iris

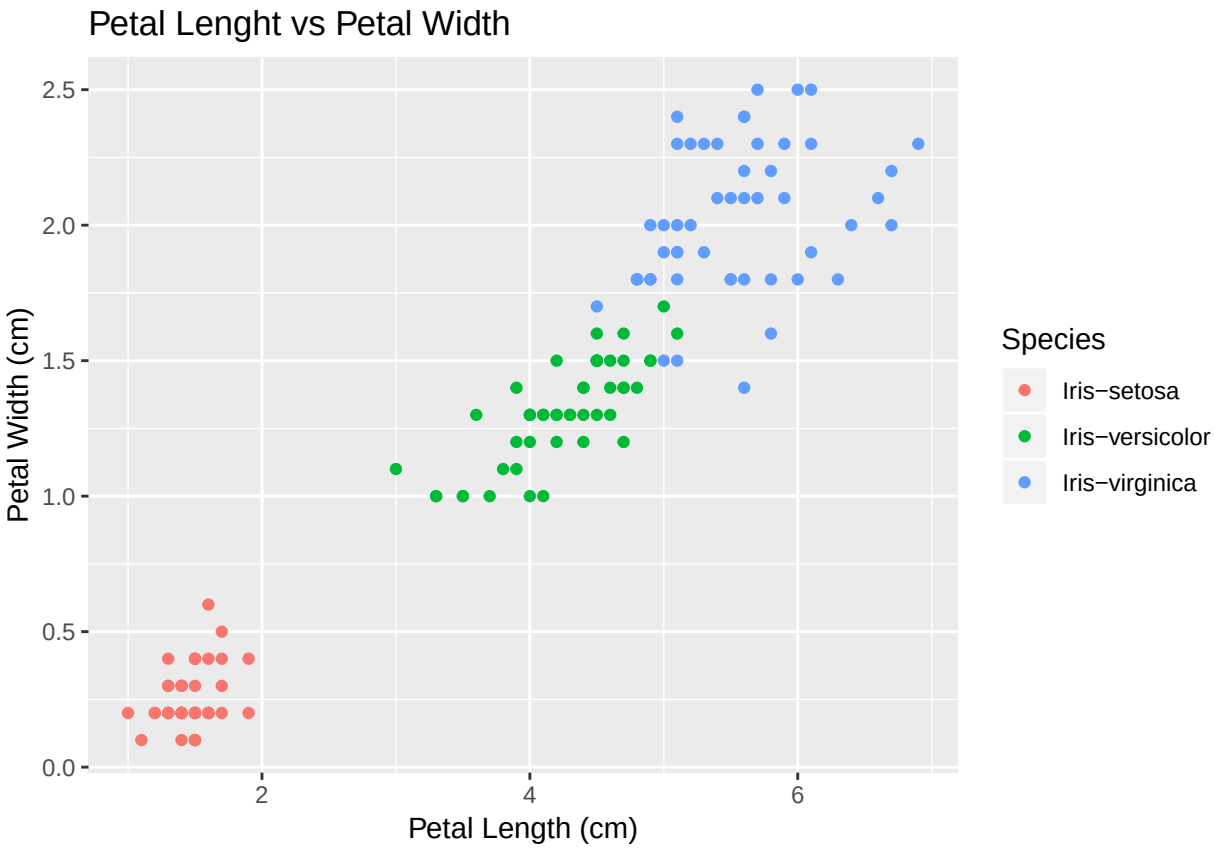
Daniel Kojis

January 16, 2019

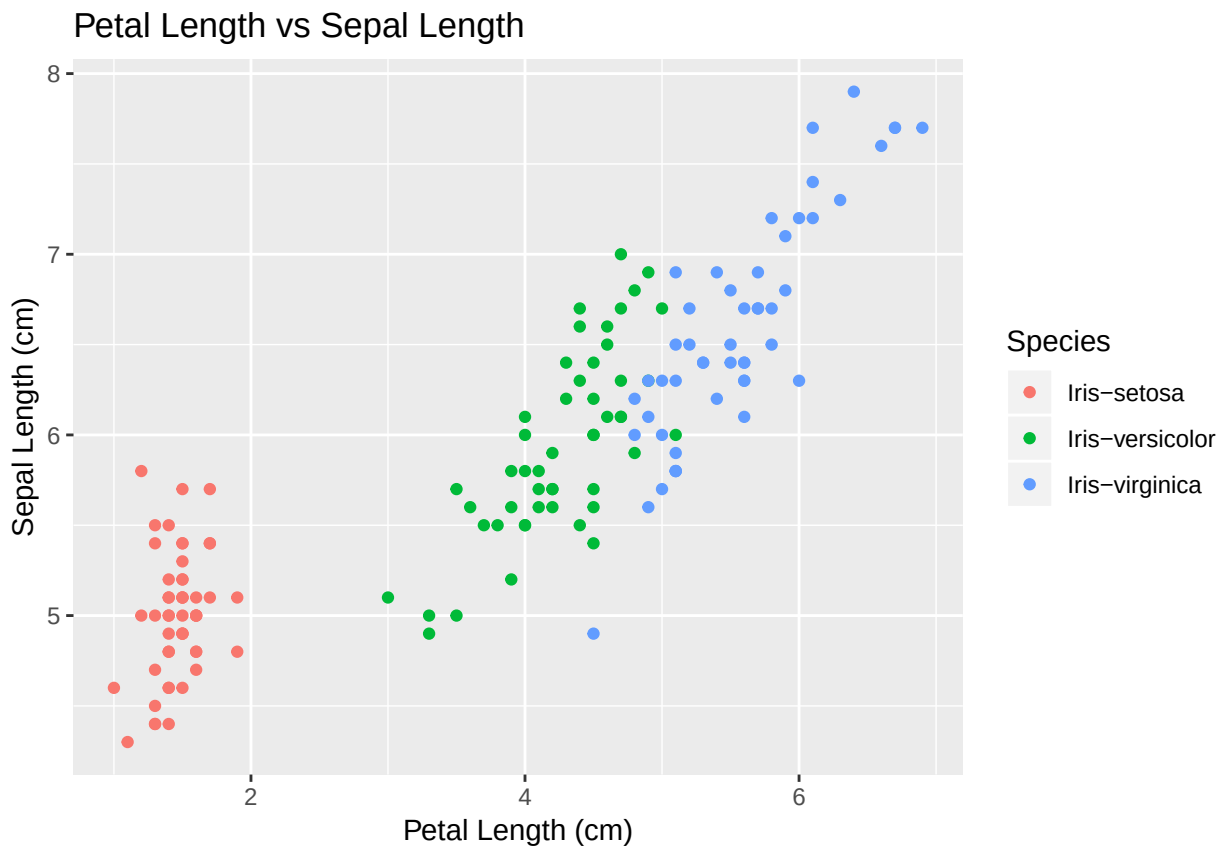
Today I'll be using a very simple kNN approach with the Iris dataset to predict the species of plants. This excersie is designed to familirize myself with some of the most basic machine learning techniques in R. More sophisticated approaches will be taken in the future with different and more challenging datasets.

```
data <- read.csv("Iris.csv")
```

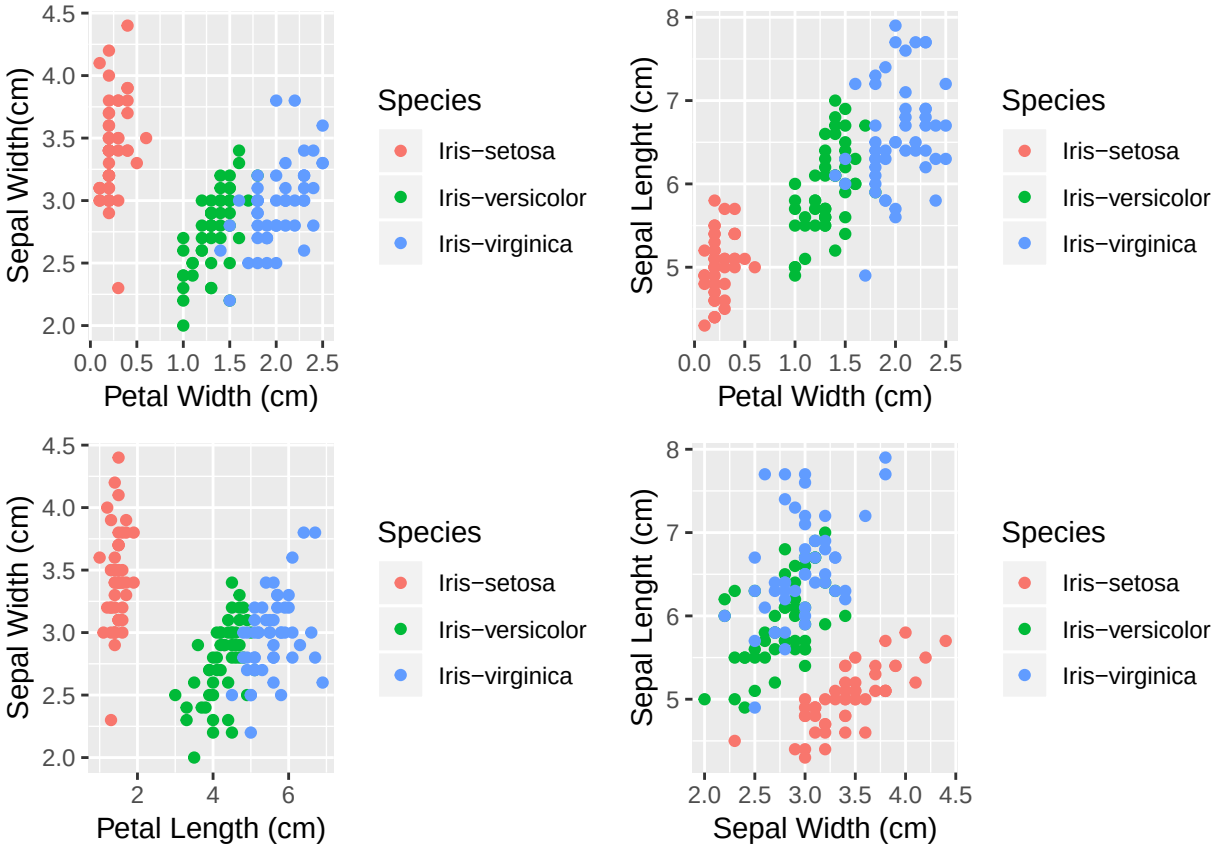
```
# Sepal Width vs Length
p1 <- ggplot(data, aes(x = SepalWidthCm, y = SepalLengthCm, color = Species)) +
  geom_point() +
  labs( y="Sepal Lenght (cm)", x="Sepal Width (cm)")

# Petal Length vs Width
p2 <- ggplot(data, aes(x = PetalLengthCm, y = PetalWidthCm, color = Species)) +
  geom_point() +
  labs(title="Petal Lenght vs Petal Width", y="Petal Width (cm)", x="Petal Length (cm)")

# Petal Width vs Sepal Width
p3 <- ggplot(data, aes(x = PetalWidthCm, y = SepalWidthCm, color = Species)) +
  geom_point() +
  labs( y="Sepal Width(cm)", x="Petal Width (cm)")

# Petal Width vs Sepal Length
p4 <- ggplot(data, aes(x = PetalWidthCm, y = SepalLengthCm, color = Species)) +
  geom_point() +
  labs( y="Sepal Lenght (cm)", x="Petal Width (cm)")

# Petal Length vs Sepal Width
p5 <- ggplot(data, aes(x = PetalLengthCm, y = SepalWidthCm, color = Species)) +
  geom_point() +
  labs( y="Sepal Width (cm)", x="Petal Length (cm)")

# Petal Length vs Sepal Length
p6 <- ggplot(data, aes(x = PetalLengthCm, y = SepalLengthCm, color = Species)) +
  geom_point() +
  labs(title="Petal Length vs Sepal Length", y="Sepal Length (cm)", x="Petal Length (cm)")

print(p2)
```


Petal Lenght vs Petal Width
Petal Width (cm)
Petal Length (cm)
2.5
2.0
1.5
1.0
0.5
0.0
2
4
6
Species
Iris−setosa
Iris−versicolor
Iris−virginica

```
print(p6)
```




```
ggarrange(p3, p4, p5, p1,
          ncol = 2, nrow = 2)
```



From observing the scatterplots, the data already locates itself in identifiable clusters and shouldn't be hard to classify with a simple machine learning model. A good classifier could be built by using just two features, such as Petal Width and Petal Length, but we'll include all features in the model because they seem to contain meaningful information that should marginally improve the model. Also, it's probably not neccessary to scale/normalize the data either because they are relatively similar scales.

