CONVOLUTION-BASED DEEP LEARNING APPROACH FOR ESTIMATING COMPRESSIVE STRENGTH OF FIBER REINFORCED CONCRETE AT ELEVATED TEMPERATURE

Author(s) - CougarNetID(s)
Krishnaveni Katta- 2106830
Bhanu Teja Punna- 2103890
Rajesh Reddy Vuyyuru- 2103895
Tanmayi Pantangi- 2103888

ABSTRACT

The Fiber-reinforced concrete (FRC) shows high fire resistant and retains structural integrity even at high temperatures. Traditional approaches, on the other hand, confront considerable problems in terms of efficiency, accuracy, and safety when it comes to enhancing its mixture design and anticipating its related mechanical responses after fire exposure. A convolution-based deep learning model was developed to overcome these challenges. Two datasets were used to make comparisons of compressive strength and to show effect of temperature on compressive strength. Furthermore, the proposed model's relationship between temperature and relative compressive strength demonstrates that our proposed model can accurately predict the mechanical capabilities of FRC exposed to high temperatures. The proposed deep-learning approach is intended to serve as an accurate and adaptive property evaluation tool for mixture design optimization and compressive strength calculation researchers and engineers. Implemented models like CNN, LSTM and SVM for prediction. SVM outperformed the other ANN models, with the prediction loss of the model being low than both the CNN and LSTM models.

Index Terms- *Multiple linear regression; artificial neural networks; Compressive strength; Concrete strength prediction, SVM, CNN, LSTM*

1.INTRODUCTION

Today, artificial intelligence has become a trend all over the world. It has gained a huge attraction due to its application in the field of engineering. The application of AI is very helpful and beneficial in various sectors. One of the sectors where it has gained a lot of attention is the civil sector. Engineers are using AI in various fields to perform complex jobs. The use of AI has increased over the years.

Problem: The objective of this experiment is to estimate the compressive strength of Fiber reinforced concrete at elevated temperatures. The research aims to produce data that can be used to model the compressive strength of reinforced concrete at elevated temperature. To achieve this, an experimental setup using the confinement method has been developed, together with a data acquisition system. This setup is based on the Convolution-Based Deep Learning Approach.

Importance: One of the major tasks of the engineers is to predict the strength of the material which is being used to make the structure. This task is not easy because of several reasons. First, the material which is being used may have different properties depending upon the place where it is being used. Second, a different type of material may be used as per a particular structure's needs. The material properties change as we increase the temperature of the material.
The currently practiced methods employ destructive methods to estimate the strength of the concrete. Few non-destructive methods like rebound hammering (use of ultrasonic pulse), magnetic test and radioactive tests are existing, but are highly cost ineffective, time taking and experimental errors are inevitable and hence deep learning approaches are best for estimating the compressive strength.

During cementitious material selection procedures, there is always consideration given priority to concrete strength based primarily upon slump test results. One property of particular interest is early age compressive strength (ECS). In civil engineering design, ECS values will likely determine ultimate bearing capacity. As a result, if only a single value of ECS determined during testing is available, then additional tests would potentially reveal more pertinent properties that should be examined. Herein, several new methods are proposed to estimate these remaining unknown factors that might affect the ECS behavior. Specifically, temperature influences fibres' stress–strain response embedded into hardened cement mortars.

Fiber-reinforced concrete (FRP) is a composite material that has revolutionized the construction industry in recent times. This composite material is characterized by high strength and high stiffness. As per the American Concrete Institute standards, concrete can be classified as concrete that is reinforced by steel bars embedded in a concrete matrix. The matrix is mainly used to add strength to the concrete. FRP is a composite material that is characterized by a high modulus of elasticity, high tensile strength, and high tensile modulus. The composite material is mostly used for civil engineering's, such as bridge construction. FRP is mainly composed of fibers, matrix, and reinforcement. It is a method for determining the compressive strength of reinforced concrete at extreme temperatures.

Overview of results: The compressive strength of concrete at extreme temperatures is used to simulate the strength of the concrete mixture. In the form of a finite element code, the model is used to compute the compressive strength of the concrete mixture as a function of the size distribution ratio of the aggregate. The strength of the new concrete mixture is determined as a function of the size distribution ratio of the aggregate. At increased temperatures, the compressive strength of the concrete mixture is evaluated by combining the compressive strength of the concrete mixture and the contribution of new concrete to the material's strength. The approach is used on a collection of fiber reinforced concrete mixes with varying particle size distributions. Hence currently employed Convolution-Based Deep Learning Approaches are non-destructive and cost effective methods when compared to destructive approaches.

2. RELATED WORK

1.An approach for predicting the compressive strength of cement-based materials exposed to sulphate attack Authors: Huaicheng Chen, Chunxiang Qian, Chengyao Liang, Wence Kang. Published: January 18, 2018.

In this paper, compressive strength of cement material Mortar that is exposed to sulphate attack is estimated using Support Vector Machine (SVM). A dataset containing 638 samples of data is used to establish SVM model for predicting the compressive strength of mortars. An accelerated corrosive test is conducted to collect the strength data. MAE, RMSE and Mean absolute percentage error are the performance metrics used in this paper. Sensitivity analysis is used to calculate the main factors affecting the predicted compressive strength. Main factors that were used for the model are exposure time, water-cement ratio and sulphate ions.
Reference:https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0191370

2.Using non-destructive tests for estimating uniaxial compressive strength and static Young's modulus of carbonate rocks via some modelling techniques Authors: Shekoufeh Aboutaleb, Mahmoud Behnia, Raheb Bagherpour & Behzad Bluekian Published: 11 April 2017
Estimation of elastic properties such as uniaxial compressive strength (UCS) and static Young's modulus (Es) is published in this paper. A UCS test is applied to determine them but this approach is destructive, expensive, time taking and requires quality samples. The main intension of this experiment is to find the relation between UCS and Es. Different limestone samples from five different dam sites of Iran are collected for this experiment. This paper concluded that SVM model is desirable and advantageous because of its faster run time.
Reference: https://rdcu.be/cNzu8

3.Concrete compressive strength prediction using non-destructive tests through response surface methodology Authors: Ali Poorarbabi, Mohammadreza Ghasemi, Mehdi Azhdary Moghaddam Published: 17 February 2020

In this paper, non-destructive methods for predicting concrete compressive strength through response surface methodology are used. The techniques used are ultrasonic pulse velocity, rebound number tests and Response Surface Methodology (RSM). RSM is considered as an efficient approach with more accuracy. The results from this experiment showed that ultrasonic pulse velocity is the best NDT test at beginning ages when used in RSM process.

Reference:http://creativecommons.org/licenses/by-ncnd/4.0/

Comparison:

Fiber-reinforced concrete compressive strength is regarded as the essential indication of concrete quality when it comes to the safety management of existing concrete structures (Ju et al., 2017). Because of how complicated some degradation mechanisms are involved and the numerous factors affecting each, measuring and estimating concrete compressive strength remains a difficult subject. Test methods that involve the destruction of concrete are, to some degree, reliable for determining the compressive strength of fiber-reinforced concrete; however, it is not possible to assess the concrete characteristics without destroying the concrete structure (Steenbergen & Vervuurt, 2012). As a result, non-destructive testing methods provide an appealing alternative.
The rebound hammering, use of ultrasonic pulse, testing the penetration resistance, magnetic test, and radioactive test are all existing non-destructive test procedures. The use of ultrasonic pulse velocity, rebound hammer, and a method that combines both hammering and ultrasonic pulse, are the most generally used non-destructive testing procedures, owing to their simplicity and efficacy (Trtnik & Turk, 2009)

but these methods are cost ineffective, time consuming and errors are inevitable.New estimating techniques utilizing machine learning algorithms such as Long Short Term Memory, CNN, and Support Vector Machine (SVM) to increase their accuracy and dependability are developed in this project. However, all of these approaches need costly tools and equipment upkeep and educated and skilled workers with a high level of efficiency.

