DA410_Project 4_MattGraham

In this project, we are working on k-nearest neighbors analysis, or KNN. We classify new data based on stored datapoints by similarity of measure.

KNN uses a ‘majority vote’ rule and determines classification by count of nearest closest to the calculated value. In other words, the closer your predicted point is to what actually happened, the more likely it is to fall within that classification.

We will be leveraging the iris dataset for this.

```
library(nnspat)  # used for dist2full()
library("dplyr")  # used to select numeric datatypes
library("ggplot2")
library(reshape)  # used for melting matricies
library(klaR)
library(ggvis)
library(class)
library(gmodels)
library(MASS)
```

Getting and understanding our data

```
iris
```

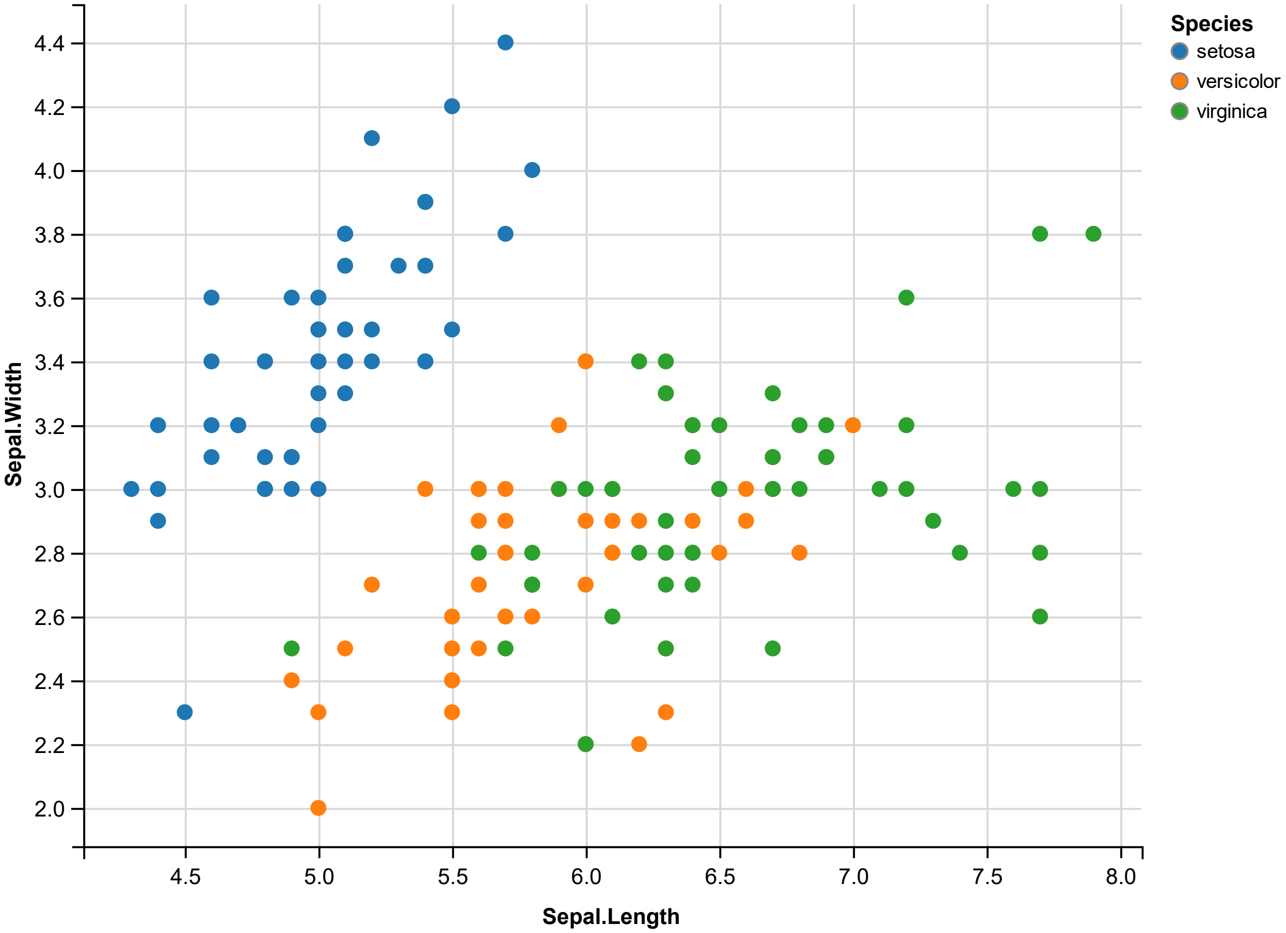
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
<dbl>	<dbl>	<dbl>	<dbl>	<fct>
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa

1-10 of 150 rows Previous **1** 2 3 4 5 6 ... 15 Next

Visualizing our data

Sepal length vs. Sepal width

```
iris %>% ggvis(~Sepal.Length, ~Sepal.Width, fill = ~Species) %>%
  layer_points()
```



