Part 1

Question 1

```
df <- read.csv('/Users/jazzopardi/Desktop/R/699/tech_salary_data.csv')

df['level'] = NULL # file description says we won't be using this for analysis, so removing from df
```

Question 2

```
str(df)
```

```
'data.frame':	62642 obs. of  23 variables:
 $ timestamp            : chr  "6/7/2017 11:33:27" "6/10/2017 17:11:29" "6/11/2017 14:53:57" "6/17/2017 0:23:14" ...
 $ company              : chr  "Oracle" "eBay" "Amazon" "Apple" ...
 $ title                : chr  "Product Manager" "Software Engineer" "Product Manager" "Software Engineering Manager" ...
 $ totalyearlycompensation: int  127000 100000 310000 372000 157000 208000 300000 156000 120000 201000 ...
 $ location             : chr  "Redwood City, CA" "San Francisco, CA" "Seattle, WA" "Sunnyvale, CA" ...
 $ yearsofexperience     : num  1.5 5 8 7 5 8.5 15 4 3 12 ...
 $ yearsatcompany        : num  1.5 3 0 5 3 8.5 11 4 1 6 ...
 $ tag                  : chr  NA NA NA NA ...
 $ basesalary           : int  107000 0 155000 157000 0 0 180000 135000 0 157000 ...
 $ stockgrantvalue       : num  20000 0 0 180000 0 0 65000 8000 0 26000 ...
 $ bonus                : num  10000 0 0 35000 0 0 55000 13000 0 28000 ...
 $ gender               : chr  NA NA NA NA ...
 $ otherdetails          : chr  NA NA NA NA ...
 $ cityid               : int  7392 7419 11527 7472 7322 11527 11521 11527 11521 11527 ...
 $ dmaid                : int  807 807 819 807 807 819 819 819 819 819 ...
 $ rowNumber            : int  1 2 3 7 9 11 12 13 15 16 ...
 $ Masters_Degree        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Bachelors_Degree      : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Doctorate_Degree      : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Highschool           : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Some_College          : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Race                 : chr  NA NA NA NA ...
 $ Education            : chr  NA NA NA NA ...
>
```

Timestamp - categorical, company - categorical, level - categorical, title - categorical, totalcompensation - numerical, location - numerical, years of exp. + years at company -

numerical, tag - categorical, basesalary + grantvalue + bonus - numerical, gender - categorical, otherdetails, categorical, cityid - categorical, dmaid - categorical, rownumber - numerical master_degree - categorical, doctoral_degree - categorical, highschool - categorical, some_college - categorical, race - categorical, education - categorical

Question 3

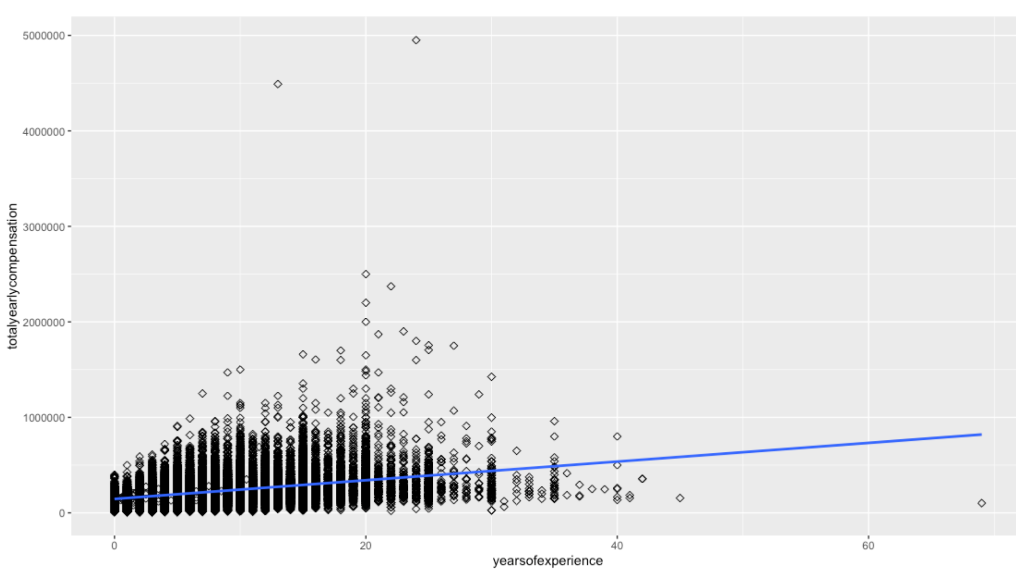
```
set.seed(10)

0.6 * nrow(df) # 37585 is 60% (for training)

sampler <- sample_n(df, nrow(df))

train.df <- slice(sampler, 1:37585) # we have selected our data for training

valid.df <- slice(sampler, 37586: nrow(df)) # we have 40% (for testing)
```

Data partitioning is important because we need to keep a subset of the available data so that we can use it to verify the model we create with the training data.

As such, the data we don't use is appropriately called testing data. In an ideal situation, you will have training, validation and testing data - but that can be expensive and time consuming.

Question 4

```
ggplot(train.df, aes(x= yearsofexperience,
                 y= totalyearlycompensation)) + geom_point(
                  size = 2, shape = 23) + geom_smooth(method = lm)
```



According to the best line fit, as years of experience increase, so does yearly compensation. This makes sense given that the more experience you have, the more knowledge and expertise you accumulate, and thus the more likely you are to be better compensated. It is important to note that there are several outliers in this training data.

Question 5

```
names(train.df)

cor.data <- train.df[, c('yearsofexperience', 'totalyearlycompensation')]

cor(cor.data) # correlation matrix = 0.4135291

cor.test(train.df$yearsofexperience, train.df$totalyearlycompensation)
```

```
                          yearsofexperience   totalyearlycompensation
yearsofexperience             1.0000000            0.4135291
totalyearlycompensation       0.4135291            1.0000000
```

```
                Pearson's product-moment correlation

data:  train.df$yearsofexperience and train.df$totalyearlycompensation
t = 88.049, df = 37583, p-value < 0.00000000000000022
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.4051130 0.4218752
sample estimates:
      cor
0.4135291
```

The p value is close to 0, which indicates that the relationship is statistically significant. The correlation sits at 0.413, which suggests a low/medium correlation between the two variables.

Question 6

```
model <- lm(totalyearlycompensation ~ yearsofexperience, data = train.df)
```

```
summary(model)
Call:
lm(formula = totalyearlycompensation ~ yearsofexperience, data = train.df)

Residuals:
    Min      1Q  Median      3Q     Max 
-718441  -71405  -14208   45121 4569490 

Coefficients:
                  Estimate Std. Error t value            Pr(>|t|)    
(Intercept)         145879       1028  141.83 <0.0000000000000002 ***
yearsofexperience     9776        111   88.05 <0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 125600 on 37583 degrees of freedom
Multiple R-squared:  0.171,	Adjusted R-squared:  0.171 
F-statistic:  7753 on 1 and 37583 DF,  p-value: < 0.00000000000000022
```

Question 7

From the summary() function, we can deduce that the minimum residual is -718441 and the maximum residual is 4569490.

```
train.df$resid <- model$residuals
```

```
View(train.df)
```

For the highest residual value, the observation earned $4,950,000.
For the lowest residual value, the observation earned $102,000.

```
View(train.df)  # highest and lowest residuals
```

```
max(model$fitted.values)
```

```
4950000 - 380509 # actual vs predicted
```

```
102000 - 820441.3 # actual vs predicted
```

The residuals are calculated by subtracting the model's predicted values from actual value.

There are more factors to take into account when calculating compensation, including company size, exogenous factors to the economy, the observation's own career goals etc. - SLR does not work for an analysis like this. What's worth noting about these results is that for the lowest residual, the yearsatcompany is listed as 69 - which is most likely a human error in the dataset.

Question 8

```
summary(model)

Call:
lm(formula = totalyearlycompensation ~ yearsofexperience, data = train.df)

Residuals:
    Min      1Q  Median      3Q     Max 
-718441  -71405  -14208   45121 4569490 

Coefficients:
                  Estimate Std. Error t value            Pr(>|t|)    
(Intercept)         145879       1028  141.83 <0.0000000000000002 ***
yearsofexperience     9776        111   88.05 <0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 125600 on 37583 degrees of freedom
Multiple R-squared:  0.171,	Adjusted R-squared:  0.171 
F-statistic:  7753 on 1 and 37583 DF,  p-value: < 0.00000000000000022
```

The regression equation is **y = 145879 + 9776(x)**

Input year = 15

```
res <- 145879 + (9776*15)
res  = 292519
```

Question 9

```
pred <- predict(model, train.df)
accuracy(pred, train.df$totalyearlycompensation)

pred_two <- predict(model, valid.df)
accuracy(pred_two, valid.df$totalyearlycompensation)
```

	ME	RMSE	MAE	MPE	MAPE
Test set	0.000000001918558	125636.1	83081.14	-40.51568	62.15659

	ME	RMSE	MAE	MPE	MAPE
Test set	-21.36998	124265.8	82168.27	-38.63684	59.99024

The purpose of this comparison is to see how well our model works on data it hasn't seen before. This is important for predictive modeling because future data will be unknown.

The MAE in both models are similar but large, suggesting that this model may not be the most accurate. Similarly the standard deviation between the residuals (the RMSE) is also large, again suggesting that this isn't a good fit. The culprit here is the outliers, which distort our model significantly given that they are considerable outliers. Also, it's a SLR model, which is ineffective given that there are undoubtedly more factors to take into account when predicting total yearly compensation.

Question 10

sd(train.df$totalyearlycompensation)

137989

summary(model)

Residual standard error: 125600 on 37583 degrees of freedom

There is a difference of 12,389 between the standard deviation of the overall model and the standard deviation of the residuals.

They are similar, all things considered, suggesting that the model fit well according to the data given - despite there being outliers.

Part 2

train.df.two <- read.csv('/Users/jazzopardi/Desktop/R/699/tech_salary_data.csv')

Question 1

anyNA(train.df.two) # there are NA values

which(colSums(is.na(train.df.two)) > 0)
# level # tag # gender # otherdetails # dmaid # race # education

These columns have missing values:

level	tag	gender	otherdetails	dmaid	Race	Education
3	9	13	14	16	23	24

b) NA values can produce biased estimates and lead to invalid conclusions.

c)

perc <- colSums(is.na(train.df.two))

x <- sort(perc, decreasing = TRUE)

names(x[(x/nrow(train.df.two)*100) > 20]) # gender, other details, race, education

train.df.two <- subset(train.df.two, select = -c(gender, otherdetails, Race, Education))

```
         3          9         13         14         16         23         24
> perc <- colSums(is.na(train.df.two))
> perc
              timestamp                 company                   level                   title totalyearlycompensation
                      0                       0                      15                       0                       0
               location       yearsofexperience          yearsatcompany                     tag              basesalary
                      0                       0                       0                     808                       0
        stockgrantvalue                   bonus                  gender            otherdetails                  cityid
                      0                       0                   19540                   22504                       0
                  dmaid               rowNumber         Masters_Degree        Bachelors_Degree        Doctorate_Degree
                      2                       0                       0                       0                       0
             Highschool            Some_College                    Race               Education
                      0                       0                   40215                   32272
> x <- sort(perc, decreasing = TRUE)
> x
                   Race               Education            otherdetails                  gender                     tag
                  40215                   32272                   22504                   19540                     808
                  level                   dmaid               timestamp                 company                   title
                     15                       2                       0                       0                       0
otalyearlycompensation                location       yearsofexperience          yearsatcompany              basesalary
                      0                       0                       0                       0                       0
        stockgrantvalue                   bonus                  cityid               rowNumber          Masters_Degree
                      0                       0                       0                       0                       0
       Bachelors_Degree        Doctorate_Degree              Highschool            Some_College
                      0                       0                       0                       0
> names(x[(x/nrow(train.df.two)*100) > 20])# gender, other details, race, education
[1] "Race"         "Education"    "otherdetails" "gender"
> train.df.two <- subset(train.df.two, select = -c(gender, otherdetails, Race, Education))
>
```

d)

names(x[(x/nrow(train.df.two)*100) < 20 & (x/nrow(train.df.two)*100) > 0]) # tag # dmaid

train.df.two$tag[is.na(train.df.two$tag)] <- sort(table(train.df.two$tag), decreasing = TRUE)[1] train.df.two$dmaid[is.na(train.df.two$dmaid)] <- sort(table(train.df.two$dmaid), decreasing = TRUE)[1]

e)

str(train.df.two)

rowid is categorical

train.df.two <- subset(train.df.two, select = -c(rowNumber)) # removing row number

A categorical variable with unique values and levels won't be good for predicting because the unique values may rarely occur in real life and thus their impact on the model would be negligent. In this case, it's better to remove the variable altogether or group it according to predefined levels.

