Abstract

It is now evident that the world needs a speedy and quicker solution to contain and tackle the further spread of COVID-19 across the world with the aid of non-clinical approaches such as data mining approaches, augmented intelligence and other artificial intelligence techniques so as to mitigate the huge burden on the healthcare system and reduce the spread of the virus, it was necessary to provide the best possible means to effectively diagnose patients.

Table of Contents

[Cover 1](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967467)

[Abstract 2](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967468)

[Table of contents 3](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967469)

[Chapter 1: Introduction 4](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967471)

[Chapter 2: Coding 4](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967472)

[2.1 Importing libraries 4](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967474)

[2.2 Loading data and One Hot encoding 5](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967475)

[2.3 Splitting data into training,testing and validation 5](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967477)

[2.4 Support Vector Machine 6](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967478)

[2.5 K-NN Classifier 7](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967474)

[2.6 Decision Tree 9](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967475)

[2.7 Logistic Regression 10](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967477)

[2.8 Naïve Bayes 13](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967478)

[2.9 Differences between the 5 classifiers 16](file:///C:\Users\mylap\Desktop\ml%20report.doc#_Toc367967478)

Chapter 1: Introduction

Severe Acute Respiratory Syndrome Coronavirus two (SARS-CoV-2), the causative agent of novel coronavirus (COVID-19 or 2019-nCoV), has emerged in late 2019 which is believed to be originated from Hubei Province, China called Wuhan. 2019nCoV or COVID-19 is rapidly spreading in humans.

The major symptoms of SARS-CoV-2 include fever, cough, and shortness of breath which in many instances appeared to be similar to that flu.

Machine learning offers a principled approach for developing sophisticated, automatic, and objective algorithms for analysis of high-dimensional and multimodal biomedical data. This review focuses on several advances in the state of the art that have shown promise in improving detection, diagnosis, and therapeutic monitoring of disease.

Chapter 2: Coding

2.1 Importing libraries

The following libraries we used in our code:

* Pandas: the best Python library to be used for data manipulation, it provides data reading and writing capabilities using various sources such as Excel, HDFS, and others.
* NumPy: It simplifies dealing with multidimensional data and improves performance of Machine Learning models
* Matplotlib: Is a Plotting Library In Python Which Is Widely used to Create Static, Animated And Interactive Visualizations
* Seaborn: is a plotting library that offers a simpler interface, sensible defaults for plots needed for machine learning
* Scikit-learn: it has a package which consists of all the methods for implementing the standard algorithms of ML, It also has a simple and consistent interface that helps fit and transform the models over any dataset and scikit-learn is also considered as the best library for the reliable deployment of Machine Learning models.
* Math: we used it to import the pi constant

2.2 Loading Data and One Hot Encoding

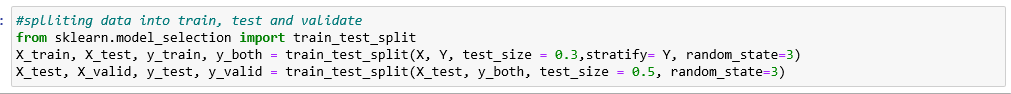
We read the “data.csv” file by using the [pandas.read\_csv()](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html) function. It read the CSV file and creates the DataFrame, then we Splitted the data into features and target variables, the first 13 column were stored in the X variable (features) and the last column was stored in the Y variable (target).

Then we applied the one hot encoding in order to convert categorical data variables into numerical values.



2.3 splitting data into training, testing and validation

**We used the train\_test\_split** function in Sklearn model selection for splitting the data set into two subsets: We set 70% from the data set for training, then we used the **train\_test\_split** function again in order to split the rest 30% into : 15% for validation and 15% for testing.



2.4 Support vector machine

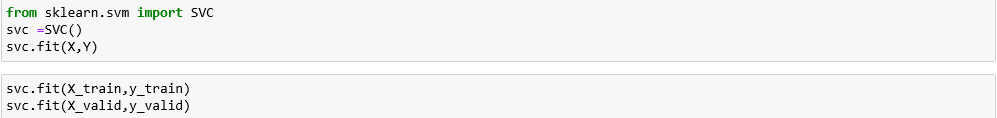
SVMs are [supervised learning](https://en.wikipedia.org/wiki/Supervised_learning) models with associated learning [algorithms](https://en.wikipedia.org/wiki/Algorithm) that analyze data for [classification](https://en.wikipedia.org/wiki/Statistical_classification) and [regression analysis](https://en.wikipedia.org/wiki/Regression_analysis). In addition to performing [linear classification](https://en.wikipedia.org/wiki/Linear_classifier), SVMs can efficiently perform a non-linear classification using what is called the [kernel trick](https://en.wikipedia.org/wiki/Kernel_method#Mathematics:_the_kernel_trick), implicitly mapping their inputs into high-dimensional feature spaces.

