

Classification

Part 2

Prof. Bisbee

Vanderbilt University

Slides Updated: 2024-03-16

Agenda

1. Introducing **logit**
2. Running logit
3. Evaluating logit

Logit Regression

- A different **type** of **regression**
 - What do we mean by **type**?
- Let's take a step back

```
require(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version  
## 4.3.2
```

```
require(scales)
```

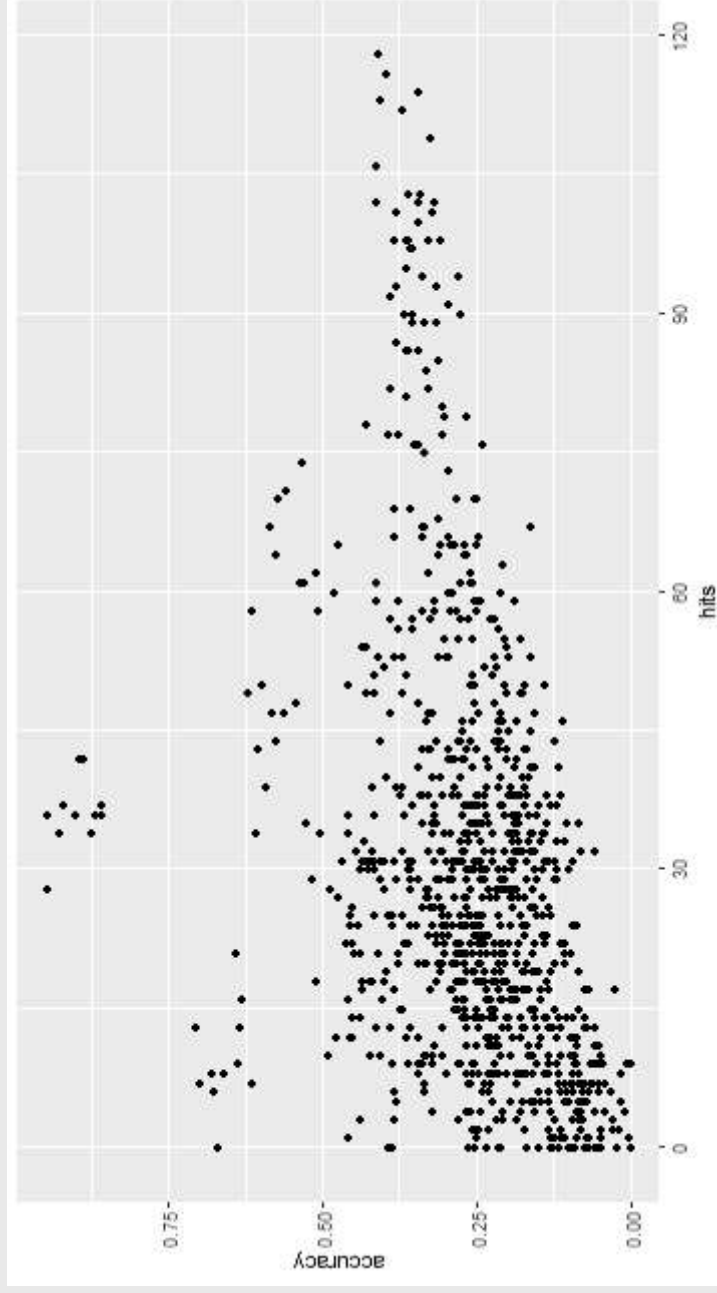
```
## Warning: package 'scales' was built under R version 4.3.3
```

```
fn <-  
read_rds('C:/Users/bisbeejh/Dropbox/2024_spring/DS1000_S2024/data/fn_cl
```

Regression Types

- "Linear" regression...why is it "linear"?

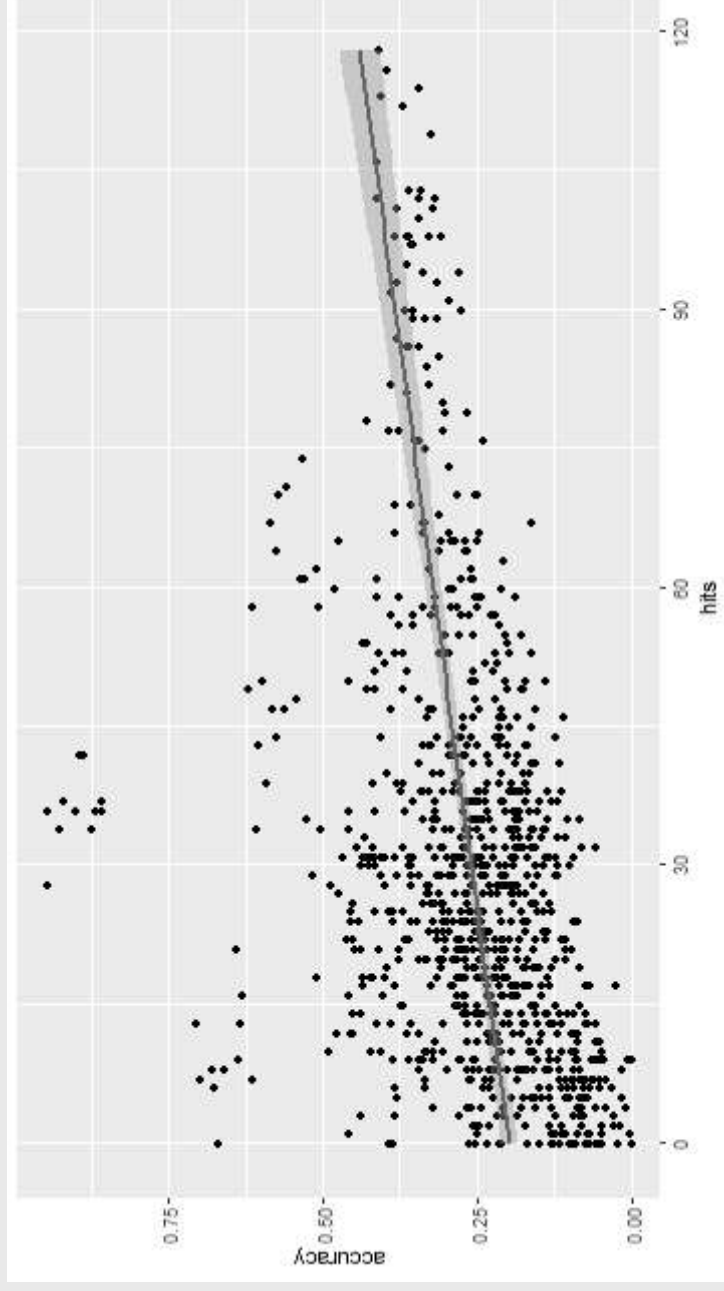
```
(p <- fn %>%  
  ggplot(aes(x = hits, y = accuracy)) +  
  geom_point())
```



Regression Types

- "Linear" regression...why is it "linear"?
- Because you can summarize it with a line!

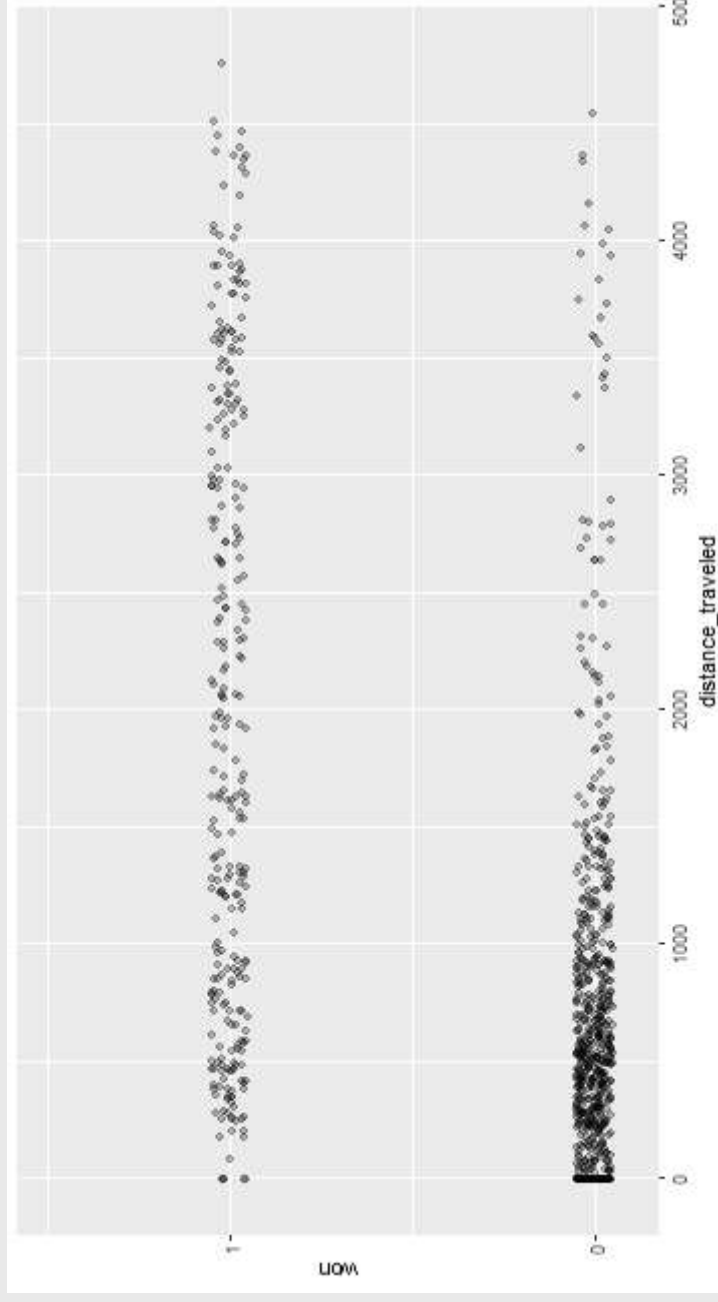
```
p + geom_smooth(method = 'lm')
```



Regression Types

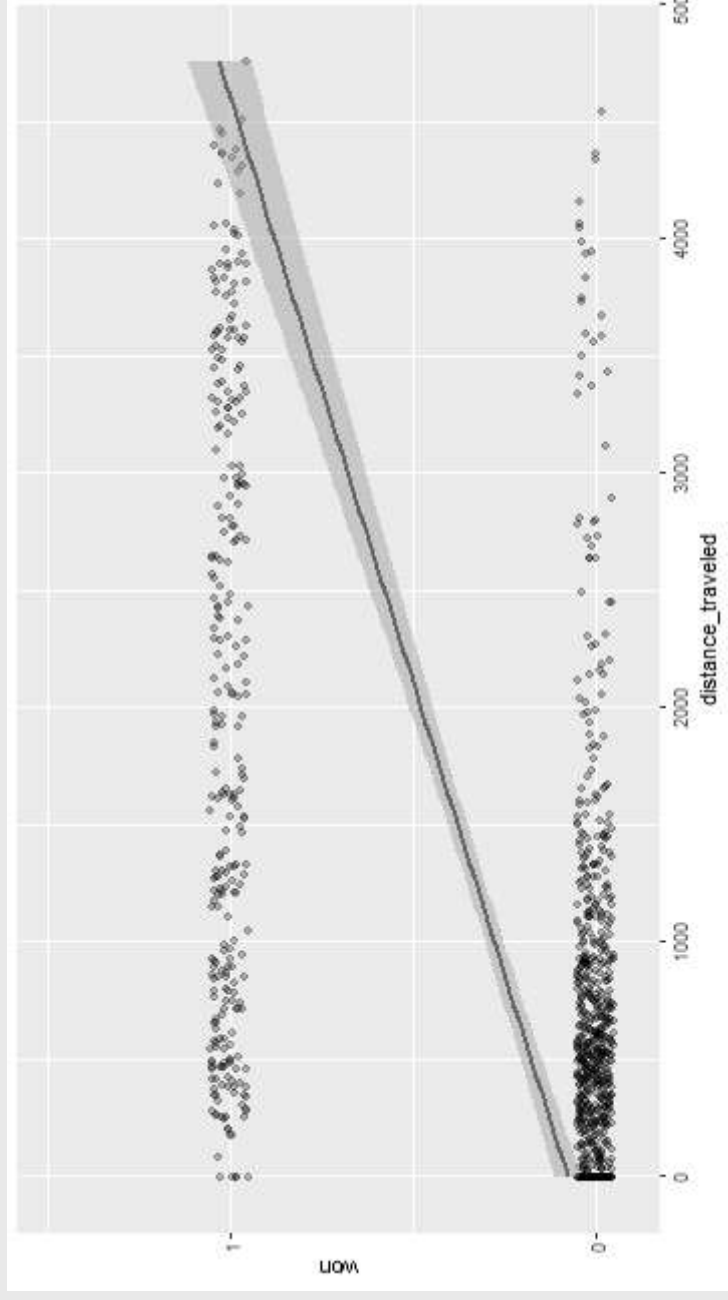
- But what if the outcome is binary?

```
(p <- fn %>% ggplot(aes(x = distance_traveled, y = won)) +  
  scale_y_continuous(breaks = c(0,1),limits = c(-.1,1.5)) +  
  geom_jitter(width = .01,height = .05,alpha = .25))
```



Regression Types

- But what if the outcome is binary?
- Lines seem too clumsy
 - If 1 = won, how can you go higher?



