DATA 621 - Business Analytics and Data Mining

Fall 2020 - Group 2 - Homework #3

Avraham Adler, Samantha Deokinanan, Amber Ferger, John Kellogg, Bryan Persaud, Jeff Shamp

10/28/2020

DATA EXPLORATION

Data Description

The training data set contains 466 records summarizing attributes of various neighborhoods in the city of Boston. The response variable is coded such that it is `1` when the neighborhood's crime rate is above the median and `0` when it is not. In all, there are 12 predictors. These include:

Predictor Variables	Description
zn	proportion of residential land zoned for large lots (over 25000 square feet)
indus	proportion of non-retail business acres per suburb
chas	a dummy var. for whether the suburb borders the Charles River (1) or not (0)
nox	nitrogen oxides concentration (parts per 10 million)
rm	average number of rooms per dwelling
age	proportion of owner-occupied units built prior to 1940
dis	weighted mean of distances to five Boston employment centers
rad	index of accessibility to radial highways
tax	full-value property-tax rate per $10,000
ptratio	pupil-teacher ratio by town
lstat	lower status of the population (percent)
medv	median value of owner-occupied homes in $1000s

We can make some initial observaations. The data set has complete cases, thus, there is no need for imputation. Based on some common summary statistics, there are more observations where the crime rate is below the median. It is already apparent that some of the predictors varies depending the crime rate. For instance, there is a noticeable difference in the means of `age`, `lstat`, `rad`, and `zn` between the crime rate groups.

Table 2: Descriptive Statistics: Crime Rate > Median

	n	mean	sd	median	trimmed	min	max	range	skew	kurtosis	se
zn	237	21.48	29.17	0.00	16.31	0.00	100.00	100.00	1.20	0.16	1.89
indus	237	7.04	5.50	5.64	6.15	0.46	27.74	27.28	1.99	4.60	0.36
chas	237	0.05	0.22	0.00	0.00	0.00	1.00	1.00	4.07	14.65	0.01
nox	237	0.47	0.06	0.45	0.47	0.39	0.62	0.23	0.72	-0.40	0.00
rm	237	6.40	0.56	6.32	6.35	5.09	8.07	2.98	0.77	0.36	0.04
age	237	50.84	25.79	47.40	50.27	2.90	100.00	97.10	0.20	-1.06	1.68
dis	237	5.08	2.07	5.12	4.96	1.67	12.13	10.46	0.48	-0.11	0.13
rad	237	4.17	1.59	4.00	4.19	1.00	8.00	7.00	-0.11	-0.25	0.10
tax	237	308.75	89.20	293.00	299.86	187.00	711.00	524.00	1.93	6.43	5.79

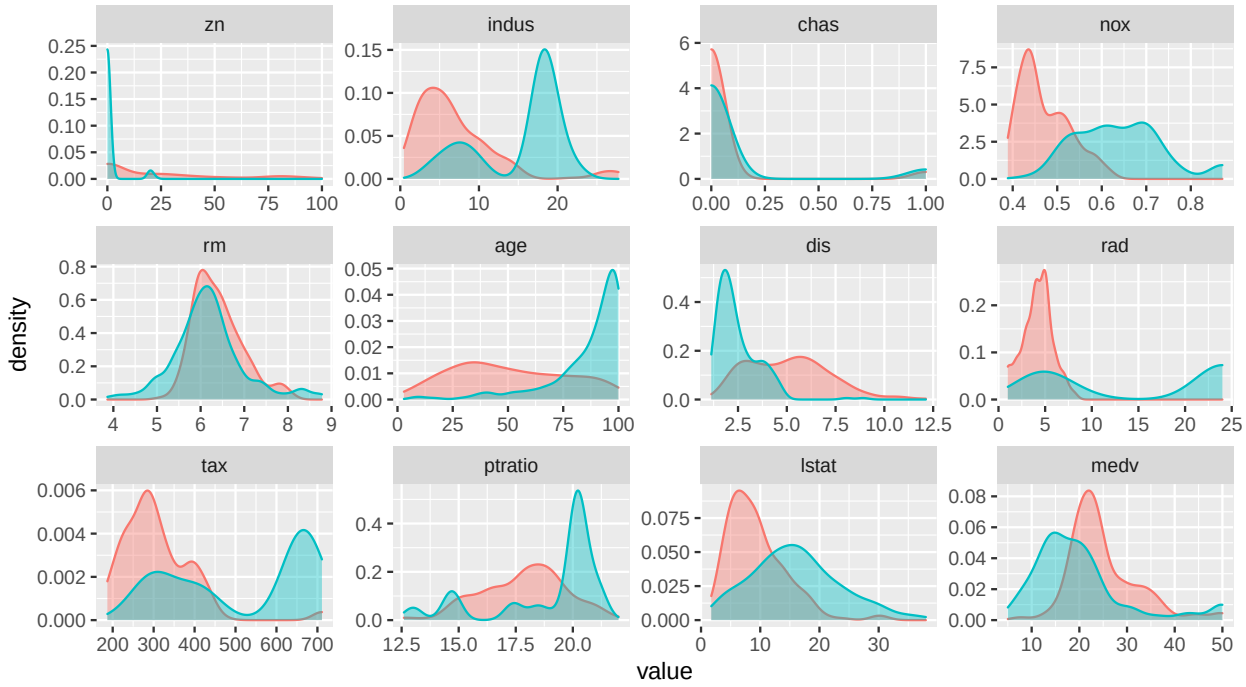
	n	mean	sd	median	trimmed	min	max	range	skew	kurtosis	se
ptratio	237	17.86	1.83	17.90	17.89	12.60	22.00	9.40	-0.30	-0.21	0.12
lstat	237	9.36	4.89	8.43	8.81	1.98	30.81	28.83	1.38	2.88	0.32
medv	237	25.04	7.34	23.10	24.23	7.00	50.00	43.00	1.20	1.85	0.48

Table 3: Descriptive Statistics: Crime Rate < Median

	n	mean	sd	median	trimmed	min	max	range	skew	kurtosis	se
zn	229	1.33	5.03	0.00	0.00	0.00	22.00	22.00	3.50	10.30	0.33
indus	229	15.31	5.41	18.10	15.88	3.97	21.89	17.92	-0.93	-0.75	0.36
chas	229	0.09	0.29	0.00	0.00	0.00	1.00	1.00	2.81	5.93	0.02
nox	229	0.64	0.10	0.62	0.63	0.43	0.87	0.44	0.50	-0.07	0.01
rm	229	6.18	0.82	6.13	6.14	3.86	8.78	4.92	0.58	1.41	0.05
age	229	86.50	17.26	92.60	90.04	8.40	100.00	91.60	-2.08	4.69	1.14
dis	229	2.47	1.08	2.12	2.34	1.13	8.91	7.78	1.95	6.97	0.07
rad	229	15.07	9.51	24.00	15.34	3.00	24.00	21.00	-0.14	-1.96	0.63
tax	229	513.77	166.69	666.00	524.27	223.00	666.00	443.00	-0.29	-1.74	11.02
ptratio	229	18.96	2.40	20.20	19.32	13.00	21.20	8.20	-1.33	0.35	0.16
lstat	229	16.02	7.45	15.39	15.71	1.73	37.97	36.24	0.41	-0.11	0.49
medv	229	20.05	10.28	17.80	18.41	5.00	50.00	45.00	1.50	2.01	0.68

