Product Demand Prediction with Machine Learning

Product Demand Prediction using Python

Let’s start by importing the necessary Python libraries and the dataset we need for the task of product demand prediction:

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
import statsmodels.api as sm

# Load your dataset (assuming it has columns like 'date', 'quantity_sold', 'price', 'promotion', etc.)
data = pd.read_csv('product_sales.csv')

# Data preprocessing and feature engineering
# Extract relevant date features
data['date'] = pd.to_datetime(data['date'])
data['year'] = data['date'].dt.year
data['month'] = data['date'].dt.month
data['day_of_week'] = data['date'].dt.dayofweek
```

```
# Split data into features (X) and target (y)
X = data[['year', 'month', 'day_of_week', 'price', 'promotion']]
y = data['quantity_sold']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train a Random Forest regressor
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred_rf = rf_model.predict(X_test)

# Evaluate the Random Forest model
mse_rf = mean_squared_error(y_test, y_pred_rf)
print(f"Random Forest Mean Squared Error: {mse_rf}")

# Optionally, you can visualize feature importances from the Random Forest model
feature_importances = rf_model.feature_importances_
features = X.columns
plt.bar(features, feature_importances)
plt.xlabel('Features')
plt.ylabel('Importance')
plt.title('Feature Importances')
plt.show()

# If you want to explore time series analysis, you can use the statsmodels library
# Fit a time series model (e.g., SARIMA) to the data
model = sm.tsa.SARIMAX(y_train, order=(1, 1, 1), seasonal_order=(1, 1, 1, 12))
results = model.fit()

# Make predictions using the time series model
y_pred_ts = results.predict(start=len(y_train), end=len(y_train) + len(y_test) - 1, dynamic=False)

# Evaluate the time series model
mse_ts = mean_squared_error(y_test, y_pred_ts)
print(f"Time Series Mean Squared Error: {mse_ts}")

# Use the models for demand forecasting as needed
# You can input future features to predict future demand

# Close any open figures from the Random Forest feature importances plot
plt.close()
```