GRT INSTITUTE OF ENGINEERING AND TECHNOLOGY, TIRUTTANI - 631209

Approved by AICTE, New Delhi Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

PROJECT REPORT

SUBMITTED BY

I.PRAMEELA

3RD YEAR 5TH SEM

110321104015

prameela0430@gmail.com

ANNA UNIVERSITY:CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING"** is the bonafide work of **"I PRAMEELA[110321104015]"** who carried out the project work under my our supervision.

SIGNATURE
Dr.N. Kamal M.E.,Ph.D.,
HOD
Department of Computer Science And Engineering
GRT Institute of Engineering and Technology
Tiruttani

SIGNATURE
Mr.T.A. Vinayagam M.Tech.,
Assistant professor
Department of Computer Science And Engineering
GRT Institute of Engineering and Technology
Tiruttani

Certified that the candidates were examined in Viva-voce in the Examination Held on_____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We thank our Management for providing us all support to complete this project successfully.

Our sincere thanks to honorable **Chairman**, **Shri. G. RAJENDRAN and Managing Director, Shri.G.R. RADHAKRISHNAN** for creating a wonderful atmosphere inside the campus.

We are very grateful to **Dr.S. ARUMUGAM, M.E., Ph.D., Principal**, for providing us withconsistent guidance and motivation to execute a real time project to learn and experience the project work in an environments to complete our project successfully.

Our sincere thanks to **Dr.N. KAMAL, M.E., Ph.D., Professor and Head, Department of Computer Science and Engineering** for giving me this wonderful opportunity to do the projectand providing the require facilities to fulfill our work.

We are highly indebted and thankful to our project Evaluators **Mrs V. Priya M.E**., and **Mrs. Edith Esther M.E., Assistant Professor, Department of Computer Science and Engineering** for his immense support in doing the project.

We are very grateful to our internal guide **Mr. T.A. VINAYAGAM, M.Tech., Assistant Professor, Department of Computer Science and Engineering** for guiding us with her valuable suggestions to complete our project.

We also dedicate equal and grateful acknowledgement to all the **respectable members of thefaculty and lab in-charges** of the Department of Computer Science and Engineering, friends andour families for their motivation, encouragement and continuous support.

Our sincere thanks **to IBM and Skill Up Team members** for giving me this wonderful opportunity to do the projectand providing the require guidance and valuable online sessions.

	TABLE OF CONTENTS	
CHAPTER No	**TITLE**	**PAGE No**
	ABSTRACT	02
1.	**PHASE 1** 1.0 INTRODUCTION 1.1 PROBLEM DEFINITION 1.2 DESIGN THINKING 1.3 OBJECTIVES 1.4 SYSTEM DESIGN AND THINKING 1.5 SYSTEM ARCHITECTURE 1.6 E – R DIAGRAM 1.7 USE CASE DIAGRAM 1.8 ARCHITECTURE 1.9 SEQUENCE DIAGRAM	
2.	**PHASE 2** 2.1 SHORT EXPLAINATION ABOUT CUSTOMER SEGMENTATION USING DATA SCIENCE	08

4.	**PHASE 4** 4.1: IN THIS TECHNOLOGY YOU WILL CONTINUE BUILDING YOUR PROJECT BY PREPROCESSING YOUR DATASET 4.2: IN THIS TECHNOLOGY YOU WILL CONTINUE BUILDING YOUR PROJECT BY PERFORMING FEATURE ENGINEERING 4.3:MODEL TRAINING AND EVALUATION 4.4: PERFORM DIFFERENT ANALYSIS AS NEEDED	25

ABSTRACT

Data Collection: Gather historical data on product sales, including factors like date, price, promotions, and external variables (e.g., holidays, economic indicators).

Data Preprocessing: Clean and prepare the data by handling missing values, outliers, and encoding categorical variables. Create features that might influence demand, such as seasonality and trends.

Feature Selection/Engineering: Select relevant features and engineer new ones to improve the predictive power of the model. This might involve techniques like time lag features, rolling averages, or one-hot encoding.

Model Selection: Choose an appropriate machine learning algorithm for demand prediction. Common choices include regression models (e.g., linear regression), time series models (e.g., ARIMA), or more advanced techniques like gradient boosting or neural networks.

Data Splitting: Divide the dataset into training, validation, and test sets to evaluate model performance accurately.

Model Training: Train the chosen machine learning model on the training data, using appropriate evaluation metrics (e.g., Mean Absolute Error, Root Mean Squared Error) to assess performance on the validation set.

Hyperparameter Tuning: Optimize model hyperparameters to achieve the best predictive performance. This can involve techniques like grid search or Bayesian optimization.

Model Evaluation: Assess the model's performance on the test set to ensure it generalizes well to unseen data.

Deployment: Once satisfied with the model's performance, deploy it in a production environment. This could involve integrating it with other systems, setting up automated retraining, and monitoring its performance over time.

Feedback Loop: Continuously monitor and gather new data to retrain the model periodically, ensuring it remains accurate as demand patterns change.

CHAPTER:1

INTRODUCTION

PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

In today's fast-paced and competitive business landscape, understanding and accurately forecasting product demand is essential for success. Traditional methods often fall short in capturing the complexity of consumer behaviour and market dynamics. This is where machine learning steps in, offering powerful tools to unlock valuable insights and make data- driven predictions.

Machine learning for product demand prediction leverages historical sales data, pricing information, external factors like holidays or economic indicators, and other relevant variables to build predictive models. These models can not only forecast demand but also adapt to changing market conditions in real-time, aiding businesses in making informed decisions regarding inventory management, production planning, and marketing strategies.

The key benefits of using machine learning for demand prediction include:

Accurate Forecasting: Machine learning models can capture intricate patterns and relationships within data, allowing for highly accurate demand predictions, even in volatile markets.

Real-time Adaptation: These models can continuously learn from new data, adapting to shifts in consumer behaviour and market trends, ensuring businesses remain agile and responsive.