Data Distribution

For each predictors, we computed and drew kernel density estimate to understand their distribution. Using density plots, we show how predictors are distributed between areas where crime rate is higher than the median, i.e. blue, and areas where crime rate is below the median, i.e. red. Our focus is to understand variables that highlight large variations between the two groups.



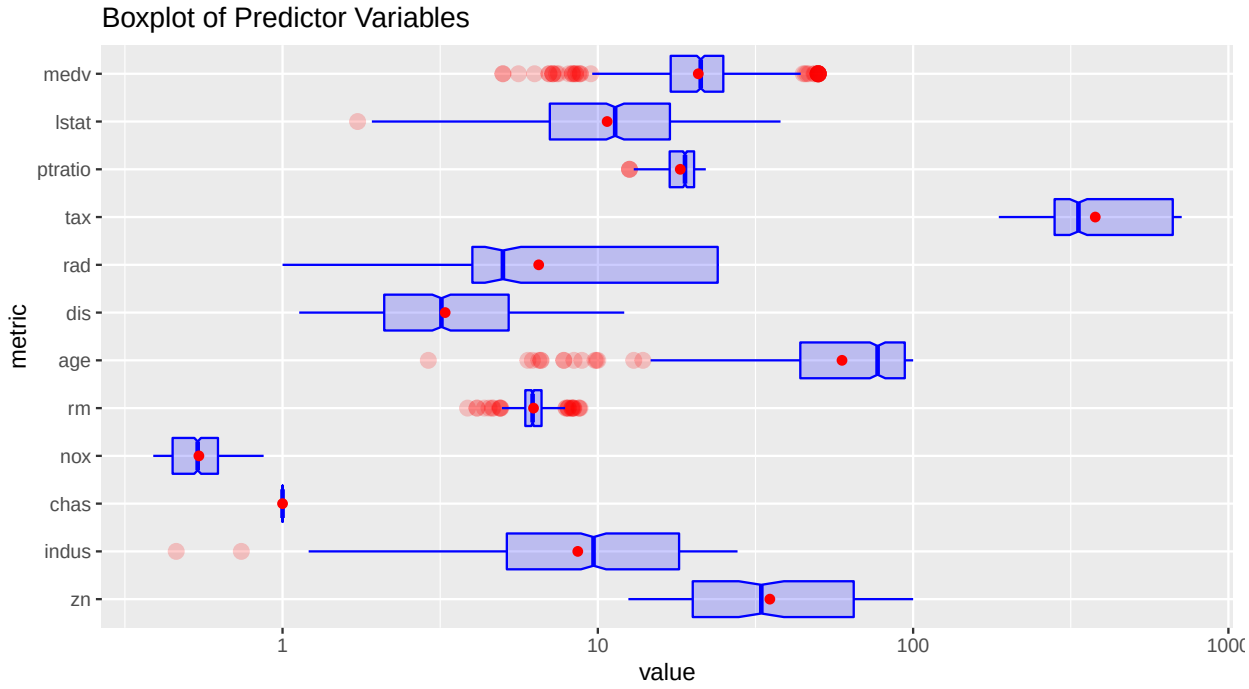
The plots reveal that most of the data does not have a normal distribution. As expected, we see the heavily

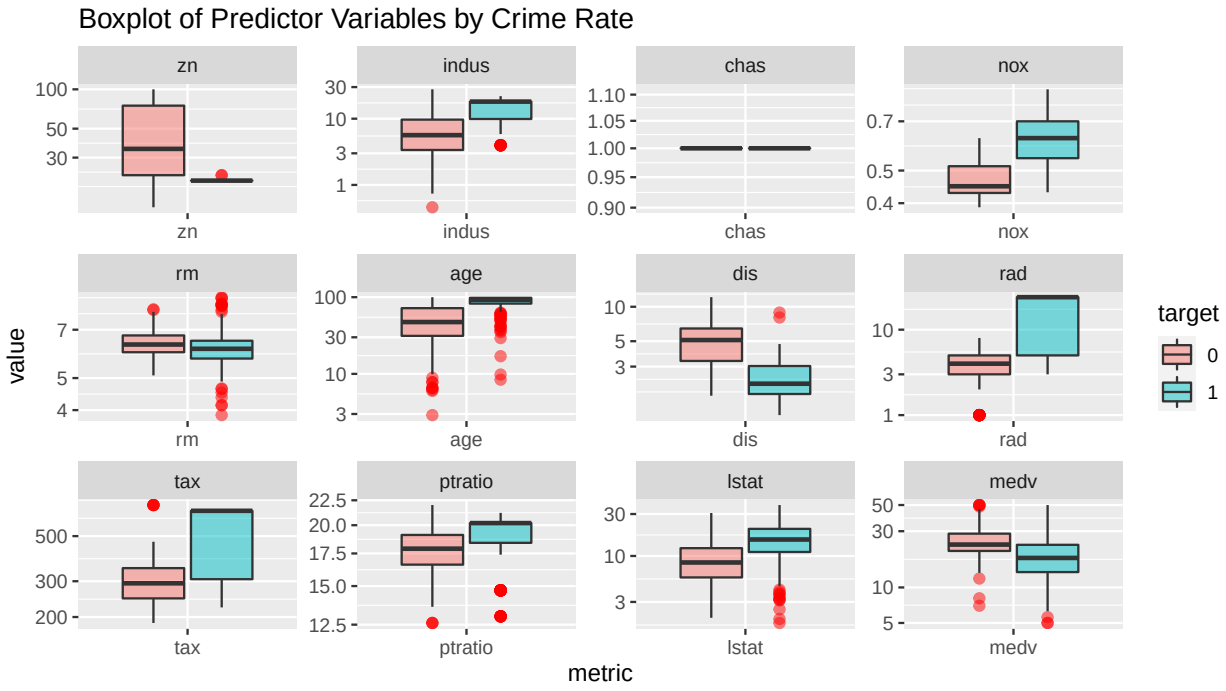
right tailed variables and others, which are multi-modal. The most Gaussian of the variables appears to be the one related to the average number of rooms per dwelling: `rm`.

