JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Unveiling the Hidden Truth: Predicting Thyroid Disease using Machine Learning Techniques

Author Info
Niharikareddy Meenigea
Data Analyst
Virginial International University
USA

Abstract : - This paper is being written to serve as a resource for research scholars interested in the prediction of thyroid disease. To predict and evaluate the performance of different machine learning techniques, three algorithms were widely used, namely logistic regression, decision trees, and k- nearest neighbour (kNN) algorithms. This study represented the intuition of how to predict thyroid disease and highlighted how to use logistic regression, decision trees, and kNN as classification tools. Thyroid data set of machine learning repository from UC Irvin knowledge discovery in databases archive was used for this.

Keywords :- Decision Tree, Thyroid Disease, K-nearest neighbour, Logistic Regression

INTRODUCTION

In India, one out of every ten people suffers from thyroid disease. Thyroid disease primarily affects women between the ages of 17 and 54. Thyroid amplification causes cardiovascular complications, an increase in blood pressure, and an increase in cholesterol levels, depression, and decreased fertility.

The thyroid gland produces two active thyroid hormones, total serum thyroxin (T4) and total serum triiodothyronine (T3), to control the body's metabolism. These hormones are required for the proper functioning of each cell, tissue, and organ, in overall energy yield and regulation, and in the production of proteins in the regulation of body temperature.

The functional behaviour of the thyroid disease represents the idea for thyroid disease diagnosis and therapy and is the key in most thyroid diseases. Thyroid disease is classified into three types: euthyroidism, hyperthyroidism, and hypothyroidism, which denote normal, excessive, or defective thyroid hormone levels. The condition euthyroidism depicts normal thyroid hormone production and poor alternate therapy.

The enormous measure of information can be taken care of utilizing the AI procedures. Characterization models are appropriate for the arrangement and qualification of the information classes. The treatment of both mathematical and straight out values should be possible by the grouping processes. Classification is a two-step grouping model in the stage one, in view of some preparation information, a model is developed, and in sync two, an obscure tuple is given to the model to classify into a class mark.

In human existence, the grouping has an extraordinary impact. The examination of various grouping procedures is a non-paltry and has an incredible reliance on the informational index properties. In the measurements local area, strategic relapse, choice tree and k-closest neighbor have a regarded position for grouping issues.

In view of the exploration works and writing audit, very little work has been finished in the characterization strategies for patients pruned by the thyroid sickness. The techniques for arrangement utilized are the notable strategies. To zero in on the above-talked about issues, this paper makes sense of the utilization of three characterization AI calculations: strategic relapse order, choice tree arrangement and closest neighbors grouping to order individuals pruned by thyroid sickness utilizing the thyroid illness information base. The paper make sense of exhaustively about the readiness, preparing and testing of the information, bit by bit depiction of every one of the procedures utilized, and an examination of the precision of the techniques utilized in the expectation.

Research Methods

Logistic regression is a generally excellent strategy to portray and test speculations for the two downright qualities. Logistic regression is utilized for characterization utilizing a straight choice limit. Logistic regression works by first searching for straight choice limits between the examples of various classes. Then, the strategic capability is utilized to get the likelihood of belongingness to each class characterized concerning the choice limits.

The overall equation for the calculated logistic regression classification is:

$$\text{logistic}(\eta) = (1/1+\exp(-\eta))$$

The decision tree utilizes the AI procedure to take care of the issue of grouping and expectation. Hubs and leaves are the two components of which the choice trees are framed. Hubs help in the testing of a specific attribute and leaves addresses a class.

The decision tree execution is hierarchical methodology. The tree is work with the objective to accomplish the greatest homogeneity in leaves as could be expected. The ceaseless division of leaves from non-homogenous to homogeneous is the main pressing issue of this calculation. The means of preparing, arrangement and testing are simple and quick in choice trees. It gives ease to the clients to acquire the data by the tree portrayal of the information.

The center calculation utilized here is the ID3. It is a voracious inquiry procedure with no backtracking of the whole possible branch. The calculation utilizes the entropy and data gain to track down the conceivable outcomes.

1. Entropy:-

E=−∑i=1Npilog2pi

2. Information Gain:-

Gain=Eparent−Echildren

Following advances are utilized to pursue a decision tree:

- Information arrangement
- Information parcel into preparing, approval and testing set
- Choice of characteristic: a technique to choose the "best" conceivable quality for the parting by the choice tree model
- Assessment of the model

In the kNN grouping, the learning depends on relationship that the test tuple is planned by contrasting and the preparation tuples that are like it. At the point when given an obscure piece of information, a k-closest neighbor classifier finds the example space for the k preparation tuples that are nearest to the obscure data of interest. The obscure tuple is grouped by a greater part of its neighbors, and gets doled out to the class generally normal among its k-closest neighbors. On giving a preparation tuple k-closest neighbor just stores it and holds on until it is given a test tuple. In this way, it is a "sluggish student" as it stores the preparation tuples or the occurrences, they are otherwise called "case based students".

The k-closest neighbor calculation depends on the distance of the closest neighbors and uses the accompanying distance formulae to find the closest neighbors:

1. Euclidean Distance:-

$$D_e = \left(\sum_{i=1}^{n} (p_i - q_i)^2 \right)^{1/2}$$

2. Manhattan Distance:-

$$D_m = \sum_{i=1}^{n} |p_i - q_i|$$

3. Minkowski Distance:-

$$D = \left(\sum_{i=1}^{n} |p_i - q_i|^p \right)^{1/p}$$

4. Hammimg Distance:-

d(C) = min{d(c1,c2) | c1, c2 € C, c1 = c2}

In this work, Euclidean distance is utilized.

Following four stages are utilized to do the kNN order:

- Gauge the distance metric between the test significant piece of information and every one of the named data of interest.
- Request the named data of interest in the rising request of distance metric
- Select the top k-marked data of interest and take a gander at the class names
- Find the class mark that larger part of these k-named information focuses have and appoint it to the test data of interest.

Result and Analysis

The perception of the preparation informational collection will be same for all the three characterization techniques. The perception of the new thyroid informational collection is displayed in the figure.

The investigation and clarification of every calculation is accounted for beneath.

