Exercise 15: Introduction to Machine LEarning

Patkhedkar Ninad

2020-11-02

Exercise 15: Introduction to Machine Learning

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 that 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.

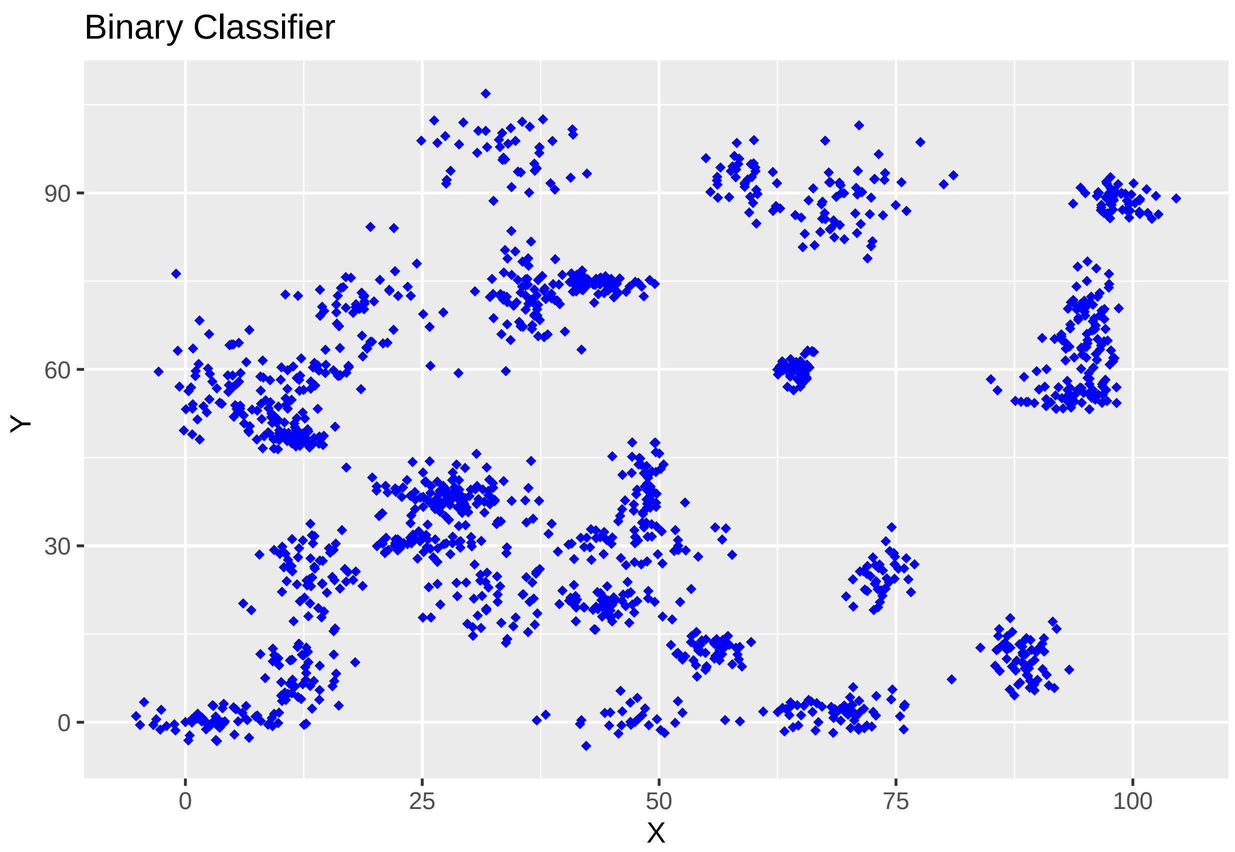
```r
library(ggplot2)
library(ggfortify)
library(caTools)
library(knitr)
library(pander)
library(class)

setwd("/cloud/project/completed/assignment09")
bi_df <- read.csv("binary-classifier-data.csv")
str(bi_df)
```

```
## 'data.frame':    1498 obs. of  3 variables:
##  $ label: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ x    : num  70.9 75 73.8 66.4 69.1 ...
##  $ y    : num  83.2 87.9 92.2 81.1 84.5 ...
```

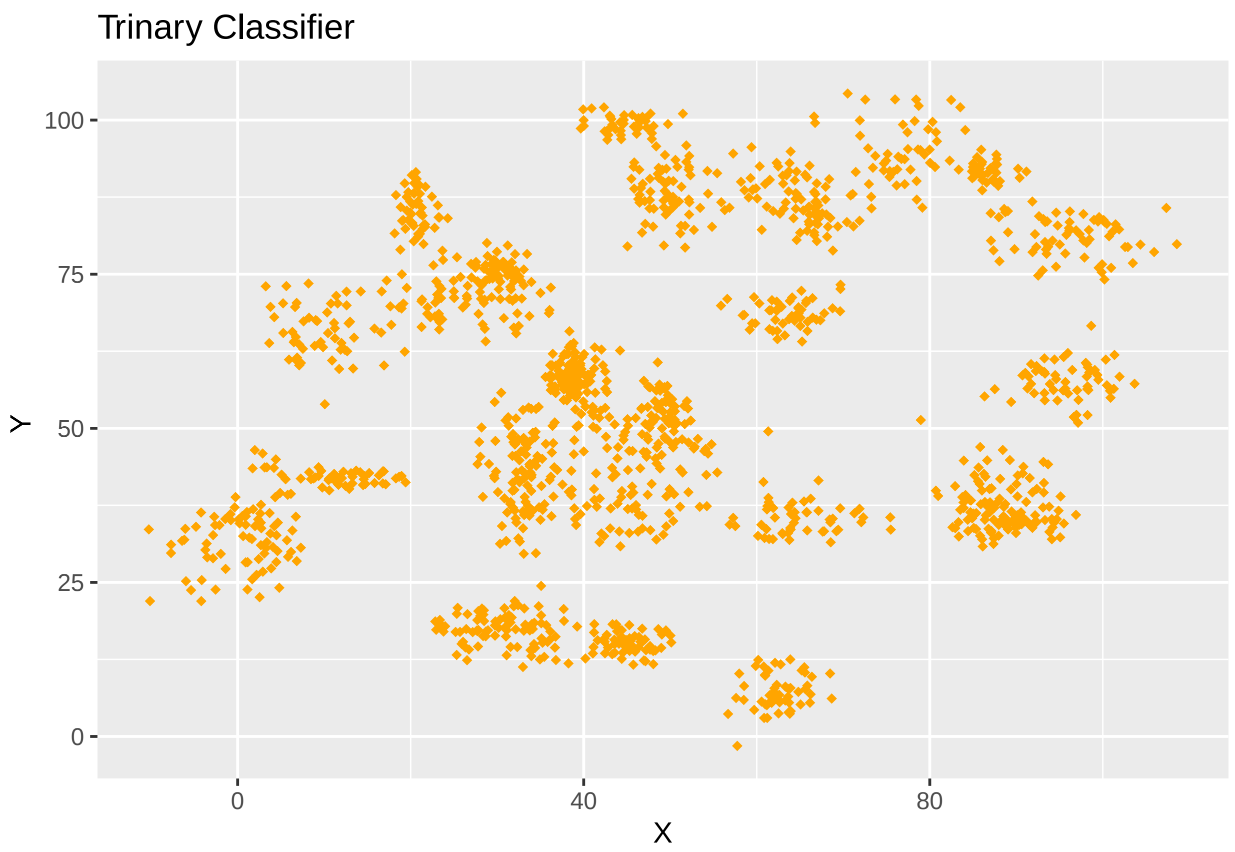
```r
tri_df <- read.csv("trinary-classifier-data.csv")
str(tri_df)
```

```
## 'data.frame':    1568 obs. of  3 variables:
##  $ label: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ x    : num  30.1 31.3 34.1 32.6 34.7 ...
##  $ y    : num  39.6 51.8 49.3 41.2 45.5 ...
```

```r
# Scattor-plot for Binary classifier
ggplot(bi_df, aes(x=x, y=y)) +
  geom_point(shape=18, color="blue") +
  labs(
    title = "Binary Classifier",
    x = "X",
    y = "Y "
  )
```



```
# Scattor-plot for Trinary classifier
ggplot(tri_df, aes(x=x, y=y)) +
  geom_point(shape=18, color="orange") +
  labs(
    title = "Trinary Classifier",
    x = "X",
    y = "Y "
  )
```



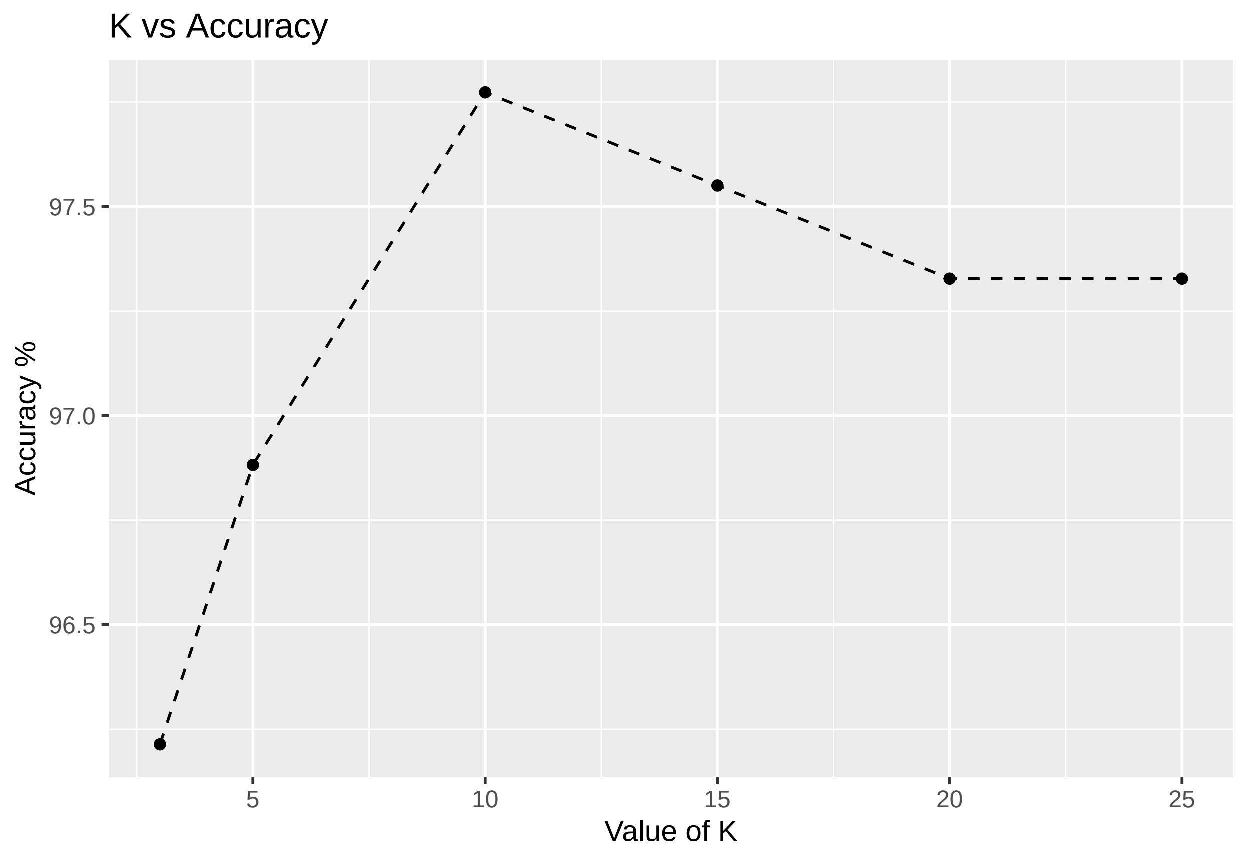
b. The k nearest neighbors algorithm categorizes an input value by looking at the labels for the k nearest points and assigning a category based on the most common label. In this problem, you will determine which points are nearest by calculating the Euclidean distance between two points. As a refresher, the Euclidean distance between two points: 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.

```r
# Vector of various k values we want to test with
set.seed(1234)
k_vals <- c(3,5,10,15,20,25)

sample <- sample.split(bi_df$label, SplitRatio = 0.70)
training_bi_data = subset(bi_df, sample == TRUE)
test_bi_data = subset(bi_df, sample == FALSE)

# Vector for accuracy values
accuracy_vals <- c()

for (i in k_vals) {
  predicted.values <- knn(training_bi_data[2:3], test_bi_data[2:3], training_bi_data$label, k = i)
  accuracy_vals <-c(accuracy_vals,100 * sum(test_bi_data$label == predicted.values)/NROW(test_bi_data$la
}
```

```r
# Create dataframe using k-values and corresponding accuracy values
k_accuracy_df <- data.frame(k_vals, accuracy_vals)
head(k_accuracy_df)
```

```
##   k_vals accuracy_vals
## 1      3      96.21381
## 2      5      96.88196
## 3     10      97.77283
## 4     15      97.55011
## 5     20      97.32739
## 6     25      97.32739
```

```r
ggplot(data=k_accuracy_df, aes(x=k_vals, y=accuracy_vals, group=1)) +
  geom_line(linetype = "dashed")+
  labs(
    title = "K vs Accuracy",
    x = "Value of K",
    y = "Accuracy %"
  ) +
  geom_point()
```



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?

- With this data and limited understanding of linear classifier,I am not sure which linear classifier would fit (if any)