Machine Learning 2019: Feature Selection

Sonali Narang

October 24, 2019

Feature Selection

In machine learning, feature selection is the process of choosing variables that are useful in predicting the response variable. Selecting the right features in your data can mean the difference between mediocre performance with long training times and great performance with short training times that are less computationally intensive.

Often, data can contain attributes that are highly correlated with each other or not useful in helping predict our response variable. Many methods perform better if such variables are removed. Feature selection is usually imporant to implement during the data pre-processing steps of machine learning.

The Breast Cancer Dataset

699 Observations, 11 variables Predictor Variable: Class- benign or malignant

```
data(BreastCancer)
head(BreastCancer)
```

```
##        Id Cl.thickness Cell.size Cell.shape Marg.adhesion Epith.c.size
## 1 1000025            5         1          1             1            2
## 2 1002945            5         4          4             5            7
## 3 1015425            3         1          1             1            2
## 4 1016277            6         8          8             1            3
## 5 1017023            4         1          1             3            2
## 6 1017122            8        10         10             8            7
##   Bare.nuclei Bl.cromatin Normal.nucleoli Mitoses     Class
## 1           1           3               1       1    benign
## 2          10           3               2       1    benign
## 3           2           3               1       1    benign
## 4           4           3               7       1    benign
## 5           1           3               1       1    benign
## 6          10           9               7       1 malignant
```

```
dim(BreastCancer)
```

```
## [1] 699  11
```

```
summary(BreastCancer$Class)
```

```
##    benign malignant
##       458       241
```

Feature Selection Using Filter Methods: Pearson's Correlation

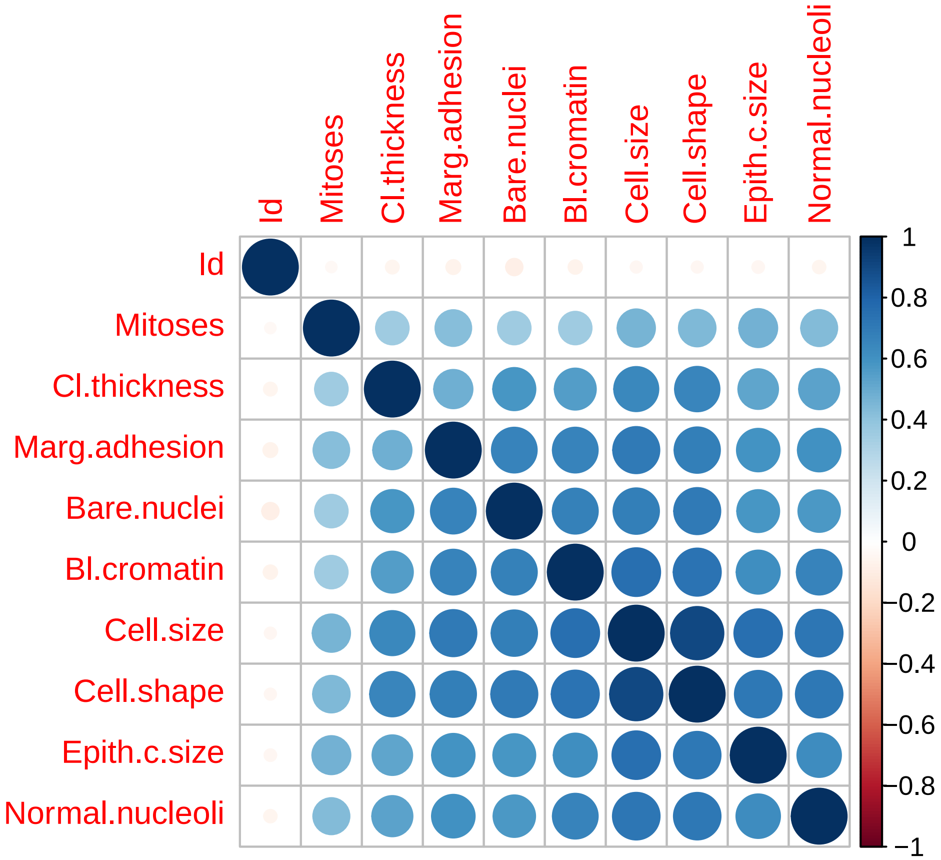
Filter Methods are generally used as a preprocessing step so the selection of features is independednt of any machine learning algorithms. Features are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable.

Below we will identify attributes that are highly correlated using Pearson's correlation which is a measure for quantifying linear dependence between X and Y. Ranges between -1 and 1.

```
BreastCancer_num = transform(BreastCancer, Id = as.numeric(Id),
                             Cl.thickness = as.numeric(Cl.thickness),
                             Cell.size = as.numeric(Cell.size),
                             Cell.shape = as.numeric(Cell.shape),
                             Marg.adhesion = as.numeric(Marg.adhesion),
                             Epith.c.size = as.numeric(Epith.c.size),
                             Bare.nuclei = as.numeric(Bare.nuclei),
                             Bl.cromatin = as.numeric(Bl.cromatin),
                             Normal.nucleoli = as.numeric(Normal.nucleoli),
                             Mitoses = as.numeric(Mitoses))

BreastCancer_num[is.na(BreastCancer_num)] = 0

#calculate correlation matrix using pearson correlation (others include spearman and kendall)
correlation_matrix = cor(BreastCancer_num[,1:10])

#visualize correlation matrix
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
corrplot(correlation_matrix, order = "hclust")
```



```
#apply correlation filter of 0.7
highly_correlated <- colnames(BreastCancer[, -1])[findCorrelation(correlation_matrix, cutoff = 0.7, verb
```

```
## Compare row 3  and column  4 with corr  0.907
##   Means:  0.631 vs 0.477 so flagging column 3
## Compare row 4  and column  8 with corr  0.736
##   Means:  0.588 vs 0.447 so flagging column 4
## All correlations <= 0.7
```

```
#which features are highly correlated and can be removed
highly_correlated
```

```
## [1] "Cell.shape"    "Marg.adhesion"
```

Feature Selection Using Wrapper Methods: Recursive Feature Elimination (RFE)

Wrapper methods are a bit more computationally intensive since we will select features based on a specific machine learning algorith.

