Problem Set 6

Classification Part 1

[YOUR NAME]

Due Date: 2023-11-03

Getting Set Up

Open `RStudio` and create a new RMarkDown file (`.Rmd`) by going to `File -> New File -> R Markdown...` . Accept defaults and save this file as `[LAST NAME]_ps6.Rmd` to your `code` folder.

Copy and paste the contents of this file into your `[LAST NAME]_ps6.Rmd` file. Then change the `author: [YOUR NAME]` (line 4) to your name.

All of the following questions should be answered in this `.Rmd` file. There are code chunks with incomplete code that need to be filled in.

This problem set is worth 10 total points, plus three extra credit points. The point values for each question are indicated in brackets below. To receive full credit, you must both have the correct code **and include a comment describing what each line does**. In addition, some questions ask you to provide a written response in addition to the code. Unlike the first two problem sets, some of the code chunks are totally empty, requiring you to try writing the code from scratch. Make sure to comment each line, explaining what it is doing!

You are free to rely on whatever resources you need to complete this problem set, including lecture notes, lecture presentations, Google, your classmates…you name it. However, the final submission must be complete by you. There are no group assignments. To submit, compiled the completed problem set and upload the PDF file to Brightspace by midnight on 2023/11/03 Also note that the TAs and professors will not respond to Campuswire posts after 5PM on Friday, so don't wait until the last minute to get started!

Good luck!

ChatGPT Link [Optional]

*Copy the link to ChatGPT you used here: _______________________.

Question 0

Require `tidyverse` and `tidymodels` (for calculating AUC), and load the `admit_data.rds` (https://github.com/jbisbee1/DS1000_F2023/blob/main/Lectures/7_Classification/data/admit_data.rds?raw=true') data to an object called `ad` . (Tip: use the `read_rds()` function with the link to the raw data.)

```
require(tidyverse)
```

```
## Loading required package: tidyverse
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✓ ggplot2 3.3.6      ✓ purrr   0.3.4
## ✓ tibble  3.1.8      ✓ dplyr   1.0.10
## ✓ tidyr   1.2.1      ✓ stringr 1.4.1
## ✓ readr   2.1.2      ✓ forcats 0.5.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()
```

```
require(tidymodels)
```

```
## Loading required package: tidymodels
## ── Attaching packages ────────────────────────────────────── tidymodels 1.0.0 ──
## ✓ broom        1.0.1     ✓ rsample      1.1.0
## ✓ dials        1.0.0     ✓ tune         1.0.0
## ✓ infer        1.0.3     ✓ workflows    1.1.0
## ✓ modeldata    1.0.1     ✓ workflowsets 1.0.0
## ✓ parsnip      1.0.2     ✓ yardstick    1.1.0
## ✓ recipes      1.0.1
## ── Conflicts ───────────────────────────────────────── tidymodels_conflicts() ──
## ✕ scales::discard() masks purrr::discard()
## ✕ dplyr::filter()   masks stats::filter()
## ✕ recipes::fixed()  masks stringr::fixed()
## ✕ dplyr::lag()      masks stats::lag()
## ✕ yardstick::spec() masks readr::spec()
## ✕ recipes::step()   masks stats::step()
## • Dig deeper into tidy modeling with R at https://www.tmwr.org
```

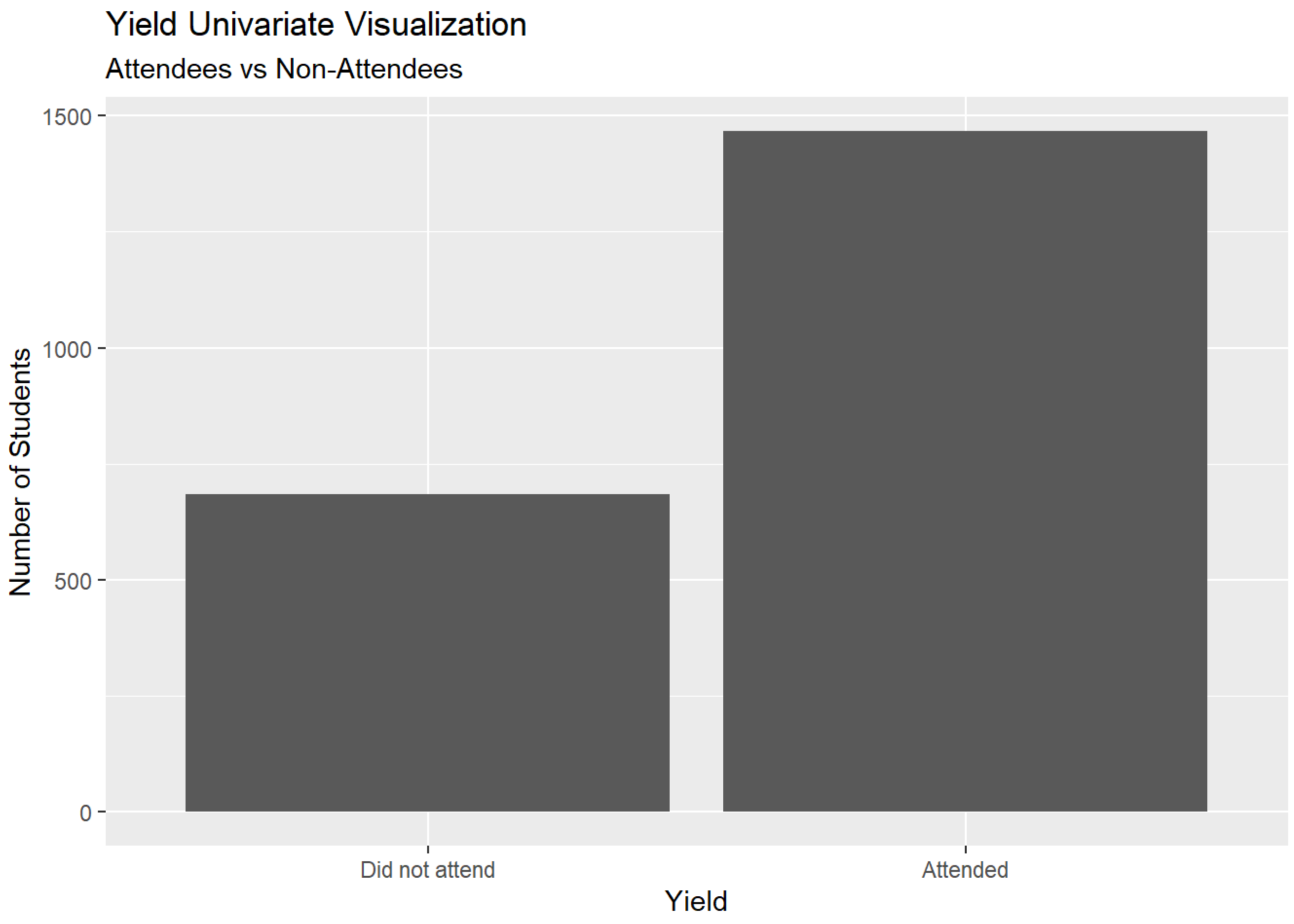
```
ad <- read_rds('https://github.com/jbisbee1/DS1000_F2023/blob/main/Lectures/7_Classification/dat
a/admit_data.rds?raw=true')
```

Question 1 [2 points + 1 EC]

Plot the univariate visualizations for `yield` , `income` , and `sat` . Justify your choices for how you are visualizing these variables. Then plot the conditional variation between `yield` and `income` , and `yield` and `sat` . Again, justify your choices and then interpret the results. Do these variables matter for `yield` ?

