**Project Based Learning Report**

on

**Implement K-nearest neighbor classifiers for the credit card defaulter dataset**

Submitted in the partial fulfillment of the requirements

For the Project based learning in **Artificial Intelligence and Data Mining**

in

Electronics & Communication Engineering

**PRN Name of the Student**

**2014111056 Sankalp Kanikar**

**2014111076 Mansi Lal**

**2014111050 Rohit Kumar Gupta**

Under the guidance of Course In-charge

**Dr. Tanuja S. Dhope**

Department of Electronics & Communication Engineering

Bharati Vidyapeeth

(Deemed to be University)

College of Engineering,

Pune – 411043

**Academic Year: 2023-24**

**Bharati Vidyapeeth**

**(Deemed to be University)**

**College of Engineering,**

**Pune – 411043**

**DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING**

**CERTIFICATE**

Certified that the Project Based Learning report entitled. “**Implement K-nearest neighbor classifiers for the credit card defaulter dataset**”

is work done by

**PRN Name of the Student**

**2014111050 ROHIT KR. GUPTA**

**2014111056 SANKALP KANIKAR**

**2014111076 MANSI LAL**

in partial fulfillment of the requirements for the award of credits for Project Based Learning (PBL) in “**Artificial Intelligence and Data Mining”** of Bachelor of Technology Semester VII, in ELECTRONICS AND COMMUNICATION ENGINEERING

**Date:**

**Dr. Tanuja S. Dhope Dr. Arundhati A .Shinde**

**Course In-charge Professor & Head**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **SERIAL NO.** | **CONTENTS** | **PAGE NO.** |
| 1 | Introduction | 5-6 |
| 2 | Procedure | 7 |
| 3 | Program | 8-14 |
| 4 | Result | 14 |
| 5 | Conclusion | 15 |
| 6 | Software Used | 16 |

**Problem Statement :**

**Implement K-nearest neighbor classifiers  for the credit card defaulter dataset**

**INTRODUCTION**

Credit card is a physical card that is used for easily paying amount of a shopping. The cardholder could use it to give a paying promise as a requital to the cost of services and goods. The issuer (possibly a bank) of the card assigns a credit for the cardholder to use it as cash advance or for payment to a dealer. For the banks the most important thing during credit card marketing is the payment capability of customers. In this study a payment status estimation has been proposed for credit card customers. For this purpose, data mining algorithms have been used.

**What is Data Mining?**

Data mining is a computational process that reveals patterns in data sets by using such methods like artificial intelligence, machine learning, statistics etc. The methods used in data mining are investigated in two groups as predictive and descriptive. In predictive methods, a model is created by using a dataset whose results are known. For example, in a bank, the properties of customers who pay their credits back can be revealed and a model can be created by using previous data sets about funding of them. Afterward this model can be used on new customers for determining the possibility of pay their credits back. In descriptive methods, a relationship can be searched between two data sets. For example, the shopping habits of two different culture may be investigated for similarity.

Data mining methods can be divided into three groups due to their function.

1. Classification and Regression

2. Clustering

3. Association Rules

Data mining methods are used to classifying the data set. In order to learn a model which can divide an input data into given categories, training samples are used in classification process. The classification operations include subsequent steps: a training dataset creation, determination of classes, describing attributes, determining more effective attributes, relevance analysis, model learning, usage of the model for classifying of an unknown data.

In this study, an estimation about whether the payment for next month is going to be done or not by the credit card clients in the default credit card clients data set with 23 attributes obtained from the UCI Machine Learning Repository, have been done. For estimation kNN and MLP algorithms have been used. The success rates and error values have been presented and compared with each other.

**Materials**

**Dataset**

In this study the default of credit card client’s data set obtained from UCI Machine Learning Repository have been used. This data set have been obtained from Credit cart customers’ default payments in Taiwan. In this data set there are 23 attributes and a binary type class.

