I. Introduction

1. Description/formulation of the problem

I use the Concrete Compressive Strength dataset to perform linear regression. The input includes first eight columns (Cement, Blast Furnace Slag, Fly Ash, Water, Superplasticizer, Coarse Aggregate, Fine Aggregate, Age) as input and the nineth column as output (Concrete compressive strength)

From the given eight input, I will first try to do Uni-variate linear regression on each feature input to try to predict the outcome of the output. Then, I will do Multi-variate linear regression on all eights feature to try to predict the output. For both the Uni-variate linear regression and Multi-variate linear regression, I will use MSE as loss function/metric to measure the performance of linear regression. The lower the MSE, the better the model would be.

Finally, I will compare 9 models' performance on both train and test dataset to draw conclusions from them.

2. Details of your algorithm

a. Uni-variate linear regression

Model formula: y_{pred} = m * x + b

Loss function (MSE): L = $\frac{1}{n}\sum(y_{\text{pred}} - y)^2 = \frac{1}{n}\sum(\text{m} * \text{x} + \text{b} - \text{y})^2$

(the summation is performed on all data point)

Goal: minimize the MSE

Update rule: as the negative gradient direction is the steepest direction, I update both parameters b and m of its direction of negative gradient (with step size α):

$\text{m}_{\text{new}} = \text{m} - \alpha\frac{\Delta \text{L}}{\Delta \text{m}} = \text{m} - \alpha * \frac{2}{n}\sum \text{x} * (\text{m} * \text{x} + \text{b} - \text{y}) = \text{m} - \alpha * \sum \text{x} * (y_{\text{pred}} - \text{y})$

$\text{b}_{\text{new}} = \text{b} - \alpha\frac{\Delta \text{L}}{\Delta \text{b}} = \text{b} - \alpha * \frac{2}{n}\sum(\text{m} * \text{x} + \text{b} - \text{y}) = \text{b} - \alpha * \frac{2}{n}\sum(y_{\text{pred}} - \text{y})$

I keeps updating with certain number of iteration (10000 in my program).

b. Multi-variate linear regression

The idea is same for uni-variate linear regression, only the formula changes.

Model formula: $y_{\text{pred}} = (\vec{a} \, . \, \vec{x})$ (dot product between a and x)

$\vec{a} = (a_0, a_1, \ldots, a_n)$ – a_0 is the intercept and $\vec{x} = (1, x_1, x_2, \ldots, x_n)$ where x_i is the input feature.

Loss function (MSE): L = $\frac{1}{n}\sum(y_{\text{pred}} - y)^2 = \frac{1}{n}\sum(\vec{a} * \text{x} - \text{y})^2 = \frac{1}{n} * |X\vec{a} - \vec{y}|^2$

(X is the whole dataset's input – n * (numFeature + 1) matrix, $\vec{y}$ is the output – n * 1 matrix)

$\vec{a}_{\text{new}} = \vec{a} - \alpha\frac{\Delta \text{L}}{\Delta \vec{a}} = \text{a} - \alpha * \frac{2}{n} X^{T} * (X\vec{a} - \vec{y})$

3. Pseudo-code

a. Uni-variate linear regression

```
n = x.shape[0]
loss = $\frac{1}{2}\sum(\mathrm{m} * \mathrm{x} + \mathrm{b} - \mathrm{y})^2$
m = b = 0
itr = 0
While itr < 10000:
    y_pred = $\mathrm{m} * \mathrm{x} + \mathrm{b}$
    m = $\mathrm{m} - \alpha * \mathrm{sum}(\mathrm{x} * (\mathrm{y\_pred} - \mathrm{y}))$
    b = b – $\alpha$ * sum($y_{\mathrm{pred}} - \mathrm{y}$)
    loss = $\frac{1}{2}\sum(\mathrm{y\_pred} - \mathrm{y})^2$
    itr += 1
return m, b
```

b. Multi-variate linear regression

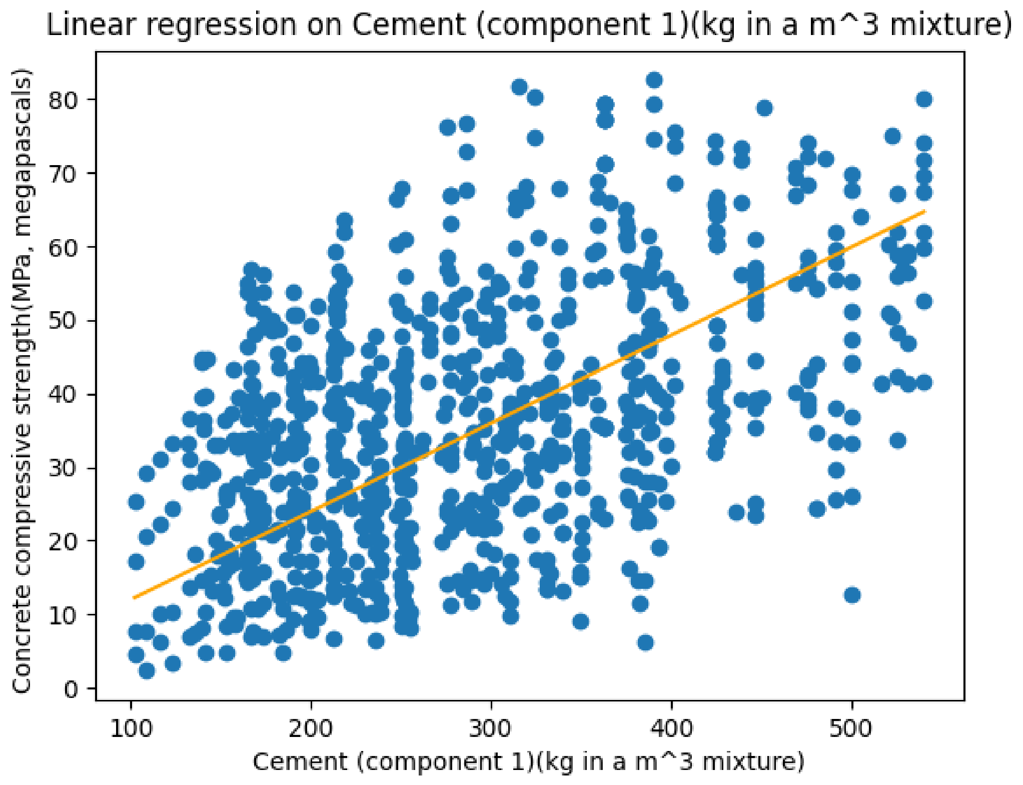
```
X = append(1, X)
a = [random() for _ in (numFeature + 1)]
loss = $\frac{1}{2} *$ norm(X * a - y)) ^ 2
itr = 0
While itr < 20000:
    a = a – $\alpha$ * X * (X * a - y)
    loss = $\frac{1}{2} *$ norm(X * a - y)) ^ 2
return a
```

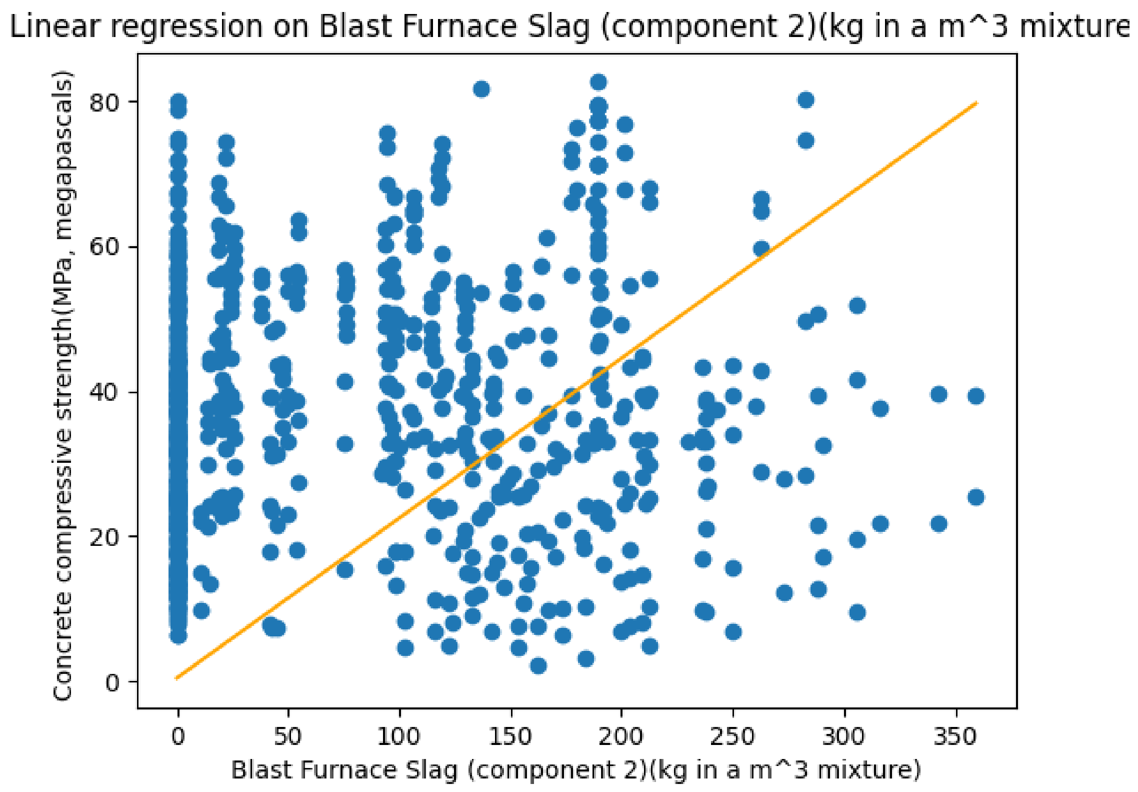
II. Results

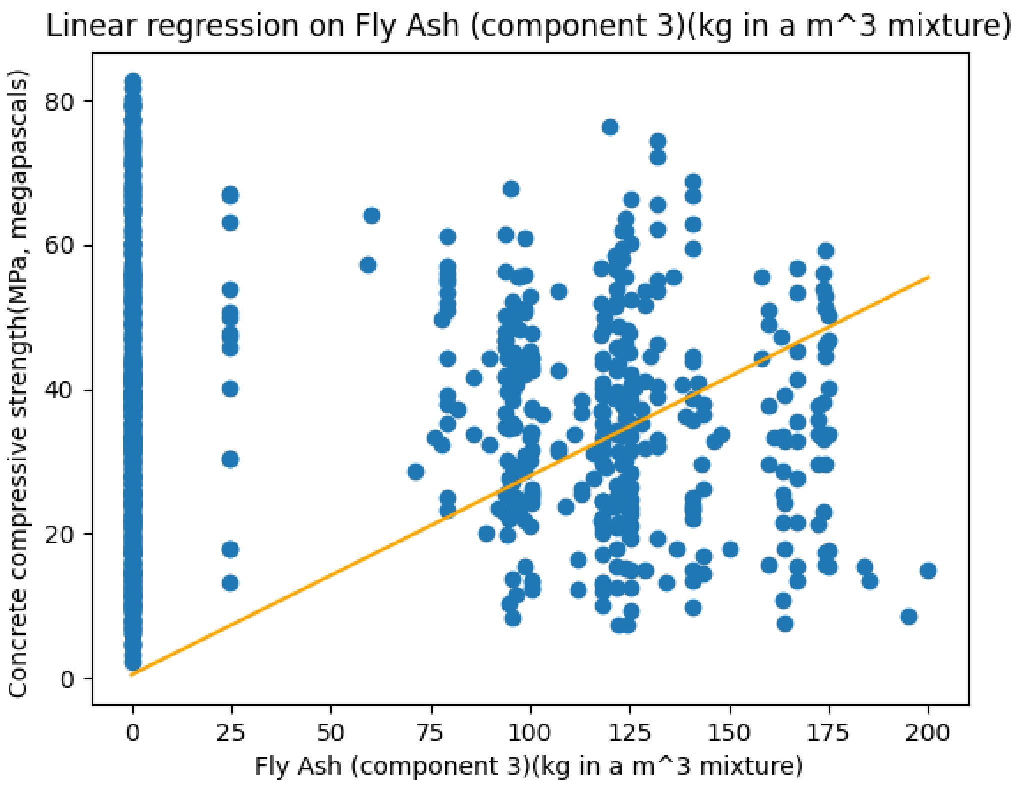
1. MSEs of models on the training/ testing dataset

