# 9.2 Intro to Machine Learning

Brandon Sams

2/9/2020

These assignments are here to provide you with an introduction to the "Data Science" use for these tools. This is your future. It may seem confusing and weird right now but it hopefully seems far less so than earlier in the semester. Attempt these homework assignments. You will not be graded on your answer but on your approach. This should be a, "Where am I on learning this stuff" check. If you can't get it done, please explain why. Include all of your answers in a R Markdown report.

Regression algorithms are used to predict numeric quantity while classification algorithms predict categorical outcomes. A spam filter is an example use case for a classification algorithm. The input dataset is emails labeled as either spam (i.e. junk emails) or ham (i.e. good emails). The classification algorithm uses features extracted from the emails to learn which emails fall into which category.

In this problem, you will use the nearest neighbors algorithm to fit a model on two simplified datasets.

The first dataset (found in binary-classifier-data.csv) contains three variables; label, x, and y. The label variable is either 0 or 1 and is the output we want to predict using the x and y variables.

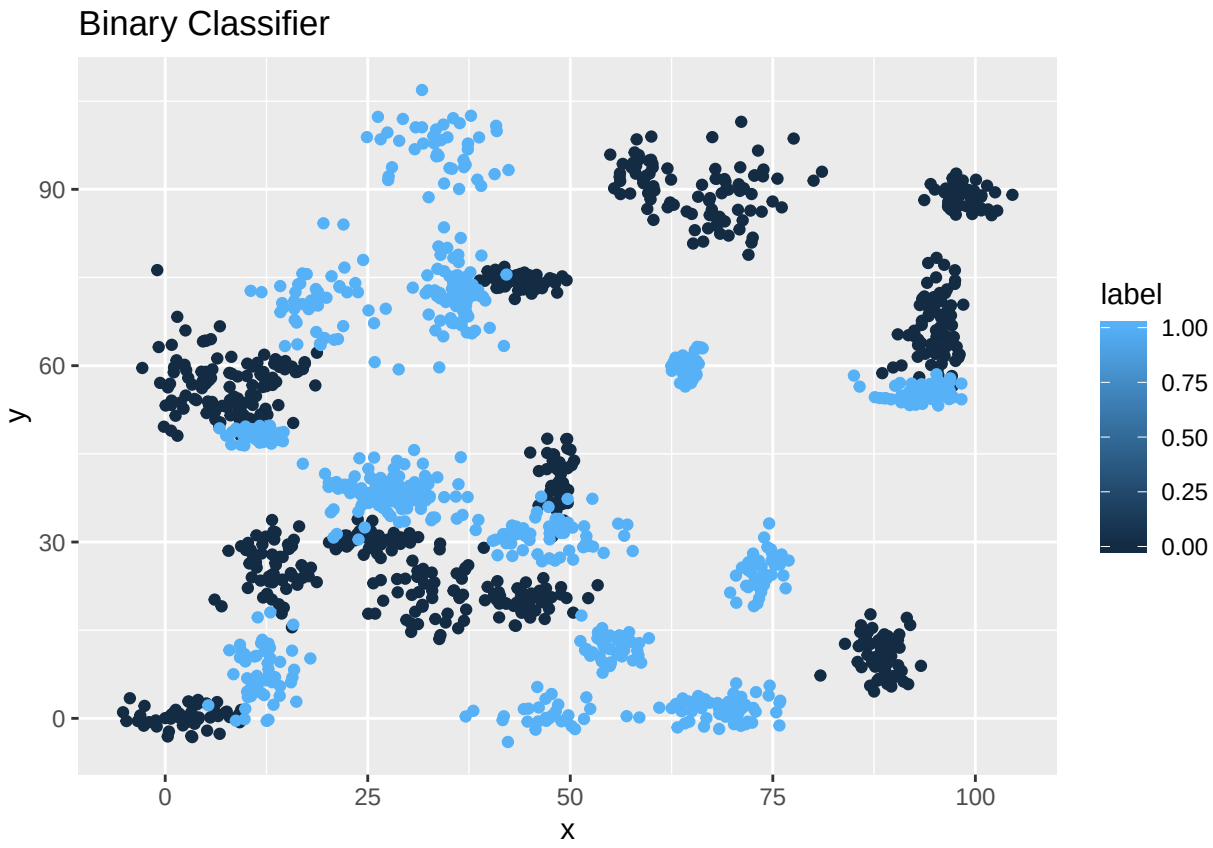
```
binary_classifier <- read.csv(url('http://content.bellevue.edu/cst/dsc/520/id/resources/binary-classifi
```