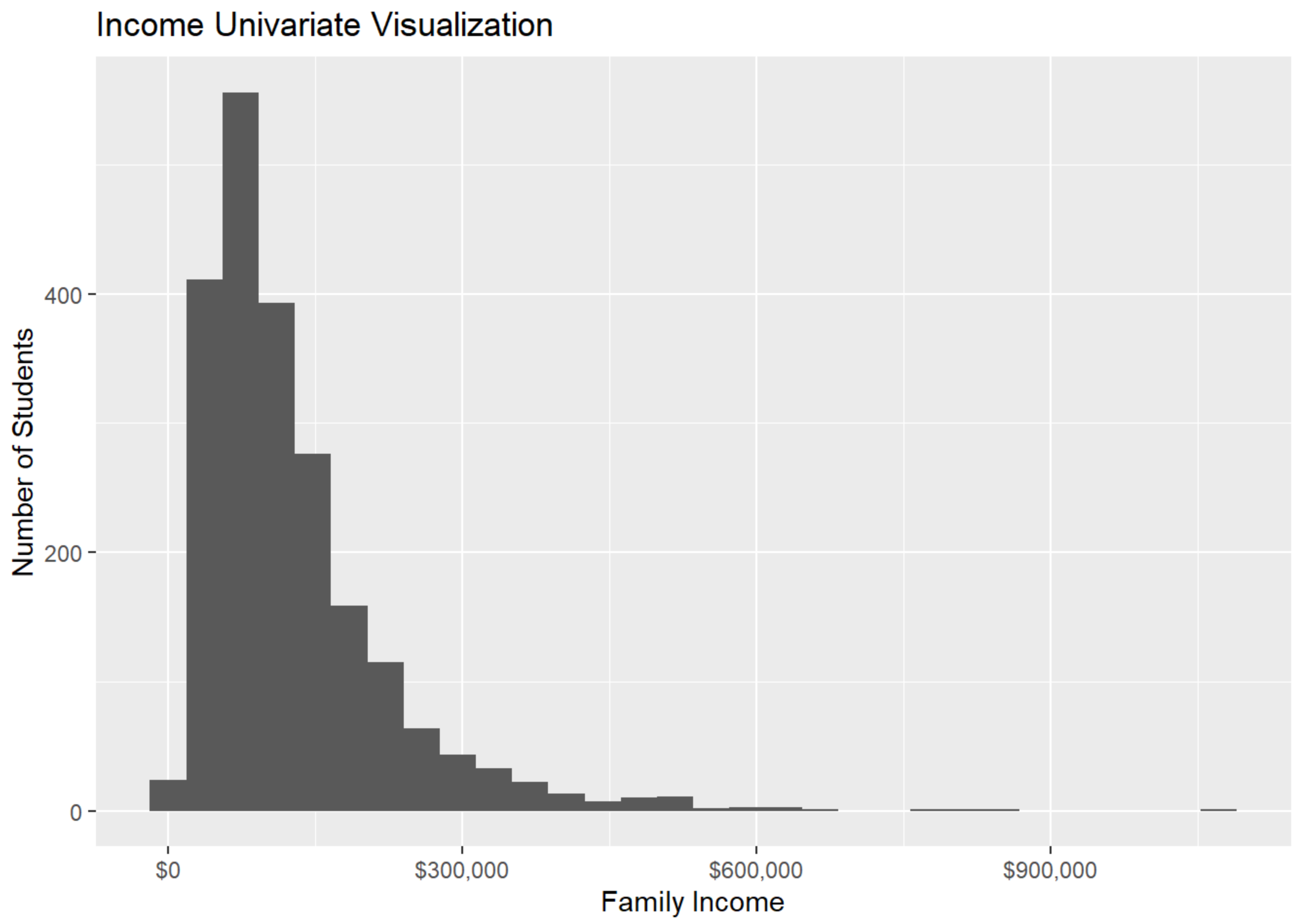
EXTRA CREDIT (+1 point): Explain the pattern you observe in the univariate visualization of the SAT scores. What might explain this?

```
# Univariate
ad %>%
  ggplot(aes(x = factor(yield))) +
  geom_bar() +
  scale_x_discrete(labels = c('Did not attend','Attended')) +
  labs(title = 'Yield Univariate Visualization',
       subtitle = 'Attendees vs Non-Attendees',
       x = 'Yield',
       y = 'Number of Students')
```