The RFE function implements backwards selection of predictors based on predictor importance ranking. The predictors are ranked and the less important ones are sequentially eliminated prior to modeling. The goal is to find a subset of predictors that can be used to produce an accurate model.

```
data(BreastCancer)
BreastCancer_num = transform(BreastCancer, Id = as.numeric(Id),
                         Cl.thickness = as.numeric(Cl.thickness),
                         Cell.size = as.numeric(Cell.size),
                         Cell.shape = as.numeric(Cell.shape),
                         Marg.adhesion = as.numeric(Marg.adhesion),
                         Epith.c.size = as.numeric(Epith.c.size),
                         Bare.nuclei = as.numeric(Bare.nuclei),
                         Bl.cromatin = as.numeric(Bl.cromatin),
                         Normal.nucleoli = as.numeric(Normal.nucleoli),
                         Mitoses = as.numeric(Mitoses))

BreastCancer_num[is.na(BreastCancer_num)] = 0

#define the control
control = rfeControl(functions = caretFuncs, number = 2)

# run the RFE algorithm
results = rfe(BreastCancer_num[,1:10], BreastCancer_num[,11], sizes = c(2,5,9), rfeControl = control, me

results
```

```
##
## Recursive feature selection
##
## Outer resampling method: Bootstrapped (2 reps)
##
## Resampling performance over subset size:
##
##  Variables Accuracy  Kappa AccuracySD  KappaSD Selected
##          2   0.9276 0.8413  0.0182731 0.029534
##          5   0.9578 0.9087  0.0021216 0.008415
##          9   0.9658 0.9258  0.0034219 0.003940        *
##         10   0.9598 0.9128  0.0006844 0.002102
##
## The top 5 variables (out of 9):
```

```
##    Cell.size, Cell.shape, Bare.nuclei, Bl.cromatin, Epith.c.size
```

```r
results$variables
```

```
##       benign malignant   Overall             var Variables  Resample
## 1  0.9739169 0.9739169 0.9739169      Cell.shape        10 Resample1
## 2  0.9711582 0.9711582 0.9711582       Cell.size        10 Resample1
## 3  0.9597112 0.9597112 0.9597112     Bare.nuclei        10 Resample1
## 4  0.9526627 0.9526627 0.9526627     Bl.cromatin        10 Resample1
## 5  0.9369575 0.9369575 0.9369575    Epith.c.size        10 Resample1
## 6  0.9085823 0.9085823 0.9085823    Cl.thickness        10 Resample1
## 7  0.9064759 0.9064759 0.9064759   Marg.adhesion        10 Resample1
## 8  0.8995950 0.8995950 0.8995950 Normal.nucleoli        10 Resample1
## 9  0.7337060 0.7337060 0.7337060         Mitoses        10 Resample1
## 10 0.5641795 0.5641795 0.5641795              Id        10 Resample1
## 11 0.9739169 0.9739169 0.9739169      Cell.shape         9 Resample1
## 12 0.9711582 0.9711582 0.9711582       Cell.size         9 Resample1
## 13 0.9597112 0.9597112 0.9597112     Bare.nuclei         9 Resample1
## 14 0.9526627 0.9526627 0.9526627     Bl.cromatin         9 Resample1
## 15 0.9369575 0.9369575 0.9369575    Epith.c.size         9 Resample1
## 16 0.9085823 0.9085823 0.9085823    Cl.thickness         9 Resample1
## 17 0.9064759 0.9064759 0.9064759   Marg.adhesion         9 Resample1
## 18 0.8995950 0.8995950 0.8995950 Normal.nucleoli         9 Resample1
## 19 0.7337060 0.7337060 0.7337060         Mitoses         9 Resample1
## 20 0.9739169 0.9739169 0.9739169      Cell.shape         5 Resample1
## 21 0.9711582 0.9711582 0.9711582       Cell.size         5 Resample1
## 22 0.9597112 0.9597112 0.9597112     Bare.nuclei         5 Resample1
## 23 0.9526627 0.9526627 0.9526627     Bl.cromatin         5 Resample1
## 24 0.9369575 0.9369575 0.9369575    Epith.c.size         5 Resample1
## 25 0.9739169 0.9739169 0.9739169      Cell.shape         2 Resample1
## 26 0.9711582 0.9711582 0.9711582       Cell.size         2 Resample1
## 27 0.9762179 0.9762179 0.9762179       Cell.size        10 Resample2
## 28 0.9674685 0.9674685 0.9674685      Cell.shape        10 Resample2
## 29 0.9451294 0.9451294 0.9451294     Bl.cromatin        10 Resample2
## 30 0.9434764 0.9434764 0.9434764     Bare.nuclei        10 Resample2
## 31 0.9167854 0.9167854 0.9167854    Epith.c.size        10 Resample2
## 32 0.9057757 0.9057757 0.9057757   Marg.adhesion        10 Resample2
## 33 0.9043875 0.9043875 0.9043875    Cl.thickness        10 Resample2
## 34 0.8854914 0.8854914 0.8854914 Normal.nucleoli        10 Resample2
## 35 0.7216058 0.7216058 0.7216058         Mitoses        10 Resample2
## 36 0.5787029 0.5787029 0.5787029              Id        10 Resample2
## 37 0.9762179 0.9762179 0.9762179       Cell.size         9 Resample2
## 38 0.9674685 0.9674685 0.9674685      Cell.shape         9 Resample2
## 39 0.9451294 0.9451294 0.9451294     Bl.cromatin         9 Resample2
## 40 0.9434764 0.9434764 0.9434764     Bare.nuclei         9 Resample2
## 41 0.9167854 0.9167854 0.9167854    Epith.c.size         9 Resample2
## 42 0.9057757 0.9057757 0.9057757   Marg.adhesion         9 Resample2
## 43 0.9043875 0.9043875 0.9043875    Cl.thickness         9 Resample2
## 44 0.8854914 0.8854914 0.8854914 Normal.nucleoli         9 Resample2
## 45 0.7216058 0.7216058 0.7216058         Mitoses         9 Resample2
## 46 0.9762179 0.9762179 0.9762179       Cell.size         5 Resample2
## 47 0.9674685 0.9674685 0.9674685      Cell.shape         5 Resample2
## 48 0.9451294 0.9451294 0.9451294     Bl.cromatin         5 Resample2
## 49 0.9434764 0.9434764 0.9434764     Bare.nuclei         5 Resample2
## 50 0.9167854 0.9167854 0.9167854    Epith.c.size         5 Resample2
## 51 0.9762179 0.9762179 0.9762179       Cell.size         2 Resample2
## 52 0.9674685 0.9674685 0.9674685      Cell.shape         2 Resample2
```

