ORF 350: Assignment 2

David Fan

3/8/2017

Collaborator: Brandon Tan

Question 1: Maximum Likelihood Estimator (MLE) and Asymptotic Normality

Part I

Slutzky's theorem tells us that if two random variables $X_n \xrightarrow{D} X$ and $Y_n \xrightarrow{P} c$, then $X_n Y_n \xrightarrow{D} cX$. Let $X_n = \sqrt{n}(\hat{\theta}_n - \theta)$ and $Y_n = \sqrt{I(\hat{\theta}_n)}$. The blog post on EdX shows how the empirical Fisher info (a random quantity) $\sqrt{I(\hat{\theta}_n)}$ converges in probability to the population Fisher info (a constant) $\sqrt{I(\theta)}$. Thus, it is implied that $\hat{\theta_n}$ is a consistent estimator of θ. By Slutzky's theorem, $\sqrt{n}(\hat{\theta}_n - \theta)\sqrt{I(\hat{\theta}_n)} \xrightarrow{D} \sqrt{I(\theta)} * N(0, \frac{1}{I(\theta)})$. The right hand side is equivalent to $N(0, \frac{I(\theta)}{I(\theta)}) \sim N(0,1)$. Thus, $\sqrt{n}(\hat{\theta}_n - \theta)\sqrt{I(\hat{\theta}_n)} \xrightarrow{D} N(0,1)$. After some transformations of the random variable, we get $\hat{\theta_n} \xrightarrow{D} N(\theta, \frac{1}{nI(\hat{\theta_n})})$.

This Gaussian has mean θ and standard deviation $\sqrt{\frac{1}{nI(\hat{\theta}_n)}}$, so the asymptotic (1-$\alpha$) confidence interval for θ is $\boxed{[\hat{\theta}_n - z_{\alpha/2} * \frac{1}{\sqrt{nI(\hat{\theta}_n)}}, \hat{\theta}_n + z_{\alpha/2} * \frac{1}{\sqrt{nI(\hat{\theta}_n)}}]}$.

We can verify this: Let F_n be the CDF of $\sqrt{n}(\hat{\theta}_n - \theta)\sqrt{I(\hat{\theta}_n)}$. Since $\sqrt{n}(\hat{\theta}_n - \theta)\sqrt{I(\hat{\theta}_n)} \xrightarrow{D} N(0,1)$, $\lim_{n\to\infty} F_n(x) = \Phi(x)$ where $\Phi(x)$ is the CDF of the standard normal distribution.

$$\begin{aligned}
\lim_{n\to\infty} F_n(z_{\alpha/2}) - F_n(z_{-\alpha/2}) &= \lim_{n\to\infty} \Phi(z_{\alpha/2}) - \Phi(-z_{\alpha/2}) = 1 - \alpha \\
F_n(z_{\alpha/2}) - F_n(z_{-\alpha/2}) &= P(z_{-\alpha/2} < \sqrt{n}(\hat{\theta}_n - \theta)\sqrt{I(\hat{\theta}_n)} > z_{\alpha/2}) \\
&= P(\hat{\theta_n} - \frac{z_{\alpha/2}}{\sqrt{nI(\hat{\theta_n})}} < \theta < \hat{\theta_n} - \frac{z_{-\alpha/2}}{\sqrt{nI(\hat{\theta_n})}}) \\
&= P(\hat{\theta_n} - \frac{z_{\alpha/2}}{\sqrt{nI(\hat{\theta_n})}} < \theta < \hat{\theta_n} + \frac{z_{\alpha/2}}{\sqrt{nI(\hat{\theta_n})}})
\end{aligned}$$

$$\lim_{n\to\infty} P(\hat{\theta_n} - \frac{z_{\alpha/2}}{\sqrt{nI(\hat{\theta_n})}} < \theta < \hat{\theta_n} + \frac{z_{\alpha/2}}{\sqrt{nI(\hat{\theta_n})}}) = 1 - \alpha$$

Part II

$p_\theta(x) = (\theta - 1)x^{-\theta} * I\{x \geq 1\}$

a) Likelihood function: L(θ, x) = $\prod_{i=1}^{n} p(x_i, \theta)$ for x $\geq$ 1

Log-likelihood: l(θ, x) $= log(\prod_{i=1}^{n} p(x_i, \theta)) = \sum_{i=1}^{n} log[(\theta-1)x_i^{-\theta}] = \sum_{i=1}^{n} [log(\theta-1) - \theta log(x_i)]$

