Concrete Strength Prediction

Maansi Bhatnagar

November 27, 2023

Contents

List of Tables

List of Figures

Phase 1

1 Introduction

Concrete strength plays a pivotal role in shaping the design, construction, and upkeep of structures, exerting a direct impact on their safety, longevity, and economic efficiency. Given my keen interest in architectural designs and the elements contributing to structural durability, the ability to anticipate concrete strength proves instrumental in making informed decisions and averting potential disasters. In this project, I will try to implement a neural network model to predict the strength of the concrete.

2 Data Source

The dataset 'Calculate Concrete Strength' has been procured from the Kaggle Data Science website. The dataset is freely available for public use and analytics. The dataset contains various information about various cements and concretes.

2.1 Dataset Details

The dataset is made up of 1030 rows and 9 columns. So there are 1030 Cement types and 8 input features that are used to measure the strength of the concrete. The 9th column is 'Strength' which is our target set or variable. The input features are:

1. Cement
2. Blast Furnace Slag
3. Fly Ash
4. Water
5. Superplasticizer
6. Coarse Aggregate
7. Fine Aggregate
8. Age

3 Data Exploration

3.1 No missing or null values.

- Cement: 0
- Blast Furnace Slag : 0
- Fly Ash : 0
- Water : 0
- Superplasticizer : 0
- Coarse Aggregate : 0
- Fine Aggregate : 0
- Age : 0
- Strength : 0
- Strength Class : 0

3.2 Numerical category

.

Column	Non-Null	Count	Dtype
Cement	1030	non-null	float64
Blast Furnace Slag	1030	non-null	float64
Fly Ash	1030	non-null	float64
Water	1030	non-null	float64
Superplasticizer	1030	non-null	float64
Fine Aggregate	1030	non-null	float64
Coarse Aggregate	1030	non-null	float64
Age	1030	non-null	int64
Strength	1030	non-null	float64

Table 1: Data Info

3.3 Input Data Visualization

The histogram plot of each input feature shows their maximum and minimum values and their distribution across the dataset.

