STT 461 HW 5

Derien Weatherspoon

2023-03-20

```
library(MASS)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(boot)
```

```
##
## Attaching package: 'boot'

## The following object is masked from 'package:lattice':
##
##     melanoma
```

```
library(bayesboot)
library(ggcorrplot)
library(DirichletReg)
```

```
## Loading required package: Formula
```

```
library(tree)
```

Question 1

A linear regression model prediction is in the form of y = 2.18x1 - 4.56x2. The root mean square error of the residuals is 0.856. (a) What is the E(y|(x1,x2) = (2, -3)? (b) What is the probability that y > 20, given that (x1,x2) = (2, -3)?

```
# setup values
x1 <- 2
x2 <- -3
y <- 2.18*x1 - 4.56*x2
rmse <- 0.856

#a
Ey <- y # predicted value of y
Ey
```

```
## [1] 18.04
```

```
#b
z <- (20-y)/rmse
1-pnorm(z)
```

```
## [1] 0.01101879
```

Question 2

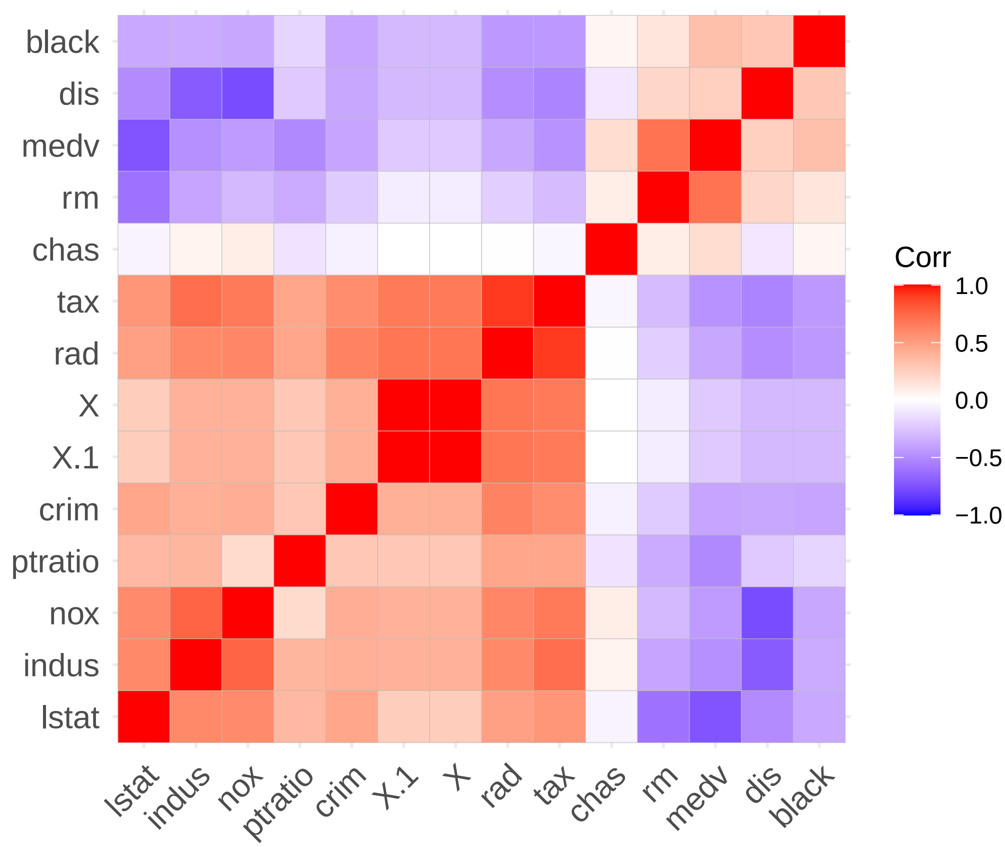
Take the Boston dataset, available in D2L. This data has information about different neighborhoods in Boston, and we will use it to predict the median housing price for the neighborhoods. Here is an explanation of the variables:

(a) Which other variable correlates the strongest with medv? Note that strongest mean largest absolute value, whether it's positive or negative.

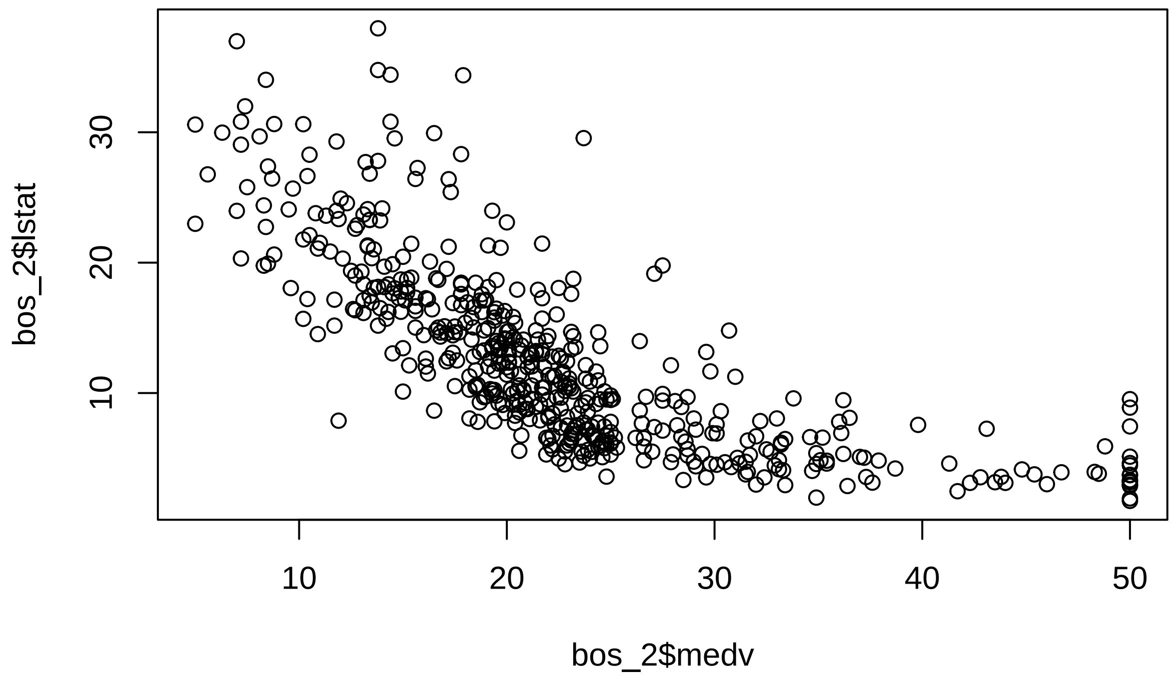
```
bos <- read.csv("Boston.csv")
bos_2 <- bos[, -c(4,9)] # remove categorical variables
round(cor(bos_2),
  digits = 2 # rounded to 2 decimals
)
```

```
##           X.1     X  crim indus  chas   nox    rm   dis   rad   tax ptratio
## X.1      1.00  1.00  0.41  0.40  0.00  0.40 -0.08 -0.30  0.69  0.67    0.29
## X        1.00  1.00  0.41  0.40  0.00  0.40 -0.08 -0.30  0.69  0.67    0.29
## crim     0.41  0.41  1.00  0.41 -0.06  0.42 -0.22 -0.38  0.63  0.58    0.29
## indus    0.40  0.40  0.41  1.00  0.06  0.76 -0.39 -0.71  0.60  0.72    0.38
## chas     0.00  0.00 -0.06  0.06  1.00  0.09  0.09 -0.10 -0.01 -0.04   -0.12
## nox      0.40  0.40  0.42  0.76  0.09  1.00 -0.30 -0.77  0.61  0.67    0.19
## rm      -0.08 -0.08 -0.22 -0.39  0.09 -0.30  1.00  0.21 -0.21 -0.29   -0.36
## dis     -0.30 -0.30 -0.38 -0.71 -0.10 -0.77  0.21  1.00 -0.49 -0.53   -0.23
## rad      0.69  0.69  0.63  0.60 -0.01  0.61 -0.21 -0.49  1.00  0.91    0.46
## tax      0.67  0.67  0.58  0.72 -0.04  0.67 -0.29 -0.53  0.91  1.00    0.46
## ptratio  0.29  0.29  0.29  0.38 -0.12  0.19 -0.36 -0.23  0.46  0.46    1.00
## black   -0.30 -0.30 -0.39 -0.36  0.05 -0.38  0.13  0.29 -0.44 -0.44   -0.18
## lstat    0.26  0.26  0.46  0.60 -0.05  0.59 -0.61 -0.50  0.49  0.54    0.37
## medv    -0.23 -0.23 -0.39 -0.48  0.18 -0.43  0.70  0.25 -0.38 -0.47   -0.51
##         black lstat  medv
## X.1     -0.30  0.26 -0.23
## X       -0.30  0.26 -0.23
## crim    -0.39  0.46 -0.39
## indus   -0.36  0.60 -0.48
## chas     0.05 -0.05  0.18
## nox     -0.38  0.59 -0.43
## rm       0.13 -0.61  0.70
## dis      0.29 -0.50  0.25
## rad     -0.44  0.49 -0.38
## tax     -0.44  0.54 -0.47
## ptratio -0.18  0.37 -0.51
## black    1.00 -0.37  0.33
## lstat   -0.37  1.00 -0.74
## medv     0.33 -0.74  1.00
```

```
ggcorrplot(cor(bos_2), hc.order = TRUE) # looks to lstat and medv due to the heatmap.
```




```
plot(bos_2$medv, bos_2$lstat) # verifying the correlation between lstat and medv
```




```r
round(cor(bos_2$medv, bos_2$lstat), digits = 3)
```

```
## [1] -0.738
```

lstat is the variable correlated most with medv, with an absolute value score of 0.738. The next two variables in line are "rm" and "ptratio".

(b) Build a simple linear regression model with that variable as the x, with

i) Constructing the normal equations AtAx = Atb, and solving for the coefficient vector x.
ii) Using lm. Do the coefficients agree?

```r
# i)
# build a simple linear regression model with lstat as x
A <- cbind(1,bos$lstat) # design matrix A
b <- bos$medv # response variable b
# solve for the coefficient vector x
x <- solve(t(A) %*% A) %*% t(A) %*% b
x
```

```
##             [,1]
## [1,] 34.5538409
## [2,] -0.9500494
```

```r
# ii)
# using lm now
bos.fit <- lm(medv ~ lstat, data = bos)
coefficients(bos.fit) #check coefficients
```

```
## (Intercept)       lstat
##  34.5538409  -0.9500494
```

The coefficients match for both modeling methods, they are the same.

(c) Next, build a linear regression model with the 2 other variables that correlate most strongly with medv, using lm. How do the adjusted R-squared values compare between the models?

```r
bos.fit_2 <- lm(medv ~ lstat + rm + ptratio, data = bos) # These two variables I mentioned earlier when
summary(bos.fit_2)
```

```
##
## Call:
## lm(formula = medv ~ lstat + rm + ptratio, data = bos)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.4871  -3.1047  -0.7976   1.8129  29.6559
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 18.56711    3.91320   4.745 2.73e-06 ***
## lstat       -0.57181    0.04223 -13.540  < 2e-16 ***
## rm           4.51542    0.42587  10.603  < 2e-16 ***
## ptratio     -0.93072    0.11765  -7.911 1.64e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.229 on 502 degrees of freedom
## Multiple R-squared:  0.6786, Adjusted R-squared:  0.6767
## F-statistic: 353.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

The adjusted R^2 gets higher in this model compared to the previous model, but this is only natural because the Adjusted R^2 only gets higher when you add more variables.

(d) For the model in (c), what are the 95% confidence intervals for the parameters, according to the t-values?

```r
confint(bos.fit_2, level = 0.95)
```

```
##                  2.5 %     97.5 %
## (Intercept) 10.8788409 26.2553821
## lstat       -0.6547754 -0.4888359
## rm           3.6787108  5.3521311
## ptratio     -1.1618769 -0.6995682
```

(e) For the model in (c), what are the 95% confidence intervals for the parameters, using (i) regular bootstrapping, and (ii) Bayesian bootstrapping?

```
# Regular Bootstrap
coeff1 <- rep(0,100)
coeff2 <- rep(0,100)
weight <- rep(0,length(bos$rm))
for (i in 1:100){
  n <- length(bos$rm)
  row_sample <- sample(1:n, n, replace = T)
  bos_sample <- bos[row_sample,]
  mod <- lm(medv ~ rm + ptratio, data = bos_sample)
  coeff1[i] <- mod$coefficients[1]
  coeff2[i] <- mod$coefficients[2]
}
quantile(coeff1, c(0.025, 0.975))
```

```
##       2.5%      97.5%
## -10.789585   6.395489
```

```
quantile(coeff2, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 6.453907 8.696030
```

```
# Bayesian Bootstrap
for (i in 1:100){
  n <- length(bos$rm)
  weight <- rdirichlet(1, rep(1,n))
  row_sample <- sample(1:n, n, replace = T, prob = weight)
  bos_sample <- bos[row_sample,]
  mod <- lm(medv ~ rm + ptratio, data = bos_sample)
  coeff1[i] <- mod$coefficients[1]
  coeff2[i] <- mod$coefficients[2]
}
quantile(coeff1, c(0.025, 0.975))
```

```
##      2.5%     97.5%
## -14.67681  11.31337
```

```
quantile(coeff2, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 5.939264 9.182108
```

Question 3

Take a look at the nndb dataset available in the sample data. There are 45 columns, 38 of which are numerical. We will build a PCA regression model for the Energy_kcal variable. After doing part (c), Transform the PCA regression equation into the original coordinates. In terms of sensitivity analysis which original variable is the most significant?

(a) Find the covariance matrix and the principal component values of the numerical fields, excluding the Energy_kcal field. How many of the 37 principal components are within 0.1% of the largest component?

```r
nndb <- read.csv("nndb_flat.csv")

numericVars <- select_if(nndb, is.numeric)
# Calculate covariance matrix
cov_mat <- cov(numericVars)
summary(princomp(cov_mat))
```

```
## Importance of components:
##                              Comp.1       Comp.2       Comp.3       Comp.4
## Standard deviation     1.344129e+07 9.575238e+04 1.032901e+04 5.370104e+03
## Proportion of Variance 9.999484e-01 5.074516e-05 5.904904e-07 1.596103e-07
## Cumulative Proportion  9.999484e-01 9.999991e-01 9.999997e-01 9.999999e-01
##                              Comp.5       Comp.6       Comp.7       Comp.8
## Standard deviation     4.119618e+03 2.772686e+03 5.168092e+02 3.572970e+02
## Proportion of Variance 9.393107e-08 4.254973e-08 1.478277e-09 7.065685e-10
## Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
##                              Comp.9      Comp.10      Comp.11      Comp.12
## Standard deviation     1.264578e+02 8.440607e+01 1.582721e+01 1.254720e+01
## Proportion of Variance 8.850879e-11 3.943145e-11 1.386450e-12 8.713429e-13
## Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
##                             Comp.13      Comp.14      Comp.15      Comp.16
## Standard deviation     5.328728e+00 3.291006e+00 2.011352e+00 1.869315e+00
## Proportion of Variance 1.571602e-13 5.994498e-14 2.239089e-14 1.934017e-14
## Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
##                             Comp.17      Comp.18      Comp.19      Comp.20
## Standard deviation     1.745867e+00 1.324869e+00 1.058268e+00 3.105775e-01
## Proportion of Variance 1.687011e-14 9.714968e-15 6.198504e-15 5.338698e-16
## Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
##                             Comp.21      Comp.22      Comp.23      Comp.24
## Standard deviation     3.796553e-02 2.130484e-02 1.735692e-02 1.701707e-02
## Proportion of Variance 7.977637e-18 2.512186e-18 1.667404e-18 1.602748e-18
## Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
##                             Comp.25      Comp.26      Comp.27      Comp.28
## Standard deviation     3.618315e-05 5.947707e-06 1.223769e-06 1.103206e-06
## Proportion of Variance 7.246166e-24 1.957918e-25 8.288846e-27 6.736102e-27
## Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
##                             Comp.29      Comp.30      Comp.31      Comp.32
## Standard deviation     6.259239e-07 3.429471e-07 1.810335e-07 8.741571e-08
## Proportion of Variance 2.168395e-27 6.509533e-28 1.813900e-28 4.229358e-29
## Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
##                             Comp.33 Comp.34 Comp.35 Comp.36 Comp.37 Comp.38
## Standard deviation     4.877899e-08       0       0       0       0       0
## Proportion of Variance 1.316925e-29       0       0       0       0       0
## Cumulative Proportion  1.000000e+00       1       1       1       1       1
##                        Comp.39
## Standard deviation           0
## Proportion of Variance       0
## Cumulative Proportion        1
```

(b) Transform the data into the principal component basis. Confirm that the data in the new basis has the multicollinearity removed.

```r
# Calculate principal components
pca <- prcomp(numericVars[-2], scale. = TRUE)

# Calculate proportion of variance explained by each principal component
var_prop <- pca$sdev^2 / sum(pca$sdev^2)

# Count the number of components within 0.1% of the largest proportion
num_components <- sum(var_prop >= 0.999 * max(var_prop))
```

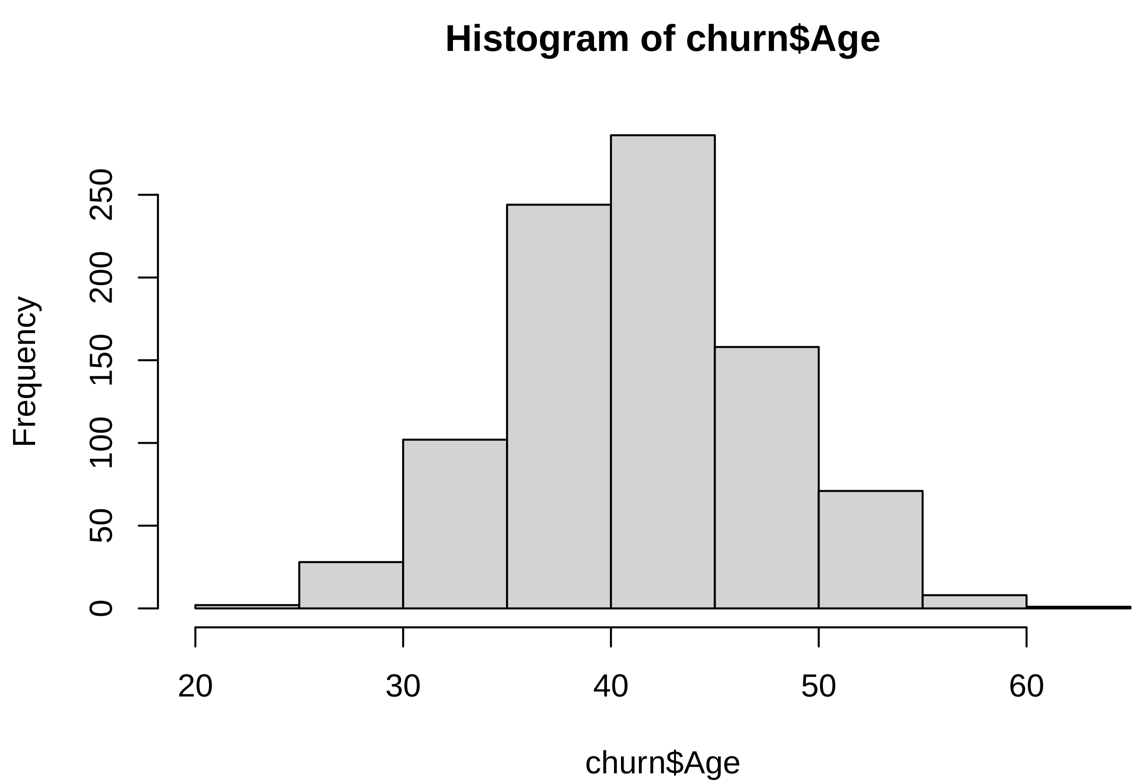
(c) Perform a linear regression using only the principal components which are within 0.1% in size of the largest component.

Question 4

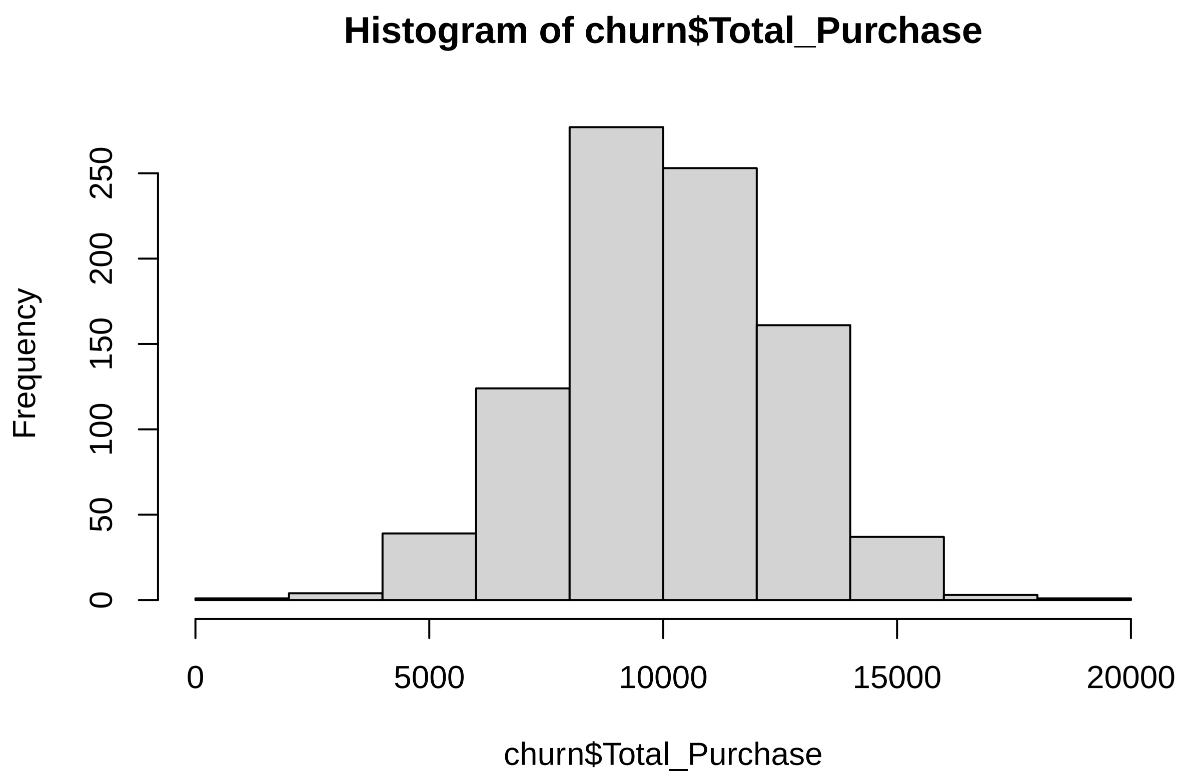
For the churn dataset, we will model the churn (whether a customer left) based on different models. Consider the fields Age, Total_Purchase, Account_Manager, Years, and Num_Sites as possible X variables. Note that Account_Manager is a binary categorical variable.

