Assignment 9.2 - Intoduction to Machine Learning

edris safari

2/7/2020

Assignment 9.2 - Intoduction to Machine Learning

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. 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.

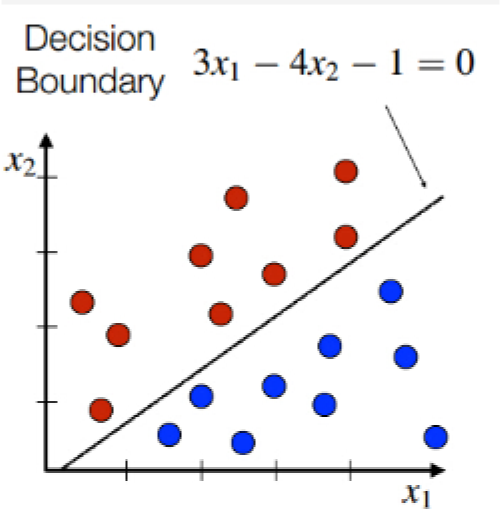
a. Plot the data from each dataset using a scatter plot.

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: p1=(x1, y1) and p2=(x2,y2) is d=$\sqrt{(x_1 - X_2)^2 + (y_1 - Y_2)^2}$.

   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.

c. In later lessons, you will learn about linear classifiers. These algorithms work by defining a decision boundary that separates the different categories.



Decision Boundary

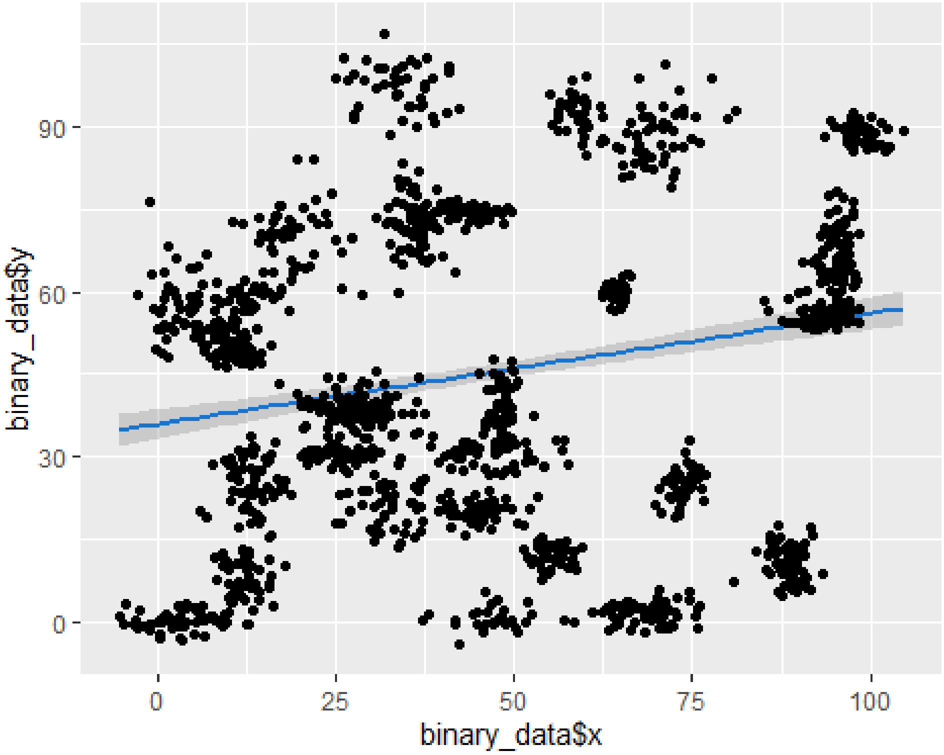
Looking back at the plots of the data, do you think a linear classifier would work well on these datasets?

