NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA
TECHNOLOGICAL UNIVERSITY, BELGAUM, APPROVED BY AICTE & GOVT.OF
KARNATAKA

PROJECT REPORT

On

PROGRAM IMPLEMENTATION OF KNN ALGORITHM FOR DETECTION OF
PROSTATE CANCER USING A GIVEN DATASET

*Submitted in partial fulfilment of the requirement for the award of Degree of Bachelor of
Engineering for the course Data Mining*

in

Computer Science and Engineering

Submitted by:

RISHABH S	1NT19CS157
BHARGAV D BHAT	1NT19CS053
ROHAN THOMAS	1NT19CS162
RAVI KIRAN REDDY B H	1NT19CS154

Under the Guidance of

Dr. Sujatha Joshi
Associate Professor
Dept of CSE

Department of Computer Science and Engineering
(Accredited by NBA Tier-1)

2021-2022

NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM
, APPROVED BY AICTE & GOVT.OF KARNATAKA)

Department of Computer Science and Engineering
(Accredited by NBA Tier-1)

CERTIFICATE

This is to certify that the **PROGRAM IMPLEMENTATION OF KNN ALGORITHM FOR DETECTION OF PROSTATE CANCER USING A GIVEN DATASET** is an authentic work carried out by **Rishabh S(1NT19CS157)**, **Bhargav D Bhat (1NT19CS053),Rohan Thomas(1NT19CS162) and Ravi Kiran Reddy B H(1NT19CS154)** bonafide students of **Nitte Meenakshi Institute of Technology**, Bangalore in partial fulfilment for the award of the degree of ***Bachelor of Engineering*** in COMPUTER SCIENCE AND ENGINEERING of Visvesvaraya Technological University, Belagavi during the academic year ***2021-2022.*** It is certified that all corrections and suggestions indicated during the internal assessment has been incorporated in the report. This project has been approved as it satisfies the academic requirement in respect of project work presented for the said degree.

Internal Guide	**Signature of the HOD**	**Signature of Principal**
_____	_____	_____
Dr. Sujatha Joshi	Dr. Sarojadevi H.	Dr. H. C. Nagaraj
Associate Professor, Dept.	Professor, Head, Dept.	Principal,
CSE, NMIT Bangalore	CSE, NMIT Bangalore	NMIT,
		Bangalore

Signature of Examiner (Internals**)**   **Signature of Examiner (**SEE**)3**

1._____   1. _____

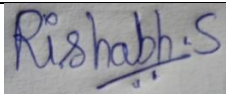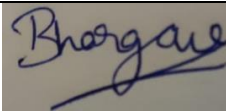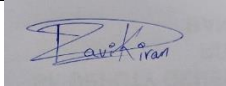2. _____   2. _____

DECLARATION

We hereby declare that

(i) The project work is our original work
(ii) This Project work has not been submitted for the award of any degree or examination at any other university/College/Institute.
(iii) This Project Work does not contain other persons' data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.
(iv) This Project Work does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
a) their words have been re-written, but the general information attributed to them has been referenced.
b) where their exact words have been used, their writing has been placed inside quotation marks, and referenced.
(v) This Project Work does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the thesis and in the References sections.

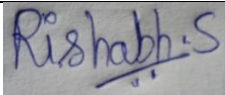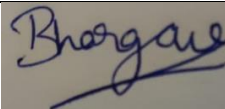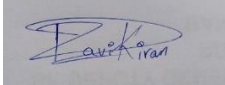| NAME	USN	Signature
RISHABH S	1NT19CS157	
BHARGAV D BHAT	1NT19CS053	
ROHAN THOMAS	1NT19CS162	
RAVI KIRAN REDDY B H	1NT19CS154	

Date: 19/01/2022

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success. I express my sincere gratitude to our Principal **Dr. H. C. Nagaraj**, Nitte Meenakshi Institute of Technology for providing facilities.

We wish to thank our HoD**, Dr. Sarojadevi H.** for the excellent environment created to further educational growth in our college. We also thank him for the invaluable guidance provided which has helped in the creation of a better project.

I hereby like to thank our *Dr. Sujatha Joshi, Associate Professor*, Department of Computer Science & Engineering on her periodic inspection, time to time evaluation of the project and help to bring the project to the present form.

Thanks to our Departmental Project coordinators. We also thank all our friends, teaching and non-teaching staff at NMIT, Bangalore, for all the direct and indirect help provided in the completion of the project.

