

The screenshot shows a Google Colab notebook environment. The notebook is titled "Untitled12.ipynb" and contains the following code:

```
[1]
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, roc_curve, roc_auc_score
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from textblob import TextBlob
import nltk
nltk.download('stopwords')

[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data] Unzipping corpora/stopwords.zip.
True

[2]
df = pd.read_csv("/content/retail_ai_dataset.csv")
df.head()
```

The output of the second cell shows the first five rows of the dataset:

	date	sales	temperature	holiday	promotion	customer_feedback
0	2023-01-01	210	25	1	1	Loved the discount, great value
1	2023-01-02	180	26	0	0	Delivery was slow
2	2023-01-03	220	27	0	1	Amazing quality and fast shipping
3	2023-01-04	160	28	0	0	Product was damaged
4	2023-01-05	300	24	1	1	Festival offer was excellent

Below the table, there are two buttons: "Generate code with df" and "New interactive sheet".

The third cell contains the following code:

```
[3]
df['date'] = pd.to_datetime(df['date'])
df.set_index('date', inplace=True)
```

Home - Classroom x SKCT x Untitled12.ipynb - Colab x +

colab.research.google.com/drive/1LbpLXtLqKzrP7rLrpHAYou-elmictf10#scrollTo=-L-sZceE42Kt

Untitled12.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text + Run all

RAM Disk

```

[1] df['date'] = pd.to_datetime(df['date'])
    df.set_index('date', inplace=True)

[4] exog = df[['temperature', 'holiday', 'promotion']]
    model = SARIMAX(df['sales'], exog=exog, order=(1,1,1), seasonal_order=(1,1,1,7))
    results = model.fit()

/usr/local/lib/python3.12/dist-packages/statsmodels/tsa/base/tsa_model.py:473: ValueWarning: A date index has been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
self._init_dates(dates, freq)
/usr/local/lib/python3.12/dist-packages/statsmodels/tsa/base/tsa_model.py:473: ValueWarning: A date index has been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
self._init_dates(dates, freq)
/usr/local/lib/python3.12/dist-packages/statsmodels/tsa/statespace/sarimax.py:966: UserWarning: Non-stationary starting autoregressive parameters found. Using zeros as starting parameters.
warn("Non-stationary starting autoregressive parameters")
/usr/local/lib/python3.12/dist-packages/statsmodels/tsa/statespace/sarimax.py:978: UserWarning: Non-invertible starting MA parameters found. Using zeros as starting parameters.
warn("Non-invertible starting MA parameters found.")
/usr/local/lib/python3.12/dist-packages/statsmodels/tsa/statespace/sarimax.py:866: UserWarning: Too few observations to estimate starting parameters for seasonal ARMA. All parameters except for variances will be set to zeros
warn("Too few observations to estimate starting parameters.")

[5] forecast = results.predict(start=len(df), end=len(df)+6,
    exog=exog.tail(7))

print("\n SALES FORECAST:")
print(forecast)

---
SALES FORECAST:
15    238.959764
16    178.880911
17    271.423371
18    269.878642
19    287.862732
20    143.554741
21    318.525002
Name: predicted_mean, dtype: float64
/usr/local/lib/python3.12/dist-packages/statsmodels/tsa/base/tsa_model.py:837: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at 'start'.
return get_prediction_index(
/usr/local/lib/python3.12/dist-packages/statsmodels/tsa/base/tsa_model.py:837: FutureWarning: No supported index is available. In the next version, calling this method in a model without a supported index will result in an e

```

Variables Terminal

29°C Partly cloudy

6:58 PM Python 3

18:58 11-02-2026

Home - Classroom x SKCT x Untitled12.ipynb - Colab x +

colab.research.google.com/drive/1LbpLXtLqKzrP7rLrpHAYou-elmictf10#scrollTo=-L-sZceE42Kt

Untitled12.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text + Run all

RAM Disk

```

21    318.525002
    Name: predicted_mean, dtype: float64
    /usr/local/lib/python3.12/dist-packages/statsmodels/tsa/base/tsa_model.py:837: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at 'start'.
    return get_prediction_index(
    /usr/local/lib/python3.12/dist-packages/statsmodels/tsa/base/tsa_model.py:837: FutureWarning: No supported index is available. In the next version, calling this method in a model without a supported index will result in an e
    return get_prediction_index(

[6] plt.figure(figsize=(10,5))
    plt.plot(df['sales'], label='Actual')
    plt.plot(forecast, label='forecast', color='red')
    plt.legend()
    plt.title("Sales Forecast")
    plt.show()

---

```

Variables Terminal

29°C Partly cloudy

6:58 PM Python 3

18:58 11-02-2026

The screenshot shows a Google Colab notebook with the following code and output:

```
def get_sentiment(text):
    return TextBlob(text).sentiment.polarity

df['sentiment_score'] = df['customer_feedback'].apply(get_sentiment)
df['sentiment'] = df['sentiment_score'].apply(lambda x: "Positive" if x>0 else "Negative")

print("\n SENTIMENT RESULTS:")
print(df[['customer_feedback', 'sentiment']].head())
```

SENTIMENT RESULTS:

date	customer_feedback	sentiment
2023-01-01	loved the discount, great value	Positive
2023-01-02	Delivery was slow	Negative
2023-01-03	Amazing quality and fast shipping	Positive
2023-01-04	Product was damaged	Negative
2023-01-05	Festival offer was excellent	Positive

```
df['label'] = df['sentiment'].map({"Positive":1, "Negative":0})

X_train, X_test, y_train, y_test = train_test_split(
    df['customer_feedback'], df['label'], test_size=0.2)

text_model = Pipeline([
    ('tfidf', TfidfVectorizer(stop_words='english')),
    ('clf', LogisticRegression())
])

text_model.fit(X_train, y_train)
accuracy = text_model.score(X_test, y_test)

print("\n Feedack Classifier Accuracy: [accuracy*100:.2f]%"
```

The screenshot displays a Google Colab notebook titled 'Untitled12.ipynb'. The notebook contains a Python script that defines a function 'generate_insight()' and prints its output. The function calculates the average sentiment and provides a forecast based on the sentiment analysis.

```
[10]: def generate_insight():
    if avg_sentiment > 0:
        mood = "positive"
    else:
        mood = "negative"

    return f"""
    * AI INSIGHT REPORT
    -----
    Average Sales: {avg_sales:.2f}
    Overall Customer Mood: {mood}
    Forecast Trend: Increasing sales expected due to promotions & holidays.
    Key Driver: Customer sentiment strongly influences sales performance.
    Recommendation: Increase marketing during high-sentiment periods.
    """

print(generate_insight())
```

The output of the script is as follows:

```
* AI INSIGHT REPORT
-----
Average Sales: 231.33
Overall Customer Mood: positive
Forecast Trend: Increasing sales expected due to promotions & holidays.
Key Driver: Customer sentiment strongly influences sales performance.
Recommendation: Increase marketing during high-sentiment periods.
```

The notebook also shows a confirmation message: "END-TO-END AI SYSTEM EXECUTED SUCCESSFULLY".