Politechnika Warszawska

WYDZIAŁ ELEKTRONIKI
I TECHNIK INFORMACYJNYCH

Instytut Informatyki

Studia Podyplomowe
Wizualna analityka danych

PRACA KOŃCOWA

Sebastian Gumula

Amazon w Liczbach: Analiza Trendów Sprzedaży i Profilu Klienta w Branży E-commerce

Opiekun pracy
Tytuł, imię i nazwisko
opiekuna

Warszawa, 2024

STRESZCZENIE

Praca ta koncentruje się na wszechstronnej analizie danych w kontekście branży e-commerce, ze szczególnym uwzględnieniem klientów, popytu na konkretne typy produktów, analizy sentymentu, dekompozycji szeregów czasowych oraz tworzenia na jej podstawie modelu prognostycznego ARIMA.
Dogłębna analiza zachowań klientów oraz trendów sprzedażowych, a także badanie popytu na konkretne produkty umożliwiają lepsze zrozumienie preferencji konsumentów.
Analiza sentymentu pozwala na ocenę opinii klientów, podczas gdy dekompozycja szeregów czasowych dostarcza wglądu w sezonowe i trendowe zmiany występujące w danych.
Wreszcie, model prognostyczny umożliwia prognozowanie przyszłych wyników sprzedaży, co jest kluczowe dla opracowywania skutecznych strategii biznesowych.
Praca ta skupia się na kompleksowym podejściu do analizy danych e-commerce, kładąc nacisk na zrozumienie klientów, dynamikę popytu oraz efektywne prognozowanie przyszłych trendów.

Słowa kluczowe: Analiza Danych, Analiza Sentymentu, Prognozowanie, ARIMA

THESIS TITLE IN ENGLISH

Amazon in Numbers: Analyzing Sales Trends and Customer Profile in the E-commerce Industry.

Summary in English.

This thesis focuses on comprehensive data analysis in context of e-commerce industry, with a particular focus on customers, demand for specific types of products, sentiment analysis, time series decomposition and the creation of an ARIMA predictive model.
An in-depth analysis of customer behavior and sales trends, as well as the study of demand for specific products, enables a better understanding of consumer preferences.
Sentiment analysis allows assessment of customer opinions, while time series decomposition provides insight into seasonal and trend changes occurring in the data.
Finally, a predictive model enables forecasting of future sales performance, which is crucial for developing effective business strategies.
This work focuses on a comprehensive approach to analyzing e-commerce data, emphasizing understanding customers, demand dynamics and effectively forecasting future trends.

Keywords: Data Analysis, Sentiment Analysis, Time series Forecasting, ARIMA

SPIS TREŚCI

1. Sformułowanie problemu biznezowego

Firma Amazon, będąca liderem w branży e-commerce, stawia sobie za cel zwiększenie efektywności operacyjnej i wzrostu przychodów.

W tym celu proponuję przeprowadzenie zaawansowanej analizy danych, skupiając się na analizie klienta, popycie na konkretne produkty, analizie sentymentu, dekompozycji szeregów czasowych oraz implementacji modelu prognostycznego ARIMA.

Cele:

1. **Zrozumienie klientów:** Analiza zachowań klientów pozwoli nam lepiej zrozumieć ich preferencje, co umożliwi dostosowanie oferty do oczekiwań konsumentów.
2. **Optymalizacja popytu**: Badanie popytu na konkretne produkty pozwoli na identyfikację najbardziej atrakcyjnych produktów, co skutkować będzie lepszym zarządzaniem dostawą i zapasami.
3. **Ewaluacja sentymentu**: Analiza sentymentu opinii klientów pomoże w identyfikacji obszarów wymagających poprawy oraz w budowaniu pozytywnego wizerunku marki.
4. **Dekompozycja szeregów czasowych**: Zastosowanie dekompozycji szeregów czasowych pozwoli na zidentyfikowanie sezonowych i trendowych zmian, co usprawni planowanie marketingowe.
5. **Prognozowanie przyszłych trendów:** Implementacja modelu prognostycznego ARIMA umożliwi precyzyjne prognozowanie przyszłych trendów sprzedażowych, co jest kluczowe dla skutecznej strategii biznesowej.

2. Eksploracyjna analiza danych

Celem realizacji przedstawionych założeń biznesowych, przeanalizowane zostały następujące zbiory danych:

- retail_sales_dataset.csv
- women_clothing_ecommerce_sales.csv
- reviews.csv

Poniżej przedstawiona została eksploracyjna analiza danych dla każdego datasetu:

	Transaction ID	Date	Customer ID	Gender	Age	Product Category	Quantity	Price per Unit	Total Amount
521	522	2023-01-01	CUST522	Male	46	Beauty	3	500	1500
179	180	2023-01-01	CUST180	Male	41	Clothing	3	300	900
558	559	2023-01-01	CUST559	Female	40	Clothing	4	300	1200
302	303	2023-01-02	CUST303	Male	19	Electronics	3	30	90
978	979	2023-01-02	CUST979	Female	19	Beauty	1	25	25

