Self Reflection

Kushal

4/27/2023

#Objective 1-Describe probability as a foundation of statistical modeling, including inference and maximum likelihood estimation

Probability is always considered as a foundation of statistical modeling. When we have a data which is randomly distributed and has uncertainty, we consider probability like happening something if some conditions are met or the chances of it. On most of the statistical scenario, our goal is to understand the probability of something, either it may be the probability of something appearing, error occuring, false interpretation happening and so on. So on every aspect and on most statistical alsgorithms and technique, probability is the foundation.

In our course, considering the linear regression, I see it as a probability of happening something depending on the other certain variables. For an example, finding out the probability of being broke based on the income and living standards parameters.

Inference is to predict something based on the other parameters. For an example, predicting loan amount of a student which is dependent on the rent, car price, distance from university, credits take and so on. We can implement multiple linear regression to predict the loan amount based on those factors. Both estimation and hypothesis comes under the inference. In inference, we may consider the sample data which represents the population data and use statistical techniques to predict and find out the conclusions assuming them as genralized population analysis.

For the maximum likelihood estimation, we focus on the statistical model parameters and try to find them correctly so that it can help to make the correct estimates or prediction. During our class activities, we built linear regression model using lm() to estimate the slope and intercept of the regression line. For and example with a simple regression model of y=mx+c, the parameters of the model intercept(c) and slope (m) makes the estimation of the variable y on a particular value of x. glance() function in r was used to find out the standard errors and confidence interval while estimating intercept and slope of the model.

