Multiclass Lab Notebook

Codie Wood

2023-01-11

Multi-class Classification

In this section we consider the multi-class classification task of predicting the type of a reported crime, given data on the crime. We also extend this to address the secondary question of if we can predict the time of day at which a crime occured, given data on the crime. We will focus here on the method of multinomial logistic regression for both of these tasks.

Firstly, we perform feature selection by manual investigation of our variables. We will identify variables to include in our classifier, taking into account both their real world interpretations and computational complexity of the model. We will select several feasible models, and use multiple metrics to assess their performance. We will use repeated k-fold cross validation to calculate average values of these metrics across different training sets. We will then compare our models using these averages.

```
#wont need this when loading package
require(nnet) # For multinomial regression
```

```
## Loading required package: nnet
```

```
require(tidyverse) # For data manip
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
require(lubridate) # For date/time
```

```
## Loading required package: lubridate
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
require(ggplot2) # For plots
require(sf) # For spatial plots
```

```
## Loading required package: sf
## Linking to GEOS 3.8.0, GDAL 3.0.4, PROJ 6.3.1; sf_use_s2() is TRUE
```

```
require(caret) # For confusion matrix
```

```
## Loading required package: caret
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
#shouldnt need below when package sorted
source("chicri/R/multiclass_classification.R")
source("chicri/R/utils.R")
source("chicri/R/location_analysis.R")
theme_set(theme_bw())
```

Feature Selection

Due to the extremely large size of the full data set, it is computationally intensive to download, store and use to train models. As such, we instead train and test the multinomial logistic regression models only on data from 2019.

```
dat <- load_crimes_csv("chicri/data/processed/Crimes_2019_Location_Type.csv")
```

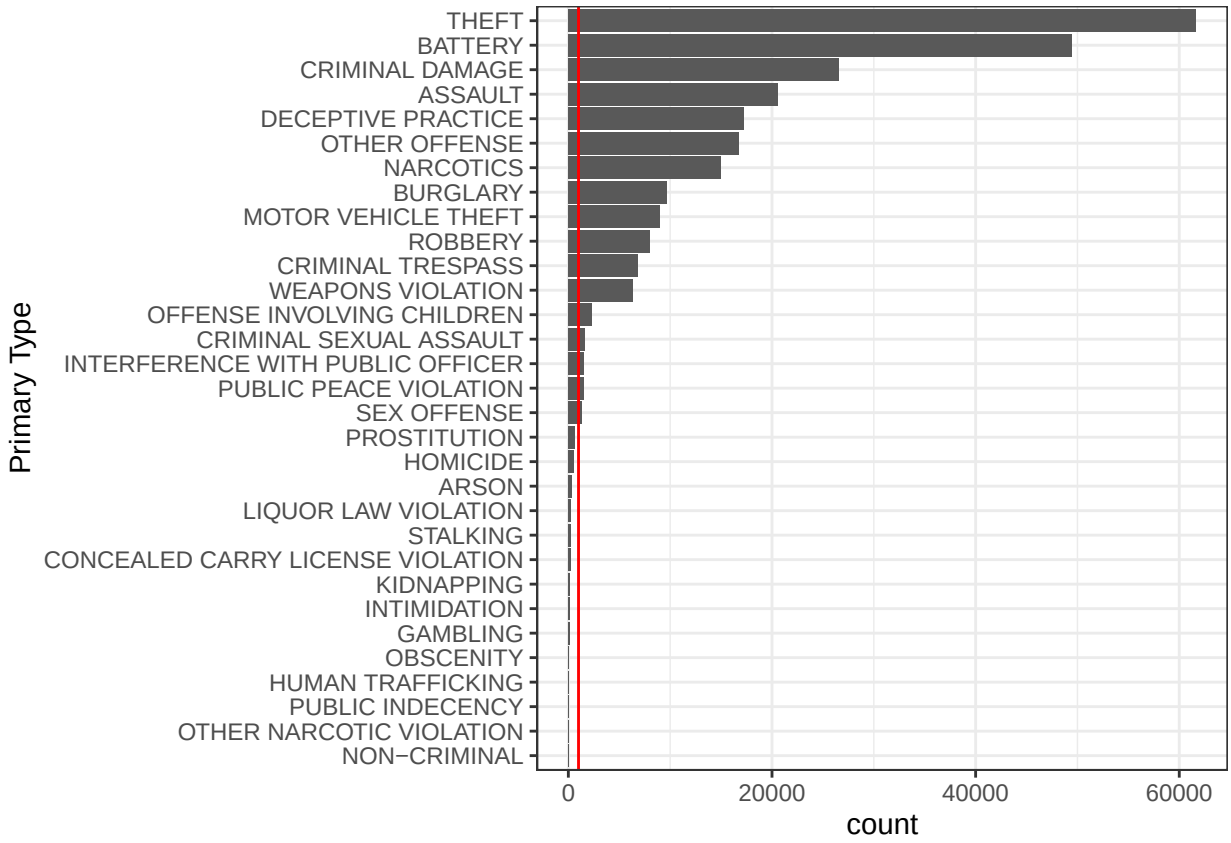
```
## Loading processed data...

## Rows: 258176 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): Primary Type, Location Description, District
## dbl  (2): X Coordinate, Y Coordinate
## lgl  (2): Arrest, Domestic
## dttm (1): Date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# API stuff below should give same data set
#dat <- load_crimes_API(year="2019")
#vars_to_remove <- c("ID", "Case Number", "Block", "IUCR", "FBI Code", "Description", "Beat", "Ward", "
#dat <- process_data(dat, remove_vars = vars_to_remove)
```

As well as the data pre-processing described earlier in the report, we also perform additional feature selection for this particular task. In order to do this we incorporate information from conducting exploratory data analysis, as well as considering computational cost.

Primary Type

```
# Reorders type factor for plotting
dat$`Primary Type` <- fct_infreq(dat$`Primary Type`) %>% fct_rev()

thr <- 1000 # set threshold for othering
ggplot(dat) +
  geom_bar(aes(y = `Primary Type`)) +
  geom_vline(xintercept=thr, colour="red")
```

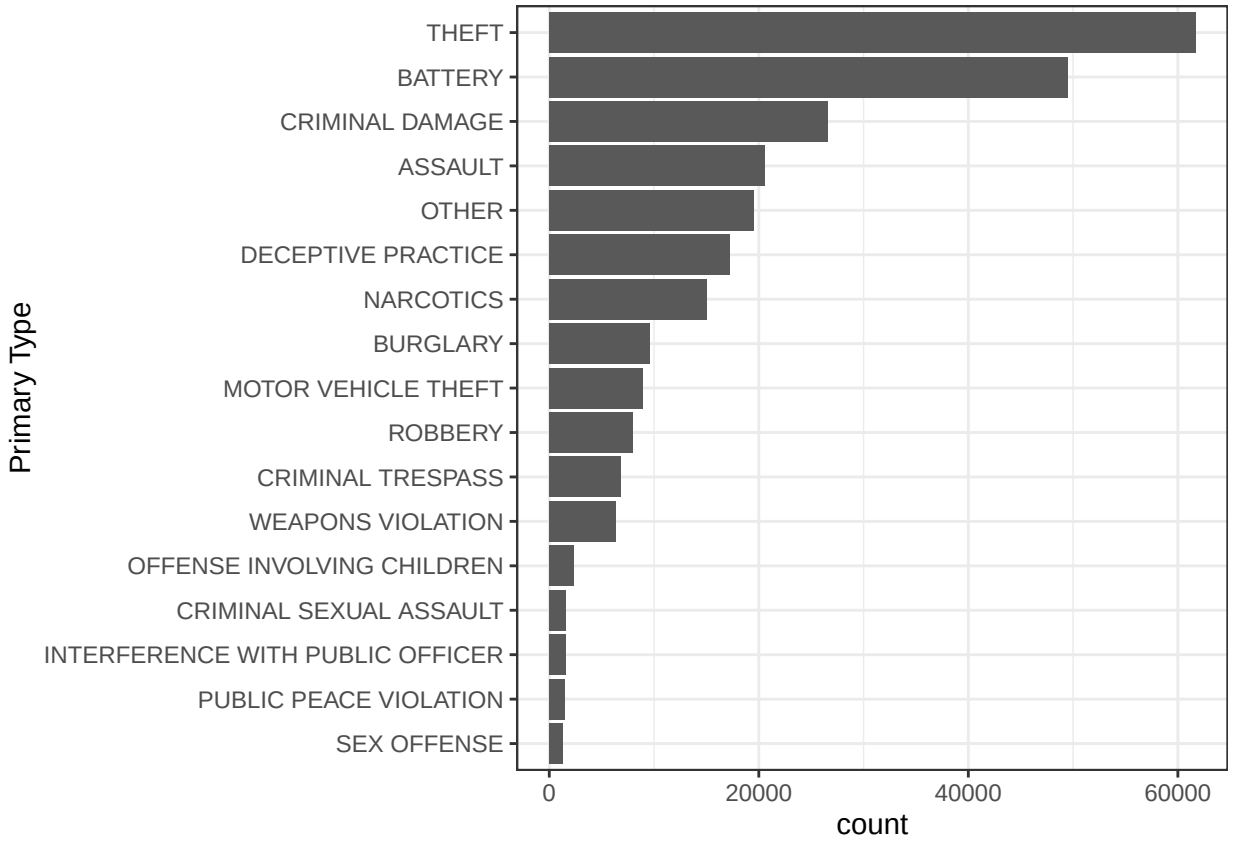


We observe that there are initially 31 crime types as classified in our data set. Of these, 14 types have fewer than 1000 observations. In order to reduce the computational complexity of our model, we merge these uncommon levels of our factor variable into the level OTHER, in which we also include the pre-existing crime type OTHER OFFENSE. This leaves us with a total of 17 levels in our variable, with 7.57% of data values in the OTHER category.

```
# Reordering and othering factor levels (for plots and analysis purposes)
dat$`Primary Type` <- fct_recode(dat$`Primary Type`,"OTHER" = "OTHER OFFENSE") %>%
  othering(thr,print_summary = T) %>%
  fct_infreq() %>%
  fct_rev()
```

```
## 14 out of 31 categories converted to OTHER, 7.57% of data values.
```