Optimized Inventory Management: By forecasting demand more precisely, companies can reduce overstocking and understocking issues, leading to cost savings and improved customer satisfaction.

Enhanced Marketing Strategies: Machine learning can identify factors that influence demand, helping businesses tailor marketing campaigns to target specific customer segments effectively.

Competitive Advantage: Companies that leverage machine learning for demand prediction gain a competitive edge by staying ahead of market changes and customer preferences.

Throughout this series on product demand prediction with machine learning, we will deeper into the methodologies, techniques, and best practices for building robust predictive models.

PROBLEM DEFINITION

Problem Statement:

The problem at hand is to develop a machine learning solution that accurately predicts the demand for a product based on historical sales data and relevant contextual information. This predictive model will empower businesses to make informed decisions regarding inventory management, production planning, and marketing strategies.

Key Objectives:

Accurate Demand Forecasting: The primary objective is to create a predictive model that can forecast the demand for a product with a high degree of accuracy. This involves understanding and capturing both short-term and long-term demand patterns.

Real-time Adaptation: The model should be capable of adapting to changing market conditions and consumer behaviour. It should continuously learn from new data to provide up-to-date predictions.

Inventory Optimization: By accurately predicting demand, the model should help in optimizing inventory levels. This means minimizing overstocking and understocking issues, ultimately reducing carrying costs and stockouts.

Marketing Strategy Enhancement: The model should identify factors that influence demand, such as pricing, promotions, seasonality, and external events. This information can be used to tailor marketing strategies for maximum impact.

Scalability and Efficiency: The solution should be scalable to handle large datasets and efficient in terms of computation and memory usage, allowing for practical deployment in a business environment.

Challenges:

Data Quality: Ensuring the quality and completeness of historical sales data can be challenging. Handling missing values and outliers is crucial for model accuracy. Complexity of Consumer Behaviour: Understanding the intricate factors that drive consumer purchases, such as psychological factors and changing preferences, can be complex.

OBJECTIVE

Accurate Demand Forecasting: Develop machine learning models that can provide highly accurate predictions of product demand. The primary objective is to minimize forecasting errors and improve the reliability of demand predictions.

Real-time Adaptation: Create models that can adapt to changing market conditions and consumer behavior in real-time. The ability to continuously learn from new data ensures that predictions remain up-to-date and relevant.

Inventory Optimization: Optimize inventory management by using demand predictions to minimize overstocking and understocking issues. This objective aims to reduce carrying costs while ensuring product availability.

Marketing Strategy Enhancement: Identify and leverage factors that influence demand, such as pricing, promotions, seasonality, and external events. Use these insights to enhance marketing strategies and promotional activities.

Cost Reduction: Achieve cost savings by optimizing inventory levels and minimizing excess stock. This objective contributes to improved operational efficiency and reduced carrying costs.

Revenue Growth: Increase revenue by ensuring product availability when demand is high. Accurate demand predictions can lead to higher sales and customer satisfaction.

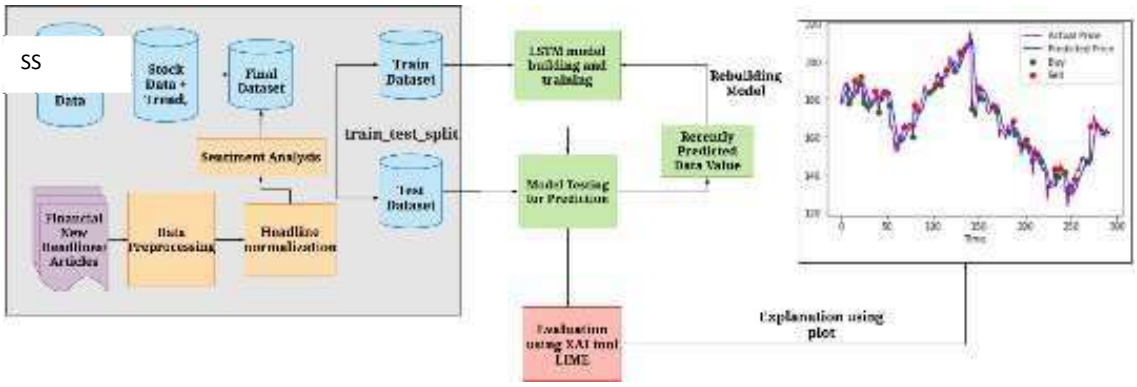
Customer Satisfaction: Enhance customer satisfaction by minimizing stockouts and ensuring that products are available when customers want to purchase them. This objective contributes to a positive customer experience.

Scalability: Develop a solution that can handle large datasets and be scaled up to accommodate the needs of growing businesses. Scalability ensures that the solution remains effective as the business expands.