(a) Create histograms to examine how each variable might predict churn.

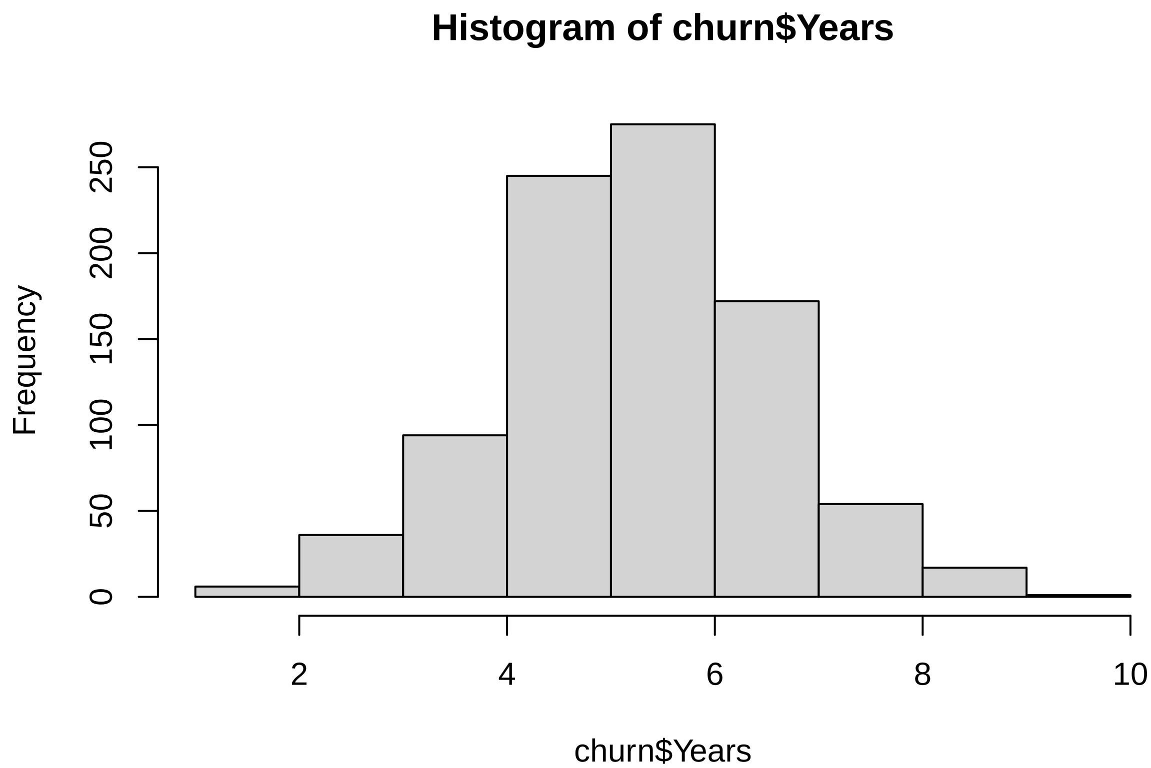
```r
churn <- read.csv("customer_churn.csv")
hist(churn$Age)
```



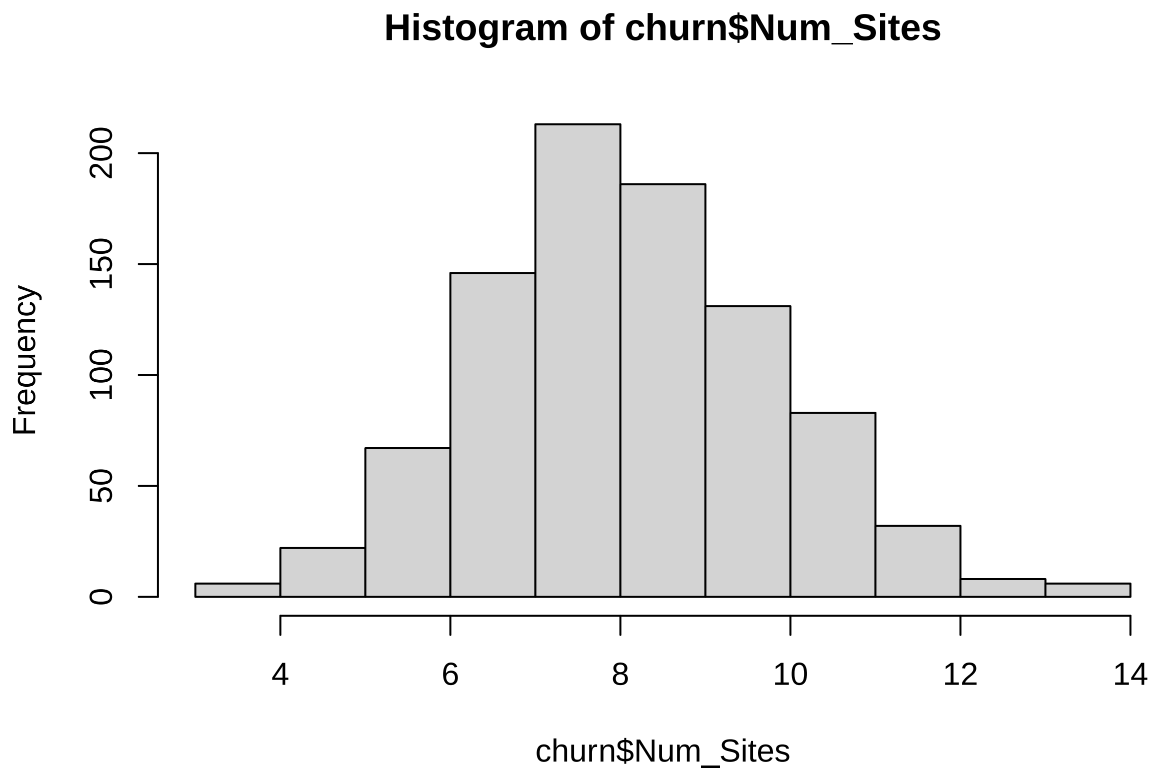
```
hist(churn$Total_Purchase)
```



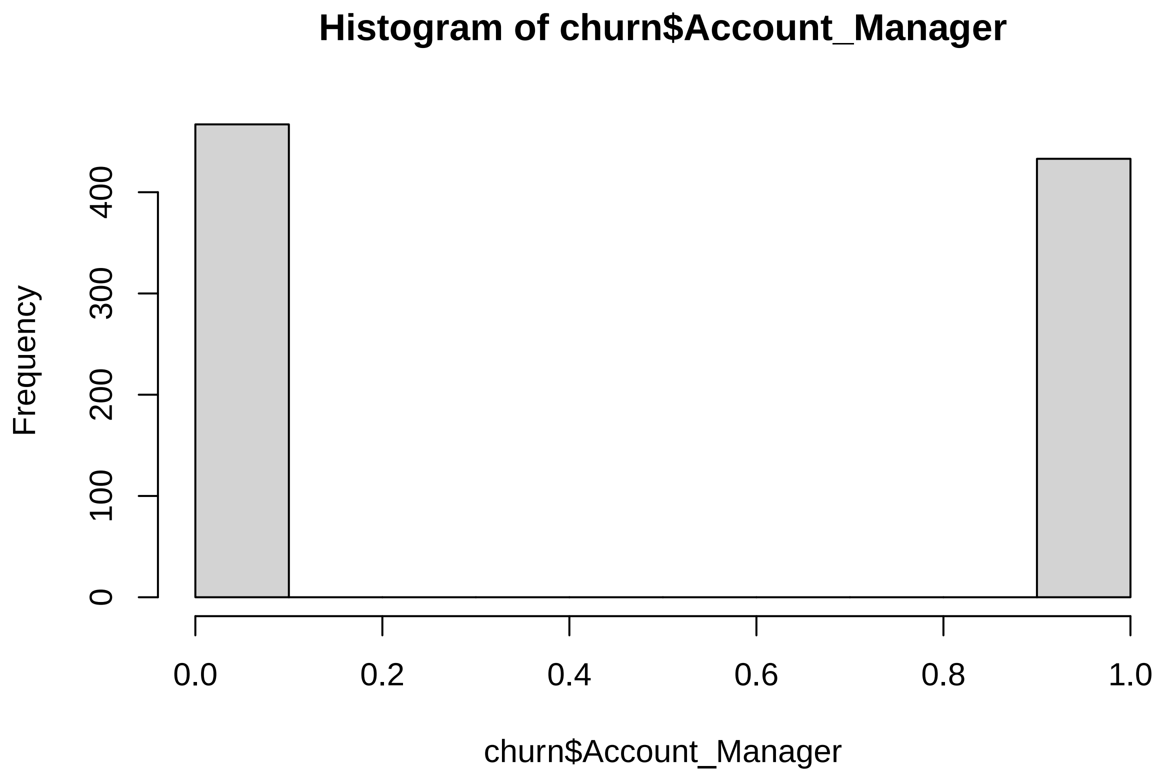