The second dataset (found in trinary-classifier-data.csv) is similar to the first dataset except for the label variable can be 0, 1, or 2. Note that in real-world datasets, your labels are usually not numbers, but text-based descriptions of the categories (e.g. spam or ham). In practice, you will encode categorical variables into numeric values.

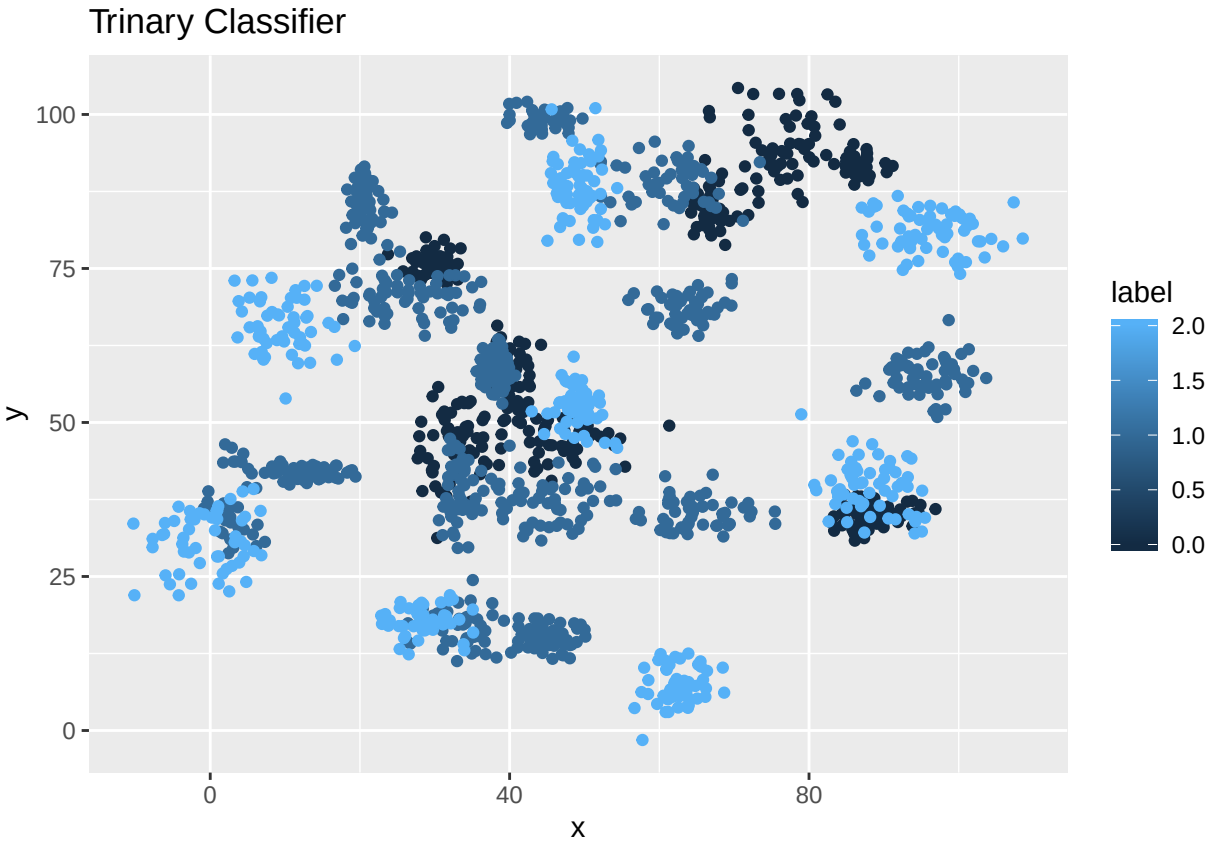
```
trinary_classifier <- read.csv(url('http://content.bellevue.edu/cst/dsc/520/id/resources/trinary-classi
```

