THYROID DISEASE

CLASSIFICATION USING

MACHINE LEARNING

ABSTRACT:

With the vast amount of data and information difficult to deal with, especially in the ith the vast amount of data and information difficult to deal with, especially in the health system, machine learning algorithms and data mining techniques have an important role in dealing with data. In our study, we used machine learning algorithms with thyroid disease. The goal of this study is to categorize thyroid disease into three categories: hyperthyroidism, hypothyroidism, and normal, so we worked on this study using data from Iraqi people, some of whom have an overactive thyroid gland and others who have hypothyroidism, so we used all of the algorithms.

INTRODUCTION:

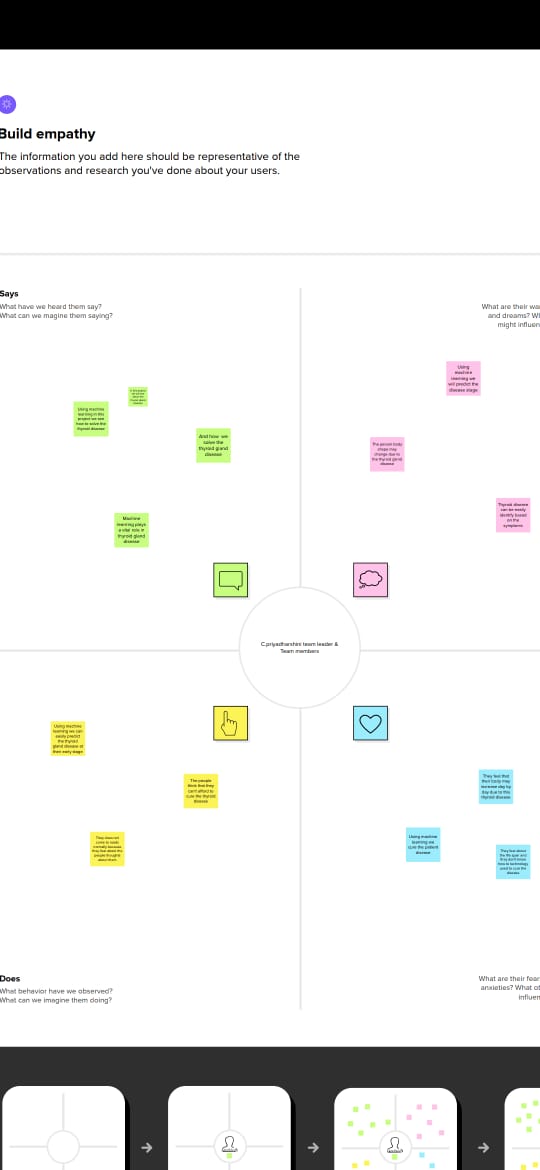
The Thyroid gland is a vascular gland and one of the most important organs of the human body. This gland secretes two hormones which help in controlling the metabolism of the body. The two types of Thyroid disorders are Hyperthyroidism and Hypothyroidism. When this disorder occurs in the body, they release certain types of hormones into the body which imbalances the body’s metabolism. A thyroid-related Blood test is used to detect this disease but it is often blurred and noise will be present. Data cleansing methods were used to make the data primitive enough for the analytics to show the risk of patients getting this disease. Machine Learning plays a very deciding role in disease prediction. Machine Learning algorithms, SVM - support vector machine, Random Forest Classifier, XGB Classifier and ANN - Artificial Neural Networks are used to predict the patient’s risk of getting thyroid disease. The web app is created to get data from users to predict the type of disease.

PURPOSE STATEMENT:

The goal of this study is to categorize thyroid disease into three categories: hyperthyroidism, hypothyroidism, and normal, so we worked on this study using data from Iraqi people, some of whom have an overactive thyroid gland and others who have hypothyroidism, so we used all of the algorithms.