NAME	USN	Signature
RISHABH S	1NT19CS157	
BHARGAV D BHAT	1NT19CS053	
ROHAN THOMAS	1NT19CS162	
RAVI KIRAN REDDY B H	1NT19CS154	

Date: 19/01/2022

Table of Contents

1.Introduction

1.1 Motivation

Significant advances in information technology results in excessive growth of data in health care informatics [1]. Health care informatics data includes hospital details, patient's details, disease details and treatment cost. These huge data are generated from different sources and format. It can have irrelevant attributes and missing data. Applying data mining techniques is a key approach to extract knowledge from large disease data. Data mining has various methods to extract knowledge from huge disease data set. Data mining techniques like classification, clustering and rule mining can be used to analyze data and extract meaningful information. Some of the important current applications of data mining in health care includes predicting the future outcomes of diseases based on previous data collected from similar diseases, diagnosis of disease based on patient data, analysing treatment costs and demand of resources, pre-processing of noisy, missing data and minimizing the time to wait for the disease diagnosis. Data mining tools like Weka, Rapid miner, and Orange [2, 3, 4] are used to analyze and predict better result for health care data. New and current data mining tools and technologies are used in disease diagnosis and health care informatics to improve the health care services in cost effective manner and minimizing the time for disease diagnosis.

1.2 Problem Domain
Data Mining

It is concerned with the process of computationally extracting unknown knowledge from huge sets of data. Extraction of useful knowledge from the enormous data sets and providing decision-making results for the diagnosis and treatment of diseases is very important. Data mining can be used to extract knowledge by analysing and predicting various diseases. Health care data mining has great potential to discover the hidden patterns in the data sets of the medical domain. Various data mining techniques are available with their suitability dependent on the health care data. Data mining applications in health care can have a wonderful potential and effectiveness. It automates the process of finding predictive information in huge databases. Disease prediction plays an important role in data mining. Finding of a disease requires the performance of a number of tests on the patient. However, use of data mining techniques, can reduce the number of tests. This reduced test set plays significant role in performance and time. Health care data mining is an

important task because it allows doctors to see which attributes are more important for diagnosis such as age, weight, symptoms etc. This will help the doctors diagnose the disease more efficiently. Knowledge discovery in databases is the process of finding useful information and patterns in data. Knowledge discovery in databases can be done using data mining. It uses algorithms to extract the information and patterns derived by the knowledge discovery in databases process.

1.3 Aim and Objectives

Our aim or objective would be to use the K-Nearest Neighbour Regression algorithm to predict and classify information taken from an unsorted dataset into majorly two type of tumours. Namely Benign and Malignant types of tumours.

2. Data Source and Data Quality

2.1 Dataset Used

The dataset we selected to implement for KNN classification is the Prostate Cancer dataset.

It consists of 100 records which we further split into Train split that is 95 records and for test and validation we select 5 records.

id	diagnosis	radius	texture	perimeter	area	smoothne	compactn	symmetry	fractal_dimension
1	M	23	12	151	954	0.143	0.278	0.242	0.079
2	B	9	13	133	1326	0.143	0.079	0.181	0.057
3	M	21	27	130	1203	0.125	0.16	0.207	0.06
4	M	14	16	78	386	0.07	0.284	0.26	0.097
5	M	9	19	135	1297	0.141	0.133	0.181	0.059
6	B	25	25	83	477	0.128	0.17	0.209	0.076
7	M	16	26	120	1040	0.095	0.109	0.179	0.057
8	M	15	18	90	578	0.119	0.165	0.22	0.075
9	M	19	24	88	520	0.127	0.193	0.235	0.074
10	M	25	11	84	476	0.119	0.24	0.203	0.082
11	M	24	21	103	798	0.082	0.067	0.153	0.057
12	M	17	15	104	781	0.097	0.129	0.184	0.061
13	B	14	15	132	1123	0.097	0.246	0.24	0.078
14	M	12	22	104	783	0.084	0.1	0.185	0.053
15	M	12	13	94	578	0.113	0.229	0.207	0.077
16	M	22	19	97	659	0.114	0.16	0.23	0.071
17	M	10	16	95	685	0.099	0.072	0.159	0.059
18	M	15	14	108	799	0.117	0.202	0.216	0.074
19	M	20	14	130	1260	0.098	0.103	0.158	0.054
20	B	17	11	87	566	0.098	0.081	0.189	0.058

The dataset contains an ID that is the index number and Diagnosis result in the form of 'M' and 'B' for malignant and benign respectively.

