DATA_621_HW3

Chi Pong, Euclid Zhang, Jie Zou, Joseph Connolly, LeTicia Cancel

3/8/2022

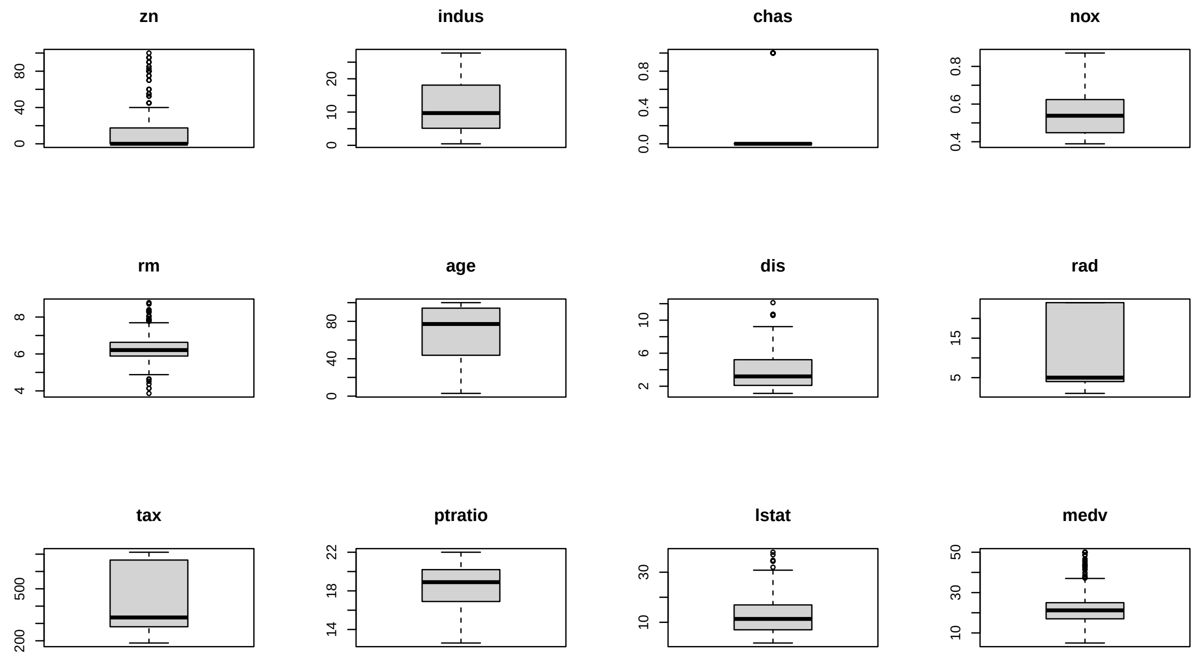
```
train_df <- read.csv("https://raw.githubusercontent.com/ezaccountz/DATA_621/main/HW3/crime-training-data
test_df <- read.csv("https://raw.githubusercontent.com/ezaccountz/DATA_621/main/HW3/crime-evaluation-da
```

DATA EXPLORATION

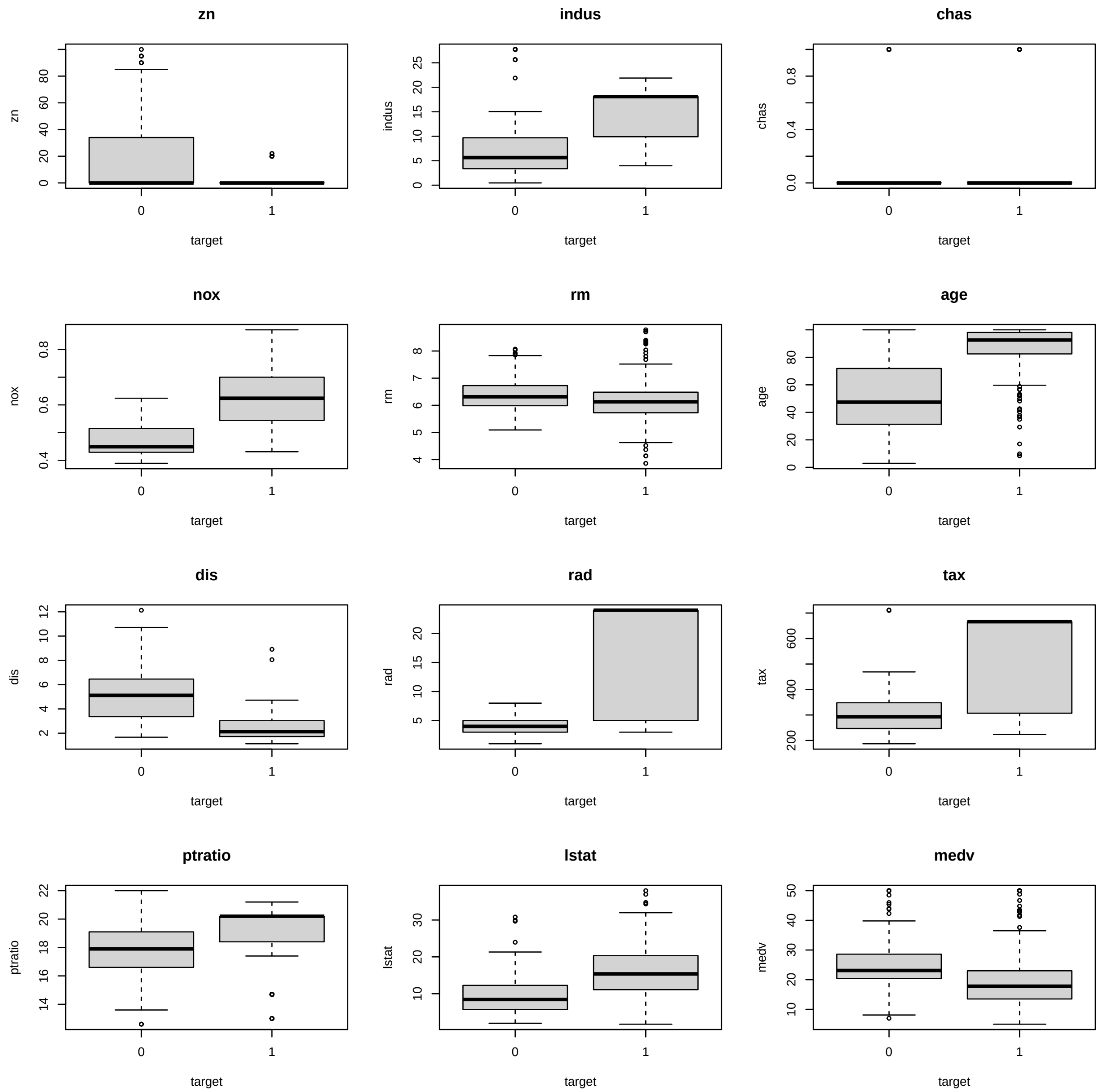
```
summary(train_df)
```

```
##        zn             indus             chas              nox
##  Min.   :  0.00   Min.   : 0.460   Min.   :0.00000   Min.   :0.3890
##  1st Qu.:  0.00   1st Qu.: 5.145   1st Qu.:0.00000   1st Qu.:0.4480
##  Median :  0.00   Median : 9.690   Median :0.00000   Median :0.5380
##  Mean   : 11.58   Mean   :11.105   Mean   :0.07082   Mean   :0.5543
##  3rd Qu.: 16.25   3rd Qu.:18.100   3rd Qu.:0.00000   3rd Qu.:0.6240
##  Max.   :100.00   Max.   :27.740   Max.   :1.00000   Max.   :0.8710
##        rm             age              dis              rad
##  Min.   :3.863   Min.   :  2.90   Min.   : 1.130   Min.   : 1.00
##  1st Qu.:5.887   1st Qu.: 43.88   1st Qu.: 2.101   1st Qu.: 4.00
##  Median :6.210   Median : 77.15   Median : 3.191   Median : 5.00
##  Mean   :6.291   Mean   : 68.37   Mean   : 3.796   Mean   : 9.53
##  3rd Qu.:6.630   3rd Qu.: 94.10   3rd Qu.: 5.215   3rd Qu.:24.00
##  Max.   :8.780   Max.   :100.00   Max.   :12.127   Max.   :24.00
##       tax           ptratio         lstat             medv
##  Min.   :187.0   Min.   :12.6   Min.   : 1.730   Min.   : 5.00
##  1st Qu.:281.0   1st Qu.:16.9   1st Qu.: 7.043   1st Qu.:17.02
##  Median :334.5   Median :18.9   Median :11.350   Median :21.20
##  Mean   :409.5   Mean   :18.4   Mean   :12.631   Mean   :22.59
##  3rd Qu.:666.0   3rd Qu.:20.2   3rd Qu.:16.930   3rd Qu.:25.00
##  Max.   :711.0   Max.   :22.0   Max.   :37.970   Max.   :50.00
##      target
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.4914
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

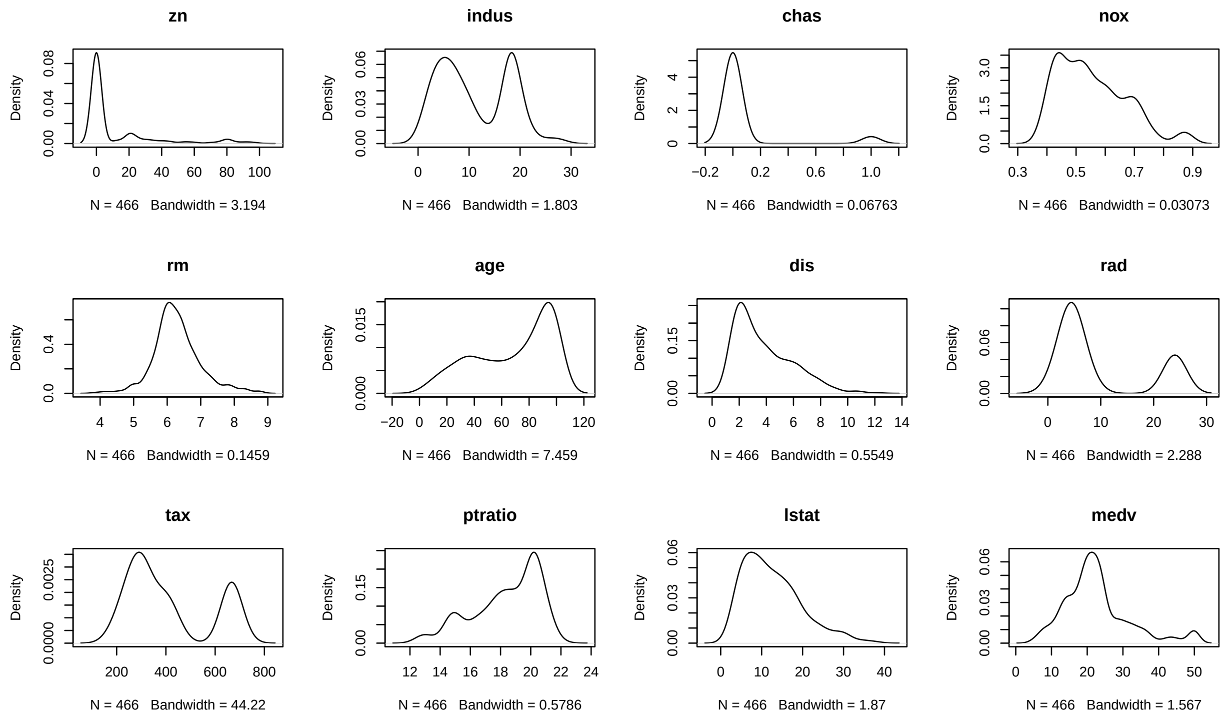
```r
par(mfrow=c(3,4))
predictors <- colnames(train_df)
predictors <- predictors[!predictors %in% c("target")]

for(preditor in predictors) {
  boxplot(train_df[,preditor],main=preditor)
}
```




```r
par(mfrow=c(4,3))
predictors <- colnames(train_df)
predictors <- predictors[!predictors %in% c("target")]

for(preditor in predictors) {
  boxplot(train_df[,preditor]~train_df$target,main=preditor,
          xlab = "target", ylab = preditor)
}
```



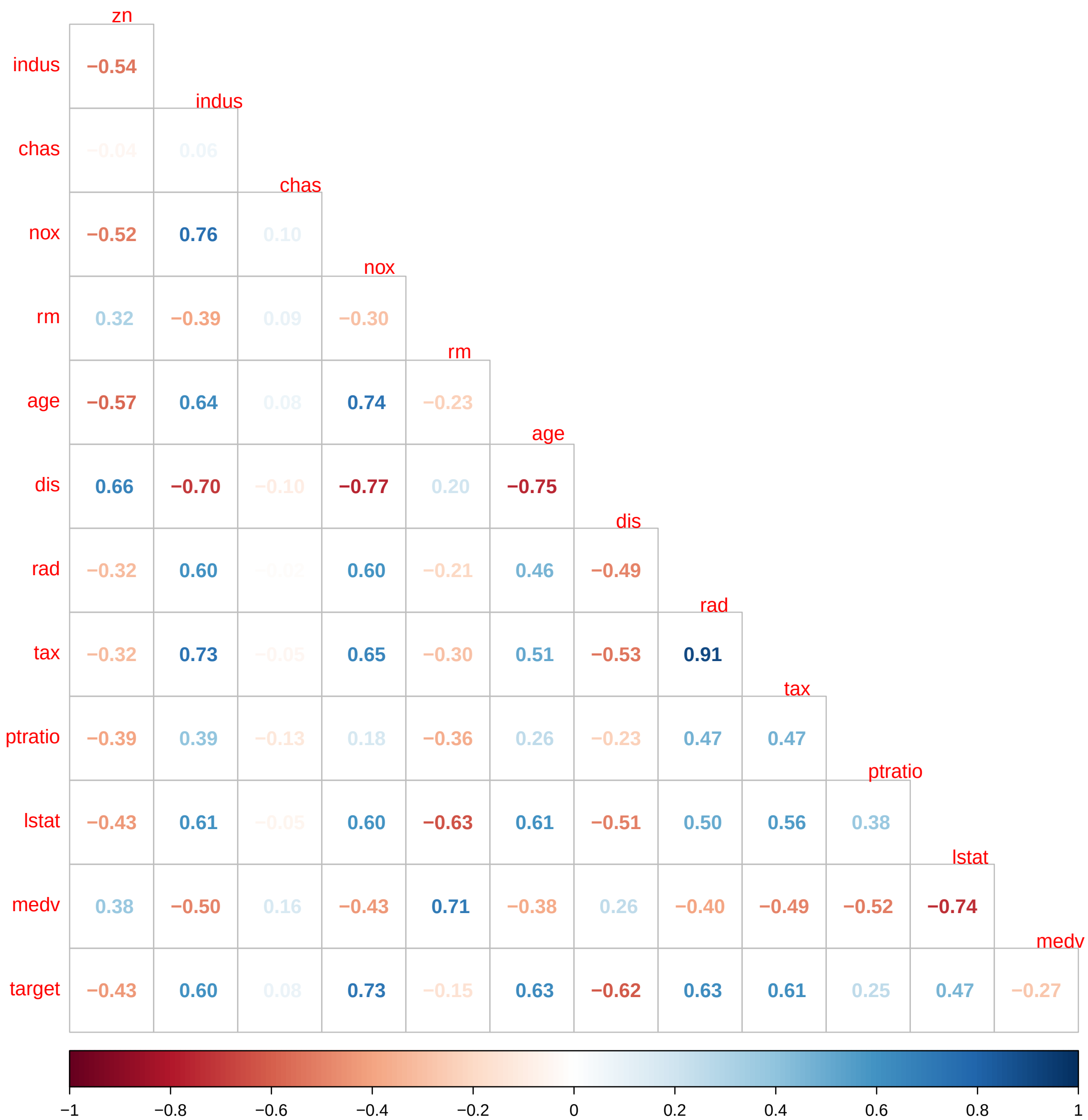

```
par(mfrow=c(3,4))

predictors <- colnames(train_df)
predictors <- predictors[!predictors %in% c("target")]

for(preditor in predictors) {
  plot(density(train_df[,preditor],na.rm=TRUE),main=preditor)
}
```



* Correlations

Now let's look at the correlations between the variables

```
corrplot(cor(train_df, use = "na.or.complete"), method = 'number', type = 'lower', diag = FALSE, tl.srt
```



DATA PREPARATION

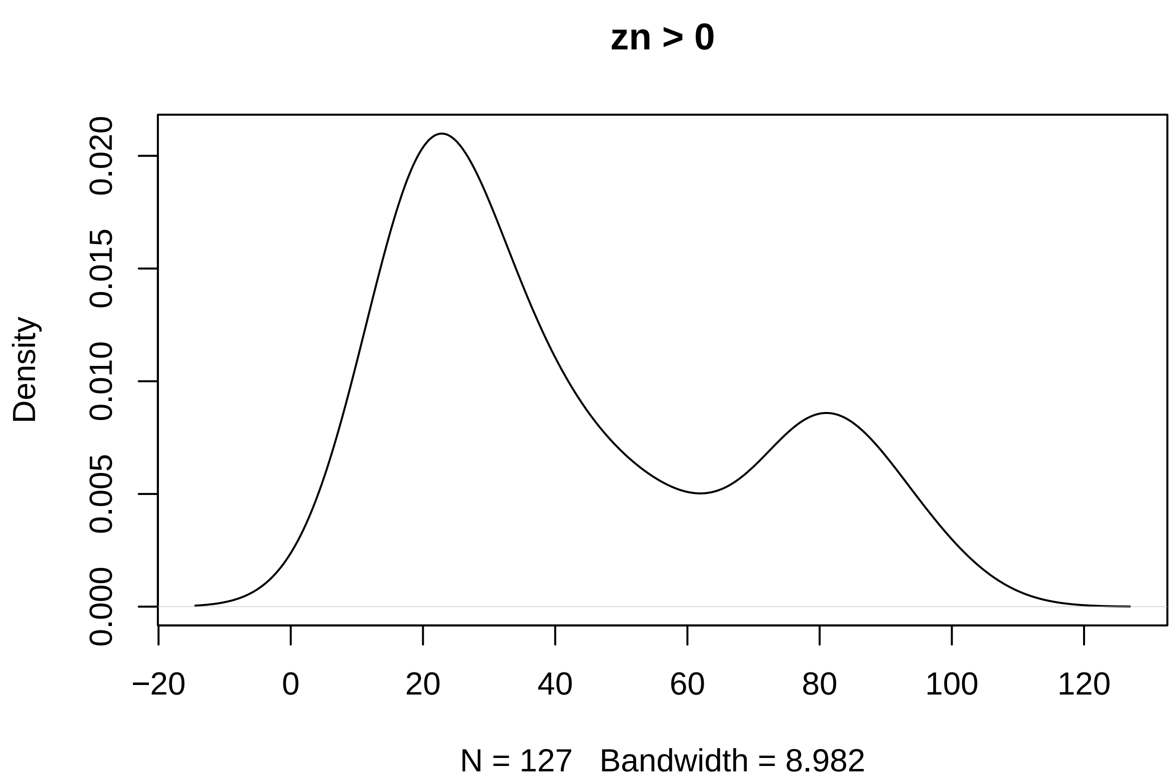
From the density plot of **zn**, we know that the variable is zero-inflated. The percentage of 0 values is

```
nrow(train_df[train_df$zn==0,])/nrow(train_df)
```

```
## [1] 0.7274678
```

Let's check the distribution of the **zn** without the 0 values

```
plot(density(train_df[train_df$zn>0,]$zn,na.rm=TRUE), main = "zn > 0")
```



The distribution looks a lot better.
We will add a new dummy variable zn_y indicating if **zn** is >0. The interaction **zn x zn_y = zn** so we don't need to do anything to it. If **zn_y** is deemed to be insignificant by our models, then we can simply drop it.

```
train_df$zn_y <- 0
train_df$zn_y[train_df$zn>0] <- 1
```

According to the text book *A Modern Approach To Regression With R*, "when the predictor variable X has a Poisson distribution, the log odds are a linear function of x". Let's check if any of the predictors follows a Poisson distribution

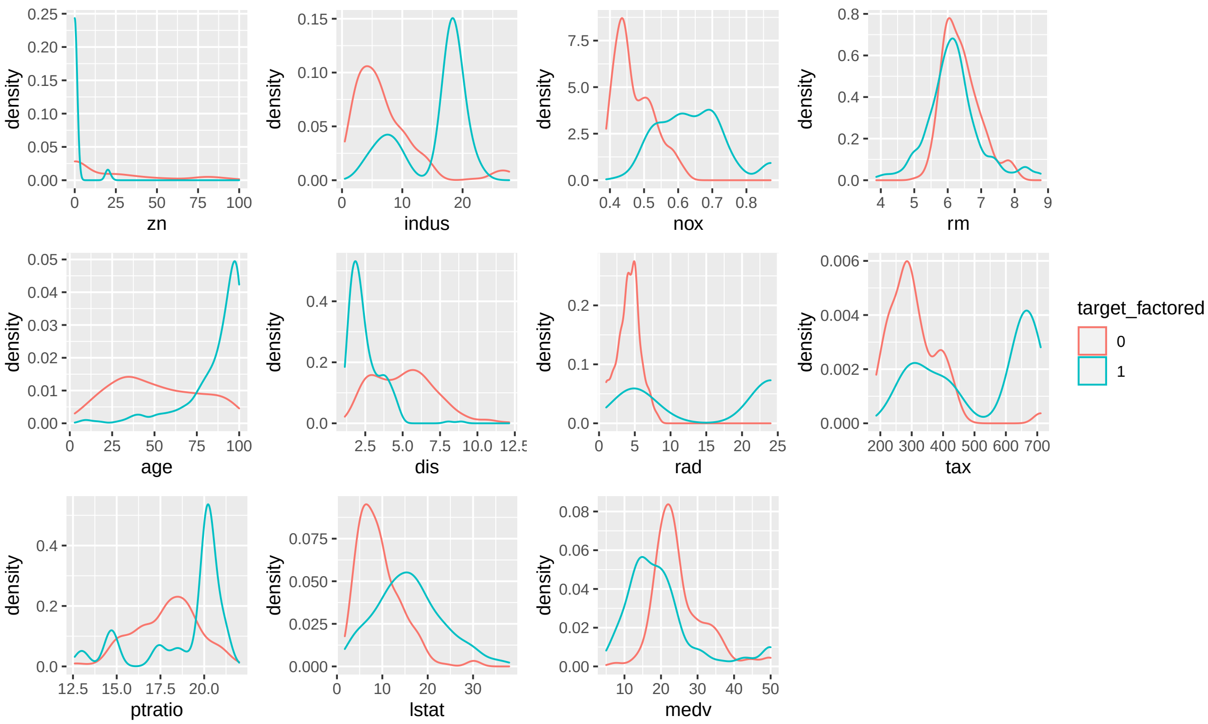
```
#Method of possion distribution test is from https://stackoverflow.com/questions/59809960/how-do-i-know

#two tail test
p_poisson <- function(x) {
  return (1-2 * abs((1 - pchisq((sum((x - mean(x))^2)/mean(x)), length(x) - 1))-0.5))
}

predictors <- colnames(train_df)
predictors <- predictors[!predictors %in% c("target","chas","zn_y")]

data.frame(mean = round(apply(train_df[,predictors],2,mean),2),
```

```r
           variance = round(apply(train_df[,predictors],2,var),2),
           probability_of_poisson = round(apply(train_df[,predictors],2,p_poisson),2))
```

```
##            mean variance probability_of_poisson
## zn        11.58   545.91                   0.00
## indus     11.11    46.87                   0.00
## nox        0.55     0.01                   0.00
## rm         6.29     0.50                   0.00
## age       68.37   802.10                   0.00
## dis        3.80     4.44                   0.01
## rad        9.53    75.45                   0.00
## tax      409.50 28190.44                   0.00
## ptratio   18.40     4.83                   0.00
## lstat     12.63    50.44                   0.00
## medv      22.59    85.37                   0.00
```

None of the predictors follows a poisson distribution

```r
target_factored <- as.factor(train_df$target)

plot_zn <- ggplot(train_df, aes(x=zn, color=target_factored)) + geom_density()
plot_indus <- ggplot(train_df, aes(x=indus, color=target_factored)) + geom_density()
plot_nox <- ggplot(train_df, aes(x=nox, color=target_factored)) + geom_density()
plot_rm <- ggplot(train_df, aes(x=rm, color=target_factored)) + geom_density()
plot_age <- ggplot(train_df, aes(x=age, color=target_factored)) + geom_density()
plot_dis <- ggplot(train_df, aes(x=dis, color=target_factored)) + geom_density()
plot_rad <- ggplot(train_df, aes(x=rad, color=target_factored)) + geom_density()
plot_tax <- ggplot(train_df, aes(x=tax, color=target_factored)) + geom_density()
plot_prtatio <- ggplot(train_df, aes(x=ptratio, color=target_factored)) + geom_density()
plot_lstat <- ggplot(train_df, aes(x=lstat, color=target_factored)) + geom_density()
plots_medv <- ggplot(train_df, aes(x=medv, color=target_factored)) + geom_density()

plot_zn+plot_indus+plot_nox+plot_rm+plot_age+plot_dis+plot_rad+plot_tax+
  plot_prtatio+plot_lstat+plots_medv+plot_layout(ncol = 4, guides = "collect")
```



The distributions for rm with target = 0 and target = 1 are approximately normal with the same variance. Hence we don't need to transform the variable The distributions for lstat and medv are skewed, we will add a log-transformed variable for each of them The distributions for indus, nox, age, dis, tax, ptratio look significantly different for the target values. We will not perform transformation for them unless transformations have to be done to meet the model assumptions

```
train_df$log_lstat <- log(train_df$lstat)
train_df$log_medv <- log(train_df$medv)
```

```
predictors <- colnames(train_df)
predictors <- predictors[!predictors %in% c("target","chas","zn_y","log_lstat","log_medv")]

interaction_test <- data.frame(matrix(ncol = 3, nrow = 0))
colnames(interaction_test) <- c("Preditor","Interaction","p-value")
class(interaction_test$`p-value`) = "Numeric"

for (predictor in predictors) {
  interaction_test[nrow(interaction_test) + 1,] <-
    c(predictor, paste0(predictor, ":chas"),
      round(anova(glm(target ~ train_df[,predictor]*chas,data = train_df, family = "binomial"),test="Chi
}
```

```
interaction_test
```

```
##    Preditor  Interaction p-value
## 1        zn      zn:chas  0.0645
## 2     indus   indus:chas   0.954
## 3       nox     nox:chas  0.6638
## 4        rm      rm:chas  0.6647
```

```
## 5       age     age:chas  0.0719
## 6       dis     dis:chas  0.0681
## 7       rad     rad:chas  0.0191
## 8       tax     tax:chas       0
## 9   ptratio ptratio:chas  0.3917
## 10    lstat   lstat:chas  0.1006
## 11     medv    medv:chas  0.1555
```

We will add an interaction between **tax** and **chas** and an interaction between **rad** and **chas** to our preditor candidates.

```r
train_df$tax_chas <- train_df$tax * train_df$chas
train_df$rad_chas <- train_df$rad * train_df$chas
```

```r
full_model <- glm(target~.,family = binomial, train_df)
```

```r
summary(full_model)
```

```
##
## Call:
## glm(formula = target ~ ., family = binomial, data = train_df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9768  -0.1345   0.0000   0.0014   3.6544
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.078e+01  1.080e+01  -2.849  0.00439 **
## zn          -1.837e-02  3.967e-02  -0.463  0.64331
## indus       -6.174e-02  4.923e-02  -1.254  0.20979
## chas        -6.492e+02  3.308e+04  -0.020  0.98434
## nox          5.241e+01  8.421e+00   6.224 4.86e-10 ***
## rm          -7.434e-01  8.094e-01  -0.918  0.35838
## age          3.481e-02  1.439e-02   2.420  0.01554 *
## dis          7.740e-01  2.580e-01   3.000  0.00270 **
## rad          6.990e-01  1.789e-01   3.907 9.34e-05 ***
## tax         -7.913e-03  3.205e-03  -2.469  0.01354 *
## ptratio      3.199e-01  1.369e-01   2.336  0.01947 *
## lstat        1.712e-01  1.468e-01   1.166  0.24350
## medv         2.292e-01  1.451e-01   1.580  0.11408
## zn_y        -1.347e+00  1.242e+00  -1.084  0.27826
## log_lstat   -2.039e+00  1.767e+00  -1.154  0.24848
## log_medv    -2.206e+00  3.210e+00  -0.687  0.49208
## tax_chas     2.578e+00  1.312e+02   0.020  0.98432
## rad_chas    -1.568e+01  9.121e+02  -0.017  0.98629
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 177.00  on 448  degrees of freedom
```

```
## AIC: 213
##
## Number of Fisher Scoring iterations: 21
```

```
model_AIC <- step(full_model, trace=0)
```

```
summary(model_AIC)
```

```
##
## Call:
## glm(formula = target ~ chas + nox + age + dis + rad + tax + ptratio +
##     medv + zn_y + tax_chas, family = binomial, data = train_df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8710  -0.1667   0.0000   0.0025   3.5538
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.939e+01  6.410e+00  -6.146 7.96e-10 ***
## chas        -5.251e+03  3.082e+05  -0.017 0.986407
## nox          4.750e+01  7.247e+00   6.555 5.55e-11 ***
## age          3.292e-02  1.143e-02   2.879 0.003984 **
## dis          7.585e-01  2.260e-01   3.357 0.000789 ***
## rad          6.870e-01  1.592e-01   4.316 1.59e-05 ***
## tax         -7.915e-03  2.696e-03  -2.935 0.003331 **
## ptratio      2.859e-01  1.245e-01   2.297 0.021594 *
## medv         1.099e-01  3.697e-02   2.973 0.002953 **
## zn_y        -1.639e+00  7.866e-01  -2.084 0.037150 *
## tax_chas     1.897e+01  1.113e+03   0.017 0.986401
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 183.72  on 455  degrees of freedom
## AIC: 205.72
##
## Number of Fisher Scoring iterations: 24
```

```
drop1(glm(target ~ .-rad_chas-zn-log_medv-rm-log_lstat-lstat-indus, family=binomial, train_df), test="Cl
```

```
## Single term deletions
##
## Model:
## target ~ (zn + indus + chas + nox + rm + age + dis + rad + tax +
##     ptratio + lstat + medv + zn_y + log_lstat + log_medv + tax_chas +
##     rad_chas) - rad_chas - zn - log_medv - rm - log_lstat - lstat -
##     indus
##          Df Deviance    AIC    LRT  Pr(>Chi)
## <none>        183.72 205.72
```

```
## chas      1   196.02 216.02 12.299 0.0004532 ***
## nox       1   263.30 283.30 79.581 < 2.2e-16 ***
## age       1   192.73 212.73  9.017 0.0026746 **
## dis       1   195.78 215.78 12.067 0.0005133 ***
## rad       1   231.47 251.47 47.752 4.838e-12 ***
## tax       1   194.28 214.28 10.565 0.0011526 **
## ptratio   1   189.12 209.12  5.399 0.0201474 *
## medv      1   193.58 213.58  9.866 0.0016839 **
## zn_y      1   188.45 208.45  4.733 0.0295913 *
## tax_chas  1   196.31 216.31 12.593 0.0003871 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_p <- glm(target ~ .-rad_chas-zn-log_medv-rm-log_lstat-lstat-indus, family=binomial, train_df)
```

```
summary(model_p)
```

```
##
## Call:
## glm(formula = target ~ . - rad_chas - zn - log_medv - rm - log_lstat -
##     lstat - indus, family = binomial, data = train_df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8710  -0.1667   0.0000   0.0025   3.5538
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.939e+01  6.410e+00  -6.146 7.96e-10 ***
## chas        -5.251e+03  3.082e+05  -0.017 0.986407
## nox          4.750e+01  7.247e+00   6.555 5.55e-11 ***
## age          3.292e-02  1.143e-02   2.879 0.003984 **
## dis          7.585e-01  2.260e-01   3.357 0.000789 ***
## rad          6.870e-01  1.592e-01   4.316 1.59e-05 ***
## tax         -7.915e-03  2.696e-03  -2.935 0.003331 **
## ptratio      2.859e-01  1.245e-01   2.297 0.021594 *
## medv         1.099e-01  3.697e-02   2.973 0.002953 **
## zn_y        -1.639e+00  7.866e-01  -2.084 0.037150 *
## tax_chas     1.897e+01  1.113e+03   0.017 0.986401
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 183.72  on 455  degrees of freedom
## AIC: 205.72
##
## Number of Fisher Scoring iterations: 24
```

```
model_p <- glm(target ~ .-rad_chas-zn-log_medv-rm-log_lstat-lstat-indus-chas-tax_chas, family=binomial,
```

```r
summary(model_p)
```

```
##
## Call:
## glm(formula = target ~ . - rad_chas - zn - log_medv - rm - log_lstat -
##     lstat - indus - chas - tax_chas, family = binomial, data = train_df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8321  -0.1891  -0.0112   0.0030   3.5349
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -37.511394   5.986257  -6.266 3.70e-10 ***
## nox          43.985459   6.683768   6.581 4.67e-11 ***
## age           0.034869   0.011108   3.139  0.00169 **
## dis           0.704363   0.215670   3.266  0.00109 **
## rad           0.719840   0.146091   4.927 8.34e-07 ***
## tax          -0.007773   0.002607  -2.981  0.00287 **
## ptratio       0.283659   0.117259   2.419  0.01556 *
## medv          0.108308   0.035554   3.046  0.00232 **
## zn_y         -1.718763   0.734165  -2.341  0.01923 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 197.45  on 457  degrees of freedom
## AIC: 215.45
##
## Number of Fisher Scoring iterations: 9
```

```r
model_AIC_2 <- step(glm(target~.-zn_y,family = binomial, train_df), trace=0)
```

```r
summary(model_AIC_2)
```

```
##
## Call:
## glm(formula = target ~ zn + chas + nox + age + dis + rad + tax +
##     ptratio + lstat + medv + log_lstat + tax_chas, family = binomial,
##     data = train_df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0678  -0.1310   0.0000   0.0019   3.4478
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.494e+01  7.087e+00  -4.930 8.23e-07 ***
## zn          -6.236e-02  3.393e-02  -1.838  0.06603 .
## chas        -5.026e+03  2.642e+05  -0.019  0.98482
```

```
## nox          4.611e+01  7.189e+00   6.415 1.41e-10 ***
## age          2.698e-02  1.175e-02   2.296  0.02170 *
## dis          6.295e-01  2.296e-01   2.742  0.00611 **
## rad          7.234e-01  1.702e-01   4.250 2.13e-05 ***
## tax         -8.834e-03  2.810e-03  -3.144  0.00167 **
## ptratio      3.235e-01  1.174e-01   2.756  0.00585 **
## lstat        2.104e-01  1.254e-01   1.679  0.09324 .
## medv         9.226e-02  4.894e-02   1.885  0.05938 .
## log_lstat   -2.330e+00  1.599e+00  -1.457  0.14505
## tax_chas     1.815e+01  9.537e+02   0.019  0.98482
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 180.82  on 453  degrees of freedom
## AIC: 206.82
##
## Number of Fisher Scoring iterations: 24
```

```
model_p_2 <- glm(target ~ zn + chas + nox + age + dis + rad + tax +
    ptratio + lstat + medv + log_lstat + tax_chas-tax_chas-chas-log_lstat-lstat, family=binomial, train_
```

```
summary(model_p_2)
```

```
##
## Call:
## glm(formula = target ~ zn + chas + nox + age + dis + rad + tax +
##     ptratio + lstat + medv + log_lstat + tax_chas - tax_chas -
##     chas - log_lstat - lstat, family = binomial, data = train_df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8295  -0.1752  -0.0021   0.0032   3.4191
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -37.415922   6.035013  -6.200 5.65e-10 ***
## zn           -0.068648   0.032019  -2.144  0.03203 *
## nox          42.807768   6.678692   6.410 1.46e-10 ***
## age           0.032950   0.010951   3.009  0.00262 **
## dis           0.654896   0.214050   3.060  0.00222 **
## rad           0.725109   0.149788   4.841 1.29e-06 ***
## tax          -0.007756   0.002653  -2.924  0.00346 **
## ptratio       0.323628   0.111390   2.905  0.00367 **
## medv          0.110472   0.035445   3.117  0.00183 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
```