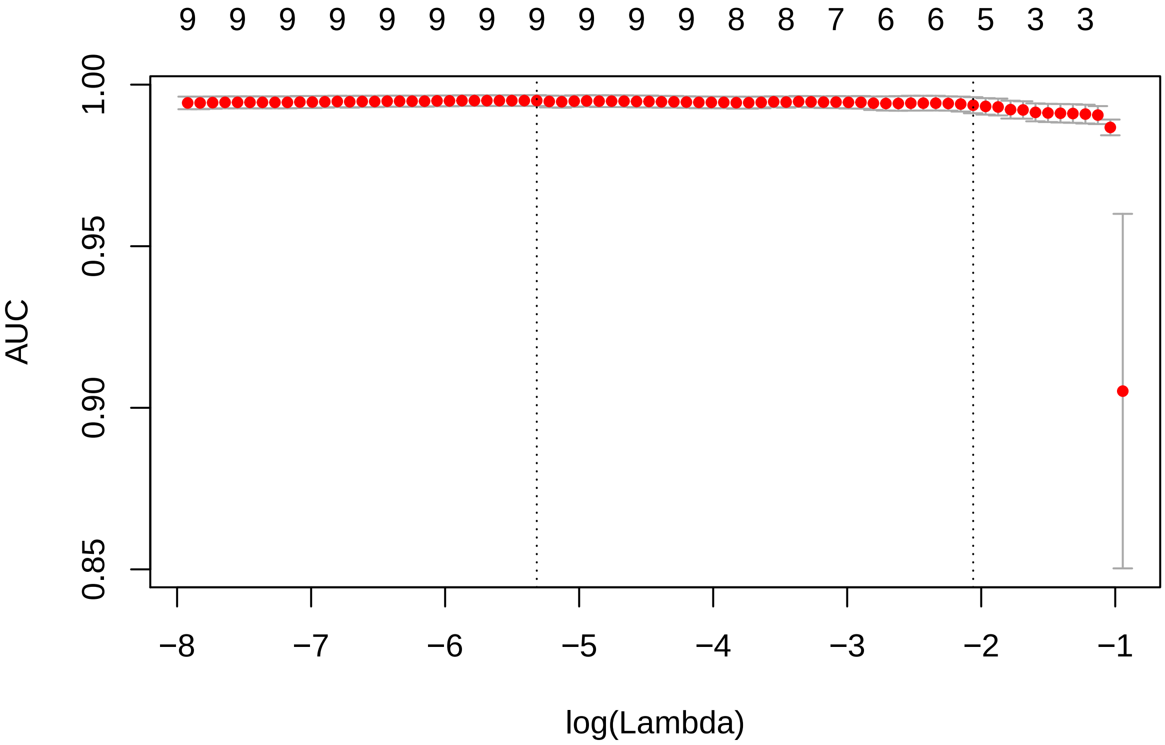
Feature Selection Using Embedded Methods: Lasso

Least Absolute Shrinkage and Selection Operator (LASSO) regression

```
set.seed(24)

#convert data
x = x <- as.matrix(BreastCancer_num[,1:10])
y = as.double(as.matrix(ifelse(BreastCancer_num[,11]=='benign', 0, 1)))

#fit Lasso model
cv.lasso <- cv.glmnet(x, y, family='binomial', alpha=1, parallel=TRUE, standardize=TRUE, type.measure='a
```

```
## Warning: executing %dopar% sequentially: no parallel backend registered
```

```
plot(cv.lasso)
```



```
cat('Min Lambda: ', cv.lasso$lambda.min, '\n 1Sd Lambda: ', cv.lasso$lambda.1se)
```

```
## Min Lambda:  0.0049116
##  1Sd Lambda:  0.1274572
```

```
df_coef <- round(as.matrix(coef(cv.lasso, s=cv.lasso$lambda.min)), 2)

# See all contributing variables
df_coef[df_coef[, 1] != 0, ]
```

```
##     (Intercept)    Cl.thickness       Cell.size      Cell.shape
##           -7.97            0.44            0.06            0.28
##   Marg.adhesion    Epith.c.size     Bare.nuclei     Bl.cromatin
##             0.16            0.05            0.37            0.33
## Normal.nucleoli         Mitoses
##             0.14            0.23
```

