Create Dataset

# We determined the data we use in machine learning and found the following data from data sites, respectively. 1.GDP 2. Inflation Rate 3. Interest Rate 4. Oil Pri 5. Per Capita Income 6. Population 7. S&P 500 8. Unemployment Rate.

```
library(readxl)

#https://data.worldbank.org/indicator/NY.GDP.PCAP.CD?end=2021&locations=US&start=1960&view=ch
art
data_gpd <- read_excel("D:/R2/gdp.xlsx")

#https://ycharts.com/indicators/us_inflation_rate#:~:text=Basic%20Info,in%20price%20over%20a%
20year.
data_enf <- read_excel("D:/R2/Inflation Rate.xlsx")

#https://tradingeconomics.com/united-states/interest-rate#:~:text=Interest%20Rate%20in%20the%
20United,percent%20in%20December%20of%202008.
data_fa <- read_excel("D:/R2/Interest Rate.xlsx")

#https://www.statista.com/statistics/262860/uk-brent-crude-oil-price-changes-since-1976/
data_be <- read_excel("D:/R2/Oil Price.xlsx")

#https://data.worldbank.org/indicator/NY.GDP.PCAP.CD?end=2021&locations=US&start=1960&view=ch
art
data_ge <- read_excel("D:/R2/Per Capita Income.xlsx")

#https://www.macrotrends.net/countries/USA/united-states/population
data_population <- read_excel("D:/R2/Population.xlsx")

#https://www.macrotrends.net/2324/sp-500-historical-chart-data
data_sp <- read_excel("D:/R2/s_P500.xlsx")

#https://www.thebalancemoney.com/unemployment-rate-by-year-3305506
data_unemp <- read_excel("D:/R2/Unemployment Rate.xlsx")
```

#We have listed the data in order to combine it.

```
data_list <- list(data_gpd,
                  data_enf,
                  data_fa,
                  data_be,
                  data_ge,
                  data_population,
                  data_sp,
                  data_unemp)
```

#Using the merge function, we first combined 1 and 2 of the data we listed, and then combined the remaining ones using for i.

```
new_df <- data.frame()
new_df <- merge(data_list[1], data_list[2], by = "Year", all = TRUE)
for (i in 3:8) {
  new_df <- merge(new_df, data_list[i], by = "Year", all = TRUE)
}
head( new_df,10)
```

```
##    Year      GDP Inflation_Rate Interest_Rates Oil._Price Per_Capita_Income
## 1  1960   543.30          1.72%          3.21%        1.9             3.007
## 2  1961   563.30          1.01%          1.95%        1.8             3.067
## 3  1962   605.10          1.00%          2.71%        1.8             3.244
## 4  1963   638.60          1.32%          3.18%        1.8             3.375
## 5  1964   685.80          1.31%          3.50%        1.8             3.574
## 6  1965   743.70          1.61%          4.08%        1.8             3.828
## 7  1966   815.00          2.86%          5.11%        1.8             4.146
## 8  1967   861.70          3.09%          4.22%        1.8             4.336
## 9  1968   942.50          4.19%          5.66%        1.8             4.696
## 10 1969 1,019.90          5.46%          8.21%        1.8             5.032
##     Population S_P_500 Unemployment_Rate
## 1  176,188,578   55.85              6.6%
## 2  179,087,278   66.27              6.0%
## 3  181,917,809   62.32              5.5%
## 4  184,649,873   69.86              5.5%
## 5  187,277,378   81.37              5.0%
## 6  189,703,283   88.16              4.0%
## 7  191,830,975   85.18              3.8%
## 8  193,782,438   91.96              3.8%
## 9  195,743,427   98.38              3.4%
## 10 197,859,329   97.77              3.5%
```

Pre-Process

#While starting the preprocessing, some of the data had commas and percentiles, as these could cause us problems in machine learning, we eliminated them and then made the data numeric.

```
data <- new_df

data$GDP <- gsub("[%|,]", "", data$GDP)
data$Interest_Rates <- gsub("[%|,]", "", data$Interest_Rates)
data$Inflation_Rate <- gsub("[%|,]", "", data$Inflation_Rate)
data$Oil._Price <- gsub("[%|,]", "", data$Oil._Price)
data$Per_Capita_Income <- gsub("[%|,]", "", data$Per_Capita_Income)
data$Population <- gsub("[%|,]", "", data$Population)
data$S_P_500 <- gsub("[%|,]", "", data$S_P_500)
data$Unemployment_Rate <- gsub("[%|,]", "", data$Unemployment_Rate)



data$GDP = as.numeric(data$GDP)
data$Inflation_Rate = as.numeric(data$Inflation_Rate)
data$Interest_Rates = as.numeric(data$Interest_Rates)
data$Oil._Price = as.numeric(data$Oil._Price)
data$Per_Capita_Income = as.numeric(data$Per_Capita_Income)
data$Population = as.numeric(data$Population)
data$S_P_500 = as.numeric(data$S_P_500)
data$Unemployment_Rate = as.numeric(data$Unemployment_Rate)

head(data,10)
```

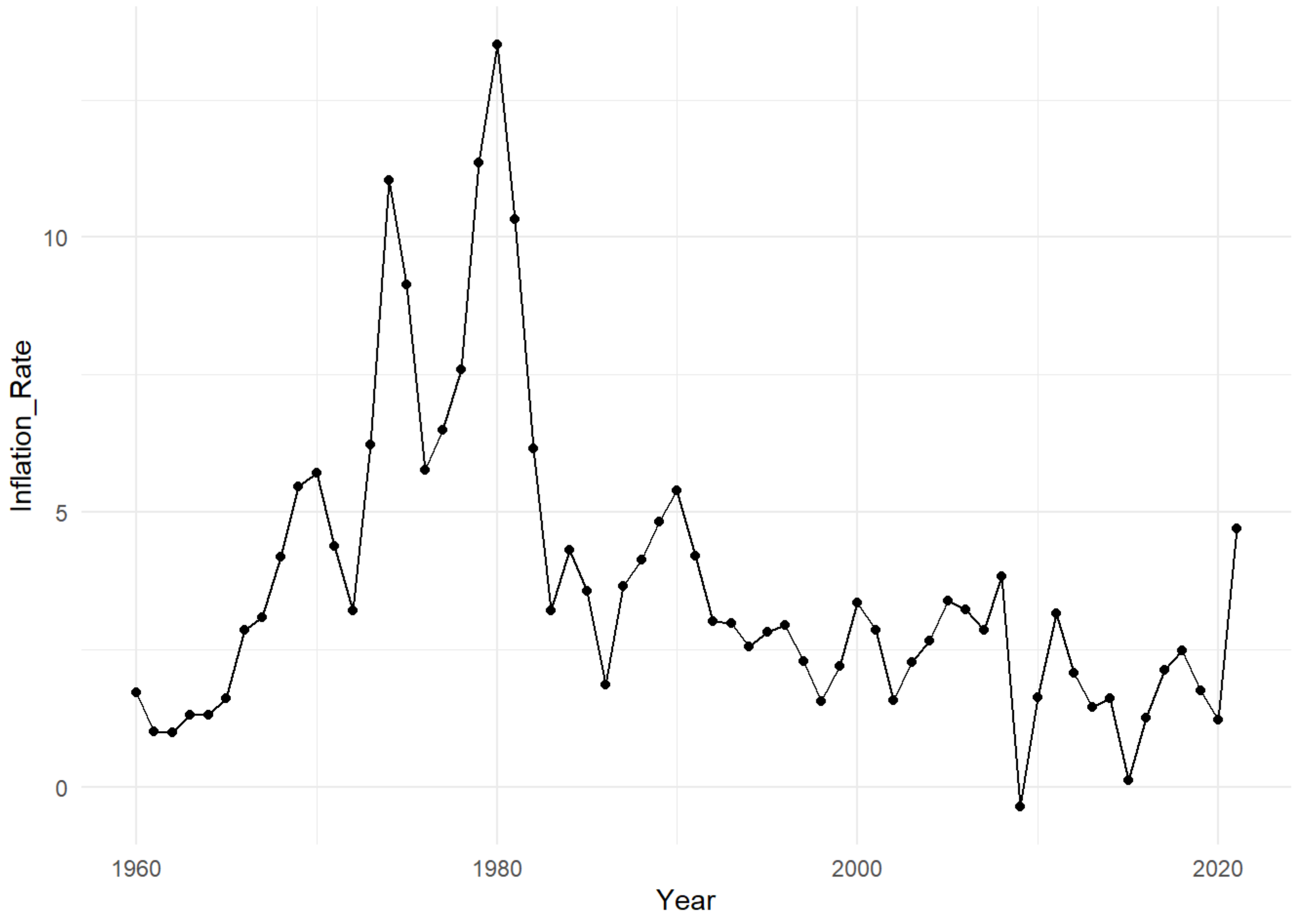
```
##    Year    GDP Inflation_Rate Interest_Rates Oil._Price Per_Capita_Income
## 1  1960  543.3           1.72           3.21        1.9             3.007
## 2  1961  563.3           1.01           1.95        1.8             3.067
## 3  1962  605.1           1.00           2.71        1.8             3.244
## 4  1963  638.6           1.32           3.18        1.8             3.375
## 5  1964  685.8           1.31           3.50        1.8             3.574
## 6  1965  743.7           1.61           4.08        1.8             3.828
## 7  1966  815.0           2.86           5.11        1.8             4.146
## 8  1967  861.7           3.09           4.22        1.8             4.336
## 9  1968  942.5           4.19           5.66        1.8             4.696
## 10 1969 1019.9           5.46           8.21        1.8             5.032
##    Population S_P_500 Unemployment_Rate
## 1   176188578   55.85               6.6
## 2   179087278   66.27               6.0
## 3   181917809   62.32               5.5
## 4   184649873   69.86               5.5
## 5   187277378   81.37               5.0
## 6   189703283   88.16               4.0
## 7   191830975   85.18               3.8
## 8   193782438   91.96               3.8
## 9   195743427   98.38               3.4
## 10  197859329   97.77               3.5
```

Analyze

#After making the data available in R, it was time to analyze them. First of all, we started to create graphics in order to examine the data visually.

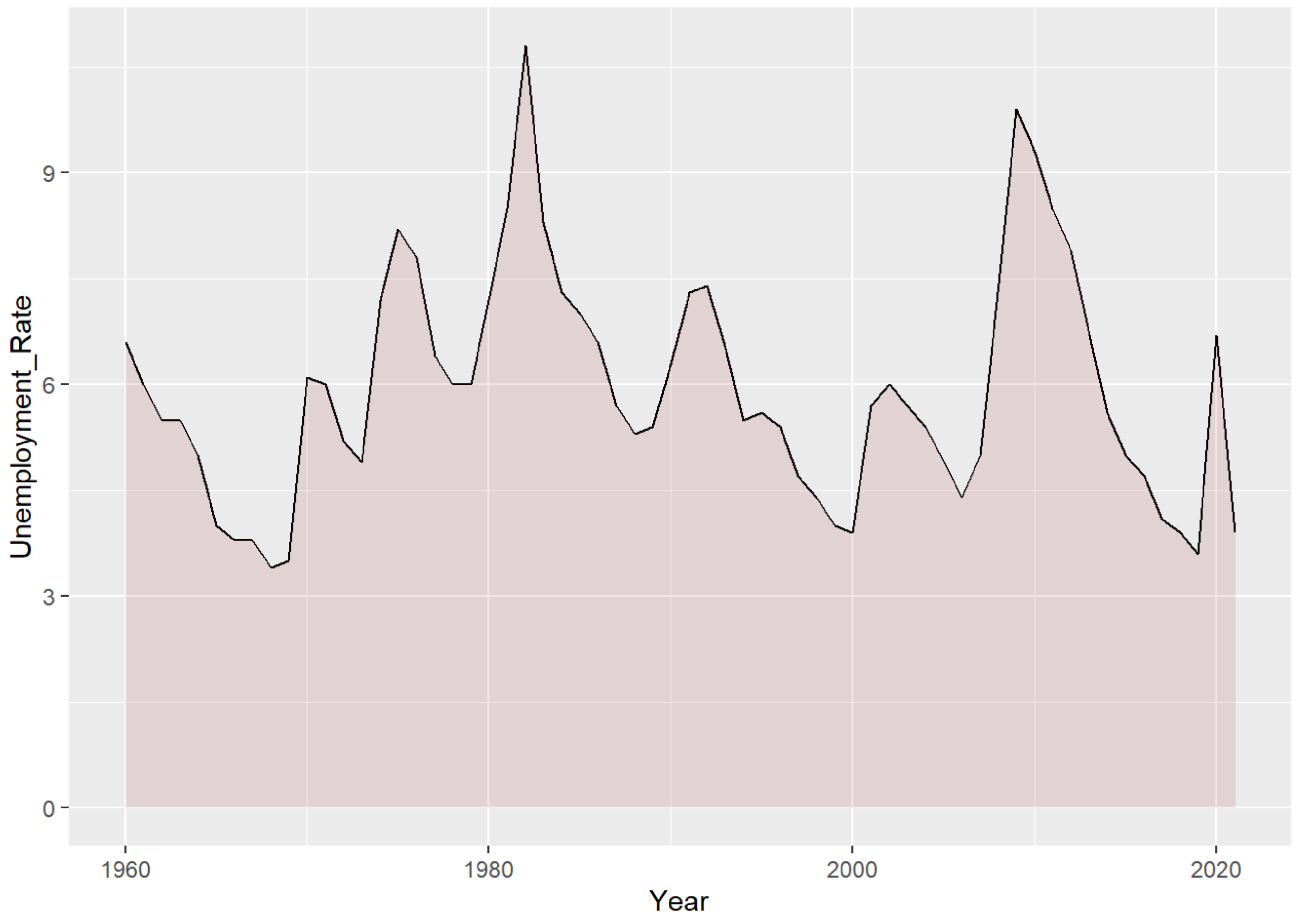
#Inflation

```
library(ggplot2)
ggplot(data,aes(x=Year,y=Inflation_Rate)) +
  geom_line()+
  geom_point()+
  theme_minimal()
```



#Unemployment

```
ggplot(data = data, aes(x = Year,y=Unemployment_Rate)) +
  geom_line() +
  geom_area(fill = 'darkred',alpha=0.1)
```

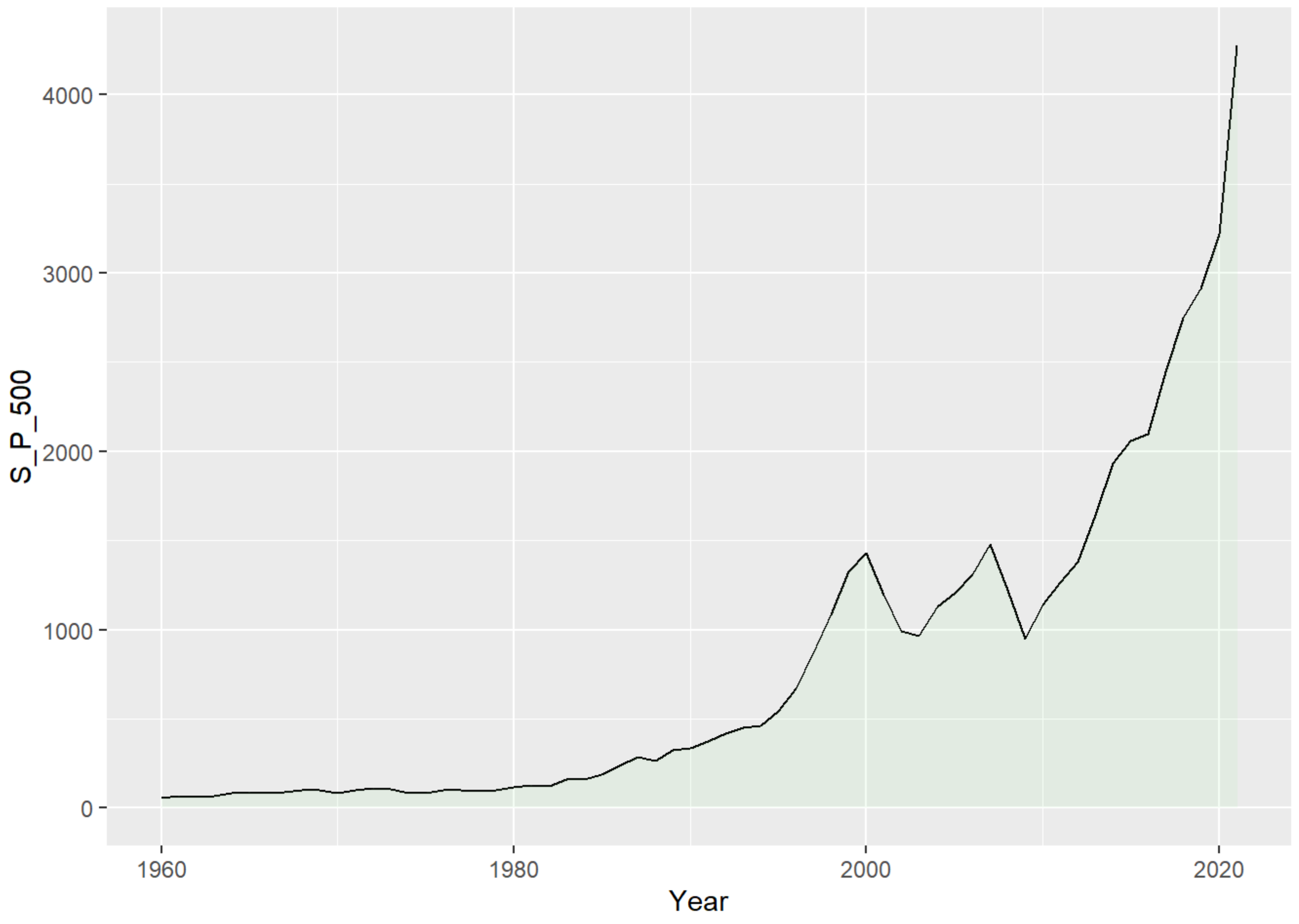


```
  labs(x = "Per Year", y = "Unemployment") +
  theme_minimal()
```

```
## NULL
```

#S&P500

```
ggplot(data = data, aes(x = Year,y=S_P_500)) +
  geom_line() +
  geom_area(fill = 'lightgreen',alpha=0.1)
```

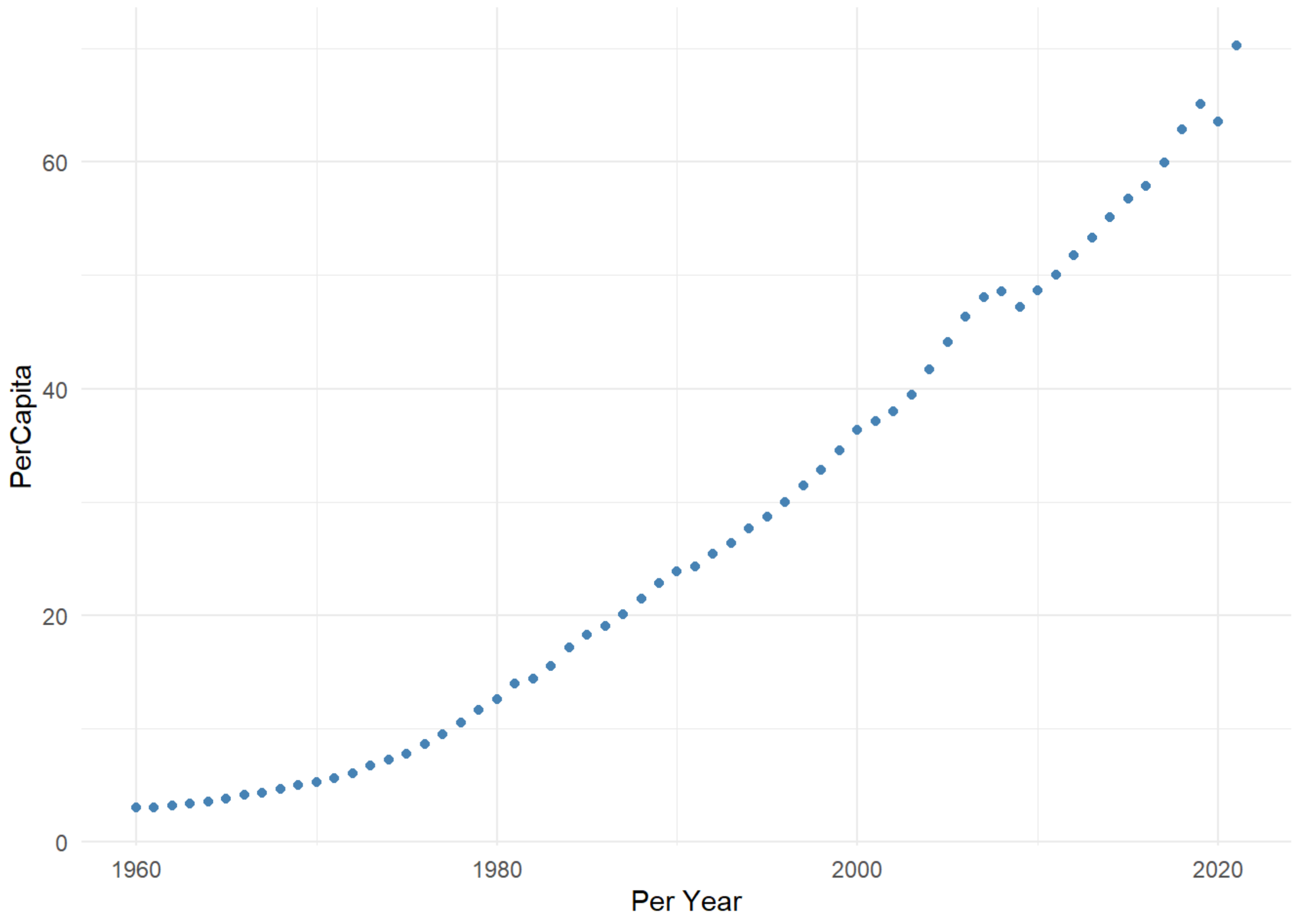


```
  labs(x = "Per Year", y = "Unemployment") +
  theme_minimal()
```

```
## NULL
```

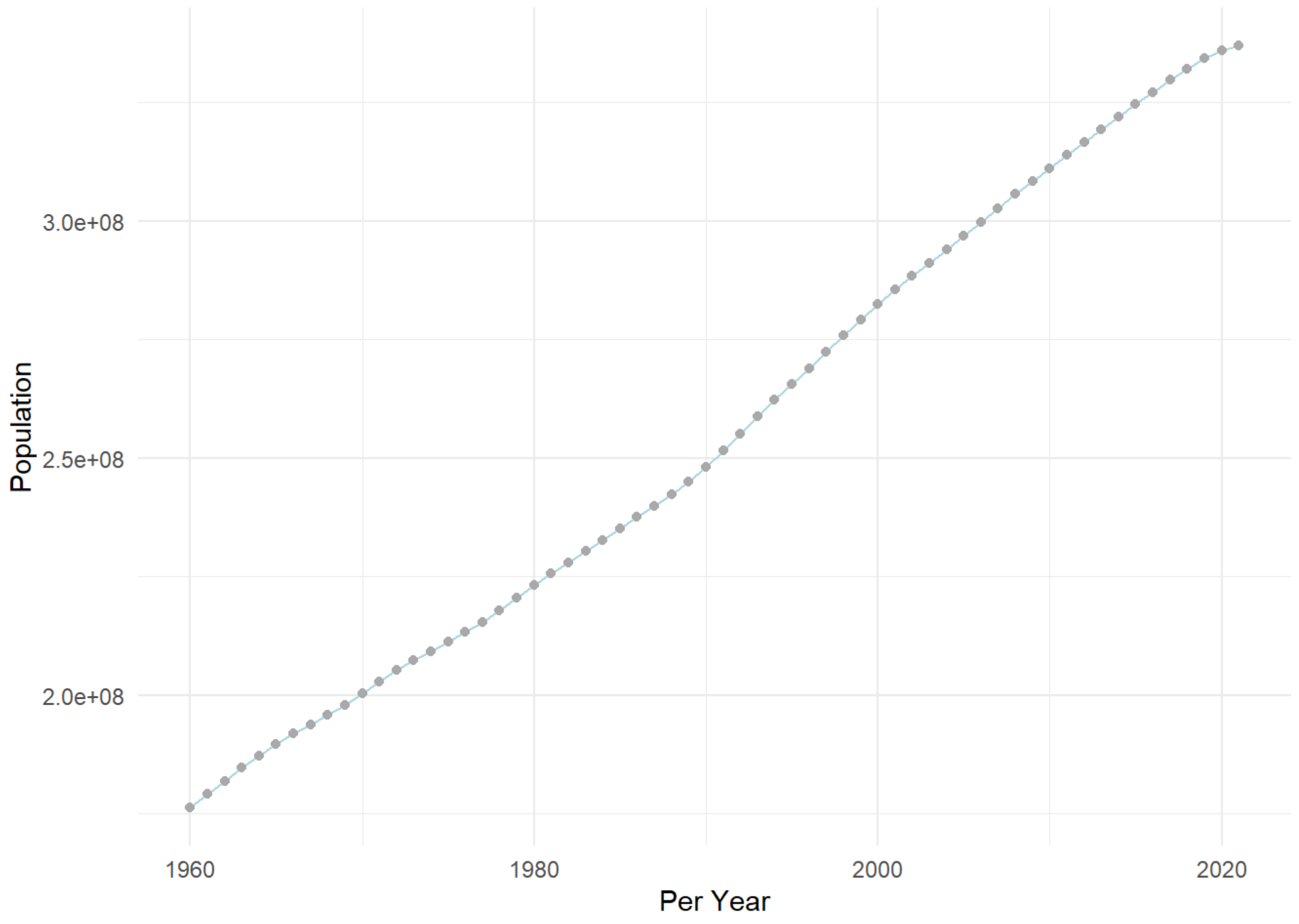
#Per Capita Income

```
ggplot(data = data, aes(x = Year,y=Per_Capita_Income)) +
  geom_point(color='steelblue') +
  labs(x = "Per Year", y = "PerCapita") +
  theme_minimal()
```



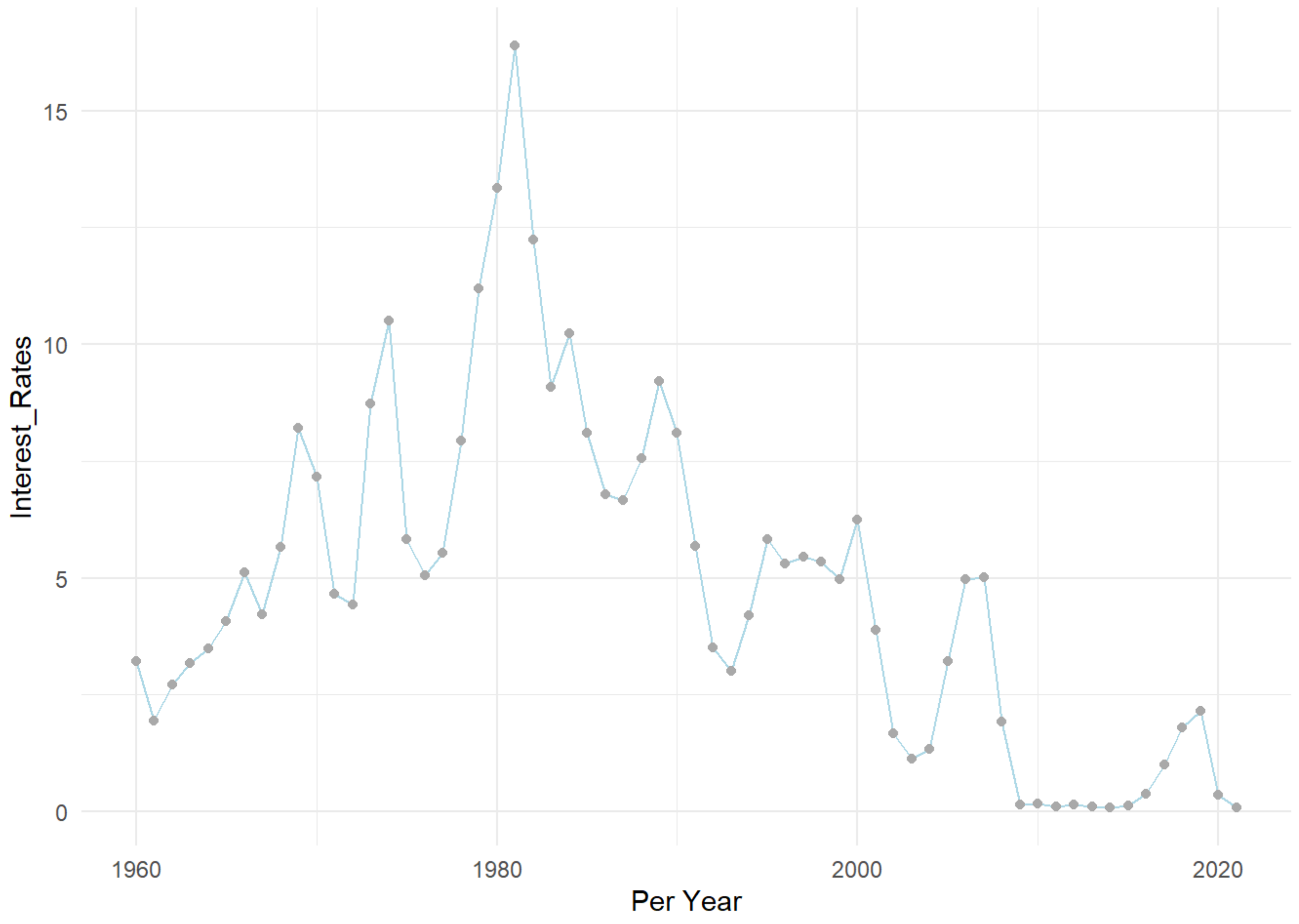
#Population

```
ggplot(data = data, aes(x = Year,y=Population)) +
  geom_line(color='lightblue')+
  geom_point(color='darkgray') +
  labs(x = "Per Year", y = "Population") +
  theme_minimal()
```



#Interest Rate

```
ggplot(data = data, aes(x = Year,y=Interest_Rates)) +
  geom_line(color='lightblue')+
  geom_point(color='darkgray') +
  labs(x = "Per Year", y = "Interest_Rates") +
  theme_minimal()
```



Correlation Matrix

#After the graphs, we tried to find out which data we could establish a relationship between, and for this we decided to use a correlation matrix. The reason for this is that the pairs with the closest results to +1 in the results we will get as a result of this analysis showed us a way.

However, since the range of data we have is very wide at first, we decided to search for the relationship between the data between 1970 and 2000, based on the graphs we have just shown.

As a result, the data pairs we have are respectively

1- Inflation Rate - Unemployment Rate

2- Interest Rate - Inflation Rate.

```r
library(dplyr)
```

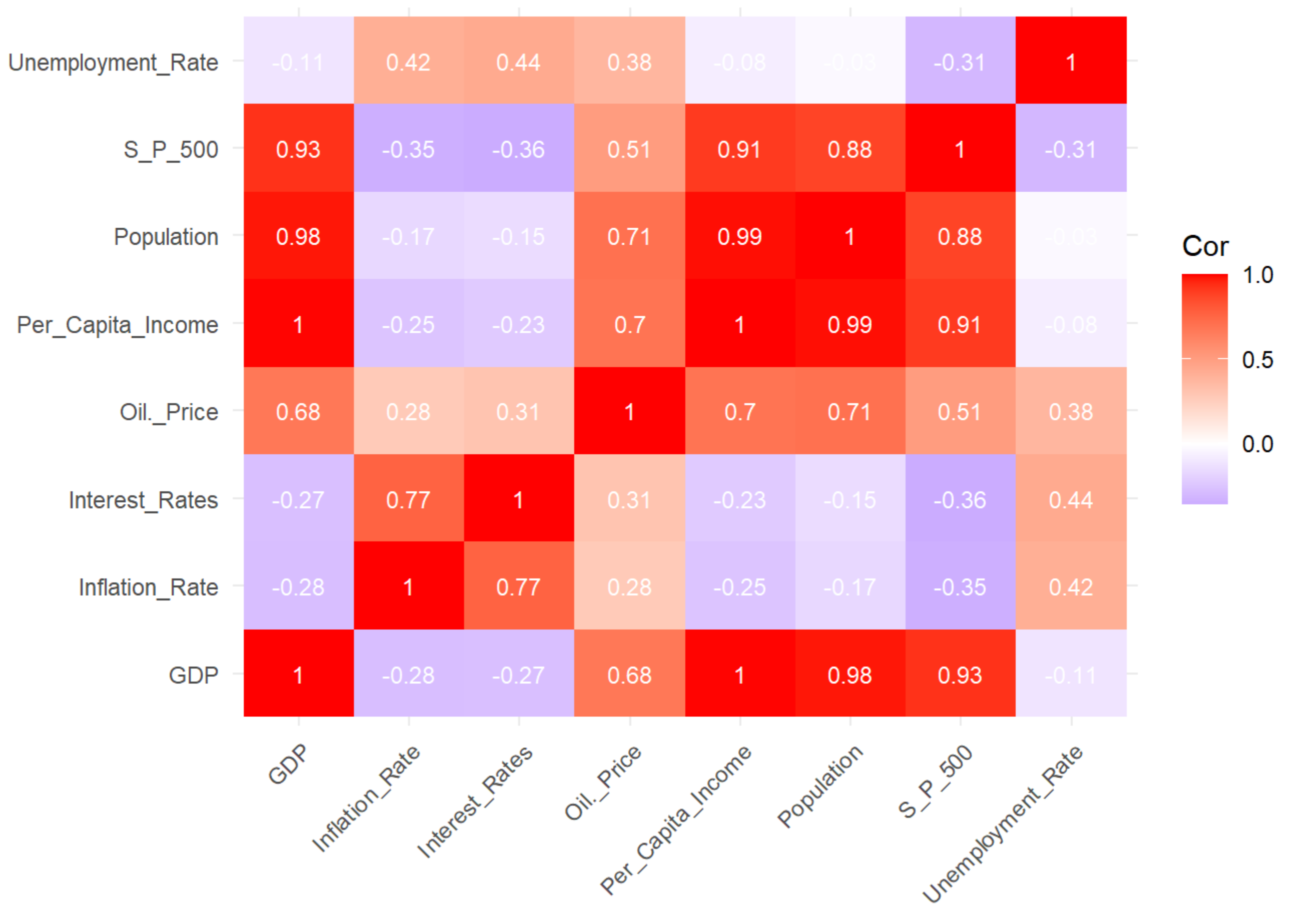
```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

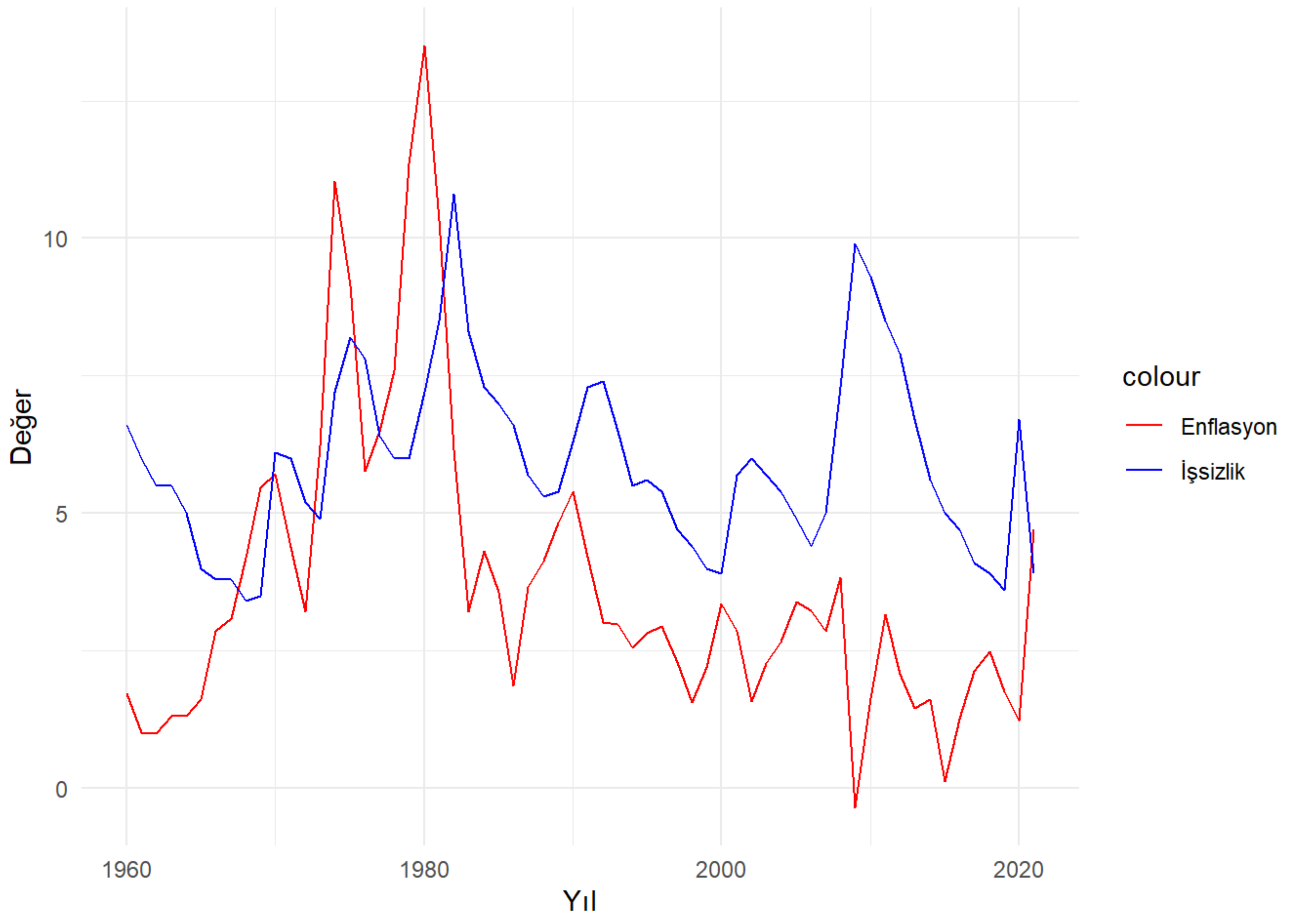
```
filtered_data=data %>% filter(Year >= 1960, Year <= 2005)
cor_matrix <- cor(filtered_data[,-1])
cor_melted <- as.data.frame(as.table(cor_matrix))
names(cor_melted) <- c('Var1','Var2','Cor')

ggplot(data = cor_melted, aes(x = Var1, y = Var2, fill = Cor)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0) +
  geom_text(aes(label=round(Cor,2)),color="white",size=3)+
  labs(x = "", y = "") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



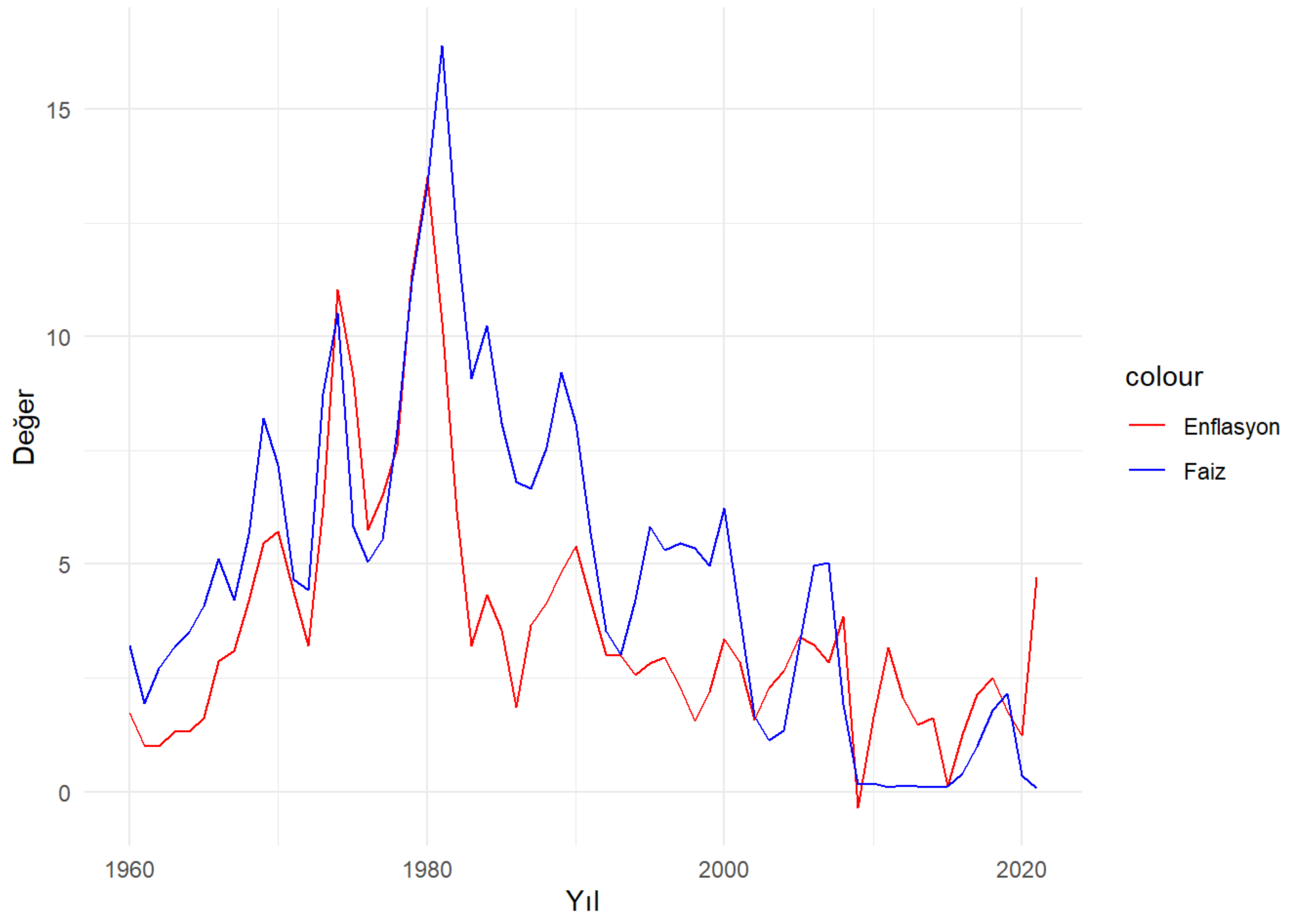
#After the correlation matrix analysis, we produced the binary graphs of the selected data. Here you see the graph of inflation and unemployment.

```
ggplot(data = data) +
  geom_line(aes(x = Year, y = Inflation_Rate, color = "Enflasyon")) +
  geom_line(aes(x = Year, y = Unemployment_Rate, color = "İşsizlik")) +
  labs(x = "Yıl", y = "Değer") +
  scale_color_manual(values = c("Enflasyon" = "red", "İşsizlik" = "blue")) +
  theme_minimal()
```



Here you see the graph of inflation and interest.

```
ggplot(data = data) +
  geom_line(aes(x = Year, y = Inflation_Rate, color = "Enflasyon")) +
  geom_line(aes(x = Year, y = Interest_Rates, color = "Faiz")) +
  labs(x = "Yıl", y = "Değer") +
  scale_color_manual(values = c("Enflasyon" = "red", "Faiz" = "blue")) +
  theme_minimal()
```



Model

#After arranging our data, it was time to create a model, first we determined the train data and then we tested these data.

1) Inflation - Unemployment

> Datasets
>
> The data set we use here can be seen, after training until 1970-1990, we estimated the period between 1990-2000.

```
train_data = data %>% filter(Year >= 1960, Year <= 2005)
test_data = data %>% filter(Year > 2005, Year <= 2020)
```

> Random Forest:
>
> Random Forest is a commonly-used machine learning algorithm which combines the output of multiple decision trees to reach a single result. Its ease of use and flexibility have fueled its adoption, as it handles both classification and regression problems.
>
> #The first method we used was random forest. This is because random forest can measure the importance of each variable and combines multiple decision trees to form a prediction model.

```
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
```

```
rf_model = randomForest(Inflation_Rate ~ Unemployment_Rate, data=train_data)
summary(rf_model)
```

```
##                   Length Class  Mode
## call                3    -none- call
## type                1    -none- character
## predicted          46    -none- numeric
## mse               500    -none- numeric
## rsq               500    -none- numeric
## oob.times          46    -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                  46    -none- numeric
## test                0    -none- NULL
## inbag               0    -none- NULL
## terms               3    terms  call
```

Linear Regression

Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.

The second method we used was Linear regression, because we thought it would help us to understand the relationship between variables.

```
lr_model <- lm(Inflation_Rate ~ Unemployment_Rate, data=train_data)
summary(lr_model)
```

```
##
## Call:
## lm(formula = Inflation_Rate ~ Unemployment_Rate, data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3302 -1.7177 -0.6994  0.9308  8.1807
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -0.5555     1.6346  -0.340    0.736
## Unemployment_Rate   0.8160     0.2686   3.038    0.004 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.671 on 44 degrees of freedom
## Multiple R-squared:  0.1734, Adjusted R-squared:  0.1546
## F-statistic: 9.228 on 1 and 44 DF,  p-value: 0.003998
```

