Predicting Compressive Strength Of Concrete

S.Raagavarshini

39110818

1. INTRODUCTION

1.1 OVERIEW

The Compressive Strength of Concrete determines the quality of Concrete. This is generally determined by a standard crushing test on a concrete cylinder. This requires engineers to build small concrete cylinders with different combinations of raw materials and test these cylinders for strength variations with a change in each raw material. The recommended wait time for testing the cylinder is 28 days to ensure correct results. This consumes a lot of time and requires a lot of labour to prepare different prototypes and test them. Also, this method is prone to human error and one small mistake can cause the wait time to drastically increase. One way of reducing the wait time and reducing the number of combinations to try is to make use of digital simulations, where we can provide information to the computer about what we know and the computer tries different combinations to predict the compressive strength. This way we can reduce the number of combinations we can try physically and reduce the amount of time for experimentation. But, to design such software we have to know the relations between all the raw materials and how one material affects the strength. It is possible to derive mathematical equations and run simulations based on these equations, but we cannot expect the relations to be same in real-world. Also, these tests have been performed for many numbers of times now and we have enough real-world data that can be used for predictive modelling.

1.2 PURPOSE

Determining accurate concrete strength is a major civil engineering problem. The

traditional way to know the strength of concrete will take about 28 days which is very time-consuming. It is important to wait 28 days to ensure the quality control of the process, although it is very time-consuming. This project aims at building a Machine Learning model to predict the compressive strength of concrete to greatly speed up prediction with higher accuracy. A web application is built where the user can enter the required parameters and see the predicted results on Web Application.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

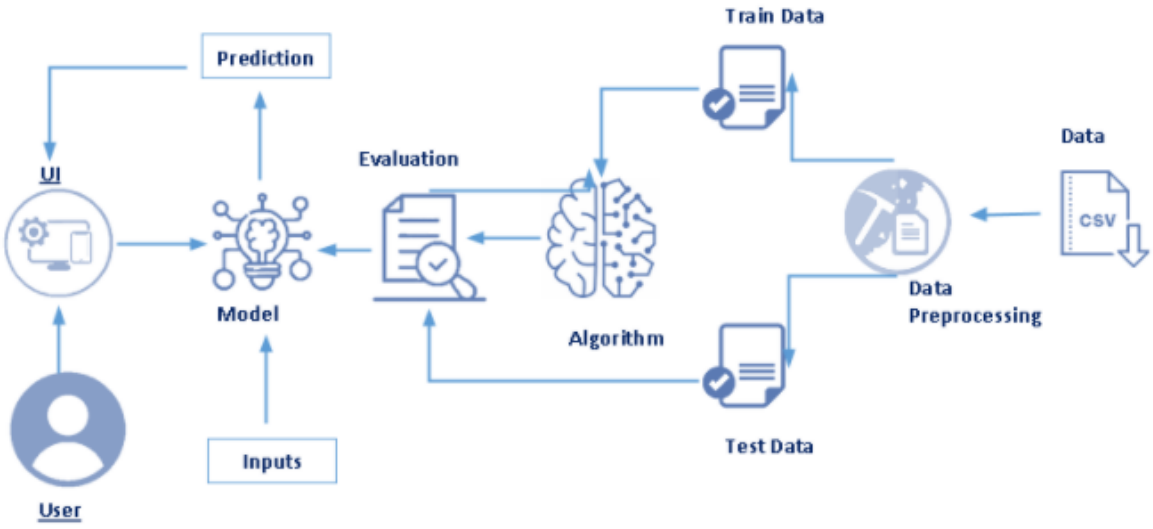
Several approaches using regression functions have been proposed for predicting the concrete strength. Data driven techniques like Linear Regression analysis, MLP (Multilayer Perceptron) and M5P modal trees were used. Linear Regression, Lasso Regression, Ridge Regression, Decision Tree Regressor, Random Forest Regressor were implemented. Also, comparative study between the different techniques has been carried out in different environments like strength of concrete mix.

2.2 PROPOSED SOLUTION

Gradient Boosting regression is used in this study. GB builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage a regression tree is fit on the negative gradient of the given loss function.

3. THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM



3.2 HARDWARE / SOFTWARE REQUIREMENTS

Anaconda Navigator :

Anaconda Navigator is a free and open-source distribution of the Python and R programming languages for data science and machine learning related applications. It can be installed on Windows, Linux, and macOS.Conda is an open-source, cross-platform, package management system. Anaconda comes with great tools like Jupyter Notebook, Spyder, Glueviz, Orange, Rstudio, Visual Studio Code.

Packages required :

Sklearn: Scikit-learn is a library in Python that provides many unsupervised and supervised learning algorithms.

NumPy: NumPy is a Python package that stands for 'Numerical Python'. It is the core library for scientific computing, which contains a powerful n-dimensional array object

Pandas: pandas is a fast, powerful, flexible, and easy to use open-source data analysis and manipulation tool, built on top of the Python programming language.

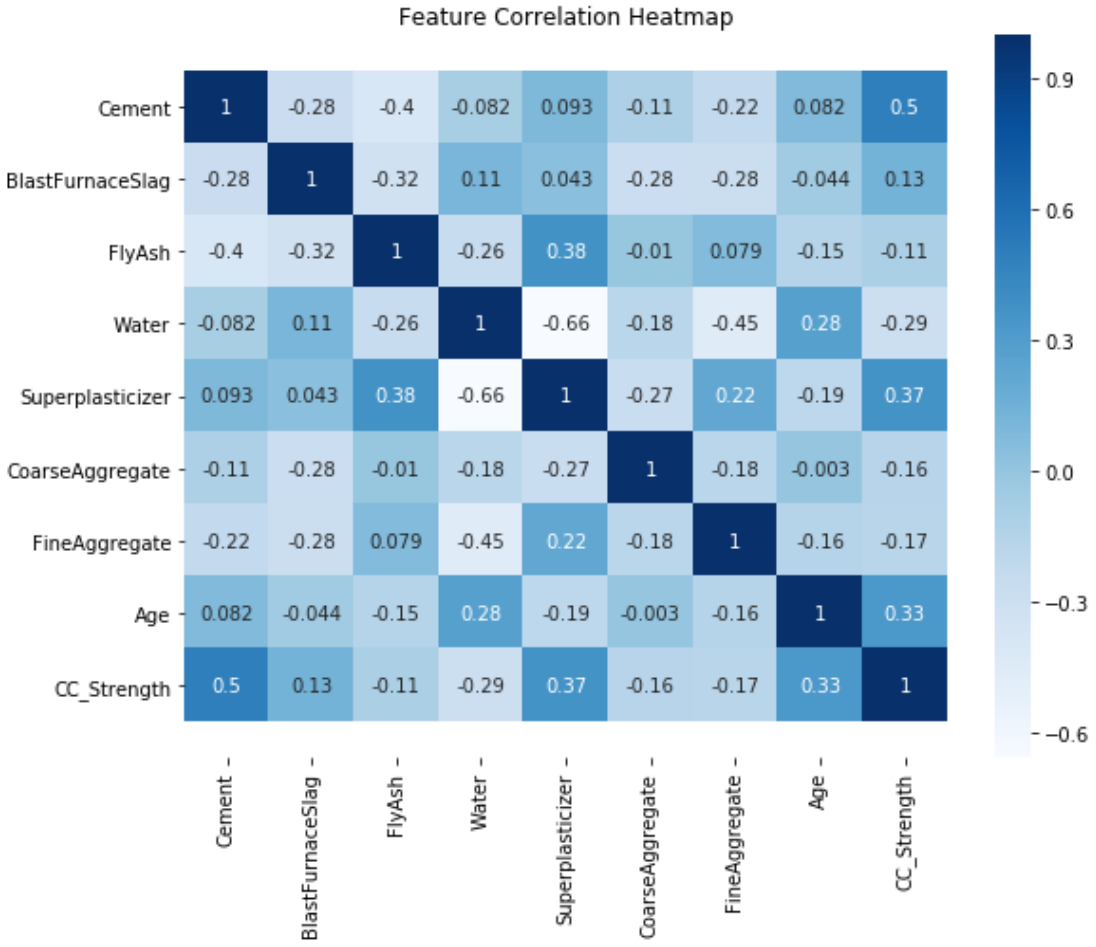
Matplotlib: It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits

Flask: Web framework used for building Web applications.