**(a) Plot the data from each dataset using a scatter plot.**

```
ggplot(binary_classifier,aes(x=x,y=y,color=label)) + geom_point() + ggtitle("Binary Classifier")
```




```
ggplot(trinary_classifier,aes(x=x,y=y,color=label)) + geom_point() + ggtitle("Trinary Classifier")
```




**(b) The k nearest neighbors algorithm categorizes an input value by looking at the labels for the k nearest points and assigning a category based on the most common label. In this problem, you will determine which points are nearest by calculating the Euclidean distance between two points. As a refresher, the Euclidean distance between two points: $(x_1, y_1)$ and $(x_2, y_2)$ is $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.**

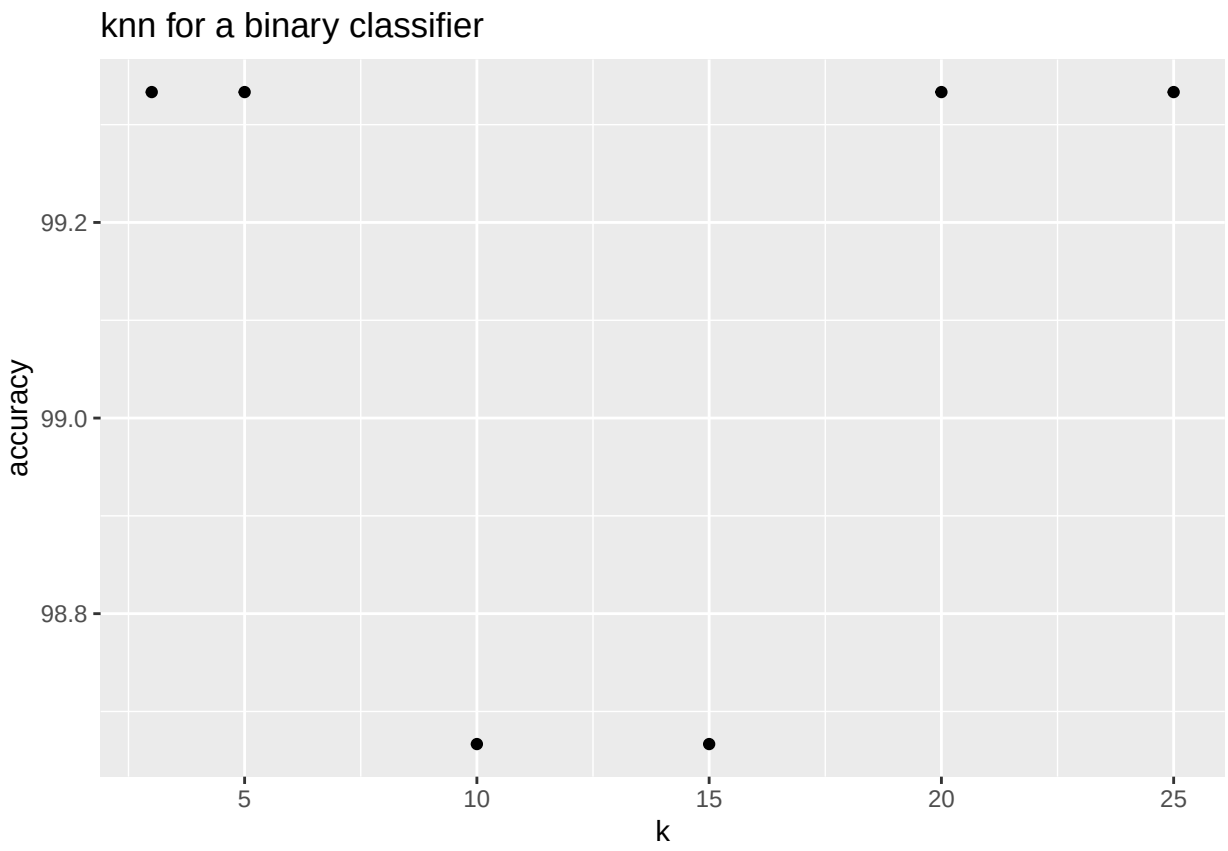
**Fitting a model is when you use the input data to create a predictive model. There are various metrics you can use to determine how well your model fits the data. You will learn more about these metrics in later lessons. For this problem, you will focus on a single metric; accuracy. Accuracy is simply the percentage of how often the model predicts the correct result. If the model always predicts the correct result, it is 100% accurate. If the model always predicts the incorrect result, it is 0% accurate.**