Another interesting predictor, `zn` i.e. proportion of residential land zoned for large lots, has a significant positive skew (skew = 2.18). Nearly 73% of the observations (339 of 466 total) have a value of zero. When analyzing the difference between the crime rate groups, it is possible that areas with high crime rate do not have land zoned for large lots suggesting suburban areas are likely to have low crime rates because there typically have large lots, whereas urban areas have a higher crime rate since lot size are smaller than 25000 square feet.

Lastly, a third highly skewed variable is `chas`, the indicator as to whether the lot borders the Charles River. Out of the 466 observations, 433 do **not** border the river.

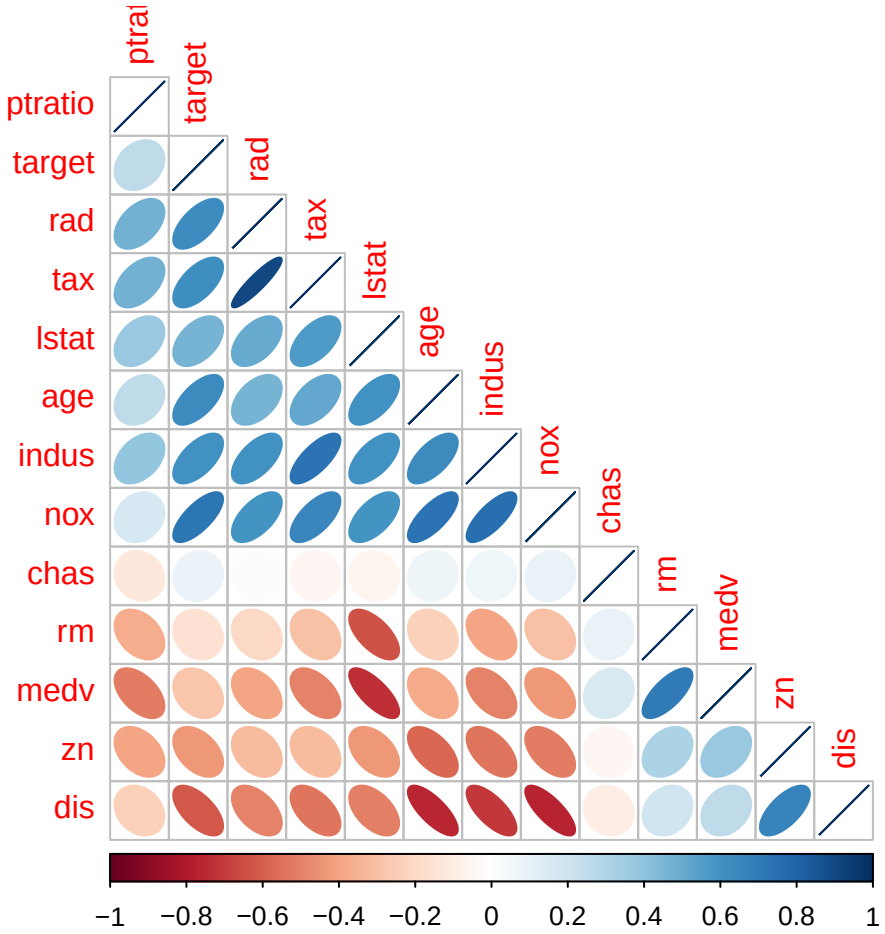
To provide another view, this time highlighting outliers, we present the data and analyze using boxplots.





It is clear from the second boxplot, similar to the densities, that the distribution of the predictor variables is different for for two outcomes. This suggests that a model will be able to extract signal from the data.

Data Correlation



Looking at the corrgram, we see that there are many variables which are moderately to highly correlated, $|\rho| > 0.50$. Of note is that `nox` has a largest positive correlation with `target` and `dis` has the largest negative correlation with `target`.

It could be possible we are seeing a reflection of the noticeable influence that socio-economic status has on the crime rate of an area. Graif, Gladfelter and Matthews (2014) state crime is generally concentrated in disadvantaged, urban neighborhoods in the United States. An economic segregation suggests that affluent neighborhoods may be further away in terms of distance, and as a result, disadvantaged areas are more attractive to crime because the probability of success is higher even if the targets are not as profitable. We prove this data, showing areas with a higher socio-economics correlating to a lower crime rate.

Fields (2004) states industrial areas correlate with higher levels of nitrous oxide concentrations. It may be an indication of why these areas are less dense with residents of a higher status. We see the same trend in `ptratio` since a higher ratio means less funding for public institutions, which is common in areas of lower status.

Lastly, it seems that the variable `chas`, which indicate whether the suburb borders the Charles River, has statistically insignificant correlation with almost all of the other variables.

DATA PREPARATION

As there are no missing values, we have no need to impute. Further different preparation techniques are utilized for different models.