3. DATA

In this project we are working on two datasets. One dataset has 9 columns without temperature attribute. This has 1030 rows. Other dataset has 326 rows and 20 columns with temperature attribute. Target Variable is compressive strength in both the datasets. Each data set has multiple input and one output variable. Some common input variables are Sand, Aggregate, silica fumes, Slag, water, cement and so on. The output variable is Compressive strength. The data has both numerical and categorical data. observing target variable with respect to temperature. Dataset is collected from GitHub the link is provided at the end of the paper. We have preprocessed the data using one hot encoding. One hot encoding makes the training data more useful and scaling easier. Also implemented scalar fit that transforms data such that its distribution will have a mean value 0 and standard deviation of 1 and also implemented MinMaxScaler as it preserves the shape of the original distribution. The dataset selected consists of various fields that are used in the training, testing, and prediction process. These fields include Temperature, Specimens, Heating rate, Maintenance, and Cement, as shown by the image below.

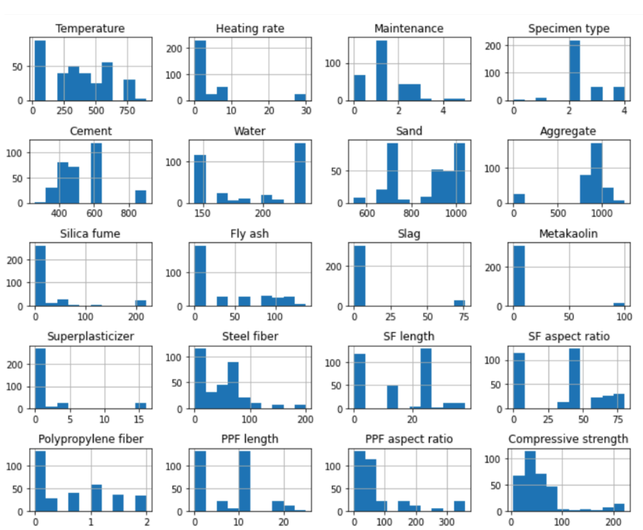

Fig shows input and output features of the dataset

The data is splitted into test set and train set. Part of analyzing data mining algorithms involves separating data into training and testing sets. It is critical to separate data into training and testing sets when analyzing data mining methods.When a data set is divided into a training set and a testing set, the majority of the data is used for training and a smaller piece of the data is utilized for testing.This method contributes to the accuracy of data models and data-driven operations.Models are trained and developed using the training data set.Training sets are frequently used to estimate various parameters or to evaluate model performance.After training, the testing data set is used to ensure that the final model functions properly, the training and test data are compared. Here in the proposed model we splitted the test data as 20% and the train data as 80%.

The other dataset used has 1023 rows and 9 columns.

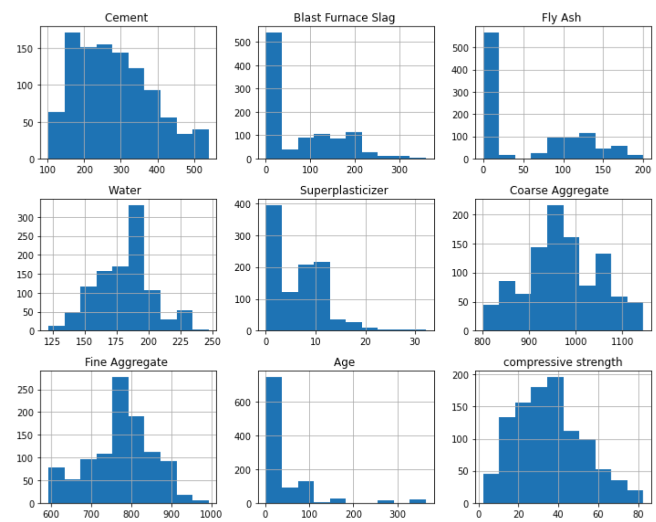

Fig shows input and output features of the dataset

4. METHODS

We implemented a total of three models to predict the compressive strength. Their result is compared in the form of mean absolute error, and the comparison plot is made in the form of a line graph. The three models are as follows:

CNN Model

Convolutional neural nets take convolutions, a type of artificial intelligence model introduced in recent times that shows promising results in solving problems, including object recognition and classification, and feature learning. In particular, these deep Convolutional Neural Networks (CNN) models yield better performance over traditional feed-forward and back-propagation networks due to their capability of capturing more abstract features at a low cost (Ferreira et al. 2018). Researchers around the globe have applied this kind of technology-based technique successfully into the area of visual image analysis and pattern classification.

Long-Short Term Memory (LSTM)

The LSTM introduced several innovations into NLP tasks, including sequence modeling and learning-based methods. In particular, LSTMs can learn sequential models through regency weighting, allowing more accurate predictions of the next input item given any past history. This enables the prediction of long sequences (many words) and thus makes applications such as machine translation possible.

Support Vector Machine (SVM)

TrueType or Postscript Type 1 fonts are preferred.

The first paragraph in each section should not be indented, but all following paragraphs within the section should be indented as these paragraphs demonstrate.

5. EXPERIMENTS

We first read the dataset with 326 rows and 20 columns and then visualized the data to check for missing values. To understand the relation between each attribute different plots were made like correlation plot, histogram plot, box plot and pair plot. The above-mentioned procedure is shown in the images below

	Temperature	Heating rate	Maintenance	Specimen type	Cement	Water	Sand	Aggregate	Silica fume	Fly ash	Slag	Metakaolin	Superplasticizer	Steel fiber	SF length
0	800	6.0	1.0	1	262.5	210.0	588.8	1251.2	0.0	87.5	0	0	0.00	0.0	0.0
1	800	6.0	1.0	1	357.5	176.0	542.7	1153.3	55.0	137.5	0	0	1.80	0.0	0.0
2	800	6.0	1.0	1	357.5	176.0	542.7	1153.3	55.0	137.5	0	0	1.80	78.0	20.0
3	800	6.0	1.0	1	357.5	176.0	542.7	1153.3	55.0	137.5	0	0	1.80	0.0	20.0
4	25	6.0	0.0	1	262.5	210.0	588.8	1251.2	0.0	87.5	0	0	0.00	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
321	200	6.0	2.5	2	500.0	150.0	980.0	780.0	0.0	0.0	0	0	1.28	0.0	0.0
322	300	6.0	2.5	2	500.0	150.0	980.0	780.0	0.0	0.0	0	0	1.28	0.0	0.0
323	400	6.0	2.5	2	500.0	150.0	980.0	780.0	0.0	0.0	0	0	1.28	0.0	0.0
324	500	6.0	2.5	2	500.0	150.0	980.0	780.0	0.0	0.0	0	0	1.28	0.0	0.0
325	600	6.0	2.5	2	500.0	150.0	980.0	780.0	0.0	0.0	0	0	1.28	0.0	0.0