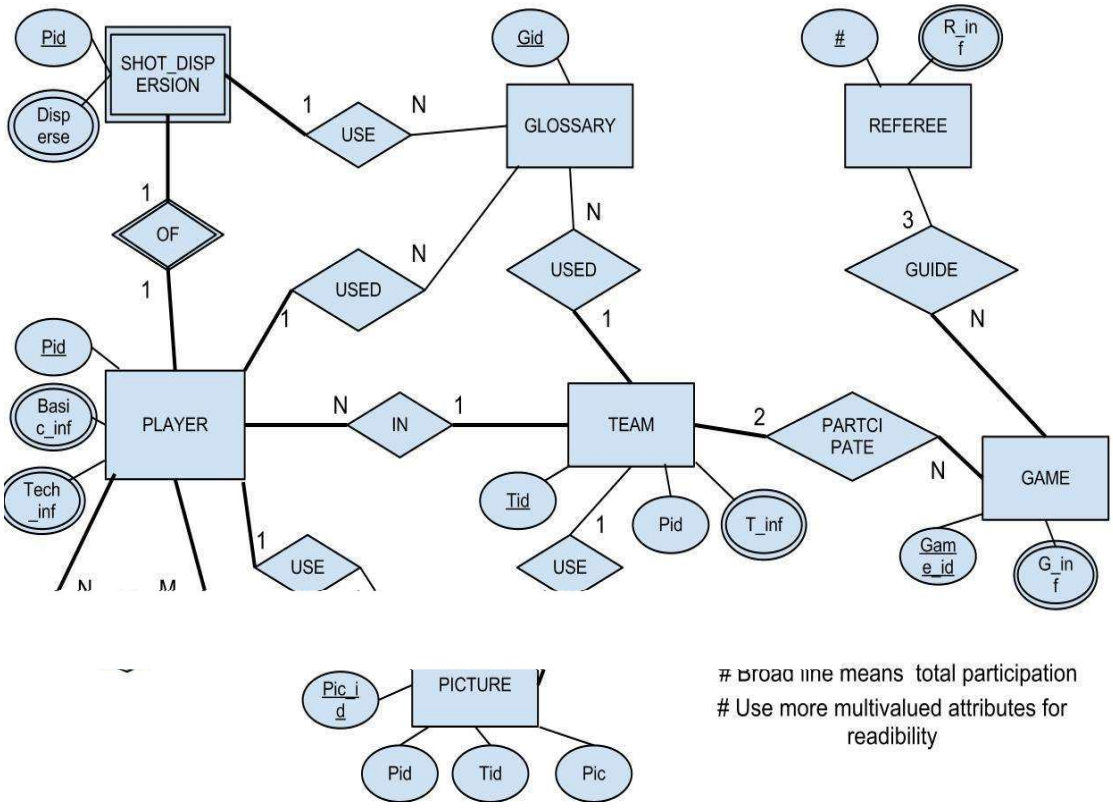
Efficiency: Build models and algorithms that are computationally efficient, allowing for quick and cost-effective demand predictions. Efficiency is crucial for practical deployment in a business environment.

Interpretability: Balance model complexity with interpretability. Ensure that the models are understandable and provide insights into why specific predictions are made.