Logit

- Theory: binary outcomes are **proxies** for some **latent** measure
 - Binary outcome won: either placed first or did not
 - Latent outcome placed: continuous measure
 - Might also imagine **ability**: continuous measure
- The higher your *ability*, the more likely you are to win
- Logit regression: model the *ability*
 - What is *ability* actually?
 - Probability of winning: $Pr(won)$
- Part of a broader class of models called "generalized linear model" (GLM)

$$Pr(y = 1|x) = G(\alpha + \beta X)$$

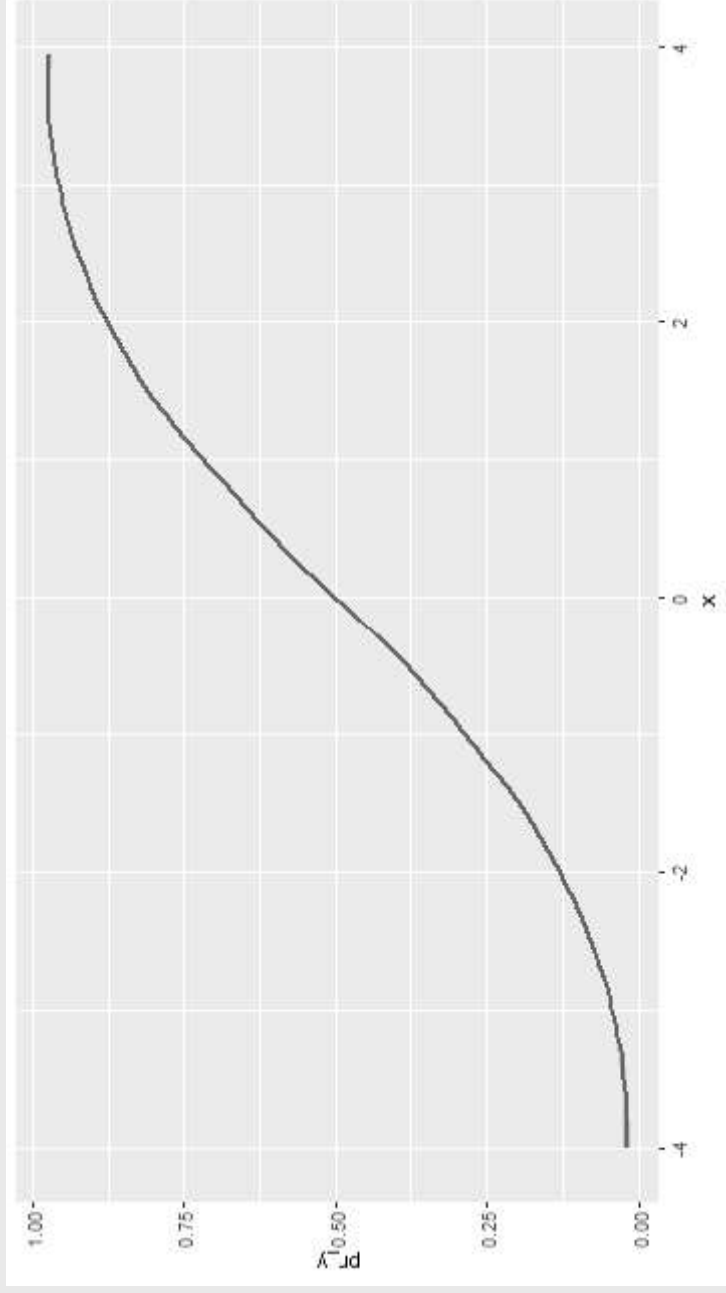
GLMs

- $Pr(y = 1|x) = G(\alpha + \beta X)$
- Does this look familiar?
- Linear regression: $Y = \alpha + \beta X$
 - Outcome: $Y \rightarrow Pr(y = 1|x)$
 - Mapping: $\alpha + \beta X \rightarrow G(\alpha + \beta X)$
- G is the "link function"
 - Transforms values of $\alpha + \beta X$ into **probabilities**
- Logistic function: specific type of link function

$$G(x) = \frac{1}{1 + \exp(-x)}$$

Logistic Function

```
x <- runif(100, -4, 4)
pr_y <- 1/(1 + exp(-x))
as_tibble(pr_y = pr_y, x = x) %>%
  ggplot(aes(x = x, y = pr_y)) +
  geom_smooth()
```

