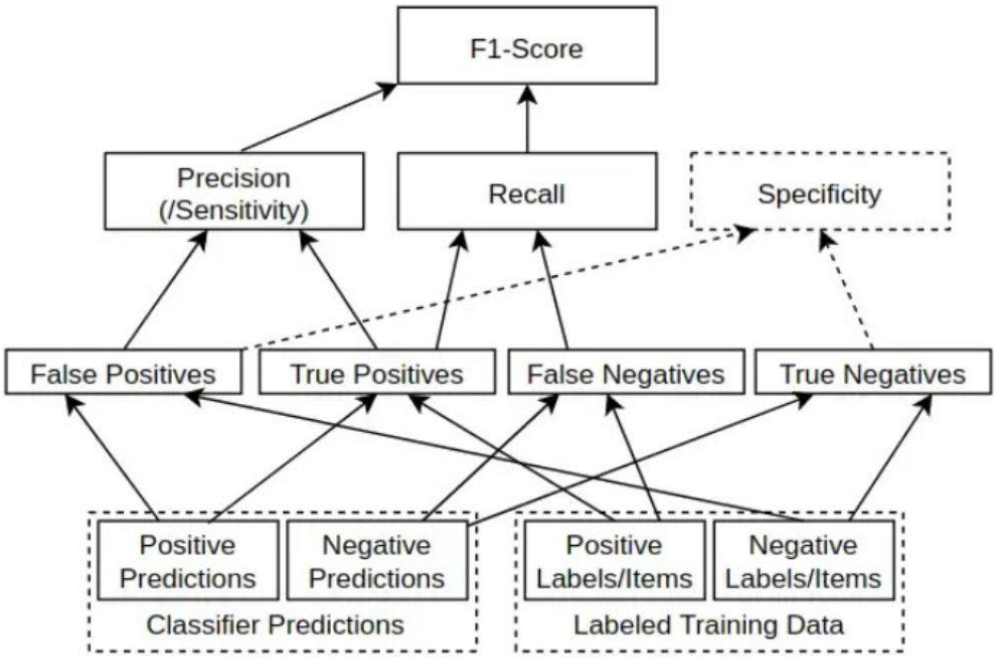
Week 4 Homework 1: KNN + Confusion Matrix + Iris Data set + Colab

Introduction:

In the field of Machine Learning, the performance of a model is crucial. To assess the performance of a model, there are several metrics that are used. One of these important metrics is the Confusion Matrix, which is used to evaluate the accuracy of a model. The Confusion Matrix is a matrix with size n x n, where n represents the number of class labels in a given problem. The purpose of the Confusion Matrix is to provide a clear picture of the model's performance in a tabular format.



A Look at Precision, Recall, and F1-Score

	Predicted=ill	Predicted=well
Actual=ill	True-Positive (TP) ◦ Correctly diagnosing an ill patient as ill.	False-Negative (FN) ◦ Incorrectly diagnosing an ill patient as well. ◦ False Negatives may be way worse than False Positives.
Actual=well	False-Positive (FP) ◦ Incorrectly diagnosing a well patient as ill. ◦ Commonly called a "false alarm"	True-Negative (TN) ◦ Correctly diagnosing an well patient as well.

Implementation on Google Colab:
https://colab.research.google.com/drive/1eshNs7h3B7Hd9EQcACsN0c3CiiX5Sh3u?usp=sharing

```
import pandas as pd
import numpy as np
import sklearn
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn import metrics
from sklearn.metrics import *
from sklearn.model_selection import *
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
```

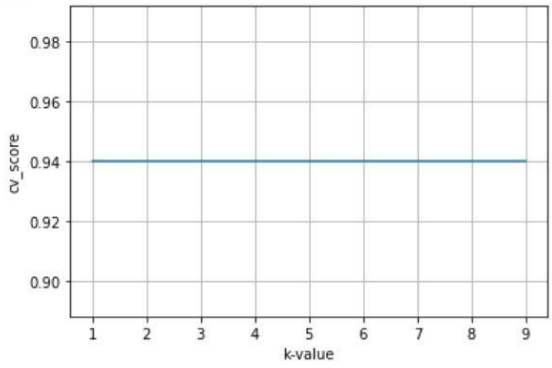
```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files iris_data.csv
• iris_data.csv(text/csv) - 4811 bytes, last modified: 2/14/2023 - 100% done
Saving iris_data.csv to iris_data (1).csv
```

```
[8] iris = pd.read_csv('iris_data.csv')
    iris.shape
    col_list = iris.columns
    print(type(col_list))
    print(col_list[:])
    iris['Species'].value_counts()
    iris_data = iris.iloc[:,1:5] # select all the rows and col indices 1 to 4
    iris_lables = iris.iloc[:,5:] # select all the rows and 5th cloumn
    iris_data.shape
    iris_data.head(2)
```

Output:

```
5
[0.9400000000000001, 0.9400000000000001, 0.9400000000000001, 0.9400000000000001, 0.9400000000000001]
(1, 0.9400000000000001)
(3, 0.9400000000000001)
(5, 0.9400000000000001)
(7, 0.9400000000000001)
(9, 0.9400000000000001)
```




```
0.98
0.96
```

```
[16] #confusion matrix and classification_report
     #precision = TP/TP+FP
     #Recall = TP/TP+FN

     print(metrics.confusion_matrix(y_test, cross_predict))
     print(metrics.classification_report(y_test, cross_predict))
```

```
[[19  0  0]
 [ 0 15  0]
 [ 0  2 14]]
              precision    recall  f1-score   support

   I. setosa       1.00      1.00      1.00        19
I. versicolor       0.88      1.00      0.94        15
 I. virginica       1.00      0.88      0.93        16

    accuracy                           0.96        50
   macro avg       0.96      0.96      0.96        50
weighted avg       0.96      0.96      0.96        50
```