Lab 02

Evan Ray

9/24/2019

Read in data set and fix variable names

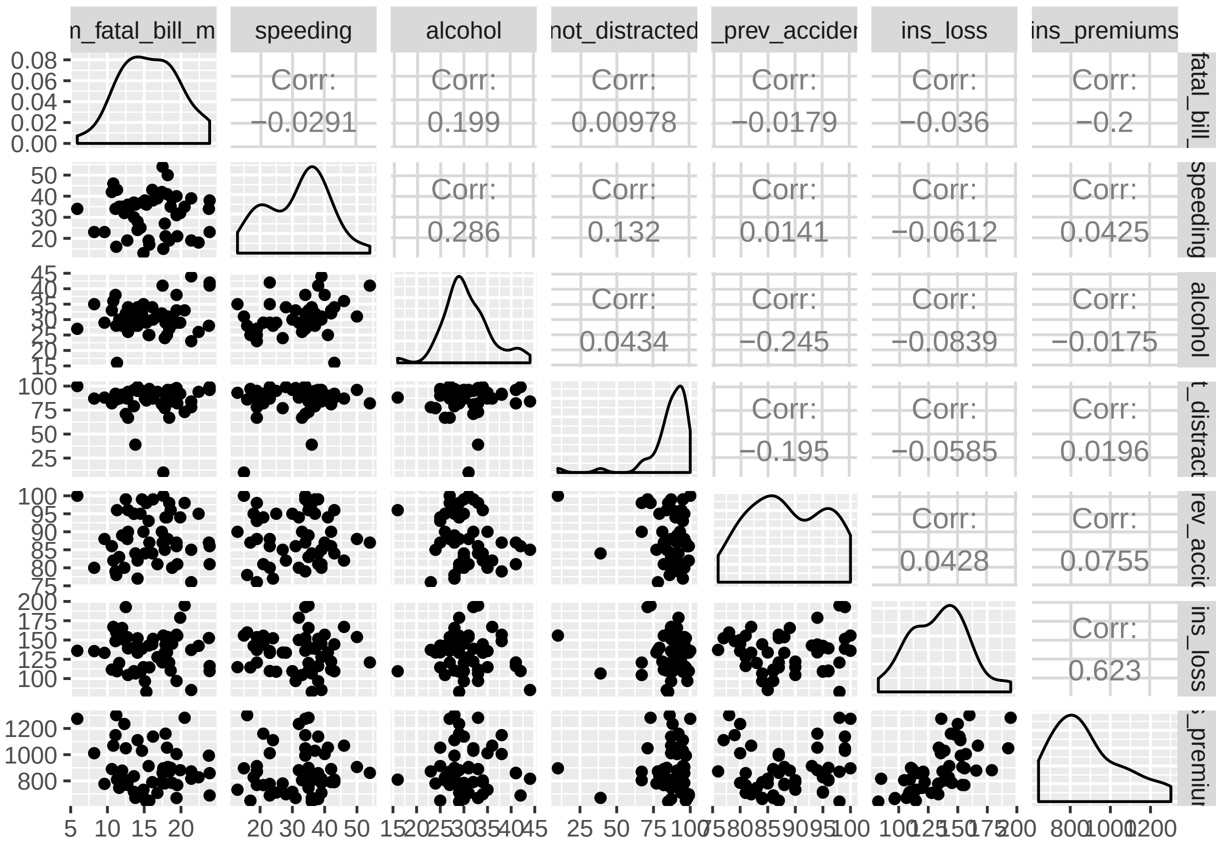
```
bad_drivers <- read_csv("data/bad-drivers.csv")
```

```
## Parsed with column specification:
## cols(
##   State = col_character(),
##   `Number of drivers involved in fatal collisions per billion miles` = col_double(),
##   `Percentage Of Drivers Involved In Fatal Collisions Who Were Speeding` = col_double(),
##   `Percentage Of Drivers Involved In Fatal Collisions Who Were Alcohol-Impaired` = col_double(),
##   `Percentage Of Drivers Involved In Fatal Collisions Who Were Not Distracted` = col_double(),
##   `Percentage Of Drivers Involved In Fatal Collisions Who Had Not Been Involved In Any Previous Acci
##   `Car Insurance Premiums ($)` = col_double(),
##   `Losses incurred by insurance companies for collisions per insured driver ($)` = col_double()
## )
```

```
names(bad_drivers) <- c(
  "state",
  "num_fatal_bill_miles",
  "speeding",
  "alcohol",
  "not_distracted",
  "no_prev_accidents",
  "ins_premiums",
  "ins_loss"
)
```

Exploratory plots

```
bad_drivers %>%
  select(c("num_fatal_bill_miles", "speeding", "alcohol", "not_distracted",
    "no_prev_accidents", "ins_loss", "ins_premiums")) %>%
  ggpairs()
```



Here's what I see in the pairs plots:

- Things about relationships between explanatory and response variables:
  - There is an increasing and not-quite-linear relationship between `ins_loss` and `ins_premiums`.
  - There is a non-linear relationship between `no_prev_accidents` and `ins_premiums`. It looks like a quadratic term will be necessary there.
  - There might be a weak quadratic relationship between `num_fatal_bill_miles` and `ins_premiums`. Will need to investigate that more.
  - There might be a relationship between `not_distracted` and `ins_premiums`, but what we're seeing is mainly driven by two influential observations with low values of `not_distracted`. I don't trust it.
  - Nothing apparent going on for `speeding` and `alcohol`
- Things about whether the model is OK
  - The response variable is skewed slightly to the right, and in the plot of `ins_loss` vs `ins_premiums` there is more variability in `ins_premiums` when `ins_loss` is large than when it is small. This suggests that we might consider a transformation of the `ins_premiums` variable.
  - I don't need to be particularly worried about multicollinearity

