MLStats: LinearRegression

Darshan Patel

1/18/2019

In this assignment, mimic the lab exercises from ISLR Chapter 3: Linear Regression.

Libraries

For this assignment, load `MASS` for access to many datasets and the `tidyverse` package which will help with exploring data and plotting. Load `car` for some functions.

```
library(MASS)
library(tidyverse)
```

```
## Warning: package 'tidyr' was built under R version 3.4.4
## Warning: package 'purrr' was built under R version 3.4.4
## Warning: package 'dplyr' was built under R version 3.4.4
```

```
library(car)
```

```
## Warning: package 'car' was built under R version 3.4.4
## Warning: package 'carData' was built under R version 3.4.4
```

Simple Linear Regression

The dataset that will be used in this assignment is the `road` dataset. In it there is information about road accident deaths in 26 US states, such as number of deaths, length of rural roads and fuel consumption per year.

The features in this dataset are

```
colnames(road)
```

```
## [1] "deaths"  "drivers" "popden"  "rural"   "temp"    "fuel"
```

Excluding the target response, `deaths`, there are 5 predictor variables. Each of these are numerical values. In the later section on qualitative predictors, an additional feature will be placed that gives information on the state's population.

Here is a look of the first 6 rows.

```
head(road)
```

```
##         deaths drivers popden rural temp  fuel
## Alabama    968     158   64.0  66.0   62 119.0
## Alaska      43      11    0.4   5.9   30   6.2
## Arizona    588      91   12.0  33.0   64  65.0
## Arkanas    640      92   34.0  73.0   51  74.0
## Calif     4743     952  100.0 118.0   65 105.0
## Colo       566     109   17.0  73.0   42  78.0
```

For the first simple linear model, try to predict `deaths` using only the number of drivers.

```r
model1 = lm(data = road, deaths~drivers)
summary(model1)
```

```
##
## Call:
## lm(formula = deaths ~ drivers, data = road)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -564.49 -138.88   34.69  120.52  862.52
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 122.0989    78.7076   1.551    0.134
## drivers       4.5951     0.2897  15.863 3.19e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 285.2 on 24 degrees of freedom
## Multiple R-squared:  0.9129, Adjusted R-squared:  0.9093
## F-statistic: 251.6 on 1 and 24 DF,  p-value: 3.192e-14
```

According to this, $\hat{\beta}_0 = 122.0989$ and $\hat{\beta}_1 = 4.5951$. This means that for a unit increase of $10,000$ drivers, the number of deaths by road accident increases by 4.4951. In addition, the p-value associated with `drivers` is close to 0 and so we can reject the null hypothesis that $H_0 : \beta_{\text{driver}} = 0$. This means `drivers` is statistically significant. This linear model has a RSE of 285.2 and R^2 value of 0.9129. This is a strong positive R^2 value. The model fits well. The confidence interval for the coefficient estimates are shown below