$\frac{dl(\theta,x)}{d\theta} = \sum_{i=1}^{n} \frac{1}{\theta-1} - \theta log(x_i)$

$\frac{dl(\theta,x)}{d\theta} = \frac{n}{\theta-1} - \sum_{i=1}^{n} log(x_i) = 0$

$\frac{n}{\theta-1} = \sum_{i=1}^{n} log(x_i)$

$$\boxed{\hat{\theta_n} = 1 + \frac{n}{\sum_{i=1}^{n} log(xi)}}$$

b) From (0.1) in the HW handout, we know that our MLE is asymptotically normal. $\hat{\theta_n} \xrightarrow{D} N(\theta, \frac{1}{nI(\theta)})$, so the asymptotic variance is $\frac{1}{nI(\theta)}$. Let's find the population Fisher info:

$$\begin{aligned}
I(\theta) &= E_\theta(-\frac{\partial^2}{\partial\theta^2} log(p_\theta(x))) \\
log(p_\theta(x)) &= log[(\theta-1)x^{-\theta}] \\
&= log(\theta-1) - \theta log(x) \\
\frac{\partial^2}{\partial\theta^2} log(p_\theta(x)) &= \frac{\partial}{\partial\theta}[\frac{1}{\theta-1} - log(x)] = -(\theta-1)^{-2} \\
I(\theta) &= E_\theta(\frac{1}{(\theta-1)^2}) \\
&= \int_1^\infty \frac{1}{(\theta-1)^2} * p_\theta(x)dx \\
&= \int_1^\infty \frac{x^{-\theta}}{\theta-1} dx \\
&= \frac{x^{-\theta+1}}{(\theta-1)(\theta+1)}\Big|_1^\infty \\
&= 0 - \frac{1}{(\theta-1)(-\theta+1)} \\
&= \frac{1}{(\theta-1)^2}
\end{aligned}$$

Finally, we plug the Fisher information back into the expression for asymptotic variance:

$$\boxed{\sigma^2 = \frac{1}{nI(\theta)} = \frac{(\theta-1)^2}{n}}.$$

c) We are given that $\sqrt{n}(\hat{\theta}_n - \theta) \xrightarrow{D} N(0, \frac{1}{I(\theta)})$ and $I(\hat{\theta_n}) \xrightarrow{P} I(\theta)$. As shown in part I, $\hat{\theta_n} \xrightarrow{D} N(\theta, \frac{1}{nI(\hat{\theta_n})})$. From part b, we just calculated that $I(\hat{\theta}) = \frac{1}{(\hat{\theta_n}-1)^2}$. Therefore, $\sigma = \sqrt{\frac{1}{n} * (\hat{\theta_n} - 1)^2} = \frac{\hat{\theta_n}-1}{\sqrt{n}}$.

Our 95% confidence interval is $\boxed{[\hat{\theta_n} \pm \frac{1.96}{\sqrt{n}}(\hat{\theta_n} - 1)]}$ where $\hat{\theta_n} = 1 + \frac{n}{\sum_{i=1}^{n} log(xi)}$.

d) Deriving the CDF for $p_\theta(x) = (\theta-1)x^{-\theta}$:

$$
\begin{aligned}
F(x) &= \int_1^x (\theta - 1)x^{-\theta} \\
F(x) &= -x^{-\theta+1}\Big|_1^x \\
F(x) &= 1 - x^{-\theta+1}
\end{aligned}
$$

Find $F^{-1}(U)$:

$$
\begin{aligned}
x &= 1 - (F^{-1}(U))^{-\theta+1} \\
F^{-1}(U) &= (1 - x)^{\frac{1}{-\theta+1}}
\end{aligned}
$$

Since $\theta = 2$, then $\boxed{F^{-1}(U) = \dfrac{1}{(1-u)}}$. The below code demonstrates the effectiveness of the 95% confidence interval.

```
n <- 100
theta <- 2
cdfi <- function(x) {
    return(1/(1 - x)^(theta - 1))
}
effective_arr <- rep(0, 10000)
for (i in 1:10000) {
    dataset <- sapply(runif(100, 0, 1), cdfi)
    # Construct 95% confidence interval
    z <- 1.96
    log_sums <- sum(sapply(dataset, log))
    theta_hat_n <- 1 + n/log_sums
    CI_lower <- theta_hat_n - z/sqrt(n) * (theta_hat_n - 1)
    CI_upper <- theta_hat_n + z/sqrt(n) * (theta_hat_n - 1)
    if (CI_lower <= theta & theta <= CI_upper) {
        effective_arr[i] <- 1
    }
}
mean(effective_arr)
```

```
## [1] 0.9522
```

Question 2: Non-Existence of MLE

$p = \frac{1}{1+log(\theta)}$ where $\theta > 1$.

