Final Project v14

2024-12-10

```
CPI <- read.csv("C:/Users/liva0287/Downloads/Data Final Project/CPI.csv")
head(CPI)
```

```
##         DATE CPALTT01USM657N
## 1 1955-02-01       0.0000000
## 2 1955-03-01       0.0000000
## 3 1955-04-01       0.0000000
## 4 1955-05-01       0.0000000
## 5 1955-06-01       0.0000000
## 6 1955-07-01       0.3745318
```

```
Unemployment_Rate <- read.csv("C:/Users/liva0287/Downloads/Data Final Project/Unemployment_Rate.csv")
head(Unemployment_Rate)
```

```
##         DATE UNRATE
## 1 1955-02-01    4.7
## 2 1955-03-01    4.6
## 3 1955-04-01    4.7
## 4 1955-05-01    4.3
## 5 1955-06-01    4.2
## 6 1955-07-01    4.0
```

```
M2SL <- read.csv("C:/Users/liva0287/Downloads/Data Final Project/M2SL.csv")
head(M2SL)
```

```
##         DATE  M2SL
## 1 1959-01-01 286.6
## 2 1959-02-01 287.7
## 3 1959-03-01 289.2
## 4 1959-04-01 290.1
## 5 1959-05-01 292.2
## 6 1959-06-01 294.1
```

```
Recession <- read.csv("C:/Users/liva0287/Downloads/Data Final Project/Recession.csv")
head(Recession)
```

```
##         DATE USREC
## 1 1955-02-01     0
## 2 1955-03-01     0
## 3 1955-04-01     0
## 4 1955-05-01     0
## 5 1955-06-01     0
## 6 1955-07-01     0
```

```
Housing_Starts <- read.csv("C:/Users/liva0287/Downloads/Data Final Project/Housing_Starts.csv")
head(Housing_Starts)
```

```
##         DATE HOUST
## 1 1959-01-01  1657
## 2 1959-02-01  1667
## 3 1959-03-01  1620
## 4 1959-04-01  1590
## 5 1959-05-01  1498
## 6 1959-06-01  1503
```

```
R_star <- read_excel("C:/Users/liva0287/Downloads/Data Final Project/Neutral_Rate.xlsx", sheet = 'data')
```

```
## New names:
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
## * `` -> `...9`
## * `` -> `...10`
## * `` -> `...11`
```

```
Fed_funds <- read.csv("C:/Users/liva0287/Downloads/Data Final Project/Fed Funds.csv")
head(Fed_funds)
```

```
##         DATE DFF
## 1 1955-12-01 2.5
## 2 1955-12-02 2.5
## 3 1955-12-03 2.5
## 4 1955-12-04 2.5
## 5 1955-12-05 2.5
## 6 1955-12-06 2.5
```

```
`1_Year_inflation_expectations` <- read.csv("C:/Users/liva0287/Downloads/Data Final Project/1_Year_Infla
head(`1_Year_inflation_expectations`)
```

```
##         DATE EXPINF1YR
## 1 1982-01-01  6.394507
## 2 1982-02-01  6.432108
## 3 1982-03-01  6.387732
## 4 1982-04-01  6.140628
## 5 1982-05-01  5.488167
## 6 1982-06-01  5.445233
```

```
SPX <- read.csv("C:/Users/liva0287/Downloads/Data Final Project/SPX.csv")
head(SPX)
```

```
##         Date    SP500 Dividend Earnings Consumer.Price.Index Long.Interest.Rate
## 1 2023-09-01 4515.770     0.00     0.00               306.13                4.09
## 2 2023-08-01 4457.359     0.00     0.00               305.98                4.17
## 3 2023-07-01 4508.076     0.00     0.00               305.69                3.90
## 4 2023-06-01 4345.373    68.71   181.17               305.11                3.75
## 5 2023-05-01 4146.173    68.54   179.17               304.13                3.57
## 6 2023-04-01 4121.467    68.38   177.17               303.36                3.46
##   Real.Price Real.Dividend Real.Earnings  PE10
## 1    4515.77          0.00          0.00 33.25
## 2    4459.48          0.00          0.00 32.91
## 3    4514.51          0.00          0.00 33.38
## 4    4359.88         68.94        181.77 32.41
## 5    4173.45         68.99        180.35 31.14
## 6    4159.03         69.00        178.78 31.15
```

Introduction

The aim of this project is to explore the relationships between macroeconomic indicators, monetary policy, and stock market performance, focusing on the S&P 500 index's year-over-year percentage change (SPX YoY%) as the target variable. Using a combination of economic, financial, and policy-related features, this study attempts to understand the factors that drive fluctuations in SPX YoY% and provide insights into how macroeconomic conditions influence market behavior.

The dataset used in this analysis was sourced from the Federal Reserve Economic Data (FRED) database, a comprehensive resource maintained by the Federal Reserve Bank of St. Louis. FRED provides access to a wide array of economic time-series data, enabling robust analyses of economic trends and their implications.

Goals of the Research

The research is designed with the following objectives:

Understanding Market Drivers: Examine how key macroeconomic indicators, such as CPI changes, unemployment rates, and monetary stance, contribute to the variation in SPX YoY%. By identifying significant predictors, the study aims to highlight the most influential factors affecting stock market returns.

Analyzing Economic Cycles: Investigate the relationship between periods of economic recession and expansion and their impact on the S&P 500's performance. This involves exploring whether recessions and other macroeconomic shifts lead to predictable changes in the stock market.

Evaluating Monetary Policy Impacts: Assess the role of monetary stance (e.g., restrictive or expansionary policies) in shaping stock market behavior, providing insights into how policy decisions influence financial markets.

Developing Predictive Insights: The study will employ linear regression and random forest models to predict SPX YoY% based on the provided features. These models are chosen to combine the simplicity and interpretability of linear regression with the flexibility and non-linear pattern recognition capabilities of random forests. The goal is to determine the extent to which macroeconomic and financial data can forecast market performance, potentially aiding investors and policymakers in decision-making.

By addressing these goals, the research contributes to a deeper understanding of the interplay between economic fundamentals and financial market dynamics, offering valuable insights for both academic and practical applications.

Basic Data Cleaning:

- **Set Column Names**: Used the 5th row as the new column headers bc the 1st 4 weren't applicable

- **Remove Extra Rows**: Deleted the first 5 rows, which were unnecessary.
- **Fix Row Numbers**: Reset row numbering to make it tidy.
- **Convert Dates**: Changed date format from Excel-style to a readable calendar date.
- **Keep Key Data**: Kept only the first and third columns for focus.
- **Rename and Format**: Renamed `rstar` to `R_star(%)`, converted to numbers, and rounded to two decimals.

```
names(R_star) <- R_star[5, ]
```

```
## Warning: The 'value' argument of 'names<-()' can't be empty as of tibble 3.0.0.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```

```
## Warning: The 'value' argument of 'names<-()' must be a character vector as of tibble
## 3.0.0.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```

```
R_star <- R_star[-(1:5), ]

rownames(R_star) <- NULL


R_star$Date <- as.Date(as.numeric(R_star$Date), origin = "1899-12-30")


R_star <- R_star[, c(1, 3)]

R_star <- R_star |>
  mutate(`R_star(%)` = as.numeric(rstar) |> round(2)) |>
  select(-rstar)

head(R_star)
```

```
## # A tibble: 6 x 2
##   Date       'R_star(%)'
##   <date>           <dbl>
## 1 1961-01-01        5.12
## 2 1961-04-01        5.34
## 3 1961-07-01        5.31
## 4 1961-10-01        5.29
## 5 1962-01-01        5.14
## 6 1962-04-01        4.64
```

- `CPI_clean`:
  - Converted `DATE` to a proper date format (`YYYY-MM-DD`).

