

THYROID DISEASE CLASSIFICATION USING MACHINE LEARNING

Submitted in partial fulfillment of requirement for the award of the Degree

Bachelor of Computer Science

In the faculty of Computer Science of Bharathiar University, Coimbatore

Submitted by

TEAM ID: NM2023TMID32548

P.POOMALAR

T.MARIKANNAN

P.THAMARAISELVI

A.RATHIKA

Under the guidance of

Dr. A.GEETHA, ASSISTANT PROFESSOR,

, Department of Computer Science.



DEPARTMENT OF COMPUTER SCIENCE

GOVERNMENT ARTS AND SCIENCE COLLEGE(CO-ED)

(Affiliated To Bharathiar University)

AVINASHI-641 654

APRIL-2023

GOVERNMENT ARTS AND SCIENCE COLLEGE(CO-ED)

NAAN MUDHALVAN PROJECT WORK

(AFFILIATED TO BHARATHIAR UNIVERSITY)

AVINASHI-641 654

TITLE :Thyroid Disease Classification Using ML

This is to certify that this is a bonafide record of work done by the above students of III B.Sc (CS) Degree **NAAN MUDHALVAN PROJECT** during the year 2022-2023

Submitted for the Naan Mudhalvan project work held
on.....20

CLASS MENTOR

Dr. A. GEETHA

HEAD OF DEPARTMENT

Prof. B. HEMALATHA

TABLE OF CONTENTS

CHAPTER NO	CONTENTS
1	INTRODUCTION
2	PROBLEM DEFINITION
3	IDEATION
4	REQUIREMENT ANALYSIS
5	INPUT DESIGN 5.1. INPUT DESIGN DESCRIPTION
6	OUTPUT DESIGN 6.1.OUTPUT DESIGN DESCRIPTION
7	DESCRIPTION OF MODULES
8	PROJECT PALNNING PHASE
9	PROJECT DEVELOPMENT PHASE
10	CONCLUSION
11	APPENDICES A.TECHNICAL FLOW B.SAMPLE INPUT C.SAMPLE OUTPUT D.SAMPLE CODING

1. INTRODUCTION

THYROID DISEASE CLASSIFICATION USING ML

The thyroid gland is a small organ that's located in the front of the neck, wrapped around the windpipe (trachea). It's shaped like a butterfly, smaller in the middle with two wide wings that extend around the side of your throat. The thyroid is a gland. The two main types of thyroid disease are hypothyroidism and hyperthyroidism. Both conditions can be caused by other diseases that impact the way the thyroid gland works.

Thyroid disease can affect anyone — men, women, infants, teenagers and the elderly. It can be present at birth (typically hypothyroidism) and it can develop as you age (often after menopause in women).

Symptoms of an overactive thyroid (hyperthyroidism) can include Experiencing anxiety, irritability and nervousness, Having trouble sleeping, Losing weight, Having an enlarged thyroid gland or a goiter, Having muscle weakness and tremors, Experiencing irregular menstrual periods or having your menstrual cycle stop, Feeling sensitive to heat and Having vision problems or eye irritation.

Symptoms of an underactive thyroid (hypothyroidism) can include Feeling tired (fatigue), Gaining weight, Experiencing forgetfulness, Having frequent and heavy menstrual periods, Having dry and coarse hair, Having a hoarse voice and Experiencing an intolerance to cold temperatures.

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, Random Forest Classifier, K-NN, logistic regression and DT Classifier 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.

2. PROBLEM SELECTION

Predicting thyroid disease using machine learning is an interesting and important project that can have significant implications in the field of healthcare. Here are some steps you can follow to select a suitable project:

1. Research existing work: Before starting any project, it's important to research existing work in the field. Look for similar projects that have been done before and see what approaches were used. This will help you understand the current state-of-the-art and identify any gaps in the literature that your project can address.
2. Define the problem: Once you have a good understanding of the current research, define the problem you want to solve. For example, you could focus on predicting the risk of developing thyroid disease in a particular population or predicting the progression of the disease in patients who have already been diagnosed.
3. Gather data: In order to train a machine learning model, you will need a dataset. Look for publicly available datasets or consider

collecting your own data. Make sure that the data you use is of high quality and is representative of the problem you are trying to solve.

4. Train and evaluate the model: Once you have chosen an algorithm, train the model on your dataset and evaluate its performance using appropriate metrics such as accuracy, precision, recall, and F1 score. If the model's performance is not satisfactory, consider tweaking the algorithm or using a different one.
5. Deploy the model: Once you have a model that performs well, deploy it in a real-world setting. This could involve integrating it into an electronic health.

3. IDEATION

- i. **Collect Data:** Collect data from various sources, including medical records, patient information, and clinical studies.

Ensure that the data is in a structured format and that it contains all the necessary features for building a model.
- ii. **Pre-process the Data:** Once you have the data, pre-process it by cleaning, normalizing, and transforming it into a format that can be used by a machine learning model.
- iii. **Feature Selection:** Choose the relevant features that will be used to predict the thyroid disease. These features can include patient demographics, lab test results, and medical history.
- iv. **Choose a Model:** Choose an appropriate machine learning algorithm that will be used for prediction. There are various algorithms available, including decision trees, random forests, and support vector machines.

- v. **Train the Model:** Use the pre-processed data and the chosen algorithm to train the model. Split the data into training and testing sets to evaluate the model's accuracy.
- vi. **Evaluate the Model:** Evaluate the model's performance using different metrics such as accuracy, precision, recall, and F1-score.
- vii. **Deploy the Model:** Finally, deploy the model into a production environment where it can be used to predict the thyroid disease of new patients.

4. REQUIREMENT ANALYSIS

Data Collection:

Gather a large dataset of patient information, including demographics, medical history, symptoms, lab test results, and imaging data (if available). The dataset should include both positive and negative cases of thyroid disease to ensure the model is trained on a balanced dataset.

Data Pre-processing:

Clean the data, remove any missing or irrelevant information, and encode categorical variables.

Feature Selection:

Identify the most important features that contribute to the prediction of thyroid disease.

Algorithm Selection:

Select the appropriate machine learning algorithm for the task, such as logistic regression, decision trees, random forests, or support vector machines.

Model training:

Split the dataset into training and validation sets, train the model on the training set, and evaluate the performance on the validation set. Adjust the model hyper-parameters to optimize performance.

Model Evaluation:

Evaluate the model's performance using metrics such as accuracy, precision, recall, and F1-score. Use cross-validation to ensure the model's generalizability to new data.

Model Deployment:

Deploy the trained model in a user-friendly interface, such as a web application or mobile app, to enable healthcare professionals to make accurate predictions of thyroid disease in their patients.

Maintenance and Updates:

Regularly update the model with new data and retrain as necessary to ensure the accuracy of the predictions.

6. INPUT DESIGN

The input design is the process of entering data to the system. The input design goal is to enter to the computer as accurate as possible. Here inputs are designed effectively so that errors made by the operation are minimized.

The inputs to the system have been designed in such a way that manual forms and the inputs are coordinated where the data elements are common to the sources document and to the input. The input is acceptable and understandable by the users who are using it.

Input design is the process of converting user-originated inputs to a computer-based format input data are collected and organized into group of similar data. Once identified, appropriate input media are selected for processing.

Input design means the physical and performance requirements of a device that are used as a basis for device design. Input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc

5.1. INPUT DESIGN DESCRIPTION

HOME

In the home page a simple concept of thyroid is displayed and it shows a predict option to get to the predict page.

PREDICT

In this page it asks for input of certain data that is necessary to predict the thyroid type it also shows a submit button which redirects it to the submit page.

7. OUTPUT DESIGN

A design output is a drawing or specification or manufacturing instruction. Design outputs describe all the components, parts, and pieces that go into your device. Design outputs describe all assemblies and subassemblies of product.

Output design is the process of converting data into hard copy that is understood by all. The various outputs have been in such a way that they represent the same format that the office and management used to.

Computer output is the most important and direct source of information to the user. Efficient, intelligible output design should improve the systems relationships with the user and help in decision making. A major form of out is the hardcopy from the printer. Output requirements are designed during system analysis.

6.1. OUTPUT DESIGN DESCRIPTION

SUBMIT

In this page after enter necessary data in predict page then click the submit button.

It displays the result of the data entered like (Normal, hypothyroid and hyperthyroid).

7. DESCRIPTION OF MODULES

Modules are unit of code written in access basic language.

❖ HOME

❖ PREDICT

❖ SUBMIT

HOME

- ❖ In the home page a simple concept of thyroid is displayed and it shows a predict option to get to the predict page.

PREDICT

- ❖ In this page it asks for input of certain data that is necessary to predict the thyroid type.

SUBMIT

- ❖ In this page the result of the predicted thyroid type is displayed like whether the user is normal, hyperthyroid or hypothyroid.

8. PROJECT PLANNING PHASE

Define the problem:

Define the problem you want to solve. For example, you may want to build a model that can accurately predict the risk of thyroid disease in patients based on certain clinical and demographic features.

Gather Data:

Identify the data sources that can be used to build the machine learning model. This may involve gathering data from electronic health records, medical imaging, or patient surveys.

Choose a Machine Learning Algorithm:

There are several machine learning algorithms that can be used to build a predictive model. You will need to choose an algorithm that is appropriate for your data and problem.

Train the model:

Once you have chosen an algorithm, you will need to train the model on the data.

This involves dividing the data into training and validation sets, and using the training set to optimize the model's parameters.

Evaluate the model:

After training the model, you will need to evaluate its performance on the validation set. This will give you an idea of how well the model will perform on new data.

Deploy the model:

Once you have a model that performs well on the validation set, you can deploy it in a clinical setting. This may involve integrating it into an electronic health record system, or creating a standalone application.

9. PROJECT DEVELOPMENT PHASE

Data Collection and Preparation:

This involves gathering relevant data from various sources and preparing it for use in machine learning algorithms. In the case of thyroid disease prediction, this might include medical records, lab results, and patient demographics.

Feature Selection and Engineering:

Once the data has been collected, the next step is to select the most relevant features (i.e., variables or attributes) that are likely to be predictive of thyroid disease. This might involve domain expertise from medical professionals, as well as statistical techniques such as correlation analysis or principal component analysis.

Model Selection and Training:

With the features selected, the next step is to choose an appropriate machine learning algorithm to use for prediction. This might include traditional models such as logistic regression or more advanced methods such as deep learning

Model Evaluation and Tuning:

Once the model has been trained, it is evaluated using various metrics such as accuracy, precision, and recall

Deployment and Monitoring:

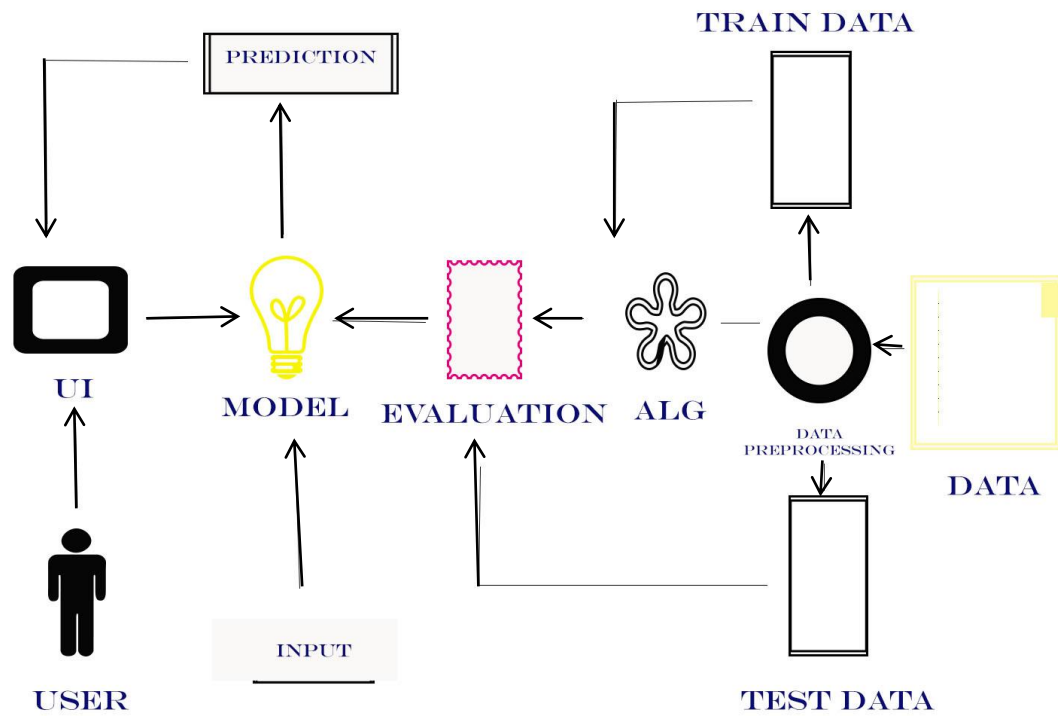
Finally, the trained and validated model can be deployed for use in clinical settings. Ongoing monitoring and evaluation of the model's performance are critical to ensure that it remains accurate and effective over time.

10. CONCLUSION

Machine Learning plays a very deciding role in disease prediction. Machine Learning algorithms such as SVM, Random Forest Classifier, K-NN, logistic regression and DT Classifier 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 thyroid disease. It predicts with 84% accuracy score.

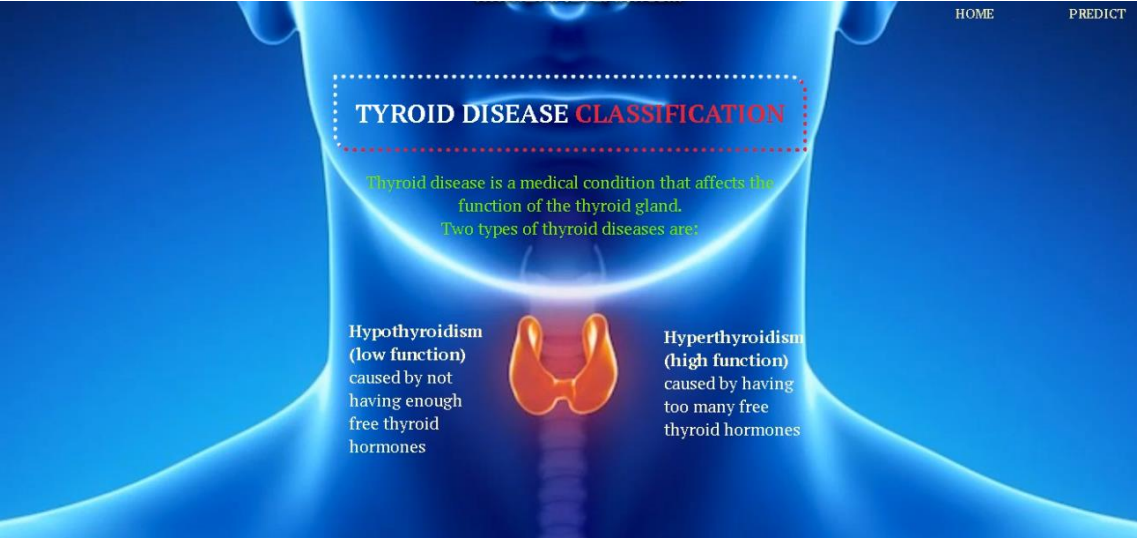
APPENDICES

A. TECHNICAL FLOW



B. SAMPLE INPUT

HOME

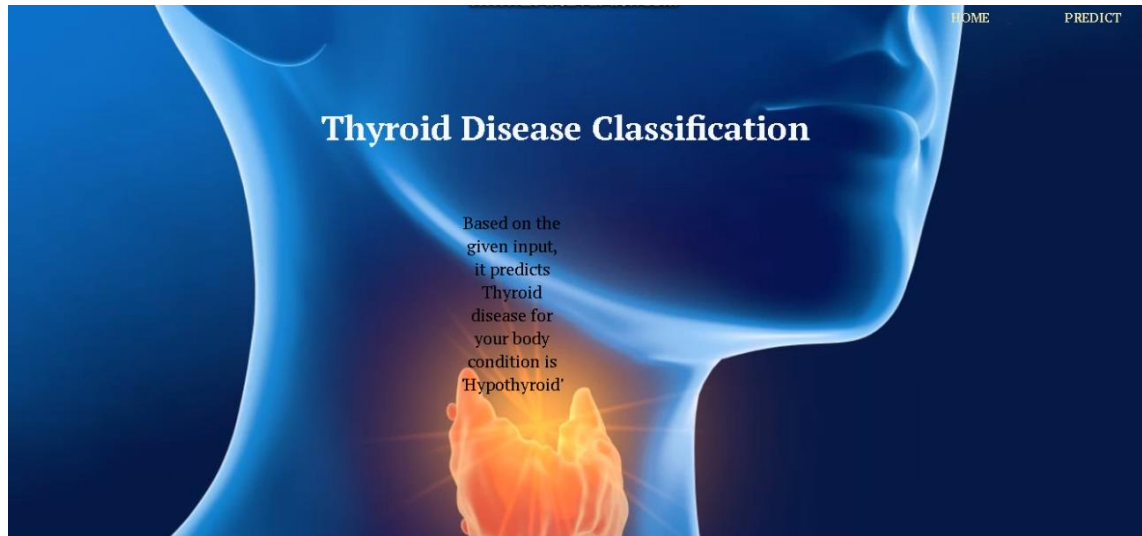


PREDICT

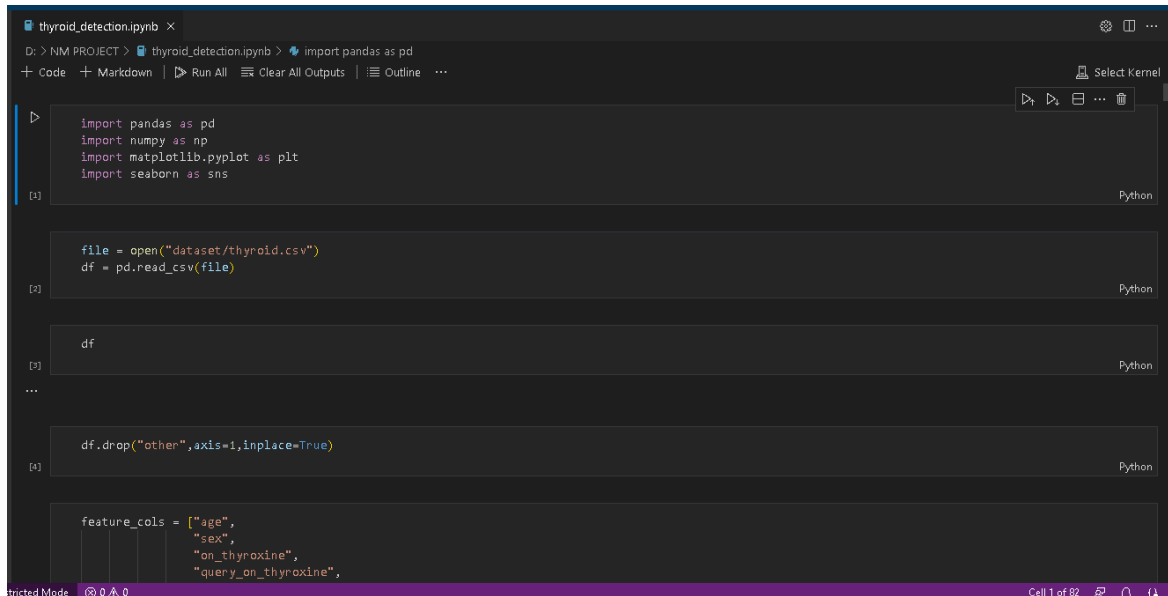
The screenshot shows the 'PREDICT' page of the application. At the top right, there are two links: 'HOME' and 'PREDICT'. The main heading is 'Thyroid Disease Classification'. On the left side, there is a list of input fields with labels in yellow and radio buttons for gender selection. The inputs are: 'GOITRE' (radio buttons for MALE and FEMALE), 'TUMOR' (radio buttons for MALE and FEMALE), 'HYPOPITUITARY' (radio buttons for MALE and FEMALE), 'PSYCH' (radio buttons for MALE and FEMALE), 'TSH' (text input with value 11), 'T3' (text input with value 2), 'TT4' (text input with value 55), 'T4U' (text input with value 7), 'FTI' (text input with value 8), 'TSG' (text input with value 48.99), and a 'Submit' button. The background features a glowing orange 3D model of a human thyroid gland.