**Fit a k nearest neighbors model for each dataset for k=3, k=5, k=10, k=15, k=20, and k=25. Compute the accuracy of the resulting models for each value of k. Plot the results in a graph where the x-axis is the different values of k and the y-axis is the accuracy of the model.**

*Example code adapted from: https://towardsdatascience.com/k-nearest-neighbors-algorithm-with-examples-in-r-simply-explained-knn-1f2c88da405c*

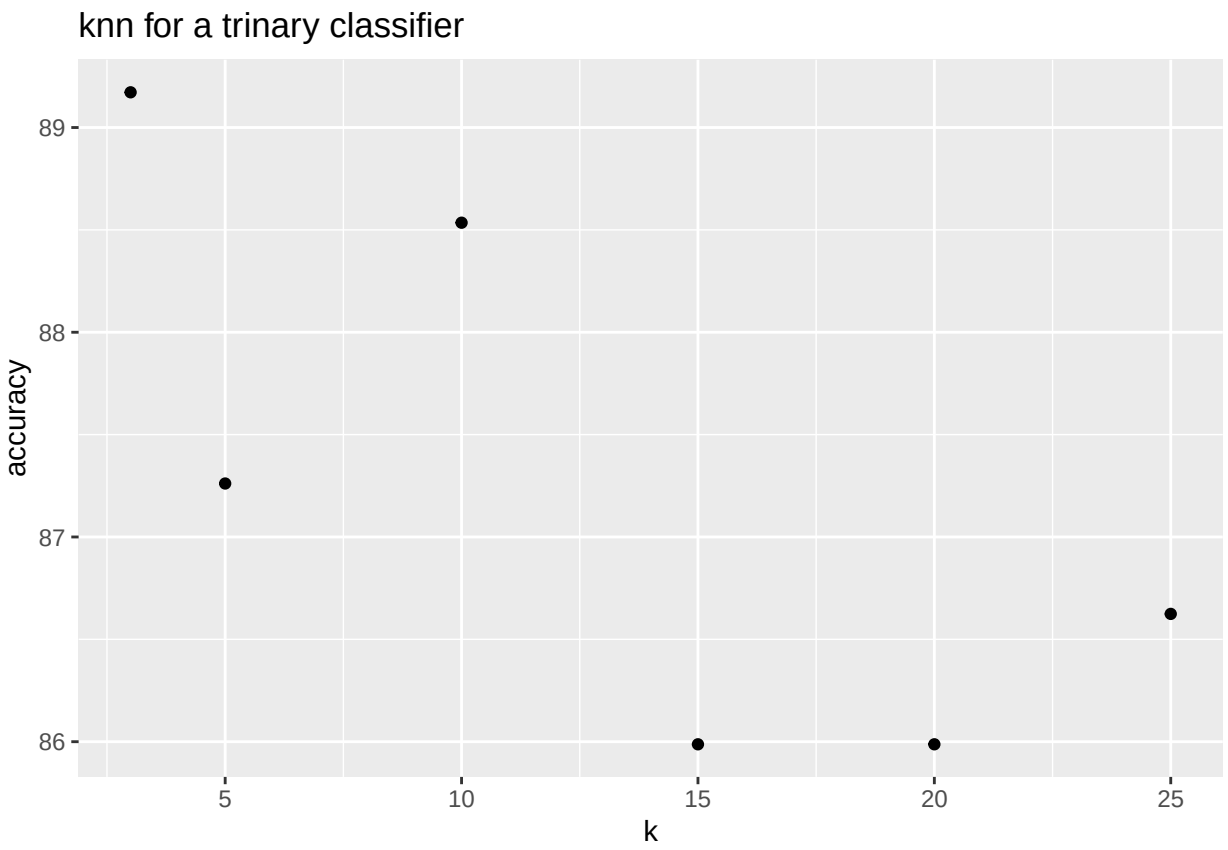
```
ran <- sample(1:nrow(binary_classifier), 0.9 * nrow(binary_classifier))
nor <-function(x) { (x -min(x))/(max(x)-min(x))}
binary_norm <- as.data.frame(lapply(binary_classifier[,c(2,3)], nor))
# extract training set
binary_train <- binary_norm[ran,]
# extract testing set
binary_test <- binary_norm[-ran,]
# extract classifier from training data set
binary_target_category <- binary_classifier[ran,1]
```

```r
# extract classifier from testing data set
binary_test_category <- binary_classifier[-ran,1]
##load the package class
library(class)

k_val <- c(3,5,10,15,20,25)
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
df <- data.frame(k=integer(),accuracy=double())
names(df) <- c("k","accuracy")
for (k in k_val) {
  pr <- knn(binary_train,binary_test,cl=binary_target_category,k=k)
  tab <- table(pr,binary_test_category)
  de <- data.frame(k,accuracy(tab))
  names(de) <- c("k","accuracy")
  df <- rbind(df, de)
}
ggplot(df,aes(x=k,y=accuracy)) + geom_point() +ggtitle("knn for a binary classifier")
```




```r
ran <- sample(1:nrow(trinary_classifier), 0.9 * nrow(trinary_classifier))