> #Prediction Dataframe ## DEGISTI
>
> After training the data, it was time to predict.

```
results_df <- data.frame(
  Year =numeric(),
  Actual_Unemployment = numeric(),
  Regression_Predict = numeric(),
  RandomForest_Predict= numeric()
)

pre_rf = predict(rf_model, newdata = data.frame(Unemployment_Rate=test_data$Unemployment_Rat
e))
pre_lr = predict(lr_model, newdata = data.frame(Unemployment_Rate=test_data$Unemployment_Rat
e))

result = data.frame(Year=test_data$Year,Actual_Inflation_Rate=test_data$Inflation_Rate,RF_Pre
dict_Rate=pre_rf,LR_Predict_Rate=pre_lr)


result$RF_Err = result$RF_Predict_Rate - result$Actual_Inflation_Rate
result$LR_Err = result$LR_Predict_Rate - result$Actual_Inflation_Rate


head(result,10)
```

```
##    Year Actual_Inflation_Rate RF_Predict_Rate LR_Predict_Rate     RF_Err
## 1  2006                  3.23        2.038819        3.034660 -1.1911805
## 2  2007                  2.85        2.607488        3.524233 -0.2425124
## 3  2008                  3.84        5.322238        5.400928  1.4822377
## 4  2009                 -0.36        7.441127        7.522410  7.8011267
## 5  2010                  1.64        7.575990        7.032837  5.9359897
## 6  2011                  3.16        7.572266        6.380073  4.4122657
## 7  2012                  2.07        5.865096        5.890501  3.7950960
## 8  2013                  1.46        2.618222        4.911355  1.1582217
## 9  2014                  1.62        2.411867        4.013805  0.7918673
## 10 2015                  0.12        2.607488        3.524233  2.4874876
##        LR_Err
## 1  -0.1953402
## 2   0.6742325
## 3   1.5609279
## 4   7.8824096
## 5   5.3928369
## 6   3.2200733
## 7   3.8205006
## 8   3.4513552
## 9   2.3938052
## 10  3.4042325
```

Mean Squared Error

After analyzing the predictions , we calculated the mean squared error to determine which estimation is more consistent, and we realized that Linear Regression is more useful for us.

```
rf_mse_1 = mean(result$RF_Err^2)
cat("Random Forest MSE", rf_mse_1, "\n")
```

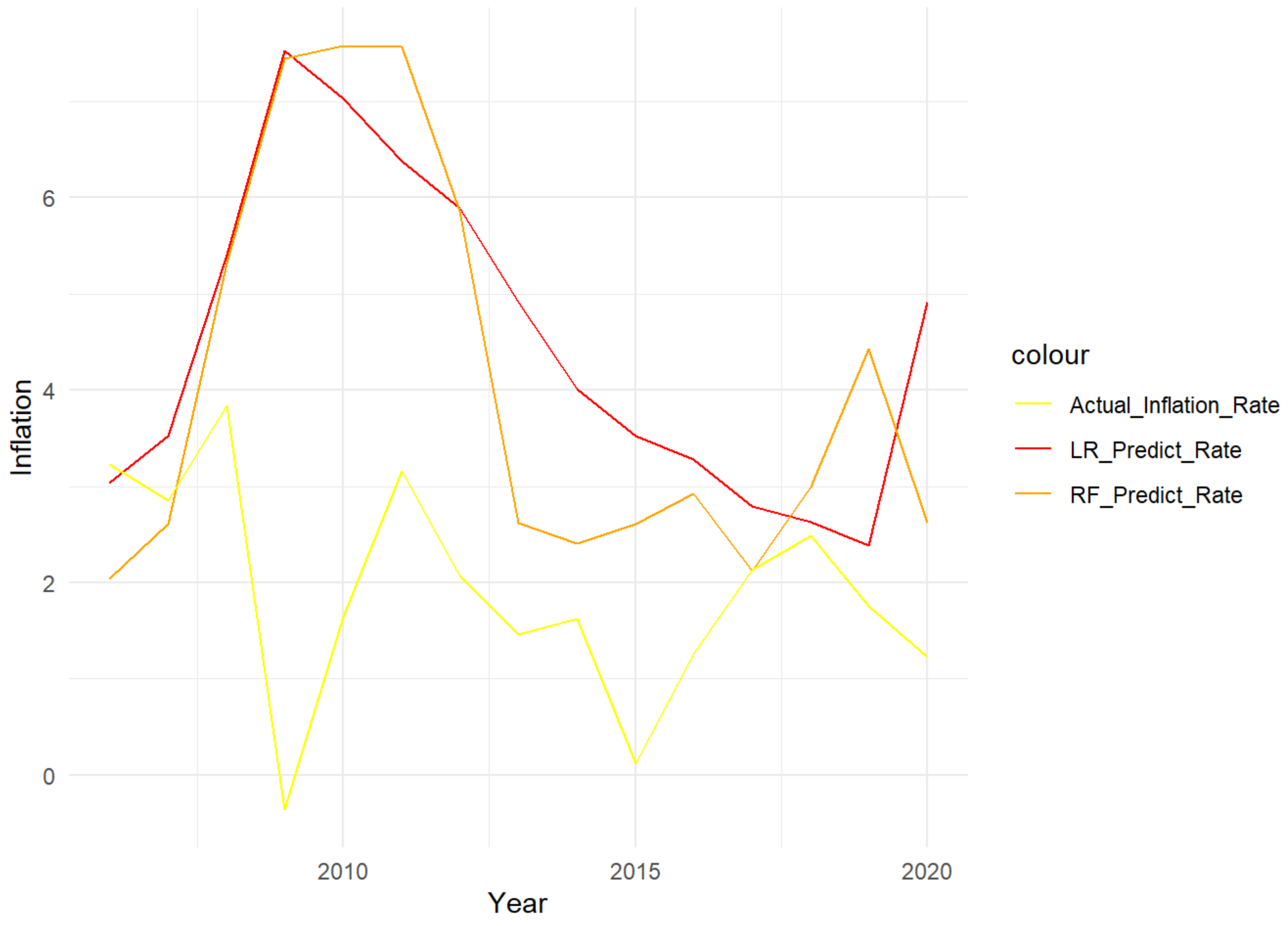
```
## Random Forest MSE 10.26008
```

```
lr_mse_1 = mean(result$LR_Err^2)
cat("Linear Regression MSE", lr_mse_1, "\n")
```

```
## Linear Regression MSE 11.12079
```

Here you can see the comparison graph made by the models with the estimated real unemployment.

```
ggplot(data = result) +
  geom_line(aes(x = Year, y = LR_Predict_Rate, color = "LR_Predict_Rate")) +
  geom_line(aes(x = Year, y = RF_Predict_Rate, color = "RF_Predict_Rate")) +
  geom_line(aes(x = Year, y = Actual_Inflation_Rate, color = "Actual_Inflation_Rate")) +
  labs(x = "Year", y = "Inflation") +
  scale_color_manual(values = c("LR_Predict_Rate" = "red", "RF_Predict_Rate"="orange","Actual
_Inflation_Rate" = "yellow")) +
  theme_minimal()
```



2) Inflation Rate - Interest Rate

This time we trained the data from 1985 to 2000. We tested from 2001 to 2005

We just decided to reuse linear regression because you thought it was more useful.

```
lr_model <- lm(Inflation_Rate ~ Interest_Rates, data=na.omit(train_data))
summary(lr_model)
```

```
##
## Call:
## lm(formula = Inflation_Rate ~ Interest_Rates, data = na.omit(train_data))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0989 -1.1437 -0.3134  0.8280  5.0722
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.05131    0.59497   0.086    0.932
## Interest_Rates  0.68841    0.08609   7.996 4.08e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.875 on 44 degrees of freedom
## Multiple R-squared:  0.5924, Adjusted R-squared:  0.5831
## F-statistic: 63.94 on 1 and 44 DF,  p-value: 4.082e-10
```

Support Vector Machine $$$

#The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.

#We wanted to use a support vector machine instead of a random forest because we thought it had good generalization capacity and was resistant to outliers.

```
rf_model <- randomForest(Inflation_Rate ~ Interest_Rates, data=train_data)
summary(rf_model)
```

```
##                 Length Class  Mode
## call              3    -none- call
## type              1    -none- character
## predicted        46    -none- numeric
## mse             500    -none- numeric
## rsq             500    -none- numeric
## oob.times        46    -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                46    -none- numeric
## test              0    -none- NULL
## inbag             0    -none- NULL
## terms             3    terms  call
```

After training the models, it’s time to test them. First we tested it with linear regression, then we tested it using the support vector machine.

```
results_df <- data.frame(
  Year =numeric(),
  Actual_Inflation_Rate = numeric(),
  Regression_Predict = numeric(),
  RF_Predict= numeric()
)

pre_lr = predict(lr_model, newdata = data.frame(Interest_Rates=test_data$Interest_Rates))
pre_rf = predict(rf_model, newdata = data.frame(Interest_Rates=test_data$Interest_Rates))

result = data.frame(Year=test_data$Year,Actual_Inflation_Rate=test_data$Inflation_Rate,LR_Pre
dict_Rate=pre_lr,RF_Predict_Rate=pre_rf)


result$LR_Err = result$LR_Predict_Rate - result$Actual_Inflation_Rate
result$RF_Err = result$RF_Predict_Rate - result$Actual_Inflation_Rate


head(result,10)
```

```
##    Year Actual_Inflation_Rate LR_Predict_Rate RF_Predict_Rate      LR_Err
## 1  2006                  3.23       3.4726936        3.132133  0.24269365
## 2  2007                  2.85       3.5071140        4.072742  0.65711401
## 3  2008                  3.84       1.3730515        1.484437 -2.46694851
## 4  2009                 -0.36       0.1614547        2.180642  0.52145470
## 5  2010                  1.64       0.1752228        2.180642 -1.46477715
## 6  2011                  3.16       0.1201503        2.180642 -3.03984973
## 7  2012                  2.07       0.1476866        2.180642 -1.92231344
## 8  2013                  1.46       0.1270343        2.180642 -1.33296566
## 9  2014                  1.62       0.1132662        2.180642 -1.50673381
## 10 2015                  0.12       0.1408025        2.180642  0.02080249
##         RF_Err
## 1  -0.09786733
## 2   1.22274200
## 3  -2.35556333
## 4   2.54064233
## 5   0.54064233
## 6  -0.97935767
## 7   0.11064233
## 8   0.72064233
## 9   0.56064233
## 10  2.06064233
```