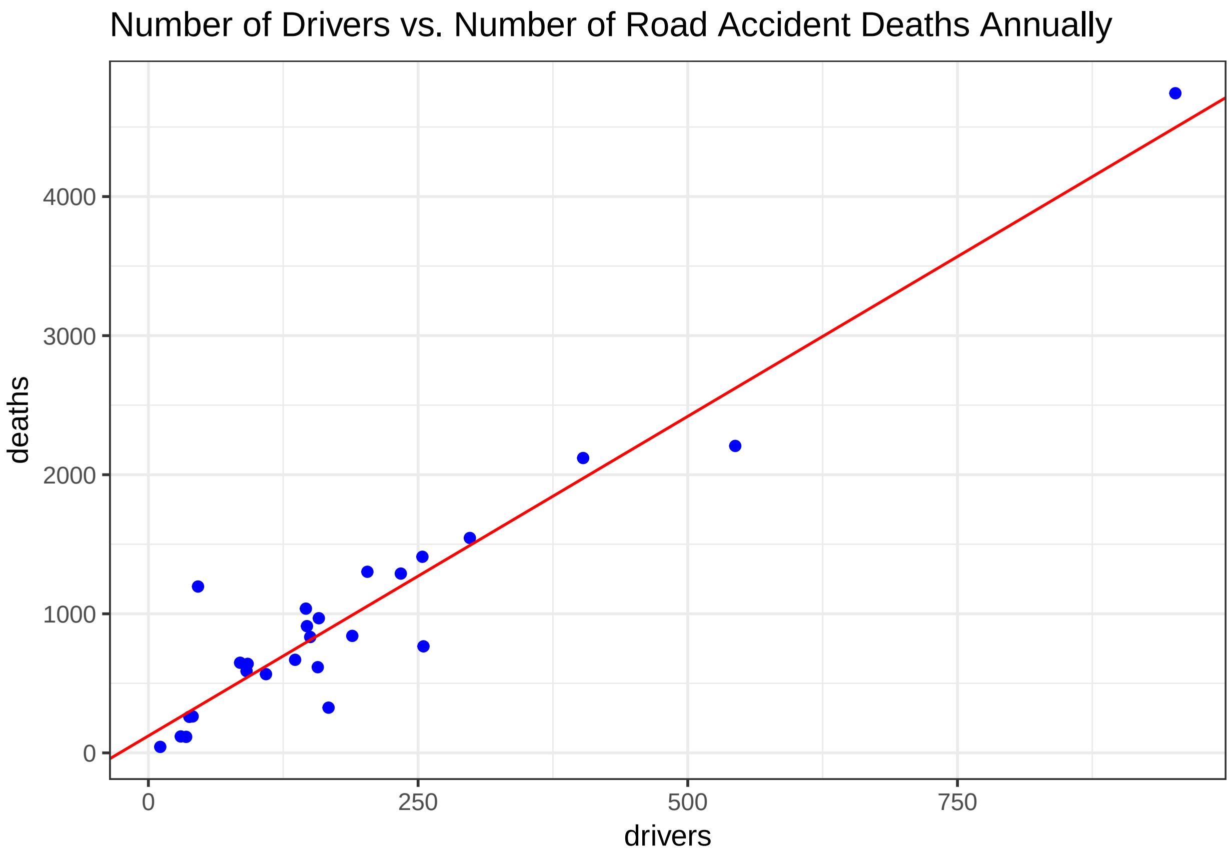
```r
confint(model1)
```

```
##                  2.5 %     97.5 %
## (Intercept) -40.345529 284.543323
## drivers       3.997271   5.193004
```

The 95% confidence interval associated with `drivers` is $(3.997271, 5.193004)$.

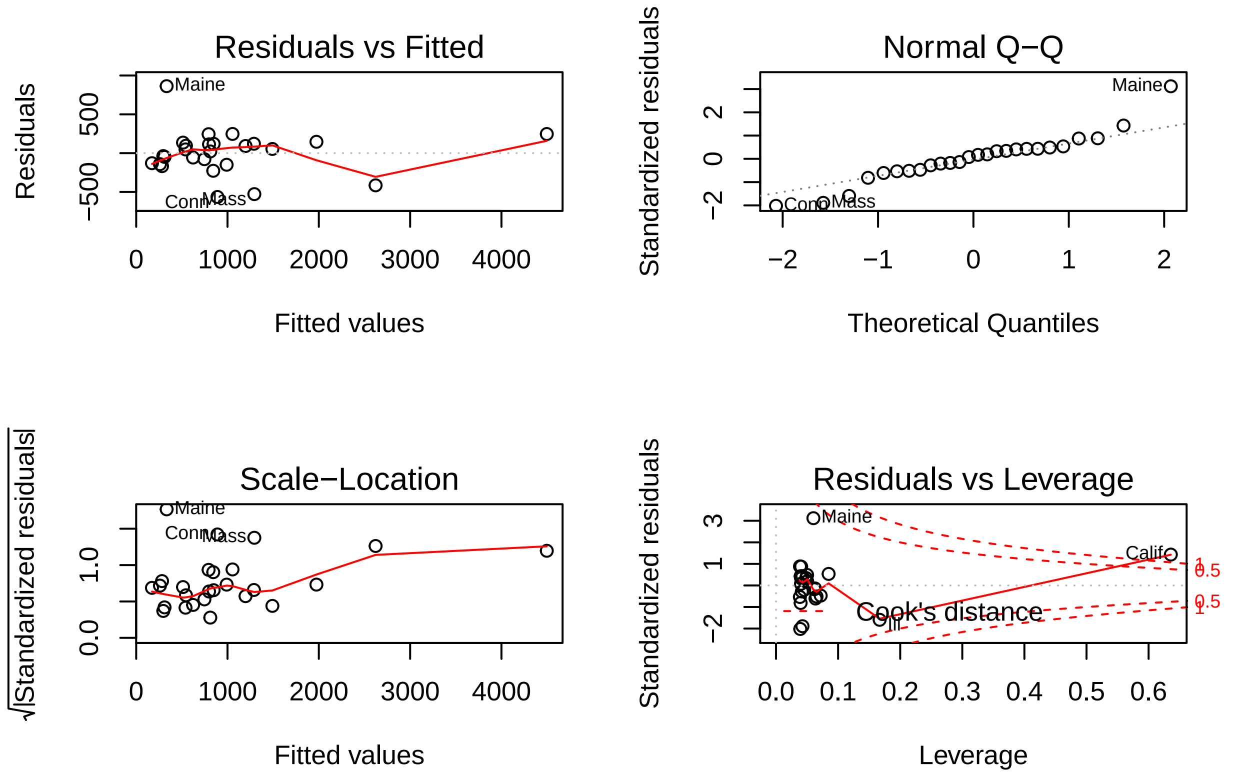
The regression line showed a nice positive R^2 value, meaning the regression line explained most of the variance. Let's see how the data looks with the regression line.

```r
ggplot(data = road, aes(x = drivers, y = deaths)) + geom_point(col = "blue") +
  geom_abline(intercept = model1$coefficients[1],
              slope = model1$coefficients[2],
              color = "red") +
  ggtitle("Number of Drivers vs. Number of Road Accident Deaths Annually") +
  theme_bw()
```



The regression line is a good fit for the data. Now let's look at some diagnostic plots.

```
par(mfrow = c(2,2))
plot(model1)
```



According to these plots, there are some conclusions that can be drawn. The residuals vs. fitted plot shows that three states showed huge residuals in the model, Maine, Massachusetts and Connecticut. In the residuals vs. leverage plot, it clear that California is a leverage point. This can also be found using the `which.max` and `hatvalues` function.

```r
which.max(hatvalues(model1))
```

```
## Calif
##     5
```

Multiple Linear Regression

Using the same dataset, use two features to predict `deaths`. Let these two features be number of drivers, `drivers` and length of rural roads, `rural`.

```r
model2 = lm(data = road, deaths~drivers+rural)
summary(model2)
```

```
##
## Call:
## lm(formula = deaths ~ drivers + rural, data = road)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -464.82  -94.21   -6.32  101.80  912.55
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   45.369    106.165   0.427    0.673
## drivers        4.401      0.341  12.903 5.13e-12 ***
```

```
## rural          1.877      1.750   1.073    0.294
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 284.3 on 23 degrees of freedom
## Multiple R-squared:  0.9171, Adjusted R-squared:  0.9099
## F-statistic: 127.2 on 2 and 23 DF,  p-value: 3.671e-13
```

This model slightly improves on the above model. The residual standard error goes slightly down while R^2 goes slightly up. However, while `drivers` is statistically significant, `rural` is not, since the associated p-value with the coefficient estimate is not smaller than $\alpha = 0.01$.

Now let's try using all features to predict `deaths`.

```r
model3 = lm(data = road, deaths~.)
summary(model3)
```

```
##
## Call:
## lm(formula = deaths ~ ., data = road)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -405.23 -123.97  -28.06   72.61  950.30
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -155.94105  238.79327  -0.653    0.521
## drivers        4.44399    0.39618  11.217 4.44e-10 ***
## popden        -0.01318    0.02458  -0.536    0.598
## rural          2.55112    1.89771   1.344    0.194
## temp           6.12376    4.55712   1.344    0.194
## fuel          -0.93411    0.87527  -1.067    0.299
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 280 on 20 degrees of freedom
## Multiple R-squared:  0.9301, Adjusted R-squared:  0.9126
## F-statistic: 53.19 on 5 and 20 DF,  p-value: 7.196e-11
```

By using all 5 predictors, the model was able to explain 93% of the variance of the `death` variable around its mean, a new high score. This model did better than the univarate and bivariate models. What is also interesting to see here is that `drivers` remains to be the only predictor that is statistically significant which others are insignificant. The variation inflation factors are calculated below.

```r
vif(model3)
```

```
##  drivers   popden    rural     temp     fuel
## 1.940666 1.145084 1.692047 1.121798 1.718427
```

Since these VIF values are low (less than 20), it can be said that the predictor variables do not show presence of multicollinearity.