These attributes and descriptions are as follow:

**X1: The credit amount (NT dollar): it involves total credit that**

**assign for the cardholder and his/her family.**

**X2: Sex (1:M; 2: F).**

**X3: Edu. (1: Graduate; 2: Unv.; 3: High School; 4: Others).**

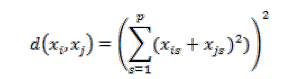
**X4: Marital (1:M; 2:S; 3: Oth).**

**X5: Age (in years)**

**Machine learning algorithms:**

**1.** K-Nearest Neighbours Algorithm (k-NN)

**K-Nearest Neighbours Algorithm:** A supervised learning algorithm, k-NN solves classification problems. Classification is the examination of the attributes of an image and the designation of this image to a predefined class. The critical point is the determination of the features of each category previously [9]. Conforming to the used classification algorithm k-NN based on the attributes drawn from the classification stage, the distance of the new individual that is wanted to be classified to all previous individuals is considered and the nearest k class is used. As an outcome of this procedure, the belonging of the test data is determined due to the k-nearest neighbour category which contains more exactly determined classes. In k-NN, the determination of the algorithm used for distance calculation and neighbour number are the critical optimization points. In the study, the optimum k number is appointed with experiments. In the calculation of distance, the Euclidean Distance is performed. Euclidean calculation method.

****

xi and xj are two points that is wanted to be learnt the distance between them.

**Procedure**

There are 7 steps involved:

**1. Importing Libraries:** In this step, we load and import the necessary programming libraries and packages, such as NumPy, Pandas, and Scikit-Learn, to access the tools and functions needed for data analysis and machine learning.

**2. Data Visualization and Analysis:** This involves exploring the dataset through graphs, charts, and statistical measures to gain insights into the data's distribution, trends, and patterns, helping you understand the information contained within the dataset.

**3. Observing Correlation between features of the Dataset:** We assess the relationships between different features or variables in the dataset to identify how they influence each other. Correlation analysis helps in understanding which attributes are most important in the dataset.

**4. Data Cleaning:** This step involves handling missing values, removing duplicates, and addressing outliers to ensure that the dataset is accurate and suitable for machine learning. Clean data is essential for building robust models.

**5. Feature Scaling of Numerical Attributes:** Scaling ensures that numerical features are on a consistent scale, preventing one variable from dominating others during model training. Common techniques include standardization (mean=0, variance=1) and normalization (scaling to a specific range).

**6. Splitting Dataset into training (70%) and test set (30%):** The dataset is divided into two parts, with 70% typically used for training the machine learning model and 30% for testing its performance. This separation helps evaluate the model's ability to generalize to unseen data.

**7. Applying Machine Learning Algorithm for Classification Problem**: Here, we select and implement a machine learning algorithm suitable for the classification problem at hand. This algorithm is trained on the training data and evaluated on the test data to make predictions and assess its performance in classifying or predicting outcomes.

**Program**

**Step1: Importing Libraries**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import os

import sys

%matplotlib inline

import warnings

warnings.filterwarnings('ignore')

**# Importing the Credit Card Default Dataset**

dataset = pd.read\_csv('UCI\_Credit\_Card.csv')

#dataset = dataset.sample(n=2000,replace =False,random\_state=1)

dataset.head(5)

dataset.shape

dataset.tail()

dataset.info()

dataset.describe()

dataset['ID'].nunique()

**There are 30000 distinct credit card clients.**

dataset.columns

dataset.index = dataset['ID']

dataset.head(3)

dataset.drop('ID',axis=1,inplace=True)

pd.set\_option('display.max\_columns', 50)

dataset.head(1)

dataset.columns

dataset.isnull().sum()

dataset['SEX'].value\_counts(dropna=False)

dataset['EDUCATION'].value\_counts(dropna=False)

dataset = dataset.rename(columns={'default.payment.next.month': 'def\_pay',

'PAY\_0': 'PAY\_1'})

dataset.head()

pd.set\_option('display.max\_columns', 30)

dataset

dataset.columns

**Step2: Data Visualisation and Analysis**

plt.style.use('fivethirtyeight')

dataset['SEX'].hist()

plt.xlabel('SEX')

plt.ylabel('COUNT')

plt.title('SEX v/s COUNT')

**Number of Male credit holder is less than Female**

plt.style.use('fivethirtyeight')

dataset['def\_pay'].hist()

plt.xlabel('DEFAULT\_PAY')

plt.ylabel('COUNT')

plt.title('Default Credit Card Clients - target value - data unbalance\n (Default = 0, Not Default = 1)')

**Percentage of Defaulters are smaller than the Non Defaulters in the given dataset**

plt.style.use('fivethirtyeight')

dataset['EDUCATION'].hist()

plt.xlabel('EDUCATION')

plt.ylabel('COUNT')

plt.title('EDUCATION v/s COUNT')

