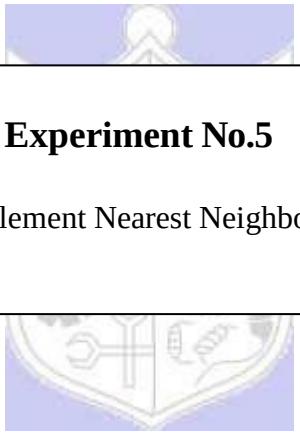


Experiment No.5

Title: Design and implement Nearest Neighbor algorithm.

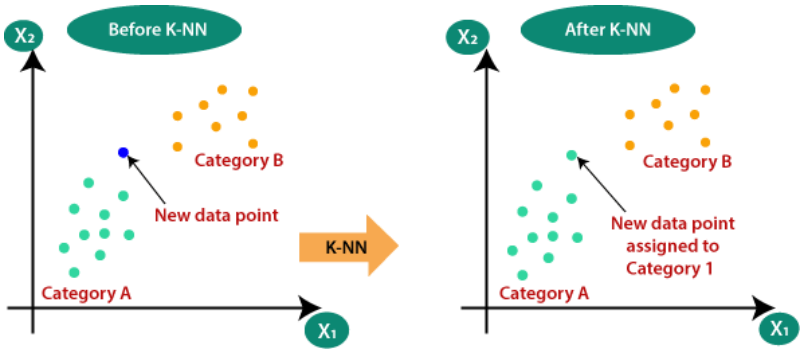
Batch: B1 **Roll No.: 1914078** **Experiment No.:5**

Aim: Design and implement Nearest Neighbor algorithm.

Resources needed: Python 3.6 onwards, RapidMiner

Theory:

- K-Nearest Neighbour 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 suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for 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 action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, and then it classifies that data into a category that is much similar to the new data.



Results: (Program printout with output / Document printout as per the format)

1. Implementation using Python:

Dataset used is Milk Quality Grading dataset:

```
[19] import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     import math
```

```
[7]  df = pd.read_csv('/content/milknew.csv')
     df.head()
```

	pH	Temprature	Taste	Odor	Fat	Turbidity	Colour	Grade
0	6.6	35	1	0	1	0	254	high
1	6.6	36	0	1	0	1	253	high
2	8.5	70	1	1	1	1	246	low
3	9.5	34	1	1	0	1	255	low
4	6.6	37	0	0	0	0	255	medium

Label encoding the target variable:

```
[11] from sklearn import preprocessing
     le = preprocessing.LabelEncoder()
     df.Grade = le.fit_transform(df.Grade)
```

```
df
```

	pH	Temprature	Taste	Odor	Fat	Turbidity	Colour	Grade
0	6.6	35	1	0	1	0	254	0
1	6.6	36	0	1	0	1	253	0
2	8.5	70	1	1	1	1	246	1
3	9.5	34	1	1	0	1	255	1
4	6.6	37	0	0	0	0	255	2
...	...	...	...	...	...	...	...	...

Implementing KNN

```
#Calculating euclidean distance
def euclidean_distance(row1, row2):
  distance = 0.0
  for i in range(len(row1)-1):
    distance += (row1[i] - row2[i])**2
  return math.sqrt(distance)

#getting neighbors
def get_neighbors(train, test_row, num_neighbors):
```

```
  distances = list()
  for train_row in train:
    dist = euclidean_distance(test_row, train_row)
    distances.append((train_row, dist))
  distances.sort(key=lambda tup: tup[1])
  neighbors = list()
  for i in range(num_neighbors):
    neighbors.append(distances[i][0])
  return neighbors

#Predicting class
def predict_classification(train, test_row, num_neighbors):
  neighbors = get_neighbors(train, test_row, num_neighbors)
  output_values = [row[-1] for row in neighbors]
  prediction = max(set(output_values), key=output_values.count)
  return prediction
```

Y_pred and Y:

```
result = pd.DataFrame(data = {'Predicted_Y ' : y_pred ,
                              'Y' : y})
result
```

