Classification

Part 3

Prof. Bisbee

Vanderbilt University

Lecture Date: 2023/03/27

Slides Updated: 2023-03-26

Agenda

1. Recap of classification
2. Recap of evaluation
3. Using classifiers

Classification Recap

- Similar to **prediction**
- Difference is in the **outcome variable** Y:
  - Continuous: use regression model
  - Categorical / Binary: use classification model
- **NB:** *classification* is a type of *prediction*

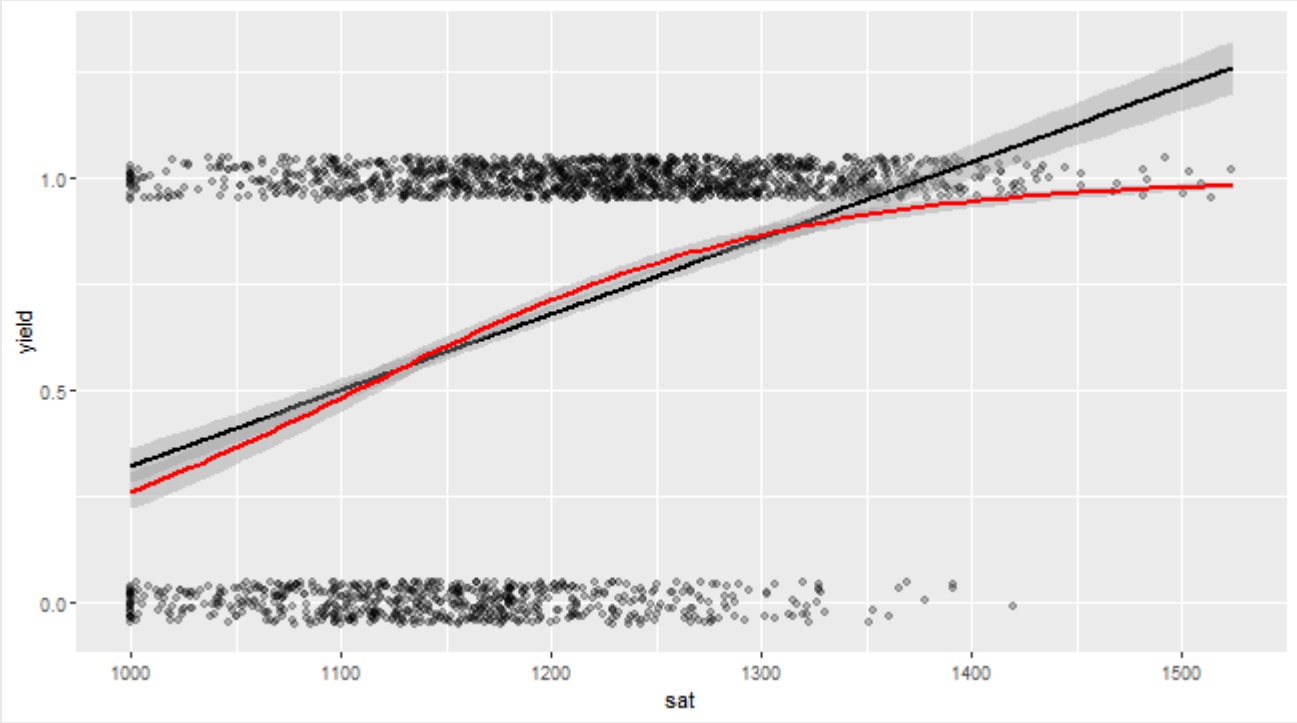
Classification Recap

- Thus far, only used binary Y
  - *Can* use linear regression model
  - *Better* to use logistic regression model
- Linear regression: `lm(formula,data)`
  - Predicts "latent" measures (`predict(mLM)`)
  - I.e., willingness to attend
- Logistic: `glm(formula,data,family = binomial(link = 'logit'))`
  - Predicts probabilities (`predict(mLG,type = 'response')`)
  - I.e., probability of attending

```
require(tidyverse)
require(tidymodels)
ad <- readRDS('../data/admit_data.rds')
```