**Support-vector clustering (SVC)**

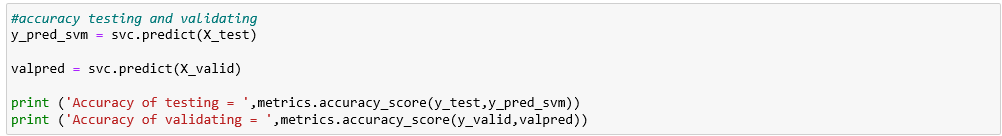
SVC is a similar method that also builds on kernel functions but is appropriate for unsupervised learning

Our analysis on COVID-19 dataset depicts that among all the other supervised models, SVM works best in predicting COVID-19 cases with maximum accuracy.

we fitted the X and Y in training and validating



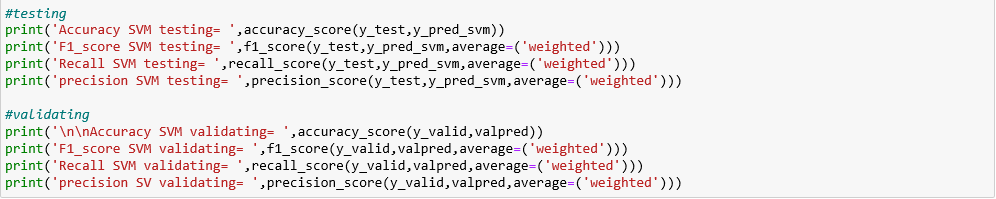
Then, we used y\_pred\_svm to predict X\_test and valpred to predict X\_valid



About that c float, its default is 1.0 and it is called the “regularization parameter”. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty, To fit svc we used linear kernel, to fit rbf\_svc we used radial basis function “rbf” kernel and to fit polynomial svc “poly\_svc” we used polynomial kernel



After that, we printed the performance “ the f1\_score, precision, recall and accuracy” for both testing and validating



Then we finally plotted the ROC and AUC Curve for SVM



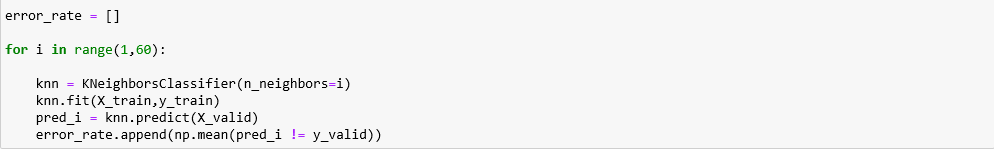
2.5 K-NN Classifier

The KNN algorithm is a machine learning algorithm based on the supervised learning model. The KNN algorithm works by assuming that similar things exist close to each other. Hence, the KNN algorithm utilizes feature similarity between the new data points and the points in the training set to predict the values of the new data points.

We started by validation with a random N value, we chose the N value to be 5 and we fitted the model



