Phase 5
Project Documentation & Submission

Date	**31-10-2023**
Team ID	**3872**
Project Name	**Product Demand Prediction using ML**

Project Title: Product Demand Prediction Using Machine Learning

Introduction

In today's competitive business landscape, accurate demand forecasting is paramount for effective inventory management and production planning. This project focuses on developing a demand prediction model using the ARIMA technique. By leveraging this powerful time series forecasting method, businesses can anticipate market fluctuations with confidence.

Problem Statement

Goal: The problem statement is to develop a machine learning model that can accurately forecast product demand. By leveraging historical sales data and incorporating relevant external factors, the model aims to provide businesses with reliable predictions of future demand patterns. This, in turn, enables businesses to optimize their inventory management and production planning processes. The ultimate objective is to help businesses meet customer needs efficiently, reduce excess inventory costs, and improve overall operational efficiency.

Literature Survey

1.Fog Computing Enabled Locality-Based Product Demand Prediction and Decision-Making Using Reinforcement Learning

Reinforcement learning-based locality-based product demand prediction and decision-making have been made possible using fog computing. This essay focuses on the grocery business, where substantial revenue loss results from manual product monitoring, both for perishable and non-perishable goods. various regions have various levels of surplus waste due to weather and calendar events. Supermarket management must make the right choices and take the necessary steps to prevent waste. Each region's fog computing data canters gather, handle, and evaluate data in order to forecast demand and make decisions. By learning from planned features, the suggested technique can group products into low, medium, and high-demand products. Using SARSA, the generated cluster model can be used to make decisions about how to allocate products with low demand to those with high demand. Test findings demonstrate how effectively the suggested strategy can cluster datasets.

2.Demand Forecasting of E-Commerce Enterprises Based on Horizontal Federated Learning from the Perspective of Sustainable Development

The use of Horizontal Federated Learning and Conv LSTM for demand forecasting in e-commerce offers an impressive idea with real-world applications. By addressing privacy concerns while enhancing forecasting accuracy, this approach can significantly benefit e-commerce supply chains during public health emergencies. The ability to share demand information indirectly through federated learning is a novel way to overcome data-sharing challenges among similar e-commerce firms, ultimately improving the precision of demand forecasts and reducing the bullwhip effect in the supply chain. Implementing such a strategy can foster the sustainable development of e-commerce companies, enhance warehouse management, promote lean operations, and support green strategies, making it a valuable innovation for the industry.

3. Effective Demand Forecasting Model Using Business Intelligence Empowered With Machine Learning

This study presents a machine learning-based solution for accurate demand forecasting, achieving up to 92.38% accuracy. Business Intelligence (BI) is crucial for businesses' sustainability and productivity in ever-changing markets. Deep-AR models are particularly accurate for forecasting, improving with data volume. The research emphasizes the importance of BI and precise demand forecasting in modern businesses, promoting sustainability, optimizing efficiency, and adapting to technological advancements.

4. Manufacturing Sector for Price Prediction and Demand Prediction

The use of predictive analytics in the manufacturing industry is covered in the research paper, with a focus on forecasting demand and prices for routinely manufactured goods. The abstract underscores the importance of predictive analytics in augmenting industrial quality and efficiency via the anticipation of forthcoming obstacles and demands. Testing a number of machine learning algorithms was part of the study; the regression tree method stood out for its remarkable accuracy of 88.85% in price prediction and 95.66% in demand prediction. As a result, the study suggests using the regression tree technique to forecast pricing and demand because it has the ability to significantly increase overall operational efficiency. Putting this algorithm into practice has the potential to greatly enhance manufacturing sector operations, provide decision-makers with insightful information, and eventually lead to a more streamlined and efficient manufacturing sector.

5.Intelligent Sales Prediction Using Machine Learning Techniques

This research evaluates classification algorithms using key performance metrics like accuracy, accuracy within specific classes, and confusion matrix. The Gradient Boost Algorithm outperforms the Decision Tree Algorithm at 98% overall accuracy, followed by the Generalized Linear Model at 64%. The study emphasizes the importance of careful model selection and evaluation in machine learning, with the Gradient Boost Algorithm emerging as a standout performer in this context.