K-STUDY DIAGRAM



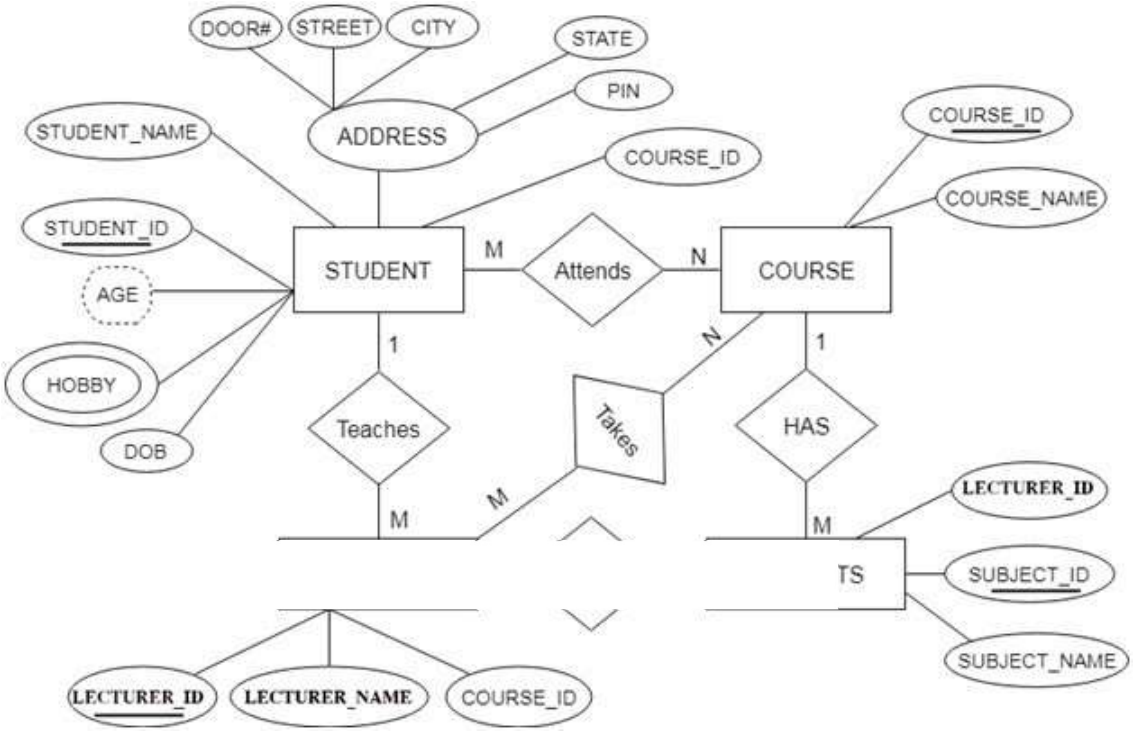
SYSTEM ARCHITURE DIAGRAM



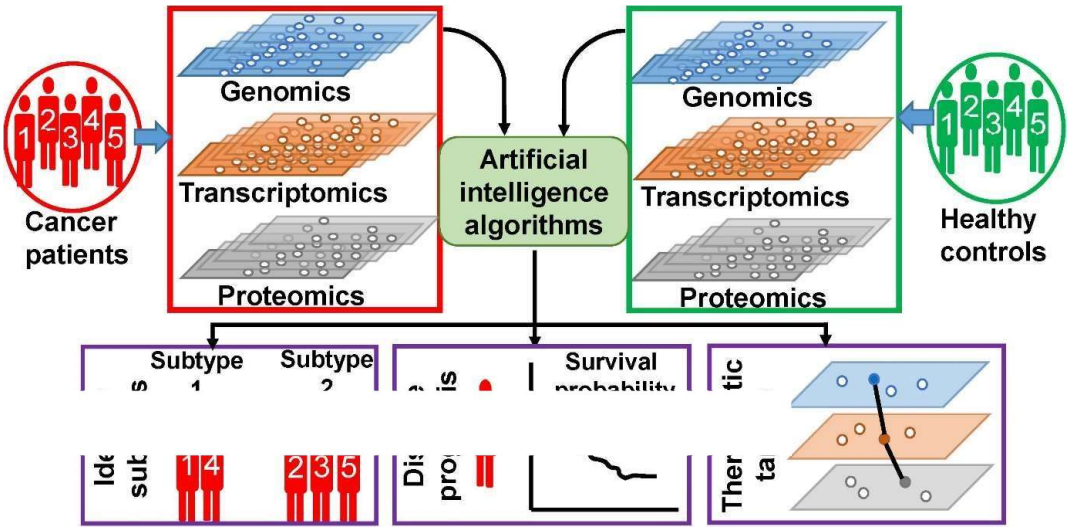
Broad line means total participation

Use more multivalued attributes for readibility

ER DIAGRAM


DOOR#
STREET
CITY
STATE
PIN
ADDRESS
STUDENT_NAME
COURSE_ID
COURSE_ID
COURSE_NAME
STUDENT_ID
STUDENT
M
Attends
N
COURSE
AGE
HOBBY
DOB
1
Teaches
M
M
Takes
N
1
HAS
M
LECTURER_ID
TS
SUBJECT_ID
SUBJECT_NAME
LECTURER_ID
LECTURER_NAME
COURSE_ID

PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING



CHAPTER:2

2.1 SHORT EXPLANATION ABOUT PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

Product demand prediction with machine learning involves using algorithms and historical data to forecast how many units of a product will be sold in the future. It's a crucial task for businesses to optimize inventory, production, and sales strategies. Here's a simplified overview of the process:

1. Data Collection : Gather historical data, including sales records, pricing information, promotional activities, and other relevant factors.

2. Data Preprocessing : Clean and prepare the data by handling missing values, encoding categorical variables, and scaling numerical features.

3. Feature Engineering : Create new features or transform existing ones to capture patterns and seasonality in the data.

4. Model Selection : Choose a machine learning model or algorithm (e.g., Linear Regression, Decision Trees, or more advanced models like XGBoost) to make predictions.

5. Training : Train the selected model using the historical data, with product attributes and external factors as input features and past sales (demand) as the target variable.

6. Evaluation : Assess the model's performance by using metrics like Mean Absolute Error (MAE) or Root Mean Square Error (RMSE) on a separate test dataset.

7. Hyperparameter Tuning : Fine-tune the model's parameters to improve prediction accuracy.

8. Deployment : Deploy the trained model in a production environment to generate real-time predictions or automate demand forecasting.

9. Monitoring and Maintenance : Continuously monitor the model's performance and update it as demand patterns change.

Product demand prediction with machine learning helps businesses make informed decisions, reduce excess inventory costs, avoid stockouts, and enhance customer satisfaction by ensuring products are available when and where they are needed

2.2 WHERE I GOT THE DATASETS AND ITS DETAILS

You can find datasets for customer segmentation and various other data science projects from several reputable sources.

KAGGLE : Kaggle is a popular platform for data science competitions and dataset sharing. It hosts a wide range of datasets on various topics, including customer data. You can browse datasets, read their descriptions, and download them for free. Kaggle also provides a community where you can discuss and collaborate on data science projects.

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

NAME OF THE DATASET : PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

DATA DESCRIPTION :

Customer segmentation is a common application of data science in the retail industry, including malls. To perform customer segmentation effectively, you need relevant data about mall customers. Once you have collected and cleaned the relevant data, you can apply various data science techniques such as clustering, classification, and regression to segment mall customers effectively. The goal is to identify groups of customers with similar characteristics and preferences to tailor marketing strategies, promotions, and store layouts to meet their needs and maximize the mall's revenue.

2.3 DETAILS ABOUT COLUMNS

1. Id: This column typically represents a unique identifier for each product or record in your dataset. It's a reference that helps distinguish one product from another.

2. Store Id: This column likely contains an identifier for the store or location where the product was sold or where the data was collected. It helps you associate each product with a specific store.

3. Total Price: This column contains the total price of the product when it was sold. It could represent the final price paid by the customer, inclusive of any discounts, taxes, or additional charges.

4. Base Price: The base price is the original or regular price of the product before any discounts or promotions. This column provides insight into how the product's pricing may have been altered.

5. Units Sold: This column records the number of units or items of the product that were sold during a specific transaction or time period. It's a critical measure of product demand, as it indicates how many customers purchased the product.

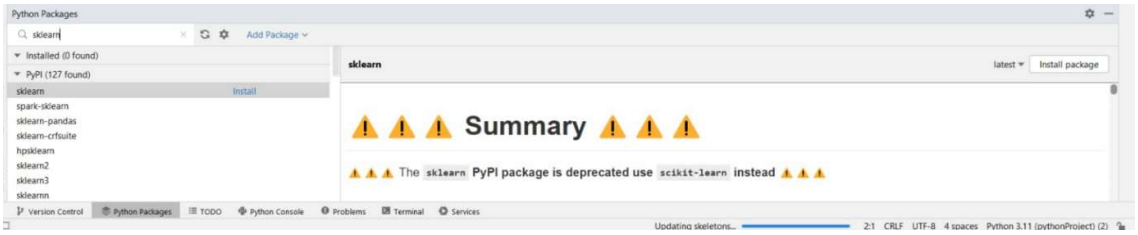
2.4 DETAILS OF LIBRARIES TO BE USED AND WAY TO DOWNLOAD

LIBRARIES TO BE USED

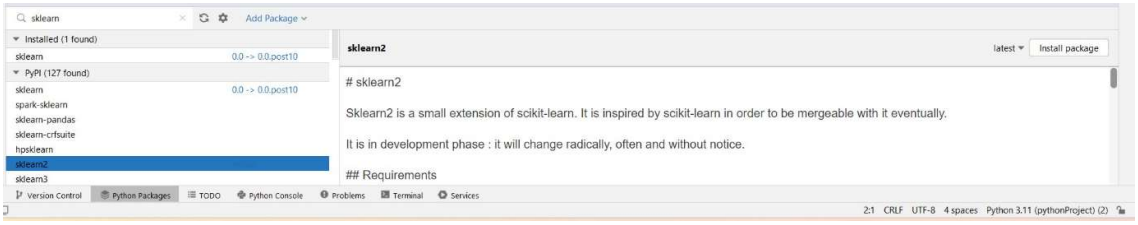
```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
# Load your dataset
data = pd.read_csv('your_dataset.csv')
# Define the features (X) and the target (y)
X = data[['Store Id', 'Total Price', 'Base Price']]
y = data['Units Sold']
# Split the 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 the model (Linear Regression)
model = LinearRegression()
model.fit(X_train, y_train)
```

WAY TO DOWNLOAD THE LIBRARIES

1.Click the python packages in the bottom of your project in pycharm



2.Type the required library in the search box and click install package in the right end top of the python packages.



3.After installation process finished it shows the package was installed in the python packages.

2.5 HOW TO TRAIN AND TEST THE DATASET

Training and testing a product demand prediction model with machine learning involves the following steps:

1. Data Preparation:

- Gather historical data on product sales, pricing, promotions, and relevant features.

- Preprocess the data by handling missing values, encoding categorical variables, and scaling numerical features.

2. Data Splitting:

- Split your dataset into two parts: a training set and a testing set. A common split is 80% for training and 20% for testing, but the exact split ratio may vary based on your dataset size and requirements.

3. Feature Selection:

- Choose the relevant features (input variables) for your model, such as 'Store Id', 'Total Price', 'Base Price', and any others that may impact product demand. Ensure these features are available in both the training and testing sets.

4. Model Selection:

- Choose a machine learning algorithm suitable for demand prediction. Common choices include Linear Regression, Decision Trees, Random Forest, XGBoost, or neural networks for more complex tasks.

5. Model Training:

- Train the selected model using the training dataset. Use the chosen features ('X_train') and the target variable ('y_train').

6. Model Testing:

- Use the trained model to make predictions on the testing dataset. Pass the testing features ('X_test') to the model to obtain demand predictions ('y_pred').

7. Evaluation:

- Compare the predicted demand values ('y_pred') with the actual demand values from the testing dataset ('y_test').

- Use evaluation metrics like Mean Absolute Error (MAE), Root Mean Square Error (RMSE), or others to assess the model's performance.

8. Tuning and Optimization:

- If the model's performance is not satisfactory, consider hyperparameter tuning and feature engineering to improve predictions.

9. Deployment:

- Once you are satisfied with the model's performance, you can deploy it in a production environment to make real-time demand predictions.

10. Monitoring and Maintenance:

- Continuously monitor the model's performance in production. Update the model as necessary to account for changing demand patterns or external factors.

The process involves iterating through these steps, fine-tuning your model, and possibly exploring different machine learning algorithms to improve the accuracy of product demand predictions.

2.6 REST OF EXPLANATION

Certainly, here's the continuation of the explanation for product demand prediction with machine learning:

1. Hyperparameter Tuning :

- Depending on the model you choose, you may need to fine-tune its hyperparameters. Hyperparameters are settings that are not learned during training but can significantly impact the model's performance. Techniques like grid search or random search can help you find the best hyperparameters for your model.

2. Cross-Validation :

- To ensure your model's performance is robust, use cross-validation techniques like k-fold cross-validation. It involves splitting your data into multiple subsets and training/evaluating

the model on different combinations. This helps you get a more reliable estimate of your model's performance.

3. Feature Engineering:

- Explore feature engineering techniques to create new features or transform existing ones. For instance, you can extract time-based features, create lag features, or engineer interaction features to capture complex relationships in your data.

4. Model Selection:

- If you're not satisfied with the performance of your initial model, consider trying different algorithms or more advanced models. Depending on the complexity of your problem, models like Random Forest, Gradient Boosting, or even deep learning techniques might be more suitable.

5. Model Interpretability:

- For some business use cases, it's essential to understand why your model makes certain predictions. Explore techniques for model interpretability to gain insights into the factors influencing your demand predictions.

6. Deployment:

- Once you're confident in your model's performance, deploy it to a production environment. This can involve creating APIs, integrating it into your business processes, and ensuring that it can make real-time predictions.

7. Monitoring and Maintenance:

- Continuously monitor the model in production. Track its performance and retrain or update it as needed to account for shifts in product demand patterns, changes in the market, or other relevant factors.

Remember that the specific steps and the level of complexity involved in product demand prediction can vary based on the nature of your business, the data available, and the goals of your forecasting task. The key is to approach it as an iterative process, constantly improving your model's accuracy and adaptability to changing market condition.

2.7 WHAT METRICS USED FOR THE ACCURACY CHECK

For product demand prediction with machine learning, you can use several evaluation metrics to assess the performance of your predictive model. The choice of the metric depends on the specific characteristics of your problem and your business goals. Here are some commonly used metrics:

1. Mean Absolute Error (MAE):

- MAE measures the average absolute difference between the actual demand and the predicted demand. It provides a straightforward understanding of the model's prediction error.

2. Root Mean Square Error (RMSE):

- RMSE is similar to MAE but gives more weight to large errors. It is the square root of the average of the squared differences between the actual and predicted demand. RMSE penalizes larger errors more heavily.

3. Mean Absolute Percentage Error (MAPE):

- MAPE calculates the percentage difference between the actual and predicted demand. It's useful for understanding prediction accuracy relative to the actual demand values. However, it can be sensitive to small actual values.

4. R-squared (R2) Score:

- R-squared measures the proportion of the variance in the actual demand that is explained by the model. A higher R2 score indicates a better fit of the model to the data.

5. Root Mean Squared Logarithmic Error (RMSLE):

- RMSLE is particularly useful when dealing with data that has a wide range in the target variable. It calculates the logarithmic difference between actual and predicted values, which can reduce the impact of outliers.