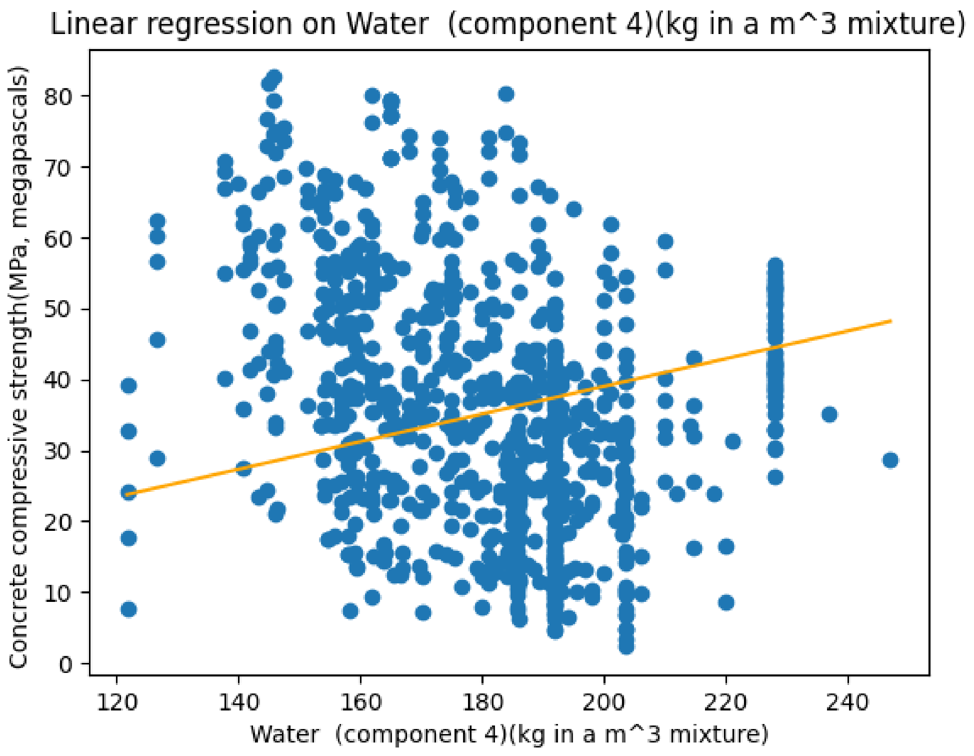
Index	Feature used	MSE train	MSE test
1	Cement	249.3	110.8
2	Blast Furnace Slag	1002.9	343.6
3	Fly Ash	1115.1	750.2
4	Water	358.1	186.9
5	Superplasticizer	741.6	303.4
6	Coarse Aggregate	322.3	166.0
7	Fine Aggregate	333.4	171.5
8	Age	931.5	666.2
All (multi regression)	All	118.6	62.1

