**Assignment-5 Report**

**Team Members:** 1) Hardik Punjabi hvp160030

2) Aakash Shah axs165231

* **Dataset Used:**

**Breast Cancer Wisconsin (Diagnostic) Data Set**

**url:** <https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.data>

**Dataset characteristic: Multivariate**

**Number of Attributes:32**

**Number of Records:569**

**Missing Values: None**

**Associated Task: Classification**

**Area: Life**

**Attribute Characteristic: Real**

**Detailed description of dataset:** <https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.names>

* **Pre-processing:**

The data-set is directly loaded using the scikitlearn library.( using scikitlearn.datasets and using load\_breast\_cancer() function)

Pseudocode:

from sklearn.datasets import load\_breast\_cancer

.

.

.

dataset=load\_breast\_cancer()

The dataset does not contain any missing values, however it includes unnecessary data such as ID, which is discarded in the pre-processing step. Also the data is standardized using the scikitlearn preprocessing library.

Pseudocode:

from sklearn import preprocessing

:

:

X=dataset.data

Scaled\_x=preprocessing.scale(X)

* **Pseudocode:**

import libraries

:

:

:

:

dataset=load\_dataset

preprocessed\_dataset=preprocess\_scale(dataset)

:

:

List\_model=[]

Model1=Load\_model1(best\_parameters)

List\_model.append(model1)

Model2=Load\_model2(best\_parameters)

List\_model.append(model2)

:

:

Modeln=Load\_modeln(best\_parameters)

List\_model.append(modeln)

:

:

//cross validation, running all models one by one and displaying accuracy of each model

for name, model in models:

kfold = model\_selection.KFold(n\_splits=10, random\_state=seed)

cv\_results = model\_selection.cross\_val\_score(model,X, Y, cv=kfold, scoring=scoring)

results.append(cv\_results)

names.append(name)

print('{:20} {:0.4f}'.format(name, cv\_results.mean()\*100))

final.append(cv\_results.mean()\*100)

**Evaluation Metric Used:**

The evaluation of each model is done using two different evaluation matrices.

One of them is the accuracy of each model that we normally use. The accuracy is calculated by checking how many test instances were predicted correctly by the model. Here we are using 10-fold cross validation so the accuracy is the average accuracy of the model over 10 folds.

The other evaluation matric used is the precision matric which is calculated as follows:

Precision= True Positives / (True positives+ False positives)

The precision is averaged using the weighted average over 10 folds, the weighted average allows to take attribute imbalance into consideration.

**Results:**

Number of instances in dataset: 569

Number of attributes in dataset: 32

How many fold cross-validation performed: 10

|  |  |  |  |
| --- | --- | --- | --- |
| Classifier | Best Parameters Used | Accuracy | Precision |
| Decision Tree | maximum\_feature=n  min\_impurity\_split=0.1 | 92.79 % | 93.49 % |
| Perceptron | penalty=NONE  n\_iter=5  warm\_start=false | 90.15 % | 91.06 % |
| Neural Net | hidden\_layer\_sizes=(50,45)  activation='logistic'  learning\_rate\_init=0.001 | 91.56 % | 92.64 % |
| Deep Learning | hidden\_layer\_sizes=(50,45,35,30,28,25,20)  activation='identity'  learning\_rate\_init=0.001 | 84.55 % | 80.10 % |
| SVM | c=10  kernel='linear'  tol=0.09 | 94.91 % | 95.42 % |
| Naïve Bayes | prior=default | 93.68 % | 94.27 % |
| Logistic Regression | dual=false  max\_iter=100  tol=0.0001 | 95.08 % | 95.80 % |
| k-Nearest Neighbors | n\_neighbours=9  algorithm='ball\_tree'  weights='distance' | 92.63 % | 94.20 % |
| Bagging | n\_estimator=50  bootstrap=true  bootstrap\_features=false | 95.96 % | 96.73 % |
| Random Forests | n\_estimator=30  bootstrap=True  min\_impurity\_split=0.00001 | 95.79 % | 96.51 % |
| AdaBoost | n\_estimator=200  algorithm='SAMME' | 97.89 % | 97.94 % |
| Gradient Boosting | n\_estimator=200  max\_depth=3  min\_impurity\_split=0.0000001 | 96.84 % | 96.92 % |

**Analysis:**

Each model was tested on the same data separately several times to find the best parameters for each model. Individual results of each experiment are mentioned in the logfile submitted with the assignment. Also we tried to test the model on different datasets to compare the results and found out that if the data type varies the model accuracy may vary and we found that some of the model that perform bad on some data sets perform better on other data sets. So in practice you cannot say for sure that a single model is best for all datasets, different models needs to tested and the model best suited for the particular data set should be selected.

There are some models that perform better than other models in general because of their complexity and efficiency. The models that perform better in general on each type of dataset are Bagging , AdaBoost and Gradient boosting (according to our experiments). The reason for these models being better than other is obvious. These models uses other models to predict the outcome several times and gradually overcomes the limitations of a single model. Also the dataset is modified by such models to avoid noise in the data.

The weakest model is the perceptron because a single perceptron tries to linearly separate which might not be possible in complex datasets. Perceptron model can still give good accuracy on some datasets if they are linearly separable and the model is tuned to the best parameters. Knn can also be considered as a week model as it just considers its neighbours to predict the outcome, but surprisingly it performs better most of the times if the model is tuned to the best suitable parameters but finding such parameters easily may not be possible for each dataset.

Accuracy is used as the evaluation metric for each model but it may not be the best idea to use accuracy to compare all the models. As the accuracy only considers the correctly predicted data, it does not consider false positives and false negatives which may be required in some application of the model. We have used The precision matric to evaluate the model which is better than accuracy for comparison of models.