APPLIED DATA SCIENCE - ML COURSE

PROJECT DOCUMENTATION

PREDICTING COMPRESSIVE STRENGTH OF CONCRETE

NAME : R.PRIYADHARSHINI

REG NO : 39110805

CONCRETE STRENGTH PREDICTION PROJECT

CONTENTS

1.INTRODUCTION

1.1 Overview :

Predicting Compressive Strength Of Concrete

Concrete is a material used in construction that has great versatility and which is

used across the globe. Concrete has several advantages, including good compressive strength, durability, workability, construction availability, and low cost. 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.

1.2 Purpose :

- This project aims at building a Machine Learning model to predict the compressive strength of concrete to greatly speed up prediction with higher accuracy.
- Able to understand the problem to classify if it is a regression or a classification kind of problem.
- Able to know how to pre-process/clean the data using different data pre-processing techniques.
- Applying different algorithms according to the dataset.
- Able to know how to find the accuracy of the model.
- Able to build web applications using the Flask framework.

2. LITERATURE SURVEY

2.1 Existing problem :

EXISTING METHOD :

Some of the methods for Testing Compressive Strength of Concrete are Rebound Hammer or Schmidt Hammer (ASTM C805),Penetration Resistance Test (ASTM C803),Ultrasonic Pulse Velocity (ASTM C597), Pullout Test (ASTM C900),Drilled Core (ASTM C42),Wireless Maturity Sensors (ASTM C1074),etc.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. Determining accurate concrete strength is a major civil engineering problem. The two major tests mainly considered as quality tests are the **compression tests and slump tests.**The traditional way to know the strength of concrete will take about 28 days which is very time-consuming.

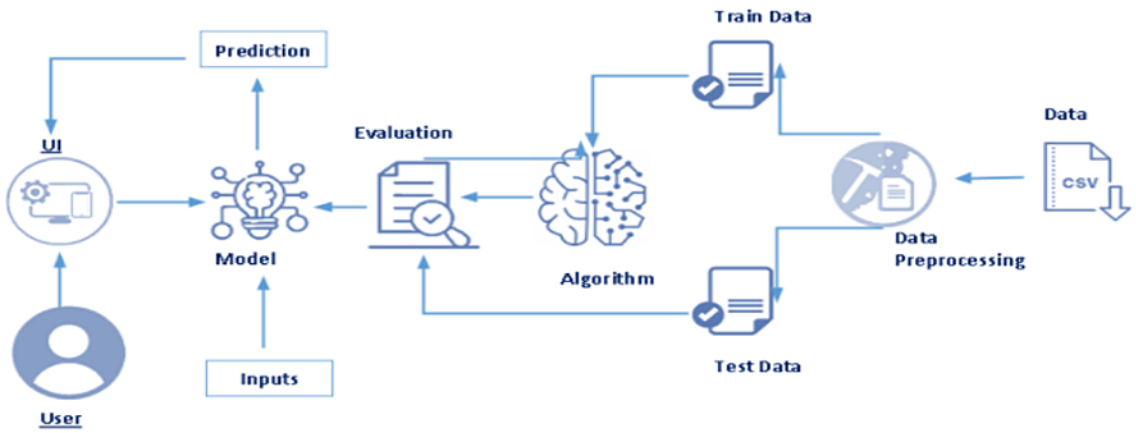
Method to solve it : One way of reducing the wait time and reducing the number of combinations to try is to make use of digital simulations, where we can provide information to the computer about what we know and the computer tries different combinations to predict the compressive strength.

2.2 Proposed solution:

We can reduce the number of combinations we can try physically and reduce the amount of time for experimentation with the use of digital simulations. Here, we apply Machine learning (Applied data science) concept to make concrete strength predictions better. We analyse the given Concrete Compressive Strength dataset and build Machine Learning models to predict the compressive strength.

3. THEORITICAL ANALYSIS

3.1 Block diagram:



3.2 Hardware and software requirements:

HARDWARE REQUIREMENTS:

CLIENT-SIDE	• Processor • Disk Space • RAM

SOFTWARE REQUIREMENTS:

In order to develop this project we need to install the following software/packages:

Step-1: 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 JupyterLab, Jupyter Notebook, QtConsole, Spyder, Glueviz, Orange, Rstudio, Visual StudioCode. For this project, we will be using **Jupyter** notebook and **Spyder.**

Step-2: To build Machine learning models you must require the following packages:

* **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.
* **If you are using anaconda navigator, follow the below steps to download the required packages:**

1. Open anaconda prompt.
2. Type “pip install numpy” and click enter.
3. Type “pip install pandas” and click enter.
4. Type “pip install matplotlib” and click enter.
5. Type “pip install scikit-learn” and click enter.
6. Type “pip install Flask” and click enter.

