DATA SCIENCE SALARIES DATASET

EFFORTS BY:-

SHAGUN PREET KAUR

CONTENT

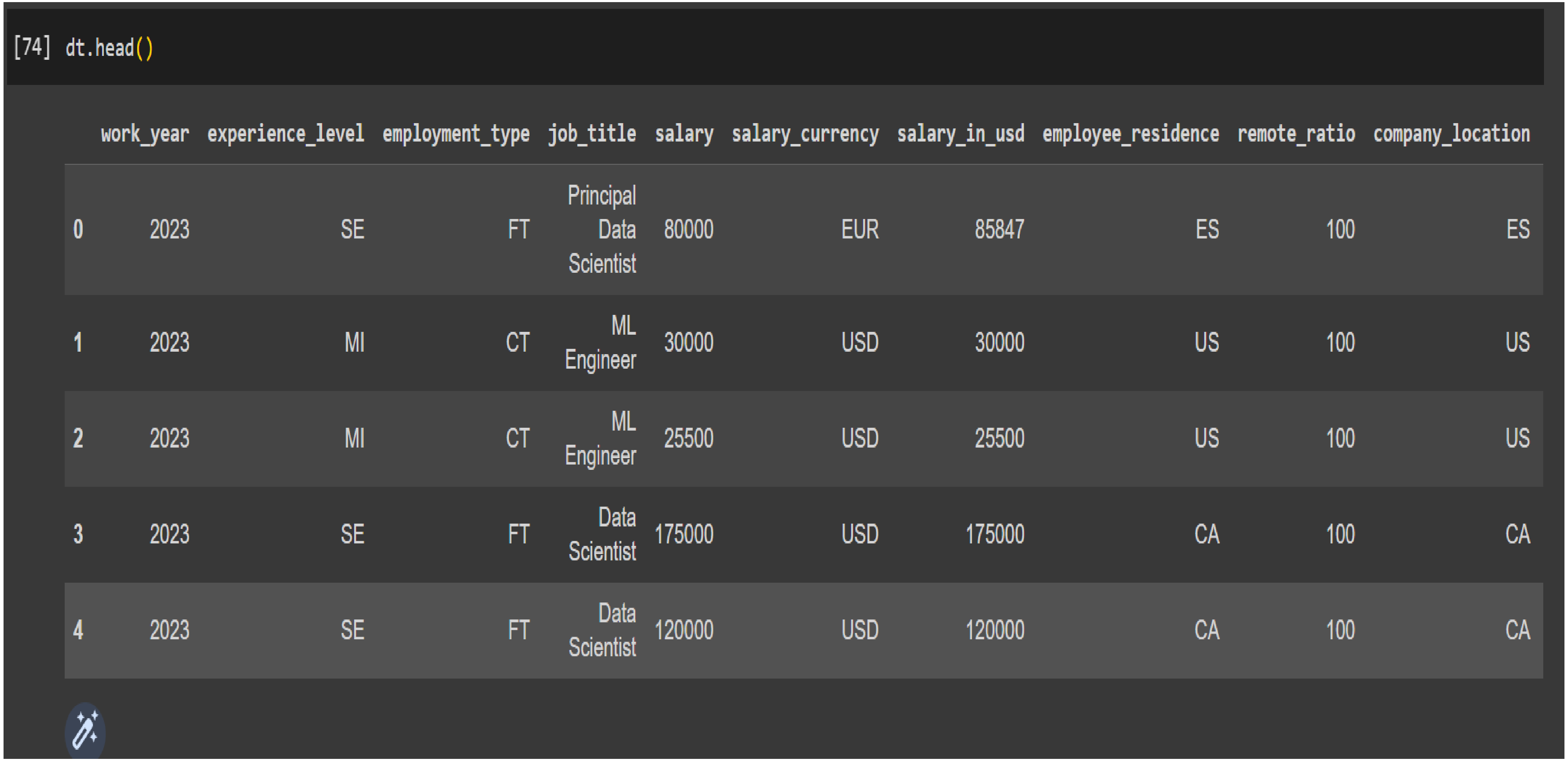
- Introduction to datset
- Data Visualisation
- Inroduction to ML
- Validation Techniques(Splitting the data)
- MinMaxScaler
- Confusion Matrix

INTRODUCTION TO DATASET

This dataset provides comprehensive information about Data Science Salaries

1) work_year: The year the salary was paid.
2) employment_type: The type of employment for the role
3) job_title: The role worked in during the year.
4) experience_level: The experience level in the job during the year
5) salary: The total gross salary amount paid.
6) salary_currency: The currency of the salary paid as an ISO 4217 currency code.
7) salaryinusd: The salary in USD
8) employee_residence: Employee's primary country of residence in during the work year as an ISO 3166 country code.
9) remote_ratio: The overall amount of work done remotely
10) company_location: The country of the employer's main office or contracting branch
11) company_size: The median number of people that worked for the company during the year

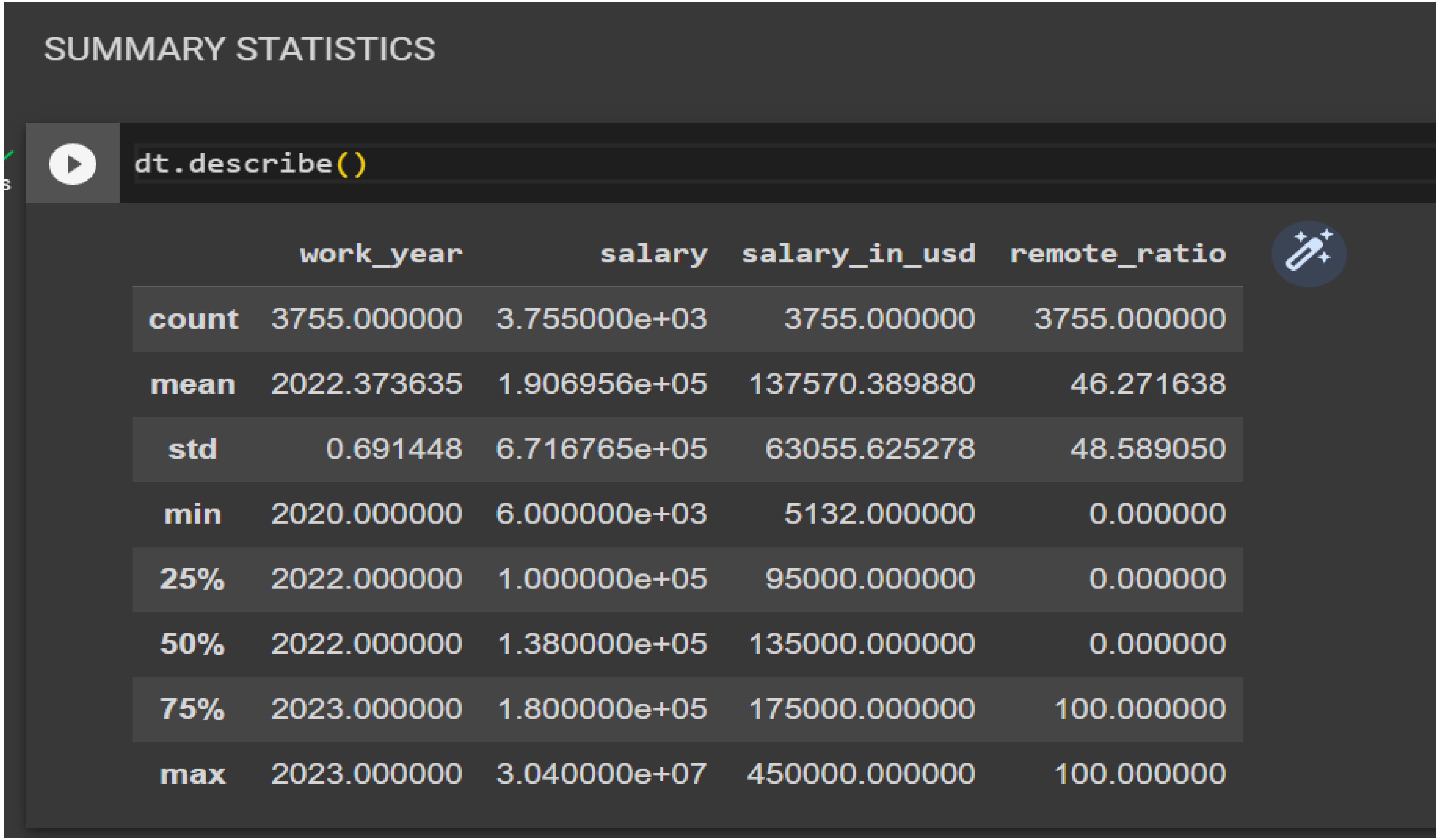
ABOUT DATASET



```
[74] dt.head()
```

	work_year	experience_level	employment_type	job_title	salary	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location
0	2023	SE	FT	Principal Data Scientist	80000	EUR	85847	ES	100	ES
1	2023	MI	CT	ML Engineer	30000	USD	30000	US	100	US
2	2023	MI	CT	ML Engineer	25500	USD	25500	US	100	US
3	2023	SE	FT	Data Scientist	175000	USD	175000	CA	100	CA
4	2023	SE	FT	Data Scientist	120000	USD	120000	CA	100	CA