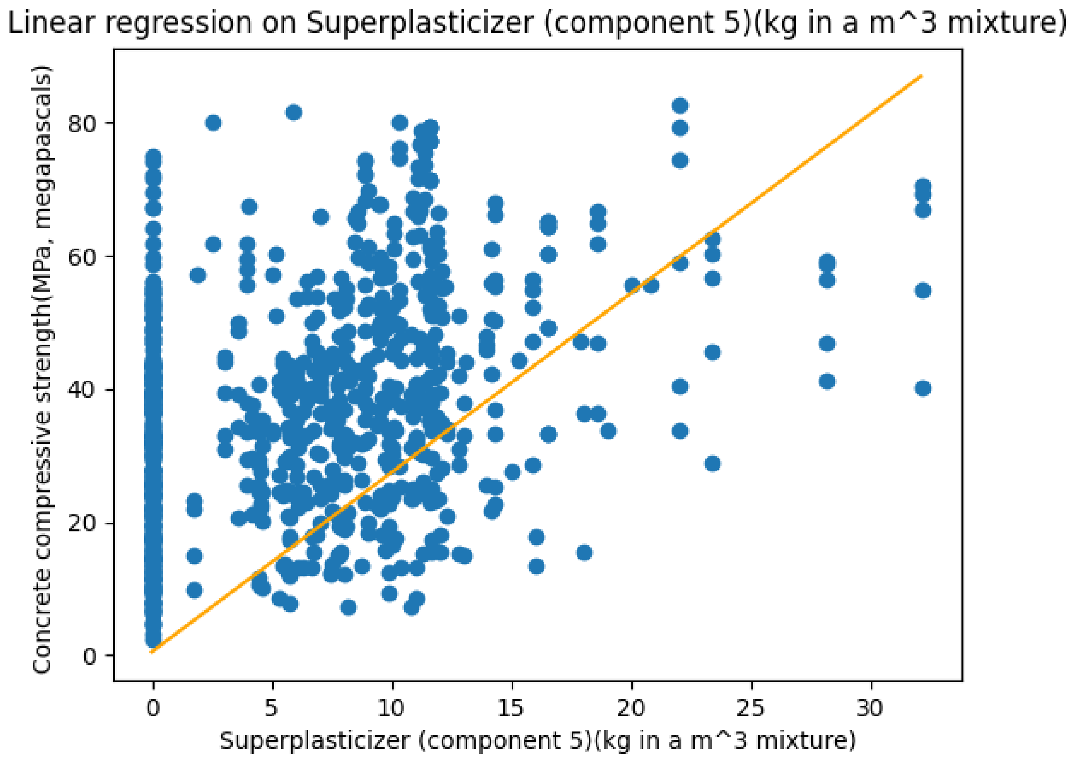
2. Plots of trained uni-variate models

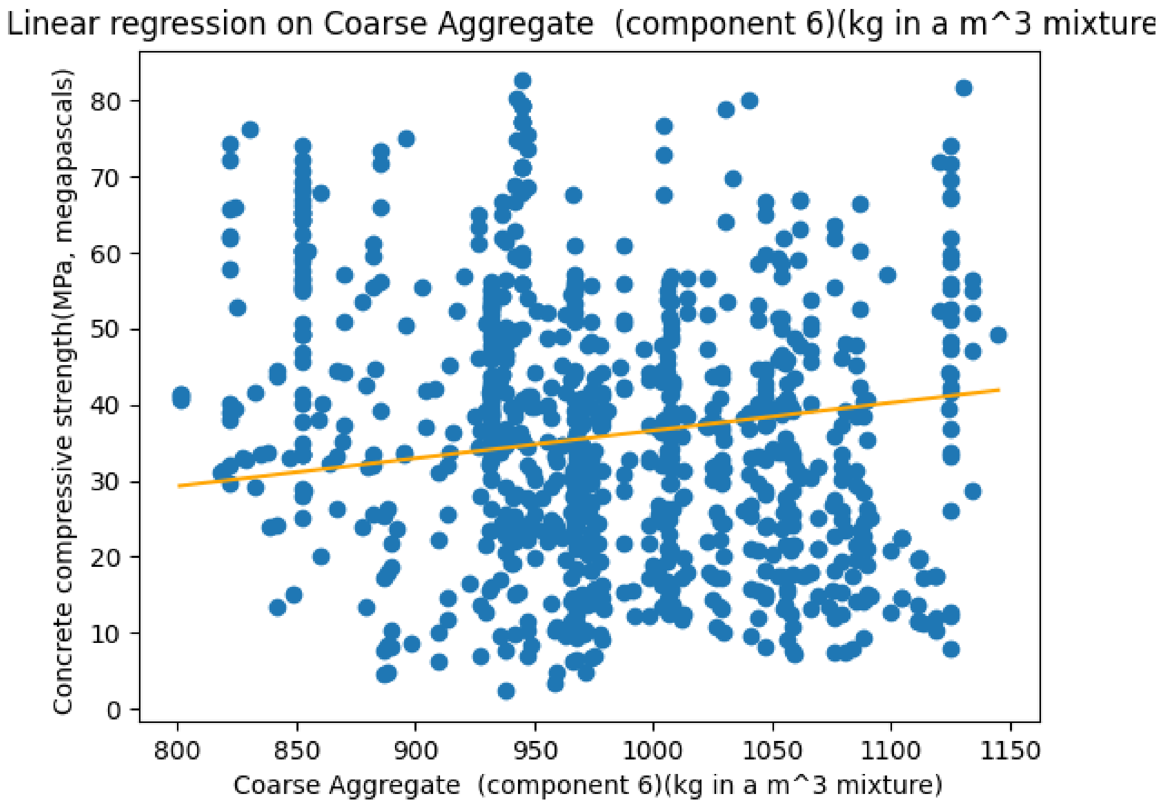

Linear regression on Cement (component 1)(kg in a m^3 mixture)
Concrete compressive strength(MPa, megapascals)
Cement (component 1)(kg in a m^3 mixture)
0
10
20
30
40
50
60
70
80
100
200
300
400
500

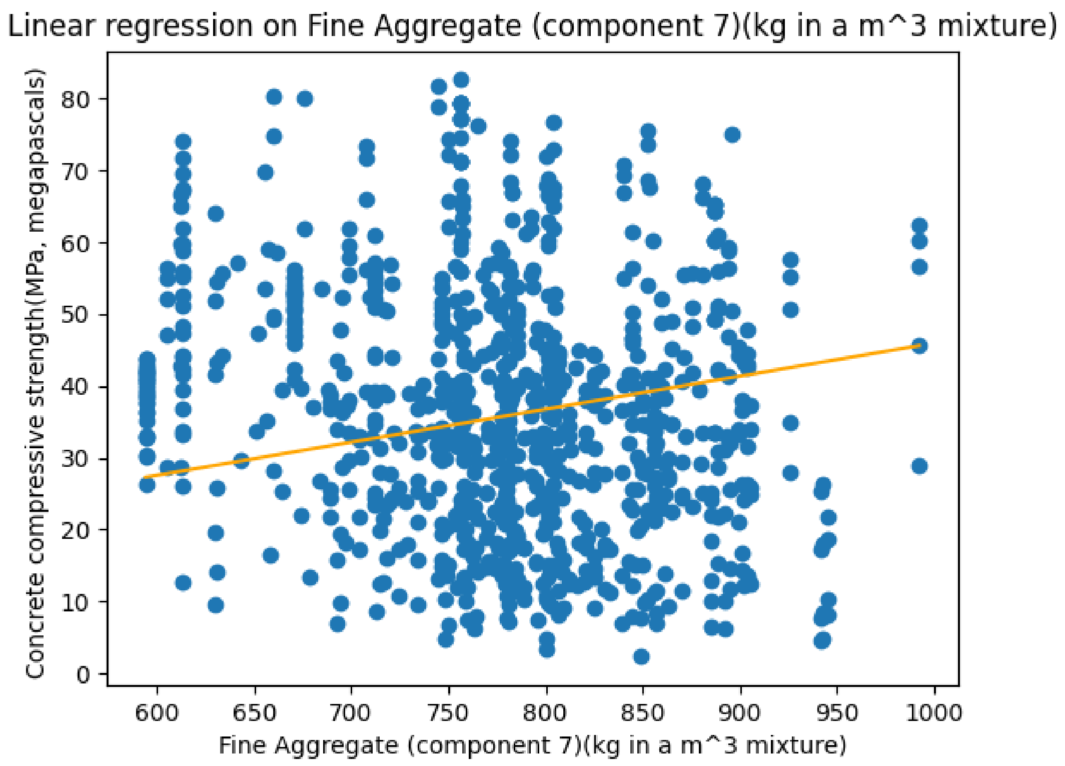

Linear regression on Blast Furnace Slag (component 2)(kg in a m^3 mixture
Concrete compressive strength(MPa, megapascals)
Blast Furnace Slag (component 2)(kg in a m^3 mixture)
0
20
40
60
80
0
50
100
150
200
250
300
350

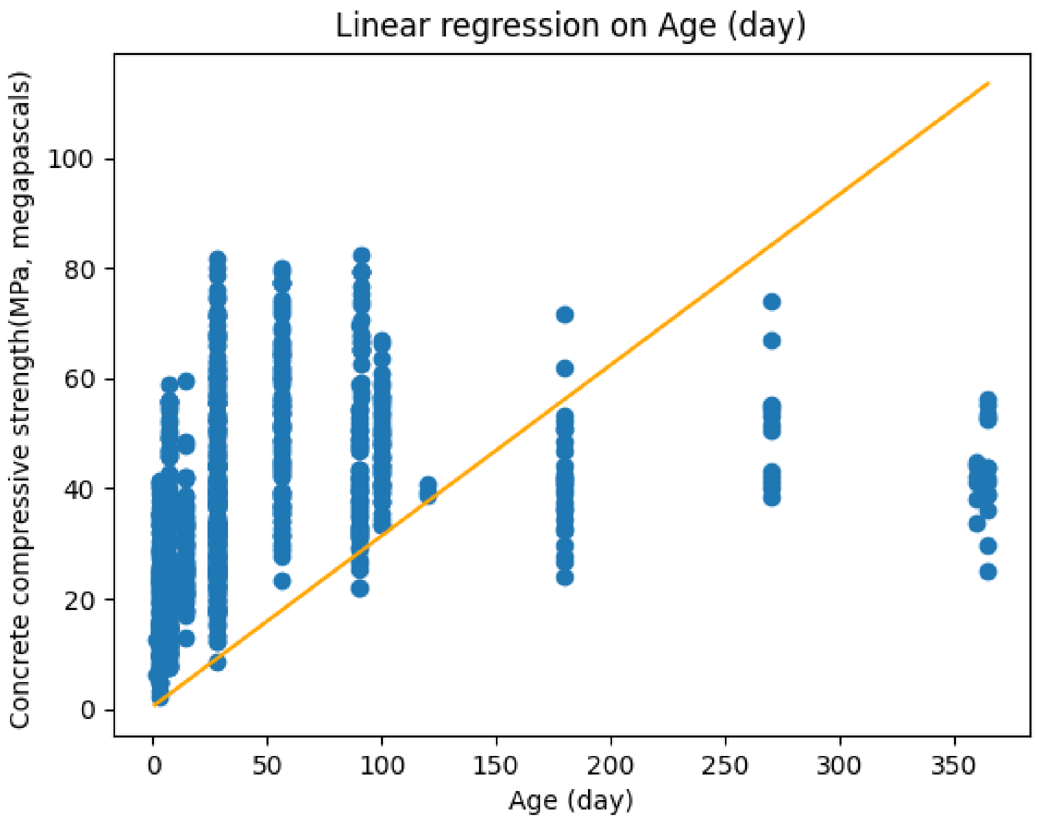

Linear regression on Fly Ash (component 3)(kg in a m^3 mixture)
Concrete compressive strength(MPa, megapascals)
0
20
40
60
80
0
25
50
75
100
125
150
175
200
Fly Ash (component 3)(kg in a m^3 mixture)


Linear regression on Water (component 4)(kg in a m^3 mixture)
Concrete compressive strength(MPa, megapascals)
0
10
20
30
40
50
60
70
80
120
140
160
180
200
220
240
Water (component 4)(kg in a m^3 mixture)