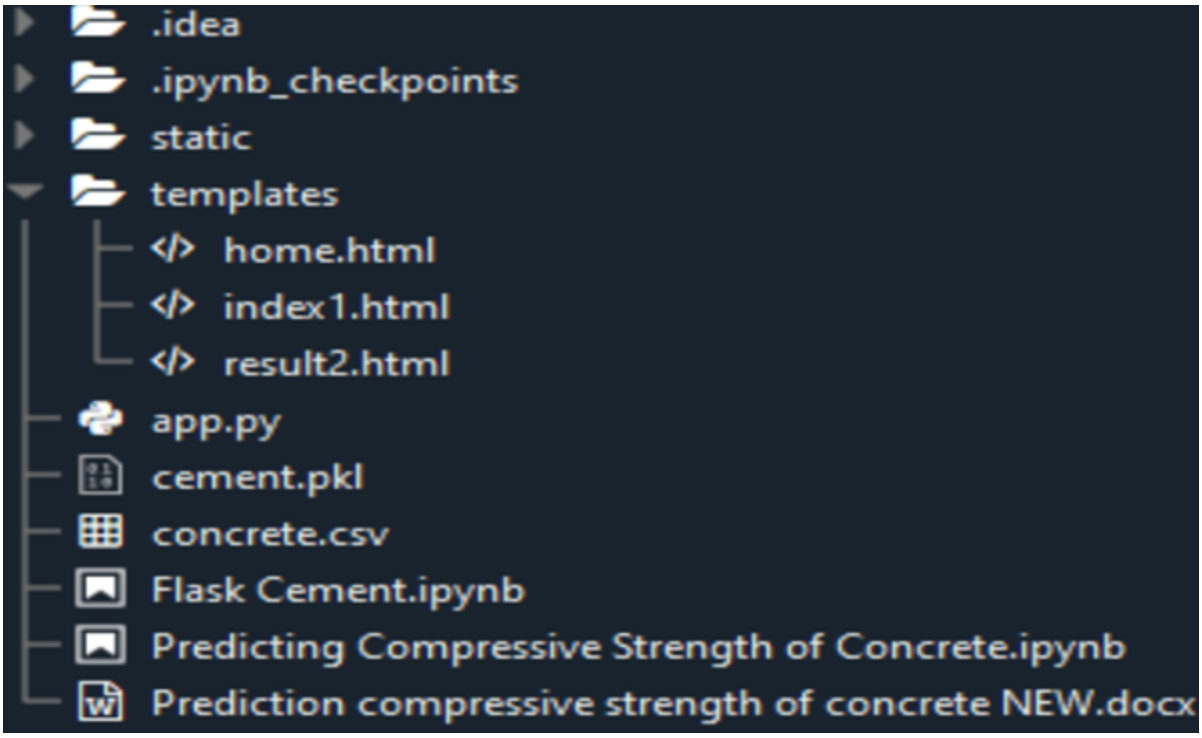
Now we have downloaded and set up all the necessary software and packages to start building the project.

4. EXPERIMENTALINVESTIGATIONS

4.1 ANALYSIS OR THE INVESTIGATION MADE WHILE WORKING ON THE SOLUTION:

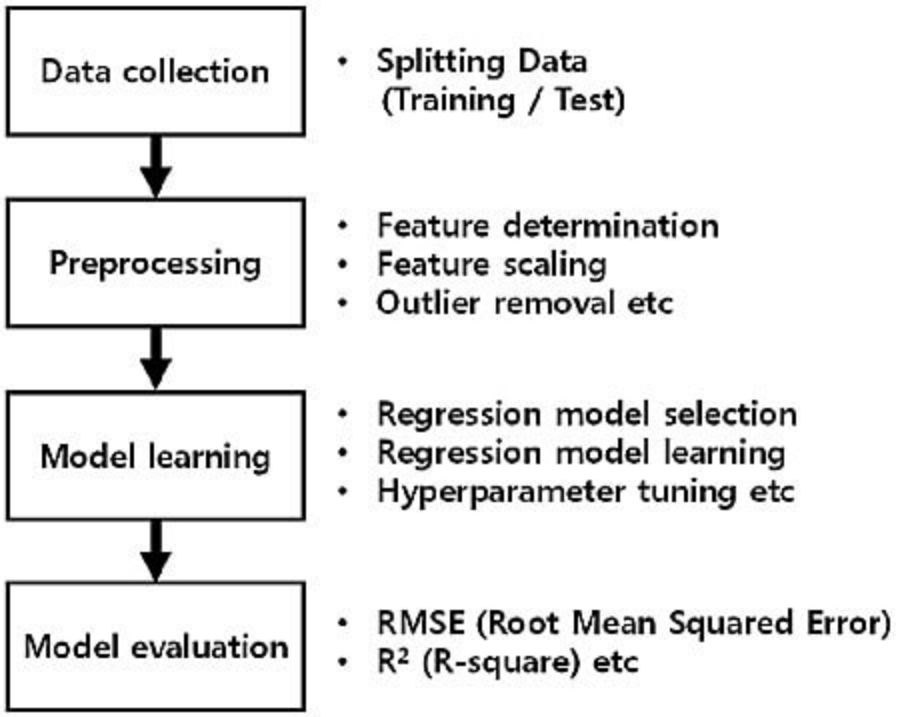
The first step in a Data Science project is to understand the data and gain insights from the data before doing any modelling. This includes checking for any missing values, plotting the features with respect to the target variable, observing the distributions of all the features and so on. Firstly we need to import the data and start analysing. Check the correlations between the input features, this will give an idea about how each variable is affecting all other variables. This can be done by calculating Pearson correlations between the features.

Project Folder structures :



5. FLOWCHART

5.1 DIAGRAM SHOWING THE CONTROL FLOW OF THE SOLUTION :



5.2 PROCESS FLOW:

Process to be followed while developing the project:

- User interacts with the UI (User Interface) to enter the input features
- Entered input features is analyzed by the model which is integrated
- Once model analyses the input, the prediction is showcased on the UI.