C. SAMPLE OUTPUT

SUBMIT



D. SAMPLE CODING

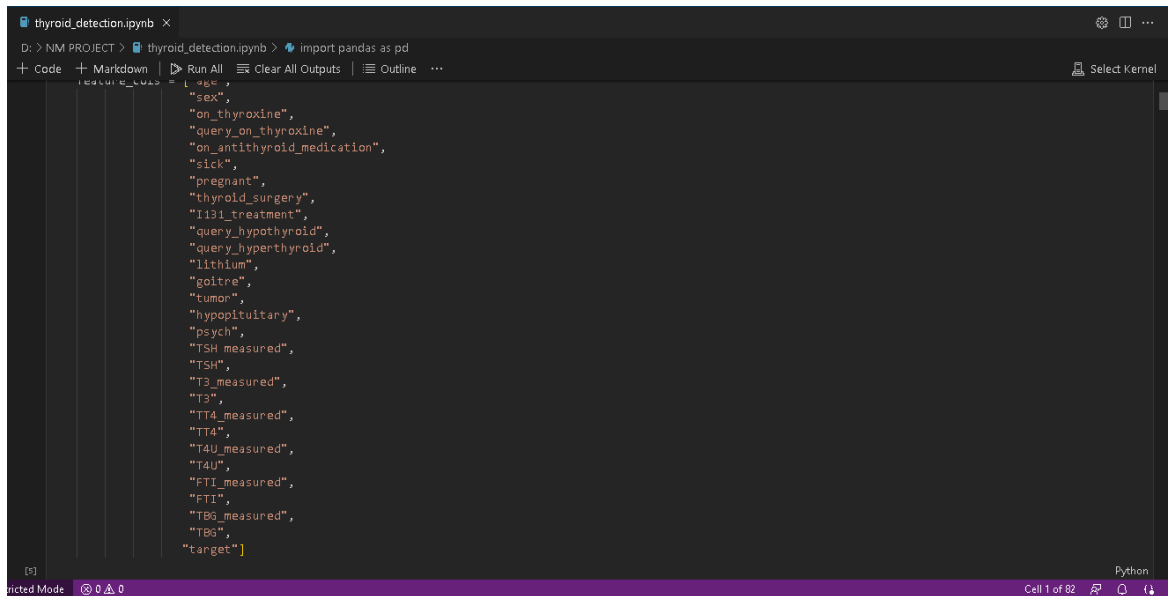


```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

file = open("dataset/thyroid.csv")
df = pd.read_csv(file)

df

df.drop("other",axis=1,inplace=True)
```



```
feature_cols = ["age",
               "sex",
               "on_thyroxine",
               "query_on_thyroxine",
               "on_antithyroid_medication",
               "sick",
               "pregnant",
               "thyroid_surgery",
               "t131_treatment",
               "query_hypothyroid",
               "query_hyperthyroid",
               "lithium",
               "goitre",
               "tumor",
               "hypopituitary",
               "psych",
               "TSH_measured",
               "TSH",
               "T3_measured",
               "T3",
               "TT4_measured",
               "TT4",
               "T4U_measured",
               "T4U",
               "FTI_measured",
               "FTI",
               "T8G_measured",
               "T8G",
               "target"]
```

```
thyroid_detection.ipynb
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel

df.columns = feature_cols

df

...

Splitting target

Now we can check that the target columns as many categorial names with some
indicate numbers so going to split with the respective features

thyroid.names file content

The diagnosis consists of a string of letters indicating diagnosed conditions. A diagnosis "-" indicates no condition requiring comment. A diagnosis of the form "X[Y]" is
interpreted as "consistent with X, but more likely Y". The conditions are divided into groups where each group corresponds to a class of comments.

Letter  Diagnosis
-----