It also contains 8 features used for prediction namely:

1. Radius
2. Texture
3. Perimeter
4. Area
5. Smoothness
6. Compactness
7. Symmetry
8. Fractal Dimension

2.2 Data Pre-processing

The predict function:

```python
raw_data = []
with open('Prostate_Cancer.csv','r') as md:
    next(md)
    for line in md.readlines():
        data_row = line.strip().split(',')
        raw_data.append(data_row)
#print(raw_data)

process_data = []
for row in raw_data:
    if(row[1]=='M'):
        row[1]=1
    else:
        row[1]=0
    data_row = list(map(float,row[1:]))
    process_data.append(data_row)
#print(process_data)
```

Initially Data is read from the CSV file and stored in a python list. We first skip the initial line which contains the heading of the attributes and later as Diagnosis result is in the form of Alphabets, we assign binary numbers to them and store it in its place (1 for M and 0 for B) . As data is in the form of string, we convert each attribute into float and store it in process_data python list.

3. Methods & Models

3.1 Data Mining Questions

What is KNN?

KNN: K Nearest Neighbor is one of the fundamental algorithms in machine learning. Machine learning models use a set of input values to predict output values. It is mostly used for classification. It classifies the data point on how its neighbor is classified. It classifies the new data points based on the similarity

measure of the earlier stored data points. For example, if we have a dataset of tomatoes and bananas. It will store similar measures like shape and colour. When a new object comes it will check its similarity with the colour (red or yellow) and shape. K in KNN represents the number of the nearest neighbors we used to classify new data points.

When and where do we use KNN?

We use KNN

- When we have properly labelled data. For example, if we are predicting someone is having diabetes or not the final label can be 1 or 0. It cannot be NaN or -1.
- When data is noise-free. For the diabetes data set we cannot have a Glucose level as 0 or 10000. It's practically impossible.
- Dataset is small.

What are the Pros and Cons of KNN?

Pros:

- No assumptions about data — useful, for example, for nonlinear data
- Simple algorithm — to explain and understand/interpret
- High accuracy (relatively) — it is pretty high but not competitive in comparison to better supervised learning models
- Versatile — useful for classification or regression

Cons:

- Computationally expensive — because the algorithm stores all the training data
- High memory requirement   Stores all (or almost all) of the training data
- Prediction stage might be slow (with big K)
- Sensitive to irrelevant features and the scale of the data

What are the Applications and Examples of KNN?

- KNN can be used for Recommendation Systems. Although in the real world, more sophisticated algorithms are used for the recommendation system. KNN is not suitable for high dimensional data, but KNN is an excellent baseline approach for the systems. Many companies make a personalized recommendation for its consumers, such as Netflix, Amazon, YouTube, and many more.
- KNN can search for semantically similar documents. Each document is considered as a vector. If documents are close to each other, that means the documents contain identical topics.

- KNN can be effectively used in detecting outliers. One such example is Credit Card fraud detection.

3.2 Data Mining Algorithm

Working of KNN Algorithm:

- K-nearest neighbors (KNN) algorithm uses 'feature similarity' to predict the values of new datapoints which further means that the new data point will be assigned a value based on how closely it matches the points in the training set. We can understand its working with the help of following steps
- Step 1 − For implementing any algorithm, we need dataset. So, during the first step of KNN, we must load the training as well as test data.
- Step 2 − Next, we need to choose the value of K i.e., the nearest data points. K can be any integer.
- Step 3 − For each point in the test data do the following −
- 3.1 − Calculate the distance between test data and each row of training data with the help of any of the method namely: Euclidean, Manhattan or Hamming distance. The most commonly used method to calculate distance is Euclidean.
- 3.2 − Now, based on the distance value, sort them in ascending order.
- 3.3 − Next, it will choose the top K rows from the sorted array.
- 3.4 − Now, it will assign a class to the test point based on most frequent class of these rows.
- Step 4 – End

Algorithm implementation:

```
res, _ = knn(
    process_data, query , knum,
    distance_fn=euclidean_distance, choice_fn=mean
)
```