```
## Residual deviance: 197.32  on 457  degrees of freedom
## AIC: 215.32
##
## Number of Fisher Scoring iterations: 9
```

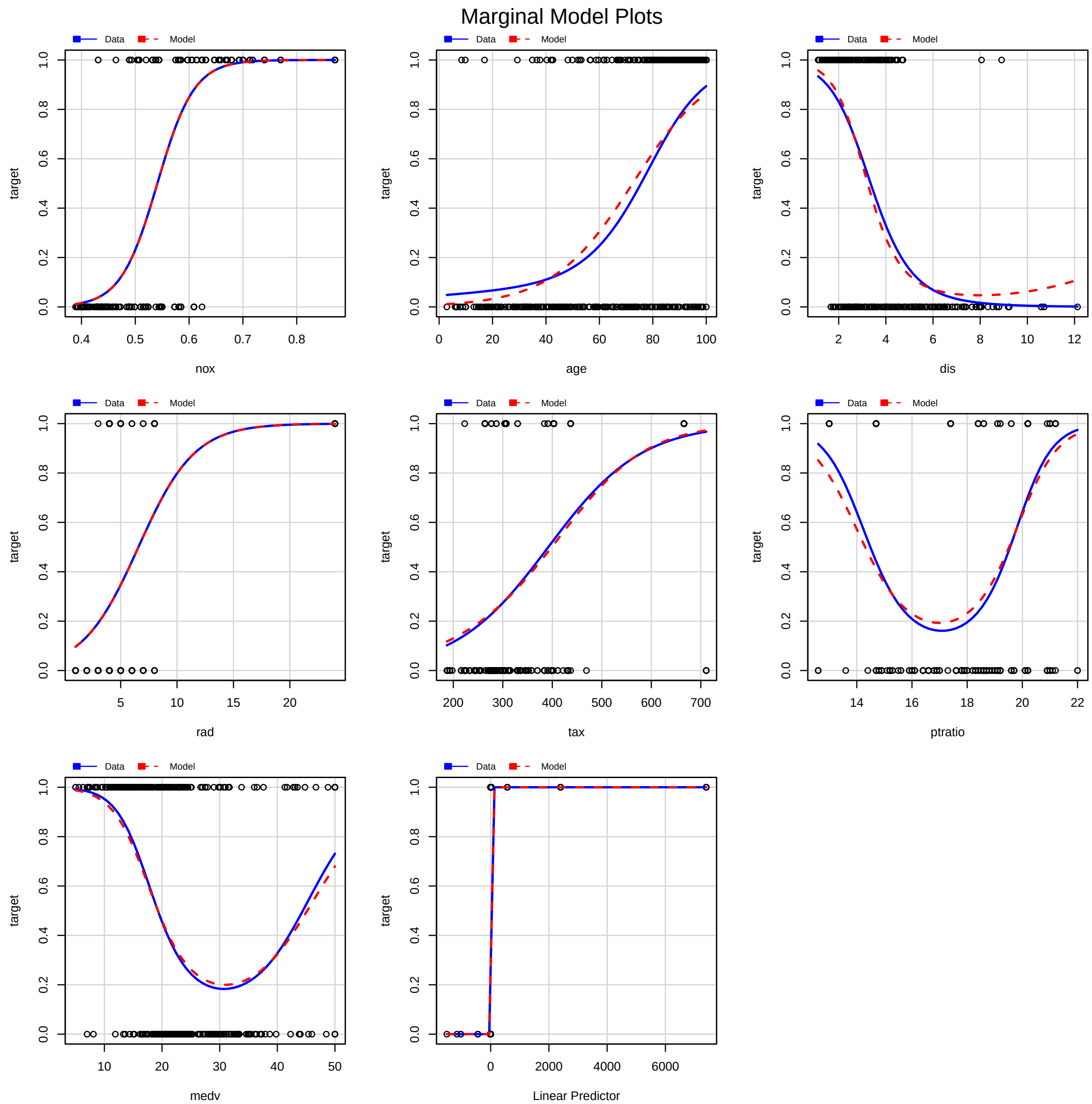
```
model_compare <- data.frame(
    model = c("full_model","model_AIC","model_p","model_AIC_2","model_p_2"),
    Deviance = rep(0.0000,5),
    AIC = rep(0.0000,5),
    Accurarcy = rep(0.0000,5),
    Sensitivity = rep(0.0000,5),
    Specificity = rep(0.0000,5),
    Precision = rep(0.0000,5),
    F1 = rep(0.0000,5),
    AUC = rep(0.0000,5),
    Nagelkerke_R_squared = rep(0.0000,5)
)
```

```
models <- list(full_model, model_AIC, model_p, model_AIC_2, model_p_2)

for (i in c(1:5)) {
  predicted_class <- ifelse(models[[i]]$fitted.values>0.5,1,0)
  confusion_matrix <- confusionMatrix(as.factor(predicted_class),
                                      as.factor(train_df$target),positive = "1")

  model_compare[i,] <- c(model_compare[i,1],round(models[[i]]$deviance,4), models[[i]]$aic,
                            confusion_matrix$overall[1],
                            confusion_matrix$byClass[1],
                            confusion_matrix$byClass[2],
                            confusion_matrix$byClass[3],
                            2*confusion_matrix$byClass[1]*confusion_matrix$byClass[3]/
                              (confusion_matrix$byClass[1]+confusion_matrix$byClass[3]),
                            auc(roc(train_df$target, models[[i]]$fitted.values)),
                            (1-exp((models[[i]]$dev-models[[i]]$null)/
                                     length(models[[i]]$residuals)))/
                              (1-exp(-models[[i]]$null/length(models[[i]]$residuals)))
                            )
}
```

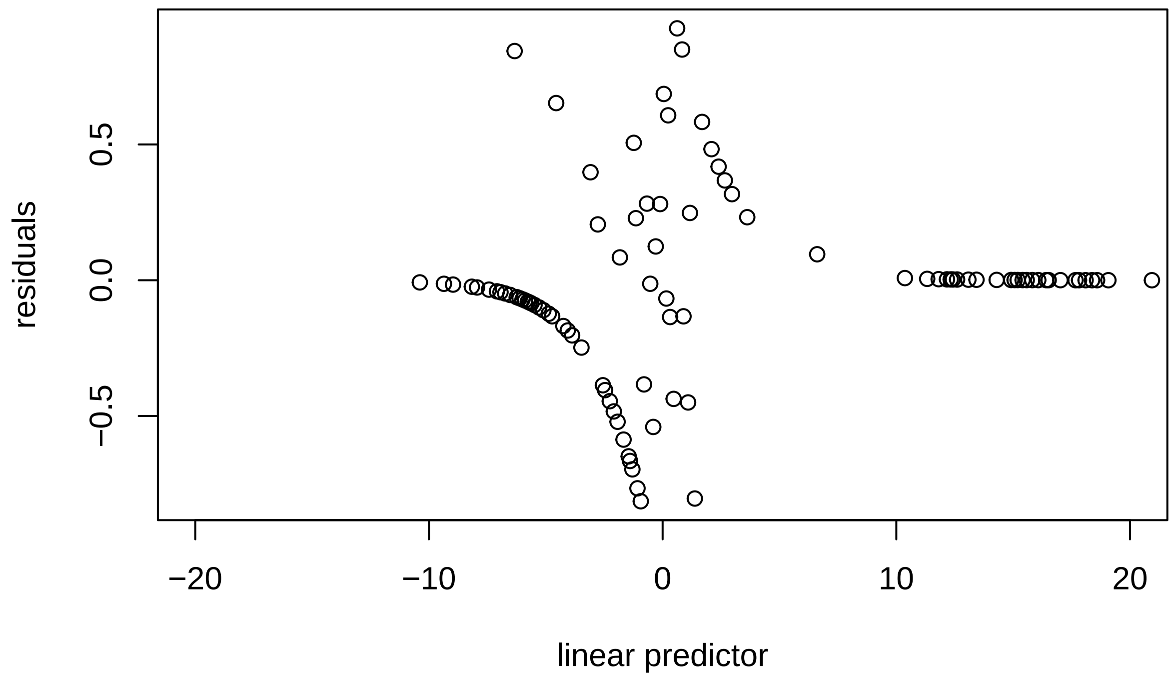
```
model_compare[,c(2:10)] <- sapply(model_compare[,c(2:10)],as.numeric)
model_compare
```

```
##         model Deviance      AIC Accurarcy Sensitivity Specificity Precision
## 1  full_model 176.9952 212.9952 0.9291845   0.9170306   0.9409283 0.9375000
## 2   model_AIC 183.7167 205.7167 0.9206009   0.9126638   0.9282700 0.9247788
## 3     model_p 197.4519 215.4519 0.9141631   0.9039301   0.9240506 0.9200000
## 4 model_AIC_2 180.8188 206.8188 0.9206009   0.9170306   0.9240506 0.9210526
## 5   model_p_2 197.3229 215.3229 0.9120172   0.9039301   0.9198312 0.9159292
##          F1       AUC Nagelkerke_R_squared
## 1 0.9271523 0.9774473            0.8459333
## 2 0.9186813 0.9756048            0.8388503
## 3 0.9118943 0.9718645            0.8240547
## 4 0.9190372 0.9757338            0.8419166
## 5 0.9098901 0.9719382            0.8241958
```

```r
marginalModelPlots(model_AIC,~nox+age+dis+rad+tax+ptratio+medv,layout =c(3,3))
```




```r
residual_df <- mutate(train_df, residuals=residuals(model_AIC,type="deviance"), linpred=predict(model_AI
gdf <- group_by(residual_df, cut(linpred, breaks=unique(quantile(linpred,(1:100)/101))))
diagdf <- summarise(gdf, residuals=mean(residuals), linpred=mean(linpred))
plot(residuals ~ linpred, diagdf, xlab="linear predictor",xlim=c(-20,20))
```

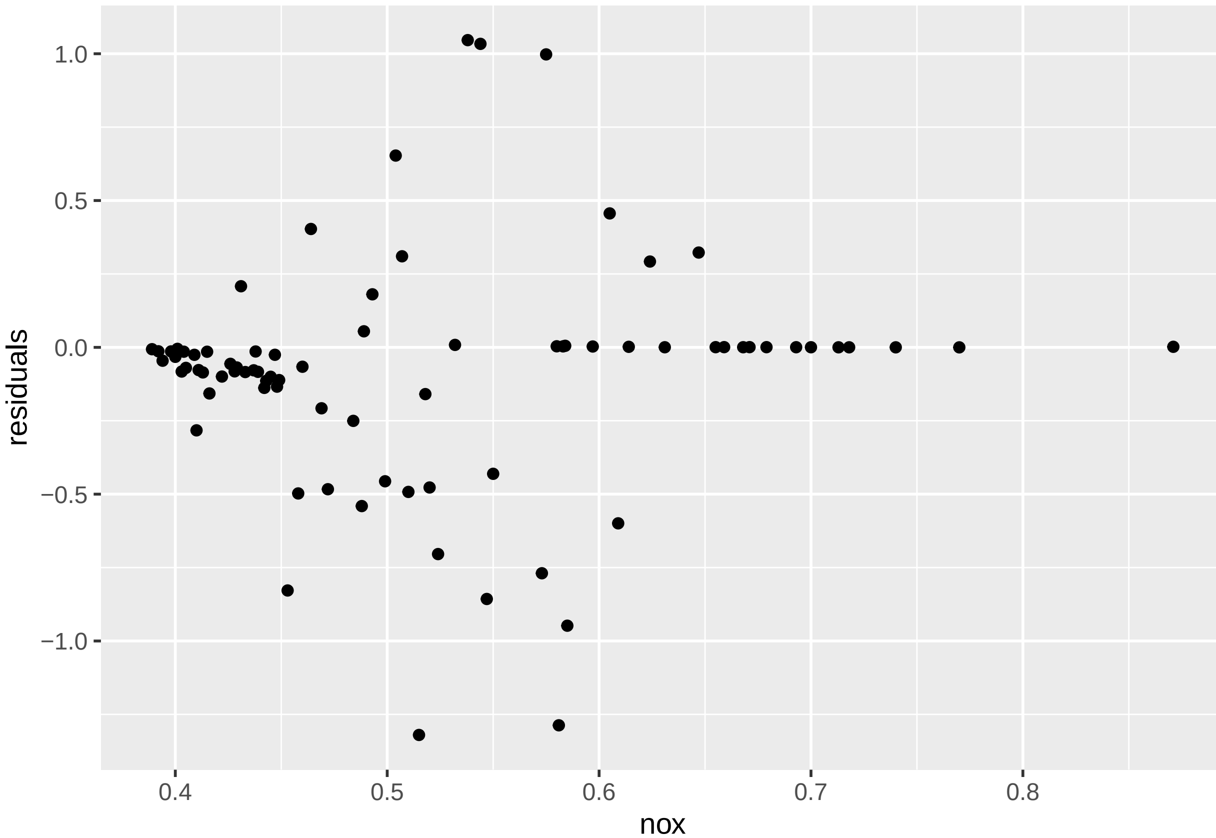


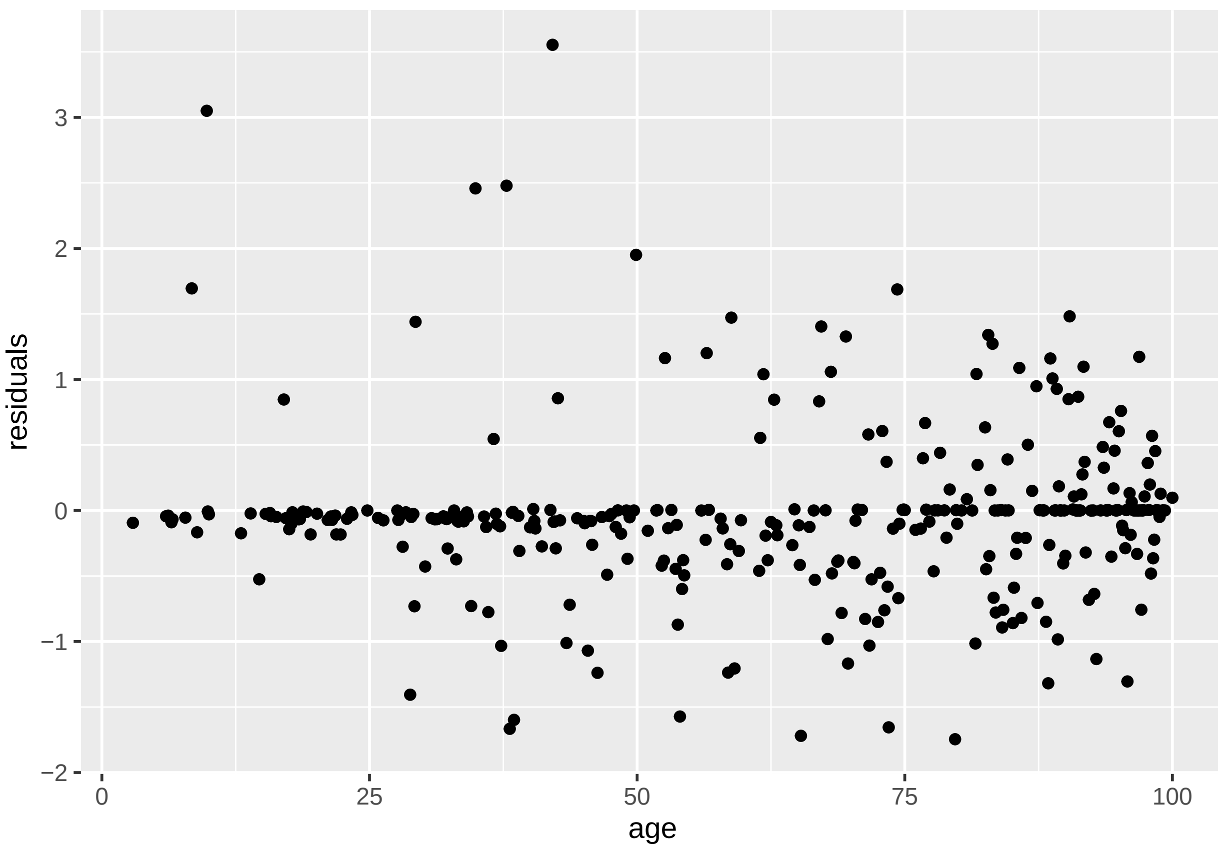

```
model_AIC
```

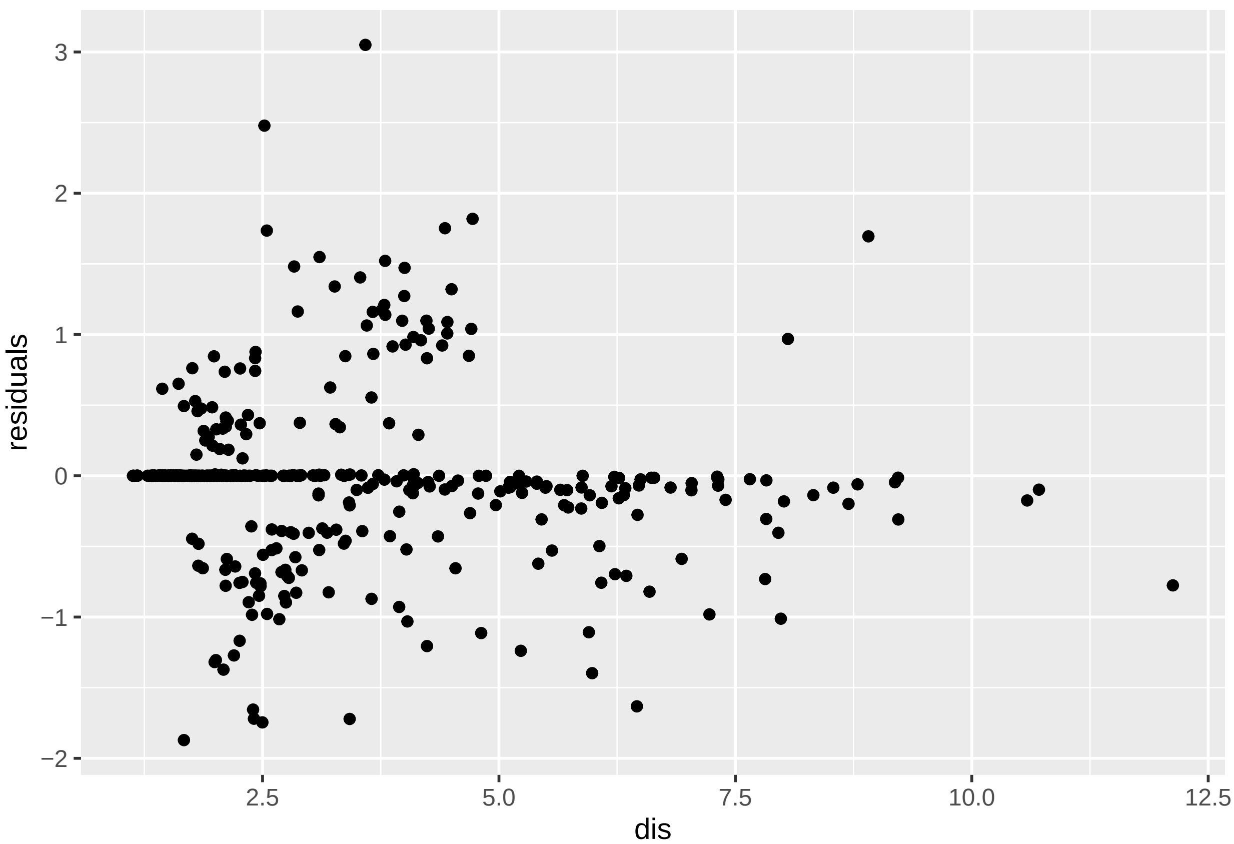
```
##
## Call:  glm(formula = target ~ chas + nox + age + dis + rad + tax + ptratio +
##     medv + zn_y + tax_chas, family = binomial, data = train_df)
##
## Coefficients:
## (Intercept)         chas          nox          age          dis          rad
##  -3.939e+01   -5.251e+03    4.750e+01    3.292e-02    7.585e-01    6.870e-01
##         tax      ptratio         medv         zn_y     tax_chas
##  -7.915e-03    2.859e-01    1.099e-01   -1.639e+00    1.897e+01
##
## Degrees of Freedom: 465 Total (i.e. Null);  455 Residual
## Null Deviance:       645.9
## Residual Deviance: 183.7     AIC: 205.7
```

```
predictors <- c("nox","age","dis","rad","tax","ptratio","medv")

residual_df <- mutate(train_df, residuals=residuals(model_AIC,type="deviance"))
gg_plots <- list()

for (i in c(1:length(predictors))) {
    gdf <- group_by(residual_df, .dots = predictors[i])
    diagdf <- summarise(gdf, residuals=mean(residuals))
    print(ggplot(diagdf, aes_string(x=predictors[i],y="residuals")) + geom_point())
}
```

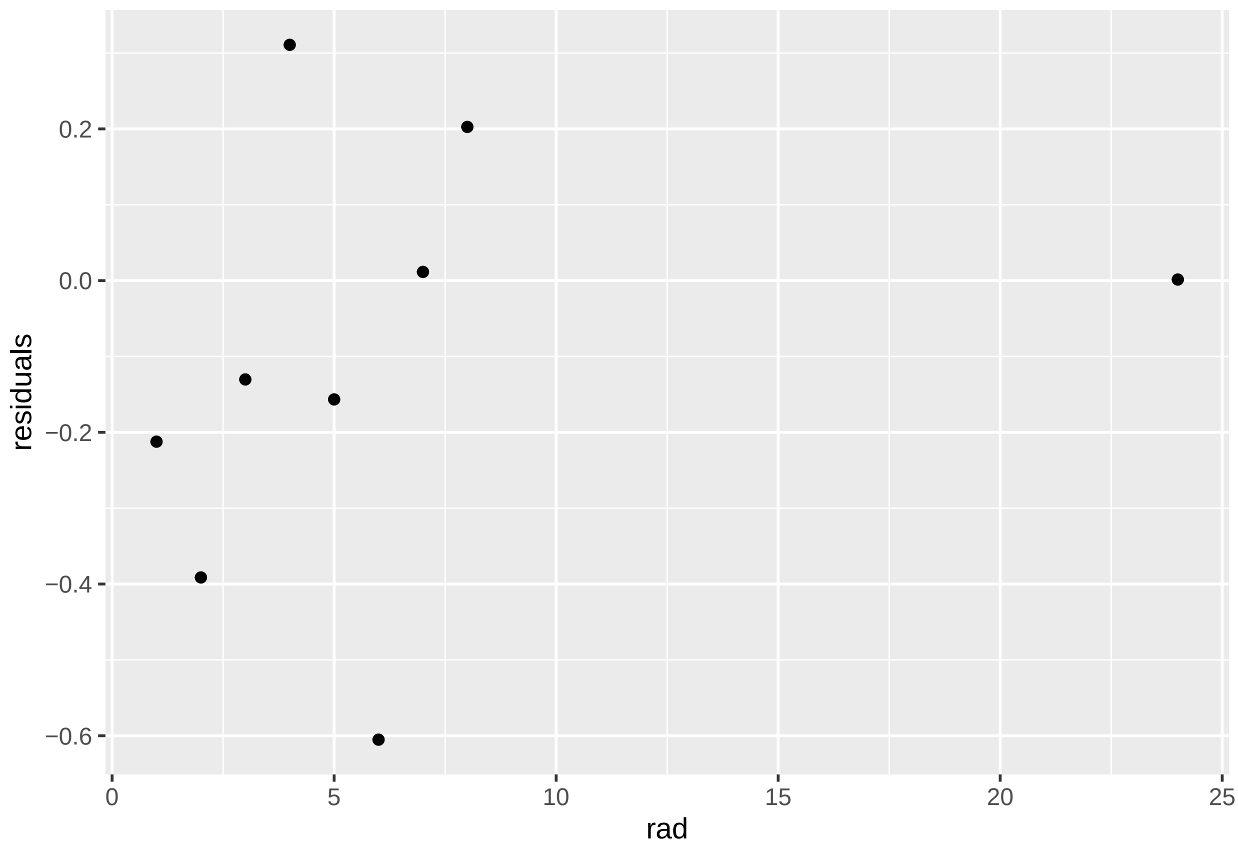
```
## Warning: The `.dots` argument of `group_by()` is deprecated as of dplyr 1.0.0.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```