**More number of credit holders are university students followed by Graduates and then High school students**

plt.style.use('fivethirtyeight')

dataset['MARRIAGE'].hist()

plt.xlabel('MARRIAGE')

plt.ylabel('COUNT')

plt.title('MARRIAGE v/s COUNT')

**\*\*More number of credit cards holder are Married\*\***

sns.barplot(x='SEX',y='LIMIT\_BAL',data=dataset,hue='SEX')

sns.countplot(x='SEX',data=dataset,hue = 'SEX')

**# Checking the number of counts of defaulters and non defaulters sexwise**

sns.countplot(x='SEX', data=dataset,hue="def\_pay", palette="muted")

**It is evident from the above output that females have overall less default payments wrt males**

**Non-Defaults have a higher proportion of Females (Sex=2)**

g=sns.countplot(x="MARRIAGE", data=dataset,hue="def\_pay", palette="muted")

**From the above plot it is clear that those people who have marital status single have less default payment wrt married status people**

g=sns.countplot(x="EDUCATION", data=dataset,hue="def\_pay", palette="muted")

**From the above plot it is clear that those people who are university students have less default payment wrt graduates and high school people**

def getColumnsNames(prefix):

return [prefix+str(x) for x in range(1,7)]

**# PAY\_1 , PAY\_2 , PAY\_3 , PAY\_4 , PAY\_5, PAY\_6**

pay\_status\_columns = getColumnsNames('PAY\_')

figure, ax = plt.subplots(2,3)

figure.set\_size\_inches(18,8)

for i in range(len(pay\_status\_columns)):

row,col = int(i/3), i%3

d = dataset[pay\_status\_columns[i]].value\_counts()

x = dataset[pay\_status\_columns[i]][(dataset['def\_pay']==1)].value\_counts()

ax[row,col].bar(d.index, d, align='center', color='red')

ax[row,col].bar(x.index, x, align='center', color='yellow', alpha=0.7)

ax[row,col].set\_title(pay\_status\_columns[i])

plt.show()

**The above figure shows bar plot for each month payment status which show the count of defaulters and non-defaulter.**

pay\_amt\_columns = getColumnsNames('PAY\_AMT')

figure, ax = plt.subplots(3,2)

figure.set\_size\_inches(18,8)

for i in range(len(pay\_status\_columns)):

row,col = i%3, int(i/3)

ax[row,col].hist(dataset[pay\_amt\_columns[i]], 30, color ='red')

ax[row,col].hist(dataset[pay\_amt\_columns[i]][(dataset['def\_pay']==1)],30,color='yellow',alpha = 0.7)

ax[row,col].set\_title(pay\_amt\_columns[i])

#adding scaling to make the graph more helpful

ax[row,col].set\_yscale('log', nonposy='clip')

plt.tight\_layout()

plt.show()

bill\_atm\_columns = getColumnsNames('BILL\_AMT')

figure, ax = plt.subplots(3,2)

figure.set\_size\_inches(10,10)

for i in range(len(pay\_status\_columns)):

row,col = i%3, int(i/3)

ax[row,col].hist(dataset[bill\_atm\_columns[i]], 20,rwidth=0.9, color ='red')

ax[row,col].hist(dataset[bill\_atm\_columns[i]][(dataset['def\_pay']==1)],20,rwidth=0.9,color='yellow',alpha = 0.7)

ax[row,col].set\_title(bill\_atm\_columns[i])

#adding scaling to make the graph more helpful

ax[row,col].set\_yscale('log', nonposy='clip')

plt.tight\_layout()

plt.show()

sns.boxplot(x='def\_pay',y='AGE',data=dataset,palette='rainbow')

sns.boxplot(x='def\_pay',hue='MARRIAGE', y='AGE',data=dataset,palette="rainbow")

sns.boxplot(x='def\_pay',hue='EDUCATION', y='AGE',data=dataset,palette="rainbow" )

sns.boxplot(x='SEX',hue='def\_pay', y='LIMIT\_BAL',data=dataset,palette="rainbow")

sns.boxplot(x='EDUCATION',hue='def\_pay', y='LIMIT\_BAL',data=dataset,palette="rainbow")

sns.boxplot(x='MARRIAGE',hue='def\_pay', y='LIMIT\_BAL',data=dataset,palette="rainbow")

sns.distplot(dataset['LIMIT\_BAL'],kde=True,bins=30)