Above image shows how the function is called. It takes 5 parameters namely the final processed data and the query list which contains the test set, knum or the k nearest neighbours number , distance function to ctypesct distance between the train set and test condition , choice function to calculate the majority voting condition.

```
def knn(data, query, k, distance_fn, choice_fn):
    neighbor_distances_and_indices = []
    for index, data1 in enumerate(data):
        #print(index,data1)
        distance = distance_fn(data1[1:], query)

        neighbor_distances_and_indices.append((distance, index))


    sorted_neighbor_distances_and_indices = sorted(neighbor_distances_and_indices)
    ypoints = []
    xpoints = []
    for dist,index in sorted_neighbor_distances_and_indices:
        ypoints.append(dist)
        xpoints.append(index)
        label = "{:.2f}".format(dist)
        plt.annotate(label,(index,dist),ha='left')
    plt.plot(xpoints,ypoints,'o')
    plt.show()
    #print(sorted_neighbor_distances_and_indices)

    k_nearest_distances_and_indices = sorted_neighbor_distances_and_indices[:k]

    k_nearest_labels = [data[i][0] for distance, i in k_nearest_distances_and_indices]
    return k_nearest_distances_and_indices , choice_fn(k_nearest_labels)
```

Here we have defined the knn algorithm. Initially we declare the neighbors empty list. Then for each of the data in the process_data we calculate the distance from the test condition and store it in the distance list and then with their respective index we store it in the neighbors list.

 We then sort them according to their distance in ascending order. then we select the first k neighbours and retrieve their labels and store them in k_nearest_labels list. When returning we return the k nearest distances with their index and majority voting result using the choice function that is the mean function.

Distance function (Euclidean distance):

The Euclidean distance between the 2 points is calculated using the formula shown below

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

p, q = two points in Euclidean n-space

$q_i, p_i =$ Euclidean vectors, starting from the origin of the space (initial point)

n = n-space

3.3 Data Mining Models

KNN: Regression and Classification:

A simple implementation of KNN regression is to calculate the average of the numerical target of the K nearest neighbors. Another approach uses an inverse distance weighted average of the K nearest neighbors. KNN regression uses the same distance functions as KNN classification.

Distance functions

$$\text{Euclidean} \qquad \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

The Euclidian distance measure is only valid for continuous variables. In the case of categorical variables, you must use the Hamming distance, which is a measure of the number of instances in which corresponding symbols are different in two strings of equal length.

Hamming Distance

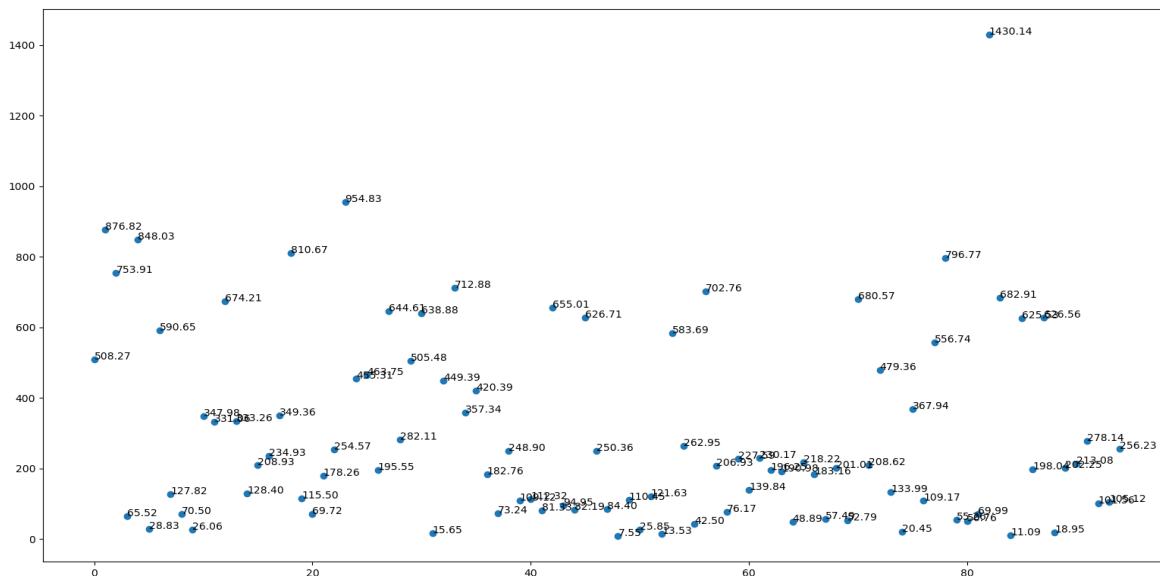$$D_H = \sum_{i=1}^{k}|x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

Choosing the optimal value for K is best done by first inspecting the data. In general, a large K value is more precise as it reduces the overall noise; however, the compromise is that the distinct boundaries within the feature space are blurred. Cross-validation is another way to retrospectively determine a good K value by using an independent data set to validate your K value. The optimal K for most datasets is 10 or more. That produces much better results than 1-NN.