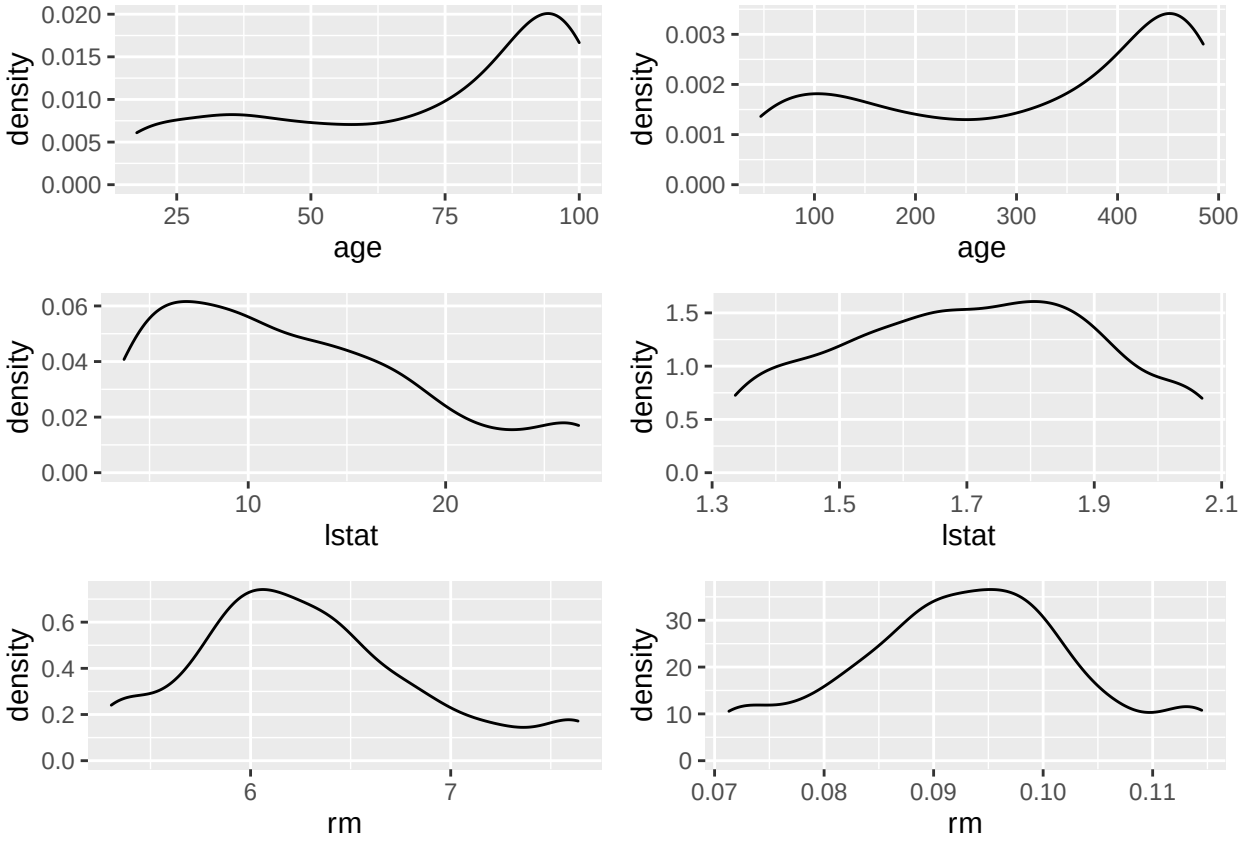
Models 1 & 2

The variable `chas` will be removed due to having statistically insignificant correlation with almost all the variables in the dataset. Next, outliers are tempered by capping below the 5th percentile and above the 95th percentile.

The Box-Cox transformation will be applied to the `age`, `lstat`, and `rm` variables.

The transform suggests a λ of 1.3427856 for `age`, 0.2212862 for `lstat`, and -1.2990027 for `rm`

Below are the densities of the variables prior (left) and post (right) the Box-Cox transform. The variables `lstat` and `rm` have benefited from the transform, but the strong bimodality of `age` has not been affected significantly.



Model 3

For model 3, we chose very little data manipulation, as there may be valuable information in the outliers. A reasonable model may be able to extract something from `chas`. Some variables, such as `chas` and `rad`, an index of accessibility, are converted to a factor. Additionally, while `rad` may have an ordering, it will be converted to a normal factor, as ordered factors are processed via orthogonal polynoial regression, which is out of the scope of this class. Given that there are now two factors in the predictors, we require creation of dummy variables. Lastly, the **labels** of `target` will be changed from integer to character for compliance with some of R's naming conventions. Based on the definitions, 0 will be mapped to *Below* and 1 to *Above*.

BUILD MODELS

First, we will split the training data into two sets. This will allow us to perform a cross validation scheme on the models to tune parameters for optimum performance.

Model 1

Base Model

We will start with a simple logistic regression model to serve as a baseline. This includes all variables in the dataset.

Table 4: Base Model Logistic Regression Output

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-50.767	10.225	-4.965	0.000
zn	-0.054	0.037	-1.473	0.141
indus	-0.008	0.057	-0.143	0.886
nox	44.401	8.259	5.376	0.000
rm	99.666	45.463	2.192	0.028
age	0.008	0.003	2.662	0.008
dis	0.695	0.254	2.739	0.006
rad	0.764	0.185	4.141	0.000
tax	-0.009	0.003	-2.757	0.006
ptratio	0.464	0.141	3.299	0.001
lstat	-1.340	2.063	-0.649	0.516
medv	0.236	0.073	3.247	0.001

We can immediately see that a few variables *exceed* the 0.05 p-value threshold for significance.

Enhanced Model

We will use **backwards stepwise regression** to remove features that are not statistically significant in predicting the target. The result is a model that includes the following features: `zn`, `nox`, `age`, `dis`, `rad`, `tax`, `ptratio`, `lstat`, and `medv`.

Coefficient Discussion

It is important to note that the coefficients from the model are predicting whether or not the target variable is a **1** (the crime rate *above* the median value). Additionally, since the numeric coefficients are relative to the range of values that the variable encompasses it is possible to have a coefficient that seems small when we look at the absolute magnitude, but that actually has a very strong effect when applied to the data.

Table 5: Backwards Model Regression Output

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-38.018	7.483	-5.080	0.000
zn	-0.064	0.035	-1.832	0.067
nox	42.812	7.261	5.896	0.000
age	0.004	0.002	1.821	0.069
dis	0.627	0.242	2.588	0.010
rad	0.713	0.169	4.209	0.000
tax	-0.008	0.003	-2.798	0.005
ptratio	0.348	0.121	2.871	0.004
lstat	0.654	1.801	0.363	0.716
medv	0.133	0.051	2.607	0.009

Looking at the *signs* of the coefficients, we can see the variables `zn` and `tax` are negative. This is indicative

of an inverse relationship; i.e., the higher these values, the less likely the crime rate is above the median. The relationship for the zone variable aligns with the findings from our initial data exploration. However, the relationship for the tax variable does not; in fact, we saw that there was a positive correlation between the two variables. As the tax rate increases so does the probability that the crime rate will be above the median.

The rest of the variables have positive coefficients and many of these are expected. For example, in our exploratory data analysis, we noticed that the `nox` variable (nitrogen oxides concentration (parts per 10 million)) has the greatest positive correlation with the target variable. This is surprising!
We had not thought that the nitrogen oxide concentration would have as large of an impact on the crime rate. According to an article in Science Daily (2019), researchers at Colorado State University found that there may actually be a strong correlation between exposure to air pollution and aggressive behavior.

