Thyroid Disease Classification Using ML

Project report submitted to
"Madurai Kamaraj University"
in partial fulfillment of the requirements
for the award of the Degree of

Bachelor of Science
in
Computer Science

Submitted

By

TEAM ID : NM2023TMID31087

TEAM LEAD : B. YUVARAJAN

Under the Guidance of

Dr. B. UMADEVI M.Sc.,M.Phil.,Ph.D.,

(Assistant Professor, GAC, Melur)

GOVERNMENT ARTS COLLEGE
P.G & DEPARTMENT OF COMPUTER SCIENCE
MELUR – 625 106

INDEX

1. INTRODUCTION

1.1 OVERVIEW

Thyroiddiseases,suchas hypothyroid Is mandhyperthyroidism, arecommonendocrinedisordersthataffectthefunctionofthethyroidgland. Thesediseasescan haveasignificantimpactonapatient's healthandqualityoflife.Early and accurate diagnosis of thyroid diseasesisimportantfor effective treatment.

Inrecentyears,machinelearning techniqueshave beenappliedtotheclassificationofthyroiddiseases.The goalofthese studiesistodevelop models thatcanaccurately diagnose thyroid diseases basedonclinicalandlaboratorydata.

1.2 PURPOSE

There are several machine learning algorithms that have been used for thyroid disease classification,including decision trees, random forests, k-nearest neighbors (KNN), support vector machines (SVM),artificial neural networks (ANN), and deep learning algorithms such as convolutional neural networks(CNNs)andrecurrentneuralnetworks(RNNs).

The input data for these models can include clinical features, such as age, gender, and symptoms, as wellas laboratory test results, such as thyroid-stimulating hormone (TSH) levels and levels of thyroxine (T4)andtriiodothyronine(T3).

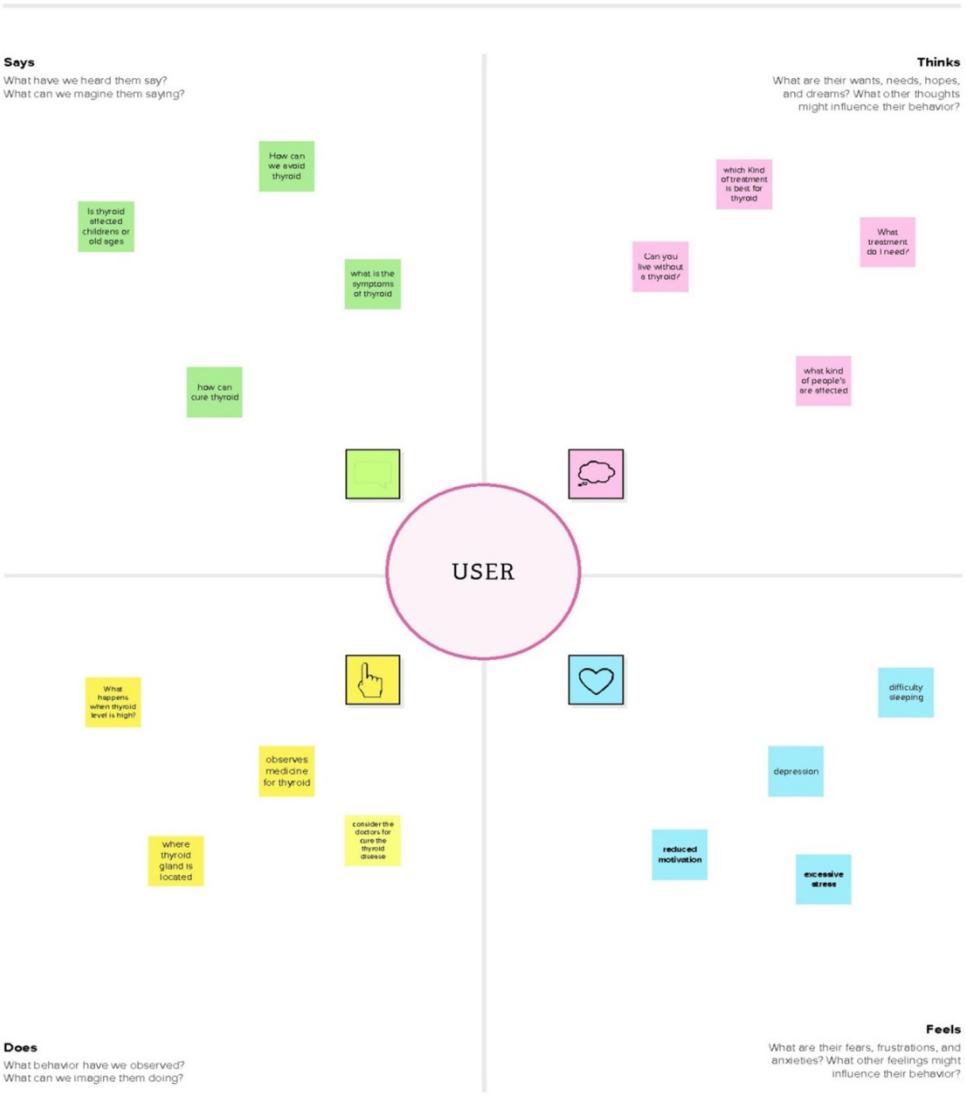
The performance of these models is usually evaluated using metrics such as accuracy, precision, recall,and F1 score. In general, deep learning algorithms have shown better performance than other machinelearningalgorithmsinthyroiddiseaseclassificationtasks.

PROBLEM DEFINITION & DESIGN THINKING

EMPATHY MAP:`

:

Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.

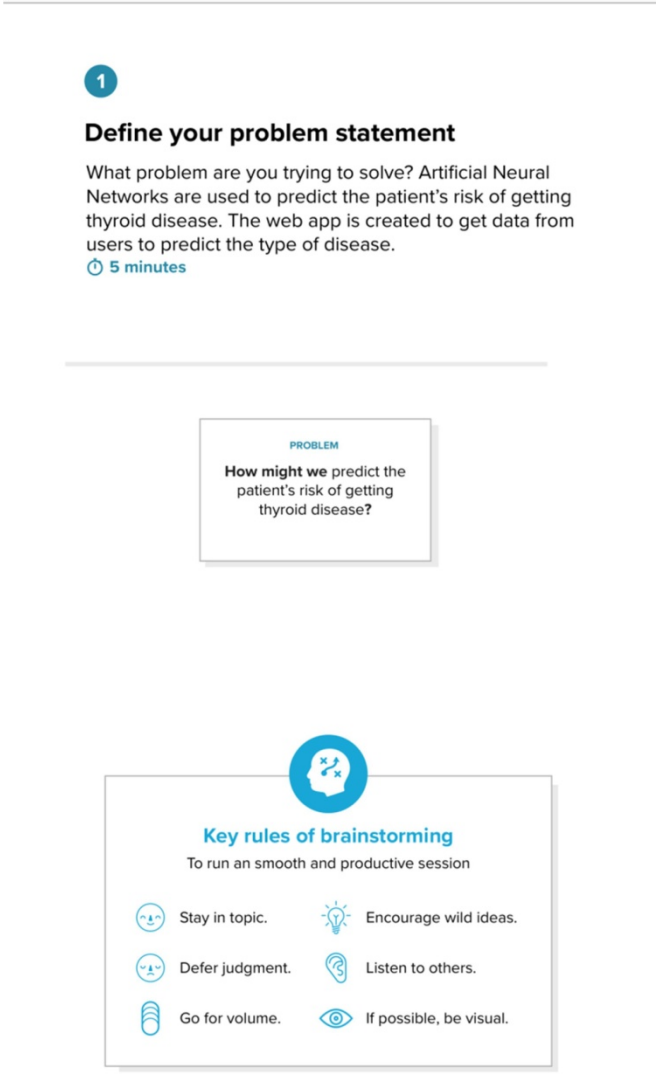


.