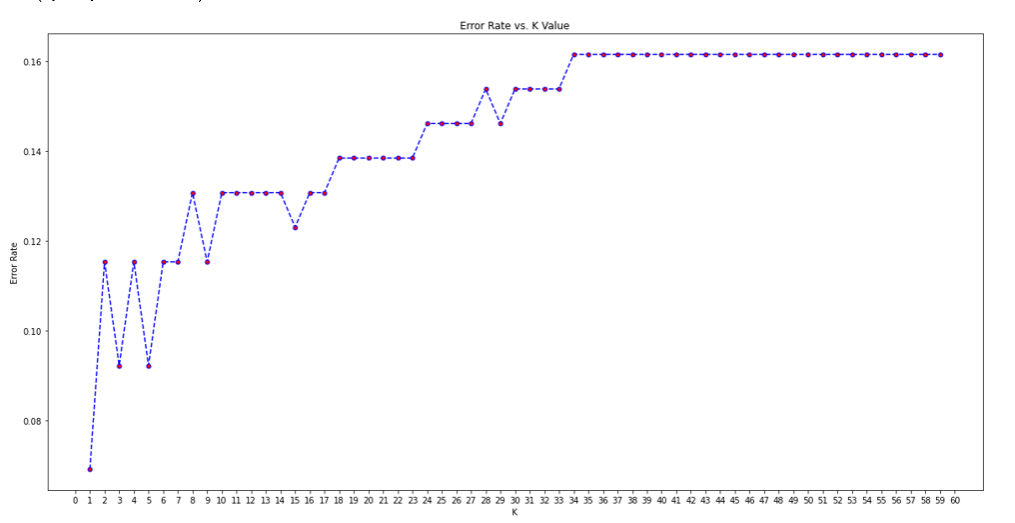
Then, we used the elbow method to pick a good K Value. We checked the error rate for k=1 to k=60. For every value of k, we called the KNN classifier in order to choose the value of k which has the least error rate.



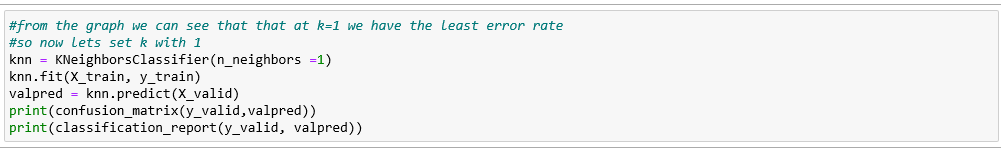
We needed to plot a Line graph of the error rate to find the corresponding k value to the least error rate, we did it using the plt.plot()



We got the following graph, which shows that the least error rate value is at k=1



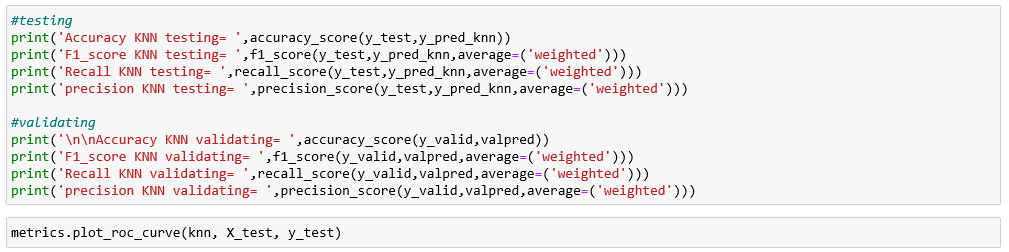
So we retrained our model but this time using k=1, and it gave us a better accuracy



Now, the testing step using knn.predict()

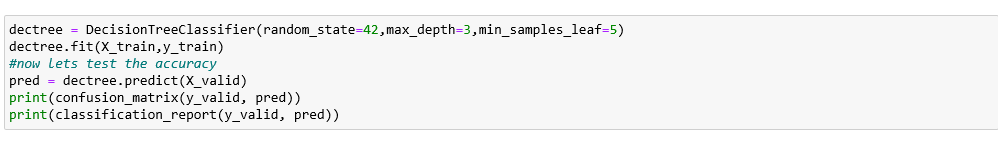


And finally, we printed the performance “the f1\_score, precision, recall and accuracy” for both testing and validating , Then we finally plotted the ROC and AUC Curve for KNN

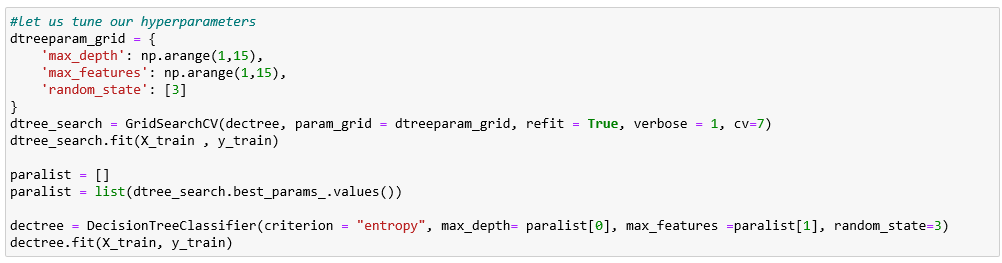


2.6 Decision Tree

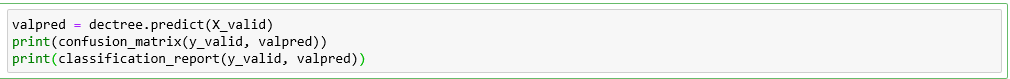
We started by creating the decision tree and fitting it with X\_train and y\_train using DecisionTreeClassifier () from **sklearn** Library, we will perform the validation using predict() and then we will print the confusion matrix and the classification report