Model 2

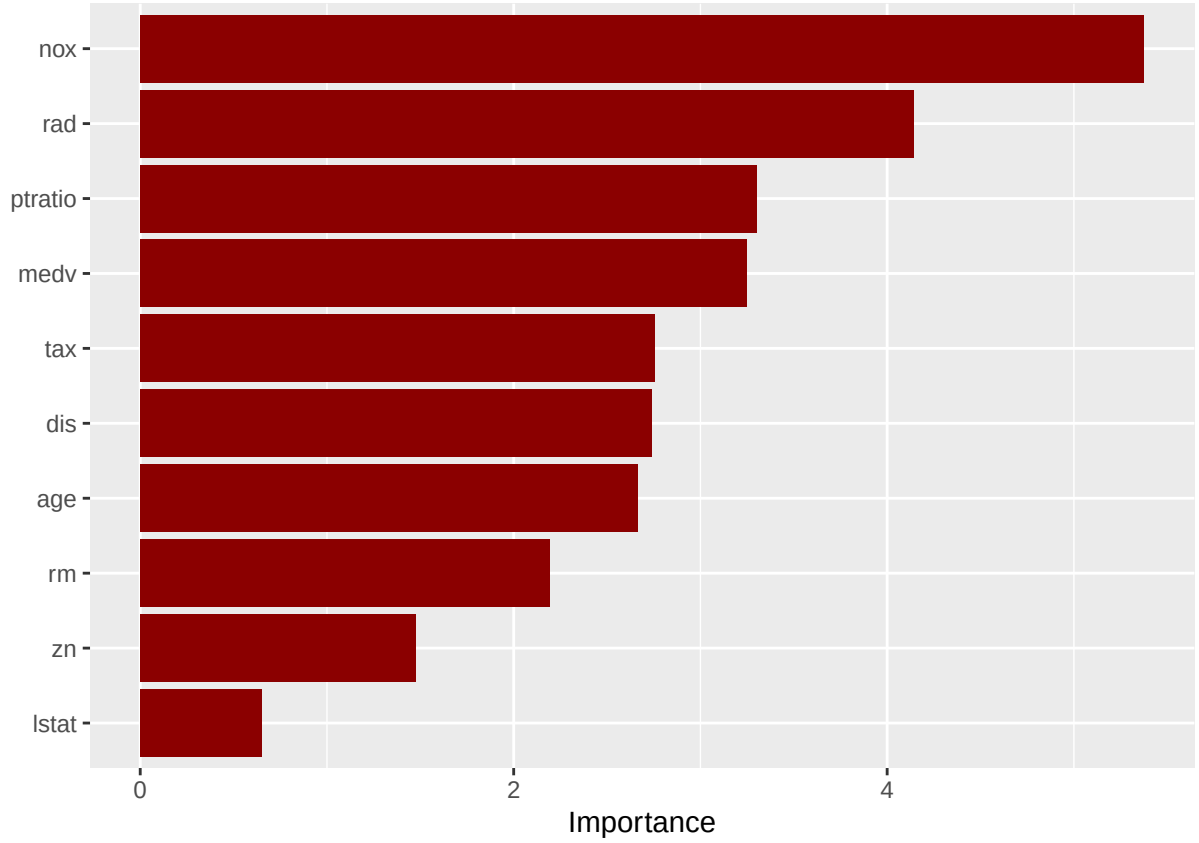
Logistic Classifier

The first step is to setup a workflow which encapsulates the logistic regression and cross-validation.

Once the workflow is set up, the model is tuned to return the best parameters.

The metric used to select the "best" model will be the area under the receiver operating curve (AUC).

Once the algorithms select a best model, the test metrics can be computed.



We again see that the primary predictor in the logistic regression is the `nox` with `rad` and `ptratio` rounding out the top three predictors staying in line with the base model. Living close to the center of town, having teachers, and not having polluted air are the largest predictors in terms of staying above median crime rates.

Model 3

Model 3 will be a binary logistic regression allowing for all variables and selected pair-wise interactions pruned using both forward and backward stepwise regression based on AICc to select the optimal parameters. The captured indices of the test set for models 1 & 2 are used to split the data for model 3 as well. With the vanilla logistic regression model in R, `glm` with a binomial family and a logit link, there are no hyper parameters available for tuning. More sophisticated logistic regression implementation have these parameters for fine-tuning. Lastly, the stepAIC method which will be used to select the best model from an information-criterion perspective, prevents the use of cross-validation.

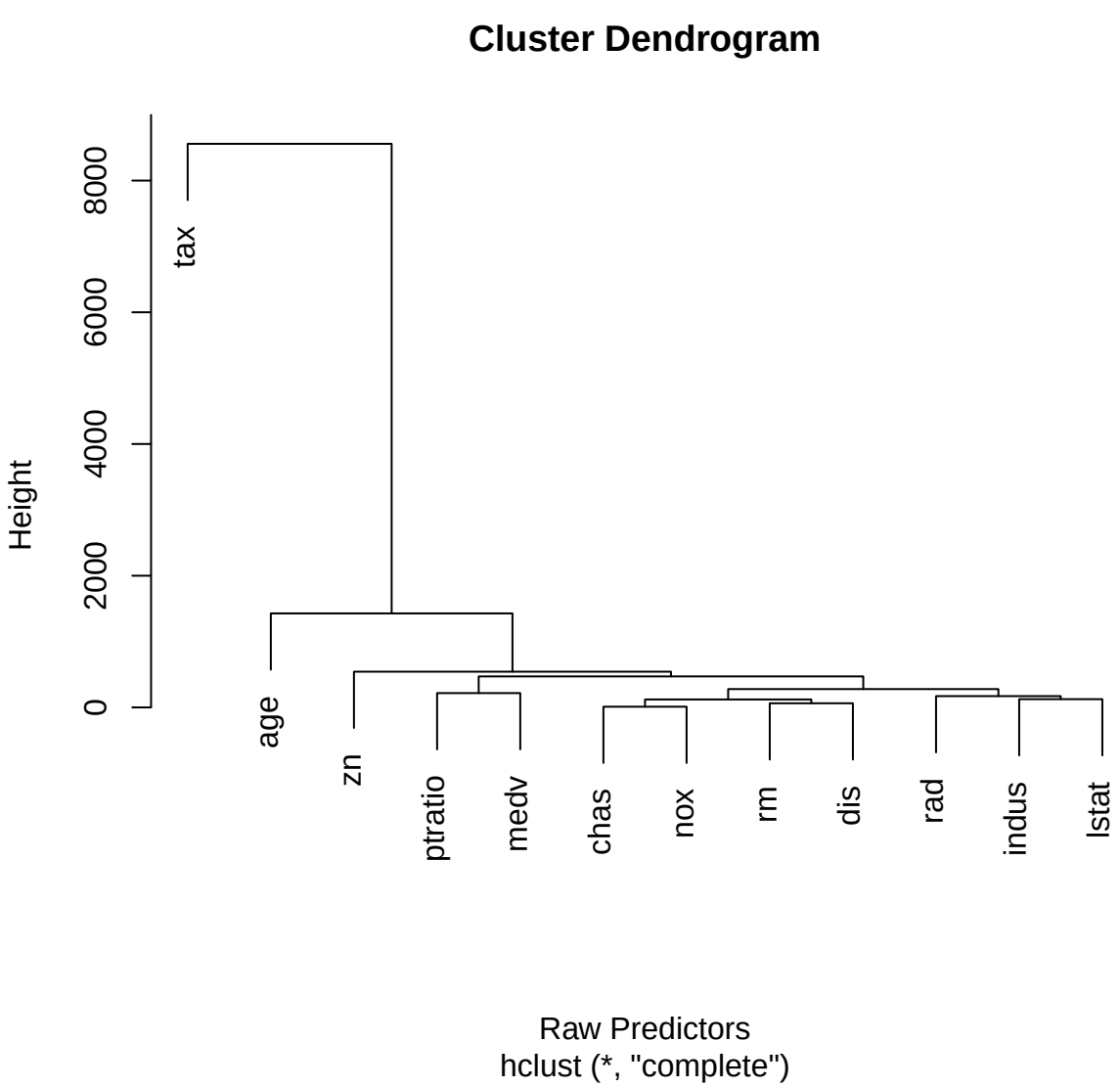
Considering *all* interactions would be foolhardy. Given the need for dummy variables, there will 19 predictors and ${}_{19}C_2 = 171$ possible pairs—too much. Therefore, intentional selection of pairs is required.

