# Homework 3

*PSTAT 131/231, Winter 2018*

***Due on March 11, 2018 at 11:55 pm***

---

In this homework, the problems will be more computationally intensive than previous problems in this class. You should expect some problems to take a couple of minutes to complete. Re-knitting your file can take a long time so you should consider using `cache=TRUE` option in R chunks that involve code which takes a while to run.

Another option would be to complete some of the more computationally demanding problems in separate Rmd

Following packages will be used in this homework. Please install any packages that are not currently installed.

```
library(tidyverse)
library(e1071)
library(cluster)
library(NbClust)
library(bmp)

library(imager)
library(tree)
library(randomForest)
library(gbm)
library(ROCR)
```

---

1. **Fundamentals of the bootstrap**

   In the first part of this problem we will explore the fact that approximately 1/3 of the observations in a bootstrap sample are *out-of-bag*.

   (a) Given a sample of size $n$, what is the probability that any observation $j$ is *not* in in a bootstrap sample? Express your answer as a function of $n$.

   (b) Compute the above probability for $n = 1000$.

   (c) Verify that your calculation is reasonable by resampling the numbers 1 to 1000 with replace and printing the number of missing observations. Hint: use the `unique` and `length` functions to identify how many unique observations are in the sample.

   Here we'll use the bootstrap to compute uncertainty about a parameter of interest.

   (a) As of November 18, an NBA basketball player, Robert Covington, had made 50 out of 101 three point shot attempts this season. This field goal percentage (0.495), would rank in the 10 ten all time for single season three point shooting. Use bootstrap resampling on a sequence of 50 1's (makes) and 51 0's (misses). For each bootstrap sample compute and save the sample mean (e.g. bootstrap FG% for the player). Use 1000 bootstrap samples to plot a histogram of those values. Compute the 95% bootstrap confidence interval for Robert Covington's "true" FG% using the `quantile` function in R. Print the endpoints of this interval? Do you expect his season field goal percentage to be higher or lower than his current percentage, 0.495? Justify your answer using a statistical argument.

---

2. **Clustering methods**. `Seeds` dataset contains various physical measurements of three different types of seeds. Instead of constructing a classification model to predict the type of seed, we will try to find if there are natural groupings using clustering methods and see if the grouping correspond to types of seeds.

```
seeds = read.table('http://archive.ics.uci.edu/ml/machine-learning-databases/00236/seeds_dataset.txt')
names(seeds) = c('area', 'perimeter', 'compactness', 'length',
                 'width', 'asymmetry', 'groovelength', 'type')
```

```
seeds.label = factor(seeds$type)
seeds.orig = seeds[,-ncol(seeds)]
seeds = scale(seeds.orig)
seeds = as.data.frame(seeds)
```

(a) Use default settings in `prcomp()` to perform PCA and name the result `seeds.pca`. Summarize `seeds.pca` and state how much of the variance in `seeds` is explained by first three principal components.

The first three components explain such a large porportion of variance (i.e. information) of the data because the columns in `seeds` are closely related quantities: `area` would be function of `length` and `width`, etc. As a result, some of the columns do not provide much additional information. Verify this claim by computing the correlation matrix between all predictor variables.

(b) **K-means clustering** Save the first two principle components `PC1` and `PC2` into a two-column data frame, call it `seeds.pc2`. Treat `seeds.pc2` as a new dataset which contains 88.98% information in `seeds`. We want to perform clustering based on `seeds.pc2`.

i. Use `NbClust()` and `index=trcovw` to determine the best number of clusters. Describe what index `trcovw` is in plain words. Save the optimal number as `best.km`. What is the value of index being computed? (refer to https://www.jstatsoft.org/article/view/v061i06/v61i06.pdf

ii. Set seed value by `set.seed(10)`. Use `kmeans()` and `best.km` to perform a k-means clustering. Use basic `plot()` function to visualize cluster assignments (use different colors to indicate different clusters) and use `points()` to mark the cluster centers.

(c) **Hierachical clustering** Compare different linkages.

i. Obtain the euclidean distance matrix for `seeds.pc2` and call it `dist.pc2`.

ii. Use `NbClust()` and `index="all"` to determine the best number of clusters for a hierarchical clustering with **s**ingle linkage, **c**omplete linkage and **a**verage linkage, respectively. Call the best number of clusters `best.s`, `best.c` and `best.a`.

iii. Set seed value by `set.seed(10)`. Use `hclust()` to find three clustering results with different linkage functions. Name them `hc.s`, `hc.c` and `hc.a`. (Specify different linkage names in `method` to have hierarchical clustering results for single linkage, complete linkage and average linkage respectively).

