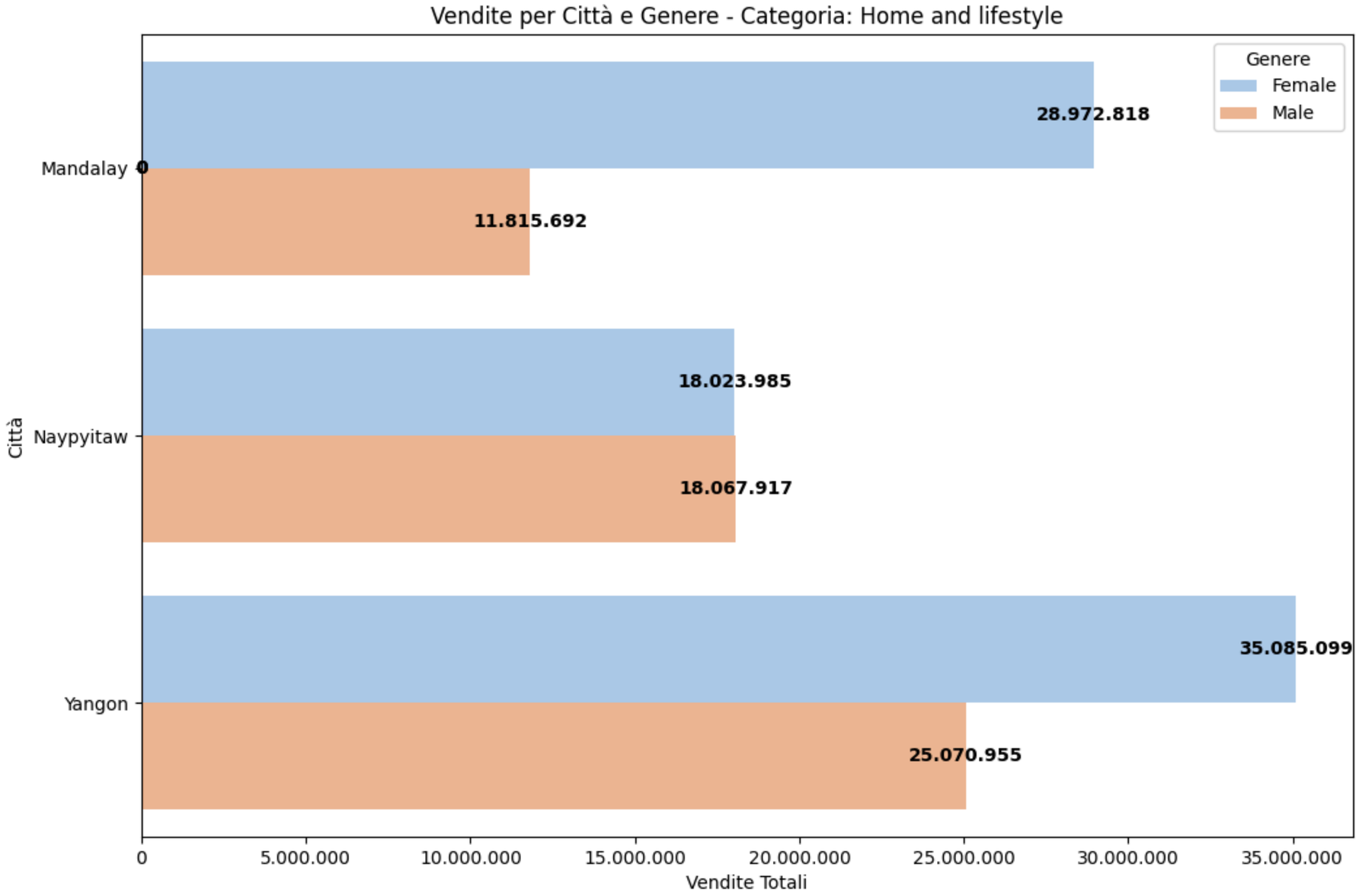
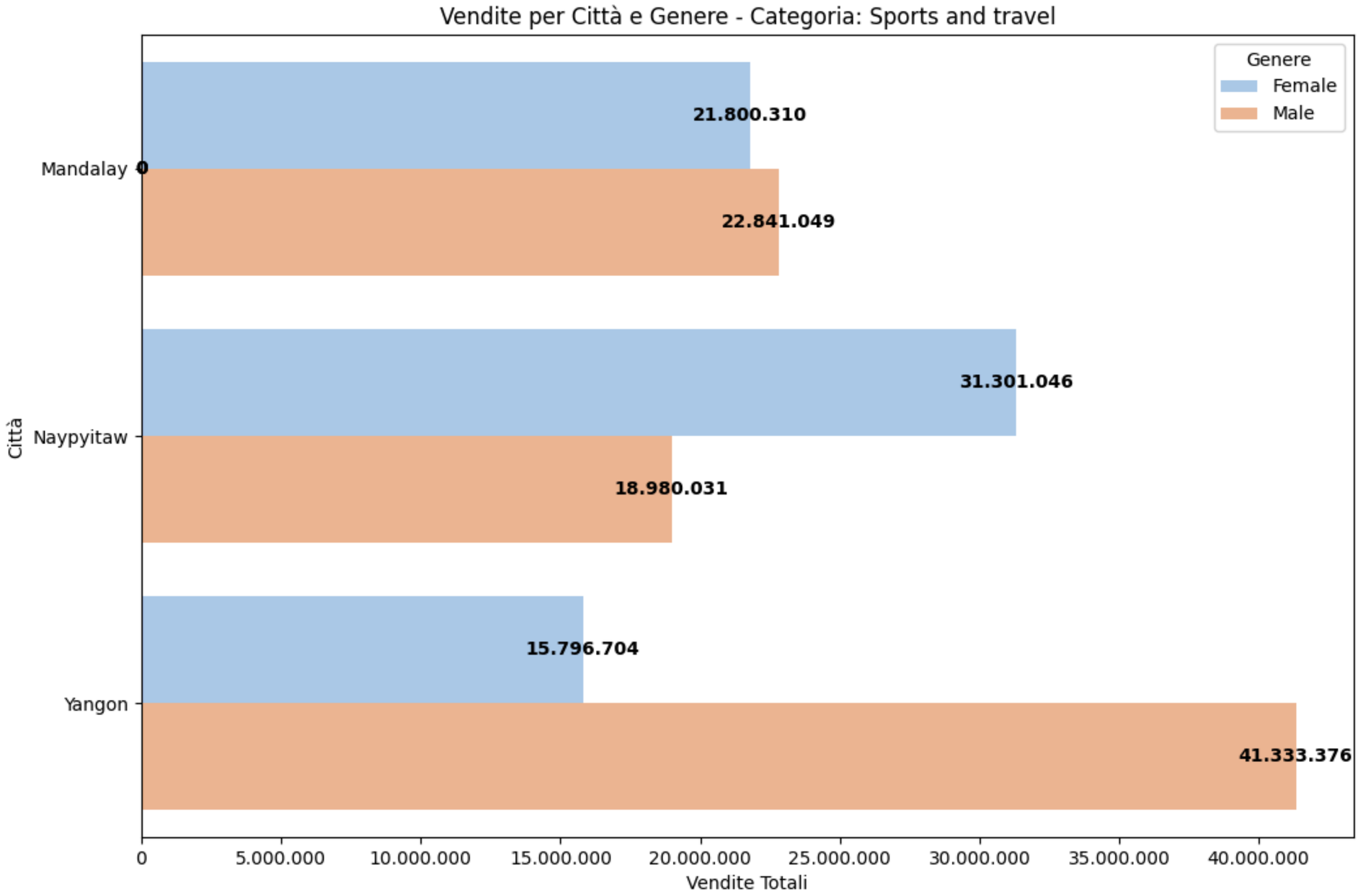


