Customer Purchase Behavior Analysis & Sales Forecasting Dashboard

This project involves analyzing e-commerce sales data to extract insights into customer purchasing behavior and forecast future sales trends. It uses Python for data cleaning, exploration, and modeling, and Power BI for interactive dashboard creation. Key Features: 1. Data Cleaning: Handle missing values, duplicates, and incorrect data types. 2. Exploratory Data Analysis (EDA): Identify top-selling products, seasonal trends, and customer segments. 3. Visualization Dashboard: Create interactive visuals for sales trends, category analysis, and geographical sales distribution. 4. Predictive Analytics: Implement ARIMA to forecast future sales with high accuracy. Tech Stack: - Python (Pandas, NumPy, Matplotlib, Seaborn, scikit-learn, statsmodels) - Power BI/Tableau - Dataset: E-commerce Superstore Data

Sample Python Code:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error
import numpy as np

# Load dataset
df = pd.read_csv('ecommerce_sales.csv')

# Data cleaning
df.dropna(inplace=True)
df['Order Date'] = pd.to_datetime(df['Order Date'])
df = df.sort_values('Order Date')

# Monthly sales aggregation
monthly_sales = df.groupby(pd.Grouper(key='Order Date', freq='M'))['Sales'].sum()

# Visualization
plt.figure(figsize=(10,5))
sns.lineplot(x=monthly_sales.index, y=monthly_sales.values)
plt.title('Monthly Sales Trend')
plt.show()

# Forecasting using ARIMA
train_size = int(len(monthly_sales) * 0.8)
train, test = monthly_sales[0:train_size], monthly_sales[train_size:]

model = ARIMA(train, order=(5,1,0))
model_fit = model.fit()
forecast = model_fit.forecast(steps=len(test))

# Evaluation
error = np.sqrt(mean_squared_error(test, forecast))
print(f"RMSE: {error}")

# Plot forecast
plt.figure(figsize=(10,5))
plt.plot(train.index, train, label='Train')
plt.plot(test.index, test, label='Test')
plt.plot(test.index, forecast, label='Forecast')
plt.legend()
plt.show()
```