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1000 entries, 521 to 649
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Transaction ID    1000 non-null   int64
 1   Date              1000 non-null   datetime64[ns]
 2   Customer ID       1000 non-null   object
 3   Gender            1000 non-null   object
 4   Age               1000 non-null   int64
 5   Product Category  1000 non-null   object
 6   Quantity          1000 non-null   int64
 7   Price per Unit    1000 non-null   int64
 8   Total Amount      1000 non-null   int64
dtypes: datetime64[ns](1), int64(5), object(3)
memory usage: 78.1+ KB
```

	Transaction ID	Age	Quantity	Price per Unit	Total Amount
count	1000.000000	1000.00000	1000.000000	1000.000000	1000.000000
mean	500.500000	41.39200	2.514000	179.890000	456.000000
std	288.819436	13.68143	1.132734	189.681356	559.997632
min	1.000000	18.00000	1.000000	25.000000	25.000000
25%	250.750000	29.00000	1.000000	30.000000	60.000000
50%	500.500000	42.00000	3.000000	50.000000	135.000000
75%	750.250000	53.00000	4.000000	300.000000	900.000000
max	1000.000000	64.00000	4.000000	500.000000	2000.000000

	order_id	order_date	sku	color	size	unit_price	quantity	revenue
0	1	2022-06-01 16:05:00	708	Dark Blue	2XL	298	1	298
1	1	2022-06-01 16:05:00	89	Dark Blue	2XL	258	1	258
39	32	2022-06-02 12:57:00	799	Dark Blue	XL	288	1	288
63	53	2022-06-03 11:52:00	799	Dark Blue	L	288	1	288
64	54	2022-06-03 22:00:00	708	Dark Blue	2XL	278	1	278

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 527 entries, 0 to 468
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   order_id    527 non-null    int64
 1   order_date  527 non-null    datetime64[ns]
 2   sku         527 non-null    object
 3   color       527 non-null    object
 4   size        490 non-null    object
 5   unit_price  527 non-null    int64
 6   quantity    527 non-null    int64
 7   revenue     527 non-null    int64
dtypes: datetime64[ns](1), int64(4), object(3)
memory usage: 37.1+ KB
```

	order_id	unit_price	quantity	revenue
count	527.000000	527.000000	527.000000	527.000000
mean	115.313093	274.973435	1.011385	278.024668
std	73.106716	34.517412	0.106193	44.741095
min	1.000000	191.000000	1.000000	191.000000
25%	54.000000	266.000000	1.000000	266.000000
50%	110.000000	278.000000	1.000000	278.000000
75%	166.000000	288.000000	1.000000	288.000000
max	273.000000	899.000000	2.000000	899.000000

	Clothing ID	Age	Title	Review Text
0	767	33	NaN	Absolutely wonderful - silky and sexy and comf...
1	1080	34	NaN	Love this dress! it's sooo pretty. i happene...
2	1077	60	Some major design flaws	I had such high hopes for this dress and reall...
3	1049	50	My favorite buy!	I love, love, love this jumpsuit. it's fun, fl...
4	847	47	Flattering shirt	This shirt is very flattering to all due to th...

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 23486 entries, 0 to 23485
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   Clothing ID              23486 non-null  int64 
 1   Age                      23486 non-null  int64 
 2   Title                    19676 non-null  object
 3   Review Text              22641 non-null  object
 4   Rating                   23486 non-null  int64 
 5   Recommended IND          23486 non-null  int64 
 6   Positive Feedback Count  23486 non-null  int64 
 7   Division Name            23472 non-null  object
 8   Department Name          23472 non-null  object
 9   Class Name               23472 non-null  object
dtypes: int64(5), object(5)
memory usage: 2.0+ MB
```

	Clothing ID	Age	Rating	Recommended IND	Positive Feedback Count
count	23486.000000	23486.000000	23486.000000	23486.000000	23486.000000
mean	918.118709	43.198544	4.196032	0.822362	2.535936
std	203.298980	12.279544	1.110031	0.382216	5.702202
min	0.000000	18.000000	1.000000	0.000000	0.000000
25%	861.000000	34.000000	4.000000	1.000000	0.000000
50%	936.000000	41.000000	5.000000	1.000000	1.000000
75%	1078.000000	52.000000	5.000000	1.000000	3.000000
max	1205.000000	99.000000	5.000000	1.000000	122.000000

3. Przetwarzanie danych

Celem stworzenia wartościowych wizualizacji oraz modeli, zbiory danych zostały odpowiednio przetworzone:

```
df = pd.read_csv("retail_sales_dataset.csv")

bins = [18, 22, 28, 33, 40, 45, 50, 55, 100]
labels = ['Early Adult Transition',
          'Entering the Adult World',
          'Age 30 Transition',
          'Settling Down',
          'Mid-Life Transition',
          'Entering the Middle Years',
          'Age 50 Transition',
          'Late Adulthood'
          ]


df['Levinson Age Group'] = pd.cut(df['Age'], bins=bins, labels=labels, right=False)

age_groups = df.groupby('Levinson Age Group')['Age'].mean()

print(df)

df.to_csv('retail_grouped.csv')
```

Do ‘retail_sales_dataset.csv’ dodana została nowa kolumna [‘Levinson Age Group’], celem przypisania klientów do określonych grup wiekowych, przedstawiających określone fazy życia człowieka [1].

Analiza sprzedaży w kontekście faz życiowych Levinsona może dostarczyć cennych informacji na temat preferencji zakupowych, potrzeb i oczekiwań klientów w różnych etapach ich życia dorosłego.

Działanie w zgodzie z tymi oczekiwaniami może przyczynić się do lepszego dostosowania strategii marketingowej oraz efektywniejszej sprzedaży.

