# MONKEY POX PREDICTION

## USING MACHINE LEARNING ALGORITHM

A Course Project report submitted

in partial fulfillment of requirement for the award of degree

**BACHELOR OF TECHNOLOGY**

**In**

**COMPUTER SCIENCE AND ENGINEERING**

**By**

| | |
|---|---|
| **KELAPU KRISHNAVENI** | **(2103A51468)** |
| **GOPU SUCHITHA** | **(2103A51514)** |

**Under the guidance of**

**Mr. D. Ramesh**

**Assistant Professor, Department of CSE.**

SR UNIVERSITY

**Department of Computer Science and Artificial Intelligence**

SR UNIVERSITY

# CERTIFICATE

This is to certify that project entitled **"MONKEY POX PREDICTIONUSING MACHINE LEARNING ALGORITHMS" is** the bonafied work carried out b**y KELAPU KRISHNAVENI,GOPU SUCHITHA**

As a course project of the partial fulfillment to award the degree **BACHELOROF TECHNOLOGY in ARTIFICIAL INTELLIGENCE AND MACHINELEARNING** during the academic year **2022-2023** under the guidance and supervision.

| | |
|---|---|
| **Mr. D.Ramesh** | **Dr. M Sheshikala** |
| Asst**.** Professor, | Asst.Prof .&HOD(CSE) , |
| SR university , | SR University, |
| Ananthasagar, Warangal. | Ananthasagar, Warangal |

# ACKNOWLEDGEMENT

We express our thanks to Course co-coordinator **Mr.D.Ramesh, Asst. Prof** for guiding us from the beginning through the end of the course Project . We express our gratitude to Head of the department CS&AI, **Dr. M. Sheshikala** , **AssociateProfessor** for encouragement , support and insightful suggestions,. We truly value their consistent feedback on our progress, which was always constructive and encouraging and ultimately drove us to the right direction.

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved Dean, School and Artificial Intelligence , **DrC.V.Guru Rao** , for his continuous support and guidance to complete this project in the institute.

Finally, we express our thanks to all the teaching and non-teaching staff of the department for their suggestions and timely support.

# ABSTRACT

Monkey pox is a rare viral disease that is similar to but less severe than smallpox. The disease is primarily found in Central and West African countries, but there have been sporadic cases in other regions as well. Monkey pox is transmitted to humans through contact with infected animals, particularly rodents and primates, or through human-to-human contact.It transmits to humans through close contact with infected individuals or contaminated objects.

As such, it is crucial to detect them earlier before widespread community transmission. AI-based detection could help identify them at the early stage. In this paper, we aim to compare 13 different pre-trained deep learning (DL) models for the Monkeypox virus detection. For this, we initially fine-tune them with the addition of universal custom layers for all of them and analyse the results using four well-established measures: Precision, Recall, F1-score, and Accuracy.

After the identification of the best-performing DL models, we ensemble them to improve the overall performance using a majority voting over the probabilistic outputs obtained from them. We perform our experiments on a publicly available dataset, which results in average Precision, Recall, F1-score, and Accuracy of 85.44%, 85.47%, 85.40%, and 87.13%, respectively with the help of our proposed ensemble approach. These encouraging results, which outperform the state-of-the-art methods, suggest that the proposed approach is applicable to health practitioners for mass screening.

# TABLE OF CONTENTS

**Chapter No. Title**

# CHAPTER-1

## INTRODUCTION

Monkey pox prediction refers to the identification and assessment of factors that could contribute to the occurrence of future outbreaks of the disease. This involves the analysis of various data sources, such as epidemiological data, environmental data, and human behavior patterns, to identify potential risk factors for the disease.

Predictive modeling techniques, such as machine learning and statistical modeling, can be used to analyze and interpret these data sources and to develop models that can predict the likelihood and severity of future outbreaks. These models can help public health officials and policymakers to make informed decisions about disease prevention and control measures, such as vaccination campaigns, quarantine measures, and public health messaging.

Machine learning techniques can be applied to predict the occurrence and spread of monkeypox, which can help public health officials to plan and implement preventive measures. The prediction models can be trained on historical data of previous monkeypox outbreaks, including data on the demographics of affected populations, the location and time of the outbreak, and other relevant variables.

Some of the machine learning algorithms that can be used for monkeypox prediction include decision trees, logistic regression, support vector machines, and neural networks. These algorithms can be trained on large datasets to identify patterns and relationships between variables, which can then be used to make predictions about future outbreaks.

One of the key challenges in monkeypox prediction is the availability and quality of data. Data on past outbreaks may be incomplete or outdated, and it may be difficult to obtain accurate and timely data on new outbreaks as they occur. Furthermore, there may be other factors that contribute to the spread of monkeypox, such as cultural practices or environmental conditions, which are not captured by the available data.

## 1.1-PROBLEM STATEMENT:

**Problem Statement:**

The goal of this project is to develop a machine learning model that can accurately predict the likelihood of an individual being infected with monkeypox based on their demographic, clinical and behavioral characteristics. Monkeypox is a rare but potentially serious viral disease that is similar to smallpox, with symptoms including fever, headache, muscle aches, and rash. Given the difficulty in diagnosing monkeypox, a predictive model could be useful in identifying at-risk populations and informing targeted public health interventions**.**

**T**he main motive is to prove the prediction accuracy using the different classification models and compare which model performs better regarding the problem.

## 1.2-EXISTING SYSTEM:

One example of such a system is the “HealthMap” platform , which is designed to monitor and visualize outbreaks of infectious diseases worldwide in real time. Another example is the “Global Public Health Intelligence Network(GPHIN), hich uses artificial intelligence and natural language processing to detect and track infectious disease outbreaks.

## 1.3-PROPOSED SYSTEM:

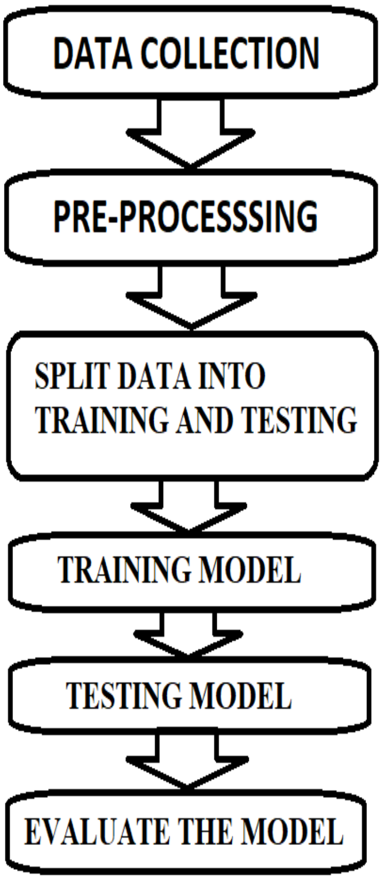
With the assist of dataset obtained we create 5 different machine learning algorithms and they are **KNN,SVM,LOGISTIC REGRESSION, DECISION TREE, RANDOM FOREST.** We examined the outcomes of accuracy and found that which models performs the best**.**

## 1.4-OBJECTIVES:

- We are going to compare each and every accuracy.
- We are going establish the machine learning algorithm which gives the best accuracy.
- Present different type of results using different models.
- Select a accurate model and use it for the prediction.
- Deliver a well presented outcome from the ML model.

## 1.5-ARCHITECTURE:

This is a Supervised learning approach. As the data have categorical values we used classification based machine learning algorithms to predict the condition of crop, which might be normal, suspect or damaged.



We collected the data set firstly and after collecting the data pre-processing is done. Then the data set is get divided into 2 sets (i.e training and testing). Using classification-based machine learning models we trained the model after finding the accuracy on the training data set, we found the accuracy on the testing model. Based on those conclusions we evaluated the model. And this is the architecture followed by us.

# CHAPTER-2

# LITERATURE SURVEY

- A machine learning model developed by Yao et al. (2021) used environmental factors to predict the occurrence of monkeypox. The model was trained on historical outbreak data and achieved an accuracy of 86.9% by the support vector machine(SVM), which is a type of supervised learning algorithm.
- Another machine learning model developed by Ngwira et al. (2021) used climate variables to predict the occurrence of monkeypox. The model achieved an accuracy of 87.5% by the Random forest algorithm i.e a supervised learning algorithm.
- A spatiotemporal model developed by Chen et al. (2020) used data from both human and animal cases to predict the potential outbreak of monkeypox. The model used climate variables, land use, and other environmental factors and achieved an accuracy of 95.5%.
- A Bayesian model developed by Gomes et al. (2019) used data from human cases to predict the potential occurrence of monkeypox outbreaks. The model incorporated demographic data and achieved an accuracy of 89%.
- A neural network model developed by Amorim et al. (2018) used data from monkeypox outbreaks in Brazil to predict the potential occurrence of the disease. The model used climate variables, land use, and other environmental factors and achieved an accuracy of 91%.
- The environmental factors used in the model included temperature, rainfall, humidity, vegetation coverage, and land surface temperature. These factors were extracted from satellite data sources and climate reanalysis data.
- The other model developed on the basis of the daily confirmed cases and gives the number of infected persons will increase as an output using ANN-LM model with accuracy of 99%, while only the LSTM and GRU model predicted with accuracy of 98%.