Case 1: all observations are 1

$$
\begin{aligned}
L_n &= \prod_{i=1}^{n} p = p^n \\
ln &= log(Ln) \\
&= \sum_{i=1}^{n} log(p) \\
\frac{dl_n}{dp} &= \sum_{i=1}^{n} \frac{1}{p} \\
&= \frac{n}{p} = 0
\end{aligned}
$$

There is no MLE! There is no value of θ that makes the above derivative 0, and no smallest value of $\theta > 1$ that can maximize the log-likelihood function ln since ln is a monotonically decreasing function with negative derivative for all $\theta < 1$.

Case 2: all observations are 0

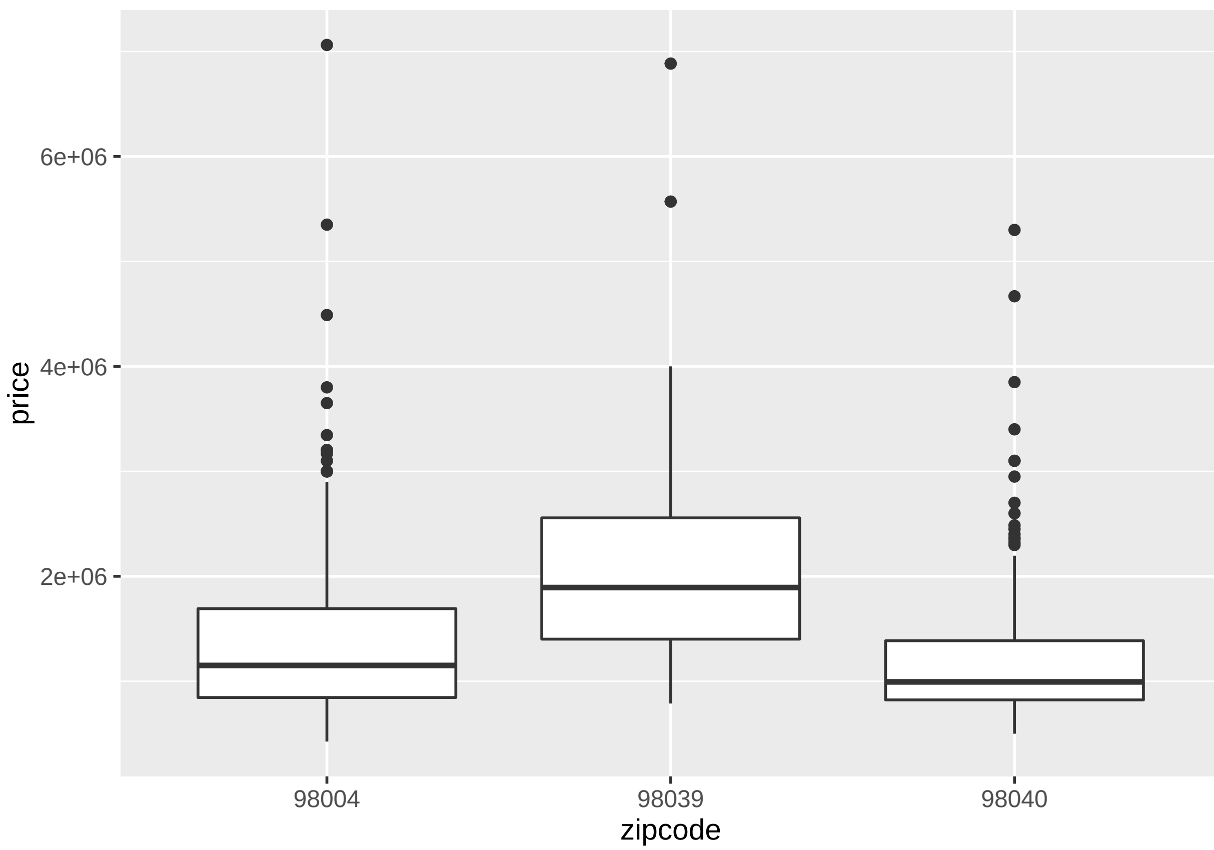
$$
\begin{aligned}
L_n &= \prod_{i=1}^{n} (1-p) = (1-p)^n \\
ln &= log(Ln) \\
&= \sum_{i=1}^{n} log(1-p) \\
\frac{dl_n}{dp} &= \sum_{i=1}^{n} \frac{1}{1-p} \\
&= \frac{n}{1-p} = 0
\end{aligned}
$$

