Student ID: 700747817

Name: Manikanta Venkata Kondasai Kalyan Badeti Parameswara

NN & DL - Assignment 1

**Video Link:** <https://drive.google.com/file/d/17hGlr-On_-QMgLb24uXSOm1Q0jgggYs2/view>

GitHub Link: <https://github.com/Kalyansai6/Neural-Network---Assignment-I>

Question 1

*import pandas as pd*

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

*from sklearn.naive\_bayes import GaussianNB*

*from sklearn.metrics import accuracy\_score, classification\_report*

*# Load the glass dataset*

*data = pd.read\_csv('glass.csv')*

*# Split the data into training and testing sets*

*X\_train, X\_test, y\_train, y\_test = train\_test\_split(data.drop('Type', axis=1), data['Type'], test\_size=0.25)*

*# Create a Gaussian Naïve Bayes classifier*

*nb\_classifier = GaussianNB()*

*# Fit the classifier to the training data*

*nb\_classifier.fit(X\_train, y\_train)*

*# Predict the class labels for the test data*

*y\_pred = nb\_classifier.predict(X\_test)*

*# Calculate accuracy score*

*accuracy = accuracy\_score(y\_test, y\_pred)*

*print("Accuracy using Naïve Bayes:", accuracy)*

*A screenshot of a computer code

Description automatically generated*

*# Print the classification report*

*report = classification\_report(y\_test, y\_pred)*

*print("Classification Report for Naïve Bayes:")*

*print(report)*

*A screenshot of a computer

Description automatically generated*

**Comments:**

* Firstly, the libraries were imported such as pandas, sklearn.svm, sklearn.naive\_bayes.
* Then, read the provided dataframe from read() function and printed.
* The dataframe is splitted into train & test.
* Now feature the columns of the dataframe.
* Implement the naïve bayes classifier & train it.
* Now, the naïve bayes classifier is predicted and the accuracy is printed.
* Now, the naïve bayes classifier performance is printed.
* Implement the linear support vector classification & train it.
* Now, the performance and the accuracy are printed.

**Question 2**

*import numpy as np*

*import pandas as pd*

*from sklearn.metrics import classification\_report*

*from sklearn.svm import LinearSVC*

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

*# Loading the glass dataset*

*data = pd.read\_csv('glass.csv')*

*# Splitting the data into training and testing sets*

*X\_train, X\_test, y\_train, y\_test = train\_test\_split(data.drop('Type', axis=1), data['Type'], test\_size=0.25)*

*# Creating a linear SVM classifier*

*clf = LinearSVC()*

*# Fitting the classifier to the training data*

*clf.fit(X\_train, y\_train)*

*# Predicting the class labels for the test data*

*y\_pred = clf.predict(X\_test)*

*# Calculating the accuracy of the model*

*accuracy = clf.score(X\_test, y\_test)*

*print('Accuracy:', accuracy)*

*# Printing the classification report*

*print(classification\_report(y\_test, y\_pred))*

*A screenshot of a computer program

Description automatically generated*

**Comments:**

* Then Support vector classifier (SVC) with the radial basis function kernel (RBF) is trained and predicted.
* Now, the performance and accuracy is printed.

**Justification:**

By comparing both the Naïve Bayes method and the linear SVM method, the accuracy of Naïve Bayes method is more with 59.25%.

This is because as they both are parameter optimization. Naïve bayes treats them as independent, whereas SVM looks at the interactions between them to a certain degree, as long as you’re using a non-linear kernel.

**Question 3**

*import pandas as pd*

*import matplotlib.pyplot as plt*

*from sklearn.linear\_model import LinearRegression*

*from sklearn.metrics import mean\_squared\_error*

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

*# Importing the Salary\_Data.csv file*

*data = pd.read\_csv('Salary\_Data.csv')*

*# Splitting the data into train and test partitions*

*X = data.iloc[:, :-1].values*

*y = data.iloc[:, -1].values*

*X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=1/3, random\_state=42)*

*# Creating a linear regression model*

*regressor = LinearRegression()*

*# Training the model*

*regressor.fit(X\_train, y\_train)*

*# Predicting with the model*

*y\_pred\_train = regressor.predict(X\_train)*

*y\_pred\_test = regressor.predict(X\_test)*

*# Calculating the mean squared error*

*mse\_train = mean\_squared\_error(y\_train, y\_pred\_train)*

*mse\_test = mean\_squared\_error(y\_test, y\_pred\_test)*

*print("Mean Squared Error (Train):", mse\_train)*

*print("Mean Squared Error (Test):", mse\_test)*

*# Visualizing the train and test data using a scatter plot*

*plt.scatter(X\_train, y\_train, color='blue', label='Train Data')*

*plt.scatter(X\_test, y\_test, color='red', label='Test Data')*

*plt.plot(X\_train, y\_pred\_train, color='black', label='Linear Regression')*

*plt.xlabel('Years of Experience')*

*plt.ylabel('Salary')*

*plt.title('Linear Regression - Train and Test Data')*

*plt.legend()*

*plt.show()*

*A graph of a linear graph

Description automatically generated*