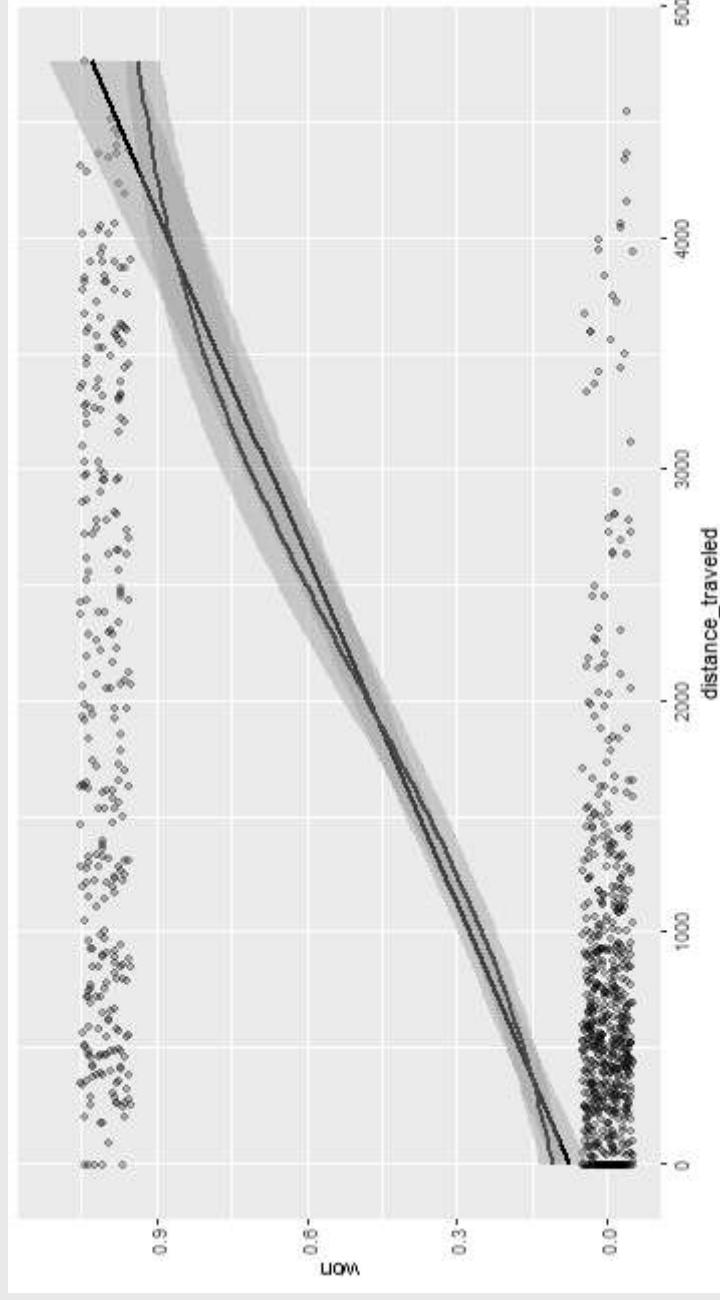


Logistic Function

- But what about real data like $\alpha + \beta X$?
- $G(X) = \frac{\exp(\alpha + \beta X)}{1 + \exp(\alpha + \beta X)}$
- We estimate this with `glm(formula, data, family)`
 - Note similarity to `lm(formula, data)`
- `family = binomial(link = "logit")`

Logistic Regression (logit)

```
fn %>% ggplot(aes(x = distance_traveled, y = won)) +  
  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'))))
```



Logistic Regression (logit)

```
# Train model
mlogit <- glm(formula = won ~ distance_traveled, data = fn, family =
binomial(link = 'logit'))

# Predict model
fn <- fn %>%
  mutate(prob_won = predict(mlogit, type = 'response')) %>%
  mutate(pred_won = ifelse(prob_won > .5, 1, 0))

# Evaluate model
eval <- fn %>%
  group_by(won) %>%
  mutate(total_games = n()) %>%
  group_by(won, pred_won, total_games) %>%
  summarise(nGames=n(), .groups = 'drop') %>%
  mutate(prop = nGames / total_games) %>%
  ungroup() %>%
  mutate(accuracy = percent(sum((won == pred_won)*nGames) /
sum(nGames)))
```