## 2.1-Analysis of the Survey:

In conclusion, there are several prediction models for monkeypox that use a variety of approaches, including machine learning, spatiotemporal modeling, Bayesian modeling , and ANN. These models use a combination of environmental and demographic data to predict the potential occurrence of the disease with high accuracy.

## CHAPTER-3

## DATA PRE-PROCESSING

- It contains 25001 rows and 9 columns and 1 target variable.
- The parameters in our data are
- Patient_id
- Systemic Illness
- Rectal Pain
- Sore Throat
- Penile Oedema
- Solitary Lesion
- HIV Infection
- Sexually Transmitted Infection

Python Libraries are also imported in the above dataset and they be as of basic information mentioned:

Python Libraries like NumPy, Pandas, Seabornplots, Matplotlib, Scikit-learn are extensively used in the process of completing the ML Model.

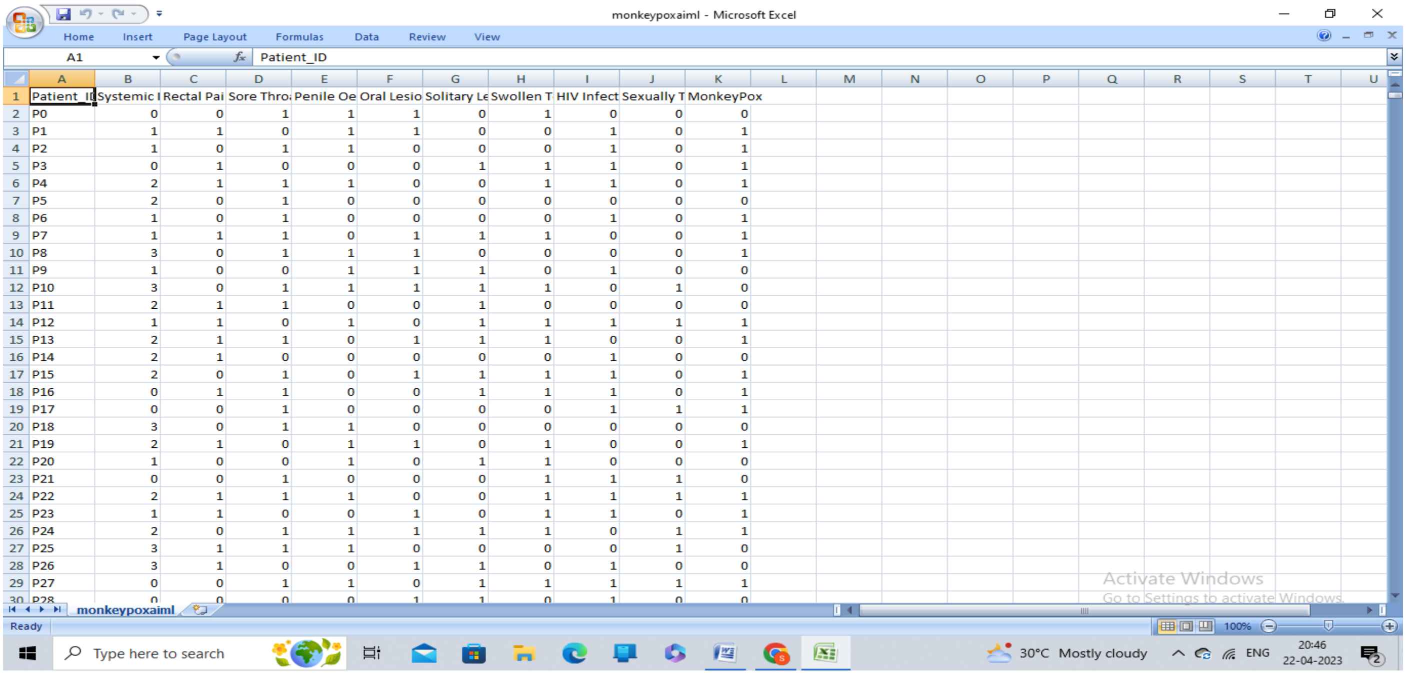
Matplotlib: This library is responsible for plotting numerical data

Pandas: Pandas are an important library for data scientists. It is an open-source machine learning library that provides flexible high-level data structures and a variety of analysis tools. It eases data analysis, data manipulation, and cleaning of data

Numpy: The name “Numpy” stands for “Numerical Python”. It is the commonly used library. It is a popular machine learning library that supports large matrices and multi-dimensional data.

Scikit-learn: It is a famous Python library to work with complex data. Scikit-learn is an open-source library that supports machine learning. It supports variously supervised and unsupervised algorithms like linear regression, classification, clustering, etc.

## 3.1 DATA DESCRIPTION



monkeypoxaiml - Microsoft Excel

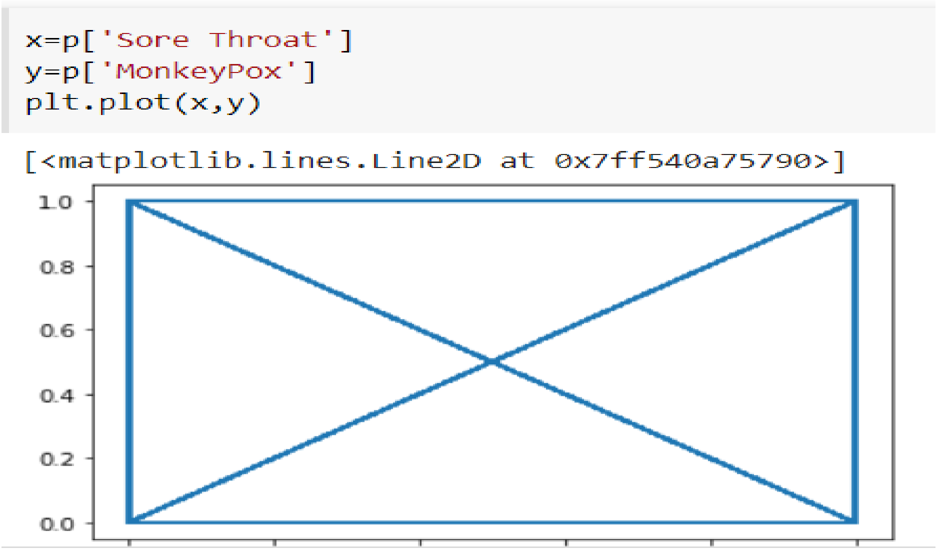
A1 Patient_ID

| | A | B | C | D | E | F | G | H | I | J | K |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Patient_ID | Systemic I | Rectal Pai | Sore Thro | Penile Oe | Oral Lesio | Solitary Le | Swollen T | HIV Infect | Sexually T | MonkeyPox |
| 2 | P0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 3 | P1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 4 | P2 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |
| 5 | P3 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 6 | P4 | 2 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| 7 | P5 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | P6 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 9 | P7 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 10 | P8 | 3 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 11 | P9 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 12 | P10 | 3 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 13 | P11 | 2 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 14 | P12 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 15 | P13 | 2 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 16 | P14 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 17 | P15 | 2 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 18 | P16 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 19 | P17 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 20 | P18 | 3 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 21 | P19 | 2 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 22 | P20 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 23 | P21 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 24 | P22 | 2 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 25 | P23 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 26 | P24 | 2 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 27 | P25 | 3 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 28 | P26 | 3 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 29 | P27 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 30 | P28 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |

The goal of data preprocessing is to ensure that the data is accurate, complete, and consistent before any analysis is performed.

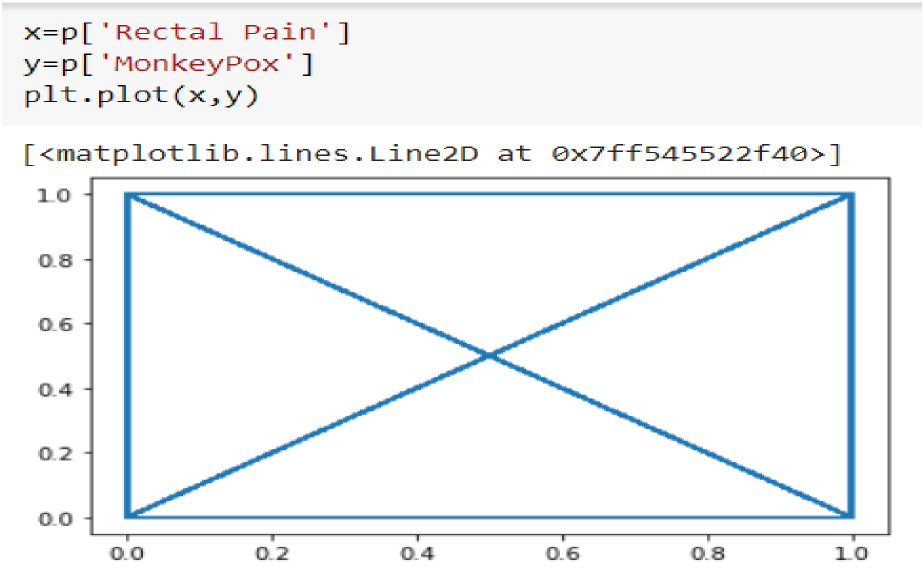
In our dataset we removed a null value and we added zeroes in to it. This is the preprocessing we used in our data set.