Import dataset

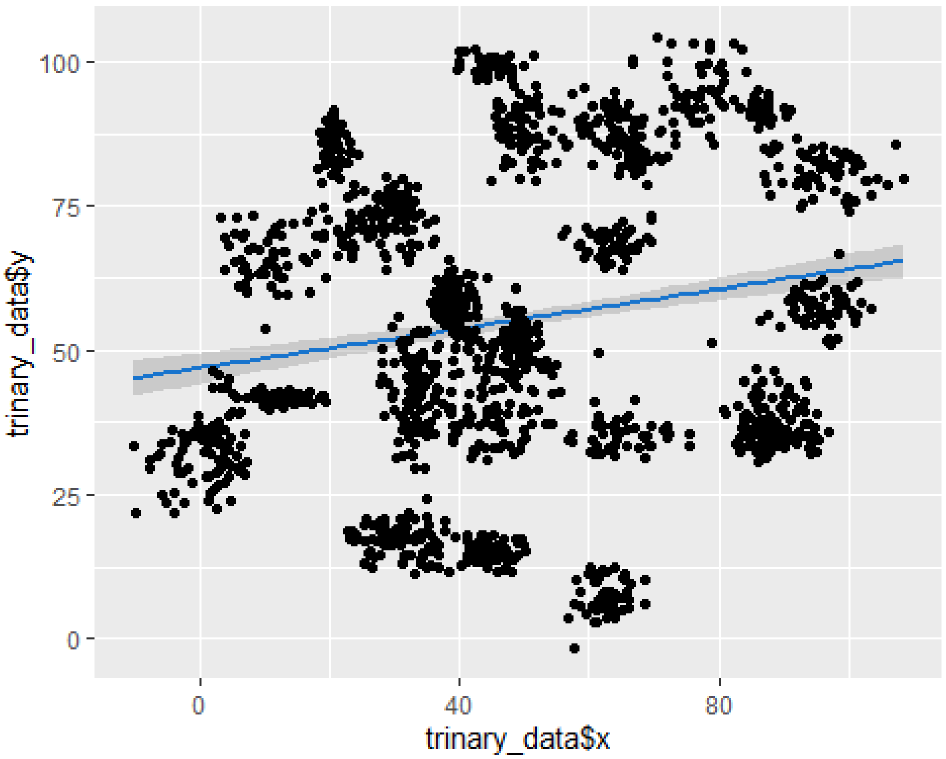
```
## [1] "C:/Users/safar/Documents/GitHub/Safarie1103/Bellevue
University/Courses/DSC520/Week9"

## [1] "C:/Users/safar/Documents/GitHub/Safarie1103/Bellevue
University/Courses/DSC520/Week9"

##          x        y label
## 1 70.88469 83.17702     0
## 2 74.97176 87.92922     0
## 3 73.78333 92.20325     0
## 4 66.40747 81.10617     0
## 5 69.07399 84.53739     0
## 6 72.23616 86.38403     0

##          x        y label
## 1 30.08387 39.63094     0
## 2 31.27613 51.77511     0
## 3 34.12138 49.27575     0
## 4 32.58222 41.23300     0
## 5 34.65069 45.47956     0
## 6 33.80513 44.24656     0
```

