SDS 323: Exercises 2

Aaron Grubbs Kaushik Koirala Khue Tran Matthew Tran

Problem 1: KNN Practice

Problem Overview

The data in sclass.csv contains data on over 29,000 Mercedes S class vehicles. Each data row will have information such as the mileage, price, trim, color, and year. This report will attempt to separate and examine **ONLY** two of the trims of the Mercedes S class – *the 350 and the 65 AMG* – and analyze them separately and compare the two analyses. Specifically, for each trim, the predictability of price using only one other variable (mileage) will be assessed. This will be done by using the K-Nearest-Neighbors(KNN) model. For each trim, the best K-value that predicts price (based on the minimization of RMSE between the predicted price and the actual price) will be used to fit the model and look at it one final time.

Data and Analysis Process

The only two trims of vehicles that will be examined, as mentioned before are the *350 trim* and the *65AMG trim.* The following plots depict the distribution of mileage vs price for each of the two trims, foreshadowing the challenge of predicting the price of a vehicle given its mileage.

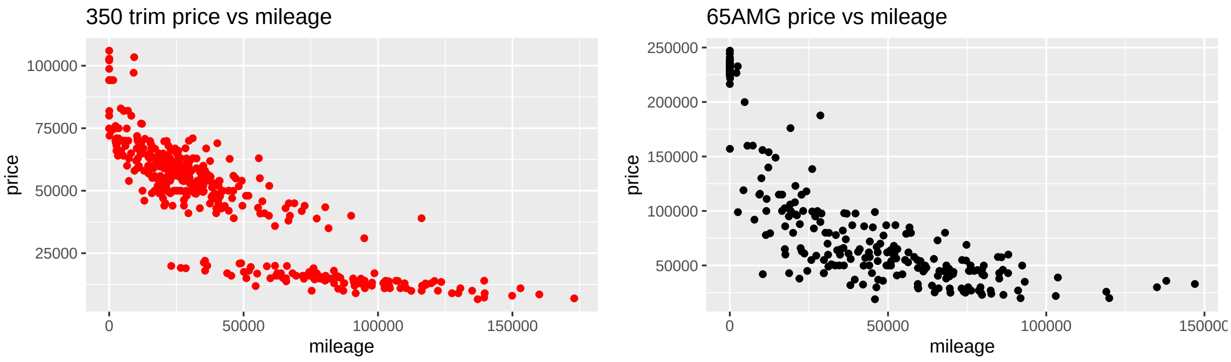


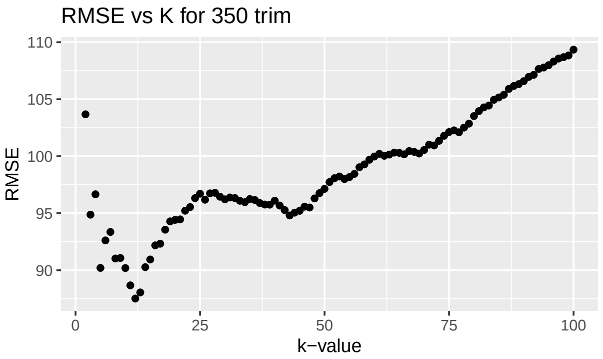
Figure 1: price vs mileage distribution for each trim

In these data distributions, there were 416 datapoints for the 350 trim and 292 datapoints for the 65AMG trim. There were 17 different variables or features for each trim, but the only two variables of interest for this prediction problem are mileage and price.

For each trim, the data depicted above was partitioned into random training and test set several (100) times so that one specific random partition wasn't used to generalize for the whole dataset. Within each random partition of the data, the knn model was trained with the training partition of the data and the calculation of the RMSE calculation was done against the testing set (as compared to the prediction of the KNN model). The k-value within each of the random 100 partitions was varied from 3 to 100 in increments of 1. The RMSE for a specific k-value was summed up across the 100 iterations of the 100 different random test-train partitions. In the end, these summed RMSE-values were averaged and the k-value that minimized RMSE was chosen to fit another partition of the testing data one last time.

Results

The following is the average (across 100 train-test split partitions) RMSE vs k-value plot for the 350 trim:


RMSE vs K for 350 trim

The best value of k to use in the KNN model is 12. The average RMSE across all 100 random train test-splits was 87.5184199.

