CS 6375

SciKit Lab 2

PSD170000

Paril Doshi

Number of free late days used: \_\_\_\_\_\_\_0\_\_\_\_\_\_\_\_\_\_\_\_\_   
Note: You are allowed a **total** of 4 free late days for the **entire semester**. You can use at most 2 for each assignment. After that, there will be a penalty of 10% for each late day.

Output:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Algorithm | Best Parameter | Avg Precision | Avg Recall | Avg F1 | Accuracy Score |
| Decision Tree | 'max\_depth': 5,  'max\_features': 3, 'max\_leaf\_nodes': 1600, 'min\_impurity\_decrease':0.0001,  'min\_samples\_leaf': 4, 'min\_samples\_split': 9, 'min\_weight\_fraction\_leaf': 0 | 0.9483 | 0.9606 | 0.9831 | 0.9730 |
| Neural Net  **(Found best among all)** | 'activation': 'tanh', 'alpha': 0.1, 'early\_stopping': False, 'hidden\_layer\_sizes': (5, 5, 5), 'learning\_rate': 'constant', 'max\_iter': 1000, 'momentum': 0.9, 'tol': 0.0001 | 1.0 | 1.0 | 1.0 | 1.0 |
| Support Vector Machine | 'C': 10, 'gamma': 0.001, 'kernel': 'linear' | 0.9755 | 1.0 | 0.9881 | 0.9902 |
| Gaussian Naïve Bayes | 'priors': [0.5, 0.5] | 0.7786 | 0.9528 | 0.8443 | 0.8398 |
| Logistic Regression | 'C': 1.0, 'fit\_intercept': True, 'intercept\_scaling': 1,  'max\_iter': 100,  'multi\_class': 'ovr', 'penalty': 'l2', 'solver': 'newton-cg',  'tol': 0.0001 | 0.9619 | 1.0 | 0.9729 | 0.9757 |
| K-Nearest Neighbors | 'algorithm': 'auto', 'n\_neighbors': 10, 'p': 1, 'weights': 'uniform' | 1.0 | 1.0 | 1.0 | 1.0 |
| Bagging | 'max\_features': 3, 'max\_samples': 10, 'n\_estimators': 17, 'random\_state': 8 | 0.8586 | 0.9943 | 0.8901 | 0.9077 |
| Random Forest | 'criterion': 'gini',  'max\_depth': 20,  'max\_features': 1, 'min\_samples\_leaf': 2, 'min\_samples\_split': 5, 'n\_estimators': 10 | 0.9862 | 0.9673 | 0.9863 | 0.9927 |
| Ada Boost Classifier | 'algorithm': 'SAMME', 'base\_estimator': DecisionTreeClassifier(  class\_weight=None, criterion='gini', 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=False, random\_state=None,  splitter='best'), 'learning\_rate': 1, 'n\_estimators': 10, 'random\_state': 4 | 0.9575 | 0.9787 | 0.9735 | 0.9757 |
| Gradient Boosting Classifier | 'learning\_rate': 0.4,  'loss': 'deviance',  'max\_features': 1, 'n\_estimators': 125 | 0.9809 | 1.0 | 0.9919 | 0.9902 |
| XGBoost | 'learning\_rate': 1, 'max\_delta\_step': 1, 'min\_child\_weight': 1, 'n\_estimators': 10, 'seed': 1 | 1.0 | 1.0 | 1.0 | 1.0 |

I have submitted the code in file(Learn SciKit 1.py). I have also uploaded the code on google colab.

Colab Link:-

<https://colab.research.google.com/drive/13Hx5ySQfUMnQrBWcYuTjN_2fRaE9l7wH>

There is a need to input the index number of classifier while running the code to select the algorithm needed to be run on the dataset. The number ranges from (1 to 11).

1. Decision Tree

2. Neural Net

3. Support Vector Machine

4. Gaussuian Naive Bayes

5. Logistic Regression

6. K-Nearest Neighbors

7. Bagging

8. Random Forest

9. AdaBoost Classifier

10. Gradient Boosting Classifier

11. XGBoost

Please enter the respective number of classifier you want to use

The dataset link is:- <https://archive.ics.uci.edu/ml/machine-learning-databases/00267/data_banknote_authentication.txt>

Report on results:

* Having used the best combination of parameters for all Algorithms, the accuracy score for all the algorithms have reached above 90% except Naïve Bayes.
* The best classifier for bank note authentication dataset found is Neural Net, K-Nearest Neighbors, XGBoost.

Why is Neural Net best algorithm:

Firstly, tanh is the best activation function. It is because of there are only two classes(0 and 1) the data needs to be classified in.

The second property of tanh is that it maps the negative inputs strongly negative and zero inputs maps near zero. As the current dataset is having negative values tanh outperformed other activation functions.

The confusion matrix of this displayed that there were 0 points misclassified.

The runner up algorithm found is K-Nearest Neighbors Algorithm.

* There are already three classifier(Neural Net, Gradient Boosting Classifier, K-Nearest Neighbours) who predicts the test data to 99% not much scope of further increase in accuracy.