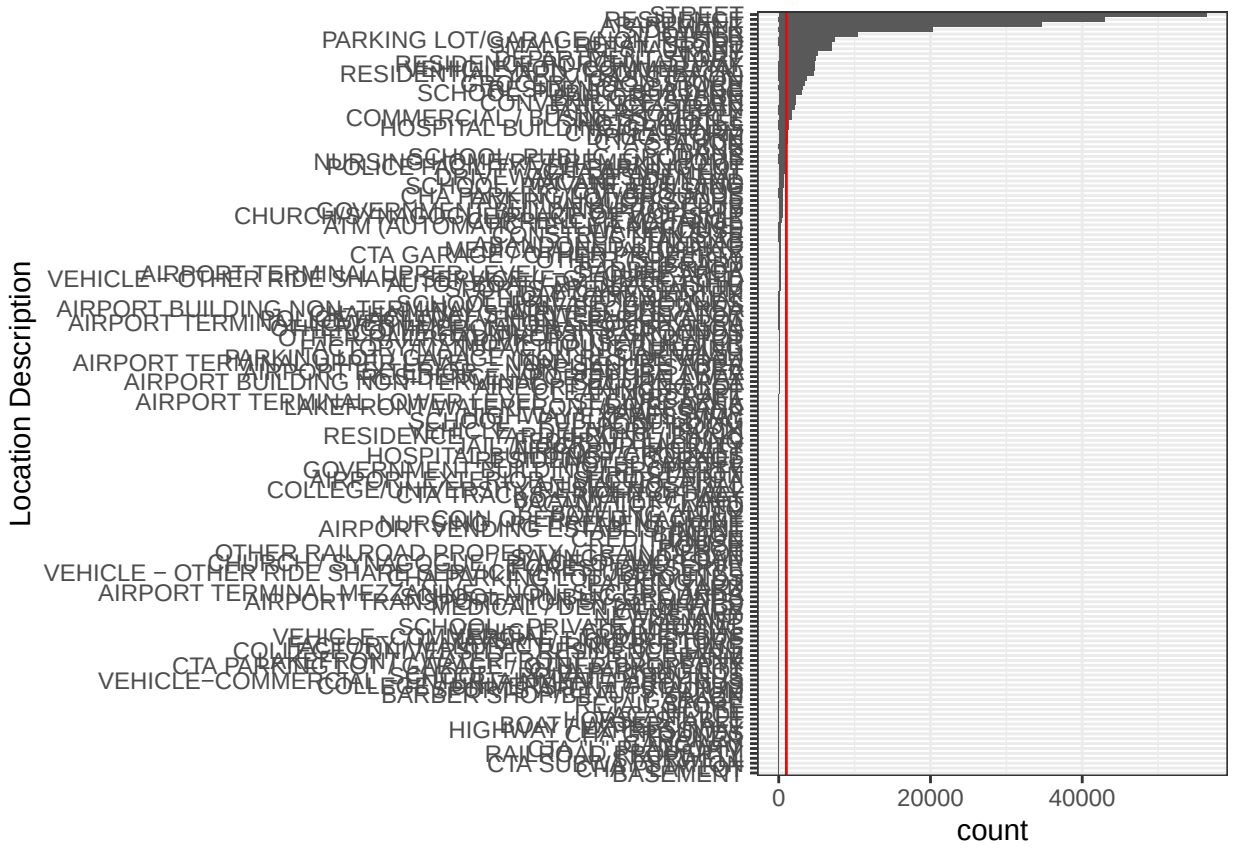
```
ggplot(dat) +
  geom_bar(aes(y = `Primary Type`))
```



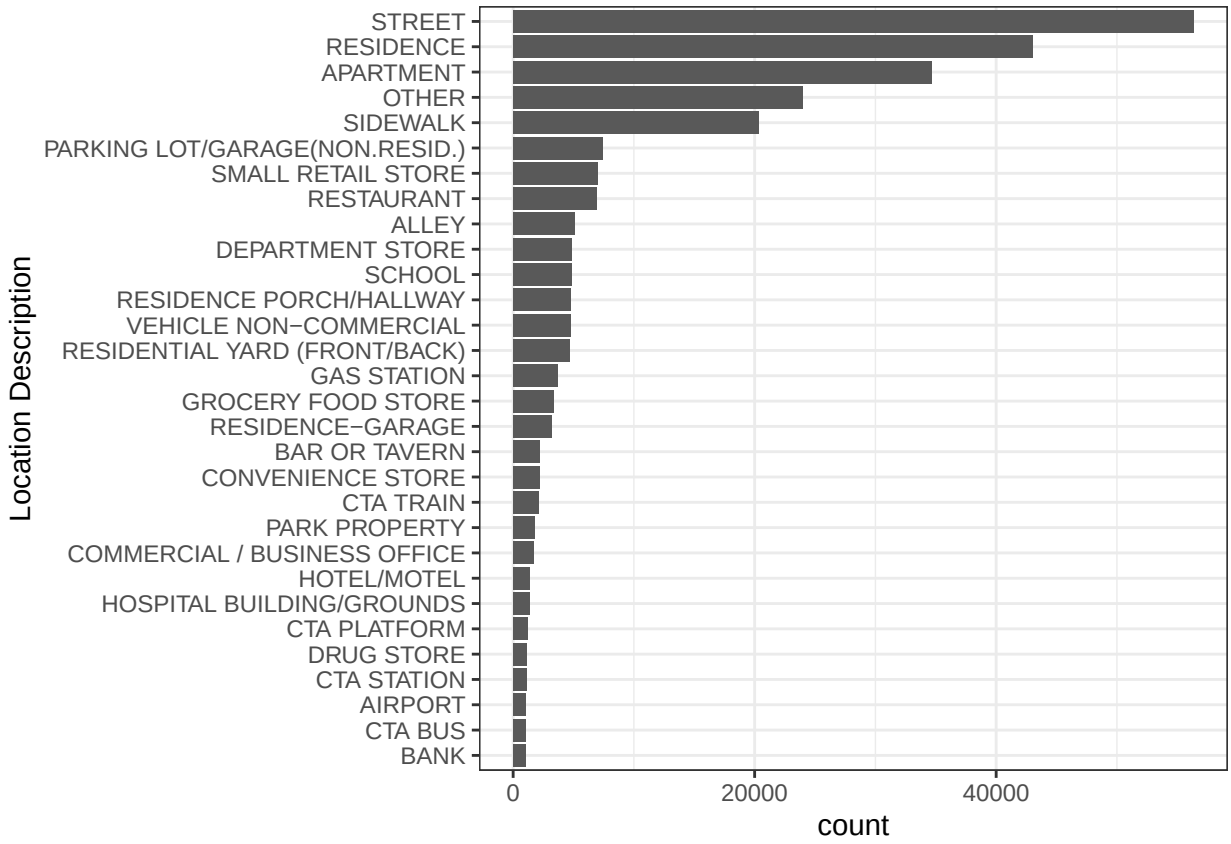
Location Description

BEEP move this to an EDA and just remove the variable earlier? We also wish to reduce the number of levels for the Location Descriptions in our data set. Initially, we have 156 location descriptions. Of these, 127 have less than 1000 observations. In order to reduce the computational complexity of our model, we merge these uncommon levels of our factor variable into the level OTHER

```
# Reordering factor levels (for plots and analysis purposes)
dat$`Location Description` <- fct_infreq(dat$`Location Description`) %>%
  fct_rev()

ggplot(dat) +
  geom_bar(aes(y = `Location Description`)) +
  geom_vline(xintercept=thr, colour="red")
```




```
dat <- regroup_locations(dat,thr)

dat$`Location Description` <- fct_infreq(dat$`Location Description`) %>% fct_rev()

ggplot(dat) +
  geom_bar(aes(y = `Location Description`))
```

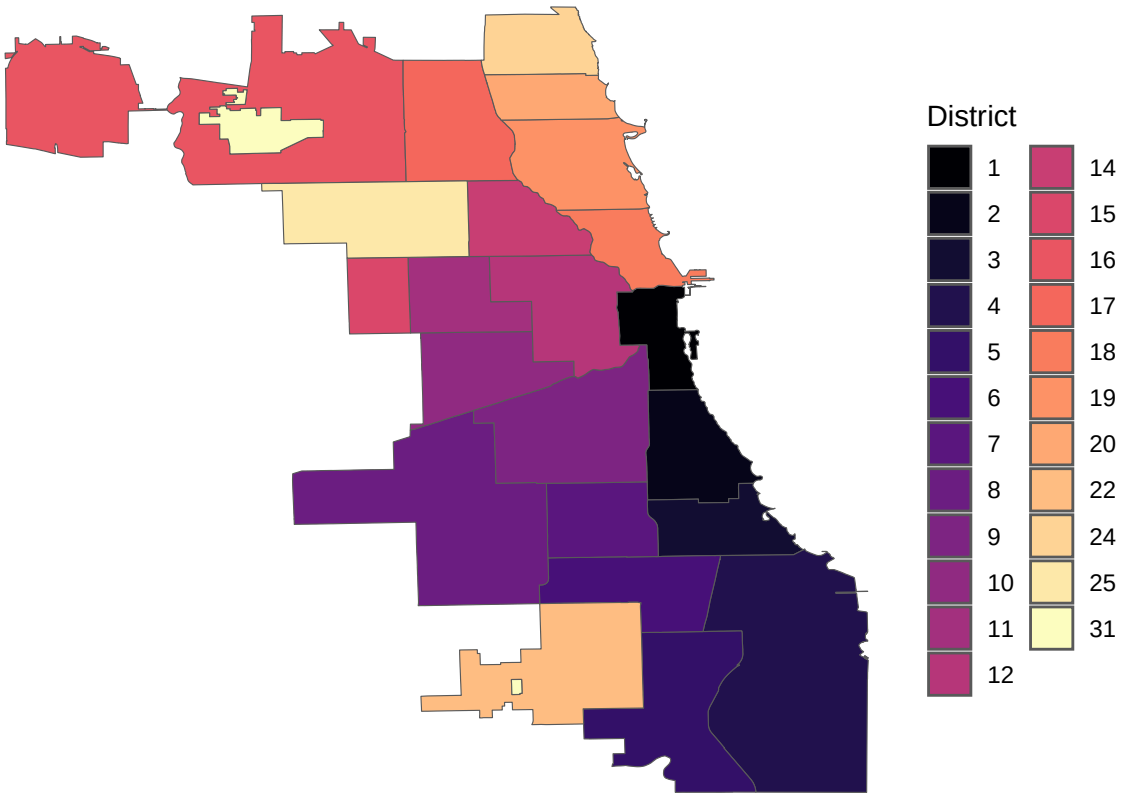


District

```
# Loading mapping data for plots
district_map <- RSocrata::read.socrata("https://data.cityofchicago.org/resource/24zt-jpfn.csv")
```

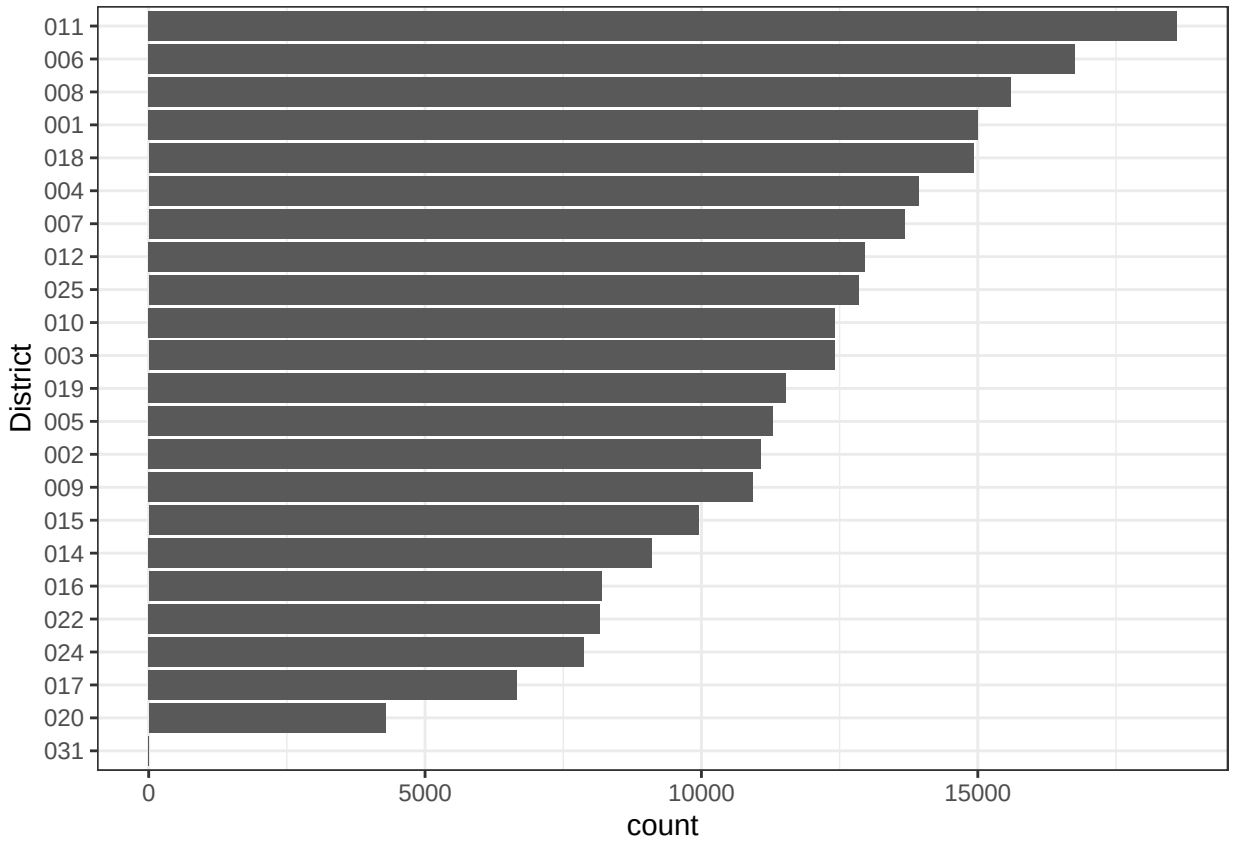
In order to include spatial information, we consider the District variable. We observe that all districts experience reasonable numbers of crimes to be incorporated into analysis, with the exception of District 031.

```
district_map <- district_map %>%
  select(c(the_geom, District = dist_num)) %>%
  st_as_sf(wkt = "the_geom")
plot_heat_map_d(district_map,"District")
```



This district is made up of multiple different areas, as can be seen in the geographical plots. In order to simplify analysis, we exclude the 7 crimes from District 031, and simply note that as a result our model will not be suitable for prediction in this district. As such we are left with 22 districts in our analysis.

```
# Reordering factor levels (for plots and analysis purposes)
dat$District <- dat$District %>%
  fct_infreq() %>%
  fct_rev()

ggplot(dat) +
  geom_bar(aes(y = District))
```




```
# Removing district 31 from analysis
dat <- dat[dat$District != "031",]
dat$District <- fct_drop(dat$District)
```

In order to reduce the computational complexity of our model, whilst still maintaining location information, we can also consider grouping districts further into policing district areas, of which there are 5.

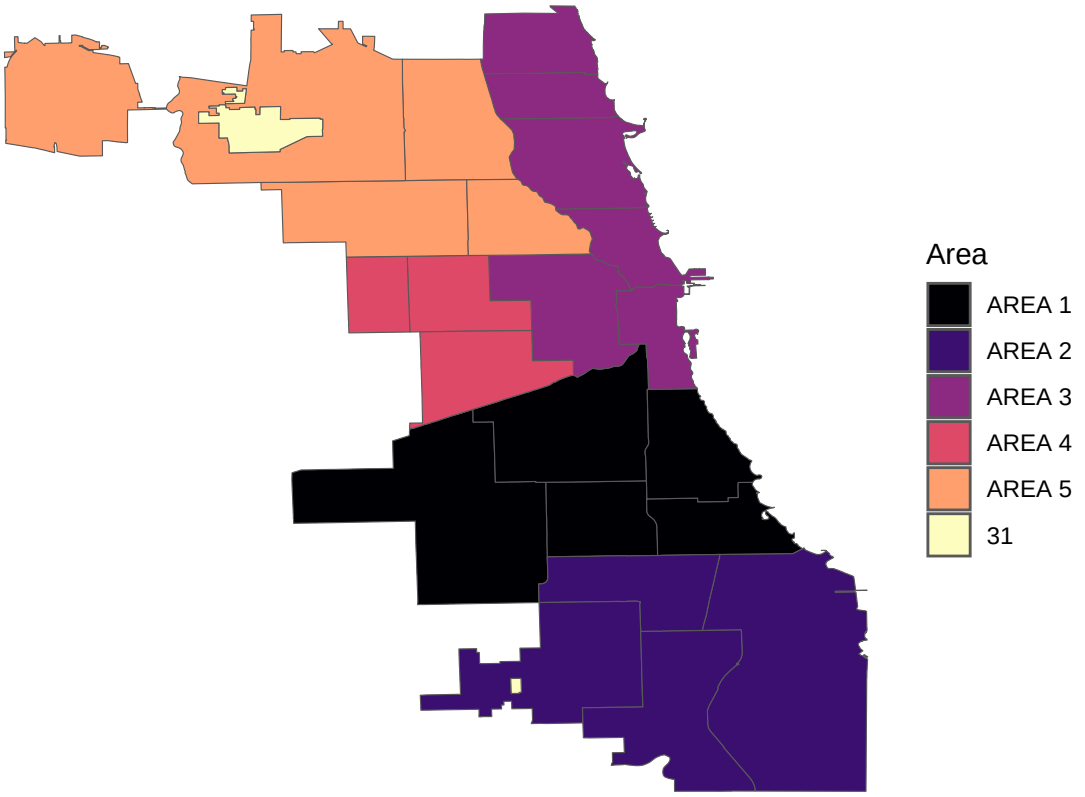
BEEP Source http://oururbantimes.com/district-stations/command-changes-12th-and-14th-chicago-police-department-districts

https://news.wttw.com/2020/04/30/chicago-police-opening-2-new-operation-areas-expand-resources-across-city

```
# Adding area groupings
area1 <- c("002","003","008","009","007")
area2 <- c("004","005","006","022")
area3 <- c("001","012","018","019","020","024")
area4 <- c("010","011","015")
area5 <- c("014","016","017","025")
dat$`District Area` <- fct_collapse(dat$District, "AREA 1" = area1, "AREA 2" = area2, "AREA 3" = area3,
  fct_relevel(sort)
```

```
district_map$Area <- fct_collapse(as.factor(district_map$District),
                                  "AREA 1" = as.character(as.integer(area1)),
                                  "AREA 2" = as.character(as.integer(area2)),
                                  "AREA 3" = as.character(as.integer(area3)),
                                  "AREA 4"= as.character(as.integer(area4)),
```

```
                                  "AREA 5" = as.character(as.integer(area5))) %>%
  fct_relevel(sort) %>%
  fct_relevel("31", after = Inf)

#Plot for just area
area_plot <- plot_heat_map_d(district_map,"Area")
area_plot
```