Now, in the above model, it is clear that see that `popden` has a high p-value, a staggering 0.598! Let's make a model without this feature.

```r
model4 = lm(data = road, deaths~.-popden)
summary(model4)
```

```
##
## Call:
## lm(formula = deaths ~ . - popden, data = road)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -386.82 -123.11  -28.38   68.44  971.57
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -186.2586   228.0298  -0.817    0.423
## drivers        4.4283     0.3883  11.403 1.85e-10 ***
## rural          2.8487     1.7837   1.597    0.125
## temp           6.2075     4.4765   1.387    0.180
## fuel          -0.9017     0.8582  -1.051    0.305
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 275.2 on 21 degrees of freedom
## Multiple R-squared:  0.9291, Adjusted R-squared:  0.9155
## F-statistic: 68.75 on 4 and 21 DF,  p-value: 9.249e-12
```

By removing `popden`, the RSE went down. This shows that it is important to use p-values to remove extraneous features that hinder model performance.

Interaction Terms

Try creating an interaction between `rural` and `temp` to see if it can predict `deaths` well.

```r
model5 = lm(data = road, deaths~rural*temp)
summary(model5)
```

```
##
## Call:
## lm(formula = deaths ~ rural * temp, data = road)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1311.03  -489.03    24.74   336.59  1221.07
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1715.6672   965.9815   1.776  0.08955 .
## rural        -23.8896    12.6818  -1.884  0.07288 .
## temp         -40.3167    22.6555  -1.780  0.08897 .
## rural:temp     0.9447     0.3046   3.101  0.00521 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 643.5 on 22 degrees of freedom
## Multiple R-squared:  0.5935, Adjusted R-squared:  0.5381
## F-statistic: 10.71 on 3 and 22 DF,  p-value: 0.0001534
```

This model does poorly, with a RSE of 643.5. Too high! Try another interaction.

```r
model6 = lm(data = road, deaths~drivers*fuel)
summary(model6)
```

```
##
## Call:
## lm(formula = deaths ~ drivers * fuel, data = road)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -595.25 -123.80   14.56   93.65  864.94
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   90.897492 141.158008   0.644    0.526
## drivers        5.092261   0.536700   9.488 3.11e-09 ***
## fuel           0.304986   1.560835   0.195    0.847
## drivers:fuel  -0.003153   0.003974  -0.793    0.436
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 289 on 22 degrees of freedom
## Multiple R-squared:  0.918,  Adjusted R-squared:  0.9068
## F-statistic: 82.11 on 3 and 22 DF,  p-value: 4.184e-12
```

This interaction did better than the previous interaction but not as well as the 4 predictor model, in terms of the RSE metric.

```r
model7 = lm(data = road, deaths~drivers*popden)
summary(model7)
```

```
##
## Call:
## lm(formula = deaths ~ drivers * popden, data = road)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -262.50 -135.07  -23.13   85.00  819.10
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    133.027983  65.756091   2.023  0.05539 .
## drivers          5.359694   0.316745  16.921 4.25e-14 ***
## popden           0.204256   0.062167   3.286  0.00338 **
## drivers:popden  -0.006316   0.001674  -3.773  0.00105 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 228.9 on 22 degrees of freedom
## Multiple R-squared:  0.9486, Adjusted R-squared:  0.9416
## F-statistic: 135.2 on 3 and 22 DF,  p-value: 2.516e-14
```

This interaction made even less residual standard errors. Now 94.86% of the variance in the response variable is explained by the regression line.

I think here is a good place to stop looking for improvements by interactions. Otherwise we will be overfitting the data..

