SHALAKA FOUNDATION'S

KEYSTONE SCHOOL OF ENGINEERING

Department of Computer Engineering

LABORATORY MANUAL

LABORATORY PRACTICE III

MACHINE LEARNING

SEMESTER-I

Subject Code: 410246

TEACHING SCHEME
Lectures:3Hrs/Week
Practical:2Hrs/Week

EXAMINATION SCHEME
Practical:50Marks
TermWork:50Marks

Name of Faculty:
Dr. Rupali Pawar

Group B: Machine Learning

Any 4 assignments and 1 Mini project are mandatory.

Sr. No.	Title
1	Predict the price of the Uber ride from a given pickup point to theagreed drop-off location. Dataset link: https://www.kaggle.com/datasets/yasserh/uber-fares-dataset
2	Classify the email using the binary classification method. Email Spam detection has two states: a) Normal State – Not Spam, b) Abnormal State – Spam. Use K-Nearest Neighbors and Support Vector Machine for classification. Analyze their performance. Dataset link: https://www.kaggle.com/datasets/balaka18/email-spam- classification-dataset-csv
3	Given a bank customer, build a neural network-based classifier that can determine whether they will leave or not in the next 6 months. Dataset link: https://www.kaggle.com/barelydedicated/bank-customer-churn-modeling
4	Implement Gradient Descent Algorithm to find the local minima of a function. For example, find the local minima of the function $y=(x+3)^2$ starting from the point x=2.
5	Implement K-Nearest Neighbors algorithm on diabetes.csv dataset. Compute confusion matrix, accuracy, error rate, precision and recall on the given dataset. Dataset link: https://www.kaggle.com/datasets/abdallamahgoub/diabetes
6	Implement K-Means clustering/ hierarchical clustering on sales_data_sample.csv dataset. Determine the number of clusters using the elbow method. Dataset link : https://www.kaggle.com/datasets/kyanyoga/sample- sales-data
7	Use the following dataset to analyze ups and downs in the market and predict future stock price returns based on Indian Market data from 2000 to 2020. Dataset Link: https://www.kaggle.com/datasets/sagara9595/stock-data OR Build a machine learning model that predicts the type of people who survived the Titanic shipwreck using passenger data (i.e. name, age, gender, socio-economic class, etc.). Dataset Link: https://www.kaggle.com/competitions/titanic/data

GROUP B:
MACHINE LEARNING
LAB MANUAL

Assignment No.	1 (GROUP B)
Title	Predict the price of the Uber ride from a given pickup point to the agreed drop-off location. Perform following tasks: 1. Pre-process the dataset. 2. Identify outliers. 3. Check the correlation. 4. Implement linear regression and random forest regression models. Evaluate the models and compare their respective scores like R2, RMSE, etc. Dataset link: https://www.kaggle.com/datasets/yasserh/uber-fares-dataset
Subject	Machine Learning
Class	
Roll No.	
Date	
Signature	

Assignment 1

Title: Predict the price of the Uber ride from a given pickup point to the agreed drop-off location.

Problem Statement: Predict the price of the Uber ride from a given pickup point to the agreed drop-off location. Perform following tasks:

1. Pre-process the dataset.
2. Identify outliers.
3. Check the correlation.
4. Implement linear regression and random forest regression models.
5. Evaluate the models and compare their respective scores like R2, RMSE, etc.

Prerequisites: LR and Random Forest Algorithm

Objectives

1.Understand the Dataset & cleanup (if required).

2.Build Regression models to predict the fare price of uber ride.

3.Also evaluate the models & compare their respective scores like R2, RMSE, etc.

Software Used:

Linux/Unix/Ubuntu Operating Systems, Jupyter Notebook

Programming Language: Python 3.7.8

Dataset link: https://www.kaggle.com/datasets/yasserh/uber-fares-dataset

Theory:

The Uber Datasets

We will perform data analysis on two types of rider data from Uber. The first dataset contains information about the rides taken by one particular user, and the second contains similar detailsabout the rides taken by Uber users in two cities.

Uber Personal Data

This Kaggle Uber dataset contains information about 1155 rides of a single Uber user. The features include the trip date, source, destination, distance traveled, and purpose of the trip. This dataset is a good starting point for performing basic EDA. Based on this, we might also be able to generate some insights by relating the data to real-world events and user habits

1. Linear Regression

If you want to start machine learning, linear regression is the best place to start. Linear Regression is a regression model, meaning, it'll take features and predict a continuous output,

eg : stock price, salary etc.

Linear regression as the name says, finds a linear curve solution to every problem.

LR allocates weight parameter, theta for each of the training features. The predicted output (h(θ)) will be a linear function of features and θ coefficients.

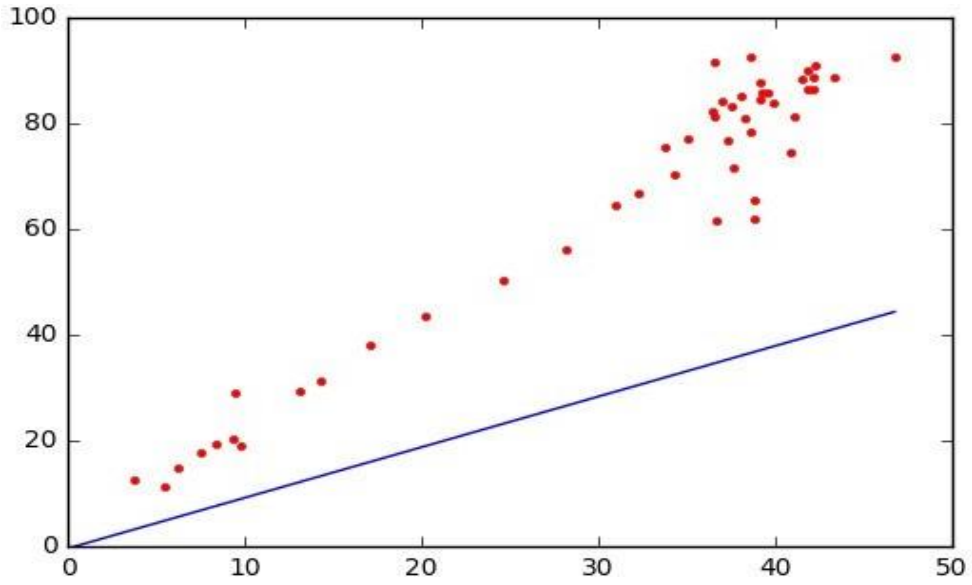
$$h_\theta = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ...$$

linear regression output. Eqn (1) During the start of training, each theta is randomly initialized. But during the training, wecorrect the theta corresponding to each feature such that, the loss (metric of the deviationbetween expected and predicted output) is minimized. Gradient descend algorithm will beused to align the θ values in the right direction.

In the below diagram, each red dots represent the training data and the blue line shows thederived

solution.

Loss function:
In LR, we use mean squared error as the metric of loss. The deviation of expected and actual outputs will be squared and sum up. Derivative of this loss will be used by gradient descend algorithm.



Advantages:

- Easy and simple implementation.
- Space complex solution.
- Fast training.
- Value of θ coefficients gives an assumption of feature significance.

Disadvantages:

- Applicable only if the solution is linear. In many real life scenarios, it may not be thecase.
- Algorithm assumes the input residuals (error) to be normal distributed, but may not besatisfied always.
- Algorithm assumes input features to be mutually independent (no co-linearity).