Logistic Regression (logit)

```
eval
```

```
## # A tibble: 4 x 6
##   won pred_won total_games nGames   prop accuracy
##   <dbl>   <dbl>      <int>   <int>   <dbl>   <chr>
## 1     0     0         666     620  0.931  78%
## 2     0     1         666     46  0.0691  78%
## 3     1     0         291    163  0.560  78%
## 4     1     1         291    128  0.440  78%
```

14 / 28

Logistic Regression (logit)

- Can also calculate ROC Curve and AUC

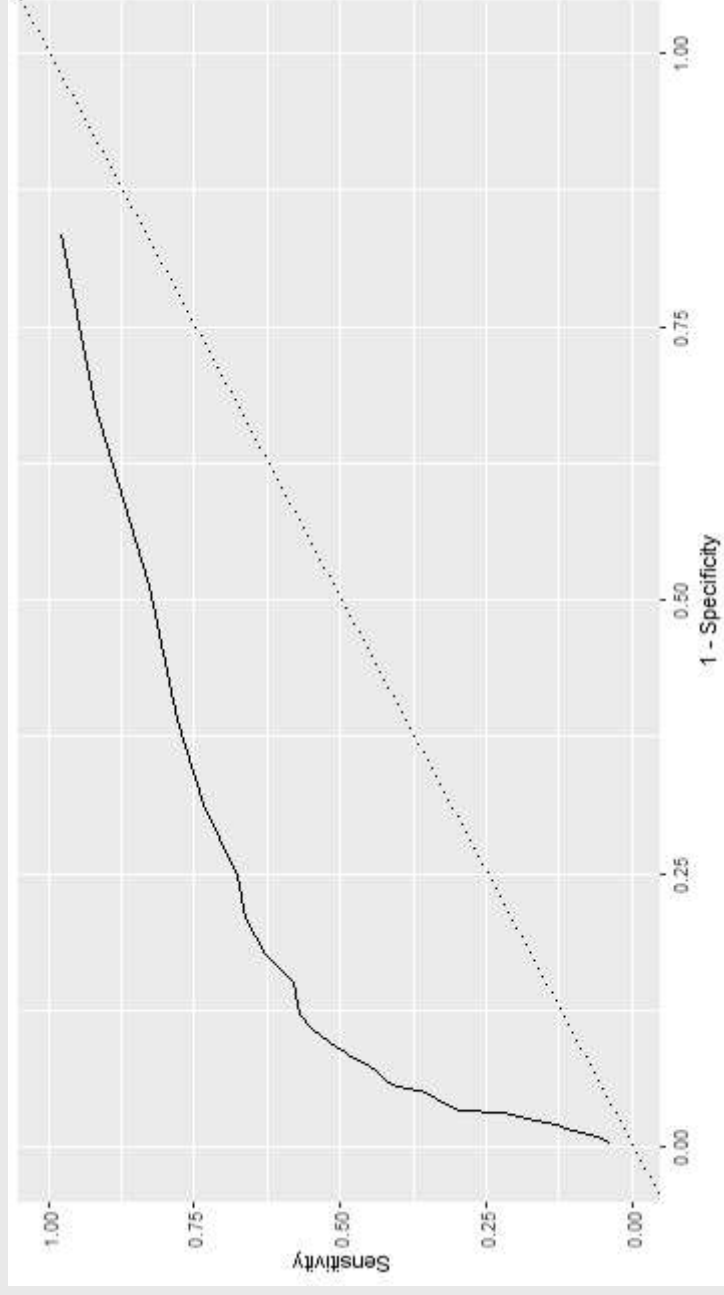
```
toplot <- NULL
for(thresh in seq(0,1,by = .025)) {
  toplot <- fn %>%
    mutate(pred_won = ifelse(predict(mLogit,type = 'response') >
      thresh,1,0)) %>%
    group_by(won) %>%
    mutate(total_games = n()) %>%
    group_by(won,pred_won,total_games) %>%
    summarise(nGames=n(),.groups = 'drop') %>%
    mutate(prop = nGames / total_games) %>%
    ungroup() %>%
    mutate(threshold = thresh) %>%
    bind_rows(toplot)
}
```