Logistic Regression Classification:-

The calculated grouping characterizes the information in light of the sigmoid capability. The order of the thyroid informational collection by calculated relapse grouping is displayed in Fig. 1b. The information are separated into three sections:

- Preparing set (70%)
- Approval set (15%)
- Test set (15%)

On assessing the calculated relapse classifier on this thyroid informational collection, it shows an approval misclassification level of 18.75% and test misclassification level of 15.625%. The disarray network drawn on the irregular choice of test information on the arbitrary determination of preparing information is displayed in Fig. 1c. The confusion matrix makes sense of about the how much the model is precise. The equation for the estimation of accuracy from the confusion matrix is given as:-

Accuracy = (TP + TN)/ (TP + TN + FP + FN)

Decision Tree:-

Absolute serum thyroxin and complete serum triiodothyronine are chosen as the component names for settling on the choices. The class that the result produce will be class 0 (having thyroid) and class 1 (ordinary). To set up the model, informational index is partitioned into preparing set (70%), approval set (15%) and test set (15%).

On assessing the exhibition of the calculation, it shows approval misclassification level of 12.5% and test misclassification level of 3.125%.

The confusion matrix is drawn here for computing the exactness of the model is displayed in Fig. 1d.

While applying the calculation at irregular picked a point [4.2 1.2] as question point. The genuine class of the question point is 0. On applying the calculation, the closest neighbors of the question point are: ([4.2 1.2] [4.2 0.7] [4.7 1.1] [3.6 1.5] [4.7 1.8]), classes of the closest neighbors are: ([1] [0] [0] [0] [0]) and anticipated class for inquiry point is likewise 0. The perception of working of kNN is displayed in Fig. 1e.

On assessing the exhibition of the k-NN classifier, the test misclassification rate = 3.125%.
The confusion matrix of the test information is displayed in Fig. 1f.

Table 1 Result analysis

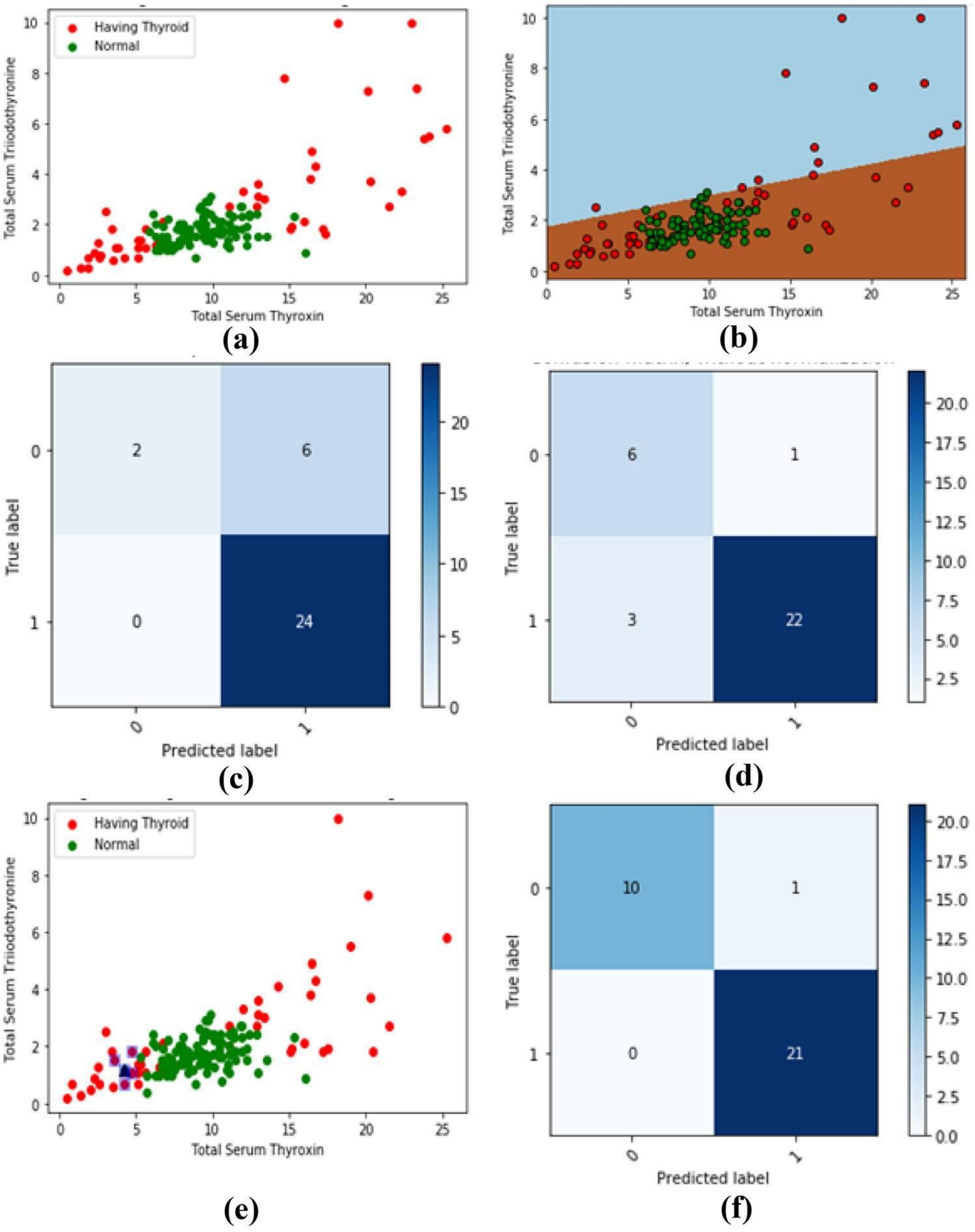
	Logistic regression classification (%)	**Decision tree classification (%)**	***k*-NN classifier (%)**
Test miscIclassification percentage	18.75	12.5	3.125
Validation misclassification percentage	15.625 .	3.125	6.25
Accuracy	81.25	87.5	96.875

Table 2 Compare with previous work

Researc algorithms	**Decision tree accuracy**	***k*NN acc Accuracy**
Ankita Ty Ritika mehra	75.76% (Much lower accuracy)	98.62% (lilittle better accuracy)
Proposed method	87.5%	96.875%

Table 3 Compare with previous work

Researc algorithms	**Decision tree accuracy**	***k*NN acc Accuracy**
Rafi khan	98.89% (B better accuracy)	91.62% (much lower accuracy)
Proposed method	87.5%	96.87%



(a) (b) (c) (d) (e) (f)

Fig.1 **(a)** Visualization of data set. **(b)** Visualization decision boundary of logistic regression model. **(c)** Confusion matrix of logistic model. **(d)** Confusion matrix of decision tree. **(e)** Working of the *k*NN algorithm. **(f)** Confusion matrix of *k*NN model