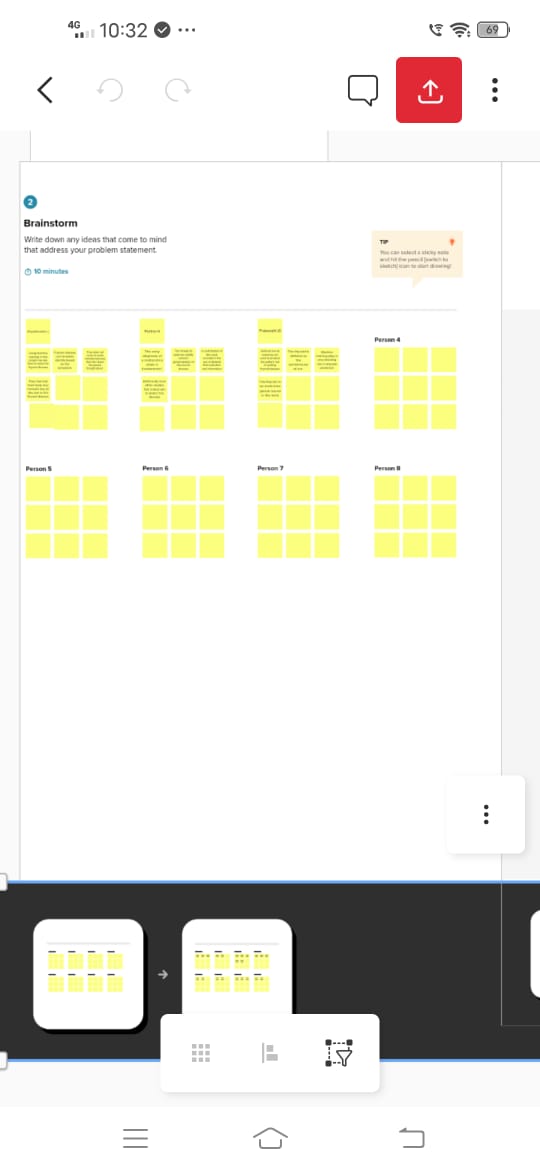
PROBLEM DEFINITION & DESIGN THINKING

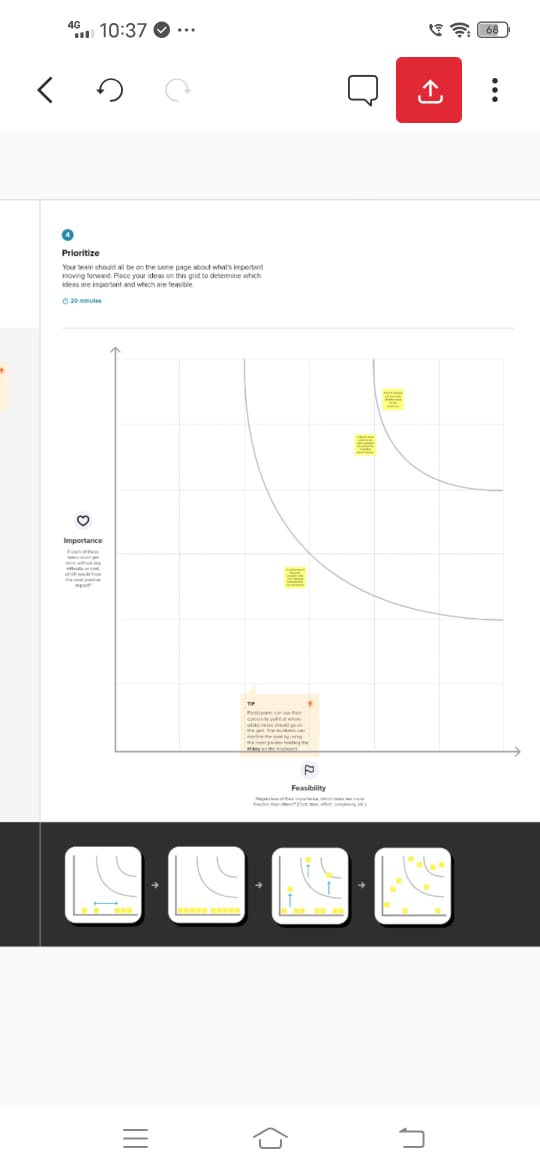
Empthy Map



In the above Mural picture our group done the map with our ideas about optimizing spam filtering with machine learning. We talk about the topic screenshot of our mural map.

BRAINSTORMING:





In the below we added screenshot of our Brainstorm and ideation. In this template we give our brainstorm ideas about our project.

RESULT

A large goiter may cause problems with swallowing or breathing. Heart problems. Hypothyroidism can lead to a higher risk of heart disease and heart failure. That's mainly because people with an underactive thyroid tend to develop high levels of low-density lipoprotein (LDL) cholesterol — the "bad" cholesterol.

ADVANTAGES & DISADVANTAGES

ADVANTAGE:

It helps to regulate many body functions by constantly releasing a steady amount of thyroid hormones into the bloodstream. If the body needs more energy in certain situations – for instance, if it is growing or cold, or during pregnancy – the thyroid gland produces more hormones.

DISAVANTAGE:

When the thyroid makes too much thyroid hormone, your body uses energy too quickly. This is called hyperthyroidism. Using energy too quickly will do more than make you tired it can make your heart beat faster, cause you to lose weight without trying and even make you feel nervous.