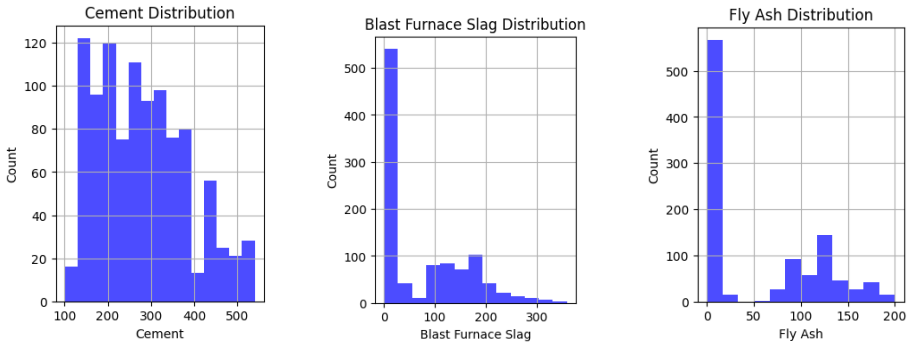


Figure 1: Input Feature Histograms - 1

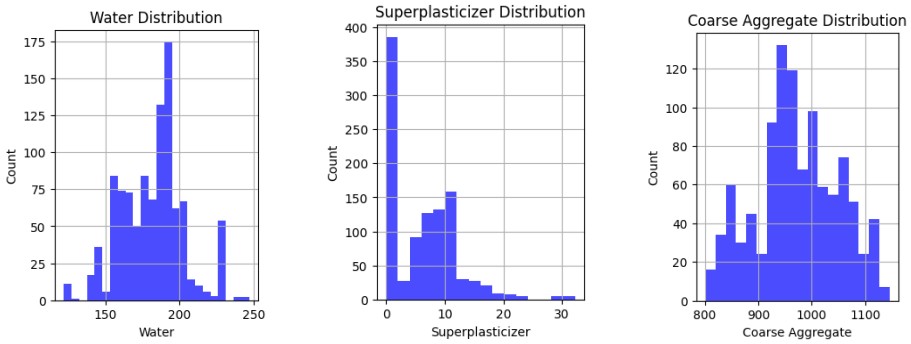


Figure 2: Input Feature Histograms - 2

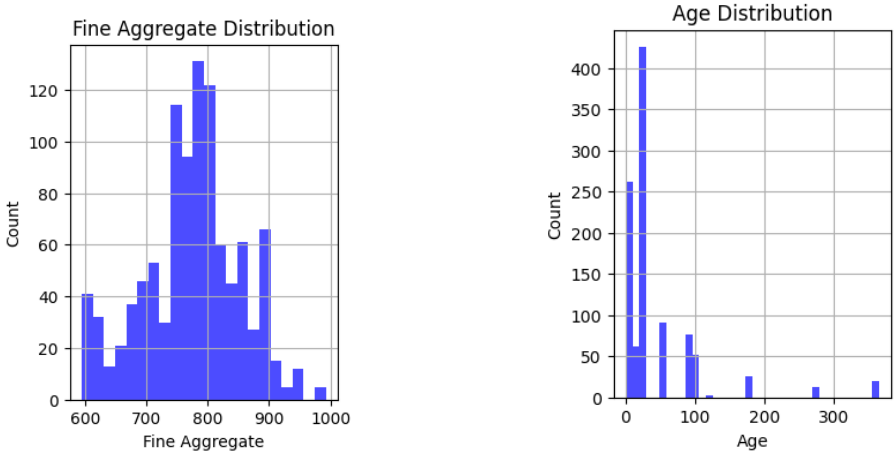


Figure 3: Input Feature Histograms - 3

3.4 Input Feature Statistics

	Min	Max	Mean	Median	Std
Cement	102.0	540.0	281.167864	272.9	104.506364
Blast Furnace Slag	0.0	359.4	73.895825	22.0	86.279342
Fly Ash	0.0	200.1	54.188350	0.0	63.997004
Water	121.8	247.0	181.567282	185.0	21.354219
Superplasticizer	0.0	32.2	6.204660	6.4	5.973841
Coarse Aggregate	801.0	1145.0	972.918932	968.0	77.753954
Fine Aggregate	594.0	992.6	773.580485	779.5	80.175980
Age	1.0	365.0	45.662136	28.0	63.169912

Table 2: Input Feature Satistics

4 Data Processing

4.1 Normalization of Dataset

As seen through the Histograms, the dataset has input features with varying scales, and normalizing the data might be beneficial. If I do choose to use a Neural Network model, normalizing the data helps improve convergence during training. So we pre-process the data using normalization techniques. Normalization helps in faster convergence of algorithms as well. The formula to calculate the min-max value:

$$\frac{value - min}{max - min}$$

4.2 Input Featuring

I have added a column 'Strength Class' to act as our output variable. I have preprocessed our target set 'Strength' to convert the problem into a Binary Classification problem and test the Logistic Regression model. I initially took the threshold value as 0.5 to classify the cement as 'Strong' or 'Weak'. However, this threshold value led to an imbalanced output variable.

Weak = 711	Strong = 319

Table 3: Threshold value = 0.5

Therefore, to have a more balanced output, I chose the threshold value to be 0.4

Weak = 515	Strong = 515

Table 4: Threshold value = 0.5

Output Data Visualization

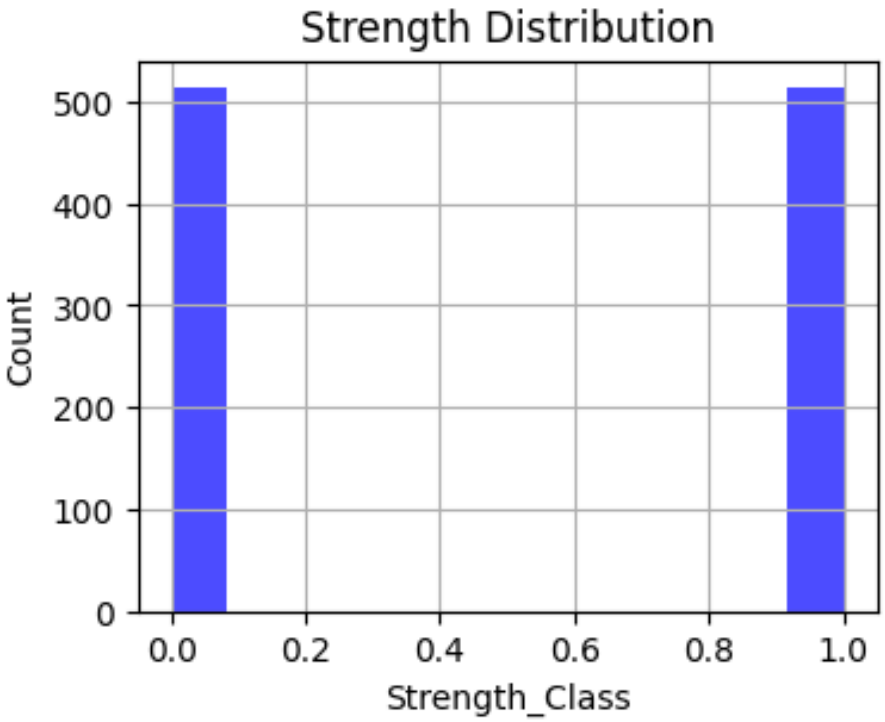


Figure 4: Output Feature Visualization

4.3 Data Analysis

After normalization of the input data, and adding 'Strength Class', as our target output variable, I have manually encoded 'Strong' as 1 and 'Weak' as 0. So there are 2 output labels: 1 and 0.

Relation between Input Feature and Output Feature

I have plotted a scatter plot along with the Regression line to analyze the relationship between each input feature and 'Strength'.

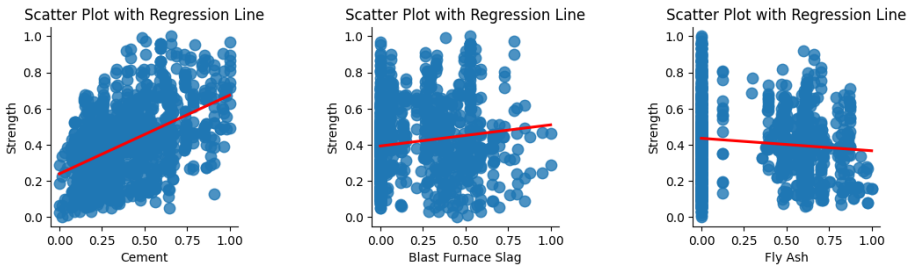


Figure 5: Input Feature vs Output variable -1

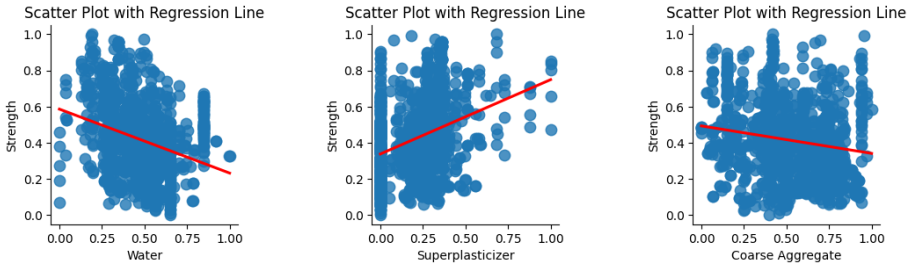


Figure 6: Input Feature vs Output variable -2

Figure 7: Input Feature vs Output variable -3

The regression line in each scatter plot signifies the relation of the input feature with the target feature 'Strength'. As we can observe the input features 'Cement number', 'Blast Furnace Slag', Superplasticizer', and 'Age' positively impact the Strength of the

cement, while input features 'Fly Ash', 'Water', 'Coarse Aggregate, 'Fine Aggregate' contribute negatively towards the cement Strength.

Phase 2

5 Building Models

To start with modeling my data, I first determined how big a model was needed to overfit my data. Without splitting the data into training and validation sets, I first experimented with the Logistic Regression model and trained my data. I first randomly shuffled all the columns for a more robust training. I then fit the logistic regression model over the entire dataset. The accuracy score of the model will be discussed in the next phase of the project.

To achieve a better accuracy score, I experimented with 3 Neural Network models, each containing a different number of neurons in the layers. The Neural Network model with the highest accuracy score on training and validation sets was chosen as the final model to be trained.

Models that have been experimented on:

- Logistic Regression
- Neural Network Model (16-8-4-1)
- Neural Network Model (16-8-1)
- Neural Network Model (8-4-1)

Phase 3

6 Data Modelling, Evaluation and Comparison

6.1 Logistic Regression

For an easier application, I converted my problem into a binary classification problem and my target set had only 2 output labels; 0 and 1. The column 'Strength Class' was our target variable. The table below shows the accuracy achieved by applying Logistic Regression.

Model	Training Acc	Validation Acc
Logistic Regression	95.56172 %	97.73462 %

Table 5: Logistic Regression - Accuracy

6.2 Neural Network Model

Since I was not able to reach a near 100 % accuracy using Logistic Regression, I experimented with Neural Network models with varying layers and neurons in layers. The 3 Neural Network Models that were experimented with:

- Neural Network Model (16-8-4-1)
- Neural Network Model (16-8-1)
- Neural Network Model (8-4-1)

The table below shows the comparison between all the 4 data models:

Model	Training Acc.	Validation Acc.
Logistic Regression	95.56172 %	97.73462 %
Neural Network (16-8-4-1)	99.7226 %	99.35275 %
Neural Network (16-8-1)	97.91955 %	98.0582 %
Neural Network (8-4-1)	96.9486 %	96.1165 %

Table 6: Accuracy Comparison of all data models

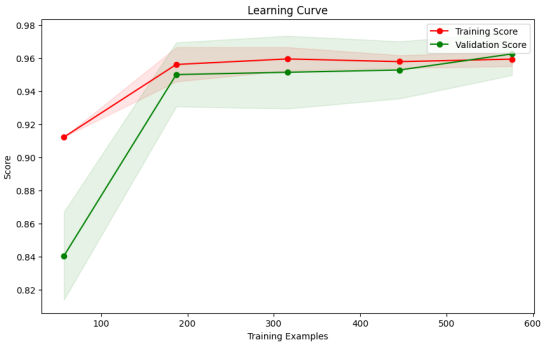
The table below compares the F1 score, precision and Recall of the 4 data models:

Model	Precision	Recall	F1 score
Logistic Regression	0.961	0.943	0.952
Neural Network Model (16-8-4-1)	0.98	1.0	0.99
Neural Network Model (16-8-1)	0.95	0.98	0.96
Neural Network Model (8-4-1)	0.96	0.99	0.98

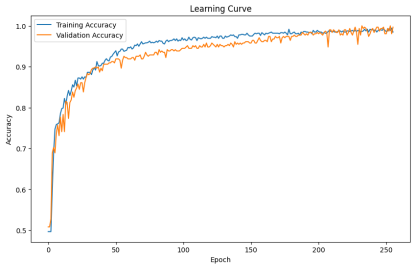
Table 7: Precision, Recall, F1 score values

6.3 Learning Curves

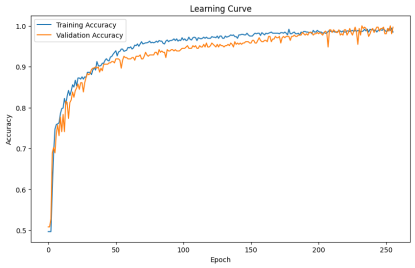
The learning curves show that the Neural Network Models are performing better than the Logistic Regression Model and Table 6 and Table 7, showcase that the Neural Network Model (16-8-4-1) performs the best among all the 4 data models considered.



(a) Learning Curve - Logistic Regression



(a) Learning Curve - NN model (16-8-4-1)



(b) Learning Curve - NN model (16-8-1)

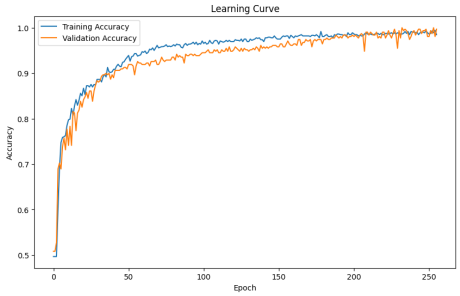


Figure 10: Learning Curve - NN model (8-4-1)

6.4 Output as Input Feature

Overfitting occurs when a model performs well on the training data but fails to generalize to new, unseen data. Providing output as an input feature may lead to overfitting as the model will learn on the basis of the target set and form patterns and may not give correct predictions for unseen data.
We can use regularization technique on the NN model, and add dropout layers to our Neural Network model (16-8-4-1) to prevent overfitting.

6.5 Custom function for predictions

I worked on building a custom function to serve as a prediction model and it has been evaluated using Recall, Precision, and F1 score. I used the weights of my selected Neural Network model and put them in my custom function to gain the predictions. The table below compares the Precision, Recall and F1 scores of NN model and Custom funciton.

	Precision	Recall	F1
NN model	0.98	1.0	0.99
Custom Function	0.96	0.97	0.97

Table 8: Comparing NN model and Custom Function

7 Model Evaluation using ROC and AUC

Since, I converted my problem into a binary classification problem, I also evaluated he NN model using ROC and AUC.

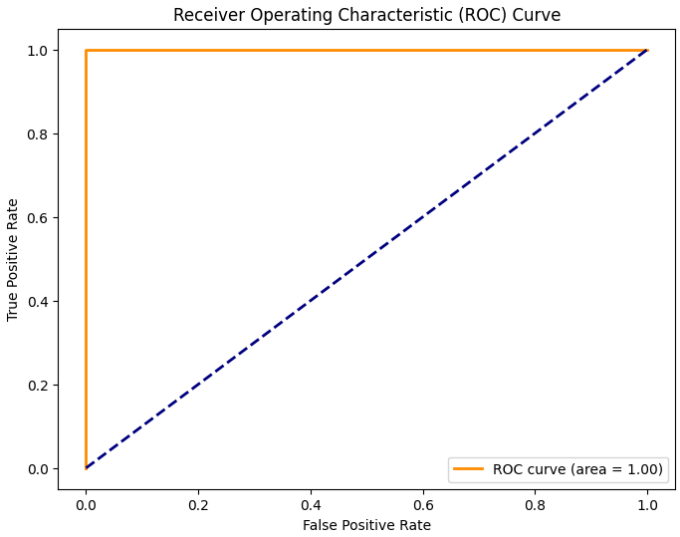


Figure 11: Model evaluation using ROC and AUC

AUC: 0.9999999999999999

Phase 4

8 Feature Importance and Reduction

The chosen dataset has 9 input features. To study the importance of features and iteratively remove them, I have trained models for each input feature calculated their accuracy against validation sets, and plotted these validation accuracies in a bar graph. The validation accuracy determines the importance of the input feature.

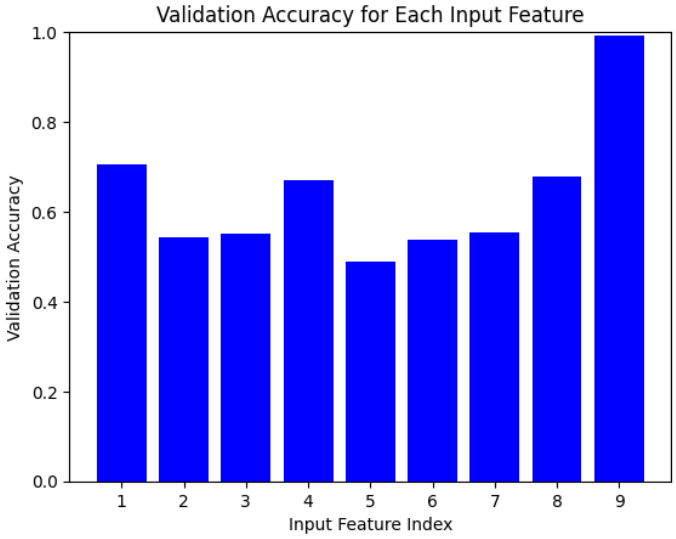


Figure 12: Validation accuracy of Input Features

As we can see in Figure 11, input feature 5 i.e. 'Superplasticizer' seems to be an unimportant feature when compared to others. After removing the input feature 'Superplasticizer' there was an improvement in the accuracy of the model prediction. The table shows the performance improvement:

Feature Removed	Training Acc	Validation Acc
Superplasticizer	1.0	1.0

Table 9: 1 input feature removed

Removing 2 input features: 'Superplasticizer' and 'Coarse Aggregate'

Feature Removed	Training Acc	Validation Acc
Superplasticizer	1.0	1.0
Coarse Aggregate	1.0	1.0

Table 10: 2 input features removed

9 Challenges faced

During the course of the project, I faced a couple of challenges:

- Choosing a dataset was tricky. The dataset had to be balanced with decent usability. I had to read through quite a few datasets to be able to find one decent enough to work with.
- Choosing a data visualization technique to showcase the relation between the input feature and output variable.

10 Conclusion

In conclusion, in this project, I have created a Neural Network Model to predict the strength of concrete based on certain input features. The data was thoroughly explored and analyzed. Four data models were tested and the most accurate one was used to train and for predictions. I also coded a custom function for predictions which has also performed well. At last, the importance of Input Features was determined and the model's accuracy was measured by iteratively removing the least important ones.

11 References

- https://christophm.github.io/interpretable-ml-book/
- https://www.youtube.com/watch?v=4jRBRDbJemM
- https://developers.google.com/machine-learning/data-prep/transform/normalization