Lab02

Alice Ji and Victoria Wang

9/20/2019

```r
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
library(GGally)
```

```
##
## Attaching package: 'GGally'
## The following object is masked from 'package:dplyr':
##
##     nasa
```

```r
library(caret)
```

```
## Loading required package: lattice
```

```r
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
## The following object is masked from 'package:dplyr':
##
##     combine
```

Exploratory Analysis

```r
# Read in and look at dataset
bad_dr <- 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_integer(),
##   `Percentage Of Drivers Involved In Fatal Collisions Who Were Alcohol-Impaired` = col_integer(),
##   `Percentage Of Drivers Involved In Fatal Collisions Who Were Not Distracted` = col_integer(),
##   `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()
## )
```

```r
names(bad_dr) <- c('state', 'num_dr_fatal_pbm', 'speeding', 'alc', 'undistr', 'no_prev_acc', 'insurance
head(bad_dr)
```

```
## # A tibble: 6 x 8
##   state num_dr_fatal_pbm speeding   alc undistr no_prev_acc insurance
##   <chr>            <dbl>    <int> <int>   <int>       <int>     <dbl>
## 1 Alab~             18.8       39    30      96          80      785.
## 2 Alas~             18.1       41    25      90          94     1053.
## 3 Ariz~             18.6       35    28      84          96      899.
## 4 Arka~             22.4       18    26      94          95      827.
## 5 Cali~             12         35    28      91          89      878.
## 6 Colo~             13.6       37    28      79          95      836.
## # ... with 1 more variable: loss_pdrv <dbl>
```

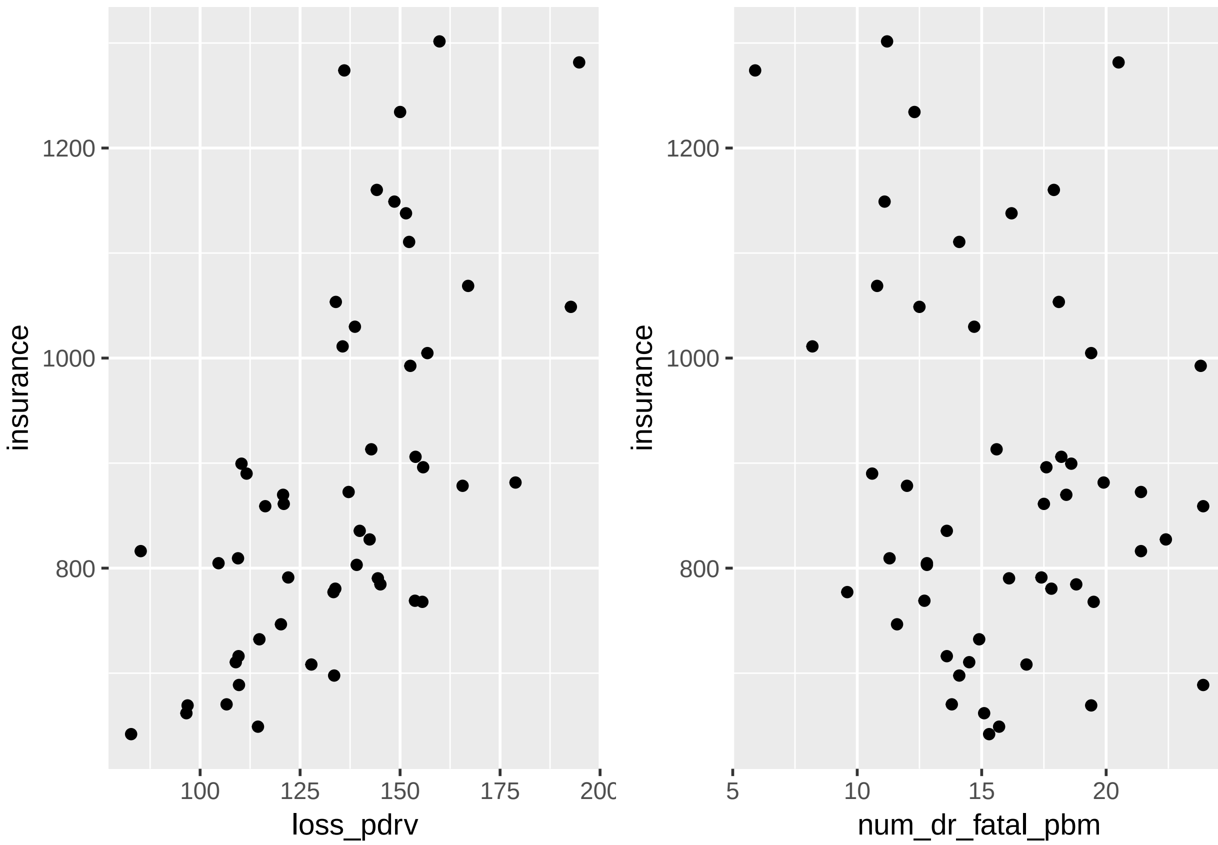
```r
# Check correlation between variables
cor(bad_dr[,-1])
```

```
##                  num_dr_fatal_pbm    speeding         alc      undistr
## num_dr_fatal_pbm      1.000000000 -0.02908015  0.19942634  0.009781764
## speeding             -0.029080146  1.00000000  0.28624417  0.131737796
## alc                   0.199426344  0.28624417  1.00000000  0.043379788
## undistr               0.009781764  0.13173780  0.04337979  1.000000000
## no_prev_acc          -0.017941877  0.01406622 -0.24545506 -0.195264522
## insurance            -0.199701946  0.04254126 -0.01745071  0.019578112
## loss_pdrv            -0.036011082 -0.06124052 -0.08391593 -0.058466772
##                  no_prev_acc   insurance   loss_pdrv
## num_dr_fatal_pbm -0.01794188 -0.19970195 -0.03601108
## speeding          0.01406622  0.04254126 -0.06124052
## alc              -0.24545506 -0.01745071 -0.08391593
## undistr          -0.19526452  0.01957811 -0.05846677
## no_prev_acc       1.00000000  0.07553314  0.04277014
## insurance         0.07553314  1.00000000  0.62311644
## loss_pdrv         0.04277014  0.62311644  1.00000000
```

Apparently, $loss_pdrv$ has the most significant correlation with our response variable $insurance$. Another slightly significantly correlated variable is $num_dr_fatal_pbm$. We will scatterplot these variables.

```r
# Check some plots
plotList <- list()
p1 <- ggplot() +
  geom_point(data = bad_dr, mapping = aes(x = loss_pdrv, y = insurance))#Plot loss_pdrv~insurance
p2 <- ggplot() +
  geom_point(data = bad_dr, mapping = aes(x = num_dr_fatal_pbm, y = insurance)) # Plot num_dr_fatal_pbm

grid.arrange(p1, p2, nrow = 1)
```



The first scatterplot (loss_pdrv~insurance) shows a positive relationship between the two variables loss_pdrv and insurance, but it is unclear if the relationship is linear.

The second scatterplot (num_dr_fatal_pbm$_{\text{insurance}}$insurance) does not show a clear relationship between the two variables.

Model Construction

Next, we fit a simple linear regression to predict insurance from percentage of drivers involved in fatal collisions who had not been involved in any previous accidents, which is defined as "loss_pdrv".

```
# Simple linear regression
reg01 <- lm(log(insurance) ~ loss_pdrv, data = bad_dr)
summary(reg01)
```

```
##
## Call:
## lm(formula = log(insurance) ~ loss_pdrv, data = bad_dr)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.2304 -0.1048 -0.0469  0.1113  0.3730
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.0993354  0.1163642  52.416  < 2e-16 ***
## loss_pdrv   0.0049799  0.0008511   5.851 3.96e-07 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1495 on 49 degrees of freedom
## Multiple R-squared:  0.4113, Adjusted R-squared:  0.3993
## F-statistic: 34.24 on 1 and 49 DF,  p-value: 3.965e-07
```

The p-value of our simple linear model is 3.96e-07 which is significantly smaller than 0.05, indicating that there is enough evidence to predict insurance from "loss_pdrv".

We then fit a multiple linear regression to predict insurance from percentage of drivers involved in fatal collisions, noted as "loss_pdrv" and the number of driver fatal per billion miles, noted as "num_dr_fatal_pdm". When we fit the multiple regression, we need to make sure that there is no collinearity of our chosen explanatory variables. We therefore look at the correlations coefficients between "loss_pdrv" and other possible factors, and conclude that the collinearity effect of "num_dr_fatal_pdm" is significantly small, with a coefficient of -0.03.

```r
# Multiple linear regression
reg02 <- lm(log(insurance) ~ loss_pdrv+num_dr_fatal_pbm, data = bad_dr)
summary(reg02)
```

```
##
## Call:
## lm(formula = log(insurance) ~ loss_pdrv + num_dr_fatal_pbm, data = bad_dr)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.24166 -0.11955 -0.03797  0.11230  0.30044
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       6.2211646  0.1426099  43.624  < 2e-16 ***
## loss_pdrv         0.0049360  0.0008423   5.860  4.1e-07 ***
## num_dr_fatal_pbm -0.0073418  0.0050751  -1.447    0.155
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1478 on 48 degrees of freedom
## Multiple R-squared:  0.4359, Adjusted R-squared:  0.4124
## F-statistic: 18.55 on 2 and 48 DF,  p-value: 1.077e-06
```

Our fitted model has a p-value of 1.077e-06, which is significantly smaller than 0.05. Although p-value of "num_dr_fatal_pbm" is larger than 0.05, we still want to conclude "num_dr_fatal_pbm" in our model because the R-Square is larger than simple linear model.

