iNeuron

Phishing Domain Detection (Classification [Machine Learning]).

Low Level Design

Project Member: Aman Gupta.

Introduction:-

What is low-level design document???

The goal of LLD (low-level design document) is to give internal logical

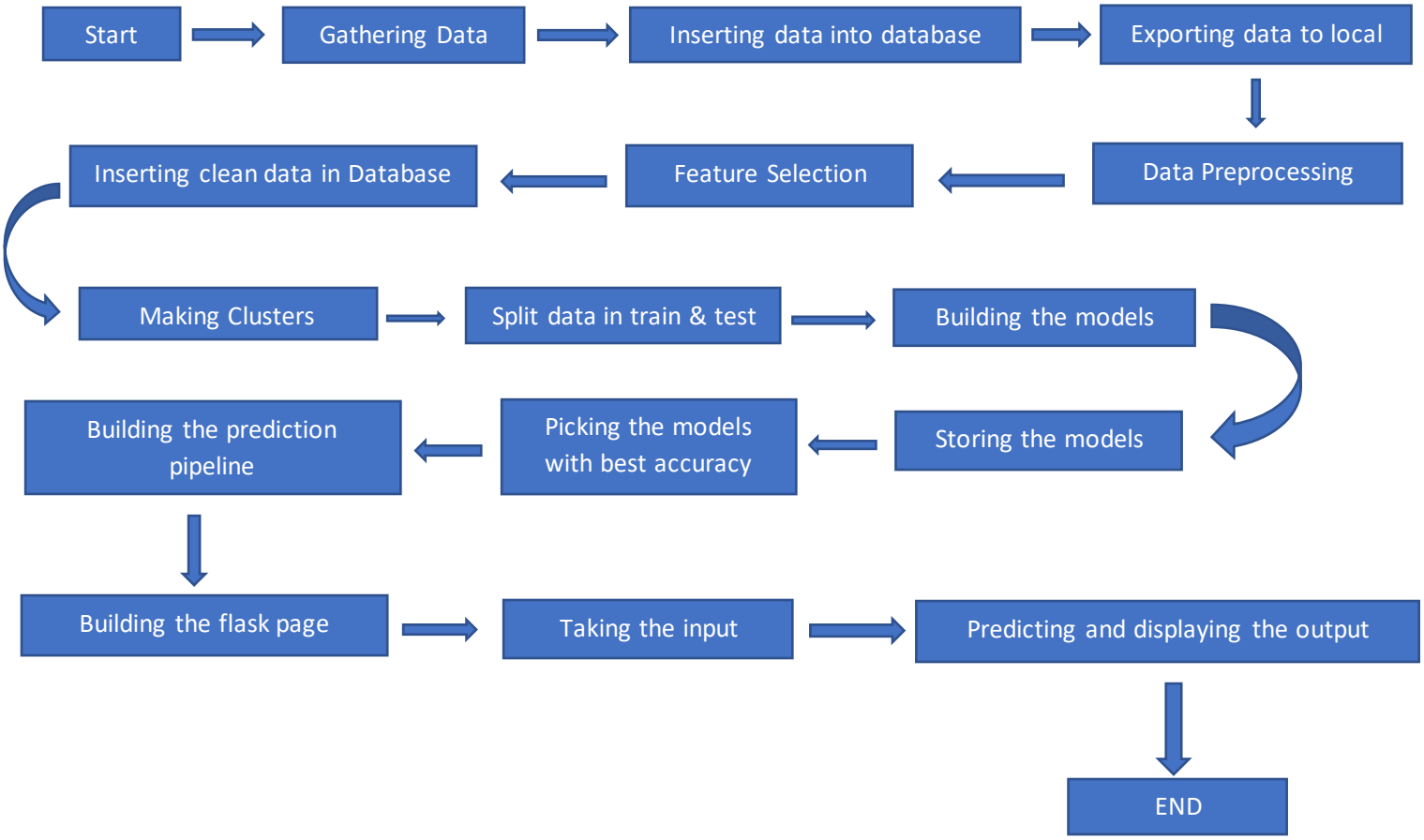
Design of the actual program code for Phishing Domain Detection.

LLD describes the class diagrams with the methods and relation between feature and level column. It describes the modules so that the programmer can directly code the program from the document.

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 algorithm. Overall, the data organization may be defined during requirements analysis and then refined during data design work.

Architecture:-



- ## Architecture Description:-

Data Description:-

This data set consist of 88,647 websites labelled as legitimate or phishing and allow the researchers to train their classification models, build phishing detection systems.

For more details of the dataset visit:

Datasets for phishing websites detection - ScienceDirect

Datasets for phishing websites detection - ScienceDirect

Data insertion into database:-

In this project we are using Cassandra database. Cassandra has limit the numbers of columns up to 75 columns so we are storing data in two tables.

Creating a connection and the storing the data into database.

Export data to local system:-

Exporting the data to the local system and making a dummy data to build the app later when all things are ready we will use the whole dataset.

Data Pre-processing:-

Dropping the duplicates and null values and some pre-processing the data.

Feature Selection:-

Selecting the best features to the good output by reducing the computation. (8 features are selected.)

Inserting the clean data into database:-

After feature selection and pre-processing storing the clean data into database again.

Clustering:-

Separating the data based on same variance to the best results.

Then building separate model for each clusters. (In this we are using 2 clusters)

Train test split and Scaling: -

Splitting the data into training and testing then scaling to bring them into same scale.

Model Building and evaluating:

Building different algorithms like Logistic Regression, SVC, Random forest, Decision tree classifier, Gradient boosting, Ada boosting. And checking the

Precision, recall, specificity and f1 score because our dataset has a ratio of 65:35 we can't use accuracy we will use recall to reduce the false negative.

Model Storing: -

After building every model and scaler object we are going to store that models in a pickle format. And using that for prediction purpose.

Selecting the best model:-

Out of all the models from all the clusters we need to select best model for all the clusters we are using recall as a scoring parameter higher the score better the model.

Building the prediction pipeline and flask app:-

In the prediction pipeline we are going to pass the input through the same process as we did while training the model. Clustering, scaling, and etc.

Taking inputs and predicting the outputs:-

All the input are coming from the flask app we need to predict the output and display the information in the flask app.

Conclusion :

It turns out model is performing well with a recall score of 78% in the cluster one and in the cluster two it is giving recall of 93% but there are some False Positives which will affect.