**Documentation of Agricultural-production-optimization**

The model can achieve the optimum production plan of an agricultural region combining in one utility function different conflicting criteria .

Machine learning is used to analyze data

These data set suggesting crops according to the given data set of soil and using the predicition of weather In given dataset we have different value of Nitrogen ,phosphorous,potassium,temperature,humidity,ph,rainfall

Machine learning process is used for prediction of best soil,weather and climatic condition and get the best result for our crops grow

Example :

Nitrogen=90,phosphorous=40,potassium=40,temperature=20,humidity=80,ph =7,rainfall=200

input=> prediction = model.predict((np.array([[90,40,40,20,80,7,200]])))

print("the suggested crop for given climatic condition id :", prediction)

output=> the suggested crop for given climatic condition id : ['rice'] => prediction of best soil , weather and climatic condition

Result => Through all these process we are getting the best prediction to grow the crops in depending soil ,weather and climatic condition

**# data manipulate**

1>**import NumPy as np**

**What is the use of NumPy in data analysis?**

NumPy is a commonly used Python data analysis package. By using NumPy, you can speed up your workflow, and interface with other packages in the Python ecosystem, like scikit-learn, that use NumPy under the hood. NumPy was originally developed in the mid 2000s, and arose from an even older package called Numeric.

Numpy library is used for solving high computation mathematical problems like machine learning algorithms etc

==><https://www.dataquest.io/blog/numpy-tutorial-python/>

2>**import pandas as pd**

Pandas is an open-source python library that is used for data manipulation and analysis. It provides many functions and methods to speed up the data analysis process. Pandas is built on top of the NumPy package, hence it takes a lot of basic inspiration from it. The two primary data structures are Series which is 1 dimensional and DataFrame which is 2 dimensional.

It is one of the most important and useful tools in the arsenal of a Data Scientist and a Data Analyst.

pandas (all lowercase) is a popular Python-based data analysis toolkit which can be imported using import pandas as pd . It presents a diverse range of utilities, ranging from parsing multiple file formats to converting an entire data table into a NumPy matrix array.

==><https://www.analyticsvidhya.com/blog/2021/03/pandas-functions-for-data-analysis-and-manipulation/>

# **data visualization**

1>**import matplotlib.pyplot as plt**

matplotlib. pyplot is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. In matplotlib. (like graph)

1. **import seaborn as sas**

Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data.

1. i**mport klib #==> new model for visualization**

# **klib.describe - functions for visualizing datasets**

* klib.cat\_plot(df) # returns a visualization of the number and frequency of categorical features
* klib.corr\_mat(df) # returns a color-encoded correlation matrix
* klib.corr\_plot(df) # returns a color-encoded heatmap, ideal for correlations
* klib.dist\_plot(df) # returns a distribution plot for every numeric feature
* klib.missingval\_plot(df) # returns a figure containing information about missing values
* <https://pypi.org/project/klib/>

# **for interactivity (ui)**

1>**from ipywidgets import interact**

The interact function ( ipywidgets. interact ) automatically creates user interface (UI) controls for exploring code and data interactively. It is the easiest way to get started using IPython's widgets.

1. **what is null data and used** ?

Pandas isnull() and notnull()

While making a Data Frame from a csv file, many blank columns are imported as null value into the Data Frame which later creates problems while operating that data frame. Pandas isnull() and notnull() methods are used to check and manage NULL values in a data frame.

Dataframe.isnull()

Syntax: Pandas.isnull(“DataFrame Name”) or DataFrame.isnull()

Parameters: Object to check null values for

Return Type: Dataframe of Boolean values which are True for NaN values

Mean - in the data set in have numerical value(1,2,5,9) then replace the missing value helping through mean

**Median -**

The mean value of numerical data is without a doubt the most commonly used statistical measure. mode

Categorical variables represent types of data which may be divided into groups. Examples of categorical variables are race, sex, age group, and

**educational link** => <http://www.stat.yale.edu/Courses/1997-98/101/catdat.htm>

In statistics, the mode is the most commonly observed value in a set of data. For the normal distribution, the mode is also the same value as the mean and median. In many cases, the modal value will differ from the average value in the data

**Distribution**

Term “distribution ” in data science or statistics usually means a probability distribution . Distribution is nothing but a function which provides the possible value of variables and how often they occur. Probability distribution is a mathematical function which provides the possibilities of occurrence of various possible outcomes that can occur in an experiment.

Checking the distribution we can know the pattern of all the column or  field and not only this rather checking the distribution we can know about the outliner and anomalies of the particular column or filed

Q=>>**What is clustering analysis ? All information**

Clustering is an unsupervised machine learning method of identifying and grouping similar data points in larger datasets without concern for the specific outcome. Clustering (sometimes called cluster analysis) is usually used to classify data into structures that are more easily understood and manipulated

Q⇒ **k mean clustering analysis?**

**k mean clustering analysis**

K-means clustering is one of the simplest and popular unsupervised machine learning algorithms.

**How the K-means algorithm works**

To process the learning data, the K-means algorithm in data mining starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative (repetitive) calculations to optimize the positions of the centroids

Q⇒ f**rom sklearn.cluster import KMeans ?**

K-means algorithm example problem

Here is the entire K-means clustering algorithm code in Python:

Step 1: Import libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans

%matplotlib inline

Step 2: Generate random data

Here is the code for generating some random data in a two-dimensional space:

X= -2 \* np.random.rand(100,2)

X1 = 1 + 2 \* np.random.rand(50,2)

X[50:100, :] = X1

plt.scatter(X[ : , 0], X[ :, 1], s = 50, c = ‘b’)

plt.show()

Step 3: Use Scikit-Learn

We’ll use some of the available functions in the Scikit-learn library to process the randomly generated data.

Here is the code:

from sklearn.cluster import KMeans

Kmean = KMeans(n\_clusters=2)

Kmean.fit(X)

Step 4: Finding the centroid

Here is the code for finding the center of the clusters:

Kmean.cluster\_centers\_

Let’s display the cluster centroids (using green and red color).

plt.scatter(X[ : , 0], X[ : , 1], s =50, c=’b’)

plt.scatter(-0.94665068, -0.97138368, s=200, c=’g’, marker=’s’)

plt.scatter(2.01559419, 2.02597093, s=200, c=’r’, marker=’s’)

plt.show()

Step 5: Testing the algorithm

Here is the code for getting the labels property of the K-means clustering example dataset; that is, how the data points are categorized into the two clusters.

Kmean.labels\_

For example, let’s use the code below for predicting the cluster of a data point:

sample\_test=np.array([-3.0,-3.0])

second\_test=sample\_test.reshape(1, -1)

Kmean.predict(second\_test)

Q⇒ **elbow method**?

In cluster analysis, the elbow method is a heuristic used in determining the number of clusters in a data set. The method consists of plotting the explained variation as a function of the number of clusters, and picking the elbow of the curve as the number of clusters to use. The same method can be used to choose the number of parameters in other data-driven models, such as the number of principal components to describe a data set.

Using the "elbow" or "knee of a curve" as a cutoff point is a common heuristic in mathematical optimization to choose a point where diminishing returns are no longer worth the additional cost. In clustering, this means one should choose a number of clusters so that adding another cluster doesn't give much better modeling of the data.

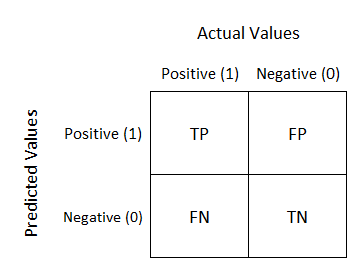
Q⇒ **rcParams?**

Each time Matplotlib loads, it defines a runtime configuration (rc) containing the default styles for every plot element you create. This configuration can be adjusted at any time using the plt. ... matplotlibrc file, which you can read about in the Matplotlib documentation.

Q⇒ **confusion\_matrix?**

**What is Confusion Matrix and why you need it?**

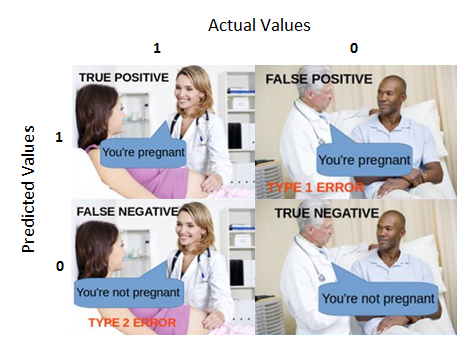
Well, it is a performance measurement for machine learning classification problem where output can be two or more classes. It is a table with 4 different combinations of predicted and actual values.



**Confusion Matrix**

It is extremely useful for measuring Recall, Precision, Specificity, Accuracy, and most importantly AUC-ROC curves.

Let’s understand TP, FP, FN, TN in terms of pregnancy analogy.



Confusion Matrix True Positive:

Interpretation: You predicted positive and it’s true.

You predicted that a woman is pregnant and she actually is.

True Negative:

Interpretation: You predicted negative and it’s true.

You predicted that a man is not pregnant and he actually is not.

False Positive: (Type 1 Error)

Interpretation: You predicted positive and it’s false.

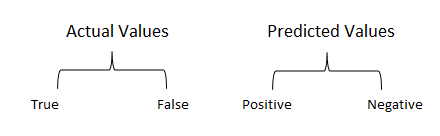
You predicted that a man is pregnant but he actually is not.

