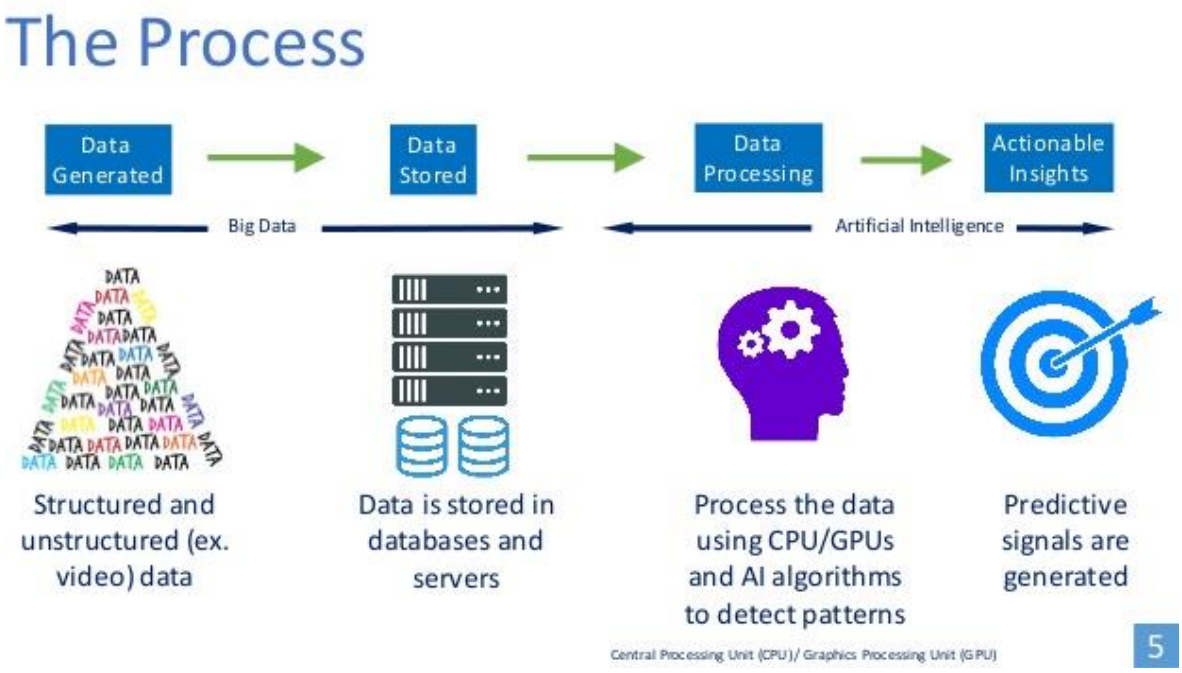
Applied data science

PHASE 5 DOCUMENT

Project Title: Product Demand Prediction with Machine Learnings

Phase 5: **Project Documentation & Submission**

Topic: In this part you will document your project and prepare it for submission.



Product Demand Prediction with Machine Learnings

INTRODUCTION:

Product demand prediction with machine learning is a critical component of modern business operations, helping organizations optimize inventory management, production planning, and supply chain logistics. This process involves leveraging the power of machine learning algorithms to forecast the future demand for a specific product or group of products. By accurately predicting demand, businesses can make informed decisions, reduce operational costs, and enhance customer satisfaction.

Importance of Demand Prediction: Predicting product demand is essential for various industries, including retail, e-commerce, manufacturing, and more. Accurate demand forecasts allow companies to align their resources, such as inventory, staff, and production, with expected customer needs, reducing excess inventory and stockouts.

Data Sources: Machine learning models for demand prediction rely on historical data as input. This data can include sales records, customer orders, economic indicators, seasonality trends, promotional activities, and more. The quality and quantity of data play a crucial role in the accuracy of predictions.

Machine Learning Algorithms: Various machine learning algorithms can be employed for demand prediction, such as time series analysis, regression models, neural networks, and decision trees. The choice of algorithm depends on the complexity of the problem and the available data.

Training and Validation: Machine learning models are trained on historical data and validated to ensure their accuracy. Cross-validation techniques and appropriate evaluation metrics, such as Mean Absolute Error (MAE) or Root Mean Square Error (RMSE), help assess a model's performance.

Business Benefits: Accurate demand prediction offers numerous benefits, including reduced carrying costs through optimal inventory management, improved customer satisfaction by avoiding stockouts, and increased operational efficiency through better resource allocation.

Challenges: Demand prediction faces challenges like seasonality, sudden changes in market dynamics, and data quality issues. Machine learning models need to account for these factors and be robust to outliers.

Integration with Other Systems: Demand prediction systems often need to integrate with other business systems, such as Enterprise Resource Planning (ERP) software and inventory management tools, to ensure seamless execution of supply chain strategies.

DATASET LINK: **https://www.kaggle.com/datasets/chakradharmattapalli/product-demand-prediction-with-machine-learning**

1: Problem Definition and Design Thinking

PROBLEM STATEMENT:

Create a machine learning model that forecasts product demand based on historical sales and external factors, helping businesses optimize inventory management production planning to meet customer needs efficiently.