Looking at the predictors, there are some, which logically may interact with each other, such as:

- `indus`
  - The concentration of non-retail businesses in the area
- `nox` * Nitrous Oxide is a common industrial pollutant.

There is a high correlation in the training set between the two. Allowing for its interaction may temper the result of their relationship without losing more information.

What may prove interesting is to look at some unsupervised clustering techniques to provide us with insights as to which variables may be "close" enough to warrant interactions. Instead of treating the observations as samples, we will treat the predictors as samples!



It seems that there may be reason to consider an interaction between `chas` and `nox`. Can that many industrial plants be on the river? Similarly with `indus` and `lstat`. Higher-cost homes tend to be further away from the traffic and pollution of industrial areas.

Our selected starting model will include the following interactions:

- indus:nox
- indus:lstat
- chas:nox
- ptratio:medv
- rm:dis

Train Model

Coefficient Discussion

Table 6: Stepwise Interaction-Allowed Logistic Regression Output

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-94.694	24.758	-3.825	0.000
indus	3.440	1.415	2.431	0.015
nox	156.391	39.938	3.916	0.000
lstat	0.240	0.125	1.921	0.055
ptratio	0.423	0.239	1.770	0.077
medv	0.237	0.094	2.520	0.012
rm	-2.160	1.102	-1.960	0.050
rad.3	10.805	3.480	3.105	0.002
rad.4	7.977	1.589	5.020	0.000
rad.5	5.669	1.686	3.362	0.001
rad.7	15.367	3.831	4.011	0.000
rad.8	13.555	3.098	4.376	0.000
rad.24	24.576	1430.919	0.017	0.986
`indus:nox`	-5.960	2.471	-2.413	0.016
`indus:lstat`	-0.019	0.009	-2.182	0.029
`rm:dis`	0.079	0.042	1.864	0.062

The selected model has some interesting results. First, the predictor with the greatest effect is `nox`. Second, the only model variable which serves to **decrease** the crime rate from the baseline is `rm`.

The interactions between `indus` and both `nox` and `lstat` are significant. As theorized, these interactions serve to temper their individual contributions to crime. Remember, a `1` indicates crime above the median, so positive coefficients indicate contributions to *increasing* the crime rate. A similar observation holds `rm`. While `dis` on its own did not decrease the model's AICc, its interaction to temper `rm` did help the model's deviance. No other selected interactions are left in the model.

A number of variables fell out of the model, such as `chas`, `dis`, and `zn` among others. The remaining predictors are all significant, at least at the 10% level, except for `rad.24`. However, from the last trace of the stepping routine shown below, removing it would ***drastically*** reduce the models performance.

Lastly, while `rad.1` is the base case, `rad.2` and `rad.6` have disappeared as well. This indicates that both `rad = 2` and `rad = 6` should be considered indistinguishable from `rad = 1`.

```
                    Df Deviance    AIC
<none>                   90.565 122.56
+ dis                1   89.191 123.19
+ zn                 1   89.558 123.56
+ tax                1   89.700 123.70
- ptratio            1   93.763 123.76
+ age                1   90.156 124.16
+ `nox:chasOnRiver`  1   90.224 124.22
- `rm:dis`           1   94.254 124.25
+ chas.OnRiver       1   90.273 124.27
+ rad.6              1   90.315 124.31
- lstat              1   94.364 124.36
+ `ptratio:medv`     1   90.494 124.49
+ rad.2              1   90.554 124.55
- rm                 1   95.001 125.00
- `indus:lstat`      1   95.633 125.63
- `indus:nox`        1   97.351 127.35
- indus              1   97.597 127.60
- medv               1   98.568 128.57
```

```
- rad.3              1  101.594 131.59
- rad.5              1  105.594 135.59
- rad.7              1  118.832 148.83
- nox                1  134.786 164.79
- rad.8              1  149.967 179.97
- rad.4              1  152.757 182.76
- rad.24             1  152.797 182.80
```

SELECT MODELS

Criteria

In binary classification, often, the measures of focus are precision and recall, as opposed to accuracy. A measure combining this is the F1 score, defined as twice the sum of precision and recall divided by their product. Another oft-used metric is the area under the receiver operating curve, AUC. We will look at the three metrics of accuracy, F1, and AUC, and select the model that performs best in at least two of the three metrics.

Performance

Below we show the confusion matrices for the three models, followed by a table with the selected performance.

Model 1

```
##            Reference
## Prediction  0  1
##          0 44  4
##          1  3 41
```

Model 2

```
##            Truth
## Prediction  0  1
##          0 45  4
##          1  2 41
```

Model 3

```
##            Reference
## Prediction Below Above
##      Below    46     3
##      Above     1    42
```

Selected Performance Statistics

Table 7: Model Test Results

Models	ACC	F1	AUC
Model 1	0.924	0.926	0.983
Model 2	0.935	0.938	0.983
Model 3	0.957	0.958	0.992

All the models performed very well and classified the majority of the test set properly. The difference between the three models was very slight. Only 7, 6, and 4 observations out of 92 were misclassified by the three

models respectively. Given the above statistics and confusion matrices, **Model 3** barely edged out the other two and will be used to predict on the `eval` set.

