Homework 5 – Due Monday October 12, 2015

Cheryl Calhoun

09/29/2015

Assignment #5

Partly inspired by recent adventures, but connected to everyone's experiences when travelling to Gainesville, we will analyze delays associated with airline travel. In particular, the outcome of interest is whether or not a particular flight was noticeably delayed in its arrival to its destination.

The Data Set

Our data set contains the following variables about flights:

- ArrDelay: arrival delay in minutes (negative means an early arrival)
- Month: month of the flight (1 = January; 12 = December)
- DayofWeek: day of the week (1 = Monday; 7 = Sunday)
- AirTime: total time in the air in minutes
- Distance: total distance of the flight in miles
- TaxiIn: time taxiing out to the runway in minutes
- TaxiOut: time taxiing to the gate after landing in minutes
- DepDelay: departure delay in minutes (negative means an early departure)
- Diverted: was the plane diverted? 1= Yes, 0 = No
- Carrier: was there a delay due to the carrier/company? 1 = Yes, 0 = No
- Weather: was there a delay due to weather? 1 = Yes, 0 = No
- NAS: was there a delay due to the National Air System? 1 = Yes, 0 = No
- Security: was there a delay due to security problems? 1 = Yes, 0 = No
- LateAircraft: was there a delay due to a late aircraft? 1 = Yes, 0 = No

We will begin by loading the data file from its source, `http://www.acthomas.ca/FSSS/data/airline-data.csv` , using the read.table command. We then review the data set to see how many observations (rows) our data set has and confirm that the column names correspond to the variables above.

```
## Load airline data set.
airline <- read.table ("http://www.acthomas.ca/FSSS/data/airline-data.csv", se
p = ",",  header=TRUE)

## Review contents of airline data set.
is.data.frame(airline)
```

```
## [1] TRUE
```

```
## Review column (variable) names and data types for each variable.
str(airline)
```

```
## 'data.frame':    4887 obs. of  15 variables:
##  $ X           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ ArrDelay    : int  -7 57 -15 -4 -5 19 -5 30 106 20 ...
##  $ Month       : int  2 11 12 3 12 5 12 1 12 3 ...
##  $ DayOfWeek   : int  4 4 2 3 7 3 5 5 1 1 ...
##  $ AirTime     : int  244 66 118 73 52 88 111 68 93 177 ...
##  $ Distance    : int  1814 413 972 444 304 629 758 308 675 1389 ...
##  $ TaxiIn      : int  8 4 8 3 6 5 6 6 12 4 ...
##  $ TaxiOut     : int  17 13 12 8 13 12 11 22 19 14 ...
##  $ DepDelay    : int  0 79 2 -3 -1 33 1 9 97 34 ...
##  $ Diverted    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Carrier     : int  0 1 0 0 0 1 0 1 1 1 ...
##  $ Weather     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NAS         : int  0 0 0 0 0 0 0 1 1 0 ...
##  $ Security    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ LateAircraft: int  0 1 0 0 0 0 0 0 0 1 ...
```

```
summary(airline)
```

```
##        X           ArrDelay          Month          DayOfWeek
##  Min.   :   1   Min.   : -60.000   Min.   : 1.000   Min.   :1.000
##  1st Qu.:1250   1st Qu.: -10.000   1st Qu.: 4.000   1st Qu.:2.000
##  Median :2496   Median :  -2.000   Median : 6.000   Median :4.000
##  Mean   :2498   Mean   :   8.456   Mean   : 6.408   Mean   :3.915
##  3rd Qu.:3746   3rd Qu.:  11.000   3rd Qu.: 9.000   3rd Qu.:6.000
##  Max.   :5000   Max.   :1092.000   Max.   :12.000   Max.   :7.000
##     AirTime         Distance          TaxiIn            TaxiOut
##  Min.   :  9.0   Min.   :  31.0   Min.   :  0.000   Min.   :  2.00
##  1st Qu.: 56.0   1st Qu.: 328.0   1st Qu.:  4.000   1st Qu.: 10.00
##  Median : 84.0   Median : 573.0   Median :  6.000   Median : 14.00
##  Mean   :102.8   Mean   : 719.1   Mean   :  6.755   Mean   : 16.38
##  3rd Qu.:131.0   3rd Qu.: 944.0   3rd Qu.:  8.000   3rd Qu.: 19.00
##  Max.   :580.0   Max.   :4502.0   Max.   :171.000   Max.   :192.00
##     DepDelay          Diverted    Carrier           Weather
##  Min.   : -29.00   Min.   :0   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:  -4.00   1st Qu.:0   1st Qu.:0.00000   1st Qu.:0.00000
##  Median :  -1.00   Median :0   Median :0.00000   Median :0.00000
##  Mean   :  10.21   Mean   :0   Mean   :0.09597   Mean   :0.01514
##  3rd Qu.:   8.00   3rd Qu.:0   3rd Qu.:0.00000   3rd Qu.:0.00000
##  Max.   :1099.00   Max.   :0   Max.   :1.00000   Max.   :1.00000
##       NAS            Security          LateAircraft
##  Min.   :0.0000   Min.   :0.0000000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000000   1st Qu.:0.0000
##  Median :0.0000   Median :0.0000000   Median :0.0000
##  Mean   :0.1306   Mean   :0.0004092   Mean   :0.1076
##  3rd Qu.:0.0000   3rd Qu.:0.0000000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.0000000   Max.   :1.0000
```

Results: The data file, airline-data.csv, is a comma delimeted file. We used the `read.table()` command with `sep=","` to read the .csv file into a data frame. Next we confirmed it was a data frame using `is.data.frame()` which returned a value of TRUE. Then we used `str()` to determine there are 4887 observations with 15 total variables, that the column names do in fact correspond to those provided in the assignment, and to determine the data types of each of the variables. We then used `summary()` to get an overview of the descriptive statistics for the variables contained in the data set.

Converting to factors

The first step in our analysis, will be to determine which two numerical columns must be converted to factors and use `mutate()` to do so. We will then confirm the counts of these factors using the `summary` function applied only to those two columns.

```
## Convert variables Month and DayofWeek to data type factor.
airline.2 <- airline %>%
  mutate (Month=as.factor(Month),
          DayOfWeek=as.factor(DayOfWeek))

## Review data set to verify conversion worked properly.
str(airline.2)
```

```
## 'data.frame':    4887 obs. of  15 variables:
##  $ X           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ ArrDelay    : int  -7 57 -15 -4 -5 19 -5 30 106 20 ...
##  $ Month       : Factor w/ 12 levels "1","2","3","4",..: 2 11 12 3 12 5 12
1 12 3 ...
##  $ DayOfWeek   : Factor w/ 7 levels "1","2","3","4",..: 4 4 2 3 7 3 5 5 1
1 ...
##  $ AirTime     : int  244 66 118 73 52 88 111 68 93 177 ...
##  $ Distance    : int  1814 413 972 444 304 629 758 308 675 1389 ...
##  $ TaxiIn      : int  8 4 8 3 6 5 6 6 12 4 ...
##  $ TaxiOut     : int  17 13 12 8 13 12 11 22 19 14 ...
##  $ DepDelay    : int  0 79 2 -3 -1 33 1 9 97 34 ...
##  $ Diverted    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Carrier     : int  0 1 0 0 0 1 0 1 1 1 ...
##  $ Weather     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NAS         : int  0 0 0 0 0 0 0 1 1 0 ...
##  $ Security    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ LateAircraft: int  0 1 0 0 0 0 0 0 0 1 ...
```

```
## Use summary to look at each of the converted variables to verify new data lo
oks as is expected.
summary(airline.2$Month)
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12
## 416 381 414 441 428 440 415 425 358 424 355 390
```

```
summary(airline.2$DayOfWeek)
```

```
##   1   2   3   4   5   6   7
## 715 730 736 726 716 580 684
```

Results: Used `as.factor()` to convert Month and DayOfWeek to factors, and `mutate()` to replace the original values for Month and DayOfWeek with the new factor variables. Saved new data.frame as `airline.2`. Used `str()` to verify Month and DayOfWeek are now factors. Used 'summary()` to view the levels and frequencies of each level.