nor <-function(x) { (x -min(x))/(max(x)-min(x))}
trinary_norm <- as.data.frame(lapply(trinary_classifier[,c(2,3)], nor))
# extract training set
trinary_train <- trinary_norm[ran,]
# extract testing set
trinary_test <- trinary_norm[-ran,]
# extract classifier from training data set
trinary_target_category <- trinary_classifier[ran,1]
```

```r
# extract classifier from testing data set
trinary_test_category <- trinary_classifier[-ran,1]

k_val <- c(3,5,10,15,20,25)
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
df <- data.frame(k=integer(),accuracy=double())
names(df) <- c("k","accuracy")
for (k in k_val) {
  pr <- knn(trinary_train,trinary_test,cl=trinary_target_category,k=k)
  tab <- table(pr,trinary_test_category)
  de <- data.frame(k,accuracy(tab))
  names(de) <- c("k","accuracy")
  df <- rbind(df, de)
}
ggplot(df,aes(x=k,y=accuracy)) + geom_point() +ggtitle("knn for a trinary classifier")
```




**(c) In later lessons, you will learn about linear classifiers. These algorithms work by defining a decision boundary that separates the different categories. Looking back at the plots of the data, do you think a linear classifier would work well on these datasets?**

The wikipedia article appears to indicate that a boundary is defined between the two sets that is of a linear form. There does appear to be some clustering that is happening in the original data set, but I am not confident that a linear function can reasonably partition the sample into a variety of classifications.