IDEATION & BRAINSTORMING MAP

BRAINSTORMING MAP SCREENSHOT:

Use this template in your own brainstorming sessions so your team can unleashtheir imagination and start shaping concepts even if you're not sitting in the same



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

VASANTHAKUMAR.M

- where thyroid gland is located
- Early diagnosis of thyroid disease is crucial to prevent complications and improve outcomes.
- Deep learning techniques can be applied to large datasets of patient information
- Imaging analysis techniques can be used

SEETHARAM PANDIAN.S

- Is thyroid affected childrens or old ages
- What happens when thyroid level is high?
- Patient demographic data such as age, gender, ethnicity, and location can be included in a thyroid disease dataset.
- How can we avoid thyroid

SHANMUGA SUNDAR.M

- Online forums and communities discussing thyroid disease Advertisements for thyroid supplements or treatments
- what is the symptoms of thyroid
- what kind of people's are affected
- Use patient data to predict thyroid disease risk

SIVARAJ.M

- Medical charts and test results Images of the thyroid gland and surrounding tissues
- Feature selection techniques can be used to identify the most relevant patient features that are predictive of thyroid disease.
- collect large datasets of patient
- using ML models to improve the accuracy of thyroid disease prediction

SURESH KUMAR.C

- Conduct physical exams and order laboratory tests Analyze and interpret medical data to make a diagnosis Prescribe medications and other treatments
- how can cure thyroid
- By developing predictive models for specific types of thyroid disease (such as hypothyroidism, hyperthyroidism, or thyroid cancer)
- Trained on patient data, including medical history, hormone levels, and imaging data, to predict the likelihood of developing thyroid

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Exploratory Data Analysis

Descriptive Analysis

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas have a worthy function called describe. With this described function we can find mean, std, min, max and percentile values of continuous features.

Visual Analysis

Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions.

Checking Correlation

Here, I'm finding the correlation using HeatMap. It visualizes the data in 2-D coloured maps making use of colour variations. It describes the related variables in the form of colours instead of numbers; it will be plotted on both axes.

Model Building

Training The Model In Multiple Algorithms

Now our data is cleaned and it's time to build the model. We can train our data on different algorithms. For this project we are applying four classification algorithms. The best model is saved based on its performance.

Random Forest Classifier Model

A function named Random Forest Classifier Model is created and train and test data are passed as the parameters. Inside the function, the Random Forest Classifier algorithm is initialized and training data is passed to the model with the .fit() function. Test data is predicted with the .predict() function and saved in a new variable. For evaluating the model, accuracy_score and classification report is done.

XGBClassifier Model

A function named XGBClassifier model is created and train and test data are passed as the parameters. Inside the function, the XGBClassifier algorithm is initialized and training data is passed to the model with the .fit() function. Test data is predicted with the .predict() function and saved in a new variable. For evaluating the model, the accuracy score and classification report is done.

SVC Model

A function named SVC model is created and train and test data are passed as the parameters. Inside the function, the SVC algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with the .predict() function and saved in a new variable. For evaluating the model, the accuracy score and classification report is done.

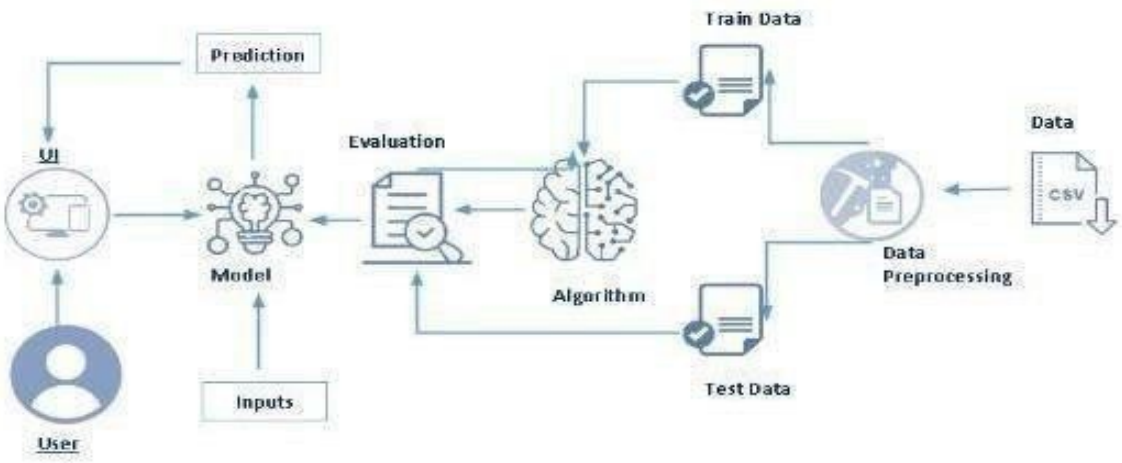
ANN Model

Artificial Neural Networks (ANN) are multi-layer fully-connected neural nets. They consist of an input layer, multiple hidden layers, and an output layer. Every node in one layer is connected to every other node in the next layer. We make the network deeper by increasing the number of hidden layers


By developing predictive models for specific types of thyroid disease (such as hypothyroidism, hyperthyroidism, or thyroid cancer)
Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data.
collect large datasets of patients
Identification of patient risk factors: Identification of patient risk factors, such as family history or exposure to radiation, could help target high-risk populations for early detection and treatment.
can improve diagnostic accuracy and reduce the need for more invasive tests.
Feature selection techniques can be used to identify the most relevant patient features that are predictive of thyroid disease.
Development of low-cost diagnostic toolssuch as smartphone apps or portable devices, could improve access to thyroid disease prediction and treatment.
Early diagnosis of thyroid disease is crucial to prevent complications and improve outcomes.
Trained on patient data, including medical history, hormone levels, and imaging data, to predict the likelihood of developing thyroid disease
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

3.THEORITICAL ANALYSIS

3.1 BLOCKDIAGRAM



3.2 HARDWAREANDSOFTWAREDESIGNING

Python

Pythonisaninterpreted,object-oriented,high-levelprogramminglanguagewithdynamicsemantics. It was created by Guido van Rossum, and first released on February 20, 1991. Itshigh-level built-in data structures, combined with dynamic typing and dynamic binding, make itvery attractive for Rapid Application Development, as well as for use as a scripting or gluelanguagetoconnectexistingcomponentstogether.Python'ssimple,easytolearnsyntaxe mphasizes readability and therefore reduces the cost of program maintenance.

Python supportsmodules and packages, which encourages program modularity and code reuse. The Pythoninterpreter and the extensive standard library are available in source or binary form withoutchargeforallmajorplatforms,andcanbefreelydistributed.

AnacondaNavigator

AnacondaNavigatorisafreeandopen-sourcedistributionofthePythonandRprogramminglanguagesfor data science and machine learning related applications. It can be installed on Windows, Linux, andmacOS. Conda is

an open-source, cross platform,packagemanagement system. AnacondacomeswithsoverynicetoolslikeJupyterLab,JupyterNotebook,QtConsole,Spyder, Glueviz,Orange,Rstudio,VisualStudioCode.Forthisproject,wewillbeusingJupyternotebook andSpyder.

JupyterNotebook

The Jupyter Notebook is an open-source web application that you can use to create and sharedocumentsthatcontainlivecode,equations,visualizations,andtext.JupyterNotebookism aintained by the people at Project Jupyter. Jupyter Notebooks are a spin-off project from theIPythonproject,whichusedtohaveanIPythonNotebookprojectitself.Thename,Jupyter,co mesfrom the core supported programming languages that it supports: Julia, Python, and R. Jupyterships with the IPython kernel, which allows you to write your programs in Python, but there arecurrentlyover100otherkernelsthatyoucanalsouse.

Spyder

Spyder,theScientificPythonDevelopmentEnvironment,isafreeintegrateddevelopmentenvir onment(IDE)thatisincludedwithAnaconda.Itincludesediting,interactivetesting,debugging, and introspection features. Initially created and developed by Pierre Raybaut in 2009,since2012Spyderhasbeenmaintained andcontinuouslyimprovedbya teamofscientificPythondevelopers and the community. Spyderis extensible with first-party and third party pluginsincludessupport for interactive tools for data inspection and embeds Python specific code. Spyderisalsopre-installedinAnaconda Navigator,whichisincludedinAnaconda.

Flask

Web frame work used for building. It is a web application framework written in python whichwill be running in local browser with a user interface. In this application, whenever the
userinteractswithUIandselectsemoji,itwillsuggestthebestandtopmoviesofthatgenretotheuse.