Let's build some kNN models and test the accuracy scores for values of k 1 through 15.

```
set.seed(99) # Set Seed so that same sample can be reproduced in future also


sample <- sample.int(n = nrow(data), size = floor(.6667*nrow(data)), replace = F)

train <- data[sample,  c("PetalLengthCm", "PetalWidthCm","SepalLengthCm", "SepalWidthCm")] #%>% scale()
train.labels <- data[sample,  c("Species")]
test  <- data[-sample, c("PetalLengthCm", "PetalWidthCm","SepalLengthCm", "SepalWidthCm") ] #%>% scale(
test.labels <- data[-sample,  c("Species")]

for(i in 1:15){
predictions <- knn(train = train, test =  test, cl = train.labels, k = i )

test.labels.df <- data.frame(test.labels)
df <- data.frame(test.labels.df, predictions)
names(df) <- c("Observed", "Predicted")

acc <- sum(df$Observed == df$Predicted) / length(df$Observed)
print(paste0("For k=", i , ", accuracy is ", acc, "%"))
```

```
}
```

```
## [1] "For k=1, accuracy is 0.96%"
## [1] "For k=2, accuracy is 0.94%"
## [1] "For k=3, accuracy is 0.96%"
## [1] "For k=4, accuracy is 0.98%"
## [1] "For k=5, accuracy is 0.98%"
## [1] "For k=6, accuracy is 0.96%"
## [1] "For k=7, accuracy is 0.98%"
## [1] "For k=8, accuracy is 0.98%"
## [1] "For k=9, accuracy is 0.96%"
## [1] "For k=10, accuracy is 0.94%"
## [1] "For k=11, accuracy is 0.96%"
## [1] "For k=12, accuracy is 0.96%"
## [1] "For k=13, accuracy is 0.96%"
## [1] "For k=14, accuracy is 0.96%"
## [1] "For k=15, accuracy is 0.98%"
```

From the results we can see which models performed best. However, it's important to note that selecting the best model in this fashion is poor practive. Rather, we should use training and validation set for model selection and set aside the test set for a final evaluation of the one model we chose. Otherwise, we may select a model that performed unusually higher than expected just because of the random train-test split.

We will graph the confusion matrix of the kNN model with value of 7 for k.

```
predictions <- knn(train = train, test =  test, cl = train.labels, k = 7 )
acc <- sum(df$Observed == df$Predicted) / length(df$Observed)


CrossTable(x = df$Observed, y = df$Predicted, prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  50
##
##
##                  | df$Predicted
##     df$Observed |     Iris-setosa | Iris-versicolor |  Iris-virginica |       Row Total |
## ----------------|-----------------|-----------------|-----------------|-----------------|
##     Iris-setosa |              15 |               0 |               0 |              15 |
##                 |           1.000 |           0.000 |           0.000 |           0.300 |
##                 |           1.000 |           0.000 |           0.000 |                 |
##                 |           0.300 |           0.000 |           0.000 |                 |
## ----------------|-----------------|-----------------|-----------------|-----------------|
```

```
## Iris-versicolor |               0 |              15 |               1 |              16 |
##                 |           0.000 |           0.938 |           0.062 |           0.320 |
##                 |           0.000 |           1.000 |           0.050 |                 |
##                 |           0.000 |           0.300 |           0.020 |                 |
## ----------------|-----------------|-----------------|-----------------|-----------------|
##  Iris-virginica |               0 |               0 |              19 |              19 |
##                 |           0.000 |           0.000 |           1.000 |           0.380 |
##                 |           0.000 |           0.000 |           0.950 |                 |
##                 |           0.000 |           0.000 |           0.380 |                 |
## ----------------|-----------------|-----------------|-----------------|-----------------|
##    Column Total |              15 |              15 |              20 |              50 |
##                 |           0.300 |           0.300 |           0.400 |                 |
## ----------------|-----------------|-----------------|-----------------|-----------------|
##
##
```

We can see that this model did a very good job at classifyint the data. It classified 49 out of 50 plants correctly, with the one exception classifying a Iris-versicolor as a Iris-virginica.

This excersie was just to familiarize myself with kNN tecnhniques in R. For future exercises with kNN, we should take into consideration scaling feature variables, stratifying the train-test split on the species type, using a validation set for hyperparamter optimization and model comparison, as well as introducing cross validation for more robust results.