326 rows × 20 columns

Fig loading the dataset

	Temperature	Heating rate	Maintenance	Specimen type	Cement	Water	Sand	Aggregate	Silica fume	Fly ash	Slag	Metakaolin
count	326.000000	326.000000	326.000000	326.000000	326.000000	326.000000	326.000000	326.000000	326.000000	326.000000	326.000000	326.000000
mean	347.607362	4.751534	1.233129	2.411043	533.556942	191.552129	874.238007	857.134920	24.557301	36.490675	5.595092	4.601227
std	246.130258	7.366112	0.966806	0.790171	137.548126	38.675199	146.117282	259.298705	58.934546	46.690163	19.877978	20.983375
min	20.000000	0.000000	0.000000	0.000000	262.500000	143.000000	542.700000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	100.000000	2.125000	1.000000	2.000000	442.500000	150.000000	710.000000	857.000000	0.000000	0.000000	0.000000	0.000000
50%	300.000000	2.500000	1.000000	2.000000	500.000000	203.280000	919.811563	904.000000	0.610000	0.000000	0.000000	0.000000
75%	600.000000	5.000000	2.000000	3.000000	605.139186	229.952891	1005.000000	950.855203	2.420000	90.770000	0.000000	0.000000
max	900.000000	30.000000	5.000000	4.000000	900.000000	235.600000	1040.000000	1251.200000	220.000000	137.500000	76.000000	100.000000

Fig statical parameters

Looking for missing data is significant since, depending on the nature, it might occasionally skew the results. Because the data came from an unrepresentative sample, our findings may not be generalizable if we don't take care of the missing values. To understand the missing values the dataset is visualized as you can see in below image there are no missing values we can proceed further.

