COMPSCIX 415.2 Homework 7

Bryan Hee

March 19, 2018

Exercise 1

There are 81 columns or variables and 1,460 observations in the train data set from the kaggle competition, House Prices: Advanced Regression Techniques.

```
train <- read_csv(file = "C:/Users/BryanHee/OneDrive - stok LLC/Intro to Data Science/HW Assignments/As
glimpse(train)
```

Exercise 2

The following code represents a random sample of the train dataset via a 70/30 split. 70% of the train dataset will be used as a training dataset. The remaining 30% will be used as the testing dataset.

```
set.seed(29283)

train_set <- train %>% sample_frac(0.7)
test_set <- train %>% filter(!(train$Id %in% train_set$Id))
```

Exercise 3

The following code provides a linear regression with only a y-intercept for the variable train_set$SalePrice. The mean Sale Price is $181,176. This is confirmed by the function broom::tidy. The R-squared value is derived from the glance() function which is 0.

```
mod_0 <- lm(formula = SalePrice ~ 1, data = train_set)

mean(train_set$SalePrice)
```

```
## [1] 182176
```

```
tidy(mod_0)
```

```
##          term estimate std.error statistic p.value
## 1 (Intercept)   182176  2492.072  73.10222       0
```

```
glance(mod_0)
```

```
##   r.squared adj.r.squared    sigma statistic p.value df    logLik      AIC
## 1         0             0 79668.37        NA      NA  1 -12983.57 25971.13
##        BIC     deviance df.residual
## 1 25980.99 6.480338e+12        1021
```

Exercise 4

It makes sense that the above ground living square footage of the building (GrLivArea) is positively correlated to the house price. The coefficient provides the price per square foot valuation of the linear model, $54.50/sf. The overall quality of the building material (OverallQual) also makes sense that it is positievely correlated.

The higher the 1-10 rating, the more expensive the house (all else being equal). The coefficient is the price effect on the overall home sale price associated with increasing the material and finish rating by one.

I would interpret the Neighborhood coefficients in the following way: the sign describes whether or not the neighborhood is beneficial for the home price (i.e. whether it is a desirable neighborhood (+) or not (-), and the absolute value relates the weight factor of the desirability (i.e. the larger the positive number the more desirable the neighborhood).

Using a cutoff p-value of 0.5, all of the features are significant. However, in my opinion Neighborhood feature is not a practically significant feature due to the small sample size of the train_set dataset. There are 6 neighborhoods with less than 15 home sales within the sample set. That is not enough to have this represent a meaningful feature. See the code below for the count per each neighborhood.

The model is a relatively good fit for the training set given the high R-squared value of 0.787.

```
mod_1 <- lm(SalePrice ~ GrLivArea + OverallQual + Neighborhood, data = train_set)
tidy(mod_1)
```

```
##                    term     estimate    std.error  statistic      p.value
## 1           (Intercept) -45017.87483 12933.341808 -3.4807612 5.216927e-04
## 2             GrLivArea     62.77735     3.006033 20.8837885 1.337222e-80
## 3           OverallQual  21692.23178  1353.714104 16.0242342 1.389020e-51
## 4   NeighborhoodBlueste -38288.88063 36531.907177 -1.0480942 2.948497e-01
## 5    NeighborhoodBrDale -43314.05372 14524.693991 -2.9820975 2.932566e-03
## 6   NeighborhoodBrkSide -14064.37052 11318.850018 -1.2425618 2.143221e-01
## 7   NeighborhoodClearCr  27839.00662 13561.346871  2.0528202 4.035110e-02
## 8   NeighborhoodCollgCr   4297.67432 10372.304467  0.4143413 6.787135e-01
## 9   NeighborhoodCrawfor   7423.05573 11371.511784  0.6527765 5.140512e-01
## 10 NeighborhoodEdwards -15284.11495 10994.287187 -1.3901870 1.647830e-01
## 11 NeighborhoodGilbert  -8357.55930 10894.173472 -0.7671586 4.431692e-01
## 12  NeighborhoodIDOTRR -32689.43085 12603.712743 -2.5936350 9.636216e-03
## 13 NeighborhoodMeadowV -14446.06504 14190.148622 -1.0180348 3.089089e-01
## 14 NeighborhoodMitchel   1922.31487 11788.608170  0.1630655 8.705000e-01
## 15   NeighborhoodNAmes  -7719.67883 10375.956174 -0.7439969 4.570540e-01
## 16 NeighborhoodNoRidge  47685.16790 12567.432633  3.7943444 1.569690e-04
## 17 NeighborhoodNPkVill -20240.71145 16548.664867 -1.2231024 2.215806e-01
## 18 NeighborhoodNridgHt  63872.80848 10880.456671  5.8704161 5.917964e-09
## 19  NeighborhoodNWAmes -12279.33299 11047.502893 -1.1115030 2.666204e-01
## 20 NeighborhoodOldTown -36107.07577 10849.170903 -3.3280954 9.064637e-04
## 21  NeighborhoodSawyer  -4121.92502 11252.369778 -0.3663162 7.142070e-01
## 22 NeighborhoodSawyerW  -5391.97074 11230.758221 -0.4801075 6.312565e-01
## 23 NeighborhoodSomerst  18700.96725 10772.212794  1.7360377 8.286672e-02
## 24 NeighborhoodStoneBr  65712.45881 12745.312907  5.1558137 3.045915e-07
## 25   NeighborhoodSWISU -45451.86707 13564.792586 -3.3507233 8.363074e-04
## 26  NeighborhoodTimber  27925.08619 11985.325857  2.3299397 2.000859e-02
## 27 NeighborhoodVeenker  54913.12768 16521.075497  3.3238228 9.203087e-04
```

```
train_set %>%
  group_by(Neighborhood) %>%
summarise(count = n()) %>%
  arrange(count)
```

```
## # A tibble: 25 x 2
##    Neighborhood count
##    <chr>        <int>
##  1 Blueste          1
##  2 NPkVill          7
```

```
##   3 Veenker          7
##   4 BrDale          11
##   5 Blmngtn         13
##   6 MeadowV         13
##   7 ClearCr         15
##   8 SWISU           16
##   9 StoneBr         19
##  10 IDOTRR          22
## # ... with 15 more rows
```

```r
glance(mod_1)
```

```
##   r.squared adj.r.squared   sigma statistic p.value df    logLik      AIC
## 1 0.8099927     0.8050277 35178.1  163.1401       0 27 -12134.95 24325.91
##        BIC     deviance df.residual
## 1 24463.93 1.231311e+12         995
```