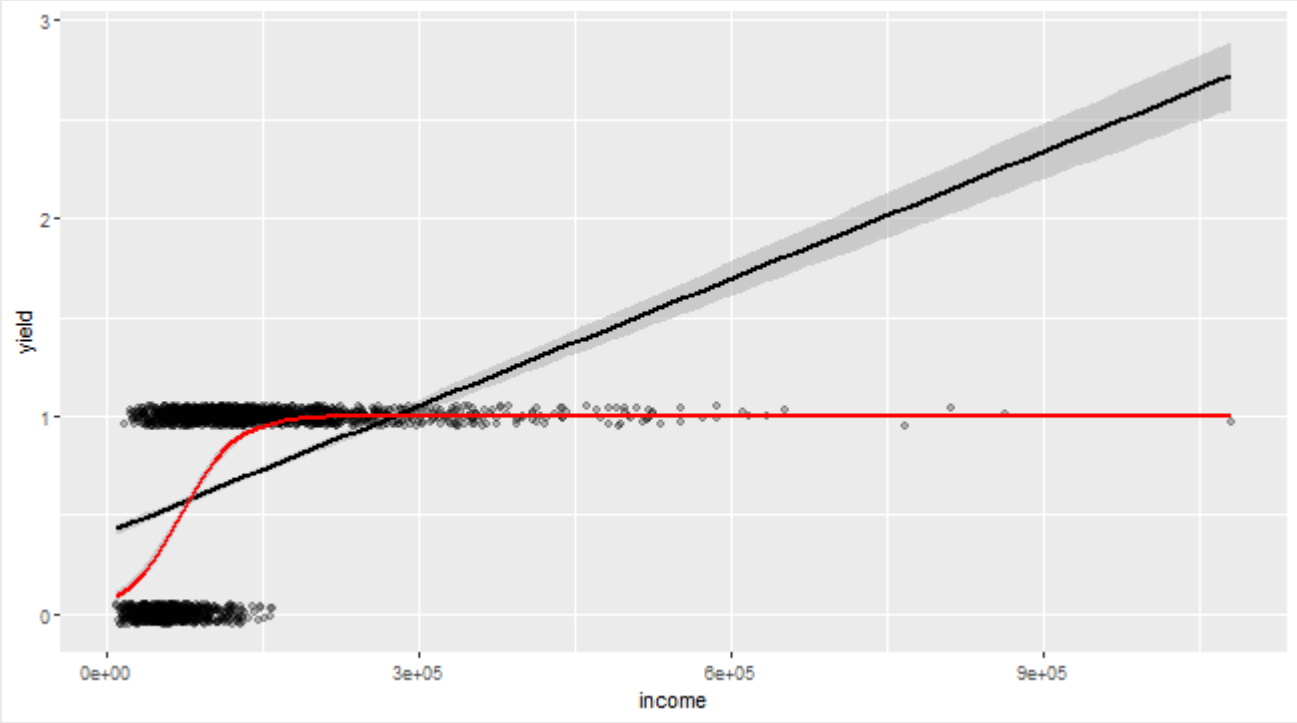
Classification Recap

```
ad %>% ggplot(aes(x = sat,y = yield)) +
  geom_jitter(width = .01,height = .05,alpha = .25) +
  geom_smooth(method = 'lm',color = 'black') +
  geom_smooth(method = 'glm',color = 'red',
              method.args = list(family = binomial(link = 'logit')))
```



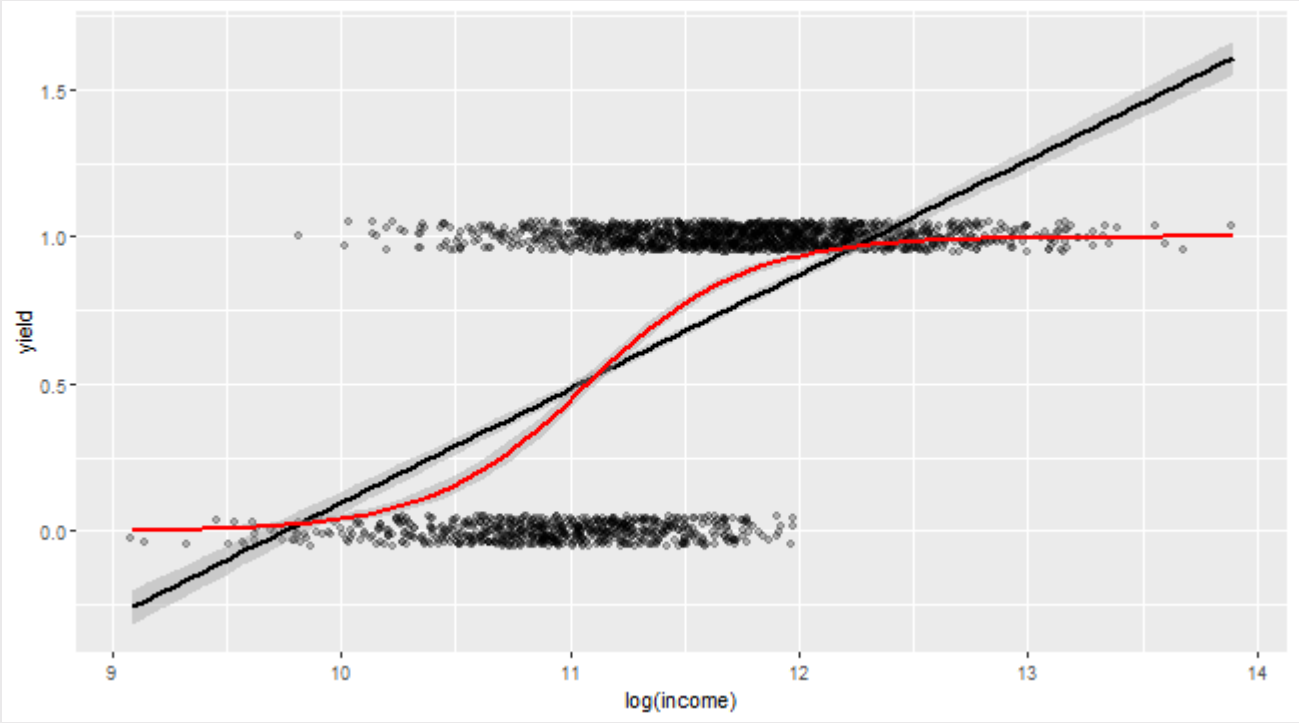
Classification Recap

```
ad %>% ggplot(aes(x = income,y = yield)) +
  geom_jitter(width = .01,height = .05,alpha = .25) +
  geom_smooth(method = 'lm',color = 'black') +
  geom_smooth(method = 'glm',color = 'red',
              method.args = list(family = binomial(link = 'logit')))
```