Predictions

Table 8: Predicted Classes and Probabilities

TestID	PredictedClass	Prob_Is_0	Prob_Is_1
1	0	1.000	0.000
2	1	0.084	0.916
3	1	0.126	0.874
4	1	0.058	0.942
5	0	0.992	0.008
6	0	0.725	0.275
7	0	0.878	0.122
8	0	0.999	0.001
9	0	0.999	0.001
10	0	1.000	0.000
11	0	1.000	0.000
12	0	1.000	0.000
13	1	0.062	0.938
14	1	0.237	0.763
15	0	0.505	0.495
16	0	0.880	0.120
17	1	0.037	0.963
18	1	0.010	0.990
19	0	0.803	0.197
20	0	1.000	0.000
21	0	1.000	0.000
22	0	0.595	0.405
23	1	0.153	0.847
24	0	0.992	0.008
25	0	0.992	0.008
26	0	0.904	0.096
27	0	1.000	0.000
28	1	0.000	1.000
29	1	0.000	1.000
30	1	0.000	1.000
31	1	0.000	1.000
32	1	0.000	1.000
33	1	0.000	1.000
34	1	0.000	1.000
35	1	0.000	1.000
36	1	0.000	1.000
37	1	0.000	1.000
38	1	0.000	1.000
39	0	0.827	0.173
40	0	0.982	0.018

REFERENCES

- Fields S. (2004). Global nitrogen: cycling out of control. *Environmental health perspectives*, *112*(10), A556–A563. https://doi.org/10.1289/ehp.112-a556
- Graif, C., Gladfelter, A. S., & Matthews, S. A. (2014). Urban Poverty and Neighborhood Effects on Crime: Incorporating Spatial and Network Perspectives. *Sociology compass*, *8*(9), 1140–1155. https://doi.org/10.1111/soc4.12199
- Colorado State University. (2019, October 3). Exposure to air pollution increases violent crime rates. *ScienceDaily*. Retrieved October 26, 2020 from www.sciencedaily.com/releases/2019/10/191003114007.htm

CODE APPENDIX

The code chunks below represent the R code called in order during the analysis. They are reproduced in the appendix for review and comment.

```
library(tidyverse)
library(tidymodels)
library(data.table)
library(vip)
library(summarytools)
library(corrplot)
library(knitr)
library(rsample) # model 1 libraries
library(caret)
library(geoR)

set.seed(9450)

urlRemote  = "https://raw.githubusercontent.com/"
pathGithub = "aadler/DT621_Fall2020_Group2/master/HW3/data/"
fileTrain = "crime-training-data_modified.csv"
fileTest = "crime-evaluation-data_modified.csv"

df <- read.csv(paste0(urlRemote, pathGithub, fileTrain))
nobs <- dim(df)[[1]]
DT <- as.data.table(df)
DT[, target := factor(target)]
eval <- read.csv(paste0(urlRemote, pathGithub, fileTest))
```

```
# Summary Statistics
summarystat = stby(data = df, INDICES = df$target, FUN = psych::describe)
kable(summarystat[[1]][-13,-c(1,7)],
      caption = "Descriptive Statistics: Crime Rate > Median",
      digit = 2L)
kable(summarystat[[2]][-13,-c(1,7)],
      caption = "Descriptive Statistics: Crime Rate < Median",
      digit = 2L)
```

```
# Density plots
DT[, IDX := .I]
DTM <- melt(DT, id.vars = c('IDX', 'target'), measure.vars = 1:12,
            variable.name = 'metric', value.name = 'value')
ggplot(DTM, aes(x = value, fill = target, color = target)) +
  geom_density(alpha = 0.4, show.legend = FALSE) +
  facet_wrap(~ metric, scales = 'free')
```

```r
# Corrgram
corrplot(cor(df), method = 'ellipse', type = 'lower', order = 'hclust')
```

```r
# Model 1&2 Prep: Remove chas
new_df <- subset(df, select = -chas)

# Model 1&2 Prep:  Cap all observations at their 5th and 95th percentiles
low5 <- apply(new_df, 2, quantile, prob = 0.05)
up95 <- apply(new_df, 2, quantile, prob = 0.95)
for (i in seq_along(new_df)) {
  new_df[, i] <- pmin(new_df[, i], up95[i])
  new_df[, i] <- pmax(new_df[, i], low5[i])
}

# Model 1&2 Prep: Store density plots prior to Box-Cox transform
dens_age0 <- ggplot(new_df, aes(age)) + geom_density()
dens_lstat0 <- ggplot(new_df, aes(lstat)) + geom_density()
dens_rm0 <- ggplot(new_df, aes(rm)) + geom_density()
```

```r
# Model 1&2 Prep:  Calculate Box-Cox lambdas
ageBC <- boxcoxfit(new_df$age)
lstatBC <- boxcoxfit(new_df$lstat)
rmBC <- boxcoxfit(new_df$rm)
```

```r
# Model 1&2 Prep: Apply the Box-Cox transform
new_df$age <- new_df$age ^ ageBC$lambda
new_df$lstat <- new_df$lstat ^ lstatBC$lambda
new_df$rm <- new_df$rm ^ rmBC$lambda
```

```r
# Model 1&2 Prep: Graphically display the effect of the Box-Cox transform
dens_age <- ggplot(new_df, aes(age)) + geom_density()
dens_lstat <- ggplot(new_df, aes(lstat)) + geom_density()
dens_rm <- ggplot(new_df, aes(rm)) + geom_density()
grid.arrange(dens_age0, dens_lstat0, dens_rm0, dens_age, dens_lstat, dens_rm,
             layout_matrix = cbind(c(0, 1, 2), c(3, 4, 5)))
```

```r
# Model 1: 10-fold cross validation
train_control <- trainControl(method = "cv", number = 10)

# Model 1: Train the model on training set
model1 <- train(target ~ zn + nox + age + dis + rad + tax + ptratio + lstat +
                  medv,
                data = train_df,
                trControl = train_control,
                method = "glm",
                family=binomial())
kable(summary(model1)$coefficients, digits = 3L,
      caption = 'Backwards Model Regression Output')
```

```r
# Model 2: Setup
cv_folds <- vfold_cv(train_df, v = 10, repeats = 1)

crime_recipe <- recipe(target ~., data=train_df)
```

```r
crime_wf <- workflow() %>%
  add_recipe(crime_recipe)
```

```r
# Model 2: Call logistic regression
logit_specs <-
  logistic_reg(
    penalty = tune(),
    mixture = tune()
    ) %>%
  set_engine("glm") %>%
  set_mode("classification")
```

```r
# Model 2: Setup tuning grid
logit_wf <-
  crime_wf %>%
  add_model(logit_specs)

crtl_grid <-control_grid(verbose = FALSE)

logit_results <-
  tune_grid(
    logit_wf,
    resamples = cv_folds,
    control = crtl_grid,
    grid = 10,
    save_pred = TRUE,
    save_workflow = FALSE
)
```

```r
# Model 2: Select AUC as optimizing metric
best_model_auc <- select_best(logit_results, "roc_auc")
final_auc <-
  finalize_workflow(
    logit_wf,
    best_model_auc
)

final_model_pred <-
  final_auc %>%
  last_fit(data_split) %>%
  collect_predictions()
```

```r
# Model 2: Compute metrics on test set (done here instead of below)
m2_roc<- yardstick::roc_auc(
  final_model_pred,
  truth = target,
  contains(".pred_1")
  )

m2_acc<- yardstick::accuracy(
  final_model_pred,
  truth = target,
  estimate = .pred_class
)
```

```r
m2_recall<- yardstick::recall(
  final_model_pred,
  truth = target,
  estimate = .pred_class
)

m2_precise<- yardstick::precision(
  final_model_pred,
  truth = target,
  estimate = .pred_class
)

m2_metrics_df<-
  bind_rows(m2_roc, m2_acc, m2_recall, m2_precise)
```

```r
# Model 2: Plot variable importance
logit_wf %>%
  fit(data = train_df) %>%
  pull_workflow_fit() %>%
  vip(geom = "col", aesthetics = list(fill='red4'))
```

```r
# Model 3: Create cluster plot of features
DT_train <- DT[!(IDX %in% tstIDX)]
predClust <- hclust(dist(t(DT_train[, -c('target', 'IDX')]), diag = TRUE))
plot(predClust, xlab = "Raw Predictors")
```

```r
# Model 3: Data Preparation
DT[, `:=`(chas = factor(chas, labels = c('OffRiver', 'OnRiver')),
          rad = factor(rad),
          target = factor(target, labels = c('Below', 'Above')))]
DT_test <- DT[IDX %in% tstIDX]
DT_train <- DT[!(IDX %in% tstIDX)]

# No CV, use precision/recall
trC <- trainControl(method = 'none', classProbs = TRUE,
                    summaryFunction = twoClassSummary)

# Model 3: Create dummy variables
dVars <- dummyVars(target ~ indus * nox +
                     indus * lstat +
                     chas * nox +
                     ptratio * medv +
                     rm * dis + . - IDX, data = DT_train, fullRank = TRUE)
DTtrnx <- predict(dVars, newdata = DT_train)
DTtrny <- DT_train$target

# Model 3: Train Model
set.seed(1)
m3Fit <- train(x = DTtrnx, y = DTtrny, method = 'glmStepAIC', trControl = trC,
               family = 'binomial', trace = 0, direction = 'both')
```

```r
# Model 3: Model summary
kable(summary(m3Fit$finalModel)$coefficients, digits = 3L,
             caption = 'Stepwise Interaction-Allowed Logistic Regression Output')
```

```r
# Create table to hold model comparison statistics
compTable <- data.frame(Models = c('Model 1', 'Model 2', 'Model 3'),
                        ACC = double(3),
                        F1 = double(3),
                        AUC = double(3))
```

```r
# Model 1 Performance
model1_preds <- predict(model1, test_df, type = "raw")
model1_probs <- predict(model1, test_df, type = "prob")
colnames(model1_probs) <- c('pred0', 'pred1')

model1_results <- test_df %>%
  bind_cols(pred = model1_preds, model1_probs)

# Model 1 Metrics
m1_roc <- yardstick::roc_auc(
  model1_results,
  truth = target,
  pred0 # select the prob class that corresponds to first level of target
)

m1_acc <- yardstick::accuracy(
  model1_results,
  truth = target,
  estimate = pred
)

m1_recall <- yardstick::recall(
  model1_results,
  truth = target,
  estimate = pred
)

m1_precise <- yardstick::precision(
  model1_results,
  truth = target,
  estimate = pred
)

m1_metrics_df<-
  bind_rows(m1_roc, m1_acc, m1_recall, m1_precise)

# Model 1 Confusion Matrix
m1CM <- confusionMatrix(model1_preds, model1_results$target,
                        mode = 'prec_recall')
m1CM$table

# Model 1 Populate comparison table
compTable[1, 2] <- m1_acc$.estimate
compTable[1, 3] <- 2 * m1_precise$.estimate * m1_recall$.estimate /
  (m1_precise$.estimate + m1_recall$.estimate)
compTable[1, 4] <- m1_roc$.estimate
```

```r
# Model 2 calculate metrics and print confusion matrix
final_model_pred %>%
  conf_mat(truth = target, estimate = .pred_class)

m2.metrics <- logit_wf %>%
  last_fit(data_split) %>%
  collect_metrics()

# Model 2 Populate comparison table
compTable[2, 2] <- m2.metrics[1, 3]
compTable[2, 3] <- 2 * m2_precise$.estimate * m2_recall$.estimate /
  (m2_precise$.estimate + m2_recall$.estimate)
compTable[2, 4] <- m2.metrics[2, 3]
```

```r
# Model 3: Process test set
tstdVars <- dummyVars(target ~ indus * nox +
                        indus * lstat +
                        chas * nox +
                        ptratio * medv +
                        rm * dis + . - IDX, data = DT_test, fullRank = TRUE)
DTtstx <- predict(tstdVars, newdata = DT_test)
DTtsty <- DT_test$target

# Model 3: Predict on test set
m3Pred <- predict(m3Fit, newdata = DTtstx)
m3PredP <- predict(m3Fit, newdata = DTtstx, type = 'prob')

# Model 3: Confusion Matrix
m3CM <- confusionMatrix(m3Pred, DTtsty, mode = 'prec_recall')
m3CM$table

# Model 3: Populate comparison table
compTable[3, 2] <- sum(diag(m3CM$table)) / sum(m3CM$table)
compTable[3, 3] <- m3CM$byClass[7]
compTable[3, 4] <- yardstick::roc_auc_vec(DTtsty, m3PredP[, 1])
```

```r
# Display Model Comparison
kable(compTable, digits = 3L, caption = 'Model Test Results')
```

```r
# Selected Model: Process eval file data as per train and test sets
setDT(eval)
neval <- dim(eval)[[1]]
eval[, `:=`(chas = factor(chas, labels = c('OffRiver', 'OnRiver')),
          rad = factor(rad))]
evalVars <- dummyVars( ~ indus * nox + indus * lstat + chas * nox +
                         ptratio * medv + rm * dis + .,
                       data = eval, fullRank = TRUE)
evaltstx <- predict(evalVars, newdata = eval)

# Selected Model: Predict classes and probabilities
evalPred <- predict(m3Fit, newdata = evaltstx)
evalPredP <- predict(m3Fit, newdata = evaltstx, type = 'prob')

# Selected Model: Display results
```

```r
predTable <- data.frame(TestID = seq_len(neval),
                        PredictedClass = ifelse(evalPred == "Below", 0, 1),
                        Prob_Is_0 = evalPredP$Below,
                        Prob_Is_1 = evalPredP$Above)
kable(predTable, digits = 3L, caption = "Predicted Classes and Probabilities")
```