After testing, we calculated mean squared errors to decide which of the two models worked better and decided that linear regression was better.

```
lr_mse_2 = mean(result$LR_Err^2)
cat("Linear Regression MSE", lr_mse_2, "\n")
```

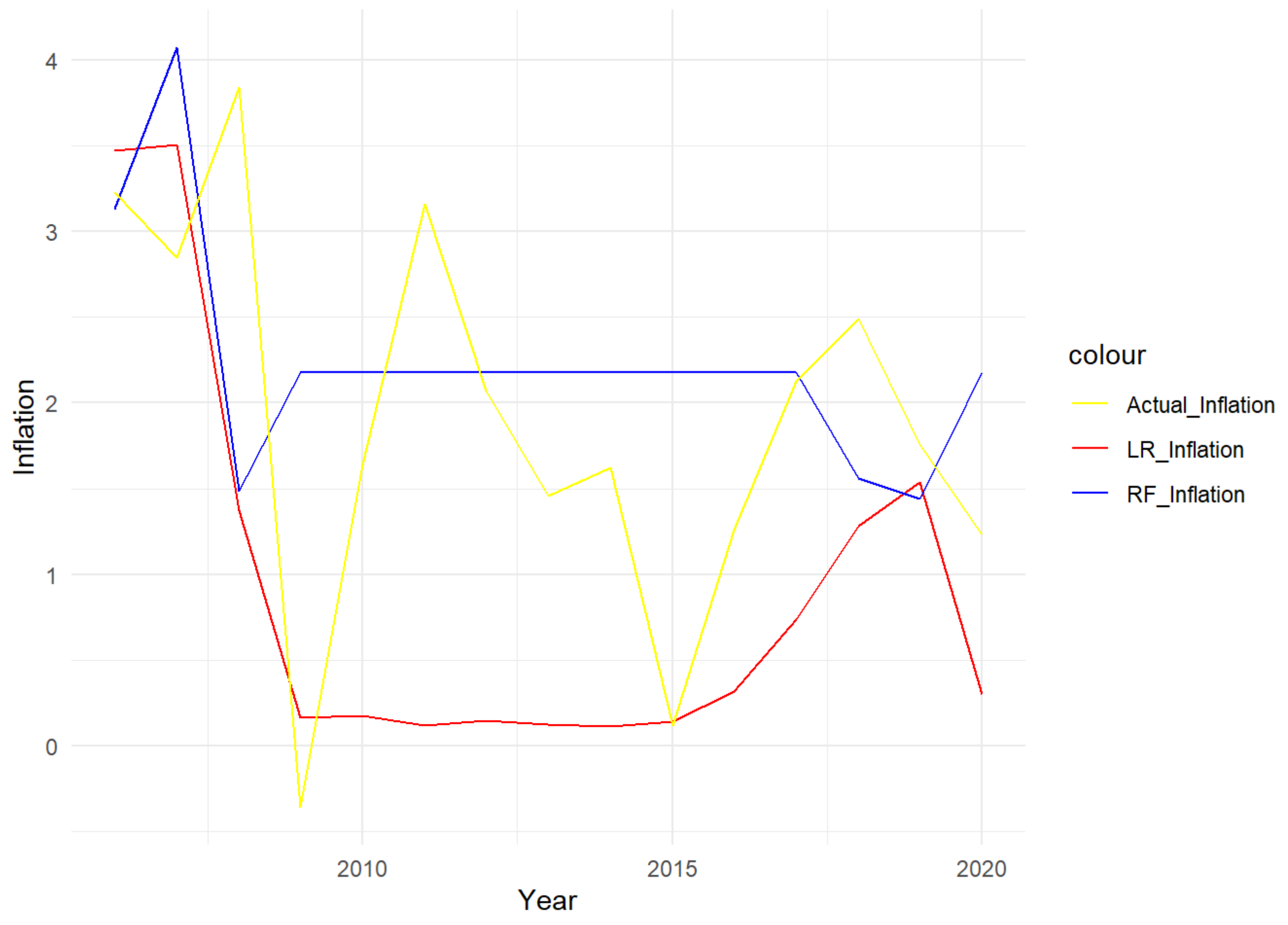
```
## Linear Regression MSE 2.077702
```

```
rf_mse_2 = mean(result$RF_Err^2)
cat("RF MSE", rf_mse_2, "\n")
```

```
## RF MSE 1.505035
```

We re-graphed our predictions.

```
ggplot(data = result) +
  geom_line(aes(x = Year, y = LR_Predict_Rate, color = "LR_Inflation")) +
  geom_line(aes(x = Year, y = RF_Predict_Rate, color = "RF_Inflation")) +
  geom_line(aes(x = Year, y = Actual_Inflation_Rate, color = "Actual_Inflation")) +
  labs(x = "Year", y = "Inflation") +
  scale_color_manual(values = c("LR_Inflation" = "red", "RF_Inflation"="blue","Actual_Inflati
on" = "yellow")) +
  theme_minimal()
```



3) 4lü

```
lr_model <- lm(Inflation_Rate ~ Unemployment_Rate + S_P_500 + Oil._Price + Interest_Rates, da
ta=train_data)
summary(lr_model)
```

```
##
## Call:
## lm(formula = Inflation_Rate ~ Unemployment_Rate + S_P_500 + Oil._Price +
##     Interest_Rates, data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6330 -1.0397  0.0354  0.7510  4.8403
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.712456   1.650902   0.432    0.668
## Unemployment_Rate -0.012169   0.263399  -0.046    0.963
## S_P_500           -0.001583   0.001224  -1.293    0.203
## Oil._Price         0.045878   0.039874   1.151    0.257
## Interest_Rates     0.564329   0.117066   4.821 1.99e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.886 on 41 degrees of freedom
## Multiple R-squared:  0.6158, Adjusted R-squared:  0.5783
## F-statistic: 16.43 on 4 and 41 DF,  p-value: 4.152e-08
```

```
rf_model <- randomForest(Inflation_Rate ~ Unemployment_Rate + S_P_500 + Oil._Price + Interest
_Rates, data=train_data)
summary(rf_model)
```

```
##                 Length Class  Mode
## call              3    -none- call
## type              1    -none- character
## predicted        46    -none- numeric
## mse             500    -none- numeric
## rsq             500    -none- numeric
## oob.times        46    -none- numeric
## importance        4    -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                46    -none- numeric
## test              0    -none- NULL
## inbag             0    -none- NULL
## terms             3    terms  call
```

```
results_df <- data.frame(
  Year =numeric(),
  Actual_Inflation_Rate = numeric(),
  Regression_Predict = numeric(),
  RF_Predict= numeric()
)

pre_lr = predict(lr_model, newdata = data.frame(Unemployment_Rate=test_data$Unemployment_Rat
e,S_P_500=test_data$S_P_500,Oil._Price=test_data$Oil._Price,Interest_Rates=test_data$Interest
_Rates))

pre_rf = predict(rf_model, newdata = data.frame(Unemployment_Rate=test_data$Unemployment_Rat
e,S_P_500=test_data$S_P_500,Oil._Price=test_data$Oil._Price,Interest_Rates=test_data$Interest
_Rates))

result = data.frame(Year=test_data$Year,Actual_Inflation_Rate=test_data$Inflation_Rate,LR_Pre
dict_Rate=pre_lr,RF_Predict_Rate=pre_rf)


result$LR_Err = result$LR_Predict_Rate - result$Actual_Inflation_Rate
result$RF_Err = result$RF_Predict_Rate - result$Actual_Inflation_Rate


head(result,10)
```

```
##    Year Actual_Inflation_Rate LR_Predict_Rate RF_Predict_Rate     LR_Err
## 1  2006                  3.23       4.3782606        3.520679  1.1482606
## 2  2007                  2.85       4.4679488        3.730656  1.6179488
## 3  2008                  3.84       4.2384794        4.613287  0.3984794
## 4  2009                 -0.36       2.0112484        4.880380  2.3712484
## 5  2010                  1.64       2.5441230        4.905136  0.9041230
## 6  2011                  3.16       3.7637658        4.938319  0.6037658
## 7  2012                  2.07       3.6352502        4.766719  1.5652502
## 8  2013                  1.46       3.0767295        4.077863  1.6167295
## 9  2014                  1.62       2.1791507        3.552437  0.5591507
## 10 2015                  0.12      -0.1364499        3.581379 -0.2564499
##       RF_Err
## 1  0.2906793
## 2  0.8806560
## 3  0.7732867
## 4  5.2403797
## 5  3.2651363
## 6  1.7783193
## 7  2.6967190
## 8  2.6178627
## 9  1.9324369
## 10 3.4613792
```

