PSET 05 - Poisson and Logistic Regression

S&DS 361

Due 2024-03-28

Shot success in hockey

In this part, we'll study NHL shots data. In particular, we'll examine what kinds of shot attempts are more likely to be goals.

```
d = readRDS('data/nhl.shots.rds')
head(d,2)
```

```
##   goal shot shot.att shot.att.with.blocks event dist dist10 angle abs.angle
## 1    0    0        0                    1 BLOCK   41     NA   -31        31
## 2    0    1        1                    1  SHOT   59      6    27        27
##   ang10 abs.ang10 xcoord ycoord house zone1 shot.type prev.event home
## 1  -3.1       3.1     54    -21     1   off                  FAC    0
## 2   2.7       2.7     37     27     0   off     wrist      SHIFT    1
##   score.diff            goalie glove        shooter pos hand str empty reb
## 1          0       Steve Mason     R    Nazem Kadri   C    L 5v5     0   0
## 2          0 Frederik Andersen    76 Josh Morrissey   D    L 5v5     0   0
##   own.reb angle.diff angle.diff10 angle.diff.left angle.diff.right team1 team2
## 1       0         NA           NA              NA               NA   TOR   WPG
## 2       0         NA           NA              NA               NA   WPG
##   hometeam awayteam year  game       date row.id time
## 1      WPG      TOR 2017 20001 2017-10-04      6   12
## 2      WPG      TOR 2017 20001 2017-10-04      9   38
```

Let's focus only on 5-on-5 situations where both goalies are on the ice, let's remove a few shots missing location information, and let's remove blocked shots.

```
d = d %>%
  filter(str == '5v5' &  ## 5-on-5 only
           !is.na(xcoord) &  ## remove shots with missing coordinates
           event != 'BLOCK')   ## remove blocks
```

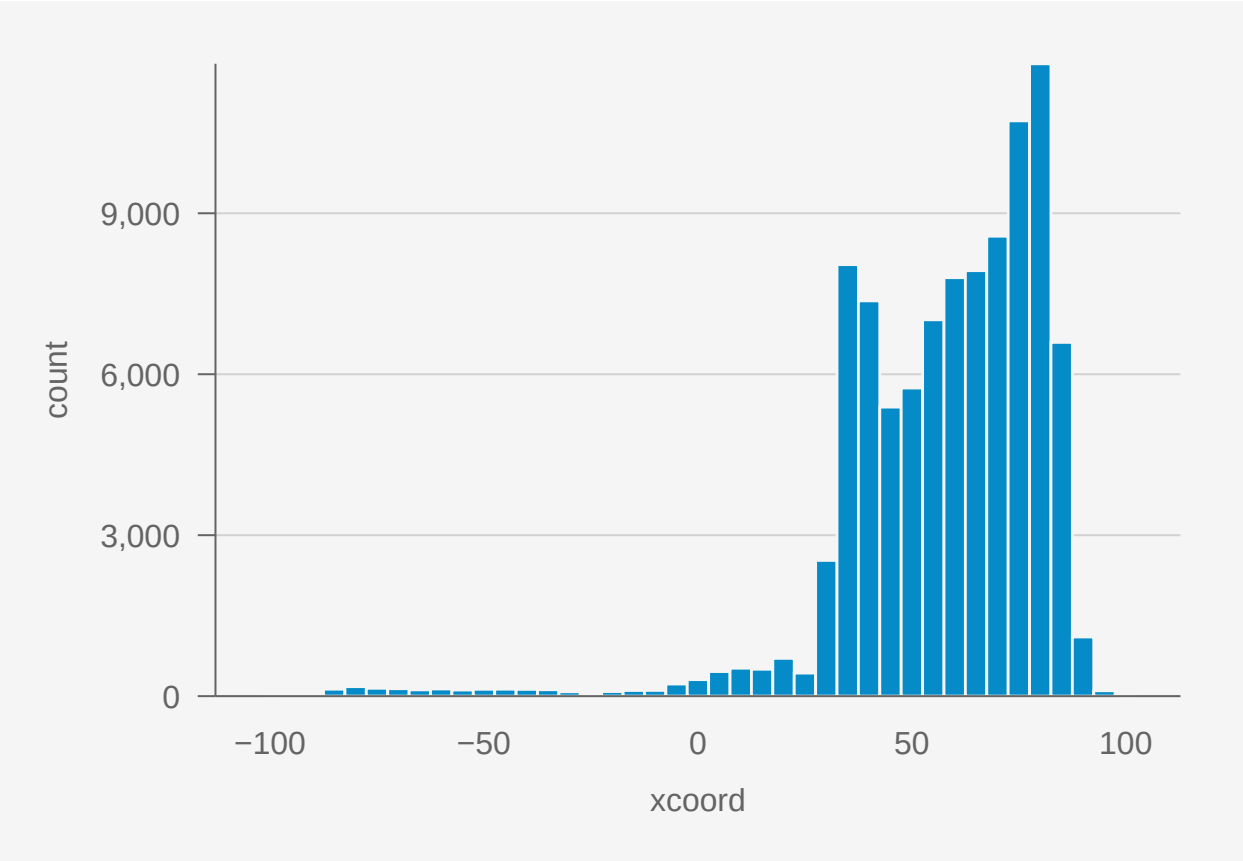
We'll focus on shot location (`distance` and `angle`) as predictors, and `goal` as the outcome:

- `goal` = 1 if the shot attempt was a goal, 0 if it missed the net or was saved by the goalie.
- `dist` = the distance of the shot attempt from the net
- `angle` = the angle from the center of the ice that the shot was taken from (0 is center of the ice, -90 is left side along the red line, 90 is right side along the red line)
- `abs.angle` = the absolute value of the `angle`

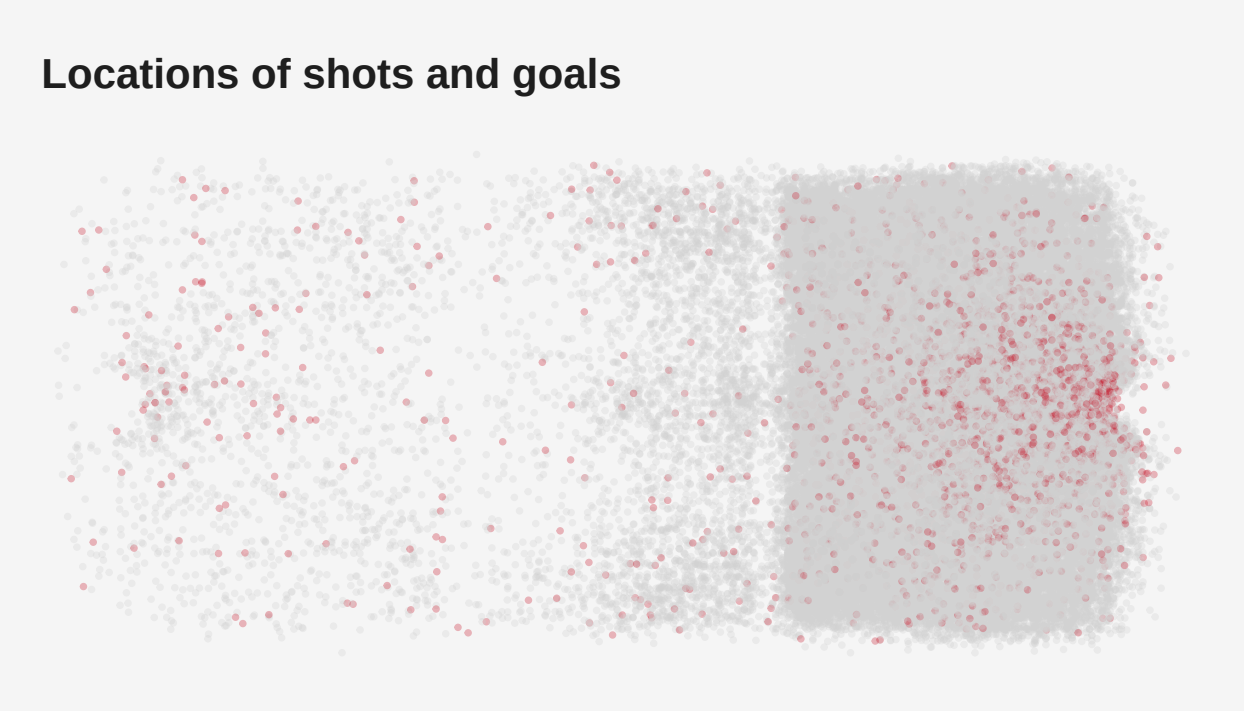
We first make a plot of the data, getting shot locations from `xcoord` and `ycoord`. We color the dots by `goal`.

Note that for each period, the coordinates for one team are flipped so that both teams are shooting in the same direction (to the right). So we'll see far more shots on the right side of the rink.

```r
g = ggplot(d,
           aes(x = xcoord)) +
  geom_histogram(color = pubbackgray,
                 binwidth = 5)

g %>%
  pub(type = 'hist')
```



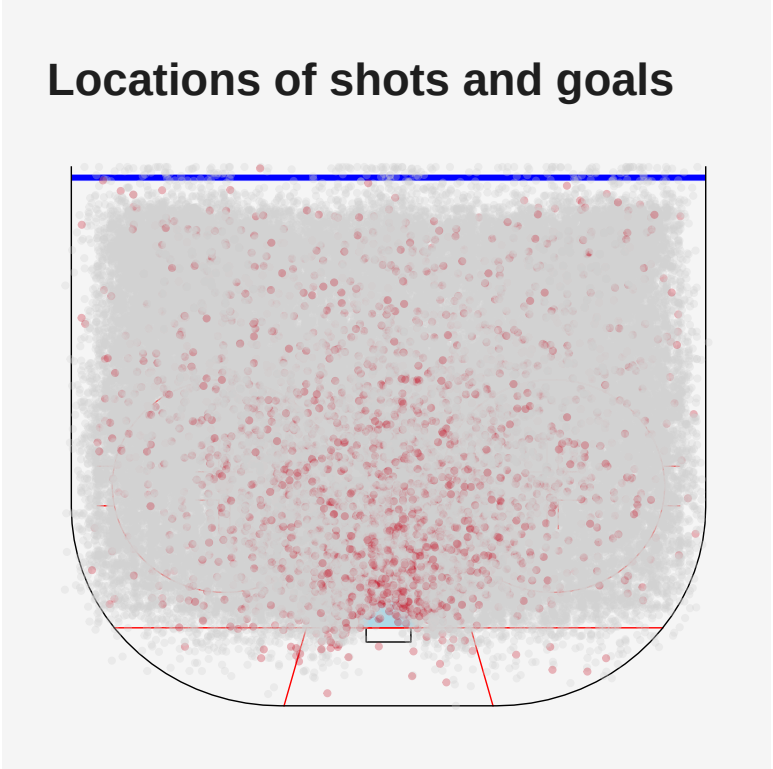

```r
d = d %>%
  mutate(Goal = as.character(goal))

title = "Locations of shots and goals"
g = ggplot(d,
           aes(x = xcoord,
               y = ycoord,
               color = Goal))+
  geom_jitter(alpha = 0.3,
              size  = 0.75,
              show.legend = F)+
  labs(title = title) +
  scale_color_manual(values = c(publightgray, pubred))


g %>%
  pub(type = 'map')
```



The goal is on the right, and we probably don't care about the shots that are coming from outside the offensive zone. So let's remove those.

Also, it would look nicer if this were plotted on a picture of a rink. So first we'll load a picture of the rink and then plot points on top of it. Because of the way the rink is oriented, we'll have to flip-flop our x and y coordinates and let `x = ycoord` and `y = -xcoord` (note the minus sign). You'll need to download the script `half.rink.r` from Canvas.

```
d = d %>%
  filter(xcoord >= 24,
         xcoord <= 100)

source('R/half.rink.r')

g = half.rink +
  geom_jitter(data = d,
              aes(x =  ycoord,
                  y = -xcoord,
                  color = Goal),
              alpha = 0.3,
              size  = 0.75,
              show.legend = F) +
  labs(title = title) +
  scale_color_manual(values = c(publightgray, pubred))

g %>%
  pub(type = 'map',
      ylim = c(-100, -24))
```

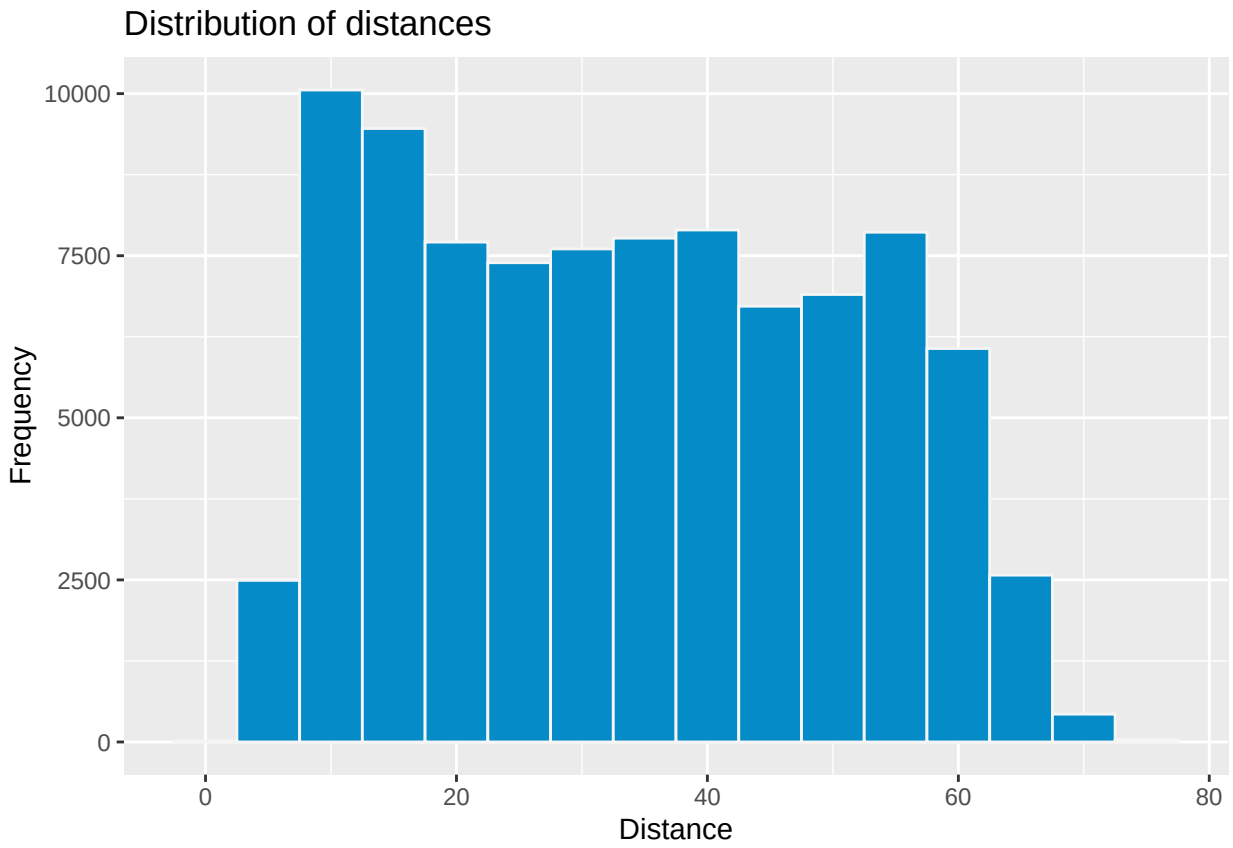


We see mostly gray dots (shot attempts that weren't goals), but there are red dots (goals) too, and there are more of them closer to the goal (smaller values of `dist`) and very few far away from the goal (higher values of `dist`, near the blue line). (The goal is the black rectangle at the bottom, below the blue half circle.) There are also more red dots in the center of the ice (`angle` near 0) and fewer on the left and right sides (angles near -90 and 90, or `abs.angle` near 90). This suggests `dist`, `angle`, `abs.angle` are related to the probability that a shot attempt will be a `goal`. The data are not very well-separated, so we don't expect our error rates to be that low. But hopefully we can build a useful model.

1. Observed proportion of goals by `dist` Find the observed proportion of goals for different subsets of `dist` and plot the result. Do these values appear to follow a logistic curve? Does the curve you found make sense given the overall proportion of goals in the data?

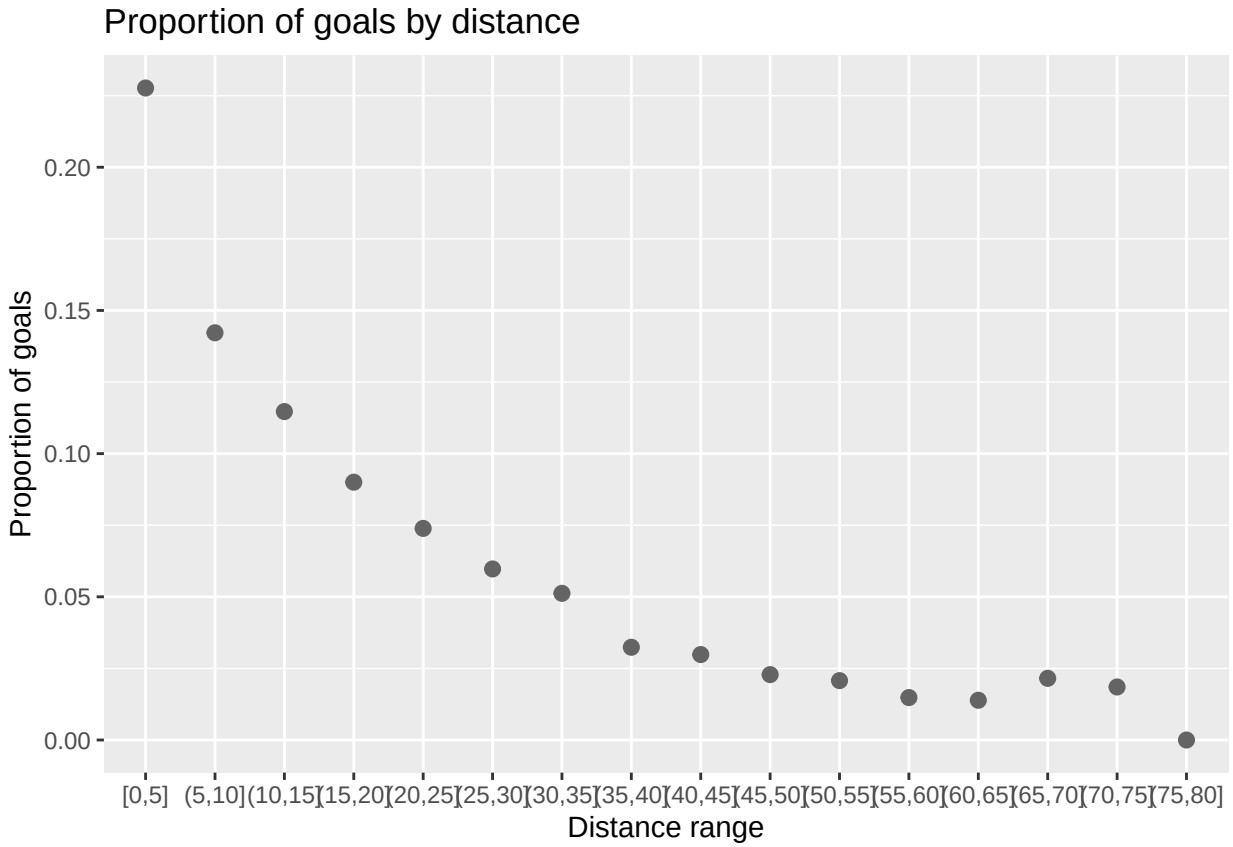
I might want to see the distribution of the distances first to make a decision on how will I group the distances.

```
## first let us find out the distribution of the distances
d %>%
  ggplot(aes(x = dist)) +
  geom_histogram(binwidth = 5,
                 color = pubbackgray) +
  labs(title = "Distribution of distances",
       x = "Distance",
       y = "Frequency")
```



Seems that binwidth = 5 is a good choice. I might just stick to this binwidth and try to plot the proportion of goals for each bin.

```
d = d %>%
  mutate(dist.group = cut(dist,
                          breaks = seq(0, 100, 5),
                          include.lowest = T))

d %>%
  group_by(dist.group) %>%
  summarise(goals = sum(goal),
            attempts = n()) %>%
  mutate(prop = goals/attempts) %>%
  ggplot(aes(x = dist.group,
             y = prop)) +
  geom_point() +
  geom_line() +
  labs(title = "Proportion of goals by distance",
       x = "Distance range",
       y = "Proportion of goals")
```

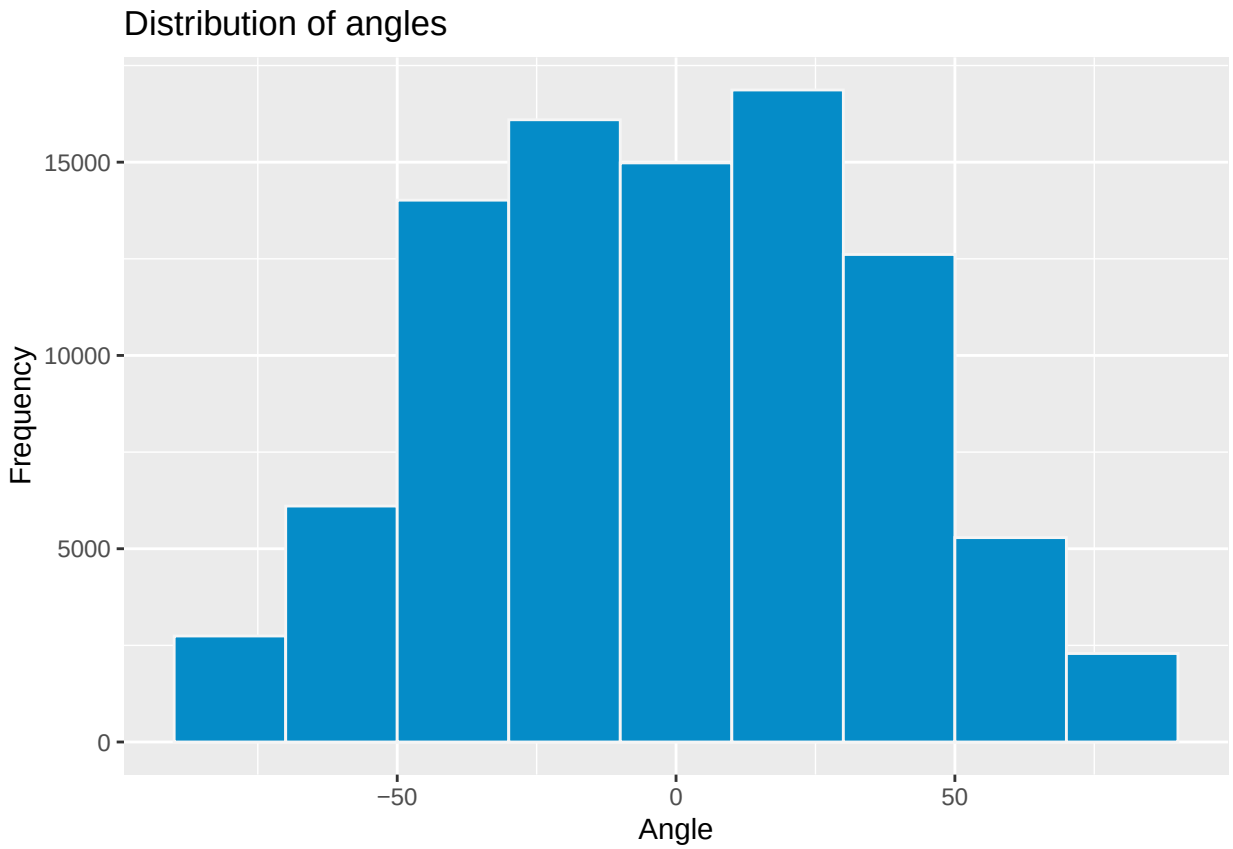


The curve does not seem to follow a rigorous logistic S-curve, but it does have the trend of being an S-curve. So I'd say that we should be at least somewhat reasonable to use logistic regression here. This plot makes sense in terms of the data because the proportion of goals is higher for shots that are closer to the goal, and lower for shots that are farther away.

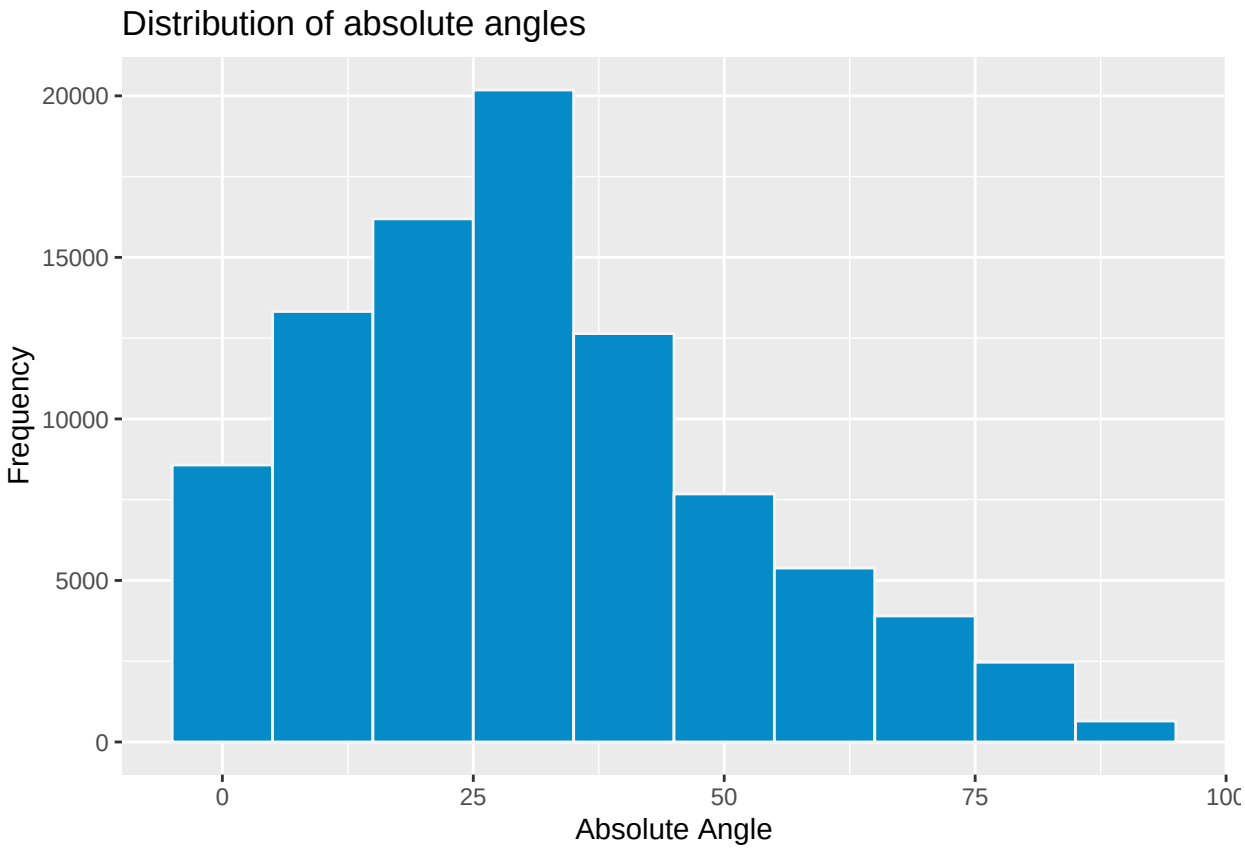
2. Observed proportion of goals by `angle` and `abs.angle`. Repeat the above question for `angle` and `abs.angle`. Which one should you try in the model? Why?

I think we should still begin with a exploration on the distribution of `angle` and `abs.angle`.

```
d %>%
  ggplot(aes(x = angle)) +
  geom_histogram(binwidth = 20,
                 color = pubbackgray) +
  labs(title = "Distribution of angles",
       x = "Angle",
       y = "Frequency")
```

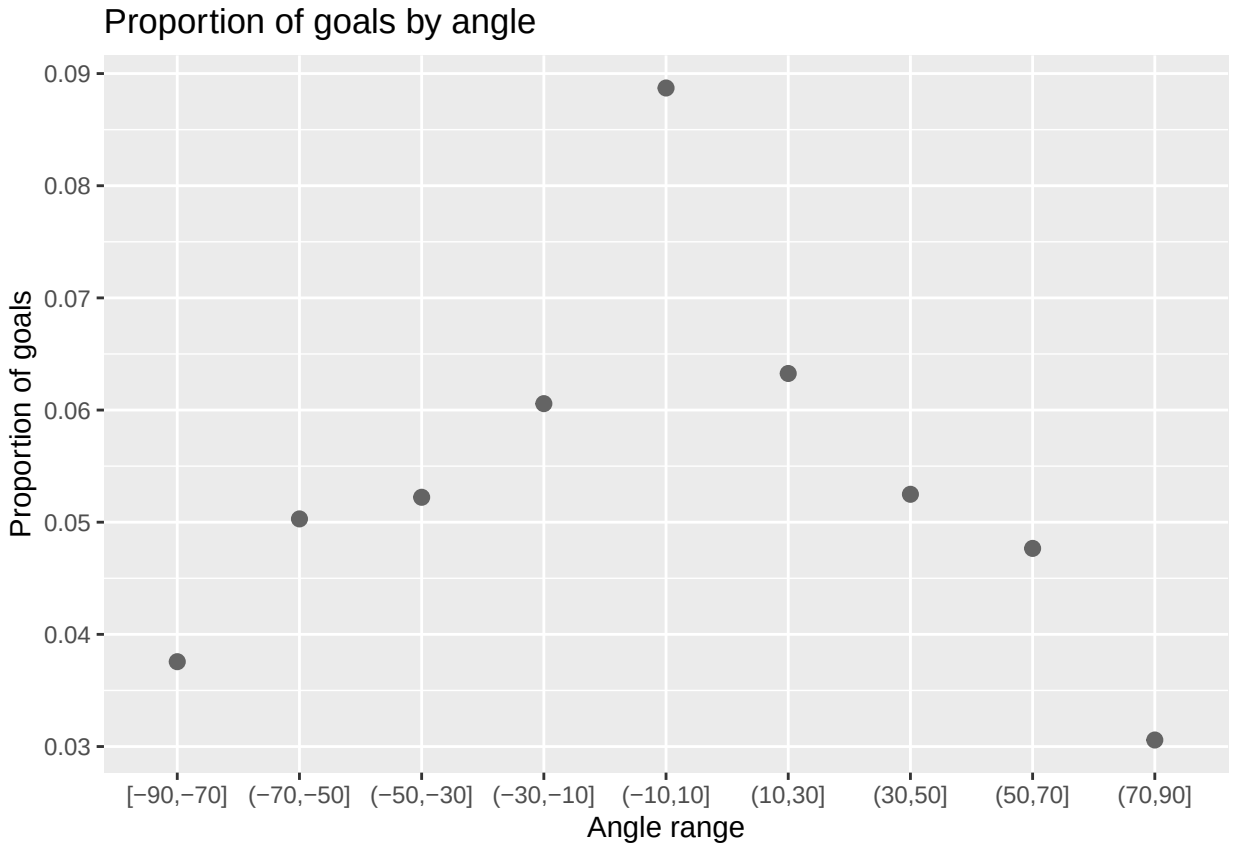



```
d %>%
  ggplot(aes(x = abs.angle)) +
  geom_histogram(binwidth = 10,
                 color = pubbackgray) +
  labs(title = "Distribution of absolute angles",
       x = "Absolute Angle",
       y = "Frequency")
```

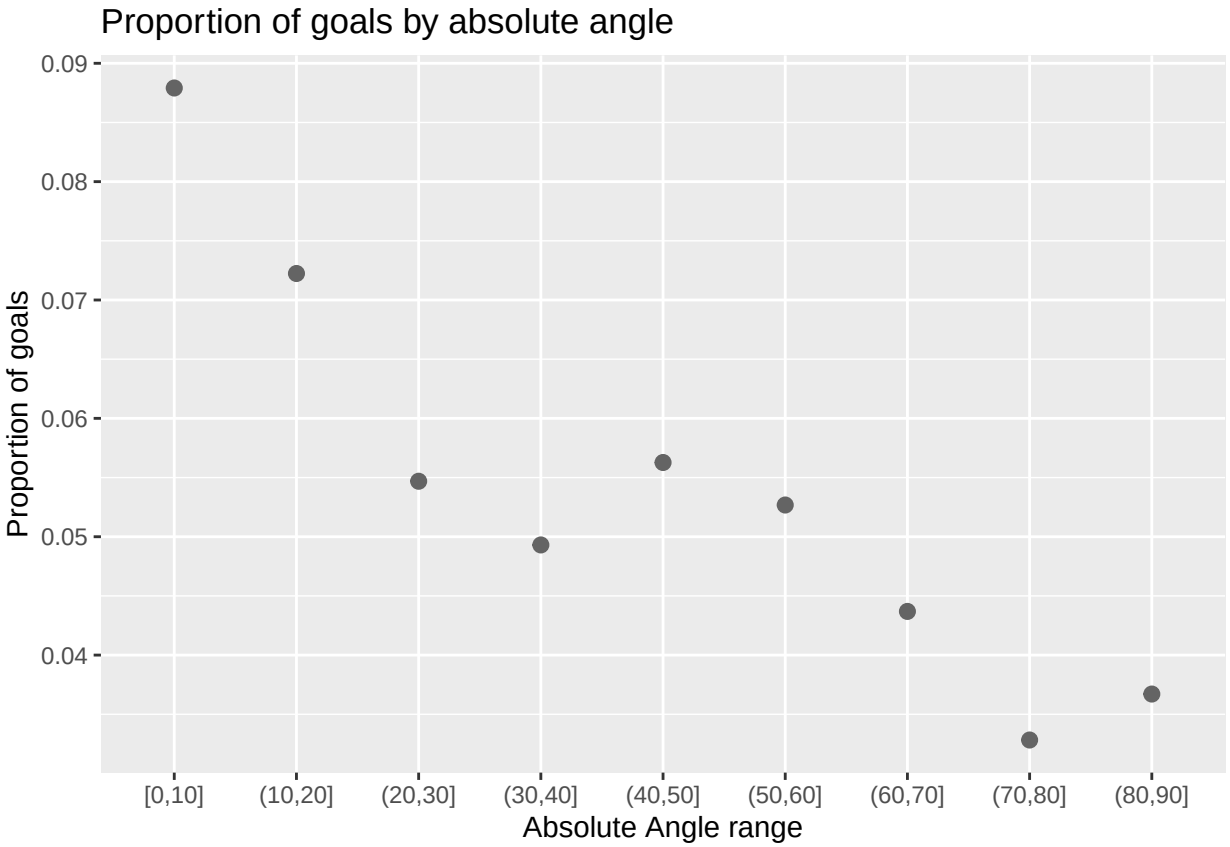


Now I am about to use binwidth = 20 for `angle` and binwidth = 10 for `abs.angle`, this is because naturally `abs.angle` should be 2 times tighter than `angle`. Mathematically this makes sense.

```
d = d %>%
  mutate(angle.group = cut(angle,
                           breaks = seq(-90, 90, 20),
                           include.lowest = T),
         abs.angle.group = cut(abs.angle,
                               breaks = seq(0, 90, 10),
                               include.lowest = T))


d %>%
  group_by(angle.group) %>%
  summarise(goals = sum(goal),
            attempts = n()) %>%
  mutate(prop = goals/attempts) %>%
  ggplot(aes(x = angle.group,
             y = prop)) +
  geom_point() +
  geom_line() +
  labs(title = "Proportion of goals by angle",
       x = "Angle range",
       y = "Proportion of goals")
```




```
d %>%
  group_by(abs.angle.group) %>%
  summarise(goals = sum(goal),
            attempts = n()) %>%
  mutate(prop = goals/attempts) %>%
  ggplot(aes(x = abs.angle.group,
             y = prop)) +
  geom_point() +
  geom_line() +
  labs(title = "Proportion of goals by absolute angle",
       x = "Absolute Angle range",
       y = "Proportion of goals")
```



We could tell from the first `angle` plot, there is not a huge impact on goal rate (or proportion of goals in the plots) on whether the player attacks from the left or right side. However, the second plot shows that the proportion of goals is higher for shots that are taken from the center of the ice (`abs.angle` near 0), and lower for shots that are taken from the sides of the ice (`abs.angle` near 90). Moreover, the second plot also roughly looks like a S-curve, even this is not a sharp match. So I would choose `abs.angle` as a predictor in the model.

3. Build three logistic regression models that predict `goal` using the following predictors. Which model is "best"? Why?

- Model 1: `dist`
- Model 2: `abs.angle`
- Model 3: `dist` and `abs.angle`

```
## Model 1

model1 = glm(goal ~ dist,
             data = d,
             family = binomial)

summary(model1)
```

```
##
## Call:
## glm(formula = goal ~ dist, family = binomial, data = d)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept) -1.4148550  0.0267904  -52.81   <2e-16 ***
## dist        -0.0478062  0.0009994  -47.83   <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: 41511  on 90995  degrees of freedom
## Residual deviance: 38728  on 90994  degrees of freedom
## AIC: 38732
##
## Number of Fisher Scoring iterations: 6
```

```
## Model 2

model2 = glm(goal ~ abs.angle,
             data = d,
             family = binomial)

summary(model2)
```

```
##
## Call:
## glm(formula = goal ~ abs.angle, family = binomial, data = d)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.3808595  0.0244433  -97.40   <2e-16 ***
## abs.angle   -0.0124433  0.0007398  -16.82   <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: 41511  on 90995  degrees of freedom
## Residual deviance: 41212  on 90994  degrees of freedom
## AIC: 41216
##
## Number of Fisher Scoring iterations: 5
```

```
## Model 3

model3 = glm(goal ~ dist + abs.angle,
             data = d,
             family = binomial)

summary(model3)
```

```
##
## Call:
## glm(formula = goal ~ dist + abs.angle, family = binomial, data = d)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.9368424  0.0335283  -27.94   <2e-16 ***
## dist        -0.0492659  0.0010008  -49.23   <2e-16 ***
```

```
## abs.angle   -0.0147557  0.0006898  -21.39   <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: 41511  on 90995  degrees of freedom
## Residual deviance: 38240  on 90993  degrees of freedom
## AIC: 38246
##
## Number of Fisher Scoring iterations: 6
```

From the summary of the three models, I prefer using the third model (both `dist` and `abs.angle` as predictors) because it has the lowest AIC value. This means that the third model is the best model among the three. Moreover, the third model has the most significant drop on the Null deviance to the Residual deviance, which means that the third model is the most effective model in explaining the data.

```r
## Assessing the accuracy and error rates of the models
library(Metrics)
## Model 1
pred1 = predict(model1,
                newdata = d,
                type = 'response')

goal1 = ifelse(pred1 > 0.5, 1, 0)

accuracy1 = mean(goal1 == d$goal)

error_rate1 = 1 - accuracy1

## Model 2

pred2 = predict(model2,
                newdata = d,
                type = 'response')

goal2 = ifelse(pred2 > 0.5, 1, 0)

accuracy2 = mean(goal2 == d$goal)

error_rate2 = 1 - accuracy2

## Model 3

pred3 = predict(model3,
                newdata = d,
                type = 'response')

goal3 = ifelse(pred3 > 0.5, 1, 0)

accuracy3 = mean(goal3 == d$goal)
```

```
error_rate3 = 1 - accuracy3

## AUC and log loss

roc1 = roc(d$goal, pred1)

roc2 = roc(d$goal, pred2)

roc3 = roc(d$goal, pred3)



log_loss1 = logLoss(d$goal, pred1)

log_loss2 = logLoss(d$goal, pred2)

log_loss3 = logLoss(d$goal, pred3)

cat("The accuracy of model 1 is:", accuracy1, "; the error rate of model 1 is:", error_rate1, "The AUC
```

4. Find the accuracy and error rate of each model, meaning, how often it correctly or incorrectly predicts the outcome. Also compute the AUC and log loss for all three models. What do you notice? Which model would you use to predict the probability that a new shot will be a goal?

```
## The accuracy of model 1 is: 0.9395908 ; the error rate of model 1 is: 0.06040925 The AUC of model 1
```

```
cat("The accuracy of model 2 is:", accuracy2, "; the error rate of model 2 is:", error_rate2, "The AUC
```

```
## The accuracy of model 2 is: 0.9395908 ; the error rate of model 2 is: 0.06040925 The AUC of model 2
```

```
cat("The accuracy of model 3 is:", accuracy3, "; the error rate of model 3 is:", error_rate3, "The AUC
```

```
## The accuracy of model 3 is: 0.9395908 ; the error rate of model 3 is: 0.06040925 The AUC of model 3
```

Surprisingly, the accuracy and error rates of the three models are the same. However, the third model achieves the greatest AUC value among the three models. The log loss of the third model is also the lowest among the three models. This means that the third model is the best model to predict the probability that a new shot will be a goal.

5. How are the accuracy, error rates, and proportion of 1's related? Why are they related that way? How could you change your method of determining the predicted classes and obtain a different result? Let's find out the proportion of 1's in the data.

```
proportion_1 = mean(d$goal)
print(proportion_1)
```

```
## [1] 0.06040925
```

Now things are clear. The accuracy and error rates are negatively related because mathematically accuracy = 1 - error rate. The proportion of 1's in the data exactly equals to the error rate of all three models. This means all models are just predicting 0s for all shot attempts. Tracing back to problem 1 and 2 where we analyzed the data. Distance-wise, the highest goal rate zone is [0,5) with approximately 0.2 as the goal rate. Angle-wise, the highest goal rate zone is [0,10), with approximately 0.1 as the goal rate. This is a very low goal rate, so the models are just predicting 0s for all shot attempts. To obtain different results, we need to take a threshold of approximately 0.05-0.15 that fits the actual goal rate.

EV Stations

The following data set is census and electric vehicle charging stations data set from earlier in the course. It contains one row per census tract (`GEOID`), several columns with census information along with columns that have the number of Level 2 (`lev2`) and Level 3 (`lev3`) electric vehicle charging stations for each census tract.

```
ev = readRDS('data/tracts.and.census.with.EV.stations.rds')
ev = ev@data
head(ev,2)
```

```
##   STATEFP COUNTYFP TRACTCE            AFFGEOID       GEOID    NAME LSAD
## 1      01      047  956202 1400000US01047956202 01047956202 9562.02   CT
## 2      01      073  003001 1400000US01073003001 01073003001   30.01   CT
##       ALAND  AWATER    meters      miles state   tract           county
## 1 182237800 1051253 183289053 70.7682990    AL 9562.02    Dallas County
## 2   2440163       0   2440163  0.9421522    AL   30.01 Jefferson County
##   state.full  pop male female  age male.age female.age white black
## 1    Alabama 1987  898   1089 42.9     40.0       45.1  1483   479
## 2    Alabama 3481 1488   1993 21.9     21.9       22.0  1089  2144
##   indian.alaskan asian pacific other two.or.more white.not.hisp hisp white.hisp
## 1              5     0       0     0          20           1461   22         22
## 2              0    27       0   180          41           1042  229         47
##   black.hisp households i10orless i10to14 i15to19 i20to24 i25to29 i30to34
## 1          0        851        62      42      62      43      46      33
## 2          6        905       114     164     152      72       0      55
##   i35to39 i40to44 i45to49 i50to59 i60to74 i75to99 i100to124 i125to149 i150to199
## 1      55      29      36      68     109     125        88         7        46
## 2      26      18       0     110      77      22         8        15        52
##   i200ormore hh.income house.value   male.p female.p  white.p  black.p
## 1          0     51458      113800 45.19376 54.80624 74.63513 24.10669
## 2         20     22525       73200 42.74634 57.25366 31.28411 61.59150
##     asian.p   hisp.p white.not.hisp.p white.hisp.p black.hisp.p  other.p
## 1 0.0000000 1.107197         73.52793     1.107197    0.0000000 1.258178
## 2 0.7756392 6.578569         29.93393     1.350187    0.1723643 6.348750
##   rescaled.house.value hh.income.and.house tot.hh.income tot.house.value
## 1             45477.94            48467.97      43790758        96843800
## 2             39975.12            31250.06      20385125        66246000
##   tot.hh.income.and.house pop.density hh.density income.density
## 1                41246242    28.07754   12.02516       618790.6
## 2                28281305  3694.73212  960.56667     21636764.2
##   house.value.density house.and.income.density lev2 lev3
## 1             1368463                  582835   NA   NA
## 2            70313480                30017767    2    0
```

6. Data exploration

Perform some data exploration and visualization that helps you learn about the data, with a focus on predicting the probability that a census tract has at least one Level 2 charging station. Discuss any notable observations.

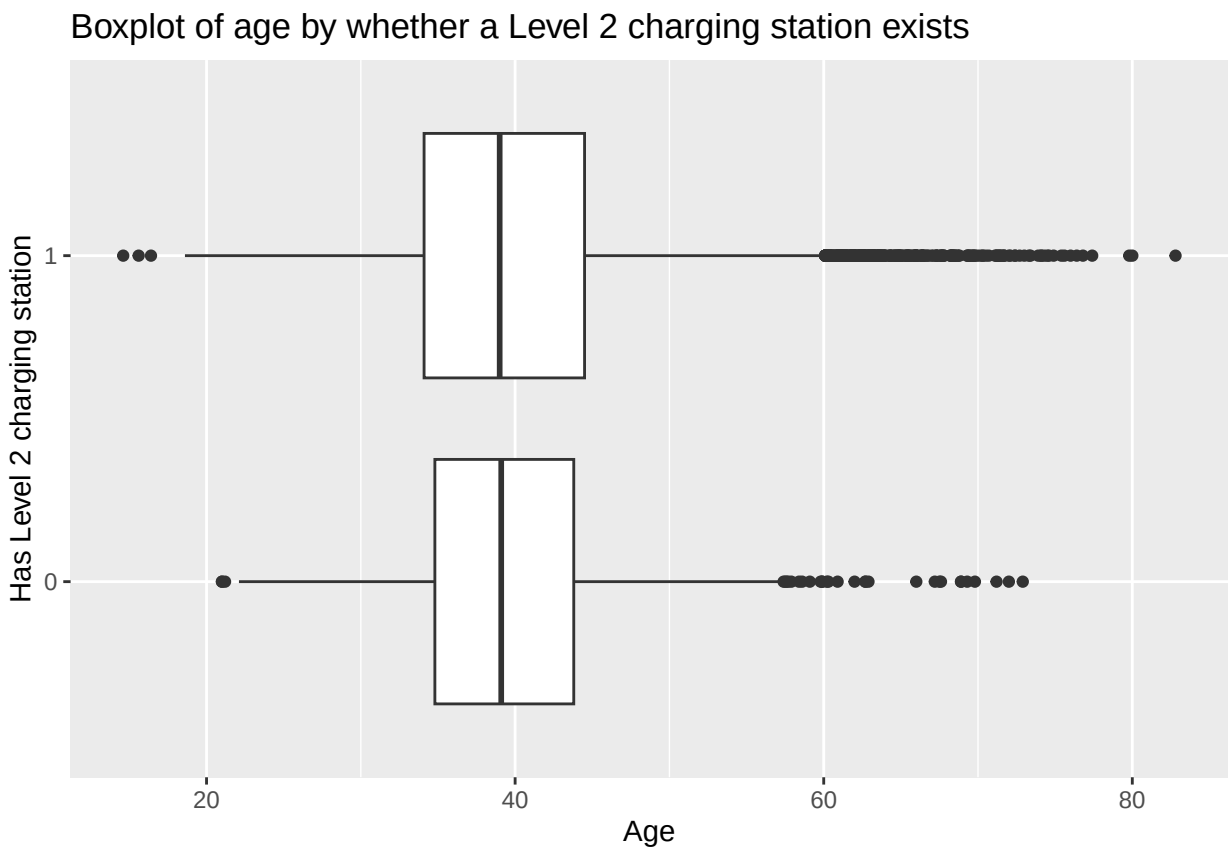
Before we start the data exploration, let's first pre-process the data a little bit.

```
# drop NA and mutate a new column to indicate whether a census tract has at least one Level 2 charging

ev = ev %>%
  drop_na() %>%
```

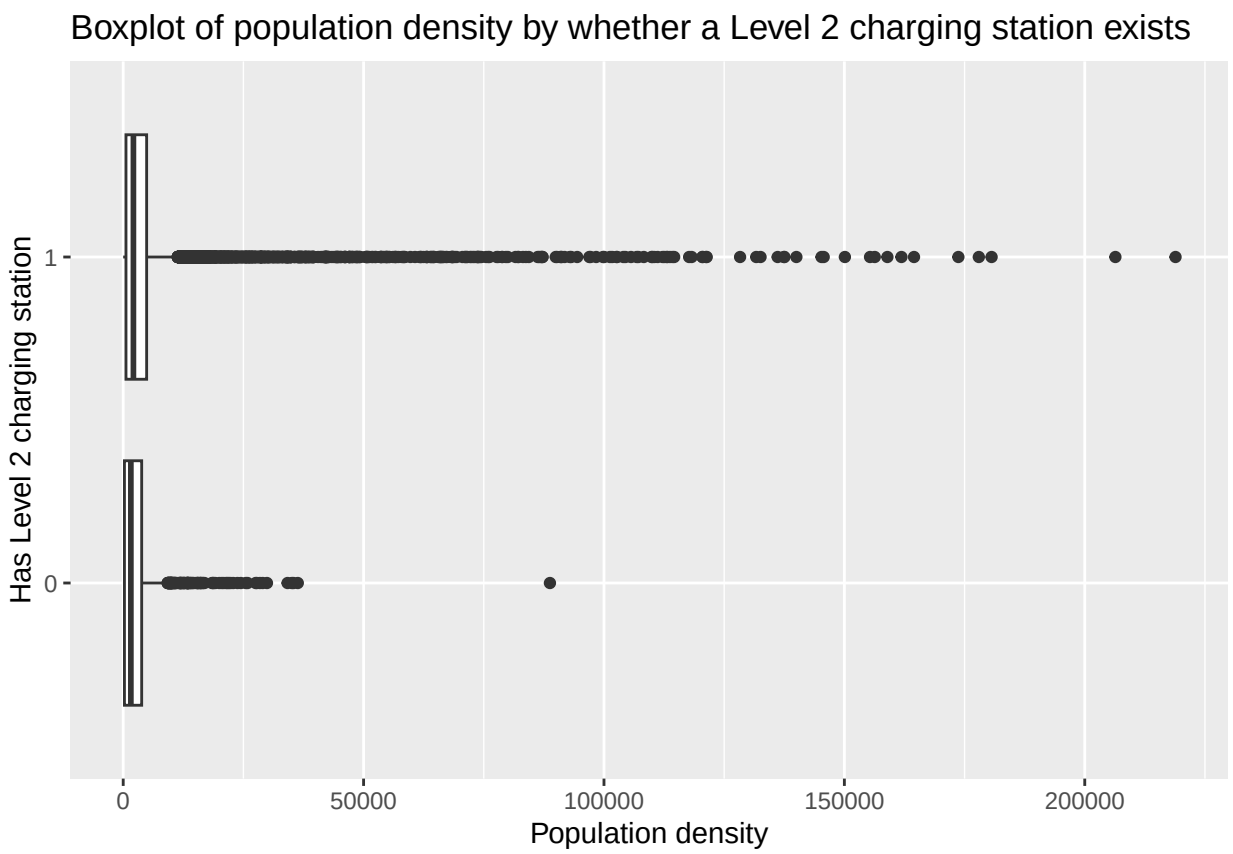
```
  mutate(has_lev2 = ifelse(lev2 > 0, 1, 0)) %>%
  mutate(has_lev2_factor = as.factor(has_lev2))
```

Now let's start the data exploration. What could be related to whether a level2 ev charging station exists or not? I want to focus on 1. the average age (`age`) of the tract; 2. the population density (`pop.density`) of the tract; 3. the rescaled household value ('rescaled.house.value'); 4. the housela income (`hh.income`).

```
ev %>%
  ggplot(aes(x = age,
             y = has_lev2_factor)) +
  geom_boxplot() +
  labs(title = "Boxplot of age by whether a Level 2 charging station exists",
       x = "Age",
       y = "Has Level 2 charging station")
```

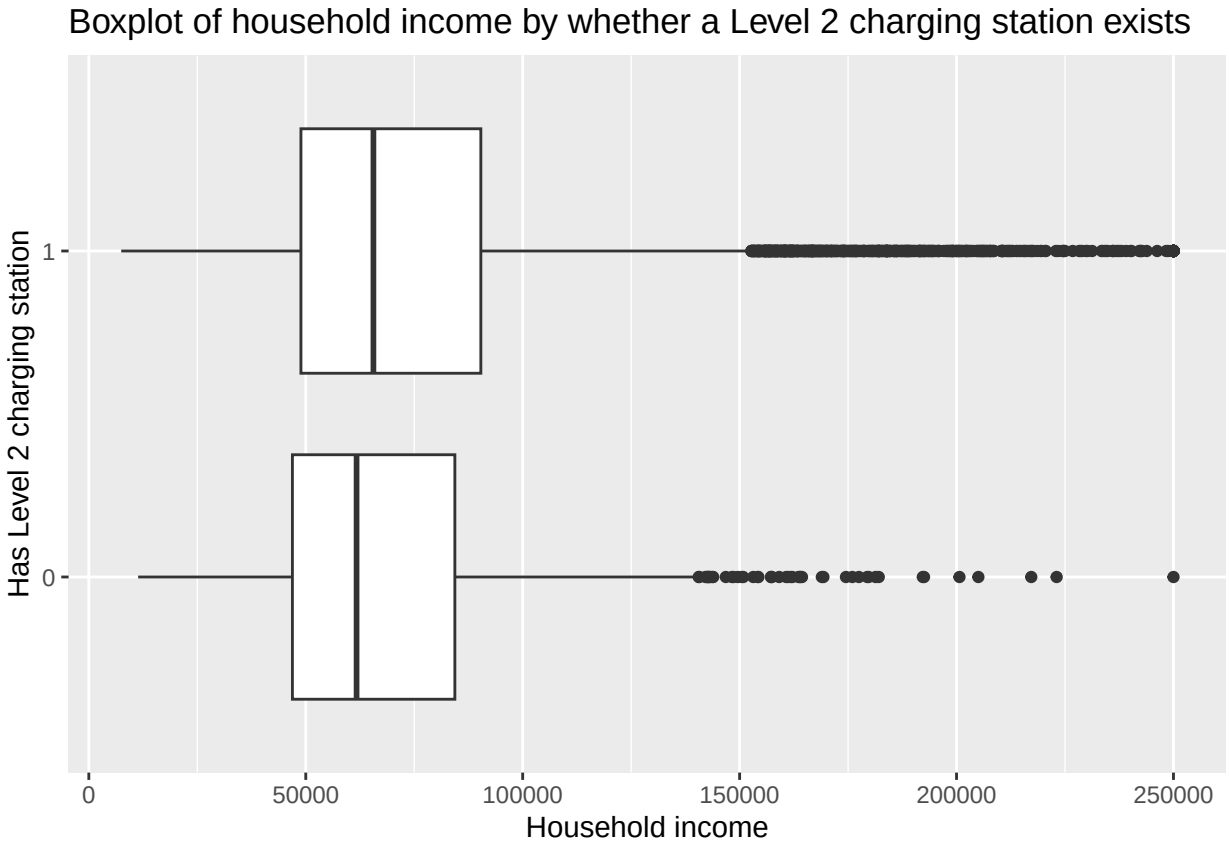



```
ev %>%
  ggplot(aes(x = pop.density,
             y = has_lev2_factor)) +
  geom_boxplot() +
  labs(title = "Boxplot of population density by whether a Level 2 charging station exists",
       x = "Population density",
       y = "Has Level 2 charging station")
```




```
ev %>%
  ggplot(aes(x = rescaled.house.value,
             y = has_lev2_factor)) +
  geom_boxplot() +
  labs(title = "Boxplot of rescaled house value by whether a Level 2 charging station exists",
       x = "Rescaled house value",
       y = "Has Level 2 charging station")
```


```
ev %>%
  ggplot(aes(x = hh.income,
             y = has_lev2_factor)) +
  geom_boxplot() +
  labs(title = "Boxplot of household income by whether a Level 2 charging station exists",
       x = "Household income",
       y = "Has Level 2 charging station")
```



From the above plots, we could tell that the average age of the tract does not seem to have a significant impact on whether a Level 2 charging station exists or not. However, the population density of the tract, the rescaled house value, and the household income seem to have an impact on whether a Level 2 charging station exists or not. The population density of the tract, the rescaled house value, and the household income are higher for tracts that have a Level 2 charging station.

7. Estimate the probability of at least one `lev2` station

Estimate the probability that a census tract has at least one Level 2 charging station. Justify the information you use in the model. Briefly discuss the results and any takeaways.

This is a binary classification problem. I will use logistic regression to estimate the probability that a census tract has at least one Level 2 charging station. I will use the population density of the tract, the rescaled house value, and the household income as predictors in the model.

```
model = glm(has_lev2_factor ~ pop.density + rescaled.house.value + hh.income,
            data = ev,
            family = binomial)


summary(model)
```

```
##
## Call:
## glm(formula = has_lev2_factor ~ pop.density + rescaled.house.value +
##     hh.income, family = binomial, data = ev)
##
## Coefficients:
```

```
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)           1.620e+00  7.414e-02  21.853  < 2e-16 ***
## pop.density           2.101e-05  5.893e-06   3.564 0.000365 ***
## rescaled.house.value  1.227e-05  1.482e-06   8.282  < 2e-16 ***
## hh.income            -3.707e-06  1.192e-06  -3.110 0.001869 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10772  on 17709  degrees of freedom
## Residual deviance: 10583  on 17706  degrees of freedom
## AIC: 10591
##
## Number of Fisher Scoring iterations: 6
```

```
## Assessing the accuracy and error rates of the model

pred = predict(model,
               newdata = ev,
               type = 'response')

accuracy = mean(ifelse(pred > 0.5, 1, 0) == ev$has_lev2_factor)

roc = roc(ev$has_lev2_factor, pred)

cat("The accuracy of the model is:", accuracy, "The AUC of the model is:", roc$auc, "\n")
```

```
## The accuracy of the model is: 0.9093168 The AUC of the model is: 0.6145616
```

```
### the actual probabilities of having at least one Level 2 charging station

actual_prob = mean(ev$has_lev2)

cat("The actual probability of having at least one Level 2 charging station is:", actual_prob, "\n")
```

```
## The actual probability of having at least one Level 2 charging station is: 0.9093168
```

The major takeaway is that we still falls into the problem of having a extreme unbalanced dataset. The actual probability of having at least one Level 2 charging station is 0.9, which is very high. The model accuracy is the same as the actual probability, which means the model tends to predict everything to 1. This means that most the level2 ev charger is viral now in most regions, no matter if the tract has more/less population or the tract has a high/low average income.

Landcover classification

For this part our goal is to use `NDVI` and `B7` to classify each location as `natforest`, `cropland`, `orchard`, or `builtup`. Note that `landcover` is a categorical outcome with 4 categories, which isn't something we focused on in class. But not to worry, we'll be doing some of the work for you.

First, here is some data prep to help you get started.

```
load('data/labeled_points.Rdata')

## Create a data set that is one row per location
## with mean(NDVI), mean(B7), and landcover type for each location
```

```r
labeled = labeled %>%
  dplyr::select(ID, landcover)

d = labeled_train %>%
  left_join(labeled,
            by = 'ID') %>%
  group_by(ID) %>%
  summarise(NDVI = mean(NDVI, na.rm=T),
            B7   = mean(B7  , na.rm=T),
            landcover = unique(landcover)) %>%
  as.data.frame()
head(d)
```

```
##     ID        NDVI       B7 landcover
## 1 2043  0.15067196 51.30000   orchard
## 2 2069  0.13429618 57.92000   orchard
## 3 2095  0.04711515 79.20833   orchard
## 4 2100  0.07668241 71.13043   orchard
## 5 2114 -0.09528784 76.96429   builtup
## 6 2118 -0.09528784 76.96429   builtup
```

We see now that there are 400 rows,

```r
nrow(d)
```

```
## [1] 400
```

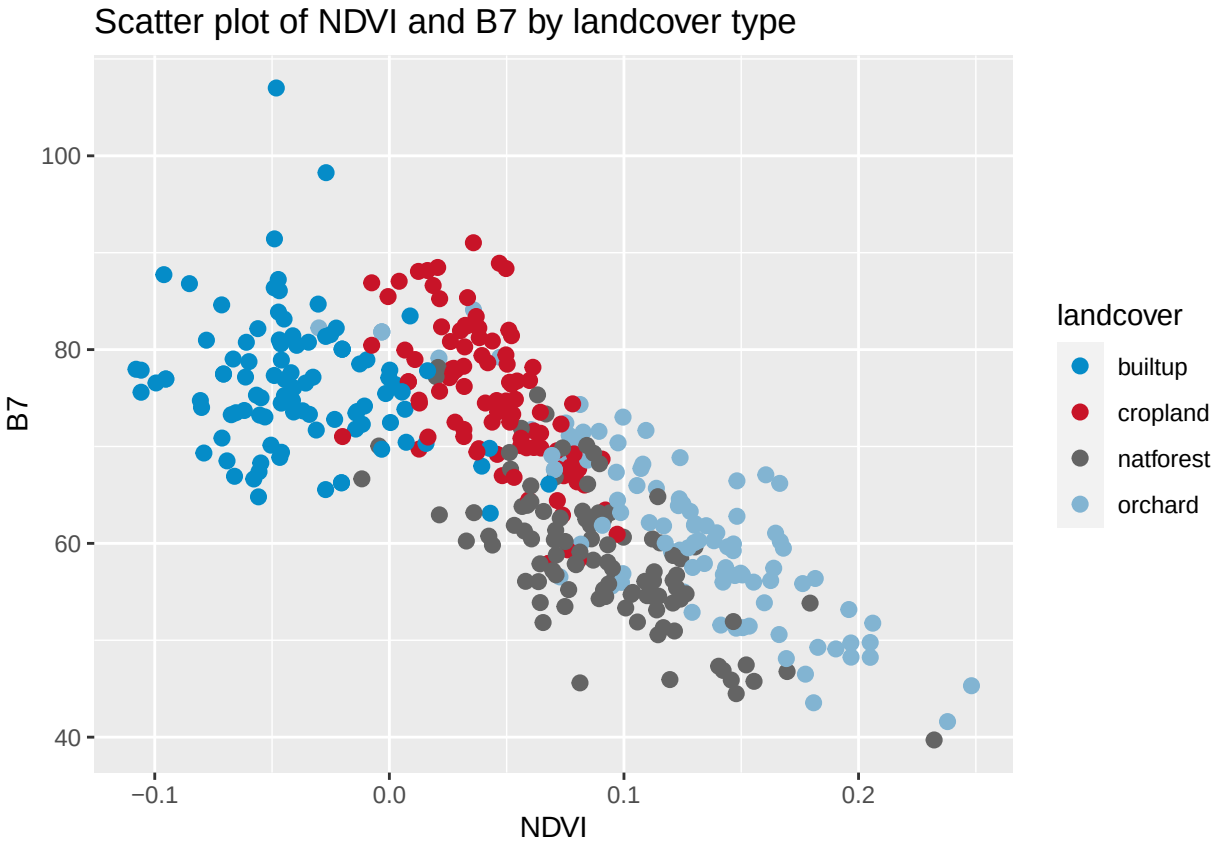
one row per location, and each row has `ID`, mean `NDVI`, mean `B7`, and `landcover` type for each location.

8. Data exploration

Perform some data exploration and visualization that helps you learn about the data, with a focus on predicting `landcover`. Discuss any notable observations. Which `landcover` types will be harder/easier to classify based on what you learned?

Let me begin with a scatter plot of `NDVI` and `B7` with different colors for different `landcover` types.

```r
d %>%
  ggplot(aes(x = NDVI,
             y = B7,
             color = landcover)) +
  geom_point() +
  labs(title = "Scatter plot of NDVI and B7 by landcover type",
       x = "NDVI",
       y = "B7")
```



From the plot, we could tell that `builtup` and `cropland` are easier to classify because they are more separated from the other two types. `orchard` and `natforest` are harder to classify because they are more mixed with each other.

9. Modeling

One approach to this classification problem is to use multinomial logistic regression. This is a generalization of logistic regression for when there are more than 2 categories. For each observation, instead of getting an estimated probability of success (or, 1, or "yes", etc.), we get estimated probabilities for all of the categories.

We use `NDVI`, `B7`, and an interaction term as predictors, and `landcover` as the outcome in the model below.

```
library(nnet) ## install.packages("nnet") if you don't have the package
mn1 = multinom(landcover ~ NDVI + B7 + B7:NDVI,
               data = d)
```

```
## # weights:  20 (12 variable)
## initial  value 554.517744
## iter  10 value 247.040612
## iter  20 value 235.566061
## iter  30 value 232.404384
## iter  40 value 230.534413
## iter  50 value 227.114240
## iter  60 value 227.018340
## iter  70 value 225.836746
## final  value 225.717675
## converged
```

```
#summary(mn1) ## in case you are curious, but not necessary
```

We can add predicted probabilities and the predicted class to our data frame using `predict` as usual.

```
mn1.landcover.probs = c('mn1.builtup',
                        'mn1.cropland',
                        'mn1.natforest',
                        'mn1.orchard')

d[,mn1.landcover.probs] = predict(mn1, newdata=d, type = 'probs') %>% round(5)
d$pred.landcover        = predict(mn1, newdata=d, type = 'class') ## predicted class
d %>% head()
```

```
##     ID        NDVI       B7 landcover mn1.builtup mn1.cropland mn1.natforest
## 1 2043  0.15067196 51.30000   orchard     0.00017      0.00113       0.46530
## 2 2069  0.13429618 57.92000   orchard     0.00017      0.01292       0.24472
## 3 2095  0.04711515 79.20833   orchard     0.00551      0.81355       0.01823
## 4 2100  0.07668241 71.13043   orchard     0.00217      0.53411       0.08546
## 5 2114 -0.09528784 76.96429   builtup     0.99987      0.00005       0.00008
## 6 2118 -0.09528784 76.96429   builtup     0.99987      0.00005       0.00008
##   mn1.orchard pred.landcover
## 1     0.53339        orchard
## 2     0.74219        orchard
## 3     0.16270       cropland
## 4     0.37826       cropland
## 5     0.00000        builtup
## 6     0.00000        builtup
```

We now have one row per ID, and probability of each landcover type in the `mn1.xxxxxx` columns.

Another approach is to build 4 logistic regression models, one for each `landcover` type:

1. outcome: 1 = `builtup` , 0 = not `builtup`. Predictors: `NDVI`, `B7`, and an interaction
2. outcome: 1 = `cropland` , 0 = not `cropland`. Predictors: `NDVI`, `B7`, and an interaction
3. outcome: 1 = `natforest`, 0 = not `natforest`. Predictors: `NDVI`, `B7`, and an interaction
4. outcome: 1 = `orchard` , 0 = not `orchard`. Predictors: `NDVI`, `B7`, and an interaction

Build these 4 logistic regression models, find the predicted probabilities for each class, and show that the estimated probabilities are similar to those from our multinomial logistic regression model above. Also, find the sum of the estimated probabilities from the multinomial logistic regression model, and the sum of estimated probabilities from the 4 logistic regression models. Based on these results, what is one downside of using 4 logistic regression models instead of multinomial logistic regression?

```
### builtup

d$landcover_builtup = ifelse(d$landcover == "builtup", 1, 0)

model_builtup = glm(landcover_builtup ~ NDVI + B7 + B7:NDVI,
                    data = d,
                    family = binomial)

d$pred_builtup = predict(model_builtup,
                         newdata = d,
                         type = 'response')

abs_builtup_diff = mean(abs(d$pred_builtup - d$mn1.builtup))
```

```r
### cropland

d$landcover_cropland = ifelse(d$landcover == "cropland", 1, 0)

model_cropland = glm(landcover_cropland ~ NDVI + B7 + B7:NDVI,
                     data = d,
                     family = binomial)

d$pred_cropland = predict(model_cropland,
                          newdata = d,
                          type = 'response')

abs_cropland_diff = mean(abs(d$pred_cropland - d$mn1.cropland))


### natforest

d$landcover_natforest = ifelse(d$landcover == "natforest", 1, 0)

model_natforest = glm(landcover_natforest ~ NDVI + B7 + B7:NDVI,
                     data = d,
                     family = binomial)

d$pred_natforest = predict(model_natforest,
                          newdata = d,
                          type = 'response')

abs_natforest_diff = mean(abs(d$pred_natforest - d$mn1.natforest))


### orchard

d$landcover_orchard = ifelse(d$landcover == "orchard", 1, 0)

model_orchard = glm(landcover_orchard ~ NDVI + B7 + B7:NDVI,
                     data = d,
                     family = binomial)

d$pred_orchard = predict(model_orchard,
                          newdata = d,
                          type = 'response')

abs_orchard_diff = mean(abs(d$pred_orchard - d$mn1.orchard))

cat("The absolute difference between the estimated probabilities from the multinomial logistic regressi
```

```
## The absolute difference between the estimated probabilities from the multinomial logistic regression
```

```r
cat("The sum of estimated probabilities from the multinomial logistic regression model is:", sum(d$mn1.
```

```
## The sum of estimated probabilities from the multinomial logistic regression model is: 400
```

```r
cat("The sum of estimated probabilities from the 4 logistic regression models is:", sum(d$pred_builtup
```

```
## The sum of estimated probabilities from the 4 logistic regression models is: 400
```

```
cat('The sum of the absolute difference between the estimated probabilities from the multinomial logisti
```

```
## The sum of the absolute difference between the estimated probabilities from the multinomial logistic
```

```
cat('The sum of the absolute difference between the estimated probabilities from the 4 logistic regressi
```

```
## The sum of the absolute difference between the estimated probabilities from the 4 logistic regressio
```

From the results, we see that the difference between the multinomial logistic regression model and the 4 logistic regression models is very small. The sum of the estimated probabilities from the multinomial logistic regression model is 400, and the sum of the estimated probabilities from the 4 logistic regression models is also 400, which is good. However, if we take the sum of the absolute difference of the sum of the probabilities of one specific location and 1, we see that the multinomial model is much better than the 4 separate models. Hence the downside of using 4 models is that the sum of probabilities of each type is not guaranteed to be 1.

Simulation

10. Repeat #4 from PSET4, but for Poisson Regression.

In particular, suppose $y \sim Pois(\lambda)$ and $\lambda = exp(1 + 2x)$. Simulate some data by doing the following:

a. Generate a random sample of 10,000 x values on $[0, 1]$.
b. Generate 10,000 λs using $\lambda = e^{1+2x}$.
c. Generate a random sample of 10,000 y values using $y \sim Pois(\lambda)$.

Then,

d. Fit a Poisson regression model to the data and report the estimated coefficients. Are the estimates what you would expect? Why or why not?

e. Plot a scatter plot of y versus x, and add a smooth curve to the plot. Does this visualization look as you would have expected? Do the Poisson regression assumptions appear to hold? Why or why not?

Let's begin from a,b,c.

```
set.seed(123)

x = runif(10000, 0, 1)

lambda = exp(1 + 2*x)

y = rpois(10000, lambda)
```

Proceed to d.

```
df <- data.frame(x = x, y = y)
m <- glm(y ~ x, data = df, family = poisson)
summary(m)
```

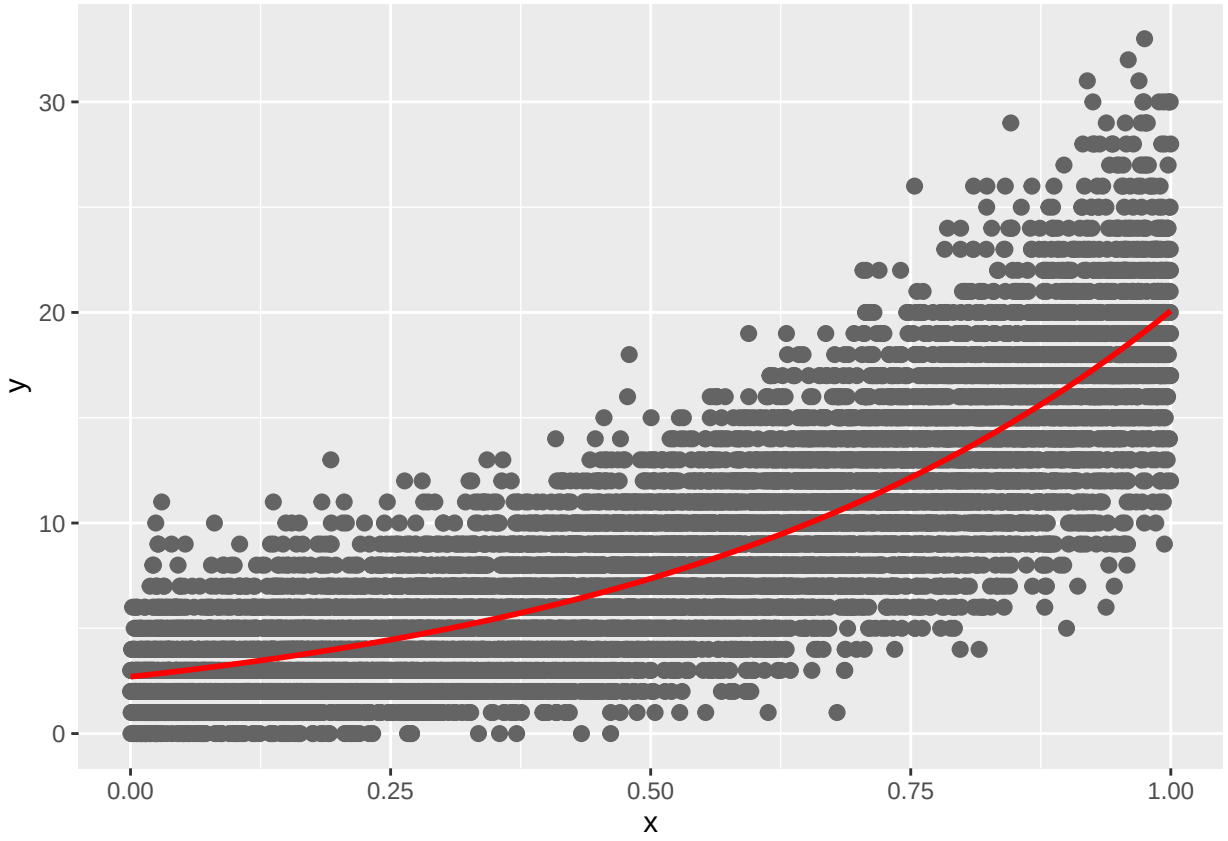
```
##
## Call:
## glm(formula = y ~ x, family = poisson, data = df)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 0.992966   0.009165   108.3   <2e-16 ***
## x           2.006677   0.013032   154.0   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 36277  on 9999  degrees of freedom
## Residual deviance: 10295  on 9998  degrees of freedom
## AIC: 47909
##
## Number of Fisher Scoring iterations: 4
```

We tell that the Poisson model estimates the coefficient of x to be 2.006677 and the intercept to be 0.992966. This is exactly what we expect because we generated the data using $\lambda = e^{1+2x}$.

Now let's move to e.

```
ggplot(df, aes(x = x, y = y)) +
    geom_point() +
    geom_smooth(method = "glm", method.args = list(family = poisson), se = F, color = "red")
```



The visualization looks as expected. The assumptions for Poisson regression appear to be satisfied. The first assumption is that the response variable is a independently distributed non-negative integer. Given that the way we generate y is `y = rpois(10000, lambda)`, this assumption is automatically satisfied. The second assumption is that the mean of the response variable is equal to the variance of the response variable. This is satisfied by looking the plot, as the mean increases, the variance also increases approximately the same amount. The third assumption is that the response variable is Poisson distributed, this is also automatically satisfied by the way we generate y. The last assumption is that the logarithm of the mean of the response variable is linear of the predict variable, which is also automatically satisfied.