Assumptions for LR:

- Linear relationship between the independent and dependent variables.
- Training data to be homoscedastic, meaning the variance of the errors should be somewhat constant.
- Independent variables should not be co-linear.

Collinearity & Outliers:

Two features are said to be collinear when one feature can be linearly predicted from the other with somewhat accuracy.

- Collinearity will simply inflate the standard error and causes some significant features tobecome insignificant during training. Ideally, we should calculate the collinearity prior to training and keep only one feature from highly correlated feature sets.

  Outlier is another challenge faced during training. They are data-points that are extreme to normal observations and affects the accuracy of the model.

- Outliers inflate the error functions and affects the curve function and accuracy of the linear regression. Regularization (especially L1) can correct the outliers, by not allowingthe θ parameters to change violently.
- During Exploratory data analysis phase itself, we should take care of outliers and correct/eliminate them. Box-plot can be used for identifying them.

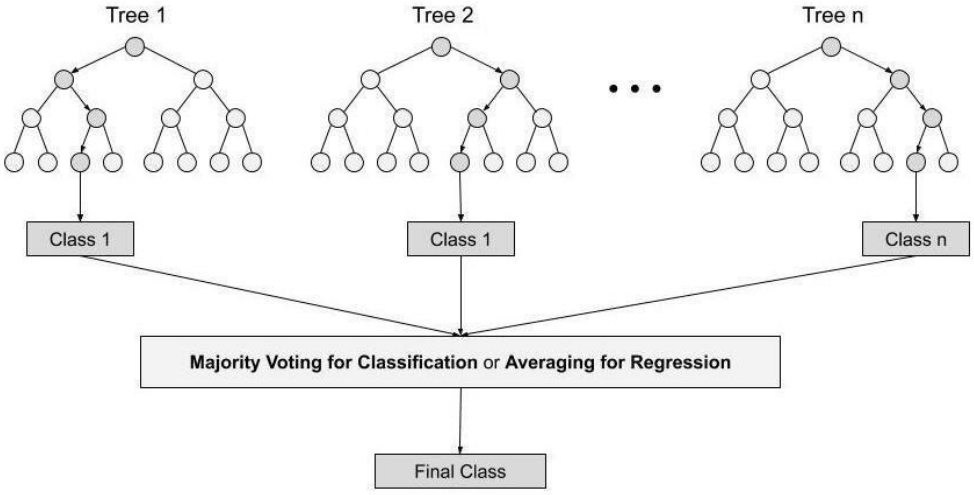
Comparison with other models:

As the linear regression is a regression algorithm, we will compare it with other regression algorithms. One basic difference of linear regression is, LR can only support linearsolutions. There are no best models in machine learning that outperforms all others (no free Lunch), and efficiency is based on the type of training data distribution.

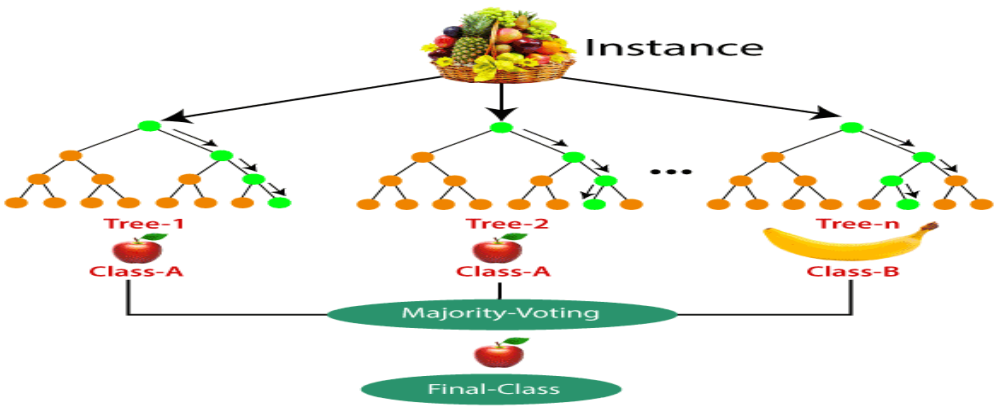
Random Forest:

Algorithm:

Steps involved in random forest algorithm:
Step 1: In Random forest n number of random records are taken from the data set having k number of records.
Step 2: Individual decision trees are constructed for each sample.
Step 3: Each decision tree will generate an output.
Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.



For example: consider the fruit basket as the data as shown in the figure below. Now n number of samples are taken from the fruit basket and an individual decision tree is constructed for each sample. Each decision tree will generate an output as shown in the figure. The final output is considered based on majority voting. In the below figure you can see that the majority decision tree gives output as an apple when compared to a banana, so the final output is taken as an apple.



Important Features of Random Forest

- Diversity- Not all attributes/variables/features are considered while making an individual tree,each tree is different.
- Immune to the curse of dimensionality- Since each tree does not consider all the features, thefeature space is reduced.
- Parallelization-Each tree is created independently out of different data and attributes. This meansthat we can make full use of the CPU to build random forests.
- Train-Test split- In a random forest we don't have to segregate the data for train and test as therewill always be 30% of the data which is not seen by the decision tree.
- Stability- Stability arises because the result is based on majority voting/ averaging.

In this Assignment, we're looking to predict the fare for their future transactional cases. Uber delivers service to lakhs of customers daily. Now it becomes really important to manage their data properly to come up with new business ideas to get best results. Eventually, it becomes really important to estimate the fare prices accurately

We aim to solve the problem statement by creating a plan of action, Here are some of the necessarysteps:

- Import the required Libraries
- Pre-process the dataset.
- Identify outliers.
- Check the correlation.
- Haversine distance calculation
- Scaling of Features
- Dividing dataset into train and Test
- Implement linear regression and random forest regression models.
- Evaluate the models and compare their respective scores like R2, RMSE, MSE,etc

Dataset link: https://www.kaggle.com/datasets/yasserh/uber-fares-dataset

Performance Metrics for Regression Problems

Here, we are going to discuss various performance metrics that can be used to evaluate predictions for regression problems. If one metric is perfect, there is no need for multiple metrics. To understand the benefits and disadvantages of Evaluation metrics because different evaluation metric fits on a different set of a dataset.

1) Mean Absolute Error(MAE)

MAE is a very simple metric which calculates the absolute difference between actual and predicted values.

```
from sklearn.metrics import mean_absolute_error
print("MAE",mean_absolute_error(y_test,y_pred))
```

2) Mean Squared Error (MSE)

MSE is a most used and very simple metric with a little bit of change in mean absolute error. Mean squared error states that finding the squared difference between actual and predicted value.

```
from sklearn.metrics import mean_squared_error
print("MSE",mean_squared_error(y_test,y_pred))
```

3) Root Mean Squared Error (RMSE)

As RMSE is clear by the name itself, that it is a simple square root of mean squared error

```
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
```

4) R Squared (R2)

R2 score is a metric that tells the performance of your model

```
from sklearn.metrics import r2_score r2 =
r2_score(y_test,y_pred) print(r2)
```

Facilities:

Google Colab, Jupiter notebook

Input: https://www.kaggle.com/datasets/yasserh/uber-fares-dataset uber dataset file in .csv format

Output:

Confusion Matrix showing Final Results of accuracy comparison

Conclusion:

We have implemented and compared performance of linear regression and randomforest algorithms for predicting the price of the Uber ride from a given pickup point to the agreed drop-off location.