Creating a variable for delays over 10 minutes

Our objective will be to see if a flight's arrival is delayed by more than 10 minutes. To do this, we will create a new variable in this data frame using `mutate()` corresponding to this True/False outcome.

```
## Calculate number of flights where arrival delay was more than 10 minutes, as
sign to new variable `Delayed10`.
airline.3 <- airline.2 %>%
  mutate(Delayed10 = (ArrDelay > 10))

## Review data set to verify calculation worked correctly.
str(airline.3)
```

```
## 'data.frame':    4887 obs. of  16 variables:
##  $ X           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ ArrDelay    : int  -7 57 -15 -4 -5 19 -5 30 106 20 ...
##  $ Month       : Factor w/ 12 levels "1","2","3","4",..: 2 11 12 3 12 5 12
1 12 3 ...
##  $ DayOfWeek   : Factor w/ 7 levels "1","2","3","4",..: 4 4 2 3 7 3 5 5 1
1 ...
##  $ AirTime     : int  244 66 118 73 52 88 111 68 93 177 ...
##  $ Distance    : int  1814 413 972 444 304 629 758 308 675 1389 ...
##  $ TaxiIn      : int  8 4 8 3 6 5 6 6 12 4 ...
##  $ TaxiOut     : int  17 13 12 8 13 12 11 22 19 14 ...
##  $ DepDelay    : int  0 79 2 -3 -1 33 1 9 97 34 ...
##  $ Diverted    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Carrier     : int  0 1 0 0 0 1 0 1 1 1 ...
##  $ Weather     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NAS         : int  0 0 0 0 0 0 0 1 1 0 ...
##  $ Security    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ LateAircraft: int  0 1 0 0 0 0 0 0 0 1 ...
##  $ Delayed10   : logi  FALSE TRUE FALSE FALSE FALSE TRUE ...
```

```
summary(airline.3$Delayed10)
```

```
##    Mode   FALSE    TRUE    NA's
## logical    3606    1281       0
```

Results: Created new variable `Delayed10` to indicate whether or not a flight's arrival is delayed by more than 10 minutes. Saved new data.frame as `airline.3`. Used `str()` to verify the new variable has been created, and view the resulting data type. Used `summary()` to view descriptive statisticss and verify the new variable was calculated correctly. `Delayed10` now contains logical data indicating true for flights with delays > 10 minutes, and false for flights with delays <= 10 minutes.

Fitting the binary linear model

Now we will use `glm()` to fit a binary linear model to the Delayed10 outcome. We will include all variables initially, and then look at whether or not any variables can be omitted if it is determined they have no bearing on the analysis.

```
## Use `glm` to fit a binary linear model to `Delayed10` with all variables in the data set
delayed.model <- glm (Delayed10 ~ Month + DayOfWeek + AirTime + Distance + TaxiIn +  TaxiOut + DepDelay + Carrier + Weather + NAS + Security + LateAircraft - 1, data=airline.3, family=binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## View resulting model and summary statistics.
summary(delayed.model)
```

```
##
## Call:
## glm(formula = Delayed10 ~ Month + DayOfWeek + AirTime + Distance +
##     TaxiIn + TaxiOut + DepDelay + Carrier + Weather + NAS + Security +
##     LateAircraft - 1, family = binomial, data = airline.3)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7468  -0.2281  -0.1311   0.0000   3.3459
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## Month1       -7.045e+00  4.867e-01 -14.475  < 2e-16 ***
## Month2       -6.944e+00  4.992e-01 -13.909  < 2e-16 ***
## Month3       -7.295e+00  5.194e-01 -14.045  < 2e-16 ***
## Month4       -6.440e+00  4.454e-01 -14.458  < 2e-16 ***
## Month5       -6.252e+00  4.298e-01 -14.545  < 2e-16 ***
## Month6       -6.683e+00  4.583e-01 -14.583  < 2e-16 ***
## Month7       -7.120e+00  4.957e-01 -14.363  < 2e-16 ***
## Month8       -7.029e+00  4.923e-01 -14.276  < 2e-16 ***
## Month9       -6.889e+00  4.998e-01 -13.782  < 2e-16 ***
## Month10      -7.053e+00  4.855e-01 -14.528  < 2e-16 ***
## Month11      -6.997e+00  5.104e-01 -13.708  < 2e-16 ***
## Month12      -6.974e+00  4.822e-01 -14.464  < 2e-16 ***
## DayOfWeek2    4.340e-01  3.135e-01   1.384   0.1663
## DayOfWeek3    3.467e-01  3.157e-01   1.098   0.2721
## DayOfWeek4    4.061e-01  3.112e-01   1.305   0.1919
## DayOfWeek5   -5.248e-02  3.395e-01  -0.155   0.8771
## DayOfWeek6    5.465e-01  3.317e-01   1.648   0.0994 .
## DayOfWeek7    3.349e-01  3.218e-01   1.041   0.2980
## AirTime       5.354e-02  7.866e-03   6.806 1.00e-11 ***
## Distance     -6.744e-03  9.918e-04  -6.799 1.05e-11 ***
## TaxiIn        1.072e-01  1.710e-02   6.270 3.62e-10 ***
## TaxiOut       1.087e-01  9.977e-03  10.895  < 2e-16 ***
## DepDelay      1.794e-01  1.061e-02  16.903  < 2e-16 ***
## Carrier       1.949e+01  7.112e+02   0.027   0.9781
## Weather       1.659e+01  1.734e+03   0.010   0.9924
## NAS           2.159e+01  6.912e+02   0.031   0.9751
## Security     -1.125e+01  1.725e+04  -0.001   0.9995
## LateAircraft  1.774e+01  6.866e+02   0.026   0.9794
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6774.8  on 4887  degrees of freedom
## Residual deviance: 1096.9  on 4859  degrees of freedom
## AIC: 1152.9
```

```
##
## Number of Fisher Scoring iterations: 20
```

Results: The resulting model shows 6 variables which are statistically significant. One of these is `Month` which appears as all 12 months. `DayOfWeek` is not producing statistically significant results. `Diverted` has been eliminated because it does not seem to contain anysignificant data as all observations = 0.

Note: `glm` creates a warning "## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred" when run for "family=binomial". This error is generated due to a condition where one or more of the fitted probabiites are extremely close to zero or one. This is not a fatal flaw but should be noted.

Standardizing the variables

Now lets look at the standardized model to see if we can improve the fit. We will create a new data frame that standardizes the variables in question by subtract their mean and divide by their standard deviation. We will use `qplot` to look at the resulting standardized variables to confirm that they appear to have the correct distributions.