```
# Print the DataFrame with sentiment analysis results
bins = [18, 22, 28, 33, 40, 45, 50, 55, 100]
bins2 =[1,2,3,4,5,6,7,8]
labels = ['Early Adult Transition',
          'Entering the Adult World',
          'Age 30 Transition',
          'Settling Down',
          'Mid-Life Transition',
          'Entering the Middle Years',
          'Age 50 Transition',
          'Late Adulthood'
          ]


df['Levinson Age Group'] = pd.cut(df['Age'], bins=bins, labels=labels, right=False)

result_df = df.groupby(['Levinson Age Group', 'Sentiment']).size().unstack(fill_value=0)
result_df.reset_index()
result_df.to_csv('%withingroup.csv')
```

Podobnemu zabiegowi poddany został zestaw danych, przeznaczony do analizy sentymentu. Dodatkowo, wyznaczona została nowa kolumna [‘Sentiment’] zawierająca przetworzone za pomocą modelu NLP flagi *Positive, Neutral, Negative* uzyskane dzięki interpretacji tekstu opinii na temat określonego produktu:

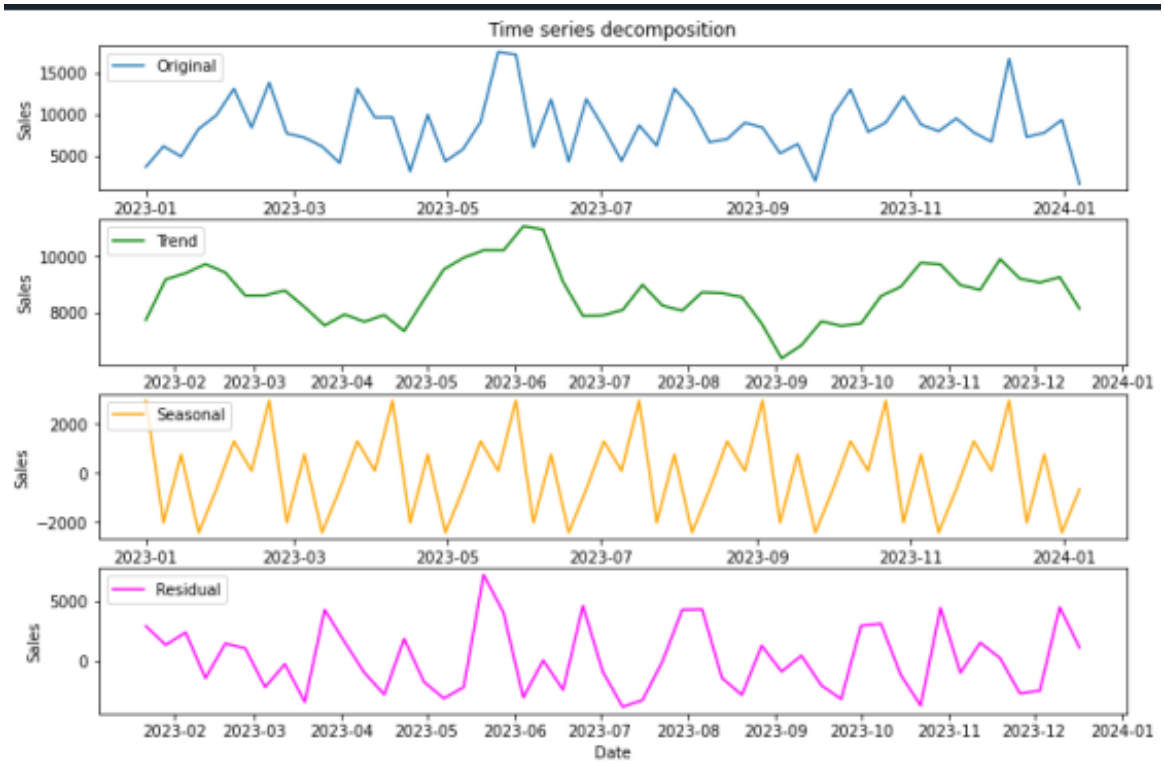
```
from textblob import TextBlob
import pandas as pd

pd.options.display.max_columns = None
df = pd.read_csv("reviews.csv")

# Define a function for sentiment analysis using TextBlob
def analyze_sentiment(text):
    blob = TextBlob(text)
    sentiment_polarity = blob.sentiment.polarity
    if sentiment_polarity > 0:
        return "Positive"
    elif sentiment_polarity < 0:
        return "Negative"
    else:
        return "Neutral"