	Predicted_Y	Y
0	2	0
1	0	0
2	1	1
3	1	1
4	2	2

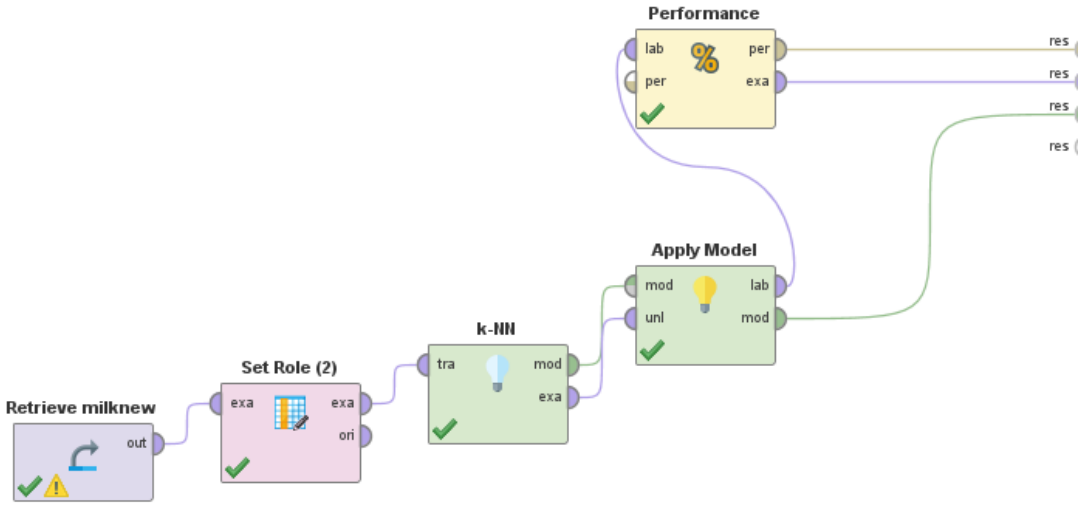
Calculating accuracy:

```
[53] count = 0
     for i in range(len(y_pred)):
       if y_pred[i] == y[i]:
         count += 1

     print('Accuracy : ',count/len(y)*100)
     print(f'{count} out of {len(y)} matches are True')
```

```
Accuracy :  99.6222851746931
1055 out of 1059 matches are True
```

2. Implementation using RapidMiner



We used 3 Neighbors

KNNClassification

```
3-Nearest Neighbour model for classification.
The model contains 1059 examples with 7 dimensions of the following classes:
  high
  low
  medium
```

accuracy: 99.62%

	true high	true low	true medium	class precision
pred. high	255	0	2	99.22%
pred. low	0	428	0	100.00%
pred. medium	1	1	372	99.47%
class recall	99.61%	99.77%	99.47%	

Questions:

1. **What are advantages and disadvantages of KNN?**

   Advantages:

   - No Training Period- KNN modeling does not include training period as the data itself is a model which will be the reference for future prediction and because of this it is very time efficient in term of improvising for a random modeling on the available data.
   - Easy Implementation- KNN is very easy to implement as the only thing to be calculated is the distance between different points on the basis of data of different features and this distance can easily be calculated using distance formula such as- Euclidian or Manhattan
   - As there is no training period thus new data can be added at any time since it won't affect the model.

   Disadvantages:-

   - Does not work well with large dataset as calculating distances between each data instance would be very costly.
   - Does not work well with high dimensionality as this will complicate the distance calculating process to calculate distance for each dimension.
   - Sensitive to noisy and missing data
   - Feature Scaling- Data in the entire dimension should be scaled (normalized and standardized) properly.

2. **Explain over fitting and under fitting problem in Machine Learning.**

   Overfitting occurs when our machine learning model tries to cover all the data points or more than the required data points present in the given dataset. Because of this, the model starts caching noise and inaccurate values present in the dataset, and all these factors reduce the efficiency and accuracy of the model. The overfitted model has low bias and high variance.

   Underfitting occurs when our machine learning model is not able to capture the underlying trend of the data. To avoid the overfitting in the model, the fed of training data can be stopped at an early stage, due to which the model may not learn enough from the training data. As a result, it may fail to find the best fit of the dominant trend in the data.In the case of underfitting, the model is not able to learn enough from the training data, and hence it reduces the accuracy and produces unreliable predictions.An underfitted model has high bias and low variance.

Outcomes: CO3 Comprehend radial-basis-function (RBF) networks and Kernel learning method work

Conclusion: (Conclusion to be based on the objectives and outcomes achieved) : Using python and rapidminer we were able to implement KNN algorithm which gave us good accuracy as well

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References:

Books/ Journals/ Websites:

1. Han, Kamber, "Data Mining Concepts and Techniques", Morgan Kaufmann 3nd Edition