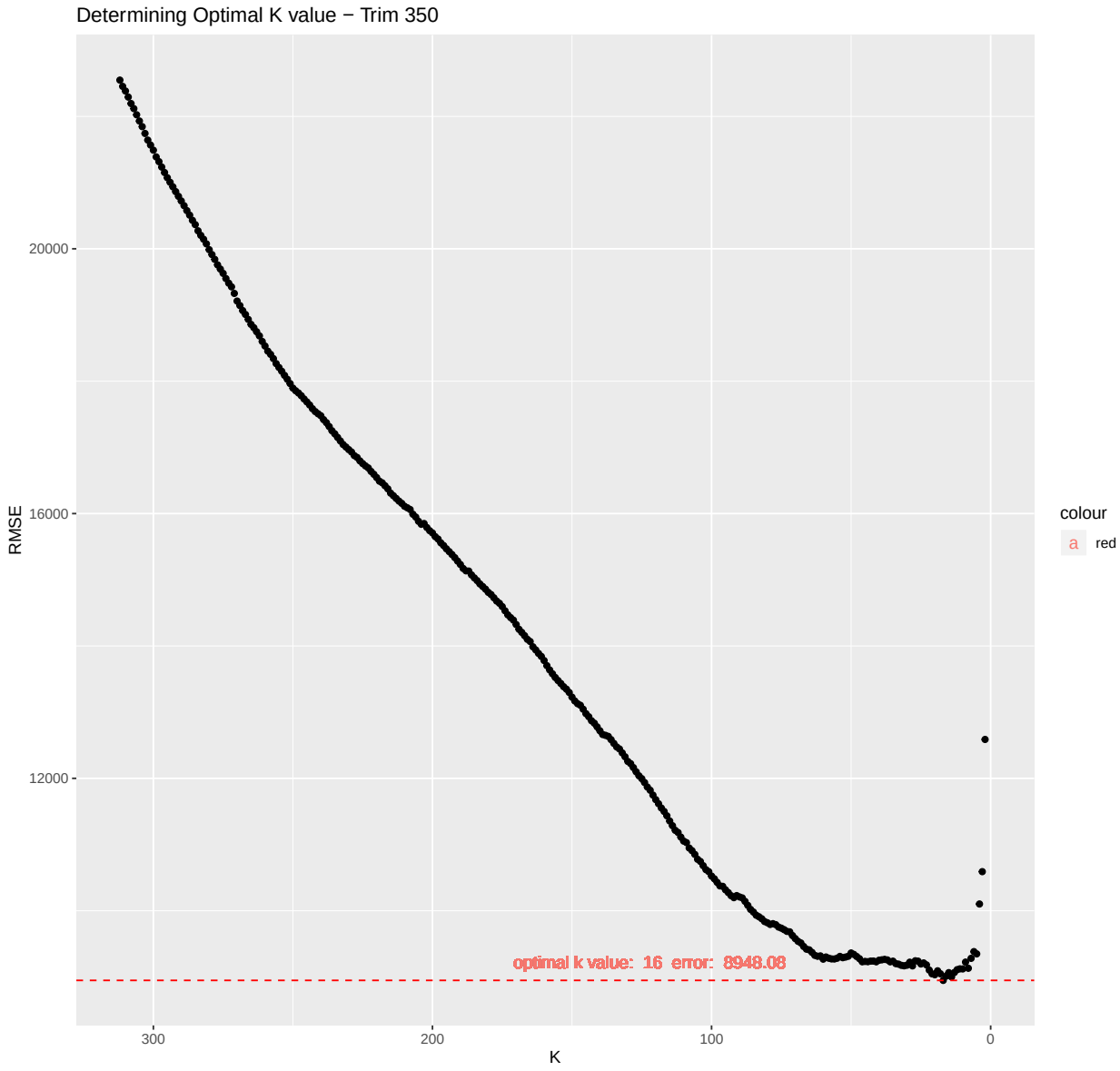
SDS 323: Exercises 2 Report

Nikhil Ajjarapu

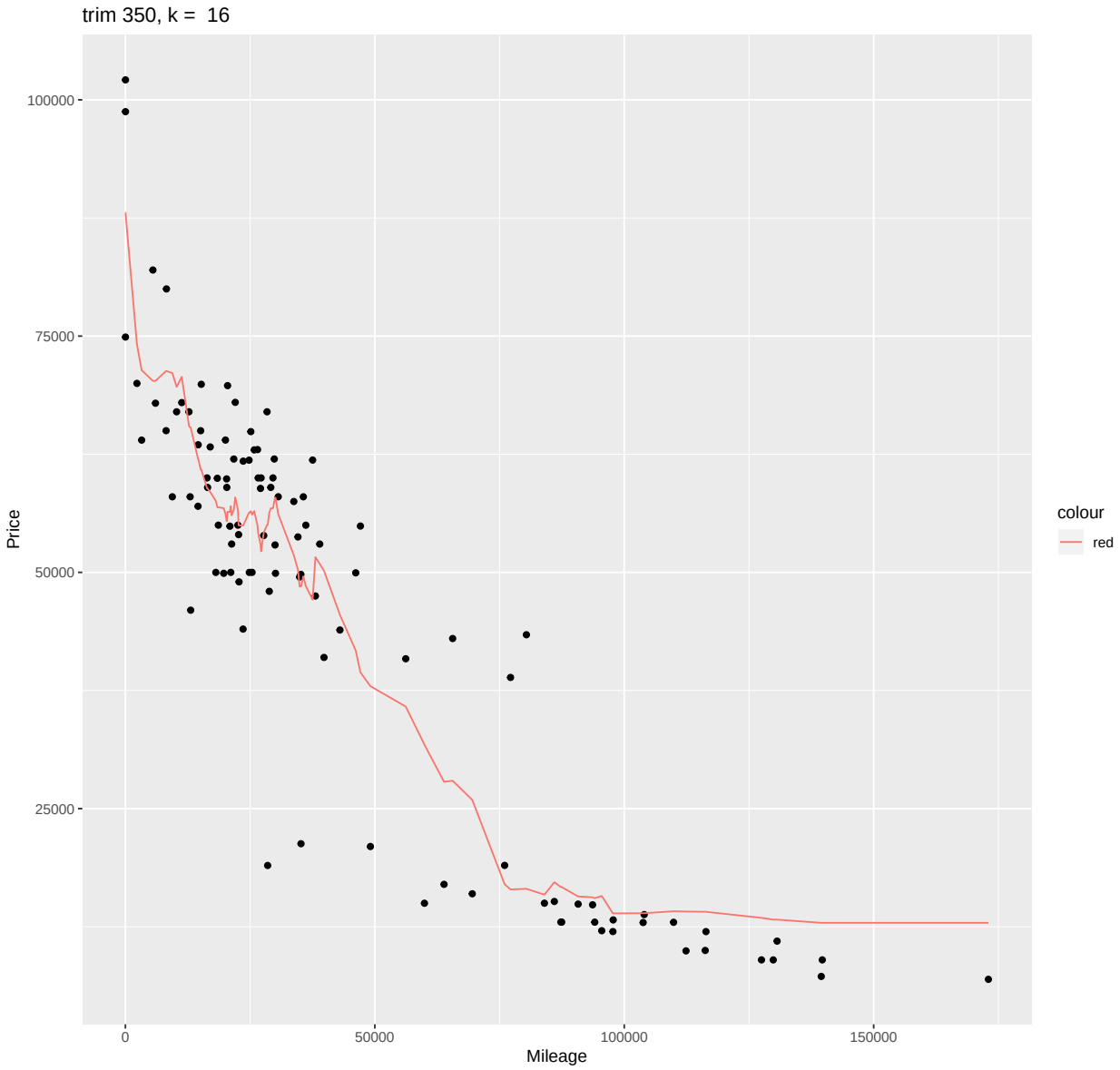
March 12, 2020

Question 1: KNN practice

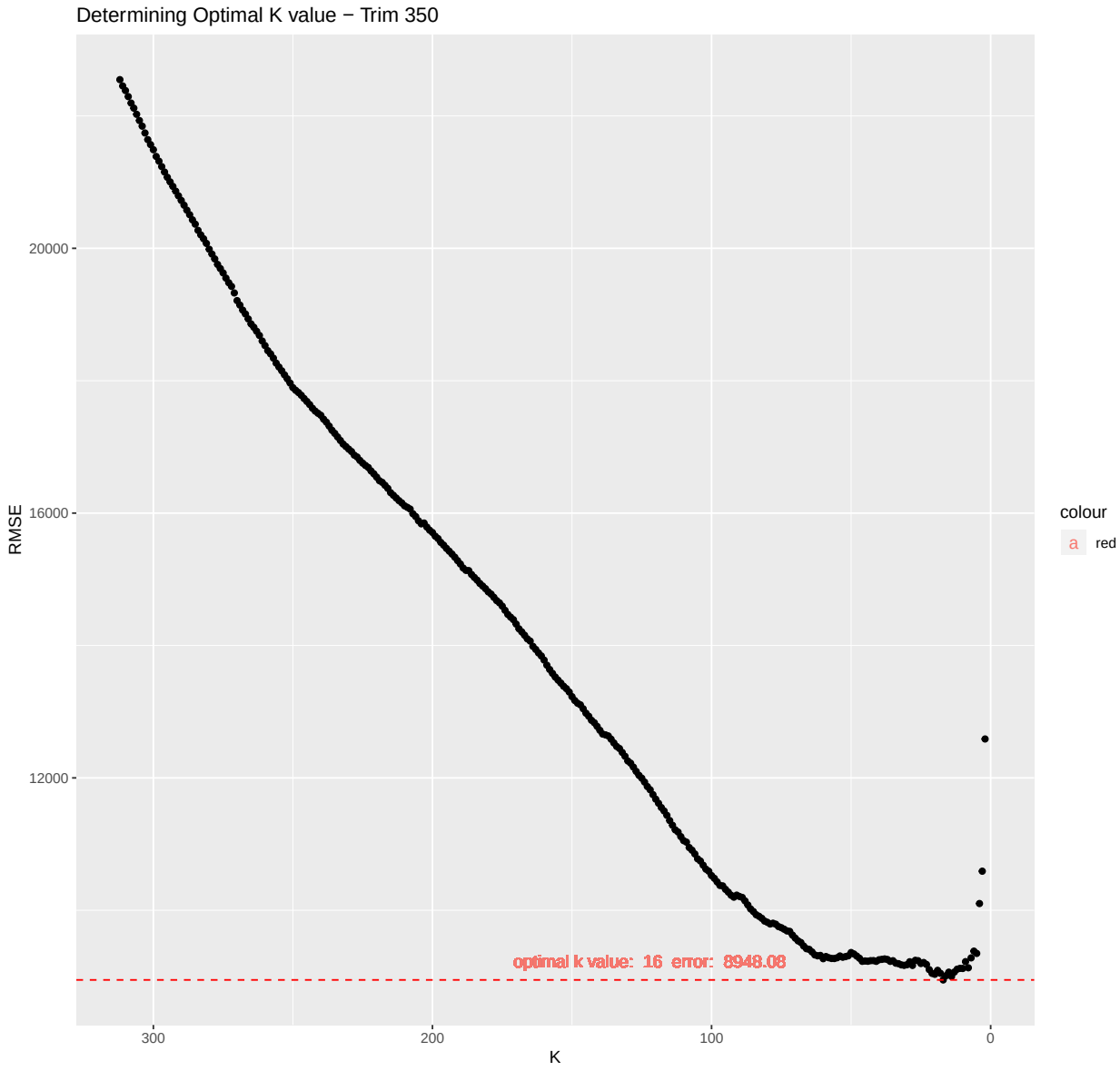
Trim 350: RMSE vs K



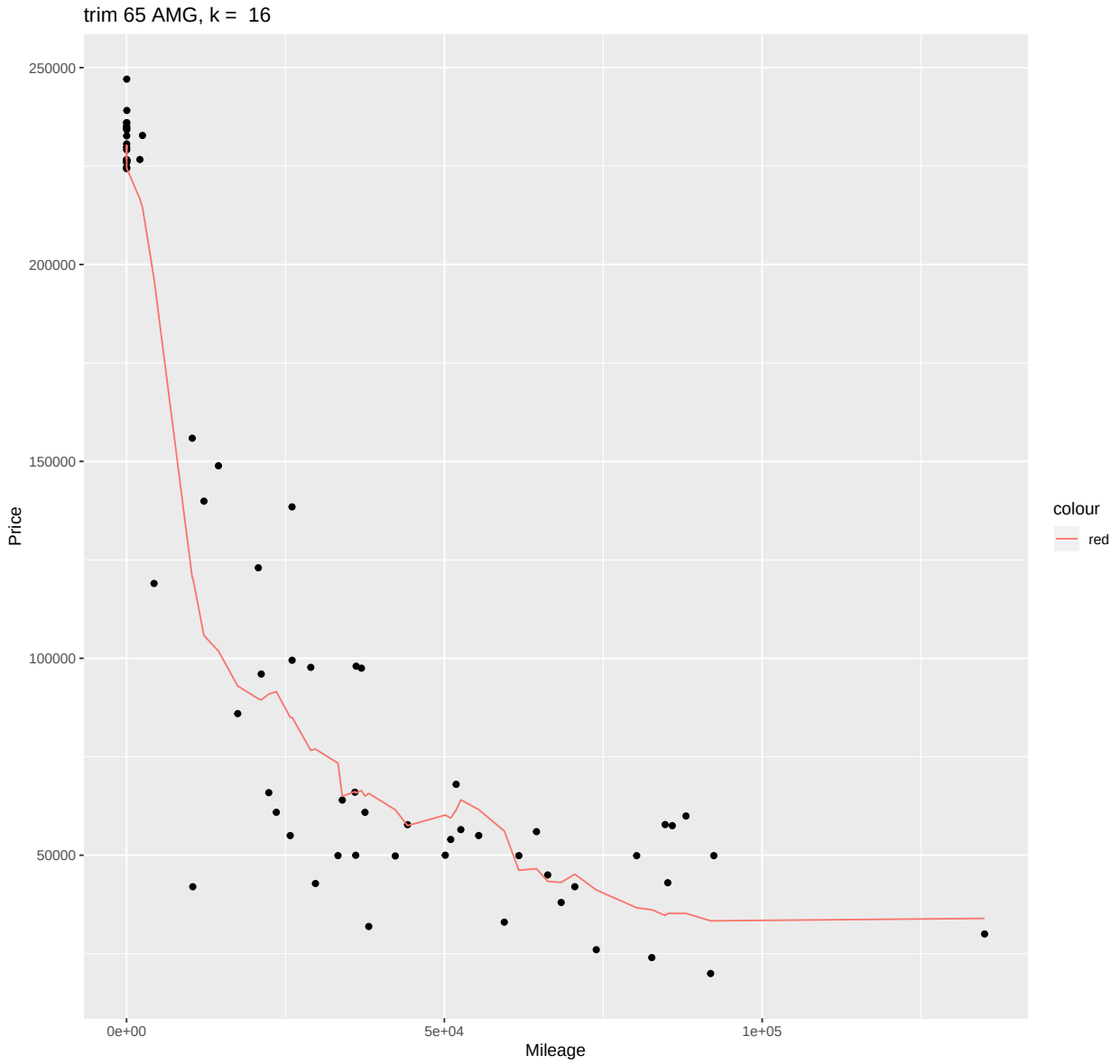
We can see the optimal k for this plot is k = 16. The optimal model looks like this:



Similarly, here is the RMSE vs K plot for Trim 65 AMG:



And the optimal model exists at k = 6, which can be seen below:



Trim 350 yields the larger optimal value of k at k = 16, compared to Trim 65 AMG, with k = 6. We believe the answer to why has to do with the bias-variance tradeoff. Using a higher k means we have much lower variance, as we take into account more of the points. But using a higher k means that we take into account points that may be far away from the x value, causing the prediction to shift away from the true value and creating bias. In other words, using a higher k causes underfitting in the model. Similarly, using a smaller k creates the opposite problem: while we know we will achieve closer to the true value because we only use the values of the data points around the x value, we are more prone to creating a model that can't generalize because it is trained too specifically to the training set. In other words, the curve will overfit the data. In this specific case, we believe that Trim 350 has the higher k because it requires a model that is simpler in nature as the data is far more "clumped" together than Trim 65 AMG's data is. Since it is a lot more dense, the bias is inherently going to be a lot lower, as more of the data points are going to be closer to the true value than would be the case in an average model. Thus, we boost the k-value a little bit to adjust for the decreased bias, and in turn, increased variance.

Question 2: Saratoga house prices

At Ajjarapu and Ajjarapu, the finest consulting firm in the state of New York, we are extremely pleased to present to you our report governing price-estimation models for houses in Saratoga, NY in order to aid your Tax Department in deciding tax policy for the upcoming 5 years.

Linear Modeling

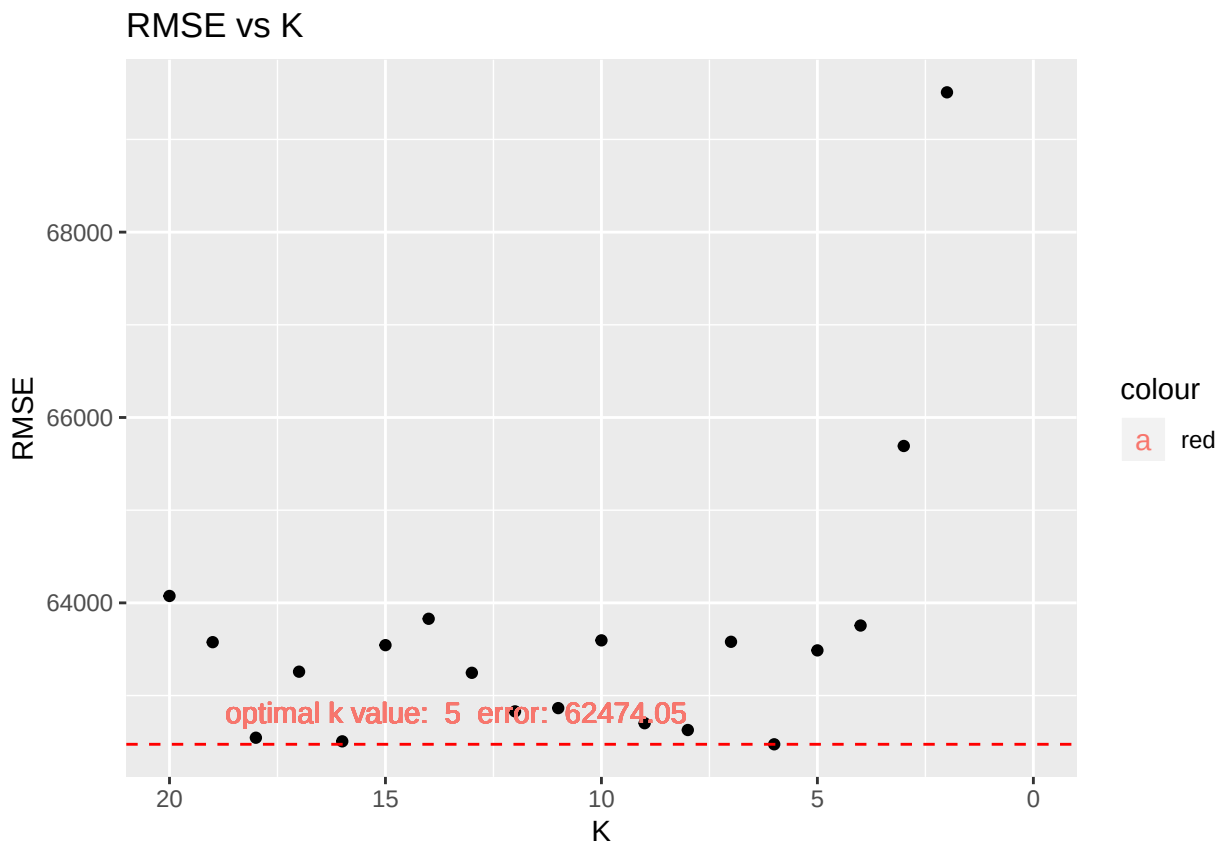
The first step we took was to create a higher quality linear model that given various attributes about the house, could produce an estimated value on the house. After taking a closer look at the current Tax Department model, we realized it was woefully inadequate and decided to update it to create more accurate predictions. Instead of using only 11 features, we decided to create a linear model that used all the features and data given to us. This was the most accurate linear model we could produce from a combination of terms as we let the linear model itself decide which features were more important than others, instead of letting human biases and irrationality cloud judgement and weed out useful features. As shown below, our linear model vastly outperforms the Tax Department's model:

```
## [1] "Tax Department model's average error over 100 iterations:  66408.6092539019"
```

```
## [1] "Ajjarapu and Ajjarapu model's average error over 100 iterations:  58678.5266184085"
```

k-Nearest-Neighbors Modeling

We decided to take it a step farther and see if the k-nearest-neighbors model could perform even better than the previously mentioned linear model. However, due to computing constraints, we were unable to test for many different Ks as it was just not feasible on a Macbook. From the values tested, however, we noticed that the linear model was outperforming the kNN model given the same exact set of features, as evidenced below:



We believe this is due to the complex nature of the data itself. While simpler data that is more categorical in nature can avail itself of the efficiency of the kNN model, we believe values as sensitive as price of a house cannot be estimated solely from "similar" houses for 2 different reasons. First, "similarity" between items in a dataset (in this case, houses) becomes a lot more vague as the number of features go up. While standardizing the data offsets this effect somewhat by equally weighting the different features, it still is an inherent problem of the model as it is difficult to estimate how "close" together two houses are as the dimensionality of the data increases. Secondly, prices are a lot more sensitive to the effects of different variables, and kNN cannot account for this, while linear modeling.

Thus, we conclude that a linear model is the most effective model for housing data, and we strongly reccomend the Tax Department immediately switch to the Ajjarapu and Ajjarapu model for a small amount of $2,000,000 to create more accurate estimates of house price.

Question 3: Predicting when articles go viral

As described in the problem set, there are multiple ways of determining whether an article will go viral given the dataset. We can roughly narrow the problem into two types: regression and classification. Overall, 4 different models were considered for this problem to see which one could produce the best results, and each model was tested with 100 random unique splits to determine a stable average for accuracy purposes. For kNN models, the highest k-value chosen was 20, with 100 random splits per k value, in the interest of computational efficiency as my laptop could not handle larger loads.

Regression - Linear Modeling

The first model tested was linear modeling on the regression problem. After trying various different combinations of features, the model was finalized with a linear combination of every single feature, as it seemed to have the best performance and minimized the error the best, which can be explained by the fact that there was no human decision-making into what features should or should not be used. A complete breakdown of the confusion matrix, overall error rate, true positive rate, and false positive rate can be found below:

```
## [1] "Confusion Matrix: "

## [1] "      yhat"

## [1] "y      0     1"

## [1] "  0  11.29    28.5"

## [1] "  1  9.91    29.3"

## [1] "Overall Error Rate:  0.48620253164557"

## [1] "True Positive Rate:  0.748105927190771"

## [1] "False Positive Rate:  0.71616583773666"

## [1] "Error Rate of Null Model:  0.503670886075949"
```

Regression - kNN model

Another model considered for the regression model was a kNN model with various values for k. The kNN model took on various k values in order to see which creates the optimal model to compare against the 3 other main models. However, the feature set was slightly different: in order to accomodate the scaling of features, various features had to be removed as they were not compatible with the scaled values. A complete breakdown of the confusion matrix, overall error rate, true positive rate, and false positive rate for this model can be found below:

```
## [1] "Confusion Matrix: "

## [1] "      yhat"

## [1] "y      0     1"

## [1] "  0  19.93    19.82"

## [1] "  1  15.1    24.15"

## [1] "Overall Error Rate:  0.442025316455696"

## [1] "True Positive Rate:  0.615655257503011"

## [1] "False Positive Rate:  0.498256189165005"

## [1] "Error Rate of Null Model:  0.503164556962025"
```

Classification - logistic model

Now we will approach the problem as a issue of classification, and this can be achieved with some preprocessing by creating a column "viral" that has binary values for whether an article went viral or not. All relevant statistics can be found below:

```
## [1] "Confusion Matrix: "

## [1] "      yhat"

## [1] "y      0     1"

## [1] "  0  37.2    2.65"

## [1] "  1  2.52    36.63"

## [1] "Overall Error Rate:  0.0654430379746836"

## [1] "True Positive Rate:  0.934295466085044"

## [1] "False Positive Rate:  0.067045086244643"

## [1] "Error Rate of Null Model:  0.504430379746835"
```

Classification - kNN

As discussed in class, kNN can not only be used for regression, but also classification problems, as every datapoint in the set will have a label assigned which kNN can predict, which gives it similar behavior to a clustering algorithm. Full statistics can be found below:

```
## [1] "Confusion Matrix: "

## [1] "      yhat"

## [1] "y      0     1"

## [1] "  0  40.31    0"

## [1] "  1  20.09    18.6"

## [1] "Overall Error Rate:  0.254303797468354"

## [1] "True Positive Rate:  0.484309484829646"

## [1] "False Positive Rate:  0"

## [1] "Error Rate of Null Model:  0.510253164556962"
```

Conclusion

Initally, we can see that every model outperforms the null model, which is ideal as there is no use in engaging in statistical techniques if we can simply predict no article can go viral, which is useful as an initial insight but nothing more than that. We can see the best performing model is the logistic model if we treat the problem as a classification problem. It beats the other 3 models by an incredible amount, having only a ~6% error rate as opposed to the near 50% error rates by the two regression models. This, along with the kNN classification model, shows that thresholding first and classifying clearly outperforms models that regress first and threshold after. Classifying after thresholding is a superior method because it is a lot easier to predict one of two labels as opposed to a specific share amount that each regression algorithm has to do. In addition, there may be various factors that a regression algorithm would prioritize when trying to determine the specific number of shares an article would receive, but those factors may not be needed or relevant when simply determining an article will go viral or not. This would cause the regression algorithm to incorrectly predict "virality" through the proxy of number of shares an article would have.