# Apply sentiment analysis to the 'Text' column and create a new 'Sentiment' column
df['Review Text'] = df['Review Text'].astype(str)
df['Sentiment'] = df['Review Text'].apply(analyze_sentiment)
```

Na koniec wyznaczona została metryka, prezentująca udział poszczególnych sentymentów w każdej grupie wiekowej.

	Levinson Age Group	Negative	Neutral	Positive
0	Early Adult Transition	9	7	238
1	Entering the Adult World	94	67	1575
2	Age 30 Transition	147	119	2282
3	Settling Down	347	299	5449
4	Mid-Life Transition	189	139	2877

4. Przetwarzanie danych do budowy modelu prognostycznego

```
from statsmodels.tsa.seasonal import seasonal_decompose
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.stattools import adfuller
from pandas.plotting import autocorrelation_plot
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf


df = pd.read_csv("retail_sales_dataset.csv", header=0, index_col=0)

time_series = df[['Date','Total Amount']]

time_series['Date'] = pd.to_datetime(df['Date'])
time_series.set_index('Date', inplace=True)

weekly_sales = time_series.resample('w').sum()

result = seasonal_decompose(weekly_sales, model = 'additive', period=7)
```



Aby stworzyć model prognostyczny, zgodnie z metodologią Boxa-Jenkinsa, badany szereg czasowy należy poddać procesom identyfikacji, estymacji oraz diagnostyki [2].

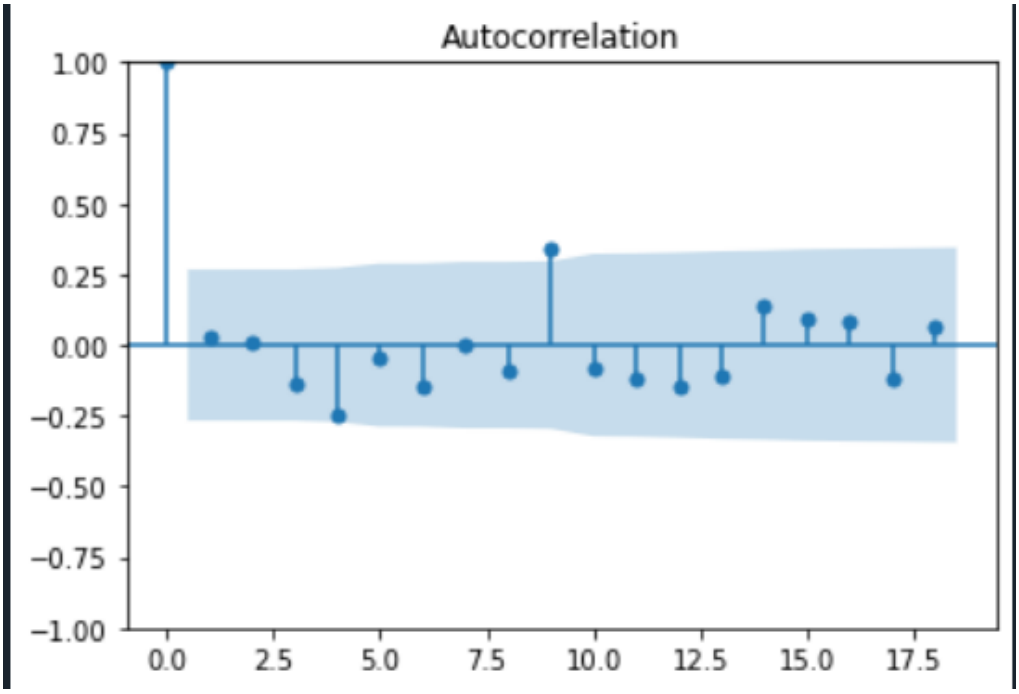
Proces identyfikacji ma na celu zbadanie zjawisk występujących w danym szeregu czasowym, tj. trend, sezonowość, zależności matematyczne między danymi (model addytywny i multiplikatywny) oraz stacjonarność.

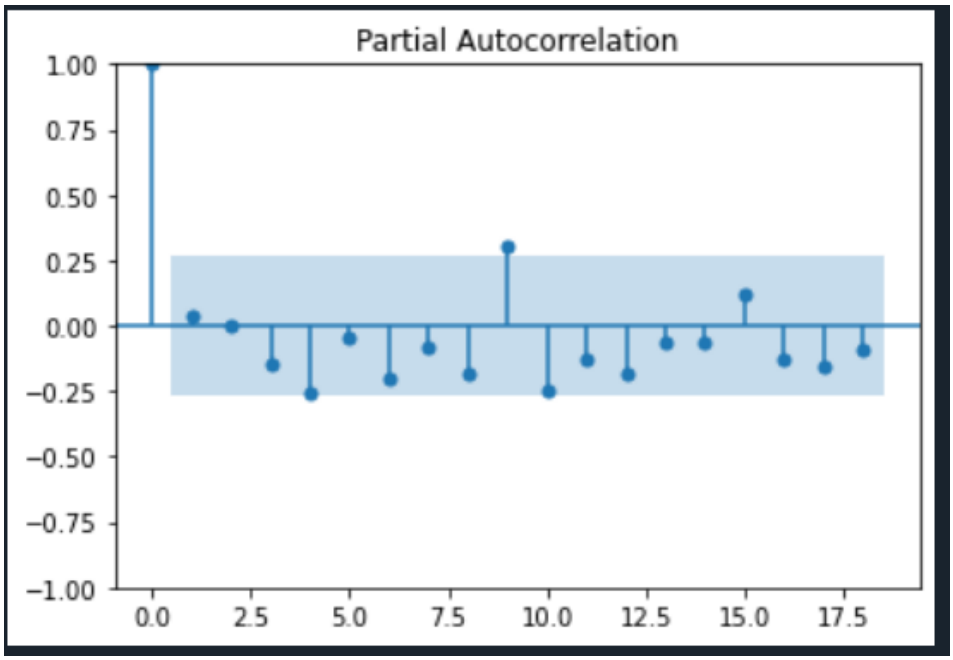
Do badania stacjonarności, można posłużyć się wykresami funkcji autokorelacji (ACF) oraz częściowej autokorelacji (PACF) , jak również testami statystycznymi, tj. rozszerzony test Dickeya – Fullera (ADF).

Polega on na badaniu hipotezy zerowej, według której każdy niestacjonarny proces zawiera pierwiastek jednostkowy, leżący w obrębie jego koła jednostkowego. W przypadku jego braku rozważana jest hipoteza alternatywna, według której wartość testu statystycznego szeregu stacjonarnego nie przekracza wartości krytycznych, znajdujących się w tablicach statystycznych testu ADF [3].