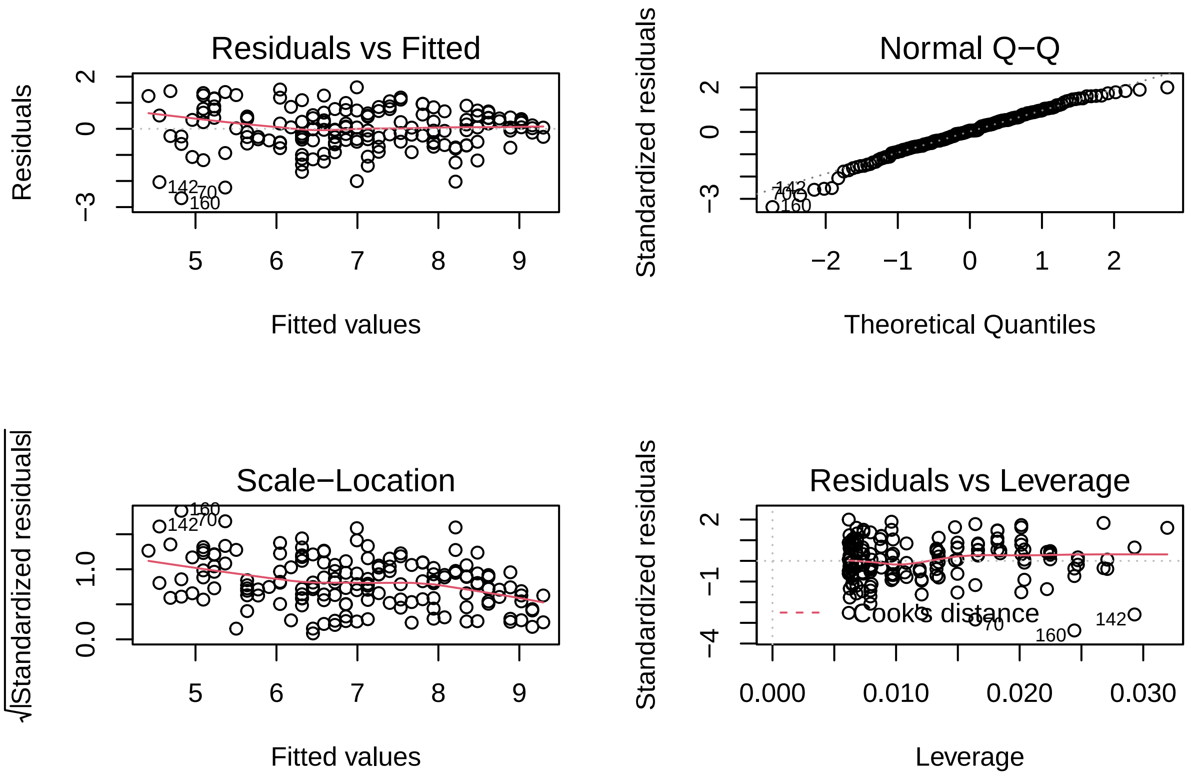
```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.0     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.1     v tibble    3.1.8
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(tidymodels)
```

```
## -- Attaching packages -------------------------------------- tidymodels 1.0.0 --
## v broom        1.0.4     v rsample      1.1.0
## v dials        1.0.0     v tune         1.0.0
## v infer         1.0.3     v workflows    1.0.0
## v modeldata     1.0.0     v workflowsets 1.0.0
## v parsnip       1.0.1     v yardstick    1.0.0
## v recipes       1.0.5
## -- Conflicts ----------------------------------------- tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Search for functions across packages at https://www.tidymodels.org/find/
```

```
hfi <- read.csv("https://www.openintro.org/data/csv/hfi.csv")
hfi_2016<- filter(hfi, year == 2016)
simple_model <- lm(pf_score ~ pf_expression_control, data = hfi_2016)
par(mfrow = c(2, 2))
plot(simple_model)
```



```
glance(simple_model)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squa~1 sigma stati~2  p.value    df logLik   AIC   BIC devia~3
##       <dbl>        <dbl> <dbl>   <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
## 1     0.714        0.712 0.799    400. 2.31e-45     1  -193.  391.  400.    102.
## # ... with 2 more variables: df.residual <int>, nobs <int>, and abbreviated
## #   variable names 1: adj.r.squared, 2: statistic, 3: deviance
```

```
tidy(simple_model)
```

```
## # A tibble: 2 x 5
##   term                  estimate std.error statistic  p.value
##   <chr>                    <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)              4.28     0.149       28.8 4.23e-65
## 2 pf_expression_control    0.542    0.0271      20.0 2.31e-45
```