Logistic Regression (logit)

```
p <- topplot %>%
  mutate(metric = ifelse(won == 1 & pred_won == 1, 'Sensitivity',
    ifelse(won == 0 & pred_won ==
0, 'Specificity', NA))) %>%
  drop_na(metric) %>%
  select(prop, metric, threshold) %>%
  spread(metric, prop) %>%
  arrange(desc(Specificity), Sensitivity) %>%
  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')
```


Logistic Regression (logit)

p



Logistic Regression (logit)

```
require(tidymodels)
roc_auc(data = fn %>%
  mutate(prob_won = predict(mLogit, type = 'response'),
         truth = factor(won, levels = c('1', '0'))) %>%
  select(truth, prob_won), truth, prob_won)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>      <dbl>
## 1 roc_auc binary      0.782
```

Comparing Models

- Two big questions in prediction:
 1. Do I have the correct predictors X ?
 2. Do I have the best model?
- Two types of outcomes (thus far)
 1. Continuous Y : use **RMSE**
 2. Binary Y : use **AUC**
- Let's determine the best model from the following:
 - X : (1) distance_traveled + mental_state vs. (2) distance_traveled + mental_state + hits
 - Model: (1) conditional means vs. (2) `lm` vs. (3) `glm`

Comparing Models

- Conditional means - simple X

```
results <- NULL

# Train & Predict
toEval <- fn %>%
  mutate(distDec = ntile(distance_traveled, n = 10)) %>%
  group_by(distDec, mental_state) %>%
  mutate(prob_won = mean(won),
         truth = factor(won, levels = c('1', '0'))) %>%
  ungroup() %>%
  select(truth, prob_won)

# Evaluate
results <- roc_auc(data = toEval, truth, prob_won) %>%
  mutate(model = 'CM',
         predictors = 'Simple') %>%
  bind_rows(results)
```

Comparing Models

- Conditional means - complex X

```
# Train & Predict
toEval <- fn %>%
  mutate(distDec = ntile(distance_traveled,n = 10),
         hitsDec = ntile(hits,n = 10)) %>%
  group_by(distDec,hitsDec,mental_state) %>%
  mutate(prob_won = mean(won),
         truth = factor(won,levels = c('1','0'))) %>%
  ungroup() %>%
  select(truth,prob_won)

# Evaluate
results <- roc_auc(data = toEval,truth,prob_won) %>%
  mutate(model = 'CM',
         predictors = 'Complex') %>%
  bind_rows(results)
```

Comparing Models

- Linear regression (lm) - simple X

```
# Train
m <- lm(won ~ distance_traveled + mental_state, fn)

# Predict
toEval <- fn %>%
  mutate(prob_won = predict(m),
         truth = factor(won, levels = c('1', '0'))) %>%
  ungroup() %>%
  select(truth, prob_won)

# Evaluate
results <- roc_auc(data = toEval, truth, prob_won) %>%
  mutate(model = 'LM',
         predictors = 'Simple') %>%
  bind_rows(results)
```

Comparing Models

- Linear regression (lm) - complex X

```
# Train
m <- lm(won ~ distance_traveled + mental_state + hits, fn)

# Predict
toEval <- fn %>%
  mutate(prob_won = predict(m),
         truth = factor(won, levels = c('1', '0'))) %>%
  ungroup() %>%
  select(truth, prob_won)

# Evaluate
results <- roc_auc(data = toEval, truth, prob_won) %>%
  mutate(model = 'LM',
         predictors = 'Complex') %>%
  bind_rows(results)
```

Comparing Models

- Logit regression (glm) - simple X

```
# Train
m <- glm(won ~ distance_traveled + mental_state, fn, family =
binomial(link = 'logit'))

# Predict
toEval <- fn %>%
  mutate(prob_won = predict(m, type = 'response'),
         truth = factor(won, levels = c('1', '0'))) %>%
  ungroup() %>%
  select(truth, prob_won)

# Evaluate
results <- roc_auc(data = toEval, truth, prob_won) %>%
  mutate(model = 'GLM',
         predictors = 'Simple') %>%
  bind_rows(results)
```