We see again there is no MLE! There is no value of θ that makes the above derivative 0. There is also no largest value of $\theta > 1$ that can maximize the log-likelihood function ln since ln is a monotonically increasing function with positive derivative for all $\theta > 1$.

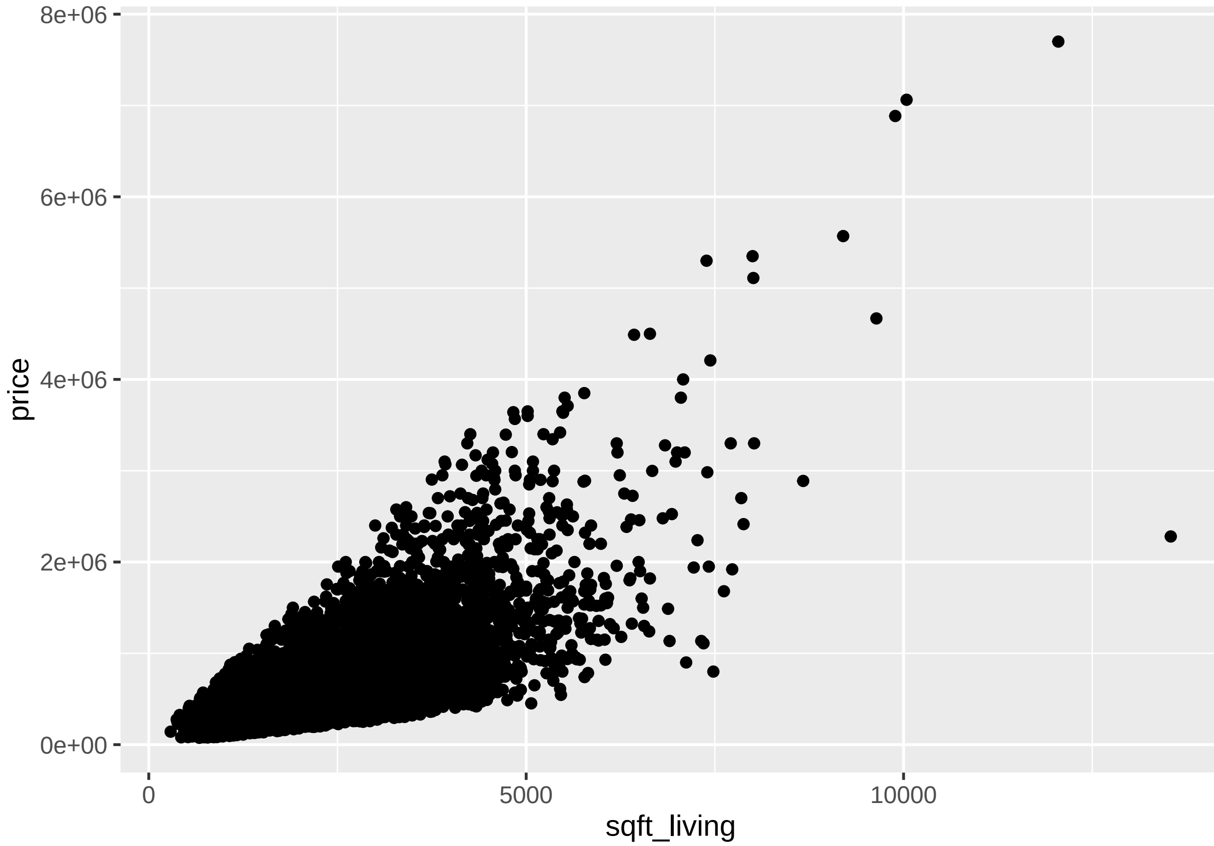
Question 3: Exploratory Data Analysis

```r
housingprices <- read.csv("housingprice.csv", header = TRUE)
# 3a)

# prices_zipcode <- aggregate(housingprices[, c('zipcode',
# 'price')], by = list(housingprices$zipcode), FUN = mean)
# prices_zipcode <- prices_zipcode[, c('zipcode', 'price')]
# prices_zipcode <-
# prices_zipcode[order(prices_zipcode$price, decreasing =
# TRUE), ]
prices_zipcode <- tapply(housingprices$price, factor(housingprices$zipcode),
    mean)
prices_zipcode <- sort(prices_zipcode, decreasing = TRUE)
max_three <- names(prices_zipcode)[1:3]
df <- data.frame(housingprices[c(which(housingprices$zipcode ==
    max_three[1]), which(housingprices$zipcode == max_three[2]),
    which(housingprices$zipcode == max_three[3])), c("zipcode",
    "price")])
df$zipcode <- factor(df$zipcode)
ggplot(df, aes(x = zipcode, y = price)) + geom_boxplot()
```




```r
# 3b)
ggplot(housingprices[, c("sqft_living", "price")], aes(x = sqft_living,
    y = price)) + geom_point()
```


8e+06
6e+06
4e+06
2e+06
0e+00
price
0
5000
10000
sqft_living

Question 4: A Simple Linear Model

```
training_data <- read.csv("train.data.csv", header = TRUE)
testing_data <- read.csv("test.data.csv", header = TRUE)

testing_r2 <- function(model, data) {
    predictions <- predict(model, data)
    RSS <- sum((data$price - predictions)^2)
    TSS <- sum((data$price - mean(data$price))^2)
    return(1 - RSS/TSS)
}

# 4a)
training_model <- lm(price ~ bedrooms + bathrooms + sqft_living +
    sqft_lot, data = training_data)
coef(training_model)
```

```
##   (Intercept)      bedrooms     bathrooms   sqft_living      sqft_lot
##  8.083299e+04 -5.929696e+04  3.681656e+03  3.166857e+02 -4.267368e-01
```

```
paste("Training data R^2:", summary(training_model)$r.squared)
```

```
## [1] "Training data R^2: 0.510113853079458"
```

```
paste("Testing data R^2:", testing_r2(training_model, testing_data))
```

```
## [1] "Testing data R^2: 0.50499446140371"
```

```
# 4b)
training_model <- lm(price ~ bedrooms + bathrooms + sqft_living +
    sqft_lot + factor(zipcode), data = training_data)
coef(training_model)
```

```
##          (Intercept)             bedrooms            bathrooms
##        -1.058001e+05        -4.609227e+04         9.793748e+03
##          sqft_living             sqft_lot factor(zipcode)98002
##         2.721327e+02         2.535685e-01         3.201302e+04
## factor(zipcode)98003 factor(zipcode)98004 factor(zipcode)98005
##         3.270864e+03         8.425761e+05         3.432664e+05
## factor(zipcode)98006 factor(zipcode)98007 factor(zipcode)98008
##         3.216824e+05         2.837032e+05         3.240625e+05
## factor(zipcode)98010 factor(zipcode)98011 factor(zipcode)98014
##         8.491370e+04         1.203289e+05         5.808457e+04
## factor(zipcode)98019 factor(zipcode)98022 factor(zipcode)98023
##         6.187852e+04         3.449420e+04        -6.954149e+03
## factor(zipcode)98024 factor(zipcode)98027 factor(zipcode)98028
##         1.629972e+05         1.756781e+05         1.261105e+05
## factor(zipcode)98029 factor(zipcode)98030 factor(zipcode)98031
##         2.286935e+05         4.632170e+03         2.262731e+04
## factor(zipcode)98032 factor(zipcode)98033 factor(zipcode)98034
##         3.002358e+04         3.976388e+05         2.283948e+05
## factor(zipcode)98038 factor(zipcode)98039 factor(zipcode)98040
##         1.670862e+04         1.379064e+06         6.294742e+05
## factor(zipcode)98042 factor(zipcode)98045 factor(zipcode)98052
##         9.769198e+03         9.368780e+04         2.438620e+05
## factor(zipcode)98053 factor(zipcode)98055 factor(zipcode)98056
```

```
##         1.887699e+05         5.795220e+04         1.222032e+05
## factor(zipcode)98058 factor(zipcode)98059 factor(zipcode)98065
##         4.082437e+04         9.247429e+04         6.522398e+04
## factor(zipcode)98070 factor(zipcode)98072 factor(zipcode)98074
##         1.739593e+05         1.568994e+05         2.121611e+05
## factor(zipcode)98075 factor(zipcode)98077 factor(zipcode)98092
##         2.170887e+05         1.352757e+05        -1.721431e+04
## factor(zipcode)98102 factor(zipcode)98103 factor(zipcode)98105
##         5.472578e+05         3.598194e+05         5.275141e+05
## factor(zipcode)98106 factor(zipcode)98107 factor(zipcode)98108
##         1.445443e+05         3.819256e+05         1.232893e+05
## factor(zipcode)98109 factor(zipcode)98112 factor(zipcode)98115
##         5.432840e+05         6.602320e+05         3.583508e+05
## factor(zipcode)98116 factor(zipcode)98117 factor(zipcode)98118
##         3.426457e+05         3.459851e+05         1.964043e+05
## factor(zipcode)98119 factor(zipcode)98122 factor(zipcode)98125
##         5.383541e+05         3.822395e+05         2.221850e+05
## factor(zipcode)98126 factor(zipcode)98133 factor(zipcode)98136
##         2.311842e+05         1.817193e+05         3.057012e+05
## factor(zipcode)98144 factor(zipcode)98146 factor(zipcode)98148
##         2.899473e+05         1.441186e+05         7.581709e+04
## factor(zipcode)98155 factor(zipcode)98166 factor(zipcode)98168
##         1.811908e+05         1.324914e+05         6.673272e+04
## factor(zipcode)98177 factor(zipcode)98178 factor(zipcode)98188
##         2.691790e+05         8.042988e+04         4.064958e+04
## factor(zipcode)98198 factor(zipcode)98199
##         6.079836e+04         4.237026e+05
```

```r
paste("Training data R^2:", summary(training_model)$r.squared)
```

```
## [1] "Training data R^2: 0.739275456819292"
```

```r
paste("Testing data R^2:", testing_r2(training_model, testing_data))
```

```
## [1] "Testing data R^2: 0.737866845356188"
```

```r
# 4c)
billgates_house <- read.csv("fancyhouse.csv", header = TRUE)
billgates_house$zipcode <- factor(billgates_house$zipcode)
paste("Predicted price of Bill Gate's House:", predict(training_model,
    billgates_house))
```

```
## [1] "Predicted price of Bill Gate's House: 14813055.9494897"
```

The predicted price seems reasonable because for every parameter with a positive coefficient in the linear model, Bill Gate's is higher than the most expensive house in the training set. For the only parameter with a negative coefficient (number of bedrooms), Bill Gate's is higher by a bit but there are more positive coefficients than negative ones. This explains why his house has a higher price than the most expensive house in the training dataset, which costs $7,700,000 (found using command *training_data[which(training_data$price == max(training_data$price)),]*). For comparison:

```r
df <- t(data.frame(c("Bill Gates", 8, 25, 50000, 225000, 98039),
    c("Most expensive training house", 6, 8, 12050, 27600, 98102)))
colnames(df) <- c("", "bedrooms", "bathrooms", "sqft_living",
    "sqft_lot", "zipcode")
rownames(df) <- rep("", 2)
kable(df)
```

	bedrooms	bathrooms	sqft_living	sqft_lot	zipcode
Bill Gates	8	25	50000	225000	98039
Most expensive training house	6	8	12050	27600	98102

4d) $R^2 = 1 - \frac{RSS}{TSS}$, $RSS = ||Y - X\hat{\beta}||_2^2$ for design matrix without an extra covariate, and $RSS = ||Y - X_1\hat{\beta}_1||_2^2$ when an extra covariate is added to the design matrix. Compared to X, X_1 has an extra column and β_1 has an extra element compared to β. We see that if the extra element in β_1 is 0, or if the extra column in X_1 is entirely zeros, then the value of RSS is the same. However, if the extra element in β_1 is non-zero and the extra column in X_1 has non-zero elements, then we see that the quantity $Y - X_1\hat{\beta}_1$ is smaller and thus RSS is smaller. Because TSS is the same for both, then R^2 must be the same or higher for the design matrix with an extra covariate, because RSS must either be the same value or lower with an extra covariate.

