**M10 Lab: Building the ML Pipeline in Scikit Learn**

In this lab you will use Scikit Learn to build a machine learning pipeline for a classification application.

### **Building the Pipeline**

Load the LibSVM file (Don't use read\_csv() function).  
Create a standard Scaler Object: SS  
Create a Decision Tree Object: DT  
Create a Pipeline that contains two steps for SS -> DT  
Use the Cross Validation Score function & Print the average score

from sklearn.datasets import load\_svmlight\_file

from sklearn.preprocessing import StandardScaler

from sklearn.tree import DecisionTreeClassifier

from sklearn.preprocessing import StandardScaler

from sklearn.pipeline import make\_pipeline

from sklearn.model\_selection import cross\_val\_score

print("Loading Dataset...")

X,y = load\_svmlight\_file("a9a.txt")

print("Loaded!!")

print("Create a Decision Tree Object: DT")

clf = DecisionTreeClassifier(random\_state=0)

print("Create a Pipeline that contains two steps for SS -> DT")

pipe = make\_pipeline(StandardScaler(with\_mean=0,with\_std=1), clf)

pipe

print("Use the Cross Validation Score function & Print the average score")

scores = cross\_val\_score(pipe, X, y, cv=5)

print("Training Complete!")

acc = scores.mean()

print("Cross Validation Mean Accuracy = %0.2f" % acc )

Graphical user interface, text, application

Description automatically generated

### **Parameter Fine Tuning**

In this part you will add a parameter grid to your code to experiment with different parameter values and select the one that gives the highest accuracy. The parameter set is:

Load LibSVM Data  
Create a Standard Scalar Object: SS  
Create a Decision Tree Object: DT  
Create a Pipeline with two steps: SS -> DT  
Create a Parameter Grid [['gini', 'entropy'],[5,10,15,20]]  
Create the Grid Search Cross Validation object  
Call the fit function to fit the pipeline to the data and try different parameter combinations  
Print the best obtained results

print("Loading Dataset...")

X,y = load\_svmlight\_file("a9a.txt")

print("Loaded!!")

print("Create a standard Scaler Object: SS")

scaler = StandardScaler(with\_mean=0, with\_std=1)

print("Create a Decision Tree Object: DT")

clf = DecisionTreeClassifier(random\_state=0)

# from sklearn.pipeline import Pipeline

print("Create a Pipeline that contains two steps for SS -> DT")

pipe =make\_pipeline(scaler, clf)

pipe

# print("Create a Parameter Grid [['gini', 'entropy'],[5,10,15,20]]")

from sklearn.model\_selection import GridSearchCV

param\_grid ={'decisiontreeclassifier\_\_criterion':['entropy', 'gini'],'decisiontreeclassifier\_\_max\_depth' :[5,10,15,20]}

print("Create the Grid Search Cross Validation object")

search = GridSearchCV(estimator=pipe, param\_grid=param\_grid, cv=5)

search.fit(X,y)

search.cv\_results\_

best\_grid = search.best\_estimator\_

print(best\_grid)

search.best\_score\_

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

**Played with different features:**

param\_grid ={'decisiontreeclassifier\_\_criterion':['entropy', 'gini'],'decisiontreeclassifier\_\_max\_depth' :[5,10,15,20]

,'decisiontreeclassifier\_\_splitter':['best','random']}

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text

Description automatically generated

from sklearn.model\_selection import GridSearchCV

param\_grid ={'decisiontreeclassifier\_\_criterion':['entropy', 'gini'],'decisiontreeclassifier\_\_max\_depth' :[5,10,15,20]

,'decisiontreeclassifier\_\_splitter':['best','random'] , 'decisiontreeclassifier\_\_min\_samples\_split':[2,3]}

Graphical user interface, text, email

Description automatically generated

Graphical user interface, application

Description automatically generated

from sklearn.model\_selection import GridSearchCV

param\_grid ={'decisiontreeclassifier\_\_criterion':['entropy', 'gini'],'decisiontreeclassifier\_\_max\_depth' :[5,10,15,20]

,'decisiontreeclassifier\_\_splitter':['best','random'] , 'decisiontreeclassifier\_\_min\_samples\_split':[2,3]

,'decisiontreeclassifier\_\_min\_samples\_leaf':[1,5]}

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated