Assignment 8

Q.1

Step 1: Collecting Data

```
# Importing dataset
teens <- read.csv("C:/Users/Meghana Nadig/Downloads/snsdata.csv")

str(teens)
```

```
## 'data.frame':    30000 obs. of  40 variables:
##  $ gradyear    : int  2006 2006 2006 2006 2006 2006 2006 2006 2006 2006 ...
##  $ gender      : Factor w/ 2 levels "F","M": 2 1 2 1 NA 1 1 2 1 1 ...
##  $ age         : num  19 18.8 18.3 18.9 19 ...
##  $ friends     : int  7 0 69 0 10 142 72 17 52 39 ...
##  $ basketball  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ football    : int  0 1 1 0 0 0 0 0 0 0 ...
##  $ soccer      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ softball    : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ volleyball  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ swimming    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ cheerleading: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ baseball    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ tennis      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ sports      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ cute        : int  0 1 0 1 0 0 0 0 0 1 ...
##  $ sex         : int  0 0 0 0 1 1 0 2 0 0 ...
##  $ sexy        : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ hot         : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ kissed      : int  0 0 0 0 5 0 0 0 0 0 ...
##  $ dance       : int  1 0 0 0 1 0 0 0 0 0 ...
##  $ band        : int  0 0 2 0 1 0 1 0 0 0 ...
##  $ marching    : int  0 0 0 0 0 1 1 0 0 0 ...
##  $ music       : int  0 2 1 0 3 2 0 1 0 1 ...
##  $ rock        : int  0 2 0 1 0 0 0 1 0 1 ...
##  $ god         : int  0 1 0 0 1 0 0 0 0 6 ...
##  $ church      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ jesus       : int  0 0 0 0 0 0 0 0 0 2 ...
##  $ bible       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hair        : int  0 6 0 0 1 0 0 0 0 1 ...
##  $ dress       : int  0 4 0 0 0 1 0 0 0 0 ...
##  $ blonde      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ mall        : int  0 1 0 0 0 0 2 0 0 0 ...
##  $ shopping    : int  0 0 0 0 2 1 0 0 0 1 ...
##  $ clothes     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hollister   : int  0 0 0 0 0 0 2 0 0 0 ...
##  $ abercrombie : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ die         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ death       : int  0 0 1 0 0 0 0 0 0 0 ...
##  $ drunk       : int  0 0 0 0 1 1 0 0 0 0 ...
##  $ drugs       : int  0 0 0 0 1 0 0 0 0 0 ...
```

Step 2: Exploring and preparing the data

```
# Spotting missing values

table(teens$gender, useNA = "ifany")
```

```
##
##     F     M  <NA>
## 22054  5222  2724
```

```
summary(teens$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   3.086  16.312  17.287  17.994  18.259 106.927    5086
```

```
# Data cleaning

teens$age <- ifelse(teens$age >= 13 & teens$age < 20, teens$age, NA)

summary(teens$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   13.03   16.30   17.27   17.25   18.22   20.00    5523
```

```
# Dummy coding missing values

teens$female <- ifelse(teens$gender == "F" & !is.na(teens$gender), 1, 0)

teens$no_gender <- ifelse(is.na(teens$gender), 1, 0)

table(teens$gender, useNA = "ifany")
```

```
##
##     F     M  <NA>
## 22054  5222  2724
```

```
table(teens$female, useNA = "ifany")
```

```
##
##     0     1
##  7946 22054
```

```
table(teens$no_gender, useNA = "ifany")
```

```
##
##     0     1
## 27276  2724
```

```
# Imputing the missing values

mean(teens$age, na.rm = TRUE)
```

```
## [1] 17.25243
```

```
aggregate(data = teens, age ~ gradyear, mean, na.rm = TRUE)
```

```
##   gradyear      age
## 1     2006 18.65586
## 2     2007 17.70617
## 3     2008 16.76770
## 4     2009 15.81957
```

```
# Imputing missing values

ave_age <- ave(teens$age, teens$gradyear, FUN = function(x) mean(x, na.rm = TRUE))

teens$age <- ifelse(is.na(teens$age), ave_age, teens$age)

summary(teens$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   13.03   16.28   17.24   17.24   18.21   20.00
```

Step 3: Training a model on the data

```
library(stats)

# Selecting 36 features
interests <- teens[5:40]

# Z-score standardization
interests_z <- as.data.frame(lapply(interests, scale))

set.seed(2345)

# Model
teen_clusters <- kmeans(interests_z, 5)
```

Step 4: Evaluating model performance

```
# Obtaining the size of clusters
teen_clusters$size
```

```
## [1]   871   600  5981  1034 21514
```

```
# Examining the coordinates of clusters
teen_clusters$centers
```

```
##    basketball   football      soccer    softball  volleyball    swimming
## 1  0.16001227  0.2364174  0.10385512  0.07232021  0.18897158  0.23970234
## 2 -0.09195886  0.0652625 -0.09932124 -0.01739428 -0.06219308  0.03339844
## 3  0.52755083  0.4873480  0.29778605  0.37178877  0.37986175  0.29628671
## 4  0.34081039  0.3593965  0.12722250  0.16384661  0.11032200  0.26943332
## 5 -0.16695523 -0.1641499 -0.09033520 -0.11367669 -0.11682181 -0.10595448
##   cheerleading    baseball      tennis      sports        cute
## 1    0.3931445  0.02993479  0.13532387  0.10257837  0.37884271
## 2   -0.1101103 -0.11487510  0.04062204 -0.09899231 -0.03265037
## 3    0.3303485  0.35231971  0.14057808  0.32967130  0.54442929
## 4    0.1856664  0.27527088  0.10980958  0.79711920  0.47866008
## 5   -0.1136077 -0.10918483 -0.05097057 -0.13135334 -0.18878627
##            sex        sexy         hot      kissed       dance        band
## 1  0.020042068  0.11740551  0.41389104  0.06787768  0.22780899 -0.10257102
## 2 -0.042486141 -0.04329091 -0.03812345 -0.04554933  0.04573186  4.06726666
## 3  0.002913623  0.24040196  0.38551819 -0.03356121  0.45662534 -0.02120728
## 4  2.028471066  0.51266080  0.31708549  2.97973077  0.45535061  0.38053621
## 5 -0.097928345 -0.09501817 -0.13810894 -0.13535855 -0.15932739 -0.12167214
##      marching      music        rock         god      church       jesus
## 1 -0.10942590  0.1378306  0.05905951  0.03651755 -0.00709374  0.01458533
## 2  5.25757242  0.4981238  0.15963917  0.09283620  0.06414651  0.04801941
```

```
## 3 -0.10880541  0.2844999  0.21436936  0.35014919  0.53739806  0.27843424
## 4 -0.02014608  1.1367885  1.21013948  0.41679142  0.16627797  0.12988313
## 5 -0.11098063 -0.1532006 -0.12460034 -0.12144246 -0.15889274 -0.08557822
##         bible        hair       dress      blonde        mall    shopping
## 1 -0.03692278  0.43807926  0.14905267  0.06137340  0.60368108  0.79806891
## 2  0.05863810 -0.04484083  0.07201611 -0.01146396 -0.08724304 -0.03865318
## 3  0.22990963  0.23612853  0.39407628  0.03471458  0.48318495  0.66327838
## 4  0.08478769  2.55623737  0.53852195  0.36134138  0.62256686  0.27101815
## 5 -0.06813159 -0.20498730 -0.14348036 -0.02918252 -0.18625656 -0.22865236
##         clothes  hollister abercrombie          die       death
## 1  0.5651537331  4.1521844  3.96493810  0.043475966  0.09857501
## 2 -0.0003526292 -0.1678300 -0.14129577  0.009447317  0.05135888
## 3  0.3759725120 -0.0553846 -0.07417839  0.037989066  0.11972190
## 4  1.2306917174  0.1610784  0.26324494  1.712181870  0.93631312
## 5 -0.1865419798 -0.1557662 -0.14861104 -0.094875180 -0.08370729
##          drunk       drugs
## 1  0.035614771  0.03443294
## 2 -0.086773220 -0.06878491
## 3 -0.009688746 -0.05973769
## 4  1.897388200  2.73326605
## 5 -0.087520105 -0.11423381
```