Question 5: Feature Engineering

```r
# 5a)
training_model <- lm(price ~ bedrooms + bathrooms + sqft_living +
    sqft_lot + factor(zipcode) + bedrooms * bathrooms, data = training_data)
coef(training_model)
```

```
##         (Intercept)            bedrooms           bathrooms
##        3.288713e+04       -8.495619e+04       -5.505102e+04
##         sqft_living            sqft_lot factor(zipcode)98002
##        2.674710e+02        2.460143e-01        2.648124e+04
## factor(zipcode)98003 factor(zipcode)98004 factor(zipcode)98005
##        3.763407e+03        8.374431e+05        3.446109e+05
## factor(zipcode)98006 factor(zipcode)98007 factor(zipcode)98008
##        3.209190e+05        2.819844e+05        3.240175e+05
## factor(zipcode)98010 factor(zipcode)98011 factor(zipcode)98014
##        8.229616e+04        1.211927e+05        5.624113e+04
## factor(zipcode)98019 factor(zipcode)98022 factor(zipcode)98023
##        6.147858e+04        3.375728e+04       -6.497852e+03
## factor(zipcode)98024 factor(zipcode)98027 factor(zipcode)98028
##        1.571200e+05        1.763106e+05        1.272569e+05
## factor(zipcode)98029 factor(zipcode)98030 factor(zipcode)98031
##        2.308649e+05        5.396524e+03        2.345869e+04
## factor(zipcode)98032 factor(zipcode)98033 factor(zipcode)98034
##        2.698915e+04        3.956476e+05        2.286537e+05
## factor(zipcode)98038 factor(zipcode)98039 factor(zipcode)98040
##        1.908755e+04        1.360260e+06        6.229801e+05
## factor(zipcode)98042 factor(zipcode)98045 factor(zipcode)98052
##        9.825421e+03        9.343987e+04        2.449436e+05
## factor(zipcode)98053 factor(zipcode)98055 factor(zipcode)98056
##        1.908912e+05        5.353038e+04        1.194875e+05
## factor(zipcode)98058 factor(zipcode)98059 factor(zipcode)98065
##        4.028932e+04        9.230562e+04        6.491609e+04
## factor(zipcode)98070 factor(zipcode)98072 factor(zipcode)98074
##        1.687050e+05        1.598808e+05        2.141598e+05
## factor(zipcode)98075 factor(zipcode)98077 factor(zipcode)98092
##        2.166563e+05        1.367925e+05       -1.513386e+04
## factor(zipcode)98102 factor(zipcode)98103 factor(zipcode)98105
##        5.414539e+05        3.552026e+05        5.231448e+05
## factor(zipcode)98106 factor(zipcode)98107 factor(zipcode)98108
##        1.376446e+05        3.767818e+05        1.176357e+05
## factor(zipcode)98109 factor(zipcode)98112 factor(zipcode)98115
##        5.417030e+05        6.559529e+05        3.528393e+05
## factor(zipcode)98116 factor(zipcode)98117 factor(zipcode)98118
##        3.395878e+05        3.391599e+05        1.877968e+05
## factor(zipcode)98119 factor(zipcode)98122 factor(zipcode)98125
##        5.343866e+05        3.800954e+05        2.176232e+05
## factor(zipcode)98126 factor(zipcode)98133 factor(zipcode)98136
##        2.227802e+05        1.765010e+05        3.015977e+05
## factor(zipcode)98144 factor(zipcode)98146 factor(zipcode)98148
##        2.834585e+05        1.368936e+05        7.042184e+04
## factor(zipcode)98155 factor(zipcode)98166 factor(zipcode)98168
##        1.757162e+05        1.308916e+05        5.810352e+04
## factor(zipcode)98177 factor(zipcode)98178 factor(zipcode)98188
```

```
##         2.672652e+05         7.662911e+04         3.878558e+04
## factor(zipcode)98198 factor(zipcode)98199   bedrooms:bathrooms
##         5.630631e+04         4.216916e+05         1.887988e+04
```

```r
paste("Training data R^2:", summary(training_model)$r.squared)
```

```
## [1] "Training data R^2: 0.741465054780292"
```

```r
paste("Testing data R^2:", testing_r2(training_model, testing_data))
```

```
## [1] "Testing data R^2: 0.739949169967571"
```

```r
# 5b)
training_poly_model <- lm(price ~ poly(bedrooms, 3) + poly(bathrooms,