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa6
```

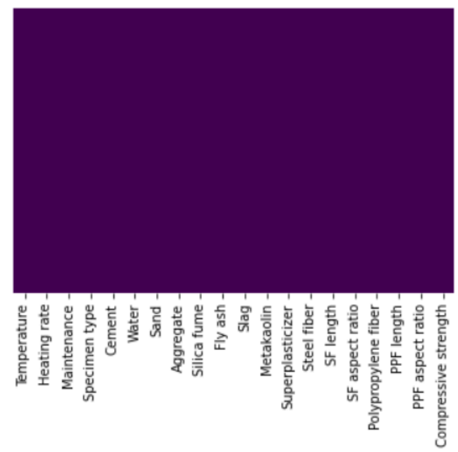

Fig checking for missing values

The yellow color shows that highly correlated columns.

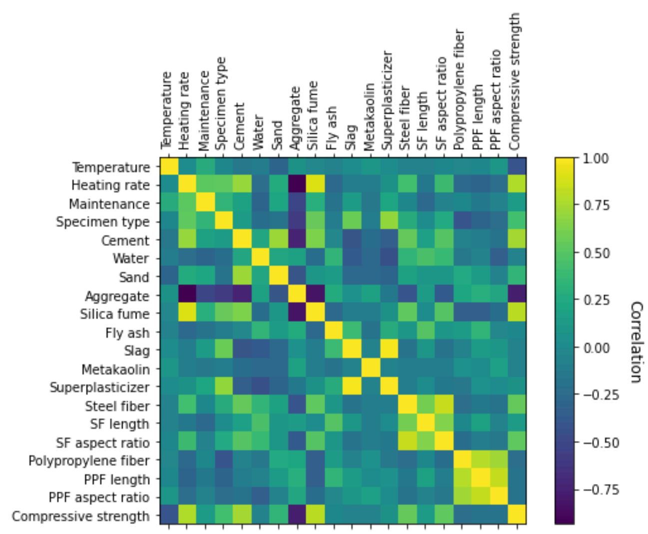

Fig correlation plot to understand the relationship

Histograms are useful for displaying the overall distribution of dataset variables. You can see roughly where the distribution's peaks are, whether it's skewed or symmetric, and whether there are any outliers.

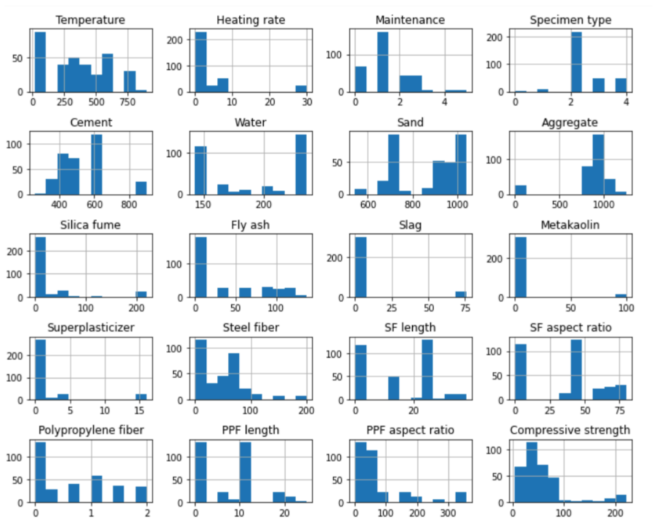

Fig Histogram plot

A pairs plot shows the distribution of single variables as well as the connections between them. Pair plots are an excellent way to spot trends for further investigation.

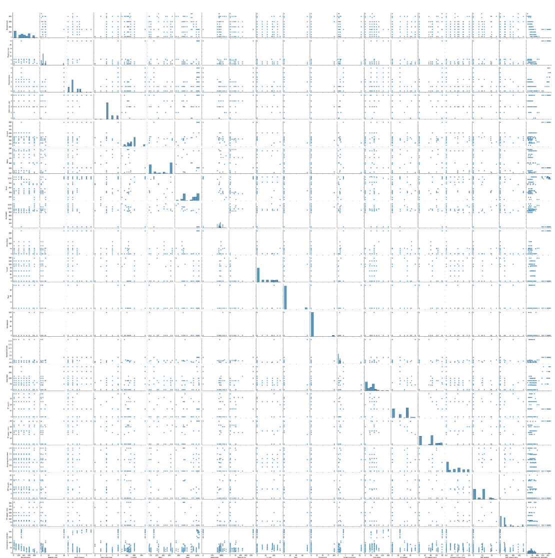

Fig pair plot

The distplot depicts the univariate distribution of data, i.e. a variable's data distribution vs the density distribution.

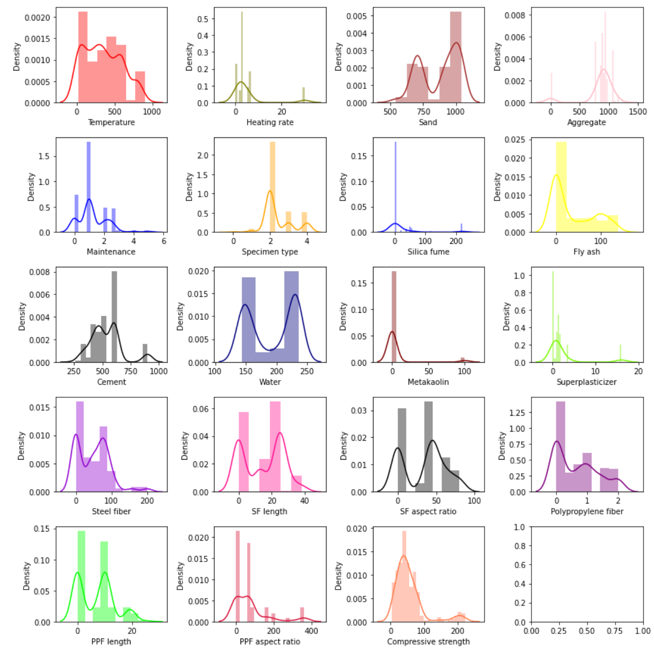

Fig distant plot

Box plots are valuable because they give a visual overview of the data that allows researchers to easily discover mean values, data set dispersion, and skewness.

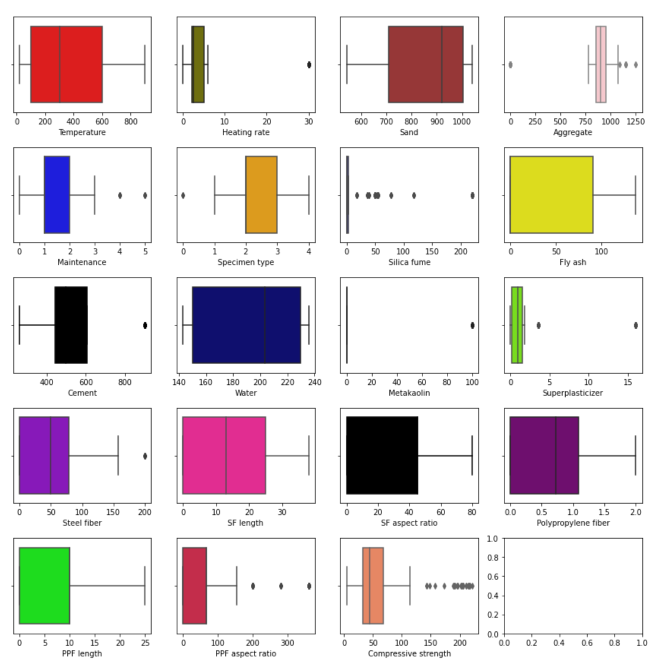

Fig box plot

We now read the other dataset with 1023 rows and 9 columns. The data was then loaded and checked for missing values. Different plots were created to comprehend the relationship between each characteristic, such as the correlation plot, histogram plot, box plot, and pair plot. The technique described above is depicted in the photos below.

	Cement	Blast Furnace Slag	Fly Ash	Water	Superplasticizer	Coarse Aggregate	Fine Aggregate	Age	compressive strength
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
...	...	...	...	...	...	...	...	...	...
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

1030 rows × 9 columns

Fig loading the dataset

	Cement	Blast Furnace Slag	Fly Ash	Water	Superplasticizer	Coarse Aggregate	Fine Aggregate	Age	compressive strength
count	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000	1030.000000
mean	281.165631	73.895485	54.187136	181.566359	6.203112	972.918592	773.578883	45.662136	35.817836
std	104.507142	86.279104	63.996469	21.355567	5.973492	77.753818	80.175427	63.169912	16.705679
min	102.000000	0.000000	0.000000	121.750000	0.000000	801.000000	594.000000	1.000000	2.331808
25%	192.375000	0.000000	0.000000	164.900000	0.000000	932.000000	730.950000	7.000000	23.707115
50%	272.900000	22.000000	0.000000	185.000000	6.350000	968.000000	779.510000	28.000000	34.442774
75%	350.000000	142.950000	118.270000	192.000000	10.160000	1029.400000	824.000000	56.000000	46.136287
max	540.000000	359.400000	200.100000	247.000000	32.200000	1145.000000	992.600000	365.000000	82.599225

Fig shows statical parameters

Looking for missing data is important since it can occasionally affect the findings depending on its type. Because the data originated from an unrepresentative sample, our findings may not be generalizable if the missing values are not taken into account. To comprehend the missing values, the dataset is represented, as seen in the figure below. Because there are no missing values, we may proceed.

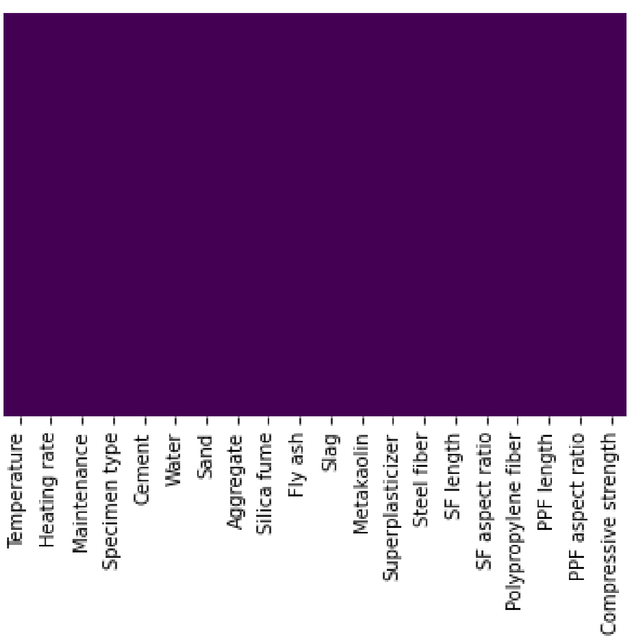

Fig there are no missing values

The brown color shows that highly correlated attributes.

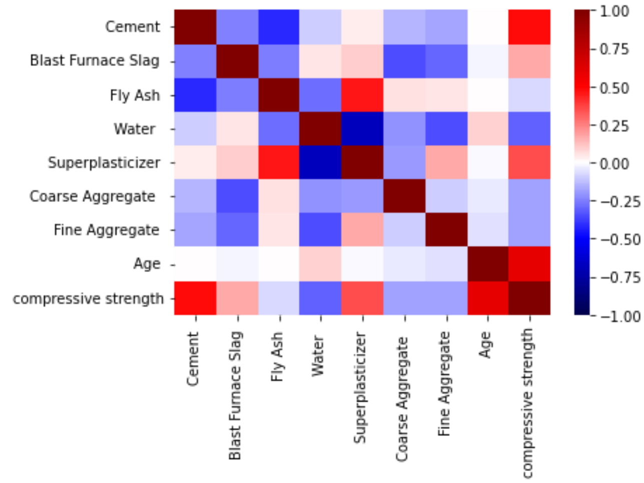

Fig correlation plot to understand the relationship

