

PREDICTING COMPRESSIVE STRENGTH OF CONCRETE

INTRODUCTION

a)OVERVIEW

To analyse Concrete Compressive Strength dataset and build Machine Learning models to predict the compressive strength. The Compressive Strength of Concrete determines the quality of Concrete. This is generally determined by a standard crushing test on a concrete cylinder. This project is about build a model for predicting the strength of concrete with components of it.

b)PURPOSE

Standard crushing test 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.

LITERATURE SURVEY

EXISTING PROBLEM& PROPOSED SOLUTION

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.

EXPERIMENTAL ANALYSIS

For speedy work and reliable concrete mixes, ready mix (RMX) concrete plants are a better choice than a laboratory and it is an integral part of the construction industry. The data used for this study was collected from a single RMX plant over a 3 month period (to try and minimise external variables such as seasonal temperature effects) and includes cement, water, fine aggregate (FA), coarse aggregate (CA) quantity and density. The data of mix and testing results pertaining to the various grades of concrete ranging in specified strength () 17.5 to 60MPa are summarized in the table below as well as the correlation of the independent variables to the 28 day compressive strength.

ADVANTAGES:

Compressive strength of concrete is one of the most important and useful properties. As a construction material, concrete is employed to resist compressive stresses. While, at locations where tensile strength or shear strength is of primary importance, the compressive strength is used to estimate the required property. The model predict the compressive strength and saves time and cost.

CODE:

MODEL BUILDING

Importing the Libraries

```
import pandas as pd
```

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

Importing the Dataset

```
dataset=pd.read_excel("Concrete_Data.xls")
```

```
dataset
```

```
dataset.columns
```

Data Visualization

```
data=dataset.iloc[0:,0:].values
```

```
data
```

```
df = pd.DataFrame(data, columns = ['Cement', 'Blast Furnace Slag',  
                                   'Fly Ash', 'Water',  
                                   'Superplasticizer', 'Coarse Aggregate', 'Fine Aggregate', 'Age', 'Concrete compressive  
strength'] )
```

```
df.plot.box()
```

```
plt.scatter(df['Water'], df['Concrete compressive strength'])
```

```
plt.show()
```

Taking care of missing data

```
dataset.isnull().any().
```

Outliers

```
data=dataset.iloc[0:,0:].values
```

```
data
```

```
from scipy import stats
```

```
z=np.abs(stats.zscore(data))
```

```
threshold=3
```

```
np.where(z>threshold)
```

```
dataset.drop(2,axis=0,inplace=True)
```

```
dataset
```

```
dataset.drop([3,4,6],axis=0,inplace=True)
```

```
dataset
```

```
dataset.drop([ 12, 17, 24, 25, 26, 30, 31, 33, 34,  
              35, 41, 42, 56, 60, 61, 63, 65, 66, 76, 79, 99, 102,  
              122, 125, 145, 148, 168, 171, 553, 559, 571, 584, 604, 610, 616,  
              620, 622, 756, 769, 792, 798, 814, 820, 873, 936],axis=0,inplace=True)
```

```
Dataset
```

```
import seaborn as sns
```

```
sns.boxplot(dataset['Cement (component 1)(kg in a m^3 mixture)'])
```

Splitting data into Train and Test

```
x=dataset.iloc[:,0:8].values
```

```
x
```

```
y=dataset.iloc[:,9].values
```

```
y
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

```
from sklearn.linear_model import LinearRegression
```

```
mr=LinearRegression()
```

```
mr.fit(x_train,y_train)
```

```
LinearRegression()
```

```
y_pred=mr.predict(x_test)
```

```
y_pred
```

```
from joblib import dump
```

```
dump(mr,"pro1.save")
```

Evaluation

```
from sklearn.metrics import mean_squared_error

from sklearn.metrics import r2_score

r2_score(y_test,y_pred)
```

index.html

```
<!DOCTYPE html>

<html >

<!--From https://codepen.io/frytyler/pen/EGdtg-->

<head>

  <meta charset="UTF-8">

  <title>ML API</title>

  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet'
type='text/css'>

<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">

<style>

.login{
top: 20%;
}

</style>

</head>

<body style="background-color:#273746;">

<div class="login">

    <h2>Compressive strength of Concrete Prediction</h2>

    <!-- Main Input For Receiving Query to our ML -->

    <form action="{{ url_for('y_predict')}}" method="post">

        <input type="text" name="Cement" placeholder="Cement" required="required" />

        <input type="text" name="Blast Furnace Slag" placeholder="Blast Furnace Slag"
```

```

required="required" />

        <input type="text" name="Fly Ash" placeholder="Fly Ash" required="required" />
        <input type="text" name="Water" placeholder="Water" required="required" />
        <input type="text" name="Superplasticizer" placeholder="Superplasticizer"
required="required" />
        <input type="text" name="Coarse Aggregate" placeholder="Coarse Aggregate"
required="required" />
        <input type="text" name="Fine Aggregate" placeholder="Fine Aggregate"
required="required" />

        <input type="text" name="Age day" placeholder="Age day" required="required" />

        <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>

    </form>

<br>

<br>

    {{ prediction_text }}

</div>

</body>

</html>

```

Style.css

```

@import url(https://fonts.googleapis.com/css?family=Open+Sans);

.btn { display: inline-block; *display: inline; *zoom: 1; padding: 4px 10px 4px; margin-bottom: 0;
font-size: 13px; line-height: 18px; color: #333333; text-align: center; text-shadow: 0 1px 1px rgba(255,
255, 255, 0.75); vertical-align: middle; background-color: #f5f5f5; background-image:
-moz-linear-gradient(top, #ffffff, #e6e6e6); background-image: -ms-linear-gradient(top, #ffffff, #e6e6e6);
background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#ffffff), to(#e6e6e6)); background-image:
-webkit-linear-gradient(top, #ffffff, #e6e6e6); background-image: -o-linear-gradient(top, #ffffff, #e6e6e6);
background-image: linear-gradient(top, #ffffff, #e6e6e6); background-repeat: repeat-x; filter:
progid:dximagetransform.microsoft.gradient(startColorstr=#ffffff, endColorstr=#e6e6e6,
GradientType=0); border-color: #e6e6e6 #e6e6e6 #e6e6e6; border-color: rgba(0, 0, 0, 0.1) rgba(0, 0, 0,
0.1) rgba(0, 0, 0, 0.25); border: 1px solid #e6e6e6; -webkit-border-radius: 4px; -moz-border-radius: 4px;
border-radius: 4px; -webkit-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0,
0.05); -moz-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05);
box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); cursor: pointer;
*margin-left: .3em; }

```

```

.btn:hover, .btn.active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }

.btn-large { padding: 9px 14px; font-size: 15px; line-height: normal; -webkit-border-radius: 5px;
-moz-border-radius: 5px; border-radius: 5px; }

.btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position: 0
-15px; -webkit-transition: background-position 0.1s linear; -moz-transition: background-position 0.1s
linear; -ms-transition: background-position 0.1s linear; -o-transition: background-position 0.1s linear;
transition: background-position 0.1s linear; }

.btn-primary, .btn-primary:hover { text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25); color: #ffffff; }

.btn-primary.active { color: rgba(255, 255, 255, 0.75); }

.btn-primary { background-color: #4a77d4; background-image: -moz-linear-gradient(top, #6eb6de,
#4a77d4); background-image: -ms-linear-gradient(top, #6eb6de, #4a77d4); background-image:
-webkit-gradient(linear, 0 0, 0 100%, from(#6eb6de), to(#4a77d4)); background-image:
-webkit-linear-gradient(top, #6eb6de, #4a77d4); background-image: -o-linear-gradient(top, #6eb6de,
#4a77d4); background-image: linear-gradient(top, #6eb6de, #4a77d4); background-repeat: repeat-x;
filter: progid:dximagetransform.microsoft.gradient(startColorstr=#6eb6de, endColorstr=#4a77d4,
GradientType=0); border: 1px solid #3762bc; text-shadow: 1px 1px 1px rgba(0,0,0,0.4); box-shadow:
inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.5); }

.btn-primary:hover, .btn-primary:active, .btn-primary.active, .btn-primary.disabled,
.btn-primary[disabled] { filter: none; background-color: #4a77d4; }

.btn-block { width: 100%; display:block; }

* { -webkit-box-sizing:border-box; -moz-box-sizing:border-box; -ms-box-sizing:border-box;
-o-box-sizing:border-box; box-sizing:border-box; }

```

```

html { width: 100%; height:100%; overflow:hidden; }

```

```

body {
    width: 100%;
    height:100%;
    font-family: 'Open Sans', sans-serif;
    background: #092756;
    color: #fff;
    font-size: 18px;
}

```

```

        text-align:center;

        letter-spacing:1.2px;

        background: -moz-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4)
10%,rgba(138,114,76,0) 40%,-moz-linear-gradient(top, rgba(57,173,219,.25) 0%, rgba(42,60,87,.4)
100%), -moz-linear-gradient(-45deg, #670d10 0%, #092756 100%);

        background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4)
10%,rgba(138,114,76,0) 40%), -webkit-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4)
100%), -webkit-linear-gradient(-45deg, #670d10 0%,#092756 100%);

        background: -o-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4)
10%,rgba(138,114,76,0) 40%), -o-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%),
-o-linear-gradient(-45deg, #670d10 0%,#092756 100%);

        background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4)
10%,rgba(138,114,76,0) 40%), -ms-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4)
100%), -ms-linear-gradient(-45deg, #670d10 0%,#092756 100%);

        background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4)
10%,rgba(138,114,76,0) 40%), linear-gradient(to bottom, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4)
100%), linear-gradient(135deg, #670d10 0%,#092756 100%);

        filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#3E1D6D',
endColorstr='#092756',GradientType=1 );

    }

    .login {

        position: absolute;

        top: 40%;

        left: 50%;

        margin: -150px 0 0 -150px;

        width:400px;

        height:400px;

    }

    .login h1 { color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-spacing:1px; text-align:center; }

    input {

```

```

width: 100%;
margin-bottom: 10px;
background: rgba(0,0,0,0.3);
border: none;
outline: none;
padding: 10px;
font-size: 13px;
color: #fff;
text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
border: 1px solid rgba(0,0,0,0.3);
border-radius: 4px;
box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
-webkit-transition: box-shadow .5s ease;
-moz-transition: box-shadow .5s ease;
-o-transition: box-shadow .5s ease;
-ms-transition: box-shadow .5s ease;
transition: box-shadow .5s ease;
}
input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px rgba(255,255,255,0.2); }

```

app.py

```

import numpy as np

from flask import Flask, request, jsonify, render_template
from joblib import load

app = Flask(__name__)

model=load('pro.save')

@app.route('/')
def home():
    return render_template('index.html')

```

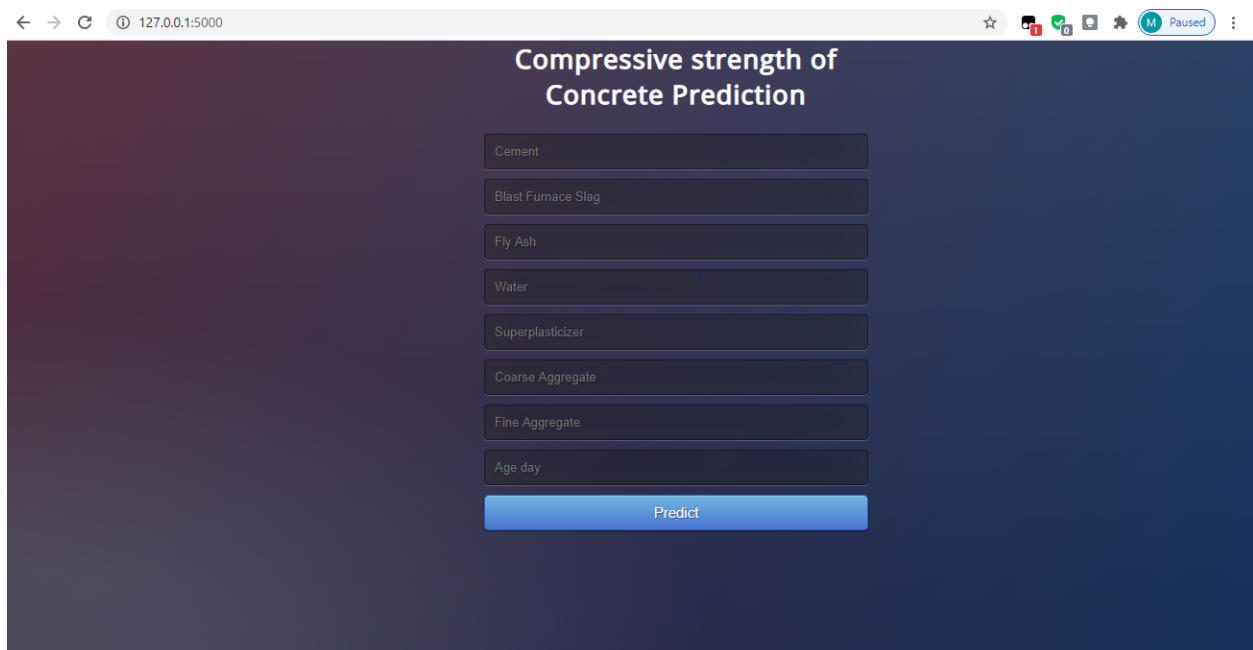


```

@app.route('/y_predict',methods=['POST'])
def y_predict():
    '''For rendering results on HTML GUI
    '''x_test = [[int(x) for x in request.form.values()]]
    #print(x_test)
    prediction = model.predict(x_test)
    output=prediction[0]
    return render_template('index.html', prediction_text='Concrete strength {}'.format(output))
'''@app.route('/predict_api',methods=['POST'])
def predict_api():
    #For direct API calls through request
    data = request.get_json(force=True)
    prediction = model.y_predict([np.array(list(data.values()))])
    output = prediction[0]
    return jsonify(output)'''
if __name__ == "__main__":
    app.run(debug=True)

```

OUTPUT:



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000'. The page title is 'Compressive strength of Concrete Prediction'. The form contains the following input fields and a button:

- Cement
- Blast Furnace Slag
- Fly Ash
- Water
- Superplasticizer
- Coarse Aggregate
- Fine Aggregate
- Age day
- Predict

Compressive strength of
Concrete Prediction

Cement

Blast Furnace Slag

Fly Ash

Water

Superplasticizer

Coarse Aggregate

Fine Aggregate

Age day

Predict

Concrete strength 59.377776570384206

CONCLUSIONS

It presents a simple mathematical model to predict the concrete compressive strength from the early age test results. In this study, the concrete strength characteristic with age is modeled by a multiple linear regression (MLR) mathematical equation. Early age test data are being used in this case to get reliable values of the 28 day strength prediction. Herein, a simple and practical approach has been described for prediction of 28-day compressive strength of concrete and the proposed technique can be used as a reliable tool for assessing the strength of concrete from quite early test results. This will help in making quick decision at site and reduce delay in the execution time of large civil construction projects.