Non-linear Transformations of the Predictors

Let's try non-linear transformations.

```
model8 = lm(data = road, deaths~log(drivers))
summary(model8)
```

```
##
## Call:
## lm(formula = deaths ~ log(drivers), data = road)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -883.4 -272.6 -100.7  116.3 2245.1
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2583.2      622.0  -4.153 0.000358 ***
## log(drivers)    740.8      126.1   5.874 4.64e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 618.9 on 24 degrees of freedom
## Multiple R-squared:  0.5898, Adjusted R-squared:  0.5727
## F-statistic: 34.51 on 1 and 24 DF,  p-value: 4.643e-06
```

By using the log value of `drivers`, the model performed worse than using the feature regularly. The RSE value is very high and a bit over a half of the variance in `deaths` is explained. Try another non-linear transformation.

```
model9 = lm(data = road, deaths~poly(rural, 2) + exp(rural) + gamma(rural))
```

```
## Warning in gamma(rural): NaNs produced
```

```
summary(model9)
```

```
##
## Call:
## lm(formula = deaths ~ poly(rural, 2) + exp(rural) + gamma(rural),
##     data = road)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -761.78 -338.86  -31.81  209.15  864.09
##
## Coefficients:
##                   Estimate  Std. Error t value Pr(>|t|)
## (Intercept)      9.205e+02   1.063e+02   8.661 3.35e-08 ***
## poly(rural, 2)1  1.940e+03   5.967e+02   3.251    0.004 **
## poly(rural, 2)2  7.196e+01   6.708e+02   0.107    0.916
## exp(rural)       1.824e-48   3.412e-49   5.347 3.11e-05 ***
## gamma(rural)    -1.070e-199  0.000e+00    -Inf  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 482.3 on 20 degrees of freedom
##   (1 observation deleted due to missingness)
```

```
## Multiple R-squared:  0.7846, Adjusted R-squared:  0.7415
## F-statistic: 18.21 on 4 and 20 DF,  p-value: 1.902e-06
```

This model uses some interesting transformations, such as polynomials, the gamma function and the normal exponential function. The RSE improved from the previous transformation by a third. What is also visible here is that some of these transformed variables coefficients are statistically significant since the p-values associated with them are close to 0, such as the exponential of `rural` and gamma of `rural`. In addition, only the coefficient of rural raised to the first power is statistically significant while when it is raised to the second power, it loses significance.

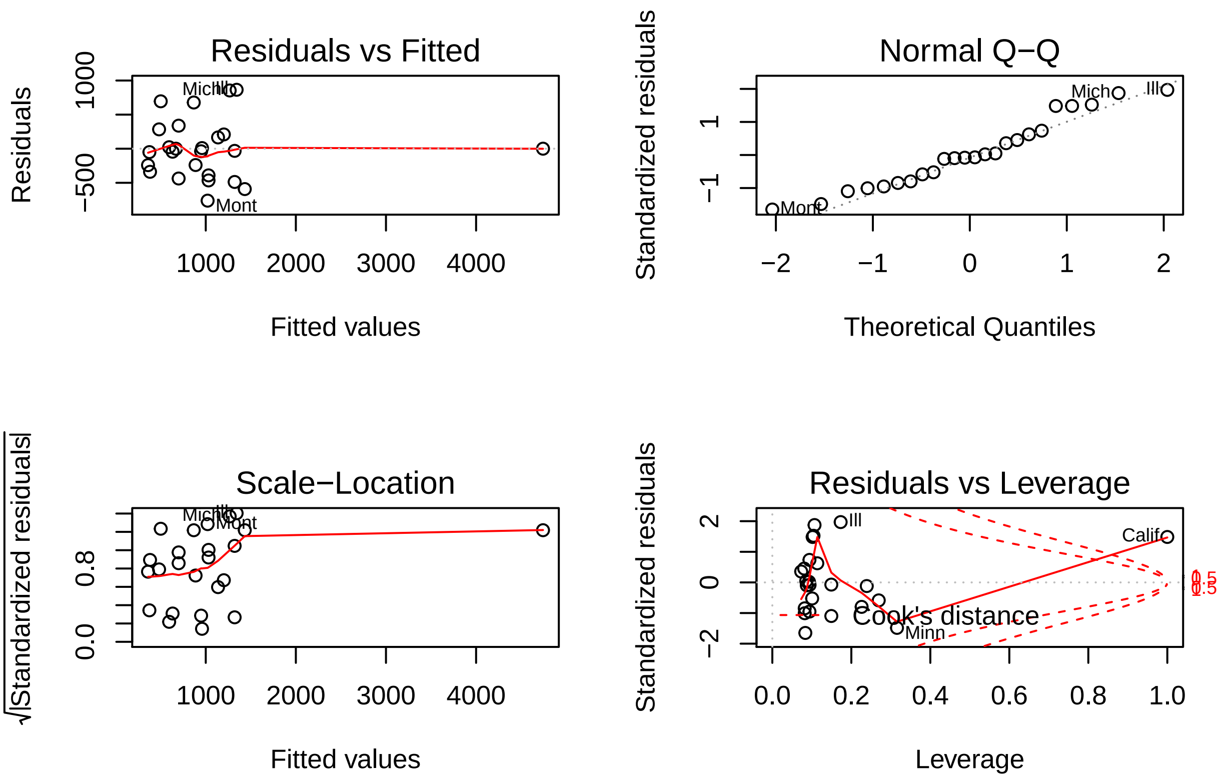
```r
par(mfrow = c(2,2))
plot(model9)
```

```
## Warning: not plotting observations with leverage one:
##   15

## Warning: not plotting observations with leverage one:
##   15

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```



These plots show that California remains to be a leverage point and that the residuals are more spread out than above.

Qualitative Predictors

This dataset does not have any qualitative predictor, so let's add one. The following `R` commands will add in a column that bins the state's population into either "low", "average", "high" or "very high" population.