PROCESS FLOW:

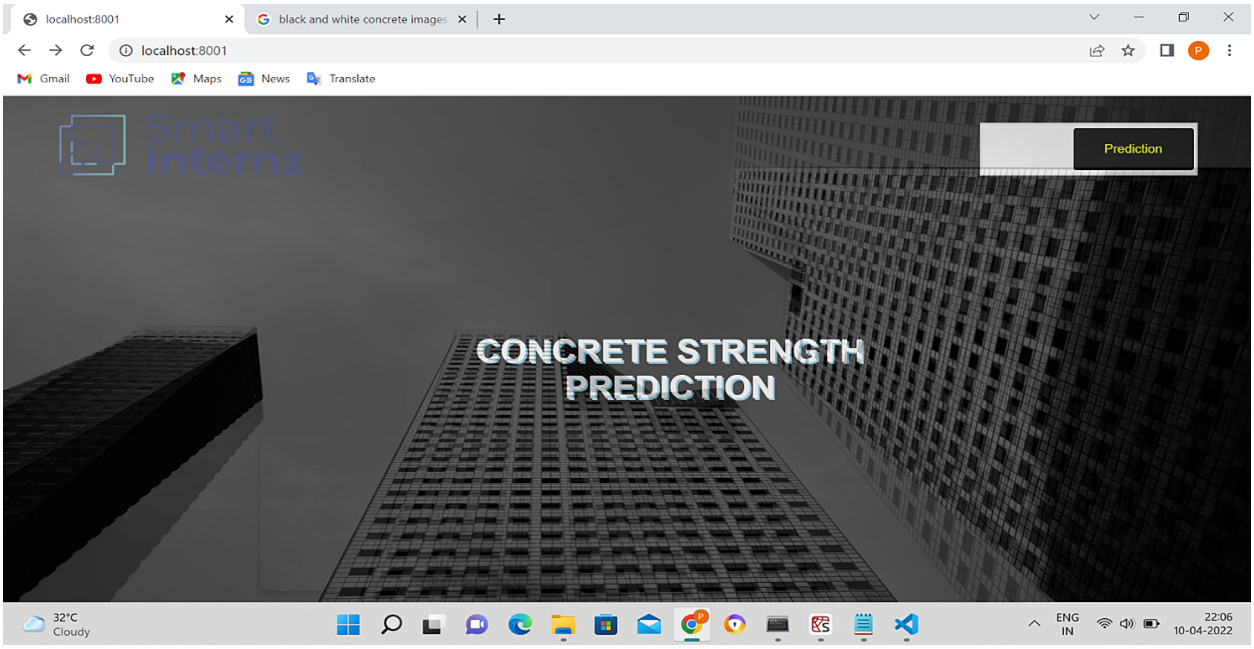
- Data Collection.
    - Collect the dataset or Create the dataset
- Data Pre- processing.
    - Import the Libraries.
    - Reading the dataset.
    - Processing of the data
    - Taking care of Missing Data.
    - Label Encoding
    - Data Visualization.
    - Splitting the Dataset into Dependent and Independent variables.
    - Splitting Data into Train and Test.
- Model Building
    - Training and testing the model
    - Evaluation of Model
- Application Building
    - Create an HTML file

- Build a Python Code

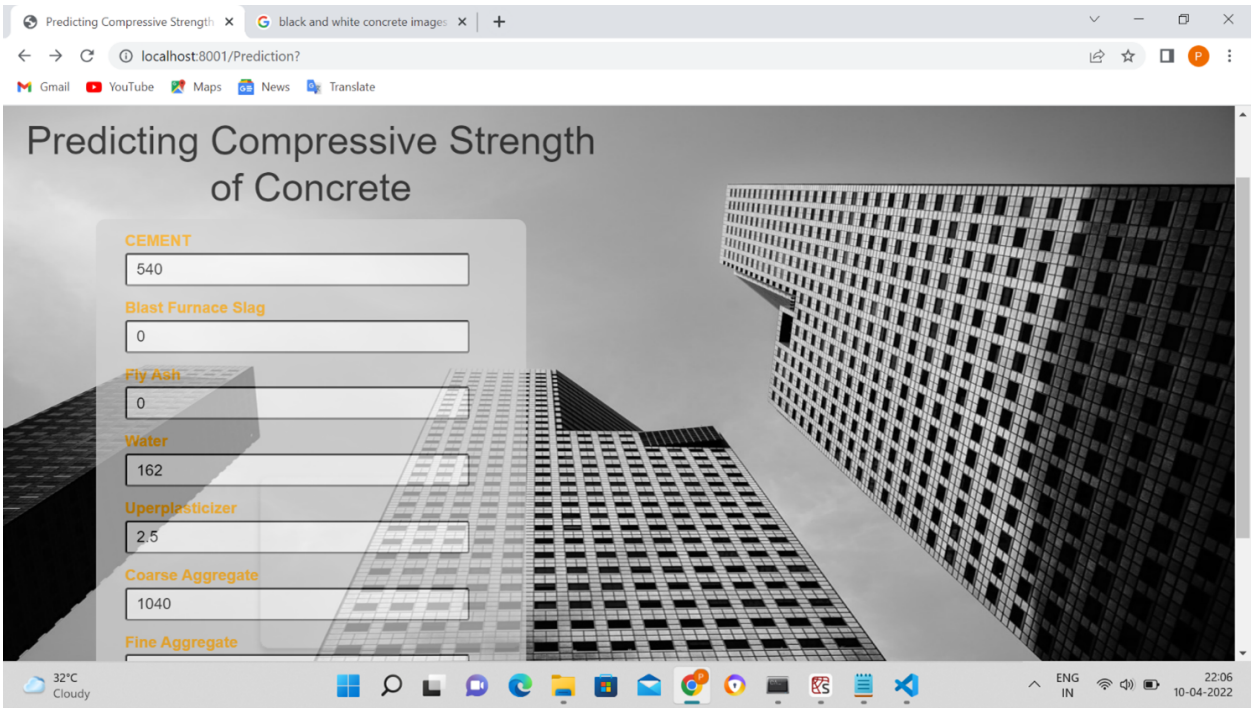
6. RESULT :