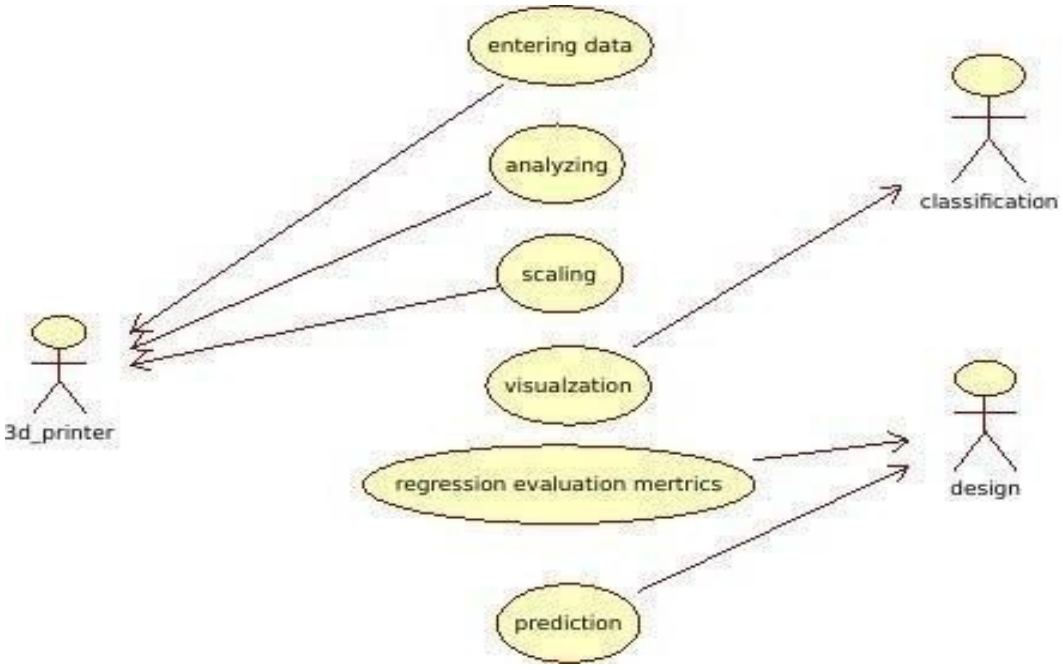
HardwareRequirements:

- Operatingsystem:window7andabovewith64bitoProcessorType-IntelCorei3-3220
- RAM:4Gbandabove
- Harddisk:min100GB

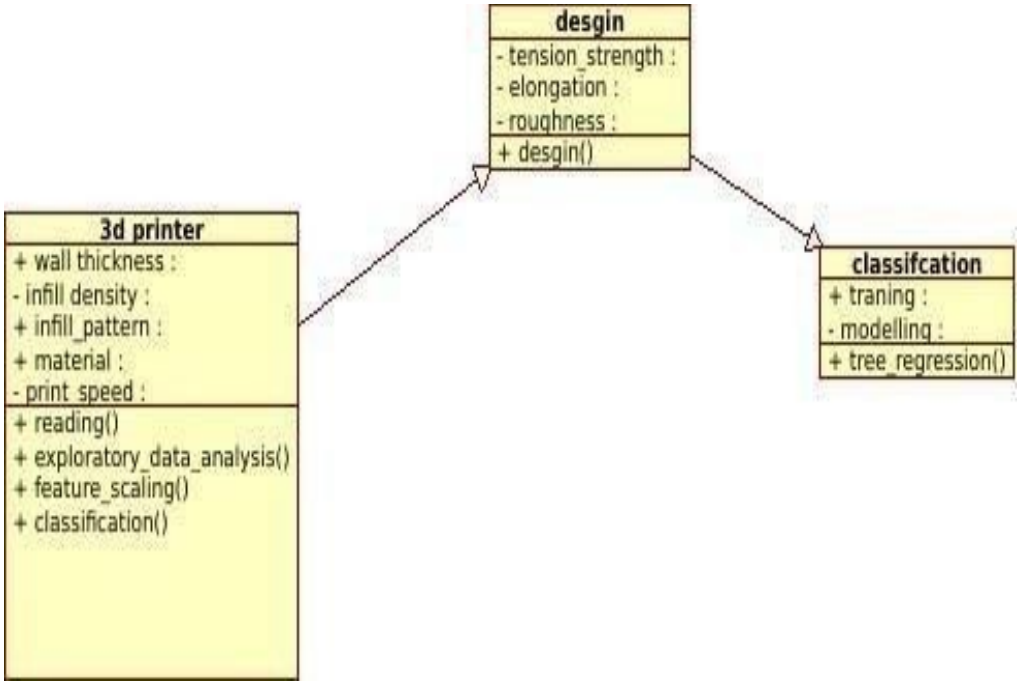
4. EXPERIMENTALINVESTIGATION

Here we are going to build a machine learning model that predicts whether the given message is aspamornot,basedontheseparametersasupervisedmachinelearningmodelisbuilt topredictthebestMaterialtobeusedfor building3Dmodels.Awebapplicationisbuildsothattheuser cantypeinthementioned partametersand thematerialwhichsuitsthe bestisshowcasedonUI.