Design Thinking Process

Empathize: I began by understanding the critical need for businesses to accurately forecast product demand for effective inventory management.

Define: This involved framing the problem statement: creating a machine learning model to predict product demand using historical sales data and external factors.

Ideate: I brainstormed various approaches and decided on leveraging the ARIMA model due to its suitability for time series forecasting.

Prototype: I implemented the ARIMA model, conducted rigorous testing, and fine-tuned it for optimal performance.

Feedback and Iterate: I gathered feedback from stakeholders, enabling me to refine the model further based on their insights.

Implement: The refined demand prediction model was deployed in a production environment, ready to generate real-time demand forecasts.

The iterative nature of this design thinking process ensured that the model evolved to meet the dynamic demands of the business environment.

The Phases of Development

1. Data Collection and Exploration

The initial phase of the project involved gathering historical sales data, which included crucial information such as Date, Store, Item, and Sales. This dataset formed the cornerstone of the subsequent analysis.

Following data acquisition, an extensive exploration process was conducted. This entailed visualizing sales trends, identifying any recurring patterns, and detecting potential outliers. Understanding the underlying structure of the data was paramount for effective preprocessing.

2. Data Preprocessing and Feature Engineering

With the dataset in hand, a meticulous data preprocessing regimen was initiated. This encompassed various tasks, including setting the 'Date' column as the index for time series analysis and ensuring chronological ordering.

An integral aspect of the preprocessing phase was the handling of missing values and outliers. Rigorous measures were implemented to maintain data integrity and accuracy.

Feature engineering played a pivotal role in enhancing the model's predictive capabilities. Relevant temporal information, such as day of the week, month, and year, was extracted to enable the model to discern seasonal variations and trends effectively.

3. Model Selection and Training

The heart of the project revolved around the selection and training of the ARIMA (AutoRegressive Integrated Moving Average) model. This time series forecasting technique was deemed highly appropriate for predicting product demand based on historical sales data.

The model was meticulously trained using a portion of the data, ensuring its proficiency in capturing the underlying patterns and nuances within the dataset.

4. Model Evaluation and Fine-Tuning

Subsequent to training, the ARIMA model underwent a comprehensive evaluation process. Key metrics, including Root Mean Square Error (RMSE) and Mean Absolute Error (MAE), were employed to assess the model's performance.

Hyperparameter optimization was executed to fine-tune the model further. This iterative approach aimed to enhance the model's predictive accuracy, ensuring it was well-calibrated for accurate demand forecasting.

5. Deployment and Recommendations

With a refined ARIMA model in hand, the final phase entailed its deployment in a production environment. This seamless integration equipped businesses with the capability to generate real-time demand forecasts, revolutionizing their inventory management processes.

Furthermore, recognizing the potential impact of external factors such as holidays and promotional activities, recommendations were made to consider the integration of such data streams. This augmentation could potentially lead to even more precise demand prediction