```python
# -*- coding: utf-8 -*-
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.seasonal import seasonal_decompose
from pandas.plotting import autocorrelation_plot
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.stats.diagnostic import acorr_ljungbox
from statsmodels.tsa.stattools import kpss

df = pd.read_csv("retail_sales_dataset.csv", header=0, index_col=0)

time_series = df[['Date','Total Amount']]

time_series['Date'] = pd.to_datetime(df['Date'])
time_series.set_index('Date', inplace=True)

weekly_sales = time_series.resample('w').sum()

df_values = weekly_sales['Total Amount'].values

#Perform augmented Dickey-Fuller test
adf_test = adfuller(weekly_sales)

autocorrelation_plot(weekly_sales)
plot_acf(weekly_sales, adjusted=True)
plot_pacf(weekly_sales, method="ols")
```

```python
#Perform augmented Dickey-Fuller test
adf_test = adfuller(weekly_sales)

autocorrelation_plot(weekly_sales)
plot_acf(weekly_sales, adjusted=True)
plot_pacf(weekly_sales, method="ols")



#Print ADF results
print("ADF Statistic:" ,adf_test[0])
print("p-value:" ,adf_test[1])
print("Critical values :" ,adf_test[4])

lb_test_stat, lb_p_value = acorr_ljungbox(weekly_sales, lags=1, return_df=False)
print(f"Ljung-Box Test Statistic: {lb_test_stat}")
print(f"P-value: {lb_p_value}")

kpss_stat, kpss_p_value, _, _ = kpss(weekly_sales)

print(f'KPSS Statistic: {kpss_stat}')
print(f'p-value: {kpss_p_value}')
```

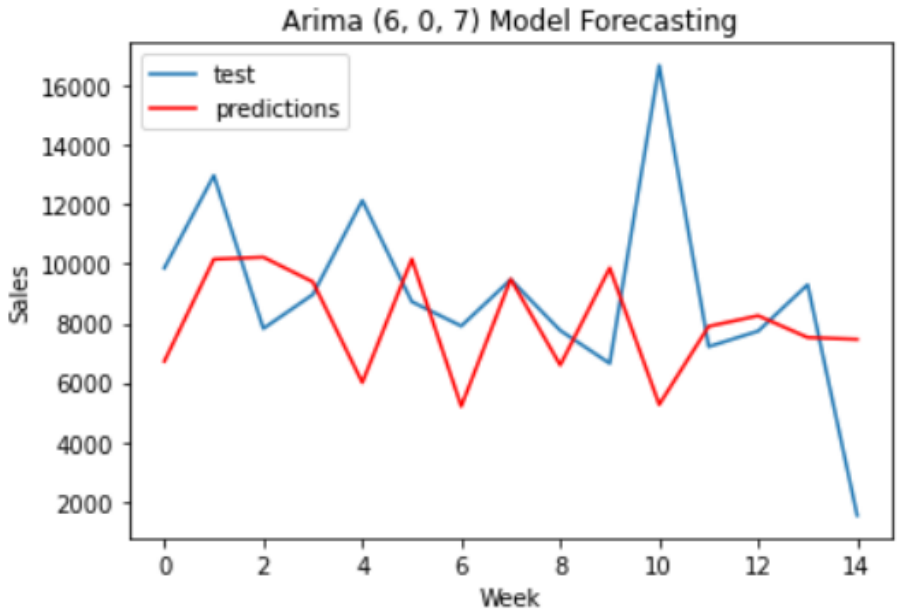




Na podstawie odczytu wartości p, q z wykresów ACF/PACF oraz optymalizacji hiperparametrów poprzez minimalizację metryk RSME oraz MAPE za pomocą algorytmu *grid search* wyznaczona została optymalna wartość parametrów p,d,q dla badanego szeregu czasowego. Ostatecznie zaproponowany został model ARIMA(6, 0, 7).