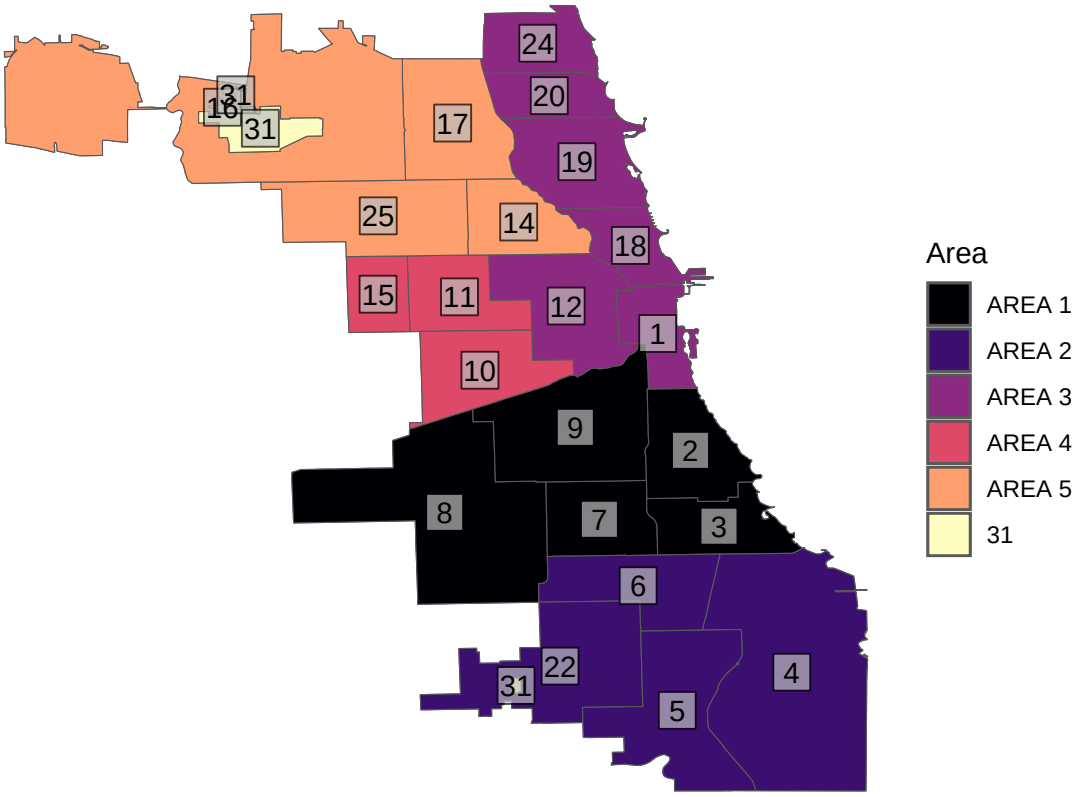

```
# Get district labels and locations
district_labels <- sort(as.integer(district_map$District))
district_markers <- district_map %>%
  arrange(as.integer(District)) %>%
  st_centroid() %>%
  st_coordinates() %>%
  as_tibble() %>%
  mutate(district_labels = district_labels)
```

```
## Warning in st_centroid.sf(.): st_centroid assumes attributes are constant over
## geometries of x
```

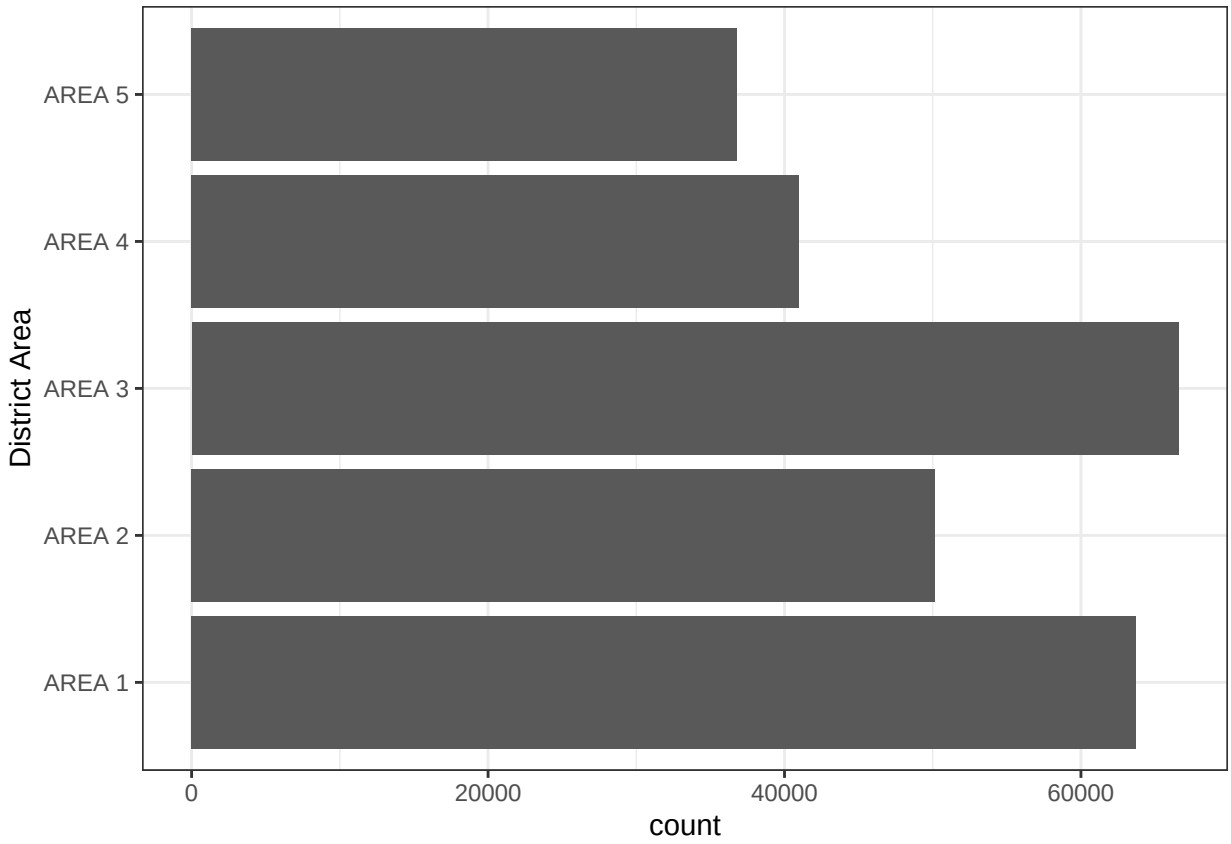
```
# Plot with district labels and coloured area: labels for 31 are a bit vom
area_plot_markers <- area_plot +
  geom_sf(data = district_map, aes(fill = Area)) +
    scale_fill_viridis_d(name = "Area",option = "magma") +
  geom_point(data = district_markers, aes(X, Y), size = 7, shape=22, fill = "Grey", alpha=0.7) +
  geom_text(data = district_markers, aes(X, Y, label = district_labels)) +
  theme_void()
```

```
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
```

```
area_plot_markers
```




```
# Plot showing number of crimes in each district
ggplot(dat) +
  geom_bar(aes(y = `District Area`))
```



As we are choosing to encode location in our model via district and district area, in order to reduce computational costs and reduce redundancy in our model we will not consider the X and Y coordinates so drop them from our analysis.

```
dat <- select(dat, -c(`X Coordinate`,`Y Coordinate`))
```

Time and Date

In order to incorporate temporal data, we include the date as the day of the year, `Day`, as well as time of day,`Time`. We standardize both of these variables to take values between 0 and 1. This is to ensure the convergence rate is not slow when we fit our model.

In order to facilitate our second model, which attempts to predict the time of day at which a crime occurs, we also add the variable `Time of Day` which encodes the time as a categorical variable with the following levels: * `MORNING`: between 5am and 12 noon * `AFTERNOON`: between 12 noon and 5pm * `EVENING`: between 5pm and 9pm * `NIGHT`: between 9pm and 5am