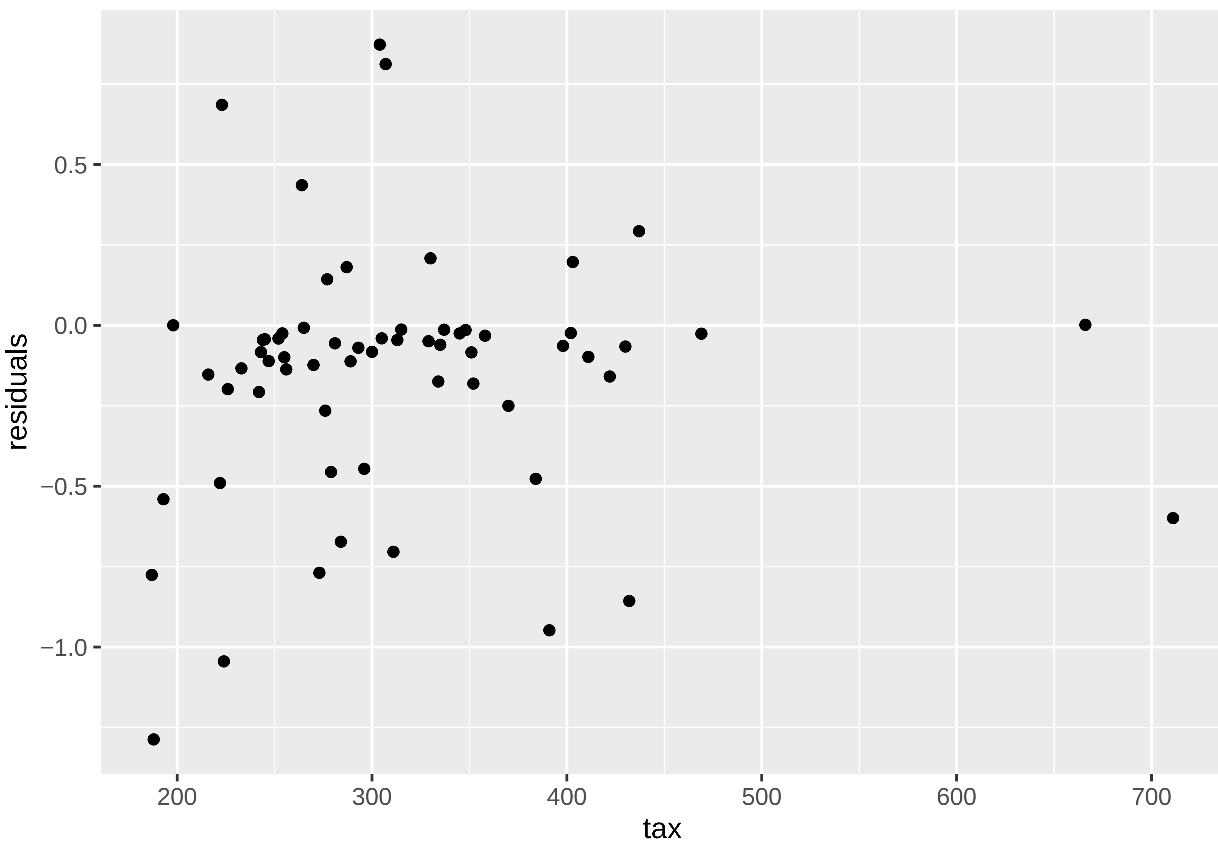

residuals
3
2
1
0
−1
−2
0
25
50
75
100
age



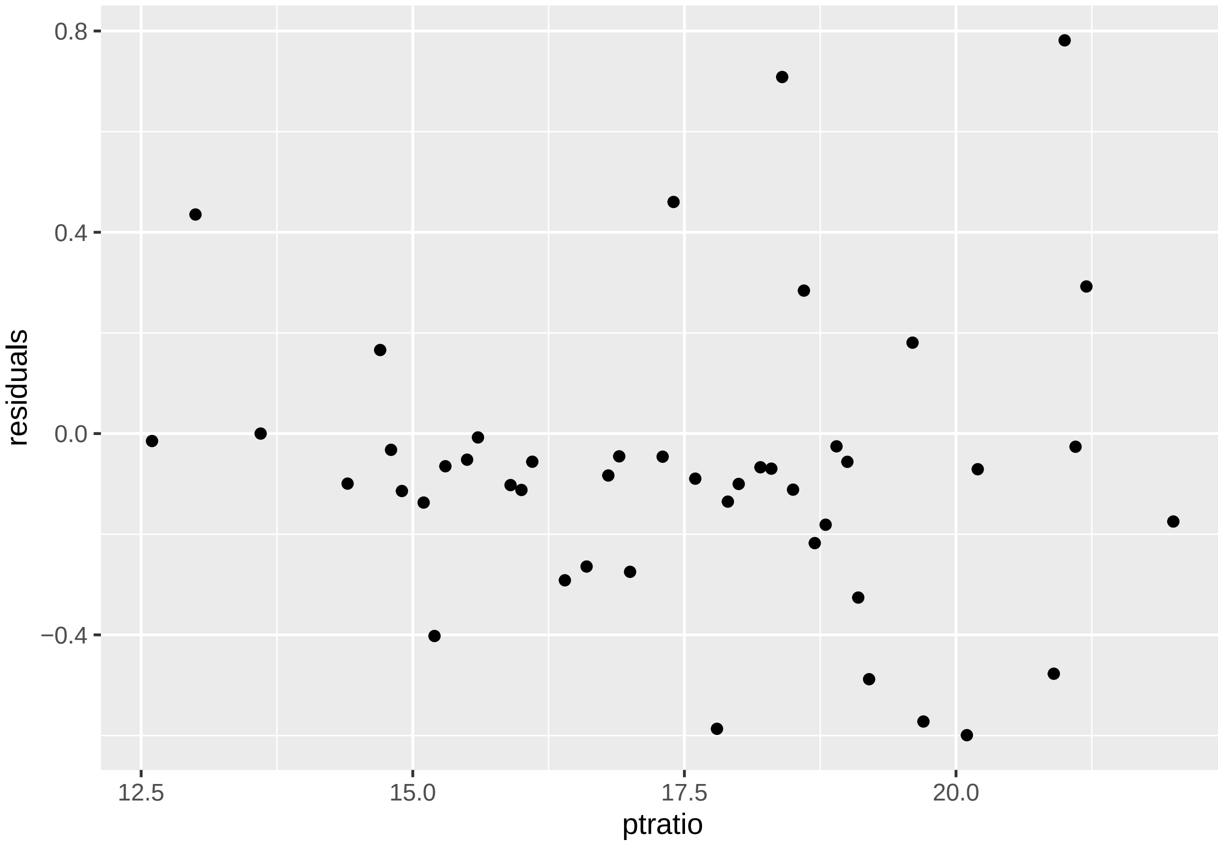
3
2
1
0
−1
−2
residuals
2.5
5.0
7.5
10.0
12.5
dis

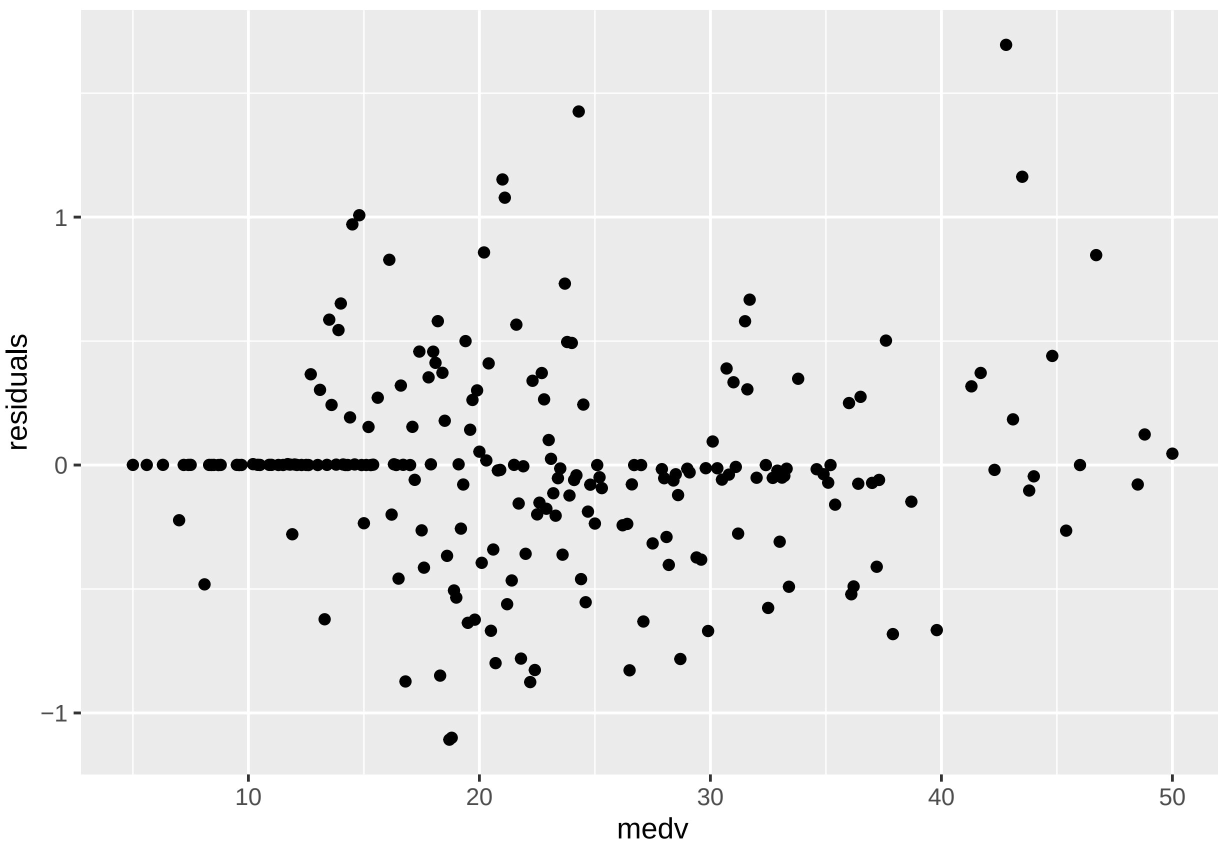


0.2
0.0
−0.2
−0.4
−0.6
residuals
0
5
10
15
20
25
rad

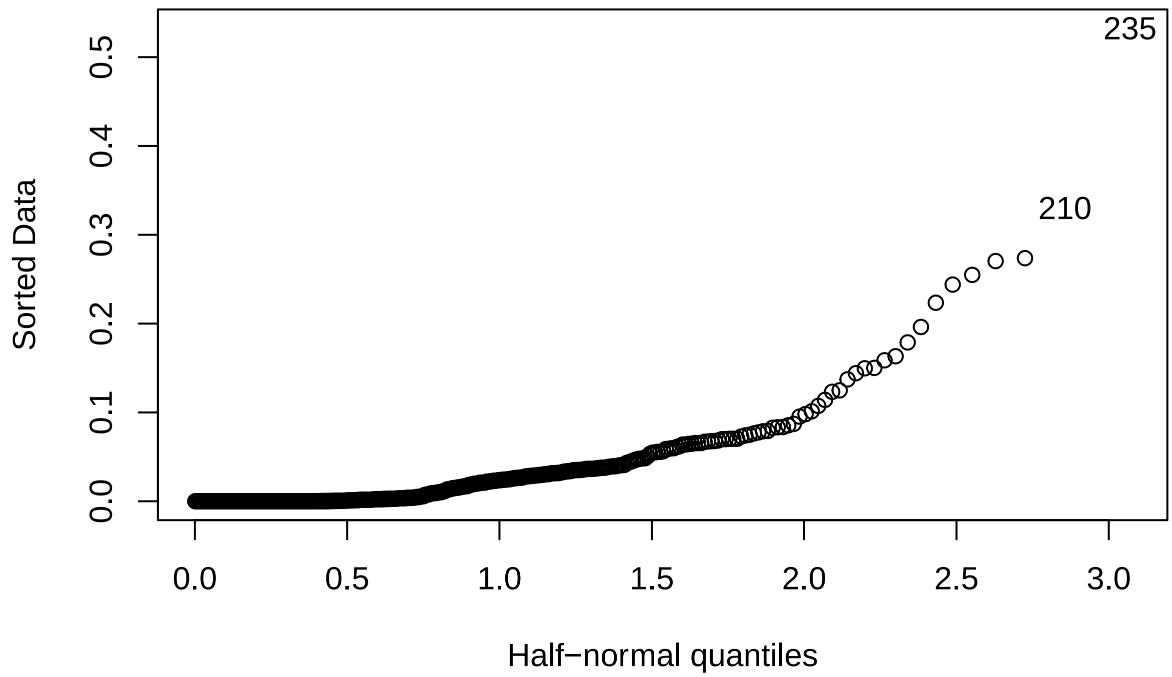


0.5
0.0
−0.5
−1.0
residuals
200
300
400
500
600
700
tax


0.8
0.4
0.0
−0.4
residuals
12.5
15.0
17.5
20.0
ptratio




```
halfnorm(hatvalues(model_AIC))
```




```
train_df[c(210,235),]
```

```
##     zn indus chas  nox    rm  age    dis rad tax ptratio lstat medv target zn_y
## 210  0 13.89    1 0.55 5.951 93.8 2.8893   5 276    16.4 17.92 21.5      0    0
## 235  0 13.89    1 0.55 6.373 92.4 3.3633   5 276    16.4 10.50 23.0      0    0
##     log_lstat log_medv tax_chas rad_chas
## 210  2.885917 3.068053      276        5
## 235  2.351375 3.135494      276        5
```