CONCRETE COMPRESSIVE STRENGTH PREDICTION

DETAILED PROJECT REPORT

REVISION No. : 1.0
LAST DATE OF REVISION : MAY 2ND,2022

PROJECT DETAIL

PROJECT TITLE	CONCRETE CEMENT STRENGTH PREDICTION
TECHNOLOGY	MACHINE LEARNING
DOMAIN	CIVIL ENGINEERING
DIFFICULTY LVL	INTERMEDIATE
PROGRAMMING LANGUAGE	PYTHON
TOOLS USED	JUPYTER NOTEBOOK, VS CODE

TABLE OF CONTENTS

1. Objective

To develop a model which can predict the Compressive Strength of Concrete based on it's constituents and it's age.

2. Scope

- Saves the time taken by manual testing procedures to determine strength.
- To produce a concrete of required strength with the optimum amount of it's constituents.
- To produce economical concrete by adjusting the quantity of it's components.

.

3. Data Sharing Agreement

- **Sample File Name :** Concrete_Data.xls
- **Number of Columns :-** There are 9 number of columns.
- **Column Names :-** Columns Names are :
    - Cement (component 1)(kg in a m^3 mixture
    - Blast Furnace Slag (component 2)(kg in a m^3 mixture)
    - Fly Ash (component 3)(kg in a m^3 mixture)
    - Water (component 4)(kg in a m^3 mixture)
    - Superplasticizer (component 5)(kg in a m^3 mixture)
    - Coarse Aggregate (component 6)(kg in a m^3 mixture)
    - Fine Aggregate (component 7)(kg in a m^3 mixture)
    - Age (day)
    - Concrete compressive strength(MPa, megapascals)
- **Column Data Type** : All the columns except the age column have float type values and the age column have integer type values.

4 Dataset Information

Number of instances (observations): 1030

Number of Attributes: 9

Attribute breakdown: 8 quantitative input variables, and

1 quantitative output variable

Missing Attribute Values: None

There are following values of columns present inside dataset:

Cement : Quantity of in Cement kg per cubic meter of concrete.

Blast Furnace Slag : Quantity of Slag in kg per cubic meter of concrete.

Fly Ash : Quantity of Fly Ash in kg per cubic meter of concrete.

Water : Quantity of Water in kg per cubic meter of concrete.

Superplasticizer : Quantity of Superplasticizer in kg per cubic meter of concrete.

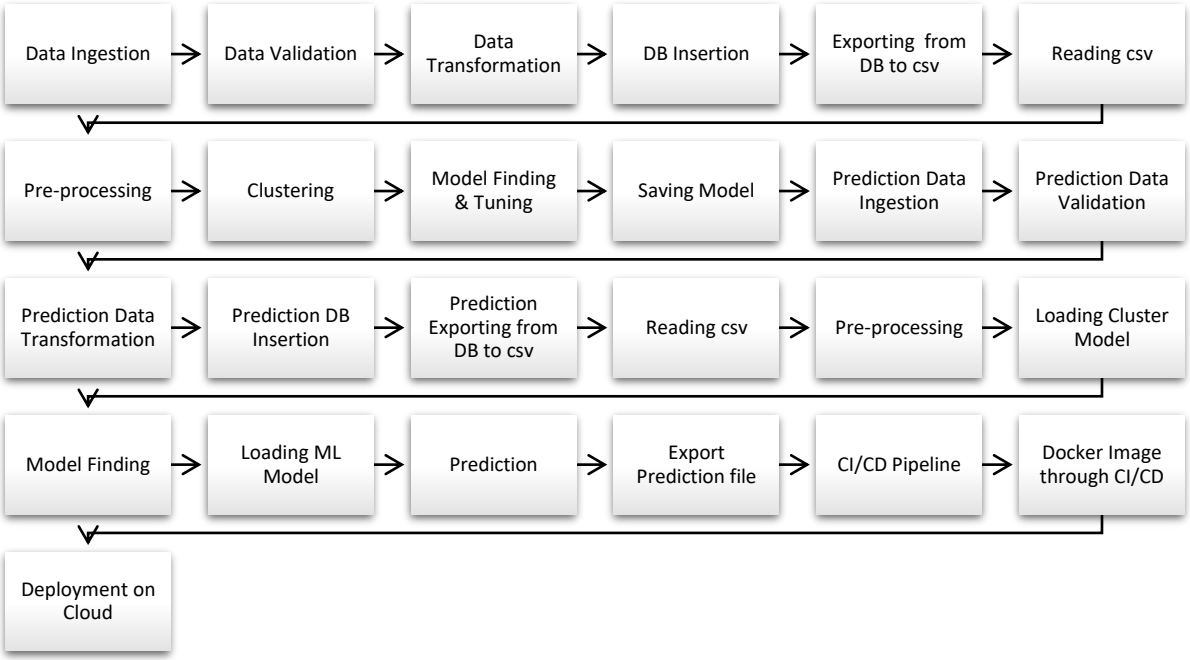
Coarse Aggregate : Quantity of C. Aggregate in kg per cubic meter of concrete.

Fine Aggregate : Quantity of F. Aggregate in kg per cubic meter of concrete.

Age : Duration at which concrete strength is tested.

Concrete Compressive Strength : Compressive strength in Megapascal which is achieved based on the above mentioned components and the duration at which it is tested.