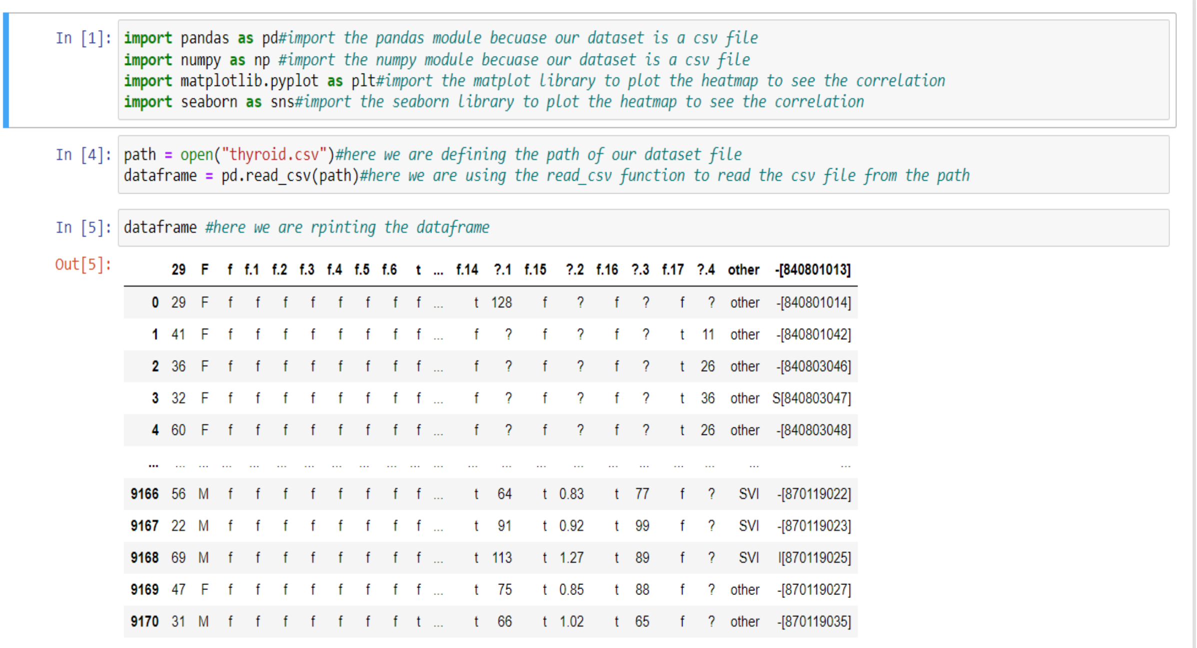
From our examination work, it is demonstrated the way that how could thyroid illness be anticipated and give an intuition how to apply the calculated relapse, choice tree order and kNN calculations. As indicated by the informational collection, the accompanying outcomes results are gotten.

The outcome (Table 1) shows that the kNN classifier is a superior calculation for this informational collection in thyroid sickness expectation.

The proficiency of a calculation relies on the informational index and its highlights chosen for the expectation. A few papers composed during 2018-2020 have less exactness than proposed calculations, and a few calculations have a superior precision which is because of the informational collection they have picked. The paper given has shown less exactness in the event of choice tree, while if there should arise an occurrence of kNN they have better precision displayed in Table 2: compare with previous work.

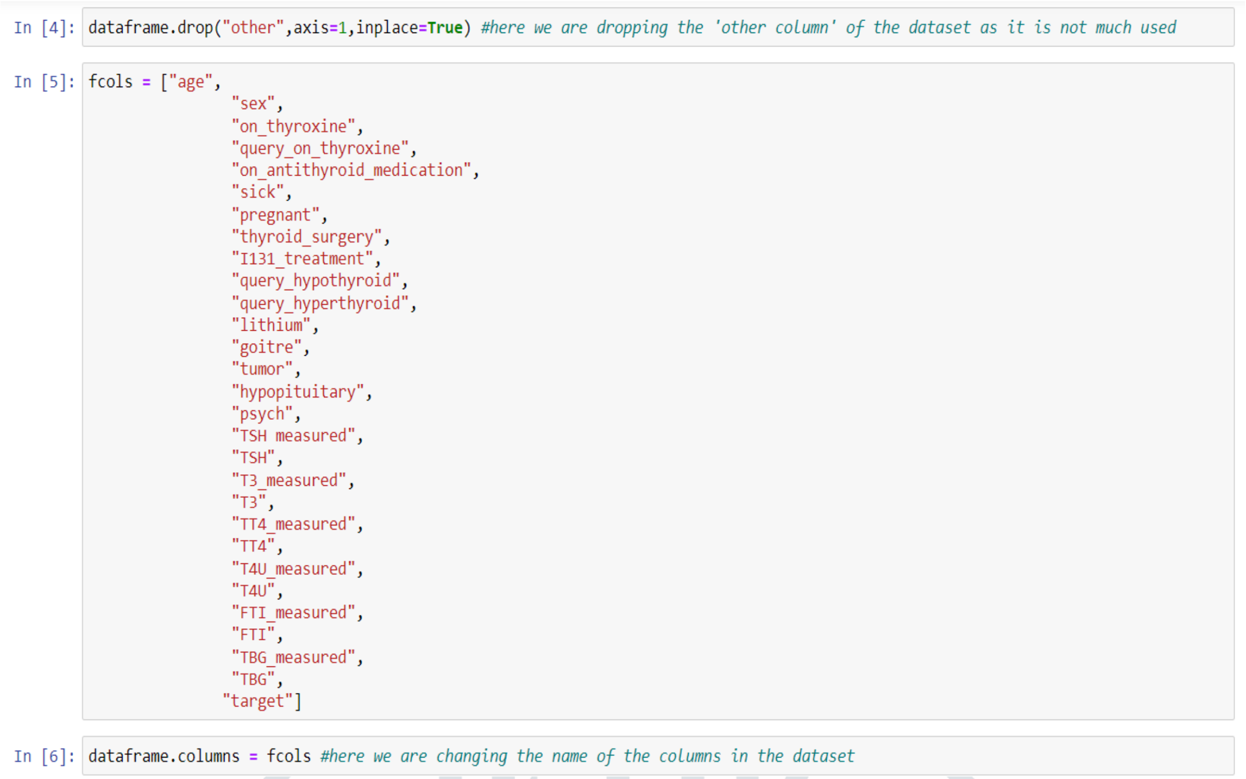
The UCI thyroid vault itself contains numerous informational indexes for thyroid illness. For proposed work, "new-thyroid" informational index has been taken. The paper creators could have taken various informational index of a similar UCI thyroid storehouse. This is the explanation of variety of result. One more work has shown substantially less precision if there should be an occurrence of kNN (91.82%) while choice tree has a superior exactness of 98.89% addressed in Table 3: compare with previous work.

Code & Output:-

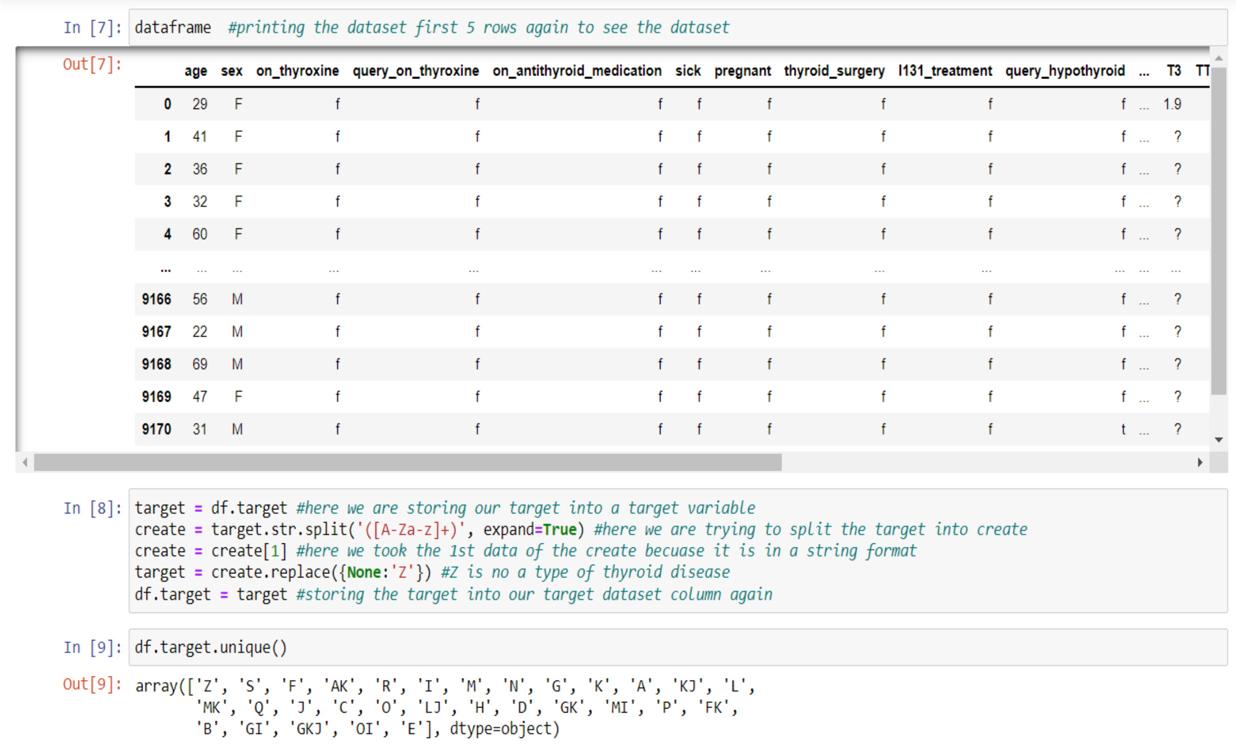


```
In [1]: import pandas as pd#import the pandas module becuase our dataset is a csv file
        import numpy as np #import the numpy module becuase our dataset is a csv file
        import matplotlib.pyplot as plt#import the matplot library to plot the heatmap to see the correlation
        import seaborn as sns#import the seaborn library to plot the heatmap to see the correlation

In [4]: path = open("thyroid.csv")#here we are defining the path of our dataset file
        dataframe = pd.read_csv(path)#here we are using the read_csv function to read the csv file from the path

In [5]: dataframe #here we are rpinting the dataframe

Out[5]:
```

	29	F	f	f.1	f.2	f.3	f.4	f.5	f.6	t	...	f.14	?.1	f.15	?.2	f.16	?.3	f.17	?.4	other	-[840801013]
0	29	F	f	f	f	f	f	f	f	f	...	t	128	f	?	f	?	f	?	other	-[840801014]
1	41	F	f	f	f	f	f	f	f	f	...	f	?	f	?	f	?	t	11	other	-[840801042]
2	36	F	f	f	f	f	f	f	f	f	...	f	?	f	?	f	?	t	26	other	-[840803046]
3	32	F	f	f	f	f	f	f	f	f	...	f	?	f	?	f	?	t	36	other	S[840803047]
4	60	F	f	f	f	f	f	f	f	f	...	f	?	f	?	f	?	t	26	other	-[840803048]
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9166	56	M	f	f	f	f	f	f	f	f	...	t	64	t	0.83	t	77	f	?	SVI	-[870119022]
9167	22	M	f	f	f	f	f	f	f	f	...	t	91	t	0.92	t	99	f	?	SVI	-[870119023]
9168	69	M	f	f	f	f	f	f	f	f	...	t	113	t	1.27	t	89	f	?	SVI	I[870119025]
9169	47	F	f	f	f	f	f	f	f	f	...	t	75	t	0.85	t	88	f	?	other	-[870119027]
9170	31	M	f	f	f	f	f	f	f	t	...	t	66	t	1.02	t	65	f	?	other	-[870119035]



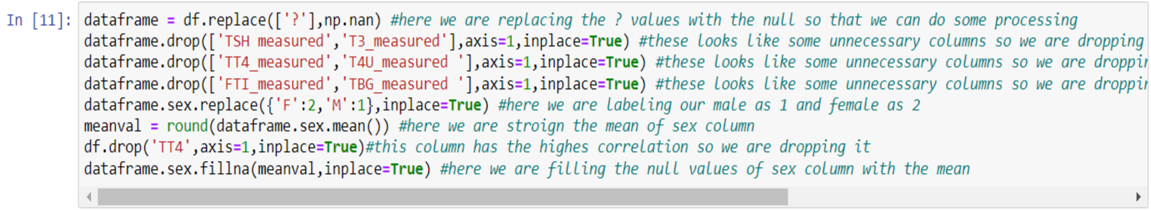
```
In [4]: dataframe.drop("other",axis=1,inplace=True) #here we are dropping the 'other column' of the dataset as it is not much used

In [5]: fcols = ["age",
                 "sex",
                 "on_thyroxine",
                 "query_on_thyroxine",
                 "on_antithyroid_medication",
                 "sick",
                 "pregnant",
                 "thyroid_surgery",
                 "I131_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",
                 "TBG_measured",
                 "TBG",
                "target"]

In [6]: dataframe.columns = fcols #here we are changing the name of the columns in the dataset
```