Now another train-test split will be done on the 350 trim data to run KNN using the best k-value of 12. The following depicts the plot of the fitted model for the optimal k-value of 12:

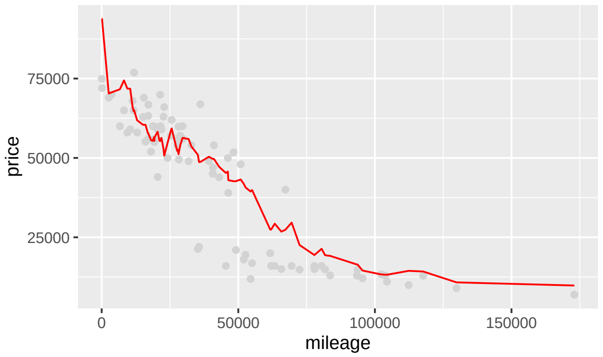
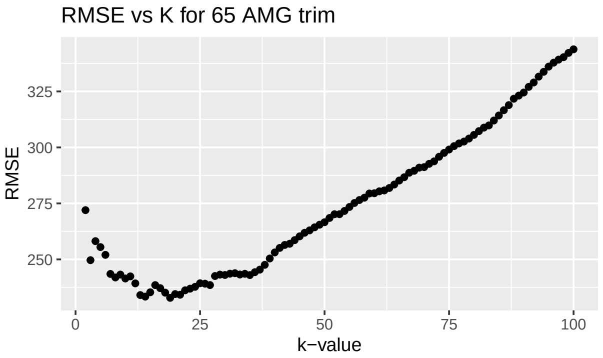


Figure 2: price vs mileage fit using optimal k-value for 350 trim

The following is the average (across 100 train-test split partitions) RMSE vs k-value plot for the 65AMG trim:


RMSE vs K for 65 AMG trim

The best value of k to use in the KNN model is 19. The average RMSE across all 100 random train test-splits was 232.8200266.

Now another train-test split will be done on the 65AMG trim data to run KNN using the best k-value of 19. The following depicts the plot of the fitted model for the optimal k-value of 19:

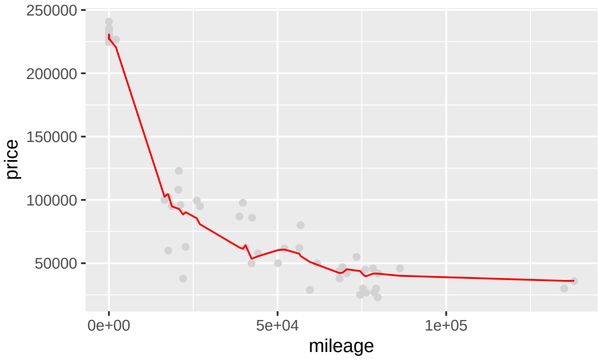


Figure 3: price vs mileage fit using optimal k-value for 65 AMG trim

Conclusions

The optimal k-value for the 350 trim was 12 and the optimal k-value for the 65 AMG trim was 19. The RMSE at the optimal k-value for the 350 trim was 87.5184199, and the RMSE at the optimal k-value for the 65 AMG trim was 232.8200266. The optimal k-value for the 65 AMG trim was higher and it follows that the RMSE at the higher k-value was higher, as a lower k-value means low-bias and a better (lower) RMSE.

The optimal k-value for the 65 AMG trim was probably higher because looking at the two plots on Figure 1, we can see that the 350 trim plot was the more distinctly clustered of the two plots, while the 65AMG data points are more spread out. This means that if a high k-value were to be used for the 350 trim, points from the alternate distinct (and distant) cluster would factor in the prediction, increasing the error. As a consequence, when optimizing the RMSE, the k-value for the 350 trim is lower than the k-value for the 65 AMG trim.

Problem 2: Saratoga Houses

Problem Overview

saratoga_lm.r proposes several linear models with which the prices of properties can be assessed for taxation purposes. The linear models take several features of a property such as the number of bedrooms, number of bathrooms, type of sewage construction, etc,. In this script, the best linear model (as identified by the lowest RMSE) is the "medium" model – a linear model that includes all features of property except its land value, whether it is a waterfront property, its type of sewage facility and whether it is a new construction. This "medium" linear model achieves an RMSE of between 65-67000. This report will attempt to create a better linear model, then using the same variables present in the linear model, attempt to predict the price using a KNN model. The two models will then be compared to see which model better predicts price for taxation purposes.

Data and Analysis Process

The SaratogaHouses dataset has property information on 1728 properties. There are 16 features for each property, including the price of the property. The features are the following:

```
##  [1] "price"           "lotSize"         "age"             "landValue"
##  [5] "livingArea"      "pctCollege"      "bedrooms"        "fireplaces"
##  [9] "bathrooms"       "rooms"           "heating"         "fuel"
## [13] "sewer"           "waterfront"      "newConstruction" "centralAir"
```

Using these features, first a linear model will be constructed that will be compared against the "medium" model found in saratoga_lm.r using the mean RMSE of each models. The performance of these linear models will be evaluated across a 100 separate, random test-train splits, where the RMSE for each out of sample fit will be computed. Then, using the variables in the model superior to the "medium" model, several KNN models will be tested for performance. In order to do this, the data (specifically, the quantitative columns) will first be standardized. After standardization, similar to the linear models after 100 random train-test splits, the average rmse will be calculated for each value of K. The results of the superior linear model will then be compared with the KNN model.

Results

The medium linear model as described above, includes all possible property features **except** whether it is a new property, whether it is a waterfront property, the value of the land, and the type of sewage facility. The better linear model developed in this report does two things, it excludes some features from consideration and prioritizes others by adding a squared polynomial term to those features when regressing those features against price.

The excluded features are:

- centralAir
- heating
- fuel
- fireplaces
- sewer

The squared features are:

- bedrooms
- bathrooms
- livingArea
- age
- landValue
- pctCollege
- rooms

In code, the equation looks like this:

```
lm(price ~ (bedrooms)^2 + (bathrooms)^2 + (livingArea)^2 + (age)^2 + (landValue)^2 +
(pctCollege)^2 + (rooms)^2 +newConstruction + waterfront + lotSize, data=saratoga_train)
```

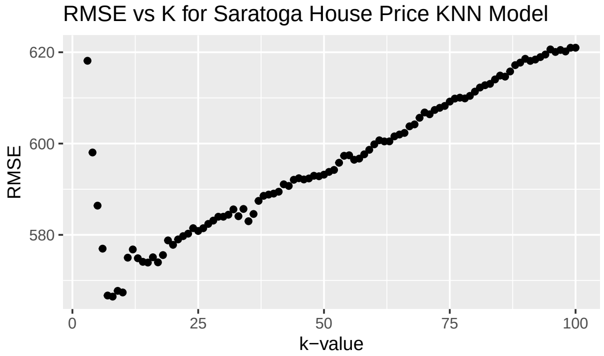
The RMSE values for the “medium” linear model and the better linear model developed in this report are reported below, with the medium model corresponding to V1 and the better linear model corresponding to V2.

```
##       V1       V2
## 65893.74 58459.43
```

The variables that have will go into the KNN model have been determined. The following is the head of the dataframe with all the features needed for KNN. The quantitative feature variables in this table have been standardized.

```
##    price lotSize        age  landValue livingArea pctCollege   bedrooms
## 1 132500    0.09  0.4821608  0.4409565 -1.3694580 -1.9903756 -1.4125065
## 2 181115    0.92 -0.9557035 -0.3499937  0.3194272 -0.4420257 -0.1890422
## 3 109000    0.19  3.5975335 -0.7783061  0.3049096 -0.4420257  1.0344221
## 4 155000    0.41 -0.5106503 -0.4527887  0.3049096 -0.4420257 -0.1890422
## 5  86060    0.11 -0.9557035 -0.5584390 -1.4759207 -0.4420257 -1.4125065
## 6 120000    0.68  0.1055773 -0.5869932 -0.9726425 -3.2484099  1.0344221
##    bathrooms      rooms waterfront newConstruction
## 1 -1.3673127 -0.8813764         No              No
## 2  0.9111023 -0.4496818         No              No
## 3 -1.3673127  0.4137073         No              No
## 4 -0.6078410 -0.8813764         No              No
## 5 -1.3673127 -1.7447656          No             Yes
## 6 -1.3673127  0.4137073          No              No
```

The following is the plot for RMSE vs the k-value used for KNN that have been averaged across the 100 random train-test splits.



The best k-value using knn is 8 and the RMSE at that k-value is 566.4734001.

Conclusions

The best RMSE using the "medium" linear model was $\$ 6.5893743 \times 10^4$. The best RMSE using the optimized linear model was $\$ 5.8459432 \times 10^4$. The best RMSE using KNN as the model was \$ 566.4734001. If the taxing authority decided to use the optimized linear model, its RMSE of a property would improve by \$ 7434.3112608. Additionally, if the taxing authority used the KNN model instead of the improved linear model, its RMSE would improve by $\$ 5.7892959 \times 10^4$, on average.

One consideration the taxing authority has to make is the tradeoff between the simplicity and interpretability of using the linear model and the more complex but more accurate KNN. The coefficients for the linear model developed in this report are:

```
##       (Intercept)          bedrooms         bathrooms        livingArea 
##      1.030676e+05     -1.121457e+04      1.797307e+04      8.125041e+01 
##               age         landValue        pctCollege             rooms 
##     -2.316311e+02      9.370195e-01      9.163015e+01      2.756645e+03 
## newConstructionNo      waterfrontNo           lotSize 
##      4.401181e+04     -1.297896e+05      5.830758e+03
```

With the linear model you simply have to plug in the coefficients and feature values for each property you are considering to get an estimate of the property value. For example, if the property in consideration has 3 bedrooms, the number 3 can simply be plugged in along with the predetermined coefficients. Furthermore, each of the coefficients in a linear model can tell the taxing authority how price estimates change per 1 unit change per particular feature while keeping other features constant. With a KNN model, the number 3 can't simply be plugged in, but rather the standardized score of the feature value has to be plugged in as compared to the training data. This technical barrier for the taxing authority might be worth it considering the significant improvement in accuracy.

