# MODULE 15

1. **Diabetes Dataset**

**Package used in R**

caTools- Used to run Random Forest.

xgboost- Used to run the Extreme Gradient Algorithm.

Adabag- Used to run the adaboost Algorithm.

Rpart- Used to control the CP value.

**Package used in python**

Pandas - Used for data manipulation

Sklearn

Train\_test\_split - Used to Split the data to train and test for modeling

accuracy\_score - Calculate the accuracy of the random Forest

AdaBoostClassifier - Used to Run Adaboost Algorithm

Xgboost - Used to Run Extreme Gradient Algorithm.

GridSearchCV - Used to Run Grid Search on Extreme Gradient Algorithm

**Loading the data**

Loading the Diabetes dataset in R and Python.

In R Import data set by base was used and converted all the strings to factors.

**EDA**

No NaN Data Found in the dataset

Normalizing the dataset with Min Max method.

Typecasting the Class.variable variables to true or false

Renaming the Dataset variables as all the variables as space in it.

Creating dummy variables for categorical variables.

**Data Partitioning**

Seed option is used to ensure same random numbers generated to split the data.

Train and test data is split into 70:30 Ratio

**ADABOOSTING.**

adaboost <- boosting(Class.variable ~ ., data = Diabetes\_train, mfinal=50, boos = TRUE, control=rpart.control(cp=.10))

Test accuracy: 0.7260 and Train accuracy : 0.815

x\_train, x\_test, y\_train, y\_test = train\_test\_split(predictors, target, test\_size = 0.25, random\_state=0)

ada\_clf = AdaBoostClassifier(learning\_rate = .1, n\_estimators = 700)

Test accuracy: 0.80 and Train accuracy : .85

**Extreme Gradient Algorithm**

xg\_boosting<-xgboost(data= Xmatrix\_train, nround =19000,objective="multi:softmax",eta=0.001, num\_class = 2, max\_depth =1)

**Test Accuracy: 0.765 and Training Accuracy 0.810**

xgb\_clf = xgb.XGBClassifier(max\_depths =2, n\_estimators = 15000, learning\_rate = .2, n\_jobs = -5)

**Accuracy for xgboost method : .757**

param\_test1 = {'max\_depth': range(3,15,1), 'gamma': [0.1,0.2,0.3],

'subsample': [0.8,0.9], 'colsample\_bytree': [0.8, 0.9],

'rag\_alpha': [1e-2, 0.1, 1]}

grid\_search = GridSearchCV(xgb\_clf, param\_test1, n\_jobs = -1, cv = 5, scoring = 'accuracy')

**Accuracy for GridSearch xgboost method : .783**

# WBCD Dataset.

**Package used in R**

caTools- Used to run Random Forest.

xgboost- Used to run the Extreme Gradient Algorithm.

Adabag- Used to run the adaboost Algorithm.

Rpart- Used to control the CP value.

**Package used in python**

Pandas - Used for data manipulation

Sklearn

Train\_test\_split - Used to Split the data to train and test for modeling

accuracy\_score - Calculate the accuracy of the random Forest

AdaBoostClassifier - Used to Run Adaboost Algorithm

Xgboost - Used to Run Extreme Gradient Algorithm.

GridSearchCV - Used to Run Grid Search on Extreme Gradient Algorithm

**Loading the data**

Loading the Diabetes dataset in R and Python.

In R Import data set by base was used and converted all the strings to factors.

**EDA**

No NaN Data Found in the dataset

Removing unwanted Variables.

Normalizing the dataset with Min Max method.

Typecasting the Class.variable variables to true or false according to the sales

Renaming the Dataset variables as all the variables as space in it.

Creating dummy variables for categorical variables.

**Data Partitioning**

Seed option is used to ensure same random numbers generated to split the data.

Train and test data is split into 70:30 Ratio

**ADABOOSTING.**

adaboost<-boosting(diagnosis~.,data=diag\_train,mfinal=50,boos=TRUE, control=rpart.control(cp=.10))

**Test accuracy: 0.98 and Train accuracy : 1**

x\_train, x\_test, y\_train, y\_test = train\_test\_split(predictors, target, test\_size = 0.30, random\_state=0)

ada\_clf = AdaBoostClassifier(learning\_rate = .07, n\_estimators = 500)

**Test accuracy: 0.9707 and Train accuracy : 1**

**Extreme Gradient Algorithm**

xg\_boosting <- xgboost(data = Xmatrix\_train, nround =7, objective = "multi:softmax", eta = 0.3,

num\_class = 5, max\_depth = 50)

**Test Accuracy: 0.982 and Training Accuracy 0.997**

xgb\_clf = xgb.XGBClassifier(max\_depths = 5,n\_estimators=10000, learning\_rate = 0.3, n\_jobs = -1)

**Accuracy for xgboost method : .973**

param\_test1 = {'max\_depth': range(3,15,1), 'gamma': [0.1, 0.2, 0.3],

'subsample': [0.7,0.8, 0.9], 'colsample\_bytree': [0.8, 0,9],

'rag\_alpha': [1e-2, 0.1, 1]}

grid\_search = GridSearchCV(xgb\_clf, param\_test1, n\_jobs = -1, cv = 5, scoring = 'accuracy')

**Accuracy for Grid Search xgboost method : .974**