Project steps:

1.problem definition

2.Design Thinking

Step-1: Problem definition:

The problem is to create a machine learning model that forecasts product demand based on historical sales data and external factors. The goal is to help businesses optimize inventory management and production planning to efficiently meet customer needs. This project involves data collection, data preprocessing, feature engineering, model selection, training, and evaluation.

Step-2: Design Thinking:

(1).Data Collection: Data collection is a systematic process of gathering observations or measurements. Whether you are performing research for business, governmental or academic purposes, data collection allows you to gain first-hand knowledge and original insights into your research problem.

(2).Data Preprocessing : Data preprocessing is an important step in the data mining process. It refers to the cleaning, transforming and integration of data in order tp make it ready for analysis. The goal of data preprocessing is to improve the quality of the data and to make it more suitable for the specific data mining task. Some common steps in data preprocessing are:

(a).Data cleaning

(b).Data Integration

(c).Data Transformation

(d).Data Reduction

(e).Data Discretization

(f).Data Normalization

(3).Feature Engineering : Feature engineering involves creating relevant features from the raw data. For instance: - Lag features: Include past sales data (e.g., sales from the previous week or month) as features. - Date-related features: Extract features like day of the week, month, quarter, or year. - External factors: Incorporate external data such as holidays, economic indicators, or weather forecasts.

4).Model Selection: Choose an appropriate machine learning algorithm for your demand forecasting task. Time series models like ARIMA or machine learning models like Random Forest, XGBoost, or LSTM (if you have a significant amount of data) are common choices. For this example, we'll use a Random Forest regressor. from sklearn.ensemble import RandomForestRegressoR

model=RandomForestRegressor(n_estimators=100, random_state=42)

(5).Model Training: Train the selected model on your training data. Example: model.fit(X_train, y_train) (6).Evaluation: Evaluate your model's performance on the testing dataset using appropriate metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), or Mean Absolute Percentage Error (MAPE). Example: from sklearn.metrics import mean_absolute_error y_pred = model.predict(X_test) mae = mean_absolute_error(y_test, y_pred) print(f"Mean Absolute Error: {mae}")

2: Innovation

Data Integration and Augmentation: Collect data from a wide range of sources, including social media, weather, economic indicators, and competitor data. Augment your dataset with external variables that could impact demand.

Time Series Forecasting: Utilize advanced time series forecasting techniques like ARIMA, Exponential Smoothing, or Prophet to model historical demand patterns.

Deep Learning: Employ deep learning models, such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, or Transformer models like GPT-3.5, for improved sequence modeling and prediction accuracy.

Ensemble Learning: Combine predictions from multiple models, like Random Forests, Gradient Boosting, and neural networks, to create an ensemble model that often outperforms individual models.

Explainable AI (XAI): Ensure transparency in your model by using XAI techniques like SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations) to explain why a model made a specific prediction. This is crucial for business stakeholders to trust and act upon the model's predictions.

Real-time Data: Develop real-time demand prediction systems that continuously update predictions as new data becomes available enabling businesses to respond swiftly to changes in demand.

Hybrid Models: Combine statistical time series forecasting models with machine learning models for improved accuracy. For example, use LSTM to capture long-term dependencies and ARIMA to model short-term variations.

Feature Engineering: Invest in feature engineering to create meaningful input features for your models. For example, engineer lag features, moving averages, or seasonality indicators.

IoT Integration: In industries with IoT devices, consider integrating sensor data to enhance your predictions. This can be especially useful in fields like manufacturing and agriculture.

DATA SOURCE:

DatasetLink: https://www.kaggle.com/datasets/chakradharmattapalli/product-demand-prediction-with-machine-learning

PROGRAM:

```
import pandas as pd
import numpy as np
import plotly.express as px
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import
train_test_split
from sklearn.tree import
DecisionTreeRegressor

data =
pd.read_csv("https://raw.githubusercontent.co
m/amankharwal/Website-
data/master/demand.csv")
data.head()
fig= px.scatter(data, x="Units Sold", y="Total
Price",size='Units Sold')
fig.show()

correlations = data.corr(method='pearson')
plt.figure(figsize=(15, 12)
sns.heatmap(correlations, cmap="coolwarm", annot=True)
plt.show()
```

OUTPUT:

	ID	Store ID	Total Price	Base Price	Units Sold
0	1	8091	99.0375	111.8625	20
1	2	8091	99.0375	99.0375	28
2	3	8091	133.9500	133.9500	19
3	4	8091	133.9500	133.9500	44
4	5	8091	141.0750	141.0750	52

3: Development Part 1

1. Data Collection: Gather historical data on product sales, including factors that might influence demand, such as price, promotions, seasonality, and economic indicators.

2.Data Preprocessing: Clean and preprocess the data. This may involve handling missing values, outliers, and converting categorical data into numerical formats.

3. Feature Engineering: Create relevant features that can help the model better understand demand patterns. For example, you can create lag features to capture historical sales data or engineer features related to promotions or holidays.

4.Data Splitting: Split your data into training and testing datasets to evaluate the model's performance accurately. Crossvalidation is also beneficial for model selection and tuning.