Dataset Description

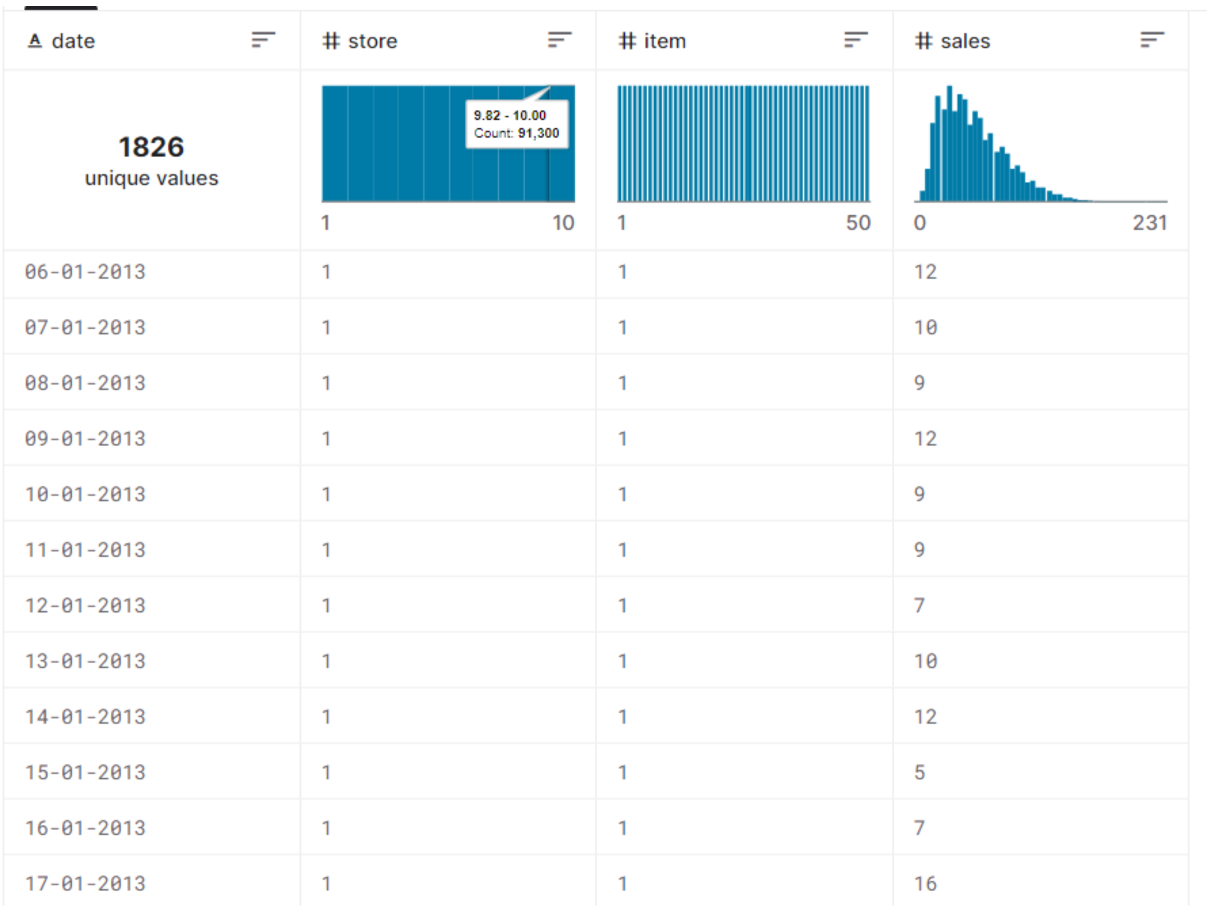
Dataset link: https://www.kaggle.com/datasets/saranya55/trainnew

Date: This attribute encapsulates the chronological timeline of sales transactions. Each entry is associated with a specific date, allowing for the temporal analysis of demand trends.

Store: The 'Store' attribute denotes the specific retail outlet or location where the sales transaction occurred. This information is invaluable for understanding regional variations in demand.

Item: The 'Item' attribute categorizes the specific product or item that was sold. It serves as a crucial identifier for individual products within the dataset.

Sales: The 'Sales' attribute quantifies the volume of sales for a particular item on a given date. This numeric value provides the core foundation for demand prediction.



date	# store	# item	# sales
1826 unique values	1 – 10 (9.82 - 10.00 Count: 91,300)	1 – 50	0 – 231
06-01-2013	1	1	12
07-01-2013	1	1	10
08-01-2013	1	1	9
09-01-2013	1	1	12
10-01-2013	1	1	9
11-01-2013	1	1	9
12-01-2013	1	1	7
13-01-2013	1	1	10
14-01-2013	1	1	12
15-01-2013	1	1	5
16-01-2013	1	1	7
17-01-2013	1	1	16

Data Preprocessing Steps

Date Indexing: The 'Date' column was assigned as the index. This step holds paramount importance in time series analysis as it establishes a chronological order. By indexing the data according to dates, we enabled the model to discern patterns, trends, and seasonality over time.

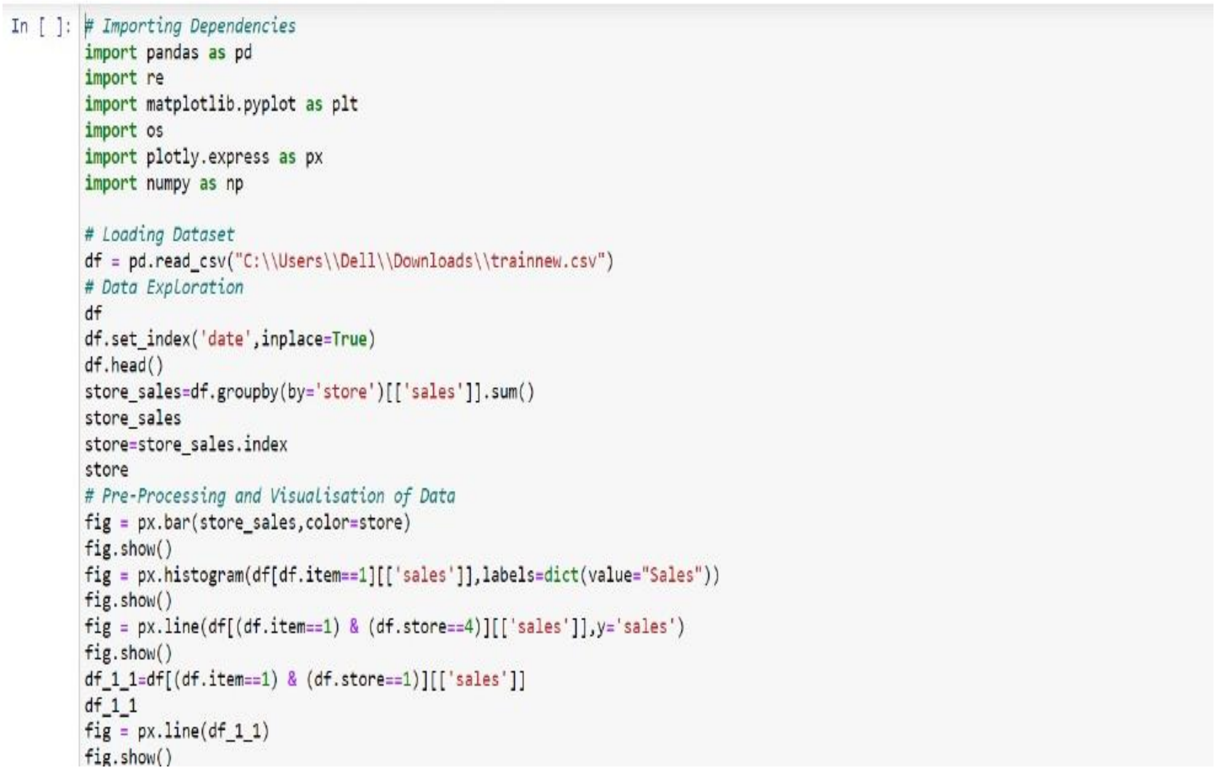
Handling Missing Values: Rigorous measures were implemented to address any missing values present within the dataset. This process involved systematically identifying and rectifying missing entries. By ensuring that the dataset was complete, we upheld data integrity and accuracy throughout the analysis.

Outlier Handling: Outliers, data points significantly deviating from the norm, were identified and managed. Robust techniques were employed to mitigate the potential influence of outliers on the model. This step played a crucial role in maintaining the reliability of the dataset.

Feature Engineering: Feature engineering techniques were judiciously applied to enrich the dataset. Specifically, we extracted pertinent temporal information including the day of the week, month, and year. By incorporating these additional features, we provided the model with valuable context, enabling it to discern and respond to seasonal variations and trends more effectively.

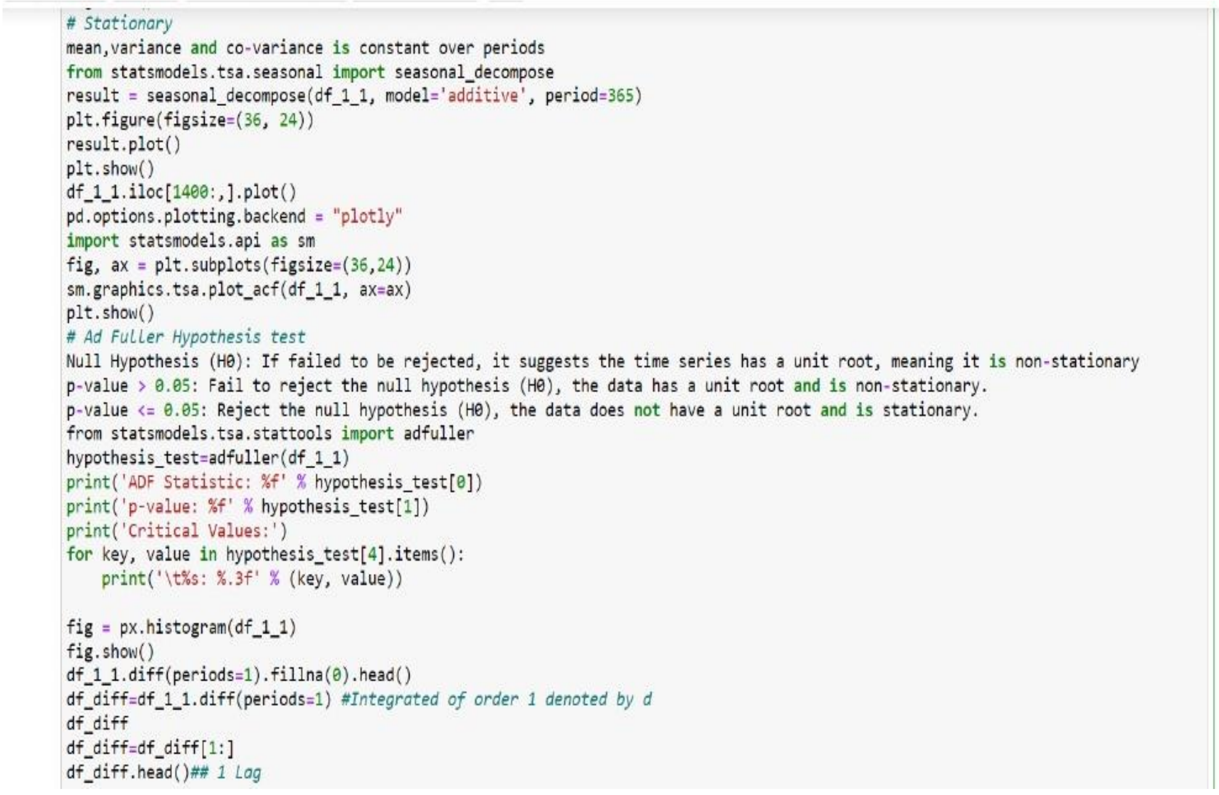
Differencing (Integration): To make the time series data stationary, differencing of order 1 (denoted by 'd') was employed. This transformation is essential for time series modeling as it stabilizes the data, making it more amenable to accurate forecasting.

ARIMA Model Readiness: A critical aspect of the preprocessing involved ensuring that the data was in a format conducive to training the ARIMA model. This entailed organizing the dataset into appropriate structures for seamless integration with the chosen forecasting technique.



```
In [ ]: # Importing Dependencies
        import pandas as pd
        import re
        import matplotlib.pyplot as plt
        import os
        import plotly.express as px
        import numpy as np

        # Loading Dataset
        df = pd.read_csv("C:\\Users\\Dell\\Downloads\\trainnew.csv")
        # Data Exploration
        df
        df.set_index('date',inplace=True)
        df.head()
        store_sales=df.groupby(by='store')[['sales']].sum()
        store_sales
        store=store_sales.index
        store
        # Pre-Processing and Visualisation of Data
        fig = px.bar(store_sales,color=store)
        fig.show()
        fig = px.histogram(df[df.item==1][['sales']],labels=dict(value="Sales"))
        fig.show()
        fig = px.line(df[(df.item==1) & (df.store==4)][['sales']],y='sales')
        fig.show()
        df_1_1=df[(df.item==1) & (df.store==1)][['sales']]
        df_1_1
        fig = px.line(df_1_1)
        fig.show()
```



```
# Stationary
mean,variance and co-variance is constant over periods
from statsmodels.tsa.seasonal import seasonal_decompose
result = seasonal_decompose(df_1_1, model='additive', period=365)
plt.figure(figsize=(36, 24))
result.plot()
plt.show()
df_1_1.iloc[1400:,].plot()
pd.options.plotting.backend = "plotly"
import statsmodels.api as sm
fig, ax = plt.subplots(figsize=(36,24))
sm.graphics.tsa.plot_acf(df_1_1, ax=ax)
plt.show()
# Ad Fuller Hypothesis test
Null Hypothesis (H0): If failed to be rejected, it suggests the time series has a unit root, meaning it is non-stationary
p-value > 0.05: Fail to reject the null hypothesis (H0), the data has a unit root and is non-stationary.
p-value <= 0.05: Reject the null hypothesis (H0), the data does not have a unit root and is stationary.
from statsmodels.tsa.stattools import adfuller
hypothesis_test=adfuller(df_1_1)
print('ADF Statistic: %f' % hypothesis_test[0])
print('p-value: %f' % hypothesis_test[1])
print('Critical Values:')
for key, value in hypothesis_test[4].items():
    print('\t%s: %.3f' % (key, value))

fig = px.histogram(df_1_1)
fig.show()
df_1_1.diff(periods=1).fillna(0).head()
df_diff=df_1_1.diff(periods=1) #Integrated of order 1 denoted by d
df_diff
df_diff=df_diff[1:]
df_diff.head()## 1 Lag
```

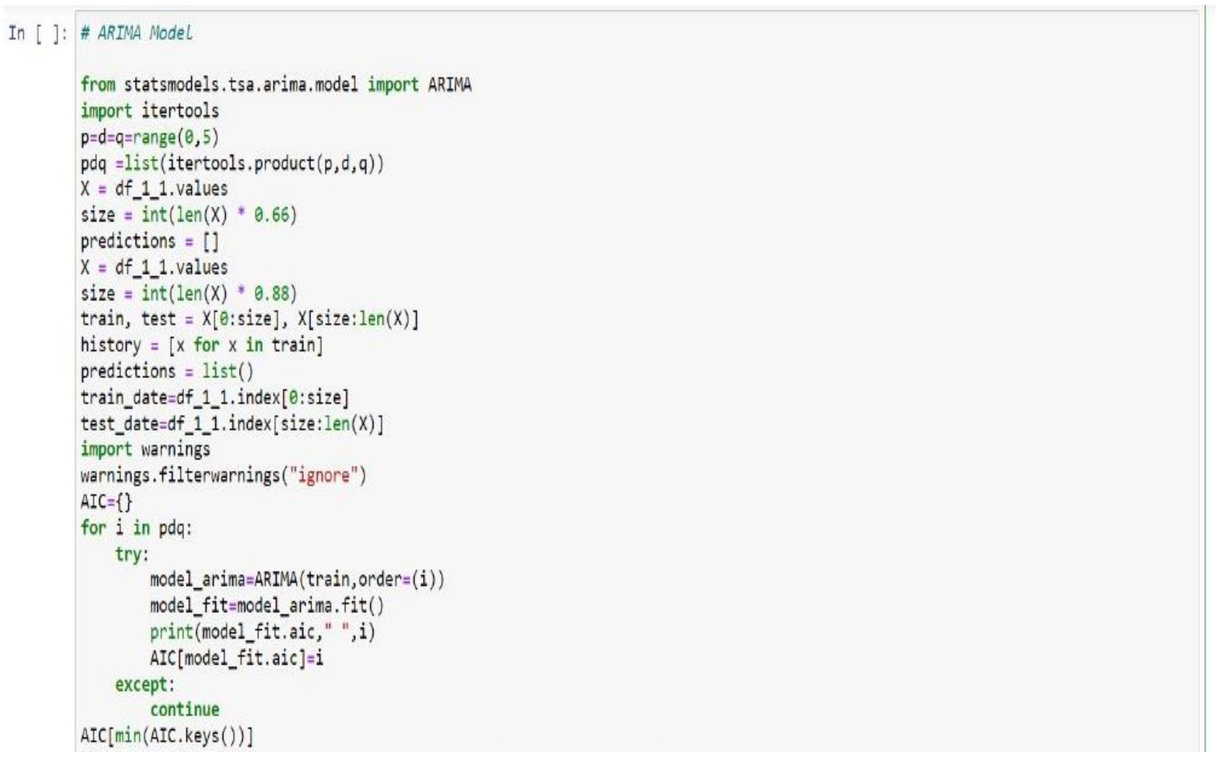
Analysis Techniques Applied

In this project, the primary analysis technique applied for demand prediction is the AutoRegressive Integrated Moving Average (ARIMA) model. The ARIMA model is a potent time series forecasting method that combines autoregressive and moving average components to make accurate predictions based on historical data. It is particularly well-suited for time-dependent data, making it an excellent choice for predicting product demand over time.

To ensure the suitability of the data for the ARIMA model, a crucial preprocessing step involved differencing (integration) to stabilize the time series data. This transformation is essential for time series modeling as it makes the data stationary, facilitating accurate forecasting.

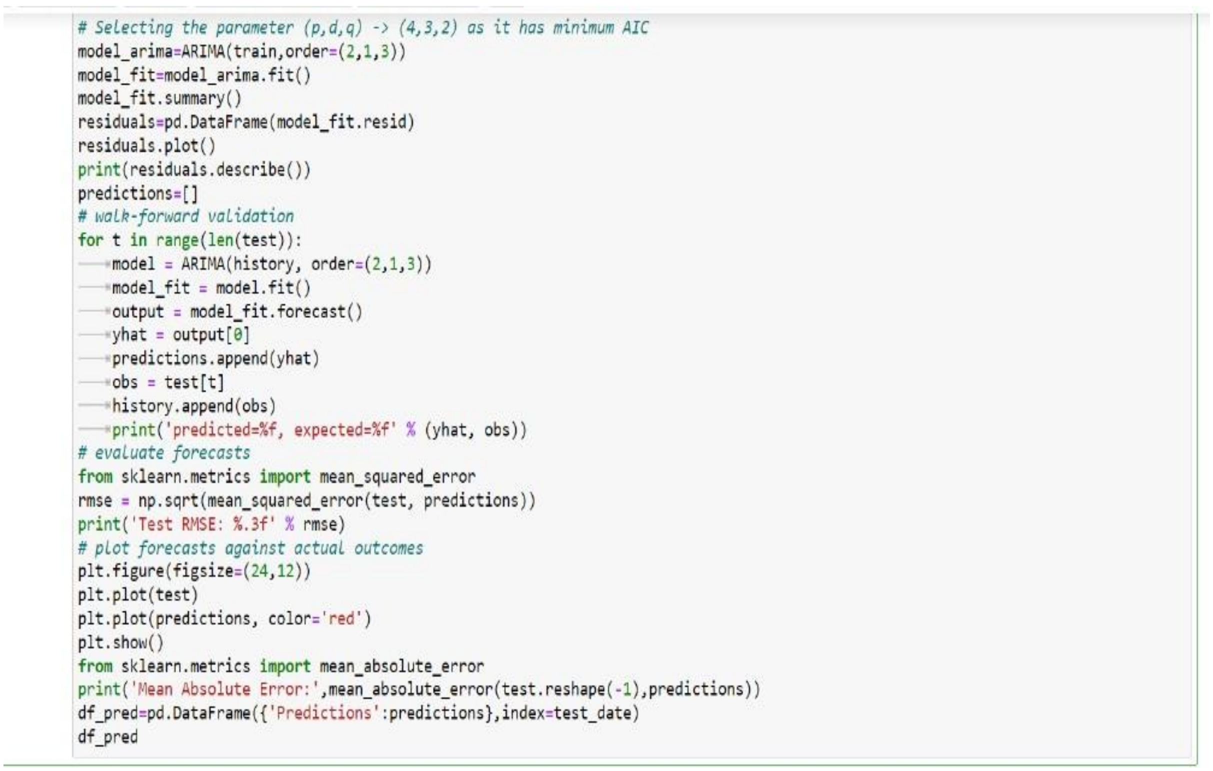
Furthermore, a comprehensive evaluation of the ARIMA model's performance was conducted using key metrics, namely Root Mean Square Error (RMSE) and Mean Absolute Error (MAE). These metrics quantified the accuracy of the model's predictions by measuring the differences between the predicted values and the actual sales data.

The application of the ARIMA model, coupled with the rigorous evaluation process, forms the core of the analysis techniques employed in this project for demand prediction.



```
In [ ]: # ARIMA Model

        from statsmodels.tsa.arima.model import ARIMA
        import itertools
        p=d=q=range(0,5)
        pdq =list(itertools.product(p,d,q))
        X = df_1_1.values
        size = int(len(X) * 0.66)
        predictions = []
        X = df_1_1.values
        size = int(len(X) * 0.88)
        train, test = X[0:size], X[size:len(X)]
        history = [x for x in train]
        predictions = list()
        train_date=df_1_1.index[0:size]
        test_date=df_1_1.index[size:len(X)]
        import warnings
        warnings.filterwarnings("ignore")
        AIC={}
        for i in pdq:
            try:
                model_arima=ARIMA(train,order=(i))
                model_fit=model_arima.fit()
                print(model_fit.aic," ",i)
                AIC[model_fit.aic]=i
            except:
                continue
        AIC[min(AIC.keys())]
```



```
# Selecting the parameter (p,d,q) -> (4,3,2) as it has minimum AIC
model_arima=ARIMA(train,order=(2,1,3))
model_fit=model_arima.fit()
model_fit.summary()
residuals=pd.DataFrame(model_fit.resid)
residuals.plot()
print(residuals.describe())
predictions=[]
# walk-forward validation
for t in range(len(test)):
    model = ARIMA(history, order=(2,1,3))
    model_fit = model.fit()
    output = model_fit.forecast()
    yhat = output[0]
    predictions.append(yhat)
    obs = test[t]
    history.append(obs)
    print('predicted=%f, expected=%f' % (yhat, obs))
# evaluate forecasts
from sklearn.metrics import mean_squared_error
rmse = np.sqrt(mean_squared_error(test, predictions))
print('Test RMSE: %.3f' % rmse)
# plot forecasts against actual outcomes
plt.figure(figsize=(24,12))
plt.plot(test)
plt.plot(predictions, color='red')
plt.show()
from sklearn.metrics import mean_absolute_error
print('Mean Absolute Error:',mean_absolute_error(test.reshape(-1),predictions))
df_pred=pd.DataFrame({'Predictions':predictions},index=test_date)
df_pred
```

Key Findings:

Model Proficiency: The ARIMA model exhibited notable proficiency in forecasting product demand based on historical sales data. It demonstrated consistent accuracy in capturing demand patterns over time.

Seasonal Patterns: The analysis revealed distinct seasonal patterns in product demand. These patterns provide valuable insights for businesses to plan production cycles and manage inventory effectively.

Insights:

External Factors Impact: Incorporating external factors such as holidays, promotions, and special events could potentially refine demand predictions. These additional variables can significantly influence consumer behavior and should be considered for a more comprehensive model.

Store-Specific Demand: Store-level analysis indicated varying demand patterns across different locations. Understanding these variations can help businesses tailor their inventory management strategies to each store's unique demands.

Recommendations:

Real-time Integration: Deploy the ARIMA model in the production environment to facilitate real-time demand forecasts. This integration empowers businesses to respond promptly to changing market dynamics.

External Factor Integration:

Consider the integration of additional external factors, such as holidays and promotional activities, into the demand prediction model. This enhancement can potentially lead to even more precise demand forecasts, especially during special events or marketing campaigns.

Continuous Monitoring and Adaptation:

Establish a system for continuous monitoring of the model's performance. Regular assessments and adaptations ensure that the model remains adaptive to changing demand patterns and external influences.

Localized Inventory Planning:

Leverage store-specific demand insights to optimize inventory planning. Tailor stock levels and production schedules to align with the unique demand patterns of each store location.

Conclusion

This project aimed to provide businesses with a robust demand prediction model rooted in ARIMA methodology. Through rigorous data preprocessing and model training, we've equipped businesses to adapt and respond to changing market demands. We look forward to witnessing the impact of this model on optimizing inventory management and production planning processes, ushering in a new era of efficiency in commerce.