4. EXPERIMENTAL INVESTIGATIONS

4.1 Compressive strength of concrete

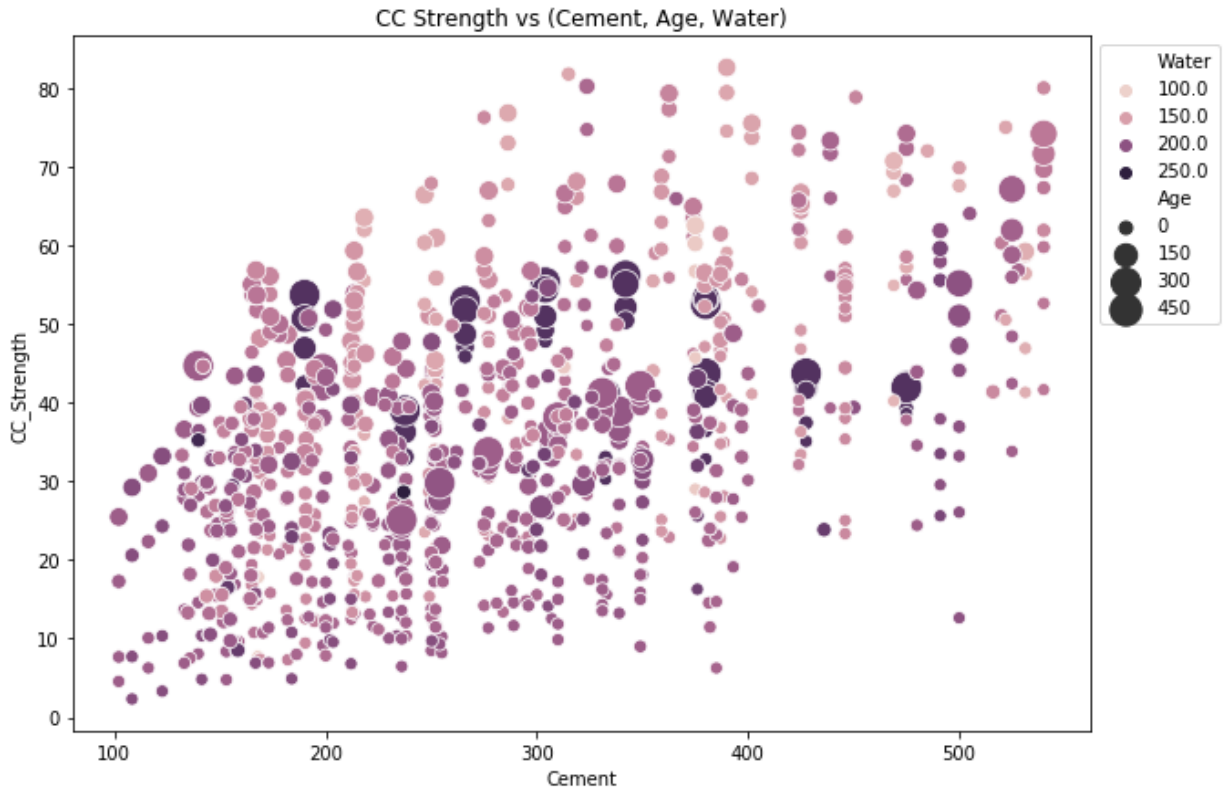
The strength of concrete is controlled by the proportioning of cement, coarse and fine aggregates, water, and various admixtures. The ratio of the water to cement is the chief factor for determining concrete strength. The lower the water-cement ratio, the higher is the compressive strength. A certain minimum amount of water is necessary for the proper chemical action in the hardening of concrete; extra water increases the workability (how easily the concrete will flow) but reduces strength. A measure of the workability is obtained by a slump test. Actual strength of concrete in place in the structure is also greatly affected by quality control procedures for placement and inspection.



The observations from this plot,

- a high positive correlation between compressive Strength (CC_Strength) and Cement.

- A strong negative correlation between Super Plasticizer and Water.
- positive correlations between Super Plasticizer and Fly Ash, Fine Aggregate



The observations from this plot,

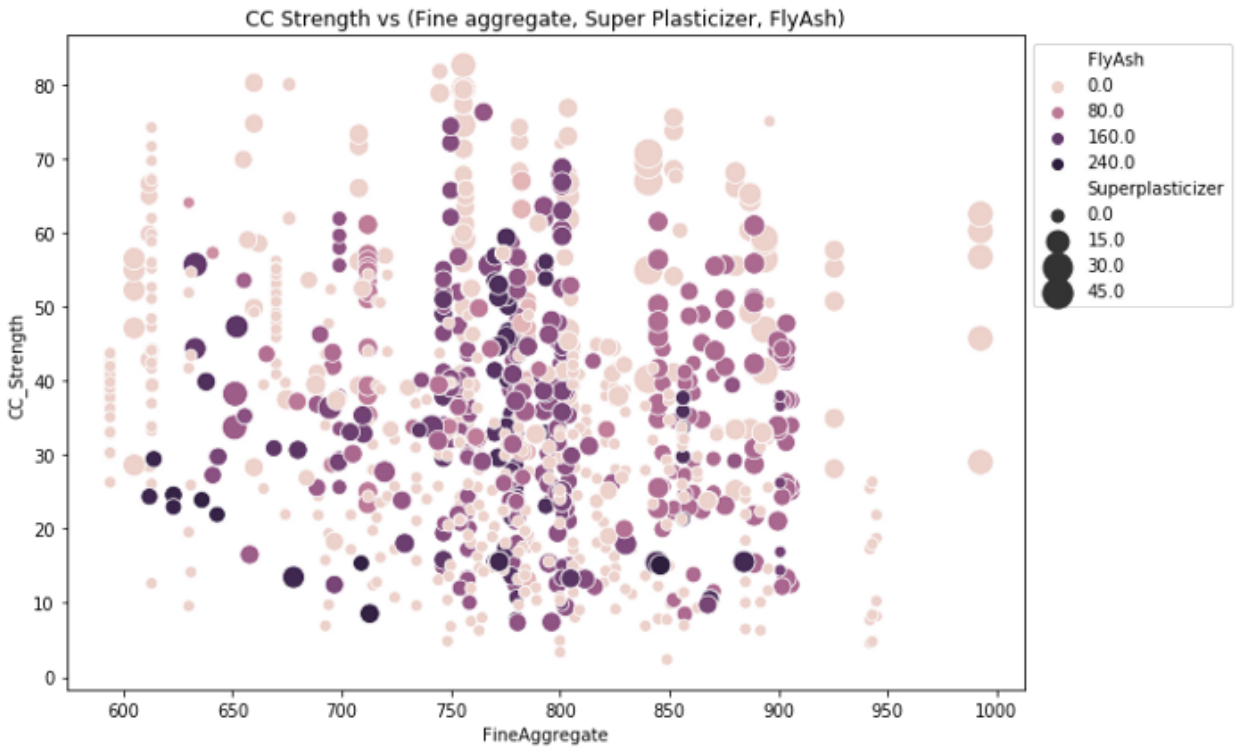
- Compressive strength increases as the amount of cement increases, as the dots move up when we move towards right on the x-axis.

- Compressive strength increases with age (as the size of dots represents the age), this not the case always but can be up to an extent.

- Cement with less age requires more cement for higher strength, as the smaller dots are moving up when we move towards right on the x-axis.

- The older the cement is the more water it requires, can be confirmed by observing the colour of the dots. Larger dots with dark colour indicate high age and more water.

- Concrete strength increases when less water is used in preparing it since the dots on the lower side (y-axis) are darker and the dots on higher-end (y-axis) are brighter.



The observations from this plot,

- Compressive strength decreases Fly ash increases, as darker dots are

concentrated in the region representing low compressive strength.

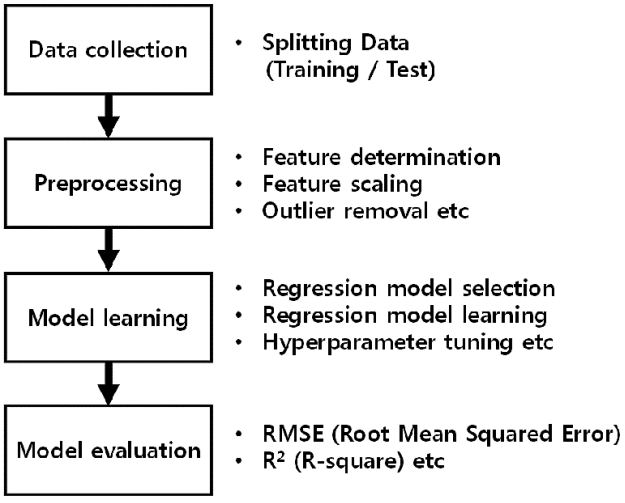
- Compressive strength increases with Superplasticizer since larger the dot the higher they are in the plot.

4.2 Gradient Boosting for regression.

GB builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage a regression tree is fit on the negative gradient of the given loss function.

Gradient Boosting algorithm represents creation of forest of fixed number of decision trees which are called as weak learners or weak predictive models. These decision trees are of fixed size or depth. The gradient boosting starts with mean of target values and add the prediction / outcome / contribution from subsequent trees by shrinking it with what is called as learning rate. The decision trees or estimators are trained to predict the negative gradient of the data samples. Gradient Boosting Regression algorithm is used to fit the model which predicts the continuous value.

5. FLOWCHART



Initially a dataset which contains a set of features through which approval status can be

identified is read. This project uses "concrete.csv". The second step is to process the data which includes handling null values and categorical values, normalize the data, identify dependent and independent variables and split the dataset into train and test sets. Then comes model building where a model is choosen depending on the data and the output and it is trained. This dataset is a regression kind of dataset, Gradient Boosting algorithm is used. Once the model is trained, it's ready to make predictions. The predict method is used on the model and pass x_test as a parameter to get the output as y_pred. The last step is model evaluation. Here the accuracy of the model is checked. Then the model is saved as pickle file and a flask application is created.

6. RESULT

Analysed the Compressive Strength Data and used Machine Learning to Predict the Compressive Strength of Concrete. The system produces results with 88.5% accuracy. The Mean absolute error is 3.8, Mean square error is 29.5 and Root mean square error is 5.4. The overall error rate can be considered low and the techniques can adequately be used to predict the concrete compressive strength, staying within the acceptable safety range for engineering practices.Thus, the present study suggests an alternative approach to evaluate compressive strength against traditional method.

```
#MODEL EVALUATION
score=gbr.score(x_test,y_test)
print("Score of Gradient Boosting model : ",score)
```

```
Score of Gradient Boosting model :  0.885770424482433
```

```
from sklearn import metrics #importing the metrics library
print("MAE : ",metrics.mean_absolute_error(y_test,y_pred)) #Mean Absolute error
print("MSE : ",metrics.mean_squared_error(y_test,y_pred)) #Mean squared error
print("RMSE :",np.sqrt(metrics.mean_squared_error(y_test,y_pred))) #Root mean Square error
```

```
MAE :  3.8350605358376098
MSE :  29.7790311873429
RMSE : 5.457016692969053
```

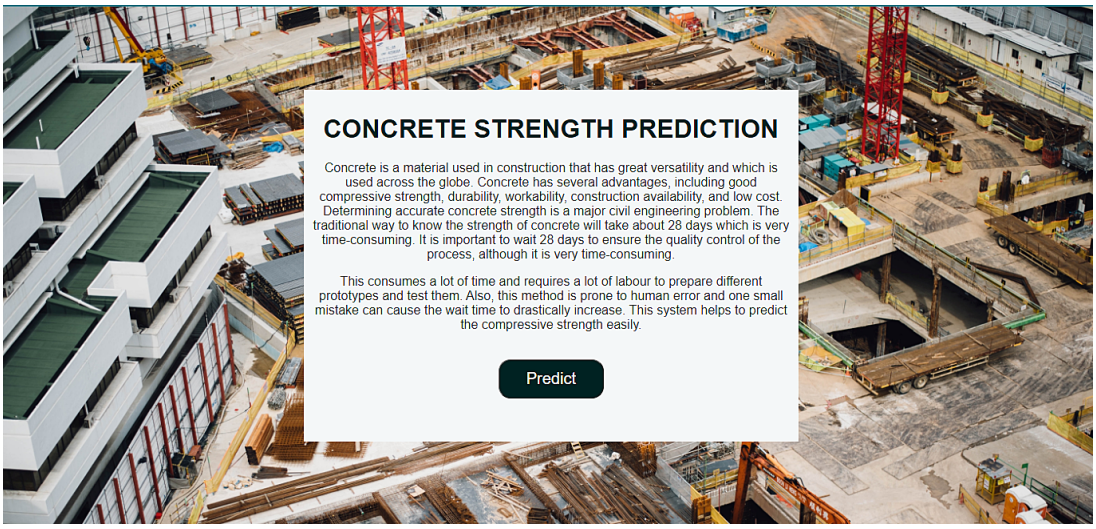
Mean_squared_error: MSE or Mean Squared Error is one of the most preferred metrics for regression problems. It is simply the average of the squared difference between the target value and the value predicted by the regression model.

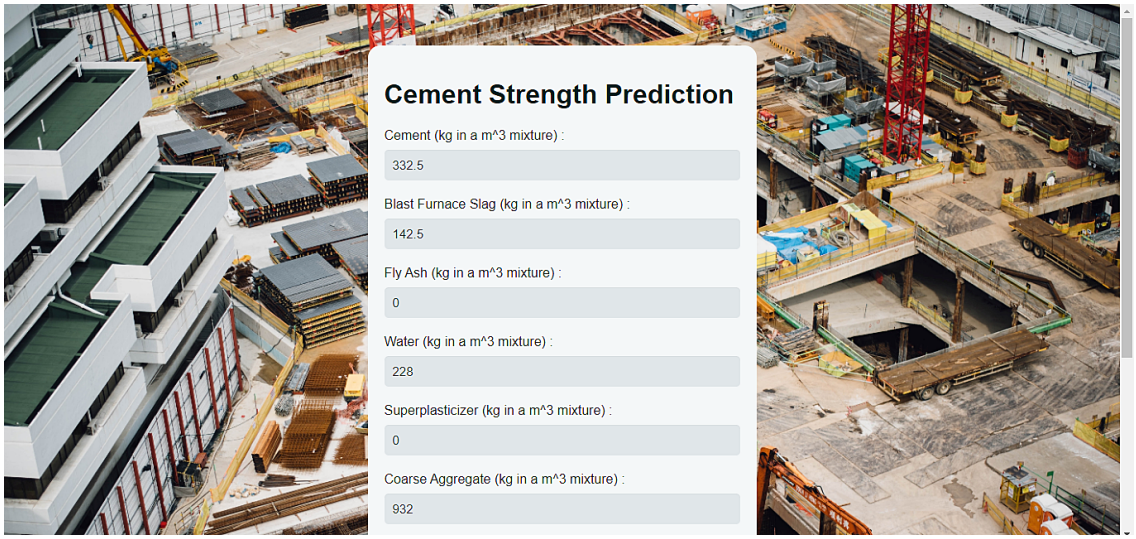
RMSE: RMSE(Root Mean Square Error) is the square root of the averaged squared

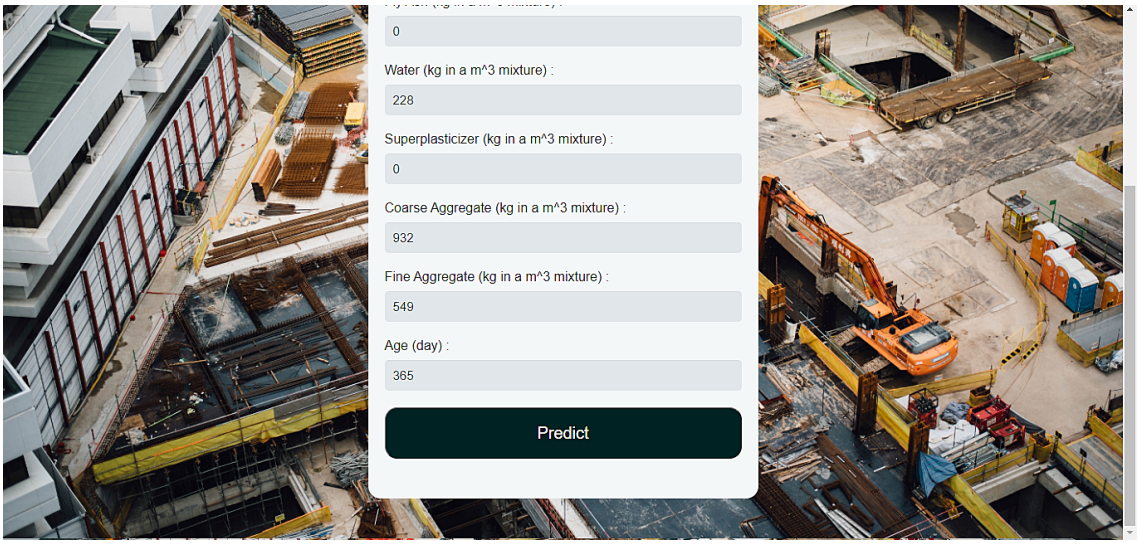
difference between the target value and the value predicted by the model.

R2 Score: Coefficient of Determination or R^2 is another metric used for evaluating the performance of a regression model. The metric helps us to compare our current model with a constant baseline and tells us how much our model is better.

6.1 OUTPUT







7.ADVANTAGES AND DISADVANTAGES

7.1.ADVANTAGES

- The traditional way to know the strength of concrete will take about 28 days which is very time-consuming. But this system helps to predict the concrete strength at great speed and high accuracy.
- This system uses Gradient boosting regresssor which provies predictive accuracy that cannot be trumped.

7.2. DISADVANTAGES

- The system provides only 88.5% accuracy.

8.APPLICATIONS

The strength of the concrete indicates the ability of the structure to withstand. In the world of construction, every constructor wants the construction to be durable life long. The compressive strength of concrete is the most common performance measurement used by engineers when designing buildings and other structures. Compressive strength is a key value for designing structures. It gives an idea about the characteristics of the concrete.

9.CONCLUSION

The work presented in this document comprises the development of regression model for predicting the compressive strength of concrete. The regression models thus developed can reliably predict the compressive strength of various mixes more efficiently. . The overall error rate can be considered low and the techniques can adequately be used to predict the concrete compressive strength, staying within the acceptable safety range for engineering practices.Thus, the present study suggests an alternative approach to evaluate compressive strength against traditional method.

10.FUTURE WORK

There are a number of potential avenues for further work. In the future, the work can be extended by applying different machine learning techniques. An in-depth evaluation of why different algorithms exhibit different regression accuracy and computational performance can be performed. Investigating the robustness of ML classification and a comparison between ML and non-ML techniques on an identical dataset would also be valuable.

11.BIBILIOGRAPHY

[1] Concrete Compressive Strength Prediction using Machine Learning from Towards Data Science

[2] Concrete Strength Prediction Using Machine Learning (with Python code) from Analyticsvidhya

[3] Kaloop, M. R., Kumar, D., Samui, P., Hu, J. W., & Kim, D. (2020). Compressive strength prediction of high-performance concrete using gradient tree boosting machine. *Construction and Building Materials*, *264*, 120198.

[4] Nguyen-Sy, T., Wakim, J., To, Q. D., Vu, M. N., Nguyen, T. D., & Nguyen, T. T. (2020). Predicting the compressive strength of concrete from its compositions and age using the extreme gradient boosting method. *Construction and Building Materials*, *260*, 119757.

[5] Web Development Tutorials from W3resource

APPENDIX

A.SOURCE CODE :

App.py :

```
import numpy as np

from flask import Flask, request, render_template

import pickle

import pandas as pd


app = Flask(__name__) # initializing a flask app

model =pickle.load(open('cement.pkl', 'rb'))


@app.route('/')# route to display the home page
```

```
def home():
    return render_template('home.html') #rendering the home page
@app.route('/Prediction',methods=['POST','GET'])
def prediction():
    return render_template('index1.html')
@app.route('/Home',methods=['POST','GET'])
def my_home():
    return render_template('home.html')
@app.route('/predict',methods=['POST']) # route to show the predictions in a web UI


def index():
            #  reading the inputs given by the user
    input_features = [float(x) for x in request.form.values()]
    features_value = [np.array(input_features)]
    features_name = ['Cement', 'Blast Furnace Slag', 'Fly Ash', 'Water',
                     'Superplasticizer','Coarse Aggregate', 'Fine Aggregate','Age']
    x = pd.DataFrame(features_value, columns=features_name)
    x_log=np.log(x)#performing log transformation
    # predictions using the loaded model file
    prediction=model.predict(x)
    print('prediction is', prediction)
```

```
    # showing the prediction results in a UI
    return render_template('result2.html',prediction_text=prediction[0])
if __name__ == "__main__":
    app.run(host='127.0.0.1', port=8001, debug=True)
    #app.run(debug=False) # running the app
    #app.run('0.0.0.0',5000) #local host 8080
```

Project.ipynb :

```
#IMPORT THE LIBRARIES
import numpy as np # used for numerical analysis
import pandas as pd # used for data manipulation
from collections import Counter as c # return counts
import seaborn as sns # used for data visualization
import matplotlib.pyplot as plt # used for data visualization(plotting graph)
from sklearn.model_selection import train_test_split# splits data in random train and test array
from sklearn.metrics import accuracy_score,mean_squared_error,mean_absolute_error # model performance
import pickle # Pyhton object hierarchy is converted into a byte stream
from sklearn.linear_model import LinearRegression #Regreassion ML algorithm
#READING THE DATASET
data = pd.read_csv('concrete.csv') # loading the csv dataset
```

```
# UNDERSTANDING THE DATA

type(data)

data.tail() # returns last 5 rows of the data

data.head() # returns first 5 rows of the data

# rename columns which have only the name

data.columns = [col[:col.find("(")].strip() for col in data.columns]

data.info() # info will give you a summary of dataset

data.describe() # returns description of the data in the DataFrame

#HANDLING MISSING VALUES

data.isnull().any()

data.isnull().sum()

# DATA VISUALIZATION

data.boxplot(figsize=(18,7))

# SPLIT THE DATA INTO INDEPENDENT AND DEPENDENT VARIABLES

x=pd.DataFrame(data,columns=data.columns[:8]) #independent columns

y=pd.DataFrame(data,columns=data.columns[8:]) #dependent columns

#SPLIT THE DATASET INTO TRAIN SET AND TEST SET

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=1)

#TRAINING AND TESTING THE MODEL

from sklearn.ensemble import GradientBoostingRegressor

gbr=GradientBoostingRegressor() #object of GradientBoostingRegressor
```

```
gbr.fit(x_train,y_train) #training our data using fit method

y_pred=gbr.predict(x_test) #predicting using predict method

print("Prediction made by Gradient Boosting model : ",y_pred)

#MODEL EVALUATION

score=gbr.score(x_test,y_test)

print("Score of Gradient Boosting model : ",score)

from sklearn import metrics #importing the metrics library

print("MAE : ",metrics.mean_absolute_error(y_test,y_pred)) #Mean Absolute error

print("MSE : ",metrics.mean_squared_error(y_test,y_pred)) #Mean squared error

print("RMSE :",np.sqrt(metrics.mean_squared_error(y_test,y_pred)))

#SAVE THE MODEL

#dumping our model in pickle format

pickle.dump(gbr, open('cement.pkl','wb'))
```