KNN + Confusion Matrix + Iris Data set + Colab

Submitted by Manish Bafna
Student Id: 19655
Instructor: Dr. Henry Chang
GIT: **Training Linear Model**

Table of Content

Introduction

In Machine Learning world model performance is everything. There are several metrics using which we will be able to evaluate our model.

Confusion matrix is one such important tool which helps us evaluate our model's performance. As the name suggests it is a matrix of size n x n .where 'n' is the number of class labels in our problem.

Introduction

Let’s take a look at confusion matrix structure. Here,I am showing the python standard matrix notation for two class classification.

		Predicted values		
		Positive	**Negative**	**Totals**
Actual Values	**Positive**	TP	FN	**P = (TP + FN) = Actual Total Positives**
	Negative	FP	TN	**N = (FP + TN) = Actual Total Negatives**
	Totals	Predicted Total Positives	Predicted Total Negatives	

Introduction

Let’s understand each of the cells :

True-Positive (TP)
Actual positives in the data, which have been correctly predicted as positive by our model. Hence True Positive.

False-Negative (FN)
Actual Positives in data, but our model has predicted them as Negative. Hence False Negative.

False-Positive (FP)
Actual Negatives in data, but our model has predicted them as Positive. Hence False Positive.

True-Negative (TN)
Actual Negatives in the data, which have been correctly predicted as negative by our model. Hence True negative.

Introduction

Classification Model Evaluation Metric: Confusion Matrix

Low Accuracy
High Precision

High Accuracy
Low Precision

High Accuracy
High Precision

Suppose you're working as a security guard at a concert and your job is to

- identify any individuals who were troublemakers at previous events (represented as "positive" class in this problem).

Recall	F1 Score

Recall measures the proportion of true positive instances (troublemakers) that were correctly identified by the classifier.
 	
• The fraction of actual troublemakers that were correctly identified by you as troublemakers and allowed into the concert.
	F1 Score is a balance between
	
	• Precision and
	  ◦ Precision is the fraction of instances classified as positive that are actually positive,
	• Recall
	  ◦ Recall is the fraction of positive instances that were correctly classified as positive.

A high recall score means that you are able to catch most of the troublemakers, but you may also let in some non-troublemakers (false positives).	In this analogy, F1 Score would represent the balance between catching all the troublemakers (high recall) and only letting in the non-troublemakers (high precision). ◦ A high F1 Score means that you have found a good balance between these two metrics.

Introduction

In ML, We have so many metrics. Out of which most known and used one is ***Accuracy.***

Accuracy: Accuracy tells the percentage of correctly predicted values out of all the data points.

$$Accuracy = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + FN + TN + FP}$$

TPR (True Positive Rate) or Recall: It tells us, out of all the ***positive*** data points, how many have been truly identified as positive by our model.

$$TPR = \frac{TP}{TP + FN}$$

Introduction

FNR (False negative Rate): It tells us, out of all the positive points, how many have been falsely identified as negative by our model.

$$FNR = \frac{FN}{TP + FN}$$

Precision: It tells use, out of all the points which have been identified as positive by our model, how many are actually true.

$$Precision = \frac{TP}{TP + FP}$$

Introduction

TNR (True Negative Rate): It tells us, out of all the negative points in our data set, how many have been truly identified as negative by our model.