Feature Selection Using Embedded Methods: RandomForest

Random Forest Importance function and caret package's varImp functions perform similarly.

```
#data
data(BreastCancer)
train_size <- floor(0.75 * nrow(BreastCancer))
set.seed(24)
train_pos <- sample(seq_len(nrow(BreastCancer)), size = train_size)

#convert to numeric
BreastCancer_num = transform(BreastCancer, Id = as.numeric(Id),
                         Cl.thickness = as.numeric(Cl.thickness),
                         Cell.size = as.numeric(Cell.size),
                         Cell.shape = as.numeric(Cell.shape),
                         Marg.adhesion = as.numeric(Marg.adhesion),
                         Epith.c.size = as.numeric(Epith.c.size),
                         Bare.nuclei = as.numeric(Bare.nuclei),
                         Bl.cromatin = as.numeric(Bl.cromatin),
                         Normal.nucleoli = as.numeric(Normal.nucleoli),
                         Mitoses = as.numeric(Mitoses))

BreastCancer_num[is.na(BreastCancer_num)] = 0

train_classification <- BreastCancer_num[train_pos, ]
test_classification <- BreastCancer_num[-train_pos, ]

#fit a model
rfmodel = randomForest(Class ~ Id + Cl.thickness + Cell.size + Cell.shape + Marg.adhesion + Epith.c.siz

#rank features based on importance
importance(rfmodel)
```

```
##                     benign malignant MeanDecreaseAccuracy MeanDecreaseGini
## Id              -0.6895607  6.306423             5.356937         4.753537
## Cl.thickness    20.1614358 21.864276            24.672793        15.817233
## Cell.size       13.2922349 16.005349            20.854493        51.850109
## Cell.shape       9.9205845 15.663444            18.547503        52.135873
## Marg.adhesion    6.9732478  8.757817            11.298839         7.495039
## Epith.c.size     8.2669770  3.558679             8.988390        14.948179
## Bare.nuclei     18.7963652 27.340734            28.123604        42.322414
## Bl.cromatin      8.5784618 14.394368            16.261622        28.772624
## Normal.nucleoli 11.9915409  9.888276            14.484327        19.497268
## Mitoses          6.3442746  2.271364             6.768548         1.615123
```