Regression Analysis and Cross-Validation

I'm going to do the cross-validation for each individual model as I go along. I'll set up the cross-validation folds here and use the same folds for all models I consider.

```
set.seed(90811)
val_folds <- createFolds(y = bad_drivers$ins_premiums, k = 5)
```

Simple Linear Regression Model

I'll use `ins_loss` as my explanatory variable since it has the closest to a linear relationship with the response.

```
reg01 <- lm(ins_premiums ~ ins_loss, data = bad_drivers)
summary(reg01)
```

```
##
## Call:
## lm(formula = ins_premiums ~ ins_loss, data = bad_drivers)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -213.33  -96.75  -40.11  112.24  379.97
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 285.3251   109.6689   2.602   0.0122 *
## ins_loss      4.4733     0.8021   5.577 1.04e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 140.9 on 49 degrees of freedom
## Multiple R-squared:  0.3883, Adjusted R-squared:  0.3758
## F-statistic:  31.1 on 1 and 49 DF,  p-value: 1.043e-06
```

```
val_mses <- rep(NA, 5)
for(i in seq_len(5)) {
  train_drivers <- bad_drivers %>% dplyr::slice(-val_folds[[i]])
  val_drivers <- bad_drivers %>% dplyr::slice(val_folds[[i]])

  fit <- lm(ins_premiums ~ ins_loss, data = train_drivers)
  val_mses[i] <- mean((val_drivers$ins_premiums - predict(fit, newdata = val_drivers))^2)
}

mean(val_mses)
```

```
## [1] 19194.81
```

Multiple Regression Model

Attempt 1

To start with I'll try all the variables with degree 2 polynomial terms where the plots suggested they might be appropriate.

```
reg02a <- lm(ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) + speeding + alcohol
             + not_distracted+ poly(no_prev_accidents, 2, raw = TRUE) + poly(ins_loss, 2, raw = TRUE),
           data = bad_drivers)
summary(reg02a)
```

```
##
## Call:
## lm(formula = ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) +
##     speeding + alcohol + not_distracted + poly(no_prev_accidents,
##     2, raw = TRUE) + poly(ins_loss, 2, raw = TRUE), data = bad_drivers)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -168.955  -88.794   -9.878   79.562  305.422
##
## Coefficients:
##                                             Estimate Std. Error t value
## (Intercept)                                1.177e+04  3.571e+03   3.295
## poly(num_fatal_bill_miles, 2, raw = TRUE)1 -5.853e+01  2.879e+01  -2.033
## poly(num_fatal_bill_miles, 2, raw = TRUE)2  1.663e+00  8.988e-01   1.850
## speeding                                   2.417e+00  2.006e+00   1.205
## alcohol                                    2.323e+00  4.029e+00   0.577
## not_distracted                             9.121e-01  1.279e+00   0.713
## poly(no_prev_accidents, 2, raw = TRUE)1    -2.585e+02  8.184e+01  -3.158
## poly(no_prev_accidents, 2, raw = TRUE)2     1.470e+00  4.615e-01   3.186
## poly(ins_loss, 2, raw = TRUE)1             6.154e+00  6.407e+00   0.961
## poly(ins_loss, 2, raw = TRUE)2            -8.862e-03  2.334e-02  -0.380
##                                            Pr(>|t|)
## (Intercept)                                 0.00203 **
## poly(num_fatal_bill_miles, 2, raw = TRUE)1  0.04855 *
## poly(num_fatal_bill_miles, 2, raw = TRUE)2  0.07155 .
## speeding                                    0.23519
## alcohol                                     0.56737
## not_distracted                              0.47971
## poly(no_prev_accidents, 2, raw = TRUE)1     0.00298 **
## poly(no_prev_accidents, 2, raw = TRUE)2     0.00276 **
## poly(ins_loss, 2, raw = TRUE)1              0.34241
## poly(ins_loss, 2, raw = TRUE)2              0.70608
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 125 on 41 degrees of freedom
## Multiple R-squared:  0.5966, Adjusted R-squared:  0.5081
## F-statistic: 6.738 on 9 and 41 DF,  p-value: 7.292e-06
```

```r
val_mses <- rep(NA, 5)
for(i in seq_len(5)) {
  train_drivers <- bad_drivers %>% dplyr::slice(-val_folds[[i]])
  val_drivers <- bad_drivers %>% dplyr::slice(val_folds[[i]])

  fit <- lm(ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) + speeding + alcohol +
    not_distracted + poly(no_prev_accidents, 2, raw = TRUE) + poly(ins_loss, 2, raw = TRUE),
    data = train_drivers)
  val_mses[i] <- mean((val_drivers$ins_premiums - predict(fit, newdata = val_drivers))^2)
}

mean(val_mses)
```

```
## [1] 17287.74
```

These cross-validation results indicate that our model is better than the simple linear regression model above.

I'm going to try to fiddle with a few things to see if we can do a little better. To start with, what happens if we take out those variables that look irrelevant (based on both the plots and the large p-values)?