APPLICATIONS:

The thyroid gland is a vital hormone gland: It plays a major role in the metabolism, growth and development of the human body. It helps to regulate many body functions by constantly releasing a steady amount of thyroid hormones into the bloodstream.

FUTURE SCOPE

Classifying possible variables (both input and output) affecting the diagnosis of thyroid disease and investigating the relationships among such variables. Generating an integrated model to `diagnose the thyroid disease of a patient and strategy used by physician using an artificial neural network Estimating the values of the output variables from the given input variables and to represent the interdependencies among these variables of interest in the ANN. Using an appropriate algorithm to find the weights between individual neurons of the individual layers of the ANN. Test and refine the integrated model and cross validated with decision tree model with accuracy to diagnose the thyroid disease. Try to get rid of any feelings or prejudices for patients and act as a specialist physician for diagnosing thyroid diseases.

APPENDIX

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import tensorflow

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Layer, Dense, Dropout

data = pd.read\_csv("/content/thyroidDF.csv")

data

| **age** | **sex** | **on\_thyroxine** | **query\_on\_thyroxine** | **on\_antithyroid\_meds** | **sick** | **pregnant** | **thyroid\_surgery** | **I131\_treatment** | **query\_hypothyroid** | **...** | **TT4** | **T4U\_measured** | **T4U** | **FTI\_measured** | **FTI** | **TBG\_measured** | **TBG** | **referral\_source** | **target** | **patient\_id** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 29 | F | f | f | f | f | f | f | f | t | ... | NaN | f | NaN | f | NaN | f | NaN | other | - | 840801013 |
| **1** | 29 | F | f | f | f | f | f | f | f | f | ... | 128.0 | f | NaN | f | NaN | f | NaN | other | - | 840801014 |
| **2** | 41 | F | f | f | f | f | f | f | f | f | ... | NaN | f | NaN | f | NaN | t | 11.0 | other | - | 840801042 |
| **3** | 36 | F | f | f | f | f | f | f | f | f | ... | NaN | f | NaN | f | NaN | t | 26.0 | other | - | 840803046 |
| **4** | 32 | F | f | f | f | f | f | f | f | f | ... | NaN | f | NaN | f | NaN | t | 36.0 | other | S | 840803047 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **9167** | 56 | M | f | f | f | f | f | f | f | f | ... | 64.0 | t | 0.83 | t | 77.0 | f | NaN | SVI | - | 870119022 |
| **9168** | 22 | M | f | f | f | f | f | f | f | f | ... | 91.0 | t | 0.92 | t | 99.0 | f | NaN | SVI | - | 870119023 |
| **9169** | 69 | M | f | f | f | f | f | f | f | f | ... | 113.0 | t | 1.27 | t | 89.0 | f | NaN | SVI | I | 870119025 |
| **9170** | 47 | F | f | f | f | f | f | f | f | f | ... | 75.0 | t | 0.85 | t | 88.0 | f | NaN | other | - | 870119027 |
| **9171** | 31 | M | f | f | f | f | f | f | f | t | ... | 66.0 | t | 1.02 | t | 65.0 | f | NaN | other | - | 870119035 |

9172 rows × 31 columns

data.shape

(9172, 31)

data.isnull().sum()

diagnoses={'A':'hyperthroid conditions',

           'B':'hyperthyroid conditions',

           'C':'hyperthyroid conditions',

           'D':'hyperthyroid conditions',

           'E':'hyperthyroid conditions',

           'F':'hyperthyroid conditions',

           'G':'hyperthyroid conditions',

           'H':'hyperthyroid conditions',

           'I':'binding protein',

           'j':'binding protien',

           'k':'general health',

           'L':'replacement therapy',

           'M':'replacement therapy',

           'N':'replacement therapy',

           'O':'antithyroid treatment',

           'P':'antithyroid treatment',

           'Q':'antithyroid treatment',

           'R':'miscellaneous',

           'S':'miscellaneous',

           'T':'miscellaneous'}

data['target']= data['target'].map(diagnoses)

data.dropna(subset=['target'],inplace=True)

data['target'].value\_counts()

hyperthyroid conditions 628

binding protein 346

replacement therapy 336

miscellaneous 281

hyperthroid conditions 147

antithyroid treatment 33

Name: target, dtype: int64

data['target'].isnull().sum()

