Modeling Data

Manda Egbert, Stephanie Renasco, Tyler Maull

2022-11-14

### Load and Import

This chunk loads multiple packages to aid in organizing and modeling our data. It also uses read.csv to import our cleaned data set.

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──  
## ✔ ggplot2 3.3.6 ✔ purrr 0.3.4  
## ✔ tibble 3.1.7 ✔ dplyr 1.0.9  
## ✔ tidyr 1.2.0 ✔ stringr 1.4.0  
## ✔ readr 2.1.2 ✔ forcats 0.5.1  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(modelr)  
library(cvTools)

## Loading required package: lattice  
## Loading required package: robustbase  
##   
## Attaching package: 'cvTools'  
##   
## The following object is masked from 'package:modelr':  
##   
## mape

library(glmnet)

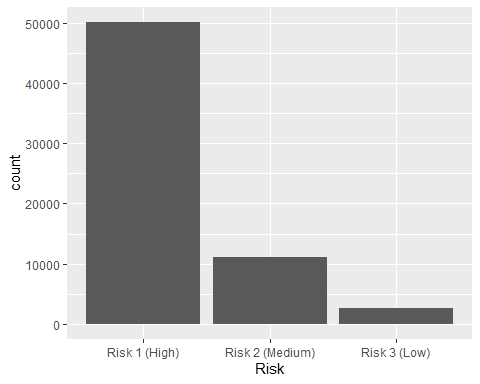
## Loading required package: Matrix  
##   
## Attaching package: 'Matrix'  
##   
## The following objects are masked from 'package:tidyr':  
##   
## expand, pack, unpack  
##   
## Loaded glmnet 4.1-4

set.seed(123)  
  
food.inspections.clean <- read.csv("Y:/Stat 220/Group Project/food.inspections.clean.csv")  
  
#food.inspections.clean <- read.csv("Y:/Stat 220/Group Project/food.inspections.clean.csv")

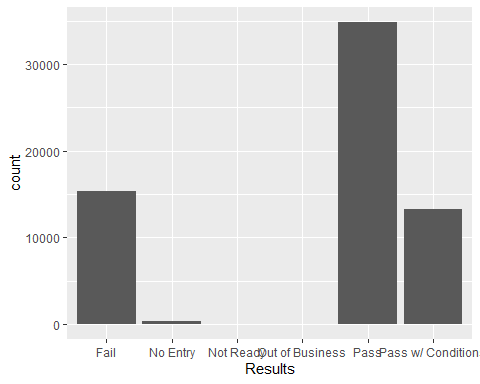
### Bar Graphs

Our data is almost entirely categorical, so even though it would be nice to use the pairs function to see relationships between our variables, it would not be wise to use scatter plots to visualize our data. Instead, we have used bar graphs. The chunk below displays the distribution of violation risk and inspection results. We can draw a few conclusions from these graphs. First, a vast majority of businesses are evaluated at high risk, or considered to be more likely to have health violations. A very small portion were considered to be low risk. This would lead one to believe that most businesses fail their inspection, however, the results graph shows that most businesses do pass. This is an interesting contradiction and suggest that the way businesses are evaluated for their risk level might not be the most accurate.

ggplot(data = food.inspections.clean) +  
 geom\_bar(mapping = aes(x=Risk))



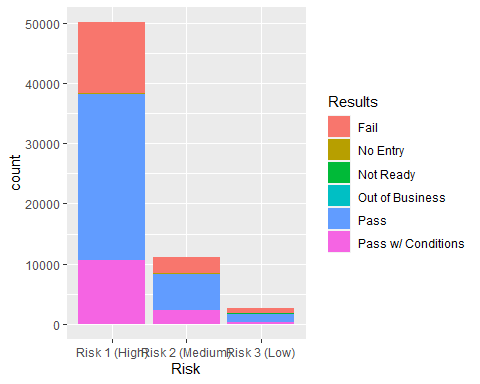
ggplot(data = food.inspections.clean) +  
 geom\_bar(mapping = aes(x=Results))



### Results as related to risk

This stacked bar graph displays similar information to the two above. However, this graph allows us to get a better idea of the distribution of results within each risk level. As we noticed before, a very large portion of high risk establishments pass their inspections. The same can be said for the other two risk levels. Something we do notice here that we didn’t before is that the distribution of results evens out as their risk gets lower. Low risk establishments have a similar chance of failing that they do to passing, while high rish establishments are much more likely to pass.

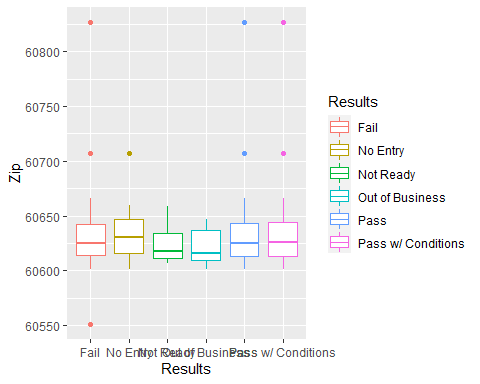
ggplot(data = food.inspections.clean) +  
 geom\_bar(mapping = aes(x=Risk, fill=Results))



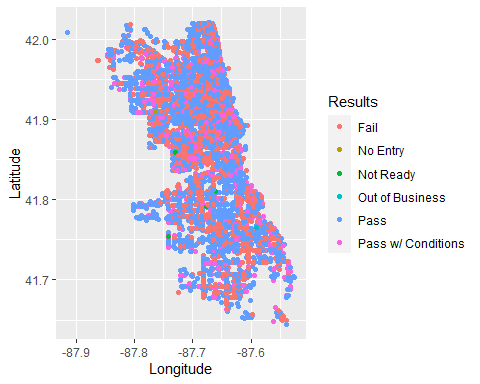
### Results as related to Zip Code

Given the zip code of each food establishment, we wondered if there were particular locations that were more prone to a passing or failing than others. Using a boxplot, we looked at the relationship between results and zip code. The first thing we noticed from this graph is that the majority of businesses, regardless of pass or fail, are located near each other. Looking at the distribution of pass and fail, we notice that their box is in nearly the same spot, with their outliers in similar distributions, with passing having a large number. This suggest that location is not a strong predictor of results. We can confirm this by using correlation coefficients and linear regression.

ggplot(data=food.inspections.clean, aes(x=Results, y=Zip ))+  
 geom\_boxplot(aes(color=Results))

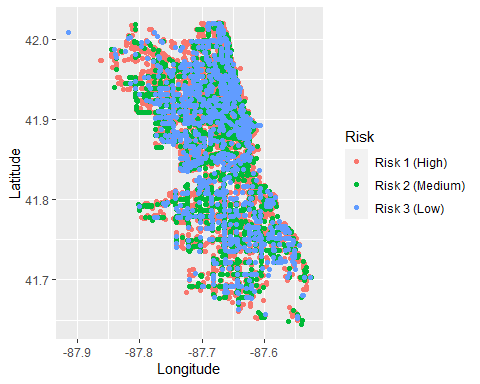
 ### Location distribution Given a longitude and latitude, we can use these as our axis in a scatter plot to see the location distribution of points. This makes a pretty neat image of Chicago color coded to show where certain results took place. We can also see the establishment density in certain parts of the map, where most businesses are located. In the dense locations, it is quite hard to see whether there is more passing or failing happening, but looking at the more sparse sections, we can see there are a lot more blue (or passing) points.

ggplot(data = food.inspections.clean) +  
 geom\_point(mapping = aes(x=Longitude, y=Latitude, color=Results))



Similar to the above map, we have a scatter plot of the points by location in the shape of Chicago. This time, our points are color coded to show their assigned risk value. Something interesting we noticed is that the farther we go from the inner city, or more populated locations, risk seems to increase. There are not many blue, or low risk points on the outskirts, they are condensed inside. We are unaware of what leads an establishment to recieve a higher risk rating, but this would be interesting to explore.

ggplot(data = food.inspections.clean) +  
 geom\_point(mapping = aes(x=Longitude, y=Latitude, color=Risk))



### Prepare for model

We would like to make a model that can accurately predict whether an establishment will pass or fail. To do this, we’ve made Results our response variable. However, Results, along with nearly all of our other variables are categorical or character. This is not idea for a regression analysis. To combat this, we’ve changed the data types of a few of our variables.

This chunk drop NA values so that they do not intervene in our analysis. It also uses mutate to assign a numerical value to each level of our Results variable.

food.inspections.clean <- food.inspections.clean %>% drop\_na()  
food.inspections.clean <- food.inspections.clean %>% mutate(Results = case\_when(  
 Results == "Business not Located" ~ 1,  
 Results == "Fail" ~ 2,  
 Results == "No Entry" ~ 3,  
 Results == "Not Ready" ~ 4,  
 Results == "Out of Business" ~ 5,  
 Results == "Pass" ~ 6,  
 Results == "Pass w/ Conditions" ~ 7))  
  
#as.numeric(food.inspections.clean$Results)

This chunk does the same as the above one, but uses mutate to assign a numerical value to each level of the Inspection Type variable.

food.inspections.clean <- food.inspections.clean %>% mutate(Inspection.Type = case\_when(  
 Inspection.Type == "Canvass" ~ 1,  
 Inspection.Type == "Complaint" ~ 2,  
 Inspection.Type == "Consultation" ~ 3,  
 Inspection.Type == "Fire" ~ 4,  
 Inspection.Type == "License" ~ 5,  
 Inspection.Type == "Non-Inspection" ~ 6,  
 Inspection.Type == "Not Ready" ~ 7,  
 Inspection.Type == "Re-Inspection" ~ 8,  
 Inspection.Type == "Recent Inspection" ~ 9,  
 Inspection.Type == "Special Event" ~ 10,  
 Inspection.Type == "Suspected Food Poisoning" ~ 11,  
 Inspection.Type == "Tag Removal" ~ 12,  
 Inspection.Type == "Not Ready" ~ 13))

This chunk also uses mutate to assign numerical values to each level of a variable, this time, Risk.

food.inspections.clean <- food.inspections.clean %>% mutate(Risk = case\_when(  
 Risk == "Risk 1 (High)" ~ 1,  
 Risk == "Risk 2 (Medium)" ~ 2,  
 Risk == "Risk 3 (Low)" ~ 3))

This chunk has similar results to the last three chunk, but uses a different method. The Inspection ID and Zip code variables contain far too many levels to be individually mutated. So we will set these factors as characters using as.character, then set them as numeric using as.numeric.

food.inspections.clean$Inspection.ID <- as.numeric(as.character(food.inspections.clean$Inspection.ID))  
food.inspections.clean$Zip <- as.numeric(as.character(food.inspections.clean$Zip))  
#food.inspections.clean$Facility.Type <- as.numeric(food.inspections.clean$Facility.Type)

### Correlations

Now that our variables are numeric, we can use them in statistical analysis. The following chunks generate the correlation coefficient between Results (our response) and an explanatory variable. The closer the coefficient is to -1 or 1, the stronger the relationship between our variables. The closer to 0, the weaker. A negative value means a negative relationship, while a positive value means a positive relationship.

Results and Risk have a weak, negative relationship.

cor(food.inspections.clean$Results, food.inspections.clean$Risk)

## [1] -0.04147528

Results and Insepction Type have a weak, positive relationship. This is the strongest relationship out of all the variables.

cor(food.inspections.clean$Results, food.inspections.clean$Inspection.Type)

## [1] 0.06307521

Results and Inspection ID have a weak, positive relationship.

cor(food.inspections.clean$Results, food.inspections.clean$Inspection.ID)

## [1] 0.02089477

Results and Zip have a weak, negative relationship. This is the weakest relationship.

cor(food.inspections.clean$Results, food.inspections.clean$Zip)

## [1] -0.00232801

Results and Latitude have a weak, positive relationship.

cor(food.inspections.clean$Results, food.inspections.clean$Latitude)

## [1] 0.01457115

Results and Longitude have a weak, positive relationship.

cor(food.inspections.clean$Results, food.inspections.clean$Longitude)

## [1] 0.007338448

### Simple Linear Regression

The following chunks create a linear model using a single explanatory variable to predict our response variable, Results. We will use the output to consider whether the variable is significant (p-value less than 5%) and whether r-squared is high enough (preferably greater than 80%, but we will be satisfied with just the highest value we can find) to suggest the regression does a good job at explaining the variation in our response.

Inspection ID is significant with a p-value below 5% at 1.31e-07. However, it has a small r-squared of 0.0004366. This suggests that only 0.04366% of the variation in the response variable is explained by the regression.

model.ID <- lm(Results ~ Inspection.ID, data=food.inspections.clean)  
summary(model.ID)

##   
## Call:  
## lm(formula = Results ~ Inspection.ID, data = food.inspections.clean)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.2854 0.7173 0.7611 0.8294 1.9164   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 4.947e+00 5.401e-02 91.592 < 2e-16 \*\*\*  
## Inspection.ID 1.485e-07 2.813e-08 5.278 1.31e-07 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.873 on 63782 degrees of freedom  
## Multiple R-squared: 0.0004366, Adjusted R-squared: 0.0004209   
## F-statistic: 27.86 on 1 and 63782 DF, p-value: 1.309e-07

Inspection Type is significant with a p-value of 2.2e-16, but has a small r-squared of 0.003978. This is the best model out of all our simple linear models as it has the highest r-squared and explains the largest amount of the variation in the response.

#best  
model.inspection.type <- lm(Results ~ Inspection.Type, data=food.inspections.clean)  
summary(model.inspection.type)

##   
## Call:  
## lm(formula = Results ~ Inspection.Type, data = food.inspections.clean)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.5725 0.5551 0.8527 0.8527 1.8527   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.104815 0.010754 474.68 <2e-16 \*\*\*  
## Inspection.Type 0.042513 0.002663 15.96 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.869 on 63782 degrees of freedom  
## Multiple R-squared: 0.003978, Adjusted R-squared: 0.003963   
## F-statistic: 254.8 on 1 and 63782 DF, p-value: < 2.2e-16

Risk is significant with a p-value of 2.2e-16, but has a small r-squared of 0.00172.

model.risk <- lm(Results ~ Risk, data=food.inspections.clean)  
summary(model.risk)

##   
## Call:  
## lm(formula = Results ~ Risk, data = food.inspections.clean)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.2674 0.7326 0.7326 0.8818 2.0310   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.41654 0.01933 280.19 <2e-16 \*\*\*  
## Risk -0.14918 0.01423 -10.48 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.871 on 63782 degrees of freedom  
## Multiple R-squared: 0.00172, Adjusted R-squared: 0.001705   
## F-statistic: 109.9 on 1 and 63782 DF, p-value: < 2.2e-16

Zip is insignificant with a p-value greater than 5% at 55.7%, it has an extremely small r-squared of 5.42e-06.

model.zip <- lm(Results ~ Zip, data=food.inspections.clean)  
summary(model.zip)

##   
## Call:  
## lm(formula = Results ~ Zip, data = food.inspections.clean)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.2470 0.7643 0.7692 0.7769 1.8152   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 18.8957313 23.2444048 0.813 0.416  
## Zip -0.0002254 0.0003834 -0.588 0.557  
##   
## Residual standard error: 1.873 on 63782 degrees of freedom  
## Multiple R-squared: 5.42e-06, Adjusted R-squared: -1.026e-05   
## F-statistic: 0.3457 on 1 and 63782 DF, p-value: 0.5566

Longitude is insignificant with a p-value of 0.0638, it has a small r-squared of 5.385e-05.

model.longitude <- lm(Results ~ Longitude, data=food.inspections.clean)  
summary(model.longitude)

##   
## Call:  
## lm(formula = Results ~ Longitude, data = food.inspections.clean)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.2640 0.7411 0.7671 0.7915 1.8254   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 25.3977 10.8819 2.334 0.0196 \*  
## Longitude 0.2300 0.1241 1.853 0.0638 .  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.873 on 63782 degrees of freedom  
## Multiple R-squared: 5.385e-05, Adjusted R-squared: 3.818e-05   
## F-statistic: 3.435 on 1 and 63782 DF, p-value: 0.06383

Latitude is significant with a p-value of 0.000233, but has a small r-squared of 0.0002123.

model.latitude <- lm(Results ~ Latitude, data=food.inspections.clean)  
summary(model.latitude)