Attempt 2

```r
val_mses <- rep(NA, 5)
for(i in seq_len(5)) {
  train_drivers <- bad_drivers %>% dplyr::slice(-val_folds[[i]])
  val_drivers <- bad_drivers %>% dplyr::slice(val_folds[[i]])

  fit <- lm(ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) +
    poly(no_prev_accidents, 2, raw = TRUE) + poly(ins_loss, 2, raw = TRUE),
    data = train_drivers)
  val_mses[i] <- mean((val_drivers$ins_premiums - predict(fit, newdata = val_drivers))^2)
}

mean(val_mses)
```

```
## [1] 17081.58
```

```r
reg02b <- lm(ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) +
  poly(no_prev_accidents, 2, raw = TRUE) + poly(ins_loss, 2, raw = TRUE),
  data = bad_drivers)
summary(reg02b)
```

```
##
## Call:
## lm(formula = ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) +
##     poly(no_prev_accidents, 2, raw = TRUE) + poly(ins_loss, 2,
##     raw = TRUE), data = bad_drivers)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -172.37  -91.08  -16.46   80.46  287.53
##
## Coefficients:
##                                              Estimate Std. Error t value
## (Intercept)                                 1.039e+04  3.468e+03   2.997
## poly(num_fatal_bill_miles, 2, raw = TRUE)1 -6.566e+01  2.749e+01  -2.388
## poly(num_fatal_bill_miles, 2, raw = TRUE)2  1.884e+00  8.519e-01   2.212
## poly(no_prev_accidents, 2, raw = TRUE)1    -2.174e+02  7.802e+01  -2.786
## poly(no_prev_accidents, 2, raw = TRUE)2     1.234e+00  4.389e-01   2.812
## poly(ins_loss, 2, raw = TRUE)1              4.332e+00  6.098e+00   0.710
## poly(ins_loss, 2, raw = TRUE)2             -2.202e-03  2.222e-02  -0.099
##                                            Pr(>|t|)
## (Intercept)                                 0.00447 **
## poly(num_fatal_bill_miles, 2, raw = TRUE)1  0.02128 *
## poly(num_fatal_bill_miles, 2, raw = TRUE)2  0.03224 *
## poly(no_prev_accidents, 2, raw = TRUE)1     0.00784 **
## poly(no_prev_accidents, 2, raw = TRUE)2     0.00733 **
## poly(ins_loss, 2, raw = TRUE)1              0.48123
## poly(ins_loss, 2, raw = TRUE)2              0.92150
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 125.2 on 44 degrees of freedom
## Multiple R-squared:  0.5662, Adjusted R-squared:  0.507
## F-statistic: 9.571 on 6 and 44 DF,  p-value: 9.918e-07
```

The degree 2 polynomial term on `ins_loss` is very close to 0 and has the opposite sign as what I expected from the initial plot. The p-value is also very large. Maybe I should get rid of that?

Attempt 3

```
val_mses <- rep(NA, 5)
for(i in seq_len(5)) {
  train_drivers <- bad_drivers %>% dplyr::slice(-val_folds[[i]])
  val_drivers <- bad_drivers %>% dplyr::slice(val_folds[[i]])

  fit <- lm(ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) +
    poly(no_prev_accidents, 2, raw = TRUE) + ins_loss,
    data = train_drivers)
  val_mses[i] <- mean((val_drivers$ins_premiums - predict(fit, newdata = val_drivers))^2)
}

mean(val_mses)
```

```
## [1] 16448.77
```

```
reg02c <- lm(ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) + poly(no_prev_accidents, 2, raw
             data = bad_drivers)
summary(reg02c)
```

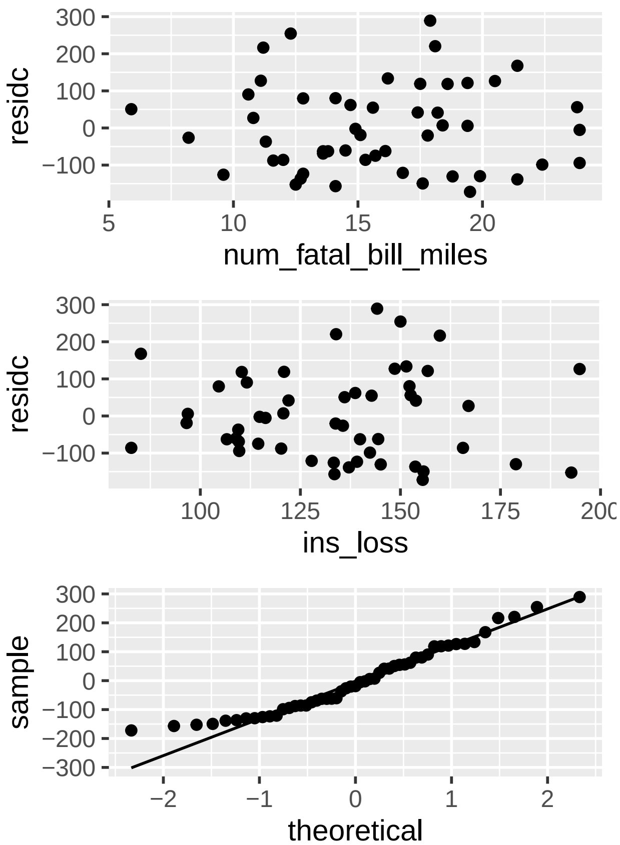
```
##
## Call:
## lm(formula = ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) +
##     poly(no_prev_accidents, 2, raw = TRUE) + ins_loss, data = bad_drivers)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -172.11  -91.07  -18.92   80.00  289.21
##
## Coefficients:
##                                              Estimate Std. Error t value
## (Intercept)                                10428.6912  3412.5269   3.056
## poly(num_fatal_bill_miles, 2, raw = TRUE)1   -65.8226    27.1422  -2.425
## poly(num_fatal_bill_miles, 2, raw = TRUE)2     1.8871     0.8419   2.242
## poly(no_prev_accidents, 2, raw = TRUE)1     -217.1650    77.1285  -2.816
## poly(no_prev_accidents, 2, raw = TRUE)2        1.2326     0.4338   2.842
## ins_loss                                       3.7319     0.7389   5.050
##                                            Pr(>|t|)
## (Intercept)                                 0.00376 **
## poly(num_fatal_bill_miles, 2, raw = TRUE)1  0.01938 *
## poly(num_fatal_bill_miles, 2, raw = TRUE)2  0.02997 *
## poly(no_prev_accidents, 2, raw = TRUE)1     0.00720 **
## poly(no_prev_accidents, 2, raw = TRUE)2     0.00672 **
## ins_loss                                   7.78e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 123.8 on 45 degrees of freedom
## Multiple R-squared:  0.5661, Adjusted R-squared:  0.5179
## F-statistic: 11.74 on 5 and 45 DF,  p-value: 2.702e-07
```