0

data.head()

| **age** | **sex** | **on\_thyroxine** | **query\_on\_thyroxine** | **on\_antithyroid\_meds** | **sick** | **pregnant** | **thyroid\_surgery** | **I131\_treatment** | **query\_hypothyroid** | **...** | **TT4** | **T4U\_measured** | **T4U** | **FTI\_measured** | **FTI** | **TBG\_measured** | **TBG** | **referral\_source** | **target** | **patient\_id** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **4** | 32 | F | f | f | f | f | f | f | f | f | ... | NaN | f | NaN | f | NaN | t | 36.0 | other | miscellaneous | 840803047 |
| **18** | 63 | F | t | f | f | t | f | f | f | f | ... | 48.0 | t | 1.02 | t | 47.0 | f | NaN | other | hyperthyroid conditions | 840815067 |
| **32** | 41 | M | f | f | f | f | f | f | f | f | ... | 39.0 | t | 1.00 | t | 39.0 | f | NaN | other | miscellaneous | 840816013 |
| **33** | 71 | F | t | f | f | f | f | f | f | f | ... | 126.0 | t | 1.38 | t | 91.0 | f | NaN | other | binding protein | 840816014 |
| **39** | 55 | F | t | f | f | f | f | f | f | t | ... | 136.0 | t | 1.48 | t | 92.0 | f | NaN | other | replacement therapy | 840816047 |

5 rows × 31 columns

data.describe()

ageTSHT3TT4T4UFTITBGpatient\_idcount1771.0000001625.0000001198.0000001676.0000001607.0000001608.00000097.0000001.771000e+03mean49.81140618.7041172.379299125.1280791.069266124.76837147.9515468.531494e+08std19.19046751.7415601.48783564.2041560.28182178.53879332.4835647.531264e+06min1.0000000.0050000.0500002.0000000.2800001.4000009.2999998.408030e+0825%34.0000000.2100001.50000082.0000000.90000080.00000032.0000008.504165e+0850%52.0000003.2000002.100000125.0000001.000000110.00000036.0000008.510240e+0875%65.00000012.0000003.000000165.0000001.190000165.00000047.0000008.607180e+08max91.000000530.00000018.000000600.0000002.330000881.000000200.0000008.701190e+08

data[data.age>100]

| **age** | **sex** | **on\_thyroxine** | **query\_on\_thyroxine** | **on\_antithyroid\_meds** | **sick** | **pregnant** | **thyroid\_surgery** | **I131\_treatment** | **query\_hypothyroid** | **...** | **TT4** | **T4U\_measured** | **T4U** | **FTI\_measured** | **FTI** | **TBG\_measured** | **TBG** | **referral\_source** | **target** | **patient\_id** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

0 rows × 31 columns

data

| **age** | **sex** | **on\_thyroxine** | **query\_on\_thyroxine** | **on\_antithyroid\_meds** | **sick** | **pregnant** | **thyroid\_surgery** | **I131\_treatment** | **query\_hypothyroid** | **...** | **TT4** | **T4U\_measured** | **T4U** | **FTI\_measured** | **FTI** | **TBG\_measured** | **TBG** | **referral\_source** | **target** | **patient\_id** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **4** | 32 | F | f | f | f | f | f | f | f | f | ... | NaN | f | NaN | f | NaN | t | 36.0 | other | miscellaneous | 840803047 |
| **18** | 63 | F | t | f | f | t | f | f | f | f | ... | 48.0 | t | 1.02 | t | 47.0 | f | NaN | other | hyperthyroid conditions | 840815067 |
| **32** | 41 | M | f | f | f | f | f | f | f | f | ... | 39.0 | t | 1.00 | t | 39.0 | f | NaN | other | miscellaneous | 840816013 |
| **33** | 71 | F | t | f | f | f | f | f | f | f | ... | 126.0 | t | 1.38 | t | 91.0 | f | NaN | other | binding protein | 840816014 |
| **39** | 55 | F | t | f | f | f | f | f | f | t | ... | 136.0 | t | 1.48 | t | 92.0 | f | NaN | other | replacement therapy | 840816047 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **9138** | 76 | M | f | f | f | f | f | f | f | f | ... | 122.0 | t | 0.94 | t | 130.0 | f | NaN | SVI | hyperthyroid conditions | 870115042 |
| **9142** | 15 | F | f | f | f | f | f | f | f | f | ... | 92.0 | t | 0.96 | t | 96.0 | f | NaN | other | hyperthyroid conditions | 870116005 |
| **9149** | 75 | F | t | f | f | f | f | f | f | t | ... | 54.0 | t | 1.03 | t | 53.0 | f | NaN | SVI | hyperthyroid conditions | 870116038 |
| **9162** | 36 | F | f | f | f | f | f | f | f | f | ... | 84.0 | t | 1.26 | t | 67.0 | f | NaN | other | binding protein | 870119008 |
| **9169** | 69 | M | f | f | f | f | f | f | f | f | ... | 113.0 | t | 1.27 | t | 89.0 | f | NaN | SVI | binding protein | 870119025 |

1771 rows × 31 columns

x=data.iloc[:,0:-1]

y=data.iloc[:,-1]

data.isnull().sum()

age 0

sex 76

on\_thyroxine 0

query\_on\_thyroxine 0

on\_antithyroid\_meds 0

sick 0

pregnant 0

thyroid\_surgery 0

I131\_treatment 0

query\_hypothyroid 0

query\_hyperthyroid 0

lithium 0

goitre 0

tumor 0

hypopituitary 0

psych 0

TSH\_measured 0

TSH 146

T3\_measured 0

T3 573

TT4\_measured 0

TT4 95

T4U\_measured 0

T4U 164

FTI\_measured 0

FTI 163

TBG\_measured 0

TBG 1674

referral\_source 0

target 0

patient\_id 0

dtype: int64

x

x['sex'].unique()

array(['F', 'M', nan], dtype=object)

x['sex'].replace(np.nan,'F',inplace=True)

x['sex'].value\_counts()

F 1428

M 343

Name: sex, dtype: int64

x.isnull().sum()

age 0

sex 0

on\_thyroxine 0

query\_on\_thyroxine 0

on\_antithyroid\_meds 0

sick 0

pregnant 0

thyroid\_surgery 0

I131\_treatment 0

query\_hypothyroid 0

query\_hyperthyroid 0

lithium 0

goitre 0

tumor 0

hypopituitary 0

psych 0

TSH\_measured 0

TSH 146

T3\_measured 0

T3 573

TT4\_measured 0

TT4 95

T4U\_measured 0

T4U 164

FTI\_measured 0

FTI 163

TBG\_measured 0

TBG 1674

referral\_source 0

target 0

dtype: int64

data.info()

<class 'pandas.core.frame.DataFrame'>

Int64Index: 1771 entries, 4 to 9169

Data columns (total 31 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 age 1771 non-null int64

1 sex 1695 non-null object

2 on\_thyroxine 1771 non-null object

3 query\_on\_thyroxine 1771 non-null object

4 on\_antithyroid\_meds 1771 non-null object

5 sick 1771 non-null object

6 pregnant 1771 non-null object

7 thyroid\_surgery 1771 non-null object

8 I131\_treatment 1771 non-null object

9 query\_hypothyroid 1771 non-null object

10 query\_hyperthyroid 1771 non-null object

11 lithium 1771 non-null object

12 goitre 1771 non-null object

13 tumor 1771 non-null object

14 hypopituitary 1771 non-null object

15 psych 1771 non-null object

16 TSH\_measured 1771 non-null object

17 TSH 1625 non-null float64

18 T3\_measured 1771 non-null object

19 T3 1198 non-null float64

20 TT4\_measured 1771 non-null object

21 TT4 1676 non-null float64

22 T4U\_measured 1771 non-null object

23 T4U 1607 non-null float64

24 FTI\_measured 1771 non-null object

25 FTI 1608 non-null float64

26 TBG\_measured 1771 non-null object

27 TBG 97 non-null float64

28 referral\_source 1771 non-null object

29 target 1771 non-null object

30 patient\_id 1771 non-null int64

dtypes: float64(6), int64(2), object(23)

memory usage: 442.8+ KB

x.info()

<class 'pandas.core.frame.DataFrame'>

Int64Index: 1771 entries, 4 to 9169

Data columns (total 30 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 age 1771 non-null int64

1 sex 1771 non-null object

2 on\_thyroxine 1771 non-null object

3 query\_on\_thyroxine 1771 non-null object

4 on\_antithyroid\_meds 1771 non-null object

5 sick 1771 non-null object

6 pregnant 1771 non-null object

7 thyroid\_surgery 1771 non-null object

8 I131\_treatment 1771 non-null object

9 query\_hypothyroid 1771 non-null object

10 query\_hyperthyroid 1771 non-null object

11 lithium 1771 non-null object

12 goitre 1771 non-null object

13 tumor 1771 non-null object

14 hypopituitary 1771 non-null object

15 psych 1771 non-null object

16 TSH\_measured 1771 non-null object

17 TSH 1625 non-null float64

18 T3\_measured 1771 non-null object

19 T3 1198 non-null float64

20 TT4\_measured 1771 non-null object

21 TT4 1676 non-null float64

22 T4U\_measured 1771 non-null object

23 T4U 1607 non-null float64

24 FTI\_measured 1771 non-null object

25 FTI 1608 non-null float64

26 TBG\_measured 1771 non-null object

27 TBG 97 non-null float64

28 referral\_source 1771 non-null object

29 target 1771 non-null object

dtypes: float64(6), int64(1), object(23)

memory usage: 428.9+ KB

x['age']=x['age'].astype('float')  
x['TSH']=x['TSH'].astype('float')  
x['T3']=x['T3'].astype('float')  
x['TT4']=x['TT4'].astype('float')  
x['T4U']=x['T4U'].astype('float')  
x['FTI']=x['FTI'].astype('float')  
x['TBG']=x['TBG'].astype('float')

from sklearn.preprocessing import OrdinalEncoder,LabelEncoder  
ordinal\_encoder = OrdinalEncoder(dtype='int64')  
x.iloc[:, 1:16] = ordinal\_encoder.fit\_transform(x.iloc[:, 1:16])