```
(6, 0, 7) 3204.2852972957076
(6, 0, 7) 0.4479469887448478
```

```
#Splitting data into train and test sets
size = 39
train, test = df_values[0:size], df_values[size:len(df_values)]

# Define the parameter combinations for the grid search
p_values = range(0, 10)  # Adjust based on your expectation
d_values = range(0, 10)  # Adjust based on your expectation
q_values = range(0, 10)  # Adjust based on your expectation

# Generate all possible combinations of p, d, and q
combinations = list(itertools.product(p_values, d_values, q_values))

best_rmse = float('inf')
best_mape = float('inf')
best_order = None

# Iterate through all combinations
for order in combinations:
    try:
        model = ARIMA(train, order=order)
        model_fit = model.fit()
        predictions = model_fit.forecast(steps=len(test))

        # Calculate RMSE for the current combination
        rmse = sqrt(mean_squared_error(test, predictions))
        mape = mean_absolute_percentage_error(test, predictions)
        # Update the best model if the current combination performs better
        if (rmse < best_rmse and mape < best_mape):
            best_rmse = rmse
            best_order = order
            best_mape = mape

    except Exception as e:
        print(f"Error for order {order}: {e}")

print(best_order, best_rmse)
print(best_order, best_mape)
```

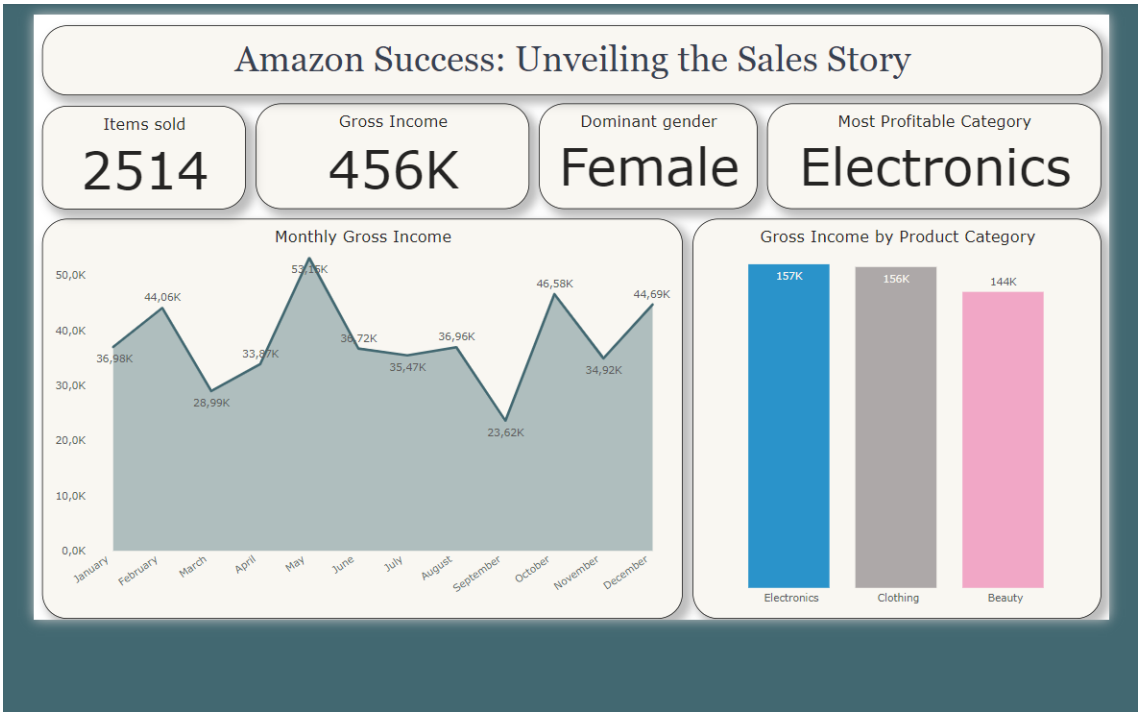



