WEB PHISHING DETECTION PROJECT REPORT

Submitted By

HARINI B	210819205011
JAYASRI S	210819205017
HARINI D	210819205012
SAFANA PARVEEN J	210819205041

In partial fulfilment for the award of the degree

Of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY

KINGS ENGINEERING COLLEGE, IRUNGATTUKOTAI

ANNA UNIVERSITY: CHENNAI 600025

CHAPTER 1

INTRODUCTION:

: PROJECT OVERVIEW

The terms "heart disease" and "cardiovascular disease" are frequently used interchangeably. Heart disease is a general term that covers a wide range of heart related medical conditions. The irregular health state that directly affects the heart and all of its components is characterized by these medical conditions. In order to forecast cardiac disease, this study discusses various data mining, big data, and machine learning techniques. Building an important model for the medical system to forecast heart disease or cardiovascular illness requires the use of data mining and machine learning. Our application helps the user in finding out if they have heart disease or not. They can find out by entering details such as their heart rate, cholesterol, blood pressure etc. A dashboard is also attached along with the results for better understanding where they can compare their blood pressure and similar metrics with other users. This project focuses on Random Forest Classifier. The accuracy of our project is 87% for which is better than most other systems in terms of achieving accuracy quickly.

: PURPOSE

This project's goal is to determine, depending on the patient's medical characteristics—such as gender, age, chest pain, fasting blood sugar level, etc.—whether they are likely to be diagnosed with any cardiovascular heart illnesses. The leading cause of death in the developed world is heart disease. Heart disease cases are rising quickly every day, thus it's crucial and worrisome to predict any potential illnesses in advance. This diagnosis is a challenging task that requires accuracy and efficiency. Therefore, there needs to be work done to help prevent the risks of having a heart attack or stroke. It is the main factor in adult deaths. By using a person's medical history, our initiative can identify

Empathy Map Canvas

Gain insight and understanding on solving customer problems.

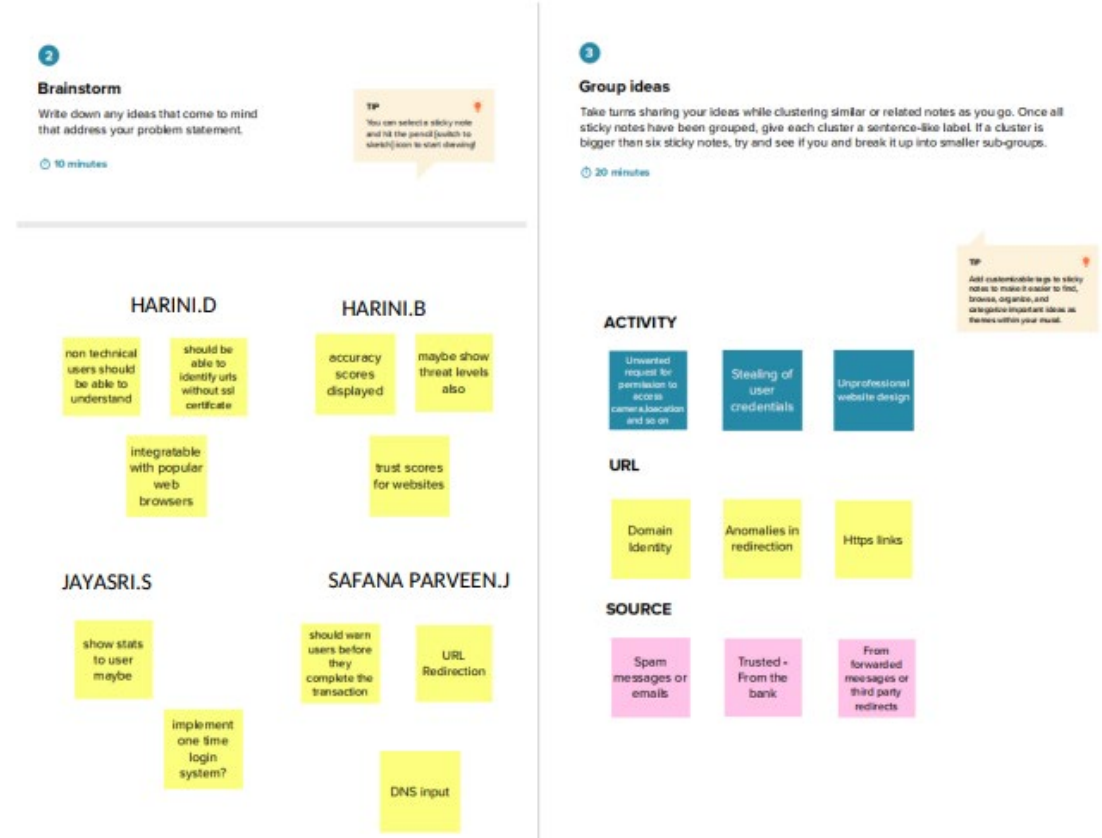
1

Build empathy and keep your focus on the user by putting yourself in their shoes.