Questions:

i. Compare the accuracy for Random Forest and Linear Regression
ii. Justify is the accuracy difference between both the models.
iii. What is Haversine distance calculation, why is it required
iv. What is Rsquared, MSE, RMSE, MAE, Adjusted Rsquare, Write their formulas

Assignment No.	2 (GROUP B)
Title	Classify the email using the binary classification method. Email Spam detection has two states: a) Normal State – Not Spam, b) Abnormal State – Spam. Use K-Nearest Neighbors and Support Vector Machine for classification. Analyze their performance.
Subject	Machine Learning
Class	
Roll No.	
Date	
Signature	

Assignment No: 2

Title: Classify the email using the binary classification method. Email Spam detection has two states: a) Normal State – Not Spam, b) Abnormal State – Spam. Use K-Nearest Neighbors and Support Vector Machine for classification. Analyze their performance.

Problem Statement: Develop a Python program to implement Classify the email using the binary classification method.

Prerequisites: SVM, K-Nearest algorithm

Objectives:

Understand the Dataset & cleanup.

Build models to predict the class as spam/not spam
Also evaluate the models & compare their respective scores like R2, RMSE, etc.

Software Used:

Linux/Unix/Ubuntu Operating Systems, Jupyter Notebook
Programming Language: Python 3.7.8
Dataset link: https://www.kaggle.com/datasets/balaka18/email-spam-classification-dataset-csv

Theory:

- K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suitecategory by using K-NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is usedfor the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an actionon the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.
- Example: Suppose, we have an image of a creature that looks similar to cat and dog, but wewant to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it willput it in either cat or dog category

Working of KNN Algorithm

Step-1: Select the number K of the neighbors

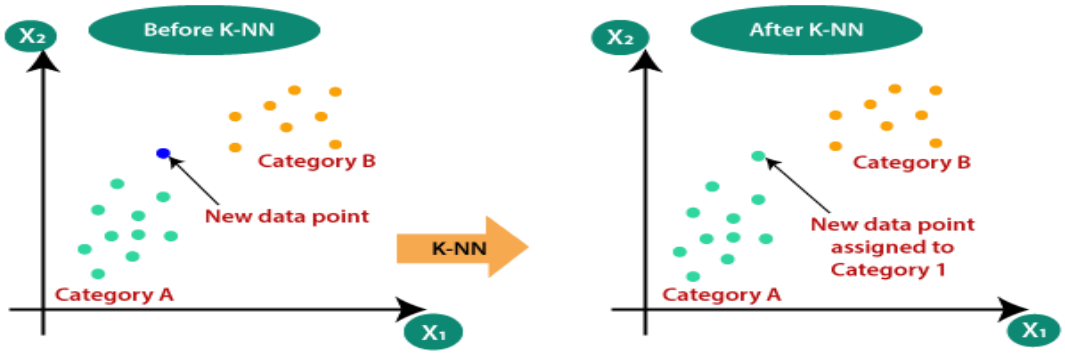
Step-2: Calculate the Euclidean distance of K number of neighbors
Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

Step-4: Among these k neighbors, count the number of the data points in each category.
Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.

Why do we need a K-NN Algorithm?

- Suppose there are two categories, i.e., Category A and Category B, and we have a new datapoint x1, so this data point will lie in which of these categories. To solve this type of problem,we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:



Implementation in Python

As we know K-nearest neighbors (KNN) algorithm can be used for both classification as well as regression. The following are the recipes in Python to use KNN as classifier as well as regressor −

KNN as Classifier

1. Start with importing necessary python packages.
2. Download the dataset from its web link
3. Assign column names to the dataset
4. Read dataset to pandas data frame
5. Data Preprocessing will be done.
6. Data scaling will be done
7. Divide the data into train and test split. Following code will split the dataset into 70% training data and 30% of testing data
8. Train the model with the help of K Neighbors Classifier class of sklearn
9. At last we need to make prediction
10. Print the results using confusion matrix, Classification report and accuracy.

Pros and Cons of KNN

Pros

- It is very simple algorithm to understand and interpret.
- It is very useful for nonlinear data because there is no assumption about data in thisalgorithm.
- It is a versatile algorithm as we can use it for classification as well as regression.
- It has relatively high accuracy but there are much better supervised learning models than KNN.

Cons

- It is computationally a bit expensive algorithm because it stores all the training data.
- High memory storage required as compared to other supervised learning algorithms.
- Prediction is slow in case of big N.
- It is very sensitive to the scale of data as well as irrelevant features

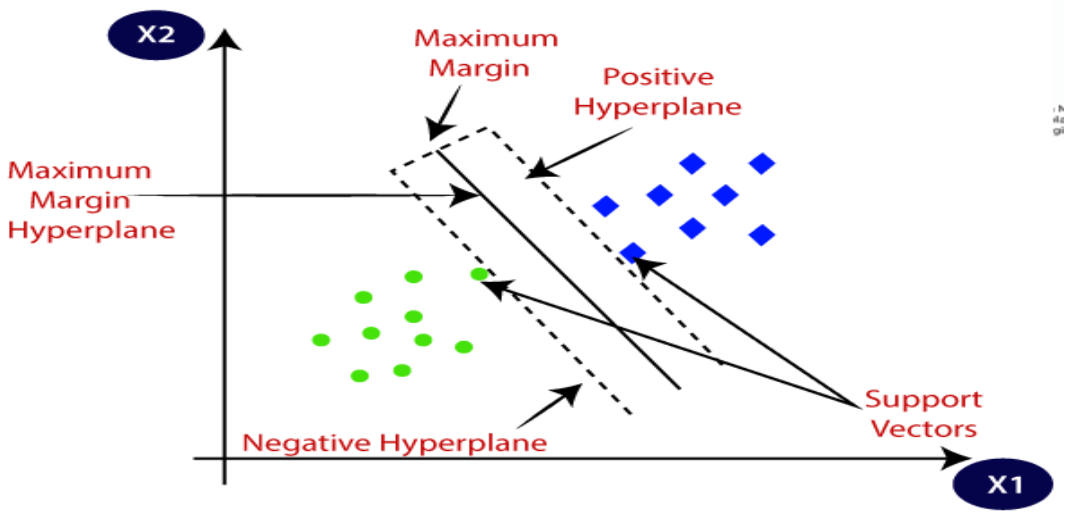
Introduction to SVM

Support vector machines (SVMs) are powerful yet flexible supervised machine learning algorithms which are used both for classification and regression. But generally, they are usedin classification problems. In 1960s, SVMs were first introduced but later they got refined in1990. SVMs have their unique way of implementation as compared to other machine learning algorithms. Lately, they are extremely popular because of their ability to handle multiple continuous and categorical variables.

Working of SVM

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, whichis used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct categoryin the future. This best decision boundary is called a hyperplane.