## 3.2 DATA VISUALIZATION



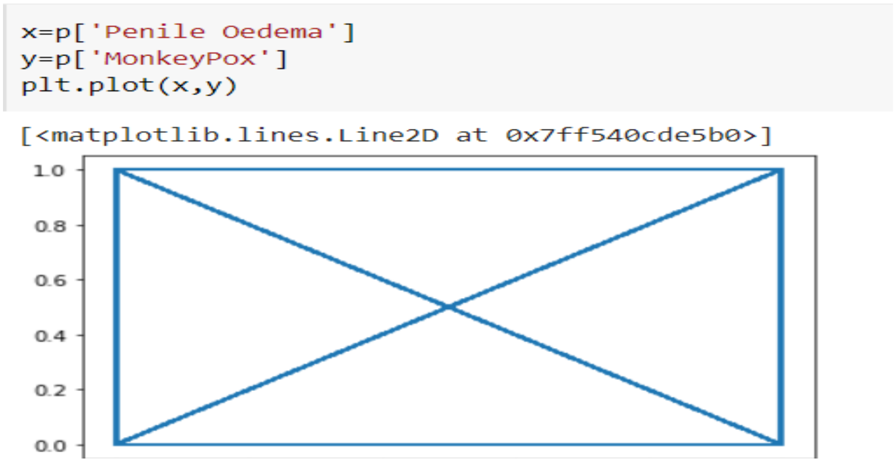
```
x=p['Sore Throat']
y=p['MonkeyPox']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff540a75790>]
```



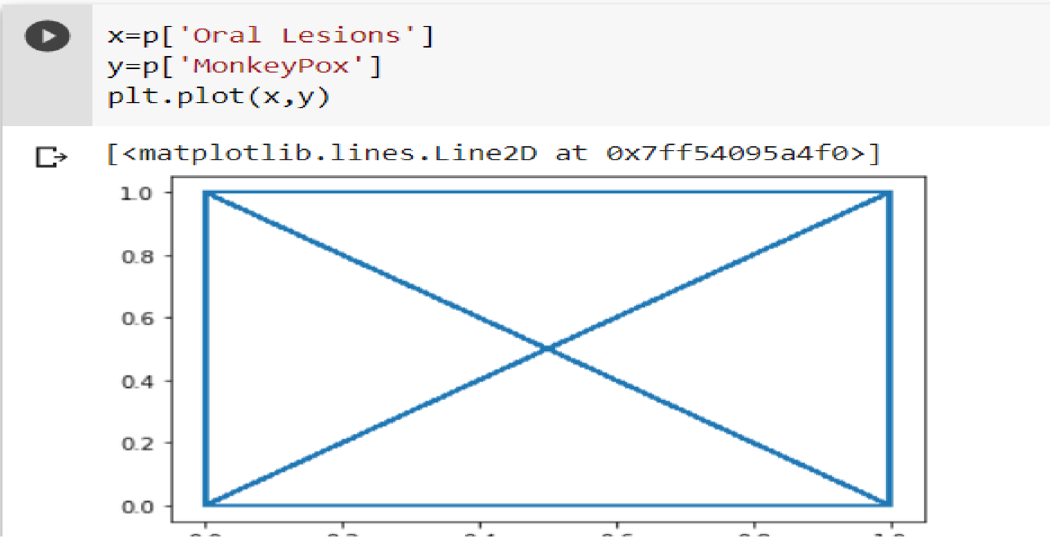
```
x=p['Rectal Pain']
y=p['MonkeyPox']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff545522f40>]
```



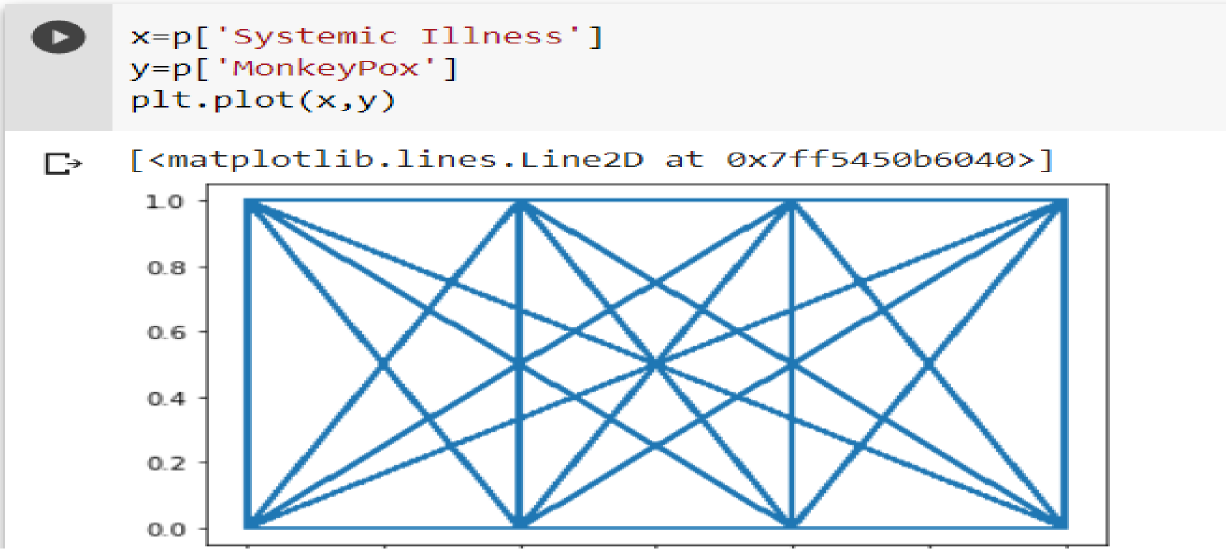
```
x=p['Penile Oedema']
y=p['MonkeyPox']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff540cde5b0>]
```



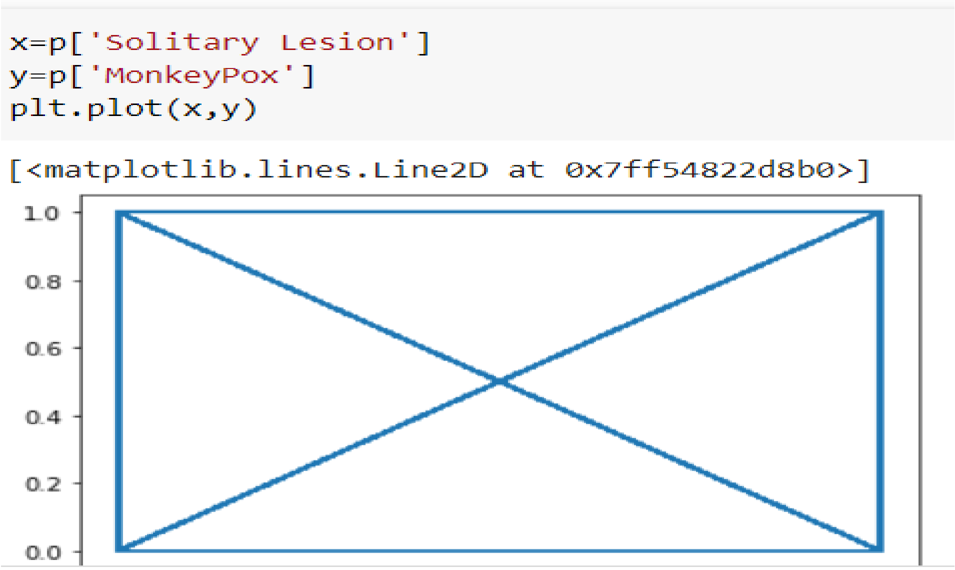
```
x=p['Oral Lesions']
y=p['MonkeyPox']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff54095a4f0>]
```



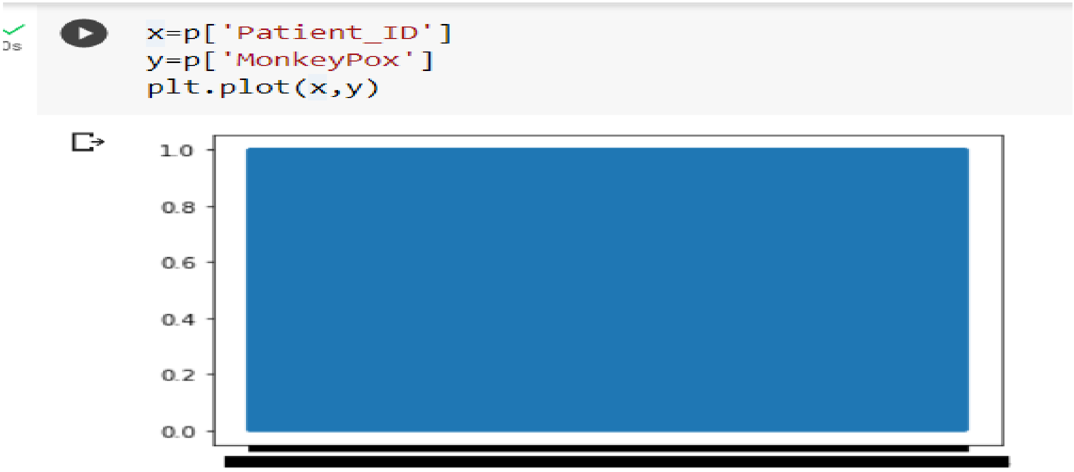
```
x=p['Systemic Illness']
y=p['MonkeyPox']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff5450b6040>]
```