<ipython-input-23-c93efbd70f82>:3: DeprecationWarning: In a future version, `df.iloc[:, i] = newvals` will attempt to set the values inplace instead of always setting a new array. To retain the old behavior, use either `df[df.columns[i]] = newvals` or, if columns are non-unique, `df.isetitem(i, newvals)`

x.iloc[:, 1:16] = ordinal\_encoder.fit\_transform(x.iloc[:, 1:16])

x.head()

x

x.replace(np.nan,'0',inplace=True)

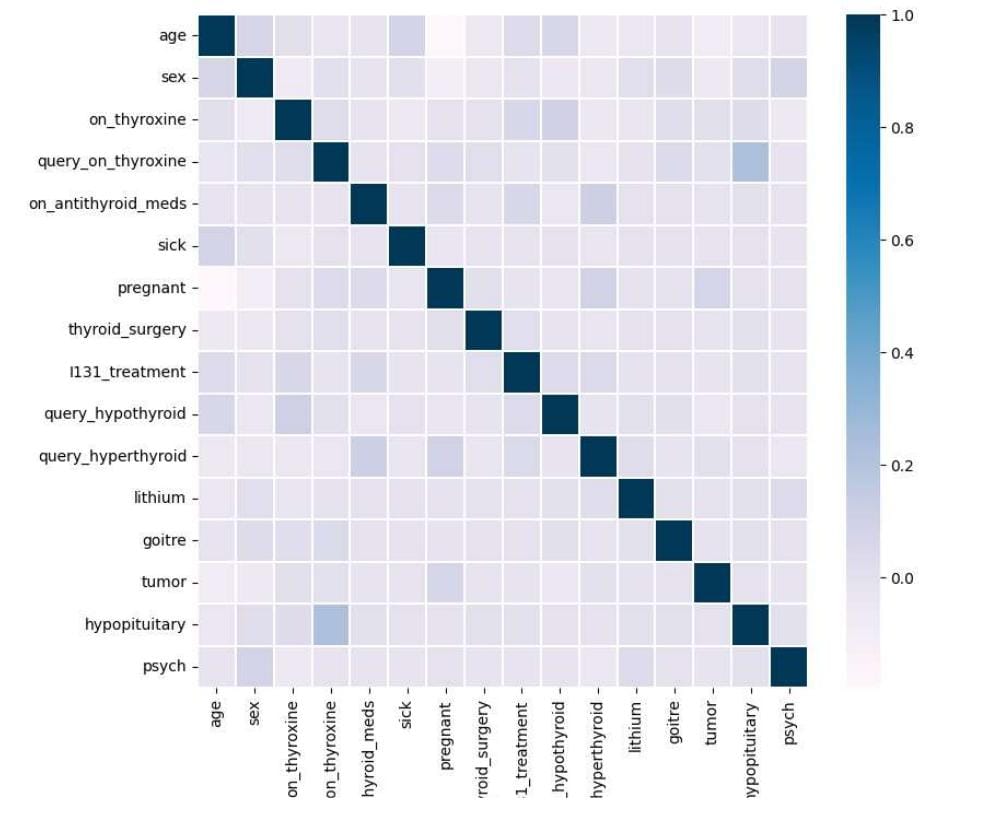
x

label\_encoder = LabelEncoder()  
y\_dt=label\_encoder.fit\_transform(y)

y=pd.DataFrame(y\_dt,columns=['target'])

y

  import seaborn as sns  
  cormat = x.corr()  
  f,ax = plt.subplots(figsize=(9,8))  
  sns.heatmap(cormat,ax = ax,cmap ="PuBu",linewidths = 0.1)

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

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.20,random\_state=0)

y\_train.value\_counts()

target

0 1

1224 1

1191 1

1190 1

1189 1

..

603 1

602 1

601 1

600 1

1769 1

Length: 1416, dtype: int64

[ ]

from imblearn.over\_sampling import SMOTE

os = SMOTE(random\_state=0,k\_neighbors=1)

x\_bal,y\_bal=os.fit\_resample(x\_train,y\_train)

x\_test\_bal,y\_test\_bal=os.fit\_resample(x\_test,y\_test)

from imblearn.over\_sampling import SMOTE

os = SMOTE(random\_state=0,k\_neighbors=1)

x\_bal,y\_bal=os.fit\_resample(x\_train,y\_train)

x\_test\_bal,y\_test\_bal=os.fit\_resample(x\_test,y\_test)

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

x\_bal = sc.fit\_transform(x\_bal)

x\_test\_bal= sc.transform(x\_test\_bal)

x\_bal

columns=['age','sex','on\_thyroxine','query\_on\_thyroxine','on\_antithyroid\_meds','sick','pregnant','thyroid\_surgery','I131\_treatment']

x\_test\_bal

x\_bal= pd.DataFrame(x\_bal,columns=columns)

x\_bal

from sklearn.inspection import permutation\_importance

results = permutation\_importance(rfr,x\_bal,y\_bal,scoring='accuracy')

feature\_importance=['age','sex','on\_thyroxine','query\_on\_thyroid\_meds','sick','pregnant','thyroid\_surgery','I131\_treament']

importance = results.importances\_mean

importance = np.sort(importance)

for i,v in enumerate(importance)

  i=feature\_importance[i]

  print('feature:{:<20}Score:{}'.format(i,v))

  plt.figure(figsize(10,10))

  plt.bar(x=feature\_importance,height=importance)

  plt.xticks(rotation=30,ha='right')

  plt.show()

x.head()

| **age** | **sex** | **on\_thyroxine** | **query\_on\_thyroxine** | **on\_antithyroid\_meds** | **sick** | **pregnant** | **thyroid\_surgery** | **I131\_treatment** | **query\_hypothyroid** | **...** | **TT4\_measured** | **TT4** | **T4U\_measured** | **T4U** | **FTI\_measured** | **FTI** | **TBG\_measured** | **TBG** | **referral\_source** | **target** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **4** | 32.0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | f | 0 | f | 0 | f | 0 | t | 36.0 | other | miscellaneous |
| **18** | 63.0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | ... | t | 48.0 | t | 1.02 | t | 47.0 | f | 0 | other | hyperthyroid conditions |
| **32** | 41.0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | t | 39.0 | t | 1.0 | t | 39.0 | f | 0 | other | miscellaneous |
| **33** | 71.0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | t | 126.0 | t | 1.38 | t | 91.0 | f | 0 | other | binding protein |
| **39** | 55.0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | ... | t | 136.0 | t | 1.48 | t | 92.0 | f | 0 | other | replacement therapy |

5 rows × 30 columns

x\_bal.head()

data.info()

<class 'pandas.core.frame.DataFrame'>

Int64Index: 1771 entries, 4 to 9169

Data columns (total 31 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 age 1771 non-null int64

1 sex 1695 non-null object

2 on\_thyroxine 1771 non-null object

3 query\_on\_thyroxine 1771 non-null object

4 on\_antithyroid\_meds 1771 non-null object

5 sick 1771 non-null object

6 pregnant 1771 non-null object

7 thyroid\_surgery 1771 non-null object

8 I131\_treatment 1771 non-null object

9 query\_hypothyroid 1771 non-null object

10 query\_hyperthyroid 1771 non-null object

11 lithium 1771 non-null object

12 goitre 1771 non-null object

13 tumor 1771 non-null object

14 hypopituitary 1771 non-null object

15 psych 1771 non-null object

16 TSH\_measured 1771 non-null object

17 TSH 1625 non-null float64

18 T3\_measured 1771 non-null object

19 T3 1198 non-null float64

20 TT4\_measured 1771 non-null object

21 TT4 1676 non-null float64

22 T4U\_measured 1771 non-null object

23 T4U 1607 non-null float64

24 FTI\_measured 1771 non-null object

25 FTI 1608 non-null float64

26 TBG\_measured 1771 non-null object

27 TBG 97 non-null float64

28 referral\_source 1771 non-null object

29 target 1771 non-null object

30 patient\_id 1771 non-null int64

dtypes: float64(6), int64(2), object(23)

memory usage: 442.8+ KB

from sklearn.ensemble import RandomForestClassifier  
rfr1 = RandomForestClassifier().fit(x\_os,y\_os.values.ravel())  
y\_pred = rfr1.predict(x\_test\_os)  
rfr1 = RandomForestClassifier()

rfr1.fit(x\_os,y\_os.values.ravel())

y\_pred = rfr1.predict(x\_test\_os)

y\_pred = rfr1.predict(x\_test\_os)

train\_score = accuracy\_score(y\_os,rfr1.predict(x\_0s))

train\_score

print(classification\_report(y\_test\_os,y\_pred))