The followings are important concepts in SVM

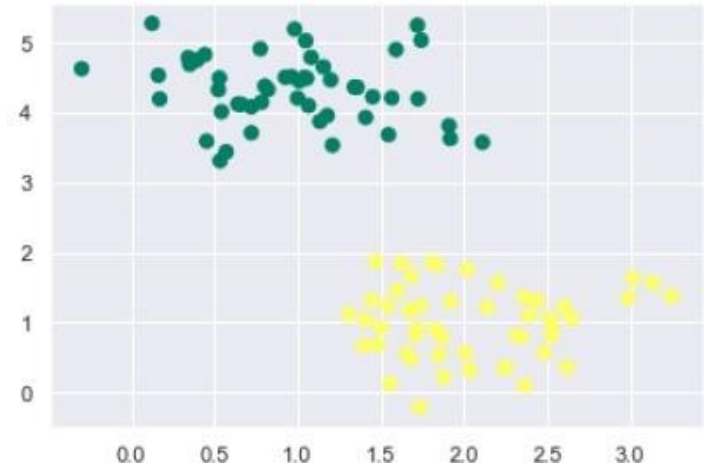
Support Vectors − Data points that are closest to the hyperplane is called support vectors. Separating line will be defined with the help of these data points.

Hyperplane − As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes.

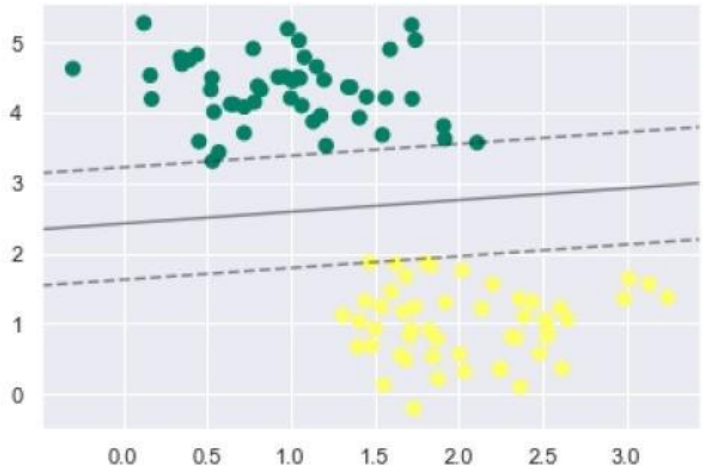
Margin − It may be defined as the gap between two lines on the closet data points of different classes. It can be calculated as the perpendicular distance from the line to the support vectors. Large margin is considered as a good margin and small margin is consideredas a bad margin.

The main goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH) and it can be done in the following two steps −

First, SVM will generate hyperplanes iteratively that segregates the classes in best way.



Then, it will choose the hyperplane that separates the classes correctly.



SVM Kernels

In practice, SVM algorithm is implemented with kernel that transforms an input data space into the required form. SVM uses a technique called the kernel trick in which kernel takes a low dimensional input space and transforms it into a higher dimensional space. In simple words, kernel converts non-separable problems into separable problems by adding more dimensions to it. It makes SVM more powerful, flexible and accurate.

The following are some of the types of kernels used by SVM −

Linear Kernel

It can be used as a dot product between any two observations. The formula of linear kernel isas below −

K (x, xi) = sum(x*xi)

From the above formula, we can see that the product between two vectors say x & xi is thesum of the multiplication of each pair of input values

Polynomial Kernel

It is more generalized form of linear kernel and distinguish curved or nonlinear input space.Following is the formula for polynomial kernel −

K(x, xi) = 1 + sum(x * xi) ^d

Here d is the degree of polynomial, which we need to specify manually in the learningalgorithm.

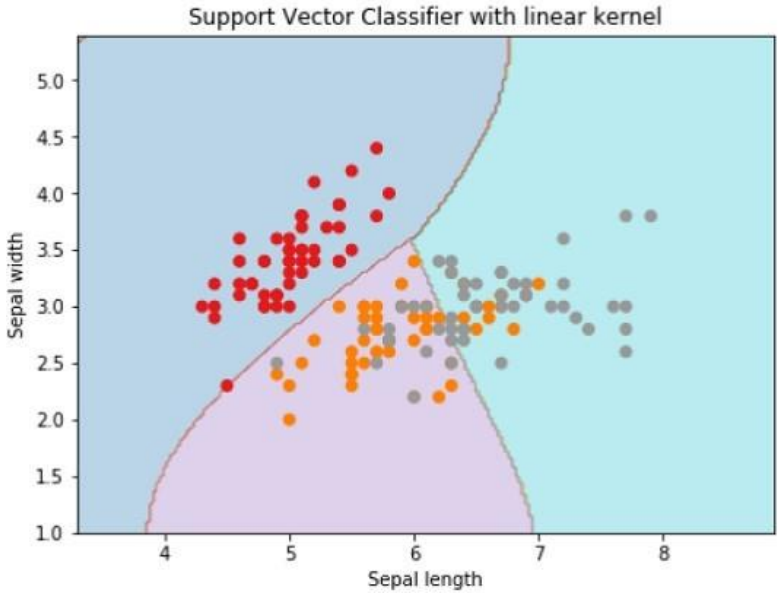
Radial Basis Function (RBF) Kernel

RBF kernel, mostly used in SVM classification, maps input space in indefinite dimensionalspace. Following formula explains it mathematically −

K(x,xi) = exp(-gamma * sum((x – xi^2))

Here, gamma ranges from 0 to 1. We need to manually specify it in the learning algorithm. Agood default value of gamma is 0.1.

As we implemented SVM for linearly separable data, we can implement it in Python for thedata that is not linearly separable. It can be done by using kernels.



Implementing SVM in Python

1. For implementing SVM in Python we will start with the standard libraries import.
2. Next, we are download a spam dataset from kaggle, having linearly separable data.
3. We know that SVM supports discriminative classification. It divides the classes from each other by simply finding a line in case of two dimensions or manifold in case of multiple dimensions. It is implemented on the given dataset.
4. As discussed, the main goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH) hence rather than drawing a zero line between classes we can draw around each line a margin of some width up to the nearest point.
5. Next, we will use Scikit-Learn's support vector classifier to train an SVM model on thisdata.
6. Now, for a better understanding, plot the decision functions for 2D SVC
7. For evaluating model, we need to create grid.
8. Next, we need to plot decision boundaries and margins.
9. Now, similarly plot the support vectors.
10. Now, use SVM function to fit our models.

Pros of SVM classifiers

SVM classifiers offers great accuracy and work well with high dimensional space. SVMclassifiers basically use a subset of training points hence in result uses very less memory.

Cons of SVM classifiers

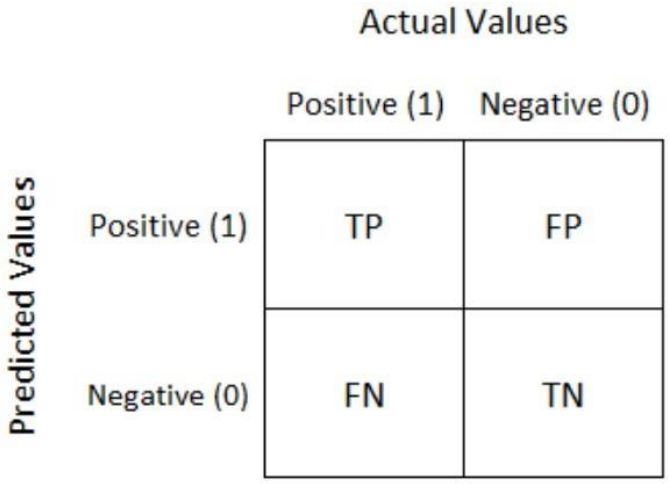
They have high training time hence in practice not suitable for large datasets. Another disadvantage is that SVM classifiers do not work well with overlapping classes.