SUMMARY STATISTICS

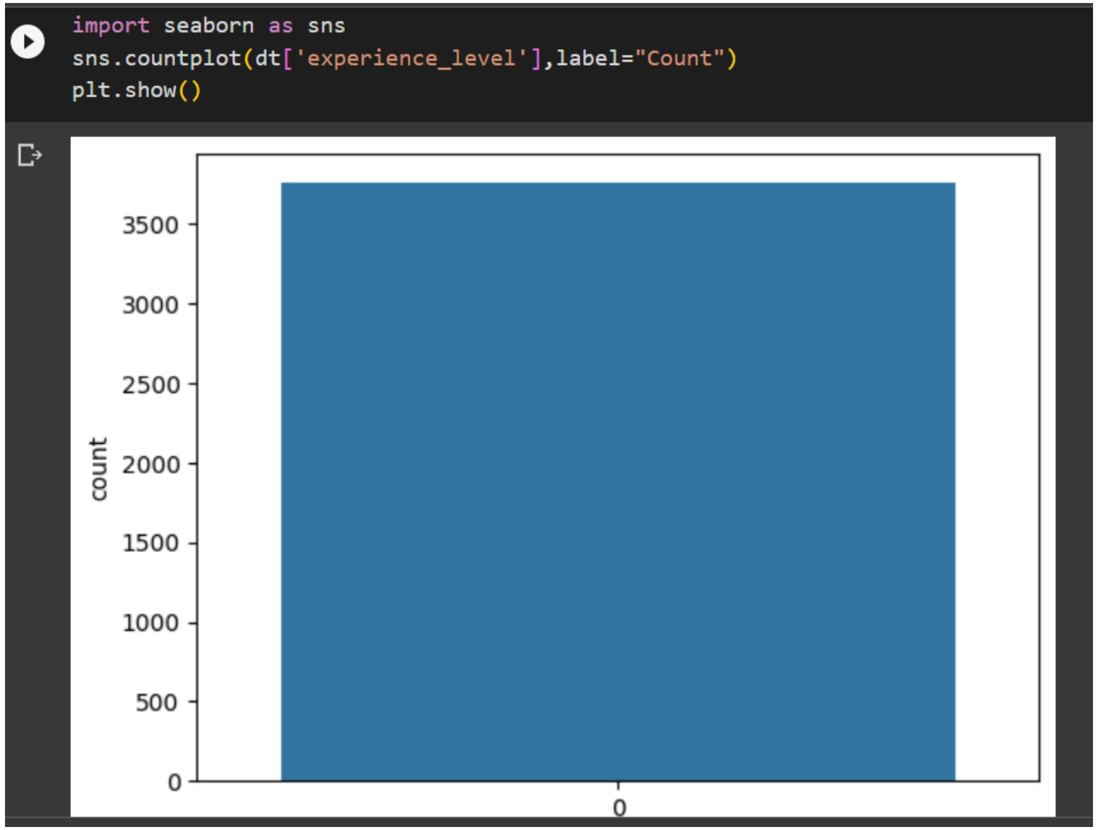


SUMMARY STATISTICS

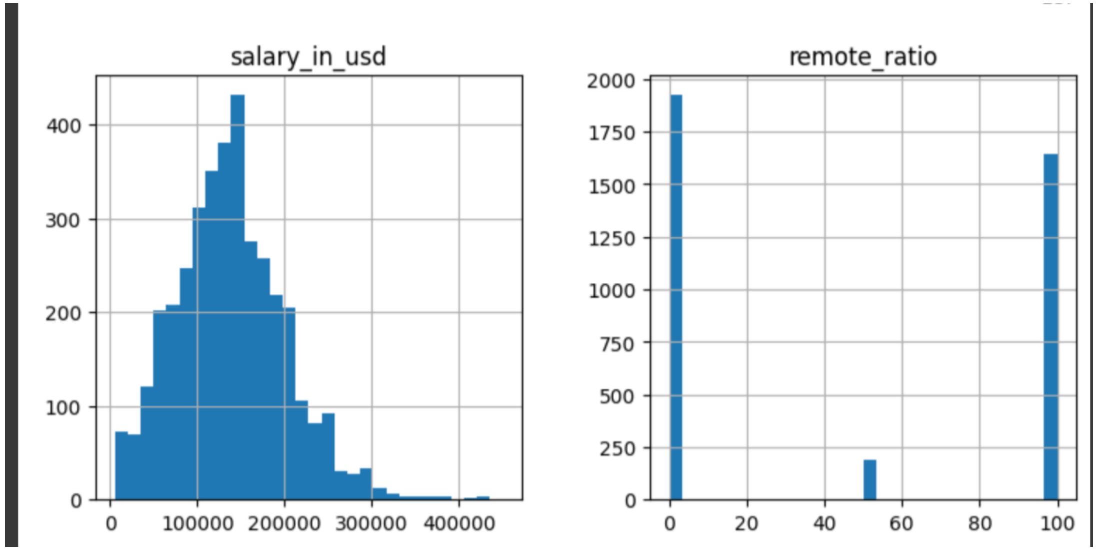
```
dt.describe()
```

```
         work_year        salary   salary_in_usd  remote_ratio
count  3755.000000  3.755000e+03     3755.000000   3755.000000
mean   2022.373635  1.906956e+05   137570.389880     46.271638
std       0.691448  6.716765e+05    63055.625278     48.589050
min    2020.000000  6.000000e+03     5132.000000      0.000000
25%    2022.000000  1.000000e+05    95000.000000      0.000000
50%    2022.000000  1.380000e+05   135000.000000      0.000000
75%    2023.000000  1.800000e+05   175000.000000    100.000000
max    2023.000000  3.040000e+07   450000.000000    100.000000
```

DATA VISUALISATION

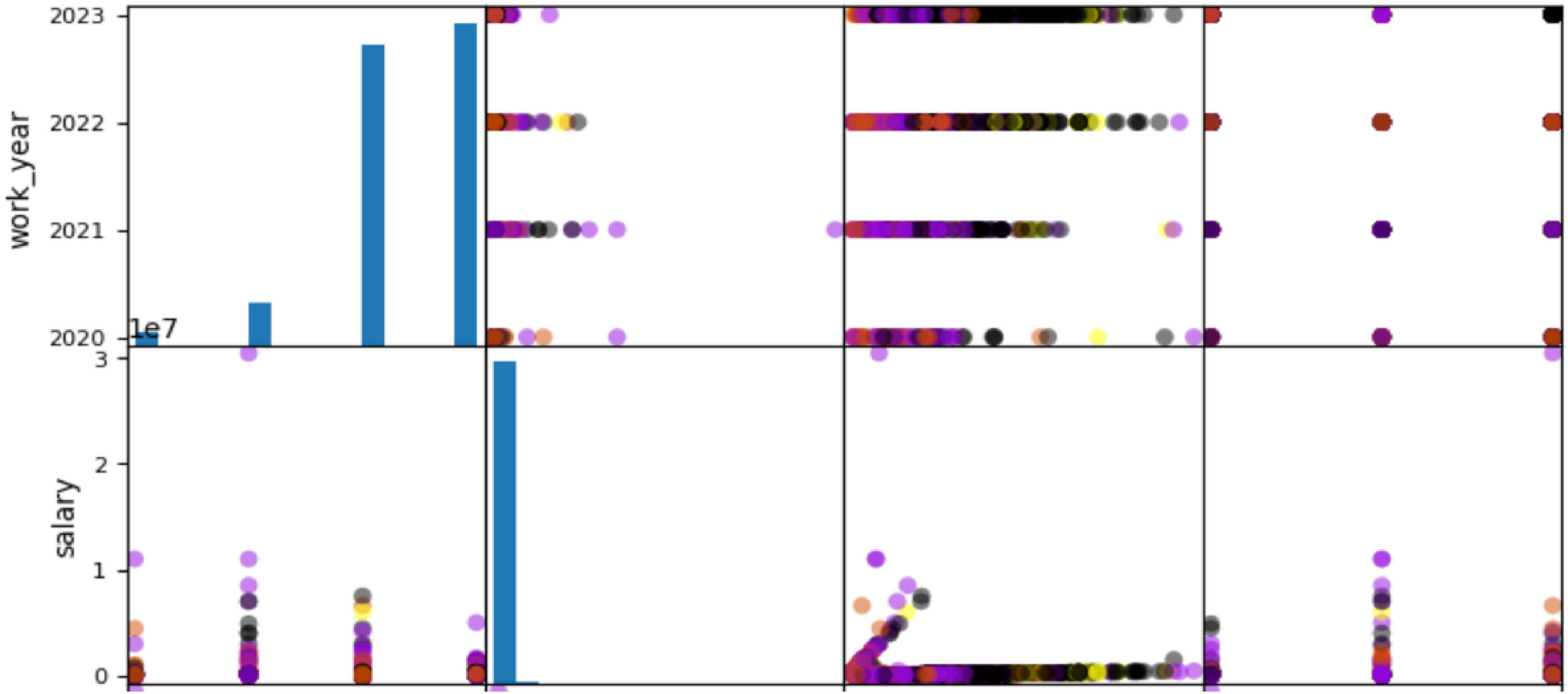


```
import seaborn as sns
sns.countplot(dt['experience_level'],label="Count")
plt.show()
```