    3) + sqft_living + sqft_lot + factor(zipcode), data = training_data)
coef(training_poly_model)
```

```
##          (Intercept)   poly(bedrooms, 3)1   poly(bedrooms, 3)2
##        -2.037385e+05        -4.734582e+06         9.038881e+05
##   poly(bedrooms, 3)3  poly(bathrooms, 3)1  poly(bathrooms, 3)2
##         2.104956e+06         1.797358e+06         6.257305e+06
##  poly(bathrooms, 3)3          sqft_living             sqft_lot
##         6.098551e+05         2.565749e+02         1.795807e-01
## factor(zipcode)98002 factor(zipcode)98003 factor(zipcode)98004
##         1.594996e+04         4.157833e+03         8.372914e+05
## factor(zipcode)98005 factor(zipcode)98006 factor(zipcode)98007
##         3.563414e+05         3.283897e+05         2.892097e+05
## factor(zipcode)98008 factor(zipcode)98010 factor(zipcode)98011
##         3.284092e+05         8.428998e+04         1.312653e+05
## factor(zipcode)98014 factor(zipcode)98019 factor(zipcode)98022
##         5.935352e+04         7.076723e+04         3.558774e+04
## factor(zipcode)98023 factor(zipcode)98024 factor(zipcode)98027
##        -8.942204e+03         1.488786e+05         1.790612e+05
## factor(zipcode)98028 factor(zipcode)98029 factor(zipcode)98030
##         1.318567e+05         2.333477e+05         8.612534e+03
## factor(zipcode)98031 factor(zipcode)98032 factor(zipcode)98033
##         2.507691e+04         1.591416e+04         3.956459e+05
## factor(zipcode)98034 factor(zipcode)98038 factor(zipcode)98039
##         2.270451e+05         2.914092e+04         1.269776e+06
## factor(zipcode)98040 factor(zipcode)98042 factor(zipcode)98045
##         6.186875e+05         1.217596e+04         9.251938e+04
## factor(zipcode)98052 factor(zipcode)98053 factor(zipcode)98055
##         2.542675e+05         1.976016e+05         4.899447e+04
## factor(zipcode)98056 factor(zipcode)98058 factor(zipcode)98059
##         1.169018e+05         4.321144e+04         9.843907e+04
## factor(zipcode)98065 factor(zipcode)98070 factor(zipcode)98072
##         6.948825e+04         1.721166e+05         1.673148e+05
## factor(zipcode)98074 factor(zipcode)98075 factor(zipcode)98077
##         2.198974e+05         2.214289e+05         1.396477e+05
## factor(zipcode)98092 factor(zipcode)98102 factor(zipcode)98103
##        -8.817786e+03         5.302535e+05         3.467289e+05
## factor(zipcode)98105 factor(zipcode)98106 factor(zipcode)98107
##         5.278381e+05         1.320419e+05         3.731281e+05
## factor(zipcode)98108 factor(zipcode)98109 factor(zipcode)98112
##         1.153430e+05         5.333587e+05         6.603222e+05
## factor(zipcode)98115 factor(zipcode)98116 factor(zipcode)98117
```

```
##         3.507414e+05         3.327696e+05         3.350527e+05
## factor(zipcode)98118 factor(zipcode)98119 factor(zipcode)98122
##         1.806900e+05         5.302867e+05         3.781770e+05
## factor(zipcode)98125 factor(zipcode)98126 factor(zipcode)98133
##         2.106042e+05         2.127758e+05         1.702163e+05
## factor(zipcode)98136 factor(zipcode)98144 factor(zipcode)98146
##         2.973260e+05         2.787402e+05         1.246372e+05
## factor(zipcode)98148 factor(zipcode)98155 factor(zipcode)98166
##         5.960636e+04         1.670980e+05         1.227383e+05
## factor(zipcode)98168 factor(zipcode)98177 factor(zipcode)98178
##         4.042458e+04         2.634848e+05         6.734860e+04
## factor(zipcode)98188 factor(zipcode)98198 factor(zipcode)98199
##         3.072127e+04         5.139431e+04         4.192384e+05
```

```r
paste("Training data R^2:", summary(training_poly_model)$r.squared)
```

```
## [1] "Training data R^2: 0.756505031665591"
```

```r
paste("Testing data R^2:", testing_r2(training_poly_model, testing_data))
```

```
## [1] "Testing data R^2: 0.747827655562612"
```