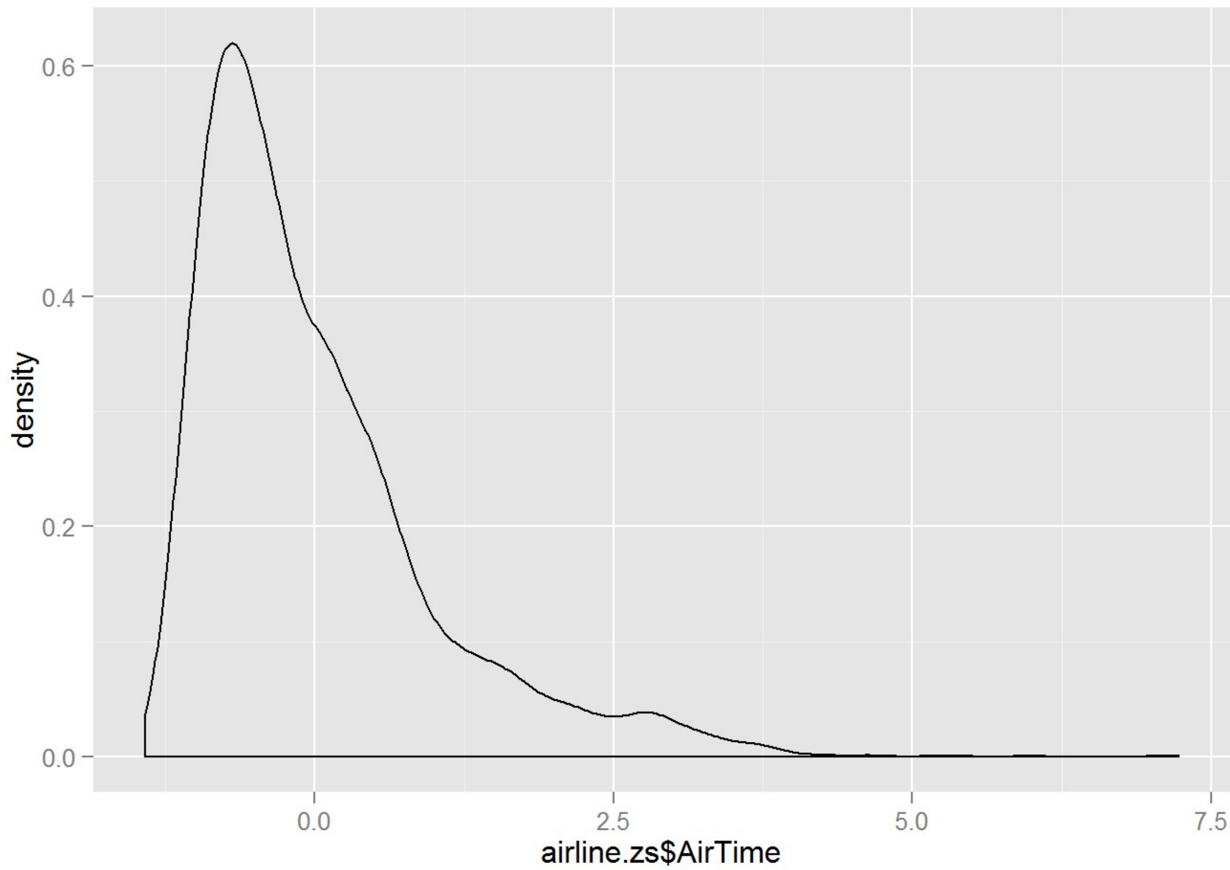
```
## Convert AirTime, Distance, TaxiIn, TaxiOut and DepDelay to standardized valu
es.
airline.zs <- airline.3 %>%
  mutate(AirTime = (AirTime - mean(AirTime)) / sd(AirTime)) %>%
  mutate(Distance = (Distance - mean(Distance)) / sd(Distance)) %>%
  mutate(TaxiIn = (TaxiIn - mean(TaxiIn)) / sd(TaxiIn))  %>%
  mutate(TaxiOut = (TaxiOut - mean(TaxiOut)) / sd(TaxiOut))  %>%
  mutate(DepDelay = (DepDelay - mean(DepDelay)) / sd(DepDelay))

# View resulting data set and summary statistics to verify standardized values
appear correct.
str(airline.zs)
```

```
## 'data.frame':    4887 obs. of  16 variables:
##  $ X           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ ArrDelay    : int  -7 57 -15 -4 -5 19 -5 30 106 20 ...
##  $ Month       : Factor w/ 12 levels "1","2","3","4",..: 2 11 12 3 12 5 12
1 12 3 ...
##  $ DayOfWeek   : Factor w/ 7 levels "1","2","3","4",..: 4 4 2 3 7 3 5 5 1
1 ...
##  $ AirTime     : num  2.139 -0.557 0.23 -0.451 -0.769 ...
##  $ Distance    : num  1.991 -0.557 0.46 -0.5 -0.755 ...
##  $ TaxiIn      : num  0.232 -0.514 0.232 -0.7 -0.141 ...
##  $ TaxiOut     : num  0.0557 -0.3009 -0.39 -0.7465 -0.3009 ...
##  $ DepDelay    : num  -0.277 1.868 -0.223 -0.359 -0.305 ...
##  $ Diverted    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Carrier     : int  0 1 0 0 0 1 0 1 1 1 ...
##  $ Weather     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NAS         : int  0 0 0 0 0 0 0 1 1 0 ...
##  $ Security    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ LateAircraft: int  0 1 0 0 0 0 0 0 0 1 ...
##  $ Delayed10   : logi  FALSE TRUE FALSE FALSE FALSE TRUE ...
```

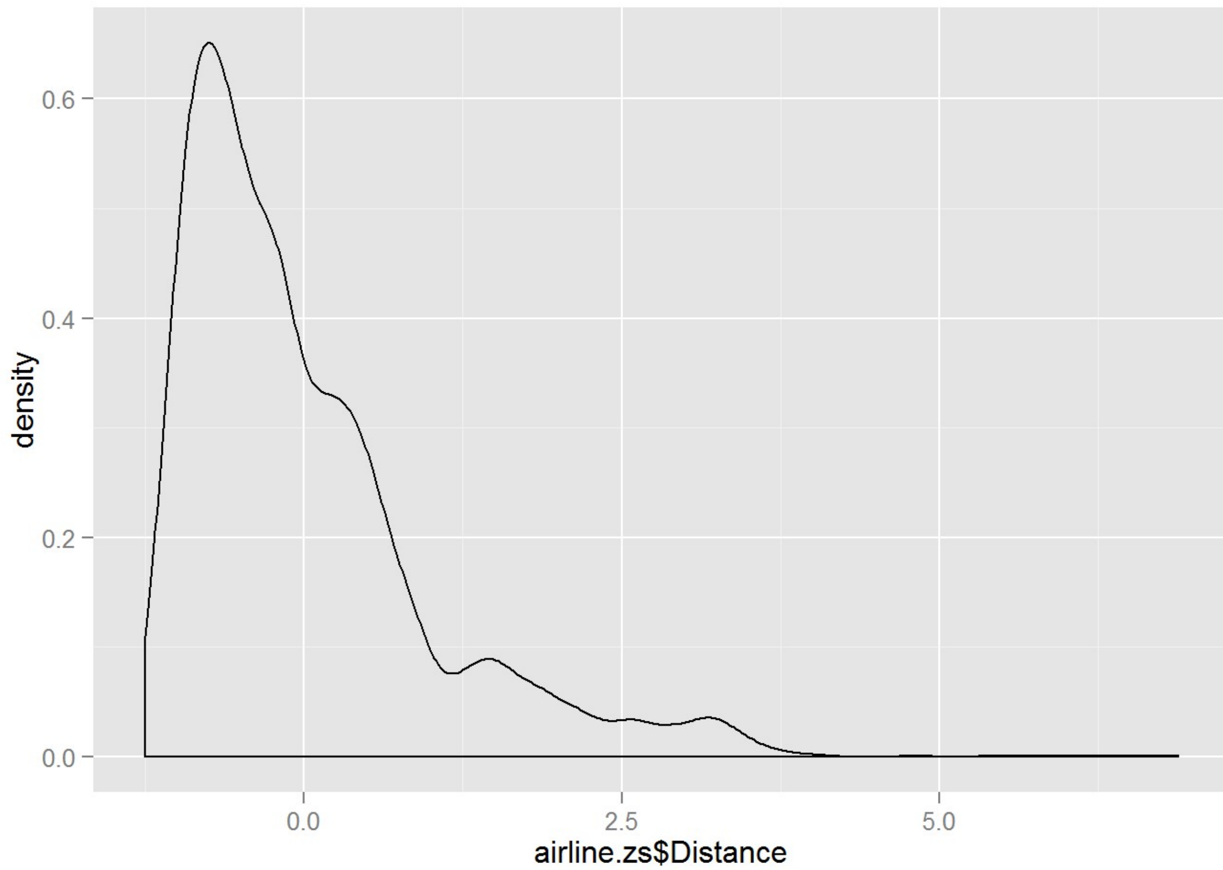
```
summary(airline.zs)
```

```
##        X           ArrDelay           Month      DayOfWeek
##  Min.   :   1   Min.   : -60.000   4      : 441   1:715
##  1st Qu.:1250   1st Qu.: -10.000   6      : 440   2:730
##  Median :2496   Median :  -2.000   5      : 428   3:736
##  Mean   :2498   Mean   :   8.456   8      : 425   4:726
##  3rd Qu.:3746   3rd Qu.:  11.000   10     : 424   5:716
##  Max.   :5000   Max.   :1092.000   1      : 416   6:580
##                                    (Other):2313   7:684
##     AirTime          Distance           TaxiIn           TaxiOut
##  Min.   :-1.4206   Min.   :-1.2513   Min.   :-1.2601   Min.   :-1.2814
##  1st Qu.:-0.7087   1st Qu.:-0.7112   1st Qu.:-0.5139   1st Qu.:-0.5683
##  Median :-0.2845   Median :-0.2657   Median :-0.1408   Median :-0.2117
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.4274   3rd Qu.: 0.4090   3rd Qu.: 0.2323   3rd Qu.: 0.2340
##  Max.   : 7.2283   Max.   : 6.8793   Max.   :30.6399   Max.   :15.6545
##
##     DepDelay           Diverted    Carrier           Weather
##  Min.   :-1.06495   Min.   :0   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:-0.38598   1st Qu.:0   1st Qu.:0.00000   1st Qu.:0.00000
##  Median :-0.30451   Median :0   Median :0.00000   Median :0.00000
##  Mean   : 0.00000   Mean   :0   Mean   :0.09597   Mean   :0.01514
##  3rd Qu.:-0.06008   3rd Qu.:0   3rd Qu.:0.00000   3rd Qu.:0.00000
##  Max.   :29.56996   Max.   :0   Max.   :1.00000   Max.   :1.00000
##
##       NAS            Security          LateAircraft    Delayed10
##  Min.   :0.0000   Min.   :0.0000000   Min.   :0.0000   Mode :logical
##  1st Qu.:0.0000   1st Qu.:0.0000000   1st Qu.:0.0000   FALSE:3606
##  Median :0.0000   Median :0.0000000   Median :0.0000   TRUE :1281
##  Mean   :0.1306   Mean   :0.0004092   Mean   :0.1076   NA's :0
##  3rd Qu.:0.0000   3rd Qu.:0.0000000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.0000000   Max.   :1.0000
##
```

```
## Plot two of the standardized variable to verify they are in the range of sta
ndardized scores.
qplot(airline.zs$AirTime, geom="density")
```




```
qplot(airline.zs$Distance, geom="density")
```



Results: We now have a new data frame, airline.zs, which contains the standardized z scores for each non 0/1 predictor. The resulting z scores produce plots with distributions clustered around "0", verifying the z scores were correctly calculated.

Fitting the binary linear model with standardized data

Again, we will use `glm()` to fit a binary linear model to the Delayed.10 outcome using the new data set which contains standardized variables for predictors.. We will include all variables initially, and then look at whether or not any variables can be omitted if it is determined they have no bearing on the analysis. but with these standardized variables for predictors instead. Which variables have the greatest effect size in each regression?

```
## Use `glm` to fit a binary linear model to Delayed10 using the standardized v
ariables in the data set
delay.model.zs <- glm (Delayed10 ~ Month + DayOfWeek + AirTime + Distance + Tax
iIn + TaxiOut + DepDelay + Carrier + Weather + NAS + Security + LateAircraft
-1, data=airline.zs, family=binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## View resulting model and summary statistics.
##delay.model.zs
summary(delay.model.zs)
```

```
##
## Call:
## glm(formula = Delayed10 ~ Month + DayOfWeek + AirTime + Distance +
##     TaxiIn + TaxiOut + DepDelay + Carrier + Weather + NAS + Security +
##     LateAircraft - 1, family = binomial, data = airline.zs)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7468  -0.2281  -0.1311   0.0000   3.3459
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## Month1       -2.055e+00  3.673e-01  -5.596 2.20e-08 ***
## Month2       -1.955e+00  3.785e-01  -5.164 2.42e-07 ***
## Month3       -2.306e+00  4.034e-01  -5.716 1.09e-08 ***
## Month4       -1.450e+00  3.388e-01  -4.279 1.87e-05 ***
## Month5       -1.263e+00  3.157e-01  -4.000 6.34e-05 ***
## Month6       -1.694e+00  3.364e-01  -5.035 4.79e-07 ***
## Month7       -2.130e+00  3.704e-01  -5.751 8.87e-09 ***
## Month8       -2.039e+00  3.864e-01  -5.277 1.31e-07 ***
## Month9       -1.899e+00  3.845e-01  -4.939 7.84e-07 ***
## Month10      -2.063e+00  3.768e-01  -5.476 4.34e-08 ***
## Month11      -2.007e+00  4.092e-01  -4.905 9.32e-07 ***
## Month12      -1.985e+00  3.674e-01  -5.402 6.60e-08 ***
## DayOfWeek2    4.340e-01  3.135e-01   1.384   0.1663
## DayOfWeek3    3.467e-01  3.157e-01   1.098   0.2721
## DayOfWeek4    4.061e-01  3.112e-01   1.305   0.1919
## DayOfWeek5   -5.248e-02  3.394e-01  -0.155   0.8771
## DayOfWeek6    5.465e-01  3.317e-01   1.648   0.0994 .
## DayOfWeek7    3.349e-01  3.218e-01   1.041   0.2980
## AirTime       3.535e+00  5.193e-01   6.806 1.00e-11 ***
## Distance     -3.708e+00  5.454e-01  -6.799 1.05e-11 ***
## TaxiIn        5.747e-01  9.166e-02   6.270 3.62e-10 ***
## TaxiOut       1.219e+00  1.119e-01  10.895  < 2e-16 ***
## DepDelay      6.605e+00  3.908e-01  16.903  < 2e-16 ***
## Carrier       1.949e+01  7.112e+02   0.027   0.9781
## Weather       1.659e+01  1.734e+03   0.010   0.9924
## NAS           2.159e+01  6.912e+02   0.031   0.9751
## Security     -1.125e+01  1.725e+04  -0.001   0.9995
## LateAircraft  1.774e+01  6.866e+02   0.026   0.9794
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6774.8  on 4887  degrees of freedom
## Residual deviance: 1096.9  on 4859  degrees of freedom
## AIC: 1152.9
```

```
##
## Number of Fisher Scoring iterations: 20
```

Results: The resulting model shows 6 variables which are statistically significant. One of these is `Month` which appears as all 12 months. `DayOfWeek` is not producing statistically significant results. `Diverted` has been eliminated because it does not seem to contain anysignificant data as all observations = 0. DepDelay has the greatest effect size in both models.

Note: `glm` creates a warning “## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred” when run for “family=binomial”. This error is generated due to a condition where one or more of the fitted probabiites are extremely close to zero or one. This is not a fatal flaw but should be noted.

Creating the design/predictors matrixs

Now we will produce the design/predictors matrix using the `model.matrix()` function. We will verify the number of columns corresponds to the number of coefficients in your previous `glm()` output.

```
## Produce the design/predictors matrix for arline.delays.zs.
delay.matrix <- model.matrix(~ 0 + Month + DayOfWeek + AirTime + Distance + Tax
iIn + TaxiOut + DepDelay + Carrier + Weather + NAS + Security + LateAircraft, a
irline.zs)

## Verify the number of columns in airline.delay.matrix corresponds to the numb
er of coefficients in the previous `glm()` output.
length(delay.model.zs$coefficients)
```

```
## [1] 28
```

```
dim(delay.matrix)[2]
```

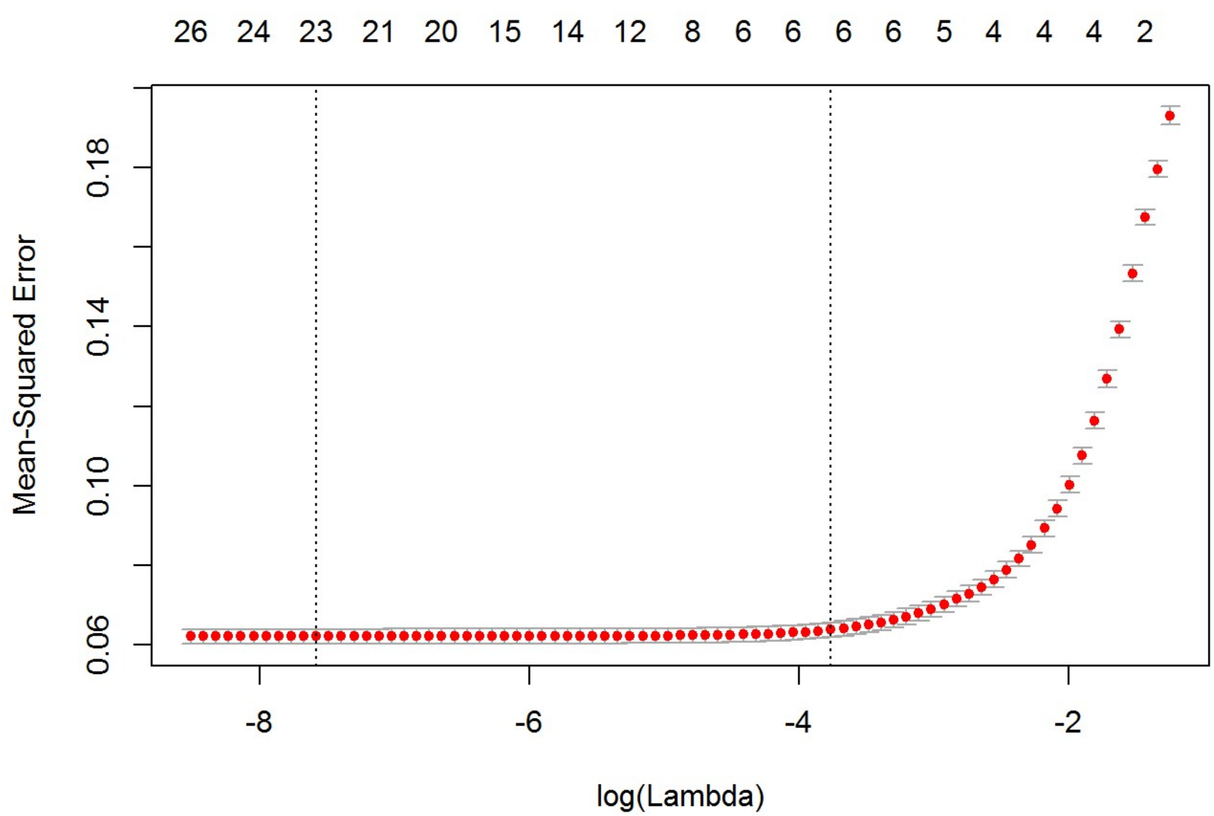
```
## [1] 28
```

Results: The delay.matrix has the same number of columns (28) as the number of coefficients in the previous ‘delay.model.zs’ (28).

Lasso effect

We will use `cv.glmnet()` with the Lasso (`alpha=1`) to run a penalized linear model for quality as the outcome with all predictors as previously done, for this data frame.

```
##Use cv.glmnet() with Lasso (`alpha=1`) to run the penalized lineral model for
delayed.10.
## First Validation Step
delay.model.cv = cv.glmnet (delay.matrix, airline.zs$Delayed10, alpha=1 )
picked <- which (delay.model.cv$lambda == delay.model.cv$lambda.min)
plot(delay.model.cv)
```



Results: This plot shows the cross-validation mean squared error (MSE) as a function of log(lambda) curve (red dotted line), including the upper and lower standard deviation curves. The dotted lines represent lambda.min and lambda.min plus one standard error. As lambda gets smaller, the curve flattens out. The numbers across the top of the plot indicate how many non-zero predictors are in the model at each level of lambda.

```
## The `lambda` that produces the smallest cross-validated error is:
delay.model.cv$lambda.min
```

```
## [1] 0.0005114175
```

```
## The cross validated error is:
delay.model.cv$cvm[picked]
```

```
## [1] 0.06210824
```

```
## How much of a reduction is this in cross-validated error from the basic fit model in Question 3?
summary(delay.model.zs)
```

```
##
## Call:
## glm(formula = Delayed10 ~ Month + DayOfWeek + AirTime + Distance +
##     TaxiIn + TaxiOut + DepDelay + Carrier + Weather + NAS + Security +
##     LateAircraft - 1, family = binomial, data = airline.zs)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7468  -0.2281  -0.1311   0.0000   3.3459
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## Month1       -2.055e+00  3.673e-01  -5.596 2.20e-08 ***
## Month2       -1.955e+00  3.785e-01  -5.164 2.42e-07 ***
## Month3       -2.306e+00  4.034e-01  -5.716 1.09e-08 ***
## Month4       -1.450e+00  3.388e-01  -4.279 1.87e-05 ***
## Month5       -1.263e+00  3.157e-01  -4.000 6.34e-05 ***
## Month6       -1.694e+00  3.364e-01  -5.035 4.79e-07 ***
## Month7       -2.130e+00  3.704e-01  -5.751 8.87e-09 ***
## Month8       -2.039e+00  3.864e-01  -5.277 1.31e-07 ***
## Month9       -1.899e+00  3.845e-01  -4.939 7.84e-07 ***
## Month10      -2.063e+00  3.768e-01  -5.476 4.34e-08 ***
## Month11      -2.007e+00  4.092e-01  -4.905 9.32e-07 ***
## Month12      -1.985e+00  3.674e-01  -5.402 6.60e-08 ***
## DayOfWeek2    4.340e-01  3.135e-01   1.384   0.1663
## DayOfWeek3    3.467e-01  3.157e-01   1.098   0.2721
## DayOfWeek4    4.061e-01  3.112e-01   1.305   0.1919
## DayOfWeek5   -5.248e-02  3.394e-01  -0.155   0.8771
## DayOfWeek6    5.465e-01  3.317e-01   1.648   0.0994 .
## DayOfWeek7    3.349e-01  3.218e-01   1.041   0.2980
## AirTime       3.535e+00  5.193e-01   6.806 1.00e-11 ***
## Distance     -3.708e+00  5.454e-01  -6.799 1.05e-11 ***
## TaxiIn        5.747e-01  9.166e-02   6.270 3.62e-10 ***
## TaxiOut       1.219e+00  1.119e-01  10.895  < 2e-16 ***
## DepDelay      6.605e+00  3.908e-01  16.903  < 2e-16 ***
## Carrier       1.949e+01  7.112e+02   0.027   0.9781
## Weather       1.659e+01  1.734e+03   0.010   0.9924
## NAS           2.159e+01  6.912e+02   0.031   0.9751
## Security     -1.125e+01  1.725e+04  -0.001   0.9995
## LateAircraft  1.774e+01  6.866e+02   0.026   0.9794
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6774.8  on 4887  degrees of freedom
## Residual deviance: 1096.9  on 4859  degrees of freedom
## AIC: 1152.9
```

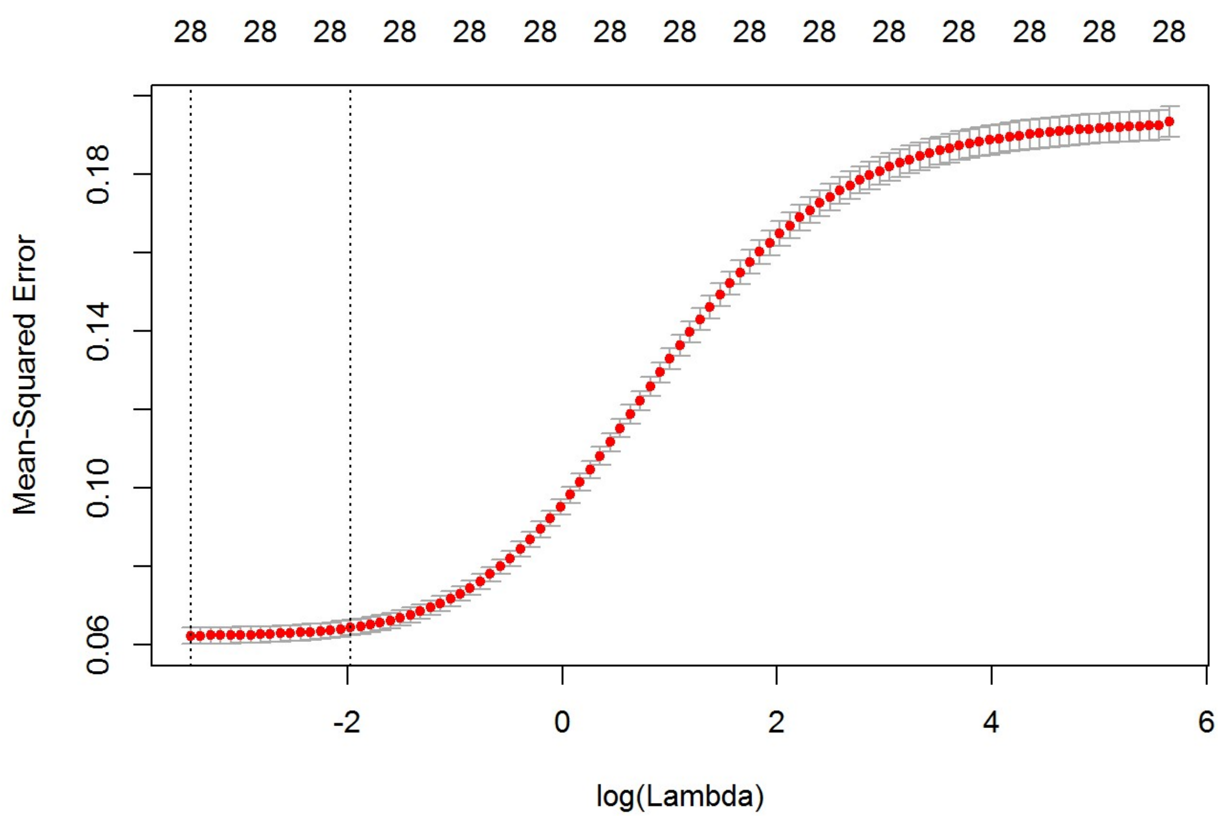
```
##
## Number of Fisher Scoring iterations: 20
```

Results: The `lambda` that produces the smallest value is [69] = 5.114175310^{-4}. The cross-validated error for red.quality.cv [69] is 0.0621082.

Ridge penalty

Let's run `cv.glmnet()` again, but now using the Ridge penalty (`alpha=0`).

```
##Use cv.glmnet() with Lasso (`alpha=1`) to run the penalized lineral model for
delayed.10.
## First Validation Step
delay.model.cv2 = cv.glmnet (delay.matrix, airline.zs$Delayed10, alpha=0 )
picked2 <- which (delay.model.cv2$lambda == delay.model.cv2$lambda.min)
plot(delay.model.cv2)
```



Results: This plot shows the cross-validation mean squared error (MSE) as a function of log(lambda) curve (red dotted line), including the upper and lower standard deviation curves. The dotted lines represent lambda.min and lambda.min plus one standard error. As lambda gets smaller, the curve flattens out. The numbers across the top of the plot indicate how many non-zero predictors are in the model at each level of lambda.

```
## The `lambda` that produces the smallest cross-validated error is:
delay.model.cv2$lambda.min
```

```
## [1] 0.03138011
```

```
## The cross validated error is:
delay.model.cv2$cvm[picked2]
```

```
## [1] 0.06236552
```

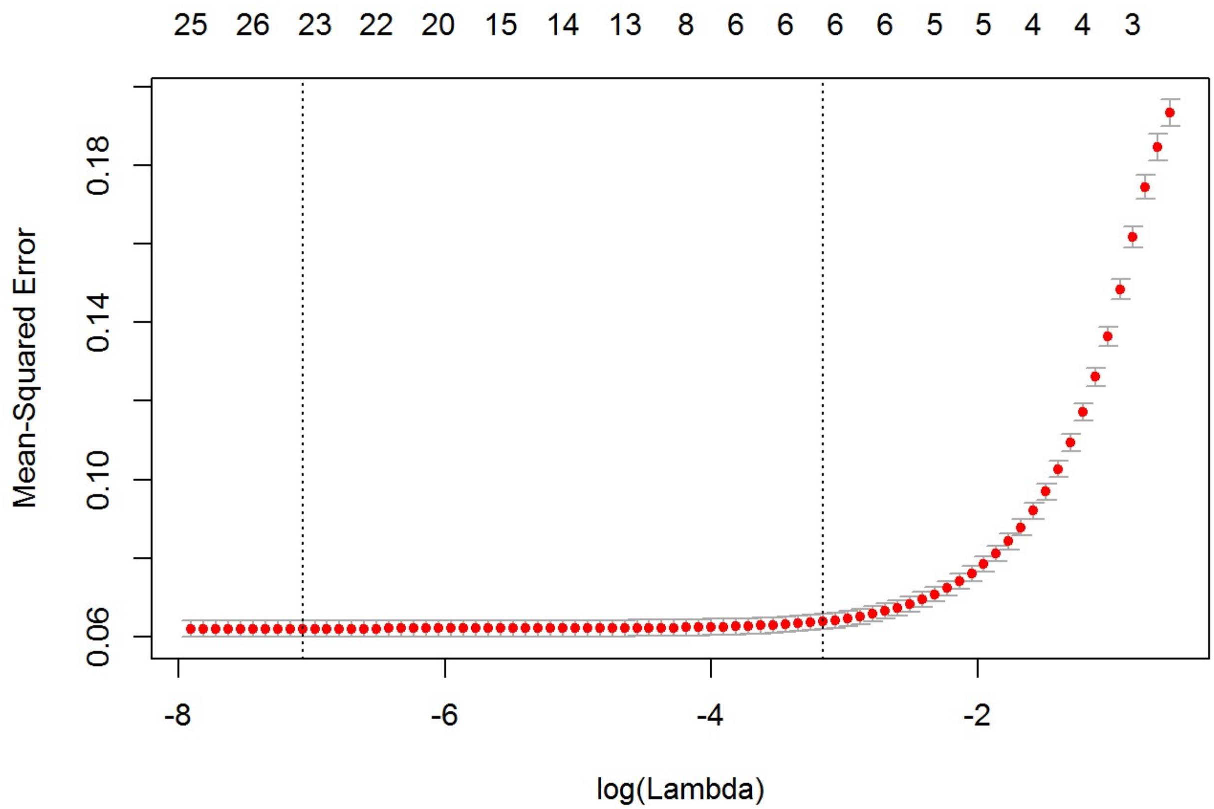
```
## How much of a reduction is this in cross-validated error from the basic fit
model in Question 3?
## summary(delay.model.zs)
## delay.model.zs
```

Results: The `lambda` that produces the smallest value is [99] = 0.0313801. The cross-validated error for red.quality.cv [99] is NA.

Balancing both penalties

Let's run `cv.glmnet` one more time, but now using an even combination of the two penalties (`alpha=0.5`).

```
##Use cv.glmnet() with Lasso (`alpha=1`) to run the penalized lineral model for
delayed.10.
## First Validation Step
delay.model.cv3 = cv.glmnet (delay.matrix, airline.zs$Delayed10, alpha=0.5 )
picked3 <- which (delay.model.cv3$lambda == delay.model.cv3$lambda.min)
plot(delay.model.cv3)
```



Results: This plot shows the cross-validation mean squared error (MSE) as a function of log(lambda) curve (red dotted line), including the upper and lower standard deviation curves. The dotted lines represent lambda.min and lambda.min plus one standard error. As lambda gets smaller, the curve flattens out. The numbers across the top of the plot indicate how many non-zero predictors are in the model at each level of lambda.

```
## The `lambda` that produces the smallest cross-validated error is:
delay.model.cv3$lambda.min
```

```
## [1] 0.0008491756
```

```
## The cross validated error is:
delay.model.cv3$cvm[picked3]
```

```
## [1] 0.06199374
```

```
## How much of a reduction is this in cross-validated error from the basic fit
model in Question 3?
## summary(delay.model.zs)
## delay.model.zs
```

Results: The `lambda` that produces the smallest value is [71] = 5.114175310^{-4}. The cross-validated error for red.quality.cv [71] is 0.0621096. The output from the `glm` generated model does not show adjusted r-squared for comparison.

Finding the smallest lambda

Now lets look at each of our models to determine which one contain the the smallest value of `cvm` for the `lambda` we selected.

```
delay.model.cv$cvm[picked]
```

```
## [1] 0.06210824
```

```
delay.model.cv2$cvm[picked2]
```

```
## [1] 0.06236552
```

```
delay.model.cv3$cvm[picked3]
```

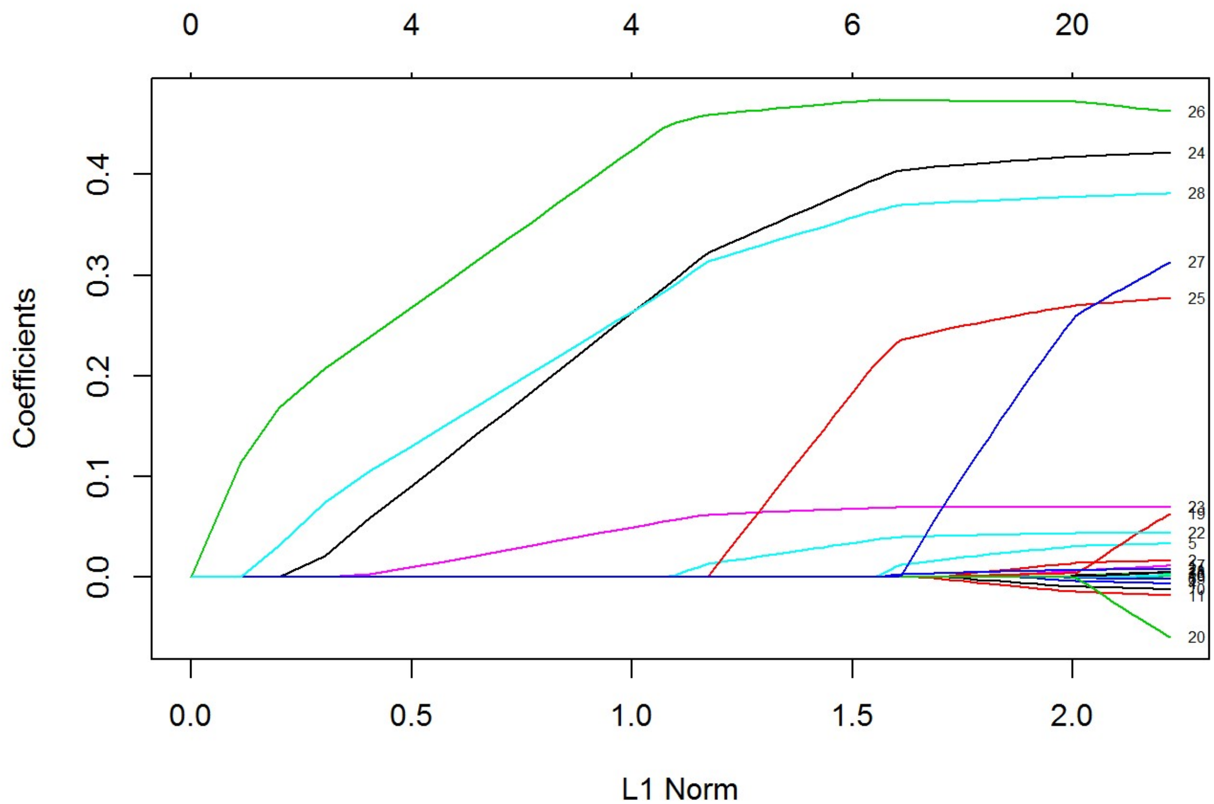
```
## [1] 0.06199374
```

Results: The smallest `cvm` value comes from delay.model.cv3[71] at 0.0619937.

Fitting the "shrinkage model"

We will use `glmnet()` to fit the "shrinkage" model to this data set using `delay.model.cv3$lambda` which gave us the smallest `cvm` . We will use the same `lambda` series as outputted in the previous steps for each mode to look at the `beta` matrix that corresponds to this ideal `lambda` .

```
airline.delay.model = glmnet (delay.matrix, airline.zs$Delayed10, lambda = dela
y.model.cv3$lambda)
plot(airline.delay.model, label = TRUE)
```



Results: Each curve in the plot represents a coefficient in the model. As the norm increases the coefficients deviate from 0. The numeric scale across the top indicates how many coefficients have deviated from 0.

```
airline.delay.model$beta[,picked3]
```

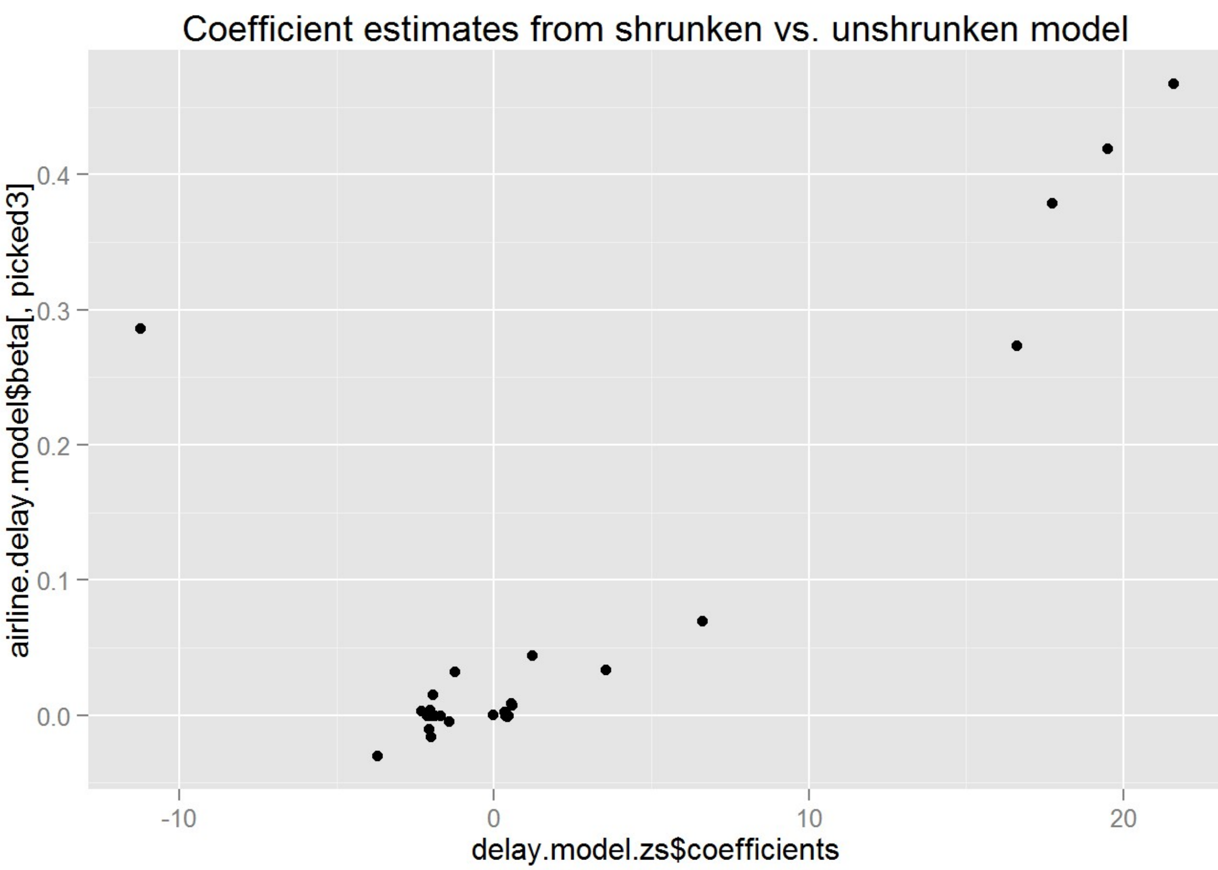
```
##        Month1        Month2        Month3        Month4        Month5 
##  0.0037187127  0.0156473331  0.0030852216 -0.0047970955  0.0324174794 
##        Month6        Month7        Month8        Month9       Month10 
##  0.0000000000  0.0000000000  0.0000000000  0.0000000000 -0.0103927007 
##       Month11       Month12    DayOfWeek2    DayOfWeek3    DayOfWeek4 
## -0.0159285624  0.0000000000  0.0000000000  0.0000000000 -0.0012428638 
##    DayOfWeek5    DayOfWeek6    DayOfWeek7       AirTime      Distance 
##  0.0005185574  0.0090237269  0.0027634698  0.0338186086 -0.0299654600 
##        TaxiIn       TaxiOut      DepDelay       Carrier       Weather 
##  0.0076587143  0.0441382760  0.0699790545  0.4195586937  0.2736560054 
##           NAS      Security  LateAircraft 
##  0.4673484890  0.2864899728  0.3793499846
```

Results: In this step we calculated the model which corresponds to the minimum `lambda` identified in step 7. The resulting plot and betas are shown. The estimates of `airline.delay.model$beta` for `Month6` , `Month7` , `Month8` , `Month9` , `Month12` , `DaysOfWeeks2` , and `DaysOfWeeks3` have shrunken to zero.

Comparing the unshrunken model to the shrunken model

Finally we will plot the coefficient estimates from the unshrunken models (`delay.model.zs`) compared to the ideal shrunken model (`airline.delay.model`),to demonstrate whether this shrunken estimation produced a noticeably different response.

```
qplot (delay.model.zs$coefficients, airline.delay.model$beta[,picked3], main="C
oefficient estimates from shrunken vs. unshrunken model")
```



Results: In this step we ploted the original coefficient estimates from `delay.model.zs` to the coefficient estimates from the shrunken model `airline.delay.model$beta[,picked3]` . There appear to be some differences in coefficients. If there were no differences, the plot would create a straight line.