Classification Recap

```
ad %>% ggplot(aes(x = log(income),y = yield)) +
  geom_jitter(width = .01,height = .05,alpha = .25) +
  geom_smooth(method = 'lm',color = 'black') +
  geom_smooth(method = 'glm',color = 'red',
              method.args = list(family = binomial(link = 'logit')))
```



Classification Workflow

1. Train: `mLG <- glm(formula,data,family = binomial(link = 'logit'))`
2. Predict: `data$predY <- predict(mLG,type = 'response')`
3. Evaluate: `roc_auc(data,truth,estimate)`

- **NB:** Evaluation stage is *DEEP*
  - Classify observations based on threshold: `ifelse(predY > 0.5,1,0)`
  - Calculate accuracy by group across thresholds
  - **Sensitivity** and **Specificity** → ROC curve → AUC
  - Should be estimated via cross validation to prevent **overfitting**
  - Make sure `truth` is ordered in **reverse**: `factor(truth,levels = c('1','0'))`

Classification Workflow

- What do we get out of all this effort?
- A single classification algorithm that we think is best

```
# Train
form <- 'yield ~ sat + legacy + visit + registered + sent_scores + income + gpa + distance + net_price'
mLG <- glm(formula = as.formula(form),
           data = ad,family = binomial(link = 'logit'))
# Predict
pred <- ad %>%
  mutate(predY = predict(mLG,type = 'response'), # NB: type = 'response' for glm()!
         truth = factor(yield,levels = c('1','0'))) # NB: reorder outcome so that 1 is first!
# Evaluate
roc_auc(data = pred,truth = 'truth',estimate = 'predY')
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.915
```

