**IMMENSPHERE**

**Minor Project**

**Problem statement:**

Create a classification model to predict whether credit risk is good or bad.

* The above problem is solved using 3 different data Models
* KNN
* Logistic Regression
* SVN

**Source Code:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

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

from sklearn.preprocessing import LabelEncoder

from sklearn.linear\_model import LogisticRegression

from sklearn.neighbors import KNeighborsClassifier

from sklearn.svm import SVC

from sklearn.metrics import classification\_report, confusion\_matrix,accuracy\_score

import streamlit as st

df = pd.read\_csv('credit\_customers (1).csv')

st.header("\_Credit\_Risk\_ :red[Prediction Model]")

st.subheader(':blue[Customer\_Credit\_Data: ]')

st.dataframe(df.head())

nav = st.sidebar.radio("Select Countplot Feature",["credit\_history","savings\_status","housing","existing\_credits"])

st.subheader(':blue[Based on] '+ nav)

f1 = plt.figure(figsize=(3,3))

sns.countplot(x='class',hue=nav,data=df)

st.pyplot(f1)

# Handling null values

df = df.dropna()

# Handling duplicate records

df = df.drop\_duplicates()

# Label Encoding

categorical\_cols = ['checking\_status', 'credit\_history', 'purpose','credit\_amount','savings\_status','employment', 'existing\_credits', 'housing', 'job', 'own\_telephone','class']

df1 = df[['checking\_status', 'credit\_history', 'purpose','credit\_amount','savings\_status','employment', 'existing\_credits', 'housing', 'job', 'own\_telephone','class']].copy()

lb = LabelEncoder()

for col in categorical\_cols:

df1[col] = lb.fit\_transform(df[col])

st.subheader('\n :blue[Data after LabelEncoding: ]')

st.dataframe(df1.head(5))

x = df1.drop('class', axis=1)

y = df1['class']

st.write('Shape of features:\n')

st.write(x.shape,y.shape)

# Split data into training and test data

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.25, random\_state=42)

classifier\_name = st.sidebar.selectbox('Choose classifier for Prediction',('KNN', 'SVM','Logistic Regression'))

# Parameter Selection

def select\_param(clf\_name):

params = {}

if clf\_name == 'SVM':

C = st.sidebar.slider('C', 0.01, 10.0)

params['C'] = C

ker = st.sidebar.selectbox('Select kernel',('linear','rbf'))

params['kernel'] = ker

elif clf\_name == 'KNN':

K = st.sidebar.slider('K', 1, 15)

params['K'] = K

else:

None

return params

params = select\_param(classifier\_name)

# Classifier Selection

def get\_classifier(clf\_name, params):

clf = None

if clf\_name == 'SVM':

clf = SVC(kernel=params['kernel'],C=params['C'])

elif clf\_name == 'KNN':

clf = KNeighborsClassifier(n\_neighbors=params['K'])

else:

clf = LogisticRegression(max\_iter=10000)

return clf

model = get\_classifier(classifier\_name, params)

model.fit(x\_train,y\_train)

y\_pred = model.predict(x\_test)

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

cm = confusion\_matrix(y\_test,y\_pred)

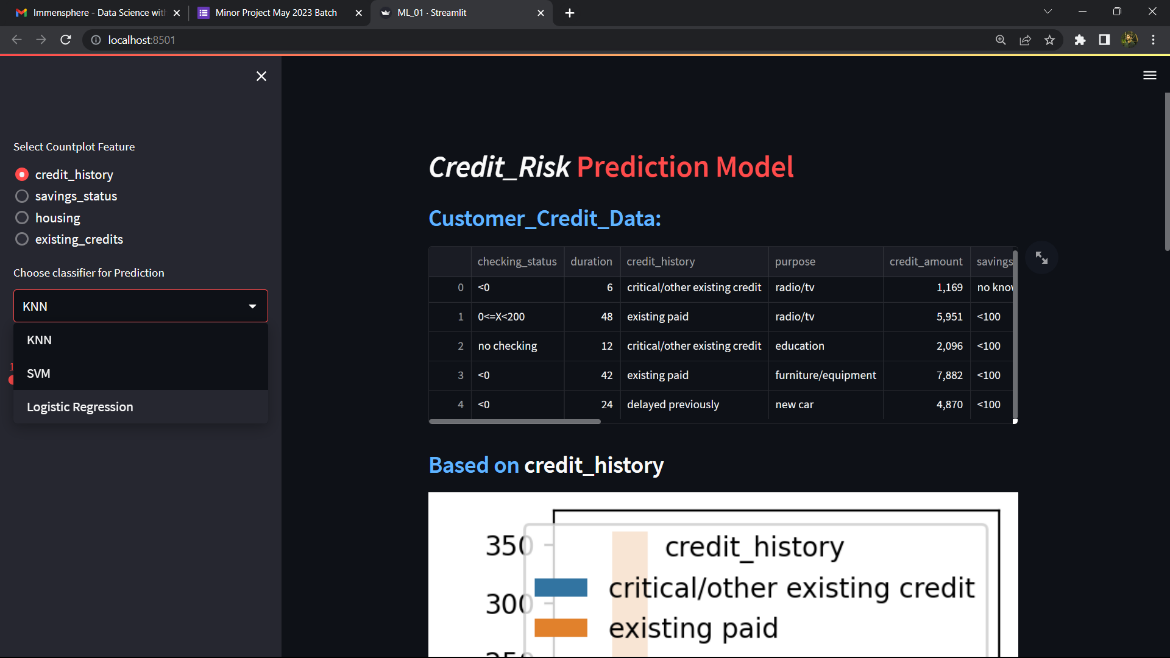
st.write(f'Classifier = {classifier\_name}')

st.write(f'Accuracy =', acc)

st.write(cm)

st.write(f'Classification Report\n',classification\_report(y\_test,y\_pred))

**Output:**

****

**SUBMITTED BY:**

**SHIVAM SONI**

**shivam.soni@ptuniv.edu.in**