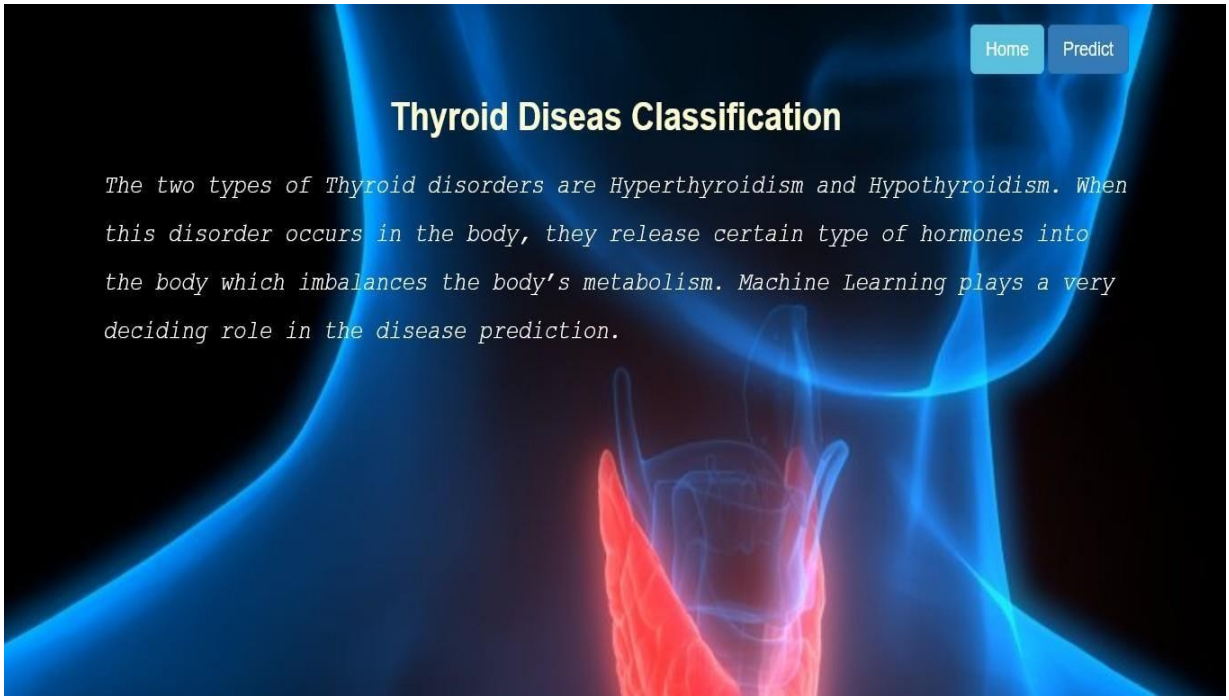
5.FLOWCHART



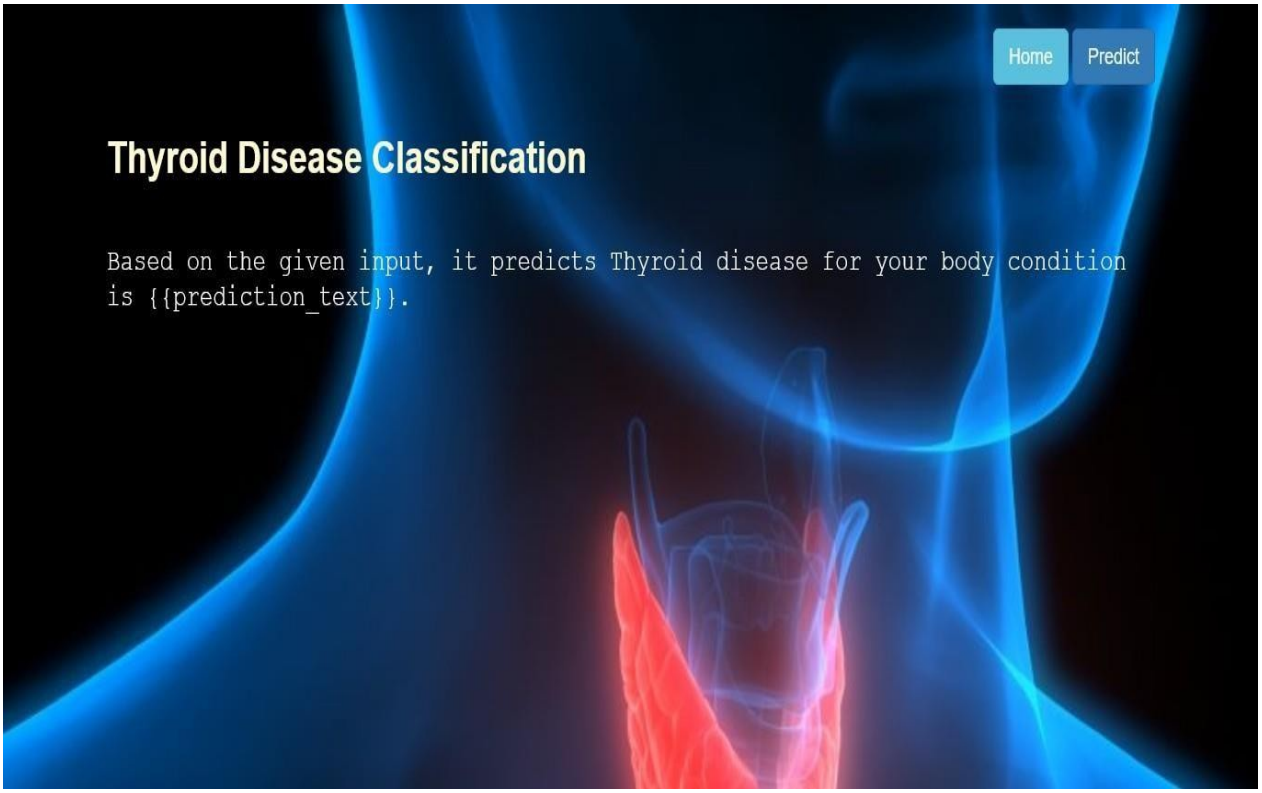
USECASEDIAGRAM



6. RESULT


Home
Predict
Thyroid Diseas Classification
The two types of Thyroid disorders are Hyperthyroidism and Hypothyroidism. When this disorder occurs in the body, they release certain type of hormones into the body which imbalances the body's metabolism. Machine Learning plays a very deciding role in the disease prediction.


Home
Predict
Thyroid Disease Classification
goitre
Male
tumor
Male
hypopituitary
Male
psych
Male
TSH
TSH
T3
T3
TT4
TT4
T4U
T4U
FTI
FTI
TBG
TBG
Submit


Home
Predict
Thyroid Disease Classification
Based on the given input, it predicts Thyroid disease for your body condition is {{prediction_text}}.

7. ADVANTAGES&DISADVANTAGES

ADVANTAGES

- Easytouse
- Costefficient
- Timeefficient

DISADVANTAGE

- Initialcostsofprinter
- Postprocessing
- Printingtime
- Specialskillrequiredfor3Dmodels
- ManufacturingJobLosses

8. APPLICATIONS

3D printing has gone through a number of changes over the years. In the early days, 3D printing wastime-consuming and costly, and not very practical for applications outside of industry. However, with theadventoftoday's moreflexibleandcost-effective3Dprintingmethods,thereareareaswhere3Dprintinghasbecomeapracticaltool.

Itisapplicableindifferentsectorssuchas

- EngineeringAndDesign
- Consumerproducts
- Manufacturing
- Education
- Aerospace
- Medical
- Movies/Theatres
- Architectures

9. CONCLUSION

3D printing technology could revolutionize and re-shape the world. Advance in 3D technology cansignificantlychangeandimprovethe waywe manufacture productsgoods worldwide.

If the last industrial revolution brought us mass production and the advent of economics of scale – thedigital 3D printing revolution could bring mass manufacturing back a full of circle – to an era of mass personalization, and returntoindividualcraftsmanship.

10. FUTURESCOPE

Future applications for 3D printing might include creating open-source scientific equipment to createopen-sourcelabsScience-basedapplicationslikereconstructingfossilsinpaleontology.Replicatingancientand priceless sartifacts inarchaeologyReconstructingbonesandbodypartsinforensicpathology.Thetechnologycurrentlybeingresearchedforbuildingconstruction.

BIBILOGRAPHY

- http://mashable.com/2014/03/06/3d-printed-blood-vessels/
- http://www.3dprinter.net/

APPENDIX

In [1]:
```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
client_3055a99af3464697994102c4e129439e = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='eNIjS9ifKkb2Dl9X6CwgXNYfpJMCpNEeLiE6j8RyJ754',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

body = client_3055a99af3464697994102c4e129439e.get_object(Bucket='thyroid-donotdelete-pr-qqkstuhbfy1zjo',Key='data.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

data = pd.read_csv(body)
data.head()
data.shape
```