Performance Metrics for Classification Problems

Here, we are going to discuss various performance metrics that can be used to evaluatepredictions for classification problems.

Confusion Matrix

Confusion Matrix is a performance measurement for the machine learning classification problems where the output can be two or more classes. It is a table with combinations of predicted and actual values.
A confusion matrix is defined as the table that is often used to describe the performance of a classification model on a set of the test data for which the true values are known.



It is extremely useful for measuring the Recall, Precision, Accuracy, and AUC-ROC curves.
True Positive: We predicted positive and it's true.
True Negative: We predicted negative and it's true.
False Positive (Type 1 Error) - We predicted positive and it's false.
False Negative (Type 2 Error) - We predicted negative and it's false.

We can use confusion_matrix function of sklearn.metrics to compute Confusion Matrix ofour classification model.

Accuracy

Accuracy simply measures how often the classifier correctly predicts. We can define accuracy as the ratio of the number of correct predictions and the total number of predictions.

We can use accuracy_score function of sklearn.metrics to compute accuracy of our classification model.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Classification Report

This report consists of the scores of Precisions, Recall, F1 and Support. They are explainedas follows −

1. **Precision—** precision explains how many of the correctly predicted cases actually turnedout to be positive. Precision is useful in the cases where False Positive is a higher concern than False Negatives.

The importance of Precision is in music or video recommendation systems, e-commerce websites, etc. where wrong results could lead to customer churn and this could be harmful to the business.

Precision for a label is defined as the number of true positives divided by the number ofpredicted positives.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

2. Recall (Sensitivity)—Recall explains how many of the actual positive cases we were ableto predict correctly with our model. It is a useful metric in cases where False Negative is of higher concern than False Positive. It is important in medical cases where it doesn't matter whether we raise a false alarm but the actual positive cases should not go undetected.

Recall for a label is defined as the number of true positives divided by the total numberof actual positives.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

3. F1 Score— it gives a combined idea about Precision and Recall metrics. It is maximum when Precision is equal to Recall.

F1 Score is the harmonic mean of precision and recall.

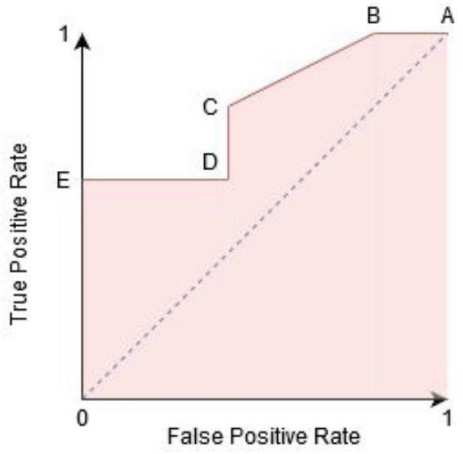
$$F1 = 2.\frac{Precision \times Recall}{Precision + Recall}$$

The F1 score punishes extreme values more. F1 Score could be an effective evaluation metricin the following cases:

- When FP and FN are equally costly.
- Adding more data doesn't effectively change the outcome
- True Negative is high

4. AUC-ROC—The Receiver Operator Characteristic (ROC) is a probability curve that plots the TPR (True Positive Rate) against the FPR (False Positive Rate) at various thresholdvalues and separates the 'signal' from the 'noise'.

The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes. From the graph, we simply say the area of the curve ABDE and the X and Y-axis. From the graph shown below, the greater the AUC, the better is the performance of the model at different threshold points between positive and negative classes. This simply meansthat When AUC is equal to 1, the classifier is able to perfectly distinguish between all Positive and Negative class points. When AUC is equal to 0, the classifier would bepredicting all Negatives as Positives and vice versa. When AUC is 0.5, the classifier is not able to distinguish between the Positive and Negative classes.



Steps to implement the K-NN /SVM algorithm:

1. Start with importing necessary python packages.
2. Download the dataset from its web link
3. Assign column names to the dataset
4. Read dataset to pandas data frame
5. Data Preprocessing will be done
6. Data scaling will be done
7. Divide the data into train and test split. Following code will split the dataset into 70% training data and 30% of testing data
8. Train the model with the help of K Neighbors Classifier// SVC class of sklearn
9. At last we need to make prediction
10. Print the results using confusion matrix, Classification report and accuracy.

Facilities:

Google colab, Jupiter notebook

Input:*https://www.kaggle.com/datasets/balaka18/email-spam-classification-dataset-csv Email dataset in csv File Format*

Output:

1)Data preprocessing visuals if any
2)Confusion matrix of accuracy

Conclusion: Thus we have studied KNN and SVM algorithm successfully

Questions:

1.What do you mean by KNN algorithm?
2.What do you mean by SVM.
3.Compare both the algorithms and justify the accuracy?

Assignment No.	3
Title	Implement Gradient Descent Algorithm to find the local minima of a function
Subject	Machine Learning Laboratory Practice-III
Class	
Roll No.	
Date	
Signature	

Title: Implement Gradient Descent Algorithm.

Problem Statement: Implement Gradient Descent Algorithm to find the local minima of a function. For example, find the local minima of the function $y=(x+3)^2$ starting from the point x=2.

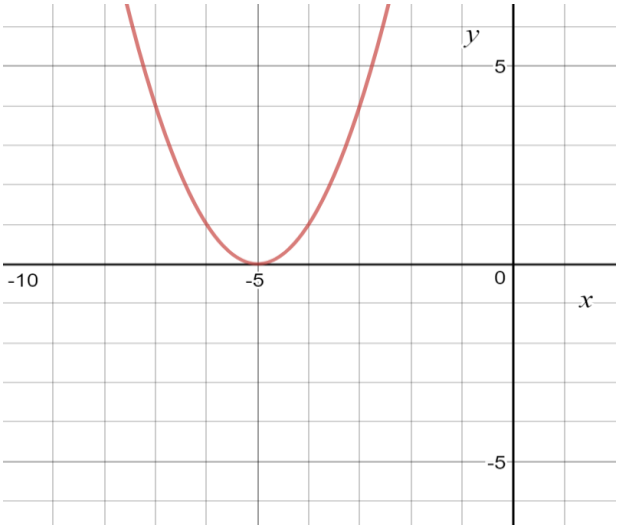
Prerequisites: Cost Function, Gradient Descent Algorithm

Objectives: Implement Gradient Descent Algorithm to find the local minima of a function

Theory:

It is an optimization algorithm to find the minimum of a function. We start with a random point on the function and move in the negative direction of the gradient of the function to reach the local/global minima.

To find the local minima of the function $y=(x+5)^2$ starting from the point x=3



We know the answer just by looking at the graph. $y = (x+5)^2$ reaches it's minimum valuewhen x = -5 (i.e when x=-5, y=0). Hence x=-5 is the local and global minima of the function.