from xghoost import XBGclassifier

xgb1 = XGBClassifier()

xgb1.fit(x\_os,y\_os)

y\_pred = xgb1.predict(x\_test\_os)

print(classification\_report(y\_test\_os,y\_pred))

accuracy\_score(y\_test\_os,y\_pred)

sv.fit(x\_bal,y\_bal)

y\_pred = sv.predict(x\_test\_bal)

print(classification\_report(y\_test\_bal,y\_pred))

train\_score=accuracy\_score(y\_bal,sv.predict(x\_bal))

train\_score

model = Sequential()

model.add(Dense(units = 128,activation='relu',input\_shape=(10,)))

model.add(Dense(units = 128,activation='relu',kernel\_initializer='random\_uniform'))

model.add(Dropout(0.2))

model.add(Dense(units=256,activation='relu',kernel\_initializer='random\_uniform'))

model.add(Dropout(0.2))

model.add(Dense(units=128,activation='relu',kernel\_initializer='random\_uniform'))

model.summary()

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

dense (Dense) (None, 128) 1408

dense\_1 (Dense) (None, 128) 16512

dropout (Dropout) (None, 128) 0

dense\_2 (Dense) (None, 256) 33024

dropout\_1 (Dropout) (None, 256) 0

dense\_3 (Dense) (None, 128) 32896

dense\_4 (Dense) (None, 1) 129

=================================================================

Total params: 83,969

Trainable params: 83,969

Non-trainable params: 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

model.compile(loss='binary\_crossentropy',optimizer='adam',metrics=['accuracy'])

model.fit(x\_bal,y\_bal,validation\_data=[x\_test\_bal,y\_test\_bal]),epochs=15)

rfr1.predict([[0,0,0,0,0.000000,0.0,0.0,1.00,0.0,40.0]])

CodeText

sv.predict([[0,0,0,0,0.000000,0.0,0.0,1.00,0.0,40.0]])

col = ['goitre','tumor','hypopiturary','psych','TSH','T3','TT4','T4U','FTI','TBG']

da = [[0,0,0,0,0.000000,0.0,0.0,1.00,0.0,40.0]]

da1 = pd.DataFrame(data = da,columns=col)

xgb1.predict(dal)

model.predict([[0,0,0,0,0.000000,0.0,0.0,1.00,0.0,40.0]])

print(classification\_report(y\_test\_bal,y\_pred))

train\_score = accuracy\_score(y\_bal,rfr1.predict(x\_bal))

train\_score

y\_pred=xgb.predict(x\_test\_bal)

print(classification\_report(y\_test\_bal,y\_pred))

accuracy\_score(y\_test\_bal,y\_pred)

params ={

    'c':[0.1,1,10,100,1000],

     'gamma':[1,0.1,0.01,0.001,0.0001],

     'kernal':['rbf','sqrt']

}

random\_svc = RandomizedSearchCV(sv,params,scoring='accuracy',cv=5,n\_jobs=1)

random\_svc.fit(x\_bal,y\_bal)

random\_svc.best\_params\_

sv1=SVC(kernel= 'rbf',gamma=0.1,c=100)

sv1.fit(x\_bal,y\_bal)

y\_pred= sv1.predict(x\_test\_bal)

print(classification\_report(y\_test\_bal,y\_pred))

train\_score=accuracy\_score(y\_bal,sv1.predict(x\_bal))

train\_score

import pickle

pickle.dump(sv1,open('thyroid\_1\_model.pkl','wb'))

features=np.array([[0,0,0,0,0,000000,0.0,0.0,1.00,0.0,40.0]])

print(label\_encoder.inverse\_transform(xgb1.predict(features)))

pickle.dump(label\_encoder,open('label\_encoder.pkl','wb'))

data['target'].unique()

y['target'].unique()

array([ 0, 1, 2, ..., 1768, 1769, 1770])

import pickle

pickle.dump(sv1,open('thyroid\_1\_model.pkl','wb'))

from flask import Flask,render\_template,request

import numpy as np

import pickle

import pandas as pd

model=pickle.load

le=pickle.load

app=Flask(\_name)

def about():

  return render\_template('home.html')

def predict():

  x=[[float(x)for x in request.form.values()]]

  print(x)

  col=['goitre','tumor','hypopitutiary','psych','TSH','T3','TT4','T4U','FTI','TGB']

  x=pd.DataFrame(x,columns=col)

  print(x)

  pred=model.predict(X)

  pred=le.inverse\_transform(pred)

  print(pred[0])

  return render\_template('sumbit.html',prediction\_text=str(pred))

if\_\_name\_\_=="\_\_main\_\_":

  app.run(debug=False)