Grafico Vendite per Città e Genere “Sports and Travel”



Vendite Totali per Genere

```
for index, row in gender_sales.iterrows():
    print(f"{row['Gender']}: {thousand_separator(round(row['Total'], 0))}")
```

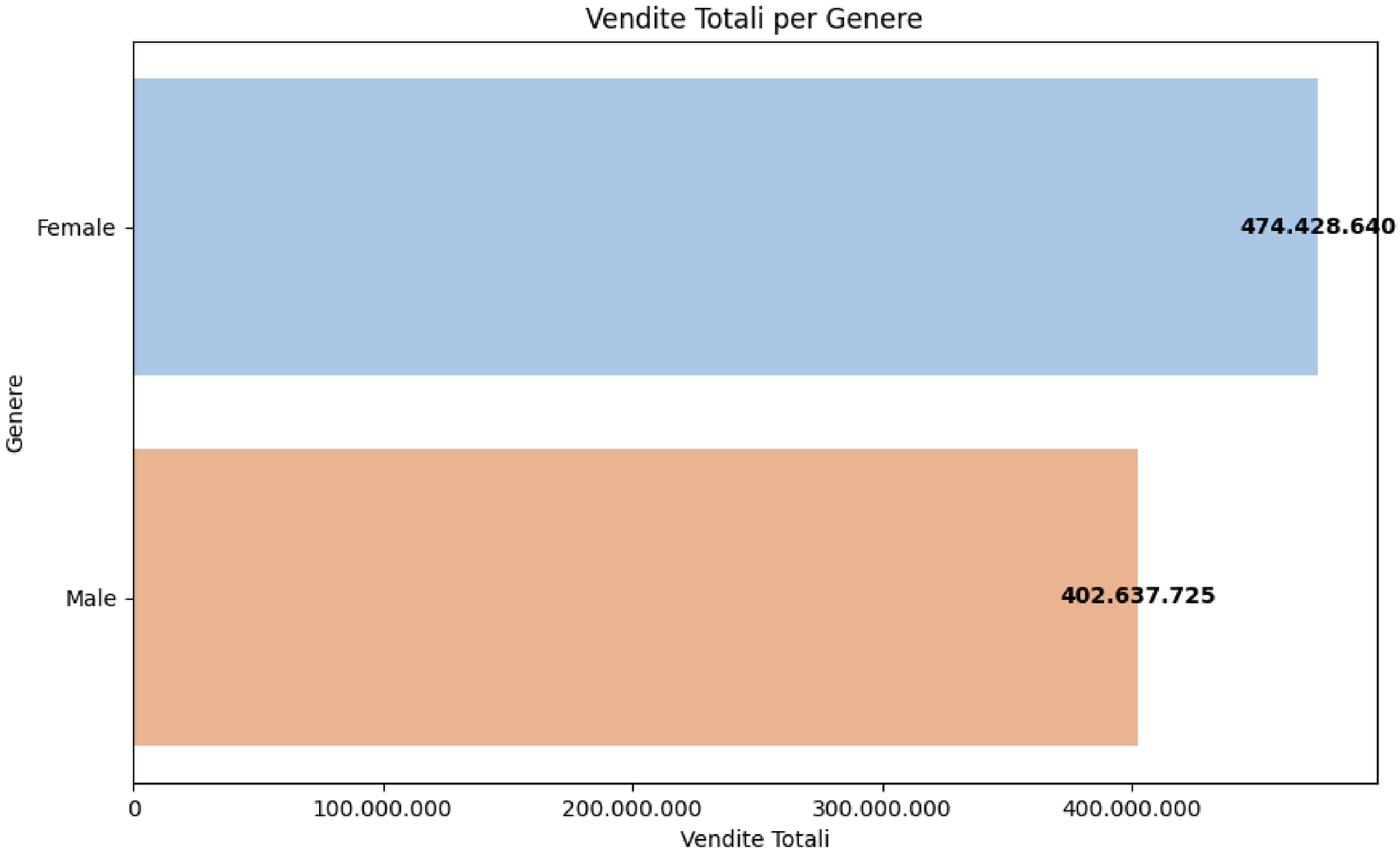
```
Vendite per genere:
Female: 474.428.640
Male: 402.637.725
```

Per ogni genere, stampa il nome del genere e il totale delle vendite formattato, 'thousand_separator(row['Total'])' applica la formattazione numerica definita per separare le migliaia e arrotonda al numero intero più vicino.

*Nel complesso, le vendite totali indicano che le **donne** hanno speso significativamente di più rispetto agli **uomini.***
Questo dato può suggerire che, sebbene gli uomini mostrino una spesa significativa in alcune categorie (come salute e bellezza, e sport e viaggi), le donne hanno una spesa più elevata in generale, in particolare nei settori della moda, cibo e articoli per la casa.
Considerazioni Finali

- *Preferenze di Acquisto: I dati suggeriscono che le donne tendono a investire di più in moda, cibo e lifestyle, mentre gli uomini sembrano più interessati ai prodotti di bellezza e agli sport. Ciò riflette le tendenze sociali e culturali attuali.*
- *Crescita nel Settore Maschile: La crescita delle vendite in categorie come salute e bellezza per gli uomini potrebbe rappresentare una tendenza emergente che potrebbe essere ulteriormente esplorata per potenziali opportunità di mercato.*

Grafico Vendite Totali per Genere



Vendite per Città, Genere e Categoria di prodotto

```python
city_gender_category_sales = sales_data.groupby(['City', 'Gender', 'Product line'])['Total'].sum().reset_index()
print("Vendite per città, genere e categoria di prodotto:")

for index, row in city_gender_category_sales.iterrows():
    print(f"{row['City']} ({row['Gender']}, {row['Product line']}): {int(row['Total']):,.0f}".replace(',', '.'))
```

