Low-Level Design

Thyroid Disease Detection

Developed by	Saurabh Gupta
Version	1.0
Date	24-06-2023

Table of Contents

1 Document Change/History Control

Version	Date	Done By	Review By	Remarks
1.0	24-06-2023	Saurabh Gupta		

2 Reviews:

Version	Date	Reviewer	Remarks

3 Approval Status:

Version	Review Date	Reviewed By	Approved By	Remarks

4 Introduction

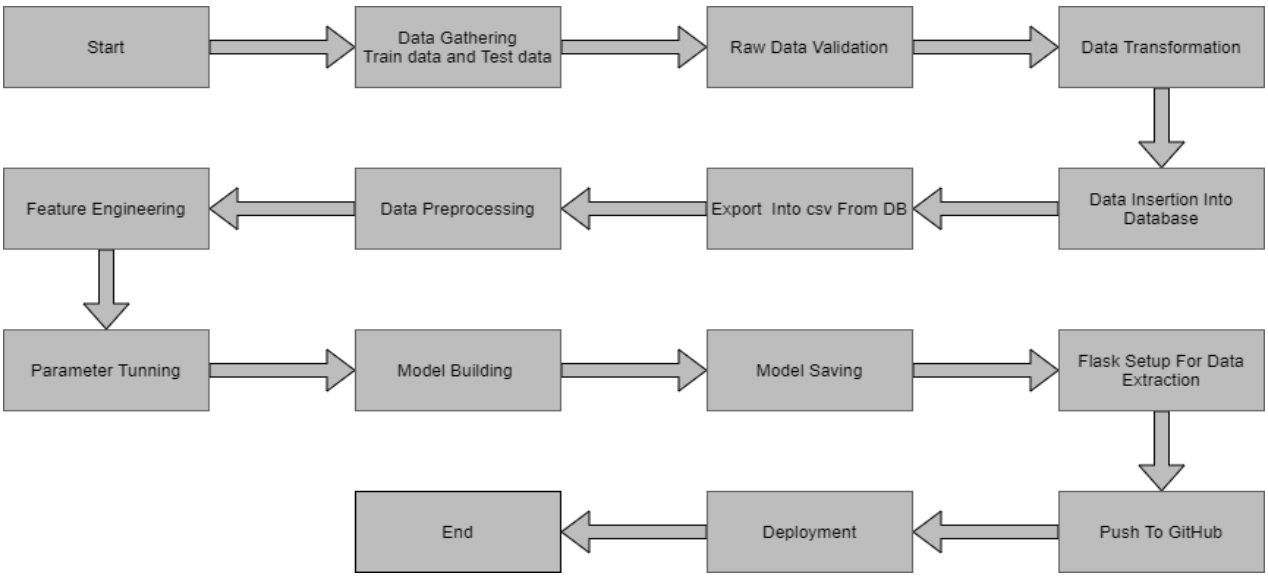
4.1 What is Low-Level Design Document.

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for **'Stores Sales Prediction'**. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document.

4.2 Scope

Low-level design (LLD) is a component-level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required software architecture, source code, and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work.

5 Architecture



6 Architecture Description

The dataset looks like as follow:

	count	unique	top	freq
age	3772	94	59	95
sex	3772	3	F	2480
on thyroxine	3772	2	f	3308
query on thyroxine	3772	2	f	3722
on antithyroid medication	3772	2	f	3729
sick	3772	2	f	3625
pregnant	3772	2	f	3719
thyroid surgery	3772	2	f	3719
I131 treatment	3772	2	f	3713
query hypothyroid	3772	2	f	3538
query hyperthyroid	3772	2	f	3535
lithium	3772	2	f	3754
goitre	3772	2	f	3738
tumor	3772	2	f	3676
hypopituitary	3772	2	f	3771
psych	3772	2	f	3588
TSH measured	3772	2	t	3403
TSH	3772	288	?	369
T3 measured	3772	2	t	3003
T3	3772	70	?	769
TT4 measured	3772	2	t	3541
TT4	3772	242	?	231
T4U measured	3772	2	t	3385
T4U	3772	147	?	387
FTI measured	3772	2	t	3387
FTI	3772	235	?	385
TBG measured	3772	1	f	3772
TBG	3772	1	?	3772
referral source	3772	5	other	2201
binaryClass	3772	2	P	3481

The data set consists of object data type shown in Fig.

#	Column	Non-Null Count	Dtype
0	age	3772 non-null	object
1	sex	3772 non-null	object
2	on thyroxine	3772 non-null	object
3	query on thyroxine	3772 non-null	object
4	on antithyroid medication	3772 non-null	object
5	sick	3772 non-null	object
6	pregnant	3772 non-null	object
7	thyroid surgery	3772 non-null	object
8	I131 treatment	3772 non-null	object
9	query hypothyroid	3772 non-null	object
10	query hyperthyroid	3772 non-null	object
11	lithium	3772 non-null	object
12	goitre	3772 non-null	object
13	tumor	3772 non-null	object
14	hypopituitary	3772 non-null	object
15	psych	3772 non-null	object
16	TSH measured	3772 non-null	object
17	TSH	3772 non-null	object
18	T3 measured	3772 non-null	object
19	T3	3772 non-null	object
20	TT4 measured	3772 non-null	object
21	TT4	3772 non-null	object
22	T4U measured	3772 non-null	object
23	T4U	3772 non-null	object
24	FTI measured	3772 non-null	object
25	FTI	3772 non-null	object
26	TBG measured	3772 non-null	object
27	TBG	3772 non-null	object
28	referral source	3772 non-null	object
29	binaryClass	3772 non-null	object

6.1 Raw Data Validation

After data is loaded, various types of validation are required before we proceed further with any operation. Validations like checking for zero standard deviation for all the columns, checking for complete missing values in any columns, etc. These are required because The attributes which contain these are of no use. It will not play role in contributing to the sales of an item from respective outlets.

