Predicting Fuel Efficiency by Minimizing MAPE with Weighted Least Squares

Leopold Marx
ISDS 7103

October 7th, 2020

1 Introduction

Transportation is critical in modern day society. Nearly everyone uses some sort of transportation daily as an intermediate to go about our lives. In most of America, to get to the places you need in a reasonable amount of time, the most common choice of transportation is the automobile. Vehicles come with various properties including sizes, engines, drive trains, air conditioning, make, year, etc. One of the most important features of an automobile is the cost which includes fuel efficiency.

In this project, we will be trying to predict fuel efficiency based on different properties of 1500 automobiles including the variables listed in Table 1.

Table 1: Available Variables

Variable	Description
`origin`	Origin of the car
`maker`	Auto Maker
`model`	Model name
`yr`	Year
`cyl`	Number of Cylinders
`2wd`	2=Two wheel drive, 4=Four wheel drive
`auto`	1 = Automatic, 0 = otherwise
`p/s`	1 = Power steering, 0 = otherwise
`a/c`	O=Air Conditioning, X = otherwise
`fro`	1 = Front wheel drive, 0 = otherwise
`wght`	Weight (pounds)
`disp`	Displacement (cubic inches)
`hp`	Horsepower
`lngth`	Length (inches)
`wdth`	Width (inches)
`wb`	Wheel base (inches)
`reli`	Reliability index (from Consumer Reports)
`fid`	Firm identification number
`dom`	yes = U.S. built, no = imported
`eur`	Y = European Model, N = otherwise
`sales`	Sales
`price`	List price (dollars)
`markup`	Estimated mark up (thousands of dollars)
`mpg`	Miles per gallon

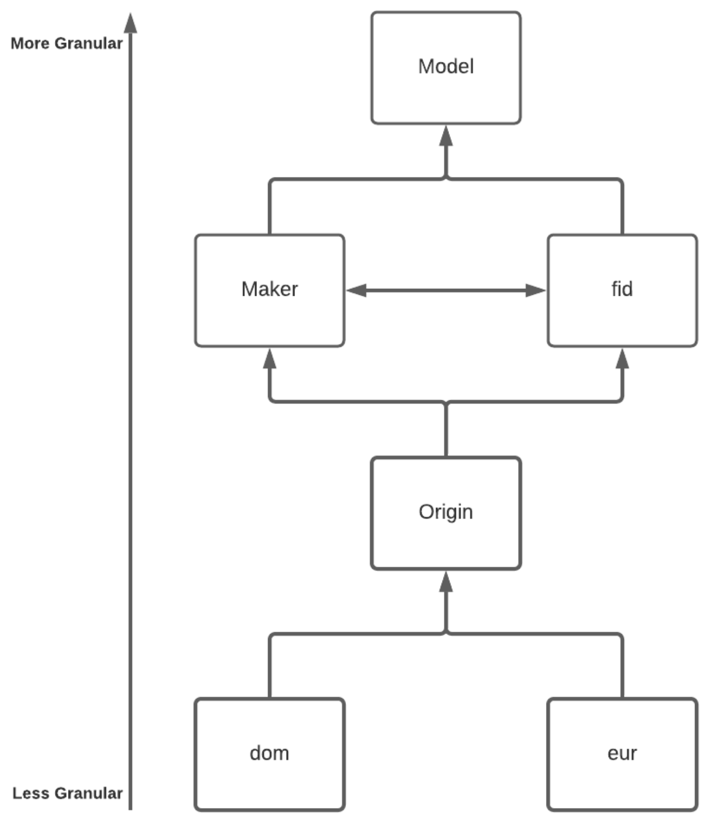
2 Analysis and Methods

2.1 Data Preprocessing

2.1.1 Redundant Categorical Variables

To start off this analysis, we should look at which variables are redundant. One form of redundancy is if any variables are subsets or copies of others. For example, `dom` and `eur` is a subset of `origin`, `origin` is a subset of `maker`, and `maker` is equal to `fid`. This implies that if we use `maker`, then we should not use `fid`, `origin`, `dom`, and `eur` because `maker` already covers all the cases of the subsets. If we use `origin`, we should exclude `maker` because it is a super set and would provide more information that `origin` alone. Similarly if we want to use `dom` and `eur` in our analysis, then we should exclude `maker` and `origin`. See Figure 1 for a visual on which variables are subsets or supersets of others.