```

```
thyroid_detection.ipynb
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel

target = df.target
create = target.str.split('([A-Za-z]+)', expand=True)
create = create[1]
target = create.replace({None:'Z'}) #here z is none type
df.target = target

df.target.unique()

array(['Z', 'S', 'F', 'AK', 'R', 'I', 'M', 'N', 'G', 'K', 'A', 'KJ', 'L',
      'MK', 'Q', 'J', 'C', 'O', 'LO', 'H', 'D', 'GK', 'MI', 'P', 'FK',
      'B', 'GI', 'GKJ', 'OI', 'E'], dtype=object)

df

...

Now we want to impute the null values but this case the null values are marked as in '?' so we can do some tricks

df = df.replace(['?'], np.nan)
```

```
thyroid_detection.ipynb
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel
Python

df.isnull().sum()
[12] Python
...
age                0
sex                307
on_thyroxine       0
query_on_thyroxine 0
on_antithyroid_medication 0
sick               0
pregnant           0
thyroid_surgery    0
l131_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                 2603
TT4_measured       0
TT4                441
T4U_measured       0
T4U                808

Restricted Mode 0 0 0 Cell 1 of 81
```

```
thyroid_detection.ipynb
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel
Python

FTI_measured       0
FTI                801
TBG_measured       0
TBG                8822
target             0
dtype: int64

# here we can see the TBG has more null observations it will tremendously occur problem so we can remove and some of the other
# feature rows which is not useful

df.drop(['TBG_measured', 'TBG', 'T3_measured', 'TSH_measured', 'TT4_measured', 'T4U_measured', 'FTI_measured'], axis=1, inplace=True)
[13] Python

df
[14] Python
...

df.isnull().sum()
[15] Python
...
age                0
sex                307
on_thyroxine       0
query_on_thyroxine 0

Restricted Mode 0 0 0 Cell 1 of 81
```

```
thyroid_detection.ipynb
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel

[15] Python
...
age                0
sex                307
on_thyroxine       0
query_on_thyroxine 0
on_antithyroid_medication 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                 2603
TT4                441
T4U                808
FTI                801
target             0
dtype: int64

df.sex.replace({'F':2,'M':1},inplace=True)

[16] Python
```

```
thyroid_detection.ipynb
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel

[17] Python
round_values = round(df.sex.mean())
df.sex.fillna(round_values,inplace=True)

[18] Python
df.sex.unique()

... array([2., 1.])

[19] Python
df.isnull().sum()

...
age                0
sex                0
on_thyroxine       0
query_on_thyroxine 0
on_antithyroid_medication 0
sick               0
pregnant          0
thyroid_surgery    0
I131_treatment     0
query_hypothyroid  0
query_hyperthyroid 0
lithium            0
goitre             0
tumor              0