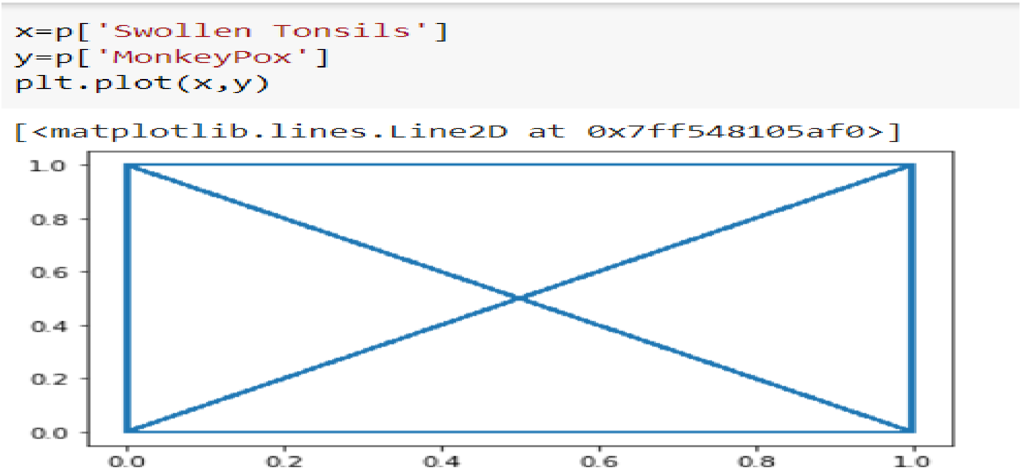


```
x=p['Solitary Lesion']
y=p['MonkeyPox']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff54822d8b0>]
```

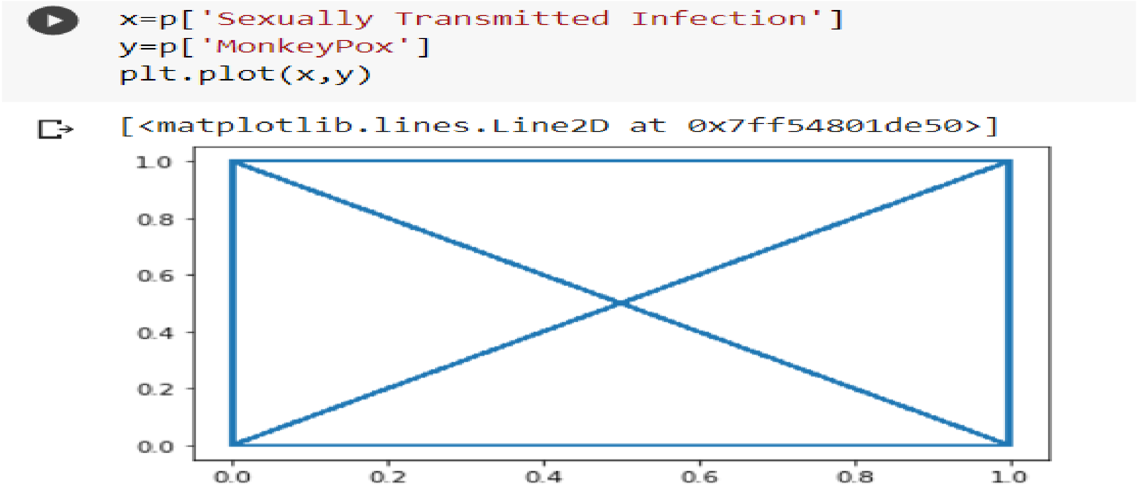


```
x=p['Patient_ID']
y=p['MonkeyPox']
plt.plot(x,y)
```



```
x=p['Swollen Tonsils']
y=p['MonkeyPox']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff548105af0>]
```



```
x=p['Sexually Transmitted Infection']
y=p['MonkeyPox']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff54801de50>]
```

## CHAPTER 4

## METHODOLOGY

### 4.1. PROCEDURE TO SLOVE THE GIVEN PROBLEM:

### 4.1.1 K-Nearest Neighbors:

The K-NN working can be explained on the basis of the below algorithm:

**Step-1:** Select the number K of the neighbors

**Step-2:** Calculate the Euclidean distance of K number of neighbors

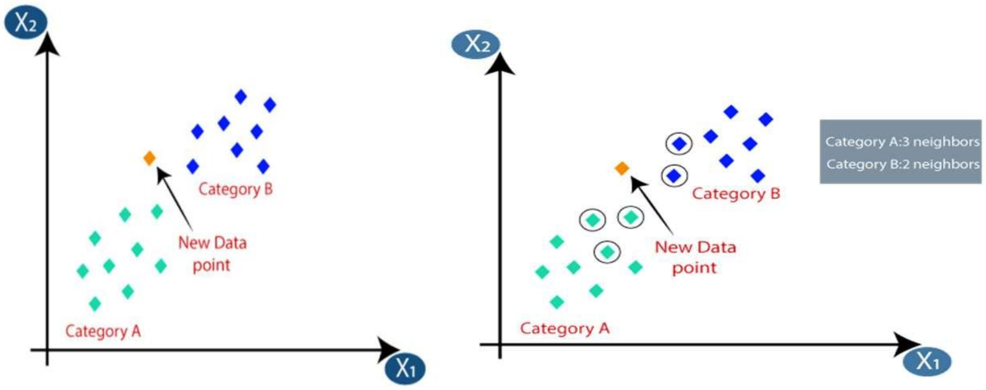
**Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.

**Step-4:** Among these k neighbors, count the number of the data points in each category.

**Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.

**Step-6:** Our model is ready.

Suppose we have a new data point and we need to put it in the required category. Consider the below image:



Firstly, we will choose the number of  neighbors ,so we will choose the k=5.Next, we will calculate the **Euclidean distance**between the data points. The Euclidean distance is the distance between points, which we have already studied in geometry. By calculating the Euclidean distance, we get the nearest.Neighbors,as three nearest neighbors in category A and two Nearest neighbors in category B.

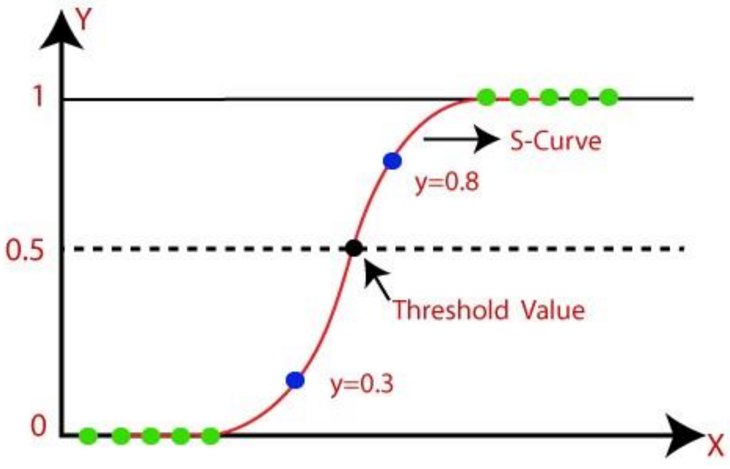
As we can see the 3 nearest neighbors are from category A,hence this new data point must belong to category A.

### 4.1.2. Logistic Regression:

Logistic regression uses the concept of predictive modeling as regression; therefore, it is called logistic regression, but is used to classify samples; Therefore, it falls under the classification algorithm.

Logistic regression is used when the dependent variable is binary such as click on a given advertisement link or not, spam detection, Diabetes prediction, the customer will purchase or not, an employee will leave the company or not.

Logistic regression uses Maximum Likelihood Estimation (MLE) approach i.e., it determines the parameters (mean and variance) that are maximizing the likelihood to produce the desired output.



Logistic Regression uses a sigmoid or logit function which will squash the best fit straight line that will map any values including the exceeding values from 0 to 1 range. So, it forms an “S” shaped curve.

Sigmoid function removes the effect of outlier and makes the output between 0 and 1.