4. Model Evaluation & Discussion

Data representation using Graph:

```
import matplotlib.pyplot as plt

  ypoints = []
  xpoints = []
  for dist,index in sorted_neighbor_distances_and_indices:
      ypoints.append(dist)
      xpoints.append(index)
      label = "{:.2f}".format(dist)
      plt.annotate(label,(index,dist),ha='left')
  plt.plot(xpoints,ypoints,'o')
  plt.show()
  #print(sorted_neighbor_distances_and_indices)
```

We consider 2 lists here i.e., ypoints and xpoints. ypoints contains the distance of each point from the test condition (considering the test condition to be at 0 distance from itself) and xpoints consists of its index. then using matplotlib.pyplot we annotate each point with its distance and plot the graph

With index of the point in the dataset on X-axis and its distance from the test set i.e. (X-axis) on the Y-axis.

5. Conclusion & Future Direction

Data mining techniques helps in finding the hidden knowledge in a group of disease data that can be used to analyze and predict the future behavior of diseases. Classification is one the data mining techniques which assigned a class label to a set of unclassified cases. The main objective of this project was to show that data mining algorithms such as KNN could certainly be used to efficiently predict and classify information. Future direction in this field is not limited to but can include creation of much faster and more efficient algorithms that perform similar functions.

References

1.) towardsdatascience

Appendices

a. Dataset Link:

https://www.kaggle.com/alihantabak/prostate-cancer-predictions-with-ml-and-dl-methods/data

b. Python Code:

knn_.py file

```
from knn_func import knn,euclidean_distance,mean




def predict(query,knum):
    raw_data = []
```

```python
    with open('Prostate_Cancer.csv','r') as md:
        next(md)
        for line in md.readlines():
            data_row = line.strip().split(',')
            raw_data.append(data_row)
    #print(raw_data)

    process_data = []
    for row in raw_data:
        if(row[1]=='M'):
            row[1]=1
        else:
            row[1]=0
        data_row = list(map(float,row[1:]))
        process_data.append(data_row)
    #print(process_data)

    res, _ = knn(
        process_data, query , knum,
        distance_fn=euclidean_distance, choice_fn=mean
    )
    #print(round(_))
    res1 = round(_)
    if(res1==1):
        print("M")
    else:
        print("B")
    print(res)


query1 = [21,24,74,413,0.09,0.075,0.162,0.066]
predict(query1,5)
```

knn_func.py file

```python
import math
import matplotlib.pyplot as plt
#import numpy as np


def knn(data, query, k, distance_fn, choice_fn):
    neighbor_distances_and_indices = []
    for index, data1 in enumerate(data):
        #print(index,data1)
        distance = distance_fn(data1[1:], query)

        neighbor_distances_and_indices.append((distance, index))
```

```
        sorted_neighbor_distances_and_indices = sorted(neighbor_distances_and_indices)
        ypoints = []
        xpoints = []
        for dist,index in sorted_neighbor_distances_and_indices:
            ypoints.append(dist)
            xpoints.append(index)
            label = "{:.2f}".format(dist)
            plt.annotate(label,(index,dist),ha='left')
        plt.plot(xpoints,ypoints,'o')
        plt.show()
        #print(sorted_neighbor_distances_and_indices)

        k_nearest_distances_and_indices = sorted_neighbor_distances_and_indices[:k]

        k_nearest_labels = [data[i][0] for distance, i in k_nearest_distances_and_indices]
        return k_nearest_distances_and_indices , choice_fn(k_nearest_labels)

def mean(nums):
    #print(sum(nums))
    return sum(nums) / len(nums)


def euclidean_distance(point1, point2):
    sum1 = 0
    for i in range(len(point1)):
        sum1 += math.pow(point1[i] - point2[i], 2)
    return math.sqrt(sum1)
```

c. Setup to execute the code:

1.Installation of python

2.Any text editor