Classification Workflow

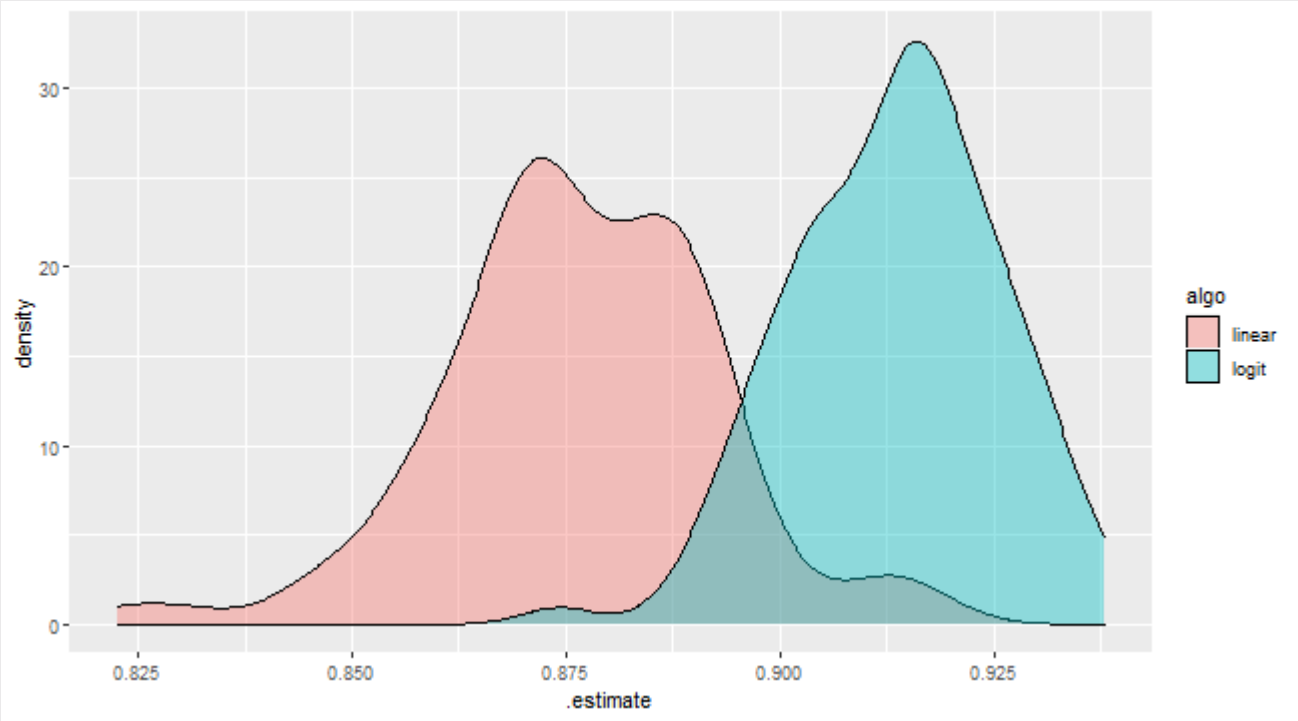
- But we should actually do this with **cross validation**

```
cvRes <- NULL
for(i in 1:100) {
  inds <- sample(1:nrow(ad),size = round(nrow(ad)*.8),replace = F)
  train <- ad %>% slice(inds)
  test <- ad %>% slice(-inds)

  # Train
  mLM <- lm(form,data = train)
  mLG <- glm(form,data = train,family = binomial(link = 'logit'))
  # Predict
  pred <- test %>%
  mutate(predLM = predict(mLM,newdata = test),
         predLG = predict(mLG,newdata = test,type = 'response'),
         truth = factor(yield,levels = c('1','0')))
  # Evaluate
  resLG <- roc_auc(data = pred,truth = 'truth',estimate = 'predLG') %>%
    mutate(algo = 'logit')
  resLM <- roc_auc(data = pred,truth = 'truth',estimate = 'predLM') %>%
    mutate(algo = 'linear')
  cvRes <- resLG %>% bind_rows(resLM) %>% bind_rows(cvRes)
}
```

Classification Workflow

```
cvRes %>%
  ggplot(aes(x = .estimate,fill = algo)) +
  geom_density(alpha = .4)
```



Using the Algorithm

- The algorithm can help us make decisions
- Recall the goals and constraints of this college:
  - **Goals**: Increase SAT to 1300
  - **Goals**: Admit 200 more students with incomes under $50,000
  - **Constraints**: Maintain total revenues of at least $30m
  - **Constraints**: Maintain entering class size of at least 1,466
- Tools:
  - Need-based and Merit-based aid
  - Effort for visit and registration
  - Other targeting? (ethics?)

Using the Algorithm

1. **Counterfactuals** for specific types of students
    - What is the probability a given student will attend?
    - `data_grid()` function from `modelr` package can help!
2. **Consulting** for changes in policy
    - If we increase price, what will happen to attendance?
    - Just predict on the **full data**

1. Counterfactuals: `data_grid()`

- What is the probability that this student will attend?

```
mLGFinal <- glm(formula = as.formula(form),
           data = ad,family = binomial(link = 'logit'))
require(modelr)
hypo_data <- ad %>%
  data_grid(legacy = 0,
            visit = 1,
            registered = 1,
            sent_scores = 1,
            sat = 1400,
            income = 95000,
            gpa = 3.9,
            distance = .1,
            net_price = 6875)

predict(mLGFinal,newdata = hypo_data,type = 'response')
```

```
##         1
## 0.9643731
```

1. Counterfactuals: `data_grid()`

- What is the probability they will attend if we increase the price by $10k?

```
hypo_data <- ad %>%
  data_grid(legacy = 0,
            visit = 1,
            registered = 1,
            sent_scores = 1,
            sat = 1400,
            income = 95000,
            gpa = 3.9,
            distance = .1,
            net_price = 16875)

predict(mLGFinal,newdata = hypo_data,type = 'response')
```

```
##         1
## 0.9392844
```

1. Counterfactuals: `data_grid()`

- Can combine with `data_grid()`

```
hypo_data <- ad %>%
  data_grid(legacy = 0,
            visit = 1,
            registered = 1,
            sent_scores = 1,
            sat = 1400,
            income = 95000,
            gpa = 3.9,
            distance = .1,
            net_price = c(6875,16875))

predict(mLGFinal,newdata = hypo_data,type = 'response')
```

```
##         1         2
## 0.9643731 0.9392844
```

1. Counterfactuals: `data_grid()`

- Can use `data_grid()` to calculate "typical" values for every variable
    - **Typical**: Mean for continuous, mode for categorical

```
(hypo_data <- ad %>%
  data_grid(.model = mLGFinal))
```

```
## # A tibble: 1 × 9
##     sat legacy visit registered sent_scores income   gpa
##   <dbl>  <dbl> <dbl>      <dbl>       <dbl>  <dbl> <dbl>
## 1 1200.      0     0          1           0 99712.  3.79
## # … with 2 more variables: distance <dbl>, net_price <dbl>
```

```
predict(mLGFinal,newdata = hypo_data,type = 'response')
```

```
##         1
## 0.8655441
```