Cross-Validation

```r
# Set seed
set.seed(128)

# Slice dataset into train&val, test
train_val <- caret::createDataPartition(
  y = bad_dr$insurance,
  p = 0.8
)
drv_train_val <- bad_dr %>% slice(train_val[[1]])
```

```r
drv_test <- bad_dr %>% slice(-train_val[[1]])

# Slice cross-validation
folds <- 5
crossval_fold_inds <- caret::createFolds(
  y = drv_train_val$insurance,
  k = folds
)
```

```r
df_mse <- data.frame()
val_fold_num <- names(crossval_fold_inds)
val_fold_num
```

```
## [1] "Fold1" "Fold2" "Fold3" "Fold4" "Fold5"
```

```r
models <- paste0("reg0", 1:2)
i = 1
for(mod in lapply(models, function(x)get(x))){
  for (num in val_fold_num){
    temp <- data.frame()
    # Get train val split
    val <- drv_train_val %>% slice(crossval_fold_inds[[num]])
    train <- drv_train_val %>% slice(-crossval_fold_inds[[num]])

    #
    train_resid <- log(train$insurance) - predict(mod)
    train_mse <- mean(train_resid^2)

    #
    val_resid <- log(val$insurance) - predict(mod)
    val_mse <- mean(val_resid^2)

    temp <- data.frame('model_num' = i, num, train_mse, val_mse)
    df_mse <- rbind(df_mse, temp)
  }
  i <- i + 1
}
```

```
## Warning in log(train$insurance) - predict(mod): longer object length is not
## a multiple of shorter object length

## Warning in log(val$insurance) - predict(mod): longer object length is not a
## multiple of shorter object length

## Warning in log(train$insurance) - predict(mod): longer object length is not
## a multiple of shorter object length

## Warning in log(val$insurance) - predict(mod): longer object length is not a
## multiple of shorter object length

## Warning in log(train$insurance) - predict(mod): longer object length is not
## a multiple of shorter object length

## Warning in log(val$insurance) - predict(mod): longer object length is not a
## multiple of shorter object length

## Warning in log(train$insurance) - predict(mod): longer object length is not
## a multiple of shorter object length
```

```
## Warning in log(val$insurance) - predict(mod): longer object length is not a
## multiple of shorter object length

## Warning in log(train$insurance) - predict(mod): longer object length is not
## a multiple of shorter object length

## Warning in log(val$insurance) - predict(mod): longer object length is not a
## multiple of shorter object length

## Warning in log(train$insurance) - predict(mod): longer object length is not
## a multiple of shorter object length

## Warning in log(val$insurance) - predict(mod): longer object length is not a
## multiple of shorter object length

## Warning in log(train$insurance) - predict(mod): longer object length is not
## a multiple of shorter object length

## Warning in log(val$insurance) - predict(mod): longer object length is not a
## multiple of shorter object length

## Warning in log(train$insurance) - predict(mod): longer object length is not
## a multiple of shorter object length

## Warning in log(val$insurance) - predict(mod): longer object length is not a
## multiple of shorter object length

## Warning in log(train$insurance) - predict(mod): longer object length is not
## a multiple of shorter object length

## Warning in log(val$insurance) - predict(mod): longer object length is not a
## multiple of shorter object length

## Warning in log(train$insurance) - predict(mod): longer object length is not
## a multiple of shorter object length

## Warning in log(val$insurance) - predict(mod): longer object length is not a
## multiple of shorter object length
```

```r
df_mse
```

```
##    model_num   num  train_mse    val_mse
## 1          1 Fold1 0.05258720 0.04992034
## 2          1 Fold2 0.05904309 0.04676713
## 3          1 Fold3 0.04817534 0.07361167
## 4          1 Fold4 0.06125182 0.04236423
## 5          1 Fold5 0.06079087 0.06482337
## 6          2 Fold1 0.05432996 0.05571951
## 7          2 Fold2 0.06108290 0.05157930
## 8          2 Fold3 0.04959034 0.07512038
## 9          2 Fold4 0.06170016 0.04244768
## 10         2 Fold5 0.06289834 0.06483142
```

```r
# Train set average MSE across 5 folds
train_mse <- df_mse %>% group_by(model_num) %>% summarize(train_mse = mean(train_mse))
train_mse
```

```
## # A tibble: 2 x 2
##   model_num train_mse
##       <dbl>     <dbl>
## 1         1    0.0564
## 2         2    0.0579
```

```r
# Validation set average MSE across 5 folds
val_mse <- df_mse %>% group_by(model_num) %>% summarize(val_mse = mean(val_mse))
val_mse
```

```
## # A tibble: 2 x 2
##   model_num val_mse
##       <dbl>   <dbl>
## 1         1  0.0555
## 2         2  0.0579
```

Summary & Analysis

The mean value of train_mse across five validation sets of our simple linear regression is 0.0564; the mean value of val_mse across five validation sets of our simple linear regression is 0.0555. The mean value of train_mse across five validation sets of multiple linear regression is 0.0579; the mean value of val_mse across five validation sets of our simple linear regression is 0.579. Even the MSE of our simple linear regression is relatively smaller, we don't think we can differentiate these two models from MSE, as the MSE of both models are approximately the same. And the multiple regression model has a larger R-Square with a number of 0.41, compared to the R-Square of our first model with a number of 0.39.