**ML Weekly Project Report-2 (18/02/23)**

*Due to mid-sems going on we weren’t able to work much on the project. But we tried the available codes for the following Algo’s and were able to understand the work flow of the code:*

1. **Linear SVM**

# Import the Libraries

import numpy as np

import matplotlib.pyplot as plt

from sklearn import svm, datasets

# Import some Data from the iris Data Set

iris = datasets.load\_iris()

# Take only the first two features of Data.

# To avoid the slicing, Two-Dim Dataset can be used

X = iris.data[:, :2]

y = iris.target

# C is the SVM regularization parameter

C = 1.0

# Create an Instance of SVM and Fit out the data.

# Data is not scaled so as to be able to plot the support vectors

svc = svm.SVC(kernel ='linear', C = 1).fit(X, y)

# create a mesh to plot

x\_min, x\_max = X[:, 0].min() - 1, X[:, 0].max() + 1

y\_min, y\_max = X[:, 1].min() - 1, X[:, 1].max() + 1

h = (x\_max / x\_min)/100

xx, yy = np.meshgrid(np.arange(x\_min, x\_max, h),

np.arange(y\_min, y\_max, h))

# Plot the data for Proper Visual Representation

plt.subplot(1, 1, 1)

# Predict the result by giving Data to the model

Z = svc.predict(np.c\_[xx.ravel(), yy.ravel()])

Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, cmap = plt.cm.Paired, alpha = 0.8)

plt.scatter(X[:, 0], X[:, 1], c = y, cmap = plt.cm.Paired)

plt.xlabel('Sepal length')

plt.ylabel('Sepal width')

plt.xlim(xx.min(), xx.max())

plt.title('SVC with linear kernel')

# Output the Plot

plt.show()

1. **Ada-boost Algorithm**

# Load libraries

from sklearn.ensemble import AdaBoostClassifier

from sklearn import datasets

# Import train\_test\_split function

from sklearn.model\_selection import train\_test\_split

#Import scikit-learn metrics module for accuracy calculation

from sklearn import metrics

# Load data

iris = datasets.load\_iris()

X = iris.data

y = iris.target

# Split dataset into training set and test set

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3) # 70% training and 30% test

# Create adaboost classifer object

abc = AdaBoostClassifier(n\_estimators=50,

learning\_rate=1)

# Train Adaboost Classifer

model = abc.fit(X\_train, y\_train)

#Predict the response for test dataset

y\_pred = model.predict(X\_test)

# Model Accuracy, how often is the classifier correct?

print("Accuracy:",metrics.accuracy\_score(y\_test, y\_pred))