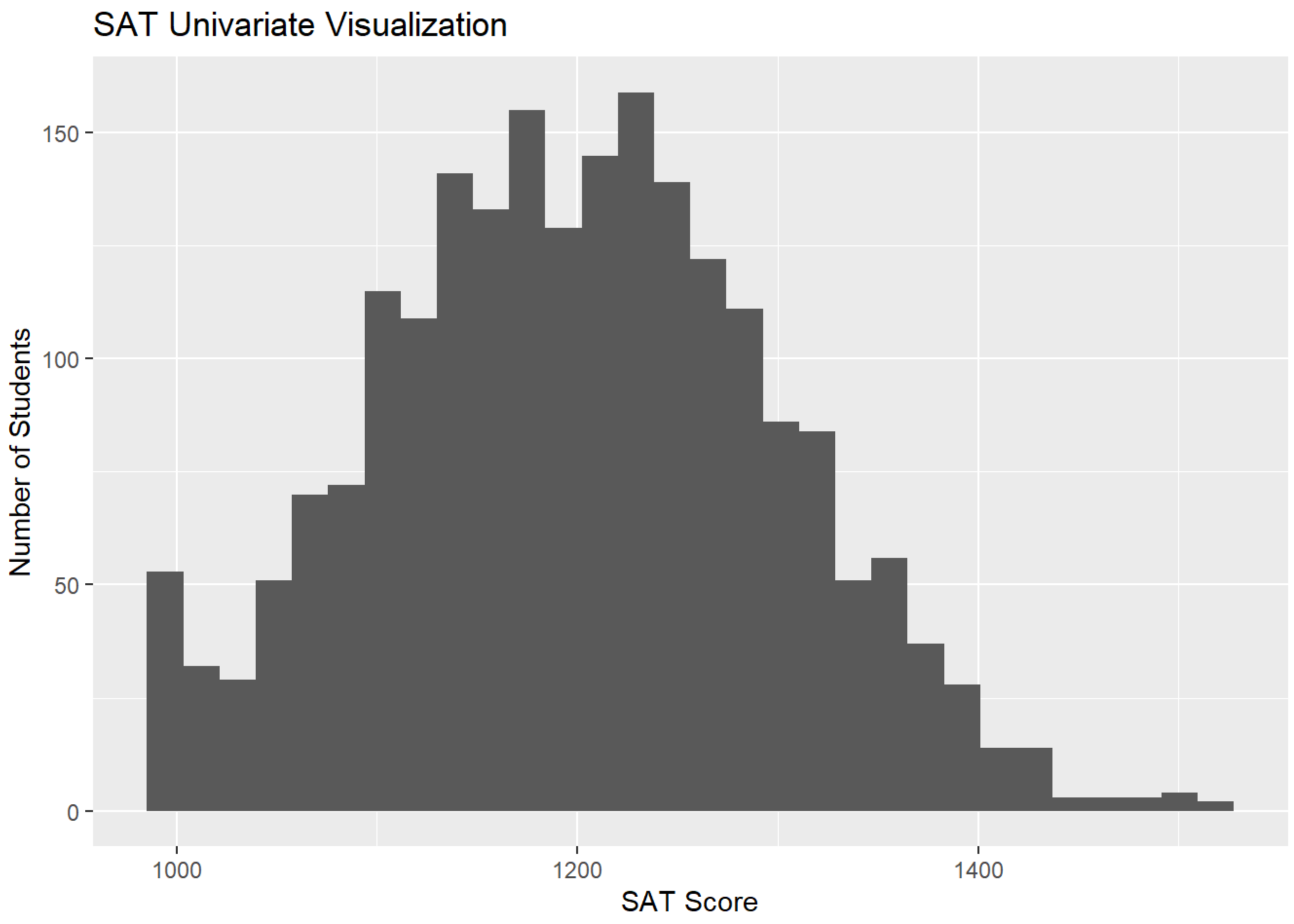
```
ad %>%
  ggplot(aes(x = income)) +
  geom_histogram() +
  scale_x_continuous(labels = dollar) +
  labs(title = 'Income Univariate Visualization',
       x = 'Family Income',
       y = 'Number of Students')
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

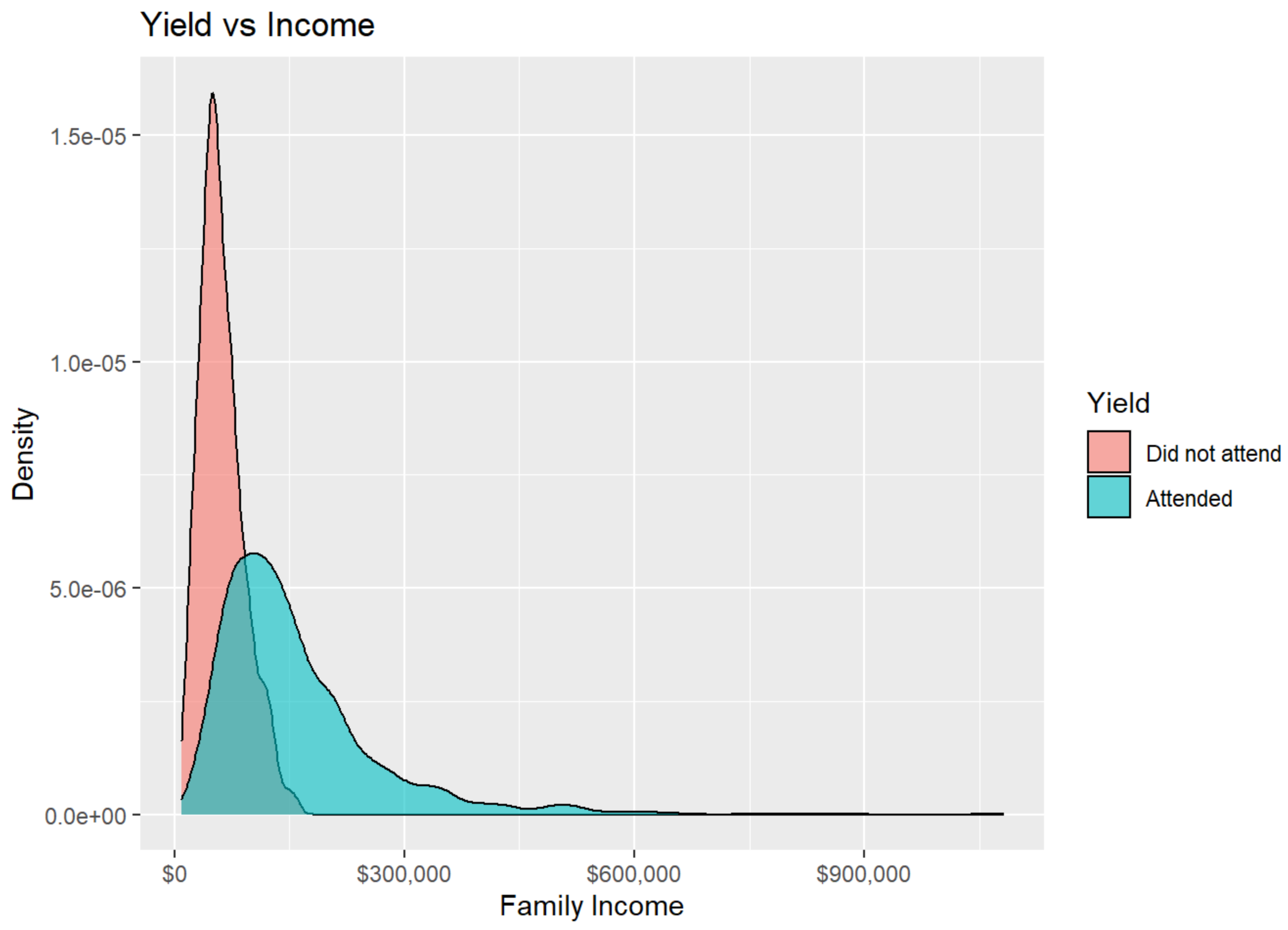


```
ad %>%
  ggplot(aes(x = sat)) +
  geom_histogram() +
  labs(title = 'SAT Univariate Visualization',
       x = 'SAT Score',
       y = 'Number of Students')
```

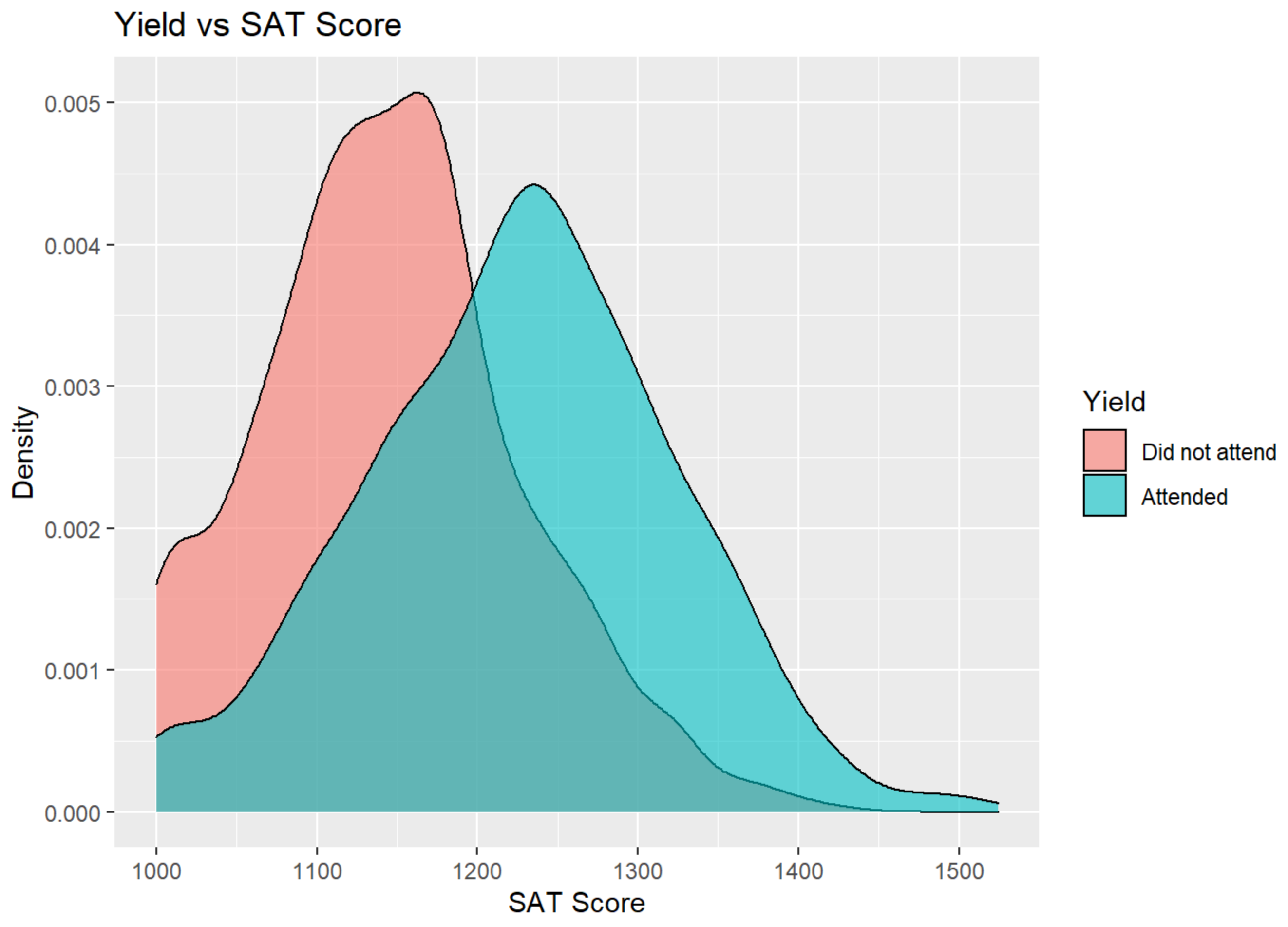
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
# Multivariate
ad %>%
  ggplot(aes(x = income,fill = factor(yield))) +
  geom_density(alpha = .6) +
  scale_x_continuous(labels = dollar) +
  scale_fill_discrete(name = 'Yield',labels = c('Did not attend','Attended')) +
  labs(title = 'Yield vs Income',
       x = 'Family Income',
       y = 'Density')
```



```
ad %>%
  ggplot(aes(x = sat,fill = factor(yield))) +
  geom_density(alpha = .6) +
  scale_fill_discrete(name = 'Yield',labels = c('Did not attend','Attended')) +
  labs(title = 'Yield vs SAT Score',
       x = 'SAT Score',
       y = 'Density')
```



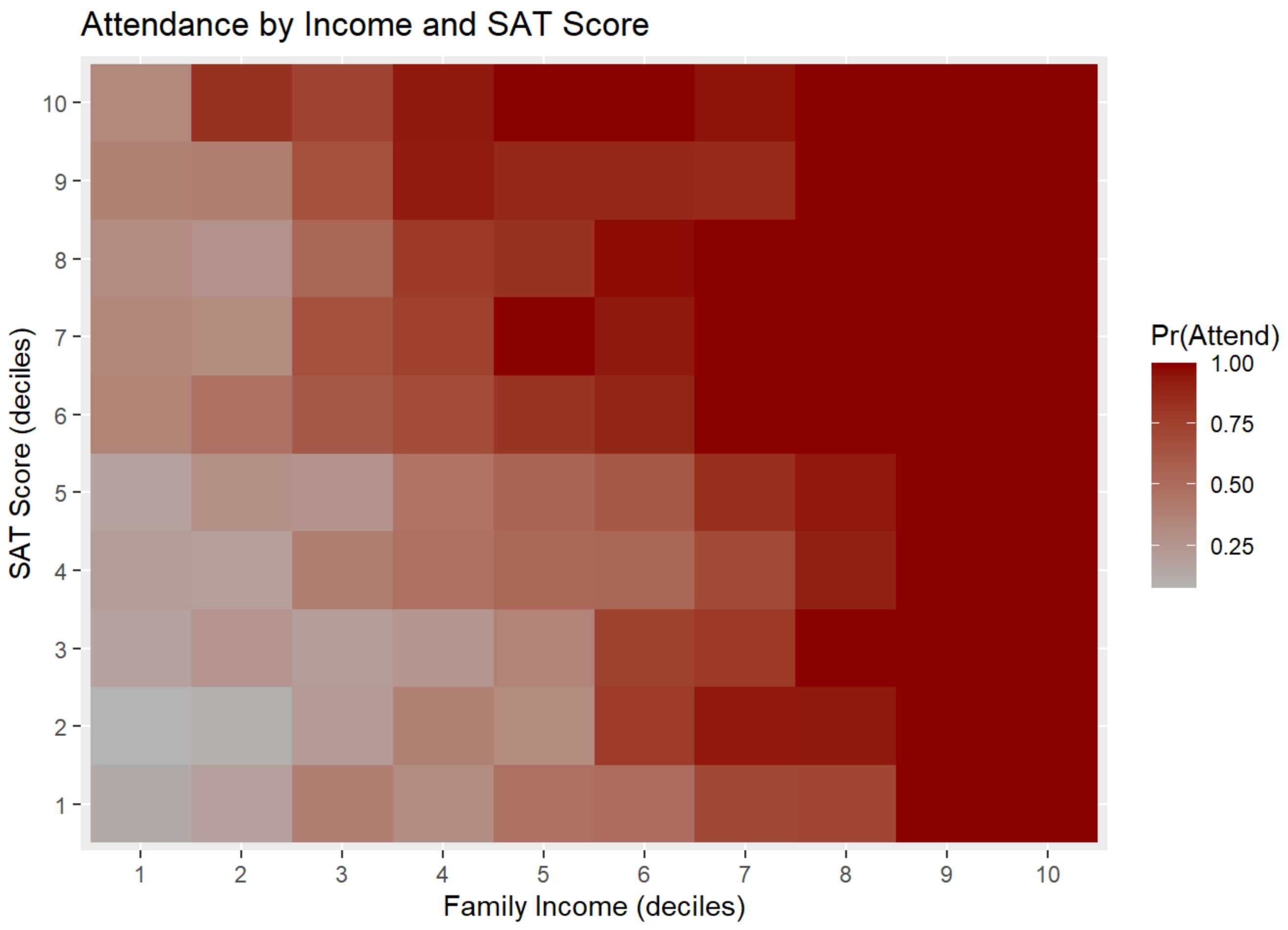
- For `yield`, I chose `geom_bar` since it is a categorical measure. For `income` and `sat`, both of which are continuous measures, I chose `geom_histogram`. For both multivariate visualizations, I chose `geom_density`, coloring the distributions by `factor(yield)`. Both of these plots indicate that students with higher income and with higher SAT scores are more likely to attend the college. EC (+1 point): the histogram for `sat` exhibits clumping around SAT scores of 1000. This might reflect one of two things. (1) this might reflect grade inflation on SAT scores, wherein scores that are lower than 1000 are rounded up. (2) this might reflect a threshold for applications established by the college, meaning that students with scores lower than 1,000 might not be able to apply.