Problem 3: Predicting when articles go viral

Problem Overview

The purpose of this model is to attempt to predict whether or not an article will be viral based on numerous characteristics of the model. Virality in this situation is whether or not the article reached 1400 shares on social media threshold.

Data and Analysis Process

The data being used to construct this model was from articles written by Mashable from 2013 and 2014. The variables included in the final model were first checked for significance and then was further reduced based on multicollinearity. The final model predicts shares/virality using number of links, number of pictures, number of videos, number of keywords, average length of words in the article, average shares of referenced Mashable articles, average polarity of negative words, average polarity of positive words, title subjectivity, and title polarity. ### Results The following are the results when regressing first and thresholding second:

```
##
## Call:
## lm(formula = shares ~ num_hrefs + num_imgs + num_videos + average_token_length +
##     num_keywords + self_reference_avg_sharess + avg_positive_polarity +
##     avg_negative_polarity + title_subjectivity + abs_title_sentiment_polarity,
##     data = online_news)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -25086  -2344  -1580   -400 839072
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   3.681e+03  3.994e+02   9.217  < 2e-16 ***
## num_hrefs                     3.643e+01  5.718e+00   6.371 1.90e-10 ***
## num_imgs                      3.152e+01  7.552e+00   4.174 3.00e-05 ***
## num_videos                    3.813e+01  1.456e+01   2.618  0.00884 **
## average_token_length         -7.032e+02  8.606e+01  -8.170 3.16e-16 ***
## num_keywords                  8.744e+01  3.082e+01   2.837  0.00456 **
## self_reference_avg_sharess    2.662e-02  2.407e-03  11.059  < 2e-16 ***
## avg_positive_polarity         1.814e+03  6.793e+02   2.670  0.00759 **
## avg_negative_polarity        -2.872e+03  4.916e+02  -5.843 5.17e-09 ***
## title_subjectivity            3.977e+01  2.568e+02   0.155  0.87692
## abs_title_sentiment_polarity  7.976e+02  3.700e+02   2.155  0.03114 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11580 on 39633 degrees of freedom
## Multiple R-squared:  0.009117,   Adjusted R-squared:  0.008867
## F-statistic: 36.47 on 10 and 39633 DF,  p-value: < 2.2e-16
```

```
##    yhat
## y       0     1
##   0    46 20036
##   1    28 19534
```

```
##
##     0     1
## 20082 19562
```

```
## [1] -0.0126627
```

```
## [1] 0.9750025
```

The following are the results when thresholding first and regressing second:

```
##
## Call:  glm(formula = viral ~ num_hrefs + num_imgs + num_videos + average_token_length +
##     num_keywords + self_reference_avg_sharess + avg_positive_polarity +
##     avg_negative_polarity + title_subjectivity + abs_title_sentiment_polarity,
##     family = binomial, data = online_news)
##
## Coefficients:
##                  (Intercept)                     num_hrefs
##                   -2.050e-01                     1.462e-02
##                     num_imgs                    num_videos
##                    7.314e-03                    -2.776e-03
##         average_token_length                  num_keywords
##                   -1.758e-01                     5.670e-02
##   self_reference_avg_sharess         avg_positive_polarity
##                    8.882e-06                     8.165e-01
##        avg_negative_polarity            title_subjectivity
##                    3.539e-02                     7.498e-02
## abs_title_sentiment_polarity
##                    1.782e-01
##
## Degrees of Freedom: 39643 Total (i.e. Null);  39633 Residual
## Null Deviance:       54950
## Residual Deviance: 54050     AIC: 54070
```

```
##    yhat
## y       0     1
##   0 19424   658
##   1 18217  1345
```

```
## [1] 0.01732923
```

```
## [1] 1.03421
```

When creating a regression before categorizing virality the absolute improvement rate of falls by 1.27% compared to the null which is predicts no articles are viral, with a relative improvement of 0.98%. The overall error of this model is 50.61% with a true positive rate of 99.86% and false positive rate of 99.77%. When categorizing the virality before creating a regression the absolute improvement rate increases by 1.73% with a relative improvement of 1.03% compared to the null. The overall error is about 47.61% with a true positive rate of 6.88% and a false positive rate of 3.28%.

Conclusions

Overall neither model is very good at predicting virality, this makes sense since both models on account for roughly 1% of the proportion of variance for shares/virality. However, thresholding before regressing seems to produce the better model. This may be because it simplifies the output of numerical shares into something binary, viral/ not viral.