Let's look at some diagnostic plots for this model to see if there are any other issues we should address.

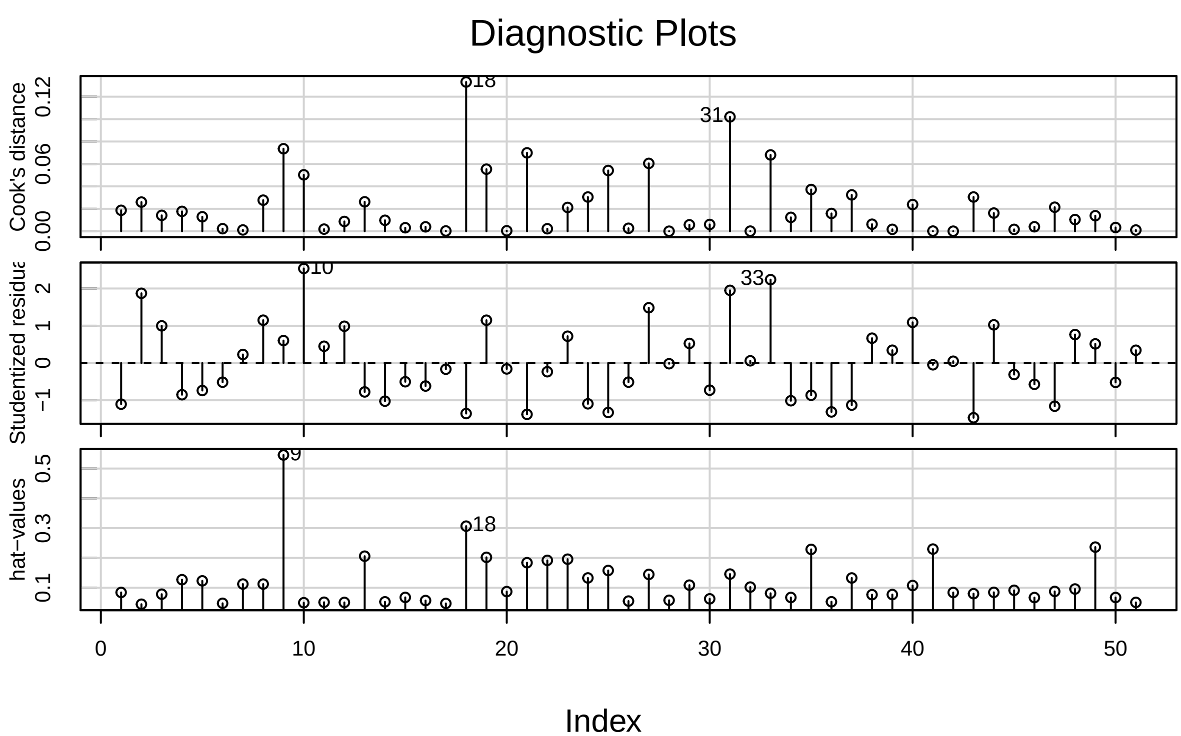
```r
bad_drivers <- bad_drivers %>%
  mutate(
    residc = residuals(reg02c)
  )

p1 <- ggplot(
    data = bad_drivers,
    mapping = aes(x = num_fatal_bill_miles, y = residc)) +
  geom_point()
p2 <- ggplot(
    data = bad_drivers,
    mapping = aes(x = no_prev_accidents, y = residc)) +
  geom_point()
p3 <- ggplot(
    data = bad_drivers,
    mapping = aes(x = ins_loss, y = residc)) +
  geom_point()
p4 <- ggplot(
    data = bad_drivers,
    mapping = aes(x = residc)) +
  geom_density()
p5 <- ggplot(
    data = bad_drivers,
    mapping = aes(sample = residc)) +
  stat_qq() +
  stat_qq_line()

grid.arrange(p1, p2, p3, p4, p5)
```




```r
car::influenceIndexPlot(reg02c,
  vars = c("Cook", "Studentized", "hat"))
```



```r
2 * 6 / nrow(bad_drivers) # threshold for when we have to worry about leverage ("hat-values")
```

```
## [1] 0.2352941
```

- The residuals are skewed right, but not horribly. This is not a serious problem.
- There are no indications of further non-linearities in any of these variables
- There is fairly constant variance of the residuals across the range of values for each explanatory variable.
- There are no outliers.
- All Cook's distances are less than 1, no need to worry
- Only a couple of studentized residuals slightly larger than 2, no need to worry
- Observation 9 has high leverage.

```r
bad_drivers$state[9]
```

```
## [1] "District of Columbia"
```

The District of Columbia may be influencing our predictions. We might consider not including it in this analysis of insurance premiums at the state level.