```

dicted Mode 0.0.0 Cell 1 of 81

```
thyroid_detection.ipynb •
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel

goitre      0
tumor       0
hypopituitary 0
psych       0
TSH         842
T3          2603
TT4         441
T4U         808
FTI         801
target      0
dtype: int64

# now we will impute the null values with knn imputer
from sklearn.impute import KNNImputer
knnimp = KNNImputer(n_neighbors=3)

[20] Python

cols = ['TSH','T3','TT4','T4U','FTI']
for i in cols:
    df[i] = knnimp.fit_transform(df[[i]])

[21] Python

df.isnull().sum() # now we can see there is no null values

[22] Python
```

```
thyroid_detection.ipynb •
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel

[22] Python

...
age      0
sex      0
on_thyroxine 0
query_on_thyroxine 0
on_antithyroid_medication 0
sick     0
pregnant 0
thyroid_surgery 0
t131_treatment 0
query_hypothyroid 0
query_hyperthyroid 0
lithium  0
goitre   0
tumor    0
hypopituitary 0
psych    0
TSH      0
T3       0
TT4      0
T4U      0
FTI      0
target   0
dtype: int64

df.info()

[23] Python
```

```
thyroid_detection.ipynb
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel

[23] Python

df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9171 entries, 0 to 9170
Data columns (total 22 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   age                                   9171 non-null   int64
1   sex                                   9171 non-null   float64
2   on_thyroxine                         9171 non-null   object
3   query_on_thyroxine                   9171 non-null   object
4   on_antithyroid_medication            9171 non-null   object
5   sick                                  9171 non-null   object
6   pregnant                             9171 non-null   object
7   thyroid_surgery                      9171 non-null   object
8   I131_treatment                       9171 non-null   object
9   query_hypothyroid                    9171 non-null   object
10  query_hyperthyroid                    9171 non-null   object
11  lithium                               9171 non-null   object
12  goitre                                9171 non-null   object