Share your feedback

).pdf

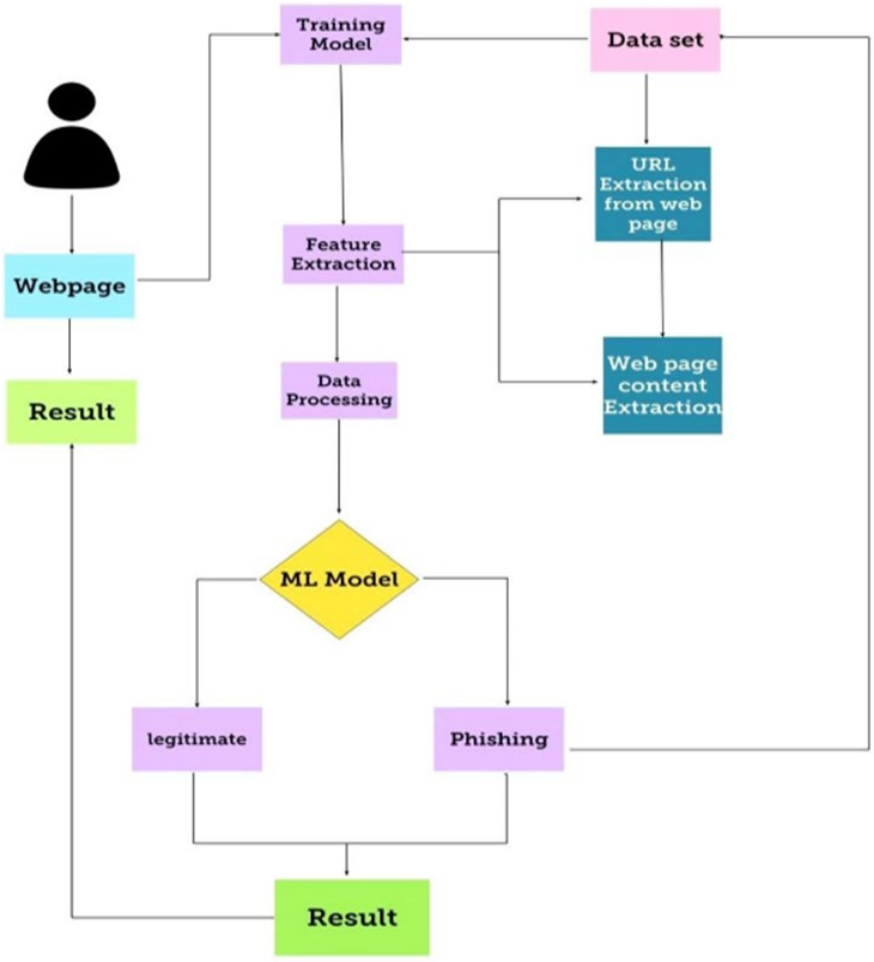
Step-2: Brainstorm, Idea Listing and Grouping

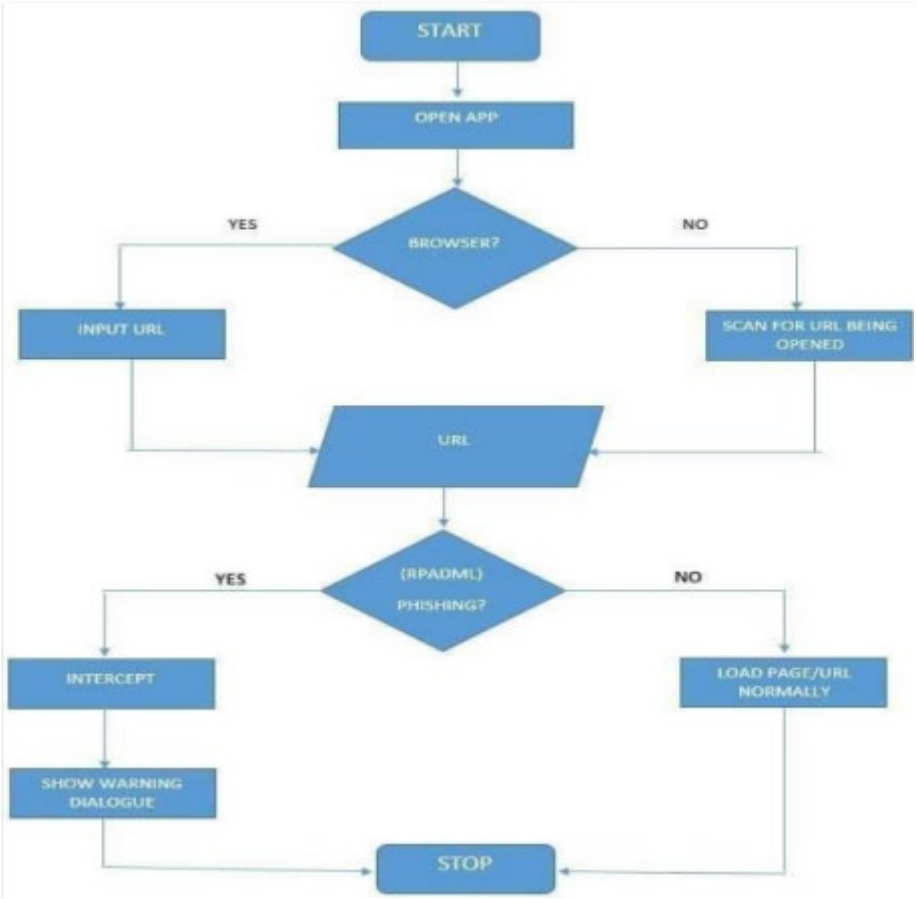


accuracy of 98.48% with 2.09% false-positive rate on benchmark dataset, which outperforms the existing baseline approaches.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Hackers are increasingly launching attacks via SMS and social media. Games and dating apps introduce yet another attack vector. However, current deep learning-based phishing detection applications are not applicable to mobile devices due to the computational burden.
2.	Idea / Solution description	Our solution that helps the user to detect and protect themselves from the spam and dangerous websites. Our application protect your personal information from the third party. We make you feel safe all the time.
3.	Novelty / Uniqueness	There are various anti-phishing techniques are there but we are different from others. We analysis the system as soon as we notice the indifference in the system. Our uniqueness is we are very user friendly application that makes yours information keeps safe and private.
4.	Social Impact / Customer Satisfaction	The customer can work with their transaction peacefully. We provide a safe and secure feel to our user. We develop a app that check the websites with the high efficiency way. We identify and remediate phishing threats before the phishing attack can cause damage.
5.	Scalability of the Solution	We automate various routine remediation process in response to threats, saving admins more time and reducing the time it takes to identify and remediate high-tier vulnerabilities

User Type	Functional Requirement	User Story Number	User Story/Task	Acceptance criteria	Priority	Release
Customer (Mobile User)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account/dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email and confirmation	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register and access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard					
Customer (Web User)	User Input	USN-1	As the user I can input the particular URL in the required field and waiting for a validation	I can go access the website without any problem	High	Sprint-1
Customer Care Executive	Feature Extraction	USN-1	After I compare in case if none found on comparison then we can extract feature using heuristic and visual similarity	In this I can have comparison between websites for security	High	Sprint-1