Linear regression on Superplasticizer (component 5)(kg in a m^3 mixture)
Concrete compressive strength(MPa, megapascals)
Superplasticizer (component 5)(kg in a m^3 mixture)
0
20
40
60
80
0
5
10
15
20
25
30


Linear regression on Coarse Aggregate (component 6)(kg in a m^3 mixture
Concrete compressive strength(MPa, megapascals)
Coarse Aggregate (component 6)(kg in a m^3 mixture)
0
10
20
30
40
50
60
70
80
800
850
900
950
1000
1050
1100
1150


Linear regression on Fine Aggregate (component 7)(kg in a m^3 mixture)
Concrete compressive strength(MPa, megapascals)
0
10
20
30
40
50
60
70
80
600
650
700
750
800
850
900
950
1000
Fine Aggregate (component 7)(kg in a m^3 mixture)


Linear regression on Age (day)
Concrete compressive strength(MPa, megapascals)
0
20
40
60
80
100
0
50
100
150
200
250
300
350
Age (day)

III. Discussion

1. Different models compared in performance on the training data

Sorted by the MSE on training data, we have:

index	feature_name	mse_train	mse_test
all	all	118.5863123	62.10907801
1	Cement (component 1)(kg in a m^3 mixture)	249.2751598	110.8091627
6	Coarse Aggregate (component 6)(kg in a m^3 mixture)	322.3370494	165.9516604
7	Fine Aggregate (component 7)(kg in a m^3 mixture)	333.4163391	171.5180711
4	Water (component 4)(kg in a m^3 mixture)	358.1442026	186.8708342
5	Superplasticizer (component 5)(kg in a m^3 mixture)	741.6108333	303.4062929
8	Age (day)	931.460403	666.1896685
2	Blast Furnace Slag (component 2)(kg in a m^3 mixture)	1002.913533	343.5963797
3	Fly Ash (component 3)(kg in a m^3 mixture)	1115.133518	750.1729157

The first row with lowest MSE on train data is multi-variate linear regression. The rest is uni-variate linear regression models.

We can see from the table that, on training data, it is the multi-variate linear regression that performs best. This makes sense, because the multi-variate linear regression has more data (8 features) to fit the output, compared to the other 8 uni-variate linear regression models that only have 1 feature to fit the output.

Between the 8 features, the ranking on performance on train data from best to worst is:

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

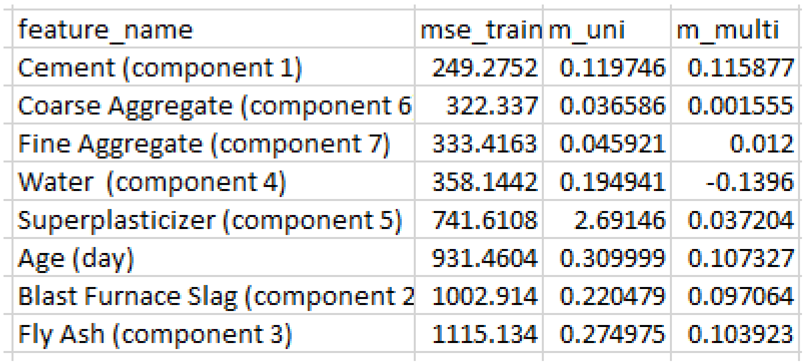
Also, from the MSE on train data, we see that same model that performs well on train data also perform well on test data (the mse_test column is sorted from lowest to highest, except for the Blast Furnace Slag – the model somehow performs not so good on train data, but perform relatively better on test data)

2. Performance of the uni-variate models predicted or failed to predict which features were "more important"in the multi-variate model(s)

As the uni-variate model of the specific feature fit the data better – having smaller MSE (or have higher slope magnitude), we may be tempted conclude that the feature is "more important" in the multi-variate model.

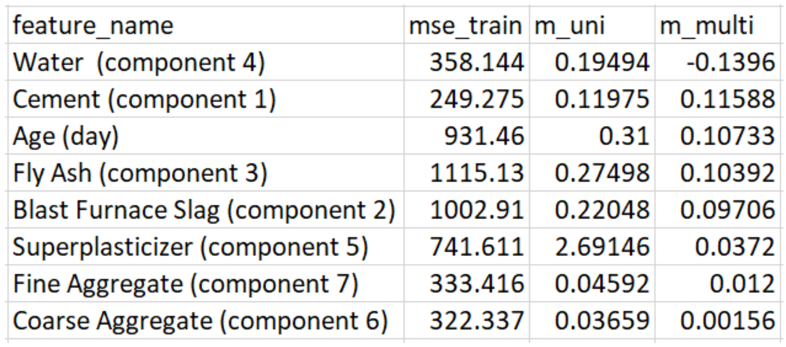
Additionally, we have the coefficient (a) of the multi-variate linear regression model. We will analyze the result and see if the uni-variate can predict which features were more important well.