Histograms are useful for displaying the overall distribution of dataset variables. It displays the distribution's peaks, determines if the distribution is skewed or symmetric, and checks for outliers. Histograms are useful for displaying the overall distribution of dataset variables. It displays the distribution's peaks, determines if the distribution is skewed or symmetric, and checks for outliers.

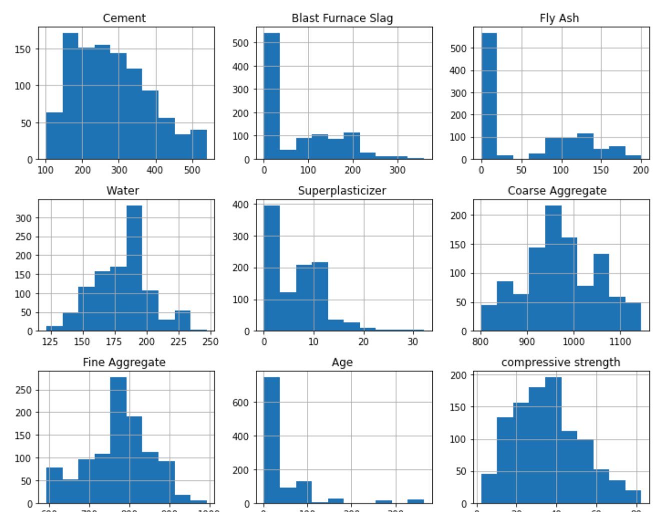

Fig Histogram plot

A pairs plot depicts the distribution of individual variables as well as the relationships between them. Pair plots are a great technique to identify tendencies for future exploration.

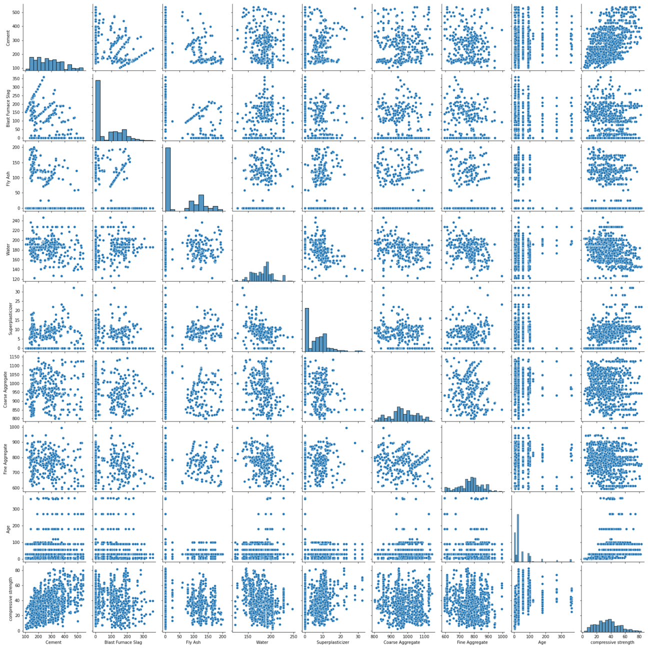

Fig pair plot

The distplot plots a variable's univariate distribution against the density distribution.

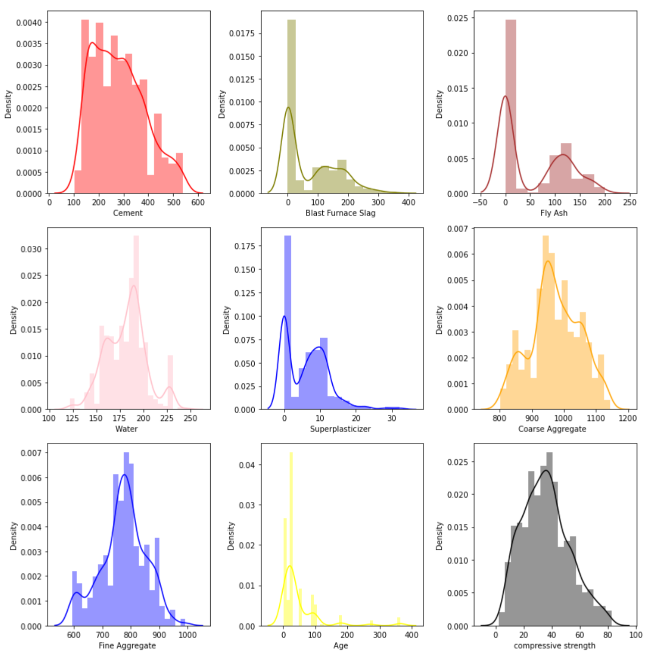

Fig dist plot

Box plots are valuable because they give a visual overview of the data, allowing researchers to easily determine mean values, data set dispersion, and skewness.

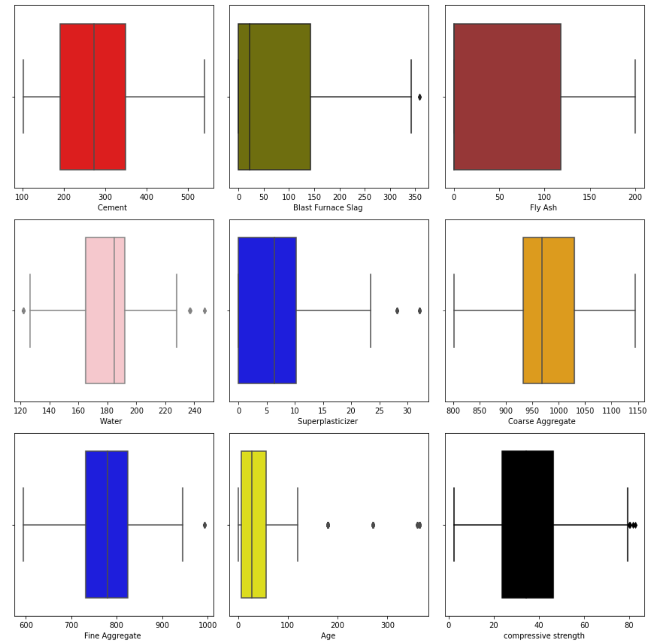

Fig box plot

PyCaret has been used to create and analyze various regression models for two of the data sets. Here we will see how to import data from PyCaret, creating a model performing cross-validation followed by tuning and evaluation of the regression models. We will declare a final model which suits best for two of the datasets. This requires python versions which are more than 3.x. First, we choose the dataset with a temperature variable in it. We have named the first dataset as 'new'. PyCaret has a setup function which needs to be declared with data and target variable and session id. Here session id is optional but we try to put it in as '123'. The target variable is "Compressive Strength". As we run the setup we get the Description and value for the dataset. As we see in the picture the original data has dimensions of (326,20). We do not have any missing values in the dataset. The dataset has 14 Numeric Features and 5 Categorical Features. The training and testing data has been split in 80:20 and the Fold Generator is taken as KFold with the optimal number of k-folds 10.

	Description	Value
0	session_id	123
1	Target	Compressive strength
2	Original Data	(326, 20)
3	Missing Values	False
4	Numeric Features	14
5	Categorical Features	5
6	Ordinal Features	False
7	High Cardinality Features	False
8	High Cardinality Method	None
9	Transformed Train Set	(228, 40)
10	Transformed Test Set	(98, 40)
11	Shuffle Train-Test	True
12	Stratify Train-Test	False
13	Fold Generator	KFold
14	Fold Number	10

Fig: Description of the data

The imputation type being used is simple and the categorical imputer is a constant. We will use the least_frequent values for the handling of unknown categoricals. We are not implementing PCA or any transformation methods, so these values are given as false. PyCaret automatically by default removes the perfect collinearity. Now, let us dive into creating a model. PyCaret has a function to create a model called “create_model()”. This function prints the output of a score grid that shows us Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R2, RMSLE and MAPE by fold values. There are 25 regressors available in the model library of PyCaret. We create SVM and AdaBoost Regressor. After creating the model we can see that C=1, degree=3, kernel=’rbf’.