Like if any attribute is having zero standard deviation, it means that's all the values are the same, its mean is zero. This indicates that either the sale is increasing or decrease that attribute will remain the same. Similarly, if any attribute is having full missing values, then there is no use in taking that attribute into an account for operation. It's unnecessary increasing the chances of dimensionality curse.

6.2 Data Transformation

Before sending the data into the database, data transformation is required so that data are converted into such form with which it can easily insert into the database. In the raw data, there can be various columns of underlying patterns which also gives an in-depth knowledge about the subject of interest and provides insights into the problem. But caution should be observed with respect to data as it may contain null values, or redundant values, or various types of ambiguity, which also demands pre-processing of data.

6.3 Data Preprocessing

Preprocessing of this dataset includes doing analysis on the independent variables like checking for null values in each column and then replacing or filling them with supported appropriate data types so that analysis and model fitting is not hindered from their way to accuracy. Shown above are some of the representations obtained by using Pandas tools which tell about variable count for numerical columns and model values for categorical columns. Maximum and minimum values in numerical columns, along with their percentile values for median, play an important factor in deciding which value to be chosen at priority for further exploration tasks and analysis. Data types of different columns are used further in label processing and a one-hot encoding scheme during the model building.

6.4 Feature Engineering

After preprocessing it was found that some of the attributes are not important to the thyroid detection for the particular outlet. So those attributes are removed. Even one hot encoding is also performed to convert the categorical features into numerical features.

6.5 Balance the data set by using SMOTE

```
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state = 2)
x_train_res, y_train_res = sm.fit_resample(x_train, y_train.ravel())
```

6.6 Parameter Tuning

Parameters are tuned using GridSearchCV. Two algorithms are used in this problem, Logistic Regression, Random Forest Classifier and XGB Classifier. The parameters of these 3 algorithms are tuned and passed into the model.

6.7 Model Building

After doing all kinds of preprocessing operations mention above and performing scaling and hyperparameter tuning, the data set is passed into 3 models, Logistic Regression, Random Forest Classifier and XGB Classifier. It was found that Random Forest Classifier and XGB Classifier performs best with the 99%.

6.8 Model Saving

Model is saved using pickle library in `.sav` format.

6.9 GitHub

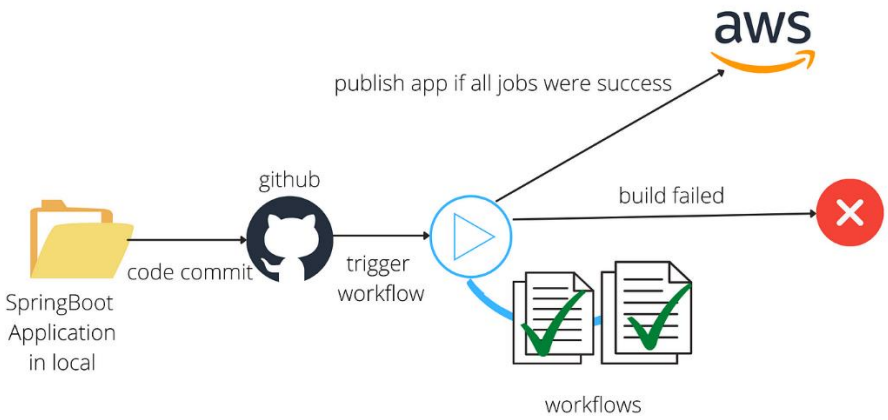
The whole project directory will be pushed into the GitHub repository.

Github: https://github.com/saurabhg2083/Thyroid_disease_detection

6.10 Deployment

The cloud environment (AWS Elastic Bean Stack) was set up and the project was deployed from GitHub into the AWS.

App link: http://thyroid-detection.eba-2nqj3nkm.ap-south-1.elasticbeanstalk.com/



7 Unit Test Cases.

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL is accessible to the user	Application-URL should be defined	Application URL should be accessible to the user
Verify whether the Application loads completely for the user when the URL is accessed	Application URL is accessible Application is deployed	The Application should load completely for the user when the URL is accessed
Verify whether a user is able to see input fields while opening the application	The user is able to see the input fields	Users should be able to see input fields on logging in
Verify whether a user is able to enter the input values.	The user is able to see the input fields	The user should be able to fill the input field
Verify whether a user gets predict button to submit the inputs	The user is able to see the input fields	Users should get Submit button to submit the inputs
Verify whether a user is presented with recommended results on clicking submit	The user is able to see the input fields. The user is able to see the submit button	Users should be presented with recommended results on clicking submit
Verify whether a result is in accordance with the input that the user has entered	The user is able to see the input fields. The user is able to see the submit button	The result should be in accordance with the input that the user has entered