Now, let's see how to obtain the same numerically using gradient descent

Step 1 : Initialize x =3. Then, find the gradient of the function, dy/dx = 2*(x+5).
Step 2 : Move in the direction of the negative of the gradient, we require a learning rate. Let us assume the **learning rate → 0.01**
Step 3 : Let's perform 2 iterations of gradient descent
Step 4 : We can observe that the X value is slowly decreasing and should converge to -5 (the local minima).
A precision variable in our algorithm which calculates the difference between two consecutive "x" values . If the difference between x values from 2 consecutive iterations is lesser than the precision we set, stop the algorithm.

Initialize Parameters :

$$X_0 = 3$$

$$Learning\ rate = 0.01$$

$$\frac{dy}{dx} = \frac{d}{dx}(x+5)^2 = 2 * (x+5)$$

Iteration 1 :

$$X_1 = X_0 - (learning\ rate) * (\frac{dy}{dx})$$

$$X_1 = 3 - (0.01) * \big(2 * (3+5)\big) = 2.84$$

Iteration 2 :

$$X_2 = X_1 - (learning\ rate) * (\frac{dy}{dx})$$

$$X_2 = 2.84 - (0.01) * \big(2 * (2.84+5)\big) = 2.6832$$

Gradient descent in Python：

Step 1 : Initialize parameters

```
cur_x = 3 # The algorithm starts at x=3
rate = 0.01 # Learning rate
precision = 0.000001 #This tells us when to stop the algorithm
previous_step_size = 1 #
max_iters = 10000 # maximum number of iterations
iters = 0 #iteration counter
df = lambda x: 2*(x+5) #Gradient of our function
```

Step 2 : Run a loop to perform gradient descent

Stop loop when difference between x values from 2 consecutive iterations is less than 0.000001or when number of iterations exceeds 10,000

```
while previous_step_size > precision and iters < max_iters:
prev_x = cur_x #Store current x value in prev_x
cur_x = cur_x - rate * df(prev_x) #Grad descent
previous_step_size = abs(cur_x - prev_x) #Change in x
iters = iters+1 #iteration count
print("Iteration",iters,"\nX value is",cur_x) #Print iterations
print("The local minimum occurs at", cur_x)
```

Facilities:

Google Colab, Jupiter notebook

Conclusion:

Hence, we have successfully studied implementation of Gradient Descent Algorithm.

Questions:

1. Find the local minima of the function $y=(x+3)^2$ starting from the point x=2.
2. What is idea behind Gradient Descent?
3. What is Local and Global Minima?
4. What are types of Gradient Descent?

Assignment No.	4 (GROUP B)
Title	Given a bank customer, build a neural network-based classifier that can determine whether they will leave or not in the next 6 months.
Subject	Machine Learning
Class	BE (Comp)
Roll No.	
Date	
Signature	

Assignment No: 4

Title: Given a bank customer, build a neural network-based classifier that can determinewhether they will leave or not in the next 6 months.

Problem Statement: Develop a Python program to implement and build a neural network-based classifier that can determine whether from the given churn data a bank customer will leave the bank or not in the next 6 months.

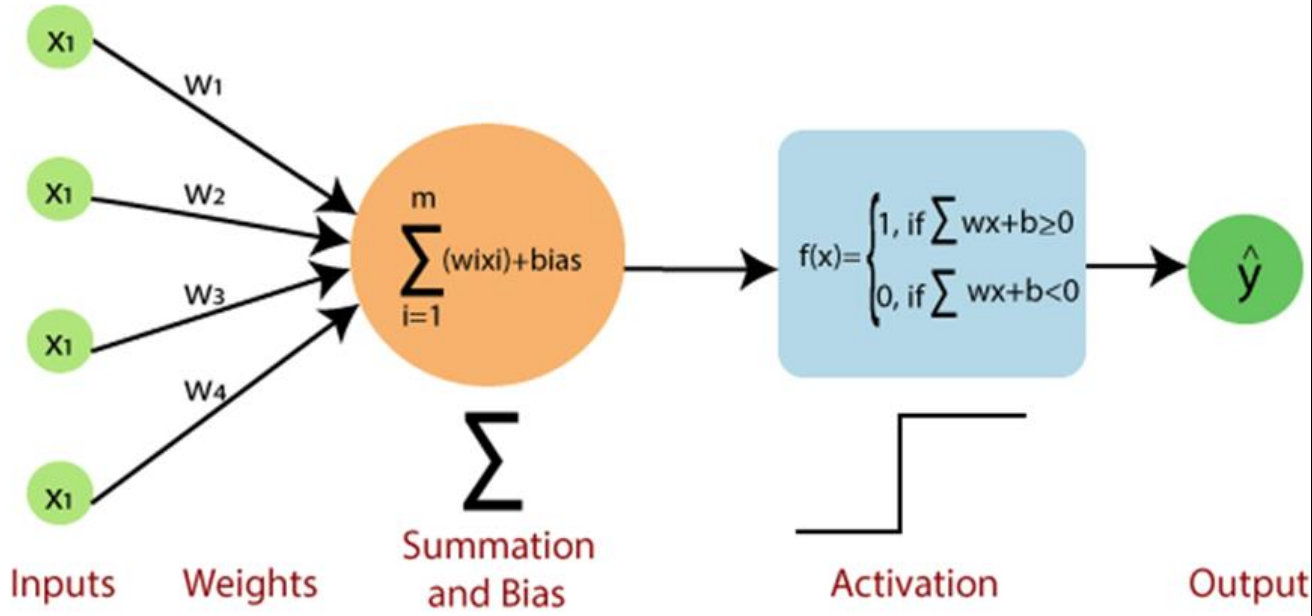
https://www.kaggle.com/barelydedicated/bank-customer-churn-modeling
Perform following steps:
1. Read the dataset.
2. Distinguish the feature and target set and divide the data set into training and test sets.
3. Normalize the train and test data.
4. Initialize and build the model. Identify the points of improvement and implement the same.
5. Print the accuracy score and confusion matrix

Prerequisites: Neural Networks

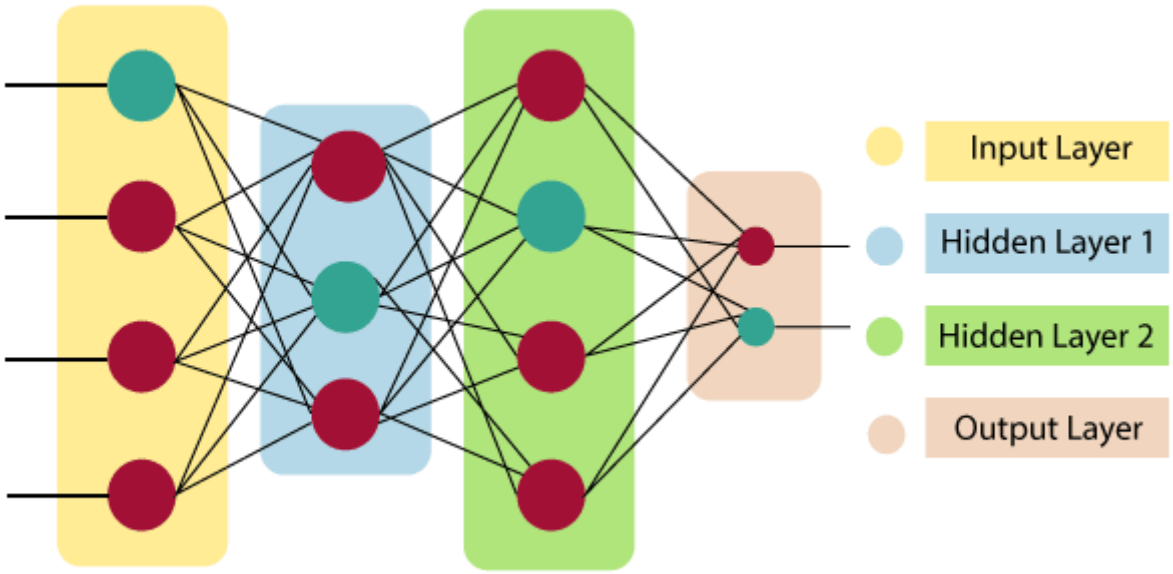
Objectives:
Understand the Dataset & cleanup (if required).
Build models to predict whether employee will leave or not
Identify the points of improvement and implement the same