Step 5: Improving model performance

```
# Adding the cluster column to dataset

teens$cluster <- teen_clusters$cluster
```

```
# First 5 teens

teens[1:5, c("cluster", "gender", "age", "friends")]
```

```
##   cluster gender    age friends
## 1       5      M 18.982       7
## 2       3      F 18.801       0
## 3       5      M 18.335      69
## 4       5      F 18.875       0
## 5       4   <NA> 18.995      10
```

```
# Demographic characteristcs

aggregate(data = teens, age ~ cluster, mean)
```

```
##   cluster      age
## 1       1 16.86497
## 2       2 17.39037
## 3       3 17.07656
## 4       4 17.11957
## 5       5 17.29849
```

```
# Females by cluster

aggregate(data = teens, female ~ cluster, mean)
```

```
##   cluster    female
## 1       1 0.8381171
## 2       2 0.7250000
```

```
## 3       3 0.8378198
## 4       4 0.8027079
## 5       5 0.6994515
```

```
# Number of friends per user

aggregate(data = teens, friends ~ cluster, mean)
```

```
##   cluster  friends
## 1       1 41.43054
## 2       2 32.57333
## 3       3 37.16185
## 4       4 30.50290
## 5       5 27.70052
```

Problem 2:

1. What are various ways to predict a binary response variable? Can you compare two of them and tell me when one would be more appropriate? What's the difference between these? (SVM, Logistic Regression, Naive Bayes, Decision Tree, etc.

The different ways to predict binary response variable is through SVM, logistic regression, naïve bayes and decision tree.

SVM: SVM works well with both linearly and non linearly separable data. It gives better accuracy when compared to other classification algorithms.It is used mahorly for text classification.

Logistic Regression: Easy to update the model and need not worry about correlatiob between features. Generally used in case of probabilistic framework or expect to receive more training data in the future.

Naïve Bayes: It is quite simple and generally requires less training data. It works surprisingly well in practice even if the naïve bayes assumption doesn't hold up.

Decision Tree: It is simple, easy to interpret white box approach for classifying so we don't have to worry about outliers or whether the data is linearly separable. Their main disadvantage is that they easily overfit.

Comparing logistic regression and decision trees, decision trees are more accurate than logistic regression but logistic regression can be updated online and gives us useful probabilities.

2. Why might it be preferable to include fewer predictors over many?

Too many predictors leads to problem of overfitting.

To avoid the problem of over-fitting:

a) Reduce the number of features by manually selecting only required features or using a model selection algorithm using feature engineering or PCA.

b) Use regularization: if we throw away lot of features which are actually useful then regularization is much more helpful than just reducing the number of features.

3. Given a database of all previous alumni donations to your university, how would you predict which recent alumni are most likely to donate?

Logistic regression measures the relationship between the categorical dependent variable, which would be likely to donate or not, and one or more independent variables. I would hope the database has data on its alumni to represent the independent variables such as age, major, address, salary, occupation and any other demographic data.

The logistic regression works by estimating probabilities using a logistic function.

4. What is R-Squared? What are some other metrics that could be better than R-Squared and why?

R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.

R-squared is always between 0 and 100%:

0% indicates that the model explains none of the variability of the response data around its mean. 100% indicates that the model explains all the variability of the response data around its mean. In general, the higher the R-squared, the better the model fits your data.

AIC and BIC are alternative metrics.The AIC (Akaike Information Criterion) Metric describes the quality of the model with the data that is given.It is is the trade-off between goodness of fit and complexity of the variables that are considered in the problem. R-Squared changes relative to the complexity of the system (variables) but AIC does not. The BIC (Bayesian Information Criterion) is closely related to AIC except for it uses a Bayesian (probability) argument to figure out the goodness to fit. It also has the same advantage over the R-Squared metric in that complex problems are less impacted with AIC or BIC vs. R-Squared method.

5. How can you determine which features are the most important in your model?

Regular regression coefficients describe the relationship between each predictor variable and the response. The coefficient value represents the mean change in the response given a one-unit increase in the predictor. Consequently, it's easy to think that variables with larger coefficients are more important because they represent a larger change in the response.

However, the units vary between the different types of variables, which makes it impossible to compare them directly.

This problem is further complicated by the fact that there are different units within each type of measurement. If you fit models for the same data set using different scale, the coefficient for weight changes by a factor of a thousand even though the underlying fit of the model remains unchanged. The coefficient value changes greatly while the importance of the variable remains constant.Thus, larger coefficients don't necessarily identify more important predictor variables.If it's a random forest, then we can derive the feature importances based on the entropy.