Homework

1. Compare the most important features from at least 2 different classes of feature selection methods covered in this tutorial with any reasonable machine learning dataset from mlbench. Do these feature selection methods provide similar results?

```r
##dataset selection and exploration
data(PimaIndiansDiabetes)
head(PimaIndiansDiabetes)
```

```
##   pregnant glucose pressure triceps insulin mass pedigree age diabetes
## 1        6     148       72      35       0 33.6    0.627  50      pos
## 2        1      85       66      29       0 26.6    0.351  31      neg
## 3        8     183       64       0       0 23.3    0.672  32      pos
## 4        1      89       66      23      94 28.1    0.167  21      neg
## 5        0     137       40      35     168 43.1    2.288  33      pos
## 6        5     116       74       0       0 25.6    0.201  30      neg
```

```r
dim(PimaIndiansDiabetes)
```

```
## [1] 768   9
```

```r
##summary of outcome variable of interest
summary(PimaIndiansDiabetes$diabetes)
```

```
## neg pos
## 500 268
```

```r
##WRecursive Feature Elimination
PimaIndians = transform(PimaIndiansDiabetes, pregnant = as.numeric(pregnant),
                         glucose = as.numeric(glucose),
                         pressure = as.numeric(pressure),
                         triceps = as.numeric(triceps),
                         insulin = as.numeric(insulin),
                         mass = as.numeric(mass),
                         pedigree = as.numeric(pedigree),
                         age = as.numeric(age))

PimaIndians[is.na(PimaIndians)] = 0
control = rfeControl(functions = caretFuncs, number = 2)
results = rfe(PimaIndians[,1:8], PimaIndians[,9], sizes = c(2,5,9), rfeControl = control, method = "svm

##RFE output
results
```

```
##
## Recursive feature selection
##
## Outer resampling method: Bootstrapped (2 reps)
##
## Resampling performance over subset size:
##
##  Variables Accuracy  Kappa AccuracySD KappaSD Selected
##          2   0.7689 0.4358   0.008928 0.02290
##          5   0.7889 0.5002   0.028926 0.08775
##          8   0.8023 0.5161   0.014681 0.06207        *
##
## The top 5 variables (out of 8):
##    glucose, mass, age, pregnant, pedigree
```

```r
results$variables
```

```
##          neg       pos    Overall      var Variables  Resample
## 1  0.7815753 0.7815753 0.7815753  glucose         8 Resample1
## 2  0.6908925 0.6908925 0.6908925     mass         8 Resample1
## 3  0.6796231 0.6796231 0.6796231      age         8 Resample1
## 4  0.6154618 0.6154618 0.6154618 pedigree         8 Resample1
## 5  0.6094349 0.6094349 0.6094349 pregnant         8 Resample1
## 6  0.5747782 0.5747782 0.5747782 pressure         8 Resample1
## 7  0.5563656 0.5563656 0.5563656  triceps         8 Resample1
## 8  0.5492751 0.5492751 0.5492751  insulin         8 Resample1
## 9  0.7815753 0.7815753 0.7815753  glucose         5 Resample1
## 10 0.6908925 0.6908925 0.6908925     mass         5 Resample1
## 11 0.6796231 0.6796231 0.6796231      age         5 Resample1
## 12 0.6154618 0.6154618 0.6154618 pedigree         5 Resample1
## 13 0.6094349 0.6094349 0.6094349 pregnant         5 Resample1
## 14 0.7815753 0.7815753 0.7815753  glucose         2 Resample1
## 15 0.6908925 0.6908925 0.6908925     mass         2 Resample1
## 16 0.8114108 0.8114108 0.8114108  glucose         8 Resample2
## 17 0.6947272 0.6947272 0.6947272     mass         8 Resample2
## 18 0.6870655 0.6870655 0.6870655      age         8 Resample2
## 19 0.6233187 0.6233187 0.6233187 pressure         8 Resample2
## 20 0.6213952 0.6213952 0.6213952 pregnant         8 Resample2
## 21 0.6015943 0.6015943 0.6015943 pedigree         8 Resample2
## 22 0.5890409 0.5890409 0.5890409  triceps         8 Resample2
## 23 0.5854424 0.5854424 0.5854424  insulin         8 Resample2
## 24 0.8114108 0.8114108 0.8114108  glucose         5 Resample2
## 25 0.6947272 0.6947272 0.6947272     mass         5 Resample2
## 26 0.6870655 0.6870655 0.6870655      age         5 Resample2
## 27 0.6233187 0.6233187 0.6233187 pressure         5 Resample2
## 28 0.6213952 0.6213952 0.6213952 pregnant         5 Resample2
## 29 0.8114108 0.8114108 0.8114108  glucose         2 Resample2
## 30 0.6947272 0.6947272 0.6947272     mass         2 Resample2
```

```
##Random Forest
data("PimaIndiansDiabetes")
train_size <- floor(0.75 * nrow(PimaIndiansDiabetes))
set.seed(24)
train_pos <- sample(seq_len(nrow(PimaIndiansDiabetes)), size = train_size)

PimaIndians = transform(PimaIndiansDiabetes, pregnant = as.numeric(pregnant),
                         glucose = as.numeric(glucose),
                         pressure = as.numeric(pressure),
                         triceps = as.numeric(triceps),
                         insulin = as.numeric(insulin),
                         mass = as.numeric(mass),
                         pedigree = as.numeric(pedigree),
                         age = as.numeric(age))
PimaIndians[is.na(PimaIndians)] = 0

train_classification <- PimaIndians[train_pos, ]
test_classification <- PimaIndians[-train_pos, ]
rfmodel = randomForest(diabetes ~ ., data=train_classification,  importance = TRUE, oob.times = 15, con

##Random Forest feature selection output
importance(rfmodel)
```

```
##                  neg        pos MeanDecreaseAccuracy MeanDecreaseGini
## pregnant 10.025340 -0.8550934             8.084583         21.63270
## glucose  28.804959 29.2639025            39.374520         66.51185
## pressure  3.813558 -2.7043237             1.381561         23.24432
## triceps   3.056432 -0.4982624             2.275994         18.21001
## insulin   8.169029  2.7038144             8.328190         19.78018
## mass     16.214681 16.8173776            23.659914         42.72148
## pedigree  6.003561  3.3776165             6.789677         33.77226
## age      13.157475  6.2415217            15.296014         35.48186
```