Theory

Neural Network:
Artificial Neural Network is biologically inspired by the neural network, which constitutes after the human brain. Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks. These neurons are known as nodes.
Neural networks are modeled in accordance with the human brain so as to imitate their functionality. The human brain can be defined as a neural network that is made up of several neurons, so is the Artificial Neural Network is made of numerous perceptron.
A simple model of a biological neuron in an artificial neural network is known as Perceptron. Perceptron is a ML algorithm for supervised learning used to build ANN



Artificial Neural Network primarily consists of three layers:



Input Layer:

The input layer of a perceptron is made of artificial input neurons that bring the initial data into the system for further processing

Hidden Layer:

The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns.

Output Layer:

The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.

The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function. $\sum_{i=1}^{n} Wi * Xi + Wo$

The weighted total, is passed as an input to an activation function to produce the output. The different activation functions are sigmoid, tanh, ReLU etc.

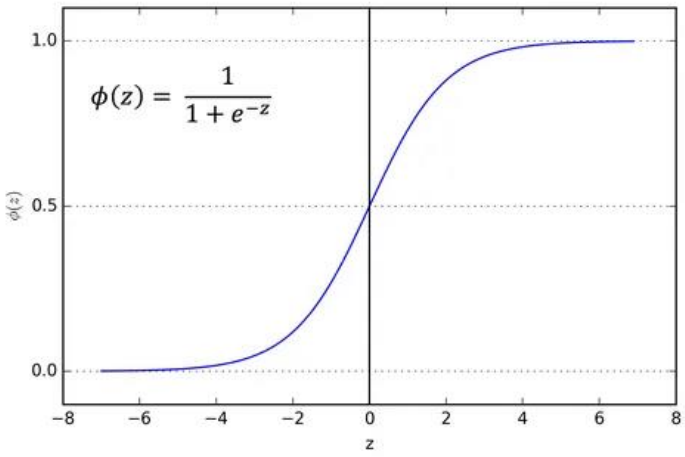
The two main categories of activation functions are:

- Linear Activation Function
- Non-linear Activation Functions

The Nonlinear Activation Functions are mainly divided on the basis of their range or curves-

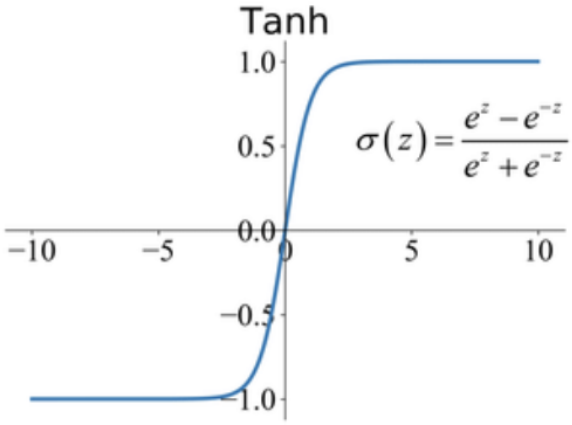
1. Sigmoid or Logistic Activation Function

The Sigmoid Function curve looks like a S-shape.



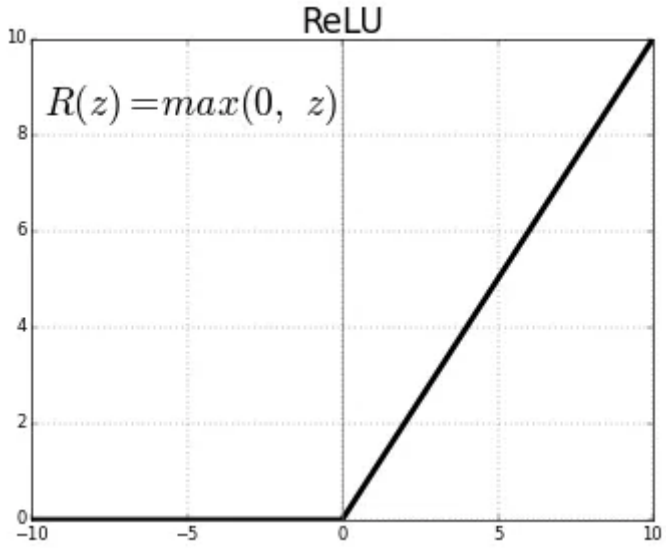
2. **tanh or hyperbolic tangent Activation Function**

tanh is also like logistic sigmoid but better. The range of the tanh function is from (-1 to 1). tanh is also sigmoidal (s - shaped).



3. ReLU

The Rectified Linear Unit (ReLU) is the most used activation function in all the convolutional neural networks or deep learning models. The function returns 0 if it receives any negative input, but for any positive value x it returns that value back.



Classification:

Classification is a supervised machine learning method where the model tries to predict the correct label of a given input data. In classification, the model is fully trained using the training data, and then it is evaluated on test data before being used to perform prediction on new unseen data

For instance, an algorithm can learn to predict whether a bank customer will leave in six months or not)

Types of Classification:

Binary Classification:

In a binary classification task, the goal is to classify the input data into two mutually exclusive categories. The training data in such a situation is labeled in a binary format: true and false; positive and negative; 0 and 1; spam and not spam, etc. depending on the problem being tackled. For instance, we might want to detect whether a given image is a truck or a boat.

Logistic Regression and Support Vector Machines algorithms are natively designed for binary classifications. However, other algorithms such as K-Nearest Neighbors and Decision Trees can also be used for binary classification.

Multi-Class Classification:

The multi-class classification, has at least two mutually exclusive class labels, where the goal is to predict to which class a given input example belongs to. For instance, we might want to detect whether a given image is a truck, a plane or a boat. Most of the binary classification algorithms can be also used for multi-class classification. These algorithms include but are not limited to: Random Forest, Naive Bayes, K-Nearest Neighbors, Gradient Boost.

One-versus-one: this strategy trains as many classifiers as there are pairs of labels. If we have a 3-class classification, we will have three pairs of labels, thus three classifiers. In general, for N labels, we will have Nx(N-1)/2 classifiers. Each classifier is trained on a single binary dataset, and the final class is predicted by a majority vote between all the classifiers. One-vs-one approach works best for SVM and other kernel-based algorithms.

Consider a multi-class classification problem with four classes: '*red*,' '*blue*,' and '*green*,' '*yellow*.'