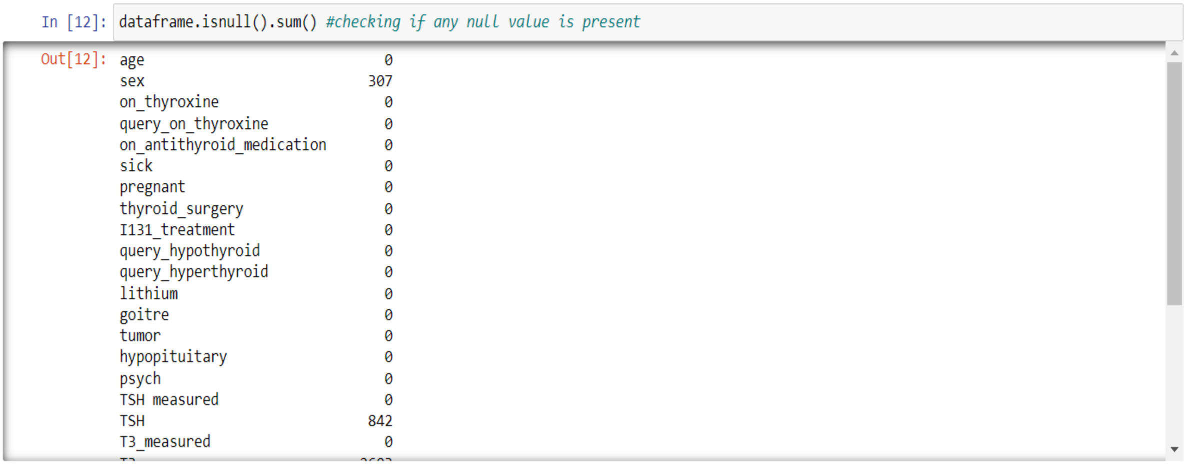
```
In [7]: dataframe  #printing the dataset first 5 rows again to see the dataset
Out[7]:
```

	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_medication	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	...	T3	TT
0	29	F	f	f	f	f	f	f	f	f	...	1.9	
1	41	F	f	f	f	f	f	f	f	f	...	?	
2	36	F	f	f	f	f	f	f	f	f	...	?	
3	32	F	f	f	f	f	f	f	f	f	...	?	
4	60	F	f	f	f	f	f	f	f	f	...	?	
...	...	...	...	...	...	...	...	...	...	...	...	...	
9166	56	M	f	f	f	f	f	f	f	f	...	?	
9167	22	M	f	f	f	f	f	f	f	f	...	?	
9168	69	M	f	f	f	f	f	f	f	f	...	?	
9169	47	F	f	f	f	f	f	f	f	f	...	?	
9170	31	M	f	f	f	f	f	f	f	t	...	?	

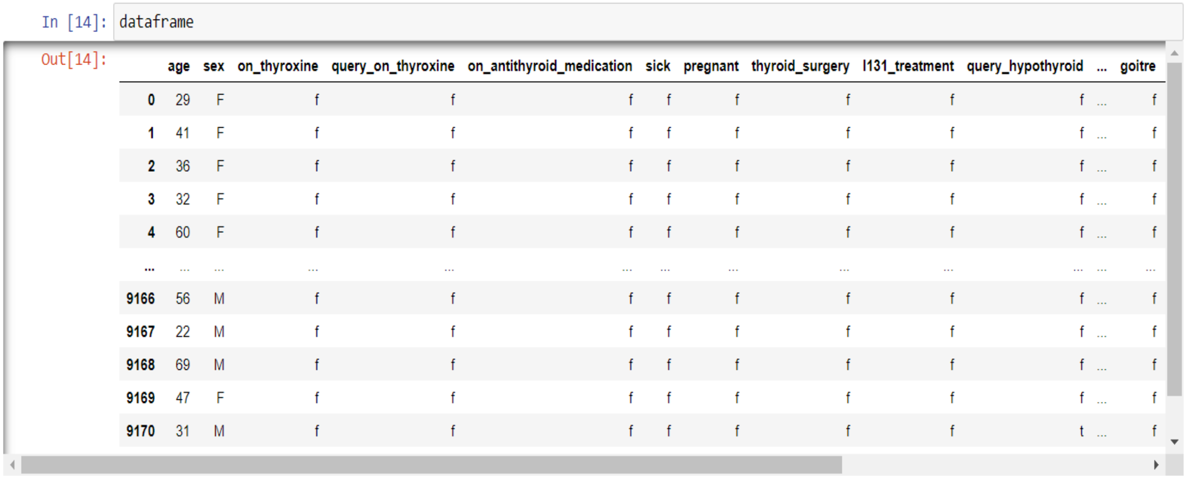
```
In [8]: target = df.target #here we are storing our target into a target variable
        create = target.str.split('([A-Za-z]+)', expand=True) #here we are trying to split the target into create
        create = create[1] #here we took the 1st data of the create becuase it is in a string format
        target = create.replace({None:'Z'}) #Z is no a type of thyroid disease
        df.target = target #storing the target into our target dataset column again

In [9]: df.target.unique()

Out[9]: array(['Z', 'S', 'F', 'AK', 'R', 'I', 'M', 'N', 'G', 'K', 'A', 'KJ', 'L',
               'MK', 'Q', 'J', 'C', 'O', 'LJ', 'H', 'D', 'GK', 'MI', 'P', 'FK',
               'B', 'GI', 'GKJ', 'OI', 'E'], dtype=object)
```



```
In [11]: dataframe = df.replace(['?'],np.nan) #here we are replacing the ? values with the null so that we can do some processing
         dataframe.drop(['TSH measured','T3_measured'],axis=1,inplace=True) #these looks like some unnecessary columns so we are dropping
         dataframe.drop(['TT4_measured','T4U_measured '],axis=1,inplace=True) #these looks like some unnecessary columns so we are droppi
         dataframe.drop(['FTI_measured','TBG_measured '],axis=1,inplace=True) #these looks like some unnecessary columns so we are droppi
         dataframe.sex.replace({'F':2,'M':1},inplace=True) #here we are labeling our male as 1 and female as 2
         meanval = round(dataframe.sex.mean()) #here we are stroign the mean of sex column
         df.drop('TT4',axis=1,inplace=True)#this column has the highes correlation so we are dropping it
         dataframe.sex.fillna(meanval,inplace=True) #here we are filling the null values of sex column with the mean
```



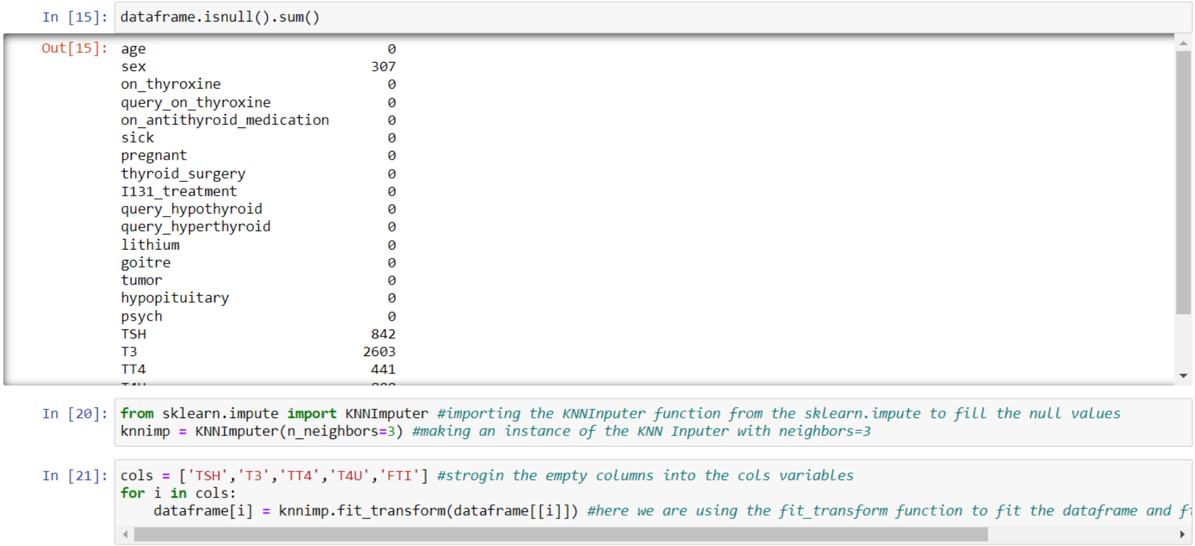
```
In [12]: dataframe.isnull().sum() #checking if any null value is present

Out[12]: 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 measured                   0
         TSH                          842
         T3_measured                    0
         T3                          2603
```



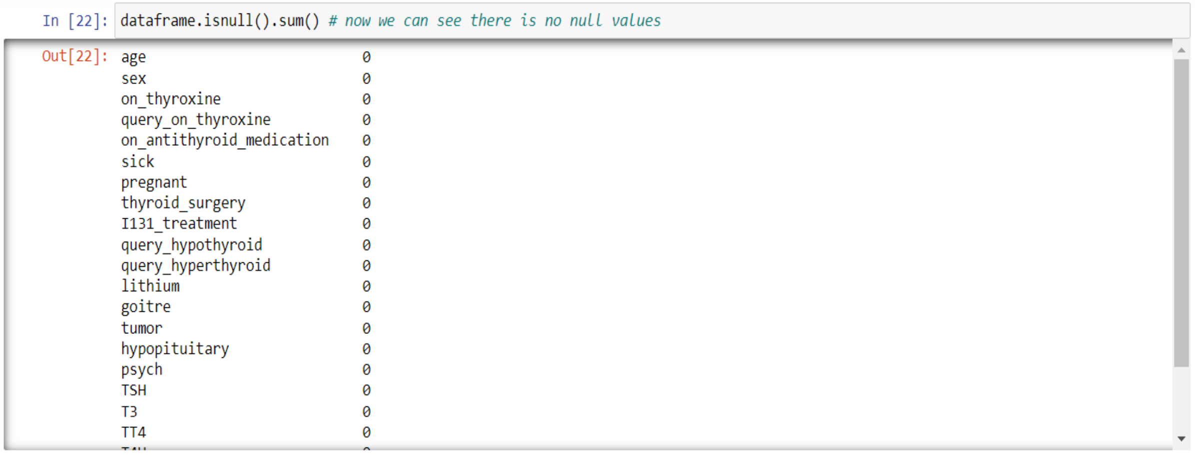
```
In [14]: dataframe
```

Out[14]:

	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_medication	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	...	goitre
0	29	F	f	f	f	f	f	f	f	f	...	f
1	41	F	f	f	f	f	f	f	f	f	...	f
2	36	F	f	f	f	f	f	f	f	f	...	f
3	32	F	f	f	f	f	f	f	f	f	...	f
4	60	F	f	f	f	f	f	f	f	f	...	f
...	...	...	...	...	...	...	...	...	...	...	...	...
9166	56	M	f	f	f	f	f	f	f	f	...	f
9167	22	M	f	f	f	f	f	f	f	f	...	f
9168	69	M	f	f	f	f	f	f	f	f	...	f
9169	47	F	f	f	f	f	f	f	f	f	...	f
9170	31	M	f	f	f	f	f	f	f	t	...	f

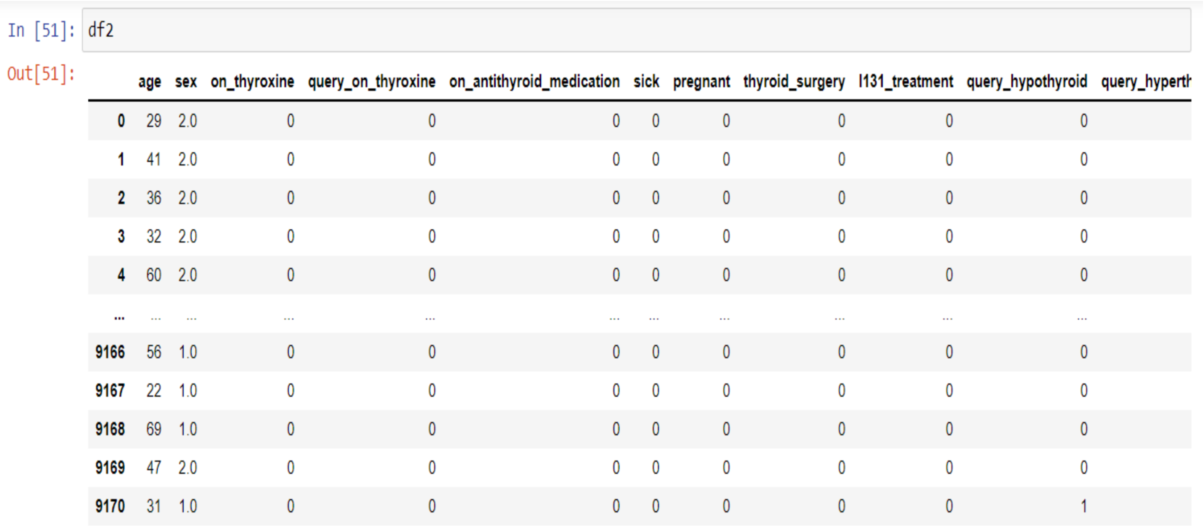


```
In [15]: dataframe.isnull().sum()

Out[15]: 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

In [20]: from sklearn.impute import KNNImputer #importing the KNNInputer function from the sklearn.impute to fill the null values
         knnimp = KNNImputer(n_neighbors=3) #making an instance of the KNN Inputer with neighbors=3

In [21]: cols = ['TSH','T3','TT4','T4U','FTI'] #strogin the empty columns into the cols variables
         for i in cols:
             dataframe[i] = knnimp.fit_transform(dataframe[[i]]) #here we are using the fit_transform function to fit the dataframe and fi
```



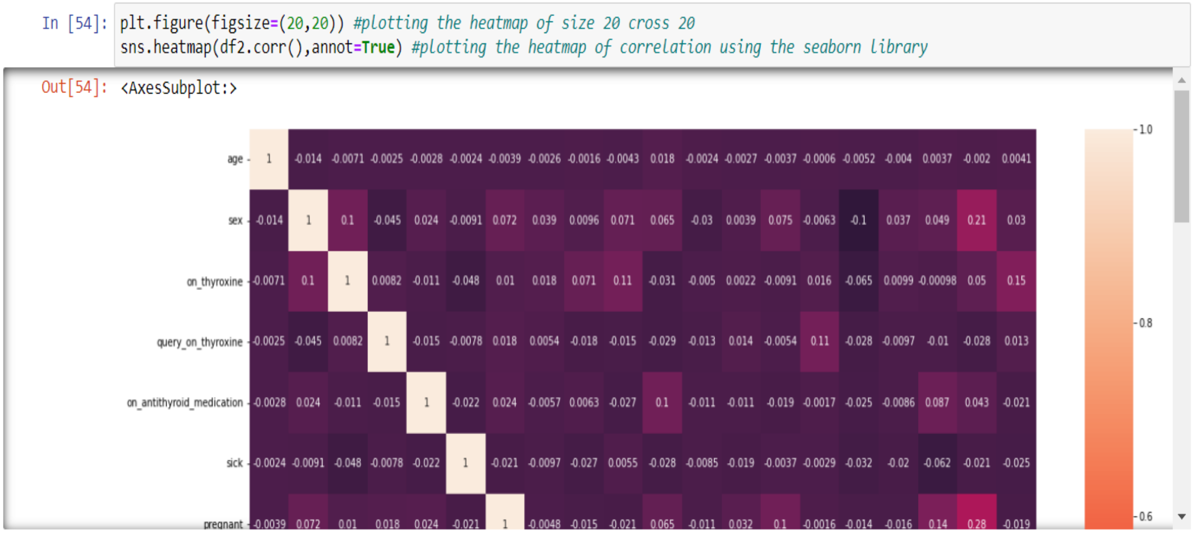
```
In [22]: dataframe.isnull().sum() # now we can see there is no null values

Out[22]: 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
         hypopituitary                0
         psych                        0
         TSH                          0
         T3                           0
         TT4                          0
         T4U                          0
```

```
In [32]: df2 = df.drop('target',axis=1) #making our x dataset by dropping our target column
         y = df.target #storing our target column into y column
```



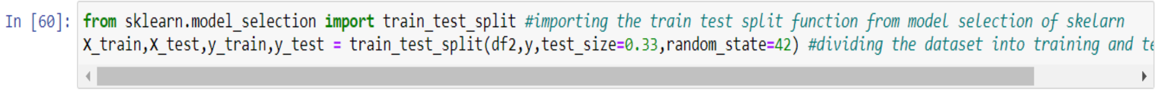
```
In [51]: df2

Out[51]:
```

	age	sex	on_thyroxine	query_on_thyroxine	on_antithyroid_medication	sick	pregnant	thyroid_surgery	I131_treatment	query_hypothyroid	query_hypertl
0	29	2.0	0	0	0	0	0	0	0	0	
1	41	2.0	0	0	0	0	0	0	0	0	
2	36	2.0	0	0	0	0	0	0	0	0	
3	32	2.0	0	0	0	0	0	0	0	0	
4	60	2.0	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	
9166	56	1.0	0	0	0	0	0	0	0	0	
9167	22	1.0	0	0	0	0	0	0	0	0	
9168	69	1.0	0	0	0	0	0	0	0	0	
9169	47	2.0	0	0	0	0	0	0	0	0	
9170	31	1.0	0	0	0	0	0	0	0	1	

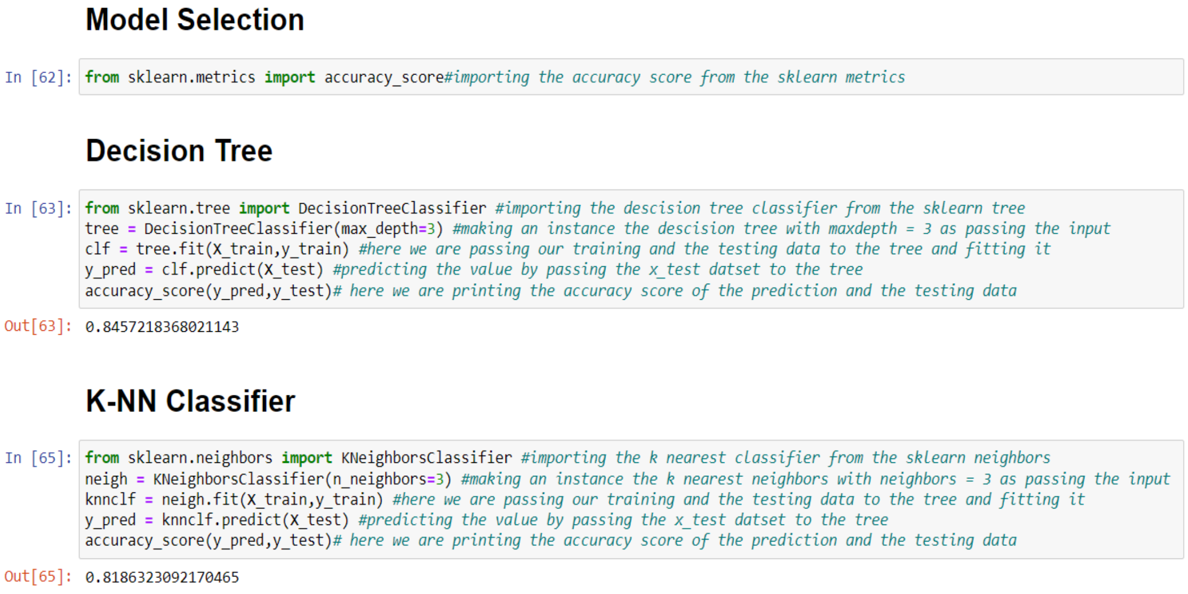


```
In [54]: plt.figure(figsize=(20,20)) #plotting the heatmap of size 20 cross 20
         sns.heatmap(df2.corr(),annot=True) #plotting the heatmap of correlation using the seaborn library

Out[54]: <AxesSubplot:>
```