Attempt 4

```r
bad_drivers_no_dc <- bad_drivers %>% slice(-9)

reg02d <- lm(ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) +
  poly(no_prev_accidents, 2, raw = TRUE) + ins_loss,
  data = bad_drivers_no_dc)
summary(reg02d)
```

```
##
## Call:
## lm(formula = ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) +
##     poly(no_prev_accidents, 2, raw = TRUE) + ins_loss, data = bad_drivers_no_dc)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -181.12  -84.31  -13.93   80.51  285.27
##
## Coefficients:
##                                             Estimate Std. Error t value
## (Intercept)                                9912.2360  3542.2648   2.798
## poly(num_fatal_bill_miles, 2, raw = TRUE)1  -51.1680    36.5938  -1.398
## poly(num_fatal_bill_miles, 2, raw = TRUE)2    1.4634     1.1017   1.328
## poly(no_prev_accidents, 2, raw = TRUE)1    -207.7048    79.2520  -2.621
## poly(no_prev_accidents, 2, raw = TRUE)2       1.1757     0.4469   2.631
## ins_loss                                      3.7816     0.7488   5.050
##                                            Pr(>|t|)
## (Intercept)                                  0.0076 **
## poly(num_fatal_bill_miles, 2, raw = TRUE)1   0.1690
## poly(num_fatal_bill_miles, 2, raw = TRUE)2   0.1909
## poly(no_prev_accidents, 2, raw = TRUE)1      0.0120 *
## poly(no_prev_accidents, 2, raw = TRUE)2      0.0117 *
## ins_loss                                   8.16e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 124.7 on 44 degrees of freedom
```

```
## Multiple R-squared:  0.5239, Adjusted R-squared:  0.4698
## F-statistic: 9.683 on 5 and 44 DF,  p-value: 2.753e-06
```

Based on the p-values, it looks like the evidence for a quadratic term in `num_fatal_bill_miles` is much weaker now that DC is not included.

Let's see what cross-validation has to say about dropping that term. Note that my validation folds were based on the data set including DC. To get comparable results, I'll just drop DC within my cross-validation loop.

First, the model with `poly(num_fatal_bill_miles, 2, raw = TRUE)`, but now fit without including DC.

```
val_mses <- rep(NA, 5)
for(i in seq_len(5)) {
  train_drivers <- bad_drivers %>%
    dplyr::slice(-val_folds[[i]]) %>%
    filter(state != "District of Columbia")
  val_drivers <- bad_drivers %>%
    dplyr::slice(val_folds[[i]]) %>%
    filter(state != "District of Columbia")

  fit <- lm(ins_premiums ~ poly(num_fatal_bill_miles, 2, raw = TRUE) +
    poly(no_prev_accidents, 2, raw = TRUE) + ins_loss,
    data = train_drivers)
  val_mses[i] <- mean((val_drivers$ins_premiums - predict(fit, newdata = val_drivers))^2)
}

mean(val_mses)
```

```
## [1] 16441.67
```

Now, the model without `num_fatal_bill_miles`

```
val_mses <- rep(NA, 5)
for(i in seq_len(5)) {
  train_drivers <- bad_drivers %>%
    dplyr::slice(-val_folds[[i]]) %>%
    filter(state != "District of Columbia")
  val_drivers <- bad_drivers %>%
    dplyr::slice(val_folds[[i]]) %>%
    filter(state != "District of Columbia")

  fit <- lm(ins_premiums ~ poly(no_prev_accidents, 2, raw = TRUE) + ins_loss,
    data = train_drivers)
  val_mses[i] <- mean((val_drivers$ins_premiums - predict(fit, newdata = val_drivers))^2)
}

mean(val_mses)
```

```
## [1] 16222.05
```

Looks like we should drop `num_fatal_bill_miles` from the model.

Final Model!

