

## **Project 10: Product Demand Prediction with Machine Learnings**

### **Data Collection:**

Collecting data for demand prediction in machine learning involves gathering relevant information that can be used to forecast the future demand for a product. Here's a breakdown of how and what data you can collect for this purpose:



#### **1. Historical Sales Data:**

- This is the most critical dataset. It includes records of past sales transactions, typically with columns such as:
  - Date of sale
  - Product ID or SKU
  - Quantity sold
  - Sales price
  - Customer ID (if applicable)

#### **2. Product Data:**

- Information about the products being sold, including attributes like:
  - Product category
  - Product description
  - Product size
  - Product color
  - Brand name
  - Supplier or manufacturer details

#### **3. Market Data:**

- Data on external factors that can affect demand, such as:
  - Economic indicators (e.g., GDP, inflation rate)
  - Competitor prices and promotions

- Market trends and industry reports
- Seasonal patterns and holidays
- Weather conditions (especially for weather-sensitive products)

#### **4. Customer Data:**

- Information about your customers can help you understand their buying behavior:
  - Customer demographics (age, gender, location)
  - Customer segmentation data
  - Customer purchase history
  - Customer loyalty program data

#### **5. Marketing and Promotion Data:**

- Details about marketing campaigns and promotions that may influence demand:
  - Start and end dates of promotions
  - Advertising channels used
  - Promotion type (discounts, buy-one-get-one, etc.)

#### **6. Inventory Data:**

- Data on current stock levels, including:
  - Inventory levels for each product
  - Reorder points and lead times
  - Stockouts and backorders

#### **7. Social Media and Online Presence:**

- Social media mentions, engagement, and sentiment related to your products or brand can provide valuable insights:
  - Social media posts and comments
  - Likes, shares, and comments on social media platforms
  - Website traffic and clickstream data

#### **8. Customer Feedback and Reviews:**

- Collecting and analyzing customer reviews and feedback:
  - Product reviews and ratings

- Customer comments and complaints

#### **9. Point of Sale (POS) Data:**

- If applicable, data from physical and online point-of-sale systems:
  - Transaction data
  - Receipt data
  - In-store foot traffic data

**10. Supplier and Supply Chain Data:** - Information related to your supply chain and suppliers: - Supplier lead times - Transportation costs - Production schedules

**11. Geographic Data:** - Location-based data that can affect demand variations: - Geographic sales data - Store locations and traffic patterns

**12. IOT and Sensor Data (Advanced):** - Data from IOT sensors on products or in-store environments for real-time insights: - Product usage data - Environmental conditions in stores

**13. External APIs:** - Access external APIs for relevant data, such as weather APIs, economic indicators APIs, or social media APIs.

**14. Market Research and Reports:** - Industry-specific reports, studies, and market research can provide valuable insights into market trends.

**15. User Behavior Data (Advanced):** - For online businesses, analyze user behavior on your website or app: - Product views - Cart additions - Checkout abandonment rates

**16. Subscription and Membership Data (Advanced):** - Data from subscription-based services or loyalty programs can offer insights into recurring revenue and customer retention.

**17. Customer Support Data (Advanced):** - Records of customer inquiries, complaints, and support interactions.

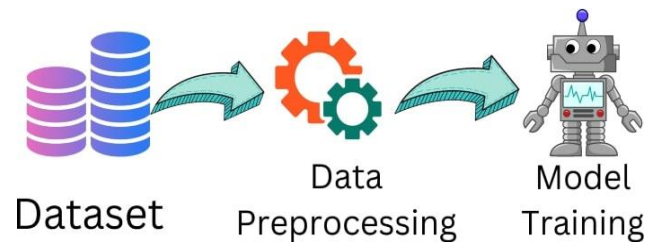
**18. Employee Data (Advanced):** - Staffing levels, shifts, and workforce data if they affect product availability.

**19. Transportation and Logistics Data (Advanced):** - Data on transportation routes, shipping times, and logistics can be relevant.

When collecting data, it's essential to ensure that you have a sufficient historical dataset for accurate prediction, and that the data is clean, properly formatted, and relevant to your specific prediction task. Data preprocessing, including data cleaning, feature engineering, and data integration, will be necessary to prepare the collected data for machine learning model training. Additionally, consider ethical and privacy considerations when handling customer data.

## **Data Preprocessing:**

Preprocessing in machine learning involves getting the data ready for analysis by making it more useful and easier for machine learning algorithms to understand. The main reason for data preprocessing is that we never get clean datasets. Always while we train the model or perform further operations, we need a clean dataset.



### **Need of Preprocessing :**

- Data Quality Improvement
- Feature Extraction and Selection
- Handling Missing Data
- Data Normalization and Scaling
- Outlier Detection and Treatment
- Reducing Computational Requirements

### **Steps of Preprocessing in Machine Learning:**

#### **Obtaining the Dataset:**

Gather the dataset you will be working with. Here is a dataset of cars. The dataset can be in any format. This dataset is in "**CSV**". It also can be in "**JSON**" or "**XLXS**" format. **CSV** stands for "**Comma Separated Values**" where all data is represented in a tabular format like a spreadsheet which is easy to understand.

#### **Importing Libraries:**

Bring in the necessary libraries for data manipulation and analysis, like Pandas, NumPy, and Scikit-learn.

- **pandas:** pandas are the Python library that is basically used for loading and managing the dataset.

- **numpy:** numpy is also a Python library used for mathematical operations, or we can say that in scientific calculations like adding two large multidimensional arrays.
- **sklearn and sci-kit-learn:** both are the same libraries used for data analytics.

Here pd and np are short naming conventions of pandas and numpy so that we can use these libraries by these short names.

### Loading the Dataset:

Load the dataset into your programming environment, which can be in different formats such as CSV, Excel, or datasets.

**read\_csv()** is the function of the panda's library, which is basically used to read the CSV file, the file can be saved locally, or we can also give a URL to this.

**dataset.drop()** is the function that is used to remove the specified column name, which is not needed in the dataset.

### Find Empty /Missing Data:

Identify any missing values in the dataset and decide on a strategy to address them. Options include removing the rows or columns with missing data or filling in the missing values with methods like mean, median, or mode. In this code snippet, missing values are handled by simply dropping the rows that contain them.

- **sklearn and sci-kit-learn:** both are the same libraries used for data analytics.

Here pd and np are short naming conventions of pandas and numpy so that we can use these libraries by these short names.

### Loading the Dataset:

Load the dataset into your programming environment, which can be in different formats such as CSV, Excel, or datasets.

**read\_csv()** is the function of the panda's library, which is basically used to read the CSV file, the file can be saved locally, or we can also give a URL to this.

**dataset.drop()** is the function that is used to remove the specified column name, which is not needed in the dataset.

### Find Empty /Missing Data:

Identify any missing values in the dataset and decide on a strategy to address them. Options include removing the rows or columns with missing data or filling in the missing values with methods like mean, median, or mode. In this code snippet, missing values are handled by simply dropping the rows that contain them.

```
# Handling missing values (if any)
df = df.dropna() # Remove rows with missing values
```

### **Encoding Categorical Data :**

Convert categorical variables into numerical representations that can be understood by machine learning algorithms. This can involve techniques like one-hot encoding or label encoding. Categorical variables need to be encoded numerically before using them in machine learning models

### **Scaling Numerical Features :**

Normalize or standardize the numerical features in the dataset to ensure they are on a similar scale. This helps prevent certain features from dominating the model's calculations. Numerical features are often scaled to a similar range to avoid the dominance of certain features during modeling.

### **Split Dataset :**

Divide the dataset into separate training and test sets. The training set is used to train the model, while the test set is used to evaluate its performance. This helps assess how well the model generalizes to new, unseen data. The code splits the preprocessed DataFrame into input features (X) and the target variable (y).

### **Feature Engineering:**

1. Feature engineering is a skill every data scientist should know how to perform, especially in the case of time series.
2. There are 6 powerful feature engineering techniques for time series

### **Introduction to Time Series:**

In a time series, the data is captured at equal intervals and each successive data point in the series depends on its past values.

### **Setting up the Problem Statement for Time Series Data:**

We'll be working on a fascinating problem to learn feature engineering techniques for time series.

#### **1. Feature Engineering for Time Series #1: Date-Related Features**

Task of forecasting the sales for a particular product. We can find out the sales pattern for weekdays and weekends based on historical data. Thus, having information about the day, month, year, etc. can be useful for forecasting the values.

## 2. Feature Engineering for Time Series #2: Time-Based Features

Extracting time-based features is very similar to what we did above when extracting date-related features. We start by converting the column to DateTime format and use the `.dt` accessor

## 3. Feature Engineering for Time Series #3: Lag Features

The lag value we choose will depend on the correlation of individual values with its past values.

## 4. Feature Engineering for Time Series #4: Rolling Window Feature

Recency is an important factor in a time series. Values closer to the current date would hold more information.

## 5. Feature Engineering for Time Series #5: Expanding Window Feature

The idea behind the expanding window feature is that it takes all the past values into account

## 6. Feature Engineering for Time Series #6: Domain-Specific Features

Having a good understanding of the problem statement, clarity of the end objective and knowledge of the available data is essential to engineer domain-specific features for the model.

### **Feature that captures Seasonal Pattern:**

Seasonal demand refers to the changes in consumer buying habits depending on the time of year. This may be related to changes in season (i.e., higher demand for snow boots during winter vs. higher demand for sunglasses during summer)

### **External Influences on the Product Demand**

- Political Factors.
- Economic Factors.
- Social Factors.
- Technological Factors.
- Legal Factors.
- Demographic Factors.
- Ethical Factors.
- Natural Factors.

## **Model Selection:**

**Model selection** is the task of selecting a model from among various candidates on the basis of performance criterion to choose the best one. In the context of learning, this may be the selection of a statistical model from a set of candidate models, given data. In the simplest cases, a pre-existing set of data is considered. However, the task can also involve the design of experiments such that the data collected is well-suited to the problem of model selection. Given candidate models of similar predictive or explanatory power, the simplest model is most likely to be the best choice

## **Regression Algorithm:**

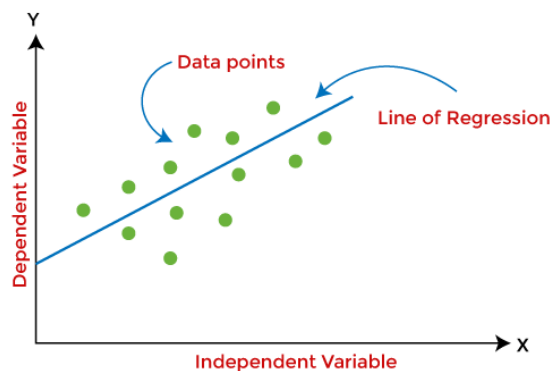
Regression analysis is often used in finance, investing, and others, and finds out the relationship between a single dependent variable(target variable) dependent on several independent ones. For example, predicting house price, stock market or salary of an employee, etc are the most common regression problems. Here is a list of top 5 regression algorithms

- Linear Regression
- Decision Tree
- Support Vector Regression
- Lasso Regression
- Random Forest

## **Linear Regression:**

Linear regression is the simplest machine learning model in which we try to predict one output variable using one or more input variables. The representation of linear regression is a linear equation, which combines a set of input values(x) and predicted output(y) for the set of those input values. It is represented in the form of a line:

$$Y = bx + c.$$





The main aim of the linear regression model is to find the best fit line that best fits the data points.

Linear regression is extended to multiple linear regression (find a plane of best fit) and polynomial regression (find the best fit curve).

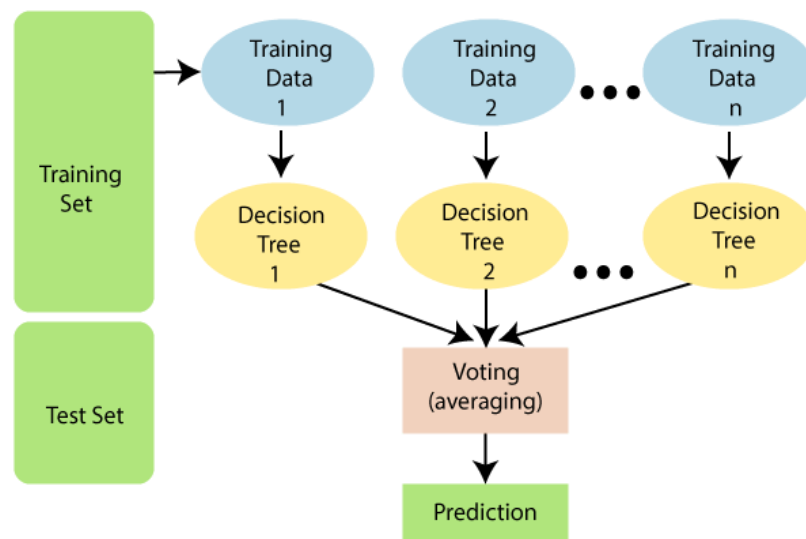
## Random Forest Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model*.

As the name suggests, ***"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."*** Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

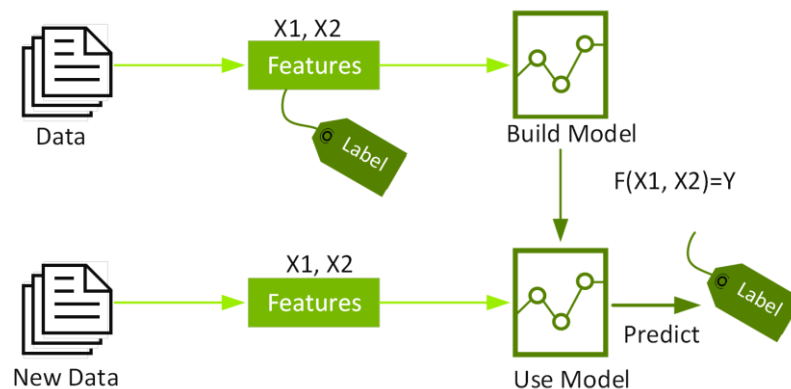
The below diagram explains the working of the Random Forest algorithm:



## XGBoost:

XGBoost, which stands for Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems. It's vital to an understanding of XGBoost to first grasp the machine learning concepts and algorithms that XGBoost builds upon: supervised machine learning, decision trees, ensemble learning, and gradient boosting.

Supervised machine learning uses algorithms to train a model to find patterns in a dataset with labels and features and then uses the trained model to predict the labels on a new dataset's features.



Decision trees create a model that predicts the label by evaluating a tree of if-then-else true/false feature questions, and estimating the minimum number of questions needed to assess the probability of making a correct decision. Decision trees can be used for classification to predict a category, or regression to predict a continuous numeric value.

## Model Training:

Training a machine learning model for demand product prediction using preprocessed data involves several steps. Here's a high-level guide on how to do it:

### 1. Split Data into Training and Testing Sets:

- Split your preprocessed data into two parts: a training set and a testing (or validation) set. The training set will be used to train the model, while the testing set will be used to evaluate its performance.

### 2. Select a Machine Learning Algorithm:

- Choose a machine learning algorithm suitable for your demand prediction task. Common choices include:
  - Linear Regression
  - Decision Trees
  - Random Forest
  - Gradient Boosting (e.g., XGBoost, LightGBM)
  - Time Series Forecasting Models (e.g., ARIMA, Prophet)
  - Deep Learning Models (e.g., LSTM, GRU)

### 3. Feature Selection (if not done during preprocessing):

- If you haven't already, select the most relevant features from your preprocessed data to use in training the model.

### 4. Train the Model:

- Fit the selected machine learning model to your training data. This involves providing the model with your historical data (features) and the corresponding target variable (demand).

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.ensemble import RandomForestRegressor
```

```
# Split data into train and test 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 (Random Forest as an example)
```

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

```
model.fit(X_train, y_train)
```

### 5. Hyperparameter Tuning (Optional):

- Fine-tune the hyperparameters of your model to optimize its performance. This can be done using techniques like grid search or random search.

## 6. Model Evaluation:

Evaluate the model's performance on the testing set using appropriate evaluation metrics. Common metrics for regression tasks include Mean

- Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R2) score.

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```
# Make predictions on the test set
```

```
y_pred = model.predict(X_test)
```

```
# Calculate evaluation metrics
```

```
mae = mean_absolute_error(y_test, y_pred)
```

```
mse = mean_squared_error(y_test, y_pred)
```

```
rmse = np.sqrt(mse)
```

```
r2 = r2_score(y_test, y_pred)
```

## 7. Model Interpretation (Optional):

- Depending on the model, you may want to interpret its predictions to understand the factors influencing demand. Techniques like feature importance analysis and SHAP (SHapley Additive exPlanations) can help with model interpretability.

## 8. Deployment (Optional):

- If you plan to use the model in a production environment to make real-time predictions, you'll need to deploy it as part of an application, API, or workflow.

## 9. Monitoring and Maintenance:

- Continuously monitor the model's performance in production and retrain it as needed to adapt to changing demand patterns. Regularly update the training data to incorporate new historical data.

## 10. Documentation and Reporting:

- Document the model, its performance, and any insights gained from it. Create reports or dashboards to present the predictions and insights to stakeholders.

## 11. Iterate and Improve:

- Based on the model's performance and feedback from stakeholders, consider making improvements to the model or the data preprocessing pipeline.

Remember that demand prediction is an iterative process, and the accuracy of your model may improve with time as you collect more data and refine your approach. Additionally, the choice of algorithm and preprocessing steps should be tailored to the specific characteristics of your data and the nature of your demand prediction problem.

## **Evaluation:**

Machine Learning is a branch of Artificial Intelligence. It contains many algorithms to solve various real-world problems. Building a Machine learning model is not only the Goal of any data scientist but deploying a more generalized model is a target of every Machine learning engineer.

Regression is also one type of supervised Machine learning and in this tutorial, we will discuss various metrics for evaluating regression Models and How to implement them using the sci-kit-learn library.

## **Regression:**

Regression is a type of Machine learning which helps in finding the relationship between independent and dependent variable.

In simple words, Regression can be defined as a Machine learning problem where we have to predict discrete values like price, Rating, Fees, etc.

## **Mean Absolute Error(MAE):**

MAE is a very simple metric which calculates the absolute difference between actual and predicted values.

To better understand, let's take an example you have input data and output data and use Linear Regression, which draws a best-fit line.

Now you have to find the MAE of your model which is basically a mistake made by the model known as an error. Now find the difference between the actual value and predicted value that is an absolute error but we have to find the mean absolute of the complete dataset.

so, sum all the errors and divide them by a total number of observations And this is MAE. And we aim to get a minimum MAE because this is a loss.

The diagram shows the formula for Mean Absolute Error (MAE) with several annotations. The formula is  $MAE = \frac{1}{N} \sum |Y - \hat{Y}|$ . An arrow points from the text "Divide by total Number of Data Points" to the fraction  $\frac{1}{N}$ . Another arrow points from "Actual Output" to the  $Y$  term in the absolute value. A third arrow points from "Predicted Output" to the  $\hat{Y}$  term. A bracket under the absolute value term  $|Y - \hat{Y}|$  is labeled "Absolute Value of residual". An arrow points from "Sum Of" to the summation symbol  $\sum$ .

### Advantages of MAE:

- The MAE you get is in the same unit as the output variable.
- It is most Robust to outliers.

### Disadvantages of MAE:

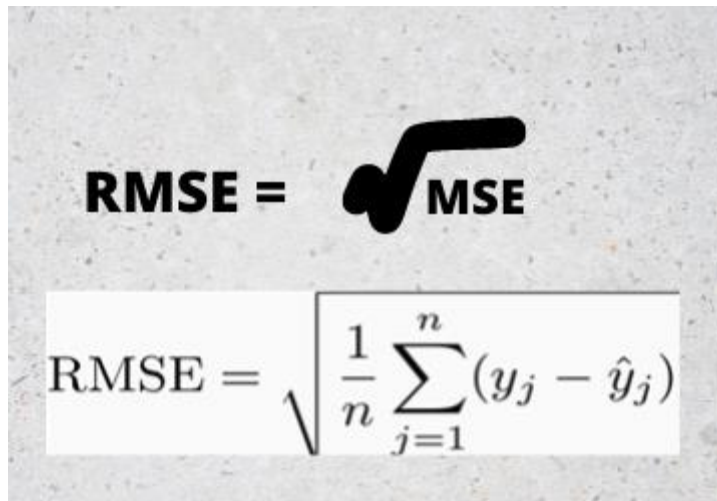
- The graph of MAE is not differentiable so we have to apply various optimizers like Gradient descent which can be differentiable.

```
from sklearn.metrics import mean_absolute_error
print("MAE",mean_absolute_error(y_test,y_pred))
```

Now to overcome the disadvantage of MAE next metric came as MSE.

### **Root Mean Squared Error(RMSE):**

As RMSE is clear by the name itself, that it is a simple square root of mean squared error.



The image shows a hand-drawn diagram on a textured background. At the top, it says "RMSE = " followed by a large, bold, hand-drawn square root symbol, and then "MSE". Below this, there is a white rectangular box containing the mathematical formula for RMSE: 
$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{j=1}^n (y_j - \hat{y}_j)^2}$$

### **Advantages of RMSE:**

- The output value you get is in the same unit as the required output variable which makes interpretation of loss easy.

### **Disadvantages of RMSE:**

- It is not that robust to outliers as compared to MAE.

for performing RMSE we have to NumPy NumPy square root function over MSE.

```
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
```

Most of the time people use RMSE as an evaluation metric and mostly when you are working with deep learning techniques the most preferred metric is RMSE.