Binary Classification Problem 1: red vs. blue
Binary Classification Problem 2: red vs. green
Binary Classification Problem 3: red vs. yellow
Binary Classification Problem 4: blue vs. green
Binary Classification Problem 5: blue vs. yellow
Binary Classification Problem 6: green vs. yellow

One-versus-rest: at this stage, we start by considering each label as an independent label and consider the rest combined as only one label. With 3-classes, we will have three classifiers. In general, for N labels, we will have **N** binary classifiers. For Classifying Banana, Orange and apples using OVR or OVA, we get

Problem 1 : Banana vs [Orange, Apple]
Problem 2 : Orange vs [Banana, Apple]
Problem 3 : Apple vs [Banana, Orange]

Algorithm:

1. Import the necessary python libraries and Read the dataset.
2. Distinguish the feature as input and output dataset after preprocessing
3. Normalize the data.
4. Divide the data set into training and test sets.
5. Train and build the model.
6. Print the accuracy score and confusion matrix
7. Identify the points of improvement (balanced or imbalanced dataset)
8. Implement Oversampling or Under sampling to Balance the dataset
9. Repeat step 5 and 6
10. Compare the precision, recall, F1score etc values of balanced and imbalanced dataset

Facilities:

GoogleColab, Jupiter notebook

Input: https://www.kaggle.com/barelydedicated/bank-customer-churn-modeling .csv file containing bank employee data

Output:

1) Data preprocessing visuals if any2) Confusion matrix of accuracy

Conclusion: Thus we have studied and implemented and churn analysis of customer data and implemented the same successfully.

Questions:

1. Explain neural Networks
2. What is Balanced and imbalanced data?
3. What are the methods to balance the data?
4. What is Macro and Micro precision, explain with an example.

Assignment No.	5
Title	Implement K-Nearest Neighbors algorithm
Subject	Laboratory Practice-III
Class	B.E. (C.E.)
Roll No.	
Date	
Signature	

Assignment No: 5

Title: Implement K-Nearest Neighbors algorithm

Problem Statement: Develop a Python program to implement K-Nearest Neighbors algorithm

Prerequisites: KNN

Objectives: Implement K-Nearest Neighbors algorithm on diabetes.csv dataset. Compute confusion matrix, accuracy, error rate, precision and recall on the given dataset.

Outcome: To understand the implementation of K-Nearest Neighbors algorithm and analyze andfind its steps.

Theory

k Nearest Neighbors algorithm

In machine learning, k Nearest Neighbors or kNN is the simplest of all machine learning algorithms. It is a non-parametric algorithm used for classification and regression tasks. Non- parametric means there is no assumption required for data distribution. So, kNN does not require any underlying assumption to be made. In both classification and regression tasks, the input consists of the k closest training examples in the feature space. The output depends upon whether kNN is used for classification or regression purposes.

- In kNN classification, the output is a class membership. The given data point is classified based on the majority of type of its neighbors. The data point is assigned to the most frequent class among its k nearest neighbors. Usually k is a small positive integer. If k=1, then the data point is simply assigned to the class of that single nearest neighbor.
- In kNN regression, the output is simply some property value for the object. This value is the average of the values of k nearest neighbors.

kNN is a type of instance-based learning or lazy learning. Lazy learning means it does not require any training data points for model generation. All training data will be used in the testing phase. This makes training faster and testing slower and costlier. So, the testing phase requires more time and memory resources.

In kNN, the neighbors are taken from a set of objects for which the class or the object property value is known. This can be thought of as the training set for the kNN algorithm, though no explicittraining step is required. In both classification and regression kNN algorithm, we can assign weightto the contributions of the neighbors. So, nearest neighbors contribute more to the average thanthe more distant ones.

k Nearest Neighbours intuition

The kNN algorithm intuition is very simple to understand. It simply calculates the distance betweena sample data point and all the other training data points. The distance can be Euclidean distance or Manhattan distance. Then, it selects the k nearest data points where k can be any integer. Finally,it assigns the sample data point to the class to which the majority of the k data points belong.

Now, we will see kNN algorithm in action. Suppose, we have a dataset with two variables which are classified as Red and Blue.

In kNN algorithm, k is the number of nearest neighbors. Generally, k is an odd number because it helps to decide the majority of the class. When k=1, then the algorithm is known as the nearest neighbor algorithm.

Now, we want to classify a new data point X into Blue class or Red class. Suppose the value of k is 3. The kNN algorithm starts by calculating the distance between X and all the other data points.It then finds the 3 nearest points with least distance to point X.

In the final step of the kNN algorithm, we assign the new data point X to the majority of the classof the 3 nearest points. If 2 of the 3 nearest points belong to the class Red while 1 belong to the class Blue, then we classify the new data point as Red.

How to decide the number of neighbors in kNN?

While building the kNN classifier model, one question that come to my mind is what should be thevalue of nearest neighbors (k) that yields highest accuracy. This is a very important question because the classification accuracy depends upon our choice of k.

The number of neighbors (k) in kNN is a parameter that we need to select at the time of model building. Selecting the optimal value of k in kNN is the most critical problem. A small value of k means that noise will have higher influence on the result. So, probability of overfitting is very high.A large value of k makes it computationally expensive in terms of time to build the kNN model. Also, a large value of k will have a smoother decision boundary which means lower variance but higher bias.

Algorithm:

1. Import the necessary python libraries and Read the dataset.
2. Distinguish the feature as input and output dataset after preprocessing
3. Normalize the data.
4. Divide the data set into training and test sets.
5. Find the value of K giving least error
6. Train and build the model using KNN.
7. Print the accuracy score and confusion matrix
8. Compare the precision, recall, F1score etc values for the dataset

Classification Report

Report includes Precision, Recall and F1-Score.

Precision Score

TP – True PositivesFP – False Positives

Precision – Accuracy of positive
predictions.Precision = TP/(TP + FP)

Recall Score

FN – False Negatives
Recall(sensitivity or true positive rate): Fraction of positives that were correctly identified.
Recall = TP/(TP+FN)

F1 Score

F1 Score (aka F-Score or F-Measure) – A helpful metric for comparing two classifiers.F1 Score takes into account precision and the recall.

It is created by finding the harmonic mean of precision and recall.

F1 = 2 x (precision x recall)/(precision + recall)

Precision - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all passengers that labeled as survived, how many actually survived? High precision relates to the low false positive rate. We have got 0.788 precision which is pretty good.

Precision = TP/TP+FP

Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. The question recall answers is: Of all the passengers that truly survived, how many did we label? A recall greater than 0.5 is good.

Recall = TP/TP+FN

F1 score - F1 Score is the weighted average of Precision and Recall. Therefore, this score takesboth false positives and false negatives into account. Intuitively it is not as easy to understand asaccuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precisionand Recall.

F1 Score = 2*(Recall* Precision) / (Recall + Precision)

Accuracy

ROC (Receiver Operating Characteristic) Curve tells us about how good the model can distinguish between two outcomes (e.g If a patient has a disease or no). Better models can accurately distinguish between the two. Whereas, a poor model will have difficulties in distinguishing between the two

Facilities:

Google Colab, Jupiter notebook

Conclusion: Hence, we have successfully studied implementation of K-NN Algorithm

Questions:

1. When do we use K-NN algorithm?
2. Would you use K-NN for large datasets?
3. What are advantages and disadvantages of K-NN algorithm?
4. Explain some cases where K-NN clustering fails to give good results