False Negative: (Type 2 Error)

Interpretation: You predicted negative and it’s false.

You predicted that a woman is not pregnant but she actually is.

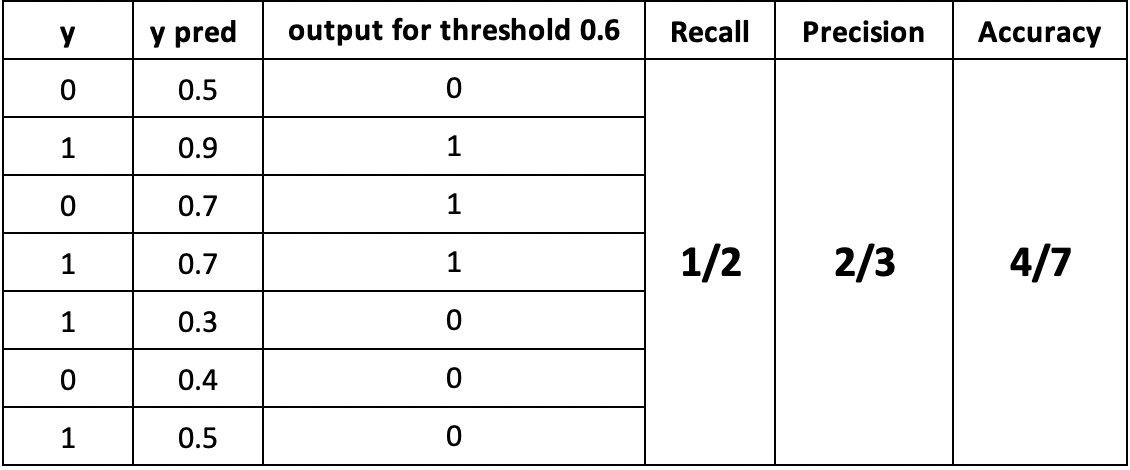
Just Remember, We describe predicted values as Positive and Negative and actual values as True and False.

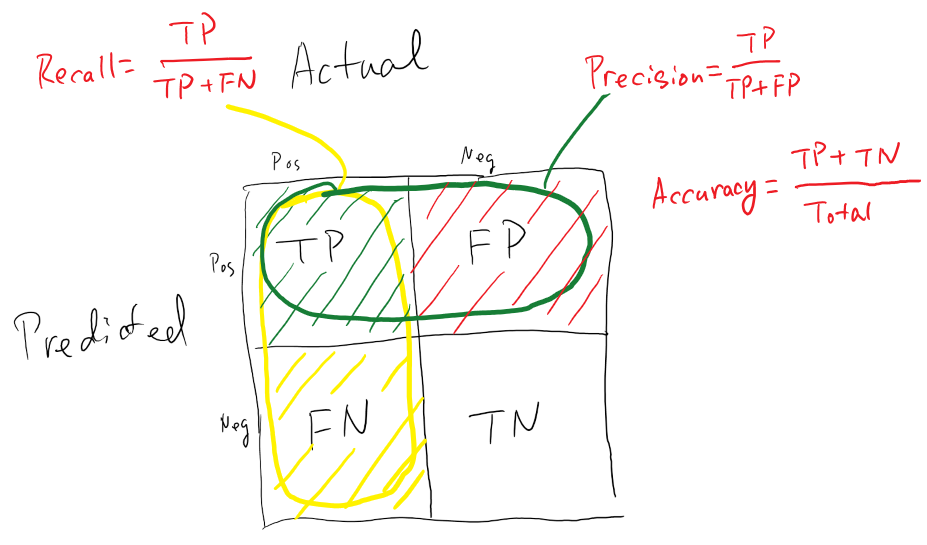


**Actual vs Predicted values**

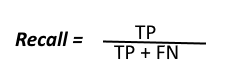
How to Calculate Confusion Matrix for a 2-class classification problem?

Let’s understand the confusion matrix through math.



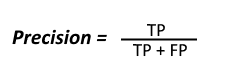


Confusion Matrix Recall



Recall

The above equation can be explained by saying, from all the positive classes, how many we predicted correctly. Recall should be high as possible.



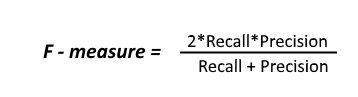
Precision

The above equation can be explained by saying, from all the classes we have predicted as positive, how many are actually positive.

Precision should be high as possible.And Accuracy From all the classes (positive and negative), how many of them we have predicted correctly. In this case, it will be 4/7.

Accuracy should be high as possible.

F-measure



F1 Score

It is difficult to compare two models with low precision and high recall or vice versa. So to make them comparable, we use F-Score. F-score helps to measure Recall and Precision at the same time. It uses Harmonic Mean in place of Arithmetic Mean by punishing the extreme values more.