Figure 1: Illustration of variable subsets



It is to note that `model` is excluded from the analysis because it is too specific of a category for trying to predict fuel efficiency. `model` has 439 unique values which means there are would only be an average of 3.4 observations per model for 1500 observations. If we were to use `model`, it would likely result in over fitting. It also would likely not necessarily explain the patterns in the data and would probably regurgitate the information in the training set. This model would also be irrelevant if future models of cars are released.

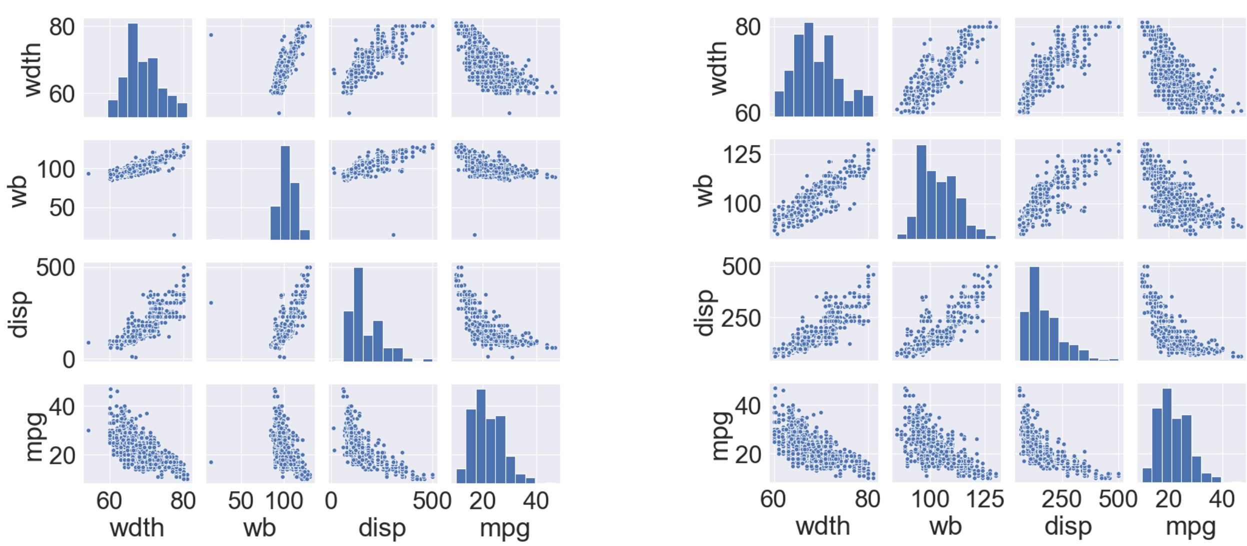
2.1.2 Outliers

In Figure 2, we can see how if we remove a couple outliers for `wdth`, `wb`, and `disp`, we can drastically improve the understanding of each variable with respect to `mpg`. Observations were removed if `wdth` was less than 55, `wb` was less than 40, or if `disp` was less than 40. The observations that fell into any of these categories have these id values: 672, 744, 1056, 1462.

Figure 2: Pair Plots Before and After Removing Outliers

(a) Raw Data

(b) After removing outliers



In addition to obvious outliers in the pair plots, observations that had large influence from a weighted least squares regression were also removed. More specifically, there was a cluster of observations that had a large Cook's Distance. Observations were removed with these id values: 159, 1166, 1281, 1391, 1392, 1442, 1498, 1499. The total amount of observations removed is 12 which is less than 1% of the original dataset.

2.1.3 Transformations

As we can see in Figure 2 (b) above, the variables in the pair plots are not linear with respect to `mpg`. To improve the linearity and normalcy of variables, a Box-Cox transformation on `wght`, `disp`, `hp`, `lngth`, `wdth`, `wb`, `price`, and `markup`. The lambda values that optimized the Box-Cox transformations for each variable are -0.09466148, -0.43856908, -0.20070702, -0.27736606, -2.31571202, -1.64987019, -0.33435441, and -0.44343978 respectively. These lambdas are selected by stabilizing variance and minimizing skewness.

2.1.4 Dummy Variables

As seen in Table 1, some of the variables are nominal meaning we need to map them to $n-1$ binary variables where n is the number of categories. This process would need to be completed for `origin`, `maker`, `a/c`, `fid`, `dom`, and `eur` since there is no definite order to the categories. Keep in mind, this would only need to be done if you want the variable in the model. The $n-1$ binary variables would more explanation if a car from Japan on average has a higher fuel efficiency rating.

2.1.5 Polynomial Interaction

Because interpretability of the model is not necessarily the goal for this project, we can try to squeeze some more explanation out of the variables by multiply multiplying them together. This also allows the dummy variables to interact with various variables to act like switches to have different weights based on various categories. For example, if cars from the US have better fuel efficiency with the same properties, we can explain that with an interaction.

In my analysis, I found polynomial interaction of orders of 1, 2, and 3 to be sufficient. A polynomial interaction on the order of 3 would return new variables of all polynomial combinations of the variables with less than or equal to 3. For example, a polynomial interaction of x and y on the order of 3 would return 1, x, y, xy, x^2, y^2, x^2y, xy^2, x^3, y^3.

2.1.6 Train/Test Split

The data set was partitioned into a training and a testing partition. 70% of the samples were placed in the training dataset at random and the remaining 30% was placed in the testing dataset. This is a critical step to make sure the model does not over-fit to the training data. In my modeling, I used the test set as a truly external validation set that allows us to gauge if the model is over fitting the to the data.

2.1.7 Test Set Imputing

The test set was altered by Dr. Chun to simulate imperfect data. Since we need to report a mpg value for each of the observation in the test set, we cannot simply remove these observations. I personally went through each column and checked if there were any anomalies. For example, in the `cyl` column, There were some cells with "acht" cylinders. For those who do not know German, "acht" means "eight" in English so I mapped this cell to the value 8. Another anomaly I found had to do with spelling. Some values of `maker` and `origin` had misspelled names as well which needed to be fixed.

Another type of anomaly in the test set was missing data. For these observations that had one or more blank rows, I imputed these cells based on the `year`, `model`, `cyl`, and `hp` if available. Through doing some research online, I tried to match the missing value with information I could find online about the automobile.

2.2 Modeling

2.2.1 Objective Function

One of our project requirements is to minimize the mean absolute percent error (MAPE). MAPE is defined as follows:

$$L_{\text{MAPE}}(y, \hat{y}) = \frac{\sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{y_i}}{n} \tag{1}$$

The MAPE loss function differs from most common loss functions, like sums of squared error, by summing the percent error of the true y value. For example a car that has 10 mpg and predicted to have 11 mpg would result in a 10% error. However, if we look at a 50 mpg car predicted to do 45 mpg, there would still be a 10% error. MAPE weights errors based on their true y values unlike least squares where everything is distance based. In other words, sums of squared error treats a 1 mpg error the same no matter what actual mpg value is. MAPE has tighter bounds for lower y values and looser bounds for larger y. This is an appropriate loss function for this problem because the marginal gain of mpg decreases as the fuel efficiency increases.

From my research, minimizing MAPE alone is quite tedious because there has to be non-linear or iterative optimization methods to find the optimal betas for the model. Minimizing sum of squared error would not do a proper job picking the optimal weights for minimizing MAPE. However, if we could weight each error by the true value, we could get pretty close to MAPE. This is where my interest with weighted least squares came to interest. Below is the loss function for weighted least squares:

$$L_{\text{WLS}}(y, \hat{y}) = \sum_{i=1}^{n} w_i (y_i - \hat{y}_i)^2 \tag{2}$$

If we use $\frac{1}{y_i}$ as w_i in the WLS loss function, we can get pretty close to the loss function of MAPE with the only difference being a squared distance in the numerator instead of the true distance. Although not exact, this will act quite similarly to how the MAPE objective function operates. It is to note that we can ignore the value of n because it is a constant with respect to the betas. Minimizing $L_{\text{MAPE}} \cdot n$ will also minimize L_{MAPE}

$$L_{\text{MAPE}^*}(y, \hat{y}) = \sum_{i=1}^{n} \frac{(y_i - \hat{y}_i)^2}{y_i} \tag{3}$$

This small change in the objective function allows us to go from a non-linear optimization to an analytical solution for optimal betas. Using a weighted least squares approach to minimizing MAPE is To use all the powerful regression tools, I will be fitting my model with MAPE* (3) and selecting variables based on minimal MAPE (1).

2.2.2 Variable selection

Variable selection is very important especially when using polynomial interaction that results in several thousand new variables. My algorithm for variable selection is a variant of forward selection. For each potential variable added to the model, I run a 5 fold cross validation. For each fold, I fit the weighted least squares model (which minimizes MAPE*, not MAPE) with the training data from the 4 folds and with the selected variables along with the new candiate. Once the model is fit, then calculate the true MAPE value. The average of the 5 true MAPE values is calculated. Which ever candiate variable has the minimum average MAPE value, it is added to the selected list. This process continues until the steps between average MAPE values is less than 0.00001 or there are no sufficient variables left.

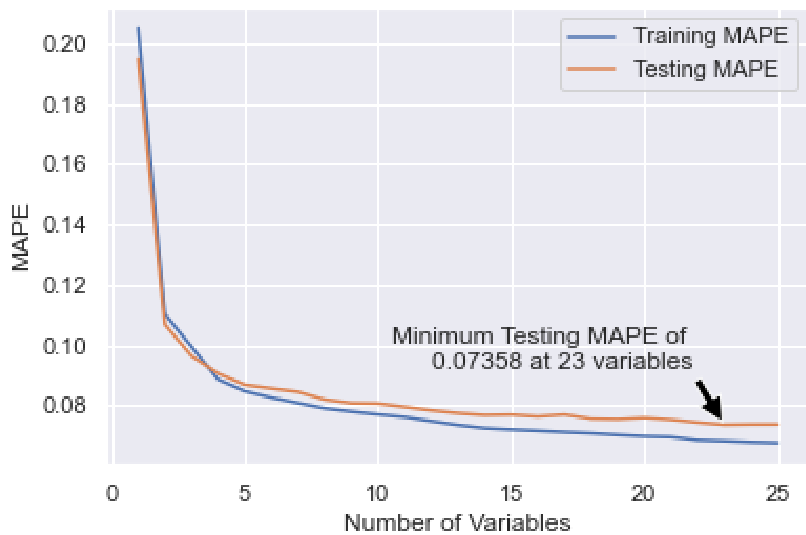
This algorithm is a sound way to rank which variables should be in the model based on minimizing MAPE however, there are some things to note about this algorithm:

- This algorithm only considers variables with significant addition (p value smaller than 0.1)
- Even though the data is being validated through 5 fold cross validation, I found that this selection process alone caused over fitting since every point was used for training and testing. This is why we need a truly external dataset discussed in section 2.1.6.
- Once this algorithm terminates, the test set is used to prune most recently added variables (next section).