13  tumor                                 9171 non-null   object
14  hypopituitary                        9171 non-null   object
15  psych                                 9171 non-null   object
16  TSH                                   9171 non-null   float64
17  T3                                    9171 non-null   float64
18  TT4                                   9171 non-null   float64
19  T4U                                   9171 non-null   float64
20  FTI                                   9171 non-null   float64
```

```
thyroid_detection.ipynb
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel

15 psych                                9171 non-null   object
16 TSH                                   9171 non-null   float64
17 T3                                    9171 non-null   float64
18 TT4                                   9171 non-null   float64
19 T4U                                   9171 non-null   float64
20 FTI                                   9171 non-null   float64
21 target                               9171 non-null   object
dtypes: float64(6), int64(1), object(15)
memory usage: 1.5+ MB

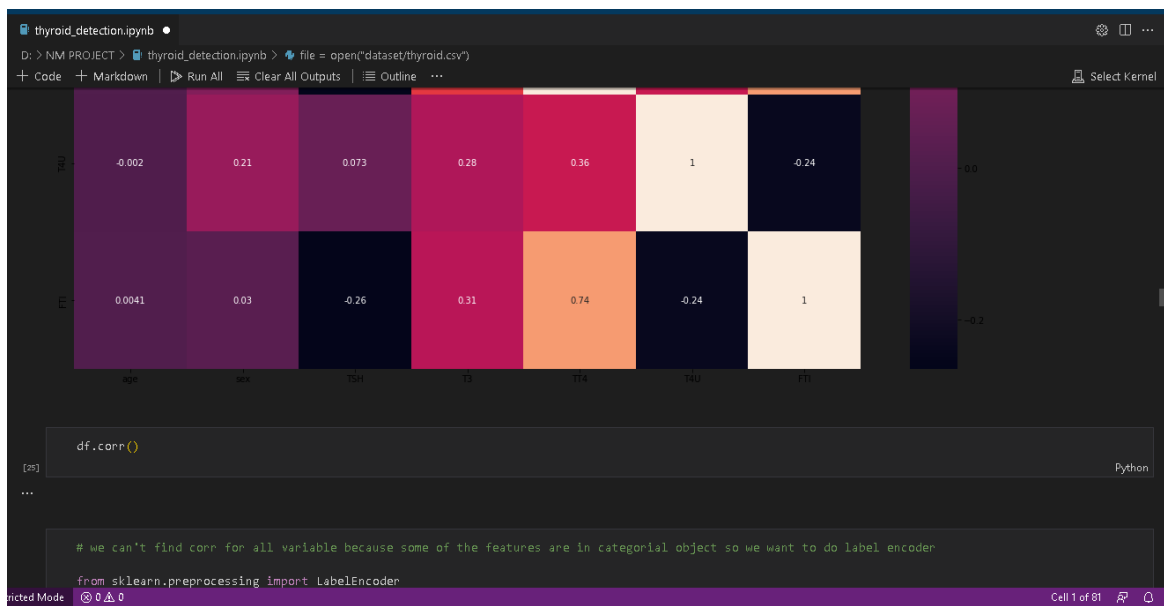
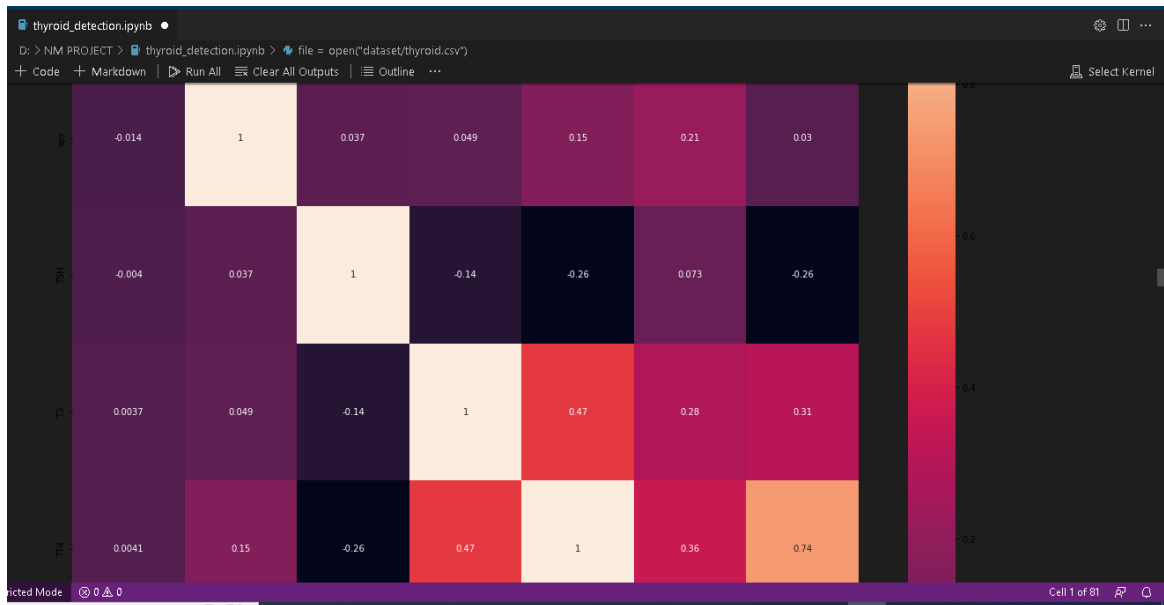
Exploratory Data Analysis

[34] Python

plt.figure(figsize=(20,20))
sns.heatmap(df.corr(),annot=True)

<AxesSubplot:~>

1 -0.014 -0.004 0.0037 0.0041 -0.002 0.0041 1.0
```




```
thyroid_detection.ipynb •
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | Run All | Clear All Outputs | Outline ...
Select Kernel

# we can't find corr for all variable because some of the features are in categorical object so we want to do Label encoder
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

[26] Python

cols = df.select_dtypes(include=['object'])

[27] Python

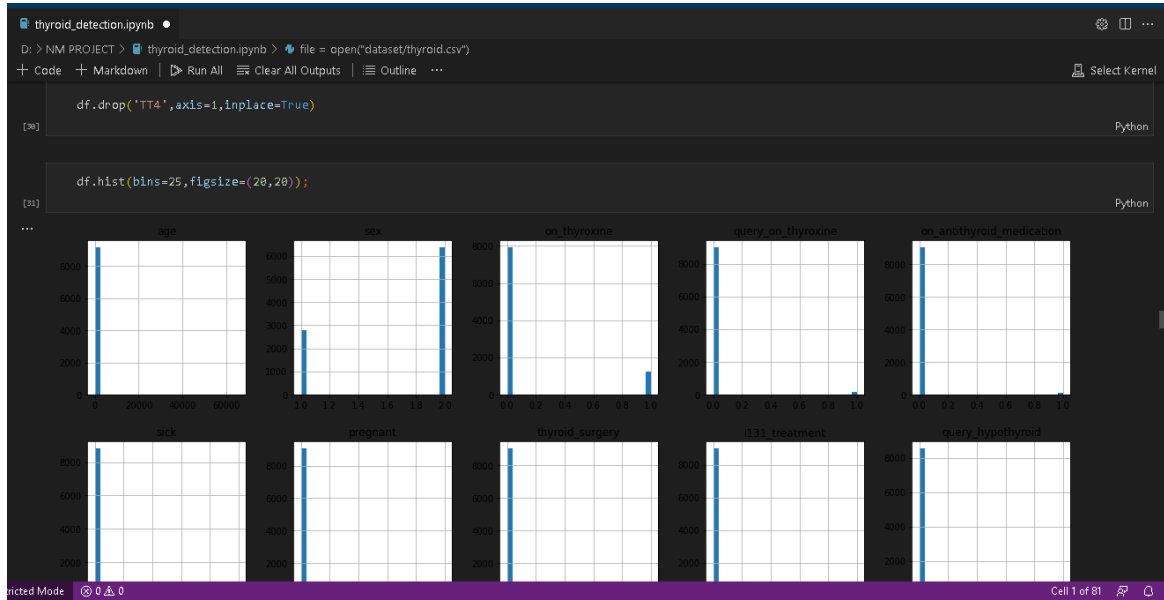
for i in cols.columns:
    try:
        df[i] = le.fit_transform(df[i])
    except:
        continue

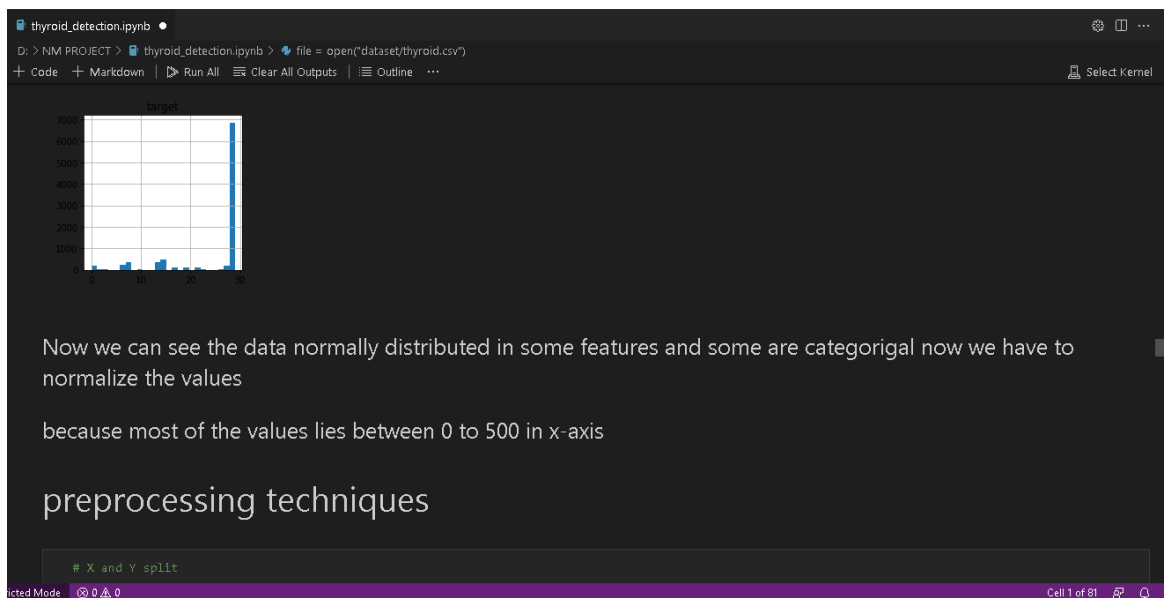
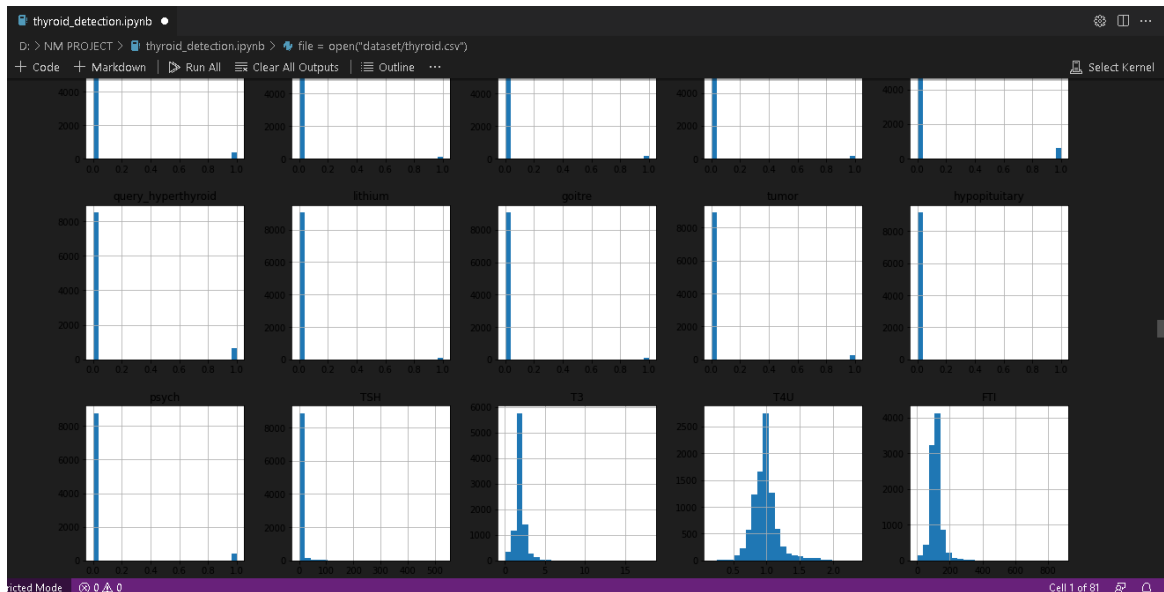
[28] Python

# now we can see their is correlation in some features
for a in range(len(df.corr())):