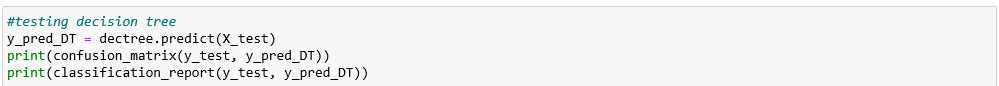
The next step was to tune our hyperparameters using the grid-search-cross-validation in order to extract best parameters with our Decision Tree classifier



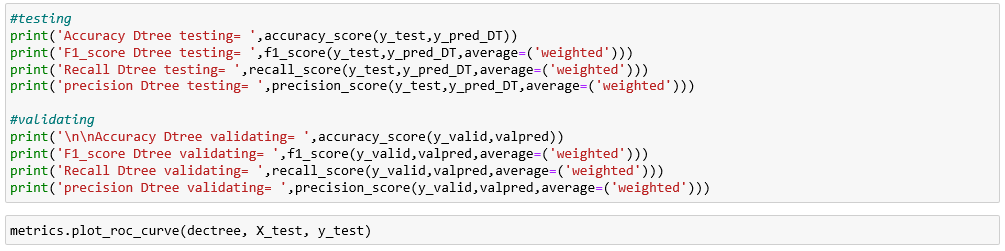
After extracting the best parameters, We will retrain and perform validation again after tuning using predict() and we will print the confusion matrix and the classification report again



Now, the testing step with printing the confusion matrix and the classification report



And finally, we printed the performance “the f1\_score, precision, recall and accuracy” for both testing and validating, Then we finally plotted the ROC and AUC Curve for the decision tree

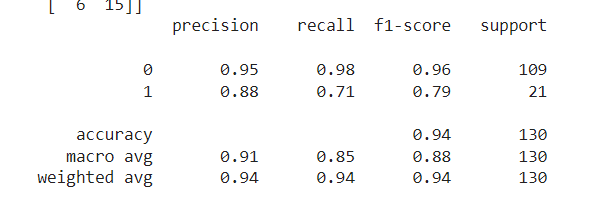


2.7 Logistic Regression

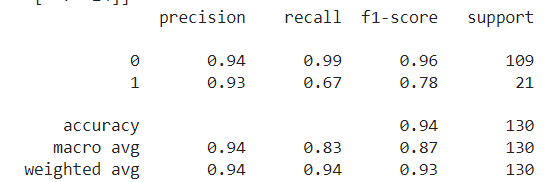
Logistic Regressionis a classification technique used in machine learning. It uses a logistic function to model the dependent variable. The dependent variable is dichotomous in nature, i.e. there could only be two possible classes. As a result, this technique is used while dealing with binary data.

To implement this classifier, we used **sklearn.linear\_model**

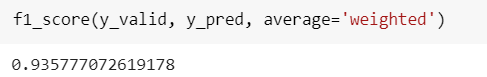
After Splitting data into Training , testing and Validation we validate before tuning the hyperparameters we found that results are :



After tuning, we performed Validation again and the results were :

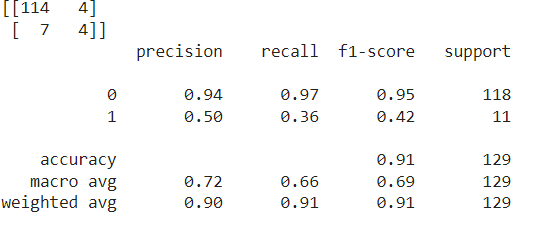


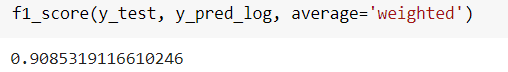
F1 score) which is a measure of a model’s accuracy on a dataset( after tuning was