2.2.3 Variable Pruning

The variable selection algorithm mentioned above is essentially a ranking of the variables that should be in the model. We still need to find out what is the best subset of variables that minimizes an external testing dataset. To do this, I plotted the first x variables selected vs the testing MAPE score. Please see Figure 3 for an example.

Figure 3: Number of Variables vs MAPE for Order 2 Polynomial Interaction with `maker`



2.2.4 Finding the Optimal Model

As alluded to in section 2.1.1, we should not be modeling with `maker` and `origin` dummy variables as `origin` is a subset of `maker`. For this reason, I also removed `eur` and `dom` from the analysis as they did not explain much. The choice between choosing `maker`, `origin`, or neither gives us 3 options.

Another dimension of complexity is to change the order of the polynomial interaction. For my analysis, I investigated orders 1 through 3.

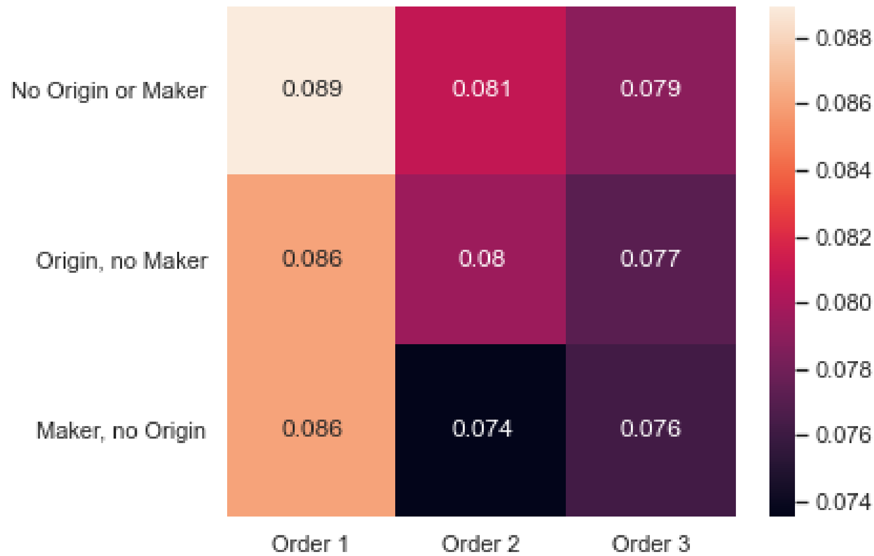
For each of these two dimensions, I ran the variable selection and pruning algorithm and got the following testing MAPE scores in Figure 4. Note that Figure 3 is the pruning graph for the optimal option of Maker, no Origin and Order 2 polynomial interaction.

2.2.5 Notable attempts at improving Modeling

- PCA: a PCA transformation on all continuous variables was made and it made testing MAPE worse for all models in contention.

- Factor Analysis: a factor analysis transformation was preformed on all variables and it made testing MAPE worse for all models in contention.

Figure 4: MAPE Test Rate by Polynomial Interactions and Varying Granularity of `maker`



- Gradient Descent: I tried setting up a non-linear optimization model with minimizing MAPE directly by changing the weights for each variable. However, I ran into issues with picking a proper step parameter. Since MAPE had an absolute value in the function, it make gradient descent pretty tricky. I even tried a variant of gradient decent where the weight on the gradient decreases as the number of iterations increases.

- Inverse transformation: initially it looked like a lot of the continuous variables had an inverse relationship. This transformation preformed worse with the same modeling techniques.

3 Conclusion and Summary

Throughout the duration of this project, I found the best model was a weighted least squares model with $w_i = \frac{1}{y_i}$. I variables I used included polynomial interaction on the order of 2 all continuous variables along with the `maker` dummy variables. This model resulted in the parameters listed in Table 4 and a validation MAPE of 7.3582%. Please note that some of these variables have been transformed as per section 2.1.3.