Stampa il nome della città, il genere, la categoria di prodotto e il totale delle vendite formattato con il separatore delle migliaia (.) in formato leggibile.

```
Vendite per città, genere e categoria di prodotto:
Mandalay (Female. Electronic accessories): 32.944.695
Mandalay (Female. Fashion accessories): 25.825.569
Mandalay (Female. Food and beverages): 22.351.035
Mandalay (Female. Health and beauty): 25.554.081
Mandalay (Female. Home and lifestyle): 28.972.818
Mandalay (Female. Sports and travel): 21.800.310
Mandalay (Male. Electronic accessories): 18.398.436
Mandalay (Male. Fashion accessories): 26.589.885
Mandalay (Male. Food and beverages): 13.556.613
Mandalay (Male. Health and beauty): 41.260.023
Mandalay (Male. Home and lifestyle): 11.815.692
Mandalay (Male. Sports and travel): 22.841.049
Naypyitaw (Female. Electronic accessories): 26.854.191
Naypyitaw (Female. Fashion accessories): 38.939.145
Naypyitaw (Female. Food and beverages): 34.289.661
Naypyitaw (Female. Health and beauty): 25.557.420
Naypyitaw (Female. Home and lifestyle): 18.023.985
Naypyitaw (Female. Sports and travel): 31.301.046
Naypyitaw (Male. Electronic accessories): 34.611.339
Naypyitaw (Male. Fashion accessories): 20.408.829
Naypyitaw (Male. Food and beverages): 10.883.523
Naypyitaw (Male. Health and beauty): 25.760.007
Naypyitaw (Male. Home and lifestyle): 18.067.917
Naypyitaw (Male. Sports and travel): 18.980.031
Yangon (Female. Electronic accessories): 15.492.561
Yangon (Female. Fashion accessories): 28.476.252
Yangon (Female. Food and beverages): 18.779.943
Yangon (Female. Health and beauty): 28.384.125
Yangon (Female. Home and lifestyle): 35.085.099
Yangon (Female. Sports and travel): 15.796.704
Yangon (Male. Electronic accessories): 25.146.114
Yangon (Male. Fashion accessories): 6.340.551
Yangon (Male. Food and beverages): 22.259.811
Yangon (Male. Health and beauty): 19.313.574
Yangon (Male. Home and lifestyle): 25.070.955
Yangon (Male. Sports and travel): 41.333.376
```

Differenze di Genere e Città

- ***Mandalay***: *Gli uomini tendono a investire di più in salute e bellezza, mentre le donne dominano in elettronica. Si suggerisce di creare campagne di marketing specifiche per attrarre entrambi i generi nelle loro aree di interesse.*
- ***Naypyitaw***: *La predominanza femminile in fashion e food presenta opportunità per l'espansione delle linee di prodotto. Le aziende dovrebbero considerare collaborazioni con influencer locali per promuovere i loro prodotti.*
- ***Yangon***: *Le donne sono i principali consumatori in home and lifestyle e fashion, quindi un potenziamento dell'e-commerce e delle promozioni sui social media potrebbe rivelarsi fruttuoso.*

Tendenze di Consumo

- *C'è un chiaro spostamento verso la cura personale tra gli uomini, con vendite significative in Health and Beauty, suggerendo opportunità per prodotti mirati.*
- *Le donne mostrano interesse per prodotti tecnologici, il che potrebbe indicare una crescente partecipazione nel mercato tecnologico.*

Strategie di Marketing

- *Le campagne pubblicitarie dovrebbero essere personalizzate per ciascuna città, tenendo conto delle preferenze di genere e delle categorie di prodotto.*
- *Considerare eventi o pop-up store per attirare i consumatori in modo diretto, soprattutto in città come Naypyitaw e Yangon.*