```
hist(churn$Years)
```




```
hist(churn$Num_Sites)
```




```
hist(churn$Account_Manager)
```



(b) Split the data into train and test datasets.

```
rows <- 1:nrow(churn)
train_split <- sample(rows, 0.7*length(rows))
test_split <- rows[-train_split]
churn_train <- churn[train_split,]
churn_test <- churn[test_split,]
```

(c) Fit a logistic regression model—first with all X's, and then remove those X's that are not statistically significant. Create a confusion matrix for this model.

```
churn.fit <- glm(Churn ~ Age + Years + Num_Sites , data = churn_train, family = binomial)
summary(churn.fit)
```

```
##
## Call:
## glm(formula = Churn ~ Age + Years + Num_Sites, family = binomial,
##     data = churn_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2594  -0.4000  -0.1789  -0.0715   3.4679
##
## Coefficients:
```

```
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -19.91422    1.92402 -10.350  < 2e-16 ***
## Age           0.05541    0.02416   2.293   0.0218 *
## Years         0.69669    0.12210   5.706 1.16e-08 ***
## Num_Sites     1.27666    0.12772   9.995  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 554.65  on 629  degrees of freedom
## Residual deviance: 314.74  on 626  degrees of freedom
## AIC: 322.74
##
## Number of Fisher Scoring iterations: 6
```

```r
churn_pred <- predict(churn.fit, newdata = churn_test, type = "response")
churn_pred_class <- ifelse(churn_pred > 0.5, 1, 0)

# confusion matrix
confusionMatrix(table(churn_pred_class, churn_test$Churn))
```

```
## Confusion Matrix and Statistics
##
##
## churn_pred_class   0   1
##                0 209  20
##                1  12  29
##
##                Accuracy : 0.8815
##                  95% CI : (0.8368, 0.9175)
##     No Information Rate : 0.8185
##     P-Value [Acc > NIR] : 0.003238
##
##                   Kappa : 0.574
##
##  Mcnemar's Test P-Value : 0.215925
##
##             Sensitivity : 0.9457
##             Specificity : 0.5918
##          Pos Pred Value : 0.9127
##          Neg Pred Value : 0.7073
##              Prevalence : 0.8185
##          Detection Rate : 0.7741
##    Detection Prevalence : 0.8481
##       Balanced Accuracy : 0.7688
##
##        'Positive' Class : 0
##
```

The variable *Total_Purchase* was the variable I removed because it was not significant.

(d) Create a decision tree model. Create a confusion matrix for this model.

```r
churn_tree <- tree(as.factor(Churn) ~ Age + Years + Num_Sites , data = churn_train)
summary(churn_tree)
```

```
##
## Classification tree:
## tree(formula = as.factor(Churn) ~ Age + Years + Num_Sites, data = churn_train)
## Number of terminal nodes:  9
## Residual mean deviance:  0.4866 = 302.2 / 621
## Misclassification error rate: 0.1 = 63 / 630
```

```r
plot(churn_tree)
text(churn_tree)
```


```r
churn_tree_pred <- predict(churn_tree, churn_test, type = 'class')

# confusion matrix

confusionMatrix(table(churn_tree_pred, churn_test$Churn))
```

```
## Confusion Matrix and Statistics
##
##
## churn_tree_pred   0   1
##               0 206  14
```

```
##               1  15  35
##
##                Accuracy : 0.8926
##                  95% CI : (0.8494, 0.9269)
##     No Information Rate : 0.8185
##     P-Value [Acc > NIR] : 0.0005626
##
##                   Kappa : 0.6413
##
##  Mcnemar's Test P-Value : 1.0000000
##
##             Sensitivity : 0.9321
##             Specificity : 0.7143
##          Pos Pred Value : 0.9364
##          Neg Pred Value : 0.7000
##              Prevalence : 0.8185
##          Detection Rate : 0.7630
##    Detection Prevalence : 0.8148
##       Balanced Accuracy : 0.8232
##
##        'Positive' Class : 0
##
```