CS 4372.501 Computational Methods For Data Scientists

HW 1

Instructor: ***Anurag Nagar***

Organized by:
Jeremiah Joseph – jsj180002
Savishwa Gaur – sxg180113

Date: Sep 20, 2022

1. Data Set Introduction

Our dataset consisted of molecular information about a set of 908 different chemicals. One of these pieces of molecular information, concentration of LC50, is known to cause death in fish and was used as the target variable. We used 6 other pieces of molecular information: MLOGP (molecular properties), CIC0 (information indices), GATS1i (2D autocorrelations), NdssC (atom-type counts), NdsCH ((atom-type counts), SM1_Dz(Z) (2D matrix-based descriptors), as attributes.

Ultimately, the goal of our project is to create a linear regression model that can predict the concentration of LC50 as well as indicate which of the molecular properties causes high concentrations of LC50.

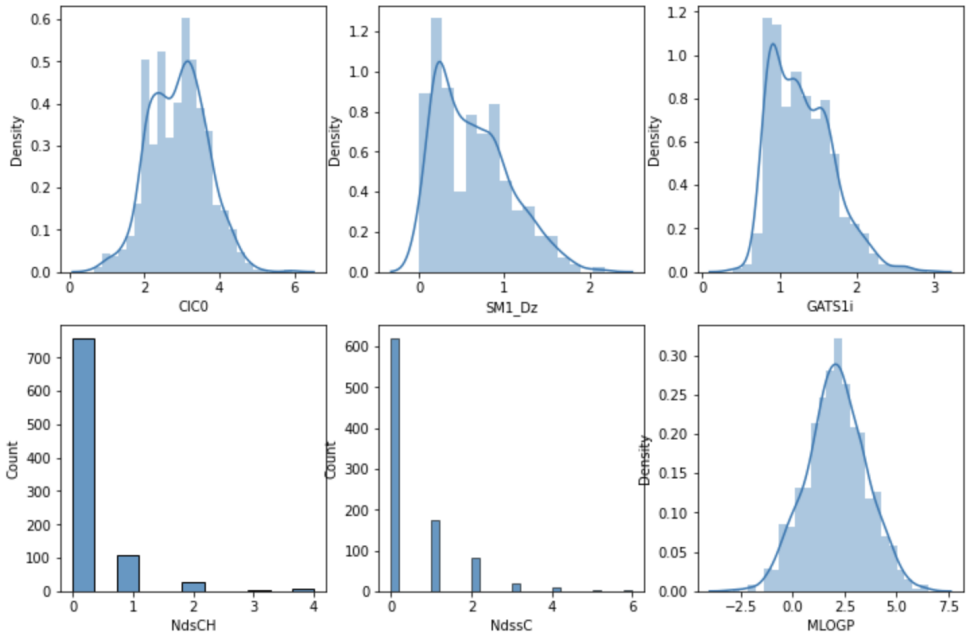
2. Preprocessing

2.1 Examination

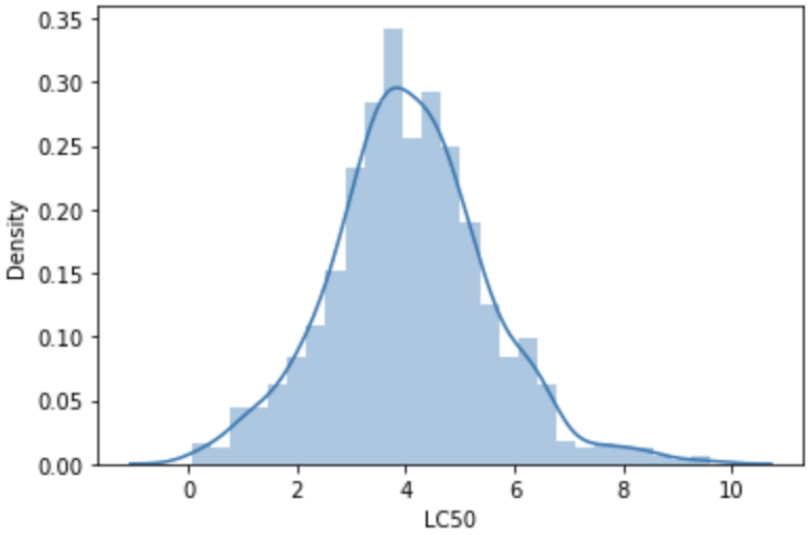
Our dataset had no null values so we did not have to remove any rows. Looking at the targets and predictors, they all had numeric data types so we did not have to do any conversion.

2.2 Target Attributes and Target Variables

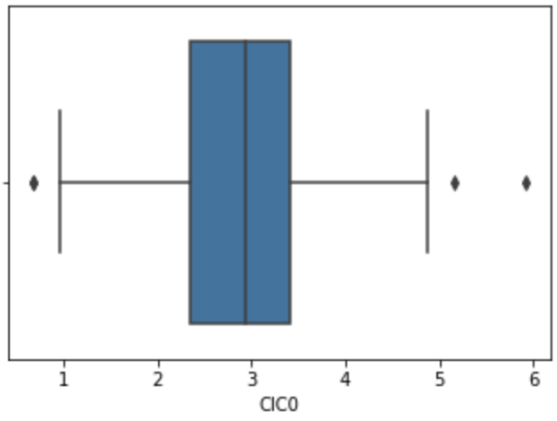
As shown by the distribution plots below, ML0GP follows a normal distribution and while CIC0 and SM1_Dz are also normally distributed, they are both bimodal and SM1_Dz is right skewed. NdsCH and NDssC were describing counts, so they were not normally distributed but were right skewed.

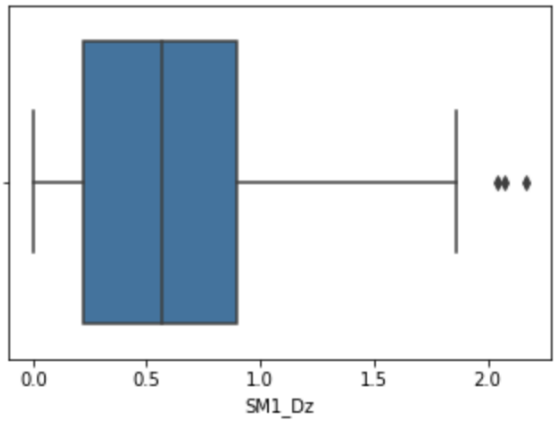


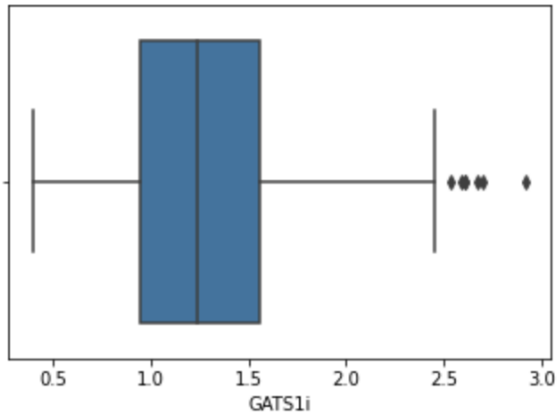
The target variable LC50 is also normally distributed, shown by the plot below

