Machine Learning Project (Concrete Compressive Strength)

Carlos Gutierrez PID: 6248381

CNT 4153 - Machine Learning in ECE

Professor Dr. Himanshu Upadhyay

I - Problem

The problem addressed throughout this project's completion is a regression problem revolving around the prediction of the compressive strength of concrete based on various input features. This is important because concrete is one of the key materials in the infrastructure of today's world due to its durability and structural integrity in buildings, so having a way to predict its strength would lead to the optimization of the material, reduction of cost, and the improvement of quality.

II - Motivation

The motivation behind the project, as it was briefly explained in the problem section, was to develop a machine learning algorithm using the knowledge learned throughout the course that can accurately predict concrete compressive strength, so engineers and construction professionals can make informed decisions about ways to develop and form concrete depending on their goal, leading to more efficient and sustainable construction practices.

III - Dataset Information

The dataset used for the completion of this project originates from the website “UC Irvine Machine Learning Repository”, and in total it contains 1030 instances and 9 total features including the target variable: Cement,kg/m^3, Blast Furnace Slag,kg/m^3, Fly Ash,kg/m^3, Water,kg/m^3, Superplasticizer,kg/m^3, Coarse Aggregate,kg/m^3, Fine Aggregate,kg/m^3, Age,day, Concrete compressive strength,MPa. The features in the dataset provide a comprehensive view of how different factors influence concrete strength as well as the “Concrete comprehensive strength” itself.

IV - Feature Processing and Feature Engineering

In the feature processing and engineering phase, I conducted an initial exploration of the dataset, analyzing data types, null values, distribution, potential feature adjustments, and outlier presence. Through functions like "data.info()" and visualization with matplotlib, I determined that all features were numerical, no null values were present and explored the distribution using histograms. Considering feature adjustments, I refrained from adding or removing features due to a clear problem understanding. Outliers were identified and removed using a boxplot and IQR method. Additionally, feature normalization was applied to ensure consistency in scale, preventing bias towards any single feature during model training. Finally, the dataset was split into training and testing sets using an 80-20 split.

V - Machine Learning Model Development

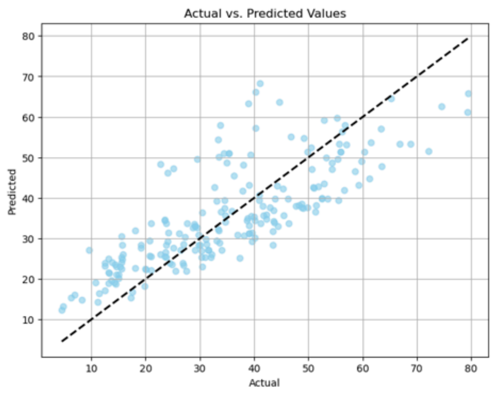
The two machine learning algorithms that were selected for this task were Linear Regression and Random Forest. Both algorithms were trained using the normalized and split sets.

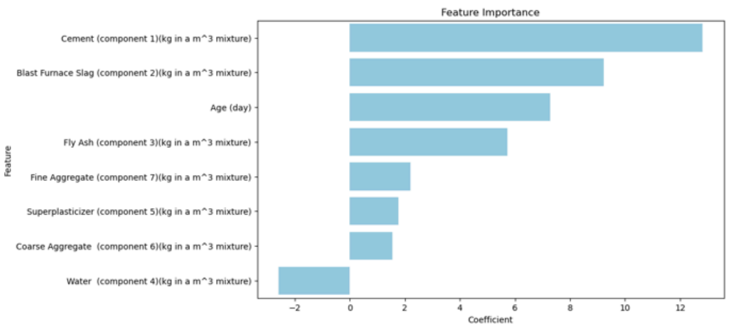
VI - Prediction/Result

The performance of each algorithm was evaluated using metrics such as Mean Squared Error (MSE) and R-squared score. Predictions were made on the test set, and the results were compared between the two algorithms. Visualizations, including scatter plots of actual vs. predicted values and feature importance plots, provided insights into the models' behavior and contributed to the evaluation process.