f) The three location-related variables are:

1) Location `1050 Levels: Aachen, NW, Germany Aarhus, AR, Denmark Aberdeen Proving Ground, MD Abingdon, MD`
2) CityID `1045 Levels: 0 10 1153 1180 1182 1205 1206 1211 1221 1222`
3) Dmaid `150 Levels: 0 500 501 502 503 504`

train.df.two <- train.df.two[train.df.two$dmaid %in% names(sort(table(train.df.two$dmaid), decreasing = TRUE)[1:8]),]

g)

check <- as.data.frame(sort(table(train.df.two$company), decreasing=TRUE)[1:8])

check <- c('Amazon','Microsoft','Google','Facebook','Apple','Oracle',
'Salesforce', 'Cisco')

saldat3 <- filter(train.df.two, company %in% check)

nrow(saldat3)

25781

h)

check_two <- as.data.frame(sort(table(train.df.two$title), decreasing=TRUE)[1:8])

```
check_two <- c('Software Engineer', 'Product Manager','Software Engineering Manager',
          'Data Scientist','Hardware Engineer','Product Designer','Technical Program Manager',
          'Solution Architect')

saldat3 <- filter(saldat3, title %in% check_two)

nrow(saldat3) # 24391
```

i)

```
check_three <- as.data.frame(sort(table(train.df.two$tag), decreasing=TRUE)[1:8])


check_three<- c('Distributed Systems (Back-End)', 'Full Stack','API Development (Back-End)',
          'ML / AI','Web Development (Front-End)','Product','Data',
          'DevOps')

saldat3 <- filter(saldat3, tag %in% check_three)

nrow(saldat3) # 16411
```

j)

```
saldat3$level <- NULL
saldat3$timestamp <- NULL
```

k)

```
saldat3$basesalary <- NULL
saldat3$bonus <- NULL
saldat3$stockgrantvalue <- NULL
```

Question 2

```
set.seed(10)

0.6 * nrow(saldat3) # 9450

sampler <- sample_n(saldat3, nrow(saldat3))

new.train.df <- slice(sampler, 1:9450) # we have selected our data for training

new.valid.df <- slice(sampler, 9451: nrow(saldat3)) # we have 40% for testing
```

Question 3

cor.data <- new.train.df[, c('totalyearlycompensation', 'yearsofexperience',
'yearsatcompany')]

cor(cor.data)

```
                        totalyearlycompensation yearsofexperience yearsatcompany
totalyearlycompensation               1.0000000         0.4865041      0.2734783
yearsofexperience                     0.4865041         1.0000000      0.5167999
yearsatcompany                        0.2734783         0.5167999      1.0000000
```

Question 4

salary.lm <- lm(totalyearlycompensation ~ . , data = new.train.df)

salary.lm.step <- step(salary.lm, direction = 'backward')

summary(salary.lm.step)

```
> salary.lm.step <- step(salary.lm, direction = 'backward')
Start:  AIC=219055.1
totalyearlycompensation ~ company + title + yearsofexperience +
    yearsatcompany + tag + dmaid + rowNumber + Masters_Degree +
    Bachelors_Degree + Doctorate_Degree + Highschool + Some_College
```

```
Step:  AIC=219053.2
totalyearlycompensation ~ company + title + yearsofexperience +
    yearsatcompany + tag + dmaid + rowNumber + Masters_Degree +
    Bachelors_Degree + Doctorate_Degree

                    Df      Sum of Sq             RSS    AIC
<none>                                109461687680538 219053
- Masters_Degree     1    30467246568 109492154927106 219054
- Bachelors_Degree   1   116352489345 109578040169883 219061
- yearsatcompany     1   222150066131 109683837746669 219070
- rowNumber          1   482227536474 109943915217012 219093
- Doctorate_Degree   1   533280298015 109994967978553 219097
- tag                7  2003723900866 111465411581404 219211
- title              7  5103524222195 114565211902732 219470
- company            7 16069742910624 125531430591162 220334
- dmaid              7 16827352714643 126289040395181 220391
- yearsofexperience  1 23140568952657 132602256633194 220864
>
```

```
> summary(salary.lm.step)

Call:
lm(formula = totalyearlycompensation ~ company + title + yearsofexperience +
    yearsatcompany + tag + dmaid + rowNumber + Masters_Degree +
    Bachelors_Degree + Doctorate_Degree, data = new.train.df)

Residuals:
    Min      1Q  Median      3Q     Max
-420073  -50095   -7802   33608 4466329

Coefficients:
                          Estimate  Std. Error t value              Pr(>|t|)
(Intercept)              5563.8908   8589.4121   0.648               0.51715
companyApple            31161.9833   5849.4094   5.327     0.000000101944625 ***
companyCisco           -57561.9101   7787.4495  -7.392     0.000000000000157 ***
companyFacebook        116492.3762   4093.7275  28.456 < 0.0000000000000002 ***
companyGoogle           55262.8401   3803.2656  14.530 < 0.0000000000000002 ***
companyMicrosoft       -28566.5178   3136.9814  -9.106 < 0.0000000000000002 ***
companyOracle          -29171.7317   5455.3337  -5.347     0.000000091328639 ***
companySalesforce       10650.9112   5858.1447   1.818               0.06907 .
titleHardware Engineer -29551.3076  27124.5453  -1.089               0.27598
titleProduct Designer  -15993.3318  18094.0307  -0.884               0.37677
```

```
tagFull Stack                      217.6260   3729.7339   0.058                0.95347
tagML / AI                       37854.9122   4596.5154   8.236 < 0.0000000000000002 ***
tagProduct                       24135.0490   8964.1917   2.692                0.00711 **
tagWeb Development (Front-End)     541.1245   5732.0459   0.094                0.92479
dmaid501                        120765.0051   5773.0826  20.919 < 0.0000000000000002 ***
dmaid506                        100225.9500   8595.7786  11.660 < 0.0000000000000002 ***
dmaid511                         80625.2104   9763.7606   8.258 < 0.0000000000000002 ***
dmaid635                         72008.7805  10299.8213   6.991    0.000000000002909 ***
dmaid803                         87661.5883  10061.7447   8.712 < 0.0000000000000002 ***
dmaid807                        134830.1864   3949.8008  34.136 < 0.0000000000000002 ***
dmaid819                        119648.3539   3552.3490  33.681 < 0.0000000000000002 ***
rowNumber                            0.3600      0.0559   6.440    0.000000000125078 ***
Masters_Degree                   -4712.0576   2910.8169  -1.619                0.10552
Bachelors_Degree                -11471.9996   3626.3706  -3.163                0.00156 **
Doctorate_Degree                 43798.5849   6467.0053   6.773    0.000000000013403 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 107800 on 9415 degrees of freedom
Multiple R-squared:  0.4932,	Adjusted R-squared:  0.4914 
F-statistic: 269.5 on 34 and 9415 DF,  p-value: < 0.00000000000000022

>
```

Question 5

sst <- new.train.df$totalyearlycompensation - mean(new.train.df$totalyearlycompensation)
sst <- sst^2
sst <- sum(sst) # the difference between the mean and the observed value

215998347738862

Question 6

ssr <- salary.lm.step$fitted.values - mean(new.train.df$totalyearlycompensation)

ssr <- ssr^2

```
ssr <- sum(ssr)
```

106536660058325

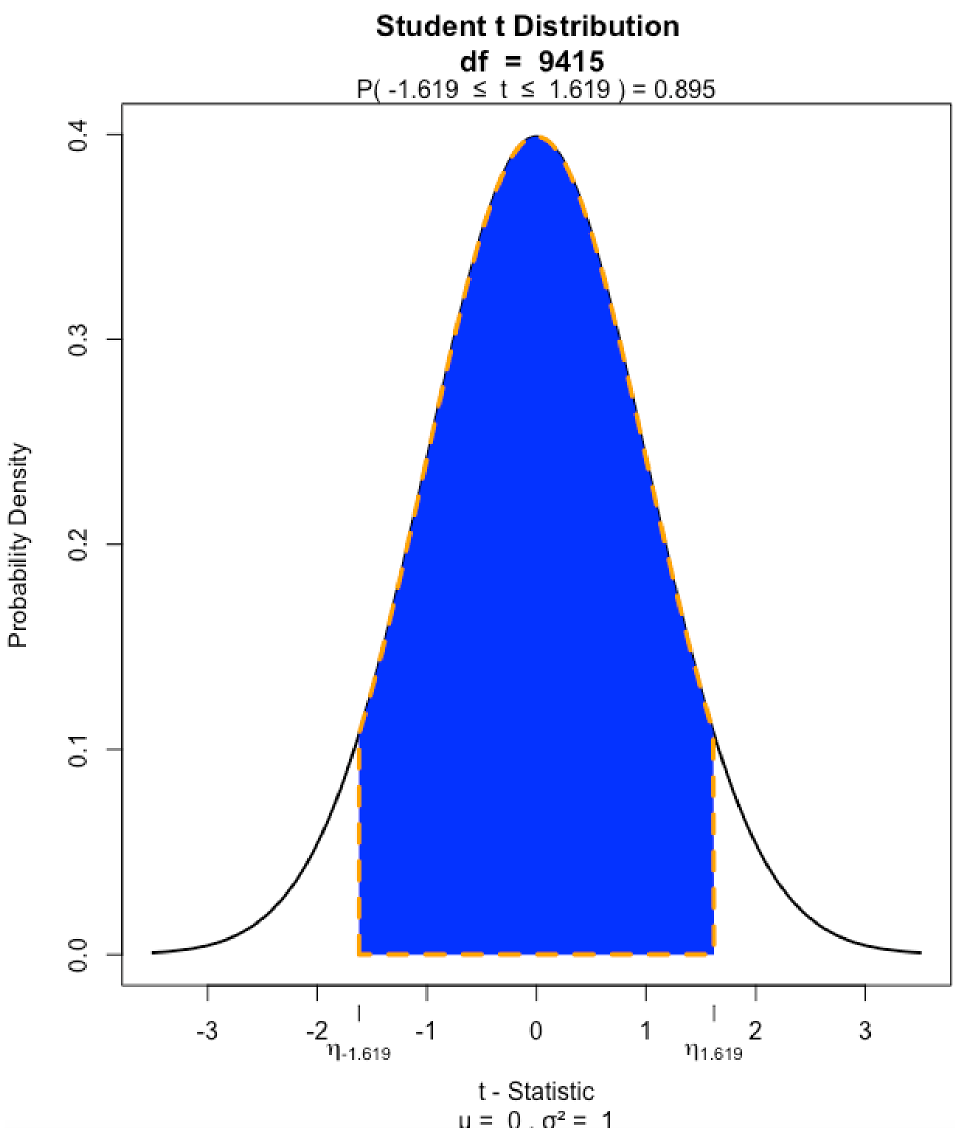
Question 7

```
ssr / sst
```

0.4932291

Question 8

The predictor I chose with “Master’s Degree.”



89.5 % of the curve is shaded

1 - 0.895 = 0.105 (p value)

The p-value is the “unshaded” part of the curve which tells us whether the value chosen is up to chance or not.

```
Masters_Degree                  -4712.0576    2910.8169   -1.619               0.10552
```

Question 10

```
newframe <- data.frame(company = 'Facebook',
                       title = 'Product Designer',
                       yearsofexperience = 5,
                       yearsatcompany = 2,
                       tag = 'Data',
                       dmaid = '501',
                       Masters_Degree = 1,
                       Bachelors_Degree = 1,
                       Doctorate_Degree = 0,
                       rowNumber = 0)

predict(salary.lm.step, newframe)
```

The tech worker will make $269888 annually based on this model.

```
> predict(salary.lm.step, newframe)
       1
269888.6
```

Question 11

```
pred <- predict(salary.lm.step, new.train.df)
accuracy(pred, new.train.df$totalyearlycompensation)
```

```
pred_two <- predict(salary.lm.step, new.valid.df)
accuracy(pred_two, new.valid.df$totalyearlycompensation)
```

```
> accuracy(pred, new.train.df$totalyearlycompensation)
                          ME     RMSE      MAE       MPE     MAPE
Test set 0.000000009145604 107867.1 62149.71 -9.531122 27.81096
> pred_two <- predict(salary.lm.step, new.valid.df)
> accuracy(pred_two, new.valid.df$totalyearlycompensation)
               ME   RMSE      MAE       MPE    MAPE
Test set 1554.252 121618 60921.05 -8.394922 26.6268
```

We can see here that the MAE for the MLR is lower than the MAE for SLR which suggests that our MLR model is more accurate. This makes sense given that the conditions of SLR rarely exist in everyday life.

The MAE in the train and testing set are similar which means our model is good at predicting based on unknown data. Of interesting note here is that the ME for the testing set is very small - perhaps a sign of overfitting.