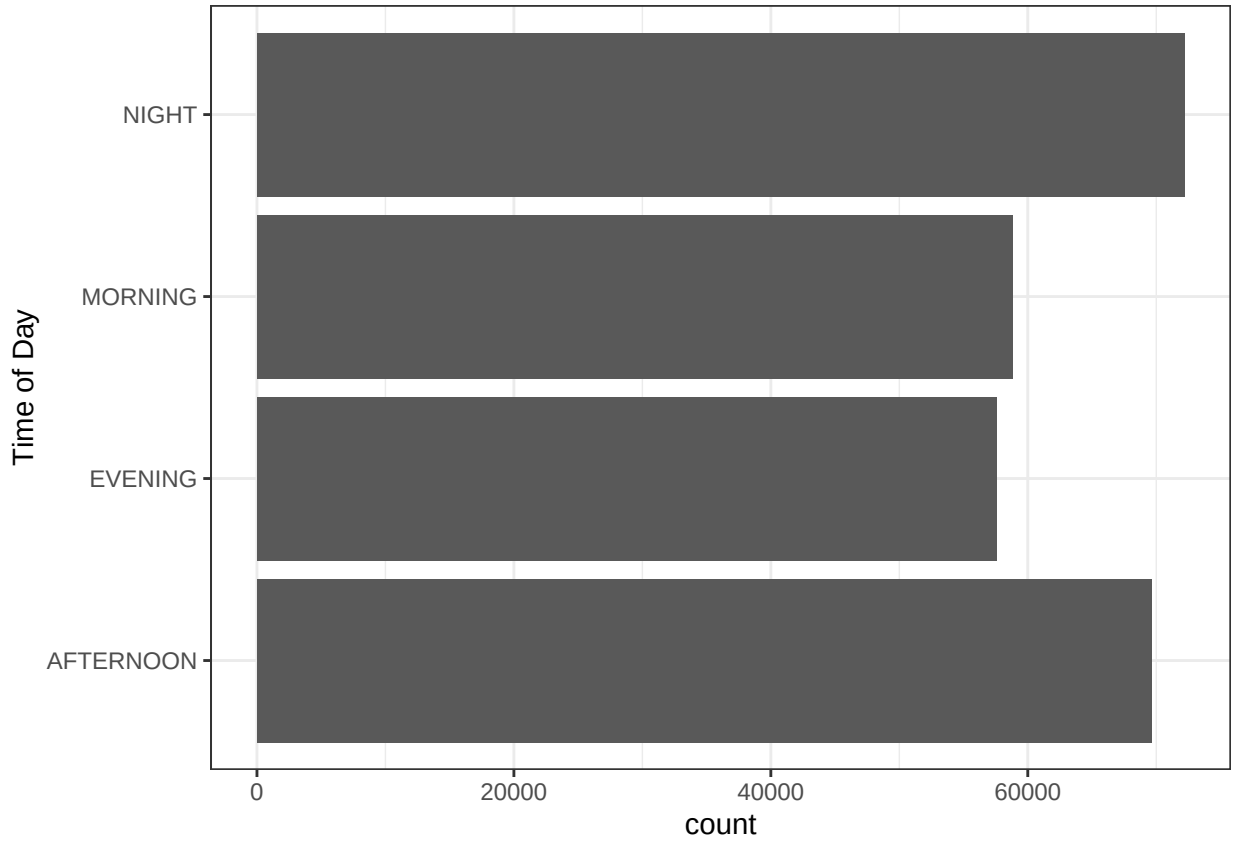
After this processing, we then remove the `Date` variable.

```
t <- lubridate::hour(dat$Date) + lubridate::minute(dat$Date)/60
tod <- rep("NIGHT",length(t))
for(i in 1:length(t)){
  if(5 <= t[i] & t[i] < 12){
    tod[i] <- "MORNING"
    }
  else if(12 <= t[i] & t[i] < 17){
    tod[i] <- "AFTERNOON"
```

```
    }
  else if(17 <= t[i] & t[i] < 21){
    tod[i] <- "EVENING"
    }
}
tod <- as.factor(tod)
```

```
dat <- dat %>%
  mutate(dat,
         Day = yday(dat$Date) / 365,
         Time = t/24,
         `Time of Day` = tod) %>%
  select(-Date)
```

```
# Plot showing number of crimes at times of day
ggplot(dat) +
  geom_bar(aes(y = `Time of Day`))
```



Multinomial Logistic Regression

We use the `multinom()` function from the `nnet` package in order to implement multinomial logistic regression. We calculate various model performance metrics using the `confusionMatrix()` function from the `caret` package. Namely we focus here on calculating overall accuracy and no-information rate (NIR), as well as class specific sensitivity, specification and accuracy values. We implement 5-fold repeated cross validation, with $n = 5$ repeats, and calculate the average of each metric across all folds for each repeat.

BEEP No-information rate is the best case scenario of what would happen if we just assigned all points the one class: we want our model to do better than that.

```
#setting seed for reprodcibility
set.seed(200899)
metrics <- c("Sensitivity", "Specificity", "Balanced Accuracy")
```

Predicting Crime Type

We begin by splitting our data into our response variable and the variables we want to include in our models. We look at three potential models: 1) No Location: Primary Type ~ Arrest + Domestic + Day + Time 2) District: Primary Type ~ Arrest + Domestic + Day + Time + District 3) District Area: Primary Type ~ Arrest + Domestic + Day + Time + District Area

```
# Splitting response data and different model variable sets
y <- dat$`Primary Type`
X_noloc <- dat %>% select(c(Arrest, Domestic, Day, Time))
X_area <- dat %>% select(c(Arrest, Domestic, Day, Time, `District Area`))
X_district <- dat %>% select(c(Arrest, Domestic, Day, Time, District))
```

We evaluate the model performance using the `mnlr_kfold_cv.df()` function, which outputs a data frame containing the metrics as evaluated for the model fitted on each fold for each repeat. Our package also contains the `mnlr_kfold_cv.list()` function, which performs the same operations and outputs a list based format of our values, as well as their averages. We use the data frame function for our analysis, however, as its output can be easily manipulated to calculate averages, investigate metric distributions. Further, the data frame outputs can be easily converted into an easy to read table, ideal for producing result documentation, which the list output cannot.

BEEP this has 80 variables, is that worth mentioning BEEP

```
mnlr_results_noloc.df <- mnlr_kfold_cv.df(X = X_noloc, y = y, k = 5, n_reps = 5, metrics = metrics)
```

BEEP this has 144 variables BEEP

```
mnlr_results_area.df <- mnlr_kfold_cv.df(X = X_area, y = y, k = 5, n_reps = 5, metrics = metrics)
```

BEEP this has 416 variables BEEP

```
mnlr_results_district.df <- mnlr_kfold_cv.df(X = X_district, y = y, k = 5, n_reps = 5, metrics = metrics
```

BEEP below loads from files on computer saved from prev run of code so that we can actually knit the pdf, variable names need to stay same BEEP

```
noloc_class_means <- mnlr_results_noloc.df$byclass %>%
  dplyr::group_by(Class,Repeat) %>%
  dplyr::summarise("Average Sensitivity" = mean(Sensitivity),
                   "Average Specificity" = mean(Specificity),
                   "Average Balanced Accuracy" = mean(`Balanced Accuracy`)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)
```

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

```
noloc_class_means
```

```
## # A tibble: 85 x 5
## # Groups:   Class [17]
##    Repeat Class                           `Average Sensitivity` Avera~1 Avera~2
##    <fct>  <fct>                                           <dbl>   <dbl>   <dbl>
##  1 1      ASSAULT                                         0       1       0.5
##  2 1      BATTERY                                         0.482   0.907   0.695
##  3 1      BURGLARY                                        0       1       0.5
##  4 1      CRIMINAL DAMAGE                                 0       1       0.5
##  5 1      CRIMINAL SEXUAL ASSAULT                         0       1       0.5
##  6 1      CRIMINAL TRESPASS                               0       1       0.5
##  7 1      DECEPTIVE PRACTICE                              0       1       0.5
##  8 1      INTERFERENCE WITH PUBLIC OFFICER                0       1       0.5
##  9 1      MOTOR VEHICLE THEFT                             0       1       0.5
## 10 1      NARCOTICS                                       0.999   0.862   0.931
## # ... with 75 more rows, and abbreviated variable names
## #   1: `Average Specificity`, 2: `Average Balanced Accuracy`
```

```
noloc_overall_means <- mnlr_results_noloc.df$overall %>%
  dplyr::group_by(Repeat) %>%
  dplyr::summarise("Average Overall Accuracy" = mean(Accuracy),
                   "Average NIR" = mean(NIR),
                   "Maximum P Value (Acc > NIR)" = max(Pval)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)

noloc_overall_means
```

```
## # A tibble: 5 x 4
##   Repeat `Average Overall Accuracy` `Average NIR` `Maximum P Value (Acc > NIR)`
##   <fct>                      <dbl>         <dbl>                          <dbl>
## 1 1                          0.357         0.239                              0
## 2 2                          0.357         0.239                              0
## 3 3                          0.357         0.239                              0
## 4 4                          0.357         0.239                              0
## 5 5                          0.357         0.239                              0
```

```
area_class_means <- mnlr_results_area.df$byclass %>%
  dplyr::group_by(Class,Repeat) %>%
  dplyr::summarise("Average Sensitivity" = mean(Sensitivity),
                   "Average Specificity" = mean(Specificity),
                   "Average Balanced Accuracy" = mean(`Balanced Accuracy`)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)
```

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

```
area_class_means
```

```
## # A tibble: 85 x 5
## # Groups:   Class [17]
##    Repeat Class                              `Average Sensitivity` Avera~1 Avera~2
##    <fct>  <fct>                                              <dbl>   <dbl>   <dbl>
##  1 1      ASSAULT                                            0       1       0.5
##  2 1      BATTERY                                            0.482   0.907   0.695
##  3 1      BURGLARY                                           0       1       0.5
##  4 1      CRIMINAL DAMAGE                                    0       1       0.5
##  5 1      CRIMINAL SEXUAL ASSAULT                            0       1       0.5
##  6 1      CRIMINAL TRESPASS                                  0       1       0.5
##  7 1      DECEPTIVE PRACTICE                                 0       1       0.5
##  8 1      INTERFERENCE WITH PUBLIC OFFICER                   0       1       0.5
##  9 1      MOTOR VEHICLE THEFT                                0       1       0.5
## 10 1      NARCOTICS                                          0.927   0.897   0.912
## # ... with 75 more rows, and abbreviated variable names
## #   1: `Average Specificity`, 2: `Average Balanced Accuracy`
```

```
area_overall_means <- mnlr_results_area.df$overall %>%
  dplyr::group_by(Repeat) %>%
  dplyr::summarise("Average Overall Accuracy" = mean(Accuracy),
                   "Average NIR" = mean(NIR),
                   "Maximum P Value (Acc > NIR)" = max(Pval)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)

area_overall_means
```

```
## # A tibble: 5 x 4
##   Repeat `Average Overall Accuracy` `Average NIR` `Maximum P Value (Acc > NIR)`
##   <fct>                       <dbl>         <dbl>                         <dbl>
## 1 1                           0.364         0.239                             0
## 2 2                           0.364         0.239                             0
## 3 3                           0.364         0.239                             0
## 4 4                           0.364         0.239                             0
## 5 5                           0.364         0.239                             0
```

```
district_class_means <- mnlr_results_district.df$byclass %>%
  dplyr::group_by(Class,Repeat) %>%
  dplyr::summarise("Average Sensitivity" = mean(Sensitivity),
                   "Average Specificity" = mean(Specificity),
                   "Average Balanced Accuracy" = mean(`Balanced Accuracy`)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)
```

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

```
district_class_means
```

```
## # A tibble: 85 x 5
## # Groups:   Class [17]
##    Repeat Class                                ‘Average Sensitivity‘ Avera~1 Avera~2
##    <fct>  <fct>                                                <dbl>   <dbl>   <dbl>
##  1 1      ASSAULT                                              0         1       0.5
##  2 1      BATTERY                                              0.482     0.907   0.695
##  3 1      BURGLARY                                             0         1       0.5
##  4 1      CRIMINAL DAMAGE                                      0.00104   1.00    0.500
##  5 1      CRIMINAL SEXUAL ASSAULT                              0         1       0.5
##  6 1      CRIMINAL TRESPASS                                    0         1       0.5
##  7 1      DECEPTIVE PRACTICE                                   0         1       0.5
##  8 1      INTERFERENCE WITH PUBLIC OFFICER                     0         1       0.5
##  9 1      MOTOR VEHICLE THEFT                                  0         1       0.5
## 10 1      NARCOTICS                                            0.916     0.903   0.909
## # ... with 75 more rows, and abbreviated variable names
## #   1: ‘Average Specificity‘, 2: ‘Average Balanced Accuracy‘
```

```
district_overall_means <- mnlr_results_district.df$overall %>%
  dplyr::group_by(Repeat) %>%
  dplyr::summarise("Average Overall Accuracy" = mean(Accuracy),
                   "Average NIR" = mean(NIR),
                   "Maximum P Value (Acc > NIR)" = max(Pval)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)

district_overall_means
```

```
## # A tibble: 5 x 4
##   Repeat ‘Average Overall Accuracy‘ ‘Average NIR‘ ‘Maximum P Value (Acc > NIR)‘
##   <fct>                      <dbl>         <dbl>                          <dbl>
## 1 1                          0.365         0.239                              0
## 2 2                          0.365         0.239                              0
## 3 3                          0.365         0.239                              0
## 4 4                          0.365         0.239                              0
## 5 5                          0.365         0.239                              0
```

BEEP interpretations of the above and get into latex tables to put in report? BEEP

BEEP think best model is using area because not much difference going to district and also has the advantage of lesser computational complexity so easier to run and interpret. Probably explain this in latex report not here? BEEP

```
set.seed(1)
ind <- sample(2,nrow(dat),replace=TRUE, prob = c(0.8,0.2))
index <- which(ind == 2)
mnlr <- mnlr_cv_indexed(X_area,y,index,return_model = TRUE)
```

BEEP load for knitting purposes BEEP

We can look at a subset of the predictive probabilities produced by the model, as well as the coefficient values, in order to interpret the relationships between our observed variables and the log odds of a particular class.

```
head(pp <- fitted(mnlr$model)) #Predictive probabilities
```

```
##   SEX OFFENSE PUBLIC PEACE VIOLATION INTERFERENCE WITH PUBLIC OFFICER
## 1 0.011119463            0.002132493                     0.0009884567
## 2 0.008029836            0.002705526                     0.0009820659
## 3 0.008409963            0.002607708                     0.0009879608
## 4 0.009681322            0.002418259                     0.0009864593
## 5 0.009492529            0.003291402                     0.0007724437
## 6 0.008317555            0.003644030                     0.0007806414
##   CRIMINAL SEXUAL ASSAULT OFFENSE INVOLVING CHILDREN WEAPONS VIOLATION
## 1             0.009732070               0.004521506      0.0024235410
## 2             0.006117134               0.003162024      0.0028839582
## 3             0.006587477               0.003370120      0.0028047923
## 4             0.007826802               0.003783111      0.0026693223
## 5             0.010156643               0.040388245      0.0003969399
## 6             0.008495572               0.035602653      0.0004276379
##   CRIMINAL TRESPASS     ROBBERY MOTOR VEHICLE THEFT    BURGLARY    NARCOTICS
## 1       0.013638497 0.032267608         0.030916348 0.039919221 6.711475e-04
## 2       0.012953862 0.036970891         0.038957033 0.027871158 8.351734e-04
## 3       0.013117388 0.036171843         0.037662853 0.029547183 8.106405e-04
## 4       0.013328606 0.034887269         0.034809589 0.033738019 7.481485e-04
## 5       0.004197899 0.003603915         0.002048627 0.006370582 3.501002e-05
## 6       0.004163433 0.003824412         0.002268672 0.005564621 3.869394e-05
##   DECEPTIVE PRACTICE      OTHER    ASSAULT CRIMINAL DAMAGE    BATTERY
## 1        0.146239261 0.03806218 0.04980537      0.08779482 0.10078118
## 2        0.094743498 0.03807443 0.06001749      0.08277294 0.09565066
## 3        0.102083189 0.03832307 0.05852711      0.08384583 0.09681585
## 4        0.118138645 0.03799385 0.05469648      0.08562850 0.09856129
## 5        0.004287115 0.11024438 0.10765331      0.09489853 0.55115416
## 6        0.003658950 0.11198900 0.11739220      0.09372311 0.54600675
##        THEFT
## 1 0.42898684
## 2 0.48727233
## 3 0.47832702
## 4 0.46010431
## 5 0.05100827
## 6 0.05410207
```

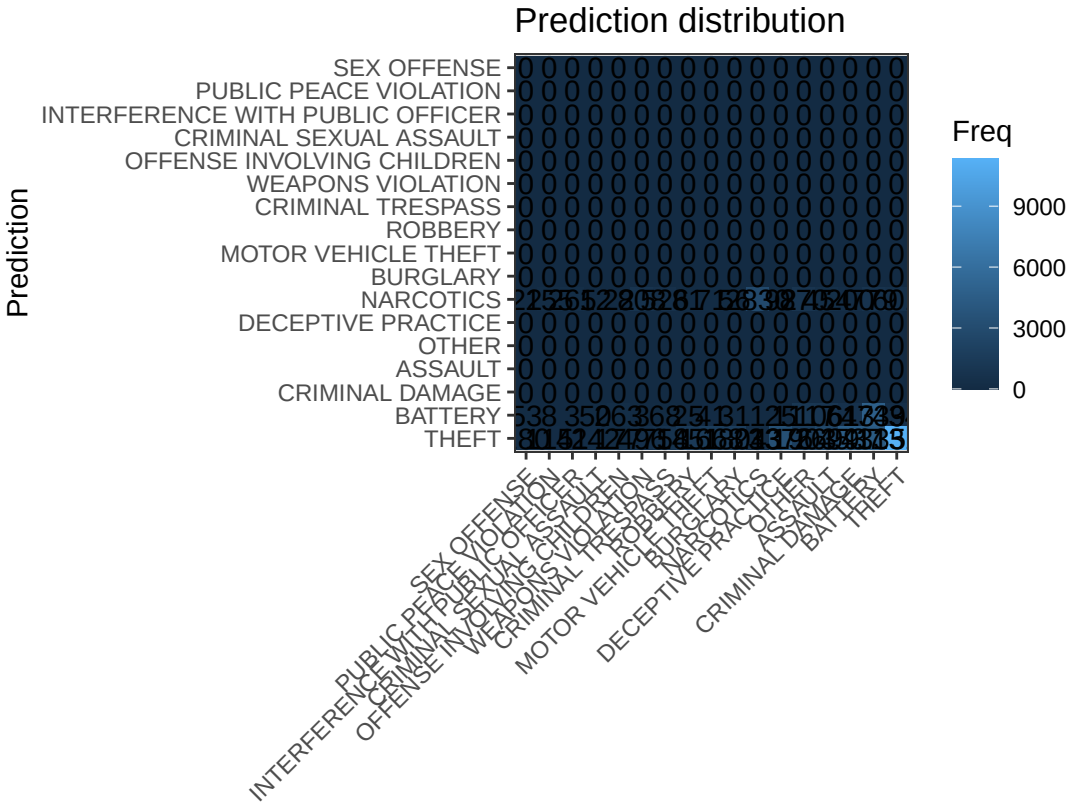
```
coef(mnlr$model)
```

```
##                                 (Intercept)  ArrestTRUE DomesticTRUE
## PUBLIC PEACE VIOLATION            0.2648455  1.54653787   -1.0594418
## INTERFERENCE WITH PUBLIC OFFICER  0.2321411  2.39386547   -2.2072318
## CRIMINAL SEXUAL ASSAULT           0.9713539 -1.02509442   -0.5513004
## OFFENSE INVOLVING CHILDREN        1.1748510 -0.15727155    0.9814242
## WEAPONS VIOLATION                 1.3267053  2.01013912   -4.2924775
## CRIMINAL TRESPASS                 1.1818599  1.58272507   -1.4896277
## ROBBERY                           1.9355198 -1.23174865   -2.6902777
## MOTOR VEHICLE THEFT               2.1946313 -1.73277621   -3.3512703
## BURGLARY                          3.0790985 -1.74385321   -3.0870144
## NARCOTICS                        -0.3525824  4.61182668   -4.7587363
## DECEPTIVE PRACTICE                3.4099079 -1.34958373   -3.5545608
## OTHER                             2.9615771  0.37466746    0.1141248
```

```
## ASSAULT                              2.9310484 -0.14669663   -0.0303311
## CRIMINAL DAMAGE                      3.7090526 -1.32427544   -0.9450071
## BATTERY                              3.5980171  0.04376494    0.9402851
## THEFT                                4.0422543 -0.91029475   -2.0352131
##                                           Day        Time `District Area`AREA 2
## PUBLIC PEACE VIOLATION           -0.5216230  1.13412585           -1.16923764
## INTERFERENCE WITH PUBLIC OFFICER -0.7932150  0.69184741           -0.47351662
## CRIMINAL SEXUAL ASSAULT          -0.4132570 -0.22519301           -0.22436840
## OFFENSE INVOLVING CHILDREN       -0.9086071  0.02917902           -0.33734661
## WEAPONS VIOLATION                -0.4288368  1.00183669           -0.42327058
## CRIMINAL TRESPASS                -0.7459413  0.60069156           -0.10913621
## ROBBERY                          -0.4233669  0.92858528           -0.21566300
## MOTOR VEHICLE THEFT              -0.6648842  1.13532965           -0.09757254
## BURGLARY                         -0.4964163 -0.01515222           -0.45514014
## NARCOTICS                        -0.8095997  1.12574772           -0.55017040
## DECEPTIVE PRACTICE               -0.8091673 -0.12752042           -0.29670249
## OTHER                            -0.9119890  0.71678987           -0.29837271
## ASSAULT                          -0.7747782  1.06056384           -0.34732904
## CRIMINAL DAMAGE                  -0.6833930  0.58023457           -0.29648670
## BATTERY                          -0.7050228  0.59516878           -0.50837109
## THEFT                            -0.5656971  0.92616397           -0.48288811
##                                  `District Area`AREA 3 `District Area`AREA 4
## PUBLIC PEACE VIOLATION                      -1.9127217            0.41678528
## INTERFERENCE WITH PUBLIC OFFICER            -2.3028221            0.37083869
## CRIMINAL SEXUAL ASSAULT                     -0.7272596            0.45878116
## OFFENSE INVOLVING CHILDREN                  -1.4197769            0.36535104
## WEAPONS VIOLATION                           -2.8704210            0.36312959
## CRIMINAL TRESPASS                           -0.6322297            0.45456850
## ROBBERY                                     -0.8699855            0.86561241
## MOTOR VEHICLE THEFT                         -1.0641190            0.83832239
## BURGLARY                                    -1.4327549           -0.02362803
## NARCOTICS                                   -2.2379025            1.69544803
## DECEPTIVE PRACTICE                          -0.1989425            0.26102900
## OTHER                                       -1.3028568            0.51710328
## ASSAULT                                     -1.2183883            0.38491291
## CRIMINAL DAMAGE                             -1.3362519            0.33594409
## BATTERY                                     -1.0764230            0.50367756
## THEFT                                       -0.2844351            0.19043542
##                                  `District Area`AREA 5
## PUBLIC PEACE VIOLATION                      -1.7494189
## INTERFERENCE WITH PUBLIC OFFICER            -1.6631295
## CRIMINAL SEXUAL ASSAULT                     -0.6569669
## OFFENSE INVOLVING CHILDREN                  -1.1405527
## WEAPONS VIOLATION                           -1.8198413
## CRIMINAL TRESPASS                           -0.6089211
## ROBBERY                                     -1.0134884
## MOTOR VEHICLE THEFT                         -0.6027693
## BURGLARY                                    -1.0048098
## NARCOTICS                                   -1.5693356
## DECEPTIVE PRACTICE                          -0.4646791
## OTHER                                       -1.0594323
## ASSAULT                                     -1.2252609
## CRIMINAL DAMAGE                             -0.8081295
## BATTERY                                     -1.0531236
```

```
## THEFT                                -0.6833454
```

```
# Extract confusion matrix for plot
tab <- mnlr$conf.matrix$table
confusion_df <- as.data.frame(tab)

confusion_plot <- ggplot(confusion_df, aes(x=Reference, y = Prediction, fill=Freq)) +
  coord_equal() +
  geom_tile() +
  labs(title = "Prediction distribution") +
  scale_y_discrete(limits = rev) +
  scale_x_discrete(expand = expansion(mult = c(0,0)), guide = guide_axis(angle = 45)) +
  geom_text(aes(label=Freq), color="black") # printing values
confusion_plot
```



In the confusion matrix, we observe that the model only makes predictions for the categories "NARCOTICS", "BATTERY" and "THEFT". We note that, from earlier exploratory analysis, "BATTERY" and "THEFT" are the two most common crimes, however "NARCOTICS" is a less common crime. BEEP this is interesting but idk what to say about it if u have any ideas feel free BEEP

Predicting Time of Crime

As an extension to this multiclass classification task, we will also evaluate the performance of another multinomial regression classifier used to predict the time of day at which a crime occurred. We use the same methods as seen previously but now time of day is response variable and we exclude Time from our model. This gives us the three models: 1) No Location: Time of Day ~ Primary Type + Arrest + Domestic + Day

2) District: Time of Day ~ Primary Type + Arrest + Domestic + Day + District 3) District Area: Time of Day ~ Primary Type + Arrest + Domestic + Day + District Area

```
#setting seed for reproducibility
set.seed(200899)
# Splitting response data and different model variable sets
y_tod <- dat$`Time of Day`
X_tod_noloc <- dat %>% select(c(`Primary Type`,Arrest,Domestic,Day))
X_tod_area <- dat %>% select(c(`Primary Type`,Arrest,Domestic,Day,`District Area`))
X_tod_district <- dat %>% select(c(`Primary Type`,Arrest,Domestic,Day,District))
```

BEEP this has 60 variables and converges (but other models none converged) BEEP

```
tod_results_noloc <- mnlr_kfold_cv.df(X = X_tod_noloc, y = y_tod, k = 5, n_reps = 5, metrics = metrics)
```

BEEP this has 72 variables and converges BEEP

```
tod_results_area <- mnlr_kfold_cv.df(X = X_tod_area, y = y_tod, k = 5, n_reps = 5, metrics = metrics)
```

BEEP this has 123 variables BEEP

```
tod_results_district <- mnlr_kfold_cv.df(X = X_tod_district, y = y_tod, k = 5, n_reps = 5, metrics = me
```

BEEP need to run the above and save then find best and do plots as before BEEP

```
tod_noloc_class_means <- tod_results_noloc$byclass %>%
  dplyr::group_by(Class,Repeat) %>%
  dplyr::summarise("Average Sensitivity" = mean(Sensitivity),
                   "Average Specificity" = mean(Specificity),
                   "Average Balanced Accuracy" = mean(`Balanced Accuracy`)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)
```

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

```
tod_noloc_class_means
```

```
## # A tibble: 20 x 5
## # Groups:   Class [4]
##    Repeat Class     `Average Sensitivity` `Average Specificity` Average Balanc~1
##    <fct>  <fct>                     <dbl>                 <dbl>            <dbl>
##  1 1      AFTERNOON                 0.514                 0.598            0.556
##  2 1      EVENING                   0.0761                0.947            0.512
##  3 1      MORNING                   0.0829                0.940            0.512
##  4 1      NIGHT                     0.572                 0.605            0.589
##  5 2      AFTERNOON                 0.510                 0.602            0.556
##  6 2      EVENING                   0.0761                0.947            0.512
##  7 2      MORNING                   0.0886                0.936            0.512
##  8 2      NIGHT                     0.572                 0.605            0.589
##  9 3      AFTERNOON                 0.514                 0.599            0.556
```

```
## 10 3      EVENING                 0.0729                 0.949           0.511
## 11 3      MORNING                 0.0864                 0.937           0.512
## 12 3      NIGHT                   0.573                  0.605           0.589
## 13 4      AFTERNOON               0.510                  0.602           0.556
## 14 4      EVENING                 0.0761                 0.947           0.512
## 15 4      MORNING                 0.0885                 0.936           0.512
## 16 4      NIGHT                   0.572                  0.605           0.589
## 17 5      AFTERNOON               0.510                  0.602           0.556
## 18 5      EVENING                 0.0761                 0.947           0.512
## 19 5      MORNING                 0.0890                 0.936           0.512
## 20 5      NIGHT                   0.572                  0.606           0.589
## # ... with abbreviated variable name 1: ‘Average Balanced Accuracy‘
```

```
tod_noloc_overall_means <- tod_results_noloc$overall %>%
  dplyr::group_by(Repeat) %>%
  dplyr::summarise("Average Overall Accuracy" = mean(Accuracy),
                   "Average NIR" = mean(NIR),
                   "Maximum P Value (Acc > NIR)" = max(Pval)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)

tod_noloc_overall_means
```

```
## # A tibble: 5 x 4
##   Repeat ‘Average Overall Accuracy‘ ‘Average NIR‘ ‘Maximum P Value (Acc > NIR)‘
##   <fct>                        <dbl>         <dbl>                         <dbl>
## 1 1                            0.335         0.280                     6.45e-137
## 2 2                            0.335         0.280                     2.13e-152
## 3 3                            0.335         0.280                     3.42e-153
## 4 4                            0.335         0.280                     1.60e-154
## 5 5                            0.335         0.280                     7.85e-154
```

```
tod_area_class_means <- tod_results_area$byclass %>%
  dplyr::group_by(Class,Repeat) %>%
  dplyr::summarise("Average Sensitivity" = mean(Sensitivity),
                   "Average Specificity" = mean(Specificity),
                   "Average Balanced Accuracy" = mean(`Balanced Accuracy`)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)
```

```
## ‘summarise()‘ has grouped output by ’Class’. You can override using the
## ‘.groups‘ argument.
```

```
tod_area_class_means
```

```
## # A tibble: 20 x 5
## # Groups:   Class [4]
##    Repeat Class     ‘Average Sensitivity‘ ‘Average Specificity‘ Average Balanc~1
##    <fct>  <fct>                     <dbl>                 <dbl>            <dbl>
##  1 1      AFTERNOON                 0.521                 0.595            0.558
##  2 1      EVENING                   0.0646                0.957            0.511
##  3 1      MORNING                   0.0862                0.937            0.512
```

```
##  4 1      NIGHT                  0.575                  0.602           0.588
##  5 2      AFTERNOON              0.520                  0.596           0.558
##  6 2      EVENING                0.0655                 0.955           0.510
##  7 2      MORNING                0.0849                 0.938           0.511
##  8 2      NIGHT                  0.575                  0.602           0.588
##  9 3      AFTERNOON              0.520                  0.595           0.558
## 10 3      EVENING                0.0640                 0.956           0.510
## 11 3      MORNING                0.0856                 0.937           0.511
## 12 3      NIGHT                  0.575                  0.601           0.588
## 13 4      AFTERNOON              0.520                  0.595           0.558
## 14 4      EVENING                0.0657                 0.955           0.510
## 15 4      MORNING                0.0851                 0.938           0.511
## 16 4      NIGHT                  0.574                  0.602           0.588
## 17 5      AFTERNOON              0.522                  0.593           0.558
## 18 5      EVENING                0.0617                 0.958           0.510
## 19 5      MORNING                0.0856                 0.938           0.512
## 20 5      NIGHT                  0.575                  0.602           0.589
## # ... with abbreviated variable name 1: `Average Balanced Accuracy`
```

```
tod_area_overall_means <- tod_results_area$overall %>%
  dplyr::group_by(Repeat) %>%
  dplyr::summarise("Average Overall Accuracy" = mean(Accuracy),
                   "Average NIR" = mean(NIR),
                   "Maximum P Value (Acc > NIR)" = max(Pval)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)

tod_area_overall_means
```

```
## # A tibble: 5 x 4
##   Repeat `Average Overall Accuracy` `Average NIR` `Maximum P Value (Acc > NIR)`
##   <fct>                      <dbl>         <dbl>                        <dbl>
## 1 1                          0.335         0.280                    3.28e-159
## 2 2                          0.335         0.280                    5.54e-147
## 3 3                          0.335         0.280                    2.29e-144
## 4 4                          0.335         0.280                    5.76e-154
## 5 5                          0.335         0.280                    2.25e-142
```

```
tod_district_class_means <- tod_results_district$byclass %>%
  dplyr::group_by(Class,Repeat) %>%
  dplyr::summarise("Average Sensitivity" = mean(Sensitivity),
                   "Average Specificity" = mean(Specificity),
                   "Average Balanced Accuracy" = mean(`Balanced Accuracy`)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)
```

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

```
tod_district_class_means
```

```
## # A tibble: 20 x 5
```

```
## # Groups:   Class [4]
##    Repeat Class     `Average Sensitivity` `Average Specificity` Average Balanc~1
##    <fct>  <fct>                     <dbl>                 <dbl>            <dbl>
##  1 1      AFTERNOON                0.538                 0.582            0.560
##  2 1      EVENING                  0.0479                0.968            0.508
##  3 1      MORNING                  0.0865                0.935            0.511
##  4 1      NIGHT                    0.572                 0.606            0.589
##  5 2      AFTERNOON                0.539                 0.581            0.560
##  6 2      EVENING                  0.0470                0.968            0.508
##  7 2      MORNING                  0.0866                0.934            0.511
##  8 2      NIGHT                    0.572                 0.607            0.589
##  9 3      AFTERNOON                0.538                 0.581            0.559
## 10 3      EVENING                  0.0462                0.969            0.508
## 11 3      MORNING                  0.0849                0.935            0.510
## 12 3      NIGHT                    0.573                 0.605            0.589
## 13 4      AFTERNOON                0.538                 0.580            0.559
## 14 4      EVENING                  0.0458                0.969            0.508
## 15 4      MORNING                  0.0850                0.935            0.510
## 16 4      NIGHT                    0.573                 0.605            0.589
## 17 5      AFTERNOON                0.537                 0.583            0.560
## 18 5      EVENING                  0.0492                0.967            0.508
## 19 5      MORNING                  0.0862                0.935            0.511
## 20 5      NIGHT                    0.573                 0.605            0.589
## # ... with abbreviated variable name 1: `Average Balanced Accuracy`
```

```
tod_district_overall_means <- tod_results_district$overall %>%
  dplyr::group_by(Repeat) %>%
  dplyr::summarise("Average Overall Accuracy" = mean(Accuracy),
                   "Average NIR" = mean(NIR),
                   "Maximum P Value (Acc > NIR)" = max(Pval)) %>%
  dplyr::relocate(Repeat) %>%
  dplyr::arrange(Repeat)

tod_district_overall_means
```

```
## # A tibble: 5 x 4
##   Repeat `Average Overall Accuracy` `Average NIR` `Maximum P Value (Acc > NIR)`
##   <fct>                      <dbl>         <dbl>                        <dbl>
## 1 1                          0.336         0.280                    1.43e-156
## 2 2                          0.335         0.280                    8.95e-157
## 3 3                          0.335         0.280                    1.82e-154
## 4 4                          0.335         0.280                    1.19e-144
## 5 5                          0.336         0.280                    5.00e-149
```

Each of the models have similar overall accuracy across our repeats, so we prefer the simplest model with the fewest variables as this has minimal computational cost to run. BEEP this is super arbitrary BEEP

```
set.seed(1)
ind <- sample(2,nrow(dat),replace=TRUE, prob = c(0.8,0.2))
index <- which(ind == 2)
tod_mnlr <- mnlr_cv_indexed(X_tod_noloc,y_tod,index,return_model = TRUE)
```

BEEP load for knitting purposes BEEP

```
head(tod_pp <- fitted(tod_mnlr$model)) #Predictive probabilities for each obs
```

```
##   AFTERNOON   EVENING   MORNING     NIGHT
## 1 0.3741678 0.1586889 0.2933461 0.1737972
## 2 0.3260413 0.2362343 0.2233024 0.2144220
## 3 0.3262113 0.2362129 0.2234191 0.2141567
## 4 0.3118771 0.2266271 0.2351205 0.2263753
## 5 0.1620191 0.1945690 0.2305151 0.4128968
## 6 0.2087007 0.1876580 0.2204456 0.3831956
```

```
coef(tod_mnlr$model)
```

```
##         (Intercept) `Primary Type`PUBLIC PEACE VIOLATION
## EVENING -0.44667170                          0.05467955
## MORNING -0.10915051                         -0.42119011
## NIGHT    0.00862079                         -0.29893268
##         `Primary Type`INTERFERENCE WITH PUBLIC OFFICER
## EVENING                                      0.5771519
## MORNING                                     -0.3109810
## NIGHT                                        0.3186401
##         `Primary Type`CRIMINAL SEXUAL ASSAULT
## EVENING                             0.2940906
## MORNING                             0.1682333
## NIGHT                               1.0533133
##         `Primary Type`OFFENSE INVOLVING CHILDREN
## EVENING                               0.02339865
## MORNING                              -0.19449155
## NIGHT                                -0.17466695
##         `Primary Type`WEAPONS VIOLATION `Primary Type`CRIMINAL TRESPASS
## EVENING                       0.6257313                       0.2301753
## MORNING                      -0.2119963                       0.1161020
## NIGHT                         0.8569053                      -0.1307179
##         `Primary Type`ROBBERY `Primary Type`MOTOR VEHICLE THEFT
## EVENING             0.4943424                          0.5434724
## MORNING            -0.1295694                          0.2266036
## NIGHT               0.5480449                          0.3669914
##         `Primary Type`BURGLARY `Primary Type`NARCOTICS
## EVENING             0.09226904             0.26364855
## MORNING             0.28886892             0.03977218
## NIGHT               0.02337576            -0.65863876
##         `Primary Type`DECEPTIVE PRACTICE `Primary Type`OTHER
## EVENING                       -0.4315074          0.17830023
## MORNING                       -0.1341563          0.02557764
## NIGHT                         -0.8341484         -0.27541349
##         `Primary Type`ASSAULT `Primary Type`CRIMINAL DAMAGE
## EVENING             0.1054950                      0.5178830
## MORNING            -0.1733129                      0.1505362
## NIGHT              -0.3919257                      0.4972346
##         `Primary Type`BATTERY `Primary Type`THEFT  ArrestTRUE DomesticTRUE
## EVENING             0.2285299           0.1025982  0.18768317    0.0899883
## MORNING            -0.1473166          -0.2692989 -0.07696303    0.3112646
## NIGHT               0.1693951          -0.4905890  0.14746034    0.3667383
```

```
##                    Day
## EVENING   2.791531e-02
## MORNING -6.089331e-05
## NIGHT     8.026164e-02
```

```r
# Extract confusion matrix for plot
tod_tab <- tod_mnlr$conf.matrix$table
tod_confusion_df <- as.data.frame(tod_tab)

tod_confusion_plot <- ggplot(tod_confusion_df, aes(x=Reference, y = Prediction, fill=Freq)) +
  coord_equal() +
  geom_tile() +
  labs(title = "Prediction distribution") +
  scale_y_discrete(limits = rev) +
  scale_x_discrete(expand = expansion(mult = c(0,0)), guide = guide_axis(angle = 45)) +
  geom_text(aes(label=Freq), color="black") # printing values
tod_confusion_plot
```