1. Counterfactuals: `data_grid()`

- Compare otherwise typical admits who differ only in terms of GPA

```
hypo_data <- ad %>%
  data_grid(.model = mLGFinal,
            gpa = c(3.5,3.9))

predict(mLGFinal,newdata = hypo_data,type = 'response')
```

```
##         1         2
## 0.7936923 0.8865843
```

1. Counterfactuals: data_grid()

- Multiple comparisons are possible

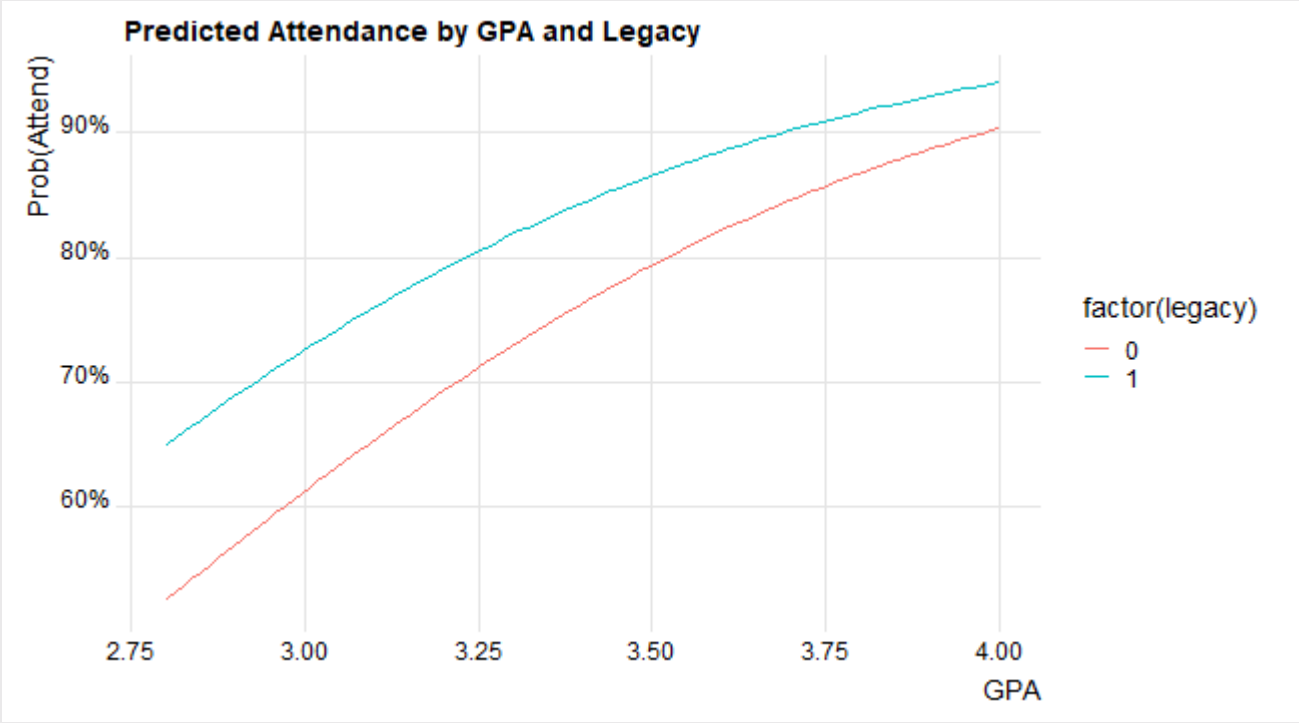
```
require(ggridges)
hypo_data <- ad %>%
  data_grid(.model = mLGFinal,
            gpa = seq_range(gpa,n = 100),
            legacy = c(0,1))

toplot <- hypo_data %>%
  mutate(preds = predict(mLGFinal,newdata = hypo_data,type =
'response'))

p <- toplot %>%
  ggplot(aes(x = gpa,y = preds,color = factor(legacy))) +
  geom_line() +
  theme_ridges() +
  scale_y_continuous(labels = scales::percent) +
  labs(title = 'Predicted Attendance by GPA and Legacy',
       x = 'GPA',
       y = 'Prob(Attend)')
```

1. Counterfactuals: `data_grid()`

```
p
```



2. Consulting: full data

- Currently, admit 1466 at ~$30.7m revenues

```
ad %>%
  filter(yield == 1) %>%
  summarise(totRev = dollar(sum(net_price)),
            totAtt = n())
```

```
##        totRev totAtt
## 1 $30,674,149   1466
```

