Module 10: August 5, 2015 (2nd half)

Main Discussion Points:
- Initializing k-means
- Latent Features/Dimensionality Reduction
    - Principal Component Analysis

Talking Points:
- How do we find the point for initializing k-means clustering?
- How do we perform statistical analysis on data with a very large feature count?

How to initialize k-means?

We want to end up picking points from things that will eventually be our clusters. However, it is not likely we will choose the best points by picking at random. We want to pick points that are spread out.

K-means++

Rather than picking all points at random, we want to ensure that we are picking points that are likely to be spread out.

This is strictly a method for how to initialize k-means:
1. Choose 1 center at random: m_1
2. Initialize cluster list as M = $\{m_1\}$
3. Repeat for 2, … , k
    a. Compute $d_i = \min_{m \in M} d(x_i, m)$
    b. Sample m_j with probability proportional to d_i^2
    c. Add m_j to M

The repeated steps of the algorithm, again, in English:
a. Compute minimum distance from centers.
b. Sample from all points with probability proportional to distance-squared, where the distance is a point's minimum distance out of the point's distances to all centers. This ensures point is far away from all current clusters.

Note: k-means++ is implemented as `cclust` in the R package `flexclust`

Latent Features (Dimensionality Reduction)

Latent features, or dimensionality reduction, is the process of reducing the data with large number of features and projecting down to lesser number of features. This is especially useful for a large number of features because the data can be difficult to summarize.

If we have a dataset that has D features (dimensions), we can do some "mapping" or "projecting" down to K features that are "interesting" where K << D (K is much less than D). The objective of dimensionality reduction is to reduce the number of features with minimal loss of information. It is often the first step in statistical analyses involving large feature data. Ideal scenario: We could take all features and map them down to 1 feature without any loss of information. Of course, this is impossible.

*Note: Clustering is essentially a form of dimensionality reduction: we reduce data to K clusters in order to gain a better understanding. However, the scope of dimensionality reduction is much broader.

Key Question: How do we define "interesting"?

Answer from **Principal Component Analysis**:
"interesting" means "high variance"

Principal components analysis (PCA) is one of a family of techniques for taking high-dimensional data, and using the dependencies between the variables to represent it in lower-dimensional form, without losing too much information.

Some Linear Algebra review first…

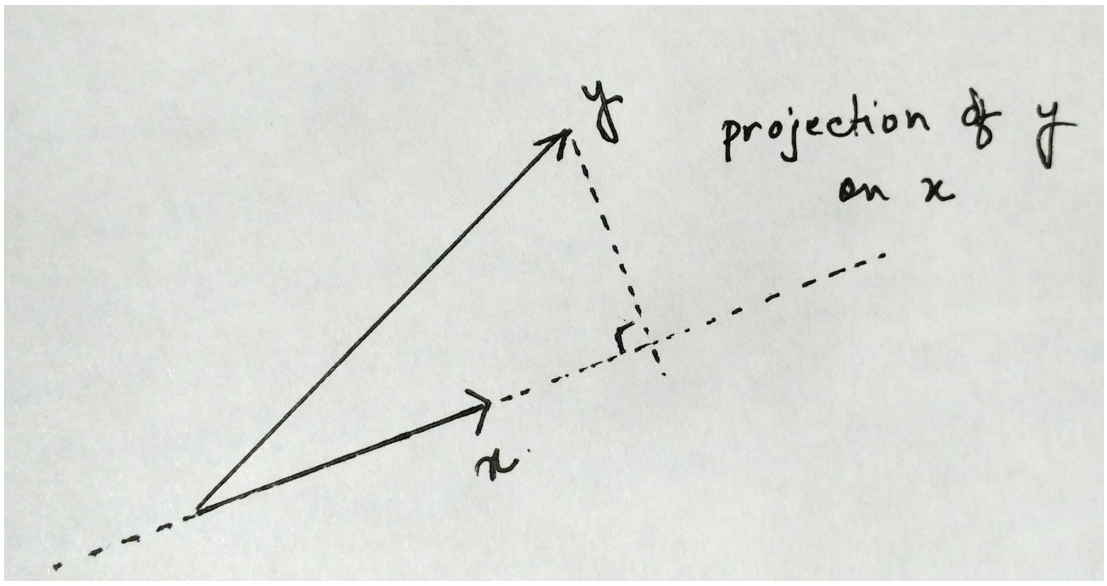
x, y are vectors in D-dimensional space
$x \in \mathbf{R}^D, y \in \mathbf{R}^D$

dot product (inner product):
$< x, y > = x \cdot y = x^T y = \sum_{i=1}^{D} x_i \, y_i$

squared norm of x:
$||x||_2^2 = x^T x$

A vector is said to have "unit norm" if:
$x^T x = ||x||_2^2 = 1$

Let x be a unit-norm vector. The projection of y onto x is $(y^T x)x$:



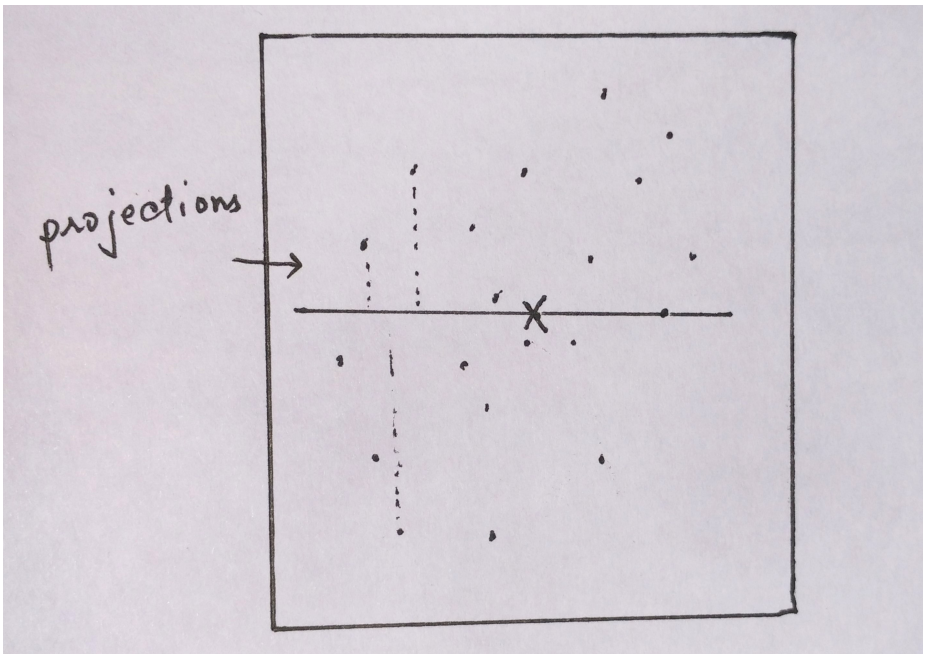
Basically: What *multiple* of x is the projection of y?

$\hat{y}$ – the projection of y onto x

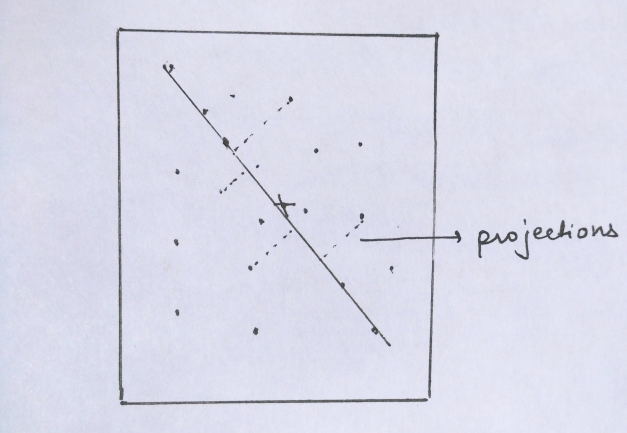
$$\hat{y} = \alpha \cdot x$$

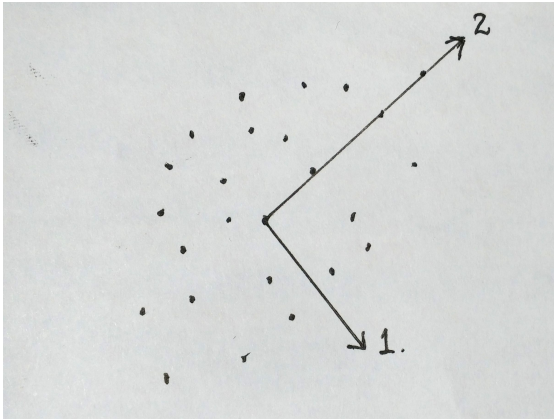
$$\alpha = y^T x = x^T y$$

What happens if we pick unit norm vector $x = (1, 0)$ to define a subspace, and we project a collection of data points in $\mathbf{R}^2$ onto that subspace?



We want to choose the vector for which the projection minimizes the distortion or, equivalently, maximizes the variance of the projected points.





This is called the first principal component. (in the picture above right: #2 is the 1st PC)

How to find first principal component:

Let x_i be data point i, $x_i \in \mathbf{R}^D$

Maximize $\mathrm{Var}[(x_i^T v)v]$ where $\{v: v \in \mathbf{R}^D\}$

We do this because features that result in minimal loss of information and maximum variance of the projections are the most interesting.

When we repeat this a 2nd time, we get the second principal component, and so on.

Summary of PCA:

We start with d-dimensional vectors, and want to summarize them by projecting down into a k-dimensional subspace. The reduced features will be the projection of the original vectors on to k directions or the principal components.

In order to perform principal component analysis, we choose the vector that minimizes the distortion and maximizes the variance of projecting points. The first principal component is the direction in space along which projections have the largest variance. The second principal component is the direction which maximizes variance among all directions orthogonal to the first. The kth component is the variance-maximizing direction orthogonal to the previous k – 1 components.

To illustrate the PCA, we run the following R scripts. Note that it is important to run PCA on a scaled data matrix.

```
#Loading the library
library(ggplot2)
```

We took the iris dataset, which gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are Iris setosa, versicolor, and virginica.

```
# Load a toy data and peak at the numbers
data(iris)
head(iris)
```

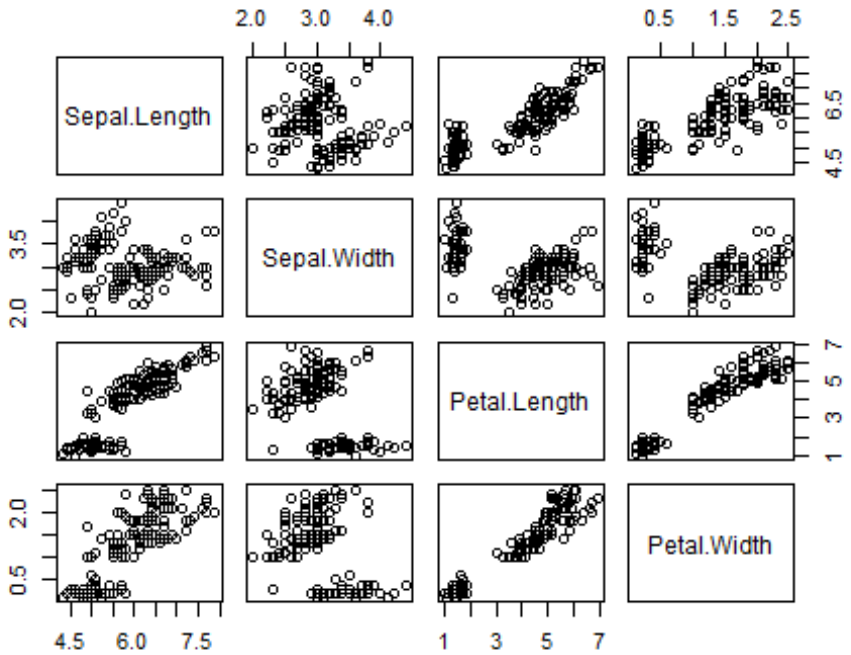
```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

We just pick out the numerical columns (4 variables) from the dataset and demonstrate PCA

```
# Pick out the numerical columns from the data set on which we want to do the
 analysis(run ?iris for further info)
Z = iris[,1:4]
```

Checking the correlation between various dimensions

```
# Clearly a lot of correlation structure in the measurements
pairs(Z)
```



Taking four dimensions computing the PCA

```
# Run PCA
pc1 = prcomp(Z, scale.=TRUE)
```

Looking at the PCA summary and plot, it shows PC1 has the highest standard deviation and the variance. Our aim is to find the fit, where we have the highest variance.

```
# Look at the basic plotting and summary methods
pc1
```

```
## Standard deviations:
## [1] 1.7083611 0.9560494 0.3830886 0.1439265
##
## Rotation:
##                      PC1         PC2        PC3        PC4
## Sepal.Length  0.5210659 -0.37741762  0.7195664  0.2612863
## Sepal.Width  -0.2693474 -0.92329566 -0.2443818 -0.1235096
## Petal.Length  0.5804131 -0.02449161 -0.1421264 -0.8014492
## Petal.Width   0.5648565 -0.06694199 -0.6342727  0.5235971
```

```
summary(pc1)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4
## Standard deviation     1.7084 0.9560 0.38309 0.14393
## Proportion of Variance 0.7296 0.2285 0.03669 0.00518
## Cumulative Proportion  0.7296 0.9581 0.99482 1.00000
```

```
plot(pc1)
```

If we want to summarize data now with PC1, we project the data points onto it, giving each point a score on the vector, α_i.

$$\alpha_{i1} = x_i^T v \text{ , where v is the first principal component.}$$

Thus, we reduce the amount of total information we need to describe the data. Instead of 4 features to describe each of the N data points (4N pieces of information), we only need N+4 "reduced" pieces of information: the 4-dimensional vector (v, or PC1 above) plus the N scores along that vector.