```
print(svm)

SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma='scale',
    kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

Fig: SVM parameters

For all the 10 fold the model calculated the mean of the folds and provided us with MAE, MSE, RMSE, R2, RMSLE and MAPE. Here in our case with the first dataset which includes the temperature the MAE was found to be 25.9027. The highest MAE was found in fold 0 (here it is fold 1) which is noted as 36.8501 and the least was recorded as 18.9087 in the last fold. The MSE is given as 1832.1182 with R2 score 0.0592 for the SVM model.

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	36.8501	3044.0359	55.1728	0.0657	0.7776	0.8503
1	30.6222	2826.0675	53.1608	0.0796	0.6430	0.5751
2	21.1322	1208.4388	34.7626	0.1081	0.6318	0.6832
3	22.4314	1085.9463	32.9537	0.1475	0.6427	0.7409
4	22.8270	1363.5838	36.9267	0.1167	0.6077	0.6253
5	26.7343	1415.3094	37.6206	0.0302	0.7181	0.8420
6	25.6275	2106.8623	45.9006	0.1270	0.6562	0.6731
7	26.2749	2451.0344	49.5079	0.0137	0.5147	0.3363
8	27.6191	2351.8706	48.4961	0.0990	0.6324	0.6037
9	18.9087	468.0326	21.6341	-0.1954	0.4344	0.3961
Mean	25.9027	1832.1182	41.6136	0.0592	0.6259	0.6326

Fig SVM Model for dataset with temperature

For AdaBoost, The highest MAE was found in fold 0 (here it is fold 1) which is noted as 14.5099 and the least was recorded as 7.4235 in the 8th fold. The MAE is recorded as 9.9137. The MSE is given as 167.4073 with R2 score 0.8852.

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	14.5099	486.3380	22.0531	0.8507	0.5006	0.4386
1	8.3170	91.8283	9.5827	0.9701	0.2830	0.2532
2	9.2880	119.1170	10.9141	0.9121	0.3866	0.3777
3	11.4378	220.0410	14.8338	0.8273	0.5001	0.5360
4	8.6497	102.4396	10.1212	0.9336	0.3156	0.2958
5	12.4339	180.6989	13.4424	0.8762	0.4465	0.4761
6	7.9931	95.7201	9.7837	0.9603	0.3253	0.3046
7	7.4235	88.4331	9.4039	0.9644	0.1946	0.1537
8	9.6493	136.4346	11.6805	0.9477	0.3261	0.2883
9	9.4345	153.0229	12.3702	0.6092	0.2586	0.2025
Mean	9.9137	167.4073	12.4186	0.8852	0.3537	0.3326
Std	2.1102	113.7853	3.6313	0.1034	0.0975	0.1160

Fig: AdaBoost Model for dataset with temperature

PyCaret has a function compare_models() that is used to evaluate performance of all the models. This function trains the models using k fold cross validation for the evaluation of the metrics. The k folds are taken as default 10. We can also sort this order by changing it to the desired parameter. Here we are not going to implement all the models but few of them. Linear regression, knn, random forest, Decision trees, Extreme Gradient Boosting, Gradient Boosting Regressor, Ridge Regressor. AdaBoost, Lasso and Support Vector Machine. With 10 folds the values for the models are given in the figure. The highlighted values are the best values for the respective parameter. Here we can see that xgboost is the best overall model with MAE=4.7374, MSE=77.5067, RMSE=7.5404, R2= 0.9466.

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
xgboost	Extreme Gradient Boosting	4.7374	77.5067	7.5404	0.9466	0.1957	0.1440	0.0730
rf	Random Forest Regressor	5.8692	86.7588	8.3579	0.9427	0.2129	0.1700	0.0910
gbr	Gradient Boosting Regressor	5.7862	96.6783	8.6591	0.9305	0.2224	0.1684	0.0240
dt	Decision Tree Regressor	6.1552	97.8545	9.3085	0.9277	0.2606	0.1885	0.0070
ridge	Ridge Regression	8.2802	133.1305	11.0811	0.8911	0.3083	0.2179	0.0060
ada	AdaBoost Regressor	9.9137	167.4073	12.4186	0.8852	0.3537	0.3326	0.0360
lr	Linear Regression	8.2791	172.8380	12.1709	0.8577	0.2943	0.2346	0.0070
lasso	Lasso Regression	13.3517	299.4939	17.1409	0.7922	0.4318	0.3990	0.0070
knn	K Neighbors Regressor	13.4870	341.5497	17.8497	0.7639	0.4398	0.4059	0.0100
svm	Support Vector Regression	25.9027	1832.1182	41.6136	0.0592	0.6259	0.6326	0.0080

Fig: Comparing all the desired models

We now choose the dataset without a temperature variable in it. We have named the second dataset as 'new2'. PyCaret has a setup function which needs to be declared with data and target variable and session id. Here session id is optional but we try to put it in as '124'. The target variable is "compressive Strength" (we are predicting the same variable in both of our datasets). As we run the setup we get the Description and value for the dataset. As we see in the picture the original data has dimensions of (1030,09). We do not have any missing values in the dataset. The dataset has 7 Numeric Features and 1 Categorical Features. The training and testing data has been split in 80:20, Trainset (720,21) and test set (310,21) and the Fold Generator is taken as KFold with the optimal number of k-folds 10.

	Description	Value
0	session_id	124
1	Target	compressive strength
2	Original Data	(1030, 9)
3	Missing Values	False
4	Numeric Features	7
5	Categorical Features	1
6	Ordinal Features	False
7	High Cardinality Features	False
8	High Cardinality Method	None
9	Transformed Train Set	(720, 21)
10	Transformed Test Set	(310, 21)
11	Shuffle Train-Test	True
12	Stratify Train-Test	False
13	Fold Generator	KFold
14	Fold Number	10

Fig: Description of the dataset without the temperature.