2. Consulting: full data

- What if we increased the price for those who submit scores?

```
# Currently, 355 students who sent scores are predicted to attend
ad %>%
  mutate(preds = predict(mLGFinal,type = 'response')) %>%
  mutate(pred_class = ifelse(preds > .5,1,0)) %>%
  count(sent_scores,pred_class)
```

```
##   sent_scores pred_class    n
## 1           0          0  627
## 2           0          1 1093
## 3           1          0   75
## 4           1          1  355
```

2. Consulting: full data

- What if we increased the price for those who submit scores?

```
# If price increases by $5k, number of predicted drops to 338
hypo <- ad %>%
  mutate(net_price = ifelse(sent_scores == 1,
                            net_price + 5000,net_price))

hypo %>%
  mutate(preds = predict(mLGFinal,newdata = hypo,type = 'response'))
%>%
  mutate(pred_class = ifelse(preds > .5,1,0)) %>%
  count(sent_scores,pred_class)
```

```
##   sent_scores pred_class    n
## 1           0          0  627
## 2           0          1 1093
## 3           1          0   92
## 4           1          1  338
```

2. Consulting: full data

- What if we increased the price for those who submit scores?

```
# BUT we make more total revenue
hypo %>%
  mutate(preds = predict(mLGFinal,newdata = hypo,type = 'response'))
%>%
  mutate(pred_class = ifelse(preds > .5,1,0)) %>%
  filter(pred_class == 1) %>%
  summarise(tot_rev = scales::dollar(sum(net_price)))
```

```
##       tot_rev
## 1 $31,264,662
```

2. Consulting: full data

- What if we increased the price for those who submit scores?

```
# Although total admits declined
hypo %>%
  mutate(preds = predict(mLGFinal,newdata = hypo,type = 'response'))
%>%
  mutate(pred_class = ifelse(preds > .5,1,0)) %>%
  filter(pred_class == 1) %>%
  count()
```

```
##      n
## 1 1431
```

2. Consulting: full data

- Can we increase SAT scores to 1300?

```
ad %>%
  filter(yield == 1) %>%
  summarise(satAvg = round(mean(sat)))
```

```
##   satAvg
## 1   1226
```

- Reduce price for those above 1300

2. Consulting: full data

- Can we increase SAT scores to 1300?
- Reduce price for those above 1300

```
hypo <- ad %>%
  mutate(net_price = ifelse(sat >= 1300,
                            net_price - 5000,net_price))

hypo %>%
  mutate(preds = predict(mLGFinal,newdata = hypo,type = 'response'))
%>%
  mutate(pred_class = ifelse(preds > .5,1,0)) %>%
  filter(pred_class == 1) %>%
  summarise(satAvg = round(mean(sat)),
            tot_rev = scales::dollar(sum(net_price)),
            totAttend = n())
```

```
##   satAvg     tot_rev totAttend
## 1   1234 $28,294,452      1450
```

2. Consulting: full data

- Can we increase SAT scores to 1300?
- BUT need to make up losses

```
hypo <- ad %>%
  mutate(net_price = ifelse(sat >= 1300,net_price - 5000,
                            ifelse(sat < 1300,net_price +
2500,net_price)))

hypo %>%
  mutate(preds = predict(mLGFinal,newdata = hypo,type = 'response'))
%>%
  mutate(pred_class = ifelse(preds > .5,1,0)) %>%
  filter(pred_class == 1) %>%
  summarise(satAvg = round(mean(sat)),
            tot_rev = scales::dollar(sum(net_price)),
            totAttend = n())
```

```
##   satAvg     tot_rev totAttend
## 1   1236 $30,084,732      1414
```

2. Consulting: full data

- Looking across many different values? **Loops**!

