Classification of Wine Data Using K-Nearest Neighbors

STT 301 Final Project

Fall 2017

Lizzy Burr, Travis Conte, Spencer Crough, Rob Gerth, Victor Ramirez, Adam Rosa

I. Introduction

K-Nearest Neighbors, or KNN, is a machine learning algorithm that can be used to classify clusters within a dataset. This dataset will have *x* known values, or features and is classified into *y* groups. This dataset is divided into two subsets: the test and the training set. KNN is summarized as follows: Choose a user-defined integer *k* closest *x* values with a known classification from the training set. We want to predict the classification of a given point *p* in the testing set. Given that each of the *k* neighbors is classified into a certain group, we classify *p* by the majority of the classifications of its *k* nearest neighbors. For example, say *p* is surrounded by *k* = 5 neighbors, 2 of those neighbors are classified as group A and the other 3 are classified as group B. Because the majority of *p*'s neighbors are of group B, KNN chooses *p* to be classified as group B.

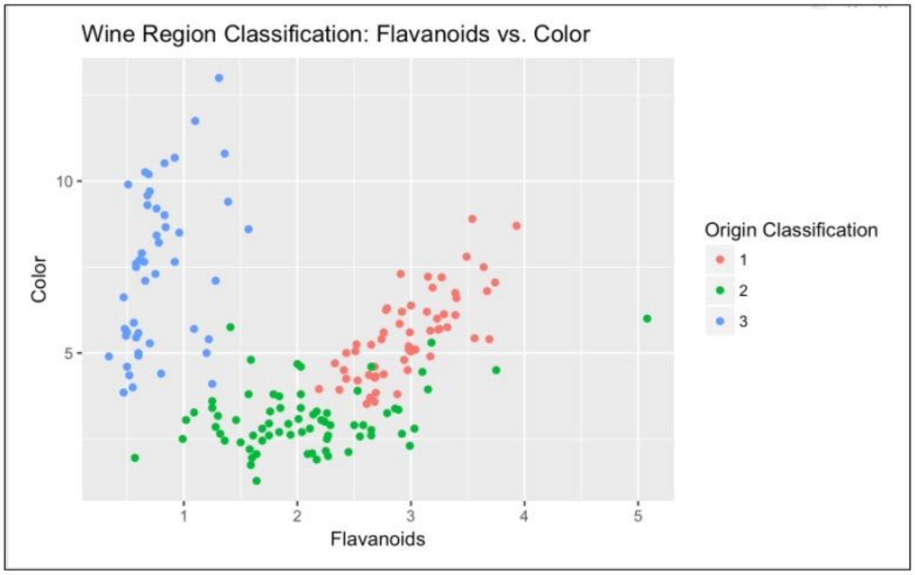
Our group was tasked with applying the KNN algorithm to a dataset of wines in order to classify their origins. First we used only two wine attributes as factors in the knn algorithm, then we applied the knn algorithm to the training data set using all wine attributes as factors. The overall goal was to predict the category of the wine (1, 2, or 3) based off the previously mentioned characteristics.

II. The Data

We used the Wine Data from the UC Irvine Machine Learning Repository. The data is classified into 3 different wine groups of wine cultivars originating from the same region in Italy. These groups are classified as group 1, 2, and 3. The features of this dataset were derived from a chemical analysis of 13 constituent quantities in each of the three groups. These quantities are: alcohol, malic, ash, alcalinity, magnesium, phenols, flavanoids, nonflavanoid, proanyhocyanins, color, hue, OD280/OD315, and proline. We performed KNN on different combinations of features to classify the origins of each wine.

III. Exploratory Analysis

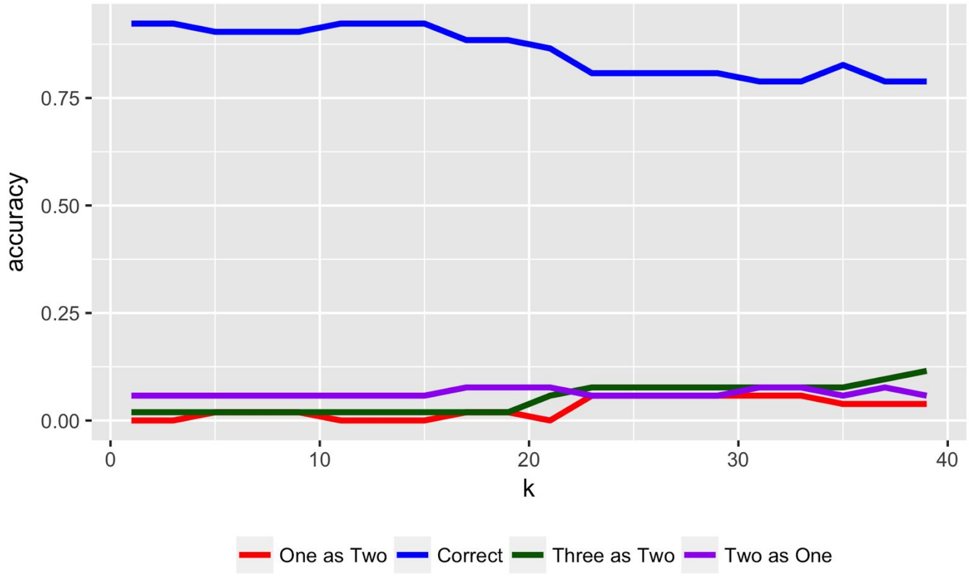
We first plotted wine flavonoids versus color grouped by wine category. The categories were reasonably separated to the point where the knn algorithm should predict categories accurately.



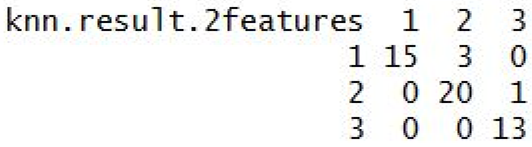
However, we noticed that there was some overlap between classification groups. When examining flavanoids vs. color, there is an overlap between wine groups 1 and 2 with flavanoid values ~0.5-1.5, and significant overlap between wine groups 2 and 3 with flavanoid values ~2-3 and color values 4-5. This is a source of error when applying the knn algorithm to classify the points. As standard in a typical machine learning pipeline, we sought to optimize our classification by finding the optimal *k* values.

IV. Classification With 2 Features

Our next step in this project was to make use of R's knn function and apply it to our wine data set which only incorporate the attributes of flavanoids and color. We first divided our dataset into 70% training set and 30% testing set. We first normalized our dataset so that our features roughly fit the same scale, this helped improve the accuracy of the algorithm. We then fit our training set with various k values and test their accuracy. We plotted each odd k value from 1-39 on the x-axis, with accuracy level represented on the y-axis.



As observed from the graph, the k = 1 provided the highest accuracy (92.3%). On a side note we did not include the other errors because they were always 0%. We also examined the confusion matrix below.

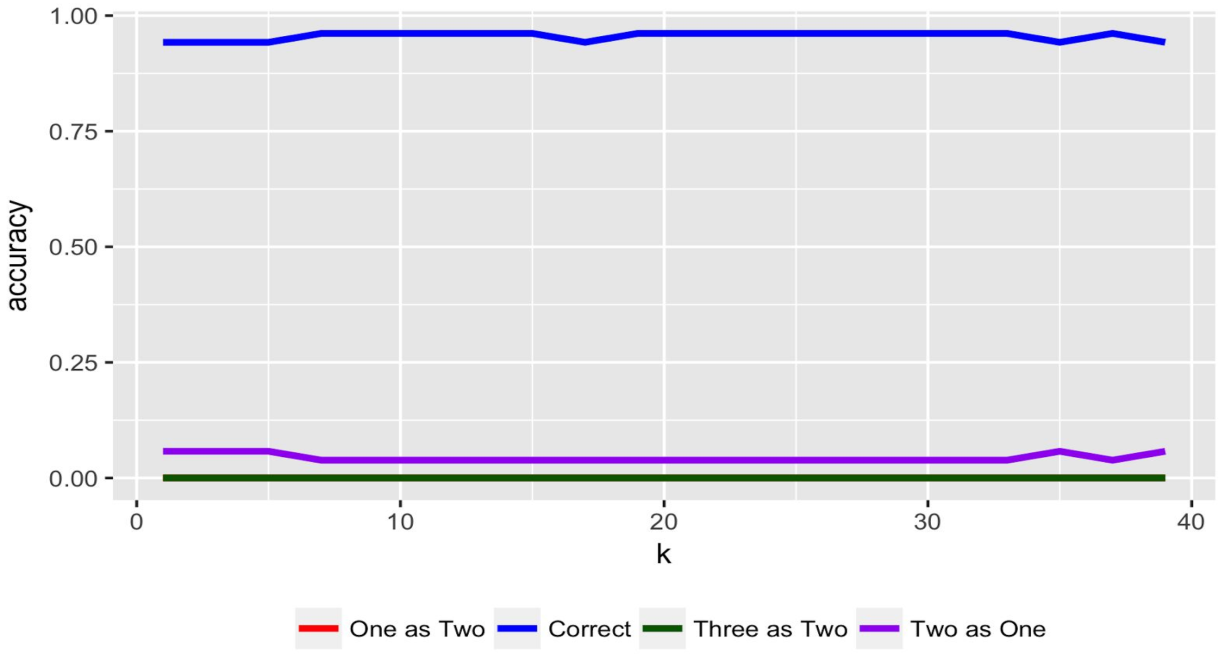


```
knn.result.2features  1  2  3
                   1 15  3  0
                   2  0 20  1
                   3  0  0 13
```