```
predicted=6723.369337, expected=9865.000000
predicted=10161.738943, expected=12980.000000
predicted=10229.849621, expected=7825.000000
predicted=9398.272343, expected=8970.000000
predicted=6004.133281, expected=12140.000000
predicted=10165.174178, expected=8730.000000
predicted=5208.407700, expected=7910.000000
predicted=9488.205267, expected=9475.000000
predicted=6592.088415, expected=7760.000000
predicted=9866.055404, expected=6650.000000
predicted=5264.131284, expected=16690.000000
predicted=7899.456612, expected=7215.000000
predicted=8257.967080, expected=7740.000000
predicted=7523.486013, expected=9305.000000
predicted=7460.689996, expected=1530.000000
Test RMSE: 4099.818
0.49677622476486244
```

Pomimo zastosowania metodologii Boxa-Jenkinsa i dostrojenia hiperparametrów uzyskano model prognostyczny o stosunkowo wysokiej wartości błędu RMSE (4099.18) i MAPE (49.6%). Jest to wynik niezadowalający, dlatego model należy odrzucić.

5. Wizualizacja danych, tworzenie narracji i wnioski końcowe

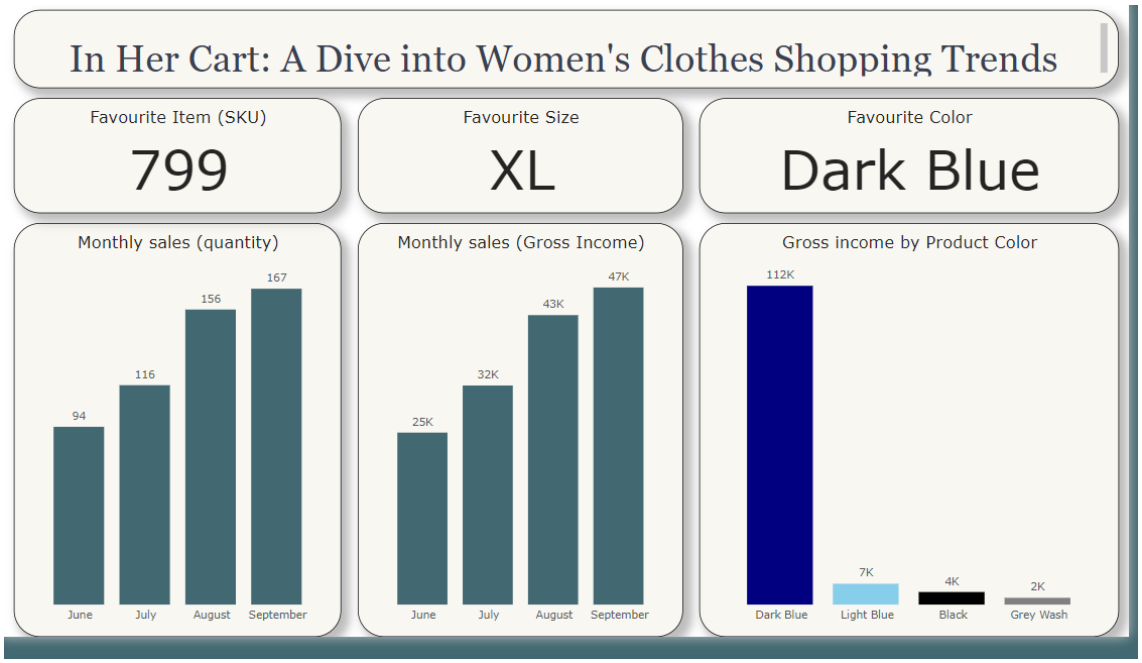
Po ukończonej analizie został stworzony raport, w celu prezentacji wniosków:



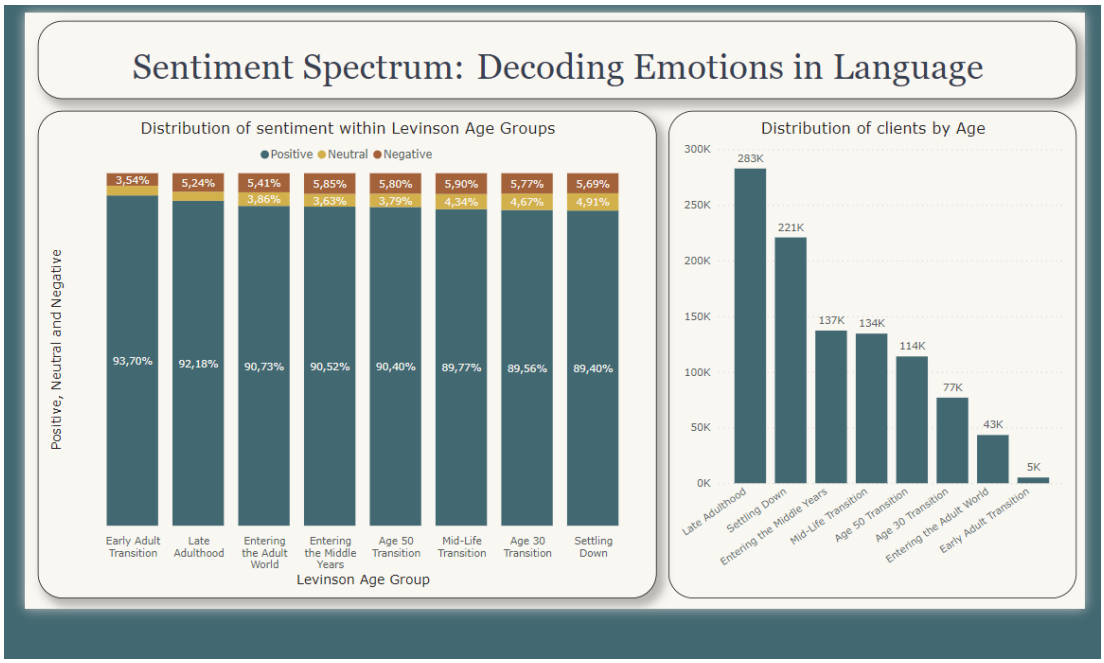
Pierwszy dashboard przedstawia obecny wynik sprzedażowy, wraz z uwzględnieniem najbardziej dochodowych grup produktowych oraz dominującej płci klienta. Celem narracji jest ekspozycja oraz skierowanie uwagi na potrzebę zdefiniowania profilu klienta, aby maksymalizować zyski.

Drugi dashboard przedstawia podział klientów ze względu na ich płeć oraz grupę wiekową. Ma on na celu wstępne utworzenie profili klientów oraz przedstawienie jak bardzo dochodowe są poszczególne grupy.

Celem narracji jest rozbudowa akcji oraz podkreślenie istotności wnikliwej analizy klienta.

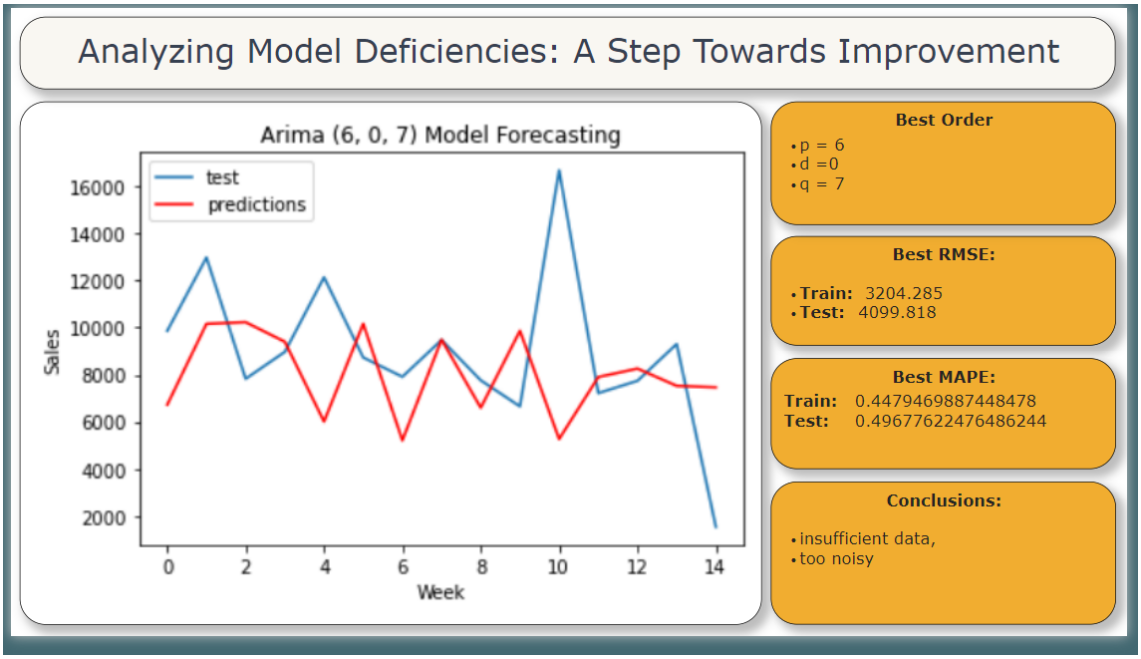


Trzeci dashboard bada głębiej preferencje dominującej grupy klientów (kobiety). Wskazuje na ulubione produkty, ich rozmiar oraz najczęściej wybierany kolor.



Czwarty dashboard pozwala na zbadanie rozkładu ilości klientów oraz procentowego udziału pozytywnych, negatywnych i neutralnych komentarzy na temat produktów, w każdej z rozważanych kategorii wiekowych.

Piąty dashboard przedstawia dekompozycję badanego szeregu czasowego, przedstawiając jego składowe trendu oraz sezonowości, a także wyniki testów ADF, KPSS i Ljung-Box, oraz interpretację wyników.



Ostatni dashboard zawiera walidację *walk-forward* modelu prognostycznego, wyznaczone optymalne hiperparametry oraz wnioski końcowe na temat jakości oraz ilości danych (dane pochodzą jedynie z 1 roku)

Pozytywne wyniki testów ADF oraz KPSS oraz widoczna składowa sezonowa na wykresie dekompozycji mogą stanowić informację o możliwości uzyskania wysokiej dokładności modelu prognostycznego w przypadku zwiększenia ilości i jakości danych.

Literatura

[1] https://www.psychologydiscussion.net/theory/levinsons-theory/levinsons-theory-stages-of-adult-life-human-development-psychology/13594

[2] https://otexts.com/fpp2/.

[3] https://coin.wne.uw.edu.pl/pstrawinski/dane_fin/adf03.pdf