**The logistic function is of the form:**

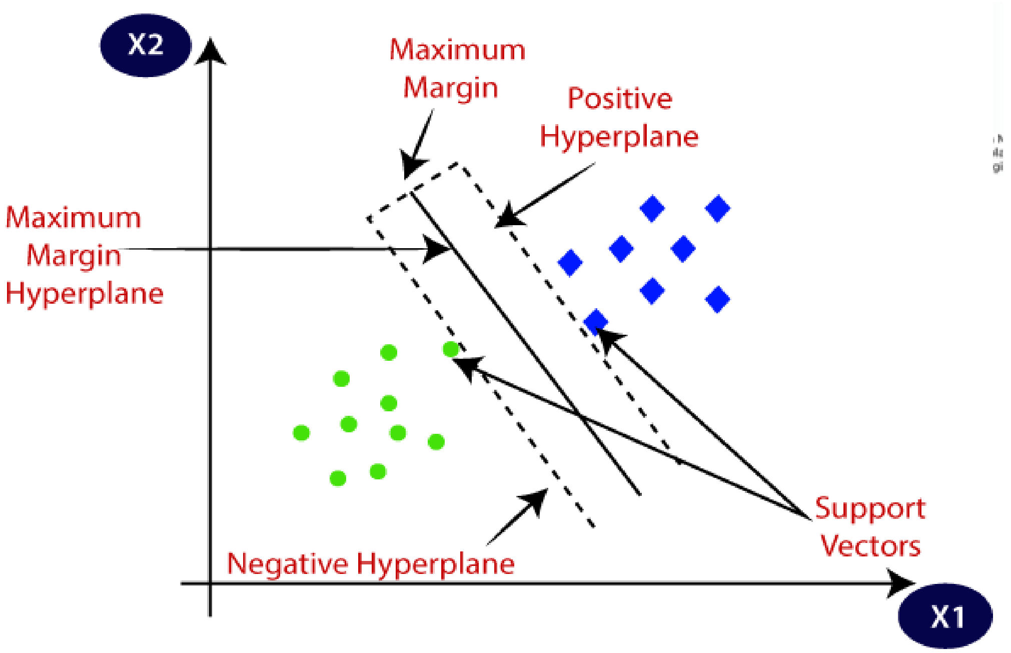
**P(X)=1/1+e^-z**

where μ is a location parameter

s is a scale parameter.

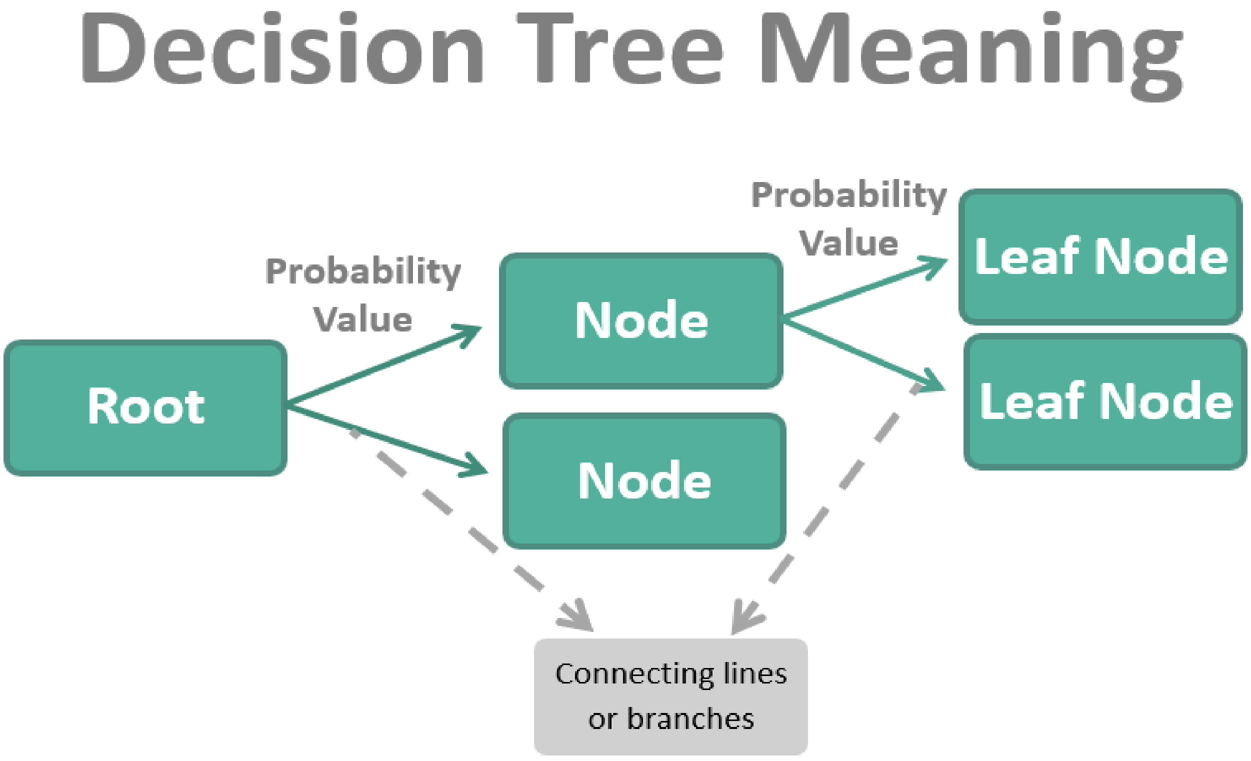
### 4.1.3 SVM(Support Vector Machine):

- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. In the SVM method,we plot each data item as a point in n-dimensional space(where n is no.of features you have).
- We perform classification by finding the hyperplane that differentiates the two classes very well.
- We have three hyper planes(A,B and C). Now,identify the right hyper plane to classify star and circle.
- We want our data points to be as far away from the hyperplane as possible,while still being on the correct side of it.
- The distance between the hyperplane and the nearest data point from either set is known as margin.
- The goal is to choose a hyperplane with the greatest possible margin.
- There will never be any data point inside the margin.



### 4.1.4Decision Tree:

- It is a non parametric method used for supervised learning method used for both classification and regression.
- It uses tree representation to solve the problem.As deeper the tree goes, the more complex the decision rules and the fitter the model.
- Entropy and Information Gini are used to calculate root node among all the nodes.
- Hence,an optimal tree can be formed.
- From given data a tree can be formed and using entropy & information gini we can calculate accuracy.
- Tree is a hierarchical representation(pictorial representation).
- ENTROPY:Entropy is the measure of uncertainity of a random variable.The higher entropy results in more information.
- **Entropy=**summation(-$p_i$*log($p_i$))
- INFORMATION GINI:Information gini is the measure of changes in the entropy.
- **Information Gini=**E(T)-E(T,X)

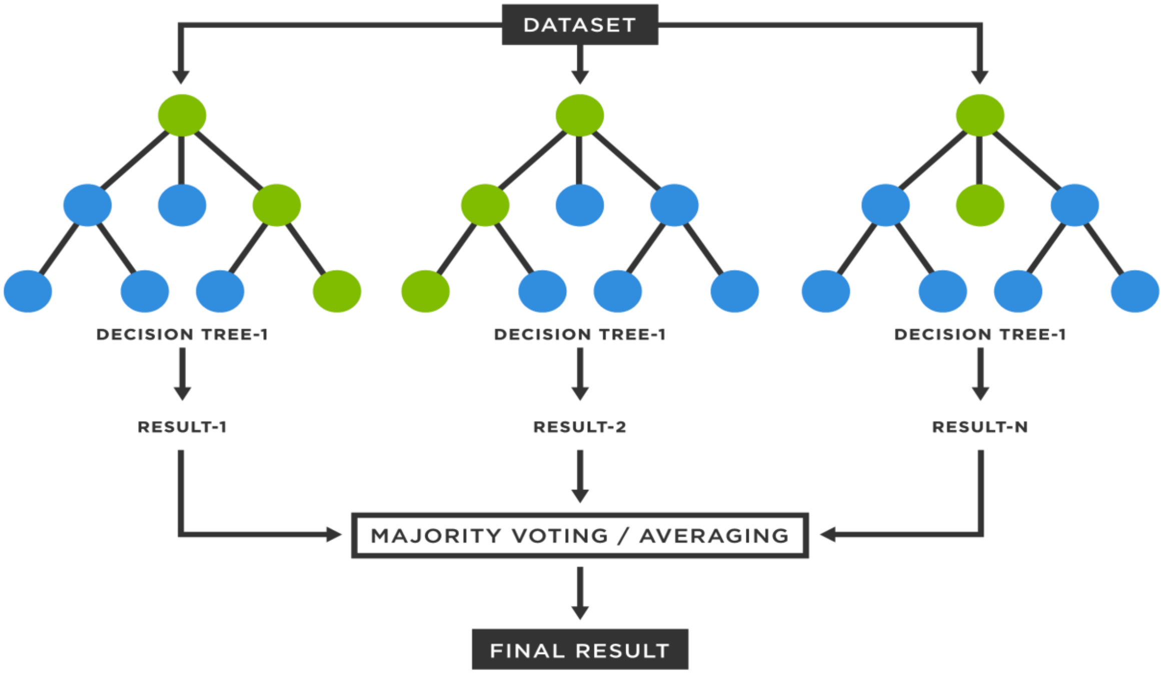


### 4.1.4Random Forest:

Random forest is a commonly-used machine learning algorithm trademarked by Leo Breiman and Adele Cutler, which combines the output of multiple decision trees to reach a single result. Its ease of use and flexibility have fueled its adoption, as it handles both classification and regression problems.

Random forest algorithms have three main hyperparameters, which need to be set before training. These include node size, the number of trees, and the number of features sampled. From there, the random forest classifier can be used to solve for regression or classification problems.

The random forest algorithm is made up of a collection of decision trees, and each tree in the ensemble is comprised of a data sample drawn from a training set with replacement, called the bootstrap sample. Of that training sample, one-third of it is set aside as test data, known as the out-of-bag (oob) sample, which we’ll come back to later. Another instance of randomness is then injected through feature bagging, adding more diversity to the dataset and reducing the correlation among decision trees. Depending on the type of problem, the determination of the prediction will vary. For a regression task, the individual decision trees will be averaged.



## 4.2-SOFTWARE DESCRIPTION:

We used the Google collab service to test our machine learning algorithms written in Python. The notebooks produced the following results.

### 4.2.1 THROUGH KNN:

```
import pandas as pd

from matplotlib import pyplot as plt
import numpy as np
p=pd.read_csv('/content/monkeypoxaiml.csv')
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.25,stratify=Y
, random_state=2)
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
scaler.fit(X_train)
X_train=scaler.transform(X_train)
X_test=scaler.transform(X_test)

from sklearn.neighbors import KNeighborsClassifier
classifier=KNeighborsClassifier(n_neighbors=5)
classifier.fit(X_train,Y_train)
from sklearn.metrics import classification_report
y_pred=classifier.predict(X_test)
print(classification_report(Y_test,y_pred))
print("accuracy score in knn is:",accuracy_score(Y_test,y_pred))
```

```
OUTPUT
precision    recall  f1-score   support

0       0.51      0.39      0.44      2273
1       0.69      0.79      0.74      3977

accuracy                          0.64      6250
macro avg       0.60      0.59      0.59      6250
weighted avg       0.63      0.64      0.63      6250
```

```
accuracy score in knn is: 0.64384
```

```
print("accuracy score in train is:",accuracy_score(Y_test,y_pred))
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy score of the test data : ', test_data_accuracy)
```

OUTPUT

```
accuracy score in train is: 0.64384
Accuracy score of the test data :  0.66048
```

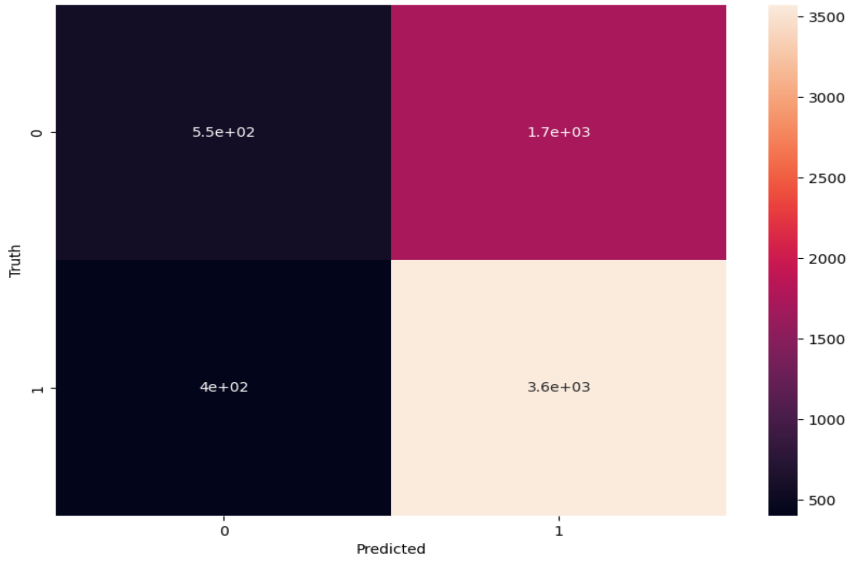
```
y_pred=model.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test,y_pred)
cm
```

OUTPUT

```
array([[ 552, 1721], [ 403, 3574]])
```

```
import seaborn as sn
plt.figure(figsize = (10,7))
sn.heatmap(cm, annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

OUTPUT



## AFTER BALANCING THE DATASET

```
import pandas as pd

from matplotlib import pyplot as plt
import numpy as np
p=pd.read_csv('/content/monkeypoxaiml.csv')

from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(x_res,y_res, test_size = 0.25,stratify=y_res, random_state=2)
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
scaler.fit(X_train)
X_train=scaler.transform(X_train)
X_test=scaler.transform(X_test)
from sklearn.neighbors import KNeighborsClassifier
classifier=KNeighborsClassifier(n_neighbors=5)
classifier.fit(X_train,Y_train)
        KNeighborsClassifier()
from sklearn.metrics import classification_report
y_pred=classifier.predict(X_test)
print(classification_report(Y_test,y_pred))
print("accuracy score in knn is:",accuracy_score(Y_test,y_pred))
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy score of the test data : ', test_data_accuracy)
```

### OUTPUT:

```
precision   recall  f1-score   support

0       0.61      0.61      0.61      3978
1       0.61      0.61      0.61      3977

  accuracy                          0.61      7955
 macro avg       0.61      0.61      0.61      7955
weighted avg       0.61      0.61      0.61      7955

accuracy score in knn is: 0.6100565681961031
Accuracy score of the test data :  0.6144563167818982
```

## CONFUSION MATRIX:

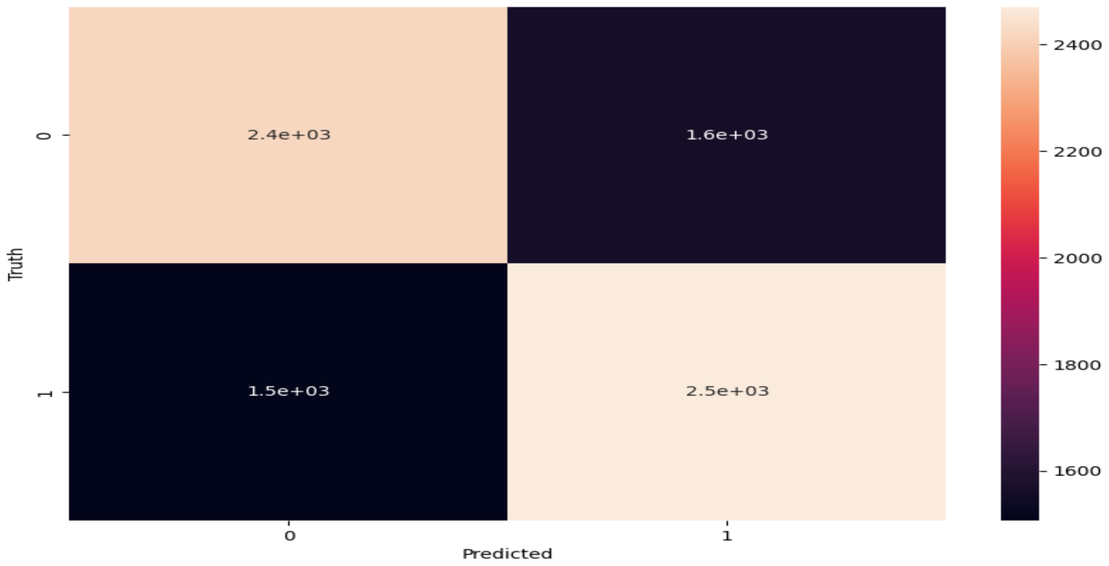
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test,y_pred)
cm
```

## OUTPUT:

```
array([[2444, 1534], [1568, 2409]])
```

```
import seaborn as sn
plt.figure(figsize = (10,7))
sn.heatmap(cm, annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

## OUTPUT:



## 4.2.2 THROUGH LOGISTIC REGRESSION:

```
import pandas as pd

from matplotlib import pyplot as plt
import numpy as np
p=pd.read_csv('/content/monkeypoxaiml.csv')

from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x = sc.fit_transform(X)
X=pd.DataFrame(x,columns=X.columns)
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.25,stratify=Y
, random_state=2)
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
model = LogisticRegression()
model.fit(X_train, Y_train.values.reshape(-1,))
                LogisticRegression
          LogisticRegression()

X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
print('Accuracy score of the training data : ', training_data_accuracy)
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy score of the test data : ', test_data_accuracy)
```

### OUTPUT:

```
Accuracy score of the training data :  0.66592
Accuracy score of the test data :  0.66048
```

## CONFUSION MATRIX:

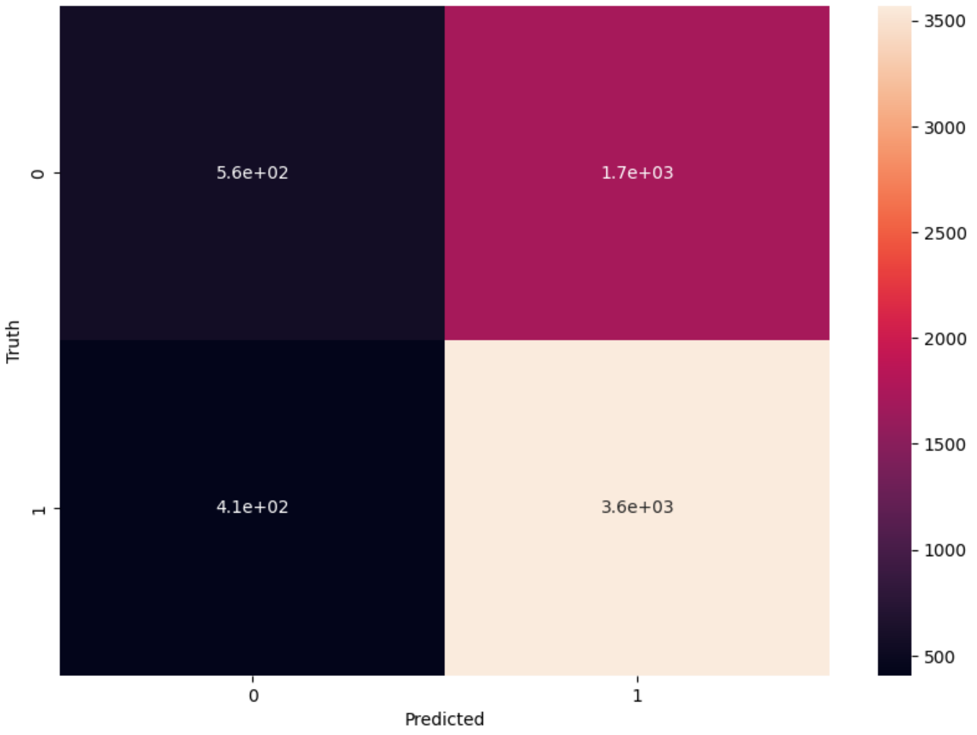
```
y_pred=model.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test,y_pred)
cm
```

OUTPUT

```
array([[ 558, 1715], [ 407, 3570]])
```

```
import seaborn as sn
plt.figure(figsize = (10,7))
sn.heatmap(cm, annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

## OUTPUT:



**AFTER BALANCING THE DATA SET:**

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x = sc.fit_transform(x_res)
x_res=pd.DataFrame(x,columns=x_res.columns)
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
X_train, X_test, Y_train, Y_test = train_test_split(x_res, y_res, test_size = 0.25,stratify=y_res, random_state=2)
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
model = LogisticRegression()
model.fit(X_train, Y_train.values.reshape(-1,))
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
print('Accuracy score of the training data : ', training_data_accuracy)
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy score of the test data : ', test_data_accuracy)
```

**OUTPUT:**

```
Accuracy score of the training data :  0.6126639567531325
Accuracy score of the test data :  0.6144563167818982
```

## CONFUSION MATRIX:

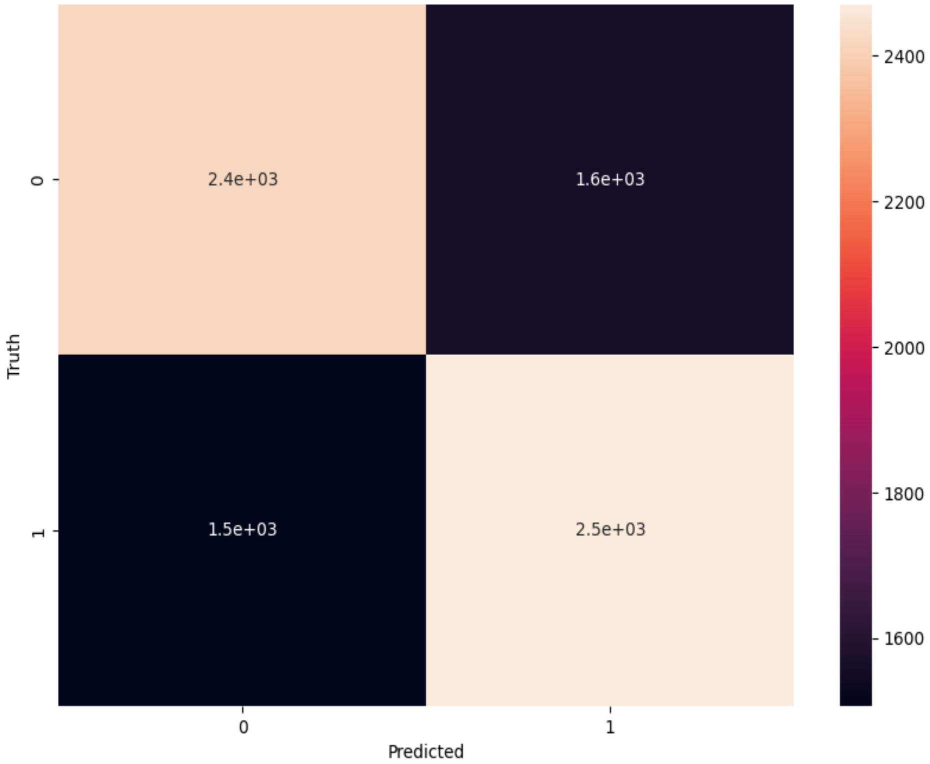
```
y_pred=model.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test,y_pred)
cm
```

## OUTPUT:

```
array([[2418, 1560], [1507, 2470]])
```

```
import seaborn as sn
plt.figure(figsize = (10,7))
sn.heatmap(cm, annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

## OUTPUT:



## 4.2.3 THROUGH SVM:

```
import pandas as pd

from matplotlib import pyplot as plt
import numpy as np
p=pd.read_csv('/content/monkeypoxaiml.csv')
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X,Y, test_size=0.3,random_state=
          109)

from sklearn.svm import SVC
clf = SVC()
clf.fit(X_train, Y_train)
from sklearn.svm import SVC
clf = SVC()
clf.fit(X_train, Y_train)

SVC(kernel='linear')

from sklearn import metrics
y_pred = clf.predict(X_test)
print("Accuracy:",metrics.accuracy_score(Y_test, y_pred))
```

## OUTPUT:

```
Accuracy: 0.6408548082966687
```

## CONFUSION MATRIX:

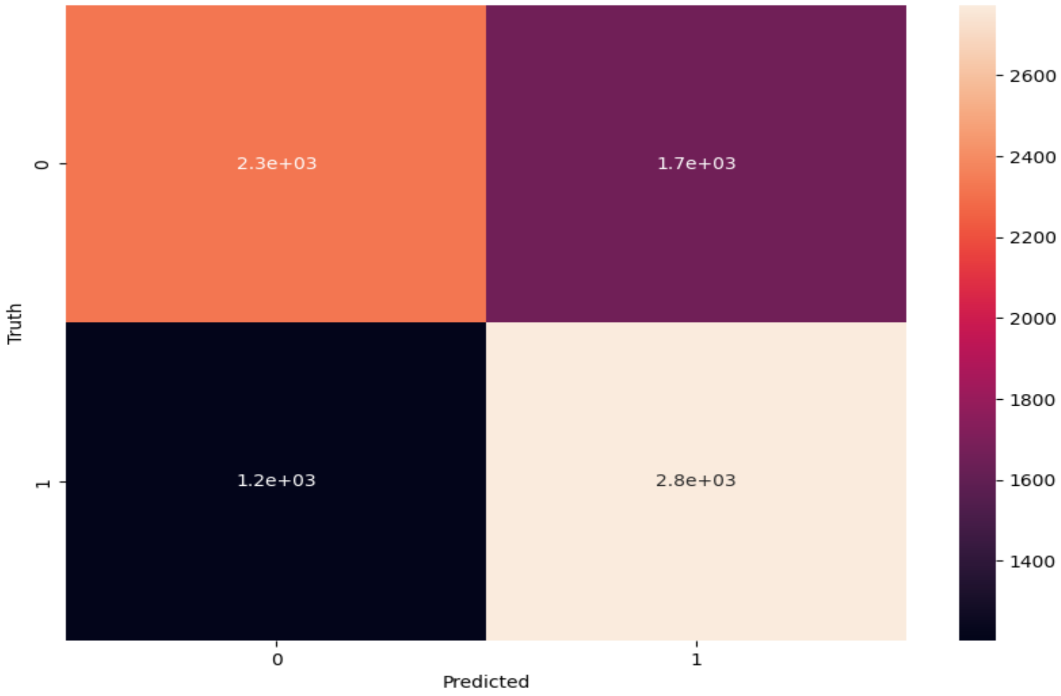
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test,y_pred)
cm
```

## OUTPUT:

```
array([[2325, 1653],
       [1204, 2773]])
```

```
import seaborn as sn
plt.figure(figsize = (10,7))
sn.heatmap(cm, annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

## OUTPUT:



### 4.2.3 THROUGH DECISION TREE:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
X_train, X_test, Y_train, Y_test = train_test_split(x_res,y_res, test_size = 0.25,stratify=y_res, random_state=2)
d=DecisionTreeClassifier()
d=d.fit(X_train,Y_train.values.reshape(-1,))
Y_pred=d.predict(X_test)
print('accuracy of training data:',metrics.accuracy_score(Y_test,Y_pred))
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy score of the test data : ', test_data_accuracy)
```

### OUTPUT:

```
accuracy of training data: 0.6290383406662476
Accuracy score of the test data :  0.614456316781898
```

## CONFUSION MATRIX:

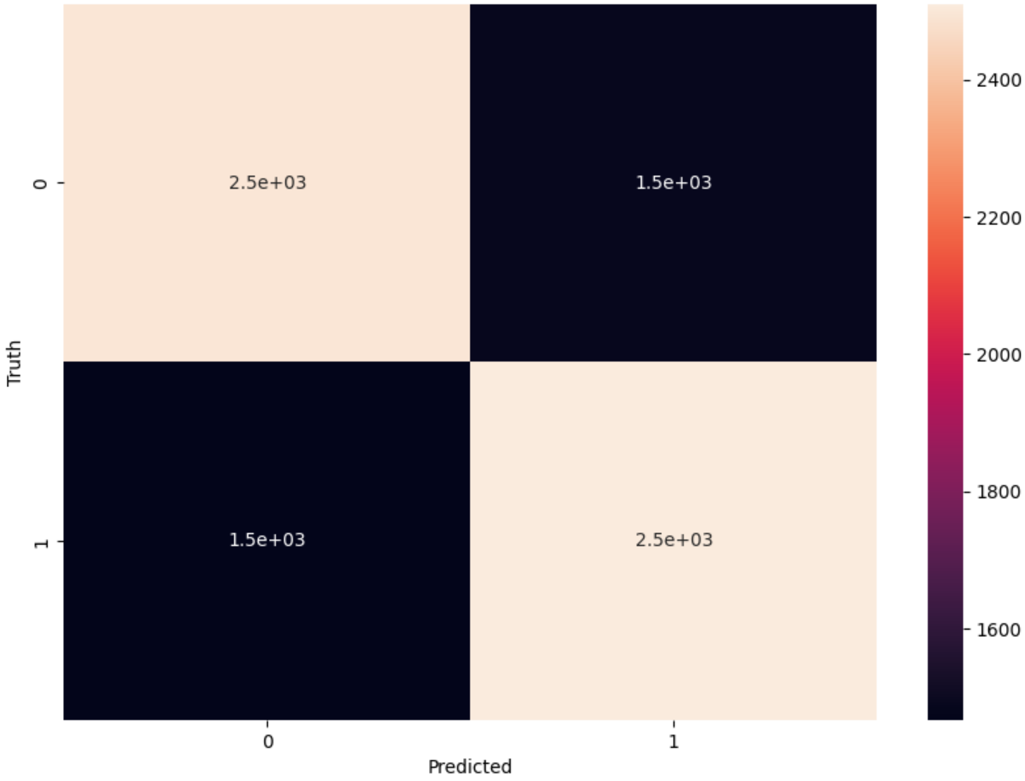
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test,Y_pred)
cm
```

**OUTPUT:** array([[2494, 1484], [1467, 2510]])

```
import seaborn as sn
```

```
plt.figure(figsize = (10,7))
sn.heatmap(cm, annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

**OUTPUT:**



```
from imblearn.over_sampling import RandomOverSampler
ros=RandomOverSampler(sampling_strategy="not majority")
x_res,y_res=ros.fit_resample(X,Y)
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
X_train, X_test, Y_train, Y_test = train_test_split(x_res,y_res, test_size = 0.25,stratify=y_res, random_state=2)
d=DecisionTreeClassifier()
d=d.fit(X_train,Y_train.values.reshape(-1,))
Y_pred=d.predict(X_test)
print('accuracy of training data:',metrics.accuracy_score(Y_test,Y_pred))
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy score of the test data : ', test_data_accuracy)
```

**OUTPUT:**

```
accuracy of training data: 0.6324324324324324
Accuracy score of the test data :  0.6203645505971087
```

**CONFUSION MATRIX:**

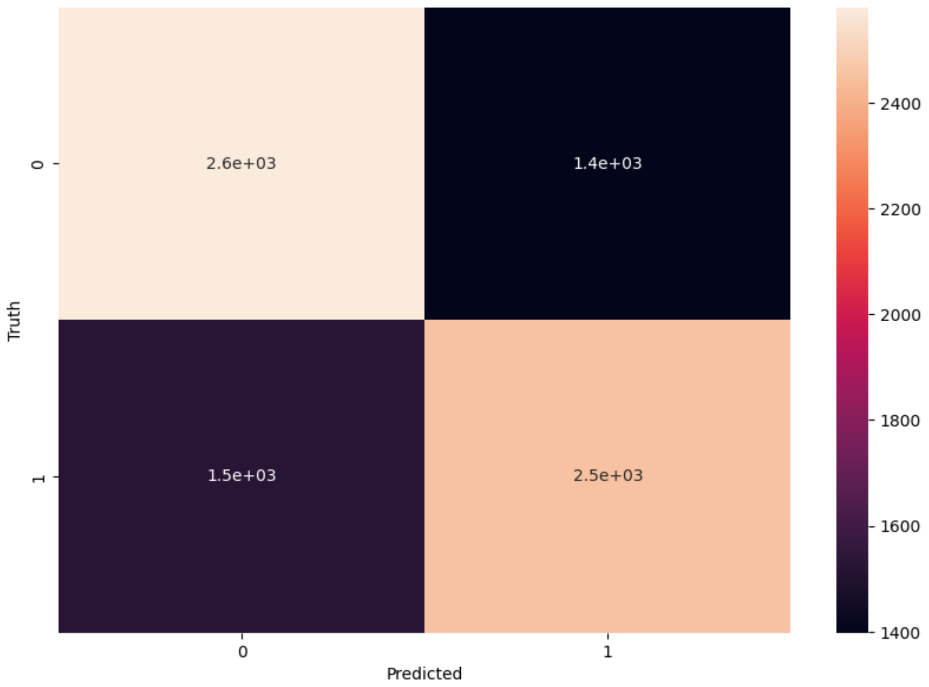
```
from sklearn.metrics import confusion_matrix
  cm = confusion_matrix(Y_test,Y_pred) cm
```

**OUTPUT:**

```
array([[2580, 1398],
      [1526, 2451]]
```

```
import seaborn as sn
        plt.figure(figsize = (10,7))
        sn.heatmap(cm, annot=True)
        plt.xlabel('Predicted')
        plt.ylabel('Truth')
```

**OUTPUT:**



# 5.RESULT

| MACHINE LEARNING ALGORITHMS | ACCURACY BEFORE BALANCING THE DATA SET | ACCURACY AFTER BALANCING THE DATA SET |
|---|---|---|
| LOGISTIC REGRESSION | **Accuracy on the training data :** 0.66592<br>**Accuracy on the test data :** 0.66048 | **Accuracy on the training data :** 0.6126639567531325<br>**Accuracy on the testdata :** 0.6144563167818982 |
| K-NEAREST NEIGHBOR | **Accuracy on training data:** 0.64384<br>**Accuracy on the test data:** 0.66048 | **Accuracy on training data:** 0.6100565681961031<br>**Accuracy on the test data:** 0.6144563167818982 |
| SUPPORT VECTOR MACHINE | ______ | **Accuracy on training data:** 0.6408548082966687<br>**Accuracy on the test data:** 0.8331307674994373 |
| DECISION TREE | ______ | **Accuracy on training data:** 0.61456316781898<br>**Accuracy on the test data:** 0.6203645505971087 |

# 6. CONCLUSION

However, in general, machine learning algorithms can be used to analyze and model large datasets to identify potential patterns and risk factors that may be associated with the occurrence and spread of diseases like monkeypox. These algorithms can help in the development of predictive models that can assist public health authorities in preparing for outbreaks and implementing appropriate response measures.

However, it's important to note that the accuracy of such models depends on the quality and quantity of the data used to train them, as well as the complexity of the underlying biological and environmental factors that contribute to the spread of the disease. Therefore, while machine learning can provide valuable insights and predictions, it should always be used in conjunction with other methods and expert knowledge in the field of public health.

# 7.FUTURE SCOPE

In future, new features from the fields can be gathered to get a perfect image of the crop damage using other machine learning algorithms and deep learning algorithms such as ANN or CNN to get more accurate predictions.

There is significant potential for the use of machine learning in predicting and mitigating the spread of monkeypox. Here are some potential future areas of development:

1. Early Detection: Machine learning algorithms can be trained to detect early signals of an outbreak based on patterns in data such as animal surveillance, social media, and news articles. This early detection can allow public health officials to act quickly and contain the spread of the disease.
2. Risk Factors Analysis: By analyzing historical data and environmental factors, machine learning algorithms can identify risk factors that contribute to the outbreak of monkeypox. This information can help public health officials to identify high-risk areas and populations and implement targeted interventions.
3. Transmission Modeling: Machine learning algorithms can be used to model the transmission of the disease and predict the potential impact of different intervention strategies. This information can help public health officials to develop effective response plans and mitigate the spread of the disease.

Overall, the future of monkeypox prediction using machine learning is promising, and it has the potential to significantly improve public health outcomes. However, continued research and collaboration between public health officials, researchers, and machine learning experts will be necessary to fully realize this potential.

.

## 8. REFERENCES

[1] https://ieeexplore.ieee.org/abstract/document/9221459

[2] https://ieeexplore.ieee.org/abstract/document/9418375

[3] https://dl.acm.org/doi/abs/10.1145/3372454.3372474

[4] https://ieeexplore.ieee.org/abstract/document/9311735

[5] https://iopscience.iop.org/article/10.1088/17426596/1767/1/012026/meta

[6] https://ieeexplore.ieee.org/abstract/document/9221459

[7] https://ieeexplore.ieee.org/abstract/document/9033611

[8] https://ieeexplore.ieee.org/abstract/document/9137868

[9] https://ieeexplore.ieee.org/abstract/document/9154036