Training Model
Data set
URL Extraction from web page
Feature Extraction
Webpage
Web page content Extraction
Result
Data Processing
ML Model
legitimate
Phishing
Result

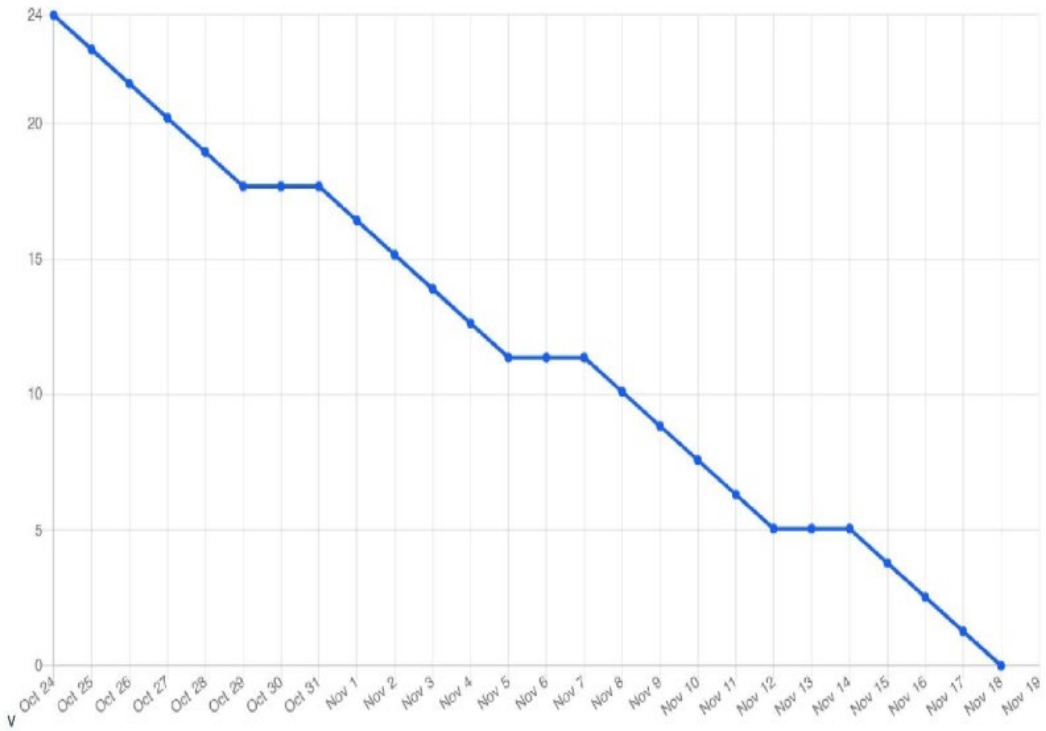

START
OPEN APP
BROWSER?
YES
NO
INPUT URL
SCAN FOR URL BEING OPENED
URL
(RPADML)
PHISHING?
YES
NO
INTERCEPT
LOAD PAGE/URL NORMALLY
SHOW WARNING DIALOGUE
STOP

7.	Improve model performance	Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right.	Machine Learning
8.	Checking accuracy	A data accuracy check, sometimes called a data sanity check, is a set of quality validations that take place before using data.	Machine Learning

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	User input	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	HARINI.B
Sprint-1	Website Comparison	USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	JAYASRI.S,SAFANA PARVEEN
Sprint-2	Feature Extraction	USN-3	As a user, I can register for the application through Facebook	2	Low	HARINI.D
Sprint-1	Prediction	USN-4	As a user, I can register for the application through Gmail	2	Medium	HARINI.B,SAFANA PARVEEN
Sprint-1	Classifier	USN-5	As a user, I can log into the application by entering email & password	1	High	JAYASRI.S


24
20
15
10
5
0
Oct 24
Oct 25
Oct 26
Oct 27
Oct 28
Oct 29
Oct 30
Oct 31
Nov 1
Nov 2
Nov 3
Nov 4
Nov 5
Nov 6
Nov 7
Nov 8
Nov 9
Nov 10
Nov 11
Nov 12
Nov 13
Nov 14
Nov 15
Nov 16
Nov 17
Nov 18
Nov 19

Code:

In [84]:
```
import numpy as np
import pandas as pd
import plotly
import plotly.express as px

import cufflinks as cf
import matplotlib.pyplot as plt
import seaborn as sns

import plotly.offline as pyo
from plotly.offline import init_notebook_mode,plot,iplot
```

In [85]:
```
pyo.init_notebook_mode(connected=True)
cf.go_offline()
```

In [86]:
```
heart
```

Out[86]:

	Age	Sex	Chest pain type	BP	Cholesterol	FBS over 120	EKG results	Max HR	Exercise angina	ST depression	Slope of ST	Number of vessels fluro	Thallium	Heart Disease
0	70	1	4	130	322	0	2	109	0	2.4	2	3	3	1
1	67	0	3	115	564	0	2	160	0	1.6	2	0	7	0
2	57	1	2	124	261	0	0	141	0	0.3	1	0	7	1
3	64	1	4	128	263	0	0	105	1	0.2	2	1	7	0
4	74	0	2	120	269	0	2	121	1	0.2	1	1	3	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
265	52	1	3	172	199	1	0	162	0	0.5	1	0	7	0
266	44	1	2	120	263	0	0	173	0	0.0	1	0	7	0
267	56	0	2	140	294	0	2	153	0	1.3	2	0	3	0
268	57	1	4	140	192	0	0	148	0	0.4	2	0	6	0
269	67	1	4	160	286	0	2	108	1	1.5	2	3	3	1

270 rows × 14 columns

In [87]:
```
heart = pd.read_csv(r'C:\Users\91967\Desktop\data\heart.csv')
```

In [88]:
```
info = ["Age","1: male, 0: female","Chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4: asymptomatic","resting blood pressure"," serum cholestoral in mg

for i in range(len(info)):
    print(heart.columns[i]+":\t\t\t"+info[i])
```

```
Age:                    Age
Sex:                    1: male, 0: female
Chest pain type:                Chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4: asymptomatic
BP:                     resting blood pressure
Cholesterol:                     serum cholestoral in mg/dl
FBS over 120:                   fasting blood sugar > 120 mg/dl
EKG results:                    resting electrocardiographic results (values 0,1,2)
Max HR:                  maximum heart rate achieved
Exercise angina:                        exercise induced angina
ST depression:                  oldpeak = ST depression induced by exercise relative to rest
Slope of ST:                    the slope of the peak exercise ST segment
Number of vessels fluro:                        number of major vessels (0-3) colored by flourosopy
Thallium:                       thal: 3 = normal; 6 = fixed defect; 7 = reversable defect
```

In [89]:
```
heart['Heart Disease']
```

Out[89]:
```
0      Presence
1       Absence
2      Presence
3       Absence
4       Absence
         ...
265     Absence
266     Absence
267     Absence
268     Absence
269    Presence
Name: Heart Disease, Length: 270, dtype: object
```

In [90]:
```
heart.groupby('Heart Disease').size()
```

Out[90]:
```
Heart Disease
Absence     150
Presence    120
dtype: int64
```

In [91]:
```
heart.groupby('Heart Disease').sum()
```

Out[91]:

Heart Disease	Age	Sex	Chest pain type	BP	Cholesterol	FBS over 120	EKG results	Max HR	Exercise angina	ST depression	Slope of ST	Number of vessels fluro	Thallium
Absence	7906	83	423	19330	36632	23	129	23750	23	93.4	210	43	568
Presence	6791	100	434	16133	30776	17	147	16663	66	190.1	218	138	700

In [92]:
```
heart.shape
```

Out[92]:
```
(270, 14)
```

In [93]:
```
heart.describe()
```

Out[93]:

	Age	Sex	Chest pain type	BP	Cholesterol	FBS over 120	EKG results	Max HR	Exercise angina	ST depression	Slope of ST	Number of vessels fluro	Thallium
count	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.00000	270.000000	270.000000	270.000000
mean	54.433333	0.677778	3.174074	131.344444	249.659259	0.148148	1.022222	149.677778	0.329630	1.05000	1.585185	0.670370	4.696296
std	9.109067	0.468195	0.950090	17.861608	51.686237	0.355906	0.997891	23.165717	0.470952	1.14521	0.614390	0.943896	1.940659
min	29.000000	0.000000	1.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.00000	1.000000	0.000000	3.000000
25%	48.000000	0.000000	3.000000	120.000000	213.000000	0.000000	0.000000	133.000000	0.000000	0.00000	1.000000	0.000000	3.000000
50%	55.000000	1.000000	3.000000	130.000000	245.000000	0.000000	2.000000	153.500000	0.000000	0.80000	2.000000	0.000000	3.000000
75%	61.000000	1.000000	4.000000	140.000000	280.000000	0.000000	2.000000	166.000000	1.000000	1.60000	2.000000	1.000000	7.000000
max	77.000000	1.000000	4.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.20000	3.000000	3.000000	7.000000

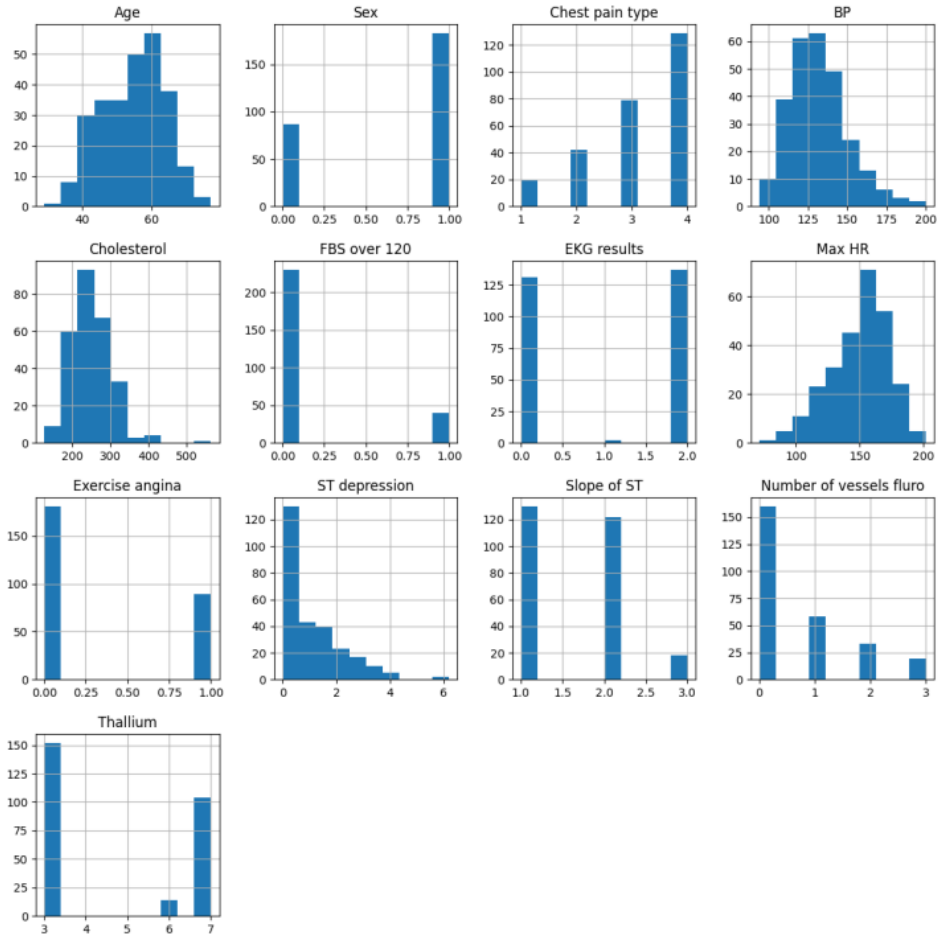
In [94]:
```
heart.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 270 entries, 0 to 269
Data columns (total 14 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Age                      270 non-null    int64
 1   Sex                      270 non-null    int64
 2   Chest pain type          270 non-null    int64
 3   BP                       270 non-null    int64
 4   Cholesterol              270 non-null    int64
 5   FBS over 120             270 non-null    int64
 6   EKG results              270 non-null    int64
 7   Max HR                   270 non-null    int64
 8   Exercise angina          270 non-null    int64
 9   ST depression            270 non-null    float64
 10  Slope of ST              270 non-null    int64
 11  Number of vessels fluro  270 non-null    int64
 12  Thallium                 270 non-null    int64
 13  Heart Disease            270 non-null    object
dtypes: float64(1), int64(12), object(1)
memory usage: 29.7+ KB
```

In [95]:
```
heart['Heart Disease'].unique()
```

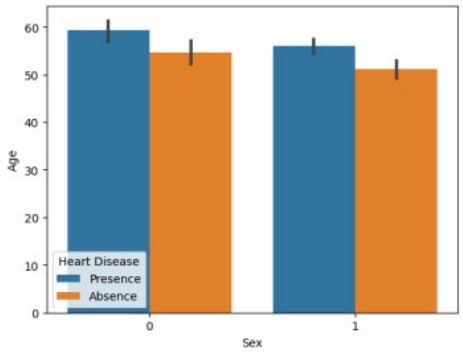
Out[95]:
```
array(['Presence', 'Absence'], dtype=object)
```

In [96]:
```
heart.hist(figsize=(14,14))
plt.show()
```


Age
Sex
Chest pain type
BP
Cholesterol
FBS over 120
EKG results
Max HR
Exercise angina
ST depression
Slope of ST
Number of vessels fluro
Thallium

In [99]:
```
sns.barplot(x=heart['Sex'],y=heart['Age'],hue=heart['Heart Disease'])
```

Out[99]:
```
<AxesSubplot: xlabel='Sex', ylabel='Age'>
```



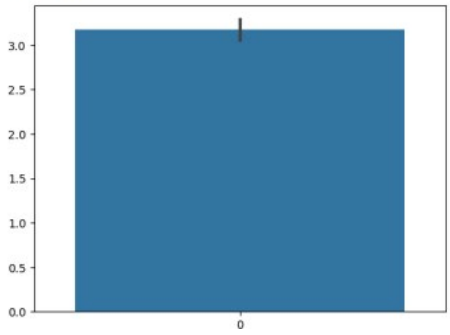
In [102…
```
heart['Heart Disease']=heart['Heart Disease'].replace({'Absence':0,'Presence':1})
```

In [103…
```
heart['Heart Disease']
```

Out[103]:
```
0      1
1      0
2      1
3      0
4      0
      ..
265    0
266    0
267    0
268    0
269    1
Name: Heart Disease, Length: 270, dtype: int64
```

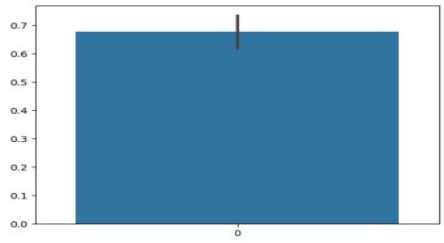
In [104…
```
sns.barplot(heart["Chest pain type"])
```

Out[104]:
```
<AxesSubplot: >
```



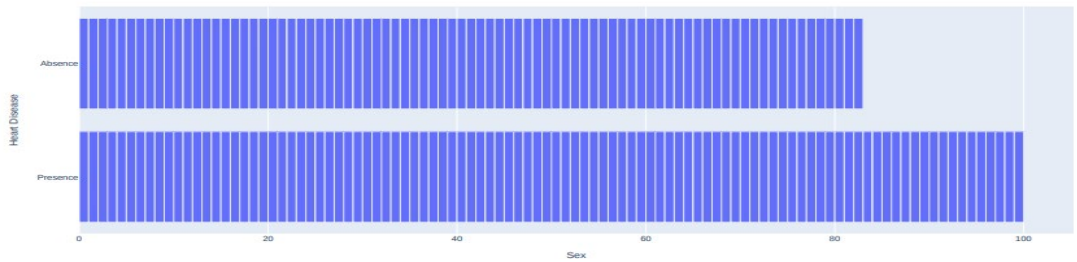
In [105…
```
sns.barplot(heart["Sex"])
```

Out[105]:
```
<AxesSubplot: >
```



In [ ]:

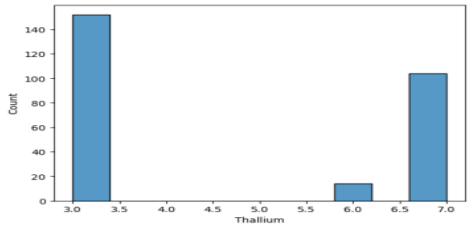
In [69]:
```
px.bar(heart,heart['Sex'],heart['Heart Disease'])
```



Sex

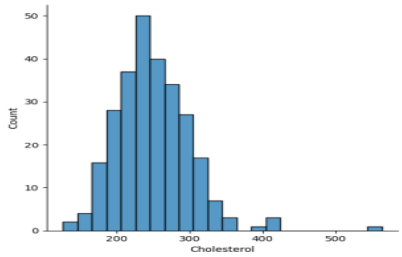
In [70]:
```
sns.histplot(heart["Thallium"])
```

Out[70]:
```
<AxesSubplot: xlabel='Thallium', ylabel='Count'>
```



In [73]:
```
sns.displot(heart["Cholesterol"])
```

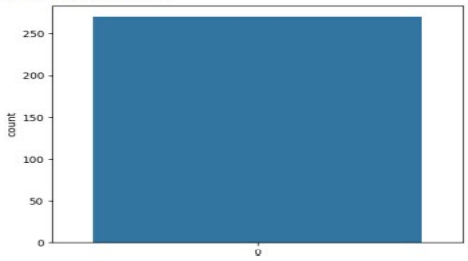
Out[73]:
```
<seaborn.axisgrid.FacetGrid at 0x12de4f95690>
```