```r
pop_type = cut(road$popden, breaks = c(quantile(road$popden, probs = seq(0,1, by = 0.25))),
               labels=c("Low", "Med", "High", "Very High"))
pop_type[2] = "Low"
df = cbind(road, pop_type)
```

The coding that `R` uses for this new predictor is as follows

```r
contrasts(df$pop_type)
```

```
##           Med High Very High
## Low         0    0         0
## Med         1    0         0
## High        0    1         0
## Very High   0    0         1
```

First make a model using all predictors except `pop_type` and then with `pop_type`.

```r
model10 = lm(data = df, deaths~.-pop_type)
model11 = lm(data = df, deaths~.)

summary(model10)$sigma
```

```
## [1] 279.9529
```

```r
summary(model11)$sigma
```

```
## [1] 262.2981
```

By adding in a categorical predictor, `pop_type`, to show population level, the residual standard error of the model went down.

```r
summary(model11)
```

```
##
## Call:
## lm(formula = deaths ~ ., data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -290.53 -131.97  -15.19   61.56  745.74
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        238.24383  280.02745   0.851   0.4067
## drivers              4.74193    0.43858  10.812 4.87e-09 ***
## popden               0.01030    0.02569   0.401   0.6934
## rural               -1.37402    2.75180  -0.499   0.6240
## temp                 0.23617    5.20792   0.045   0.9644
## fuel                 0.34017    1.07780   0.316   0.7561
## pop_typeMed        -19.69099  162.75847  -0.121   0.9051
## pop_typeHigh        44.72944  196.74264   0.227   0.8229
## pop_typeVery High -454.00178  238.93546  -1.900   0.0745 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 262.3 on 17 degrees of freedom
## Multiple R-squared:  0.9478, Adjusted R-squared:  0.9233
## F-statistic:  38.6 on 8 and 17 DF,  p-value: 2.081e-09
```

To intercept the `pop_type` coefficient, read it as follows: if the `pop_type` of the state is `low`, then the number of deaths is 238.24383 plus the values from the other coefficients. If the `pop_type` of the state is `med`, then the number of deaths is $238.24383 + -19.69099 = 218.5528$, where the coefficient for each factor of `pop_type` is added to the base level of `low` `pop_type`, plus the values from the other coefficients, and so on.

Writing Functions

```
# This function will return the polynomial degree that results in the lowest RSE
# in the linear regression model given one predictor, one target variable and
# a maximum polynomial degree.
lm_poly_max = function(x, y, max_deg = 10){

  df = data.frame(x = x, y = y)
  rse = c()
  for(i in 1:max_deg){
    model = lm(data = df, y~poly(x,i))
    rse = c(rse, summary(model)$sigma)
  }
  return(which.min(rse))
}
```

For example, the degree of `drivers` where RSE is lowest when predicting `deaths` using `drivers` is

```
lm_poly_max(road$drivers, road$deaths)
```

```
## [1] 5
```

and the degree of `fuel` when predicting `deaths` using `fuel` is

```
lm_poly_max(road$fuel, road$deaths)
```

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

All of the lab instructions in this document are taken from "An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani.