Low Level Document (LLD)

Project Title – Thyroid Disease Detection

Technologies – Machine Learning Technology

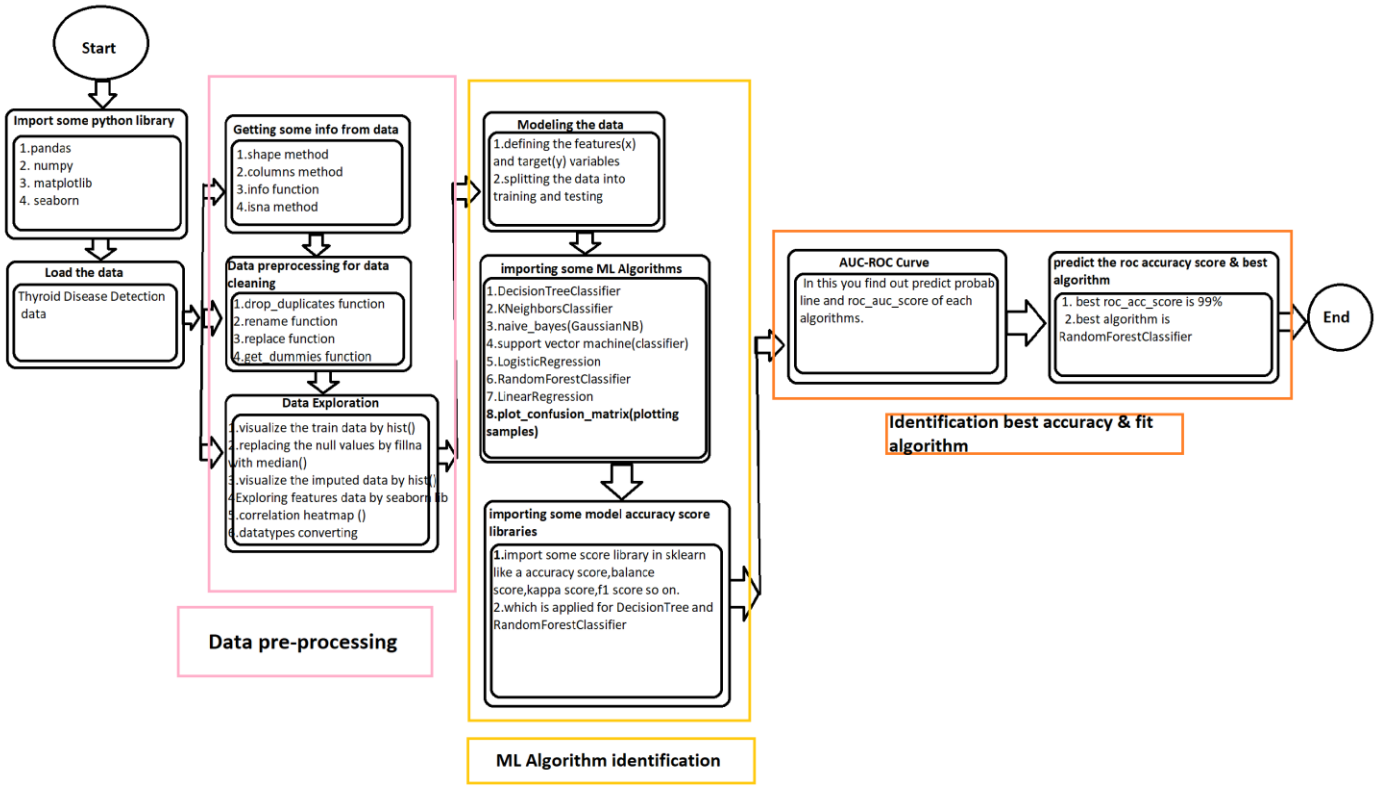
Domain – Healthcare

Authorised by – iNeuron.ai

Low Level Document (LLD)

System Design

Simple Diagram with layers:



➢ Introduction

The Problem

Thyroid disease is a common cause of medical diagnosis and prediction, with an onset that is difficult to forecast in medical research. The thyroid gland is one of our body's most vital organs. Thyroid hormone releases are responsible for metabolic regulation. Hyperthyroidism and hypothyroidism are one of the two common diseases of the thyroid that releases thyroid hormones in regulating the rate of body's metabolism. **When your thyroid makes either too much or too little of these important hormones, it's called a thyroid disease.**

Low Level Document (LLD)

➢ Awareness

When your thyroid doesn't work properly, it can impact your entire body. If your body makes too much thyroid hormone, you can develop a condition called **hyperthyroidism**. If your body makes too little thyroid hormone, it's called **hypothyroidism**. Both conditions are serious and need to be treated by your healthcare provider.

➢ How is thyroid disease diagnosed?

Sometimes, thyroid disease can be difficult to diagnose because the symptoms are easily confused with those of other conditions.
Therefore I have developed ML Algorithm which is given prediction with 100 % accuracy. That's why it is very helpful determine tests.

➢ My Approach:

The classical machine learning tasks like Data Exploration, Data Cleaning, Feature Engineering, Model Building and Model Testing. Try out different machine learning algorithms that's best fit for the above case.

➢ Content:

There are many layers within the content :--

1. Import some python libraries
2. Load the data
3. Getting some info from data
4. Data pre processing for data cleaning
5. Data Exploration

6. Modelling the data
7. Importing some ML Algorithms
8. Importing some model accuracy score library
9. AUC-ROC Curve
10. Predict the roc accuracy score and best algorithm

1. **Import some python libraries** - these some libraires have to very important to execute for load the data. You cannot execute difference types of python function without load these some python libraries.

   I. **Pandas:**
   this software library used for data manipulation and analysis.

   II. **Numpy:**
   Numpy is used for working with array and numeric data.

   III. **Matplotlib:**
   Matplotlib is used for data visualization and graphical plotting.

   IV. **Seaborn:**
   Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

   Implement:-

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

2. **Load the data -** data is important part of the project what you want to prediction base on the data.

    - Thyroid Disease Detection Data : executed the data from CSV file.

    Implement:-

    ```
    df = pd.read_csv("d:/data/hypothyroid.csv")
    ```

3. **Getting some info from data -** Initially you have to gain the information from the data. Whatever you find the info that is according to your prediction.

    I. **Shape method:**
    data shape is large in which 30 columns and 3772 rows.

    II. **Columns method:**
    Whatever columns in your data through using this method have represent all features column.

    III. **Info function:**
    Using this function get a idea which types of data type of each columns feature in out data.

    IV. **isna method:**
    why is using this method because we have getting a idea that how many NAN values there in each and every columns data.

    Implement:-

```
df.shape
df.columns
df.info()
df.isna().sum()
```

4. **Data pre-processing for data cleaning -** this part is more carefully when you go to prepare the model. Remove an unwanted thing which is use for my prediction data.

   I. **Drop_duplicates function:**

   If in your data duplicates values is there so you can use this method and remove.

   II. **Rename function:**
   I have used this function for replace the column name of Target values. This your on choice. And as well as replace function same work here.

   III. **Get_dummies function:**
   In this function you have to do separate of each column which is depend of variable contain in previous feature column.

   Implement:

```
df.drop_duplicates(keep=False,inplace=True)
df.rename(columns={'binaryClass':'Label'},inplace=True)
df = pd.get_dummies(df,columns=["referral source"])
```

5. **Data Exploration -** in this whatever structure or unstructured data have to do explored in visualization. By which you can easily analysis data.
   We have used hist function with visualize the all features in sequence. And after remove missing values with substitute fillna () with median function also use hist visualize function. After from I have used countplot method with seaborn libraries most of feature data visualize with this seaborn library method. ahead I have also used correlation heatmap function.
   Lastly whoever data gave it had to convert into int and float dtype.
   Implement:

```
#replacing null values by median because all these features show a skewed distribution.
df['age'].fillna(df['age'].median(),inplace=True)
df['sex'].fillna(df['sex'].median(),inplace=True)
df['TSH'].fillna(df['TSH'].median(),inplace=True)
df['T3'].fillna(df['T3'].median(),inplace=True)
df['TT4'].fillna(df['TT4'].median(),inplace=True)
df['T4U'].fillna(df['T4U'].median(),inplace=True)
df['FTI'].fillna(df['FTI'].median(),inplace=True)
#data type converting
df['age'] = df['age'].astype('int64')
df['FTI'] = df['FTI'].astype('float')
df['TSH'] = df['TSH'].astype('float')
df['T3'] = df['T3'].astype('float')
df['TT4'] = df['TT4'].astype('float')
df['T4U'] = df['T4U'].astype('float')
df['sex'] = df['sex'].astype('int64')
```

6. **Modelling the data:** In this you have to take two steps for modelling a data.

    I. **defining the features (x) and target (y) variables:** when your data is cleaned then you build a training and testing data which is x and y.
    II. **Splitting the data into training and testing:** When you data is built after that you have to take splitting a data for training and testing and set the test data which is my prediction.

    Implement:

```
# defining the features and target variables
x = df.drop('Label',axis=1)
y = df['Label']

#Splitting the data into training and testing
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test =
train_test_split(x,y,test_size=0.2,random_state=10)
```

7. **Importing some ML Algorithms:** in this you do try to some machine algorithm. For whom you have to do import algorithms libraries.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
```

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import plot_confusion_matrix

#Naive bayes algorithm
NB = GaussianNB()
NB.fit(x_train,y_train)
NB.score(x_test,y_test)

# LinearRegression algorithm
reg = LinearRegression()
reg.fit(x_train,y_train)
reg.score(x_test,y_test)

# KNN algorithm
KNN = KNeighborsClassifier(n_neighbors=3,weights='distance')
KNN.fit(x_train,y_train)
KNN.score(x_test,y_test)

#LogisticRegression algorithm
LR = LogisticRegression(random_state=50,max_iter=100)
LR.fit(x_train,y_train)
LR.score(x_test,y_test)

# support vector classifier
clf = SVC(kernel='linear')
clf.fit(x_train,y_train)
clf.score(x_test,y_test)

#Decision Tree algorithm
DT = DecisionTreeClassifier(criterion='entropy')
DT.fit(x_train,y_train)
DT.score(x_test,y_test)
```

```
#RandomForestClassifier algorithm
RFC = RandomForestClassifier(criterion='entropy')
RFC.fit(x_train,y_train)
RFC.score(x_test,y_test)
```

8. **Importing some model accuracy score library:** if you don't have to give clarity as accuracy from difference types of algorithms then you should use some types of accuracy score library. That's why I have used difference types of accuracy score library with decision tree and randomforestclassifier algorithms. Because these two algorithms have given best performance apart from others. Like this types of accuracy score used:-

```
from sklearn.metrics import
(accuracy_score,balanced_accuracy_score,
cohen_kappa_score,precision_score,recall_score,
f1_score,roc_curve,roc_auc_score)
```

9. **AUC-ROC Curve:** still don't have given clarity that which one is best algorithm? that's why i'm using ROC-CURVE method. only these types of algorithms I have used for ROC Curve method:---
    I. Decision tree algorithm
    II. Random Forest Classifier
    III. Linear Regression
    IV. K Neighbours Classifier
10. **Predict the ROC Accuracy score and best algorithm:** after used AUC ROC curve method you have given accuracy prediction by roc accuracy score and ROC-Curve.

Low Level Document (LLD)

I. Best roc accuracy score is 99% even more than 99%
II. Best fit algorithm is **Random Forest Classifier** for almost 100% good prediction accuracy for Thyroid Disease Detection.