VII - Evaluating the result/metrics (Include graphs)

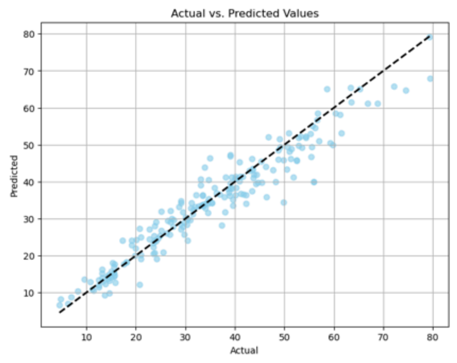
As said before both algorithm's performance was judged by comparing their MSE and R-squared scores obtained from both the test set and cross-validation, this helped me get an insight into the algorithm's performance over various subsets of the data. For the Linear regression algorithm, the results showed an MSE of 94.04 and an R-squared score of 0.62 over the regular train split, with an "Actual vs predicted Values" graph, As well as a feature importance graph where we can observe the value that each feature brings to the concrete comprehensive strength:

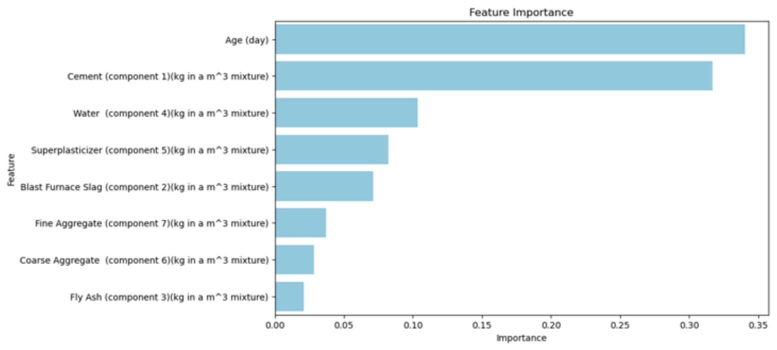




Also, trying to improve the algorithm I used cross-validation and random search which gave me the following MSE respectively: 114.43 and 94.04.

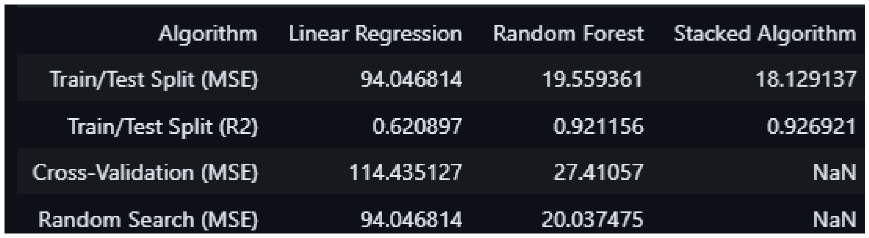
For the Random forest algorithm, the results showed an MSE of 19.55 and an R-squared score of 0.92 over the regular train split, with an "Actual vs predicted Values" graph. As well as a feature importance graph where we can observe the value that each feature brings to the concrete comprehensive strength:





Also, trying to improve the algorithm I used cross-validation and random search which gave me the following MSE respectively: 27.41 and 20.03.

Finally using "Model Staking" I combined both algorithms to improve the prediction of my model further and I obtained the following results: An MSE of 18.12 and a R-squared of 0.92.



Algorithm	Linear Regression	Random Forest	Stacked Algorithm
Train/Test Split (MSE)	94.046814	19.559361	18.129137
Train/Test Split (R2)	0.620897	0.921156	0.926921
Cross-Validation (MSE)	114.435127	27.41057	NaN
Random Search (MSE)	94.046814	20.037475	NaN

VIII - Conclusion

In conclusion, both Linear Regression and Random Forest algorithms demonstrated reasonable performance in predicting concrete compressive strength. However, the Random Forest model exhibited superior performance compared to Linear Regression, achieving lower MSE and higher R-squared scores. Hyperparameter tuning using random search showed a negligible improvement. Still, the model stacking of both algorithms proved to be the best model out of everything tested during the completion of this project having the lowest MSE and the highest R-squared score as shown in the results part.

IX - References

UCI Machine Learning Repository. (n.d.). https://archive.ics.uci.edu/dataset/165/concrete+compressive+strength

Lysett, T. (2022, January 1). Everything you need to know about concrete strength | COR-TUF. Cor-tuf UHPC. https://cor-tuf.com/everything-you-need-to-know-about-concrete-strength/

API reference — pandas 2.2.2 documentation. (n.d.). https://pandas.pydata.org/docs/reference/index.html

Matplotlib documentation — Matplotlib 3.8.4 documentation. (n.d.). https://matplotlib.org/stable/index.html

scikit-learn: machine learning in Python — scikit-learn 1.4.2 documentation. (n.d.). https://scikit-learn.org/stable/