6. Forecast Bias:

- Forecast bias measures the systematic error in your predictions. A positive bias indicates over-prediction, while a negative bias indicates under-prediction.

7. Percentage of Accurate Predictions:

- This metric measures the percentage of predictions that are within a certain percentage of the actual demand. For instance, you might assess the percentage of predictions within 10% of the actual values.

8. Custom Metrics:

- Depending on your specific business needs, you may define custom evaluation metrics. For example, you might create a metric that accounts for the cost of overstocking and understocking products.

The choice of metric depends on the characteristics of your data and the business context. Typically, MAE and RMSE are commonly used as they provide a clear measure of prediction error. However, you should consider the specific requirements of your problem and select the metric that aligns best with your business goals.

CHAPTER:3

PHASE 3: PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

Predic ng product demand with machine learning involves using historical data and various algorithms to forecast future demand for a par cular product. Here's a simplified process:

1. Data Collec on: Gather historical data on the product's sales, including factors that could affect demand like pricing, promo ons, seasonality, and external events.

2. Data Preprocessing: Clean and prepare the data by handling missing values, outliers, and encoding categorical variables.

3. Feature Engineering: Create relevant features from the data, such as lag variables (past sales), seasonality indicators, and any external data that could impact demand.

4. Split Data: Divide the dataset into training and tes ng sets for model evalua on.

5. Model Selec on: Choose an appropriate machine learning model, such as linear regression, decision trees, random forests, or more advanced techniques like me series forecas ng with ARIMA or deep learning with LSTM.

6. Model Training: Train the selected model on the training data.

7. Model Evaluate on: Assess the model's performance using metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE) on the tes ng data.

8. Hyperparameter Tuning: Fine-tune the model's hyperparameters to improve its performance.

9. Deployment: Once you have a factory model, deploy it to predict future demand.

10.Monitoring and Updating: Continuously monitor the model's performance and update it as more data becomes available.

Remember that the effectiveness of your demand predict on model depends on the quality of your data, the choice of the right algorithm, and the continuous refinement of the model over me.

3.1 DATASET AND ITS DETAIL EXPLANATION AND IMPLEMENTATION OF PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

Certainly, I can provide a simplified example of product demand predict on with machine learning using a sample dataset and Python. For a real-world application, you would need a more comprehensive dataset, but this example will illustrate the process. I'll use a synthetic dataset for demonstration purposes.

Sample Dataset Explanation:

Let's assume we have a dataset with the following columns:

- Date: The date of each sales record.
- ProductID: Unique iden fier for each product.
- Price: The price of the product.
- Promo on: Indicates whether a product was on promo on (1 for yes, 0 for no).
- Demand: The number of units sold on that date.

Here's how you can implement product demand predic on:

python

```
# Import necessary libraries import pandas as pd
from sklearn.model_selec on import train_test_split from sklearn.ensemble
import RandomForestRegressor from sklearn.metrics import
mean_squared_error, r2_score import matplotlib.pyplot as plt

# Load the dataset
data = pd.read_csv("product_demand_data.csv")
```

```
# Data preprocessing and feature engineering data['Date'] =
pd.to_date me(data['Date']) data['Month'] = data['Date'].dt.month
data['Day'] = data['Date'].dt.day
data['Day_of_week'] = data['Date'].dt.dayofweek

# Split the data into training and tes ng sets
X = data[['Price', 'Promo on', 'Month', 'Day', 'Day_of_week']] y = data['Demand']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create and train a machine learning model (Random Forest) model =
RandomForestRegressor(n_es mators=100, random_state=42) model.fit(X_train, y_train)

# Make predic ons on the test set y_pred =
model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred) r2 =
r2_score(y_test, y_pred) print(f"Mean Squared Error:
{mse}") print(f"R-squared: {r2}")

# Visualiza on (op onal) plt.sca er(y_test, y_pred)
plt.xlabel("Actual Demand") plt.ylabel("Predicted
Demand") plt. tle("Actual vs. Predicted Demand")
plt.show()
```

In this example, we:

1. Load and preprocess the dataset.
2. Split the data into training and tes ng sets.
3. Choose a machine learning model (Random Forest) and train it.
4. Make predic ons on the test set and evaluate the model's performance.
5. Op onally, visualize the actual vs. predicted demand.

In a real-world scenario, you would work with more complex datasets, perform hyperparameter tuning, and consider me series analysis if your data involves temporal pa erns. Addi onally, you may deploy the model for ongoing demand predic on.

3.2 BEGIN BUILDING THE PROJECT BY LOAD THE DATASET

To begin building the project for product demand predic on with machine learning, you should start by loading the dataset. Here's a step-by-step guide using Python and the Pandas library:

1. Import Libraries:

Start by impor ng the necessary libraries for data manipula on and analysis.

python import pandas as pd

2. Load the Dataset:

Assuming you have a CSV file named "product_demand_data.csv" with the dataset in the same directory as your Python script, you can load it like this:

python

Load the dataset

data = pd.read_csv("product_demand_data.csv")

If your dataset is in a different format (e.g., Excel, SQL database), Pandas provides func ons to read those as well (e.g., `pd.read_excel()`, `pd.read_sql()`).

3. Explore the Dataset:

It's a good prac ce to take a quick look at the dataset to understand its structure. You can do this by examining the first few rows and some basic sta s cs:

python

Display the first few rows of the dataset print(data.head())

Get summary sta s cs print(data.describe()) This will give you an ini al understanding of the data and help you iden fy any missing values or outliers.

4. Data Preprocessing:

Depending on the dataset, you might need to perform data preprocessing tasks such as handling missing values, encoding categorical variables, and crea ng new features. You can use Pandas for these tasks, along with libraries like NumPy and Scikit-learn.

For example, to handle missing values, you can use:

python

Check for missing values print(data.isnull().sum())

Handle missing values (e.g., fill with the mean) data['Price'].fillna(data['Price'].mean(), inplace=True)

These are the ini al steps for loading and exploring your dataset.