**# plot columns with similar names to check the correlation**

sns.pairplot(dataset, vars=dataset.columns[11:17], kind='scatter',hue= 'def\_pay')

sns.pairplot(dataset, vars=dataset.columns[17:23],hue = 'def\_pay')

**Step3:** **Observing Correlation between features of the Dataset**

correlation = dataset.corr()

plt.subplots(figsize=(30,10))

sns.heatmap( correlation, square=True, annot=True, fmt=".1f" )

**So it looks like the PAY\_0, PAY\_X variables are the strongest predictors of default, followed by the LIMIT\_BAL and PAY\_AMT\_X variables.**

X = dataset.drop(['def\_pay'],axis=1)

X.corrwith(dataset['def\_pay']).plot.bar(figsize = (20, 10), title = "Correlation with Default",

fontsize = 20,rot = 90, grid = True)sns.jointplot(x='LIMIT\_BAL',y='AGE',data=dataset,kind="scatter")

**# Facet Grid**

g = sns.FacetGrid(dataset, col = 'def\_pay', row = 'SEX')

g.map(plt.hist, 'AGE')

**From the above FaceGrid Plot we can see that NonDefaults have a higher proportion of people 30-40years**

g = sns.FacetGrid(dataset, col='SEX', hue='def\_pay')

g.map(plt.hist, 'AGE', alpha=0.6, bins=25) #alpha is for opacity

g.add\_legend()

**From the above FaceGrid Plot we can see that NonDefaults have a higher proportion of Female age between 30-40years .**

g = sns.FacetGrid(dataset, col='def\_pay', row= "MARRIAGE", hue='SEX')

g.map(plt.hist, 'AGE', alpha=0.3, bins=25)

g.add\_legend()

**Step4: Data Cleaning**

fil = (dataset.EDUCATION == 5) | (dataset.EDUCATION == 6) | (dataset.EDUCATION == 0)

dataset.loc[fil, 'EDUCATION'] = 4

dataset.EDUCATION.value\_counts()

dataset['EDUCATION'].value\_counts(dropna = False)

dataset.loc[dataset.MARRIAGE == 0, 'MARRIAGE'] = 3

dataset.MARRIAGE.value\_counts()

dataset.head()

dataset.tail()

fil = (dataset.PAY\_1 == -1) | (dataset.PAY\_1==-2)

dataset.loc[fil,'PAY\_1']=0

dataset.PAY\_1.value\_counts()

fil = (dataset.PAY\_2 == -1) | (dataset.PAY\_2==-2)

dataset.loc[fil,'PAY\_2']=0

dataset.PAY\_2.value\_counts()

fil = (dataset.PAY\_3 == -1) | (dataset.PAY\_3==-2)

dataset.loc[fil,'PAY\_3']=0

dataset.PAY\_3.value\_counts()

fil = (dataset.PAY\_4 == -1) | (dataset.PAY\_4==-2)

dataset.loc[fil,'PAY\_4']=0

dataset.PAY\_4.value\_counts()

fil = (dataset.PAY\_5 == -1) | (dataset.PAY\_5==-2)

dataset.loc[fil,'PAY\_5']=0

dataset.PAY\_5.value\_counts()

fil = (dataset.PAY\_6 == -1) | (dataset.PAY\_6==-2)

dataset.loc[fil,'PAY\_6']=0

dataset.PAY\_6.value\_counts()

dataset.head()

dataset.tail()

dataset.plot(y = 'PAY\_1',kind='hist')

plt.legend()

plt.show()

dataset['PAY\_1'].describe()

dataset.info()

dataset.SEX.nunique()

dataset[['PAY\_AMT1', 'PAY\_AMT2', 'PAY\_AMT3', 'PAY\_AMT4', 'PAY\_AMT5', 'PAY\_AMT6']].describe()

dataset[['BILL\_AMT1', 'BILL\_AMT2', 'BILL\_AMT3', 'BILL\_AMT4', 'BILL\_AMT5', 'BILL\_AMT6']].describe()

dataset.columns

dataset.shape

dataset.columns = dataset.columns.map(str.lower)

dataset.head()

**Step5: Feature Scaling of Numerical Attributes**

col\_to\_norm = ['limit\_bal', 'age', 'bill\_amt1', 'bill\_amt2', 'bill\_amt3', 'bill\_amt4',