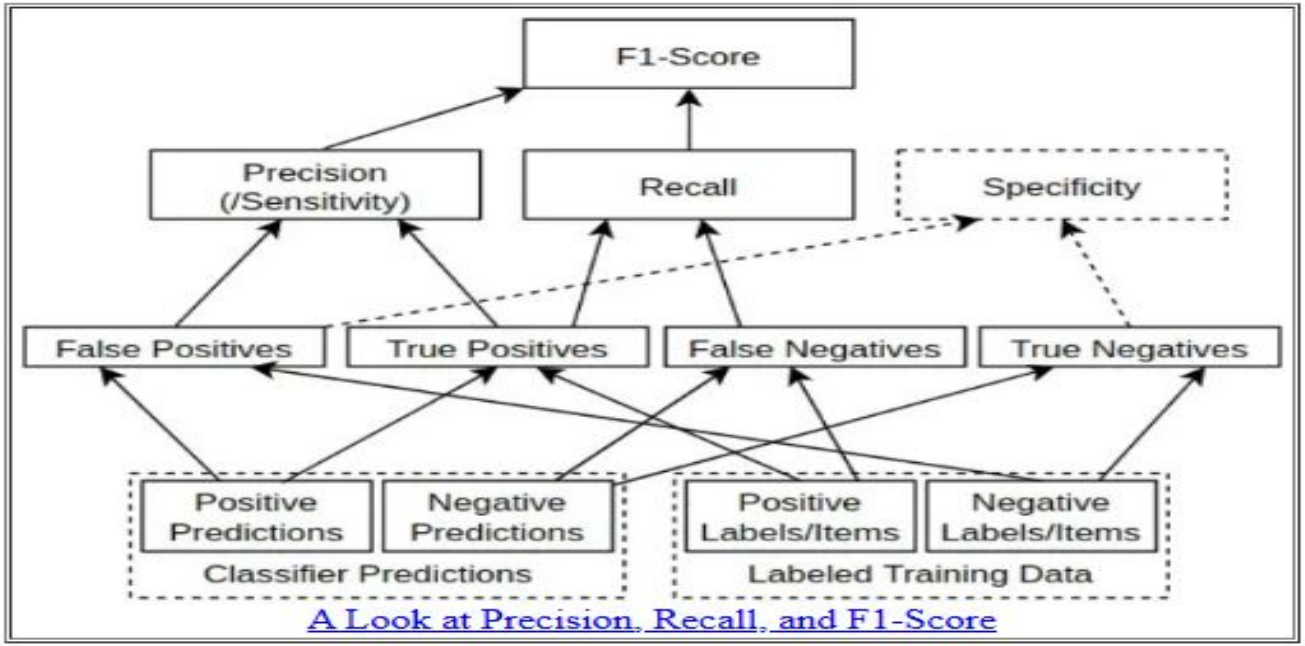
$$TNR = \frac{TN}{FP + TN}$$

FPR (False Positive Rate): It tells us, out of all the Negative points, how many have been falsely identified as positive by our model.

$$FPR = \frac{FP}{FP + TN}$$

Introduction

Definition



Implementation

Now, we understood how to read confusion matrix and how to derive all the above mentioned formulas from it.But, we looked at only two class classification. What if we have more than one class? How to read that confusion matrix. For this, let’s take Iris — Dataset

I have applied KNN- classifier on this data set and want to see confusion matrix of the test data set.

Iris data set has 150 data points in total and three labels each label has 50 points.

Out of which I trained the model using 100 data points and tested using 50 data points.

Implementation

- Google Colab: https://colab.research.google.com/

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

```
Choose Files  iris.csv
  • iris.csv(text/csv) - 4617 bytes, last modified: 2/11/2023 - 100% done
Saving iris.csv to iris (8).csv
```

```
[ ] # load iris dataset
    iris = pd.read_csv('iris (8).csv')
```

Iris Data Set: Machine-Learning/Supervised Learning/KNN + Confusion Matrix + Iris Data set + Colab at main · manishbafna22/Machine-Learning (github.com)

Execute iris_Data_knn.ipynb: Machine-Learning/Supervised Learning/KNN + Confusion Matrix + Iris Data set + Colab at main · manishbafna22/Machine-Learning (github.com)

Test

OutPut

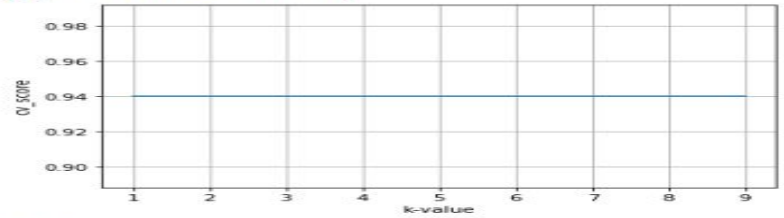
```
# based on above observations we are getting maximum accuracy when k=7,
#So we will use K-value 7 and predict on test datsset and see accuracy.

neigh_K7 = KNeighborsClassifier(n_neighbors=7)
neigh_K7.fit(x_train, y_train)
predict_array_k7 = neigh_K7.predict(x_test)
print(metrics.accuracy_score(y_test, predict_array_k7))
predict_probability = neigh_K7.predict_proba(x_test)

#zipped_pobability = zip(predict_array_k7,predict_probability)
#for i in zipped_pobability:
#    print(i)

cross_predict = cross_val_predict(cross_neigh,x_test,y_test,cv=10)
print(metrics.accuracy_score(y_test, cross_predict))
```

```
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
```

Test

Confusion Matrix for Iris Data set

```
#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

    Iris-setosa       1.00      1.00      1.00        19
Iris-versicolor       0.88      1.00      0.94        15
 Iris-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
```

Enhancement

Hyperparameter tuning: You can consider performing a more extensive hyperparameter tuning process to find the optimal hyperparameters for the KNN algorithm.

Using other algorithms: You can consider evaluating other algorithms, such as decision trees, random forests, support vector machines (SVM), and artificial neural networks (ANN), to compare their performance with KNN on this dataset.

Conclusion

we can conclude that a K-Nearest Neighbor (KNN) classification model was trained and evaluated on the iris dataset.

The iris dataset was read from a csv file, and the columns and values of the species column were analyzed.The iris data was standardized using the StandardScaler from scikit-learn's pre-processing library, and then split into training and testing datasets.

The KNN model was trained on the training dataset and used to predict the species of iris plants in the test dataset. The accuracy of the prediction was evaluated using the accuracy_score metric from scikit-learn's metrics library.

A 10-fold cross validation was also performed, and the results were plotted to determine the optimal number of neighbors (K) for the KNN model.Based on the plot, K=7 was determined to give the maximum accuracy, and the KNN model was retrained with K=7 and used to predict the species of iris plants in the test dataset, resulting in an accuracy score.

References

Understanding Confusion matrix and applying it on KNN-Classifier on Iris Data set. | by Vishwanath Beena | Artificial Intelligence in Plain English

Pick an evaluation metric (sfbu.edu)

How to Evaluate and Improve Knn Classifier Part 3 | by Jalal Mansoori | Medium