**Predicting IMDb Scores Using Machine Learning**

**TEAM MEMBER**

**732521104053 : THAMARAISELVI A**

**Phase 5 Submission Document**

**PROJECT : Predicting IMDb Scores**



**Introduction:**

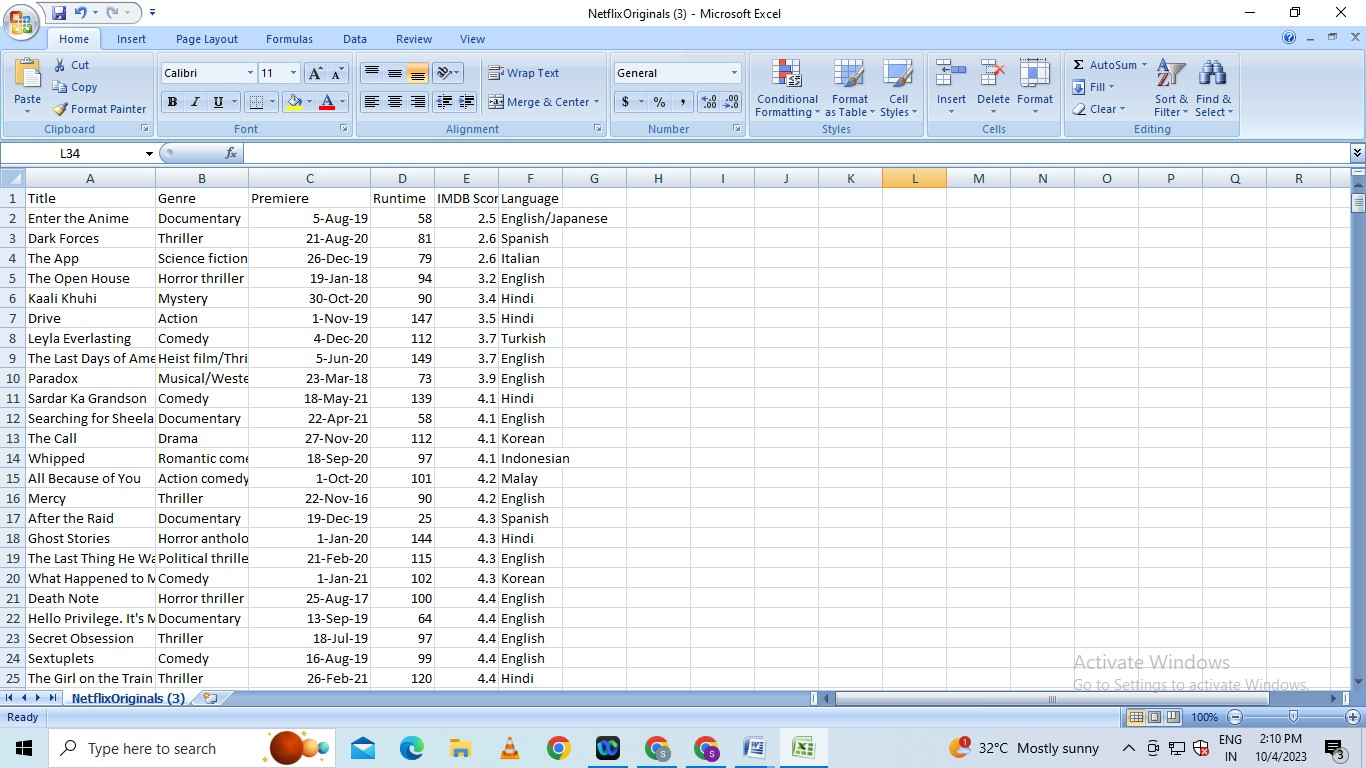
* Predicting IMDb scores for movies or TV shows typically involves using machine learning models and features such as cast, crew, genre, user reviews, and more.
* You can use regression algorithms to build a predictive model.
* The quality of your predictions depends on the quality and quantity of data, as well as the choice of features and model.
* In this project , we will explore advanced regression techniques to enhance the accuracy and robustness of IMDb scores prediction models
* Highlight the limitations of traditional linear regression models in capturing complex relationships.
* Emphasize the need for advanced regression techniques like Gradient Boosting and Neural Networks to enchance prediction accuracy.