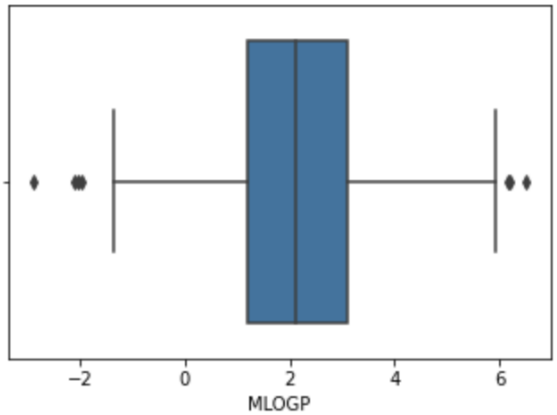


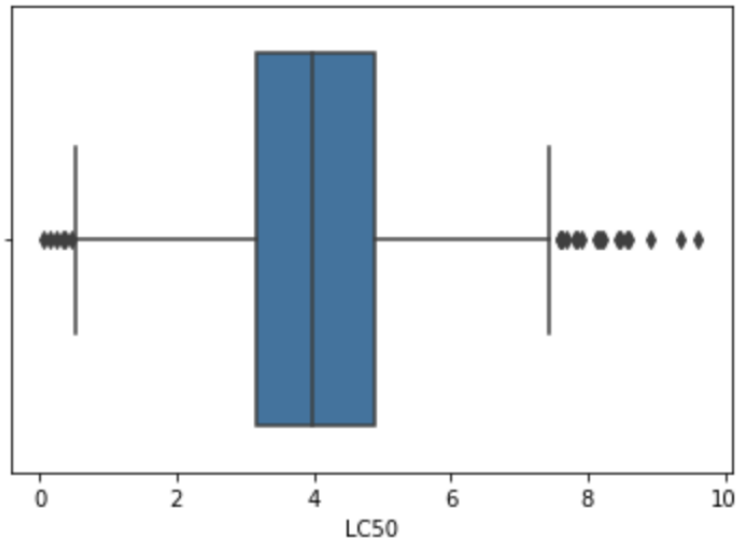
Looking at the boxplots below, we can see that the predictors do not have too many outliers, but the target variable has many:



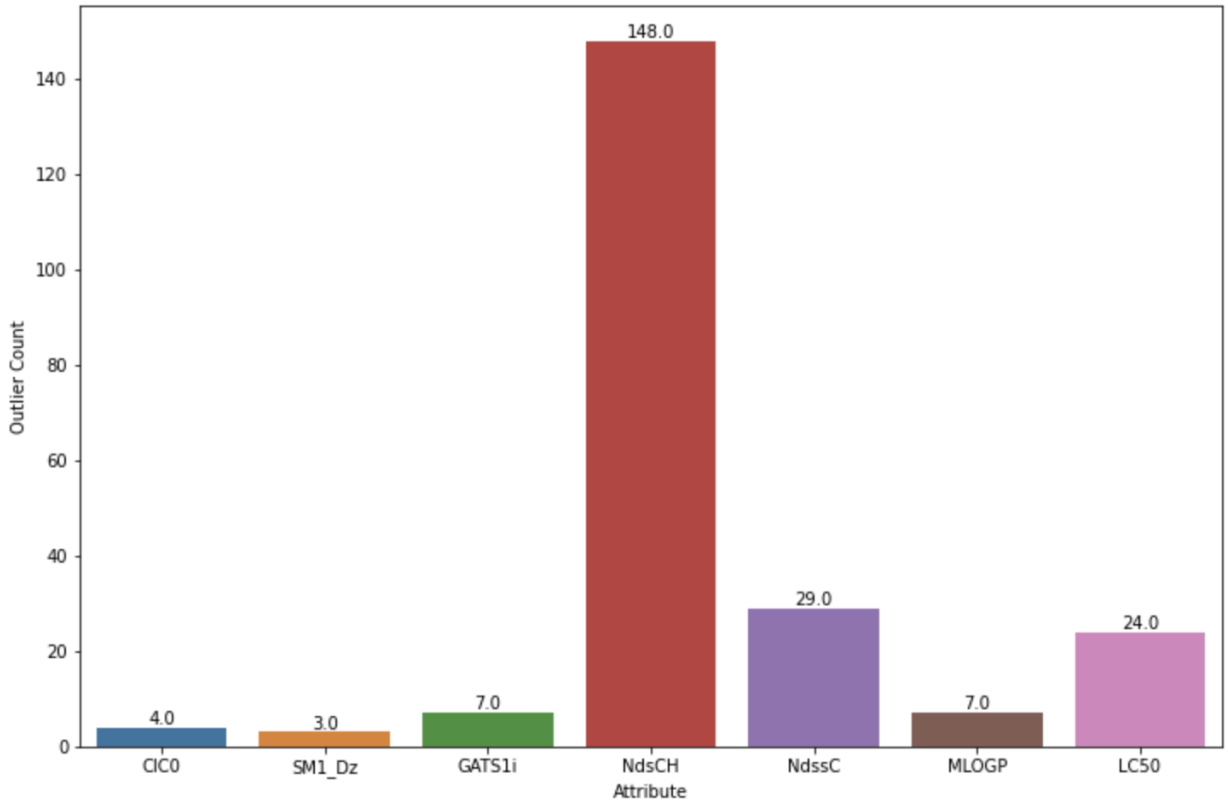




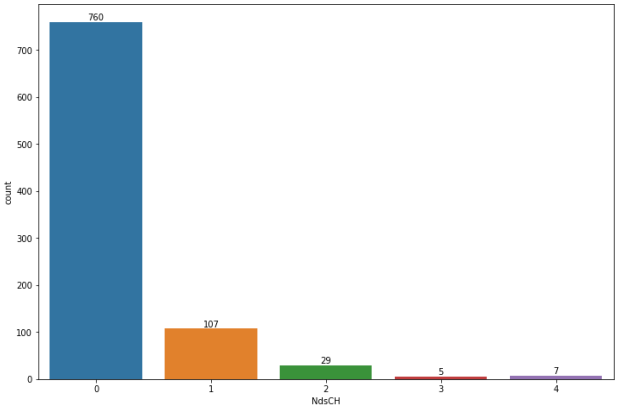


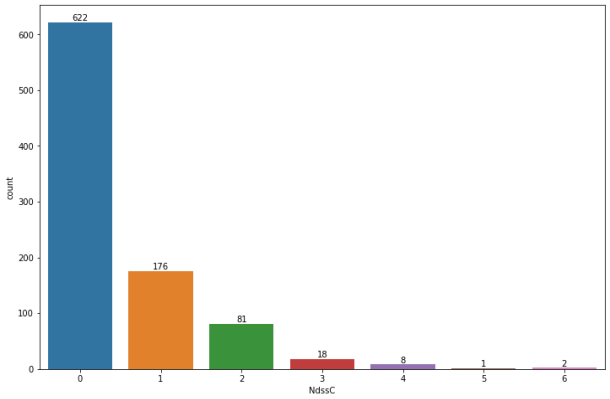


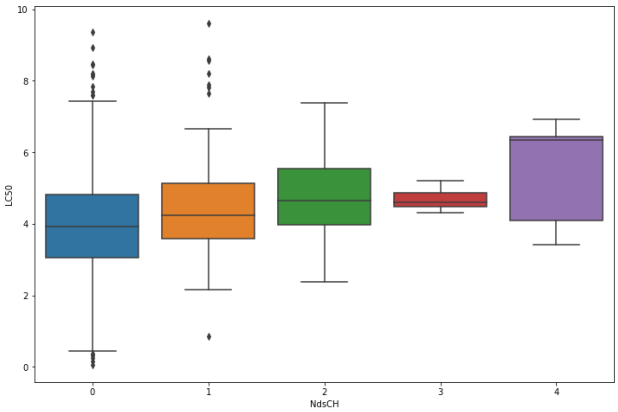
Continuing with outlier detection, according to the plot below, NdsCH and NdssC have the most outliers.

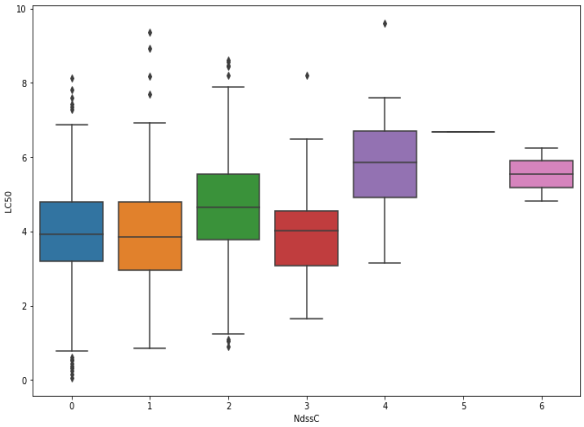


Looking further into NdsCH and NdssC, we see that a majority of the outliers come from the values 0 and 1. Also we see that the means do not vary too much in relation to LC50, meaning that the different values do not seem to affect LC50 by too much.


760
107
29
5
7
count
NdsCH

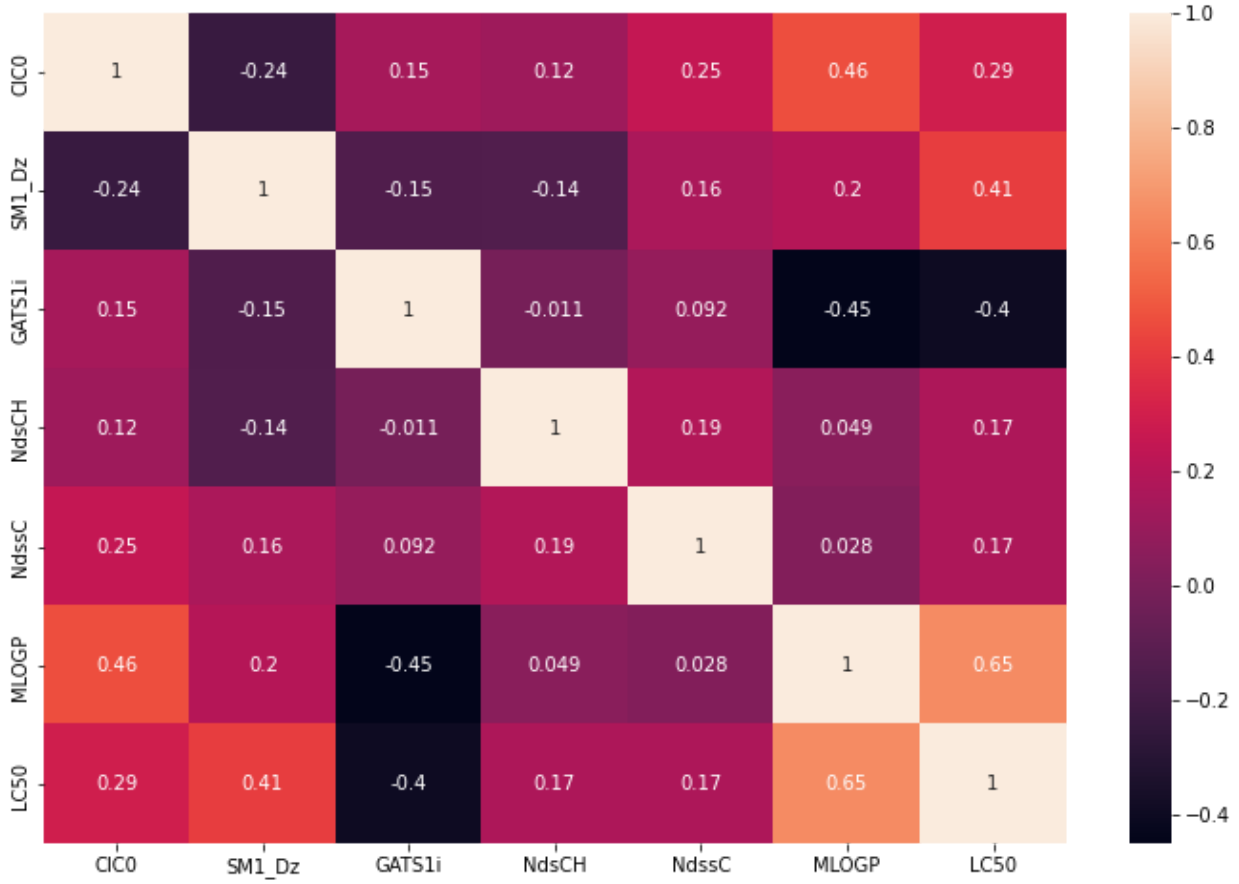

622
176
81
18
8
1
2
count
NdssC


LC50
NdsCH


LC50
NdssC

2.3 Variable Correlation/Collinearity

Looking at the heatmap below, we can see that ML0GP and SM1_Dz are the most correlated with LC50, with GATS1i and CIC0 following close behind.



Looking at the colinearity of the predictors, we see that CIC0 and GATS1i have VIF scores of around 26 and 14 respectively, much higher than 10, indicating that they are highly correlated with the other predictors in the set. Looking at the colinearity of the set of predictors with either of CIC0 or GATS1i removed, the set with GATS1i removed had higher VIF scores than the set of predictors with CIC0 removed, as shown by the tables below.

VIFs of full set

CIC0	25.598575
SM1_Dz	3.464257
GATS1i	14.480146
NdsCH	1.220603
NdssC	1.544874
MLOGP	7.428264

VIFs of set minus GATSL1I

CIC0	5.178220
SM1_Dz	2.740962
NdsCH	1.219143
NdssC	1.501488
MLOGP	4.386795

VIFs of set minus CIc0

SM1_Dz	3.178833
GATS1i	2.929123
NdsCH	1.218146
NdssC	1.439238
MLOGP	2.628803

2.4 Important Attributes

Through the analysis provided in previous sections, we decided to remove both NdsCH and NDssC from the set of predictors. This is because they both had relatively low correlation coefficients, were not distributed well, had low verity in the values making them up (mainly 0s), had too many outliers, and their means related to the target were very consistent across their possible values indicating that we would likely not gain any information by picking any particular value over another. In the case of the last reason mentioned, varying means only came from values with very low counts, making them unreliable.

We also decided to remove CIC0 because it had the highest VIF score, indicating that it was highly correlated with the other predictors in the set. Also, CIC0 had a relatively low correlation coefficient compared to the other possible predictors.

Thus, the final set of predictors was SM1_Dz, GATS1I, and ML0GP. These three variables all have relatively high correlation coefficients with the target and are approximately normally distributed. Additionally, they all have relatively low outlier counts.

2.4 Standardize/Normalize the attributes

As mentioned in our previous analysis, the final set of predictors are approximately normally distributed. Thus, it was reasonable to standardize the data instead of normalizing. The standardization transformed the data to be standard normally distributed with mean approximately 0 and standard deviation approximately 1, which can be seen from the table below.

	SM1_Dz	GATS1i	MLOGP	LC50
count	9.080000e+02	9.080000e+02	9.080000e+02	9.080000e+02
mean	-2.934510e-17	-3.990934e-16	-1.408565e-16	1.467255e-17
std	1.000551e+00	1.000551e+00	1.000551e+00	1.000551e+00
min	-1.467618e+00	-2.277656e+00	-3.485978e+00	-2.757193e+00
25%	-9.468617e-01	-8.699670e-01	-6.285190e-01	-6.273165e-01
50%	-1.365365e-01	-1.347205e-01	1.236726e-02	-5.287703e-02
75%	6.171595e-01	6.817266e-01	6.951415e-01	5.794703e-01
max	3.602169e+00	4.127044e+00	3.075776e+00	3.813033e+00

2.5 Train/Test Split

We decided to perform a 80-20 train-test split because we felt that a larger training set would help the model learn the target variable better since the predictors do not have very high correlation coefficients with the target.

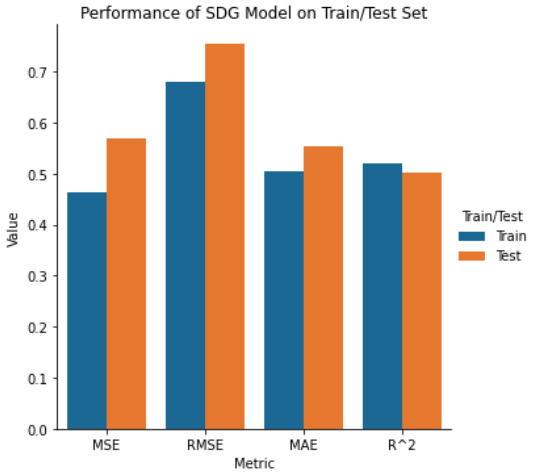
3. SGD Model

3.1 Base Model

The base SGD model was run without specifying any parameter values. The model was evaluated using the training and testing Mean Square Error (MSE), Root Mean Square Error (RMSE) Mean Absolute Error (MAE) and R^2 values.

When looking at the coefficients of the base model, SM1_Dz had a value of 0.292 meaning that for each unit increase of the standardized SM1_Dz value the standardized LC50 concentration went up by 0.292. Similarly GATSli and MLOGP have coefficient values of -0.114 and 0.531 respectively, meaning for each unit increase of those standardized values the standardized LC50 went up by their respective coefficient values. The base model also had an intercept of -0.0052, meaning that when the concentration of all the predictors is 0, the concentration of LC50 is -0.0052, which practically does not make sense. This is addressed in the parameter tuning section.

Below is a chart and report of the base model's performance on the training and testing sets.




```
Test Set Metrics
MSE Test:  0.5696354580778993
RMSE Test:  0.7547419811285836
MAE Test:  0.5533684100039337
R^2 Test:  0.5016436729707855
----------------------------------------------------
Training Set Metrics
MSE Train:  0.46249329024248764
RMSE Train:  0.6800685923070464
MAE Train:  0.505379145752092
R^2 Train:  0.5203053011180654
```

3.2 Parameter Tuning

3.2.1 Loss Function

The Loss Function parameter specifies the loss function to be used during gradient descent. The possible values are ‘huber’, ‘squared_error’, ‘epsilon_insensitive’, or ‘squared_epsilon_insensitive’.

Below are the performance plots for the possible loss functions models in the order listed above:

Loss_function = ‘huber’


```
Test Set Metrics
MSE Test:  0.6029127535676596
RMSE Test:  0.7764745672381418
MAE Test:  0.5736396469808047
R^2 Test:  0.47253040321454276
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.4833024281327412
RMSE Train:  0.6951995599342258
MAE Train:  0.5183439966597357
R^2 Train:  0.4987222136552739
```

Loss_function = ‘sqaured_error’


```
Test Set Metrics
MSE Test:  0.5695106651543215
RMSE Test:  0.7546593040268711
MAE Test:  0.5532720152753288
R^2 Test:  0.5017528504142743
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.4624680480343348
RMSE Train:  0.680050033478666
MAE Train:  0.5052842002127919
R^2 Train:  0.5203314821539733
```

Loss_function = ‘epsilon_insensitive’


```
Test Set Metrics
MSE Test:  0.5707293711058955
RMSE Test:  0.7554663269172859
MAE Test:  0.5549950802046558
R^2 Test:  0.5006866425208885
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.4628507053179432
RMSE Train:  0.6803313202535535
MAE Train:  0.5063715230789556
R^2 Train:  0.5199345927843151
```

Loss_function = ‘squared_epsilon_insensitive’


```
Test Set Metrics
MSE Test:  0.5744573190391956
RMSE Test:  0.7579296267063292
MAE Test:  0.548870181342088
R^2 Test:  0.4974251769413678
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.4676256339477784
RMSE Train:  0.6838315830288759
MAE Train:  0.5009277020833003
R^2 Train:  0.5149820713108239
```

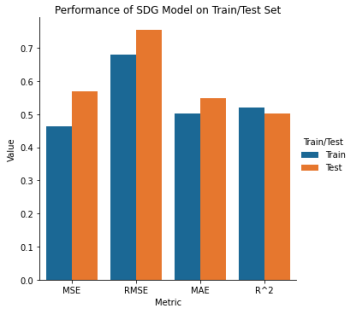
From these initial results, we then experimented with the epsilon values of the applicable loss functions. For the ‘huber’ loss function, this determines the threshold at which it becomes less important to get the prediction exactly right. For the ‘epsilon_intensive’ and ‘sqaured_epsilon_intensive’, this determines the threshold at which any differences between the predicted value and the actual value are ignored.

The following plots and report show the experiments for the ‘huber’ loss function:

Epsilon = 1


```
Test Set Metrics
MSE Test:  0.5726153173298255
RMSE Test:  0.7567134975205778
MAE Test:  0.5492173060043545
R^2 Test:  0.4990366868873264
----------
Training Set Metrics
MSE Train:  0.465609421132058
RMSE Train:  0.6823557877911332
MAE Train:  0.501591439507281
R^2 Train:  0.5170732726750897
```

Epsilon = 2 (best)




```
Test Set Metrics
MSE Test:  0.5700076860497182
RMSE Test:  0.7549885337206905
MAE Test:  0.5497189610740988
R^2 Test:  0.5013180223073259
----------
Training Set Metrics
MSE Train:  0.46359609863086404
RMSE Train:  0.6808789162772365
MAE Train:  0.5024424893473124
R^2 Train:  0.5191614762260128
```

Epsilon = 5


```
Test Set Metrics
MSE Test:  0.5695106651543215
RMSE Test:  0.7546593040268711
MAE Test:  0.5532720152753288
R^2 Test:  0.5017528504142743
----------
Training Set Metrics
MSE Train:  0.4624680480343348
RMSE Train:  0.6800500334786666
MAE Train:  0.5052842002127919
R^2 Train:  0.5203314821539733
```

The following plots and report show the experiments for the ‘epsilon_intensive’ loss function:

Epsilon = 0.5


```
Test Set Metrics
MSE Test:  0.5775878052127947
RMSE Test:  0.7599919770713338
MAE Test:  0.5661784546327536
R^2 Test:  0.4946864120538118
----------
Training Set Metrics
MSE Train:  0.4716240607295404
RMSE Train:  0.6867489066096432
MAE Train:  0.5225672966502783
R^2 Train:  0.5108349319435194
```

Epsilon = 1

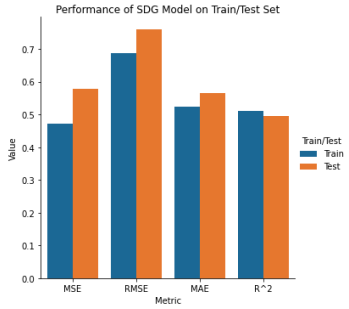

```
Test Set Metrics
MSE Test:  0.6218515985325727
RMSE Test:  0.7885756771119514
MAE Test:  0.6075090758869081
R^2 Test:  0.45596139740381414
----------
Training Set Metrics
MSE Train:  0.5138167235983702
RMSE Train:  0.7168101028852553
MAE Train:  0.5643943944361047
R^2 Train:  0.46707300688017717
```

Epsilon = 0.2


```
Test Set Metrics
MSE Test:  0.5711998331334515
RMSE Test:  0.755777634713711
MAE Test:  0.5564589981386093
R^2 Test:  0.5002750499405203
----------
Training Set Metrics
MSE Train:  0.46349578650066964
RMSE Train:  0.6808052485848428
MAE Train:  0.5090039346501648
R^2 Train:  0.5192655192426424
```

The following plots and report show the experiments for the ‘sqaured_epsilon_intensive’ loss function:

Epsilon = 0.5




```
Test Set Metrics
MSE Test:  0.5775878052127947
RMSE Test:  0.7599919770713338
MAE Test:  0.5661784546327536
R^2 Test:  0.4946864120538118
----------------------------------------------------
Training Set Metrics
MSE Train:  0.4716240607295404
RMSE Train:  0.6867489066096432
MAE Train:  0.5225672966502783
R^2 Train:  0.5108349319435194
```

Epsilon = 1


```
Test Set Metrics
MSE Test:  0.6218515985325727
RMSE Test:  0.7885756771119514
MAE Test:  0.6075090758869081
R^2 Test:  0.45596139740381414
----------------------------------------------------
Training Set Metrics
MSE Train:  0.5138167235983702
RMSE Train:  0.7168101028852553
MAE Train:  0.5643943944361047
R^2 Train:  0.46707300688017717
```

Epsilon = 2


```
Test Set Metrics
MSE Test:  0.7365085075439994
RMSE Test:  0.8582007384895445
MAE Test:  0.661270609988506
R^2 Test:  0.3556516374807521
----------------------------------------------------
Training Set Metrics
MSE Train:  0.6012399706616838
RMSE Train:  0.7753966537596637
MAE Train:  0.6033678694610228
R^2 Train:  0.376398246705106
```

Following these experiments, it was determined that a loss function of ‘hubber’ with epsilon = 2 had the best testing performance and hence we moved forward for further experiments.

3.2.2 Penalty

The penalty parameter refers to the regularization term to be used. The possible values are ‘l2’ which is the standard regularizer for linear SVM models. ‘l1’ and ‘elasticnet’ might bring sparsity to the model (feature selection) not achievable with ‘l2’. The following plots and reports shows the experiments for the ‘l1’ and ‘l2’ penalties respectively:


```
Test Set Metrics
MSE Test:  0.5691136449543286
RMSE Test:  0.7543962121818538
MAE Test:  0.5514725264404172
R^2 Test:  0.5021001910262728
----------------------------------------------------
Training Set Metrics
MSE Train:  0.4626555136060252
RMSE Train:  0.680187851704237
MAE Train:  0.5043343125005791
R^2 Train:  0.5201370442175535
```


```
Test Set Metrics
MSE Test:  0.5691100199483579
RMSE Test:  0.7543938095904272
MAE Test:  0.5514732642861679
R^2 Test:  0.5021033624311343
----------------------------------------------------
Training Set Metrics
MSE Train:  0.4626555160984128
RMSE Train:  0.6801878535363689
MAE Train:  0.5043386759635298
R^2 Train:  0.5201370416324671
```

The ‘elasticnet’ penalty has another parameter for ‘l1_ratio’ which corresponds to the proportion that L1 Regularization will be used compared to L2 Regularization. The following plots and reports who the experiments:

L1_ratio = 0.15


```
Test Set Metrics
MSE Test:  0.5691105621487209
RMSE Test:  0.7543941689519617
MAE Test:  0.5514731537043757
R^2 Test:  0.5021028880769178
----------------------------------------------------
Training Set Metrics
MSE Train:  0.46265551416535144
RMSE Train:  0.6801878521153928
MAE Train:  0.5043380215146462
R^2 Train:  0.5201370436374244
```

L1_ratio = 0.05


```
Test Set Metrics
MSE Test:  0.5691102006210166
RMSE Test:  0.7543939293373301
MAE Test:  0.5514732274292978
R^2 Test:  0.5021032043662503
----------------------------------------------------
Training Set Metrics
MSE Train:  0.4626555153929236
RMSE Train:  0.6801878530177701
MAE Train:  0.5043384578166686
R^2 Train:  0.5201370423641953
```

L1_ratio = 0.35


```
Test Set Metrics
MSE Test:  0.5691112859337186
RMSE Test:  0.7543946486645559
MAE Test:  0.5514730062097973
R^2 Test:  0.5021022548599579
----------------------------------------------------
Training Set Metrics
MSE Train:  0.4626555124438758
RMSE Train:  0.6801878508499515
MAE Train:  0.5043371488774
R^2 Train:  0.5201370454229264
```

L1_ratio = 0.5


```
Test Set Metrics
MSE Test:  0.5691118294109057
RMSE Test:  0.7543950088719474
MAE Test:  0.551472895549718
R^2 Test:  0.5021017793886877
----------------------------------------------------
Training Set Metrics
MSE Train:  0.46265551179477776
RMSE Train:  0.6801878503728053
MAE Train:  0.5043364943704122
R^2 Train:  0.5201370460961663
```

L1_ratio = 0.75


```
Test Set Metrics
MSE Test:  0.5691127364224128
RMSE Test:  0.7543956100232906
MAE Test:  0.5514727110416808
R^2 Test:  0.5021009858725594
----------------------------------------------------
Training Set Metrics
MSE Train:  0.4626555119359446
RMSE Train:  0.6801878504765758
MAE Train:  0.5043354034700887
R^2 Train:  0.520137045949749
```

Through these experiments, we saw that the penalty of l2 (default) had the best performance, so we moved forward with it for further experiments.

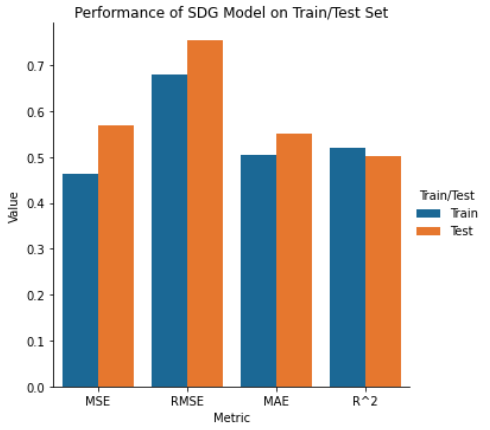
3.2.3 Alpha

This parameter is the Constant that multiplies the regularization term. The higher the value, the stronger the regularization. Also used to compute the learning rate when the learning_rate is set to ‘optimal’. The following are the most relevant experiments for alpha, many are omitted for convenience:

Alpha = 0.0002

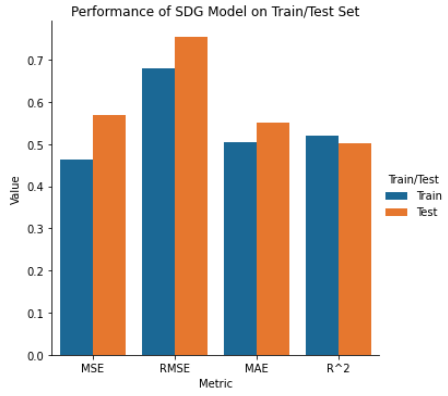

```
Test Set Metrics
MSE Test:  0.5691119286299208
RMSE Test:  0.7543950746325965
MAE Test:  0.5514804147825987
R^2 Test:  0.5021016925850605
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.4626539163904812
RMSE Train:  0.6801866776043773
MAE Train:  0.5043401910909909
R^2 Train:  0.520138700837975
```

Alpha = 0.00005




```
Test Set Metrics
MSE Test:  0.5691090688769387
RMSE Test:  0.7543931792354294
MAE Test:  0.5514696883097683
R^2 Test:  0.502104194493908
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.4626563191978663
RMSE Train:  0.6801884438873291
MAE Train:  0.5043379181371354
R^2 Train:  0.5201362086635166
```

Alpha = 0




```
Test Set Metrics
MSE Test:  0.5691081199861292
RMSE Test:  0.7543925503251799
MAE Test:  0.551466111847773
R^2 Test:  0.502105024648934
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.46265712446194945
RMSE Train:  0.6801890358289742
MAE Train:  0.5043371601355722
R^2 Train:  0.5201353734494281
```

Through these experiments, alpha = 0 gave the best performance, so moved forward with it in further experiments

3.2.4 Max Iterations

This parameter is the maximum number of passes over the training data (aka epochs). It only impacts the behavior in the fit method, and not the partial_fit method. The following plots and report show the experiments:

Max_iter = 2500


```
Test Set Metrics
MSE Test:  0.5691081199861292
RMSE Test:  0.7543925503251799
MAE Test:  0.551466111847773
R^2 Test:  0.502105024648934
---------------------------------------------------
Training Set Metrics
MSE Train:  0.46265712446194945
RMSE Train:  0.6801890358289742
MAE Train:  0.5043371601355722
R^2 Train:  0.5201353734494281
```

Max_iter = 3500


```
Test Set Metrics
MSE Test:  0.5691081199861292
RMSE Test:  0.7543925503251799
MAE Test:  0.551466111847773
R^2 Test:  0.502105024648934
---------------------------------------------------
Training Set Metrics
MSE Train:  0.46265712446194945
RMSE Train:  0.6801890358289742
MAE Train:  0.5043371601355722
R^2 Train:  0.5201353734494281
```

Max_iter = 500


```
Test Set Metrics
MSE Test:  0.5691081199861292
RMSE Test:  0.7543925503251799
MAE Test:  0.551466111847773
R^2 Test:  0.502105024648934
---------------------------------------------------
Training Set Metrics
MSE Train:  0.46265712446194945
RMSE Train:  0.6801890358289742
MAE Train:  0.5043371601355722
R^2 Train:  0.5201353734494281
```

Through these experiments, the performance did not increase, so we left max_iter = 1000 (default) for further experiments

3.2.5 Learning Rate

This parameter determines the learning schedule as follows:

- ‘constant’: eta = eta0
- ‘optimal’: eta = 1.0 / (alpha * (t + t0)) where t0 is chosen by a heuristic proposed by Leon Bottou.
- ‘invscaling’: eta = eta0 / pow(t, power_t)
- ‘adaptive’: eta = eta0

The following plots and reports show the experiments:

Learning_rate = ‘constant’


```
Test Set Metrics
MSE Test:  0.5766351097738258
RMSE Test:  0.7593649384675499
MAE Test:  0.5514892775449021
R^2 Test:  0.4955198956317901
----------------------------------------------------
Training Set Metrics
MSE Train:  0.4684470310556081
RMSE Train:  0.6844319038849724
MAE Train:  0.5031413570854432
R^2 Train:  0.5141301241656095
```

Learning_rate = ‘optimal’


```
Test Set Metrics
MSE Test:  0.6939379569169165
RMSE Test:  0.8330293853862039
MAE Test:  0.653897038579674
R^2 Test:  0.3928952868169615
----------------------------------------------------
Training Set Metrics
MSE Train:  0.6027551779500868
RMSE Train:  0.7763730919796789
MAE Train:  0.6218997274293528
R^2 Train:  0.3748266846537449
```

Learning_rate = ‘adaptive’


```
Test Set Metrics
MSE Test:  0.5691897022950331
RMSE Test:  0.7544466199109339
MAE Test:  0.5514271416331
R^2 Test:  0.5020336508268881
----------------------------------------------------
Training Set Metrics
MSE Train:  0.46264993287550704
RMSE Train:  0.6801837493468269
MAE Train:  0.5041408246357425
R^2 Train:  0.5201428325109232
```

Through these experiments, changing the learning rate away from the default ‘invscaling’ did not increase performance, so we kept it for further experiments.

3.2.6 Warm Start

This parameter will reuse the solution of the gradient descent to the previous call to fit if set to true as its initialization, otherwise it will erase the previous solution. Below are the results of the experiments:

Warm_start = True


```
Test Set Metrics
MSE Test:  0.5691081199861292
RMSE Test:  0.7543925503251799
MAE Test:  0.551466111847773
R^2 Test:  0.502105024648934
--------------------------------------------------
Training Set Metrics
MSE Train:  0.46265712446194945
RMSE Train:  0.6801890358289742
MAE Train:  0.5043371601355722
R^2 Train:  0.5201353734494281
```

Through these experiments, changing the warm start away from the default value of false does not increase performance. Thus, we kept it at False for further experiments.

3.2.7 Average

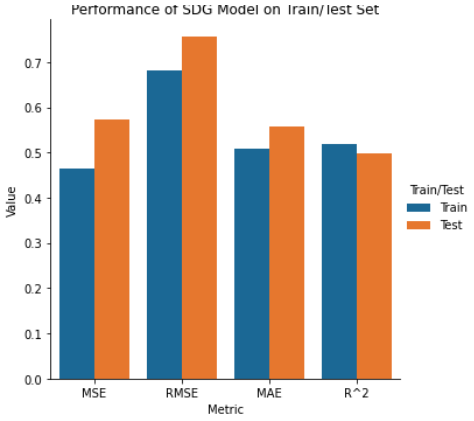
This parameter when set to True will compute the averaged stochastic gradient descent weights across all updates and store them in the coefficients. If it is set to an integer value

greater than 1, averaging will begin once the total number of samples seen reaches average. So average=10 will begin averaging after seeing 10 samples. Below are the experiments:

Average = True

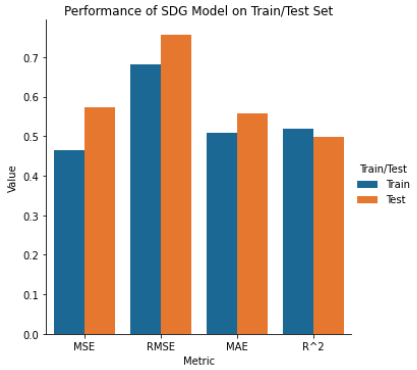

```
Test Set Metrics
MSE Test:  0.5723644431560768
RMSE Test:  0.7565477137339567
MAE Test:  0.556898680104603
R^2 Test:  0.4992561688911289
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.4634781864646645
RMSE Train:  0.6807923225658942
MAE Train:  0.5074660379331906
R^2 Train:  0.5192837738728174
```

