Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1

Article

Comparison of Classification Models for Breast Cancer Identification using Google Colab

Sundarambal Balaraman [1,*]

[1] Chennai Institute of Technology; sundariprabu@gmail.com
* Correspondence: sundariprabu@gmail.com

Abstract: Classification algorithms are very widely used algorithms for the study of various categories of data located in multiple databases that have real-world implementations. The main purpose of this research work is to identify the efficiency of classification algorithms in the study of breast cancer analysis. Mortality rate of women increases due to frequent cases of breast cancer. The conventional method of diagnosing breast cancer is time consuming and hence research works are being carried out in multiple dimensions to address this issue. In this research work, Google colab, an excellent environment for Python coders, is used as a tool to implement machine learning algorithms for predicting the type of cancer. The performance of machine learning algorithms is analyzed based on the accuracy obtained from various classification models such as logistic regression, K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Naïve Bayes, Decision Tree and Random forest. Experiments show that these classifiers work well for the classification of breast cancers with accuracy>90% and the logistic regression stood top with an accuracy of 98.5%. Also implementation using Google colab made the task very easier without spending hours of installation of environment and supporting libraries which we used to do earlier.

Keywords: classification; machine learning; breast cancer

1. Introduction

Data mining (DM) is the use of advanced data processing techniques to discover correlations between data objects. The three core data mining methods are regression, classification and clustering. DM techniques are applicable in various fields such as research, industry, intelligent data processing and healthcare sector. DM refers to the extraction of knowledge from massive amount of data.

The most commonly reported cancer in women kind is breast cancer, according to world health organization [1]. Almost 2.1 million women is subjected to breast cancer and on an average about 627,000 women died every year. The two types of breast cancer are malignant and benign. If cancer identified at benign stage, the possibility to improve the life span of patient increases whereas if developed to malignant, the cells multiply and grow in the breast tissue beyond the control making patient life miserable. Many research works are carried out to identify the early stage of cancer and to save human life. Advances in cancer detection and treatment increase the longevity of the patients. The experiments are thus carried out to identify the best classification algorithm to identify the type of cancer. In this work, classification models are used to predict the type of breast cancer.

2. Literature Review

In the literature, Ireaneus.Y et al., made an attempt to detect tumors at an early stage using SVM classifier [2]. The experiment is performed with the mammogram images in which the features are extracted and classified using SVM classifier. Arushi Agarwal et al., compared the performance of

© 2020 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1

two models built using KNN and logistic regression classifier and found that logistic regression outperforms KNN. The implementation is done using anaconda platform [3]. Abein Fred Agarap compared six machine learning algorithms such as Gated Recurrent Unit-SVM, Linear Regression, Multilayer Perceptron (MLP), Nearest Neighbor, Softmax Regression and SVM and proved that MLP gave 99.04% accuracy with test data [4].

Rafaqat Alam Khan et al., in their work showed classification accuracy of 98.24%, 98.24% and 98.07% with Logistic regression, Linear Regression and SVM respectively [5]. In another work [6], classification accuracy of 89.55% is achieved with Naïve Bayes classifier. A.Sh et al focused on different types of training testing data partition, 50-50%, 70-30% and 80-20% with SVM classifier [7]. Though 70-30% split is standardized as the best choice in research community, they proved 80-20% gave an accuracy of 99.51%

J.D.Malley et al., [8] used principal component analysis to extract the dominant features and tried with Naïve Bayes, SVM and ensembles. They also obtained accuracy of about 95% in all algorithms. S.K.Mandal applied Naïve Bayes, Logistic Regression and Decision Tree Algorithms and got the classification accuracy of 94.40%, 97.90% and 96.50% respectively [9]. The survey shows that machine learning algorithms give higher classification accuracy and hence this work focused on applying six machine learning algorithms such as Logistic Regression, KNN, SVM, Naïve Bayes, Decision Tree and Random Forest for the classification of breast cancer.

3. Materials and Methods

3.1. Libraries

Google colaboratory environment is used to implement classification algorithms in Python. The libraries imported in this work are numpy, matplotlib and pandas.

3.2. Dataset

The dataset is obtained from breast cancer Wisconsin dataset which is publicly available [10]. The dataset contains the observations of 569 patients having 357 malignant and 212 benign cases. The dataset contains the features computed from biopsy images. To address the false positive rate in breast cancer screening with mammograms, biopsy imaging is done finally to confirm the presence of cancer [11]. The various attributes available in the dataset are shown in Table 1. Each feature is computed in three aspects such as its mean, standard error and worst values contributing 30 features to feature set.

Table 1. Features of Wisconsin breast cancer dataset.

S.No	Feature
1	Radius
2	Texture
3	Perimeter
4	Area
5	Smoothness
6	Compactness
7	Concavity
8	Concave Points
9	Symmetry
10	Fractal Dimension

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1

3.3. Dataset Preprocessing

As a preprocessing step, normalization is done on dataset using StandardScaler function. The proposed model is shown in Figure 1.

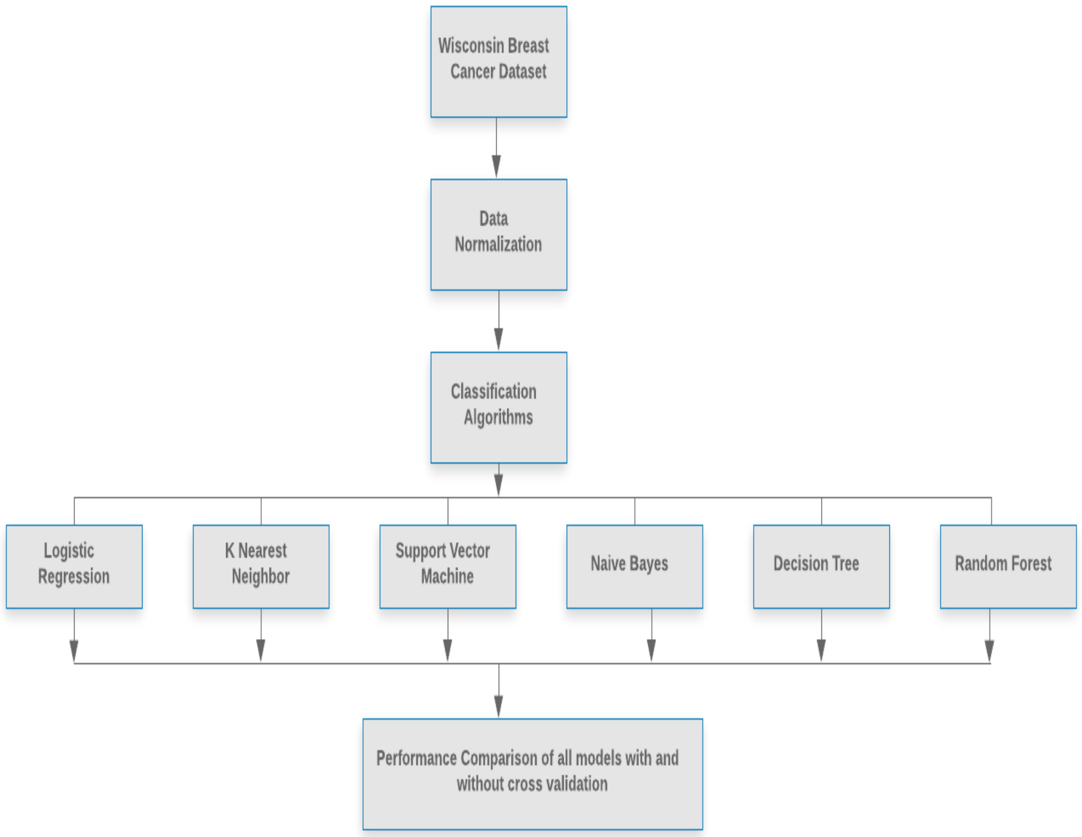

Wisconsin Breast Cancer Dataset
Data Normalization
Classification Algorithms
Logistic Regression
K Nearest Neighbor
Support Vector Machine
Naive Bayes
Decision Tree
Random Forest
Performance Comparison of all models with and without cross validation

Figure 1. Proposed model for breast cancer classification.

3.4. Logistic Regression

Logistic Regression is a supervised machine learning algorithm dedicated for classification purposes [13]. There are two types of linear regression namely simple and multiple linear regressions. Simple linear regression has one independent variable and multiple linear regression has many independent variables as shown in equations 1 and 2 respectively.

$$Y=b_o+b_1*x, \tag{1}$$

$$Y=b_0+b_1*x_1+b_2*x_2+\ldots+b_n*x_n, \tag{2}$$

Here X is the independent variable and the outcome Y is dependent variable. The objective of linear regression is to find the best fitting line for the observed information as shown in Figure 2. The best fitting line fits the observations in data set. With logistic regression, we can identify probabilities

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1

between 0 and 1. As we bother only about the prediction instead of probability, the probability greater than 0.5 is considered as yes and lower than 0.5 is considered as no.

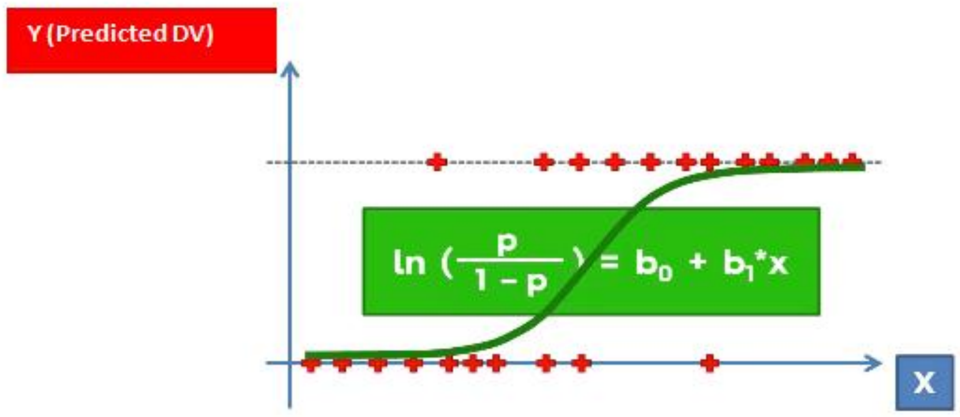


Figure 2. Illustration of Logistic Regression.

3.5. K-Nearest Neighbor (KNN)

Let there are two categories of data points as shown in Figure 3. KNN algorithm identifies the way to add a new data point to one of the categories through the following steps.

Step 1: Choose the number K of neighbors, say K=5.

Step 2: Take the K nearest neighbors of the new data point, according to the Euclidean distance.

Step 3: Among these K neighbors, count the number of data points in each category.

Step 4: Assign the new data point to the category where the most number of neighbors available.

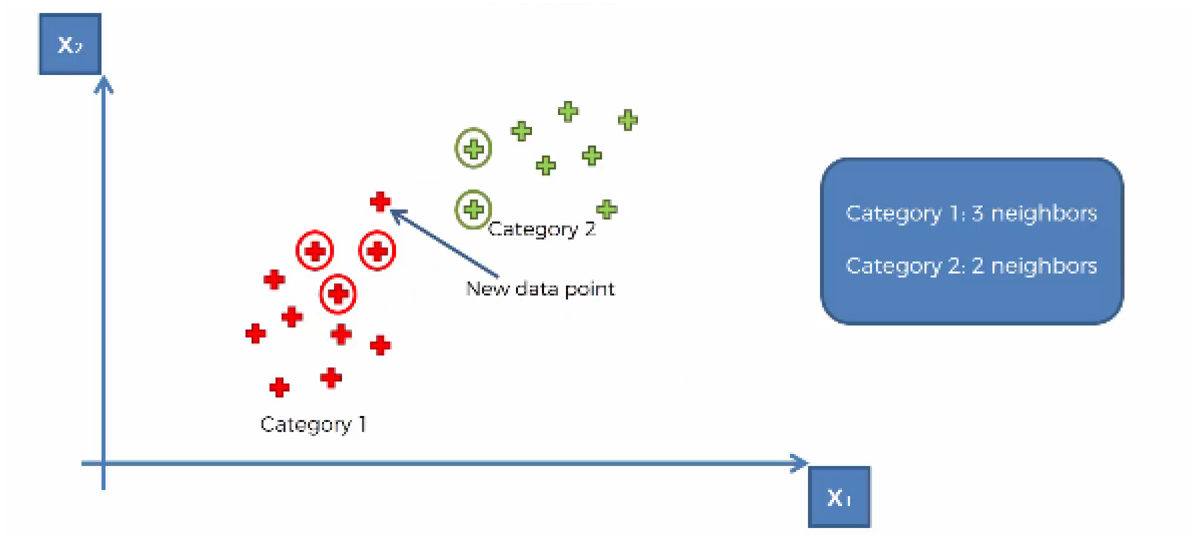


Figure 3. Illustration of K-Nearest Neighbor.

3.6. Support Vector Machine (SVM)

SVM, proposed by Vapnik, is proved to be a very powerful to other machine learning algorithms [14]. It gives the best decision boundary which separate space into classes. The decision boundary is a line that separates the classes of points as shown in Figure 4. Also it has the maximum margin and the points on the margin are termed as support vectors as these points contribute and support the algorithm. The line of decision boundary, the line in the middle of the margin, is called the maximum margin hyperplane or the maximum margin classifier.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1

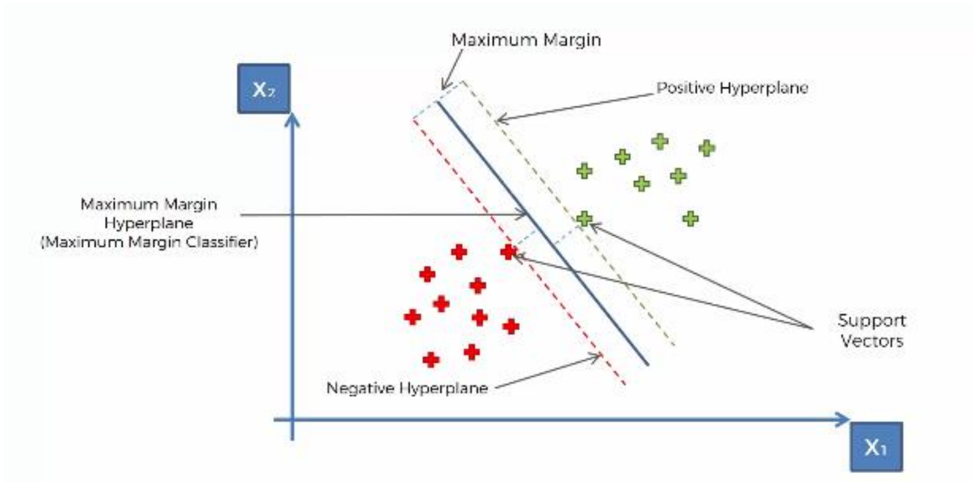


Figure 4. Illustration of Support Vector Machine.

3.7. Naive Bayes

Bayes theorem, represented by equation 3, is a method of finding the probability when certain other probabilities are known.

$$P(Y|X) = \frac{P(Y)P(X|Y)}{P(X)}, \tag{3}$$

where P(Y|X) is the probability of occurrence of Y when X occurred.
P(X|Y) is the probability of occurrence of X when Y occurred.
P(Y) is the probability of occurrence of Y without depending on occurrence of X.
P(X) is the probability of occurrence of X without depending on occurrence of Y.

Variable X represents the feature set and given as X = (X_1, X_2, X_3, … X_n).

Thus the equation takes the form

$$P(Y|X_1, \dots, X_n) = \frac{P(X_1|Y)P(X_2|Y) \dots P(X_n|Y)}{P(X_1)P(X_2) \dots P(X_n)}, \tag{4}$$

The values on the right hand side of the equation can be obtained by analyzing the given dataset and substituting back gives the solution for the left hand side. As the denominator is static, the following equations are used to predict the class.

$$P(Y|X_1, \dots, X_n) \alpha P(Y) \prod_{i=1}^{n} P(X_i|Y), \tag{5}$$

$$Y = argmax_Y P(Y) \prod_{i=1}^{n} P(X_i|Y), \tag{6}$$

3.8. Decision Tree

The decision tree works by splitting the data points as shown in Figure 5. The split is done in such a way to maximize the similar features based on minimal informational entropy in each split. The splits are called leaves and the final split is called terminal leaf.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1

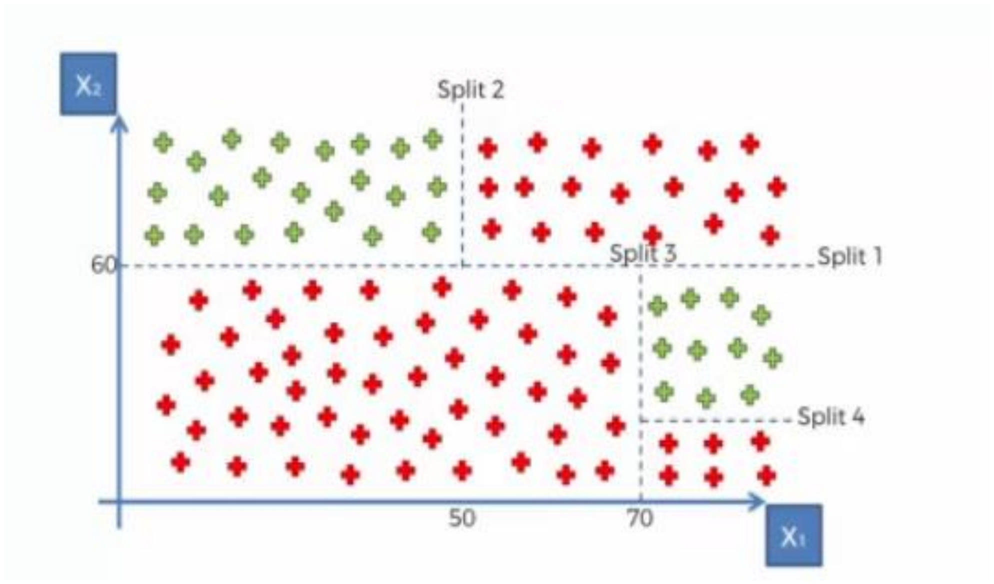


Figure 5. Illustration of Support Vector Machine.

The decision tree for the classification of breast cancer is as shown in Figure 6.

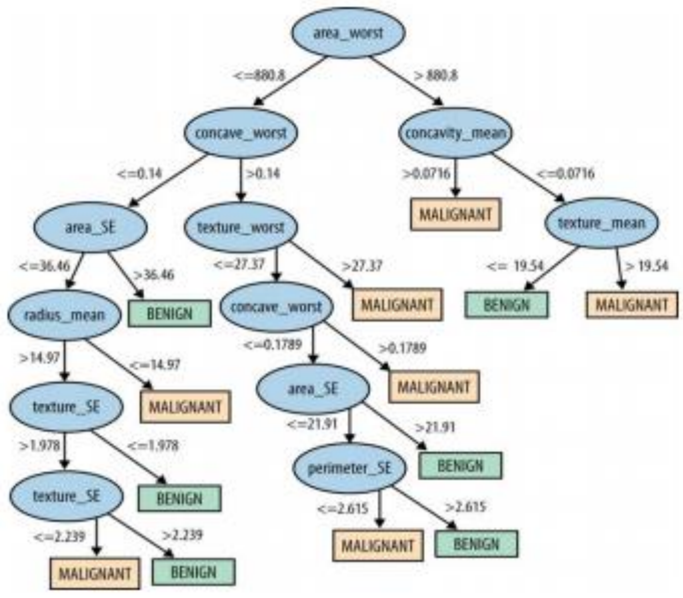


Figure 6. Decision Tree for Breast Cancer Classification.

3.9. Random Forest

Ensemble learning combines the output of multiple machine learning algorithms and gives the better predictive performance and leveraging the other machine learning algorithms. Thus random forest method combines a lot of decision tree method. The steps for implementing random forest are as follows:

Step 1: Pick K random data points from the training set.

Step 2: Build the decision tree associated to these k data points.

Step 3: Choose the number of trees to build and repeat steps 1 and 2.

Step 4: For a new data point, predict the category from each of the decision tree and finally predict the category which has the majority vote.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1

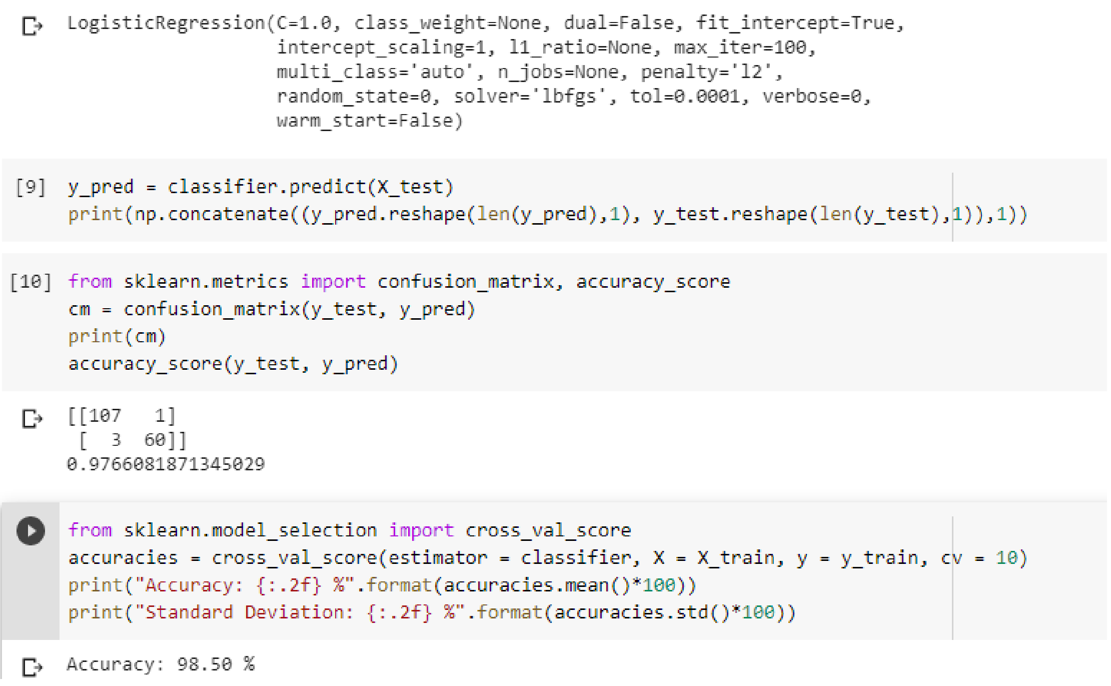
4. Results and Discussion

The learning models are implemented in google colab. The various steps performed to implement the classification algorithms are as follows:

i) Import the required libraries such as numpy, matplotlib and pandas.
ii) Import the dataset into colab.
iii) Split the dataset into 70-30% of training and testing set.

The classification model predicts whether the cancer is malignant or benign based on 30 variables. The model identifies the correlation among the 30 varibles and predicts the type of cancer. The performance of the classification models is evaluated using confusion matrix which shows the number of correct and incorrect predictions.

Though splitting the dataset into training and testing set is the correct way of evaluating the model performance, it is not the best way due to variance problem. The variance problem is getting different accuracy with different test set. So judging model performance only on accuracy obtained from one test set is not the most relavant way to judge the performance. A technique called K fold cross validation fix this variance problem by spinning the training set into 10 fold. K=10 is commonly used in machine learning [12]. The model is trained for 9 folds and tested for remaining fold. With 10 folds, we have 10 different combinations of 9 folds to train the model and one for test. Thus we can train and test the model on all ten combinations of training and test set. Taking an average of different accuracies and also computing the standard deviation to look for variance gives much better idea of the model performance. The confusion matrix, accuracy before and after applying cross validation of all the classifier models is shown in figures 7 to 12.



```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=0, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

```
[9]  y_pred = classifier.predict(X_test)
     print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
```

```
[10] from sklearn.metrics import confusion_matrix, accuracy_score
     cm = confusion_matrix(y_test, y_pred)
     print(cm)
     accuracy_score(y_test, y_pred)
```

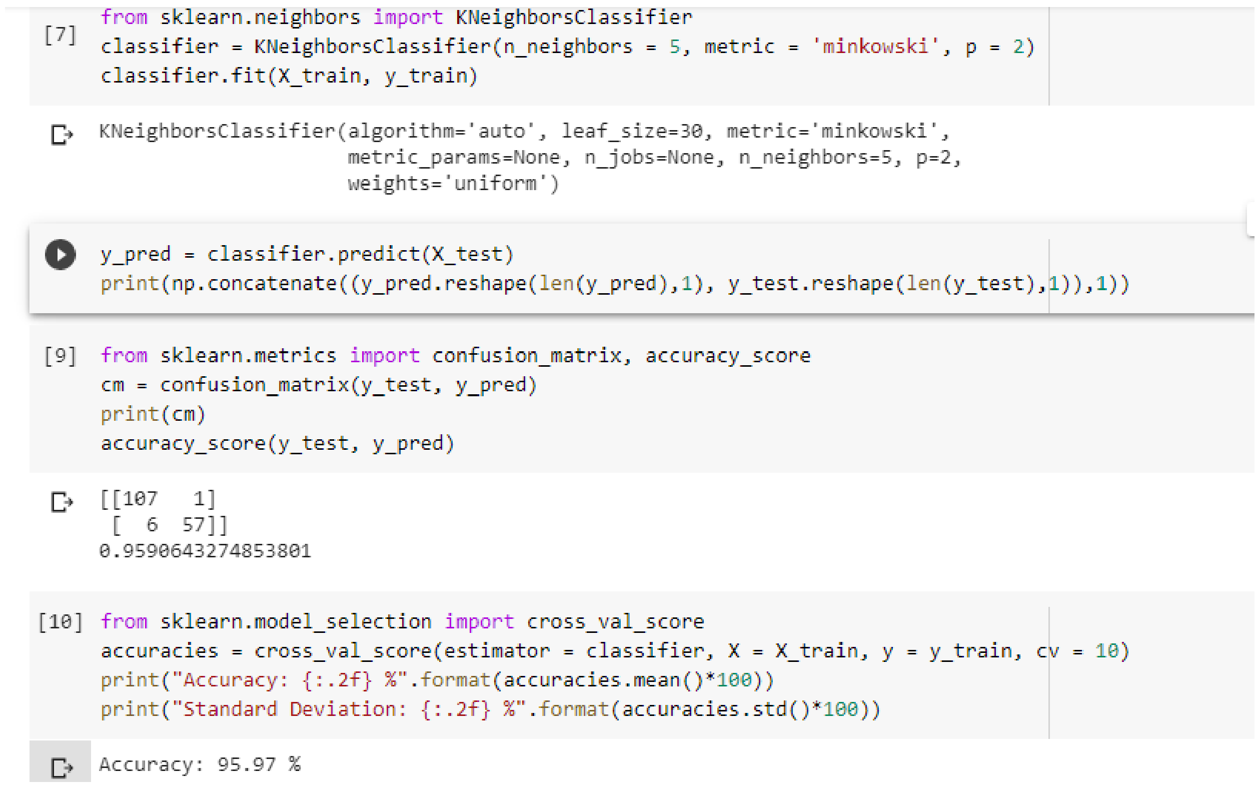
```
[[107   1]
 [  3  60]]
0.9766081871345029
```

```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = classifier, X = X_train, y = y_train, cv = 10)
print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))
```

```
Accuracy: 98.50 %
```

Figure 7. Output of Logistic Regression Classifier.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1



```
[7] from sklearn.neighbors import KNeighborsClassifier
    classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
    classifier.fit(X_train, y_train)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
```

```
y_pred = classifier.predict(X_test)
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
```

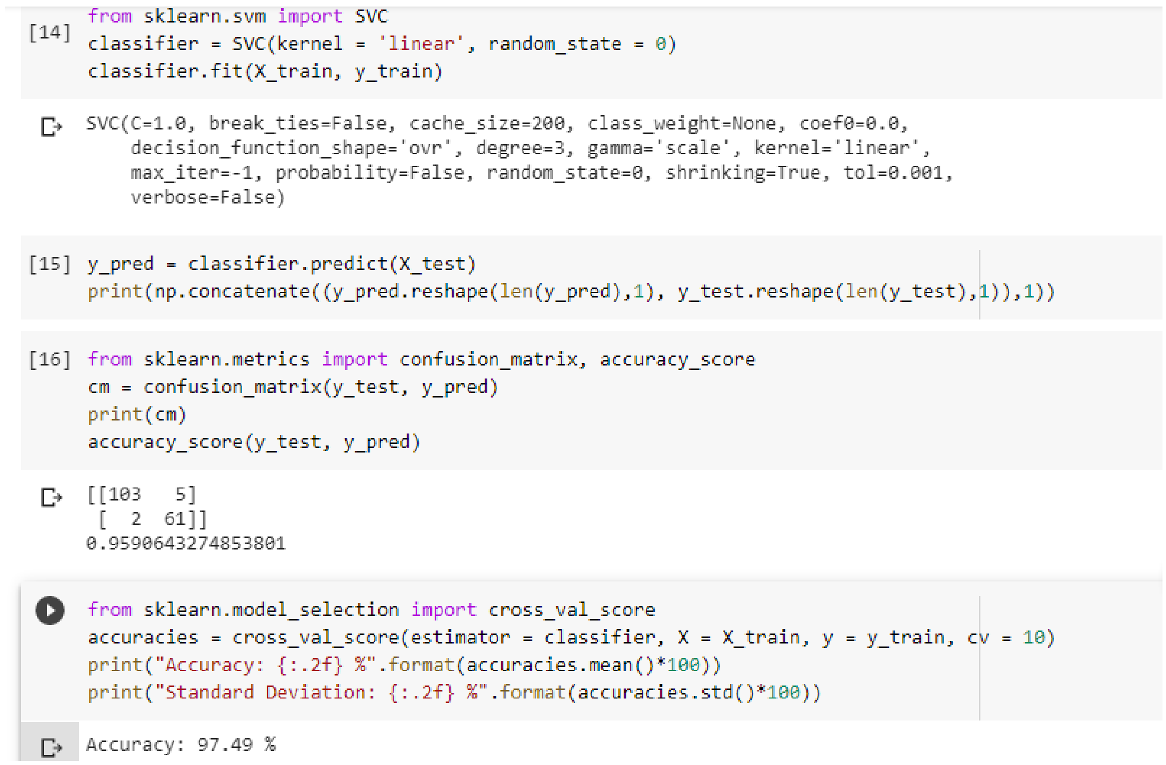
```
[9] from sklearn.metrics import confusion_matrix, accuracy_score
    cm = confusion_matrix(y_test, y_pred)
    print(cm)
    accuracy_score(y_test, y_pred)
```

```
[[107   1]
 [  6  57]]
0.9590643274853801
```

```
[10] from sklearn.model_selection import cross_val_score
     accuracies = cross_val_score(estimator = classifier, X = X_train, y = y_train, cv = 10)
     print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
     print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))
```

```
Accuracy: 95.97 %
```

Figure 8. Output of K-Nearest Neighbors Classifier.



```
[14] from sklearn.svm import SVC
     classifier = SVC(kernel = 'linear', random_state = 0)
     classifier.fit(X_train, y_train)
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='linear',
    max_iter=-1, probability=False, random_state=0, shrinking=True, tol=0.001,
    verbose=False)
```

```
[15] y_pred = classifier.predict(X_test)
     print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
```

```
[16] from sklearn.metrics import confusion_matrix, accuracy_score
     cm = confusion_matrix(y_test, y_pred)
     print(cm)
     accuracy_score(y_test, y_pred)
```

```
[[103   5]
 [  2  61]]
0.9590643274853801
```

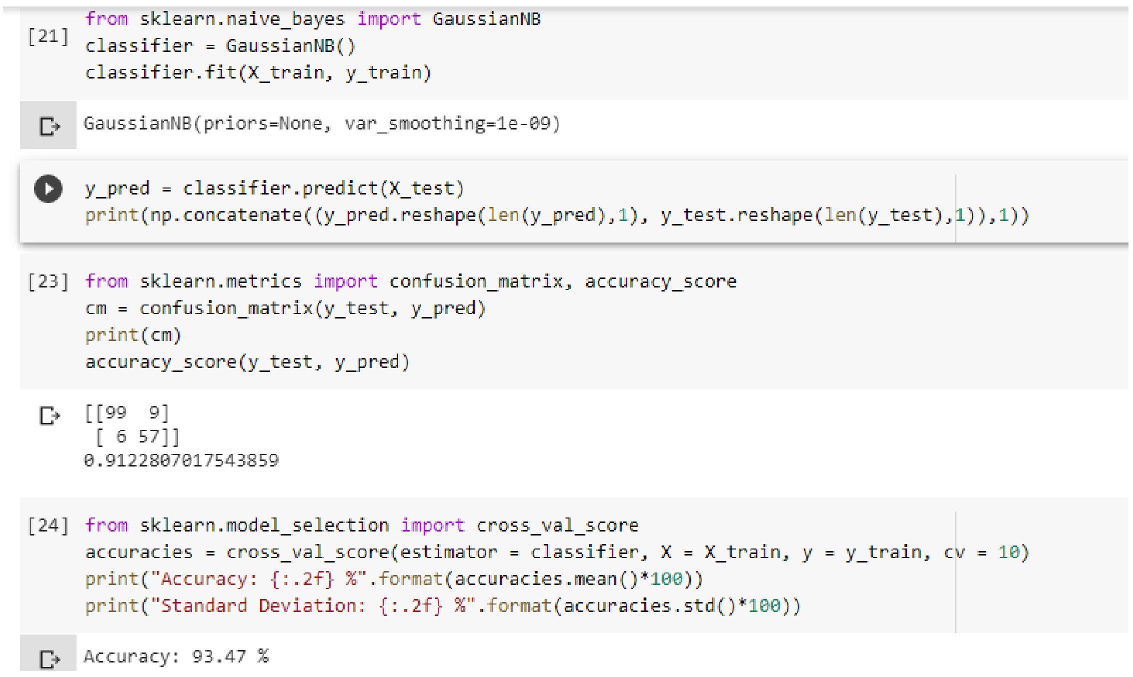
```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = classifier, X = X_train, y = y_train, cv = 10)
print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))
```

```
Accuracy: 97.49 %
```

Figure 9. Output of Support Vector Machine Classifier.

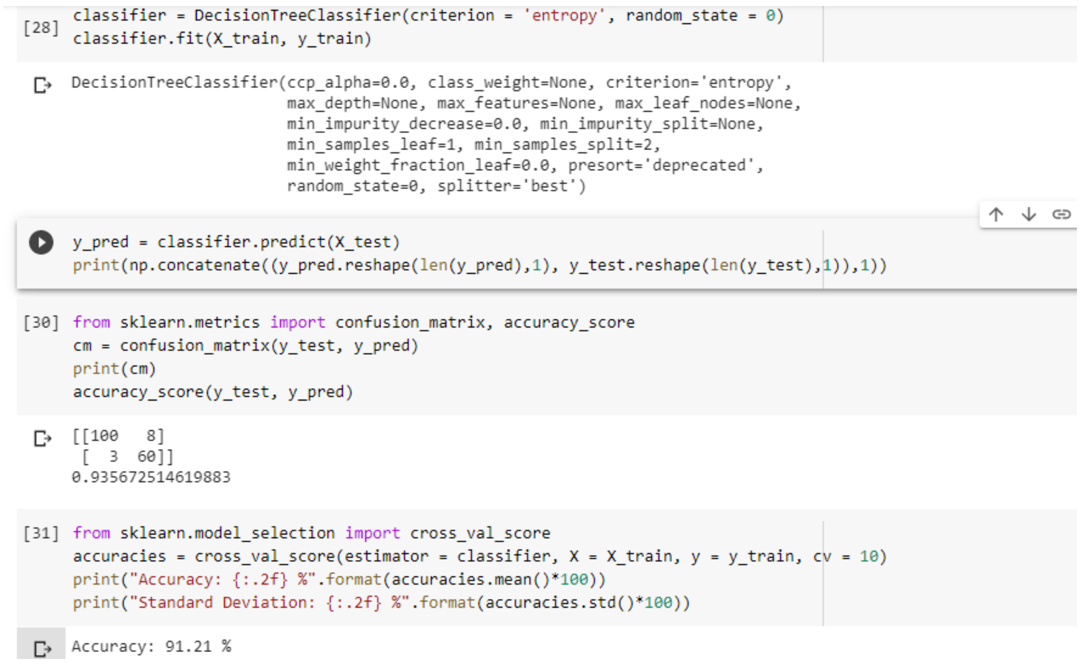
Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1

3.3.



```
[21] from sklearn.naive_bayes import GaussianNB
     classifier = GaussianNB()
     classifier.fit(X_train, y_train)

     GaussianNB(priors=None, var_smoothing=1e-09)

     y_pred = classifier.predict(X_test)
     print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))

[23] from sklearn.metrics import confusion_matrix, accuracy_score
     cm = confusion_matrix(y_test, y_pred)
     print(cm)
     accuracy_score(y_test, y_pred)

     [[99  9]
      [ 6 57]]
     0.9122807017543859

[24] from sklearn.model_selection import cross_val_score
     accuracies = cross_val_score(estimator = classifier, X = X_train, y = y_train, cv = 10)
     print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
     print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))

     Accuracy: 93.47 %
```

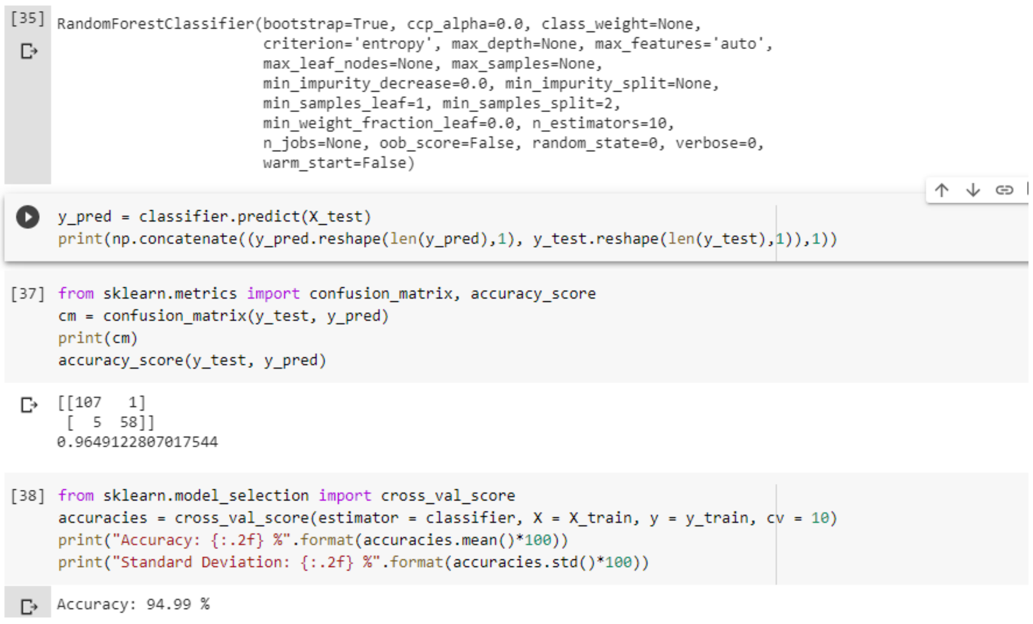
Figure 10. Output of Naïve Bayes Classifier.



```
[28] classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
     classifier.fit(X_train, y_train)

     DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='entropy',
                            max_depth=None, max_features=None, max_leaf_nodes=None,
                            min_impurity_decrease=0.0, min_impurity_split=None,
                            min_samples_leaf=1, min_samples_split=2,
                            min_weight_fraction_leaf=0.0, presort='deprecated',
                            random_state=0, splitter='best')

     y_pred = classifier.predict(X_test)
     print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))

[30] from sklearn.metrics import confusion_matrix, accuracy_score
     cm = confusion_matrix(y_test, y_pred)
     print(cm)
     accuracy_score(y_test, y_pred)

     [[100   8]
      [  3  60]]
     0.935672514619883

[31] from sklearn.model_selection import cross_val_score
     accuracies = cross_val_score(estimator = classifier, X = X_train, y = y_train, cv = 10)
     print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
     print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))

     Accuracy: 91.21 %
```

Figure 11. Output of Decision Tree Classifier.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1



```
[35] RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                            criterion='entropy', max_depth=None, max_features='auto',
                            max_leaf_nodes=None, max_samples=None,
                            min_impurity_decrease=0.0, min_impurity_split=None,
                            min_samples_leaf=1, min_samples_split=2,
                            min_weight_fraction_leaf=0.0, n_estimators=10,
                            n_jobs=None, oob_score=False, random_state=0, verbose=0,
                            warm_start=False)

y_pred = classifier.predict(X_test)
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))

[37] from sklearn.metrics import confusion_matrix, accuracy_score
     cm = confusion_matrix(y_test, y_pred)
     print(cm)
     accuracy_score(y_test, y_pred)

[[107   1]
 [  5  58]]
0.9649122807017544

[38] from sklearn.model_selection import cross_val_score
     accuracies = cross_val_score(estimator = classifier, X = X_train, y = y_train, cv = 10)
     print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
     print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))

Accuracy: 94.99 %
```

Figure 12. Output of Random Forest Classifier.

The accuracy obtained from various classification algorithms before and after cross validation is shown in Table 2.

Table 2. Performance analysis of machine learning algorithms on breast cancer classification

Algorithm	No. of images misclassified	Accuracy without Cross Validation (%)	Accuracy with Cross Validation (%)
Logistic Regression	4	97.66	98.5
K-Nearest Neighbors	7	95.90	96.97
SVM	7	95.90	97.49
Naïve Bayes	15	91.22	93.47
Decision Tree	11	93.56	91.21
Random Forest	6	96.49	94.99

The experiments are carried out instantaneously in Google colaboratory and it is proved that this excellent environment removes the hurdles of setting up environment for implementing machine algorithms written in Python. It uses tensorflow backend. The results show that the machine learning algorithms classify well the Wisconsin breast cancer dataset. With cross validation, the performance of the classifier is improved except for Decision Tree and Random Forest. Logistic regression gives the highest classification accuracy of 98.5% in which only 4 images are misclassified. The confusion matrix of the logistic regression is as shown in Table 3.

Table 3. Confusion Matrix of Logistic Regression model

N=171	Predicted Benign	Predicted Malignant
Actual : Benign	107	1
Actual : Malignant	3	60

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2020 doi:10.20944/preprints202005.0328.v1

5. Conclusions

The various classification models are implemented in this paper and their accuracy are compared. The classification accuracy of the various classifiers such as Logistic regression, KNN, SVM, Naïve Bayes, Decision Tree and Random Forest exceeds 90% in which logistic regression being at the top with 98.5%. The application of machine learning thus becomes robust in all domains. Hence it is found that healthcare sector is not an exception to this with the obtained results. Also implementation using Google colab made the task very easier without spending hours of installation of environment and supporting libraries which we used to do earlier. This work can be extended to employing real time medical information collected from various cancer centres and converted into desktop application so that the doctors can use this as a supporting tool in their diagnosis.

Author Contributions: B.S designed, implemented, analyzed and wrote the paper. The author has read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The author declares no conflict of interest.

References

1. https://www.who.int/cancer/prevention/diagnosis-screening/breast-cancer/en/ (accessed on 18/05/2020)
2. Ireaneus Anna Rejani, Y.; ThamaraiSelvi, S. Early Detection Of Breast Cancer Using SVM Classifier Technique. Int. J. Comput. Sci. Eng. **2009**, 1, 127-130.
3. Arushi Agarwal; Ankur Saxena. Malignant Tumor Detection using Machine Learning through Scikit-learn. Int. J. Pure Appl. Math **2018,** 15, 2863-2874.
4. arXiv:1711.07831 [cs.LG] (accessed on 18/05/2020)
5. Rafakat Alam Khan; Taseer Suleman; Muhammed Sajid Farooq; Muhammed Hassan Rafiq; Muhammed Arslan Tariq. Data Mining Algorithms for Classification of Diagnostic Cancer using Genetic Optimization Algorithms. Int. J. Comput. Sci. Network Secur. **2017**, 17, 207-212.
6. Mehmet Fatih Akay. Support vector machines combined with feature selection for breast cancer diagnosis. Expert Syst. Appl. **2009**, 36, 3240-3247.
7. Sh, A.; Shahraki, H.; Rowhanimanesh, AR.; Eslami,S. Feature selection using a genetic algorithm for breast cancer diagnosis: an experiment on three different datasets. Iran J Basic Med Sci **2016,** 19, 476-482.
8. Malley, J.D.; Kruppa, J.; Dasgupta, A.; Malley, K.G.; Ziegler, A. Study and Analysis of Breast Cancer Cell Detection using Naïve Bayes, SVM and Ensemble Algorithms. Int. J. Comput. Appl. **2016,** 145.
9. Mandal, S.K. Performance Analysis Of Data Mining Algorithms For Breast Cancer Cell Detection Using Naïve Bayes, Logistic Regression and Decision Tree. Int. J. Eng. Comput. Sci. **2017,** 6, 20388-20391.
10. http://archive.ics.uci.edu/ml/ (accessed on 18/05/2020)
11. Sundarambal, B.; Mathana, J.M.; Subramanian, S.; Sandesh, H.; Omprakash, A. A Detailed Investigation on Reduction of False Positive Rate in Breast Cancer Detection. Proceedings of the 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 6-7 March 2020; IEEE.
12. https://machinelearningmastery.com/k-fold-cross-validation/ (accessed on 18/05/2020)
13. https://towardsdatascience.com/logistic-regression-classifier-8583e0c3cf9 (accessed on 18/05/2020)
14. https://machinelearningmastery.com/support-vector-machines-for-machine-learning/ (accessed on 18/05/2020)