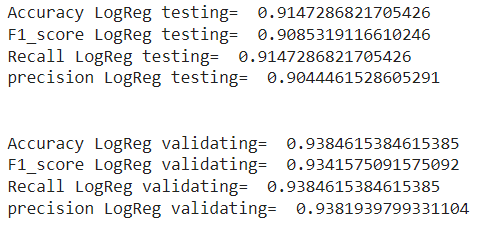
**Testing logistic regression algorithm**

confusion matrix & classification report for (y\_test, y\_pred):

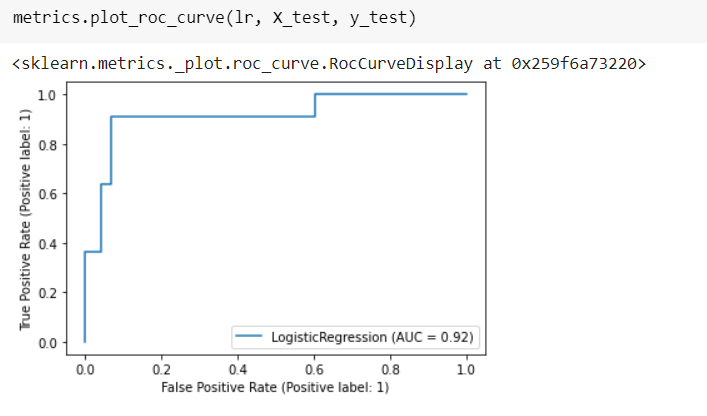


F1 score after testing Logistic regression : 

Model accuracy , F1\_score,Recall ,precision:



ROC for ((lr, X\_test, y\_test))

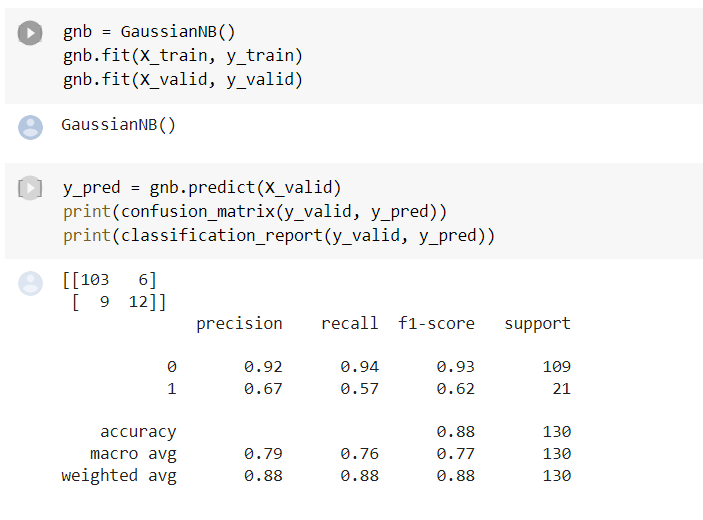


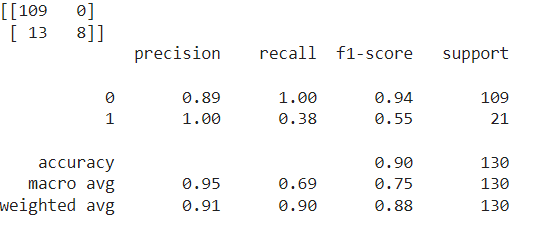
2.8 Naïve Bayes

a classification technique based on Bayes’ Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

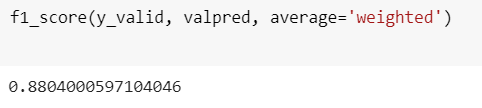
To implement this classifier we used **sklearn.naive\_bayes**

we validated before tuning the hyperparameters we found that results are :



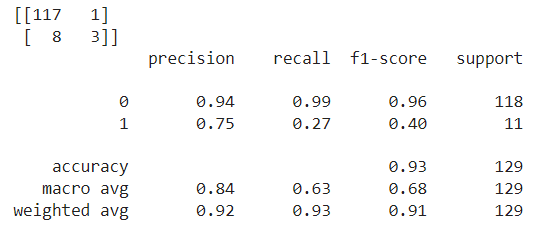
We did also Validation after tuning and the results were : 

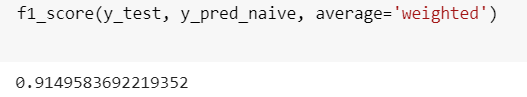
F1 score after tuning was :



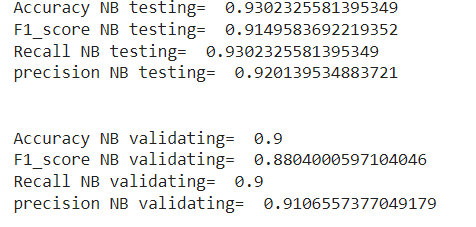
Testing Naive Bayes algorithm :

confusion\_matrix & classification\_report for (y\_test, y\_pred):

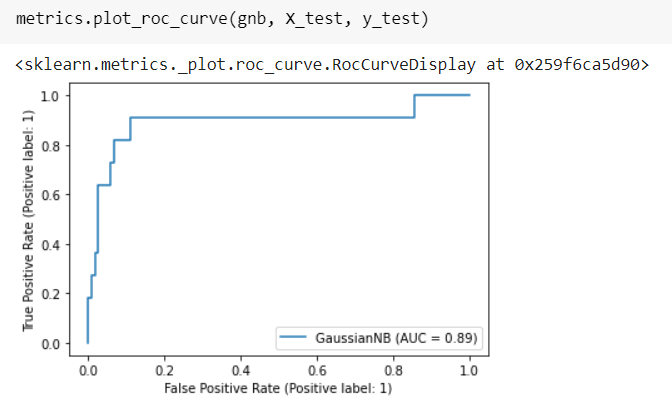


F1 score after testing:  


Model accuracy , F1\_score,Recall ,precision:

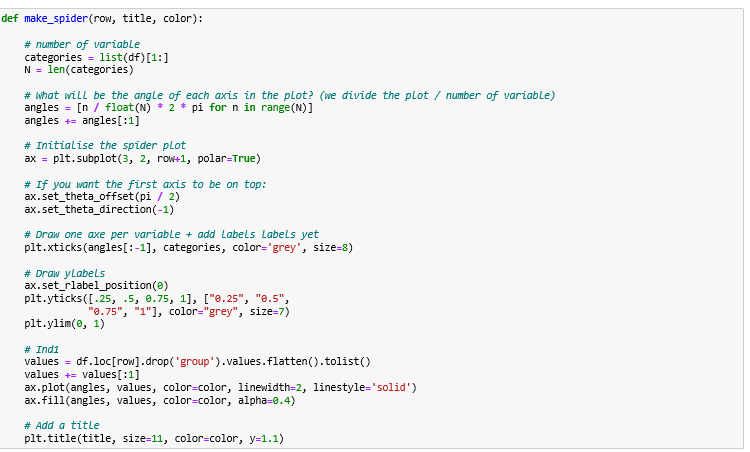


ROC for ((lr, X\_test, y\_test)):



2.9 Differences between the 5 classifiers

We used make\_spider() function in order to visualize the differences between the 5 classifiers





“The differences can be summed up with the following tables”

Difference between all performances of all classifiers (in testing):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Classifier** | **Accuracy** | **F1-score** | **Precision** | **Recall** | **AUC** |
| **SVM** | 0.91 | 0.87 | 0.84 | 0.91 | 0.91 |
| **KNN** | 0.94 | 0.95 | 0.95 | 0.94 | 0.93 |
| **Decision Tree** | 0.94 | 0.94 | 0.94 | 0.94 | 0.92 |
| **Logistic Regression** | 0.91 | 0.90 | 0.90 | 0.91 | 0.92 |
| **Naïve Bayes** | 0.93 | 0.91 | 0.92 | 0.93 | 0.89 |

Difference between all performances of all classifiers (in validating):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classifier** | **Accuracy** | **F1-score** | **Precision** | **Recall** |
| **SVM** | 0.83 | 0.76 | 0.70 | 0.83 |
| **KNN** | 0.93 | 0.93 | 0.92 | 0.93 |
| **Decision Tree** | 0.93 | 0.92 | 0.92 | 0.93 |
| **Logistic Regression** | 0.93 | 0.93 | 0.93 | 0.93 |
| **Naïve Bayes** | 0.90 | 0.88 | 0.91 | 0.90 |