5 Architecture



5.1 Architecture Description

- **Data Ingestion :** Dataset is taken from Kaggle link provided by I-Neuron
- **Data Validation :** Validation checks have been made like Name and Number of Columns, File name, Missing values in columns, etc.
- **Data Transformation :** Columns have been renamed to shorter name for easy access. No other necessary transformation required in the dataset.
- **DB Insertion :** After validation of the data and doing necessary transformations, the data is pushed to Cassandra Database. This is the last step in Training Raw Data Validation process.
- **Exporting from DB to csv :** Validated data that was pushed to Cassandran DB is retrieved into csv file. This csv file is used to train our model after necessary steps.
- **Reading csv :** The csv file that we received is now accessed using pandas
- **Data Pre-processing :** Dataset is checked for missing values, duplicate values, skewed data, outliers, scales, distributions, etc . We have found only duplicate rows are present which will not contribute anything for prediction, so have removed those rows. There are no other pre-processing steps required in this dataset.

- **Clustering :** Clustering of model is done to make the clusters of data having similarity between them.
- **Model Finding :** Model is trained based on each of the cluster dataset using appropriate ML algorithms and out of these models the best one is selected based on the R2 score for each of the clusters.
- **Model Tuning :** After finding out the best model for each of the clusters, we do hyperparameters tuning of those models to get better score. And then based on the best parameters, the model is trained.
- **Saving Model :** After training we save our model which can be used for prediction of unknown data.
- **Prediction Steps :** When we receive the new data from client for which we have to do predictions, we do all the Data Validation, Data Ingestion, Data Transformation, DB Operations, and Pre-processing steps, and then we will load our cluster model that he have created during Clustering step in Training and then using that model, it will make the clusters of prediction dataset and then based on that cluster number we use the cluster specific trained model to do the predictions of cluster and then we append those results into a single file which we have to save for Client.
- **CI/CD Pipeline :** Circle CI is used for the purpose of CI/CD operations which is used to create docker image for DockerHub and to deploy on cloud automatically just by committing our code to GitHub Repository.
- **Deployment :** The project is deployed to AWS and Heroku cloud platforms.

6. Data Validation and Transformation

- **Name Validation :** Validation of files name as per the DSA. We have created a regex pattern for validation. After it checks for date format and time format if these requirements are satisfied, we move such files to "Good_Data_Folder" else "Bad_Data_Folder.“
- **Number of Columns** : Validation of number of columns present in the files, and if it doesn't match then the file is moved to "Bad_Data_Folder.“
- Data type of columns - The data type of columns is given in the schema file. It is validated when we insert the files into Database. If the datatype is wrong, then the file is moved to "Bad_Data_Folder".
- **Null values in columns :** If any of the columns in a file have all the values as NULL or missing, we discard such a file and move it to "Bad_Data_Folder".

7. DB Operations

- **DB Connection:** The cloud version of Cassandra on astra.datastax.com is used as database. The configuration files are downloaded from website and then provided inside cloud configuration and authentication id and key are provided to connect to the database.
- **Table creation :** Table name *Good_Raw_Data* is created in the database in the *training* keyspace for inserting the files. It checks if the table is already present inside the keyspace, if present, then it deletes the existing table and creates the new one.
- **Insertion of files in the table :** All the files in the "Good_Data_Folder" are inserted in the above-created table. If any file has invalid data type in any of the columns, the file is not loaded in the table .
- **Exporting from DB to csv file:** In this step, the data from Cassandra is retrieved and exported to a csv file in *Training_FileFromDb* folder.

8. Model Training

- **Reading csv file :** File that was received from Cassandra in *Training_FileFromDb* folder is loaded using pandas.
- **Data Preprocessing :**
    - Performing EDA to get insight of data like identifying distribution , missing values, outliers , scaling , categorical data encoding, etc.
    - Removing duplicate rows that were present in dataset.
    - There were no null values present so we didn't have to take care of that.
    - As the distribution is not normal, hence we tried RobutScaler to scale the values of dataset.
- **Clustering :**
    - KMeans algorithm is used to create clusters in the preprocessed data. The optimum number of clusters is selected by plotting the elbow plot, and for the dynamic selection of the number of clusters, we are using KneeLocator function. The idea behind clustering is to implement different algorithms on the structured data
    - The Kmeans model is trained over preprocessed data, and the model is saved for further use in prediction
- **Model Selection :**
    - After the clusters are created , we find the best model for each cluster. By using 4 Algorithms "RandomForest Regressor", ""SVM Regressor" , "XGBoost", and "BaggingRegressor with Decision Tree". After that we did the hyperparameter tuning using Optuna and find the R2 score and then selects the model with best R2 score. Likewise the model is created for each cluster and saved in *models* directory which will later be used for prediction.

9. Prediction

- The testing files are shared in the batches and we perform the same Validation operations ,data transformation and data insertion on them.
- The accumulated data from db is exported in csv format for prediction
- We perform data pre-processing techniques on it.
- KMeans model created during training is loaded and clusters for the preprocessed data is predicted
- Based on the cluster number respective model is loaded and is used to predict the data for that cluster.
- Once the prediction is done for all the clusters. The predictions are saved in csv format and shared.

10. Development Strategy and Production

- We have used Flask to create API. Flask API has 4 routes
    - '/' for homepage
    - '/train' to start training
    - '/predict' to start prediction
    - '/download' to download file
- We have created our code in modular fashion so that anyone can use it.
- We have created CI/CD pipeline using CircleCI and created Docker image using CircleCI and deploying on AWS and Heroku cloud.