    for b in range(a):
        if((df.corr().iloc[a,b]) >= 0.7):
            print(df.corr().columns[b])

[29] Python

... TT4
dicted Mode 0 0 0 Cell 1 of 81
```





```
thyroid_detection.ipynb •
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ...
Select Kernel

# X and Y split
X = df.drop('target',axis=1)
y = df.target
df2 = X # for on-going process without PCA

[32] Python

y.unique() # we can see there is 29 types are present => 29 categorigal values

[33] Python

... array([29, 28, 6, 1, 27, 13, 19, 22, 8, 15, 0, 16, 17, 21, 26, 14, 3,
        23, 18, 12, 4, 10, 20, 25, 7, 2, 9, 11, 24, 5])

PCA Technique

First we use PCA then see the result then we move to normal modeling(without PCA)

from sklearn.decomposition import PCA
pca = PCA(n_components=10)

[34] Python

Restricted Mode 0 0 0 Cell 1 of 81
```

```
thyroid_detection.ipynb •
D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")
+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ...
Select Kernel

v = pca.fit_transform(X)

[35] Python

X_pca = pd.DataFrame(data = v, columns = ['component_1', 'component_2', 'component_3', 'component_4', 'component_5', 'component_6', 'component_7', 'comp

[36] Python

X_pca

[37] Python

...

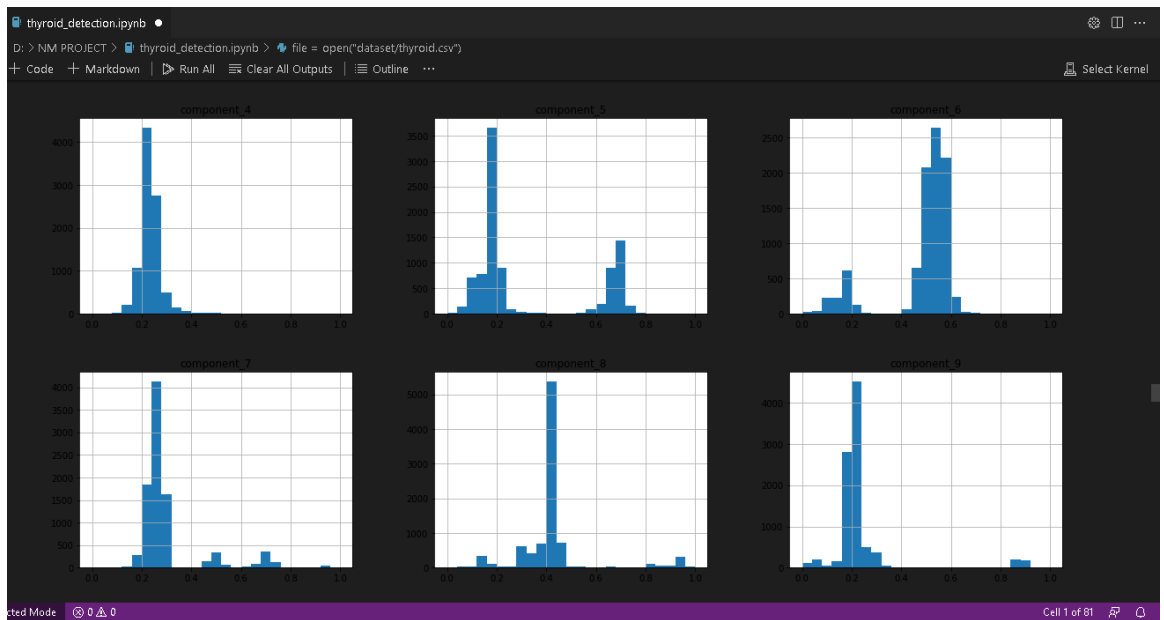
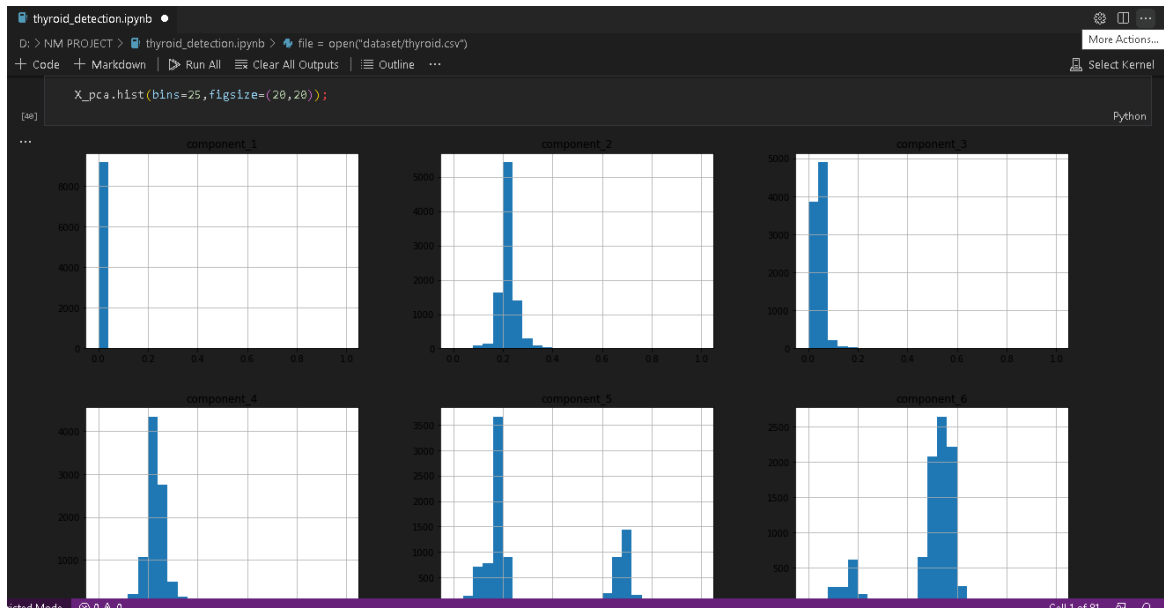
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()

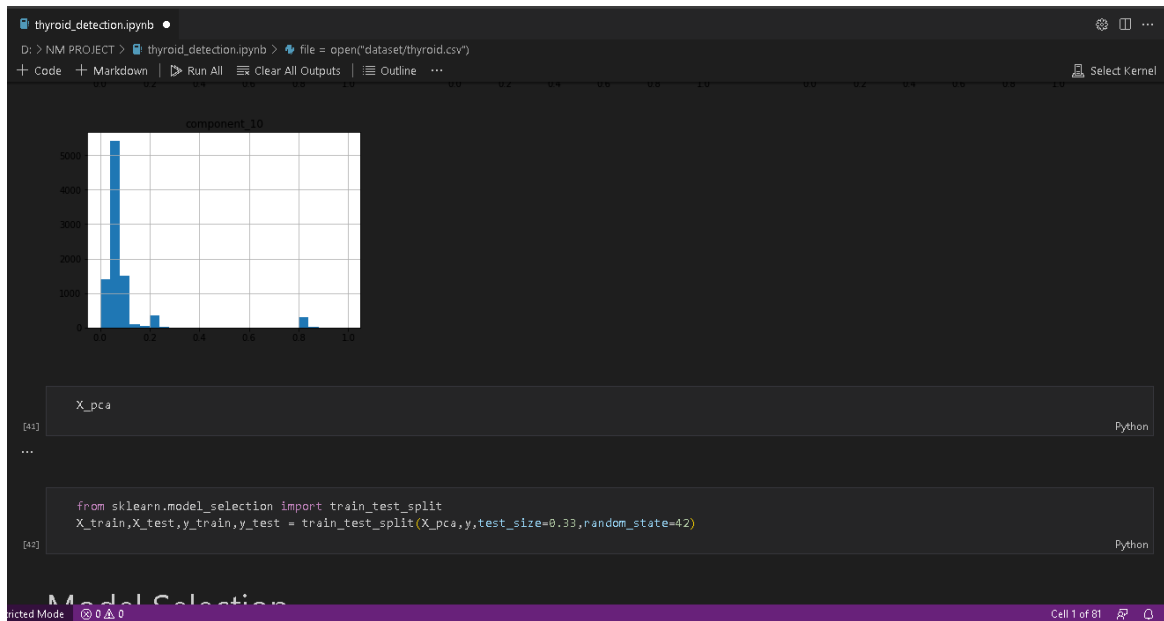
[38] Python

for i in X_pca.columns:
    X_pca[i] = scaler.fit_transform(X_pca[[i]])

[39] Python

Restricted Mode 0 0 0 Cell 1 of 81
```





thyroid_detection.ipynb

D: > NM PROJECT > thyroid_detection.ipynb > file = open("dataset/thyroid.csv")

+ Code + Markdown | Run All | Clear All Outputs | Outline ...

Select Kernel

Model Selection

```
[63] from sklearn.metrics import accuracy_score
```

Python

Decision Tree Classifier

```
[44] from sklearn.tree import DecisionTreeClassifier
tree = DecisionTreeClassifier(max_depth=3)
clf = tree.fit(X_train,y_train)
treepredict = clf.predict(X_test)
```

Python

```
[45] accuracy_score(treepredict,y_test)
```

Python

... 0.7961678229269984

Random Forest Classifier

dicted Mode 0 0 0

Cell 1 of 81