Average = 10




```
Test Set Metrics
MSE Test:  0.5722471942167301
RMSE Test:  0.7564702203105751
MAE Test:  0.5568184195592163
R^2 Test:  0.49935874633769095
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.4634267830304297
RMSE Train:  0.6807545688648955
MAE Train:  0.5074141483040033
R^2 Train:  0.5193370891433888
```

Average = 5




```
Test Set Metrics
MSE Test:  0.5723107656106043
RMSE Test:  0.7565122375815241
MAE Test:  0.5568619326348175
R^2 Test:  0.499303129704446
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.46345470807840733
RMSE Train:  0.6807750789199083
MAE Train:  0.5074430809142989
R^2 Train:  0.5193081254853997
```

Average = 1


```
Test Set Metrics
MSE Test:  0.5723644431560768
RMSE Test:  0.7565477137339567
MAE Test:  0.556898680104603
R^2 Test:  0.4992561688911289
----------------------------------------------------------
Training Set Metrics
MSE Train:  0.4634781864646645
RMSE Train:  0.6807923225658942
MAE Train:  0.5074660379331906
R^2 Train:  0.5192837738728174
```

Through these experiments, average did not increase performance, so it was kept at False for further experiments.

3.2.8 Fit Intercept

This parameter determines whether the intercept should be estimated or not. If False, the data is assumed to be already centered. This parameter addresses the unreasonable intercept from the base model as it does not make sense that, first the other compounds will have 0 concentration, and second that if the compounds were to have 0 concentration then the concentration of LC50 will be a negative number. Below are the experiments:

Fit_intercept = False


```
Test Set Metrics
MSE Test:  0.5690280796802941
RMSE Test:  0.7543394989527554
MAE Test:  0.5516076525643188
R^2 Test:  0.5021750494205044
--------------------------------------------------
Training Set Metrics
MSE Train:  0.46266341906406644
RMSE Train:  0.6801936629108407
MAE Train:  0.5046096230155906
R^2 Train:  0.5201288447335942
```

Through these experiments, a value of False for fit_intercept increased the performance. Thus, it will be used for the final parameters.

3.2.9 Final Parameters

Through the thorough experimentation described above, the list of best parameters are as follows:

- loss = 'huber'
- epsilon = 2
- penalty = '12'
- alpha = 0
- max_iter = 1000
- learning_rate = 'invscaling'
- warm_start = False
- average = False
- fit_intercept = False
- random_state = 42

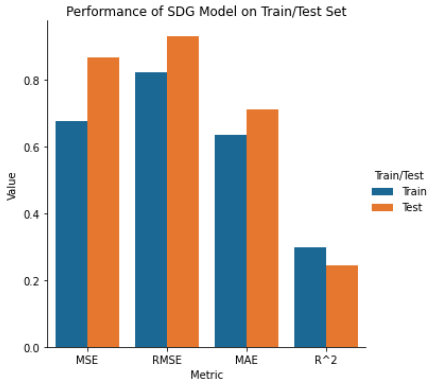
3.2.10 Variable Selection/Removal

After tuning the hyper parameters of the model, we then tuned the set of predictors again by comparing the performance of the model after adding new variables to the list. Below are the experiments:

Predictors = [‘SM1_Dz’]


```
Test Set Metrics
MSE Test:  1.0126143935115013
RMSE Test:  1.0062874308623264
MAE Test:  0.762893227896783
R^2 Test:  0.1140951942315781
----------------------------------------------------
Training Set Metrics
MSE Train:  0.7857927333864649
RMSE Train:  0.8864495097784559
MAE Train:  0.6746670844464394
R^2 Train:  0.1849814546978602
```

Predictors = ['SM1_Dz', 'GATS1i']




```
Test Set Metrics
MSE Test:  0.8657637666311045
RMSE Test:  0.9304642747742143
MAE Test:  0.7120765558263488
R^2 Test:  0.24257023558696378
----------------------------------------------------
Training Set Metrics
MSE Train:  0.6772906933231243
RMSE Train:  0.8229767270847483
MAE Train:  0.6353467486969904
R^2 Train:  0.2975189866671798
```

Predictors = ['SM1_Dz', 'GATS1i', 'MLOGP']


```
Test Set Metrics
MSE Test:  0.5690280796802941
RMSE Test:  0.7543394989527554
MAE Test:  0.5516076525643188
R^2 Test:  0.5021750494205044
----------------------------------------------------
Training Set Metrics
MSE Train:  0.46266341906406644
RMSE Train:  0.6801936629108407
MAE Train:  0.5046096230155906
R^2 Train:  0.5201288447335942
```

Through these experiments, the model performs best when all three predictors are in the set contributing to the prediction.

3.3 Final Model/Results

The final SGD model was run using the parameter values described in the section 3.2.9 along with the full set of predictors outlined in section 3.2.10. The model was evaluated using the training and testing Mean Square Error (MSE), Root Mean Square Error (RMSE) Mean Absolute Error (MAE) and R^2 values.

When looking at the coefficients of the final model, SM1_Dz had a value of 0.292 meaning that for each unit increase of the standardized SM1_Dz value the standardized LC50 concentration went up by 0.292, which did not change from the base model. Similarly GATSli and MLOGP have coefficient values of -0.0979 and 0.552 respectively, meaning for each unit increase of those standardized values the standardized LC50 went up by their respective coefficient values. The base final also had an intercept of 0 thanks to the fit_intercept parameter because it does not make sense for the concentration of the predictors to be 0, thus the regression line should pass through the origin.

Below is a chart and report of the base model's performance on the training and testing sets:


```
Test Set Metrics
MSE Test:  0.5690280796802941
RMSE Test:  0.7543394989527554
MAE Test:  0.5516076525643188
R^2 Test:  0.5021750494205044
----------------------------------------------------
Training Set Metrics
MSE Train:  0.46266341906406644
RMSE Train:  0.6801936629108407
MAE Train:  0.5046096230155906
R^2 Train:  0.5201288447335942
```

4. OLS Model

4.1 Base Model

When running the base OLS mode (on the training data)l on all the parameters the first metric we looked for was R-squared and adjusted R-squared value. These values tell us how well our data is fitting the actual model. The difference between the two is that adjusted R-squared accounts for how many terms are in the model and will lower if there are useless terms that are not adding much to the model. If the model fits well these numbers will be very close to one. Both these values are low and not close to one, meaning that our data does not fit this model well.

When looking at the F-statistic, however, we can see that the value is relatively high. Its corresponding P score is very low. The F statistic in this situation means that there is strong reason to believe that there is a relationship between the coefficients and the target (LC50 concentration). In other words, at least one of the coefficients (Sm2-Dx, Gatsli, or MLOGP) have a relationship to the LC50 concentration.

The AIC and BIC both are numerical values that are used to determine model performance. These can provide good ways to compare the models. Higher values for these scores.

When looking at the coefficients SM1_Dz has a coefficient value .2889. This means for each unit increase of the standardized SM1_Dz value the standardized LC50 concentration went up by .2889. Similarly GATSli and MLOGP have coefficient values of -.1083 and .5371 respectively, meaning for each unit increase of those standardized values the standardized LC50 went up by their respective coefficient values. The constant value coefficient refers to the intercept where the model starts.