  - Converted `CPALTT01USM657N` to numeric, rounded to two decimal places, and renamed it to `CPI MoM(%)`.
  - Removed unnecessary columns (`CPALTT01USM657N`, `DATE`).

- **`Unemployment_rate_clean`**:
  - Converted `DATE` to a proper date format.
  - Renamed `UNRATE` to `Unemployment_rate`.
  - Dropped redundant columns (`UNRATE`, `DATE`).

- **`M2SL_clean`**:
  - Converted `DATE` to a proper date format.
  - Renamed `M2SL` to `M2 (Billions $)`.
  - Removed unused columns (`DATE`, `M2SL`).

- **`Recession_clean`**:
  - Converted `DATE` to a proper date format.
  - Transformed `USREC` into a factor: `1` becomes "YES", `0` becomes "NO".
  - Removed unused columns (`DATE`, `USREC`).

- **`Housing_starts_clean`**:
  - Converted `DATE` to a proper date format.
  - Renamed `HOUST` to `Starts (Thousands)`.
  - Selected only `Date` and `Starts (Thousands)` columns.

- **`Fed_funds_clean`**:
  - Converted `DATE` to a proper date format.
  - Renamed `DFF` to `Fed_funds_rate (%)`.
  - Selected only `Date` and `Fed_funds_rate (%)` columns.

- **`One_Year_inflation_expectations_clean`**:
  - Converted `DATE` to a proper date format.
  - Renamed `EXPINF1YR` to `One_year_inflation (%)`, rounded to two decimals.
  - Removed unused columns (`EXPINF1YR`, `DATE`).

- **`SPX_clean`**:
  - Verified that `Date` was already in a valid date format.
  - Renamed `SP500` to `SPX (USD)` and rounded values to two decimals.
  - Retained only `Date` and `SPX (USD)` columns.

```
CPI_clean <- CPI |>
  mutate(
    Date = as.Date(DATE, format = "%Y-%m-%d"),
    `CPI MoM(%)` = as.numeric(CPALTT01USM657N) |> round(2)
  ) |>
  select(-CPALTT01USM657N, -DATE)
head(CPI_clean)
```

```
##         Date CPI MoM(%)
## 1 1955-02-01       0.00
## 2 1955-03-01       0.00
## 3 1955-04-01       0.00
## 4 1955-05-01       0.00
## 5 1955-06-01       0.00
## 6 1955-07-01       0.37
```

```
Unemployment_rate_clean <- Unemployment_Rate |>
  mutate(
    Date = as.Date(DATE, format = "%Y-%m-%d"),
    Unemployment_rate = UNRATE
  ) |>
  select(-UNRATE, -DATE)
head(Unemployment_rate_clean)
```

```
##         Date Unemployment_rate
## 1 1955-02-01               4.7
## 2 1955-03-01               4.6
## 3 1955-04-01               4.7
## 4 1955-05-01               4.3
## 5 1955-06-01               4.2
## 6 1955-07-01               4.0
```

```
M2SL_clean <- M2SL |>
  mutate(
    Date = as.Date(DATE, format = "%Y-%m-%d"), `M2 (Billions $)` = M2SL
  )|>
  select(-DATE, -M2SL)
head(M2SL_clean)
```

```
##         Date M2 (Billions $)
## 1 1959-01-01           286.6
## 2 1959-02-01           287.7
## 3 1959-03-01           289.2
## 4 1959-04-01           290.1
## 5 1959-05-01           292.2
## 6 1959-06-01           294.1
```

```
Recession_clean <- Recession |>
  mutate(
    Date = as.Date(DATE, format = "%Y-%m-%d"),
    Recession = as.factor(ifelse(USREC == 1, "YES", "NO"))
  ) |>
  select(-DATE, -USREC)
head(Recession_clean)
```

```
##         Date Recession
## 1 1955-02-01        NO
## 2 1955-03-01        NO
## 3 1955-04-01        NO
## 4 1955-05-01        NO
```

```
## 5 1955-06-01        NO
## 6 1955-07-01        NO
```

```
Housing_starts_clean <- Housing_Starts |>
  mutate(
    Date = as.Date(DATE, format = "%Y-%m-%d"),
    `Starts (Thousands)` = HOUST) |>
  select(Date, `Starts (Thousands)`)
head(Housing_starts_clean)
```

```
##         Date Starts (Thousands)
## 1 1959-01-01                1657
## 2 1959-02-01                1667
## 3 1959-03-01                1620
## 4 1959-04-01                1590
## 5 1959-05-01                1498
## 6 1959-06-01                1503
```

```
Fed_funds_clean <- Fed_funds |>
  mutate(
    Date = as.Date(DATE, format = "%Y-%m-%d"),
    `Fed_funds_rate (%)` = DFF) |>
  select(Date, `Fed_funds_rate (%)`)
head(Fed_funds_clean)
```

```
##         Date Fed_funds_rate (%)
## 1 1955-12-01                2.5
## 2 1955-12-02                2.5
## 3 1955-12-03                2.5
## 4 1955-12-04                2.5
## 5 1955-12-05                2.5
## 6 1955-12-06                2.5
```

```
One_Year_inflation_expectations_clean <- `1_Year_inflation_expectations` |>
  mutate(
    Date = as.Date(DATE, format = "%Y-%m-%d"),
    `One_year_inflation (%)` = EXPINF1YR |> round(2)
  ) |>
  select(-EXPINF1YR, -DATE)
head(One_Year_inflation_expectations_clean)
```

```
##         Date One_year_inflation (%)
## 1 1982-01-01                   6.39
## 2 1982-02-01                   6.43
## 3 1982-03-01                   6.39
## 4 1982-04-01                   6.14
## 5 1982-05-01                   5.49
## 6 1982-06-01                   5.45
```

```
SPX_clean <- SPX |>
  mutate(
```

```
    Date = as.Date(Date, format = "%Y-%m-%d"),
    `SPX (USD)` = SP500 |> round(2)
  )|>
  select(Date, `SPX (USD)`)
head(SPX_clean)
```

```
##         Date SPX (USD)
## 1 2023-09-01   4515.77
## 2 2023-08-01   4457.36
## 3 2023-07-01   4508.08
## 4 2023-06-01   4345.37
## 5 2023-05-01   4146.17
## 6 2023-04-01   4121.47
```