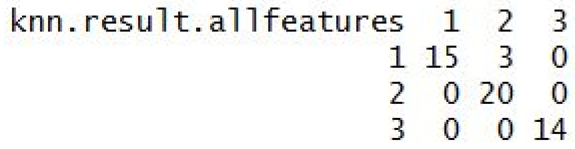
Each column of this matrix represents a wine group and the rows represent what wine each data point is classified as. In column one, we see that all the points were classified correctly. In column two of the twenty-three data points, twenty points were classified correctly and three we falsely classified as wine group 1. Finally in column three only one of the fourteen points were incorrectly classified as wine group 2, instead of wine group 3.

V. Classification With All Features

Following this, we incorporated all 13 wine attributes into the knn algorithm and observed how the accuracy levels are altered. After repeating the process of calling the knn function and storing the outputs, we graphed the results in a similar manner to the one featured above.



Our optimal *k* value 7 yielded an improved ~96.1% accuracy, this is a ~4% improvement in accuracy. One thing to note is at k=17 the correct probability dips down because there is a error as classifying wine type 2 as three, this added about a 2% error. This matches our intuition that using more features to classify a data point should be more accurate. From this plot we also observe that our algorithm is more consistently accurate even if you add more neighbors. We also examined the confusion matrix for *k* = 7.



```
knn.result.allfeatures  1  2  3
                     1 15  3  0
                     2  0 20  0
                     3  0  0 14
```

By examining percentage points, our model seemed to improve noticeably. However, the confusion matrix indicates a much more subtle improvement. We see that the only difference between the confusion matrix for all features vs. 2 features was one correct classification. Now, all 14 wine 3 points are classified instead of only 13.

VI. Conclusion

Using the KNN algorithm, we classified the origins of 3 different wine groups to within 92% for using 2 features and 96% for using all features. However this improvement is marginal and it should be noted that the increased accuracy comes with the drawback of heightened data complexity. This would normally be a concern when working with extremely large data sets, but since our set had a relatively low count of observed values this wasn't an issue. There was not a noticeable difference in software computational time required to apply the knn algorithm, so the increase in accuracy was absolutely worth the increase in complexity for our dataset.

There are many opportunities to expand our model. One possibility is to test different training/testing proportions of our dataset to see which train/test proportion gives the most accurate prediction instead of staying at a fixed 70/30 proportion. Another is to test different machine learning algorithms to classify our wine data and compare those results to our KNN results. Regardless of any of these ventures, this project lays a basic foundation into future tests of machine learning algorithms.

VII. Next Steps

The next steps of the research progress would be to try and improve the algorithm to increase accuracy. We saw that using all of the characteristics of the wine improved the accuracy, however something else to consider would be character selection. When using all of the characteristics from the data set, we use characteristics that may be similar among the three different wines. These similar characteristics would pull the accuracy down. If we used characteristics selection based on characteristics that varied more among all the different types of wine we may see an improvement in accuracy, without making the algorithm too complex.

Another potential way to improve the algorithm would be to apply boosting methods to the knn algorithm. Boosting methods would add weights to the neighbors. If the neighbor helped classify the data correctly it would receive a larger weight, if the neighbor was wrong the weight would lessen. Adding and continuously updating these weights may help improve the accuracy of our algorithm, however we do not want to over complicating the algorithm.

Bibliography

[1] Wine Dataset. https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data

[2] Wine Data Info. https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.names

[3] Melfi, Vince, Finley, Andrew, *R Programming for Data Sciences*