'bill\_amt5', 'bill\_amt6', 'pay\_amt1', 'pay\_amt2', 'pay\_amt3',

'pay\_amt4', 'pay\_amt5', 'pay\_amt6']

dataset[col\_to\_norm] = dataset[col\_to\_norm].apply(lambda x : (x-np.mean(x))/np.std(x))

dataset.head(10)

dataset.tail(10)

**Step6: Spiliting Dataset into training(70%) and test set(30%)**

X = dataset.iloc[:,:-1].values

y = dataset.iloc[:,-1].values

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

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size = 0.3,random\_state = 1)

X\_train.shape

X\_test.shape

**Step7: Applying Machine Learning Algorithm for Classification Problem**

### **K-Nearest Neighbour**

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry.

KNN focuses on easy implementation and good performance at the cost of computational time, but in our case the size of the dataset is considerably small so we can apply KNN.

We can implement a KNN model by following the below steps:

* Load the data
* Initialise the value of k
* For getting the predicted class, iterate from 1 to total number of training data points
* Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it’s the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.
* Sort the calculated distances in ascending order based on distance values
* Get top k rows from the sorted array
* Get the most frequent class of these rows
* Return the predicted class

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

knn.fit(X\_train,y\_train)

from sklearn.neighbors import KNeighborsClassifier

error\_rate = []

for i in range(1,40):

knn = KNeighborsClassifier(n\_neighbors=i,n\_jobs=-1)

knn.fit(X\_train,y\_train)

pred\_i = knn.predict(X\_test)

error\_rate.append(np.mean(pred\_i != y\_test))

plt.figure(figsize=(10,6))

plt.plot(range(1,40),error\_rate,color='blue', linestyle='dashed', marker='o',

markerfacecolor='red', markersize=10)

plt.title('Error Rate vs. K Value')

plt.xlabel('K')

plt.ylabel('Error Rate')

knn.fit(X\_train,y\_train)

from sklearn.metrics import accuracy\_score, f1\_score, precision\_score, recall\_score, roc\_auc\_score

roc=roc\_auc\_score(y\_test, y\_pred)

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

prec = precision\_score(y\_test, y\_pred)

rec = recall\_score(y\_test, y\_pred)

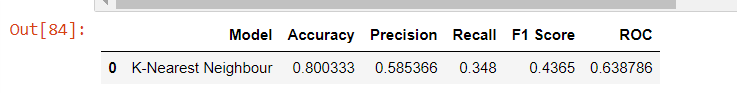
f1 = f1\_score(y\_test, y\_pred)

res = pd.DataFrame([['K-Nearest Neighbour', acc,prec,rec, f1,roc]],

columns = ['Model', 'Accuracy', 'Precision', 'Recall', 'F1 Score','ROC'])

res

**OUTPUT :**

****

**Conclusion**

In this project, credit card clients’ behaviors about payment have been estimated. For this purpose KNN (K – Nearest Neighbors ) ML Algorithm is used . The estimation success rates and error values of k-NN were calculated. The highest estimation success rate is achieved when there is only one neuron in the hidden layer was 1 and the accuracy is 80.03% and the precision rate is 58.53%

Applications Of KNN:

1. It's used in many different areas, such as handwriting detection, image recognition, and video recognition.
2. KNN is most useful when labeled data is too expensive or impossible to obtain
3. It can achieve high accuracy in a wide variety of prediction-type problems .

Hence CO6 (Design and implement the various the ML based algorithm) is achieved

**Software Used:**

****

Project Jupyter is a non-profit, open-source project, born out of the [IPython Project](https://ipython.org/) in 2014 as it evolved to support interactive data science and scientific computing across all programming languages. Jupyter will always be 100% open-source software, free for all to use and released under the liberal terms of the [modified BSD license](https://opensource.org/licenses/BSD-3-Clause).

Jupyter is developed in the open on GitHub, through the consensus of the Jupyter community. For more information on our governance, please see our [governance documentation](https://jupyter.org/governance/).

All online and in-person interactions and communications directly related to the project are covered by the [Jupyter Code of Conduct](https://jupyter.org/governance/conduct/code_of_conduct.html). This Code of Conduct sets expectations to enable a diverse community of users and contributors to participate in the project with respect and safety.