Question 2 [2 points]

Look at these same conditional relationships between `yield` and `income` and `sat`, except divide the continuous measures of `income` and `sat` into deciles using the `ntile()` function, and create a single heatmap for all three variables, where the deciles of `income` and `sat` are on the axes, and the tiles are shaded by the average attendance in each cell. Which students are most likely to attend? Which are least likely to attend? Can you determine whether income or SAT scores matter more for attendance based on this plot?

```
ad %>%
  mutate(incomeDec = ntile(income,n = 10),
         satDec = ntile(sat,n= 10)) %>%
  group_by(incomeDec,satDec) %>%
  summarise(prob_attend = mean(yield)) %>%
  ggplot(aes(x = factor(incomeDec),y = factor(satDec),fill = prob_attend)) +
  geom_tile() +
  scale_fill_gradient(low = 'grey70',high = 'darkred',name = 'Pr(Attend)') +
  labs(title = 'Attendance by Income and SAT Score',
       x = 'Family Income (deciles)',
       y = 'SAT Score (deciles)')
```

```
## `summarise()` has grouped output by 'incomeDec'. You can override using the
## `.groups` argument.
```



- Students in higher income deciles and those in higher SAT score deciles are more likely to attend. However, it appears that income matters more, since differences in SAT scores don’t matter at all among the richest students. Put another way, every row (meaning SAT decile) shows differences in probability of attending based on income, whereas not every column (meaning income decile) shows differences in probability of attending based on SAT score.

Question 3 [2 points]

Now start with the simplest way of predicting attendance: the conditional mean. As above, calculate deciles for `income` and `sat` called `incomeDec` and `satDec` using the `ntile()` function. Then calculate the average attendance in each cell using `group_by()` and `mutate()`, and finally predict attendance as 1 if the average is greater than 0.5, and 0 otherwise, using an `ifelse()` function. Evaluate the performance in terms of **accuracy**, **sensitivity**, and **specificity**, making sure to clearly define each metric.

```
ad <- ad %>%
  mutate(incomeDec = ntile(income,n = 10),
         satDec = ntile(sat,n= 10)) %>%
  group_by(incomeDec,satDec) %>%
  mutate(prob_attend = mean(yield)) %>%
  mutate(pred_attend = ifelse(prob_attend > .5,1,0)) %>%
  ungroup()

ad %>%
  group_by(yield) %>%
  mutate(total_attend = n()) %>%
  group_by(yield,pred_attend,total_attend) %>%
  summarise(nStudents=n(),.groups = 'drop') %>%
  mutate(prop = percent(nStudents / total_attend)) %>%
  ungroup() %>%
  mutate(accuracy = percent(sum((yield == pred_attend)*nStudents) / sum(nStudents))) %>%
  filter(yield == pred_attend)
```

```
## # A tibble: 2 × 6
##   yield pred_attend total_attend nStudents prop  accuracy
##   <int>       <dbl>        <int>     <int> <chr> <chr>
## 1     0           0          684       535 78.2% 83%
## 2     1           1         1466      1255 85.6% 83%
```

- The overall accuracy – meaning the proportion of students who we accurately predicted to either attend or not attend out of all students – is 83%. The sensitivity – meaning the proportion of correct predictions of attendees out of total attendees – is 85.6%. The specificity – meaning the proportion of correct predictions of non-attendees out of total non-attendees – is 78.2%.

Question 4 [2 points]

Now predict whether students will attend using a linear regression model (using the `lm()` function) that predicts `yield` as a function of `income` and `sat` (**not** using deciles, just the continuous versions). Calculate **accuracy**, **sensitivity**, and **specificity** from this model where the threshold is again 0.5, and compare to the results from Question 3. Does this model do better?

```
m1 <- lm(yield ~ income + sat,ad)

ad %>%
  mutate(pred_attend = ifelse(predict(m1) > .5,1,0)) %>%
  group_by(yield) %>%
  mutate(total_attend = n()) %>%
  group_by(yield,pred_attend,total_attend) %>%
  summarise(nStudents=n(),.groups = 'drop') %>%
  mutate(prop = percent(nStudents / total_attend)) %>%
  ungroup() %>%
  mutate(accuracy = percent(sum((yield == pred_attend)*nStudents) / sum(nStudents))) %>%
  filter(yield == pred_attend)
```

```
## # A tibble: 2 × 6
##   yield pred_attend total_attend nStudents prop  accuracy
##   <int>       <dbl>        <int>     <int> <chr> <chr>
## 1     0           0          684       366 53.5% 80%
## 2     1           1         1466      1345 91.7% 80%
```

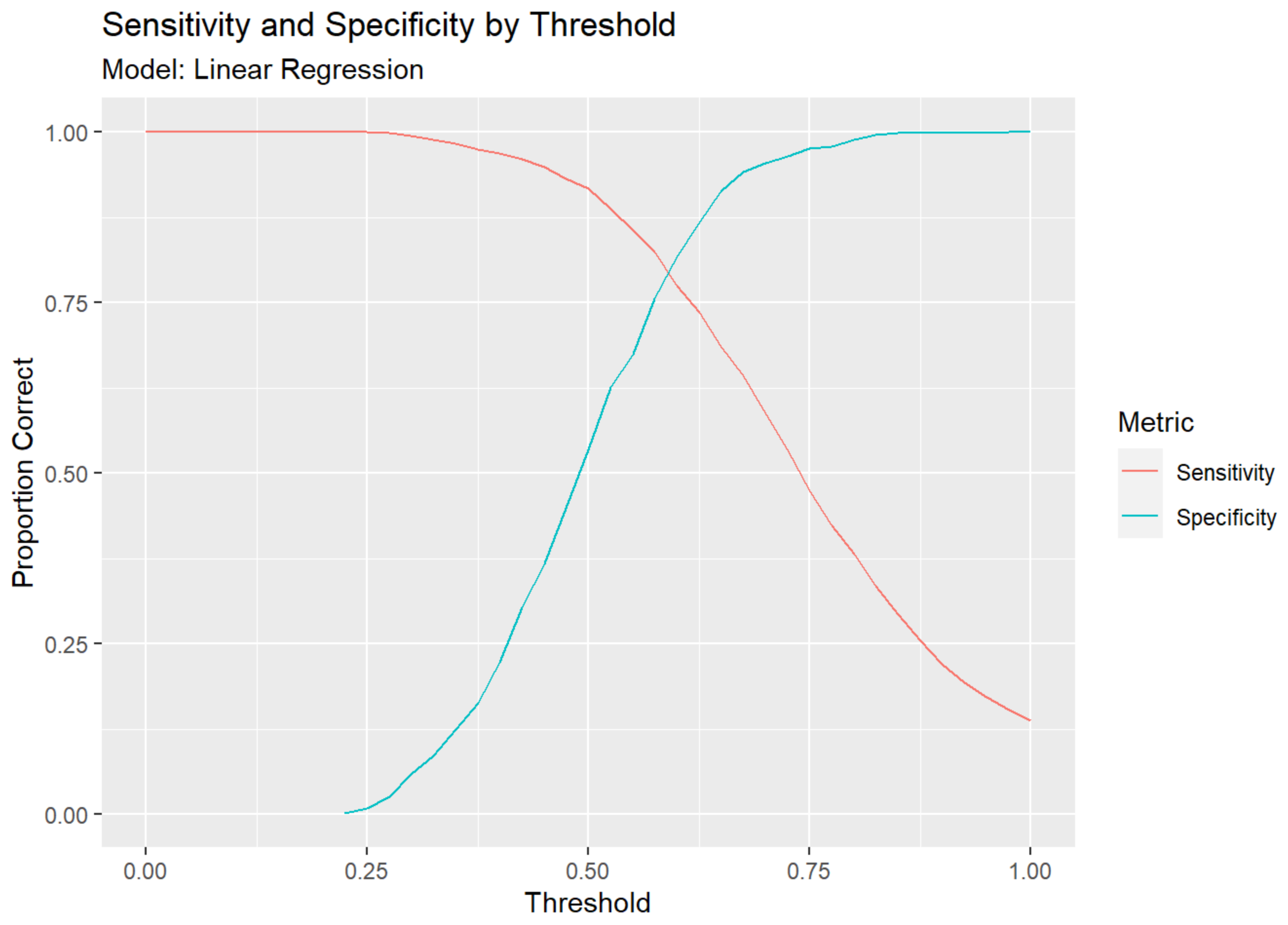
- Using a linear regression model, we have somewhat worse accuracy of 80%. Our sensitivity has increased from 85.6% to 91.7%, meaning we are doing a better job of accurately predicting attendees. However, our specificity has declined from 78.2% down to 53.5%, meaning we are doing a much worse job of accurately predicting non-attendees.

Question 5 [2 points]

Now recalculate **sensitivity**, **specificity**, and **accuracy** using different thresholds, ranging from 0 to 1, incrementing by 0.025 (use the `seq(from,to,by)` function). Plot the relationship between these thresholds and both the sensitivity and the specificity. What is the optimal threshold to balance the trade-off between **sensitivity** and **specificity**? Then plot ROC Curve and calculate the AUC.

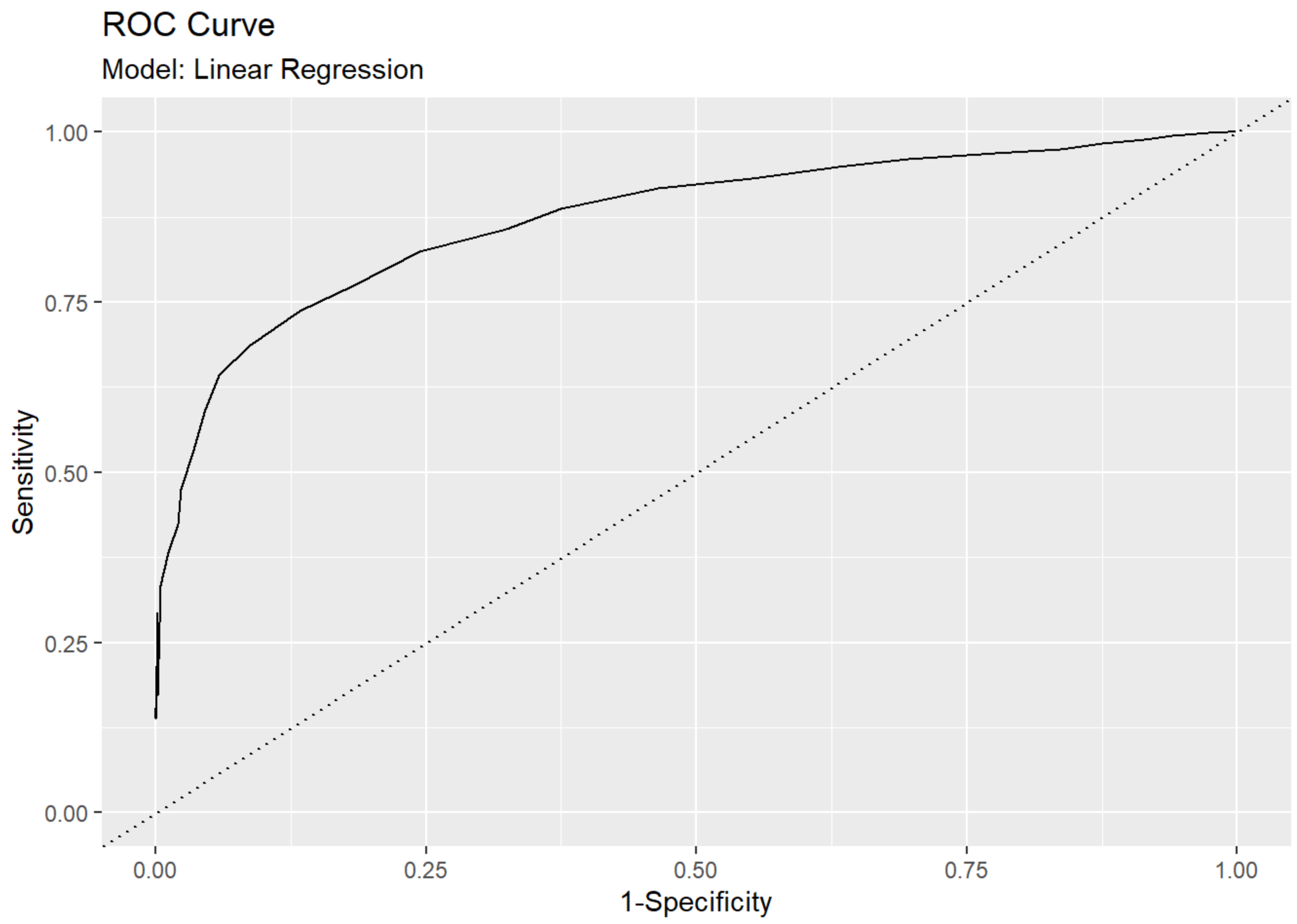
```r
toplot <- NULL
for(thresh in seq(0,1,by = 0.025)) {
  toplot <- ad %>%
  mutate(pred_attend = ifelse(predict(m1) > thresh,1,0)) %>%
  group_by(yield) %>%
  mutate(total_attend = n()) %>%
  group_by(yield,pred_attend,total_attend) %>%
  summarise(nStudents=n(),.groups = 'drop') %>%
  mutate(prop = nStudents / total_attend) %>%
  ungroup() %>%
  mutate(accuracy = sum((yield == pred_attend)*nStudents) / sum(nStudents)) %>%
  # filter(yield == pred_attend) %>% Also possible to filter here for either specificity or sens
itivity
  mutate(threshold = thresh) %>%
    bind_rows(toplot)
}

# Plot relationship between threshold and sens/spec
toplot %>%
  mutate(metric = ifelse(yield == 1 & pred_attend == 1,'Sensitivity',
                         ifelse(yield == 0 & pred_attend == 0,'Specificity',NA))) %>%
  drop_na(metric) %>%
  ggplot(aes(x = threshold,y = prop,color = metric)) +
  geom_line() +
  labs(title = 'Sensitivity and Specificity by Threshold',
       subtitle = 'Model: Linear Regression',
       x = 'Threshold',
       y = 'Proportion Correct',
       color = 'Metric')
```



```
# Plot ROC Curve
toplot %>%
  mutate(metric = ifelse(yield == 1 & pred_attend == 1,'Sensitivity',
                         ifelse(yield == 0 & pred_attend == 0,'Specificity',NA))) %>%
  drop_na(metric) %>%
  select(prop,metric,threshold) %>%
  spread(metric,prop) %>%
  ggplot(aes(x = 1-Specificity,y = Sensitivity)) +
  geom_line() +
  xlim(c(0,1)) + ylim(c(0,1)) +
  geom_abline(slope = 1,intercept = 0,linetype = 'dotted') +
  labs(title = 'ROC Curve',
       subtitle = 'Model: Linear Regression',
       x = '1-Specificity',
       y = 'Sensitivity')
```

```
## Warning: Removed 9 row(s) containing missing values (geom_path).
```



```
# Calculate AUC
require(tidymodels)
roc_auc(data = ad %>%
  mutate(pred_attend = predict(m1),
         truth = factor(yield,levels = c('1','0'))) %>%
  select(truth,pred_attend),truth,pred_attend)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.875
```