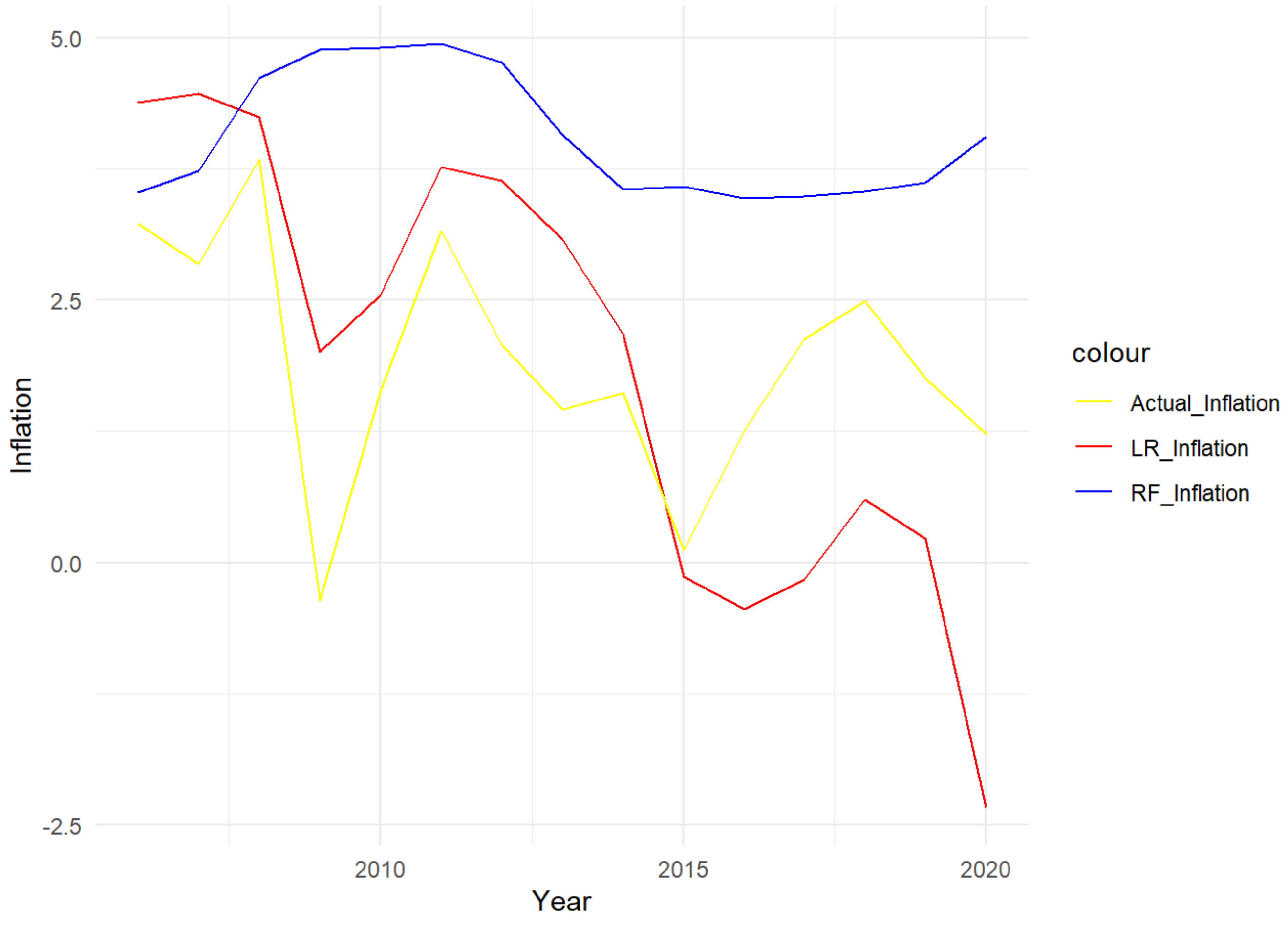
```
lr_mse_3 = mean(result$LR_Err^2)
cat("Linear Regression MSE", lr_mse_3, "\n")
```

```
## Linear Regression MSE 2.871658
```

```
rf_mse_3 = mean(result$RF_Err^2)
cat("RF MSE", rf_mse_3, "\n")
```

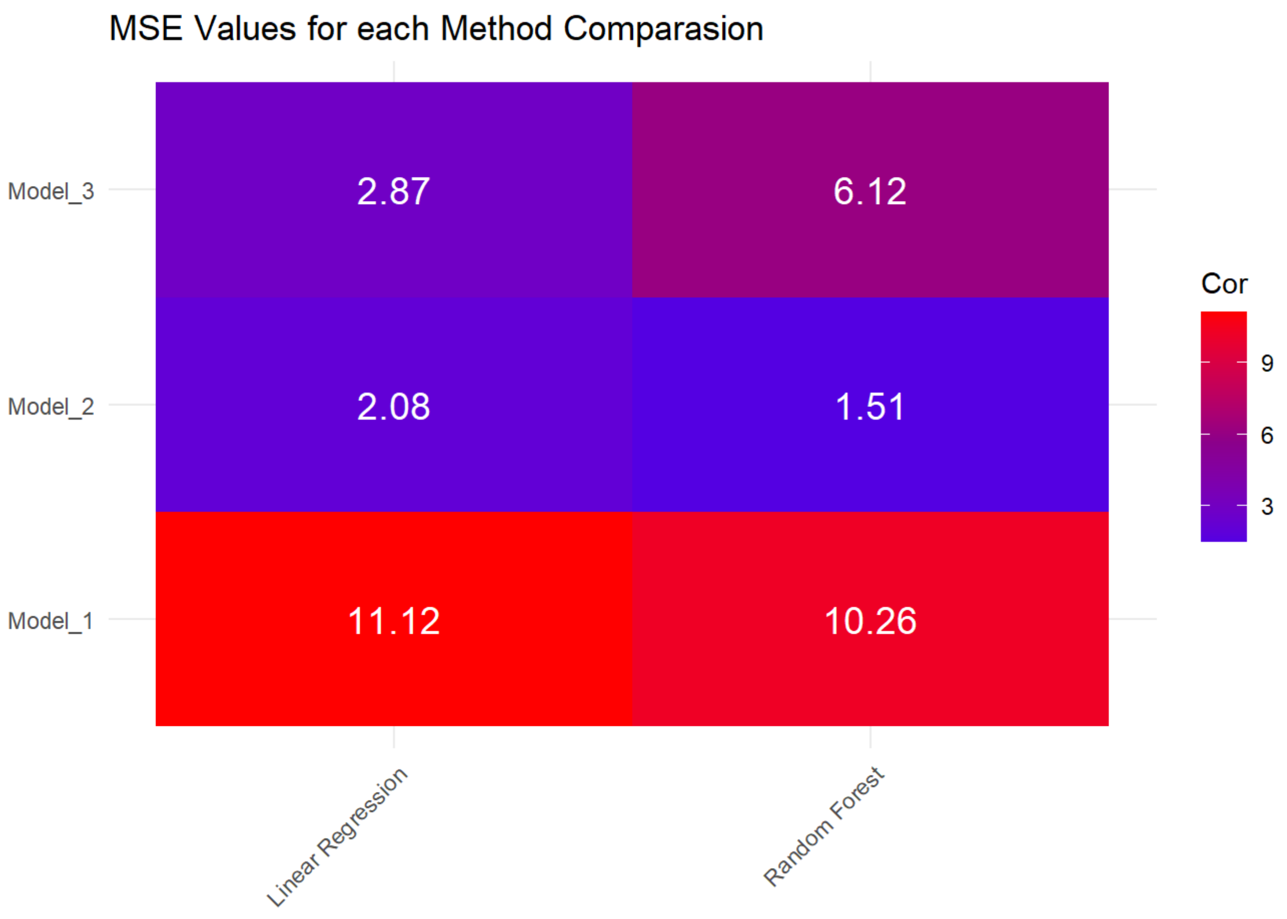
```
## RF MSE 6.122257
```

```
ggplot(data = result) +
  geom_line(aes(x = Year, y = LR_Predict_Rate, color = "LR_Inflation")) +
  geom_line(aes(x = Year, y = RF_Predict_Rate, color = "RF_Inflation")) +
  geom_line(aes(x = Year, y = Actual_Inflation_Rate, color = "Actual_Inflation")) +
  labs(x = "Year", y = "Inflation") +
  scale_color_manual(values = c("LR_Inflation" = "red", "RF_Inflation"="blue","Actual_Inflati
on" = "yellow")) +
  theme_minimal()
```



Sum

```
my_table <- data.frame(
  Var1 = c("Random Forest", "Random Forest", "Random Forest","Linear Regression", "Linear Reg
ression", "Linear Regression"),
  Var2 = c("Model_3", "Model_2", "Model_1","Model_3", "Model_2", "Model_1"),
  Cor = c(rf_mse_3, rf_mse_2, rf_mse_1,lr_mse_3,lr_mse_2,lr_mse_1)
)
mp = mean(my_table$Cor)
ggplot(data = my_table, aes(x = Var1, y = Var2, fill = Cor)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", high = "red", mid = "darkmagenta", midpoint = mp) +
  geom_text(aes(label=round(Cor,2)),color="white",size=5)+
  labs(x = "", y = "",title = "MSE Values for each Method Comparasion") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



```
me_rf=mean(my_table$Cor[1:3])
cat("In these 3 model comparisons, the average MSE value of Random Forest: \n", me_rf, "\n")
```

```
## In these 3 model comparisons, the average MSE value of Random Forest:
##  5.962457
```

```
me_lr=mean(my_table$Cor[4:6])
cat("In these 3 model comparisons, the average MSE value of Linear Regression: \n", me_lr,
"\n")
```

```
## In these 3 model comparisons, the average MSE value of Linear Regression:
##  5.356715
```