Scatter plots

Scatter plot of binary data

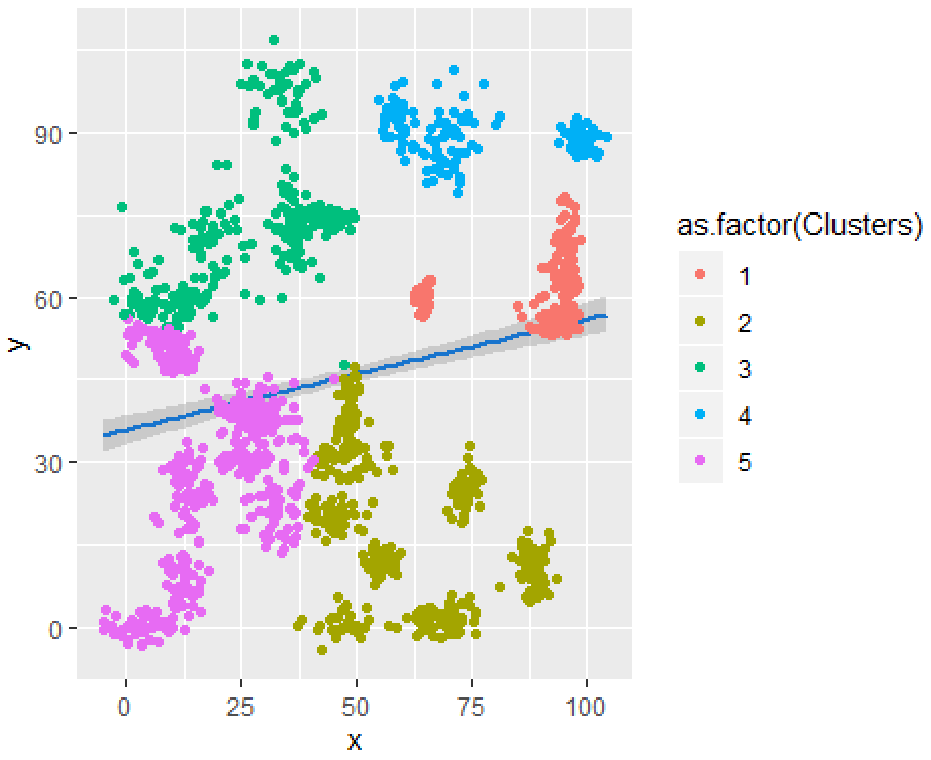


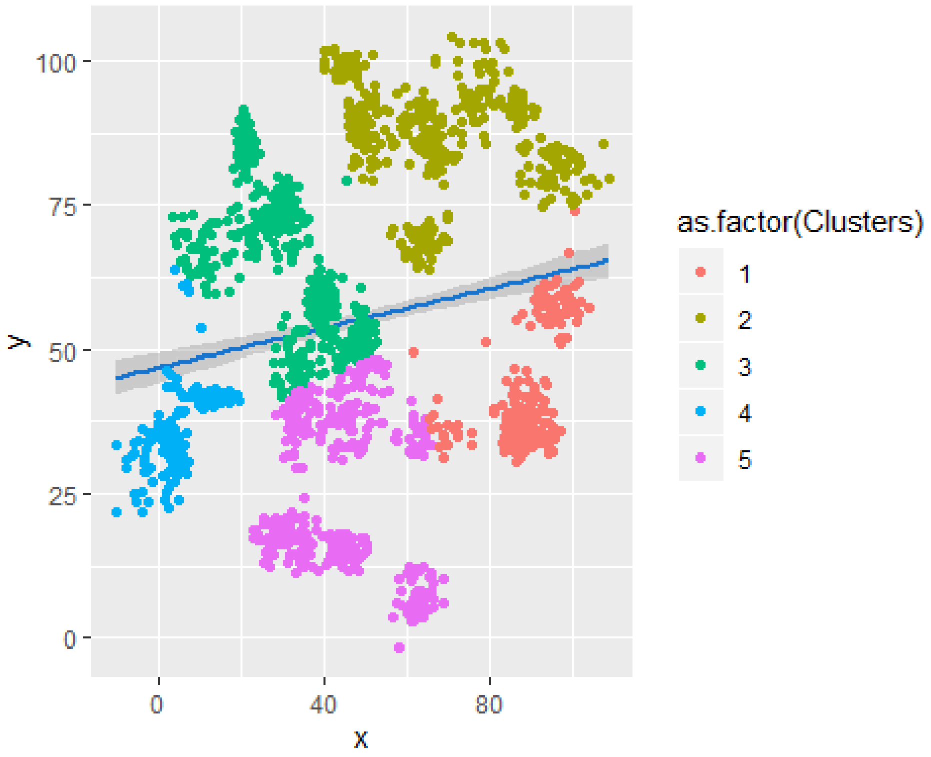
Scatter plot of trinary data



```
##          x        y label Clusters
## 1 70.88469 83.17702     0        4
## 2 74.97176 87.92922     0        4
## 3 73.78333 92.20325     0        4
## 4 66.40747 81.10617     0        4
## 5 69.07399 84.53739     0        4
## 6 72.23616 86.38403     0        4
```

```
##          x        y label Clusters
## 1 30.08387 39.63094     0        5
## 2 31.27613 51.77511     0        3
## 3 34.12138 49.27575     0        3
## 4 32.58222 41.23300     0        5
## 5 34.65069 45.47956     0        3
## 6 33.80513 44.24656     0        3
```

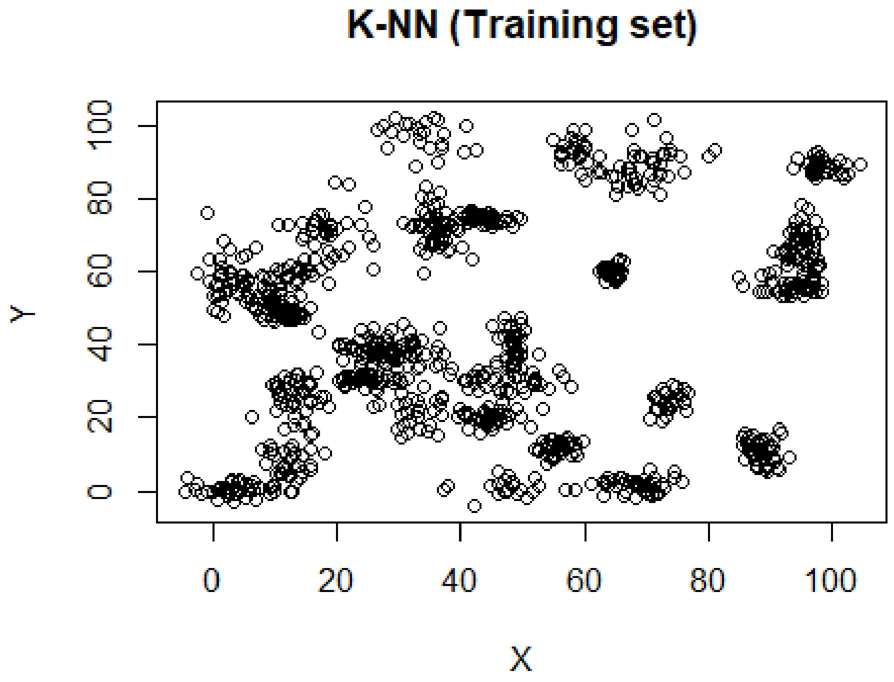


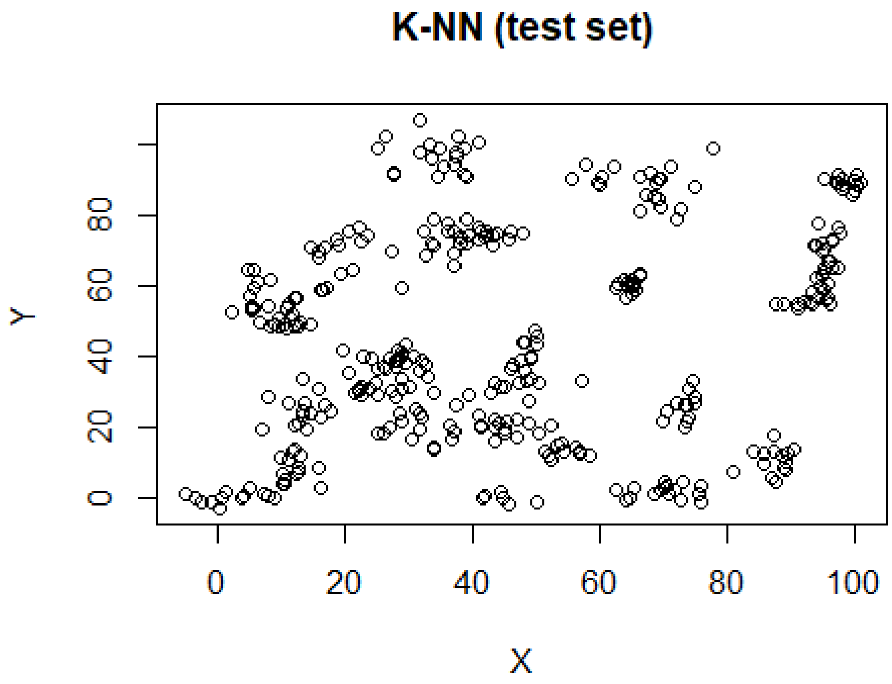


```
##             x        y label
## 2  74.97176 87.92922     0
## 4  66.40747 81.10617     0
## 5  69.07399 84.53739     0
## 8  77.57454 98.63425      0
## 11 67.20828 85.62172      0
## 16 69.23680 89.98705      0
```

```
## [1] 0 0 0 0 0 0
## Levels: 0 1
```

```
##    y_pred
##       0   1
##   0 186   6
##   1   4 179
```





K-NN (test set)
Y
0
20
40
60
80
X
0
20
40
60
80
100