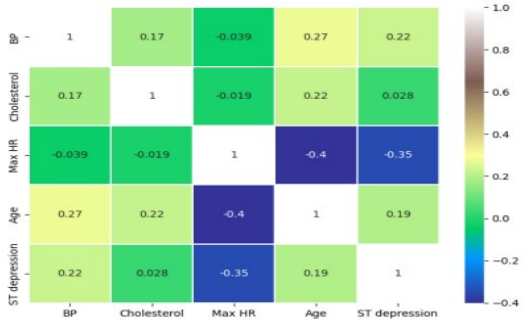
```
In [113]: heart.rename(columns={'Heart Disease':'target'},inplace=True)
```

```
In [114]: y = heart["target"]

sns.countplot(y)

target_temp = heart.target.value_counts()

print(target_temp)
```

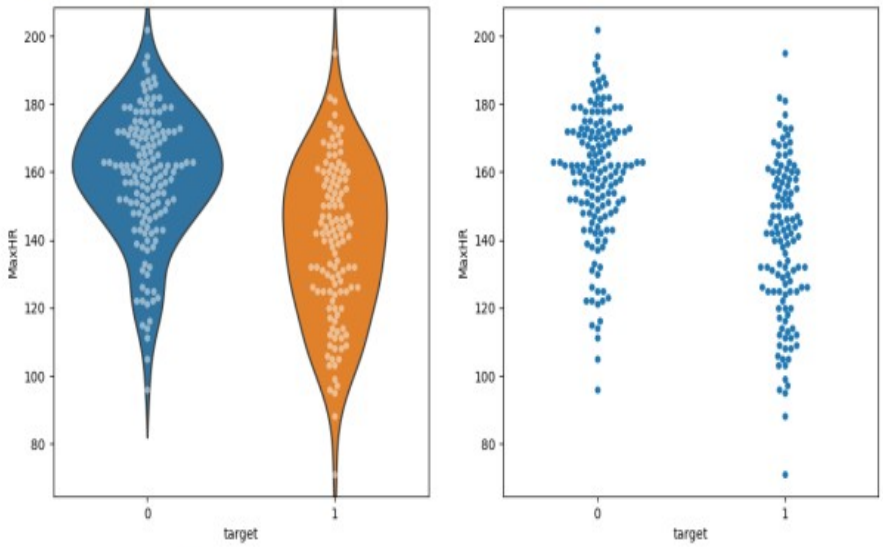
```
0    150
1    120
Name: target, dtype: int64
```



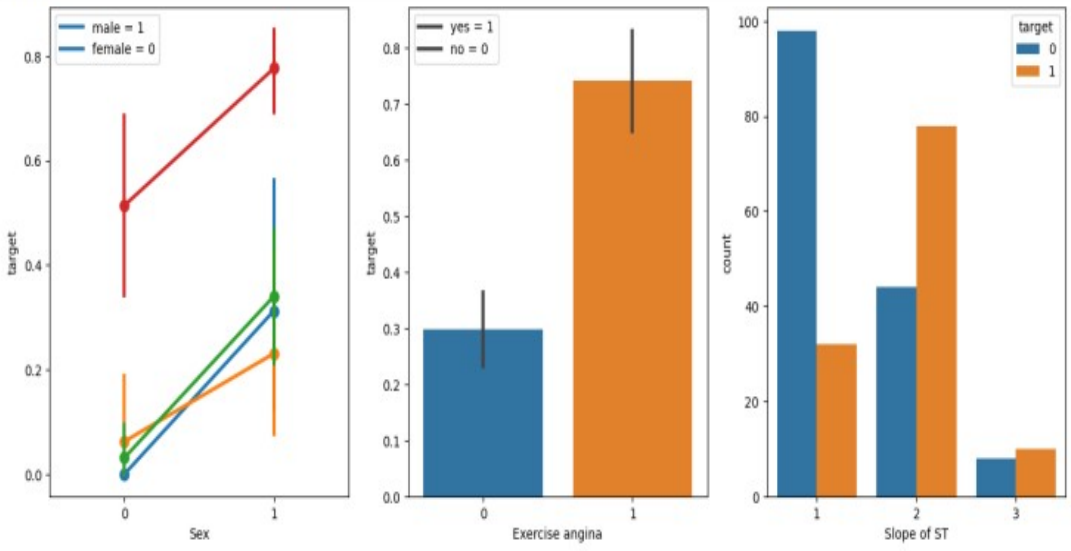

```
In [79]: sns.heatmap(heart[numeric_columns].corr(),annot=True, cmap='terrain', linewidths=0.1)
fig=plt.gcf()
fig.set_size_inches(8,6)
plt.show()
```



In [132…
```
plt.figure(figsize=(13,6))
plt.subplot(121)
sns.violinplot(x="target", y="MaxHR", data=heart, inner=None)
sns.swarmplot(x="target", y="MaxHR", data=heart, color='w', alpha=0.5)


plt.subplot(122)
sns.swarmplot(x="target", y="MaxHR", data=heart)
plt.show()
```



In [133…
```
# create pairplot and two barplots
plt.figure(figsize=(16,6))
plt.subplot(131)
sns.pointplot(x="Sex", y="target", hue='Chest pain type', data=heart)
plt.legend(['male = 1', 'female = 0'])
plt.subplot(132)
sns.barplot(x="Exercise angina", y="target", data=heart)
plt.legend(['yes = 1', 'no = 0'])
plt.subplot(133)
sns.countplot(x="Slope of ST", hue='target', data=heart)
plt.show()
```



DATA Processing

In [134]:
```
heart['target'].value_counts()
```

Out[134]:
```
0    150
1    120
Name: target, dtype: int64
```

In [135]:
```
heart['target'].isnull()
```

Out[135]:
```
0      False
1      False
2      False
3      False
4      False
       ...
265    False
266    False
267    False
268    False
269    False
Name: target, Length: 270, dtype: bool
```

In [136]:
```
heart['target'].sum()
```

Out[136]:
```
120
```

In [137]:
```
heart['target'].unique()
```

Out[137]:
```
array([1, 0], dtype=int64)
```

In [138]:
```
heart.isnull().sum()
```

Out[138]:
```
Age                       0
Sex                       0
Chest pain type           0
BP                        0
Cholesterol               0
FBS over 120              0
EKG results               0
MaxHR                     0
Exercise angina           0
ST depression             0
Slope of ST               0
Number of vessels fluro   0
Thallium                  0
target                    0
dtype: int64
```