Here we can see that in the training and the testing set the parameters have been increased, that is, the one categorical feature has been encoded automatically by the PyCaret using One-Hot Encoding. In this case the features have an inherent order to their levels and it would be more appropriate to encode them. The imputation type being used is simple, Numeric imputer as mean and the categorical imputer is a constant. We will use the least_frequent values for the handling of unknown categoricals same as in the previous dataset.. We are not implementing PCA or any transformation methods, so these values are given as false. PyCaret automatically by default removes the perfect collinearity. Now, let us dive into creating a model. PyCaret has a function to create a model called "create_model()". This function prints the output of a score grid that shows us Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R2, RMSLE and MAPE by fold values. There are 25 regressors available in the model library of PyCaret. We create SVM and AdaBoost Regressor. After creating the model we can see that C=1, degree=3, kernel='rbf'.

```
print(svm2)

SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma='scale',
    kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

Fig: SVM parameters for second dataset.

For all the 10 fold the model calculated the mean of the folds and provided us with MAE, MSE, RMSE, R2, RMSLE and MAPE. Here with the dataset without the temperature the MAE was found to be 12.6307. The highest MAE was found in fold 9 (here it is fold 10) which is noted as 14.2865 and the least was recorded as 11.4104 in the 5tht fold. The MSE is given as 245.0271 with R2 score 0.1518 for the SVM model.

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	12.3859	253.1353	15.9102	0.1575	0.5117	0.5832
1	11.8233	213.8881	14.6249	0.1507	0.5269	0.5838
2	12.9598	242.7974	15.5820	0.1484	0.5474	0.6317
3	12.5366	261.1630	16.1605	0.1441	0.5261	0.5793
4	11.4104	184.9949	13.6013	0.1398	0.4481	0.4640
5	13.7267	276.5659	16.6303	0.1367	0.4975	0.5113
6	12.6910	256.4694	16.0147	0.1443	0.5197	0.5552
7	12.3521	234.5804	15.3160	0.1560	0.4261	0.4131
8	12.1348	234.8638	15.3253	0.1535	0.4876	0.5060
9	14.2865	291.8130	17.0825	0.1866	0.5455	0.6195
Mean	12.6307	245.0271	15.6248	0.1518	0.5037	0.5447
Std	0.8117	29.0371	0.9454	0.0133	0.0380	0.0663

Fig SVM Model for dataset without temperature

For AdaBoost, The highest MAE was found in the last fold which is noted as 8.5318 and the least was recorded as 6.3258 in the 5th fold. The MAE is recorded as 7.3257. The MSE is given as 78.7120 with R2 score 0.7272.

	MAE	MSE	RMSE	R2	RMSLE	MAPE
Fold						
0	7.4994	91.6419	9.5730	0.6950	0.3600	0.3565
1	8.1772	91.7387	9.5780	0.6357	0.3816	0.3861
2	7.4824	75.5554	8.6923	0.7350	0.3475	0.3467
3	7.3625	78.2002	8.8431	0.7437	0.3254	0.3144
4	6.3258	57.0879	7.5557	0.7346	0.2808	0.2559
5	7.3047	82.3022	9.0721	0.7431	0.3336	0.3075
6	6.4697	63.2913	7.9556	0.7888	0.3445	0.3196
7	6.5208	60.7711	7.7956	0.7814	0.2605	0.2380
8	7.5830	80.4192	8.9677	0.7101	0.3254	0.3078
9	8.5318	106.1120	10.3011	0.7042	0.3588	0.3582
Mean	7.3257	78.7120	8.8334	0.7272	0.3318	0.3191
Std	0.6843	14.6320	0.8265	0.0419	0.0349	0.0436

Fig: AdaBoost Model for dataset without temperature

PyCaret has a function compare_models() that is used to evaluate performance of all the models. This function trains the models using k fold cross validation for the evaluation of the metrics. The k folds are taken as default 10. We can also sort this order by changing it to the desired parameter. Here we are not going to implement all the models but few of them. Linear regression, knn, random forest, Decision trees, Extreme Gradient Boosting, Gradient Boosting Regressor, Ridge Regressor. AdaBoost, Lasso and Support Vector Machine. With 10 folds the values for the models are given in the figure. The highlighted values are the best values for the respective parameter. Here we can see that xgboost is the best overall model with MAE=3.4097, MSE=24.6485, RMSE=4.9387, R2= 0.9133. Here we can see in both the dataset models the R2 score for the dataset with the temperature is the highest.

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
xgboost	Extreme Gradient Boosting	3.4097	24.6485	4.9387	0.9133	0.1621	0.1221	0.0910
rf	Random Forest Regressor	3.8868	29.8753	5.4367	0.8953	0.1753	0.1382	0.1270
gbr	Gradient Boosting Regressor	4.3719	32.5450	5.6803	0.8867	0.1770	0.1489	0.0390
lr	Linear Regression	5.4881	49.6372	6.9968	0.8252	0.2445	0.1924	0.0080
ridge	Ridge Regression	5.5114	50.6614	7.0722	0.8220	0.2420	0.1944	0.0070
dt	Decision Tree Regressor	5.2703	61.3022	7.6751	0.7846	0.2470	0.1788	0.0080
ada	AdaBoost Regressor	7.3257	78.7120	8.8334	0.7272	0.3318	0.3191	0.0430
lasso	Lasso Regression	8.0408	101.4843	10.0574	0.6437	0.3208	0.3011	0.0070
knn	K Neighbors Regressor	9.4478	139.7882	11.8016	0.5069	0.3943	0.3803	0.0090
svm	Support Vector Regression	12.6307	245.0271	15.6248	0.1518	0.5037	0.5447	0.0130

Fig: Comparing all the desired models with a dataset without temperature parameter.

For the experiment, the models were implemented in python and trained individually using the dataset.The training and testing make use of a batch size of 20, an input dimension of 20, and an epoch of 100.

CNN, SVM and LSTM model implementation:

We have used kernel size to be 4 for the CNN and we already know that input dimensions are 19(20). The activation function used is "relu". This is a rectified linear activation function which will output the input directly if it is a positive input else it outputs zero.

```
CNN Model
Epoch 1/100
13/13 [==============================] - 1s 31ms/step - loss: 55.5528 - val_loss: 53.0730
Epoch 2/100
13/13 [==============================] - 0s 8ms/step - loss: 52.4880 - val_loss: 47.1558
Epoch 3/100
13/13 [==============================] - 0s 9ms/step - loss: 41.7685 - val_loss: 30.6705
Epoch 4/100
13/13 [==============================] - 0s 7ms/step - loss: 27.4401 - val_loss: 25.4121
Epoch 5/100
13/13 [==============================] - 0s 8ms/step - loss: 26.4311 - val_loss: 24.1618
Epoch 6/100
13/13 [==============================] - 0s 11ms/step - loss: 25.7337 - val_loss: 23.9182
Epoch 7/100
13/13 [==============================] - 0s 8ms/step - loss: 25.8849 - val_loss: 23.7668
Epoch 8/100
13/13 [==============================] - 0s 8ms/step - loss: 25.4005 - val_loss: 23.7704
Epoch 9/100
13/13 [==============================] - 0s 9ms/step - loss: 25.2884 - val_loss: 23.5212
Epoch 10/100
13/13 [==============================] - 0s 8ms/step - loss: 25.2546 - val_loss: 23.5411
Epoch 11/100
13/13 [==============================] - 0s 10ms/step - loss: 25.1318 - val_loss: 23.3248
Epoch 12/100
13/13 [==============================] - 0s 9ms/step - loss: 25.0354 - val_loss: 23.1943
Epoch 13/100
13/13 [==============================] - 0s 9ms/step - loss: 24.9842 - val_loss: 23.1440
Epoch 14/100
13/13 [==============================] - 0s 9ms/step - loss: 24.8716 - val_loss: 23.0555
Epoch 15/100
13/13 [==============================] - 0s 9ms/step - loss: 24.5722 - val_loss: 22.8276
Epoch 16/100
```

Fig: Epoch values for CNN Model.

```
SVM Model
Epoch 1/100
13/13 [==============================] - 1s 19ms/step - loss: 55.2005 - val_loss: 52.5302
Epoch 2/100
13/13 [==============================] - 0s 6ms/step - loss: 52.3705 - val_loss: 48.3974
Epoch 3/100
13/13 [==============================] - 0s 7ms/step - loss: 45.8453 - val_loss: 39.3654
Epoch 4/100
13/13 [==============================] - 0s 7ms/step - loss: 34.7768 - val_loss: 27.3673
Epoch 5/100
13/13 [==============================] - 0s 6ms/step - loss: 26.5208 - val_loss: 23.6288
Epoch 6/100
13/13 [==============================] - 0s 6ms/step - loss: 24.7906 - val_loss: 22.5822
Epoch 7/100
13/13 [==============================] - 0s 6ms/step - loss: 23.3067 - val_loss: 22.1664
Epoch 8/100
13/13 [==============================] - 0s 5ms/step - loss: 22.3375 - val_loss: 20.8524
Epoch 9/100
13/13 [==============================] - 0s 6ms/step - loss: 21.0572 - val_loss: 19.8419
Epoch 10/100
13/13 [==============================] - 0s 7ms/step - loss: 19.7643 - val_loss: 18.7309
Epoch 11/100
13/13 [==============================] - 0s 6ms/step - loss: 18.3132 - val_loss: 17.3835
Epoch 12/100
13/13 [==============================] - 0s 6ms/step - loss: 16.7940 - val_loss: 16.0355
Epoch 13/100
13/13 [==============================] - 0s 6ms/step - loss: 15.2102 - val_loss: 14.1920
Epoch 14/100
13/13 [==============================] - 0s 6ms/step - loss: 13.0251 - val_loss: 12.6285
Epoch 15/100
13/13 [==============================] - 0s 7ms/step - loss: 11.0768 - val_loss: 11.0072
```

Fig: Epoch values for SVM Model.