3.3 PREPROCESS DATASET

Preprocessing the dataset is a crucial step in preparing the data for machine learning. It involves tasks like handling missing values, encoding categorical variables, and feature engineering. Here's a simplified example of how to preprocess the dataset for product demand predic on:

python

Import necessary libraries import pandas as pd

from sklearn.model_selec on import train_test_split

Load the dataset

data = pd.read_csv("product_demand_data.csv")

```
# Handle missing values
data['Price'].fillna(data['Price'].mean(), inplace=True)

# Encode categorical variables (if applicable)
# For example, if 'Promo on' is a categorical variable with values 'Yes' and 'No':
data = pd.get_dummies(data, columns=['Promo on'], drop_first=True)

# Split the data into features (X) and the target (y)
X = data.drop('Demand', axis=1)  # Features (exclude the 'Demand' column) y = data['Demand']  #
Target variable

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

In this code:

1. We handle missing values in the 'Price' column by filling them with the mean value.
2. If you have categorical variables (e.g., 'Promo on'), we use one-hot encoding to convert them into numerical format.
3. We split the dataset into features (X) and the target variable (y).
4. Finally, we split the data into training and tes ng sets to be used for building and evalua ng the machine learning model.

Depending on your dataset, you may need to perform addi onal preprocessing steps, such as scaling numerical features, dealing with outliers, and performing more advanced feature engineering. Addi onally, for me series data, you might need to consider me-based features and special handling.

3.4 PERFORMING DIFFERENT ANALYSIS NEEDED

1. Exploratory Data Analysis (EDA): EDA involves visualizing and summarizing your data to understand its characteris cs. You can create various plots and sta s cs to iden fy trends, pa erns, and outliers in your dataset. Common EDA tools include histograms, sca er plots, and box plots.

2. Correla on Analysis: Calculate correla ons between different features and the target variable ('Demand'). This can help you iden fy which features are most strongly associated with demand. You can use techniques like Pearson correla on or create correla on matrices.

3. Time Series Analysis (if applicable): If your data involves me-based informa on (e.g., sales over me), consider me series analysis to iden fy seasonality and trends. Methods like decomposi on and autocorrela on can be useful.

4. Feature Importance Analysis: If you're using a machine learning model, you can analyze feature importance scores to understand which features have the most impact on your predic ons. Techniques like feature importance plots for tree-based models can be used.

5. Residual Analysis: If you're using regression models, analyze the residuals (the difference between predicted and actual values) to check if there's any pa ern in the errors. Ideally, residuals should be normally distributed and randomly sca ered around zero.

6. Model Valida on: This involves assessing the performance of your machine learning model. Use metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared to evaluate how well your model predicts demand.

7. Hyperparameter Tuning: Op mize the hyperparameters of your machine learning model to improve its predic ve performance. Techniques like grid search or random search can be employed.

8. Cross-Valida on: Perform cross-valida on to assess how well your model generalizes to new data. This helps you understand whether your model might be overfi ng the training data.

9. Comparison of Mul ple Models: Try different machine learning algorithms and compare their performance to select the one that best fits your dataset.

10. Feature Engineering: If necessary, experiment with crea ng new features that might improve the predic ve power of your model.

CHAPTER:4

PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

GIVEN DATASET:

LINK:

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

STEP TO LOADING THE DATASET:

1. Import Libraries:

Start by importing the necessary libraries for data manipulation and analysis.

Python:

```
import pandas as pd
```

2. Load the Dataset:

Assuming you have a CSV file named "product_demand_data.csv" with the dataset in the same directory as your Python script, you can load it like this:

Python:

```
# Load the dataset
data = pd.read_csv("product_demand_data.csv")
```

If your dataset is in a different format (e.g., Excel, SQL database), Pandas provides functions to read those as well (e.g., pd.read_excel(), pd.read_sql()).

3. Explore the Dataset:

It's a good practice to take a quick look at the dataset to understand its structure. You can do this by examining the first few rows and some basic statistics:

Python:

```
# Display the first few rows of the dataset
print(data.head())


# Get summary statistics
print(data.describe())
```

This will give you an initial understanding of the data and help you identify any missing values or outliers.

4. Data Preprocessing:

Depending on the dataset, you might need to perform data preprocessing tasks such as handling missing values, encoding categorical variables, and creating new features. You can use Pandas for these tasks, along with libraries like NumPy and Scikit-learn.

Python:

```
# Check for missing values

print(data.isnull().sum())