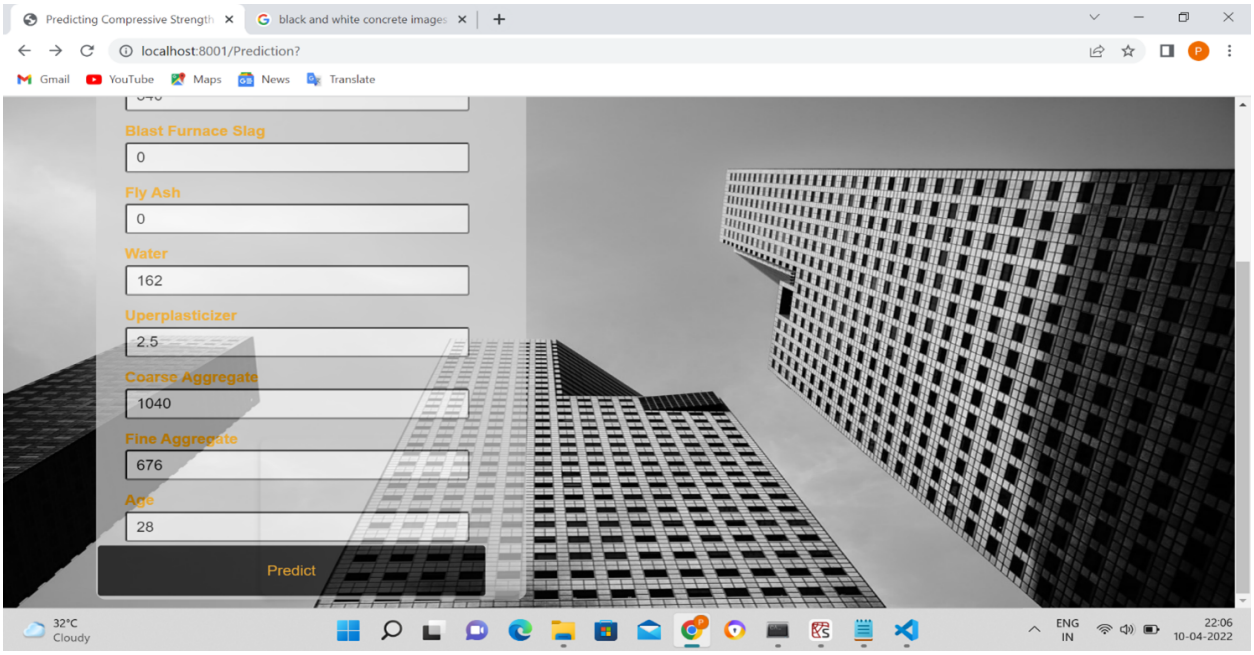
6.1 FINAL FINDINGS (OUTPUT) OF THE PROJECT ALONG WITH SCREENSHOTS:

HOME PAGE:

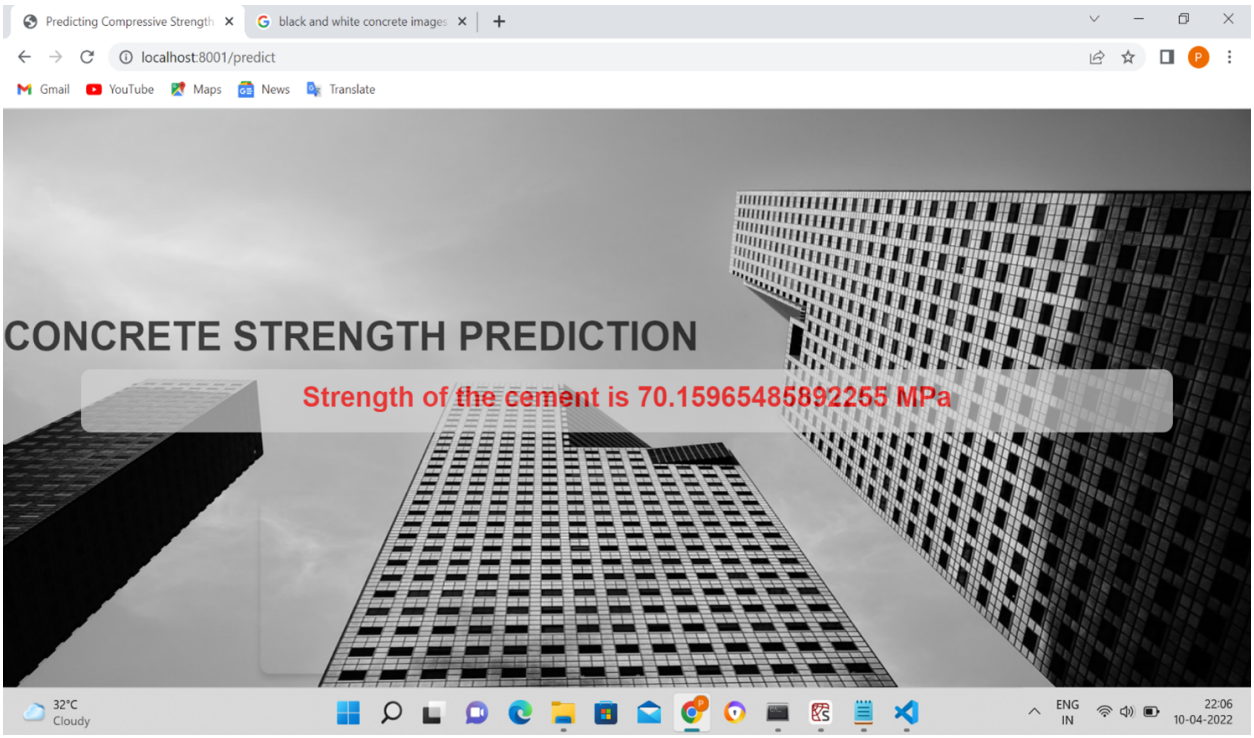


INDEX PAGE:


Predicting Compressive Strength
Predicting Compressive Strength of Concrete
localhost:8001/Prediction?
Gmail
YouTube
Maps
News
Translate
CEMENT
540
Blast Furnace Slag
0
Fly Ash
0
Water
162
Uperplasticizer
2.5
Coarse Aggregate
1040
Fine Aggregate
32°C
Cloudy
ENG
IN
22:06
10-04-2022


Predicting Compressive Strength
localhost:8001/Prediction?
Gmail
YouTube
Maps
News
Translate
Blast Furnace Slag
0
Fly Ash
0
Water
162
Uperplasticizer
2.5
Coarse Aggregate
1040
Fine Aggregate
676
Age
28
Predict
32°C
Cloudy
ENG
IN
22:06
10-04-2022

RESULT PAGE:



7. ADVANTAGES & DISADVANTAGES:

7.1 LIST OF ADVANTAGES AND DISADVANTAGES OF THE PROPOSED SOLUTION :

ADVANTAGES:

- This project aims at building a Machine Learning model to predict the compressive strength of concrete to greatly speed up prediction with higher accuracy.
- On click of single button , the concrete strength can be predicted easily.
- Able to know how to pre-process/clean the data using different data pre-processing techniques.
- Applying different algorithms according to the dataset.
- Able to know how to find the accuracy of the model.

- Able to build web applications using the Flask framework.

DISADVANTAGES:

- Accuracy may be appropriate, and it cant be perfect.

8. APPLICATIONS

8.1 THE AREAS WHERE THIS SOLUTION CAN BE APPLIED:

* It can also be applied in the field of Health care, Biotechnology , Aeronautical ,Microbiology ,etc.

* As the determination of accurate concrete strength is a major civil engineering problem, with the help of this prediction machine learning project, we can able to predict the compressive strength of concrete effectively and quickly.

* It can also be used in Image Recognition, Speech Recognition, Product recommendations and Self-driving cars and etc.

* We can apply this concept of Machine Learning ,not only for this prediction, but also for many purposes like Flight delay prediction, visa approval prediction, Rainfall prediction, Flood prediction ,Traffic volume estimation and etc.

9. CONCLUSION

SUMMARIZING THE ENTIRE WORK AND FINDINGS:

- Data Collection.
    - Collect the dataset or Create the dataset
- Data Pre- processing.
    - Import the Libraries.
    - Reading the dataset.

    - Processing of the data
    - Taking care of Missing Data.
    - Label Encoding
    - Data Visualization.
    - Splitting the Dataset into Dependent and Independent variables.
    - Splitting Data into Train and Test.
- Model Building
    - Training and testing the model
    - Evaluation of Model
- Application Building
    - Create an HTML file
    - Build a Python Code

FINDINGS:

- On click of single button , the concrete strength can be predicted easily.
- Able to know how to pre-process/clean the data using different data pre-processing techniques.
- Applying different algorithms according to the dataset.
- Able to know how to find the accuracy of the model.
- Able to build web applications using the Flask framework.

10. FUTURE SCOPE:

* This prediction of Concrete strength will become more effective, as the ML algorithms and concepts are being developed.

* In future, the predictions of any models will become more accurate.

11. BIBILOGRAPHY:

11.1 REFERENCES:

[1] Hibbeler R.C.C. Resistência dos materiais. 7th ed., Pearson, São Paulo, SP, 2010.

[2] Babanajad S.K. Application of genetic programming for uniaxial and multiaxial modeling of concrete. Springer, Switzerland, 2015.

[3] Hoang N.D., Tran X.L., Le C.H., Nguyen D.T. A backpropagation artificial neural network software program for data classification in civil engineering developed in .NET Framework, DTU J. Sci. Technol. 03, 51–56, 2019.

[4] Reuter U., Sultan A., Reischl D.S. A comparative study of machine learning approaches for modeling concrete failure surfaces. Adv. Eng. Softw., 116:67–79, 2018.

[5] Torky A.A., Aburawwash A.A. A deep learning approach to automated structural engineering of prestressed members. Int. J. Struct. Civ. Eng., 7:347–352, 2018.

[6] SmartInternz Course material and project guide materials.

APPENDIX

A. ***SOURCE CODE***:

- ***Project Model-Building:***

In [1]:
```
import pandas as pd
import numpy as np
from collections import Counter as c
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,mean_squared_error,mean_absolute_error
import pickle
from sklearn.linear_model import LinearRegression
```

In [2]:
```
data=pd.read_csv("concrete.csv")
```

In [3]:
```
data.tail()
```

Out[3]:

	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)
1025	276.4	116.0	90.3	179.6	8.9	870.1	768.3	28	44.28
1026	322.2	0.0	115.6	196.0	10.4	817.9	813.4	28	31.18
1027	148.5	139.4	108.6	192.7	6.1	892.4	780.0	28	23.70
1028	159.1	186.7	0.0	175.6	11.3	989.6	788.9	28	32.77
1029	260.9	100.5	78.3	200.6	8.6	864.5	761.5	28	32.40

In [4]:
```
data.head()
```

Out[4]:

	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)
0	540.0	0.0	0.0	162.0	2.5	1040.0	676.0	28	79.99
1	540.0	0.0	0.0	162.0	2.5	1055.0	676.0	28	61.89
2	332.5	142.5	0.0	228.0	0.0	932.0	594.0	270	40.27
3	332.5	142.5	0.0	228.0	0.0	932.0	594.0	365	41.05
4	198.6	132.4	0.0	192.0	0.0	978.4	825.5	360	44.30

In [5]:
```
data.columns = [col[:col.find("(")].strip() for col in data.columns]
data.head(1)
```

Out[5]:

	Cement	Blast Furnace Slag	Fly Ash	Water	Superplasticizer	Coarse Aggregate	Fine Aggregate	Age	Concrete compressive strength
0	540.0	0.0	0.0	162.0	2.5	1040.0	676.0	28	79.99

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1030 entries, 0 to 1029
Data columns (total 9 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Cement                        1030 non-null   float64
 1   Blast Furnace Slag            1030 non-null   float64
 2   Fly Ash                       1030 non-null   float64
 3   Water                         1030 non-null   float64
 4   Superplasticizer              1030 non-null   float64
 5   Coarse Aggregate              1030 non-null   float64
 6   Fine Aggregate                1030 non-null   float64
 7   Age                           1030 non-null   int64  
 8   Concrete compressive strength 1030 non-null   float64
dtypes: float64(8), int64(1)
memory usage: 72.5 KB
```

In [7]:
```
data.describe()
```

Out[7]:

	Cement	Blast Furnace Slag	Fly Ash	Water	Superplasticizer	Coarse Aggregate	Fine Aggregate	Age	Concrete compressive strength
count	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000
mean	281.167864	73.895825	54.188350	181.567282	6.204660	972.918932	773.580485	45.662136	35.817961
std	104.506364	86.279342	63.997004	21.354219	5.973841	77.753954	80.175980	63.169912	16.705742
min	102.000000	0.000000	0.000000	121.800000	0.000000	801.000000	594.000000	1.000000	2.330000
25%	192.375000	0.000000	0.000000	164.900000	0.000000	932.000000	730.950000	7.000000	23.710000
50%	272.900000	22.000000	0.000000	185.000000	6.400000	968.000000	779.500000	28.000000	34.445000
75%	350.000000	142.950000	118.300000	192.000000	10.200000	1029.400000	824.000000	56.000000	46.135000
max	540.000000	359.400000	200.100000	247.000000	32.200000	1145.000000	992.600000	365.000000	82.600000

In [8]:
```
data.isnull().any()
```

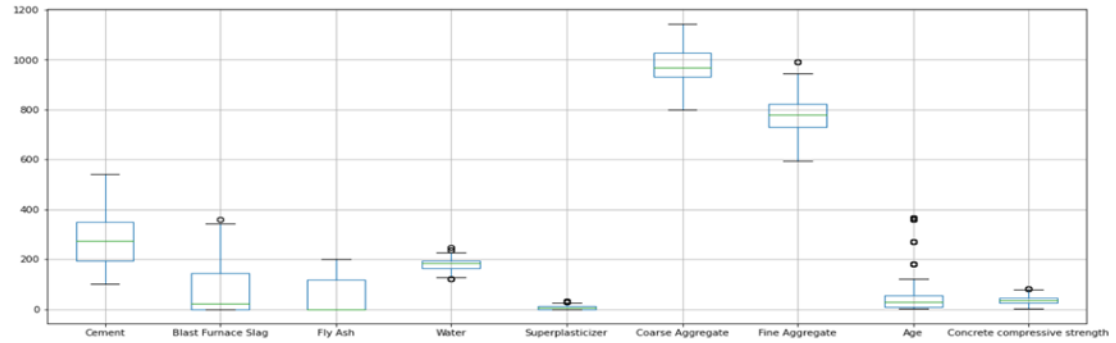
Out[8]:
```
Cement                          False
Blast Furnace Slag              False
Fly Ash                         False
Water                           False
Superplasticizer                False
Coarse Aggregate                False
Fine Aggregate                  False
Age                             False
Concrete compressive strength   False
dtype: bool
```

In [9]:
```
data.isnull().sum()
```

Out[9]:
```
Cement                          0
Blast Furnace Slag              0
Fly Ash                         0
Water                           0
Superplasticizer                0
Coarse Aggregate                0
Fine Aggregate                  0
Age                             0
Concrete compressive strength   0
dtype: int64
```

In [10]:
```
data.boxplot(figsize=(18,7))
```

Out[10]: <AxesSubplot:>



In [11]:
```
x=pd.DataFrame(data,columns=data.columns[:8])
y=pd.DataFrame(data,columns=data.columns[8:])
```

In [12]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=1)
```

In [13]:
```
from sklearn.ensemble import GradientBoostingRegressor
gbr=GradientBoostingRegressor()
gbr.fit(x_train,np.ravel(y_train))
```

Out[13]:
```
GradientBoostingRegressor()
```

In [14]:
```
y_pred =gbr.predict(x_test)
print("Prediction made by Gradient Boosting model:",y_pred)
```

```
Prediction made by Gradient Boosting model: [21.95234148 13.08162009 24.70391082 11.32025646 33.85349972 64.77465613
 18.45044902 48.72305133 40.43710467 20.12460115 38.64180627 14.73181983
 37.33435548 31.47690119  8.39548073 28.19448467 38.81451032 60.80235308
 55.41349105 36.17514766 22.09653922 50.34586419 17.54229031 59.22520866
 60.28299988 39.18419497 33.47870521 29.62105842 35.97574895 13.79206845
 42.2242261  54.78728341 27.06673021 24.87332946 34.92824549 39.36193638
 16.50942869 26.70669935 25.92994908 14.78114178 39.53782087 13.46243396
 47.04232237 38.96167764 35.78935294 27.67350801 33.28586165 27.08849056
 34.08139794 51.48267971 25.15450245 35.37386967 61.10296174 53.1944813
 56.51847802  8.85042266 38.02781274 22.95231442 32.99320901 44.64324648
 49.32434581 41.57929522 52.63226021 49.50361013 22.54109402 34.73179197
 48.47116198 35.05219693 40.66955402 27.65021804 29.83790459 27.00297825
 39.36193638 12.80090167 67.55248442 58.04726327 53.09303053 14.88848036
 60.3385503  43.22352746 37.64200033 44.22126966 49.57778314 33.2643738
 45.31391031 41.93153053 33.58746541 60.52581393 49.57778314 49.71624993
 33.9313478  36.22609241 28.20107518 40.04633608 31.31587817 68.47187226
 38.50036265 12.5517226  31.74158529 29.19674972 35.97574895 61.51911155
 36.17473814 41.82750579 46.5796786  36.20474961 26.76246129 37.07609354
 16.57585852 23.60374375 23.2620048  35.29585805  9.41309786 27.67350801
 32.39097473 16.33648516 36.37686244 40.46295946 46.66434763 56.19940098
 76.91180194 57.01508663 38.50036265 19.2760845  13.40745862 28.41321395
 54.95947943 14.10626622 69.92003204 21.70176448 16.19854885 32.73741861
 14.54489222 33.89042188 38.86275688 40.16331248 32.39097473 38.27628855
 34.27967922 11.95881816 52.54721058 38.0769414  35.39508201 44.19367588
 66.22672154 47.68704002 40.16331248 16.98263838 34.92824549 44.32035003
 67.90656588 47.17307745 80.71643822 45.72714702 32.41107987 29.63765677
 27.08849056 35.10386618 54.88493496 51.76906215 26.74362967 48.89461558
 47.37026737 22.90851667 59.46738753 43.63203337 44.66027029 32.08914728
 37.14901501 23.61010371 38.12198157 13.76147006 43.76526343 30.33367865
 47.04232237 61.75066787 45.74176051 42.06965495 33.51071773 79.33930221
 27.33493371 62.58029953 34.49132432 38.67994386 28.64681362 39.97154441
  5.87053159 24.39809138 30.20971064 34.07030098 72.88888085 24.48121141
 27.61685941 39.11765922 10.04541508 55.11080044 36.53168157 28.41321395
 28.02038331 38.96167764 36.76346037 46.27485185  6.15696776 36.76346037
 35.15677734 40.0706007 ]
```

In [15]:
```
score=gbr.score(x_test,y_test)

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

```
Score of Gradient Boosting model: 0.8857099247493276
```

In [16]:
```
from sklearn import metrics
print("MAE:",metrics.mean_absolute_error(y_test,y_pred))
print("MSE:",metrics.mean_squared_error(y_test,y_pred))
print("RMSE:",np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
```

```
MAE: 3.836299099372565
MSE: 29.79480313984124
RMSE: 5.458461609266959
```

In [17]:
```
pickle.dump(gbr,open('cement.pkl','wb'))
```

app.py codes:

```
# importing the necessary dependencies
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)

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',8080) #local host 8080
```

Html and css code of INDEX PAGE:

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="UTF-8">
```

```
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <title>Predicting Compressive Strength of Concrete</title>
                                                                    <link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css" rel="stylesheet">
    <link href="../static/css/main.css" rel="stylesheet" type="text/css">

        <style>body {
                                                            background-image: url("https://img3.goodfon.com/wallpaper/nbig/4/43/suntec-city-singapore.jpg");
 opacity: 0.7;
 background-repeat: no-repeat;
 background-attachment: fixed;
 background-size: 100% 100%;
}
h2 {
        text-align: center;
        padding-top: 100px;
        color: black;

}
h1 {
        text-align: center;
        padding-top: 100px;
        color: black;
}

.one {
        background-color: #D3D3D3;
```

```
        border-radius: 10px;
        color: orange;
        margin-left: 80px;
        margin-bottom: 10px;
    margin-right: 80px;

}
p {
        font-family: 'Poppins', sans-serif;
        padding-left: 30px;
    padding-top: 10px;
    margin-top: px;
    margin-bottom: 0.5px;
}
input {
        margin-left: 30px;
        border-radius: 3px;
    padding-left: 10px;
    padding-top: 3px;
    padding-bottom: 3px;
    margin-right: 30px;

}
.i2 {
        width: 80%
}
.i3 {
        width:5%
}
.p1 {
        font-family: 'Poppins', sans-serif;
        padding-left: 30px;
```

```
    padding-top: 5px;
}
.loginbtn{
        background-color:black;
        width: 90%;
        color: white;
        margin-top: 10px;
    margin-bottom: 20px;
}
.p2 {
        padding-bottom: 10px;
}
        *{
    margin: 0;
    padding: 0;
}
body{
    font-family: 'Lato', sans-serif;
}
.wrapper{
    width: 1170px;
    margin: auto;
}
.button1 {
  background-color: black;
  color: white;
  border: 2px solid blue; /* blue */
}
header{
background:lineargradient(rgba(0,0,0,0.8),rgba(0,0,0,0.8)),url(https://img3.goodfon.com/wallpaper/nbig/4/43/suntec-city-singapore.jpg);
    height: 100vh;
```

```
    -webkit-background-size: cover;
    background-size: cover;
    background-position: center center;
    position: relative;
}

.x {
  background-color: white;
  color: black;
  border: 2px solid #e7e7e7;
}
input[type=button], input[type=submit], input[type=reset] {
  background-color: black;
  border: none;
  color: orange;
  padding: 16px 32px;
  text-decoration: none;
  margin: 4px 2px;
  cursor: pointer;
}
.nav-area{
    float: right;
    list-style: none;
    margin-top: 30px;
     background-color: white;
  color: black;
  border: 2px solid #e7e7e7;
}
.nav-area li{
    display: inline-block;
}
.nav-area li a {
```

```
color: #fff;
text-decoration: none;
padding: 5px 20px;
font-family: poppins;
font-size: 16px;
text-transform: uppercase;
}
.nav-area li a:hover{
    background: #fff;
    color: #333;
}
.logo{
    float: left;
}
.logo img{
    width: 100%;
    padding: 15px 0;
}
.welcome-text{
    position: absolute;
    width: 600px;
    height: 300px;
    margin: 20% 30%;
    text-align: center;
}
.welcome-text h1{
    text-align: center;
    color: #fff;
    text-transform: uppercase;
    font-size: 35px;
    text-shadow: 2px 2px #87ceeb;
}
```

```
.welcome-text a{
    border: 1px solid #fff;
    padding: 10px 25px;
    text-decoration: none;
    text-transform: uppercase;
    font-size: 14px;
    margin-top: 20px;
    display: inline-block;
    color: #fff;
}
.welcome-text a:hover{
    background: #fff;
    color: #333;
}

/*resposive*/

@media (max-width:600px){
    .wrapper {
width: 100%;
}
.logo {
float: none;
width: 50%;
text-align: center;
margin: auto;
}
img {
width: ;
}
.nav-area {
float: none;
```

```
margin-top: 0;
}
   .nav-area li a {
padding: 5px;
font-size: 11px;
}
.nav-area {
text-align: center;
}
   .welcome-text {
width: 100%;
height: auto;
margin: 30% 0;
}
   .welcome-text h1 {
font-size: 30px;
}
}
        </style>
</head>
<body>

 <div class="col-md-6">
<h1>Predicting Compressive Strength of Concrete</h1>
        <div class="one">
                <form action="predict" method="post">
                        <p><b>CEMENT</b></p>

                                <input class="i2" type="text" name="Cement"
placeholder="Enter Cement Value"><br>

                        <p><b>Blast Furnace Slag</b></p>
```

```
            <input class="i2" type="text" name="Blast Furnace Slag"
                                placeholder="Enter Blast Furnace Slag value"><br>
        <p><b>Fly Ash</b></p>
                                <input class="i2" type="text" name="Fly Ash"
                                placeholder="Enter Fly Ash value"><br>
                                <p><b>Water</b></p>
                                <input class="i2" type="text" name="Water"
                                 placeholder="Enter Water quantity"><br>
                                <p><b>Uperplasticizer</b></p>
                                <input class="i2" type="text" name="Superplasticizer"
                                 placeholder="Enter Superplasticizer"><br>
                                <p><b>Coarse Aggregate</b></p>
                                <input class="i2" type="text" name="Coarse Aggregate"
                                placeholder="Enter Coarse Aggregate"><br>
                                <p><b>Fine Aggregate</b></p>
                                 <input class="i2" type="text" name="Fine Aggregate"
                                 placeholder="Enter Fine Aggregate"><br>
                                 <p><b>Age</b></p>
                                                <input class="i2" type="text" name="Age"
        placeholder="Enter Age"><br>
                        <input type="submit" class="loginbtn" value="Predict" />

        <br></form>
</div></div></body></html>
```