##   
## Call:  
## lm(formula = Results ~ Latitude, data = food.inspections.clean)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.2760 0.7260 0.7661 0.8168 1.8520   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -9.00839 3.86859 -2.329 0.019884 \*   
## Latitude 0.33993 0.09236 3.680 0.000233 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.873 on 63782 degrees of freedom  
## Multiple R-squared: 0.0002123, Adjusted R-squared: 0.0001966   
## F-statistic: 13.54 on 1 and 63782 DF, p-value: 0.0002331

### Multiple Linear Regression

The following chunks create a multiple linear model using mulitple explanatory variables to predict our response variable, Results. We will use the output to determine which variables are significant (p-value less than 5%) and whether r-squared is high enough (preferably greater than 80%, but we will be satisfied with just the highest value we can find) to suggest the regression does a good job at explaining the variation in our response.

This first model is a full model containing all variables (violations, name, and date have been omitted for incompatibility). Looking at the output, we see that Inspection ID, Inspection Type, Risk, Latitude, and Longitude are significant as they each have a p-value under 5% and have two to three stars next to them. Zip is insignificant and will be omitted from future models. The adjusted r-squared is 0.006702, which is higher than that of any simple linear model we previously made, but still small.

#omitting violations, name, date  
model.all <- lm(Results ~ Inspection.ID + Inspection.Type + Risk + Zip + Latitude + Longitude, data=food.inspections.clean)  
summary(model.all)

##   
## Call:  
## lm(formula = Results ~ Inspection.ID + Inspection.Type + Risk +   
## Zip + Latitude + Longitude, data = food.inspections.clean)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.7287 0.4290 0.7827 0.9773 2.2491   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.684e+01 2.326e+01 1.153 0.24871   
## Inspection.ID 1.281e-07 2.807e-08 4.566 4.99e-06 \*\*\*  
## Inspection.Type 4.484e-02 2.671e-03 16.788 < 2e-16 \*\*\*  
## Risk -1.603e-01 1.432e-02 -11.191 < 2e-16 \*\*\*  
## Zip -2.699e-05 4.181e-04 -0.065 0.94853   
## Latitude 4.337e-01 1.024e-01 4.238 2.26e-05 \*\*\*  
## Longitude 4.369e-01 1.468e-01 2.976 0.00292 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.867 on 63777 degrees of freedom  
## Multiple R-squared: 0.006795, Adjusted R-squared: 0.006702   
## F-statistic: 72.73 on 6 and 63777 DF, p-value: < 2.2e-16

This model is similar to the full model, but with Zip removed. Every variable is significant and the adjusted r-squared has increased by 0.000016. This is the best so far.

#omitting violations, name, date, AND ZIP  
model2 <- lm(Results ~ Inspection.ID + Inspection.Type + Risk + Latitude + Longitude, data=food.inspections.clean)  
summary(model2)

##   
## Call:  
## lm(formula = Results ~ Inspection.ID + Inspection.Type + Risk +   
## Latitude + Longitude, data = food.inspections.clean)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.7296 0.4287 0.7828 0.9775 2.2486   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.551e+01 1.088e+01 2.345 0.01905 \*   
## Inspection.ID 1.281e-07 2.806e-08 4.565 5.00e-06 \*\*\*  
## Inspection.Type 4.484e-02 2.671e-03 16.788 < 2e-16 \*\*\*  
## Risk -1.603e-01 1.432e-02 -11.192 < 2e-16 \*\*\*  
## Latitude 4.336e-01 1.023e-01 4.238 2.26e-05 \*\*\*  
## Longitude 4.404e-01 1.368e-01 3.219 0.00129 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.867 on 63778 degrees of freedom  
## Multiple R-squared: 0.006795, Adjusted R-squared: 0.006718   
## F-statistic: 87.27 on 5 and 63778 DF, p-value: < 2.2e-16

This model is created using backwards step wise regression. It gives us the model that results in the smallest AIC, which was a model containing ID, type, risk, latitude, and longitude. This is the same as our model2, which was the full model minus zip. We will use this model in place of model 2 from now on.

fit.back <- step(model.all)

## Start: AIC=79628.52  
## Results ~ Inspection.ID + Inspection.Type + Risk + Zip + Latitude +   
## Longitude  
##   
## Df Sum of Sq RSS AIC  
## - Zip 1 0.01 222225 79627  
## <none> 222225 79629  
## - Longitude 1 30.86 222256 79635  
## - Latitude 1 62.58 222288 79644  
## - Inspection.ID 1 72.63 222298 79647  
## - Risk 1 436.41 222662 79752  
## - Inspection.Type 1 982.05 223207 79908  
##   
## Step: AIC=79626.53  
## Results ~ Inspection.ID + Inspection.Type + Risk + Latitude +   
## Longitude  
##   
## Df Sum of Sq RSS AIC  
## <none> 222225 79627  
## - Longitude 1 36.11 222261 79635  
## - Latitude 1 62.57 222288 79642  
## - Inspection.ID 1 72.61 222298 79645  
## - Risk 1 436.42 222662 79750  
## - Inspection.Type 1 982.04 223207 79906

fit.back

##   
## Call:  
## lm(formula = Results ~ Inspection.ID + Inspection.Type + Risk +   
## Latitude + Longitude, data = food.inspections.clean)  
##   
## Coefficients:  
## (Intercept) Inspection.ID Inspection.Type Risk   
## 2.551e+01 1.281e-07 4.484e-02 -1.603e-01   
## Latitude Longitude   
## 4.336e-01 4.404e-01

### Akaike Information Criterion (AIC)

Comparing the AIC of each of the three models, we see that the model chosen by the backwards step wise regression is the best as it is the lowest. Second best would be the full model.

AIC(model.inspection.type, model.all, fit.back)

## df AIC  
## model.inspection.type 3 260812.3  
## model.all 8 260641.7  
## fit.back 7 260639.7

### Cross Validation (CV)

We will now use cross validation to find the average root mean squared error (RMSE). We’re looking for the smallest value. We can see that the backward step wise model and the full model are very close, but the backward step wise is smaller, therefore the best.

cv.all <- cvFit(model.all, data=food.inspections.clean, y=food.inspections.clean$Results, K=10, R=10)  
cv.inspection.type <- cvFit(model.inspection.type, data=food.inspections.clean, y=food.inspections.clean$Results, K=10, R=10)  
cv.back <- cvFit(fit.back, data=food.inspections.clean, y=food.inspections.clean$Results, K=10, R=10)  
  
data.frame(  
 cv.all=cv.all$cv,   
 cv.inspection.type=cv.inspection.type$cv,   
 cv.back=cv.back$cv  
 )

## cv.all cv.inspection.type cv.back  
## CV 1.866766 1.869256 1.866732

### Conclusion

After analyzing the data, we can conclude that inspection ID, inspection type, evaluated risk level, latitude, and longitude are all significant in determining the results of Chicago’s food inspection. Our research question, given this data set, was if inspection type and risk evaluation were related to the results of the inspections. From our simple linear model, we can conclude that inspection type has the strongest influence on whether a business passes or fails.

### Problems and solutions found during the research

This data set contains over 180,000 observations, it made it quite difficult to digest and skim over the data. The best way to combat this issue was to use tables that display the observations with their count so we could view many thousands within a single screen. Additionally, using bar graphs made it easier to understand what we were looking at as far as the distribution of each variable.

Correcting misspelling proved to be extremely time consuming, though very important. The facility variable contained over 470 different types, however, many of these “different” types were the same type, just spelled in different or incorrect ways, so the computer read them as different. In order to get an accurate idea of the number of each type of facility, we needed to fix these inaccuracies. The problem that arose while doing this was that each facility type had to be individually typed out with the case\_when function, including those that were spelled correctly in the first place. This was a daunting task as it meant writing out over 470 lines of code to fix one variable. It was extremely time consuming, but using RStudio’s multi-tab view option, a scratchpad, and copy and paste, made it much easier.

The size of this file made it challenging to have a copy of the data set on a student’s Y:Drive. Running the regressions required multiple GB of space, but the Y:Drive only contains 1GB, so we opted to use a personal computer for most of the project.

### Ideas for future investigation

Something that isn’t explained in the dataset’s description is how businesses were evaluated for their risk factor. I would be interested to know why certain businesses were high risk while others weren’t, especially because so few were considered low risk. Understanding why so many businesses were high risk, but still passed might give us further insight into what drives a passing or failing score. Furthermore, with this knowledge we would possibly make the risk evaluations more accurate, and therefore a better predictor of inspection results.