- **Removing Duplicates**:
  - Ensured no duplicate rows in `R_star`, `Fed_funds_clean`, and `One_Year_inflation_expectations_clean` by keeping only unique `Date` entries.
- **Merging Datasets**:
  - Merged `R_star`, `Fed_funds_clean`, and `One_Year_inflation_expectations_clean` using `Date` as the key.
  - Removed rows with missing values (`NA`) after merging to ensure a clean dataset.
- **Creating `Monetary_tightness`**:
  - **Neutral Rate**: Calculated as the sum of `R_star(%)` (natural interest rate) and `One_year_inflation (%)` (inflation expectations).
  - **Monetary Stance**:
    - Classified the Federal Reserve's stance as:
      - **Restrictive**: When the Fed Funds Rate is more than 0.5% above the Neutral Rate.
      - **Accommodative**: When the Fed Funds Rate is more than 0.5% below the Neutral Rate.
      - **Neutral**: When the Fed Funds Rate is within $\pm 0.5\%$ of the Neutral Rate.
    - The $\pm 0.5\%$ range accounts for minor adjustments and ensures clear classification of significant policy shifts.
- **Selecting Final Columns**:
  - Kept only `Date` and `Monetary_stance` columns for the final output (`Monetary_tightness_clean`).

```
R_star <- R_star |> distinct(Date, .keep_all = TRUE)
Fed_funds_clean <- Fed_funds_clean |> distinct(Date, .keep_all = TRUE)
One_Year_inflation_expectations_clean <- One_Year_inflation_expectations_clean |> distinct(Date, .keep_a


Real_fed_funds_and_r_star <- R_star |>
  left_join(Fed_funds_clean, by = "Date") |>
  left_join(One_Year_inflation_expectations_clean, by = "Date") |>
  na.omit()

band_width <- 0.5
```

```r
Monetary_tightness <- Real_fed_funds_and_r_star |>
  mutate(
    Neutral_rate = `R_star(%)` + `One_year_inflation (%)`,
    Monetary_stance = case_when(
      `Fed_funds_rate (%)` > (Neutral_rate + band_width) ~ "Restrictive",
      `Fed_funds_rate (%)` < (Neutral_rate - band_width) ~ "Accommodative",
      TRUE ~ "Neutral"
    )
  )
head(Monetary_tightness)
```

```
## # A tibble: 6 x 6
##   Date       `R_star(%)` `Fed_funds_rate (%)` `One_year_inflation (%)`
##   <date>           <dbl>                <dbl>                     <dbl>
## 1 1982-01-01        2.79                13.1                      6.39
## 2 1982-04-01        2.86                15.5                      6.14
## 3 1982-07-01        2.83                14.7                      6.37
## 4 1982-10-01        2.74                10.9                      5.13
## 5 1983-01-01        2.84                11.2                      4.86
## 6 1983-04-01        2.86                 9.12                     4.63
## # i 2 more variables: Neutral_rate <dbl>, Monetary_stance <chr>
```

```r
Monetary_tightness_clean <- Monetary_tightness |>
  select(Date, Monetary_stance)
head(Monetary_tightness_clean)
```

```
## # A tibble: 6 x 2
##   Date       Monetary_stance
##   <date>     <chr>
## 1 1982-01-01 Restrictive
## 2 1982-04-01 Restrictive
## 3 1982-07-01 Restrictive
## 4 1982-10-01 Restrictive
## 5 1983-01-01 Restrictive
## 6 1983-04-01 Restrictive
```

- **Combining Data**:
  - Merged all cleaned datasets (`Housing_starts_clean`, `CPI_clean`, `Unemployment_rate_clean`, `Recession_clean`, `Monetary_tightness_clean`, and `SPX_clean`) using `Date` as the key.
  - Filled missing `Monetary_stance` values by carrying forward the last observed value.
- **Handling Missing Values**:
  - Removed rows with any remaining `NA` values to ensure completeness.
- **Data Formatting**:
  - Converted `Recession` and `Monetary_stance` into categorical variables for analysis.
- **Creating New Variables**:
  - **CPI Index**:

    - Reconstructed using cumulative monthly percentage changes (`CPI MoM(%)`) starting from a base of 100.
  - **CPI YoY**:
    - Calculated the year-over-year percentage change in the CPI Index.
  - **SPX YoY(%)**:
    - Calculated the year-over-year percentage change in the S&P 500 index (`SPX (USD)`).
  - **Unemployment YoY**:
    - Calculated the year-over-year percentage change in the unemployment rate.
  - **Using a Lag of 11**:
    - For YoY calculations, a lag of 11 months was used instead of 12 because all October 1st data was removed for simplicity. This ensures accurate comparisons with the adjusted dataset.

- **Filtering Data**:
  - Removed rows with `NA` values in newly created variables (`CPI_YoY`, `SPX YoY(%)`, `Unemployment_YoY`).

- **Final Output**:
  - Sorted the dataset by `Date` and verified all transformations.

```
combined_data <- Housing_starts_clean |>
  left_join(CPI_clean, by = "Date") |>
  left_join(Unemployment_rate_clean, by = "Date") |>
  left_join(Recession_clean, by = "Date") |>
  left_join(Monetary_tightness_clean, by = "Date") |>
  left_join(SPX_clean, by = "Date")

combined_data2 <- combined_data |>
  fill(Monetary_stance, .direction = "down") |>
  na.omit()

combined_data3 <- combined_data2 |>
  mutate(
    Recession = as.factor(Recession),
    Monetary_stance = as.factor(Monetary_stance)
  )

combined_data4 <- combined_data3 |>
  arrange(Date) |>
  mutate(
    CPI_Index = 100 * cumprod(1 + `CPI MoM(%)` / 100),
    CPI_YoY = round((CPI_Index / lag(CPI_Index, 11) - 1) * 100, 2),

    `SPX YoY(%)` = round((`SPX (USD)` / lag(`SPX (USD)`, 11) - 1) * 100, 2),
    Unemployment_YoY = round((Unemployment_rate / lag(Unemployment_rate, 12) - 1) * 100, 2)
  ) |>
  filter(!is.na(CPI_YoY), !is.na(`SPX YoY(%)`), !is.na(Unemployment_YoY))

view(combined_data4)
```

Exploratory Plot 1

The plot displays a histogram of housing starts (in thousands) categorized by monetary stance (accommodative, neutral, restrictive) and separated by whether a recession is occurring ("NO" or "YES"). This visualization helps in understanding the relationship between housing starts, monetary policy stance, and recessionary periods.