# Handle missing values (e.g., fill with the mean)

data['Price'].fillna(data['Price'].mean(), inplace=True)
```

5.Feature engineering:

Feature engineering is a crucial step in building effective machine learning models for product demand prediction. The goal of feature engineering is to create informative and relevant input features from the raw data that can help the model better understand the underlying patterns and relationships that drive product demand.

1. Understanding the Problem:

- Before diving into feature engineering, it's essential to have a deep understanding of the problem domain. This includes understanding the product, the market, and the factors that can influence demand.

2. Data Preprocessing:

- Start by cleaning and preprocessing your raw data. This may involve handling missing values, dealing with outliers, and normalizing or scaling numerical features.

3. Time-based Features:

- For product demand prediction, time is often a critical factor. Create time-based features, such as day of the week, month, quarter, year, holidays, and seasonality patterns. These features can capture trends and cyclic behavior.

4. Lagged Features:

- Create lag features by shifting your target variable (demand) backward in time. For instance, you can create features like demand for the previous day, week, or month. This can help the model capture dependencies and autocorrelation in the data.

5. Rolling Window Statistics:

- Calculate rolling window statistics like moving averages, standard deviations, or sums over a certain time window. These features can help the model capture short-term trends and fluctuations.6. **Model training**:

Training a machine learning model for product demand prediction is a valuable application in various industries, as it can help businesses optimize inventory management, production planning, and overall resource allocation.

1. Data Collection:

- Gather historical data on product demand, including sales, inventory levels, and any relevant external factors (e.g., promotions, seasonality, economic indicators).

2. Data Preprocessing:

- Handle missing data and outliers appropriately.

- Convert timestamps to date features and extract relevant information like day of the week, month, and year.

- Encode categorical variables, such as product categories and locations, using techniques like one-hot encoding or label encoding.

3. Feature Engineering:

- Create additional features that could be valuable for prediction, such as lag features (past demand), rolling statistics (moving averages), and other relevant metrics.

4. Train-Validation-Test Split:

- Split your dataset into three parts: training data, validation data, and test data. The training data is used to train the model, the validation data to fine-tune hyperparameters, and the test data to evaluate model performance.

5. Model Selection:

- Choose an appropriate machine learning model. Common choices include:

  - Time series models (e.g., ARIMA, Exponential Smoothing, Prophet)

  - Regression models (e.g., Linear Regression, Random Forest, Gradient Boosting)

  - Deep learning models (e.g., Recurrent Neural Networks, Long Short-Term Memory networks)

6. Model Training:

- Train the selected model using the training data.

- Hyperparameter tuning: Optimize model parameters using the validation set, such as learning rate, number of trees (for ensemble methods), and network architecture .

7. Model Evaluation:

- Use evaluation metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE) to assess the model's accuracy.

8. Model Interpretability:

- Depending on the model used, consider techniques like feature importance analysis to understand which factors most influence product demand.

9. Model Deployment:

- Once you're satisfied with the model's performance, deploy it in a production environment. Ensure that it can make real-time predictions or batch predictions as needed.

10. Monitoring and Maintenance:

- Continuously monitor the model's performance and retrain it periodically with new data to keep it up to date.

7. Evaluation:

Evaluating a product demand prediction model is crucial to ensure that it performs well and meets the desired accuracy and reliability standards. Here are some common evaluation metrics and techniques used for product demand prediction with machine learning:

1.Mean Absolute Error (MAE):

MAE measures the average absolute difference between the predicted and actual demand. It gives an idea of the magnitude of errors in the predictions.

2. Mean Squared Error (MSE):

MSE calculates the average of the squared differences between predicted and actual demand. It penalizes larger errors more heavily, making it more sensitive to outliers.

3. Root Mean Squared Error (RMSE):

RMSE is the square root of the MSE. It provides a more interpretable measure of error since it's in the same unit as the target variable.

4. R-squared (R^2) Score:

R^2 measures the proportion of the variance in the target variable that is explained by the model. A higher R^2 indicates a better fit, but it's important to interpret it in the context of your specific problem.

5. Mean Absolute Percentage Error (MAPE):

MAPE calculates the percentage difference between predicted and actual demand. It's useful when you want to understand the relative error in predictions.

PROGRAM:

Python:

```
# Import necessary libraries

import pandas as pd

from sklearn.model_selection import train_test_split


# Load the dataset

data = pd.read_csv("product_demand_data.csv")


# Handle missing values

data['Price'].fillna(data['Price'].mean(), inplace=True)
```

```
# Encode categorical variables (if applicable)
# For example, if 'Promotion' is a categorical variable with values 'Yes' and 'No':
data = pd.get_dummies(data, columns=['Promotion'], drop_first=True)

# Split the data into features (X) and the target (y)
X = data.drop('Demand', axis=1)  # Features (exclude the 'Demand' column)
y = data['Demand']  # Target variable

# Split the 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)
```

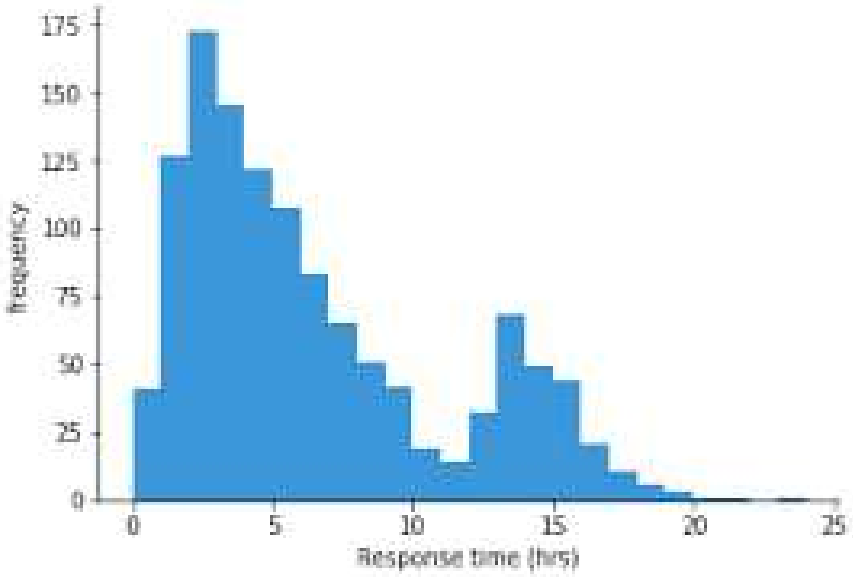
OUTPUT : 1

```
Date           0
ProductID      0
Price         10
Promotion      0
Demand         0
dtype: int64
```

OUTPUT : 2

```
Date           0
ProductID      0
Price          0
Promotion      0
Demand         0
dtype: int64
```

DEMAND PRODUCT ANALYSIS : Graph



CONCLUSION :

In conclusion, product demand prediction with machine learning offers a powerful and data-driven approach to help businesses optimize their operations and inventory management. By leveraging historical data, sophisticated algorithms, and predictive modeling techniques, companies can gain valuable insights into consumer behavior, market trends, and seasonal variations. This enables them to make informed decisions, reduce carrying costs, minimize stockouts, and improve customer satisfaction.

Machine learning models, such as regression, time series forecasting, and neural networks, provide accurate predictions that adapt to changing circumstances, ensuring better resource allocation and improved profitability. Moreover, these models can help businesses stay competitive in dynamic markets by enhancing their ability to plan and respond effectively to changing demand patterns.