Arrange the coefficient with the respective feature (sorted by MSE on train data, from low to high), we have



feature_name	mse_train	m_uni	m_multi
Cement (component 1)	249.2752	0.119746	0.115877
Coarse Aggregate (component 6	322.337	0.036586	0.001555
Fine Aggregate (component 7)	333.4163	0.045921	0.012
Water (component 4)	358.1442	0.194941	-0.1396
Superplasticizer (component 5)	741.6108	2.69146	0.037204
Age (day)	931.4604	0.309999	0.107327
Blast Furnace Slag (component 2	1002.914	0.220479	0.097064
Fly Ash (component 3)	1115.134	0.274975	0.103923

If we sort the features by their coefficient of the multi model, we have



feature_name	mse_train	m_uni	m_multi
Water (component 4)	358.144	0.19494	-0.1396
Cement (component 1)	249.275	0.11975	0.11588
Age (day)	931.46	0.31	0.10733
Fly Ash (component 3)	1115.13	0.27498	0.10392
Blast Furnace Slag (component 2)	1002.91	0.22048	0.09706
Superplasticizer (component 5)	741.611	2.69146	0.0372
Fine Aggregate (component 7)	333.416	0.04592	0.012
Coarse Aggregate (component 6)	322.337	0.03659	0.00156

The order in magnitude of the coefficient has nothing to do with the order of the uni-variate model's performance. I would explain the failure because the magnitude of features is different. The coefficient of the multi-variate linear regression model does not necessarily evaluate the importance of the feature with respect to the final output (for example, the coefficient can be small when the feature is relatively big compared to the output). We may achieve a better result with standardized data. Moreover, I am currently using the same step size and number of iterations for all the models. Some models may not be trained enough to reach the optimal position. Or maybe the step size does not fit the feature's magnitude. These are some reasons that may help explains why the performance of uni-variate models fails to predicts which features are more important.

3. Draw some conclusions about what factors predict concrete compressive strength.

From the uni-variate model's performance and the coefficient of the multi-variate linear regression model, we can pick some features that have high rank on both that I think can be the factors that predict concrete compressive strength:

- Cement (component 1)
- Water (component 4)
- Age (component 8)

I believe these 3 components contribute to the concrete compressive strength.

IV. Extra credit

model	mse_train	mse_test
Multi Variate	118.5863123	62.10907801
Closed Form Solution	114.5825168	60.51341108
Polynomial Regression	191.3718705	95.13966237
Sparse Regression	170.6191998	59.80750263

This is the result of training the model with normal Multi-Variate Linear Regression, using Closed Form Solution, Polynomial Regression, and Sparse Regression

1. Multi-variate polynomial regression

I modify the features by adding columns of the feature square to it.

For example, if my input initially is [x_1, x_2, …, x_n] then I transform it to [x_1, x_2, …, $x_{n,}$, x_1^2, x_2^2, …, x_n^2]

Then I perform normal linear regression on the transformed data.

Then, after learning, my model would be y = $(\sum ax^2 + bx)$ + c, which is in the form of quadratic regression model.

This model should have lower loss than other linear model, but maybe I did not converge on a global optimal point that the loss in practice is bigger.

2. Sparse multi-variate regression

I add a penalty term λ|a| to the loss function and minimize it to penalty the big coefficient.

The update rule now would be a = a - $\alpha \frac{\Delta L}{\Delta \vec{a}} - \lambda$ * sign(a)

As a result, although the training loss increases (118.59 -> 170.6), the testing loss decreases (62.11 -> 59.8). We successfully prevents the model from overfitting.

Moreover, the coefficient is now closer to 0 (from

[-4.57743033e-05, 1.15877312e-01 , 9.70640305e-02 , 1.03922580e-01, -1.39596850e-01 3.72044528e-02 1.55534825e-03 1.20004092e-02, 1.07326558e-01]

to [-2.83998710e-05, 8.43111852e-02, 4.39268353e-02, 3.20103993e-02, -3.38773182e-05, -5.69897893e-06, -1.53934939e-06, 5.86323985e-03, 5.31814614e-02]

All features are now closer to 0.

3. Closed Form Solution

Let a be the coefficient that we need to find

a = $(X^TX)^{-1}X^ty$

With X is the feature input, and y is the output.

The loss when we use the Closed Form Solution is 114.5825168, which is smaller than any other linear regression model.