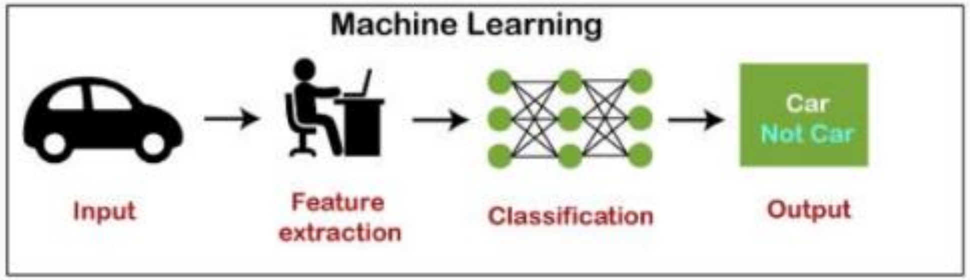
DATA VISUALISATION

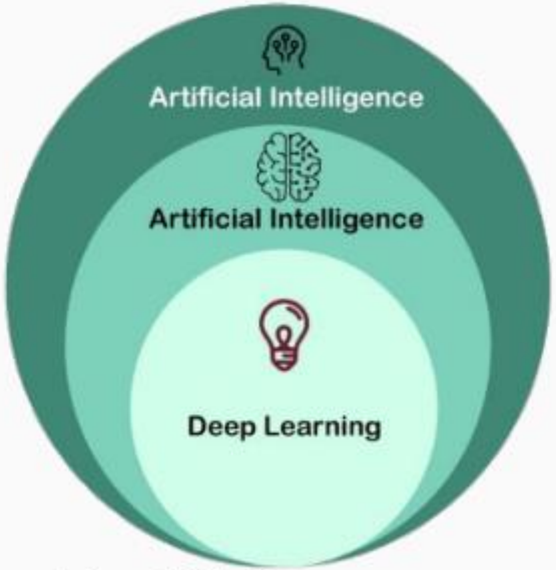
Scatter-matrix for each input variable



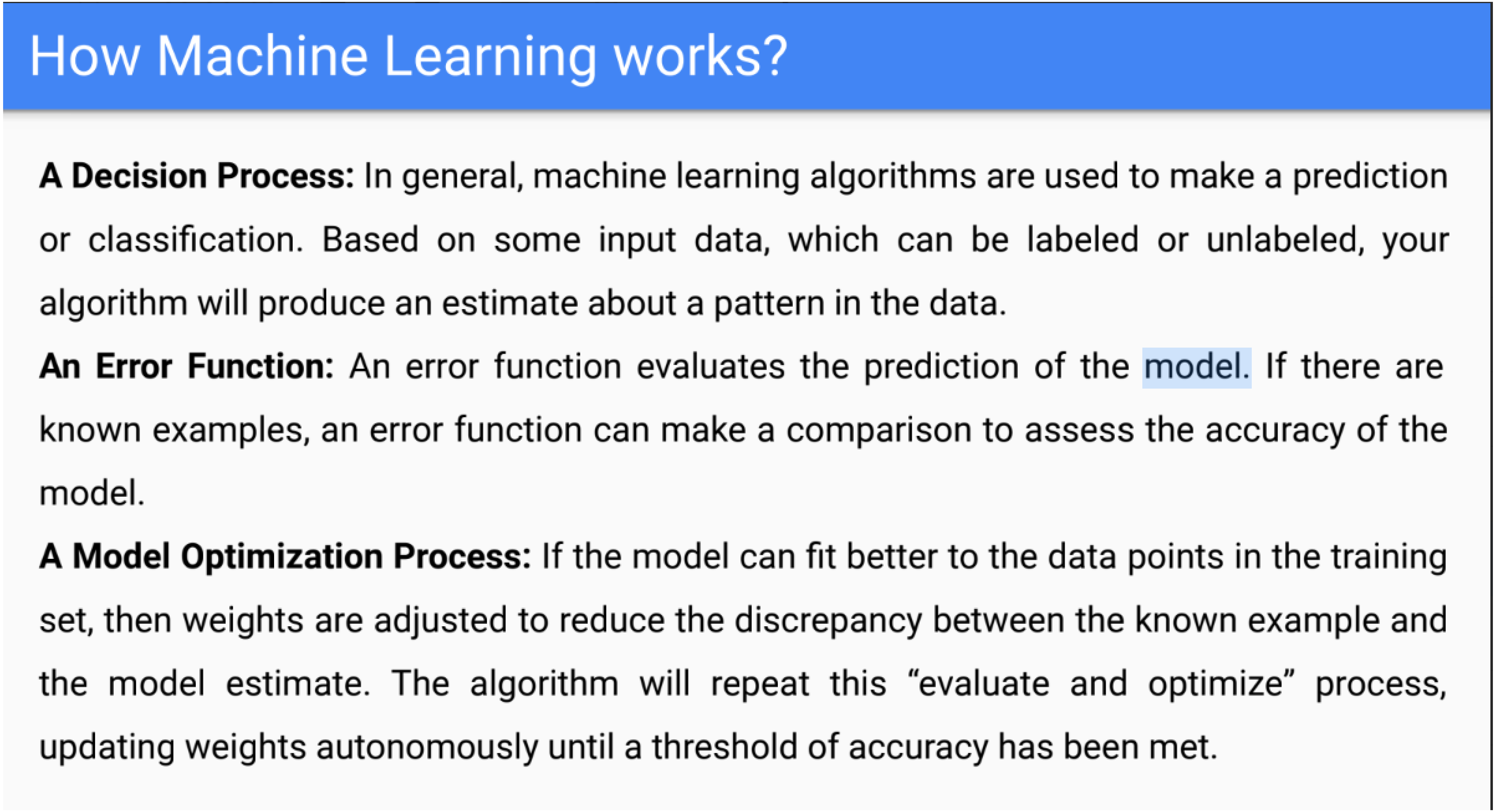
Machine Learning

Machine Learning allows the computers to learn from the experiences by its own, use statistical methods to improve the performance and predict the output without being explicitly programmed. Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.





The term machine learning was first introduced by **Arthur Samuel in 1959.**



How Machine Learning works?

A Decision Process: In general, machine learning algorithms are used to make a prediction or classification. Based on some input data, which can be labeled or unlabeled, your algorithm will produce an estimate about a pattern in the data.

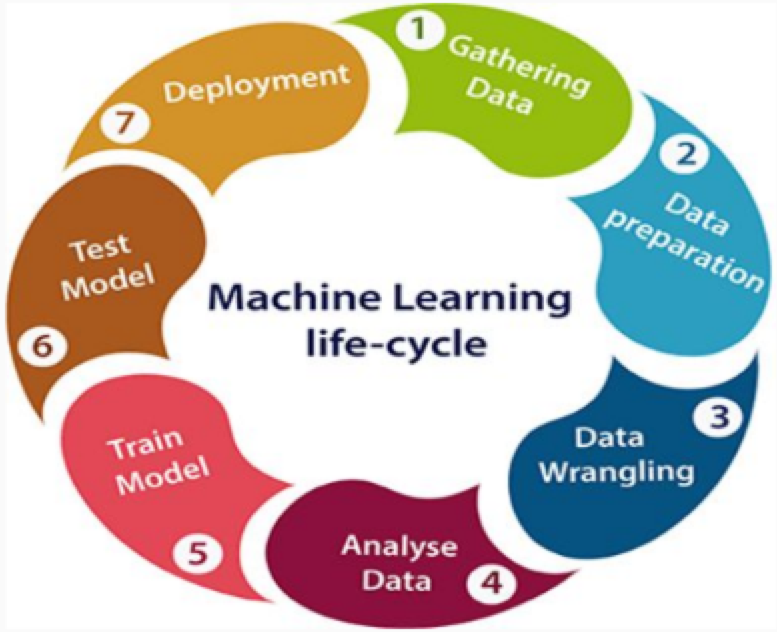
An Error Function: An error function evaluates the prediction of the model. If there are known examples, an error function can make a comparison to assess the accuracy of the model.

A Model Optimization Process: If the model can fit better to the data points in the training set, then weights are adjusted to reduce the discrepancy between the known example and the model estimate. The algorithm will repeat this “evaluate and optimize” process, updating weights autonomously until a threshold of accuracy has been met.

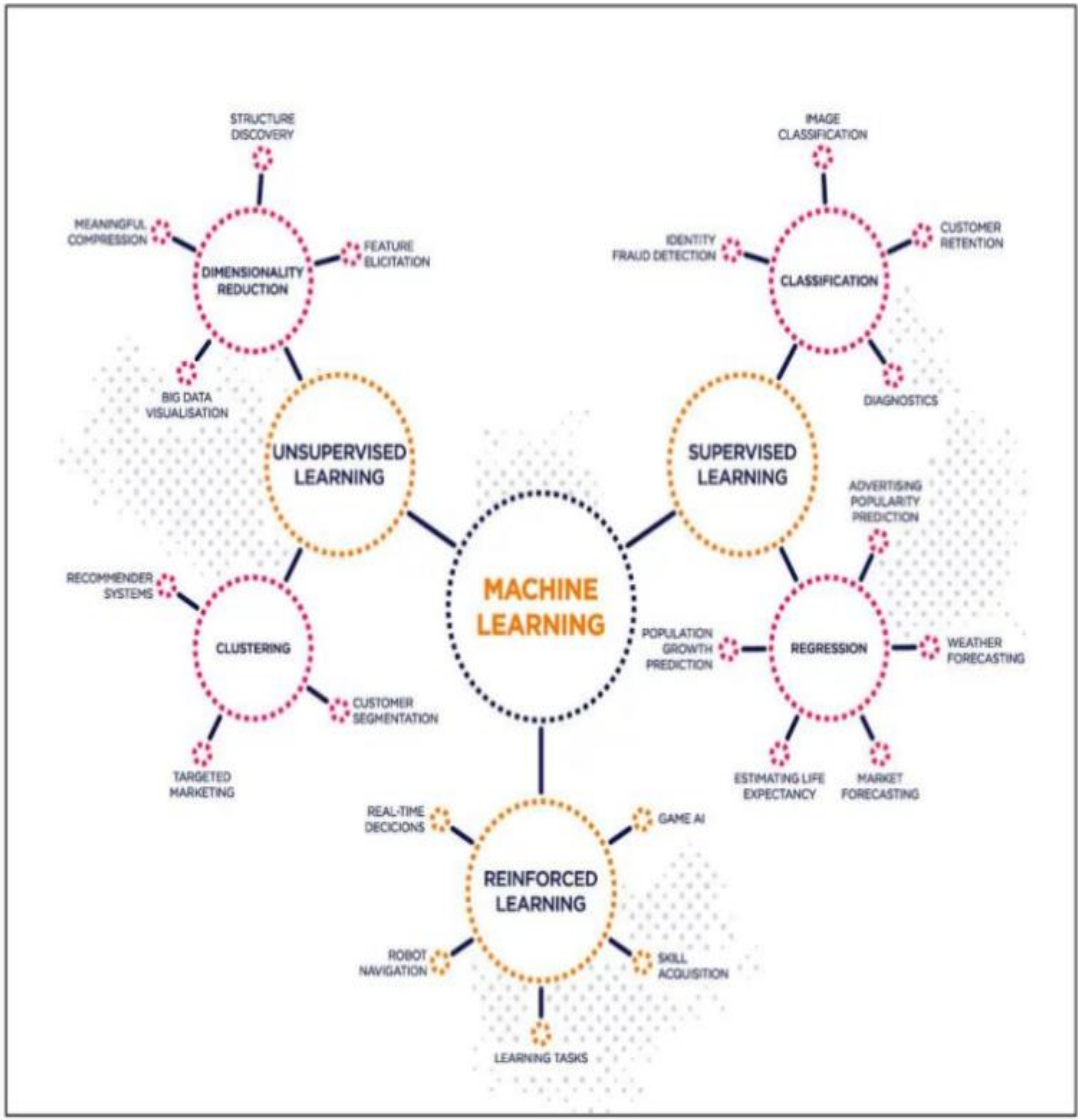
Machine learning life cycle

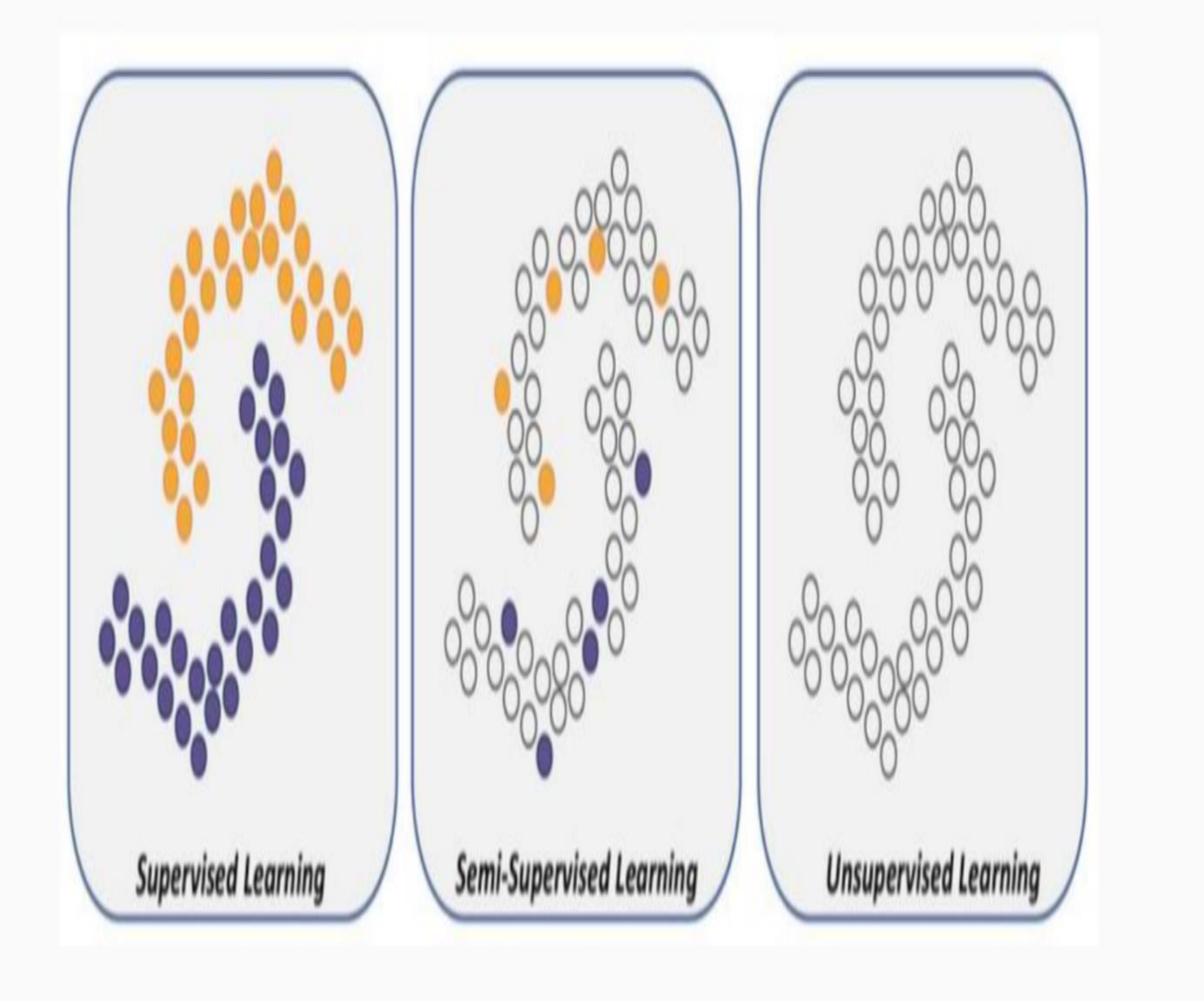
Machine learning life cycle involves seven major steps, which are given below:

1. Gathering Data
2. Data preparation
3. Data Wrangling
4. Analyse Data
5. Train the model
6. Test the model
7. Deployment



The most important thing in the complete process is to understand the problem and to know the purpose of the problem. Therefore, before starting the life cycle, we need to understand the problem because the good result depends on the better understanding of the problem.