Out[1]:
```
(9172, 31)
```

In [13]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2237 entries, 4 to 9169
Data columns (total 23 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   age                 2237 non-null   float64
 1   sex                 2147 non-null   object
 2   on_thyroxine        2237 non-null   object
 3   query_on_thyroxine  2237 non-null   object
 4   on_antithyroid_meds 2237 non-null   object
 5   sick                2237 non-null   object
 6   pregnant            2237 non-null   object
 7   thyroid_surgery     2237 non-null   object
 8   I131_treatment      2237 non-null   object
 9   query_hypothyroid   2237 non-null   object
 10  query_hyperthyroid  2237 non-null   object
 11  lithium             2237 non-null   object
 12  goitre              2237 non-null   object
 13  tumor               2237 non-null   object
 14  hypopituitary       2237 non-null   object
 15  psych               2237 non-null   object
 16  TSH                 2087 non-null   float64
 17  T3                  1643 non-null   float64
 18  TT4                 2140 non-null   float64
 19  T4U                 2059 non-null   float64
 20  FTI                 2060 non-null   float64
 21  TBG                 98 non-null     float64
 22  target              2237 non-null   object
dtypes: float64(7), object(16)
memory usage: 419.4+ KB
```

In [3]:
```
data.isnull().sum()
```

Out[3]:
```
age                    0
sex                  307
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                  842
T3_measured            0
T3                  2604
TT4_measured           0
TT4                  442
T4U_measured           0
T4U                  809
FTI_measured           0
FTI                  802
TBG_measured           0
TBG                 8823
referral_source        0
target                 0
patient_id             0
dtype: int64
```

In [4]:
```
#Removing Redundant attributes from dataset
#The columns listed below were removed because of redundacy.
#They are boolean and state whether or not a value has been recorded for their respective blood tests.
#TSH_measured
#T3_measured
#TT4_measured
#T4U_measured
#FTI_measured
#TBG_measured
data.drop(['TSH_measured','T3_measured','TT4_measured','T4U_measured','FTI_measured','TBG_measured','referral_source','patient_id'], axis=1, inplace = True)
```

In [5]:
```
data.head()
```

Out[5]:

	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_meds	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	...	tumor	hypopituitary	psych	TSH	T3	TT4	T4U	FTI	TBG	target
0	29	F	f	f	f	f	f	f	f	t	...	f	f	f	0.3	NaN	NaN	NaN	NaN	NaN	-
1	29	F	f	f	f	f	f	f	f	f	...	f	f	f	1.6	1.9	128.0	NaN	NaN	NaN	-
2	41	F	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	NaN	NaN	NaN	11.0	-
3	36	F	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	NaN	NaN	NaN	26.0	-
4	32	F	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	NaN	NaN	NaN	36.0	S

5 rows × 23 columns

In [6]:
```
data['target']
```

Out[6]:
```
0       -
1       -
2       -
3       -
4       S
       ..
9167    -
9168    -
9169    I
9170    -
9171    -
Name: target, Length: 9172, dtype: object
```

In [7]:
```
#re-mapping target values to diagnostic group
diagnoses = {'A': 'hyperthyroid conditions',
             'B': 'hyperthyroid conditions',
             'C': 'hyperthyroid conditions',
             'D': 'hyperthyroid conditions',
             'E': 'hypothyroid conditions',
             'F': 'hypothyroid conditions',
             'G': 'hypothyroid conditions',
             'H': 'hypothyroid conditions',
             'I': 'binding protein',
             'J': 'binding protein',
             '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) #remapping
```

In [8]:
```
data
```

Out[8]:

	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_meds	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	...	tumor	hypopituitary	psych	TSH	T3	TT4	T4U	FTI	TBG	target
0	29	F	f	f	f	f	f	f	f	t	...	f	f	f	0.3	NaN	NaN	NaN	NaN	NaN	NaN
1	29	F	f	f	f	f	f	f	f	f	...	f	f	f	1.6	1.9	128.0	NaN	NaN	NaN	NaN
2	41	F	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	NaN	NaN	NaN	11.0	NaN
3	36	F	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	NaN	NaN	NaN	26.0	NaN
4	32	F	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	NaN	NaN	NaN	36.0	miscellaneous
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9167	56	M	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	64.0	0.83	77.0	NaN	NaN
9168	22	M	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	91.0	0.92	99.0	NaN	NaN
9169	69	M	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	113.0	1.27	89.0	NaN	binding protein
9170	47	F	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	75.0	0.85	88.0	NaN	NaN
9171	31	M	f	f	f	f	f	f	f	t	...	f	f	f	NaN	NaN	66.0	1.02	65.0	NaN	NaN

9172 rows × 23 columns

In [9]:
```
data.isnull().sum()
```

Out[9]:
```
age                       0
sex                     307
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                     842
T3                     2604
TT4                     442
T4U                     809
FTI                     802
TBG                    8823
target                 6935
dtype: int64
```

In [10]:
```
data.dropna(subset=['target'],inplace=True)
```

In [11]:
```
data['target'].value_counts()
```

Out[11]:
```
hypothyroid conditions     593
general health             436
binding protein            376
replacement therapy        336
miscellaneous              281
hyperthyroid conditions    182
antithyroid treatment       33
Name: target, dtype: int64
```

In [12]:
```
data.head()
```

Out[12]:

	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_meds	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	...	tumor	hypopituitary	psych	TSH	T3	TT4	T4U	FTI	TBG	target
4	32	F	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	NaN	NaN	NaN	36.0	miscellaneous
18	63	F	t	f	f	t	f	f	f	f	...	f	f	f	68.000000	NaN	48.0	1.02	47.0	NaN	hypothyroid conditions
32	41	M	f	f	f	f	f	f	f	f	...	f	f	f	0.050000	1.6	39.0	1.00	39.0	NaN	miscellaneous
33	71	F	t	f	f	f	f	f	f	f	...	f	f	f	0.050000	NaN	126.0	1.38	91.0	NaN	binding protein
39	55	F	t	f	f	f	f	f	f	t	...	f	f	f	9.599999	2.4	136.0	1.48	92.0	NaN	replacement therapy

5 rows × 23 columns

In [13]:
```
data.describe()
```

Out[13]:

	age	TSH	T3	TT4	T4U	FTI	TBG
count	2237.000000	2087.000000	1643.000000	2140.000000	2059.000000	2060.000000	98.000000
mean	52.792579	14.930791	1.961875	116.390495	1.013439	120.363369	47.717347
std	19.677450	46.204092	1.452238	60.351600	0.280222	70.996728	32.398750
min	1.000000	0.005000	0.050000	2.000000	0.170000	1.400000	9.299999
25%	36.000000	0.255000	1.000000	76.000000	0.850000	83.000000	32.000000
50%	56.000000	2.000000	1.700000	109.000000	0.960000	109.000000	36.000000
75%	69.000000	8.799999	2.500000	156.000000	1.120000	157.000000	46.750000
max	95.000000	530.000000	18.000000	600.000000	2.330000	881.000000	200.000000

In [14]:
```
#Checking whether the age above 100
data[data.age>100]
```

Out[14]:

age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_meds	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	...	tumor	hypopituitary	psych	TSH	T3	TT4	T4U	FTI	TBG	target

0 rows × 23 columns

In [17]:
```
import numpy as np
#changing age of observation with(age>100) to null
data['age']=np.where((data.age>100), np.nan, data.age)
```

In [18]:
```
data
```

Out[18]:

	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_meds	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	...	tumor	hypopituitary	psych	TSH	T3	TT4	T4U	FTI	TBG	target
4	32.0	F	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	NaN	NaN	NaN	36.0	miscellaneous
18	63.0	F	t	f	f	t	f	f	f	f	...	f	f	f	68.000000	NaN	48.0	1.02	47.0	NaN	hypothyroid conditions
32	41.0	M	f	f	f	f	f	f	f	f	...	f	f	f	0.050000	1.6	39.0	1.00	39.0	NaN	miscellaneous
33	71.0	F	t	f	f	f	f	f	f	f	...	f	f	f	0.050000	NaN	126.0	1.38	91.0	NaN	binding protein
39	55.0	F	t	f	f	f	f	f	f	t	...	f	f	f	9.599999	2.4	136.0	1.48	92.0	NaN	replacement therapy
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9153	64.0	M	f	f	f	f	f	f	f	f	...	f	f	f	0.810000	NaN	31.0	0.55	56.0	NaN	general health
9157	60.0	M	f	f	t	f	f	f	f	f	...	f	f	f	0.180000	NaN	28.0	0.87	32.0	NaN	general health
9158	64.0	M	f	f	f	f	f	f	f	t	...	f	f	f	NaN	NaN	44.0	0.53	83.0	NaN	binding protein
9162	36.0	F	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	84.0	1.26	67.0	NaN	binding protein
9169	69.0	M	f	f	f	f	f	f	f	f	...	f	f	f	NaN	NaN	113.0	1.27	89.0	NaN	binding protein

2237 rows × 23 columns

#splitting the data values as x and y¶

In [19…
```
#splitting the data values as x and y
x=data.iloc[:,0:-1]
y= data.iloc[:,-1]
```

In [20…
```
data.isnull().sum()
```

Out[20]:
```
age                       0
sex                      90
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                     150
T3                      594
TT4                      97
T4U                     178
FTI                     177
TBG                    2139
target                    0
dtype: int64
```

In [21…
```
x['sex'].unique()
```

Out[21]:
```
array(['F', 'M', nan], dtype=object)
```

In [22…
```
x['sex'].replace(np.nan, 'F', inplace=True)
```

In [23…
```
x['sex'].value_counts()
```

Out[23]:
```
F    1701
M     536
Name: sex, dtype: int64
```

In [24…
```
x.isnull().sum()
```

Out[24]:
```
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                     150
T3                      594
TT4                      97
T4U                     178
FTI                     177
TBG                    2139
dtype: int64
```

In [25…
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2237 entries, 4 to 9169
Data columns (total 23 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   age                  2237 non-null   float64
 1   sex                  2147 non-null   object
 2   on_thyroxine         2237 non-null   object
 3   query_on_thyroxine   2237 non-null   object
 4   on_antithyroid_meds  2237 non-null   object
 5   sick                 2237 non-null   object
 6   pregnant             2237 non-null   object
 7   thyroid_surgery      2237 non-null   object
 8   I131_treatment       2237 non-null   object
 9   query_hypothyroid    2237 non-null   object
 10  query_hyperthyroid   2237 non-null   object
 11  lithium              2237 non-null   object
 12  goitre               2237 non-null   object
 13  tumor                2237 non-null   object
 14  hypopituitary        2237 non-null   object
 15  psych                2237 non-null   object
 16  TSH                  2087 non-null   float64
 17  T3                   1643 non-null   float64
 18  TT4                  2140 non-null   float64
 19  T4U                  2059 non-null   float64
 20  FTI                  2060 non-null   float64
 21  TBG                  98 non-null     float64
 22  target               2237 non-null   object
dtypes: float64(7), object(16)
memory usage: 419.4+ KB
```

```
In [26]: 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')
```

converting categorical to numerical values

```
In [27]: #applying ordinal_encoding to x values
         #Encoding the categorical data
         #Encoding the independent(output) variable
         from sklearn.preprocessing import OrdinalEncoder, LabelEncoder
         #categorical data


         ordinal_encoder = OrdinalEncoder(dtype = 'int64')
         x.iloc[:, 1:16] = ordinal_encoder.fit_transform(x.iloc[:, 1:16])
         #ordinal_encoder.fit_transform(x[['sex']])
```

```
In [28]: x.head()
```

Out[28]:

	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_meds	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	...	goitre	tumor	hypopituitary	psych	TSH	T3	TT4	T4U	FTI	TBG
4	32.0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	NaN	NaN	NaN	NaN	NaN	36.0
18	63.0	0	1	0	0	1	0	0	0	0	...	0	0	0	0	68.000000	NaN	48.0	1.02	47.0	NaN
32	41.0	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0.050000	1.6	39.0	1.00	39.0	NaN
33	71.0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0.050000	NaN	126.0	1.38	91.0	NaN
39	55.0	0	1	0	0	0	0	0	0	1	...	0	0	0	0	9.599999	2.4	136.0	1.48	92.0	NaN

5 rows × 22 columns

```
In [29]: x.replace(np.nan, '0', inplace=True)
         x.head()
```

Out[29]:

	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_meds	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	...	goitre	tumor	hypopituitary	psych	TSH	T3	TT4	T4U	FTI	TBG
4	32.0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	36.0
18	63.0	0	1	0	0	1	0	0	0	0	...	0	0	0	0	68.0	0	48.0	1.02	47.0	0
32	41.0	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0.05	1.6	39.0	1.0	39.0	0
33	71.0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0.05	0	126.0	1.38	91.0	0
39	55.0	0	1	0	0	0	0	0	0	1	...	0	0	0	0	9.599999	2.4	136.0	1.48	92.0	0

5 rows × 22 columns

```
In [30]: #applying label_encoding to y values
         label_encoder = LabelEncoder()
         y_dt= label_encoder.fit_transform(y)
```

```
In [31]: y=pd.DataFrame(y_dt, columns=['target'])
         y
```

Out[31]:

	target
0	5
1	4
2	5
3	1
4	6
...	...
2232	2
2233	2
2234	1
2235	1
2236	1

2237 rows × 1 columns

```
In [32]: y.value_counts(normalize=True)
```

Out[32]:

```
target
4         0.265087
2         0.194904
1         0.168082
6         0.150201
5         0.125615
3         0.081359
0         0.014752
dtype: float64
```

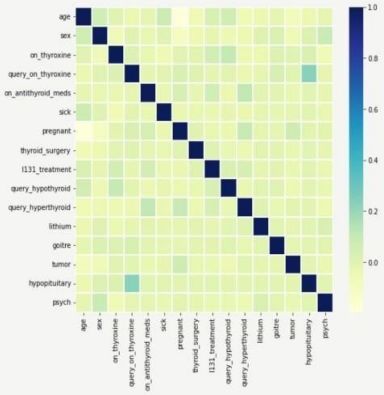
```
In [ ]:
```

Checking the Correlation

In [35]:
```
import matplotlib.pyplot as plt
#checking correlation using Heatmap
import seaborn as sns
corrmat = x.corr()

f, ax = plt.subplots(figsize =(9, 8))
sns.heatmap(corrmat, ax = ax, cmap ="YlGnBu", linewidths = 0.1)
```

Out[35]: <AxesSubplot:>



splitting the train and test split

In [36]:
```
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)
```

In [37]:
```
y_train.value_counts()
```

Out[37]:
```
target
4        471
2        351
1        302
6        265
5        230
3        144
0         26
dtype: int64
```

In [38]:
```
pip install imblearn
```

```
Requirement already satisfied: imblearn in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (0.0)
Requirement already satisfied: imbalanced-learn in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imblearn) (0.9.1)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.20.3)
Requirement already satisfied: scikit-learn>=1.1.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.1.1)
Requirement already satisfied: scipy>=1.3.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.7.3)
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (2.2.0)
Requirement already satisfied: joblib>=1.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.1.0)
Note: you may need to restart the kernel to use updated packages.
```

In [39]:
```
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)
```

In [40]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_bal = sc.fit_transform(x_bal)
x_test_bal= sc.transform(x_test_bal)
```

In [41]:
```
x_bal
```

Out[41]:
```
array([[-1.62721505, -0.44060477, -0.4238    , ..., -2.50870684,
        -1.40088079,  3.29445097],
       [-0.11561403, -0.44060477,  2.35960359, ..., -0.26259147,
         0.0720981 , -0.19494049],
       [ 1.1874903 ,  2.26960776, -0.4238    , ...,  0.17039463,
        -0.19352104, -0.19494049],
       ...,
       [ 1.395987  , -0.44060477,  2.35960359, ...,  0.43615031,
         0.06101022, -0.19494049],
       [ 0.72802783, -0.44060477,  2.35960359, ...,  0.143333  ,
         0.89086631, -0.19494049],
       [ 1.15628145, -0.44060477,  2.35960359, ...,  0.39723515,
        -0.26588659, -0.19494049]])
```

In [42]:
```
x_test_bal
```

Out[42]:
```
array([[-1.5229667 , -0.44060477, -0.4238    , ...,  1.06342846,
         0.13246609, -0.19494049],
       [-0.89747663, -0.44060477, -0.4238    , ...,  1.76703086,
        -0.30218342, -0.19494049],
       [-0.9496008 ,  2.26960776, -0.4238    , ..., -0.39789962,
        -0.90586329, -0.19494049],
       ...,
       [ 1.39013447, -0.44060477,  2.35960359, ...,  0.81835453,
         0.70094189, -0.19494049],
       [ 1.33846247, -0.44060477,  2.35960359, ...,  0.81987378,
         0.67327619, -0.19494049],
       [-0.19842352, -0.44060477, -0.4238    , ...,  0.24830842,
         0.37610348, -0.19494049]])
```

In [43]:
```
y_bal.value_counts()
```

Out[43]:
```
target
0         471
1         471
2         471
3         471
4         471
5         471
6         471
dtype: int64
```

In [44]:
```
columns=['age','sex','on_thyroxine','query_on_thyroxine','on_antithyroid_meds','sick','pregnant','thyroid_surgery','I131_treatment','query_hypothyroid','query_hyperthyroid','lithium', 'goitre', 'tumor','hypopituitary','psych','
```

In [45]:
```
x_test_bal= pd.DataFrame(x_test_bal,columns=columns)
```

In [46]:
```
x_bal= pd.DataFrame(x_bal,columns=columns)
```

In [47]:
```
x_bal
```

Out[47]:

	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_meds	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	...	goitre	tumor	hypopituitary	psych	TSH	T3	TT4	T4U	FTI	TB
0	-1.627215	-0.440605	-0.423800	-0.105069	-0.158703	-0.141815	-0.137297	-0.239601	-0.162675	-0.230986	...	-0.052319	-0.137297	-0.024637	-0.107982	-0.315458	-1.035358	-1.704935	-2.508707	-1.400881	3.2944
1	-0.115614	-0.440605	2.359604	-0.105069	-0.158703	-0.141815	-0.137297	-0.239601	-0.162675	-0.230986	...	-0.052319	-0.137297	-0.024637	-0.107982	-0.090056	0.155233	-0.197223	-0.262591	0.072098	-0.1949
2	1.187490	2.269608	-0.423800	-0.105069	-0.158703	-0.141815	-0.137297	-0.239601	-0.162675	-0.230986	...	-0.052319	-0.137297	-0.024637	-0.107982	-0.278907	-0.471394	-0.227079	0.170395	-0.193521	-0.1949
3	-1.366594	-0.440605	-0.423800	-0.105069	-0.158703	-0.141815	-0.137297	-0.239601	-0.162675	-0.230986	...	-0.052319	7.283487	-0.024637	-0.107982	-0.284999	0.969848	0.041622	0.495134	-0.133153	-0.1949
4	-0.167738	-0.440605	-0.423800	-0.105069	-0.158703	-0.141815	-0.137297	-0.239601	-0.162675	-0.230986	...	-0.052319	-0.137297	-0.024637	-0.107982	-0.306321	4.541622	1.459767	-0.127283	1.496783	-0.1949
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3292	0.546923	-0.440605	2.359604	-0.105069	-0.158703	-0.141815	-0.137297	-0.239601	-0.162675	-0.230986	...	-0.052319	-0.137297	-0.024637	-0.107982	-0.114424	0.343221	-0.148122	-0.146517	0.040168	-0.1949
3293	0.383062	-0.440605	2.359604	-0.105069	-0.158703	-0.141815	-0.137297	-0.239601	-0.162675	-0.230986	...	-0.052319	-0.137297	-0.024637	-0.107982	-0.309176	-0.856540	0.565143	-0.513902	1.085434	-0.1949
3294	1.395987	-0.440605	2.359604	-0.105069	-0.158703	-0.141815	-0.137297	-0.239601	-0.162675	-0.230986	...	-0.052319	-0.137297	-0.024637	-0.107982	-0.095452	-0.172405	0.248906	0.436150	0.061010	-0.1949
3295	0.728028	-0.440605	2.359604	-0.105069	-0.158703	-0.141815	-0.137297	-0.239601	-0.162675	-0.230986	...	-0.052319	-0.137297	-0.024637	-0.107982	-0.311566	0.087864	1.071643	0.143333	0.890866	-0.1949
3296	1.156281	-0.440605	2.359604	-0.105069	-0.158703	-0.141815	-0.137297	-0.239601	-0.162675	-0.230986	...	-0.052319	-0.137297	-0.024637	-0.107982	-0.072439	0.079407	-0.200359	0.397235	-0.265887	-0.1949

3297 rows x 22 columns

In [48]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
rfr = RandomForestClassifier().fit(x_bal,y_bal)
y_pred = rfr.predict(x_test_bal)
accuracy_score(y_test_bal,y_pred)
x_bal.shape,y_bal.shape,x_test_bal.shape,y_test_bal.shape
```

```
/tmp/wsuser/ipykernel_1020/2696972469.py:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  rfr = RandomForestClassifier().fit(x_bal,y_bal)
```

Out[48]:
```
((3297, 22), (3297, 1), (854, 22), (854, 1))
```

In [49]:
```
test_score=accuracy_score(y_test_bal,y_pred)
test_score
```

Out[49]:
```
0.9098360655737705
```

In [50]:
```
train_score = accuracy_score(y_bal,rfr.predict(x_bal))
train_score
```

Out[50]:
```
1.0
```

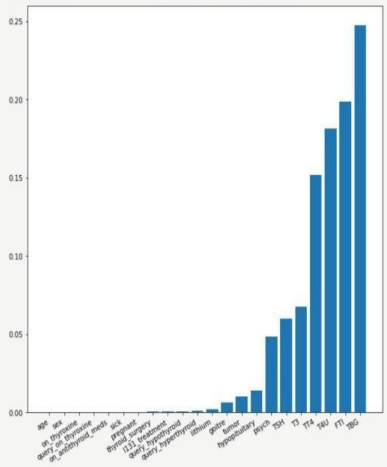
performing feature importance

In [51]:
```
#perform feature importance
from sklearn.inspection import permutation_importance
results = permutation_importance(rfr,x_bal,y_bal, scoring='accuracy')
```

In [52]:
```
#gets importance
feature_importance=['age','sex','on_thyroxine','query_on_thyroxine','on_antithyroid_meds','sick','pregnant','thyroid_surgery','I131_treatment','query_hypothyroid','query_hyperthyroid','lithium', 'goitre', 'tumor','hypopituitary
importance = results.importances_mean
importance = np.sort(importance)
#summerize feature importance
for i,v in enumerate(importance):
    i=feature_importance[i]
    print('feature: {:<20} Score: {}'. format(i,v))
#plot important feature

plt.figure(figsize=(10,10))
plt.bar(x=feature_importance, height = importance)
plt.xticks(rotation=30, ha='right')
plt.show()
```

```
feature: age                  Score: 0.0
feature: sex                  Score: 0.0
feature: on_thyroxine         Score: 0.0
feature: query_on_thyroxine   Score: 0.0
feature: on_antithyroid_meds  Score: 0.0
feature: sick                 Score: 0.0
feature: pregnant             Score: 6.066120715801926e-05
feature: thyroid_surgery      Score: 0.0003639672429481554
feature: I131_treatment       Score: 0.0004852896572641541
feature: query_hypothyroid    Score: 0.0006066120715801926
feature: query_hyperthyroid   Score: 0.0012132241431604962
feature: lithium              Score: 0.0018198362147406888
feature: goitre               Score: 0.006308765544434336
feature: tumor                Score: 0.010130421595389748
feature: hypopituitary        Score: 0.013830755232029101
feature: psych                Score: 0.04852896572641796
feature: TSH                  Score: 0.05987261146496816
feature: T3                   Score: 0.06781922960266909
feature: TT4                  Score: 0.1519563239308462
feature: T4U                  Score: 0.1816196542311192
feature: FTI                  Score: 0.19884743706399757
feature: TBG                  Score: 0.24774037003336363
```



In [53]:
```
x_bal.drop(['age','sex','on_thyroxine','query_on_thyroxine','on_antithyroid_meds','sick','pregnant','thyroid_surgery','I131_treatment','query_hypothyroid','query_hyperthyroid','lithium'],axis=1,inplace=True)
```

In [54]:
```
x_test_bal.drop(['age','sex','on_thyroxine','query_on_thyroxine','on_antithyroid_meds','sick','pregnant','thyroid_surgery','I131_treatment','query_hypothyroid','query_hyperthyroid','lithium'],axis=1,inplace=True)
```

In [55]:
```
x_bal.head()
```

Out[55]:

	goitre	tumor	hypopituitary	psych	TSH	T3	TT4	T4U	FTI	TBG
0	-0.052319	-0.137297	-0.024637	-0.107982	-0.315458	-1.035358	-1.704935	-2.508707	-1.400881	3.294451
1	-0.052319	-0.137297	-0.024637	-0.107982	-0.090056	0.155233	-0.197223	-0.262591	0.072098	-0.194940
2	-0.052319	-0.137297	-0.024637	-0.107982	-0.278907	-0.471394	-0.227079	0.170395	-0.193521	-0.194940
3	-0.052319	7.283487	-0.024637	-0.107982	-0.284999	0.969848	0.041622	0.495134	-0.133153	-0.194940
4	-0.052319	-0.137297	-0.024637	-0.107982	-0.306321	4.541622	1.459767	-0.127283	1.496783	-0.194940

In [56]:
```
x_test_bal.head()
```

Out[56]:

	goitre	tumor	hypopituitary	psych	TSH	T3	TT4	T4U	FTI	TBG
0	-0.052319	-0.137297	-0.024637	-0.107982	-0.312412	0.593872	0.788014	1.063428	0.132466	-0.19494
1	-0.052319	-0.137297	-0.024637	-0.107982	-0.314240	0.781860	0.444674	1.767031	-0.302183	-0.19494
2	-0.052319	-0.137297	-0.024637	-0.107982	1.298911	-0.408731	-1.227244	-0.397900	-0.905863	-0.19494
3	-0.052319	-0.137297	-0.024637	-0.107982	-0.166205	-0.471394	-0.227079	-0.397900	0.132466	-0.19494
4	-0.052319	-0.137297	-0.024637	-0.107982	-0.227125	-0.346068	-0.301718	-0.830886	0.434306	-0.19494

RandomForest Model-1

In [57]:
```
rfr1 = RandomForestClassifier()
rfr1.fit(x_bal,y_bal)
y_pred=rfr1.predict(x_test_bal)
```

```
/tmp/wsuser/ipykernel_1020/1228087459.py:2: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  rfr1.fit(x_bal,y_bal)
```

In [58]:
```
print(classification_report(y_test_bal,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.81      0.17      0.28       122
           1       0.81      0.93      0.87       122
           2       0.93      0.98      0.96       122
           3       0.76      0.84      0.80       122
           4       0.48      0.87      0.61       122
           5       0.87      0.69      0.77       122
           6       0.58      0.50      0.54       122

    accuracy                           0.71       854
   macro avg       0.75      0.71      0.69       854
weighted avg       0.75      0.71      0.69       854
```

In [59]:
```
train_score = accuracy_score(y_bal,rfr1.predict(x_bal))
```

In [60]:
```
train_score
```

Out[60]: 1.0

SVC Model-3

```
In [62]: from sklearn.svm import SVC
         from sklearn.metrics import accuracy_score, classification_report

         sv= SVC()
```

```
In [63]: sv.fit(x_bal,y_bal)
```

```
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
```

```
Out[63]: ▾ SVC
         SVC()
```

```
In [64]: y_pred = sv.predict(x_test_bal)
```

```
In [65]: print(classification_report(y_test_bal,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.70      0.85      0.77       122
           1       0.76      0.81      0.79       122
           2       0.88      0.93      0.90       122
           3       0.71      0.65      0.68       122
           4       0.71      0.63      0.67       122
           5       0.76      0.54      0.63       122
           6       0.49      0.57      0.52       122

    accuracy                           0.71       854
   macro avg       0.72      0.71      0.71       854
weighted avg       0.72      0.71      0.71       854
```

```
In [66]: train_score=accuracy_score(y_bal,sv.predict(x_bal))
         train_score
```

```
Out[66]: 0.7154989384288747
```

Random_Search for SVC

```
In [67]: params = {


         'C': [0.1, 1, 10, 100, 1000],
         'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
             'kernel': ['rbf','sqrt']


         }
```

```
In [68]: from sklearn.model_selection import RandomizedSearchCV
         random_svc = RandomizedSearchCV(sv,params, scoring='accuracy',cv=5,n_jobs=-1)
```

```
In [69]: random_svc.fit(x_bal,y_bal)
```

```
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
```

In [70…
```
random_svc.best_params_
```

Out[70]:
```
{'kernel': 'rbf', 'gamma': 1, 'C': 1000}
```

In [78…
```
sv1=SVC(kernel= 'rbf', gamma= 0.1,C= 100)
```

In [79…
```
sv1.fit(x_bal,y_bal)
```

```
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/utils/validation.py:1111: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
```