Exercise 5

```r
test_predictions <- as.tibble(predict(mod_1, newdata = test_set))
test_set1 <- mutate(test_set, PSalePrice = test_predictions$value)
test_set1 <- test_set1[c("GrLivArea", "OverallQual", "Neighborhood", "SalePrice", "PSalePrice")]

price_difference <- rep(NA, 438)
for(i in 1:438){
price_difference[i] <- ((test_set1$SalePrice[i] - test_set1$PSalePrice[i])^2)
}
price_difference <- as.tibble(price_difference)

price_difference %>%
  summarise(sqrt(mean(value))) -> rmse
rmse
```

```
## # A tibble: 1 x 1
##   `sqrt(mean(value))`
##                 <dbl>
## 1               41915
```

Exercise 6

```r
mod_1.5 <- lm(SalePrice ~ LotArea + OverallQual + YearRemodAdd, data = train_set)
tidy(mod_1.5)
```

```
##           term      estimate    std.error statistic       p.value
## 1  (Intercept) -9.390195e+05 1.557609e+05 -6.028596  2.310215e-09
## 2      LotArea  1.370255e+00 1.258617e-01 10.886989  3.458363e-26
## 3  OverallQual  4.260040e+04 1.234723e+03 34.501978 2.193788e-173
## 4 YearRemodAdd  4.264136e+02 8.040450e+01  5.303355  1.394496e-07
```

```r
glance(mod_1.5)
```

```
##   r.squared adj.r.squared   sigma statistic       p.value df    logLik
## 1 0.6860318     0.6851066 44706.2  741.4556 1.734936e-255  4 -12391.59
##        AIC      BIC    deviance df.residual
```

```
## 1 24793.18 24817.83 2.03462e+12        1018
```

```r
test_predictions_1.5 <- as.tibble(predict(mod_1.5, newdata = test_set))
test_set1.5 <- mutate(test_set, PSalePrice = test_predictions_1.5$value)
test_set1.5 <- test_set1.5[c("LotArea", "OverallQual", "YearRemodAdd", "SalePrice", "PSalePrice")]

price_difference1.5 <- rep(NA, 438)
for(i in 1:438){
price_difference1.5[i] <- ((test_set1.5$SalePrice[i] - test_set1.5$PSalePrice[i])^2)
}
price_difference1.5 <- as.tibble(price_difference1.5)

price_difference1.5 %>%
  summarise(sqrt(mean(value))) -> rmse
rmse
```

```
## # A tibble: 1 x 1
##   `sqrt(mean(value))`
##                 <dbl>
## 1               48661
```

Exercise 7

After running the following model several times, the biggest thing that stood out to me was the variability of the R-squared value. It ranged from 0.50 to 0.94. I added a value of y divided by x to get a rough “feel” for when the sampling provided outliers. It wasn’t always true, but for the most part the larger the y/x max value was, the smaller the R-squared value. This makes sense to me because if there are a lot of data points with large y/x values, you would expect the R-squared value to higher than if there was only one or two high y/x outliers.

```r
sim1a <- tibble(
  x=rep(1:10, each=3),
  y=x * 1.5 + 6 + rt(length(x), df=2),
  z=y/x
)
max(sim1a$z)
```

```
## [1] 9.772215
```

```r
mod_2 <- lm(formula = y ~ x, data = sim1a)
glance(mod_2)
```

```
##   r.squared adj.r.squared    sigma statistic      p.value df    logLik
## 1 0.7935687     0.7861961 2.208842  107.6383 4.244805e-11  2 -65.30731
##        AIC      BIC deviance df.residual
## 1 136.6146 140.8182 136.6115          28
```

```r
tidy(mod_2)
```

```
##          term estimate std.error statistic      p.value
## 1 (Intercept) 6.694060 0.8711790  7.683909 2.270460e-08
## 2           x 1.456668 0.1404032 10.374889 4.244805e-11
```