Storing in X and y

In [139]:
```
X,y=heart,heart.target
```

In [140]:
```
X.drop('target',axis=1,inplace=True)
```

In [141]:
```
y
```

Out[141]:
```
0      1
1      0
2      1
3      0
4      0
      ..
265    0
266    0
267    0
268    0
269    1
Name: target, Length: 270, dtype: int64
```

Or X, y = heart.iloc[:, :-1], heart.iloc[:, -1]

In [142]:
```
X.shape
```

Out[142]:
```
(270, 13)
```

In [143]:
```
y.shape
```

Out[143]:
```
(270,)
```

In [144]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

In [145]:
```
sc = StandardScaler()
X = sc.fit_transform(X)
```

```
X = sc.fit_transform(X)
```

In [146…
```
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=10,test_size=0.3,shuffle=True)
```

In [147…
```
X_test
```

Out[147]:
```
array([[-1.47745975,  0.6894997 , -1.23804513, ..., -0.95423434,
        -0.71153494, -0.87570581],
       [ 1.68210896,  0.6894997 , -2.29253153, ...,  0.67641928,
         0.34978477, -0.87570581],
       [-0.37761378,  0.6894997 , -0.18355874, ...,  0.67641928,
        -0.71153494, -0.87570581],
       ...,
       [-0.81755217,  0.6894997 , -0.18355874, ..., -0.95423434,
        -0.71153494, -0.87570581],
       [ 0.50226299, -1.45032695,  0.87092765, ...,  0.67641928,
        -0.71153494, -0.87570581],
       [-0.70756757,  0.6894997 , -0.18355874, ..., -0.95423434,
         1.41127648, -0.87570581]])
```

In [148…
```
y_test
```

Out[148]:
```
111    0
170    0
186    0
185    1
121    1
      ..
217    0
250    1
69     1
58     1
194    0
Name: target, Length: 81, dtype: int64
```

In [149…
```
print ("train_set_x shape: " + str(x_train.shape))
print ("train_set_y shape: " + str(y_train.shape))
print ("test_set_x shape: " + str(X_test.shape))
print ("test_set_y shape: " + str(y_test.shape))
```

```
train_set_x shape: (189, 13)
train_set_y shape: (189,)
test_set_x shape: (81, 13)
test_set_y shape: (81,)
```

Model

In [150…
```
# Decision Tree Classifier
scores_dict = {}
```

In [151…
```
Catagory=['No....but i pray you get Heart Disease or at leaset Corona Virus Soon...','Yes you have Heart Disease....RIP in Advance']
```

In [155…
```
from sklearn.tree import DecisionTreeClassifier
dt=DecisionTreeClassifier()
dt.fit(X_train,y_train)
```

Out[155]:
```
DecisionTreeClassifier()
```

In [156…
```
print("Accuracy on training set: {:.3f}".format(dt.score(X_train, y_train)))
print("Accuracy on test set: {:.3f}".format(dt.score(X_test, y_test)))
```

```
Accuracy on training set: 1.000
Accuracy on test set: 0.778
```

In [157…
```
prediction
```

Out[157]:
```
array([0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1,
       0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0,
       1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0], dtype=int64)
```

In [158…
```
X_DT=np.array([[63 ,1, 3,145,233,1,0,150,0,2.3,0,0,1]])
X_DT_prediction=dt.predict(X_DT)
```

In [159…
```
X_DT_prediction[0]
```

Out[159]:
```
1
```

In [160…
```
print(Catagory[int(X_DT_prediction[0])])
```

```
Yes you have Heart Disease....RIP in Advance
```

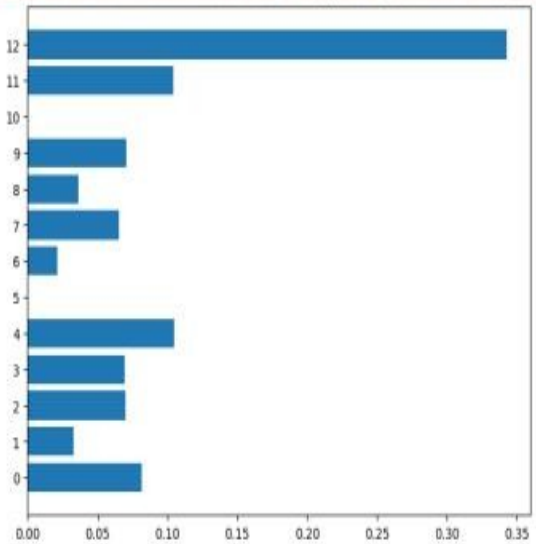
Feature Importance in Decision Trees

In [161]:
```
print("Feature importances:\n{}".format(dt.feature_importances_))
```

```
Feature importances:
[0.08148922 0.03287338 0.07020353 0.06957237 0.10489245 0.
 0.02144372 0.06531475 0.03652597 0.07047152 0.         0.10448896
 0.34272413]
```

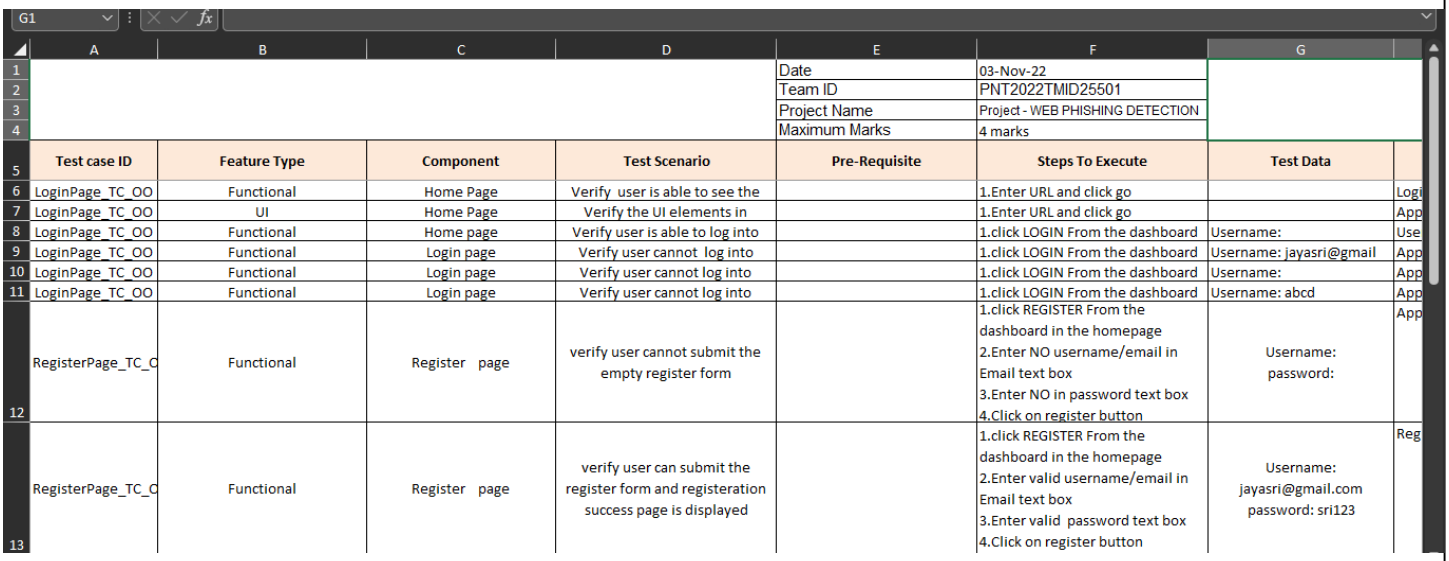
In [162]:
```
def plot_feature_importances_diabetes(model):
    plt.figure(figsize=(8,6))
    n_features = 13
    plt.barh(range(n_features), model.feature_importances_, align='center')
    plt.yticks(np.arange(n_features), X)
    plt.xlabel("Feature importance")
    plt.ylabel("Feature")
    plt.ylim(-1, n_features)
plot_feature_importances_diabetes(dt)
plt.savefig('feature_importance')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In [162], line 9
      7     plt.ylabel("Feature")
      8     plt.ylim(-1, n_features)
----> 9 plot_feature_importances_diabetes(dt)
     10 plt.savefig('feature_importance')

Cell In [162], line 5, in plot_feature_importances_diabetes(model)
      3 n_features = 13
      4 plt.barh(range(n_features), model.feature_importances_, align='center')
----> 5 plt.yticks(np.arange(n_features), X)
      6 plt.xlabel("Feature importance")
      7 plt.ylabel("Feature")

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\matplotlib\pyplot.py:1887, in yticks(ticks, labels, minor, **kwargs)
   1885         l._internal_update(kwargs)
   1886 else:
-> 1887     labels = ax.set_yticklabels(labels, minor=minor, **kwargs)
   1889 return locs, labels

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\matplotlib\axes\_base.py:73, in _axis_method_wrapper.__set_name__.<locals>.wrapper(self, *args, **kwargs)
     72 def wrapper(self, *args, **kwargs):
---> 73     return get_method(self)(*args, **kwargs)

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\matplotlib\axis.py:1968, in Axis._set_ticklabels(self, labels, fontdict, minor, **kwargs)
   1966 if fontdict is not None:
   1967     kwargs.update(fontdict)
-> 1968 return self.set_ticklabels(labels, minor=minor, **kwargs)

File ~\AppData\Local\Programs\Python\Python310\lib\site-packages\matplotlib\axis.py:1890, in Axis.set_ticklabels(self, ticklabels, minor, **kwargs)
   1886 if isinstance(locator, mticker.FixedLocator):
   1887     # Passing [] as a list of ticklabels is often used as a way to
   1888     # remove all tick labels, so only error for > 0 ticklabels
   1889     if len(locator.locs) != len(ticklabels) and len(ticklabels) != 0:
-> 1890         raise ValueError(
   1891             "The number of FixedLocator locations"
   1892             f" ({len(locator.locs)}), usually from a call to"
   1893             " set_ticks, does not match"
   1894             f" the number of ticklabels ({len(ticklabels)}).")
   1895     tickd = {loc: lab for loc, lab in zip(locator.locs, ticklabels)}
   1896     func = functools.partial(self._format_with_dict, tickd)

ValueError: The number of FixedLocator locations (13), usually from a call to set_ticks, does not match the number of ticklabels (270).
```



KNN

In [4]:
```
from sklearn.neighbors import KNeighborsClassifier
```



	A	B	C	D	E	F	G
1					Date	03-Nov-22	
2					Team ID	PNT2022TMID25501	
3					Project Name	Project - WEB PHISHING DETECTION	
4					Maximum Marks	4 marks	
5	**Test case ID**	**Feature Type**	**Component**	**Test Scenario**	**Pre-Requisite**	**Steps To Execute**	**Test Data**
6	LoginPage_TC_OO	Functional	Home Page	Verify user is able to see the		1.Enter URL and click go	
7	LoginPage_TC_OO	UI	Home Page	Verify the UI elements in		1.Enter URL and click go	
8	LoginPage_TC_OO	Functional	Home page	Verify user is able to log into		1.click LOGIN From the dashboard	Username:
9	LoginPage_TC_OO	Functional	Login page	Verify user cannot log into		1.click LOGIN From the dashboard	Username: jayasri@gmail
10	LoginPage_TC_OO	Functional	Login page	Verify user cannot log into		1.click LOGIN From the dashboard	Username:
11	LoginPage_TC_OO	Functional	Login page	Verify user cannot log into		1.click LOGIN From the dashboard	Username: abcd
12	RegisterPage_TC_O	Functional	Register page	verify user cannot submit the empty register form		1.click REGISTER From the dashboard in the homepage 2.Enter NO username/email in Email text box 3.Enter NO in password text box 4.Click on register button	Username: password:
13	RegisterPage_TC_O	Functional	Register page	verify user can submit the register form and registeration success page is displayed		1.click REGISTER From the dashboard in the homepage 2.Enter valid username/email in Email text box 3.Enter valid password text box 4.Click on register button	Username: jayasri@gmail.com password: sri123