Both Recursive Feature Selection and Random Forest for feature selection identified glucose, mass, and age as the most significant predictor variables in decending order. While there was high agreement between the two methods for the top 3 features, there was some deviation with less significant predictors. RFE found pregnancy and pedigree to be the next most significant, whereas RF found insulin and pregnancy to be the next most significant. Overall, the results of the two methods are quite similar.

2. Attempt a feature selection method not covered in this tutorial (backward elimination, forward propogation, etc.)

```r
##load required library for stepwise regression
library(MASS)
```

```r
##Backward elimination of logistic regression
##load dataset
data("PimaIndiansDiabetes")

##transform variables to numeric and eliminate any missing values
PimaIndians = transform(PimaIndiansDiabetes, pregnant = as.numeric(pregnant),
                         glucose = as.numeric(glucose),
                         pressure = as.numeric(pressure),
                         triceps = as.numeric(triceps),
                         insulin = as.numeric(insulin),
                         mass = as.numeric(mass),
                         pedigree = as.numeric(pedigree),
                         age = as.numeric(age))
PimaIndians[is.na(PimaIndians)] = 0

##subset dataset into train and test
train_size <- floor(0.75 * nrow(PimaIndiansDiabetes))
set.seed(24)
train_pos <- sample(seq_len(nrow(PimaIndiansDiabetes)), size = train_size)
train_classification <- PimaIndians[train_pos, ]
test_classification <- PimaIndians[-train_pos, ]

##build logistic regression model
logmodel <- glm(diabetes ~., data = train_classification, family = binomial)
##perform backward elimination on the model created
step <- stepAIC(logmodel, direction="backward")
```

```
## Start:  AIC=570.38
## diabetes ~ pregnant + glucose + pressure + triceps + insulin +
##     mass + pedigree + age
##
##            Df Deviance    AIC
## - insulin   1   553.26 569.26
## - triceps   1   553.28 569.28
## <none>          552.38 570.38
## - age       1   554.94 570.94
## - pressure  1   556.62 572.62
## - pregnant  1   559.33 575.33
## - pedigree  1   561.94 577.94
## - mass      1   589.90 605.90
## - glucose   1   626.37 642.37
##
## Step:  AIC=569.26
## diabetes ~ pregnant + glucose + pressure + triceps + mass + pedigree +
##     age
##
##            Df Deviance    AIC
## <none>          553.26 569.26
## - triceps   1   555.40 569.40
## - age       1   556.09 570.09
## - pressure  1   557.32 571.32
## - pregnant  1   560.52 574.52
## - pedigree  1   562.35 576.35
## - mass      1   592.13 606.13
## - glucose   1   632.26 646.26
```

```
##visualize results
step$anova
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## diabetes ~ pregnant + glucose + pressure + triceps + insulin +
##     mass + pedigree + age
##
## Final Model:
## diabetes ~ pregnant + glucose + pressure + triceps + mass + pedigree +
##     age
##
##
##        Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1                              567   552.3759 570.3759
## 2 - insulin  1 0.8842564       568   553.2602 569.2602
```

Backward Elimination seems to be in high agreement with RFE in evaluating insulin as the least significant predictor variable. The final model includes all variables except this one.