iv. Plot three dendrograms for `hc.s`, `hc.c` and `hc.a`. Give a clear title for each plot indicating which linkage was used to have the dendrogram.

v. Cut the three dendrograms of `hc.s`, `hc.c` and `hc.a` so as to get `best.s`, `best.c` and `best.a` clusters, respectively (Hint: `cutree()`). Use basic `plot()` function to visualize each cluster assignments (use different colors to indicate different clusters).

vi. Use `NbClust()` with `index="ball"` to find best number of cluster again. Show scatter plot of corresponding cluster assignments. [3]

---

3. **PCA**. Bitmap image can be read in via the following command:

```
img = read.bmp('image1.bmp')
img = t(img[ncol(img):1,])
img = img - mean(img)
```

Plot the image in grayscale:

```
gs = grey((0:255)/255)
image(img, asp=1, col=gs, yaxs='r', xaxt='n', yaxt='n')
```

(a) Using syntax `??[keyword]`, help pages can be searched for any pages with `keyword` in it. Also, if there are same function names in multiple packages, a package can be specified by `?[packagename]::[functionname]`.

Using the search method, find what the keyword `xaxt` and `yaxt` does in the above `image()` function by looking up the appropriate help page.

(b) Compute the principal components using `prcomp` and list objects in the function output: i.e. `str` function would be useful.

(c) Recall that principal components are linear combination of data columns.

$$Z_i = \phi_{i1}X_1 + \phi_{i2}X_2 + \cdots + \phi_{ip}X_p.$$

Verify that this is true by multiplying data matrix (original bitmap image `img` or a.k.a $X$) by loadings (`pca.img$rotation` object or a.k.a matrix of $\phi_{ij}$) and compare to computed principal components (`pca.img$x` object or a.k.a $Z$'s): i.e. compute to verify that

$$\|Z - X\Phi\|_F^2 \approx 0, \qquad \text{(up to numerical error)}$$

where $\|M\|_F^2 = \sum_{i,j} M_{ij}^2$.

(d) Check that `rotation` of the `prcomp` output is indeed a rotation matrix, say $Q$, by verifying a crucial property of orthonormal rotation matrices: i.e.

$$\|Q^TQ - I\|_F^2 \approx 0 \qquad \text{(up to numerical error)}$$

(e) This means we can approximately reconstruct original data using any number of principal components we choose:

$$Z\Phi^T - X\Phi\Phi^T = Z\Phi^T - X \approx Z[,1:m]\,\Phi[,1:m]^T - X$$

where $[,1:m]$ is `R` notation for taking submatrix of columns 1 through $m$.

Using this fact, reconstruct the image from 10 and 100 principal components and plot the reconstructed image.

(f) Plot proportion of variance explained as function of number of principal components and also cumulative proportional variance explained. The function `summary` returns helpful objects including PVE.

Using this information, find out how many principal components are needed to explain 90% of the variance.

---

4. **Logistic regression with polynomial features**

(a) In class, we have used polynomial linear regression several times as an example for model complexity and the bias variance tradeoff. We can also introduce polynomial logistic regression models to derive more sophisticated classification functions by introducing additional features. Use `read_csv` to load `nonlinear.csv` and plot the data. Plot each point colored according to its class, `Y`.

(b) Fit a logistic regression model of `Y` on `X1` and `X2`. Visualizing the decision boundary. The decision boundary can be visualized by making predictions of class labels over finely sampled grid points that cover your region (sample space) of interest. The following code will create grid points over the sample space as below:

```
# grid of points over sample space
gr <- expand.grid(X1=seq(-5, 5, by=0.1),  # sample points in X1
                  X2=seq(-5, 5, by=0.1))  # sample points in X2
```

For each point in `gr`, predict a class label using the logistic regression model. You can classify based on the probability being greater or less than 0.4 Plot predictions at each point on the grid, again colored by class label.

(c) Fit a model involving 2nd degree polynomial of `X1` and `X2` with interaction terms. You should use the `poly()` function. Inspect result of the fit using `summary()`. Plot the resulting decision boundary and discuss the result.

(d) Using the same procedure, fit a logistic regression model with 5-th degree polynomials without any interaction terms. Inspect result of the fit using `summary()`. Plot the resulting decision boundary and discuss the result.

(e) Qualitatively, compare the relative magnitudes of coefficients of in the two polynomial models and the linear model. What do you notice? Your answer should mention bias, variance and/or overfitting.

(f) **(231 Only)** Create 3 bootstrap replicates of the original dataset. Fit the linear model and the 5th order polynomial to each of the bootstrap replicates. Plot class predictions on the grid of values for each of both linear and 5th order fits, from each of the bootstrap samples. There should be six plots total. Discuss what you see in the context of your answer to the previous question.