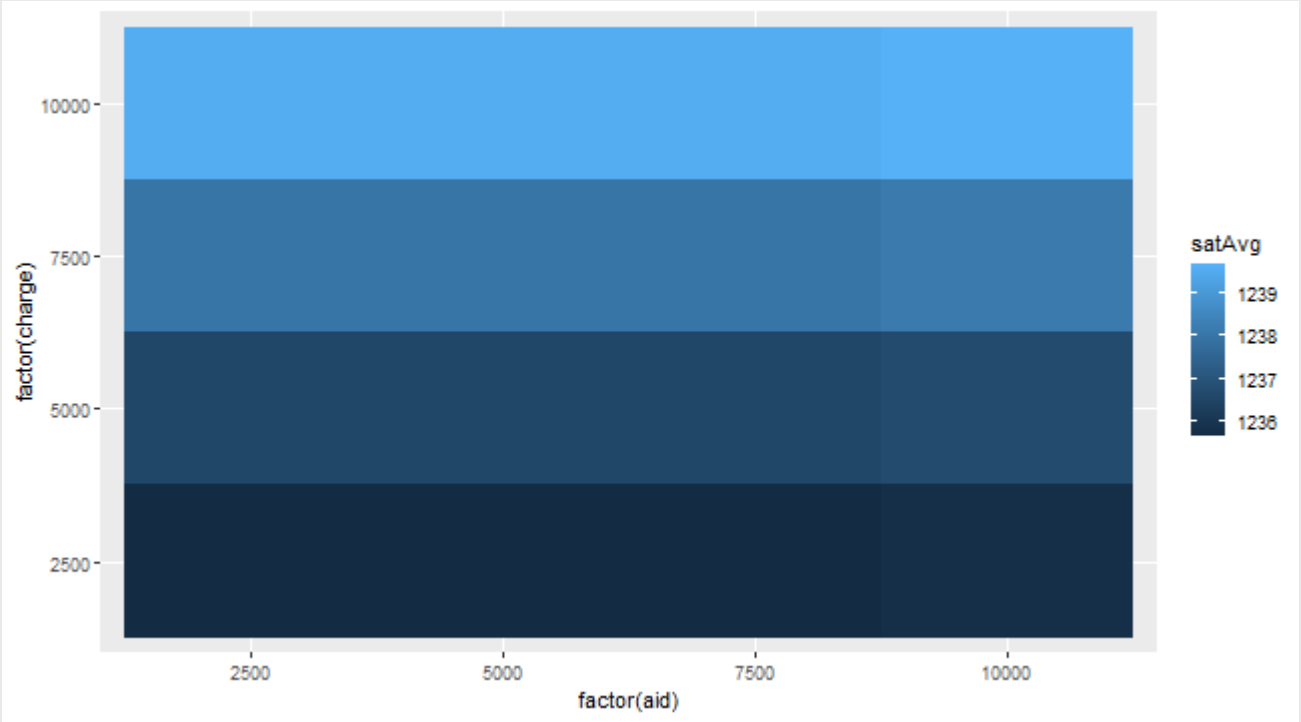
```
toplot <- NULL
for(aid in c(2500,5000,7500,10000)) {
  for(charge in seq(2500,10000,by = 2500)) {
    hypo <- ad %>%
      mutate(net_price = ifelse(sat >= 1300,net_price - aid,
                                ifelse(sat < 1300,net_price +
charge,net_price)))

  tmp <- hypo %>%
    mutate(preds = predict(mLGFinal,newdata = hypo,type =
'response')) %>%
    mutate(pred_class = ifelse(preds > .5,1,0)) %>%
    filter(pred_class == 1) %>%
    summarise(satAvg = mean(sat),
              tot_rev = sum(net_price),
              totAttend = n()) %>%
    ungroup() %>%
    mutate(aid = aid,
           charge = charge)

  toplot <- toplot %>%
```

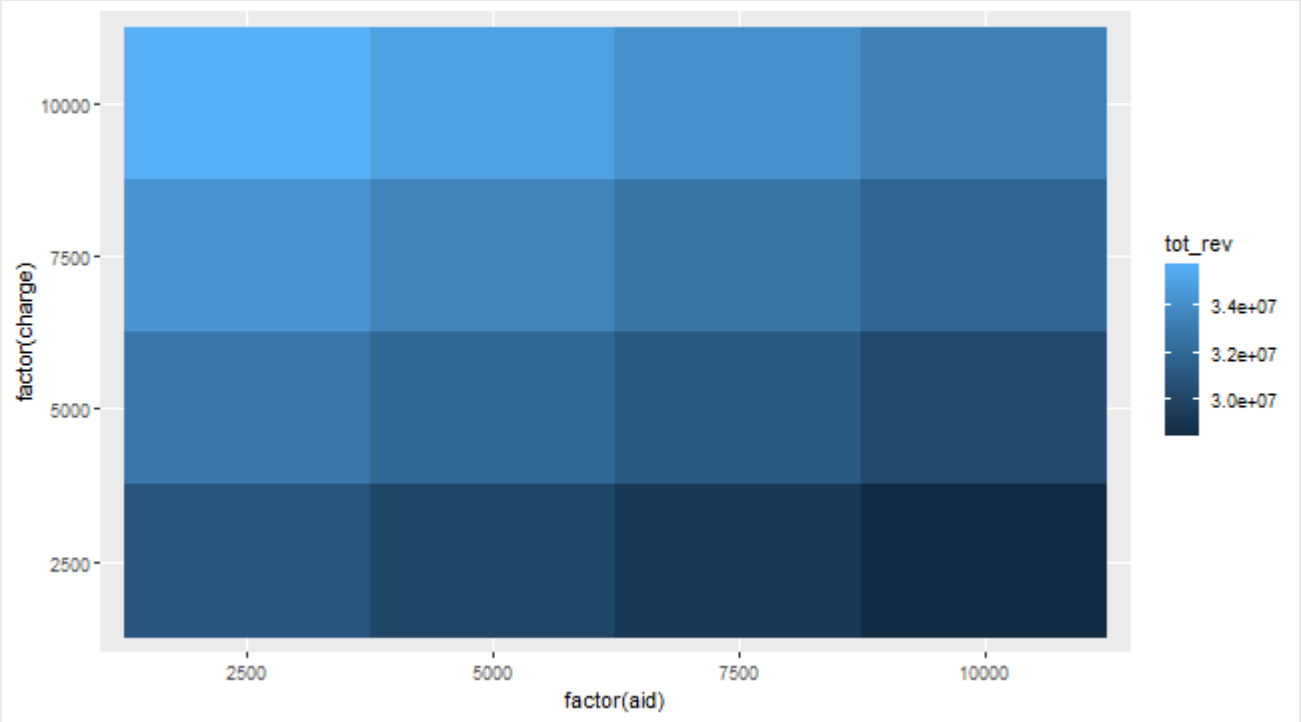
2. Consulting: full data

```
toplot %>%
  ggplot(aes(x = factor(aid),y = factor(charge),fill = satAvg)) +
  geom_tile()
```



