Architecture

Thyroid Disease Detection System

Written By	Navanshu Khare
Document Version	1.0
Last Revised Date	

Document Version Control

Change Record:

Version	Date	Author	Comments
1.0	01-06-2022	Navanshu Khare	Architecture

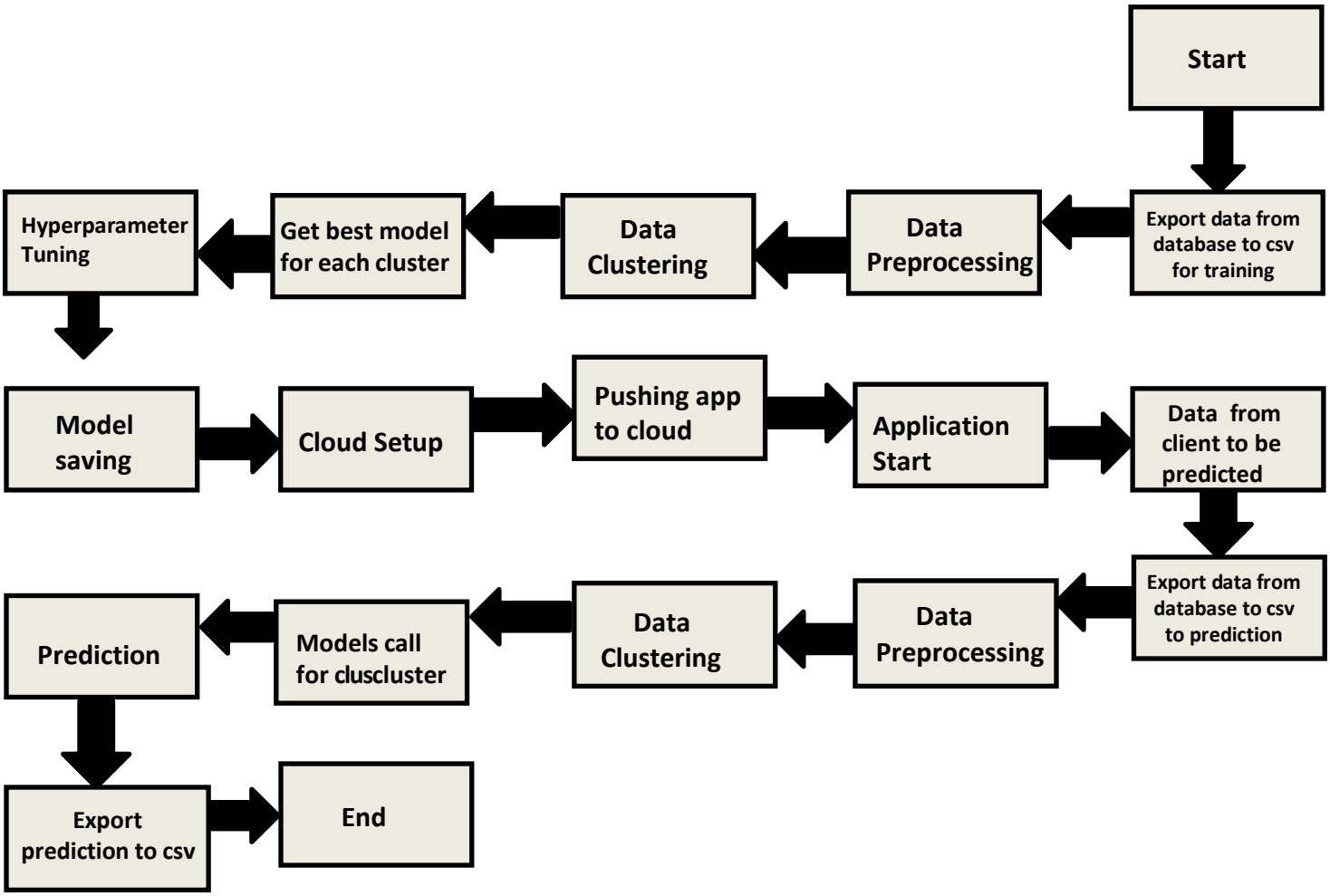
Contents

Document Version control 21

Architecture 4

2 Architecture Description 5
2.1 Data Description ..... 52.2 Export Data from DB to CSV for training ---------- 5
2.3 Data Preprocessing 5
2.4 Data Clustering ..... 52.5 Get best model of each cluster ---------- 5
2.6 Hyperparameter Tuning 5
2.7 Model saving 6
2.8 Cloud setup 6
2.9 Push App to cloud ..... 62.10 Data from client side for prediction ---------- 6
2.11 Export prediction to CSV 6

2. Architecture



3.Architecture Description

3.1 Data Description

Thyroid Disease Data Set present in UCI Machine LearningRepository is used for this project. Total 7200 instances present in different batches of data is more than enough for our requirements.

3.2 Export Data from database to CSV for Training

Exporting of all batches of data from Cassandra database into one csv file for training.

3.3 Data Preprocessing

Here, we will explore our data collection and, if necessary, execute EDA and data preprocessing, depending on the data set. To develop different modules based on our analysis and implement them for training and prediction data, we must first investigate our data set in Jupyter Notebook and decide what pre-processing and validation we need to perform, such as the imputation of null values, removing specific columns, etc.

3.4 Data Clustering

The pre-processed data will be clustered using the K-Means technique. Plotting the elbow plot allows the selection of the ideal number of clusters. Utilizing several techniques to train data in various clusters is the rationale behind clustering. The K-means model is developed using pre-processed data, and it is then stored for future prediction use.

3.5 Get best model of each cluster

Here, we'll train a variety of models on each cluster we get through data clustering to find the most effective model for each cluster.

3.6 Hyperparameter Tuning

We will perform hyperparameter tuning for each chosen model after choosing the best model for each cluster in an effort to improve model performance.

3.7 Model Saving

We will save our models after implementing hyperparameter tuning on them so that we may use them for prediction.

3.8 Cloud Setup

We'll build up the cloud to deploy the model. Here, too, we develop our Flask app and UI, and will integrate both with our models

3.9 Push app to cloud

We will push our program to the cloud to begin the application after performing cloud setup and checking the app in local machine and sanity testing.

3.10 Data from client side for prediction purpose

The cloud application is prepared for the prediction stage. The prediction data that we receive from the client side will be exported from the database and further processed using the same data cleansing techniques as for the training data. The same stages for exporting data from a database, pre-processing data, and clustering client data will also be followed. Depending on the cluster number, we will utilize our saved model to make predictions for that cluster.

3.11 Export Prediction to CSV

Finally, when all the prediction for client data is processed, We will export the prediction results to a csv file and display to download the file to the client. CSV file will contain the result of our prediction in one column