- Based on these results, the optimal point to balance between sensitivity and specificity is at a threshold of approximately 0.60.

Question 6 [2 EXTRA CREDIT points]

Re-do questions 4 and 5 using a logistic regression. Does this perform better than a linear regression model?

```
m2 <- glm(yield ~ income + sat,ad,family = binomial(link = 'logit'))
```

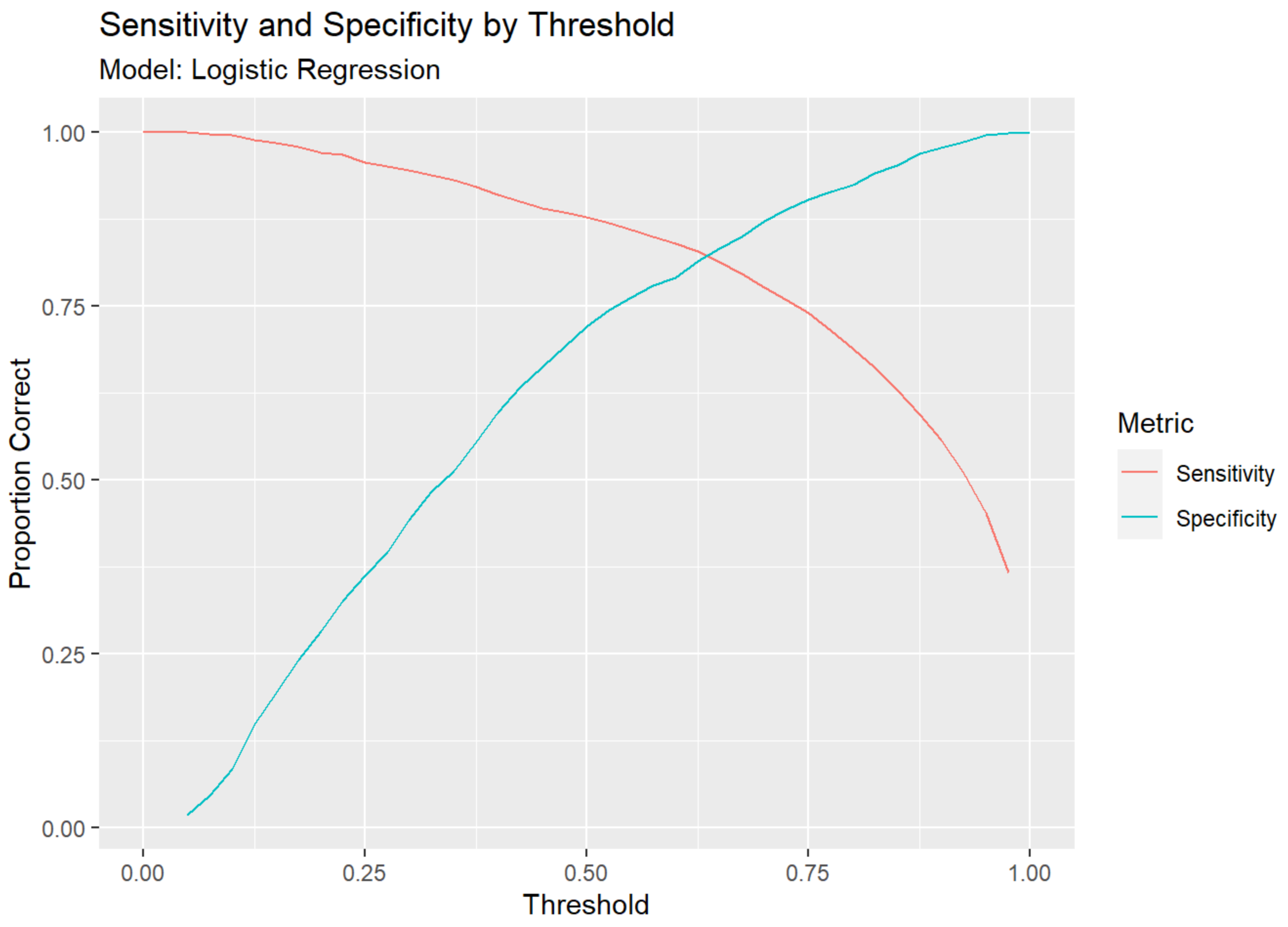
```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
ad %>%
  mutate(pred_attend = ifelse(predict(m2,type = 'response') > .5,1,0)) %>%
  group_by(yield) %>%
  mutate(total_attend = n()) %>%
  group_by(yield,pred_attend,total_attend) %>%
  summarise(nStudents=n(),.groups = 'drop') %>%
  mutate(prop = percent(nStudents / total_attend)) %>%
  ungroup() %>%
  mutate(accuracy = percent(sum((yield == pred_attend)*nStudents) / sum(nStudents))) %>%
  filter(yield == pred_attend)
```

```
## # A tibble: 2 × 6
##   yield pred_attend total_attend nStudents prop  accuracy
##   <int>       <dbl>        <int>     <int> <chr> <chr>
## 1     0           0          684       493 72%   83%
## 2     1           1         1466      1287 88%   83%
```