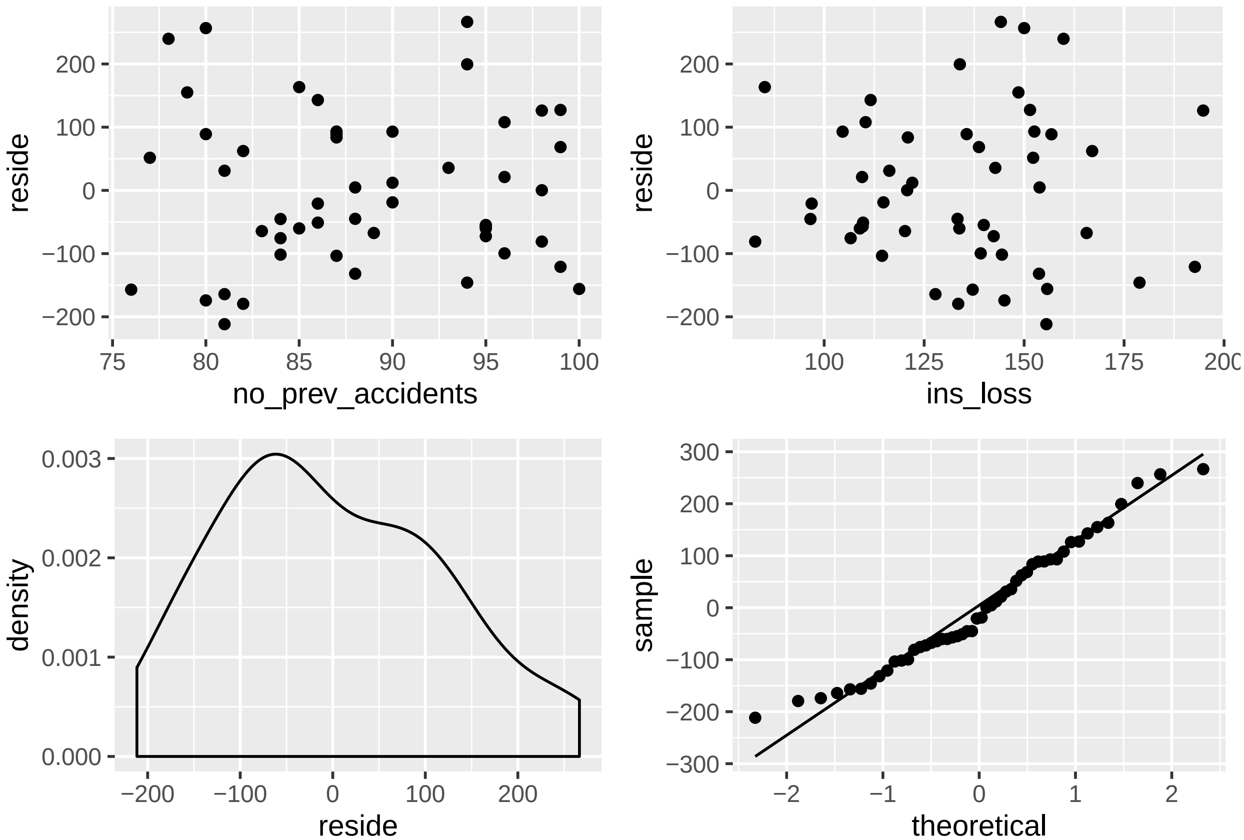
```
reg02e <- lm(ins_premiums ~ poly(no_prev_accidents, 2, raw = TRUE) + ins_loss,
  data = bad_drivers_no_dc)
summary(reg02e)
```

```
##
## Call:
## lm(formula = ins_premiums ~ poly(no_prev_accidents, 2, raw = TRUE) +
##     ins_loss, data = bad_drivers_no_dc)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -211.62  -79.65  -19.90   88.94  266.53
##
## Coefficients:
##                                          Estimate Std. Error t value
## (Intercept)                             9523.4842  3530.1025   2.698
## poly(no_prev_accidents, 2, raw = TRUE)1 -207.3916    79.2668  -2.616
## poly(no_prev_accidents, 2, raw = TRUE)2    1.1666     0.4469   2.610
## ins_loss                                   3.8592     0.7483   5.157
##                                         Pr(>|t|)
## (Intercept)                              0.00973 **
## poly(no_prev_accidents, 2, raw = TRUE)1  0.01198 *
## poly(no_prev_accidents, 2, raw = TRUE)2  0.01216 *
## ins_loss                                5.18e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 124.9 on 46 degrees of freedom
## Multiple R-squared:  0.5003, Adjusted R-squared:  0.4677
## F-statistic: 15.35 on 3 and 46 DF,  p-value: 4.662e-07
```

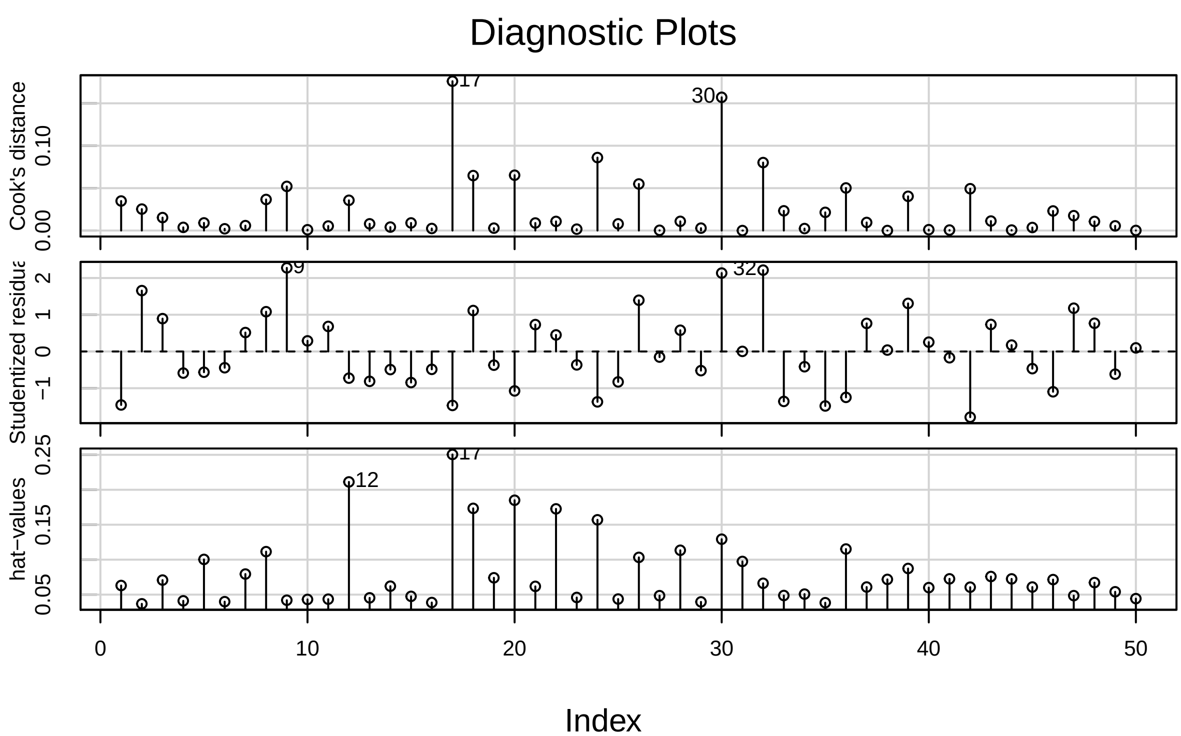
```r
bad_drivers_no_dc <- bad_drivers_no_dc %>%
  mutate(
    reside = residuals(reg02e)
  )

p1 <- ggplot(
    data = bad_drivers_no_dc,
    mapping = aes(x = no_prev_accidents, y = reside)) +
  geom_point()
p2 <- ggplot(
    data = bad_drivers_no_dc,
    mapping = aes(x = ins_loss, y = reside)) +
  geom_point()
p3 <- ggplot(
    data = bad_drivers_no_dc,
    mapping = aes(x = reside)) +
  geom_density()
p4 <- ggplot(
    data = bad_drivers_no_dc,
    mapping = aes(sample = reside)) +
  stat_qq() +
  stat_qq_line()

grid.arrange(p1, p2, p3, p4)
```




```r
car::influenceIndexPlot(reg02e,
  vars = c("Cook", "Studentized", "hat"))
```



```r
2 * 4 / nrow(bad_drivers_no_dc) # threshold for when we have to worry about leverage ("hat-values")
```

```
## [1] 0.16
```

Does anything change if we drop observations 12 and 17?

```r
reg02f <- lm(ins_premiums ~ poly(no_prev_accidents, 2, raw = TRUE) + ins_loss,
  data = bad_drivers_no_dc %>% slice(-c(12, 17)))
summary(reg02f)
```

```
##
## Call:
## lm(formula = ins_premiums ~ poly(no_prev_accidents, 2, raw = TRUE) +
##     ins_loss, data = bad_drivers_no_dc %>% slice(-c(12, 17)))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -220.42  -77.37  -18.42   93.47  274.41
##
## Coefficients:
##                                           Estimate Std. Error t value
## (Intercept)                             12914.3503  4029.9547   3.205
## poly(no_prev_accidents, 2, raw = TRUE)1  -282.3353    90.1461  -3.132
## poly(no_prev_accidents, 2, raw = TRUE)2     1.5856     0.5074   3.125
## ins_loss                                    3.4704     0.8084   4.293
##                                         Pr(>|t|)
## (Intercept)                              0.00252 **
## poly(no_prev_accidents, 2, raw = TRUE)1  0.00308 **
## poly(no_prev_accidents, 2, raw = TRUE)2  0.00314 **
## ins_loss                                9.54e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 123.8 on 44 degrees of freedom
## Multiple R-squared:  0.5114, Adjusted R-squared:  0.478
## F-statistic: 15.35 on 3 and 44 DF,  p-value: 5.65e-07
```

Nope! Parameter estimates are essentially unchanged. No need to worry about those observations.

Explaining the model to an audience

```r
summary(reg02e)
```

```
##
## Call:
## lm(formula = ins_premiums ~ poly(no_prev_accidents, 2, raw = TRUE) +
##     ins_loss, data = bad_drivers_no_dc)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -211.62  -79.65  -19.90   88.94  266.53
##
## Coefficients:
##                                          Estimate Std. Error t value
```

```
## (Intercept)                            9523.4842  3530.1025   2.698
## poly(no_prev_accidents, 2, raw = TRUE)1 -207.3916    79.2668  -2.616
## poly(no_prev_accidents, 2, raw = TRUE)2    1.1666     0.4469   2.610
## ins_loss                                   3.8592     0.7483   5.157
##                                         Pr(>|t|)
## (Intercept)                              0.00973 **
## poly(no_prev_accidents, 2, raw = TRUE)1  0.01198 *
## poly(no_prev_accidents, 2, raw = TRUE)2  0.01216 *
## ins_loss                                5.18e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 124.9 on 46 degrees of freedom
## Multiple R-squared:  0.5003, Adjusted R-squared:  0.4677
## F-statistic: 15.35 on 3 and 46 DF,  p-value: 4.662e-07
```

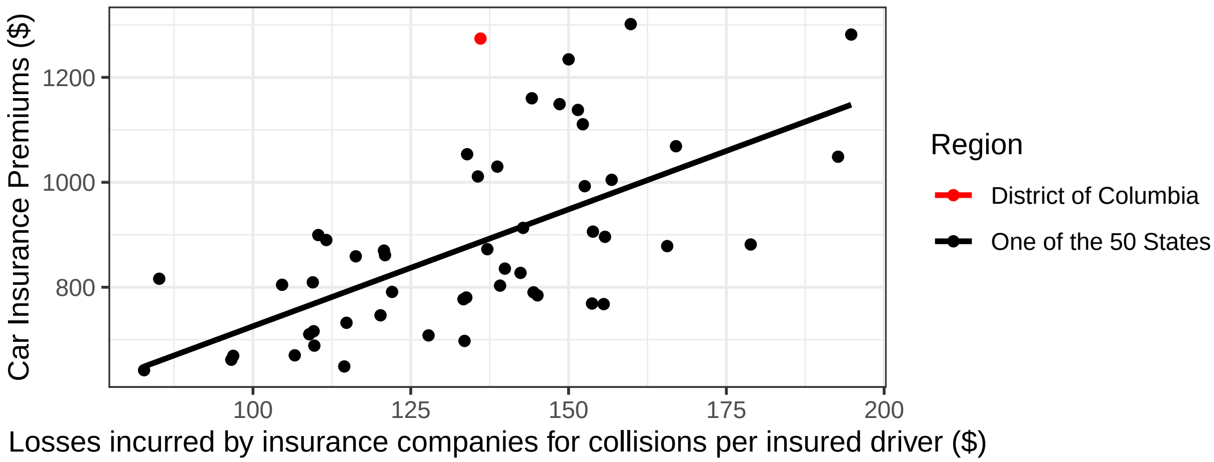
```r
confint(reg02e)
```

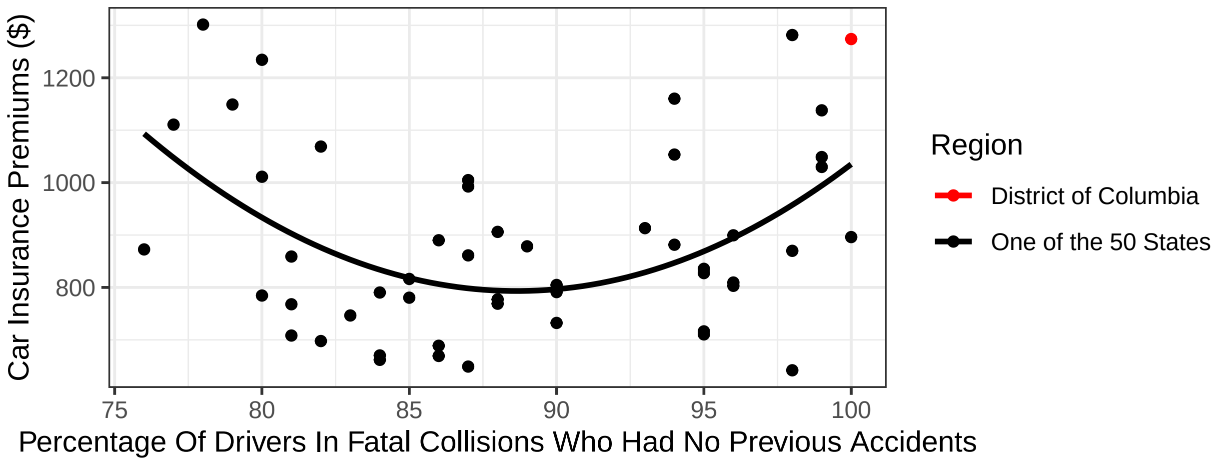
```
##                                                2.5 %       97.5 %
## (Intercept)                             2417.7564529 16629.211874
## poly(no_prev_accidents, 2, raw = TRUE)1 -366.9473736   -47.835794
## poly(no_prev_accidents, 2, raw = TRUE)2    0.2670595     2.066236
## ins_loss                                   2.3529156     5.365558
```

```r
bad_drivers <- bad_drivers %>% mutate(
  state_dc = ifelse(
    state == "District of Columbia",
    "District of Columbia",
    "One of the 50 States"
  )
)
p1 <- ggplot(data = bad_drivers,
    mapping = aes(x = ins_loss, y = ins_premiums, color = state_dc)) +
  geom_point() +
  geom_smooth(method= "lm", formula = y ~ x, se = FALSE) +
  scale_color_manual("Region", values = c("red", "black")) +
  xlab("Losses incurred by insurance companies for collisions per insured driver ($)") +
  ylab("Car Insurance Premiums ($)") +
  theme_bw()

p2 <- ggplot(data = bad_drivers, mapping = aes(x = no_prev_accidents, y = ins_premiums, color = state_d
  geom_point() +
  geom_smooth(method= "lm", formula = y ~ poly(x, 2), se = FALSE) +
  scale_color_manual("Region", values = c("red", "black")) +
  xlab("Percentage Of Drivers In Fatal Collisions Who Had No Previous Accidents") +
  ylab("Car Insurance Premiums ($)") +
  theme_bw()

grid.arrange(p1, p2)
```





The data provide strong evidence of an increasing relationship between insurance companies' losses in a given state and the premiums they charge, and a U-shaped relationship between premiums and the percentage of drivers involved in fatal collisions who had not been involved in any previous accidents. These relationships are displayed in the figure above. We have highlighted the District of Columbia in this figure because it showed unusually high premiums that did not fit the trends for the 50 states; for this reason, we excluded DC when fitting our models.

We estimate that, holding fixed the percentage of drivers in a state who had no previous accidents, an increase of one dollar in losses incurred by insurance companies for collisions per insured driver is associated with an increase of between about \$2.35 and \$5.37 in insurance premiums.

The cross-validated mean squared error of predictions was 19194.81 squared dollars for a regression model that included only insurance losses, and 16222.05 squared dollars for the model that included a quadratic term in the percent of drivers in fatal collisions who had no previous accidents. Including this variable led to a substantial improvement in predictions of insurance premiums for states that were not used to estimate the model parameters. The square root of the mean squared error for our selected model was about \$127.37. Roughly, this means that our out-of-sample predictions of insurance premiums were off by an average of about \$127.