The standard error tells us how precise each of the coefficients are in predicting the target. We want these values to be as low as possible.

When examining the t values with their corresponding p values we can see in the e that SML1_DZ and MLOGP have very high t values and very low p values. This means that there is a statistical likelihood that these values actually have a relationship between the LC50 concentration. The constant and GATSli have very low t values however. These metrics point to the possibility that these values actually do not actually relate to LC50. We will explore this option in our trimmed model. The actual output of the model is shown below.

OLS Regression Results

Dep. Variable:	LC50	R-squared:	0.520
Model:	OLS	Adj. R-squared:	0.518
Method:	Least Squares	F-statistic:	261.1
Date:	Tue, 20 Sep 2022	Prob (F-statistic):	1.02e-114
Time:	21:52:07	Log-Likelihood:	-750.20
No. Observations:	726	AIC:	1508.
Df Residuals:	722	BIC:	1527.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.0019	0.025	-0.074	0.941	-0.052	0.048
SM1_Dz	0.2889	0.026	11.171	0.000	0.238	0.340
GATS1i	-0.1083	0.028	-3.824	0.000	-0.164	-0.053
MLOGP	0.5371	0.029	18.309	0.000	0.479	0.595

Omnibus:	97.501	Durbin-Watson:	1.913
Prob(Omnibus):	0.000	Jarque-Bera (JB):	288.689
Skew:	0.658	Prob(JB):	2.05e-63
Kurtosis:	5.795	Cond. No.	1.70

4.2. Trimmed Models

When trimming the model we decided to compare the model two ways. We first took away the GATSli coefficient and then we took off both the Gatli coefficient and the intercept constant. When seeing how the model changed we see that in both cases the model R-squared values and adjusted R-squared values decreased from the base case. The F statistic and corresponding p value improved in the trimmed models, however, with the model with no constant having the lowest p values and highest f values. The AIC and BIC values also improved within the trimmed models with the trimmed model that contained the constant having the highest values.The standard errors mostly stayed the same throughout the models, but there were slightly lower MLOGP standard errors with the trimmed models. Finally the t values for the trimmed values improved from the base model, with the model with no constant performing the best.

Ultimately all these metrics point to the trimmed models improving on the base model.

OLS Regression Results

Dep. Variable:	LC50	R-squared:	0.511
Model:	OLS	Adj. R-squared:	0.509
Method:	Least Squares	F-statistic:	377.2
Date:	Tue, 20 Sep 2022	Prob (F-statistic):	6.37e-113
Time:	21:52:09	Log-Likelihood:	-757.48
No. Observations:	726	AIC:	1521.
Df Residuals:	723	BIC:	1535.
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.0031	0.026	-0.120	0.905	-0.053	0.047
SM1_Dz	0.2937	0.026	11.268	0.000	0.243	0.345
MLOGP	0.5856	0.027	21.934	0.000	0.533	0.638

Omnibus:	99.784	Durbin-Watson:	1.921
Prob(Omnibus):	0.000	Jarque-Bera (JB):	347.934
Skew:	0.621	Prob(JB):	2.80e-76
Kurtosis:	6.156	Cond. No.	1.25

OLS Regression Results

Dep. Variable:	LC50	R-squared (uncentered):	0.511
Model:	OLS	Adj. R-squared (uncentered):	0.509
Method:	Least Squares	F-statistic:	377.7
Date:	Tue, 20 Sep 2022	Prob (F-statistic):	4.49e-113
Time:	21:53:15	Log-Likelihood:	-757.49
No. Observations:	726	AIC:	1519.
Df Residuals:	724	BIC:	1528.
Df Model:	2		
Covariance Type:	nonrobust		

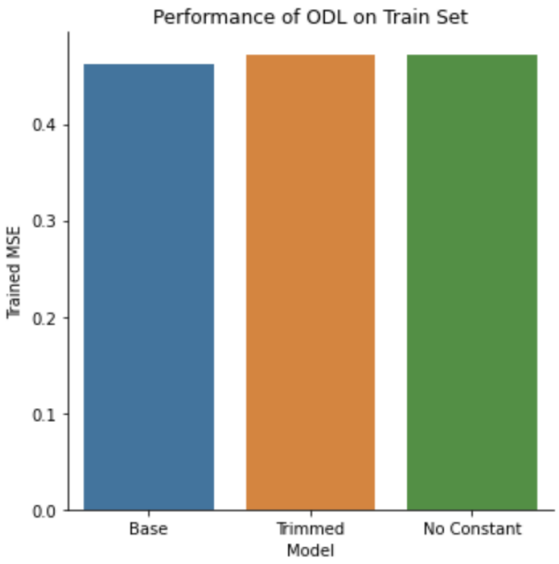
	coef	std err	t	P>|t|	[0.025	0.975]
SM1_Dz	0.2937	0.026	11.276	0.000	0.243	0.345
MLOGP	0.5856	0.027	21.949	0.000	0.533	0.638

Omnibus:	99.783	Durbin-Watson:	1.921
Prob(Omnibus):	0.000	Jarque-Bera (JB):	347.923
Skew:	0.621	Prob(JB):	2.81e-76
Kurtosis:	6.155	Cond. No.	1.25

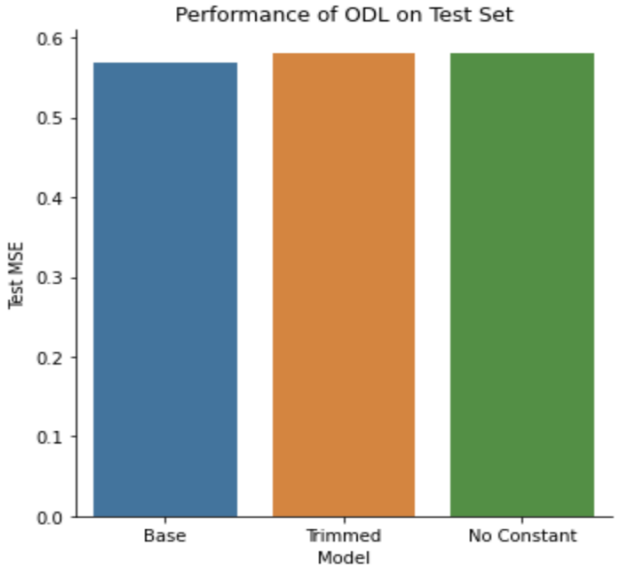
Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

4.3 Model Performances

To compare the results of the different models we decided to do MSE on both the training and test data. We are looking for the values to be lower. Doing this we got the following results. We can see that in both the test data and the train data the mean square error for the base model was lower than the others. The trimmed and no constant models had very similar MSE with the trimmed performing slightly better on the train set and the no constant performing slightly better on the test set.



	Model	Trained MSE
0	Base	0.462457
1	Trimmed	0.471823
2	No Constant	0.471833



	Model	Test MSE
0	Base	0.569364
1	Trimmed	0.580636
2	No Constant	0.580552

5. Conclusion

Overall both models had very similar MSE values with the SGD model performing slightly better. What we also learned was that SM1_Dz and MLOGP have the highest t values. This means that these two values are likely to have a relationship with LC50 concentration. The intercepts in the models tend to have low T values and actually reduce the MSE in some of the models. Overall none of the models fit particularly well, and other models will need to be evaluated for future work on this topic.