Above, we can see Setosa is a cluster having shorter, but wider sepal lengths. We can also note the linear trend among both clusters. Setosa, again, appearing to have a stronger relationship compared to the mixed cluster of Versicolor and Virginica.

```
as.data.frame(
  iris %>%
    group_by(Species) %>%
    summarize(cor=cor(Sepal.Length, Sepal.Width))
)
```

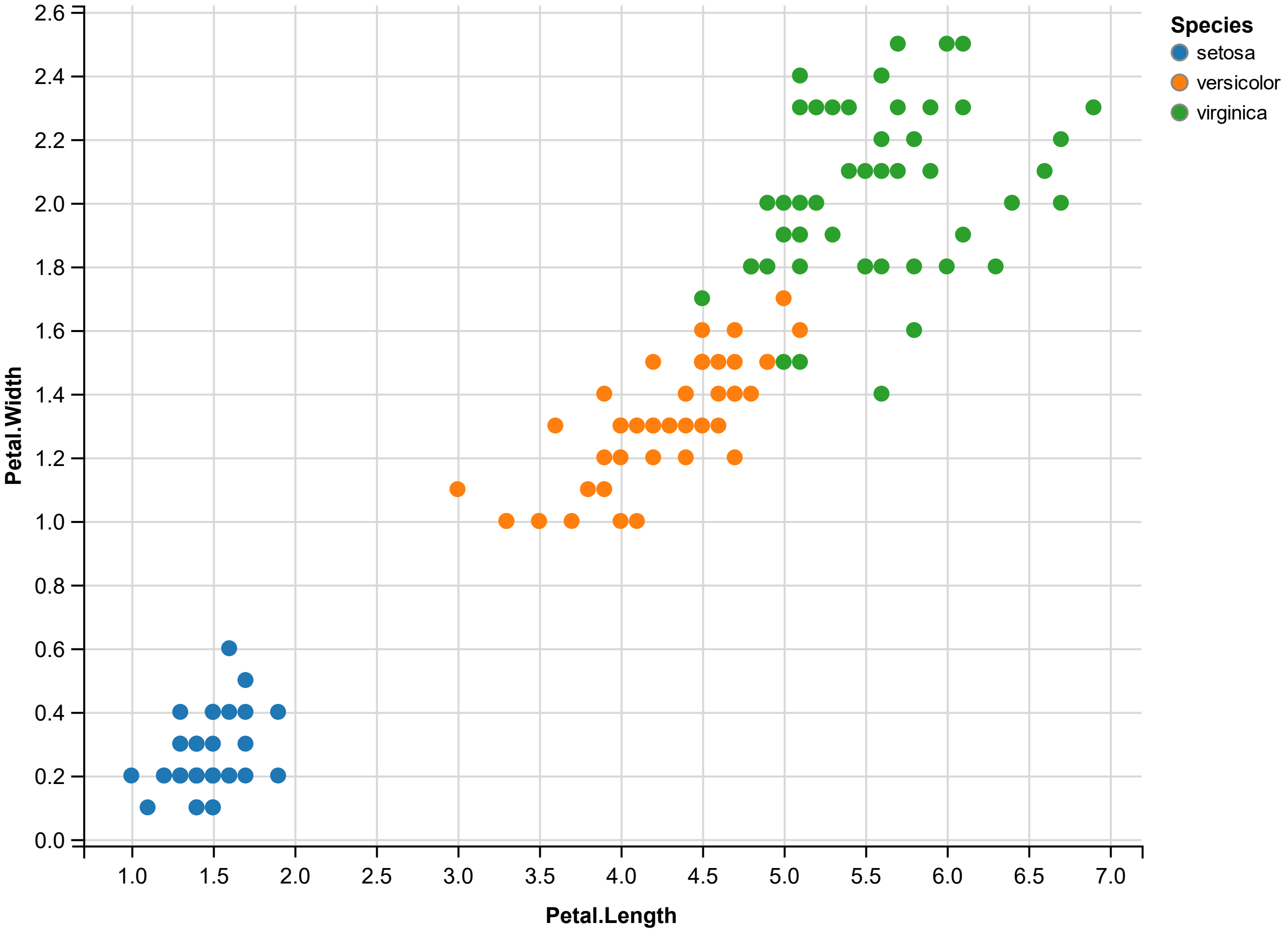
Species <fct>	cor <dbl>
setosa	0.7425467
versicolor	0.5259107
virginica	0.4572278

3 rows

According to our correlation plot, we can see the Setosa is our most correlated species.

Petal length vs. Petal width

```
iris %>%
  ggvis(~Petal.Length, ~Petal.Width, fill = ~Species) %>%
  layer_points()
```



While Setosa was our primary standout for sepal width, they have the smallest petals We can note Virginica having the largest set of petals. There are three distinct clusters.

```
as.data.frame(
  iris %>%
    group_by(Species) %>%
    summarize(cor=cor(Petal.Length, Petal.Width))
)
```

Species	**cor**
<fct>	<dbl>
setosa	0.3316300
versicolor	0.7866681
virginica	0.3221082

3 rows

We can see above that Versicolor, our middle-most cluster, is our most correlated species. Setosa and Virginica are the opposite.

Training and testing our models

```
set.seed(1234)
# splitting our data (2/3 is training, 1/3 is testing)
ind <- sample(2, nrow(iris), replace=TRUE, prob=c(0.67, 0.33))

# assigning sets for use
iris.training <- iris[ind==1, 1:4]
iris.test <- iris[ind==2, 1:4]

# creating labels to check for correct classification determination
iris.trainLabels <- iris[ind==1, 5]
iris.testLabels <- iris[ind==2, 5]
```

Training dataset

```
iris.training
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
	<dbl>	<dbl>	<dbl>	<dbl>
1	5.1	3.5	1.4	0.2
2	4.9	3.0	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
6	5.4	3.9	1.7	0.4
7	4.6	3.4	1.4	0.3
8	5.0	3.4	1.5	0.2
9	4.4	2.9	1.4	0.2
10	4.9	3.1	1.5	0.1
12	4.8	3.4	1.6	0.2

1-10 of 110 rows Previous **1** 2 3 4 5 6 ... 11 Next

Testing dataset

```
iris.test
```

	Sepal.Length <dbl>	Sepal.Width <dbl>	Petal.Length <dbl>	Petal.Width <dbl>
5	5.0	3.6	1.4	0.2
11	5.4	3.7	1.5	0.2
14	4.3	3.0	1.1	0.1
16	5.7	4.4	1.5	0.4
26	5.0	3.0	1.6	0.2
28	5.2	3.5	1.5	0.2
29	5.2	3.4	1.4	0.2
36	5.0	3.2	1.2	0.2
39	4.4	3.0	1.3	0.2
40	5.1	3.4	1.5	0.2

1-10 of 40 rows Previous **1** 2 3 4 Next

Creating our model

```
iris.pred <- knn(train = iris.training, test=iris.test, cl=iris.trainLabels, k=3)
iris.pred
```

```
##  [1] setosa     setosa     setosa     setosa     setosa     setosa
##  [7] setosa     setosa     setosa     setosa     setosa     setosa
## [13] versicolor versicolor versicolor versicolor versicolor versicolor
## [19] versicolor versicolor versicolor versicolor versicolor versicolor
## [25] virginica  virginica  virginica  virginica  versicolor virginica
## [31] virginica  virginica  virginica  virginica  virginica  virginica
## [37] virginica  virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

Above is a list of predicted classifiers that we will check against what we passed into our test.

Showing the comparison

```
merge <- data.frame(iris.pred, iris.testLabels)
merge
```

iris.pred <fct>	iris.testLabels <fct>
setosa	setosa
setosa	setosa

iris.pred <fct>	iris.testLabels <fct>
setosa	setosa
setosa	setosa
setosa	setosa
setosa	setosa
setosa	setosa
setosa	setosa
setosa	setosa
setosa	setosa

1-10 of 40 rows Previous **1** 2 3 4 Next

Making a cross table

```
CrossTable(x=iris.testLabels, y=iris.pred, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  40
##
##
##                 | iris.pred
## iris.testLabels |     setosa | versicolor |  virginica |  Row Total |
## ----------------|------------|------------|------------|------------|
##          setosa |         12 |          0 |          0 |         12 |
##                 |      1.000 |      0.000 |      0.000 |      0.300 |
##                 |      1.000 |      0.000 |      0.000 |            |
##                 |      0.300 |      0.000 |      0.000 |            |
## ----------------|------------|------------|------------|------------|
##      versicolor |          0 |         12 |          0 |         12 |
##                 |      0.000 |      1.000 |      0.000 |      0.300 |
##                 |      0.000 |      0.923 |      0.000 |            |
##                 |      0.000 |      0.300 |      0.000 |            |
## ----------------|------------|------------|------------|------------|
##       virginica |          0 |          1 |         15 |         16 |
##                 |      0.000 |      0.062 |      0.938 |      0.400 |
##                 |      0.000 |      0.077 |      1.000 |            |
##                 |      0.000 |      0.025 |      0.375 |            |
## ----------------|------------|------------|------------|------------|
##    Column Total |         12 |         13 |         15 |         40 |
##                 |      0.300 |      0.325 |      0.375 |            |
## ----------------|------------|------------|------------|------------|
##
##
```

```
mis.calc <- 1
total.calc <- 40

err.rate <- mis.calc / total.calc
err.rate
```

```
## [1] 0.025
```

Above, we can see how accurate our model was. We only had 1 misclassification across all 40 test values. Resulting in a 2.5% error rate. This is very strong! We do not need to improve this model any further.