Comparing Models

- Logit regression (glm) - complex X

```
# Train
m <- glm(won ~ distance_traveled + mental_state + hits, fn, family =
binomial(link = 'logit'))

# Predict
toEval <- fn %>%
  mutate(prob_won = predict(m, type = 'response'),
         truth = factor(won, levels = c('1', '0'))) %>%
  ungroup() %>%
  select(truth, prob_won)

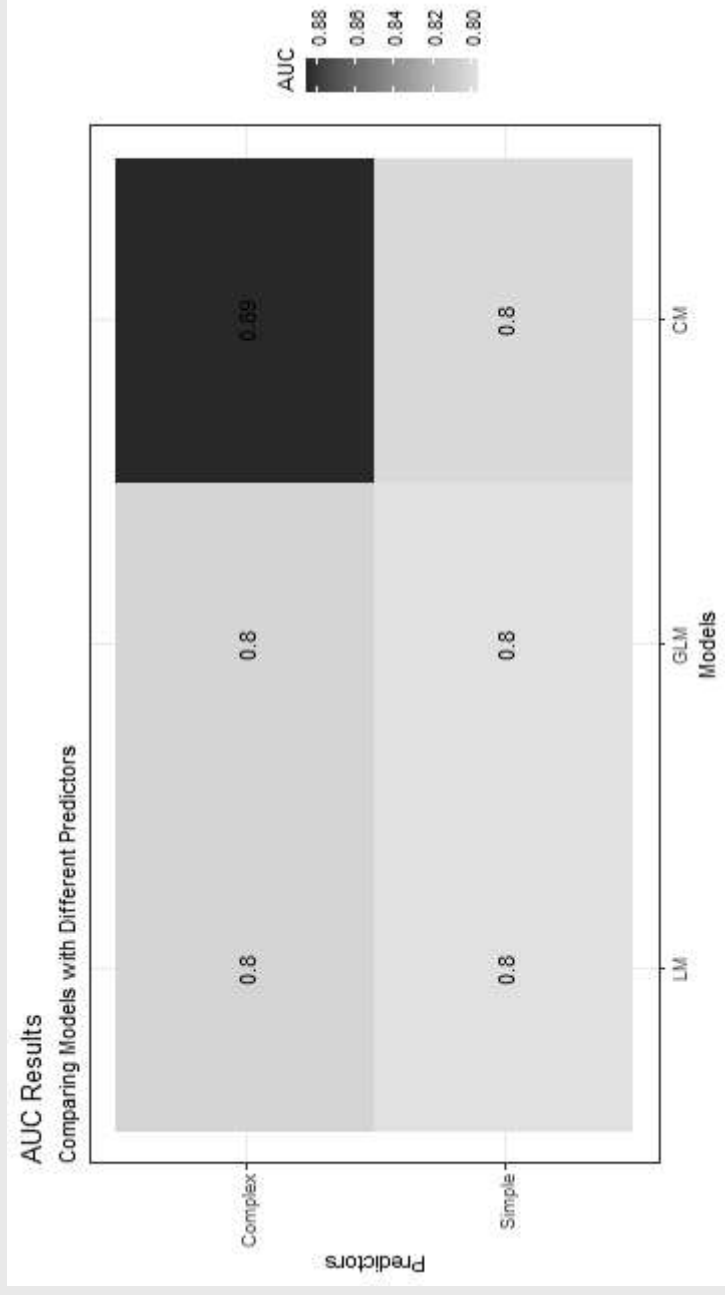
# Evaluate
results <- roc_auc(data = toEval, truth, prob_won) %>%
  mutate(model = 'GLM',
         predictors = 'Complex') %>%
  bind_rows(results)
```

Comparing Models

```
p <- results %>%  
  ggplot(aes(x = reorder(model,.estimate),  
             y = reorder(predictors,.estimate),  
             fill = .estimate,label = round(.estimate,2))) +  
  geom_tile() +  
  scale_fill_continuous(low = 'grey90',high = 'darkred') +  
  geom_text() +  
  labs(title = 'AUC Results',  
        subtitle = 'Comparing Models with Different Predictors',  
        x = 'Models',y = 'Predictors',  
        fill = 'AUC') +  
  theme_bw()
```

Comparing Models

p



Conclusion

- Conditional means outperform regression models?
 - Yes: conditional means allow for cell-specific predictions
 - No: conditional means are more susceptible to **overfitting**
- How would you re-evaluate these models-X-predictors to account for overfitting?
- Go to Brightspace and take the **15th** quiz
- **Homework:**
 - Problem Set 8 (due 2024-03-22 by 11:59PM)
 - HW 16