During the activity 2 in the class, we worked on the simple linear regression model on the hfi dataset. In the above model, our objective was to predict the pf_score depending on pf_expression_control variable. We built a simple linear regression model using lm(). For the inference, we tried to predict the pf_score. Regarding the maximum likelihood estimation, we were able to get slope and the intercept for the above model. The slope value(m)- pf_expression_control is 0.541 and intercept c is 4.283. For the model accuracy and error calculation, tidy function was used to give the standard error whereas glance is used to calcuate the r- square and p value.

Objective 2-Determine and apply the appropriate generalized linear model for a specific data context

Based on specific data context, it is always important to determine which is the generalized linear model and what combination of independent attributes can play an important role to predict the dependent one with highe accuracy and low error. For these kind of evaluation, we have several statistical metrics and methodologies to evaluate the accuracy of the models. Fitting the linear regression model with correct attributes so the model can be accurately used to predict the dependent variable is important.

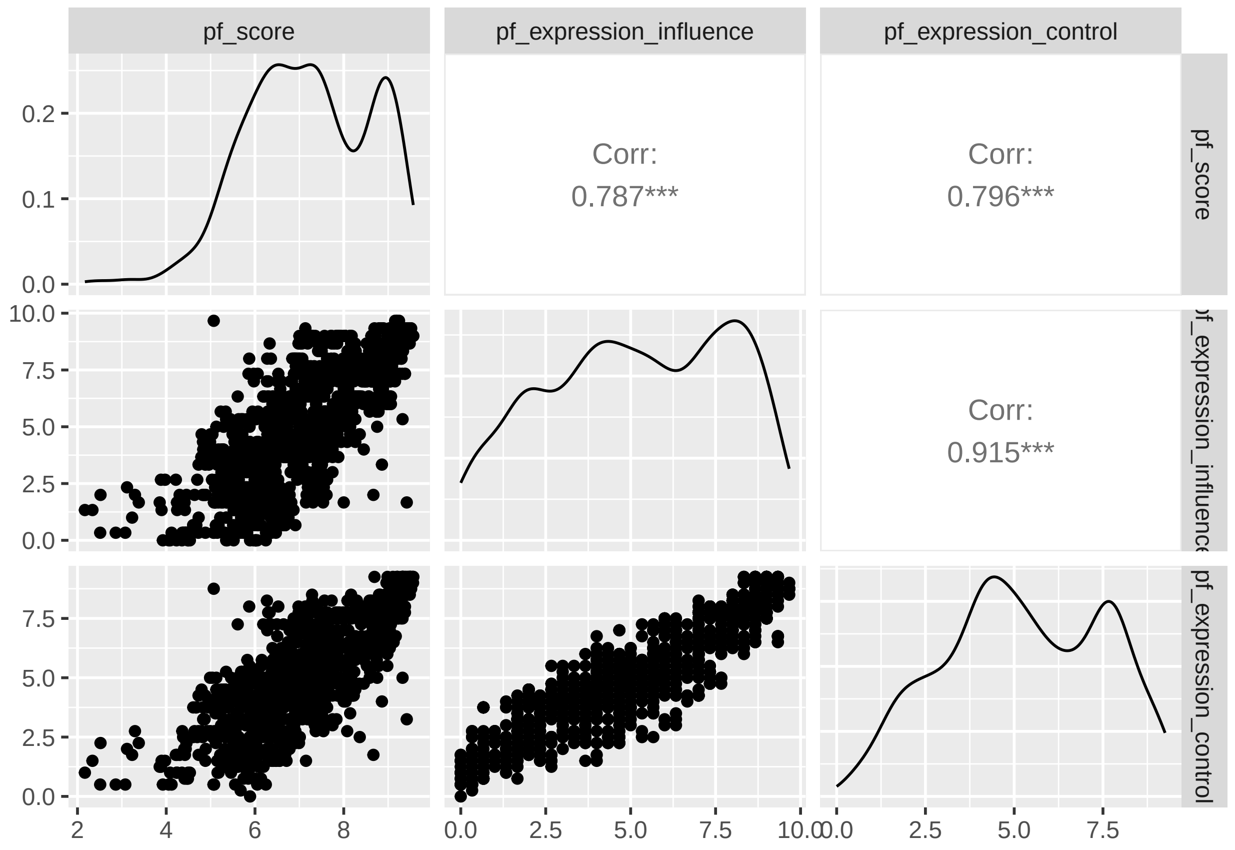
Below we have evaluated our model to predict the pf_score based on pf_expression_influence and pf_expression_control. The p-values for the slopes and intercept are below 0.05 and the standard error is also within the 0.05.

```r
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
hfi %>%
  select(pf_score, pf_expression_influence, pf_expression_control) %>%
  ggpairs()
```

```
## Warning: Removed 80 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 80 rows containing missing values


## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 80 rows containing missing values

## Warning: Removed 80 rows containing missing values (`geom_point()`).

## Warning: Removed 80 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 80 rows containing missing values

## Warning: Removed 80 rows containing missing values (`geom_point()`).
## Removed 80 rows containing missing values (`geom_point()`).

## Warning: Removed 80 rows containing non-finite values (`stat_density()`).
```



```
#fit the mlr model
m_pf <- lm(pf_score ~ pf_expression_influence + pf_expression_control, data = hfi)
tidy(m_pf)
```

```
## # A tibble: 3 x 5
##   term                    estimate std.error statistic  p.value
##   <chr>                      <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)                4.71     0.0566     83.1  0
## 2 pf_expression_influence    0.188    0.0205      9.19 1.44e-19
## 3 pf_expression_control      0.288    0.0242     11.9  2.84e-31
```

```
glance(m_pf)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squ~1 sigma stati~2   p.value    df logLik   AIC   BIC devia~3
##       <dbl>       <dbl> <dbl>   <dbl>     <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
## 1     0.655       0.655 0.808   1308. 8.19e-319     2 -1660. 3327. 3348.    897.
## # ... with 2 more variables: df.residual <int>, nobs <int>, and abbreviated
## #   variable names 1: adj.r.squared, 2: statistic, 3: deviance
```

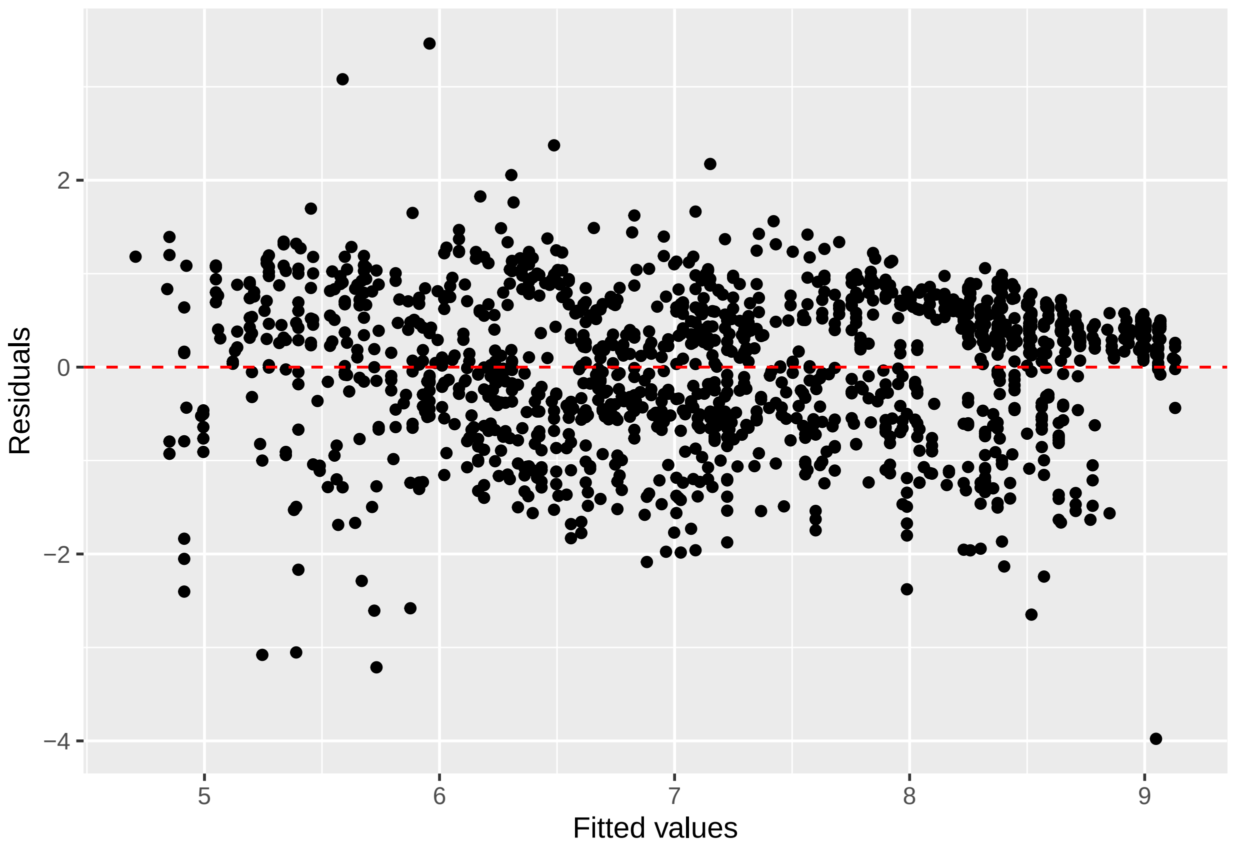
From the summary below, we can see that the F-statistic is 1308 and the p-value <0.00001 which is less than the 0.05(significance level). This means that the model has good accuracy as there is statistically significant relationship between the predictor variable and the response variable.

```
summary(m_pf)
```

```
##
## Call:
## lm(formula = pf_score ~ pf_expression_influence + pf_expression_control,
##     data = hfi)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9776 -0.5338  0.1493  0.5807  3.4627
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)             4.70699    0.05664  83.102   <2e-16 ***
## pf_expression_influence  0.18812    0.02048   9.187   <2e-16 ***
## pf_expression_control    0.28829    0.02417  11.926   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8077 on 1375 degrees of freedom
##   (80 observations deleted due to missingness)
## Multiple R-squared:  0.6554, Adjusted R-squared:  0.6549
## F-statistic:  1308 on 2 and 1375 DF,  p-value: < 2.2e-16
```

Also the plot between residuals vs. fitted (predicted) values are created which seems like randomly distributed that has the significance notifying no significant predictors are missing and there is not any systematic errors present on the model. Also from the graph below it looks like they have the predictors and response variable have linear relationship.

```r
# obtain fitted values and residuals
m_pf_aug <- augment(m_pf)

# plot fitted values and residuals
ggplot(data = m_pf_aug, aes(x = .fitted, y = .resid)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  xlab("Fitted values") +
  ylab("Residuals")
```



We have used the above created model to predict the certain values of pf_score based. For the below context, though the actual value is 8.747, the nearly predicted value was 8.23.

```
hfi %>%
  filter(countries == "United States" & year == 2016) %>%
  select(pf_score, pf_expression_influence, pf_expression_control)
```

```
##   pf_score pf_expression_influence pf_expression_control
## 1  8.74731                       8                     7
```

```
hfi %>%
  filter(countries == "United States" & year == 2016) %>%
  predict(m_pf, .)
```

```
##       1
## 8.23002
```

In this way we have applied the appropriate generalized model based on the dataset. Also we have evaluated the model using several statistical methodologies and graphs to detremine if the model is a good fit or not and how the model is doing to predict the depednent variable.

#Objective 3: Conduct model selection for a set of candidate models

Each model can have their own significance and own way of working on the dataset. In my project, to predict the stock closing price based on the date, I have implemented linear regression, random forest and polynomial regression. For the three models, I have calculated MSE and RMSE to analyse how each model is doing. for my project, model based on random forest has the lowest RMSE of 15.92 which is significantly lower than the linear and polynomial regression. It looks like for this dataset, with two variables, closing price depending only on the date, random forest is doing good compared to linear regression and polynomial regression.

But when we consider only linear and polynomial, we can see that polynomial regression has lower rmse. Also comapring AIC and BIC between the polynomial and linear regression models, polynomial has lower AIC and BIC values indicating it as a best model.

If we had more predictors, the model performance based on other attributes can be different. These evaluation metrics can play a significant role. below are some of the r programming implemented for my project.

```r
library(tidyverse)
library(quantmod)
```

```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last

## Loading required package: TTR

##
## Attaching package: 'TTR'

## The following object is masked from 'package:dials':
##
##     momentum

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
begining <- as.Date("2015-01-01")
last_date <- as.Date("2023-04-05")
raw <- getSymbols("^GSPC", src = "yahoo", from = begining, to = last_date, auto.assign = FALSE)
stock_date = index(raw)
closingprice = as.numeric(raw[, "GSPC.Close"])
df <- data.frame(stock_date, closingprice)
```

#Using Linear Regression Model

```r
linear_model <- lm(closingprice ~ stock_date, data = df)
predict_stock <- predict(linear_model, newdata = df)
mse <- mean((predict_stock - df$closingprice)^2)
cat("MSE: ",mse,"\n")
```

```
## MSE:  87223.22
```

```r
rmse <- sqrt(mse)
cat("RMSE: ",rmse)
```

```
## RMSE:  295.3358
```

```r
summary(linear_model)
```

```
##
## Call:
## lm(formula = closingprice ~ stock_date, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1126.97  -168.69   -45.86   121.57   861.77
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.273e+04  1.338e+02  -95.17   <2e-16 ***
## stock_date   8.776e-01  7.448e-03  117.83   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 295.5 on 2076 degrees of freedom
## Multiple R-squared:  0.8699, Adjusted R-squared:  0.8699
## F-statistic: 1.388e+04 on 1 and 2076 DF,  p-value: < 2.2e-16
```

#Using Random Forest Regression

```r
library(randomForest)
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

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

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
rand_model <- randomForest(closingprice ~ stock_date, data = df, ntree = 200, mtry = 1)
predict_stock <- predict(rand_model, newdata = df)
mse <- mean((predict_stock - df$closingprice)^2)
cat("MSE: ",mse,"\n")
```

```
## MSE:  254.9094
```

```r
rmse <- sqrt(mse)
cat("RMSE: ",rmse)
```

```
## RMSE:  15.96588
```

```r
summary(rand_model)
```

```
##                 Length Class  Mode
## call               5   -none- call
## type               1   -none- character
## predicted       2078   -none- numeric
## mse              200   -none- numeric
## rsq              200   -none- numeric
## oob.times       2078   -none- numeric
## importance         1   -none- numeric
## importanceSD       0   -none- NULL
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
## mtry               1   -none- numeric
## forest            11   -none- list
## coefs              0   -none- NULL
## y               2078   -none- numeric
## test               0   -none- NULL
## inbag              0   -none- NULL
## terms              3   terms  call
```

#Using polynomial regression

```
date_num <- as.numeric(df$stock_date)
poly_model <- lm(closingprice ~ poly(date_num, 2, raw = TRUE), data = df)
predict_stock <- predict(poly_model, newdata = df)
mse <- mean((predict_stock - df$closingprice)^2)
cat("MSE: ",mse, "\n")
```

```
## MSE:  81045.81
```

```
rmse <- sqrt(mse)
cat("RMSE: ",rmse)
```

```
## RMSE:  284.6855
```

```
summary(poly_model)
```

```
##
## Call:
## lm(formula = closingprice ~ poly(date_num, 2, raw = TRUE), data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1057.64  -134.95   -24.54   117.58   822.98
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     2.456e+04  2.968e+03   8.274 2.28e-16 ***
## poly(date_num, 2, raw = TRUE)1 -3.289e+00  3.314e-01  -9.925  < 2e-16 ***
## poly(date_num, 2, raw = TRUE)2  1.161e-04  9.232e-06  12.576  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 284.9 on 2075 degrees of freedom
## Multiple R-squared:  0.8791, Adjusted R-squared:  0.879
## F-statistic:  7546 on 2 and 2075 DF,  p-value: < 2.2e-16
```

```
AIC(linear_model,  poly_model)
```

```
##              df      AIC
## linear_model  3 29542.91
## poly_model    4 29392.26
```

```
BIC(linear_model,  poly_model)
```

```
##              df      BIC
## linear_model  3 29559.82
## poly_model    4 29414.82
```

Also there are several validation techniques used to increase the performance of the model. One of them is k fold cross validation where we splitted the datasets into k equal folds and used k-1 folds as the training dataset for the model whereas the other one fold as a testing dataset. There will be k iterations where each fold get the chance to be in the testing dataset. It helps to evaluate the model correctly and efficiently. Below is the eample where we used 10 fold cross validation.

```
library(tidyverse)
library(tidymodels)
library(caret)
```

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

```
##
## Attaching package: 'caret'
```

```
## The following objects are masked from 'package:yardstick':
##
##     precision, recall, sensitivity, specificity
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

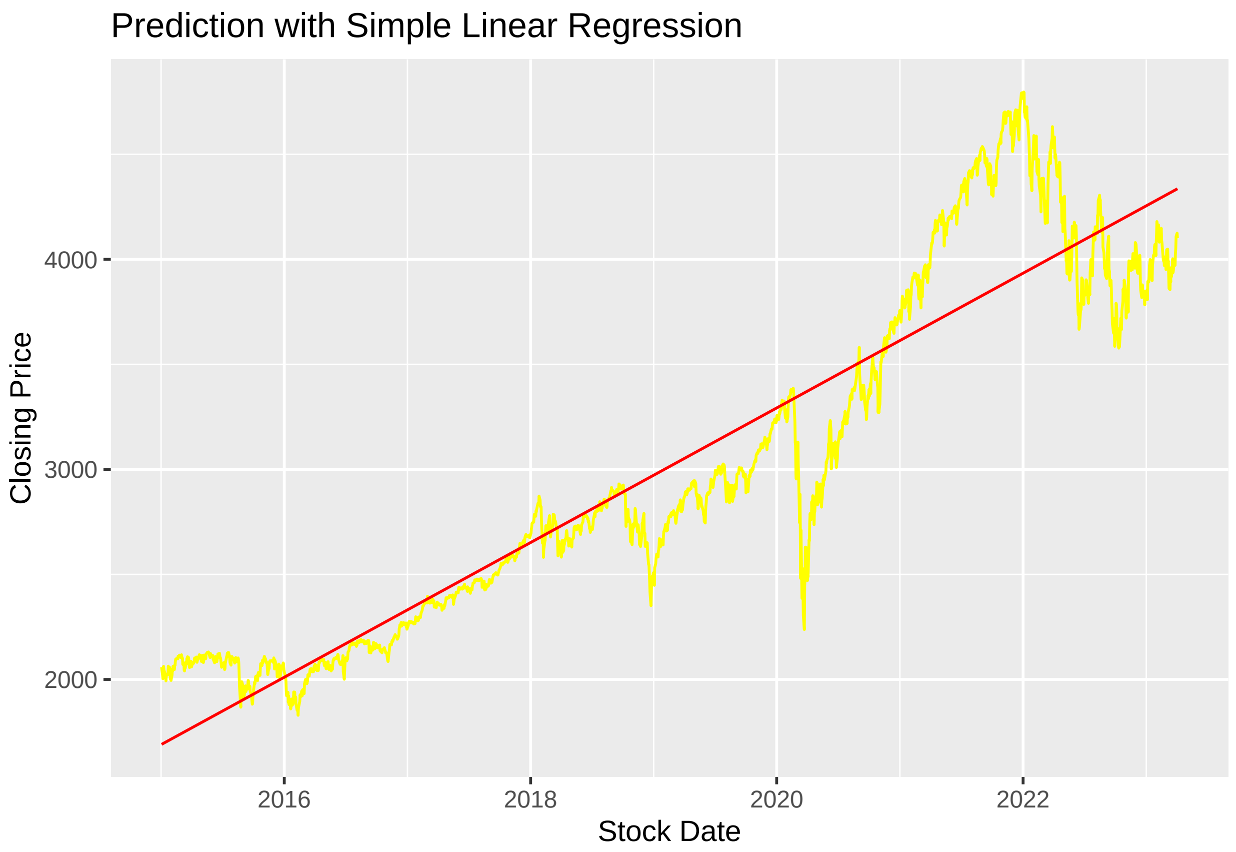
```
set.seed(105)
hfi <- read.csv("https://www.openintro.org/data/csv/hfi.csv")
hfi_2016<- filter(hfi, year == 2016)
trains <- trainControl(method = "cv",number = 10)
model <- train(pf_score ~ pf_expression_control, data = hfi_2016,
               trControl = trains,
               method = "lm")
print(model)
```

```
## Linear Regression
##
## 162 samples
##   1 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 146, 146, 146, 146, 145, 146, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.7915208  0.7345273  0.6320706
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

#Objective 4: Communicate the results of statistical models to a general audience

Creating the statistical models, optimizing them and increasing the accuracy has one kind of importance. But the major another important is how you present the model story to the audience. Making the general audience understand the significance of the statistical findings and trends is challenging. Conducting exploratory analysis and presenting the data story to the audience initially and then presenting the model working, prediction accuracy and evaluation can be the other story. I conducted several exploratory analysis of the data in the begining and then build the charts between original and predicted flow. Below is the example of my project where we can see the graph representing how close my model was to predict the values compared to the original one.

```r
predict_stock <- predict(linear_model, newdata = df)
ggplot() +
  geom_line(data = df, aes(x = stock_date, y = closingprice), color = "yellow") +
  geom_line(data = df, aes(x = stock_date, y = predict_stock), color = "red") +
  labs(title = "Prediction with Simple Linear Regression", x = "Stock Date", y = "Closing Price")
```



That was an example of the charts and graphs we can build. Also the evaluations such as RMSE, Root squared and so on can be showed and presented and informed that how the models is good based on those metrics. Also we can add p - values of the modle if it is less than 0.05 or not and the model is wignificant or not, presenting the residual plot if there is absent of patterns or not, looking at the histograms if it is normally distributed. These kind of graphs that I have explained on my first and second objective are the one that can be used to communicate with the general audience.

It's always important to present your results and findings in the simple way and inform how they can be important in the day to day cycle. Using visuals, graphs, text and blogs can be also useful for the communication. It's important to explain about the future plan of what more can be done on that models and talk about the model limitations and challenges. This can help more for the general audience to understand the story.

#Objective 5: Use programming software (i.e., R) to fit and assess statistical models

In the explanation of my first, second and third objective, I have provided the examples of several statistical models which i accomplished on the projects, class activities and brainstorming which represents the fitting and assessing the statistical models.

I have used function like lm(), glm() andso on to fit the model. Once the models is fitted, I assesed it using residual plots and model evaluation metrics like RMSE, R Square and so on and explained how well it fits the data. Several R functions are used to evaluate them. Functions like glance(), summary() etc were used. Diagnosing the model in graphical way is another important way.

#General difficulties, Challenges faced and the path followed in this course to learn.

The path to learn more about regression algorithms and implementation in this course was interesting and productive. There were a lot of new things I learnt regarding the regression. Though I have studied regression algorithms on another classes, understanding it in depth was the first time for me in this class.

The few difficulties for me was I was unknown about the assumptions of linear regression which was linearity and normality. I knew that these assumptions can affect the model performance highly. Another challenge was to how to deal with the multicollinearity problem exist between the predictors themselves. Learning the text book helped me a lot how we can avoid and handle them.

One of the major challenge that i still see is how we are gonna chose the best and efficient predictors that are good fit for the models and have higher accuracy. Exploratory analysis helped me a lot when chosing the best predictors among many of them. Building a model can be easier but evaluating the model if it is doing good as expected or not is important. Interpreting and evaluating the model with the p-value, rmse, r square, evaluating residual plots, checking linearity and normality were very important. These are some important learnings for me to conduct. The path was interesting and having findings and trends using the models were efficient. These all factors motivated me to analyse.