MACHINE LEARNING
UNSUPERVISED LEARNING
SUPERVISED LEARNING
REINFORCED LEARNING
DIMENSIONALITY REDUCTION
STRUCTURE DISCOVERY
MEANINGFUL COMPRESSION
FEATURE ELICITATION
BIG DATA VISUALISATION
CLUSTERING
RECOMMENDER SYSTEMS
CUSTOMER SEGMENTATION
TARGETED MARKETING
CLASSIFICATION
IMAGE CLASSIFICATION
IDENTITY FRAUD DETECTION
CUSTOMER RETENTION
DIAGNOSTICS
REGRESSION
ADVERTISING POPULARITY PREDICTION
POPULATION GROWTH PREDICTION
WEATHER FORECASTING
ESTIMATING LIFE EXPECTANCY
MARKET FORECASTING
REAL-TIME DECICIONS
GAME AI
ROBOT NAVIGATION
SKILL ACQUISITION
LEARNING TASKS


Supervised Learning
Semi-Supervised Learning
Unsupervised Learning

Validation Techniques

Validation techniques in machine learning are used **to get the error rate of the ML model**, which can be considered as close to the true error rate of the population. If the data volume is large enough to be representative of the population, you may not need the validation techniques. In machine learning, there is always the need to test the stability of the model. It means based only on the training dataset; we can't fit our model on the training dataset. For this purpose, we reserve a particular sample of the dataset, which was not part of the training dataset. After that, we test our model on that sample before deployment. **Refer**: https://www.upgrad.com/blog/cross-validation-in-machine-learning/

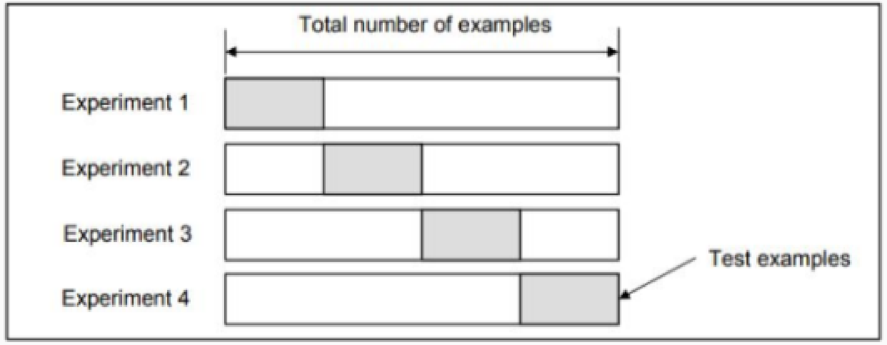
Resubstitution: If all the data is used for training the model and the error rate is evaluated based on outcome vs. actual value from the same training data set, this error is called the **resubstitution error.** This technique is called the resubstitution validation technique.

Holdout: To avoid the resubstitution error, the data is split into two different datasets labeled as a training and a testing dataset. This can be a 60/40 or 70/30 or 80/20 split. This technique is called the hold-out validation technique. In this case, there is a likelihood that uneven distribution of different classes of data is found in training and test dataset. To fix this, the training and test dataset is created with equal distribution of different classes of data. This process is called **stratification**.

Validation Techniques

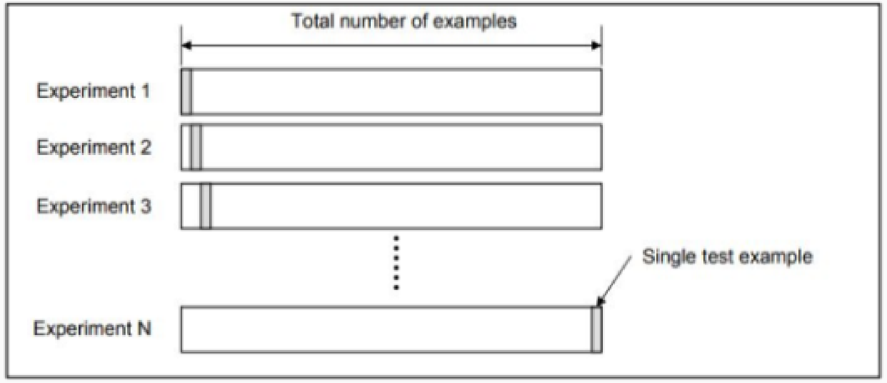
K-Fold Cross-Validation: In this technique, k-1 folds are used for training and the remaining one is used for testing as shown in the picture.

The advantage is that entire data is used for training and testing. The error rate of the model is average of the error rate of each iteration. This technique can also be called a form the repeated hold-out method. The error rate could be improved by using stratification technique.



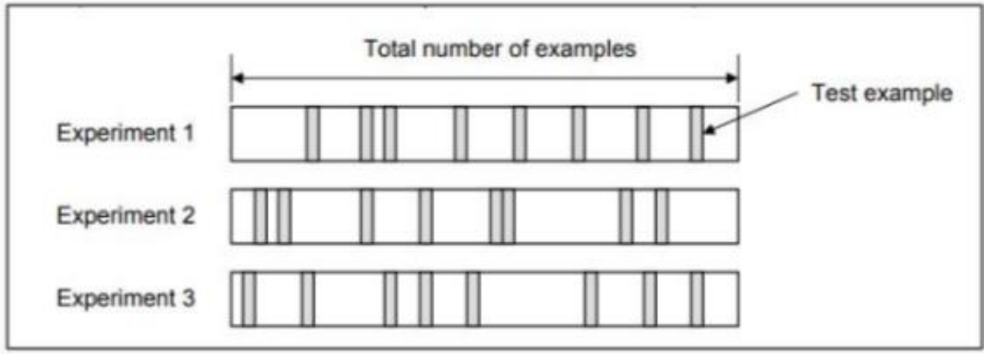
Leave-One-Out Cross-Validation (LOOCV)

In this technique, all of the data except one record is used for training and one record is used for testing. This process is repeated for N times if there are N records. The advantage is that entire data is used for training and testing. The error rate of the model is average of the error rate of each iteration. The following diagram represents the LOOCV validation technique.

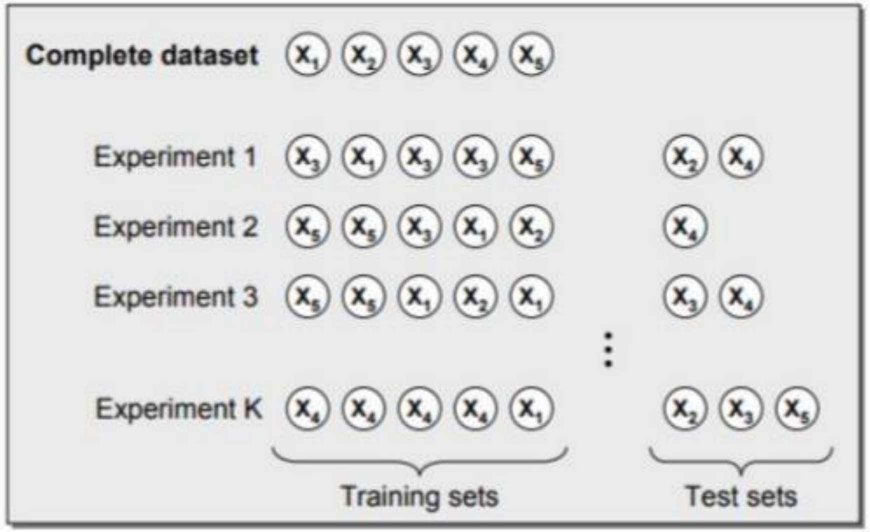


Validation Techniques

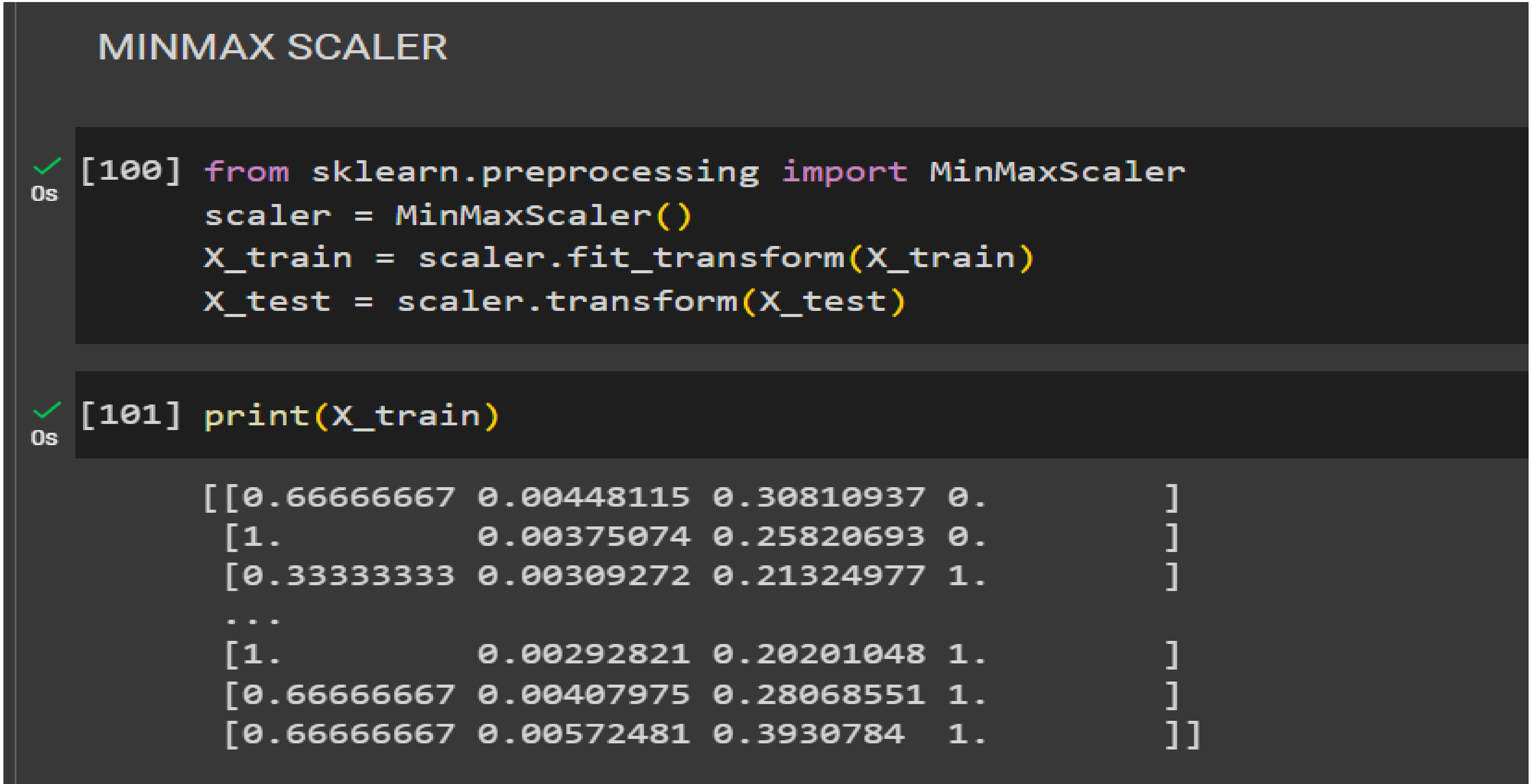
Random Subsampling: In this technique, multiple sets of data are **randomly chosen** from the dataset and combined to form a test dataset. The remaining data forms the training dataset. The following diagram represents the random subsampling validation technique. The error rate of the model is the average of the error rate of each iteration.



Bootstrapping: In this technique, the training dataset is randomly selected with replacement. The remaining examples that were not selected for training are used for testing. Unlike K-fold cross-validation, the value is likely to change from fold-to-fold. The error rate of the model is average of the error rate of each iteration. The following diagram represents the same.


Complete dataset
X_1 X_2 X_3 X_4 X_5
Experiment 1
X_3 X_1 X_3 X_3 X_5
X_2 X_4
Experiment 2
X_5 X_5 X_3 X_1 X_2
X_4
Experiment 3
X_5 X_5 X_1 X_2 X_1
X_3 X_4
Experiment K
X_4 X_4 X_4 X_4 X_1
X_2 X_3 X_5
Training sets
Test sets

MINMAX SCALER



MINMAX SCALER

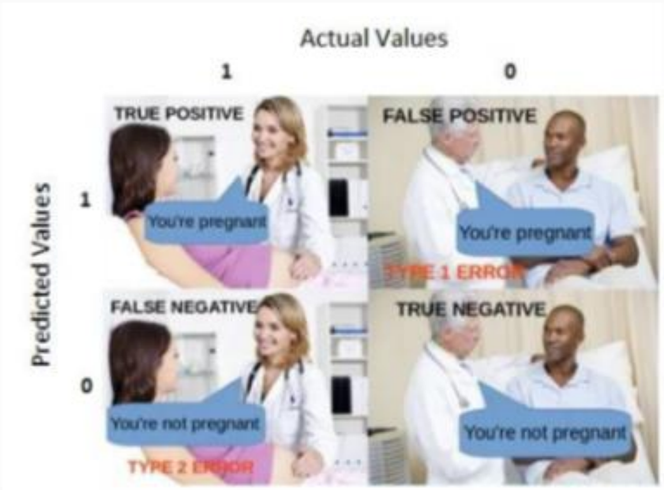
```
[100] from sklearn.preprocessing import MinMaxScaler
      scaler = MinMaxScaler()
      X_train = scaler.fit_transform(X_train)
      X_test = scaler.transform(X_test)
```

```
[101] print(X_train)
```

```
[[0.66666667 0.00448115 0.30810937 0.        ]
 [1.         0.00375074 0.25820693 0.        ]
 [0.33333333 0.00309272 0.21324977 1.        ]
 ...
 [1.         0.00292821 0.20201048 1.        ]
 [0.66666667 0.00407975 0.28068551 1.        ]
 [0.66666667 0.00572481 0.3930784  1.        ]]
```

Confusion Matrix

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an **error matrix**.



Refer: https://medium.com/analytics-vidhya/what-is-a-confusion-matrix-d1c0f8feda5

- The matrix is divided into two dimensions, that are **predicted values** and **actual values** along with the total number of predictions.
- **Predicted values** are those values, which are predicted by the model, and a**ctual values are the true values** for the given observations.

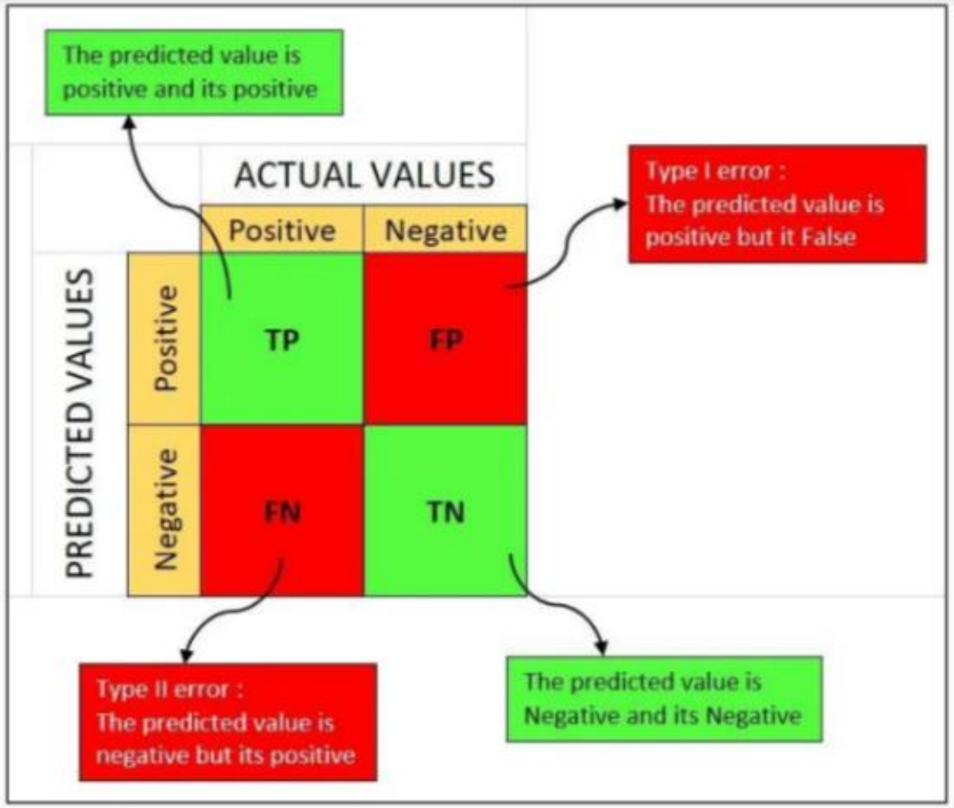
Confusion Matrix

True Positives (TP): when the actual value is Positive and predicted is also Positive.

True negatives (TN): when the actual value is Negative and prediction is also Negative.

False positives (FP): When the actual is negative but prediction is Positive. Also known as the **Type 1 error**

False negatives (FN): When the actual is Positive but the prediction is Negative. Also known as the **Type 2 error**



The predicted value is positive and its positive

Type I error : The predicted value is positive but it False

		ACTUAL VALUES	
		Positive	Negative
PREDICTED VALUES	Positive	TP	FP
	Negative	FN	TN

Type II error : The predicted value is negative but its positive

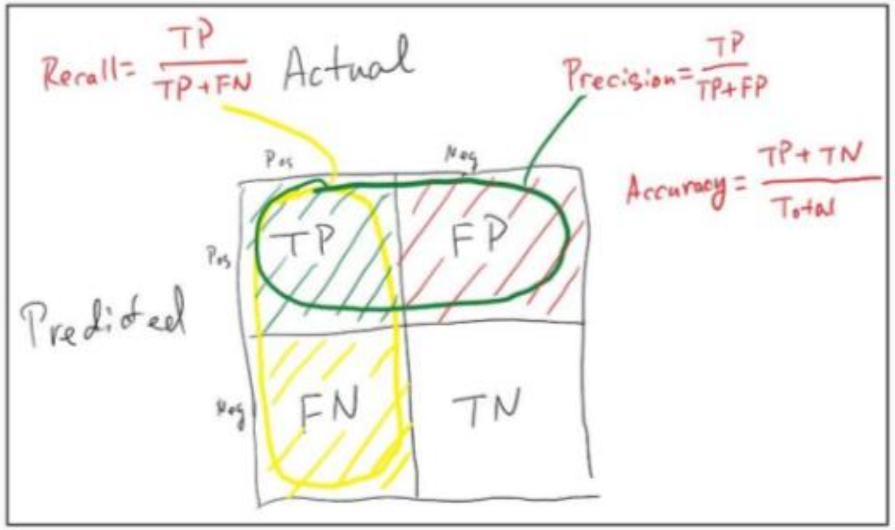
The predicted value is Negative and its Negative

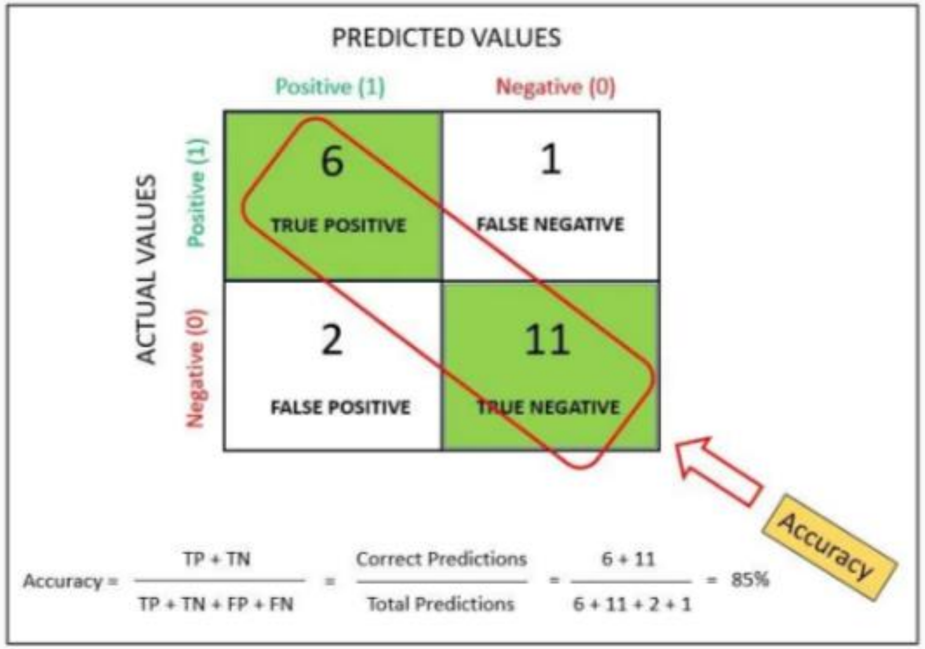
- It evaluates the **performance of the classification models**, when they make predictions on test data, and tells how good our classification model is.
- It not only tells the error made by the classifiers but also the **type of errors** such as it is either type-I or type-II error.

Confusion Matrix Parameters

With the help of the confusion matrix, we can calculate the different parameters for the model, such as:

Accuracy: It defines how often the model predicts the correct output. It can be calculated as the ratio of the number of correct predictions made by the classifier to all number of predictions made by the classifiers. The accuracy metric is not suited for **imbalanced classes**. It tells us how many predictions are actually positive out of all the total positive predicted. The formula is given below:



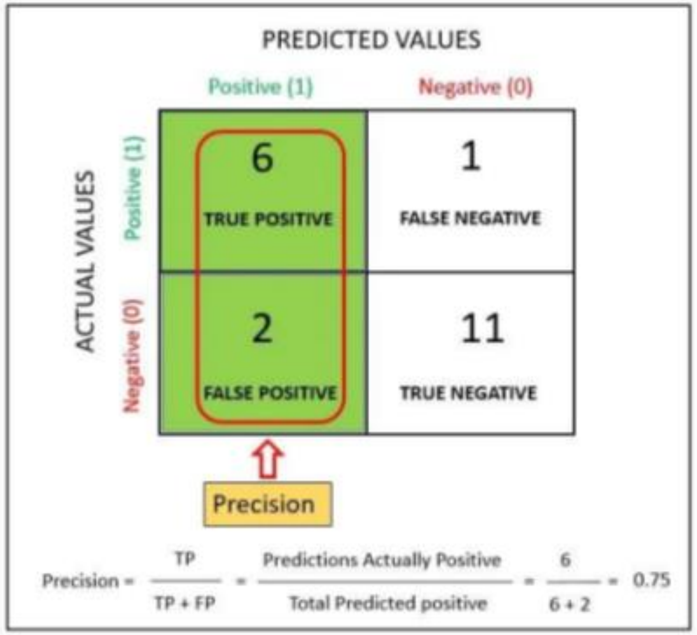


$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} = \frac{6 + 11}{6 + 11 + 2 + 1} = 85\%$$

Confusion Matrix Parameters

Precision: It can be defined as the number of correct outputs provided by the model or out of all positive classes that have predicted correctly by the model, how many of them were actually true. In simple words, it tells us how many predictions are actually positive out of all the total positive predicted.

It can be calculated using the below formula:



PREDICTED VALUES

		Positive (1)	Negative (0)
ACTUAL VALUES	Positive (1)	6 TRUE POSITIVE	1 FALSE NEGATIVE
	Negative (0)	2 FALSE POSITIVE	11 TRUE NEGATIVE

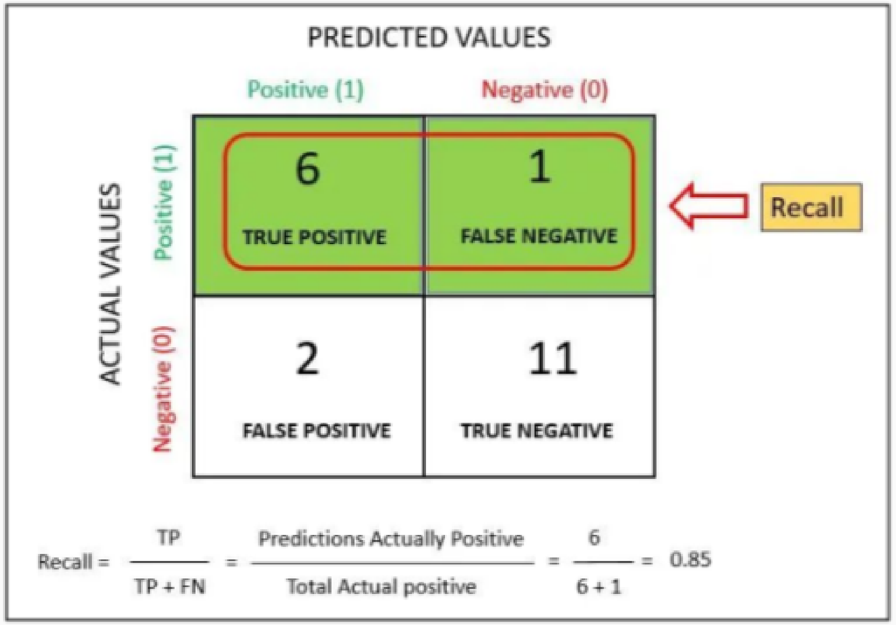
$$\text{Precision} = \frac{TP}{TP + FP} = \frac{\text{Predictions Actually Positive}}{\text{Total Predicted positive}} = \frac{6}{6 + 2} = 0.75$$

Ex1:- **In Spam Mail Detection**: Need to focus on precision

Ex2:- Precision is important in **music or video recommendation systems, e-commerce websites**, etc. Wrong results could lead to customer churn and be harmful to the business.

Confusion Matrix Parameters

Recall: It is defined as the out of total positive classes, how our model predicted correctly. The recall must be as high as possible. It is a measure of actual observations which are predicted correctly, i.e. how many observations of positive class are actually predicted as positive. It is also known as **Sensitivity**.



PREDICTED VALUES

ACTUAL VALUES		Positive (1)	Negative (0)
	Positive (1)	6 TRUE POSITIVE	1 FALSE NEGATIVE
	Negative (0)	2 FALSE POSITIVE	11 TRUE NEGATIVE

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{\text{Predictions Actually Positive}}{\text{Total Actual positive}} = \frac{6}{6+1} = 0.85$$

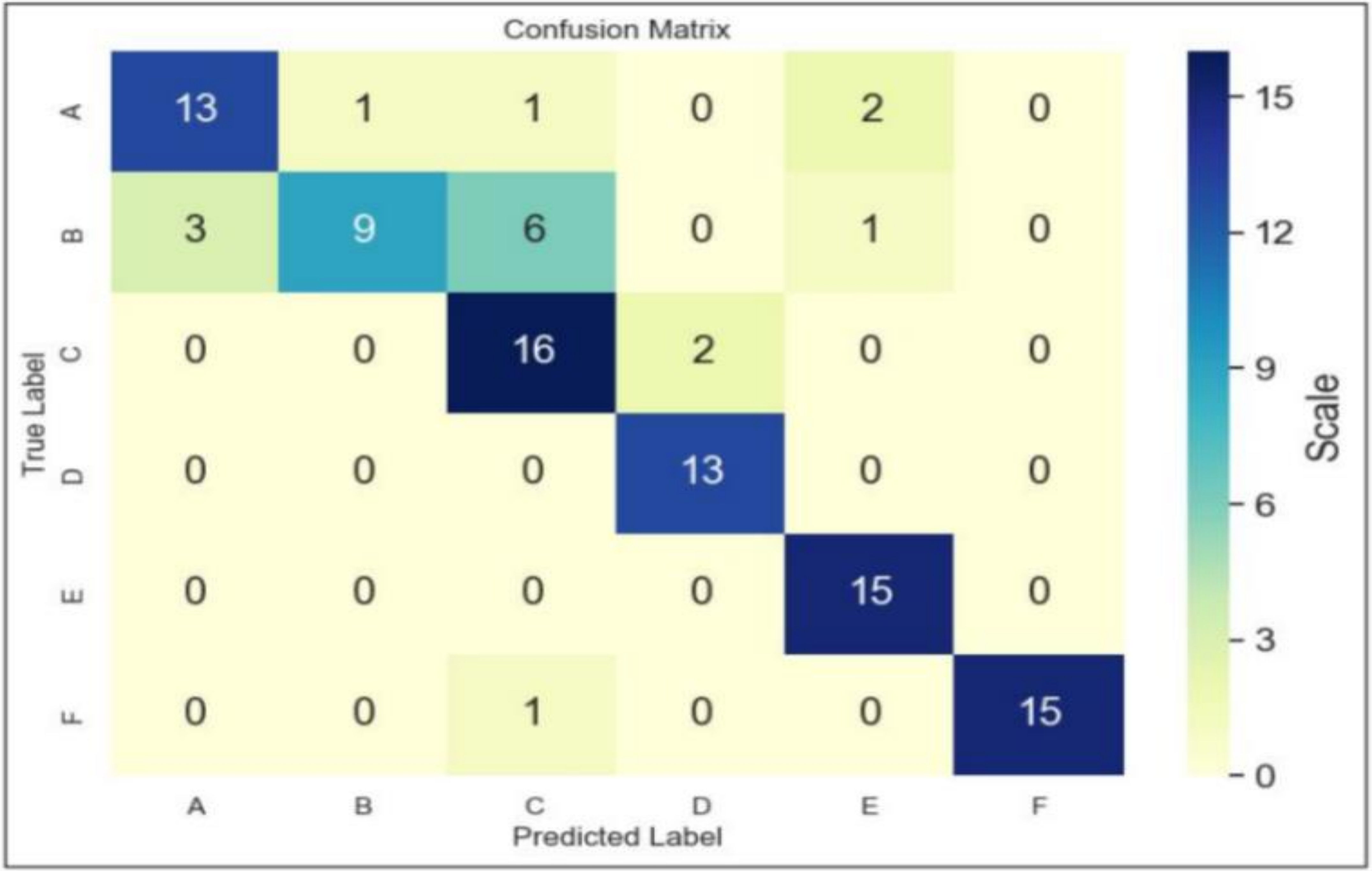
Ex 1:- suppose person having cancer (or) not?
He is suffering from cancer but model predicted as not suffering from cancer

Ex 2:- Recall is important in medical cases where it doesn’t matter whether we raise a false alarm but the actual positive cases should not go undetected!

Confusion Matrix Parameters

F-measure / F1 Score: If two models have low precision and high recall or vice versa, it is difficult to compare these models. So, for this purpose, we can use F-score. This score helps us to **evaluate the recall and precision at the same time.** The F-score is maximum if the recall is equal to the precision. F1 score is a harmonic mean of Precision and Recall. As compared to Arithmetic Mean, Harmonic Mean punishes the extreme values more. **F-score should be high (ideally 1)**. It can be calculated using the below formula:

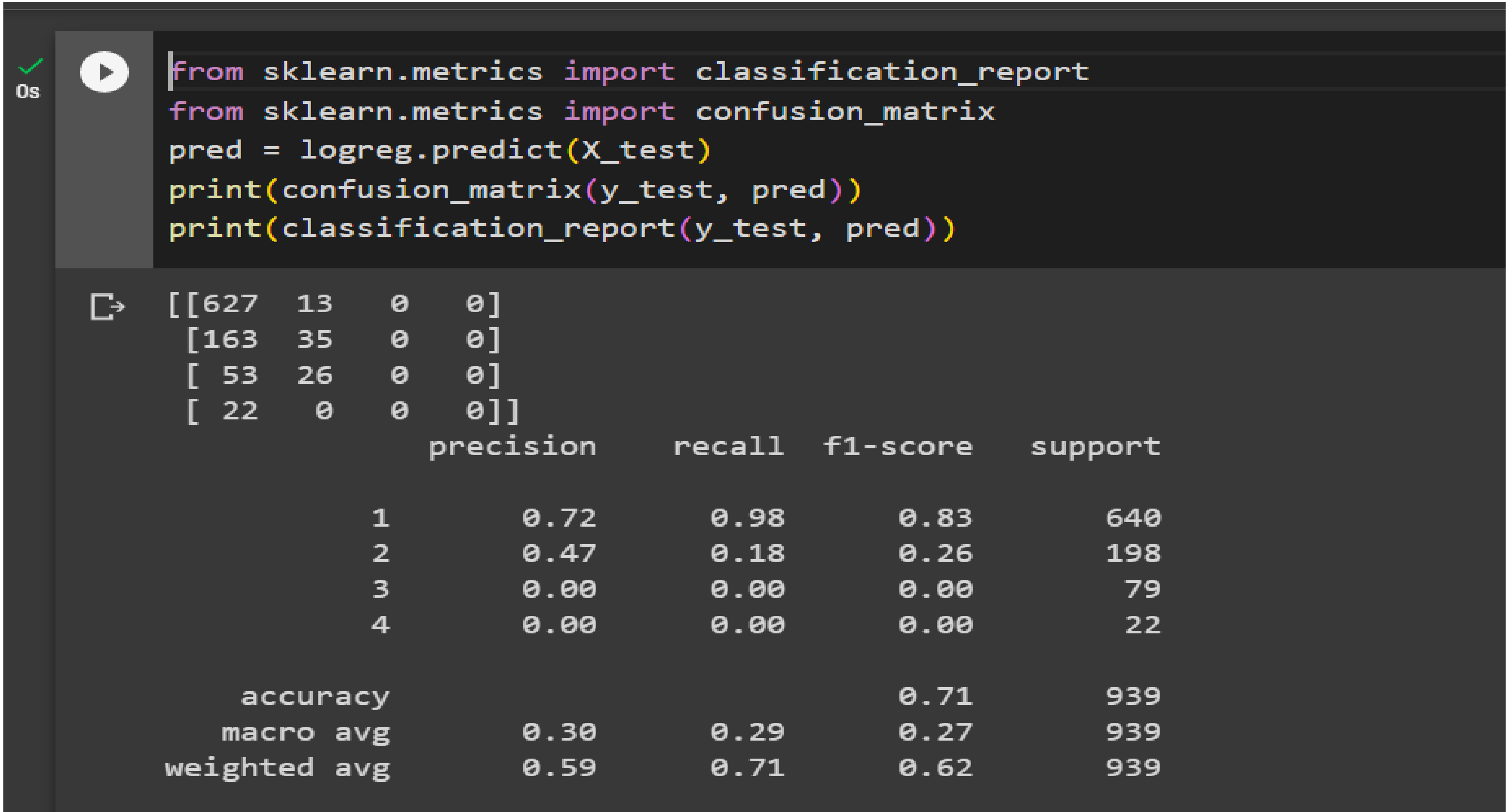
$$\text{F1-Score} = 2^* \frac{(\text{Recall}^*\text{Precision})}{(\text{Recall} + \text{Precision})} = 2^* \frac{(0.85^*0.75)}{(0.85 + 0.75)} = 0.79$$



Confusion Matrix

	A	B	C	D	E	F
A	13	1	1	0	2	0
B	3	9	6	0	1	0
C	0	0	16	2	0	0
D	0	0	0	13	0	0
E	0	0	0	0	15	0
F	0	0	1	0	0	15

CONFUSION MATRIX OF DATASET



```python
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
pred = logreg.predict(X_test)
print(confusion_matrix(y_test, pred))
print(classification_report(y_test, pred))
```

```
[[627  13   0   0]
 [163  35   0   0]
 [ 53  26   0   0]
 [ 22   0   0   0]]
              precision    recall  f1-score   support

           1       0.72      0.98      0.83       640
           2       0.47      0.18      0.26       198
           3       0.00      0.00      0.00        79
           4       0.00      0.00      0.00        22

    accuracy                           0.71       939
   macro avg       0.30      0.29      0.27       939
weighted avg       0.59      0.71      0.62       939
```

THANK YOU