age	1	-0.014	-0.0071	-0.0025	-0.0028	-0.0024	-0.0039	-0.0026	-0.0016	-0.0043	0.018	-0.0024	-0.0027	-0.0037	-0.0006	-0.0052	-0.004	0.0037	-0.002	0.0041
sex	-0.014	1	0.1	-0.045	0.024	-0.0091	0.072	0.039	0.0096	0.071	0.065	-0.03	0.0039	0.075	-0.0063	-0.1	0.037	0.049	0.21	0.03
on_thyroxine	-0.0071	0.1	1	0.0082	-0.011	-0.048	0.01	0.018	0.071	0.11	-0.031	-0.005	0.0022	-0.0091	0.016	-0.065	0.0099	-0.00098	0.05	0.15
query_on_thyroxine	-0.0025	-0.045	0.0082	1	-0.015	-0.0078	0.018	0.0054	-0.018	-0.015	-0.029	-0.013	0.014	-0.0054	0.11	-0.028	-0.0097	-0.01	-0.028	0.013
on_antithyroid_medication	-0.0028	0.024	-0.011	-0.015	1	-0.022	0.024	-0.0057	0.0063	-0.027	0.1	-0.011	-0.011	-0.019	-0.0017	-0.025	-0.0086	0.087	0.043	-0.021
sick	-0.0024	-0.0091	-0.048	-0.0078	-0.022	1	-0.021	-0.0097	-0.027	0.0055	-0.028	-0.0085	-0.019	-0.0037	-0.0029	-0.032	-0.02	-0.062	-0.021	-0.025
pregnant	-0.0039	0.072	0.01	0.018	0.024	-0.021	1	-0.0048	-0.015	-0.021	0.065	-0.011	0.032	0.1	-0.0016	-0.014	-0.016	0.14	0.28	-0.019



```
In [60]: from sklearn.model_selection import train_test_split #importing the train test split function from model selection of skelarn
         X_train,X_test,y_train,y_test = train_test_split(df2,y,test_size=0.33,random_state=42) #dividing the dataset into training and te
```



Model Selection

```
In [62]: from sklearn.metrics import accuracy_score#importing the accuracy score from the sklearn metrics
```

Decision Tree

```
In [63]: from sklearn.tree import DecisionTreeClassifier #importing the descision tree classifier from the sklearn tree
         tree = DecisionTreeClassifier(max_depth=3) #making an instance the descision tree with maxdepth = 3 as passing the input
         clf = tree.fit(X_train,y_train) #here we are passing our training and the testing data to the tree and fitting it
         y_pred = clf.predict(X_test) #predicting the value by passing the x_test datset to the tree
         accuracy_score(y_pred,y_test)# here we are printing the accuracy score of the prediction and the testing data

Out[63]: 0.8457218368021143
```

K-NN Classifier

```
In [65]: from sklearn.neighbors import KNeighborsClassifier #importing the k nearest classifier from the sklearn neighbors
         neigh = KNeighborsClassifier(n_neighbors=3) #making an instance the k nearest neighbors with neighbors = 3 as passing the input
         knnclf = neigh.fit(X_train,y_train) #here we are passing our training and the testing data to the tree and fitting it
         y_pred = knnclf.predict(X_test) #predicting the value by passing the x_test datset to the tree
         accuracy_score(y_pred,y_test)# here we are printing the accuracy score of the prediction and the testing data

Out[65]: 0.8186323092170465
```

Conclusion & Future Work

Rafikhan has utilized a clinical information of Kashmir of 807 patients and UCI thyroid storehouse of "new thyroid" has just 215 examples. Proposed strategy has not taken this informational index for thyroid expectation; it will consider in future work and measure exactness utilizing choice tree and kNN. Consequently, as per the informational collection which is utilized in this work, the exactness acquired is palatable.

The ongoing situation is of the creating of the models that assistance in the different areas of life utilizing the AI. The accessibility of information and its age step by step expanded an opportunity for the PC researchers to make expectation and examination on such informational collections that improve the human existence and solace. This study is worry with this inspiration. The forecast and grouping of any information relies upon the informational collection itself and the different algorithms that are utilized. On the off chance that anybody coordinates a superior informational index of continuous and applies different other machine inclining and profound learning calculations, for example, SVM, Naive Bayes, auto encoders, ANNs and CNNs then, at that point, further improved results might be accomplished.

References

1. Chen Ling, Li Xue, Sheng Quan Z, Peng W-C (2016) Mining health examination records—a graph-based approach. IEEE Trans Knowl Discov Eng 28:2423–2437

2. Temurtas F (2009) A comparative study on thyroid disease diagnosis using neural networks. Expert Syst Appl 36:944–949

3. Ulutagay G (2012) Modeling of thyroid disease: a fuzzy inference system approach. Wulfenia J 19(1):346–357

4. Monaco Fabrizio (2003) Classification of thyroid diseases: sug- gestions for a revision. J Clin Endocrinol Metab 88:1428–1432

5. Ionita I, Ionita L (2016) Prediction of thyroid disease using data mining techniques. Broad Res Artif Intell Neurosci 7(3):115–124

6. Gorade SM, Deo A, Purohit P (2017) A study of some data mining classification technique. Int Res J Eng Technol 4(4):3112–3115

7. Bichler M, Kiss C (2004) A comparison of logistic regression, *k*- nearest neighbor, and decision tree induction for campaign management. In: Proceedings of the tenth Americas conference on information systems, New York

8. http://archive.ics.uci.edu/ml/machine-learning-databases/thyroid- disease/

9. Peng CYJ, Lee KL, Ingersoll GM (2002) An introduction to logistic regression analysis and reporting. J Educ Res 96(1):3–14