I think it is amazing that with a deeper understanding of the mathematical background of the problem and objective, we can turn a relatively difficult problem and simplify it with some minor approximations.

If I had more time for this project, I would investigate how weights of $\frac{1}{y^2}$ would affect

Table 2: Variable coefficients for Best Model

Variable	Coefficient
yr^2	-0.040386
yr · hp	-0.014741
yr · wght	-0.030900
1	-280.401115
wght · markup	0.491332
reli^2	0.053356
cyl · maker[is_Volkswagen]	-1.624681
price · maker[is_Fiat]	4.092697
hp · markup	-0.267320
hp · maker[is_Peugeot]	-5.277230
yr	7.012234
wght · hp	0.554768
cyl	-0.443464
price · a/c[is_X]	-0.899088
auto · maker[is_Chrysler]	-0.810913
price · maker[is_Renault]	2.741458
wb · maker[is_Honda]	1.821003
disp · reli	-0.196054
cyl · wdth	0.074755
2wh · maker[is_Volkswagen]	1.231992
hp · maker[is_Mercedes-Benz]	-2.356612
price · maker[is_Mercedes-Benz]	1.601217
yr · maker[is_Yugo]	-0.086524
fro · disp	-0.509213
reli · maker[is_Subaru]	-0.375534

testing MAPE. For potential different models, I would be tempted to try a neural network or a random forest since interpretability does not mater for this project. However, I am not sure if MAPE can be minimized directly with these models.

4 Appendix

id	mpg
1501	26.828484
1502	13.769909
1503	18.725475
1504	17.432641
1505	16.914992

1506	25.136109
1507	16.600491
1508	14.784044
1509	16.412940
1510	11.830328
1511	11.092402
1512	12.247292
1513	15.371246
1514	16.445632
1515	12.033053
1516	14.700930
1517	21.801808
1518	31.325245
1519	15.529454
1520	22.189260
1521	28.078199
1522	11.277818
1523	21.216233
1524	21.758030
1525	17.113876
1526	16.819228
1527	12.193803
1528	16.386612
1529	12.942126
1530	12.329204
1531	15.534906
1532	12.077938
1533	12.149367
1534	16.903197
1535	12.608400
1536	29.005372
1537	17.742746
1538	30.012287
1539	22.377631
1540	26.160226
1541	21.701665

1542	21.697606
1543	13.609190
1544	15.886516
1545	13.634536
1546	17.303798
1547	14.297439
1548	13.189963
1549	18.685494
1550	19.370990
1551	18.678399
1552	18.513595
1553	23.169849
1554	16.384484
1555	16.541216
1556	18.528305
1557	14.555256
1558	18.150100
1559	18.991702
1560	17.671001
1561	29.365873
1562	27.738873
1563	18.895962
1564	14.918625
1565	18.018461
1566	23.107111
1567	20.123219
1568	19.087040
1569	25.222606
1570	15.706504
1571	32.093351
1572	24.176999
1573	16.183802
1574	18.153740
1575	18.643541
1576	18.035895
1577	19.163279

1578	19.314868
1579	24.229888
1580	24.972269
1581	31.803913
1582	25.296026
1583	27.214090
1584	21.768522
1585	31.931643
1586	25.134786
1587	16.727868
1588	30.178244
1589	16.682757
1590	18.506925
1591	19.695544
1592	22.339944
1593	23.319420
1594	19.059221
1595	20.563005
1596	25.386473
1597	27.916464
1598	24.474433
1599	17.870711
1600	16.229352
1601	17.306359
1602	21.714528
1603	22.769572
1604	29.237110
1605	17.700140
1606	22.118941
1607	32.422760
1608	18.257749
1609	22.157254
1610	23.808352
1611	16.605829
1612	20.457259
1613	19.330254