Why These Variables Were Chosen:

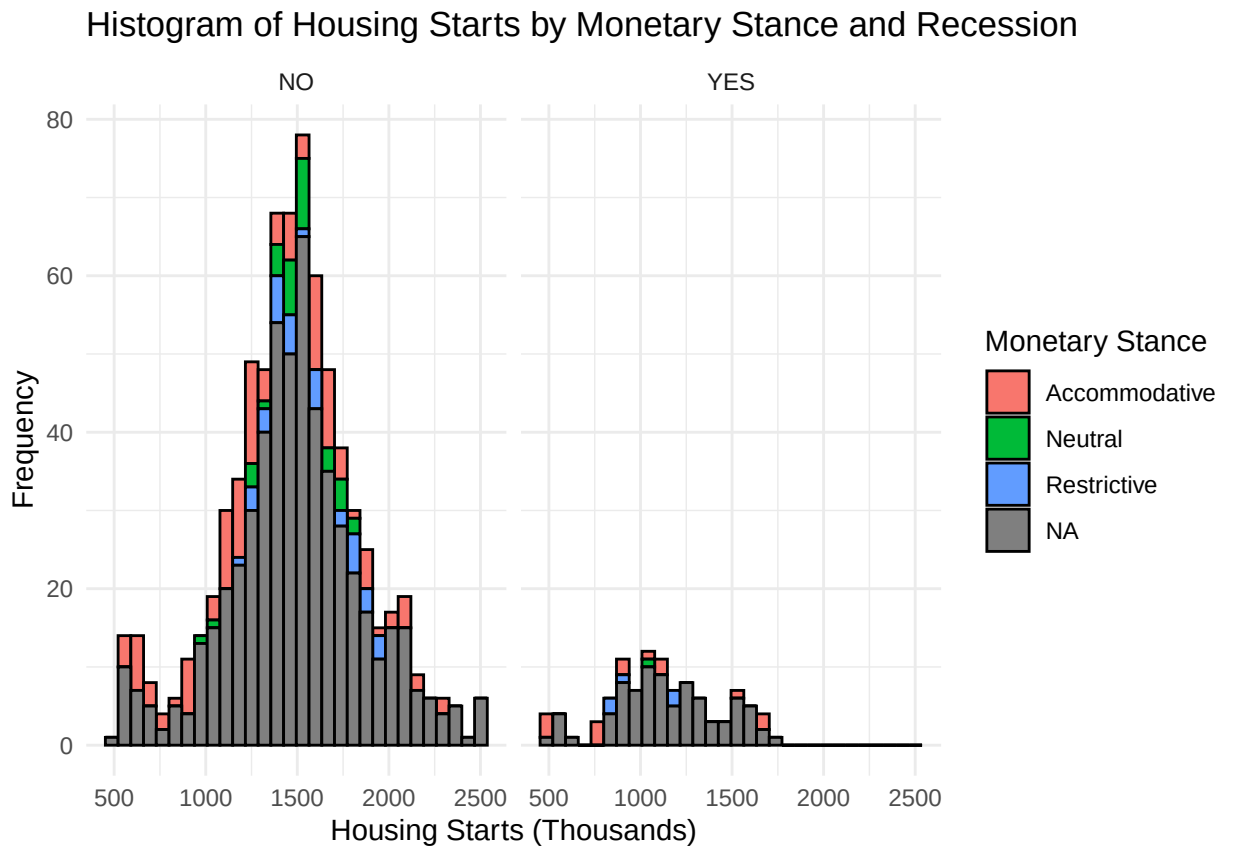
1. **Housing Starts**: This is a key economic indicator that reflects the state of the housing market and broader economic health.
2. **Monetary Stance**: Captures whether the monetary policy is accommodative, neutral, or restrictive, which influences economic activity, including housing starts.
3. **Recession**: Adding recession as a factor provides insight into how economic downturns interact with monetary policy to affect housing starts.

What the Plot Shows:

1. **Housing Starts by Monetary Stance**:
   - Under "NO" recession, housing starts are higher, with accommodative policies showing the highest frequency at higher levels of housing starts.
   - Neutral and restrictive policies have progressively fewer high housing starts.
2. **Housing Starts During a Recession ("YES")**:
   - Housing starts are significantly lower, regardless of monetary stance.
   - The restrictive stance shows even fewer housing starts, suggesting a strong suppressive effect during recessions.
3. **Distribution of Housing Starts (NO Recession)**:
   - The distribution of housing starts under "NO" recession appears **close to normal**, especially for the accommodative and neutral stances:
     - **Symmetry**: The data peaks around 1500–1700 thousands and tapers off symmetrically, resembling a bell curve.
     - **Central Tendency**: Housing starts cluster near the center of the range, indicating stability during non-recessionary periods.
     - This normal-like distribution reflects typical market behavior when the economy is stable, with monetary stances driving predictable outcomes.
   - Conversely, during recessions, this normal distribution is disrupted, with housing starts significantly lower and more scattered, reflecting the economic uncertainty.
4. **Overall Trends**:
   - Accommodative policies generally support higher housing starts.
   - Recessions strongly reduce housing starts, and restrictive stances exacerbate this decline.

This plot is valuable for analyzing how monetary policy interacts with economic conditions like recessions to influence a critical economic indicator, housing starts. It highlights the predictability and stability of housing starts during non-recessionary periods and contrasts it with the volatility and suppression observed during recessions.

```
ggplot(combined_data, aes(x = `Starts (Thousands)`, fill = Monetary_stance)) +
  geom_histogram(position = "stack", bins = 30, color = "black") +
  facet_wrap(~ Recession) +
  labs(
    title = "Histogram of Housing Starts by Monetary Stance and Recession",
    x = "Housing Starts (Thousands)",
    y = "Frequency",
    fill = "Monetary Stance"
  ) +
  theme_minimal()
```



Exploratory Plot 2

For the second exploratory plot we wanted to investigate the relationship between the percentage change of unemployment year over year with the percentage change in the rate of the CPI Inflation. Our rationale behind creating this plot was to visualize the Phillips Curve, a concept which states that the relationship between unemployment and inflation is inverse and has a "curved" relationship.

If the data did in fact show this inverse "curved" relationship, we anticipated that during periods of low inflation (recessions) we would see data points shift downward, and the curve that passed through these

points to shift downward as well. We were quite interested to see if the macro-economy of the last 60 years reflected this theory.

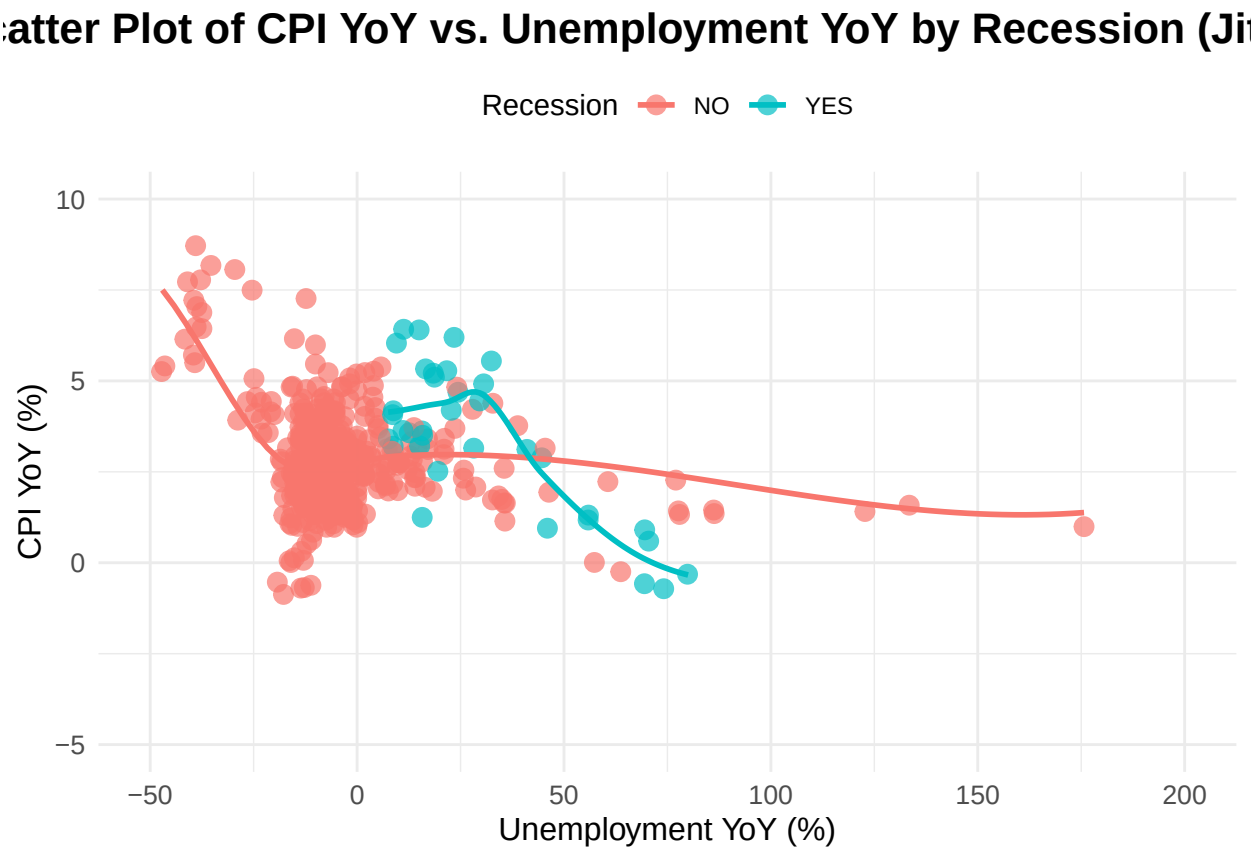
A jitter plot was used instead of a simple scatter-plot to better visualize clusters and overlapping points within the data. The plots showed that periods of non-recession characterized with higher inflation rates than during periods of recession, saw higher unemployment, as explained by the theory of the Phillips curve. The data, also exhibited the "curve" like shift (although, the data is not perfectly curved) from falling inflation during periods of recession.

```
ggplot(combined_data4, aes(x = Unemployment_YoY, y = CPI_YoY, color = Recession)) +
  geom_jitter(size = 3, alpha = 0.7, width = 0.2, height = 0.2) +
  labs(
    title = "Scatter Plot of CPI YoY vs. Unemployment YoY by Recession (Jittered)",
    x = "Unemployment YoY (%)",
    y = "CPI YoY (%)",
    color = "Recession"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    legend.position = "top"
  ) +
  xlim(-50, 200) +
  ylim(-5, 10)+
  geom_smooth(data = combined_data4, method = 'loess', se = FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 5 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 5 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



Model Plot 1

For the first model plot, we wanted to analyze the effect on monetary stance and recessions on the returns of the S&P 500. The S&P 500 being an index of the 500 fastest growing companies in the United States, is a great indicator of the US stock market at large. Therefore by looking at the effects on this index, we were essentially looking at the effects of monetary stance and recessions on the US stock market.

Theory would suggest that

1. An accomodative monetary stance, seeks to grow the economy through cuts to interest rates, increasing the money supply and encouraging businesses to invest and consumers to spend. This measure is usually taken up by the US Federal Reserve during periods of recession as it helps to slow down or reverse economic downturn. Should the Federal Reserve choose to stimulate the economy with an accomodative monetary policy during times of economic growth, they risk overstimulating the economy and pushing into a period of inflation or hyperinflation. As such, the hypothesis made was that
    1. During periods of non-recession or recession, a accomodative monetary stance should result in higher S&P returns, but a very large variation in returns as a result of varying degrees of restrictive stances taken up by the US Federal Reserve across the different economic conditions across time depending on how high or low the inflation level was at the time. These lower and high variance of values should be reflected in the boxplot.
2. A neutral monetary stance seeks to keep the economy at the same rate and make no changes to the interest rates or the money supply. As such the hypothesis made with respect to a neutral stance is that

    1. During periods of recession and non-recession a neutral monetary stance would cause little to no change in the S&P as no stimulation or restriction is applied on the economy, and the plot will reflect this with a box-plot with minimal variance on the y axis

3. A restrictive monetary stance seeks to slow down the economy by increasing interest rates and decreasing the money supply. This measure is usually taken up by the US Federal Reserve during periods of economic growth to discourage businesses from investing and consumers from spending and slow the economy down as a result. The federal reserve would be unlikely to use such a tactic during periods of economic downturn as it would reduce investing and spending further and hurt the economy. As such the hypothesis made was that:
    1. During periods of non-recession, a restrictive monetary policy should result in lower S&P returns but a very large variation in returns as a result of varying degrees of restrictive stances taken up by the US Federal Reserve across the different economic conditions across time depending on how high the inflation level was at the time. These lower and high variance of values should be reflected in the boxplot.
    2. During periods of recession, a restrictive monetary policy should hurt the economy and push S&P returns lower however it's difficult to speculate what the variance in S&P returns would be

The data reflects the accomodative hypothesis of

1. An accomodative monetary stance resulting in a high variability in S&P returns during periods of recession and non-recession

In the case of a neutral stance,

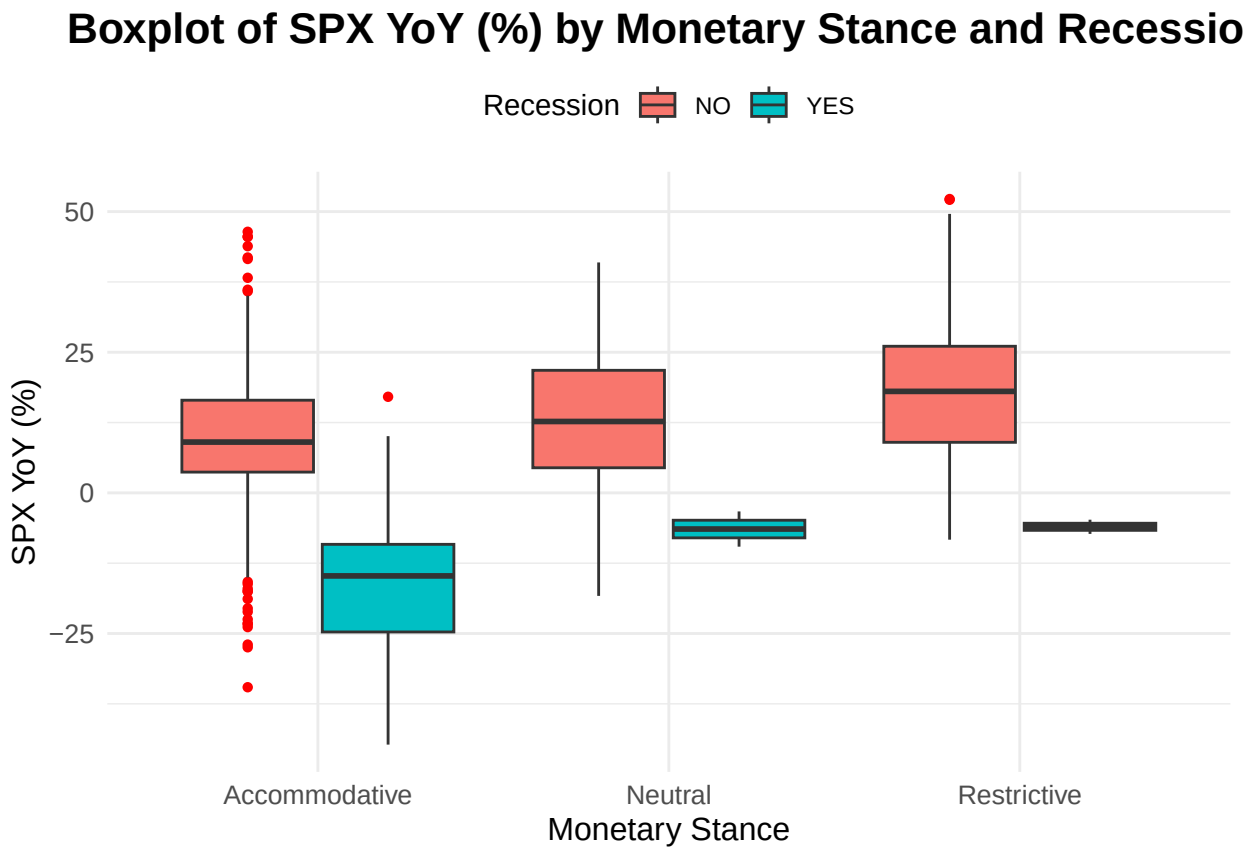
1. a neutral monetary stance results in a low variability in S&P returns during periods of recession, but...
2. a high variability in S&P returns during periods or non-recession

In the case of a restrictive stance

1. a restrictive monetary stance results in a low variability in S&P returns during periods of recession
2. a high variability in S&P returns during periods or non-recession

The high variability in S&P returns in the case of a neutral monetary stance during periods of non-recession could be a result of other economic factors such as GDP growth, market speculation, geopolitical risks etc. that would result in varying levels of percentage changes in the S&P depending on their strength or severity.

```
ggplot(combined_data4, aes(x = Monetary_stance, y = `SPX YoY(%)`, fill = Recession)) +
  geom_boxplot(outlier.color = "red", outlier.shape = 16, position = position_dodge(0.8)) +
  labs(
    title = "Boxplot of SPX YoY (%) by Monetary Stance and Recession",
    x = "Monetary Stance",
    y = "SPX YoY (%)",
    fill = "Recession"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    legend.position = "top"
  )
```



Model Plot 2

For the second model plot, we thought it might be interesting to look into the average S&P 500 returns by monetary stance during periods of recession and non-recession. We hoped that this would provide some insight into the economic conditions under which the US Federal Reserve would take on certain monetary stances since the S&P is a good aggregate of the US stock market at large.

The plot below shows that

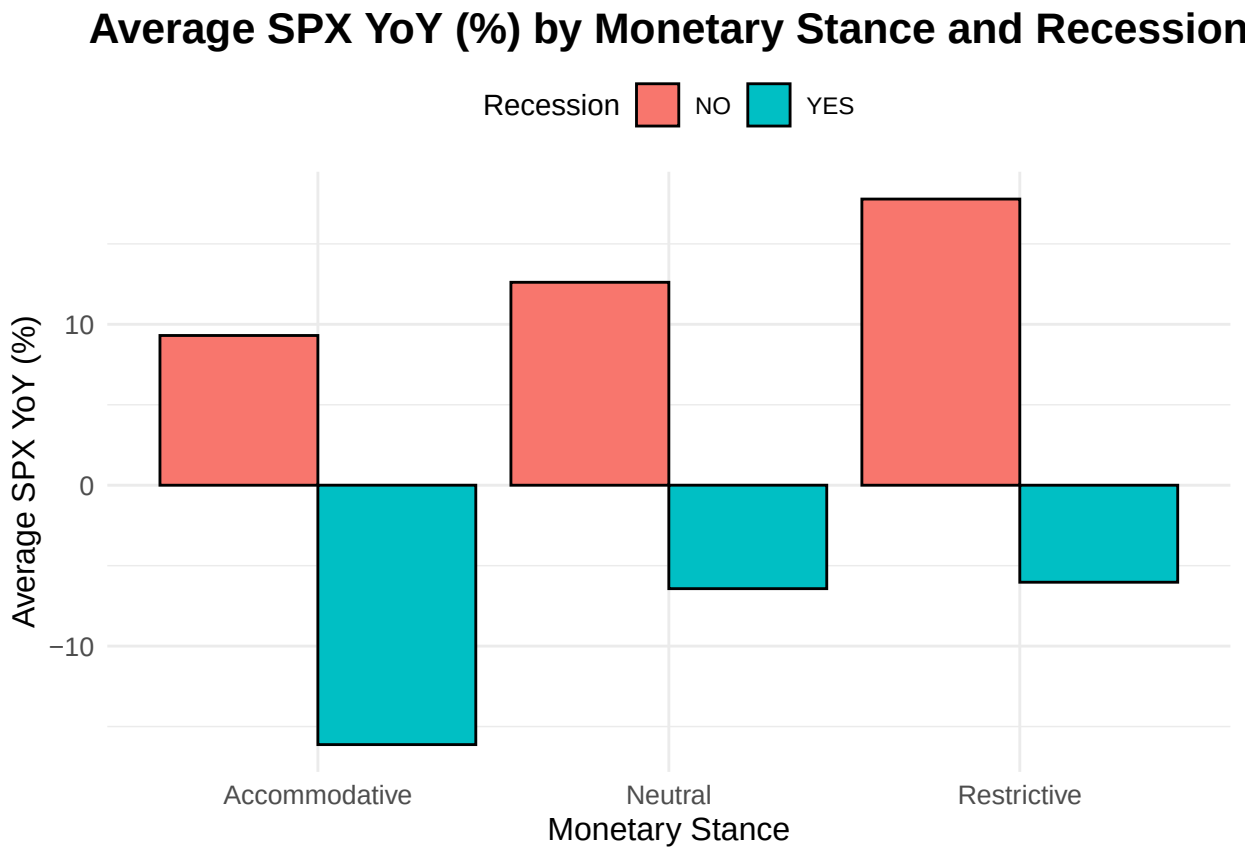
1. The Federal Reserve is more likely to take on an accomodative monetary stance when they see the S&P annual returns, and the US stock market at large growing at a rate of between -16% and 9%
    1. This makes sense considering that the S&P has seen an average return of about 10.3% since 1957, and growth under 10.3% could be a sign that the economy requires to be stimulated with an accomodative monetary stance
2. The Federal Reserve is more likely to take on a neutral monetary stance when the see S&P annual returns, and the US stock market growing at a rate of between -6% and 12%
3. The Federal Reserve is more likely to take on a restrictive monetary stance when the see S&P annual returns, and the US stock market growing at a rate of between 0 and -6% or exceeding 12%

It is important that the US Federal Reserve (Fed) may take on a neutral or restrictive monetary stance in a recessionary period despite what we would expect to combat certain economic scenarios such as during periods of stagflation (a scenario in which high inflation co-exists alongside economic downturn characterized by a recession). An example of this is during as the oil shocks and stagflation of the 1970s, when the Fed took a restrictive monetary stance, making cuts to interest rates to combat the record high inflation at the time.

```r
library(ggplot2)
library(dplyr)

# Summarize data to calculate average SPX YoY (%) for each Monetary Stance and Recession state
summary_data <- combined_data4 |>
  group_by(Monetary_stance, Recession) |>
  summarise(Average_SPX_YoY = mean(`SPX YoY(%)`, na.rm = TRUE)) |>
  ungroup()
```

```
## `summarise()` has grouped output by 'Monetary_stance'. You can override using
## the `.groups` argument.
```

```r
# Create the bar graph
ggplot(summary_data, aes(x = Monetary_stance, y = Average_SPX_YoY, fill = Recession)) +
  geom_bar(stat = "identity", position = "dodge", color = "black") +
  labs(
    title = "Average SPX YoY (%) by Monetary Stance and Recession",
    x = "Monetary Stance",
    y = "Average SPX YoY (%)",
    fill = "Recession"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    legend.position = "top"
  )
```



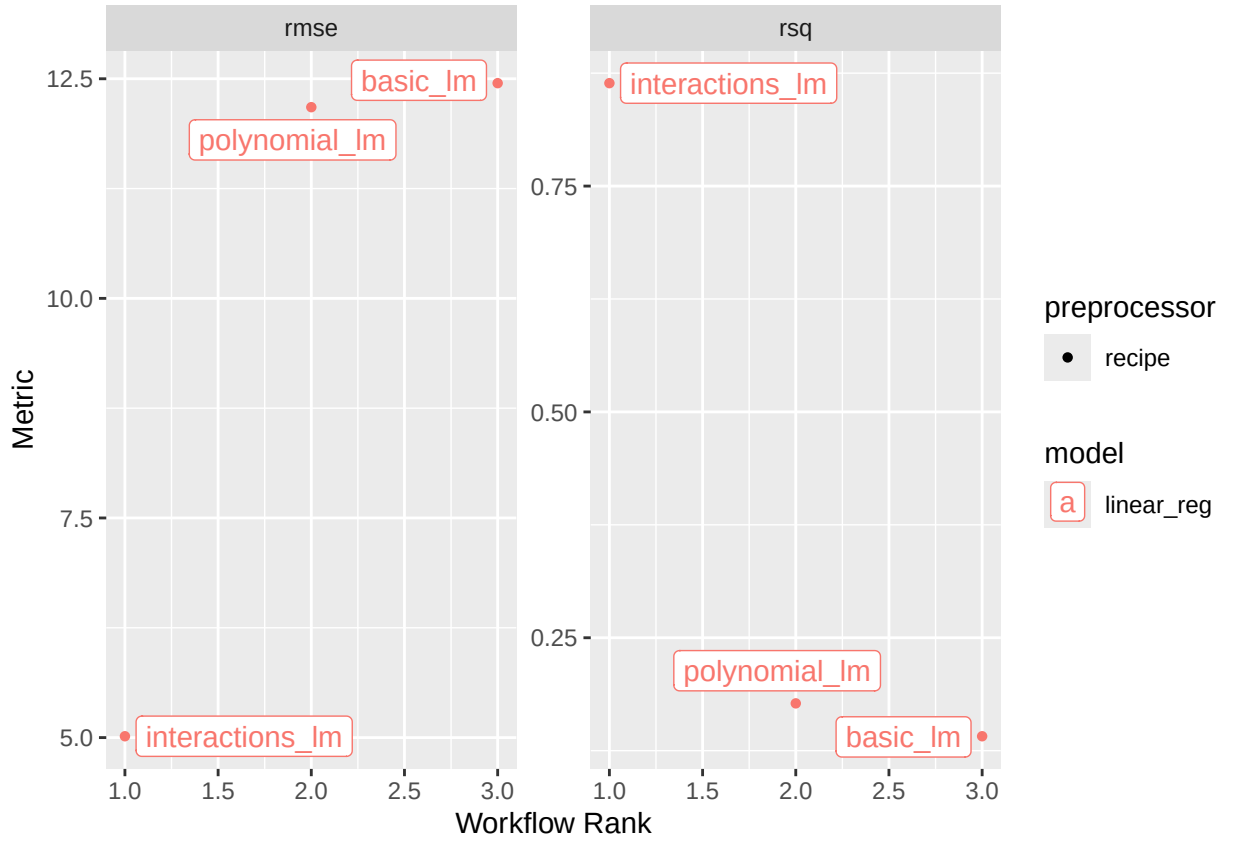
Exploratory Linear Model 1

From the exploratory and model plots, we figured out that there was a relationship between CPI Inflation and unemployment as well as the S&P 500 annual returns and the monetary stance taken on by the US Federal Reserve. Armed with this knowledge, we decided to make a model that could predict the annual return on the S&P 500 for any given year. The recipe used in the creation of this model computes the S&P 500 as a sum of the CPI Inflation, Recession and monetary stance. Three recipes with the same formula were created, each with an additional step added on to find the most optimal method of computing the index with a low RMSE (Root mean square error) and high R^2 (Coefficient of determination).

1. The first recipe "rec_basic" simply computes the S&P based on the sum of the four variables mentioned earlier.
2. The second recipe "rec_interactions" computes S&P using the same formula but by assigning monetary stance (a categorical variable) as a dummy variable (This step is recommended as it allows for the model to deal with a categorical variable by assigning it a binary variable, 0 or 1) and analyses the interaction between each of the variables by multiplying them together (this allows the model created from this recipe to better understand how each variable affects the other and factor it into the model, making the model more accurate). Lastly, all numeric predictors were normalized to make predictions more statistically accurate.
3. The third recipe "rec_polynomial" computes S&P using the same formula but aimed to look at the relationship between CPI and S&P as a more complex polynomial relationship. Degree 5 was obtained after trying different variations of degrees to see which one yielded the lowest RMSE and highest R^2 value. Once again all numeric predictors were normalized to make predictions more statistically accurate.

Having computed these recipes, we assigned each to a workflow and then a workflow set. We then re-sampled the results to improve the reliability of the predictions from the model. Finally, we graphed the models to see which one performed the best (the one which had the lowest RMSE and highest R^2 value), extracted it from the workflow set, finalized the model and analyzed metrics like RMSE and R^2 to augment how well the model would work under real world conditions. From our results we saw that rec_interactions was the best recipe with an RMSE of around 6.3 and an RSQ of around 0.87 (at best).

