K-Means Clustering

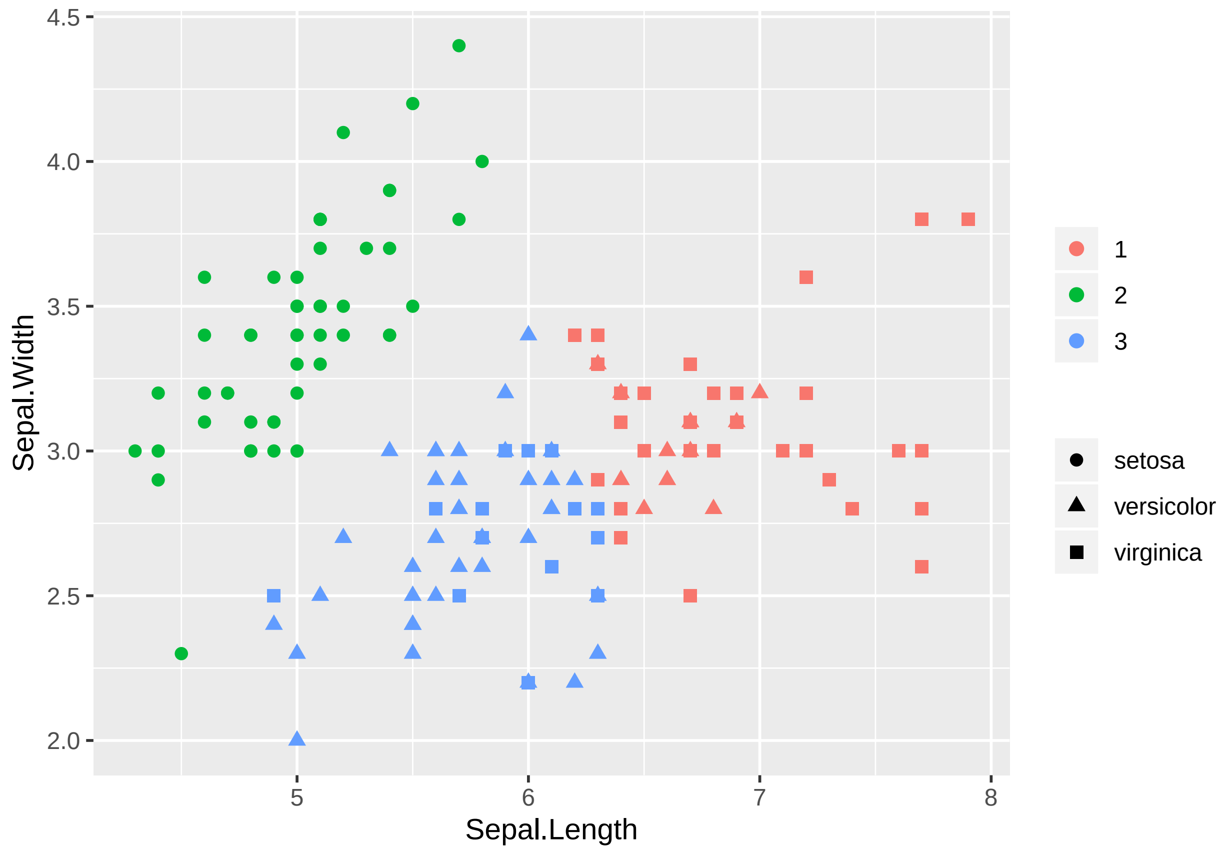
- Choose number of clusters K
- Randomly generate/select K centroids and assign each data point to the nearest centroid
- Recalculate centroids and reassign data to nearest centroid until the cluster assignments don't change

Iris Example

```
model_km = kmeans(iris[,1:2], centers=3)
iris$cluster = model_km$cluster
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.6.2
```

```
ggplot(iris, aes(Sepal.Length, Sepal.Width, color=as.factor(cluster), shape=Species)) + geom_point(size=
```



Parameters for kmeans

- nstart: how many random starts should be done? Run algorithm nstart times and choose the result with the lowest total within sum of squares. Default = 1
- iter.max: maximum number of iterations. Default is 10, change if convergence takes longer/

Elbow (Scree) Plot

Run many models with varying value of k, calculate total within-cluster sum of squares, find where 'elbow' occurs to determine optimal k to use

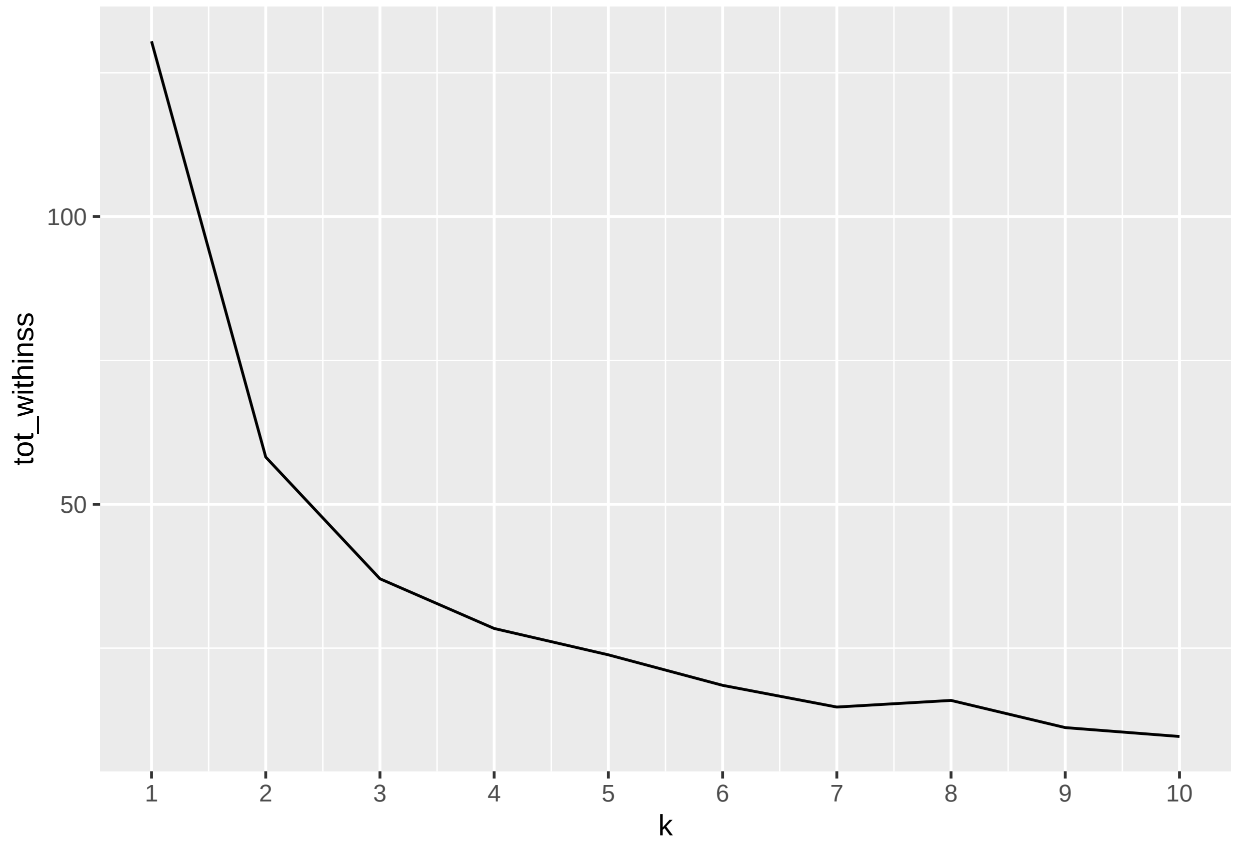
```r
library(purrr)
```

```
## Warning: package 'purrr' was built under R version 3.6.2
```

```r
# Use map_dbl to run many models with varying value of k (centers)
tot_withinss <- map_dbl(1:10,  function(k){
  model <- kmeans(x = iris[,1:2], centers = k)
  model$tot.withinss
})

# Generate a data frame containing both k and tot_withinss
elbow_df <- data.frame(
  k = 1:10,
  tot_withinss = tot_withinss
)

# Plot the elbow plot
ggplot(elbow_df, aes(x = k, y = tot_withinss)) +
  geom_line() +
  scale_x_continuous(breaks = 1:10)
```



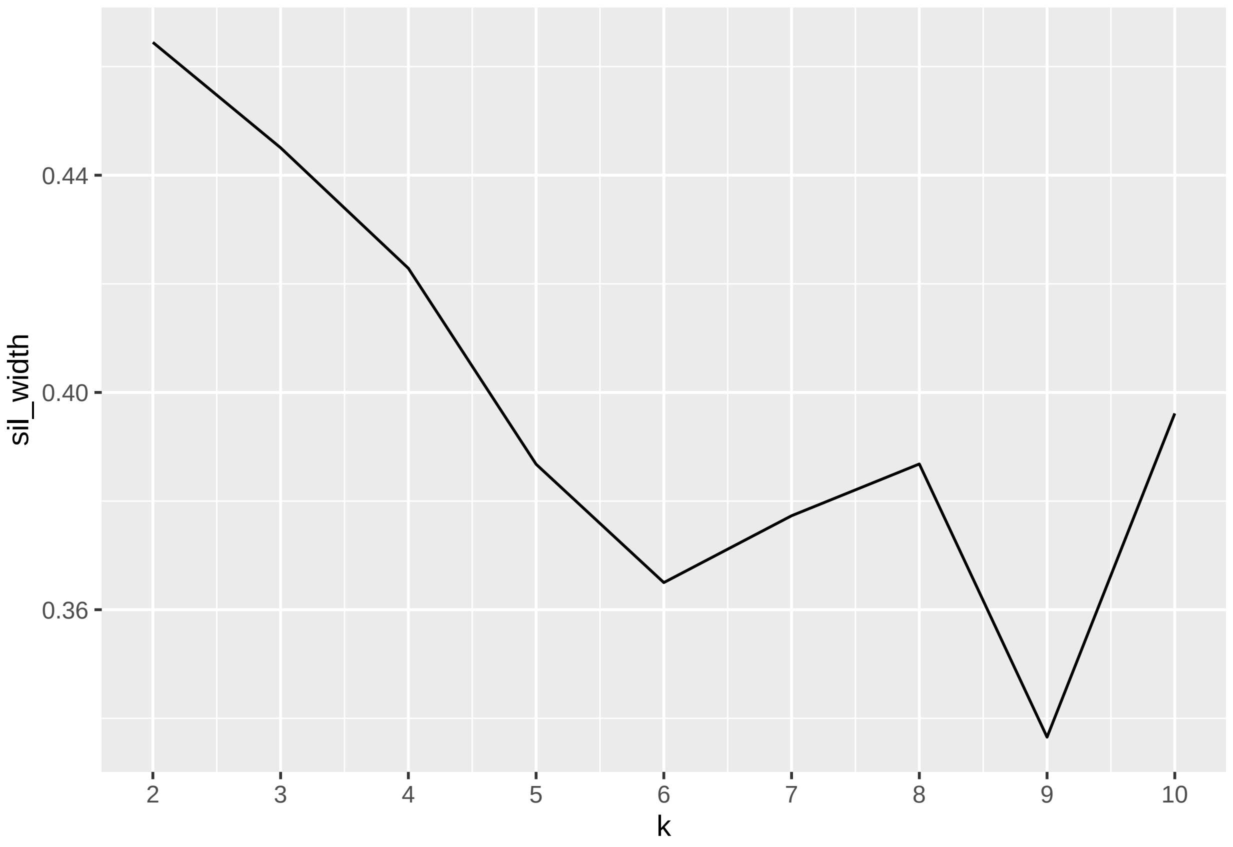
Silhouette Analysis

Calculate how similar each observation is with the cluster it's assigned relative to other clusters. Ranges from -1 (observations may be assigned to the wrong cluster) to 1 (observation is well matched to the assigned cluster). A value of 0 suggests the observation is borderline matched between two clusters

```r
library(cluster)
# Use map_dbl to run many models with varying value of k
sil_width = map_dbl(2:10,  function(k){
  model = kmeans(x = iris[,1:2], centers = k)
  ss = silhouette(model$cluster, dist(iris[,1:2]))
  mean(ss[,3]) #sil_width
})

# Generate a data frame containing both k and sil_width
sil_df = data.frame(
  k = 2:10,
  sil_width = sil_width
)

# Plot the relationship between k and sil_width
ggplot(sil_df, aes(x = k, y = sil_width)) +
  geom_line() +
  scale_x_continuous(breaks = 2:10)
```



K=2 has the largest silhouette score, indicating we should use 2 clusters (though recall we know there are actually 3 species).