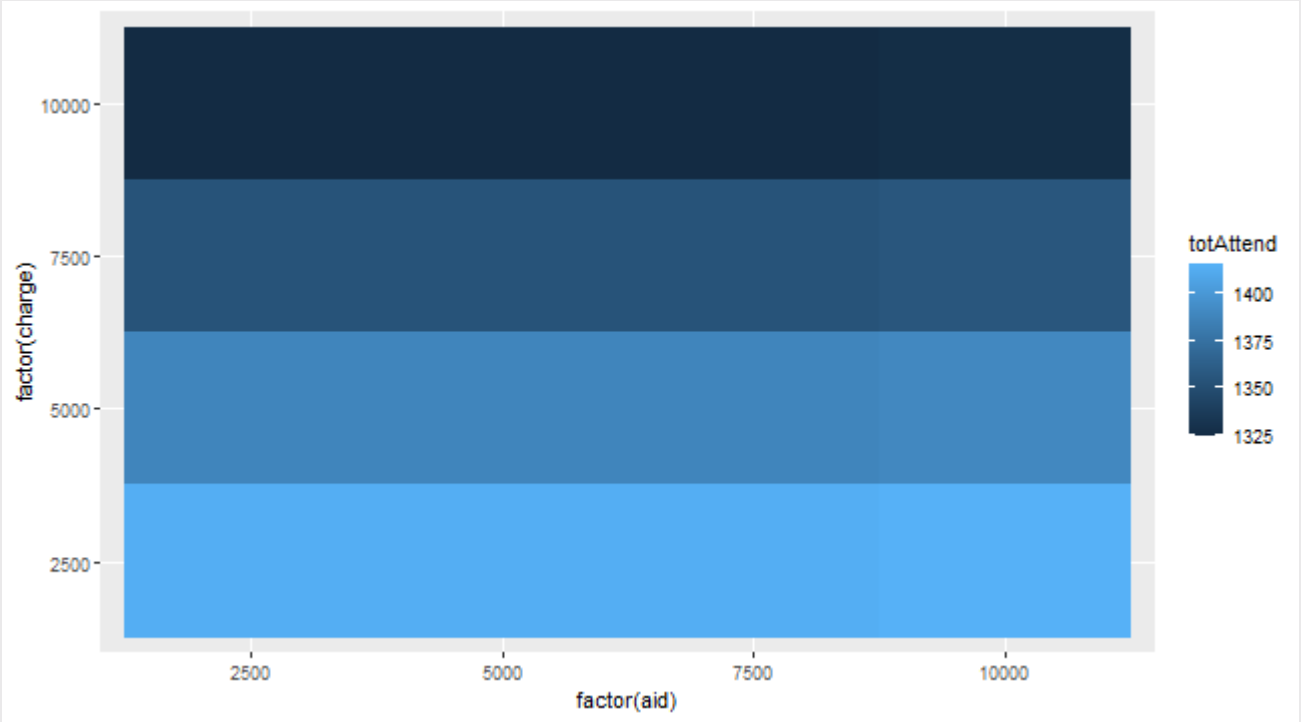
2. Consulting: full data

```
toplot %>%
  ggplot(aes(x = factor(aid),y = factor(charge),fill = tot_rev)) +
  geom_tile()
```



2. Consulting: full data

```
toplot %>%
  ggplot(aes(x = factor(aid),y = factor(charge),fill = totAttend)) +
  geom_tile()
```



Conclusion

- Remember the workflow:
    1. Train: `glm()`
    2. Predict: `predict()`
    3. Evaluate: `roc_auc()`
    4. Adjust: `net_price = ifelse()`
- Go to Brightspace and take the **15th** quiz
    - The password to take the quiz is ####
- **Homework:**
    - Problem Set 7 (due 2023-03-31 by 11:59PM)