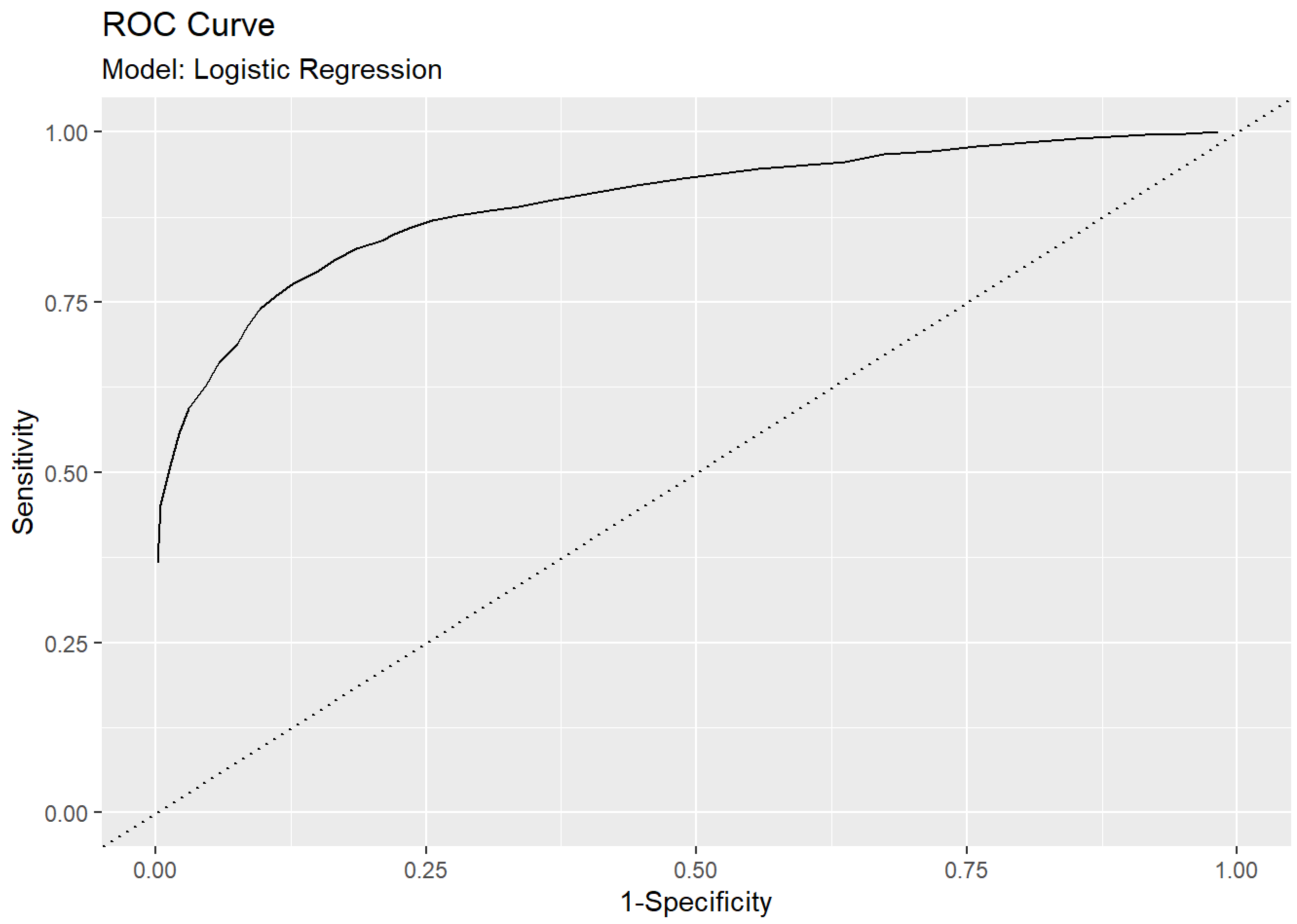
```
toplot <- NULL
for(thresh in seq(0,1,by = 0.025)) {
  toplot <- ad %>%
  mutate(pred_attend = ifelse(predict(m2,type = 'response') > thresh,1,0)) %>%
  group_by(yield) %>%
  mutate(total_attend = n()) %>%
  group_by(yield,pred_attend,total_attend) %>%
  summarise(nStudents=n(),.groups = 'drop') %>%
  mutate(prop = nStudents / total_attend) %>%
  ungroup() %>%
  mutate(accuracy = sum((yield == pred_attend)*nStudents) / sum(nStudents)) %>%
  filter(yield == pred_attend) %>%
  mutate(threshold = thresh) %>%
    bind_rows(toplot)
}

# Plot relationship between threshold and sens/spec
toplot %>%
  mutate(metric = ifelse(yield == 1 & pred_attend == 1,'Sensitivity','Specificity')) %>%
  ggplot(aes(x = threshold,y = prop,color = metric)) +
  geom_line() +
  labs(title = 'Sensitivity and Specificity by Threshold',
       subtitle = 'Model: Logistic Regression',
       x = 'Threshold',
       y = 'Proportion Correct',
       color = 'Metric')
```



```
# Plot ROC Curve
toplot %>%
  mutate(metric = ifelse(yield == 1 & pred_attend == 1,'Sensitivity','Specificity')) %>%
  select(prop,metric,threshold) %>%
  spread(metric,prop) %>%
  ggplot(aes(x = 1-Specificity,y = Sensitivity)) +
  geom_line() +
  xlim(c(0,1)) + ylim(c(0,1)) +
  geom_abline(slope = 1,intercept = 0,linetype = 'dotted') +
  labs(title = 'ROC Curve',
       subtitle = 'Model: Logistic Regression',
       x = '1-Specificity',
       y = 'Sensitivity')
```

```
## Warning: Removed 3 row(s) containing missing values (geom_path).
```



```
# Calculate AUC
require(tidymodels)
roc_auc(data = ad %>%
  mutate(pred_attend = predict(m2,type = 'response'),
         truth = factor(yield,levels = c('1','0'))) %>%
  select(truth,pred_attend),truth,pred_attend)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.897
```

- Using a logistic regression model, we have an accuracy of 83% which is better than our linear regression model overall. However, our sensitivity has decreased from 91.7% to 88%, meaning we are doing a worse job of accurately predicting attendees. But our specificity has increased from 53.5% to 72%, meaning we are doing a better job of accurately predicting non-attendees. The optimal threshold is again approximately 0.60.