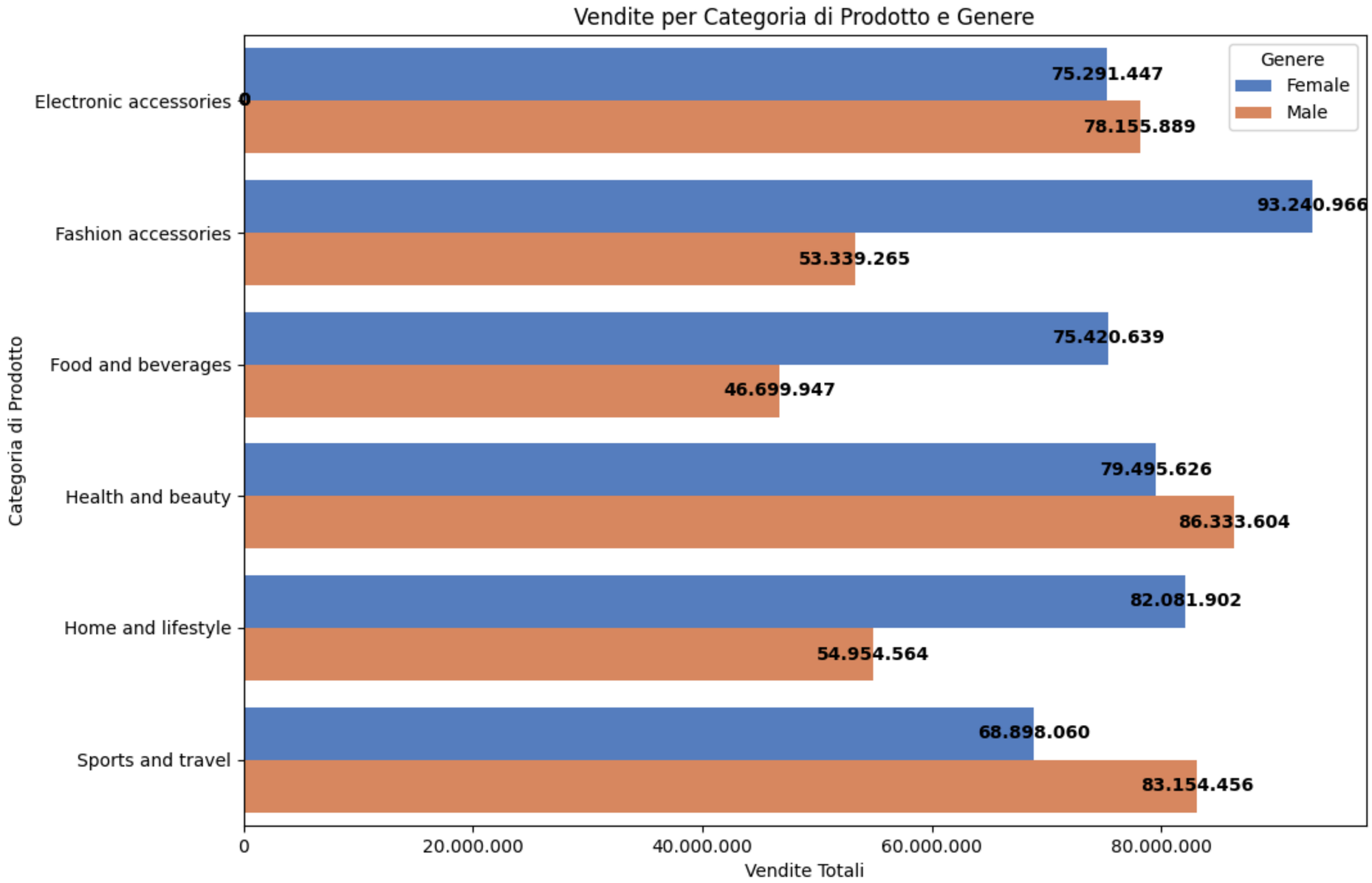
Innovazione dei Prodotti

- *A seconda delle preferenze di acquisto, le aziende potrebbero voler introdurre prodotti innovativi che combinano diverse categorie per attrarre un pubblico più ampio.*

Monitoraggio e Adattamento

- *Monitorare costantemente le tendenze di acquisto e adattare le strategie di prodotto e marketing in base ai dati emergenti per rimanere competitivi nel mercato.*

Grafico Vendite Genere e Categoria di prodotto



Classificazione della qualità delle mele

```python
apple_data['Acidity'] = pd.to_numeric(apple_data['Acidity'], errors='coerce')  # Converte la colonna 'Acidity' in valori numerici; se ci sono errori, sostituiamo con NaN.
apple_data = apple_data.dropna()  # Questo elimina tutte le righe del DataFrame che contengono valori NaN in qualsiasi colonna.
X = apple_data.drop('Quality', axis=1)  # Crea il DataFrame X con tutte le colonne tranne 'Quality', che sono le nostre caratteristiche predittive.
y = apple_data['Quality']  # Crea il vettore y che contiene solo la colonna 'Quality', che è la nostra variabile target da prevedere.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)  # Utilizza 'train_test_split' per dividere X e y in un training set e un test set, 'test_size=0.3' significa che il 30% dei dati sarà usato per il test e il 70% per l'addestramento, 'random_state=42' assicura che la divisione sia riproducibile.
rf = RandomForestClassifier(n_estimators=100, random_state=42)  # Crea un'istanza del modello RandomForestClassifier con 100 alberi e un seme random.
rf.fit(X_train, y_train)  # Allena il modello utilizzando il training set.
y_pred = rf.predict(X_test)  # Utilizza il modello addestrato per fare previsioni sui dati del test set.
```

Classificazione della qualità delle mele:

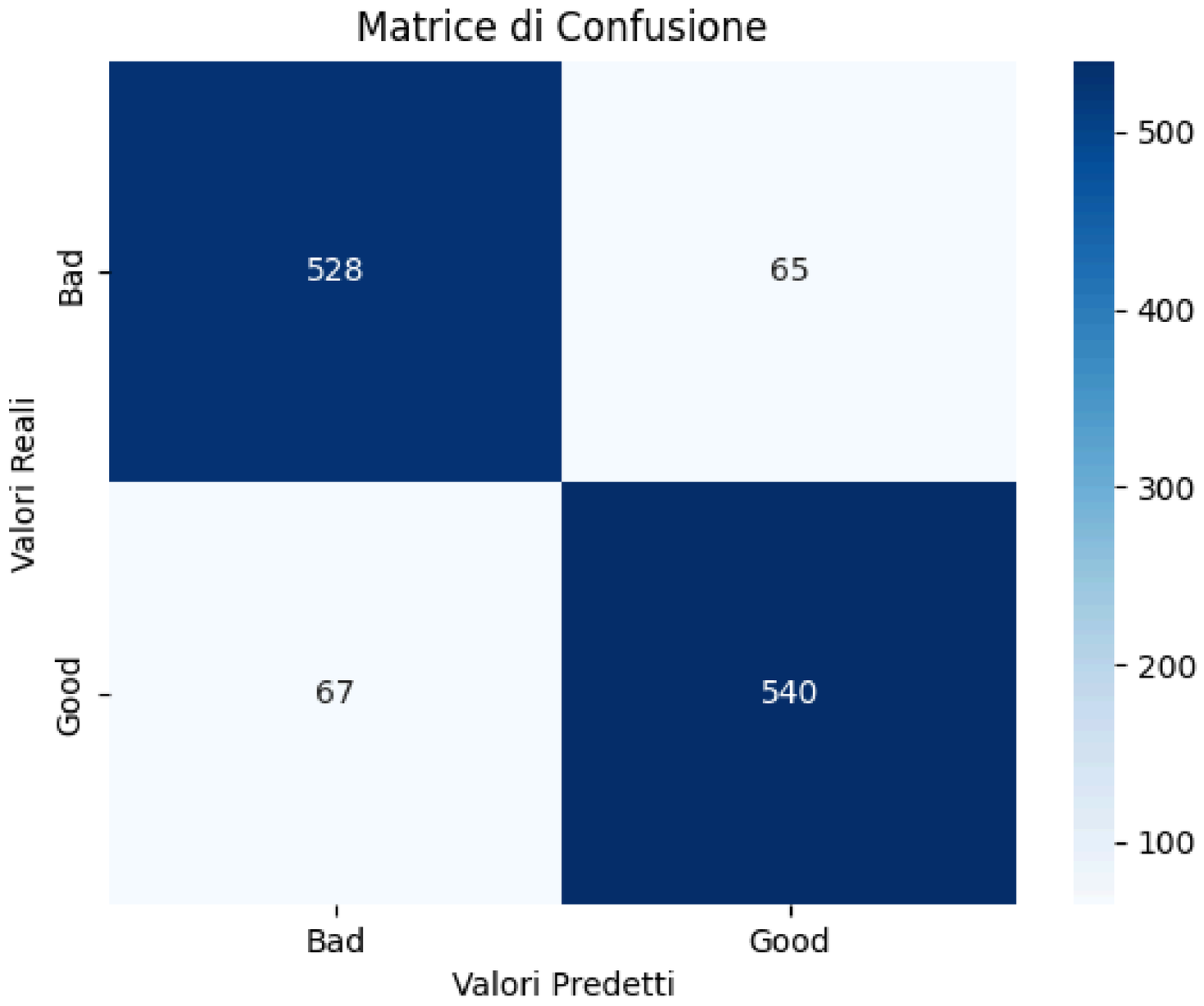
	precision	recall	f1-score	support
bad	0.89	0.89	0.89	593
good	0.89	0.89	0.89	607
accuracy			0.89	1200
macro avg	0.89	0.89	0.89	1200
weighted avg	0.89	0.89	0.89	1200

Il report di classificazione presenta informazioni dettagliate sulle prestazioni del modello:

- *La precisione indica la percentuale di classificazioni corrette tra le previste. Un valore di **0.89** suggerisce che il modello ha una buona precisione sia nella classificazione delle mele buone che cattive.*
- *Il richiamo misura la capacità del modello di identificare correttamente le classi positive. Anche in questo caso, un valore di **0.89** è rassicurante, poiché significa che il modello è in grado di identificare la maggior parte delle mele buone e cattive.*
- *L'F1-score è la media armonica tra precisione e richiamo, fornendo una misura complessiva dell'efficacia del modello. Valori intorno a **0.89** indicano un buon equilibrio tra precisione e recall.*
- *Questo valore rappresenta il numero di istanze reali per ciascuna classe. È utile per comprendere il bilanciamento dei dati.*
- *L'accuratezza del modello, pari a **0.89**, indica che il 89% delle classificazioni effettuate dal modello sono corrette. Questo è un ottimo risultato e suggerisce che il modello è robusto.*
- *Questi valori indicano che, anche se ci sono piccole differenze nel numero di campioni per ciascuna classe, il modello mantiene prestazioni coerenti attraverso entrambe le classi.*

In sintesi, il modello di classificazione ha dimostrato di avere prestazioni complessive buone e bilanciate. L'accuratezza del 89%, insieme ai valori di precisione, richiamo e F1-score, indica che il modello è efficace nel distinguere tra mele buone e cattive. Tuttavia, è fondamentale monitorare e migliorare le aree evidenziate, come i falsi positivi e i falsi negativi, per ottimizzare ulteriormente le prestazioni del modello. L'implementazione di tecniche avanzate di apprendimento automatico potrebbe contribuire a perfezionare il modello, garantendo una classificazione più accurata e riducendo al minimo gli errori.

Classificazione della qualità delle mele



La matrice di confusione è uno strumento fondamentale per valutare le prestazioni di un modello di classificazione.

- ***True Negatives (TN) = 528***
  - *Il modello ha classificato correttamente **528** mele come cattive. Questo è un segnale positivo, indicando che il modello è in grado di riconoscere le mele di bassa qualità con un buon livello di affidabilità.*

- ***False Positives (FP) = 65***
  - *Qui, il modello ha erroneamente classificato **65** mele cattive come buone. Sebbene questo numero non sia eccessivamente alto, rappresenta un'area di miglioramento, poiché mele di bassa qualità potrebbero essere vendute come buone, il che potrebbe portare a insoddisfazione dei clienti.*

- ***False Negatives (FN) = 67***
  - *Questo valore indica che il modello ha erroneamente classificato **67** mele buone come cattive. Ciò potrebbe avere un impatto negativo sulla produttività, poiché mele di buona qualità vengono scartate.*

- ***True Positives (TP) = 540***
  - *Il modello ha classificato correttamente **540** mele come buone. Questo risultato dimostra che il modello è capace di identificare con successo le mele di alta qualità.*

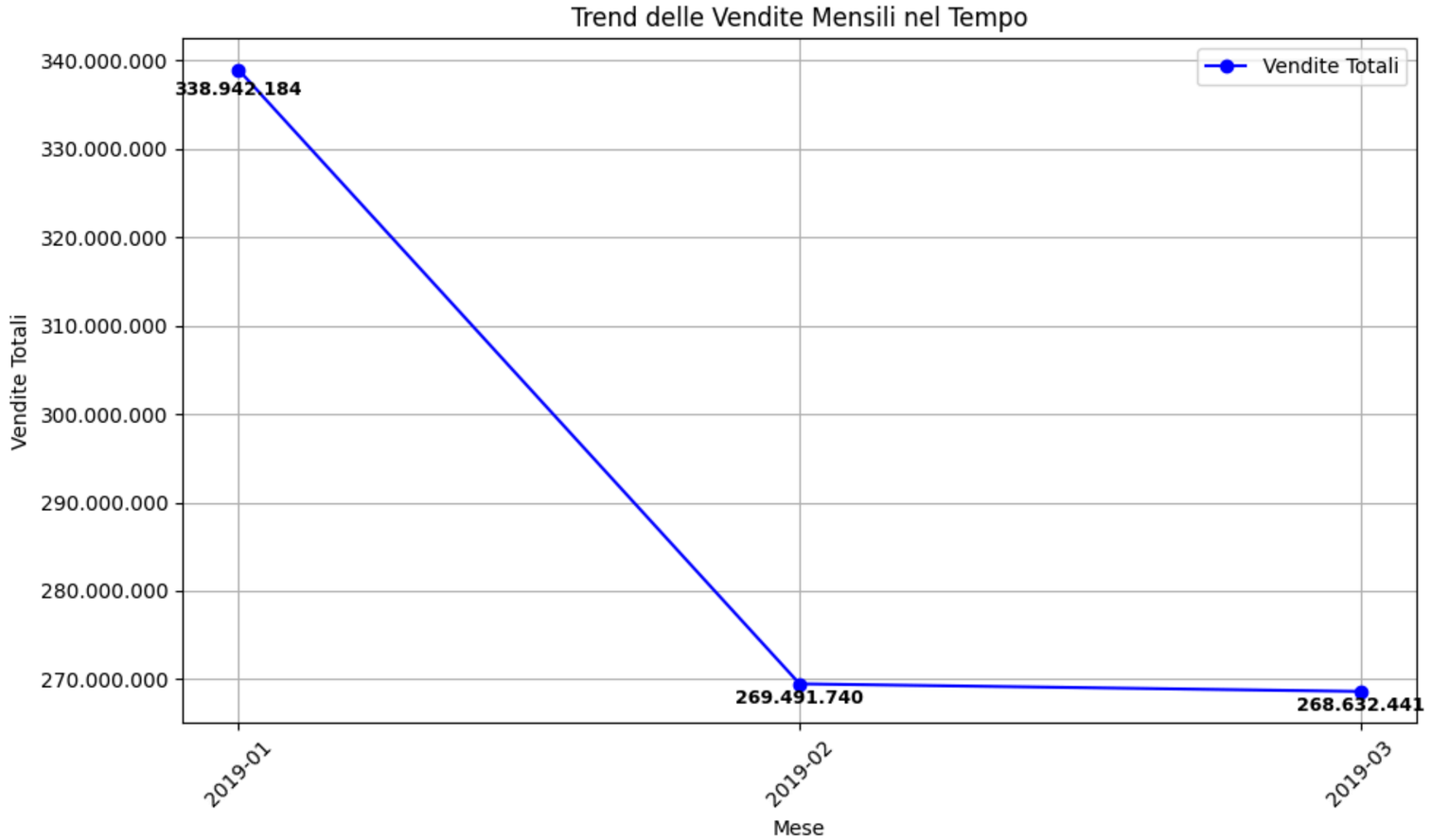
Mean Squared Error

```python
# Convertiamo la colonna Date in formato datetime e creiamo una nuova colonna "Month" per aggregare per mese
sales_data['Date'] = pd.to_datetime(sales_data['Date'])                                  # Converte la colonna 'Date' in formato datetime
sales_data['Month'] = sales_data['Date'].dt.to_period('M')                               # Estrae il mese e l'anno dalla data e lo memorizza in una nuova colonna 'Month'
# Raggruppiamo i dati per mese e sommiamo i profitti
monthly_sales = sales_data.groupby('Month')['Total'].sum().reset_index()                 # Raggruppa i dati nel DataFrame 'sales_data' per la colonna 'Month', calcolando la somma delle vendite totali
                                                                                         # ('Total') per ciascun mese e il risultato viene poi reimpostato come un nuovo DataFrame con l'indice ripristinato.
# Trasformiamo la colonna Month in numerica per la regressione
monthly_sales['Month_numeric'] = monthly_sales['Month'].astype(str).str.replace('-', '').astype(int)      # Converte i periodi mensili dalla forma 'YYYY-MM' (stringa) a un formato numerico intero e sostituisce il
                                                                                                          # trattino '-' con una stringa vuota e poi converte il risultato in un numero intero. Questo passaggio è
                                                                                                          # utile per l'analisi di regressione, poiché la variabile indipendente deve essere numerica.
# Creiamo il modello di regressione
X = monthly_sales[['Month_numeric']]                                                     # Seleziona la colonna 'Month_numeric' come feature (variabile indipendente) per il modello di regressione.
y = monthly_sales['Total']                                                               # Seleziona la colonna 'Total' come target (variabile dipendente) che vogliamo prevedere.
# Dividiamo il dataset per il training e il test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)  # Utilizza la funzione 'train_test_split' per dividere i dati in due set: uno per l'addestramento (80%) e uno
                                                                                           # per il test (20%).'random_state=42' assicura che la divisione sia riproducibile, fornendo sempre lo
                                                                                           # stesso set di dati per il training e il test.
# Inizializziamo il modello di regressione lineare
reg = LinearRegression()                                                                 # Crea un'istanza del modello di regressione lineare.
reg.fit(X_train, y_train)                                                                # Addestra il modello sui dati di addestramento, apprendendo la relazione tra 'Month_numeric' e 'Total'.
# Facciamo previsioni sui dati di test
y_pred = reg.predict(X_test)                                                             # Utilizza il modello di regressione addestrato (reg) per fare previsioni sui dati di test (X_test), producendo
                                                                                         # i valori previsti di vendita ('y_pred') per ciascun mese nel test set.
# Calcolo dell'errore quadratico medio per valutare la precisione del modello di regressione
mse = mean_squared_error(y_test, y_pred)                                 # Calcola l'errore quadratico medio (MSE), una metrica che misura quanto le previsioni differiscono dai valori
                                                                         # reali. Confronta le previsioni 'y_pred' con i dati reali 'y_test'. Più basso è il valore, migliore è la precisione del modello.
print(f"\nMean Squared Error: {thousand_separator(mse)}")                # Stampa il valore dell'errore quadratico medio formattato con il separatore delle migliaia, 'thousand_separator' applica il
                                                                         # formato per la visualizzazione.
```

```
Mean Squared Error: 4.704.745.172.411.025
```

L'elevato valore di errore quadratico medio (MSE), può essere attribuito principalmente al fatto che i dati disponibili coprono solo un periodo di tre mesi. Questo limitato intervallo temporale non consente al modello di catturare in modo efficace eventuali trend stagionali, ciclici o variazioni nel comportamento delle vendite su periodi più lunghi.

Trend delle Vendite Mensili nel Tempo



Osservazioni:

Calante dal primo al secondo mese:

- *Dopo un forte inizio nel mese di gennaio, con vendite totali di circa **339 milioni**, si nota un calo significativo a febbraio, con vendite scese a circa **269 milioni**.*

Stabilità tra febbraio e marzo:

- *Dopo il calo tra gennaio e febbraio, le vendite si stabilizzano tra febbraio e marzo, con una leggera diminuzione di circa 860.000 unità, suggerendo che la flessione delle vendite potrebbe essersi arrestata.*

Trend complessivo:

- *Complessivamente, il trend mostra un calo nelle vendite dal primo trimestre del 2019. Questo potrebbe essere dovuto a diversi fattori, come stagionalità, promozioni più forti all'inizio dell'anno o cicli economici del mercato.*

Spiegazioni possibili:

- *Effetto stagionale:*
  - *È possibile che gennaio abbia beneficiato di eventi stagionali, come vendite post-festive, saldi o promozioni. I mesi successivi potrebbero non avere avuto lo stesso tipo di impulso.*

- *Cambiamento nel comportamento dei consumatori:*
  - *La flessione delle vendite potrebbe anche riflettere un cambiamento nei modelli di spesa dei clienti, che si stabilizzano dopo una spinta iniziale.*

GitHub

Clicca qui per visualizzare il Codice