1614	18.419913
1615	27.655155
1616	23.603561
1617	20.025613
1618	32.793190
1619	23.066358
1620	26.460058
1621	15.551051
1622	27.475481
1623	23.890312
1624	20.622163
1625	22.114003
1626	31.293511
1627	17.510841
1628	21.067344
1629	20.339824
1630	17.800619
1631	19.524825
1632	19.099476
1633	19.166512
1634	26.150977
1635	29.871784
1636	19.020296
1637	21.355920
1638	21.809626
1639	31.971506
1640	27.863181
1641	28.037387
1642	21.212217
1643	19.480777
1644	19.249457
1645	17.118105
1646	33.934476
1647	31.858581
1648	31.512076
1649	20.086872

1650	25.170428
1651	31.914343
1652	23.031549
1653	17.963370
1654	16.554232
1655	24.049648
1656	16.583174
1657	29.282714
1658	24.552086
1659	27.489359
1660	18.512612
1661	34.140533
1662	27.051412
1663	33.007750
1664	30.272443
1665	21.779966
1666	24.355765
1667	21.764497
1668	30.894931
1669	33.225113
1670	26.336708
1671	15.323180
1672	23.993137
1673	26.833170
1674	17.162000
1675	24.162273
1676	20.184888
1677	25.775986
1678	24.306375
1679	20.839090
1680	24.306892
1681	24.069800
1682	24.308372
1683	31.617268
1684	19.485290
1685	27.303711

1686	21.762323
1687	30.736916
1688	26.308260
1689	24.684023
1690	17.233517
1691	15.273841
1692	32.583865
1693	29.474066
1694	23.342781
1695	20.528200
1696	23.897073
1697	19.493382
1698	16.682308
1699	19.520631
1700	16.399152
1701	22.789823
1702	18.967877
1703	19.503662
1704	24.266835
1705	24.565123
1706	25.616000
1707	19.223167
1708	19.497073
1709	22.764618
1710	25.390394
1711	21.529897
1712	24.900636
1713	36.625898
1714	25.802632
1715	26.164672
1716	29.869395
1717	14.879683
1718	23.878770
1719	33.153434
1720	23.276309
1721	25.545375

1722	15.907897
1723	22.993480
1724	24.857831
1725	16.900128
1726	19.593910
1727	31.672055
1728	23.785337
1729	18.566589
1730	17.616568
1731	25.427082
1732	18.225415
1733	17.674998
1734	20.975611
1735	25.215507
1736	23.796816
1737	25.595351
1738	24.654261
1739	20.050766
1740	13.732449
1741	23.133180
1742	31.850667
1743	24.449329
1744	22.927824
1745	18.121141
1746	25.458988
1747	18.552481
1748	16.917884
1749	23.249586
1750	18.884238
1751	19.596680
1752	16.716236
1753	17.730137
1754	17.278988
1755	15.903870
1756	22.478222
1757	23.707371

1758	22.684921
1759	25.185670
1760	40.057539
1761	16.730145
1762	22.828839
1763	17.371223
1764	26.520487
1765	23.729116
1766	35.443442
1767	21.697160
1768	18.015934
1769	22.459988
1770	21.769219
1771	19.106112
1772	21.102334
1773	22.736512
1774	24.705799
1775	20.492661
1776	23.361568
1777	16.710644
1778	41.012289
1779	23.079989
1780	22.640161
1781	24.736582
1782	16.351644
1783	19.310536
1784	17.774809
1785	15.014514
1786	16.522516
1787	16.805871
1788	13.492443
1789	24.229188
1790	23.928209
1791	22.439793
1792	20.808353
1793	17.010055

1794	23.163122
1795	22.256593
1796	16.147948
1797	22.677560
1798	17.603160
1799	16.637405
1800	12.978527

Table 3: Summary Statistics of Predicted `mpg` values for the test dataset with best model

Statistic	Value
Mean	21.7161
Median	21.4429
Standard Deviation	5.4795
Min	11.0924
Max	41.0123