```
LSTM Model
Epoch 1/100
13/13 [==============================] - 2s 73ms/step - loss: 55.4221 - val_loss: 50.3027
Epoch 2/100
13/13 [==============================] - 1s 40ms/step - loss: 47.8132 - val_loss: 50.7704
Epoch 3/100
13/13 [==============================] - 1s 40ms/step - loss: 53.2856 - val_loss: 52.2204
Epoch 4/100
13/13 [==============================] - 1s 39ms/step - loss: 53.7499 - val_loss: 52.1202
Epoch 5/100
13/13 [==============================] - 1s 41ms/step - loss: 53.4992 - val_loss: 51.7445
Epoch 6/100
13/13 [==============================] - 1s 40ms/step - loss: 53.0315 - val_loss: 51.1665
Epoch 7/100
13/13 [==============================] - 1s 40ms/step - loss: 52.1678 - val_loss: 49.6881
Epoch 8/100
13/13 [==============================] - 1s 43ms/step - loss: 46.7799 - val_loss: 28.7724
Epoch 9/100
13/13 [==============================] - 1s 42ms/step - loss: 34.6476 - val_loss: 37.1448
Epoch 10/100
13/13 [==============================] - 1s 42ms/step - loss: 29.7954 - val_loss: 31.7352
Epoch 11/100
13/13 [==============================] - 1s 41ms/step - loss: 29.1073 - val_loss: 27.5583
Epoch 12/100
13/13 [==============================] - 0s 38ms/step - loss: 28.0550 - val_loss: 23.9724
Epoch 13/100
13/13 [==============================] - 1s 39ms/step - loss: 24.6629 - val_loss: 25.7043
Epoch 14/100
13/13 [==============================] - 1s 40ms/step - loss: 25.5221 - val_loss: 23.7211
Epoch 15/100
13/13 [==============================] - 1s 39ms/step - loss: 22.2970 - val_loss: 20.7921
Epoch 16/100
13/13 [==============================] - 1s 42ms/step - loss: 23.2795 - val_loss: 18.6612
```

Fig: Epoch values for LSTM Model.

The comparison of the loss of the chosen models is shown in the image below.

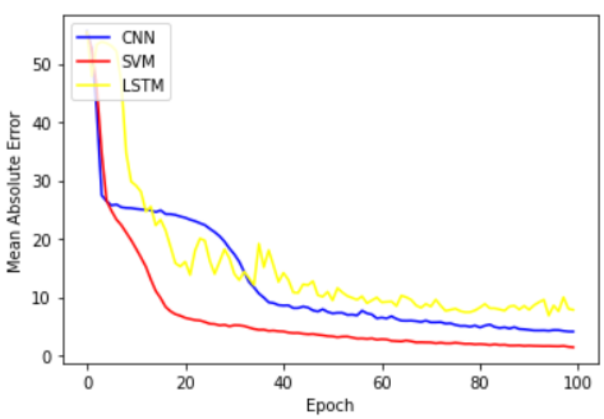

Fig: Comparison of all the models

From the figure, for the first 10 epochs we can see that the error started to be the same for all the desired models. For the LSTM we can see that there has been a lot of fluctuations in the model error but at last the LSTM has the highest error rate followed by CNN and the least error to be SVM. From the comparison, SVM appears to have the lowest loss that is epoch to MAE value tends to be lower than compared with CNN and LSTM; Generally, we think that using more complex models can result in much accurate results, but sometimes the traditional methods would work the best rather than implementing the complex models.thus, it would be the most ideal to use SVM (according to he results) in predicting compressive strength in the real-world scenario.

6. CONCLUSION

The compressive strength of fiber reinforced concrete, which is used to measure concrete quality, is an important indication in facility management. This work developed and tested the applicability of three models for calculating concrete compressive strength that employs CNN, LSTM, and SVM. The project also includes the data visualization for two types of data set where the first type contains 20 attributes with temperature playing the important role and the other one without the temperature. The implementation of different models using the pycaret with the target variable to be 'Compressive Strength'.The experimental results indicated that among all three CNN models constructed for this work, SVM outperformed the other ANN models, with the prediction loss of the model, in particular, being low than both the CNN and LSTM models. The Pycaret finding the best model for both of the datasets. We can see that XGBoost to be the best regressor with the R2 score to be 0.9466 for the dataset with temperature and R2 score for the other dataset without the temperature parameter is 0.9133. By this we can say that temperature can greatly affect the Compressive strength of the Fiber Reinforced Concrete. These findings demonstrate that though different models can be used to predict the compressive strength, in theory, not all may be utilized to predict the compressive strength of the concrete in a practical setting and that the models are capable of being improved further. The dataset presented for work is simple to apply to the actual prediction of fiber reinforced concrete due to its simple format and ease in importing the needed fields into the project models. As a result, our suggested approach offers a compelling alternative to other existing non-destructive testing methods. transform your data such that its distribution will have a mean value 0 and standard deviation of 1 MinMaxScaler preserves the shape of the original distribution.

7. REFERENCES

[1] Z. Shen, H. Zhou, A. Brooks, D. Hanna, Evolution of elastic and thermal properties of cementitious composites containing micro-size lightweight fillers after exposure to elevated temperature, Cem. Concr. Compos. 118 (2021)103931.
[2] D. Zhang, G.Y. Tan, K.H. Tan, Combined effect of flax fibers and steel fibers on spalling resistance of ultra-high performance concrete at high temperature, Cem. Concr. Compos. (2021) 104067.

[3] M. DeRousseau, J. Kasprzyk, W. Srubar III, Computational design optimization of concrete mixtures: A review, Cem. Concr. Res. 109 (2018) 42–53.
[4] T.D. Akinosho, L.O. Oyedele, M. Bilal, A.O. Ajayi, M.D. Delgado, O.O. Akinade, A.A. Ahmed, Deep learning in the construction industry: A review of present status and future innovations, J. Build. Eng. (2020) 101827.
[5] J. Yang, H. Chen, S. Hu, V. Gan, Experimental studies on the flexural behavior of steel-concrete composite beams with transverse and longitudinal hidden girders, Eng. Struct. 179 (2) (2019) 583–594.
[6] Y. Huang, V. Gan, H. Chen, J. Yang, Behavior of a two-way lightweight steel concrete composite slab voided with thin-walled core boxes towards sustainable construction, Materials 13 (18) (2020).
[7] A. Demir, İ.B. Topçu, H. Kuşan, Modeling of some properties of the crushed tile concrete exposed to elevated temperatures, Constr. Build. Mater. 25 (4) (2011)1883–1889.
[8] D. Gawin, F. Pesavento, B.A. Schrefler, What physical phenomena can be neglected when modeling concrete at high temperature? A comparative study. Part 1: Physical phenomena and mathematical model, Int. J. Solids Struct. 48(13) (2011) 1927–1944.
[9] D. Gawin, F. Pesavento, B.A. Schrefler, What physical phenomena can be neglected when modeling concrete at high temperature? A comparative study. Part 2: Comparison between models, Int. J. Solids Struct. 48 (13) (2011)1945–1961.