**Content For Project Phase 5 :**

* Clearly outline the problem statement, design thinking process, and the phases of development.
* Describe the dataset used, data preprocessing steps, and model training process.
* Explain the choice of regression algorithm and evaluation metrics

**Data Source :**

* A Good Data for Predicting IMDb Scores using machine learning model should be Accurate , complete , accessible

**Dataset Link : (**[**https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores**](https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores))

**Loading in data set :**

* Suppose if you want to start a ML project then what is the first and most important thing you would require? It is the data that we need to load for starting any of the ML project. With respect to data, the most common format of data for ML projects is CSV (comma-separated values).
* Basically, CSV is a simple file format which is used to store tabular data (number and text) such as a spreadsheet in plain text. In Python, we can load CSV data into with different ways but before loading CSV data we must have to take care about some considerations.

Why we need data preprocessing :

* **Getting the dataset**
* **Importing libraries**
* **Importing datasets**
* **Finding Missing Data**
* **Encoding Categorical Data**
* **Splitting dataset into training and test set**
* **Feature scaling**

## Get the Dataset :

* To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the **dataset**.
* Dataset may be of different formats for different purposes, such as, if we want to create a machine learning model for business purpose, then dataset will be different with the dataset required for a liver patient. So each dataset is different from another dataset. To use the dataset in our code, we usually put it into a CSV **file**. However, sometimes, we may also need to use an HTML or xlsx file.

 Importing Libraries :

* In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:
* **Numpy:** Numpy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to add large, multidimensional arrays and matrices. So, in Python, we can import it as:

1. import numpy as nm

Here we have used **nm**, which is a short name for Numpy, and it will be used in the whole program.

* **Matplotlib:** The second library is **matplotlib**, which is a Python 2D plotting library, and with this library, we need to import a sub-library **pyplot**. This library is used to plot any type of charts in Python for the code. It will be imported as below:

1. import matplotlib.pyplot as mpt

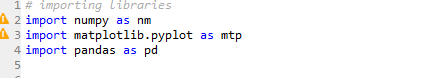
Here we have used mpt as a short name for this library.

* **Pandas:** The last library is the Pandas library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library. It will be imported as below:

Here, we have used pd as a short name for this library. Consider the below image:

 Importing the Datasets :

* Now we need to import the datasets which we have collected for our machine learning project. But before importing a dataset, we need to set the current directory as a working directory. To set a working directory in Spyder IDE, we need to follow the below steps:
* Save your Python file in the directory which contains dataset.
* Go to File explorer option in Spyder IDE, and select the required directory.
* Click on F5 button or run option to execute the file.



## Handling Missing data :

* The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

**Ways to handle missing data :**

There are mainly two ways to handle missing data, which are:

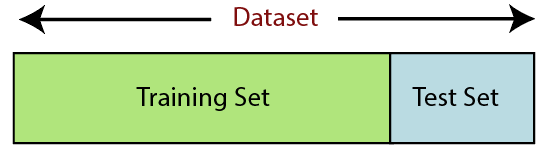
* **By deleting the particular row:** The first way is used to commonly deal with null values. In this way, we just delete the specific row or column which consists of null values. But this way is not so efficient and removing data may lead to loss of information which will not give the accurate output.
* **By calculating the mean:** In this way, we will calculate the mean of that column or row which contains any missing value and will put it on the place of missing value. This strategy is useful for the features which have numeric data such as age, salary, year, etc. Here, we will use this approach.

## Encoding Categorical data :

* Categorical data is data which has some categories such as, in our dataset; there are two categorical variable, **Country**, and **Purchased**.
* Firstly, we will convert the country variables into categorical data. So to do this, we will use **LabelEncoder()** class from **preprocessing** library.

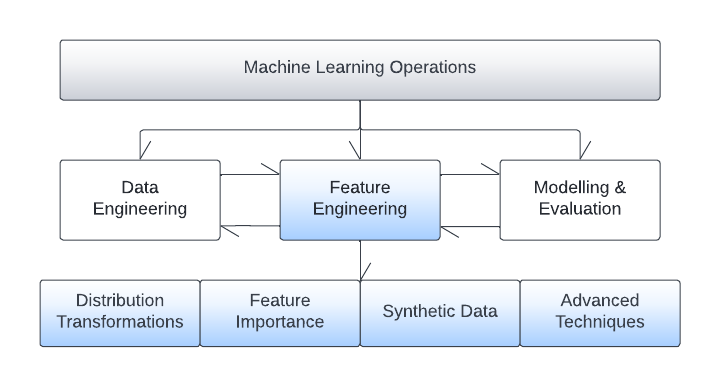
Splitting the Dataset into the Training set and Test set:

* In machine learning data preprocessing, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model.
* Suppose, if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.
* If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:



**Feature engineering :**

* Feature engineering is the pre-processing step of machine learning, which extracts features from raw data.
* It helps to represent an underlying problem to predictive models in a better way, which as a result, improve the accuracy of the model for unseen data.
* The predictive model contains predictor variables and an outcome variable, and while the feature engineering process selects the most useful predictor variables for the model.
* Since 2016, automated feature engineering is also used in different machine learning software that helps in automatically extracting features from raw data.



**Feature engineering process :**

* **Feature Creation** : Feature creation is finding the most useful variables to be used in a predictive model. The process is subjective, and it requires human creativity and intervention. The new features are created by mixing existing features using addition, subtraction, and ration, and these new features have great flexibility.
* **Transformations** : The transformation step of feature engineering involves adjusting the predictor variable to improve the accuracy and performance of the model. For example, it ensures that the model is flexible to take input of the variety of data; it ensures that all the variables are on the same scale, making the model easier to understand. It improves the model's accuracy and ensures that all the features are within the acceptable range to avoid any computational error.
* **Feature Extraction :** Feature extraction is an automated feature engineering process that generates new variables by extracting them from the raw data. The main aim of this step is to reduce the volume of data so that it can be easily used and managed for data modelling. Feature extraction methods include **cluster analysis, text analytics, edge detection algorithms, and principal components analysis (PCA**).
* **Feature Selection :**  While developing the machine learning model, only a few variables in the dataset are useful for building the model, and the rest features are either redundant or irrelevant. If we input the dataset with all these redundant and irrelevant features, it may negatively impact and reduce the overall performance and accuracy of the model.

Steps in Feature Engineering :

* The steps of feature engineering may vary as per different data scientists and ML engineers. However, there are some common steps that are involved in most machine learning algorithms, and these steps are as follows:

**Data Preparation:**

* The first step is data preparation. In this step, raw data acquired from different resources are prepared to make it in a suitable format so that it can be used in the ML model. The data preparation may contain cleaning of data, delivery, data augmentation, fusion, ingestion, or loading

**Exploratory Analysis:**

* Exploratory analysis or Exploratory data analysis (EDA) is an important step of features engineering, which is mainly used by data scientists.
* This step involves analysis, investing data set, and summarization of the main characteristics of data.
* Different data visualization techniques are used to better understand the manipulation of data sources, to find the most appropriate statistical technique for data analysis, and to select the best features for the data.

**Benchmark**:

* Benchmarking is a process of setting a standard baseline for accuracy to compare all the variables from this baseline. The benchmarking process is used to improve the predictability of the model and reduce the error rate.

Feature Engineering Techniques :

* Imputation : Feature engineering deals with inappropriate data, missing values, human interruption, general errors, insufficient data sources, etc. Missing values within the dataset highly affect the performance of the algorithm, and to deal with them "Imputation" technique is used. **Imputation is responsible for handling irregularities within the dataset.**

**Model Training :**

* Model training is at the heart of the data science development lifecycle where the data science team works to fit the best weights and biases to an algorithm to minimize the loss function over prediction range.
* Loss functions define how to optimize the ML algorithms. A data science team may use different types of loss functions depending on the project objectives, the type of data used and the type of algorithm.
* When a supervised learning technique is used, model training creates a mathematical representation of the relationship between the data features and a target label.
* In unsupervised learning, it creates a mathematical representation among the data features themselves.

Importance of Model Training.

Model training is the primary step in machine learning, resulting in a working model that can then be validated, tested and deployed. The model’s performance during training will eventually determine how well it will work when it is eventually put into an application for the end-users.

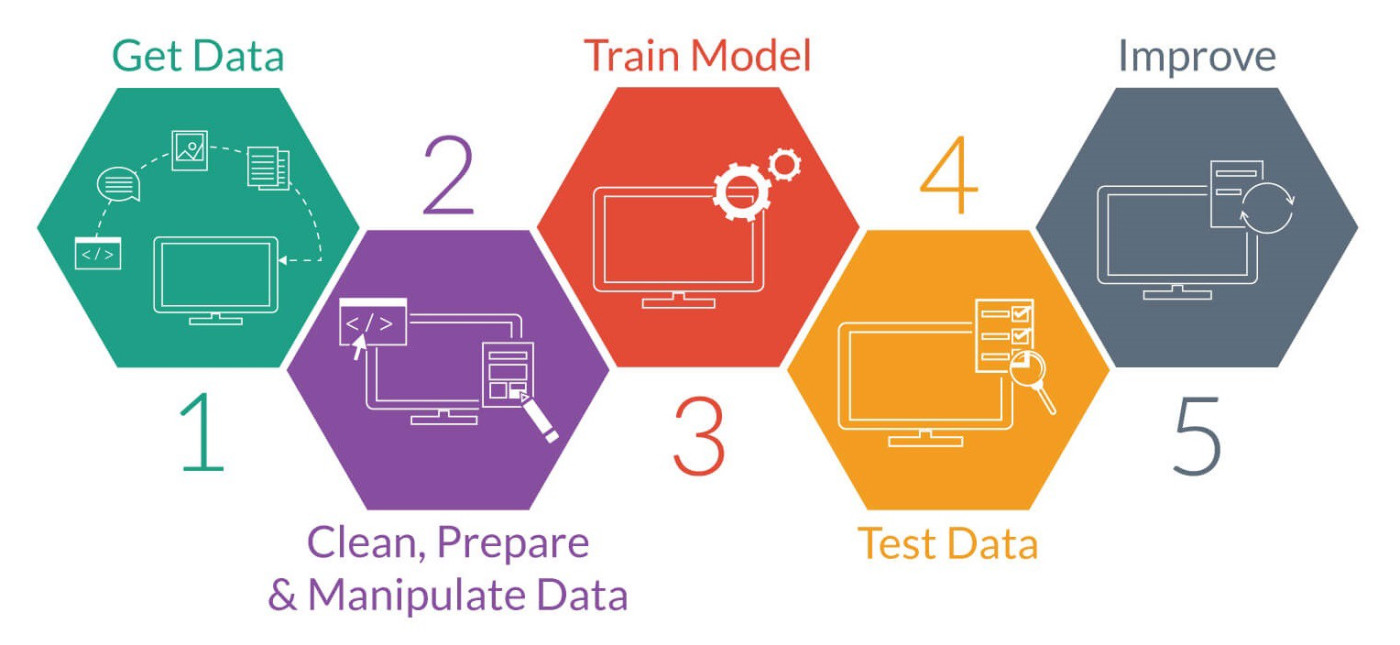
Both the quality of the training data and the choice of the algorithm are central to the model training phase. In most cases, training data is split into two sets for training and then validation and testing.

the selection of the algorithm is primarily determined by the end-use case. However, there are always additional factors that need to be considered, such as algorithm-model complexity, performance, interpretability, computer resource requirements, and speed.

Balancing out these various requirements can make selecting algorithms an involved and complicated process.

Model training is the phase in the data science development lifecycle where practitioners try to fit the best combination of weights and bias to a machine learning algorithm to minimize a loss function over the prediction range.

The p urp ose of model training is to build the best mathematical representation of the relationship between data features and a target label (in supervised learning) or among the features themselves (unsupervised learning).



**How To Train a Machine Learning Model :**

* Training a model requires a systematic, repeatable process that maximizes your utilization of your available training data and the time of your data science team. Before you begin the training phase, you need to first determine your problem statement, access your data set and clean the data to be presented to the model.
* In addition to this, you need to determine which algorithms you will use and what parameters (hyperparameters) they will run with. With all of this done, you can split your dataset into a training set and a testing set, then prepare your model algorithms for training.

Split the Dataset :

* Your initial training data is a limited resource that needs to be allocated carefully. Some of it can be used to train your model, and some of it can be used to test your model – but you can’t use the same data for each step.
* You can’t properly test a model unless you have given it a new data set that it hasn’t encountered before. Splitting the training data into two or more sets allows you to train and then validate the model using a single source of data. This allows you to see if the model is overfit, meaning that it performs well with the training data but poorly with the test data.
* Split the data into ten equal parts or folds.
* Designate one fold as the hold-out fold.
* Train the model on the other nine folds.
* Test the model on the hold-out fold.

Repeat this process ten times, each time selecting a different fold to be the hold-out fold. The average performance across the ten hold-out folds is your performance estimate, called the cross-validated score.

Select Algorithms to Test :

* In machine learning, there are thousands of algorithms to choose from, and there is no sure way to determine which will be the best for any specific model.
* In most cases, you will likely try dozens, if not hundreds, of algorithms in order to find the one that results in an accurate working model. Selecting candidate algorithms will often depend on:

Size of the training data.

* Accuracy and interpretability of the required output.
* Speed of training time required, which is inversely proportional to accuracy.
* Linearity of the training data.
* Number of features in the data seT.

Tune the Hyperparameters :

* Hyperparameters are the high-level attributes set by the data science team before the model is assembled and trained.
* While many attributes can be learned from the training data, they cannot learn their own hyperparameters.
* As an example, if you are using a [regression algorithm](http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/153-penalized-regression-essentials-ridge-lasso-elastic-net/), the model can determine the regression coefficients itself by analyzing the data.
* As another example, a model using the random forest technique can determine where decision trees will be split, but the number of trees to be used needs to be tuned beforehand.

Fit and Tune Models :

* Now that the data is prepared and the model’s hyperparameters have been determined, it’s time to start training the models. The process is essentially to loop through the different algorithms using each set of hyperparameter values you’ve decided to explore. To do this:
* Split the data.
* Select an algorithm.
* Tune the hyperparameter values.
* Train the model.
* Select another algorithm and repeat steps 3 and 4..
* Next, select another set of hyperparameter values you want to try for the same algorithm, cross-validate it again and calculate the new score.
* Once you have tried each hyperparameter value, you can repeat these same steps for additional algorithms.

Think of these trials as track and field heats.

* Each algorithm has demonstrated what it can do with the different hyperparameter values.
* Now you can select the best version from each algorithm and send them on to the final competition.

Choose the Best Model :

Now it’s time to test the best versions of each algorithm to determine which gives you the best model overall.

* Make predictions on your test data.
* Determine the [ground truth](https://domino.ai/data-science-dictionary/ground-truth) for your target variable during the training of that model.
* Determine the performance metrics from your predictions and the ground truth target variable.
* Run each finalist model with the test data.

Once the testing is done, you can compare their performance to determine which are the better models. The overall winner should have performed well (if not the best) in training as well as in testing.

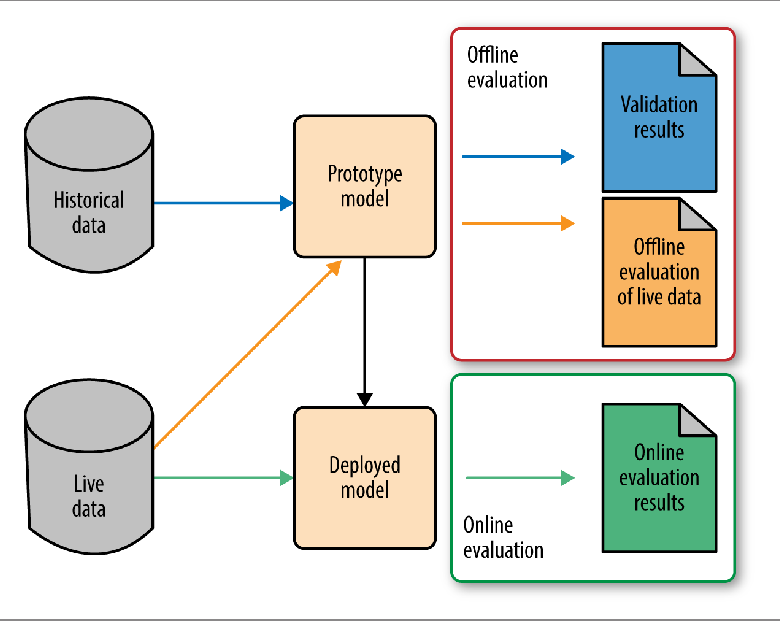
* It should also perform well on your other performance metrics (like speed and [empirical loss](https://developers.google.com/machine-learning/crash-course/descending-into-ml/training-and-loss)), and – ultimately – it should adequately solve or answer the question posed in your problem statement.

**Systematic Approach to Model Training :**

* Using a systematic and repeatable model training process is of paramount importance for any organization planning to build successful [machine learning model](https://domino.ai/blog/a-guide-to-machine-learning-models) at scale.
* Central to this is having all of your resources, [tools](https://domino.ai/blog/data-science-tools), libraries and documentation in a single enterprise platform that will foster collaboration instead of hindering it.

**Model evaluation :**

* Model evaluation in machine learning is the process of assessing the performance and effectiveness of a trained machine learning model.
* It helps determine how well the model is likely to perform on unseen or future data.
* Evaluating models is a critical step in the machine learning workflow, as it helps to select the best model, fine-tune its parameters, and make informed decisions about its deployment.



**Training and Test Data** :

* In supervised learning, the dataset is typically divided into two parts: a training set and a test set. The training set is used to train the model, and the test set is used to evaluate its performance.

**Performance Metrics**:

* There are various performance metrics used to assess the quality of a model's predictions. The choice of metric depends on the specific problem, such as classification, regression, or clustering.

Classification :

* **Accuracy**: The proportion of correct predictions.
* **Precision**: The ratio of true positive predictions to the total number of positive predictions.
* **Recall**: The ratio of true positive predictions to the total number of actual positives.
* **F1 Score**: A harmonic mean of precision and recall.
* **ROC-AUC**: Receiver Operating Characteristic - Area Under the Curve.
* Regression:
* **Mean Absolute Error (MAE)**: The average absolute difference between predicted and actual values.

**Mean Squared Error (MSE)**: The average squared difference between predicted and actual values.

* **Root Mean Squared Error (RMSE)**: The square root of MSE.
* **R-squared (R²)**: A measure of the proportion of variance in the dependent variable explained by the model.
* Clustering:
* **Silhouette Score**: Measures the quality of clustering.
* **Davies-Bouldin Index**: Measures the average similarity between each cluster and the cluster that is most similar to it.

**Cross validation :**

* Cross-validation is a technique to assess the model's generalization performance more robustly. Common methods include k-fold cross-validation and leave-one-out cross-validation.

**Overfitting and Underfitting**:

* Model evaluation helps identify issues like overfitting (the model performs well on the training data but poorly on test data) and underfitting (the model is too simple to capture the data's underlying patterns).

**Bias-Variance Tradeoff**:

* Model evaluation helps understand the trade-off between bias (underfitting) and variance (overfitting). Ideally, you want to find a balance between these two to achieve good predictive performance

**Hyperparameter Tuning**:

* Model evaluation often involves tuning hyperparameters to optimize the model's performance on the test set.

**Visualization and Interpretability**:

* Visualizations, such as confusion matrices for classification, and feature importance plots, can provide insights into the model's behavior.

**Model Comparison**:

* It's common to compare the performance of multiple models to select the best-performing one for a given task.

**Ethical Considerations**:

* Model evaluation should also consider ethical and fairness concerns, especially when the model's predictions have real-world consequences.

**Real-World Testing**:

* After initial evaluation, models are often tested in real-world settings or environments to assess their practical effectiveness.
* Model evaluation is an iterative process, and it plays a crucial role in ensuring that machine learning models are not only accurate but also reliable and suitable for their intended use cases.
* The choice of evaluation metrics and methods should align with the specific goals and characteristics of the problem you are trying to solve.