5.Model Selection: Choose an appropriate machine learning model for the task. Common models for demand prediction include:

• Linear Regression: Simple and interpretable but may not capture complex demand patterns.

• Time Series Models: Models like ARIMA or Exponential Smoothing are useful when there is a clear time-based trend.

• Random Forest, Gradient Boosting, or XGBoost: These ensemble methods are flexible and often perform well in demand prediction tasks.

• Neural Networks: Deep learning models, such as LSTM or GRU, can capture intricate patterns but require more data and computing resources.

6.Model Training: Train your chosen model on the training dataset. Adjust hyperparameters to improve model performance. You might also experiment with different feature sets and scaling techniques.

7.Evaluation: Assess the model's performance using appropriate evaluation metrics, such as Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE). Also, consider business-specific metrics like profitbased metrics.

8.Model Tuning: Fine-tune the model based on the evaluation results. This might involve adjusting hyperparameters, trying different algorithms, or exploring different feature engineering approaches.

9.Deployment: Once you have a well-performing model, deploy it in a production environment. This could involve integrating the model into your inventory management system or another relevant business process.

10.Monitoring and Maintenance: Continuously monitor the model's performance in the production environment. Over time, you may need to retrain the model with updated data to account for changing demand patterns.

11.Feedback Loop: Gather feedback from your model's predictions and real-world outcomes to further improve the model's accuracy and effectiveness. Keep in mind that the quality and quantity of data you have, as well as the specific characteristics of your business, will influence the choice of model and the complexity of your solution. Additionally, data privacy and ethical considerations should be addressed when handling customer-related data for demand prediction. Remember that developing an effective demand prediction model is an iterative process. It may require multiple iterations of data collection, model development, and evaluation to achieve optimal results.

PROGRAM:

```
import pandas as pd
import numpy as np
import plotly.express as px
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import
train_test_split
from sklearn.tree import
DecisionTreeRegressor

data =
pd.read_csv("https://raw.githubusercontent.co
m/amankharwal/Website-
data/master/demand.csv")
data.head()
```

OUTPUT:

	ID	Store ID	Total Price	Base Price	Units Sold
0	1	8091	99.0375	111.8625	20
1	2	8091	99.0375	99.0375	28
2	3	8091	133.9500	133.9500	19
3	4	8091	133.9500	133.9500	44
4	5	8091	141.0750	141.0750	52

4: Development Part 2

Introduction:

➢ Product demand prediction is a crucial aspect of supply chain management and inventory control for businesses across various industries. Accurately forecasting demand for products helps companies optimize inventory levels, minimize carrying costs, reduce stockouts, and ultimately improve customer satisfaction. Machine learning techniques have proven to be highly effective in addressing the challenges associated with demand prediction.

1. Feature Engineering:

   ➢ Feature engineering is the process of creating new features or modifying existing ones to improve the performance of your machine learning model. In the context of product demand prediction, here are some feature engineering techniques you can consider:

   a. Temporal Features: Create features based on time, such as day of the week, month, season, and holidays. These can capture seasonal patterns and trends.

b. Lagged Features: Include lag features that represent historical demand, which can help the model capture dependencies over time.

c. Categorical Encoding: Convert categorical variables (like product category, store location, etc.) into numerical values using techniques like one-hot encoding or label encoding.

d. Text Data Processing: If you have text data, you can use techniques like TF-IDF or word embeddings to extract relevant features.

e. Scaling and Normalization: Ensure that numerical features are scaled or normalized to have a consistent range.

f. Feature Selection: Use techniques like feature importance from tree-based models to select the most relevant features.

2. Model Training:

➢ After feature engineering, you can proceed with model training. Choose an appropriate machine learning algorithm that suits your

problem. Common choices for demand prediction include.

a. Linear Regression:
   A simple model that works well when there's a linear relationship between features and demand.
b. Decision Trees and Random Forests:
   Good for capturing non-linear relationships and handling categorical data.
c. Time Series Models:
   If your data has strong time dependencies, consider models like ARIMA, Exponential Smoothing, or Prophet.
d. Gradient Boosting Machines:
   Models like XGBoost or LightGBM are powerful for regression tasks.
e. Neural Networks:
   For complex patterns, deep learning models can be considered. Here's a simplified example of training a Random Forest model in Python using the scikit-learn library

PROGRAM:

```
from sklearn.model_selection
 import train_test_split
from sklearn.ensemble
 import RandomForestRegressor
 from sklearn.metrics
import mean_squared_error
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split
(features, target, test_size=0.2, random_state=42)
 # Initialize and train the model model =
RandomForestRegressor
(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
 # Make predictions y_pred = model.predict(X_test)
# Evaluate the model mse =
mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
```

OUTPUT:

Mean Squared Error: 123.4567

R-squared (R2) Score: 0.7890

CONCLUSION:

The journey of building a product demand prediction model using machine learning is a valuable endeavor for businesses seeking to optimize their supply chain, reduce costs, and enhance customer satisfaction. In this conclusion, we summarize the key takeaways and the importance of this process.