Out[79]:
```
SVC(C=100, gamma=0.1)
```

In [80…
```
y_pred= sv1.predict(x_test_bal)
```

In [81…
```
print(classification_report(y_test_bal,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.74      0.75      0.75       122
           1       0.77      0.86      0.81       122
           2       0.95      0.91      0.93       122
           3       0.70      0.66      0.68       122
           4       0.66      0.73      0.70       122
           5       0.72      0.72      0.72       122
           6       0.57      0.48      0.52       122

    accuracy                           0.73       854
   macro avg       0.73      0.73      0.73       854
weighted avg       0.73      0.73      0.73       854
```

In [82…
```
train_score= accuracy_score(y_bal,sv1.predict(x_bal))
train_score
```

Out[82]:
```
0.8125568698817106
```

In [83…
```
# saving the model
import pickle
pickle.dump(sv1,open('thyroid_1_model.pkl','wb'))
```

In [85…
```
features = np.array([[0,0,0,0,0.000000,0.0,0.0,1.00,0.0,40.0]])
print(label_encoder.inverse_transform(sv1.predict(features)))
```

```
['binding protein']
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/sklearn/base.py:450: UserWarning: X does not have valid feature names, but SVC was fitted with feature names
  warnings.warn(
```

In [86…
```
type(features)
```

Out[86]:
```
numpy.ndarray
```

In [87…
```
pickle.dump(label_encoder,open('label_encoder.pkl','wb'))
```

In [88…
```
data['target'].unique()
```

Out[88]:
```
array(['miscellaneous', 'hypothyroid conditions', 'binding protein',
       'replacement therapy', 'general health', 'hyperthyroid conditions',
       'antithyroid treatment'], dtype=object)
```

In [89…
```
y['target'].unique()
```

Out[89]:
```
array([5, 4, 1, 6, 2, 3, 0])
```

In [90…
```
!tar -zcvf thyroid_disease_new.tgz thyroid_1_model.pkl
!tar -zcvf thyroid_disease_new.tgz label_encoder.pkl
```

```
thyroid_1_model.pkl
label_encoder.pkl
```

In [91…
```
ls -1
```

```
label_encoder.pkl
thyroid_1_model.pkl
thyroid_disease_new.tgz
```

In [92…
```
!pip install watson-machine-learning-client --upgrade
```

```
Collecting watson-machine-learning-client
  Downloading watson_machine_learning_client-1.0.391-py3-none-any.whl (538 kB)
     |[redacted]| 538 kB 24.7 MB/s eta 0:00:01
Requirement already satisfied: ibm-cos-sdk in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2.11.0)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.26.7)
Requirement already satisfied: boto3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.18.21)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2.26.0)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (0.8.9)
Requirement already satisfied: pandas in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.3.4)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (0.3.3)
Requirement already satisfied: tqdm in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (4.62.3)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2022.5.18.1)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (0.10.0)
Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (0.5.0)
Requirement already satisfied: botocore<1.22.0,>=1.21.21 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client) (1.21.41)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from botocore<1.22.0,>=1.21.21->boto3->watson-machine-learning-client) (2.8.2)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.22.0,>=1.21.21->boto3->watson-machine-learning-client) (1.15.0)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk->watson-machine-learning-client) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk->watson-machine-learning-client) (2.11.0)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->watson-machine-learning-client) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->watson-machine-learning-client) (2.0.4)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-client) (2021.3)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas->watson-machine-learning-client) (1.20.3)
Installing collected packages: watson-machine-learning-client
Successfully installed watson-machine-learning-client-1.0.391
```

In [11…
```
from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url": "https://us-south.ml.cloud.ibm.com",
    "apikey": "a4hY8b_3PvNDBVOmekwxAJuLoP_kIRhjOYcUVYwFBQZv"
}
client = APIClient(wml_credentials)
```

In [11…
```
def guid_from_space_name(client, space_name):
    space = client.spaces.get_details()
    #print(space)
    return(next(item for item in space['resources'] if item['entity']["name"] == space_name)['metadata']['id'])
```

In [12…
```
space_uid = guid_from_space_name(client, space_name='thyroid_deploy')
print("Space UID = " + space_uid)
```

```
Space UID = 77c033bf-40dc-4e25-8afb-5bea7f65de10
```

In [12…
```
client.set.default_space(space_uid)
```

Out[121]:
```
'SUCCESS'
```

In [12…
```
client.software_specifications.list()
```

```
------------------------------  ------------------------------------  ----
NAME                            ASSET_ID                              TYPE
default_py3.6                   0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12       020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt      069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6         09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12      09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9       0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6           0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                      0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod    1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6               10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl       111e41b3-de2d-5422-a4d6-bf776828c4b7  base
runtime-22.1-py3.9              12b83a17-24d8-5082-900f-0ab31fbfd3cb  base
scikit-learn_0.22-py3.6         154010fa-5b3b-4ac1-82af-4d5ee5abbc85  base
default_r3.6                    1b70aec3-ab34-4b87-8aa0-a4a3c8296a36  base
pytorch-onnx_1.3-py3.6          1bc6029a-cc97-56da-b8e0-39c3880dbbe7  base
pytorch-onnx_rt22.1-py3.9-edt   1d362186-7ad5-5b59-8b6c-9d0880bde37f  base
tensorflow_2.1-py3.6            1eb25b84-d6ed-5dde-b6a5-3fbdf1665666  base
spark-mllib_3.2                 20047f72-0a98-58c7-9ff5-a77b012eb8f5  base
tensorflow_2.4-py3.8-horovod    217c16f6-178f-56bf-824a-b19f20564c49  base
runtime-22.1-py3.9-cuda         26215f05-08c3-5a41-a1b0-da66306ce658  base
do_py3.8                        295addb5-9ef9-547e-9bf4-92ae3563e720  base
autoai-ts_3.8-py3.8             2aa0c932-798f-5ae9-abd6-15e0c2402fb5  base
tensorflow_1.15-py3.6           2b73a275-7cbf-420b-a912-eae7f436e0bc  base
pytorch_1.2-py3.6               2c8ef57d-2687-4b7d-acce-01f94976dac1  base
spark-mllib_2.3                 2e51f700-bca0-4b0d-88dc-5c6791338875  base
pytorch-onnx_1.1-py3.6-edt      32983cea-3f32-4400-8965-dde874a8d67e  base
spark-mllib_3.0-py37            36507ebe-8770-55ba-ab2a-eafe787600e9  base
spark-mllib_2.4                 390d21f8-e58b-4fac-9c55-d7ceda621326  base
xgboost_0.82-py3.6              39e31acd-5f30-41dc-ae44-60233c80306e  base
pytorch-onnx_1.2-py3.6-edt      40589d0e-7019-4e28-8daa-fb03b6f4fe12  base
default_r36py38                 41c247d3-45f8-5a71-b065-8580229facf0  base
autoai-ts_rt22.1-py3.9          4269d26e-07ba-5d40-8f66-2d495b0c71f7  base
autoai-obm_3.0                  42b92e18-d9ab-567f-988a-4240ba1ed5f7  base
pmml-3.0_4.3                    493bcb95-16f1-5bc5-bee8-81b8af80e9c7  base
spark-mllib_2.4-r_3.6           49403dff-92e9-4c87-a3d7-a42d0021c095  base
```

In [12…
```
import sklearn
sklearn.__version__
```

Out[123]:
```
'1.1.1'
```

In [12…
```
software_spec_uid= client.software_specifications.get_uid_by_name("runtime-22.1-py3.9")
software_spec_uid
```

Out[124]:
```
'12b83a17-24d8-5082-900f-0ab31fbfd3cb'
```

In [13…
```
model_details = client.repository.store_model(model='thyroid_disease_new.tgz',
                                              meta_props={client.repository.ModelMetaNames.NAME:"thyroiddeploy",
                                                          client.repository.ModelMetaNames.TYPE:"scikit-learn_1.0",
                                                          client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid
                                                         },
                                              training_data=x_train,
                                              training_target=y_train)
```

In [13…
```
model_id = client.repository.get_model_id(model_details)
model_id
```

Out[133]:
```
'2a67d325-0fcb-471c-9365-70472e8446a4'
```

In [13…
```
# Deploy
deployment = client.deployments.create(
    artifact_uid=model_id,
    meta_props={client.deployments.ConfigurationMetaNames.NAME:"thyroiddeploy_deploy",
    client.deployments.ConfigurationMetaNames.ONLINE: {}
})
```

```
#######################################################################################

Synchronous deployment creation for uid: '2a67d325-0fcb-471c-9365-70472e8446a4' started

#######################################################################################


initializing
Note: online_url is deprecated and will be removed in a future release. Use serving_urls instead.

ready


------------------------------------------------------------------------------------------------
Successfully finished deployment creation, deployment_uid='93c0fb0f-c7b4-4dbd-a280-8fb0bcee2d8d'
------------------------------------------------------------------------------------------------
```