```
lin_model1_data <- combined_data4 |>
  select(`SPX YoY(%)`, Unemployment_YoY, CPI_YoY, Recession, Monetary_stance)

lm1_split <- initial_validation_split(lin_model1_data, prop = c(0.6, 0.2))
lm1_train <- training(lm1_split)
lm1_val <- validation(lm1_split)
lin_model1_test <- testing(lm1_split)

rec_basic <- recipe(`SPX YoY(%)` ~ CPI_YoY + Recession + Monetary_stance + Unemployment_YoY, data = lm1_

rec_interactions <- recipe(`SPX YoY(%)` ~ CPI_YoY + Recession + Monetary_stance + Unemployment_YoY, data
  step_dummy(Monetary_stance) |>
  step_interact(~ CPI_YoY * Unemployment_YoY + `SPX YoY(%)` * Unemployment_YoY + CPI_YoY * `SPX YoY(%)`
  step_normalize(all_numeric_predictors())

rec_polynomial <- recipe(`SPX YoY(%)` ~ CPI_YoY + Recession + Monetary_stance + Unemployment_YoY, data =
  step_poly(CPI_YoY, degree = 5) |>
  step_normalize(all_numeric_predictors())

lm1_wflow <- workflow() |>
  add_model(linear_reg()) |>
  add_recipe(rec_interactions)

final_w1 <- workflow_set(
  preproc = list(
    basic = rec_basic,
    interactions = rec_interactions,
    polynomial = rec_polynomial
  ),
  models = list(lm = linear_reg())
) |>
  workflow_map(
    fn = "fit_resamples",
    seed = 100,
    resamples = validation_set(lm1_split)
  )

final_w1 |>
  autoplot() +
  geom_label_repel(aes(label = wflow_id))
```




```
best_SPX <- final_w1 |>
  extract_workflow_set_result("interactions_lm") |>
  select_best(metric = "rmse")

final_w1 |>
  extract_workflow("interactions_lm") |>
  finalize_workflow(best_SPX) |>
  last_fit(lm1_split) |>
  collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric .estimator .estimate .config
##   <chr>   <chr>          <dbl> <chr>
## 1 rmse    standard       7.33  Preprocessor1_Model1
## 2 rsq     standard       0.839 Preprocessor1_Model1
```

Exploratory Linear Model 2

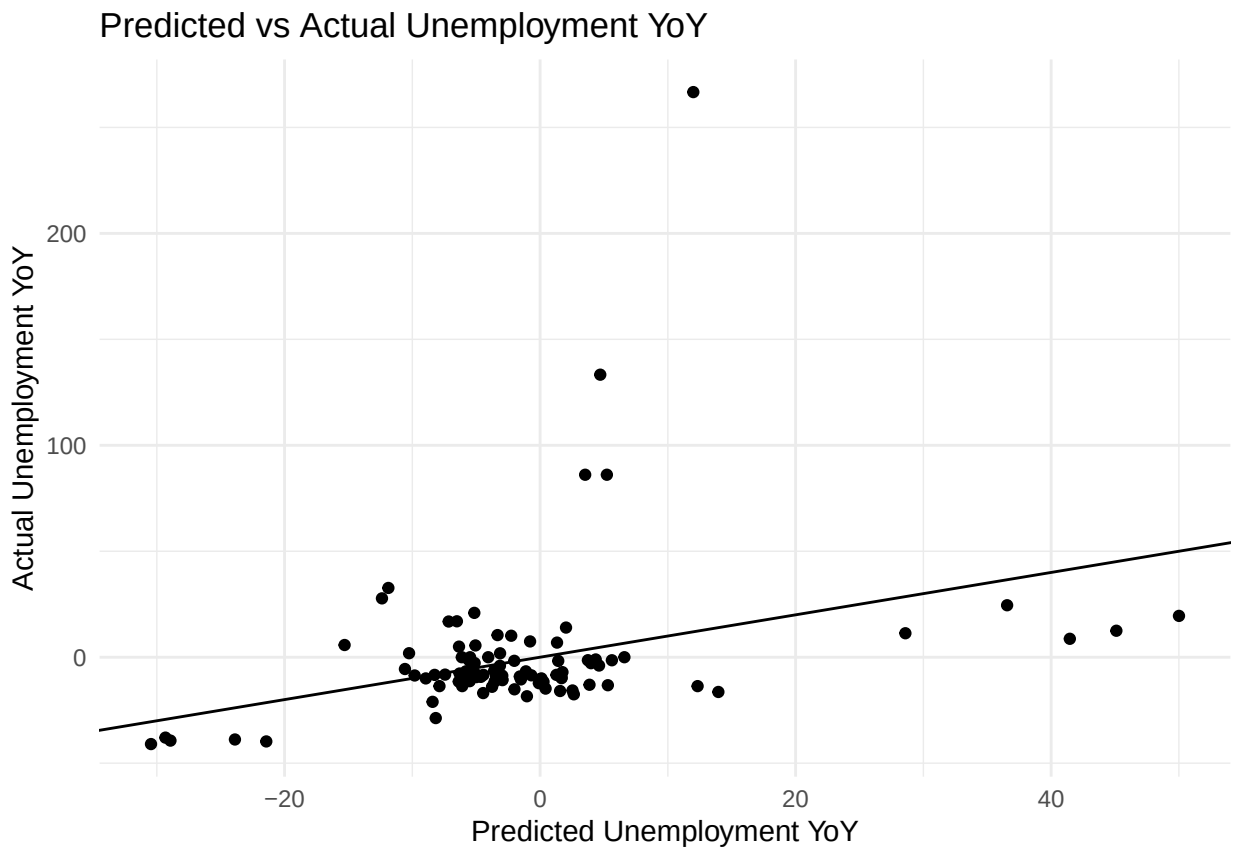
Having created a model that computed the return on the S&P 500 we thought it might be interesting to create a model that computed the unemployment rate based on the knowledge that CPI, S&P and unemployment were related to one another in some way. Once again, three recipes, all with the same formula were created, each with some steps added on to find the most optimal model that fit the data. The recipe formula was computed as the sum of CPI and the return on the S&P 500.

1. The first recipe "rec_basic1" uses the basic formula along with a step to normalize the predictions to increase the reliability and accuracy of predictions.
2. The second recipe "rec_interactions1" uses the basic formula along with a step that computes the interaction between each variable by multiplying them with each other and another step that once again normalizes all numeric predictors.
3. The third recipe "rec_polynomial1" uses the basic formula along with two steps to see if CPI and the S&P has a more complex relationship with the unemployment rate. Once again, Degree 2 was obtained after trying different variations of degrees to see which one yielded the lowest RMSE and highest R^2 value. Lastly, all numeric predictors were normalized.

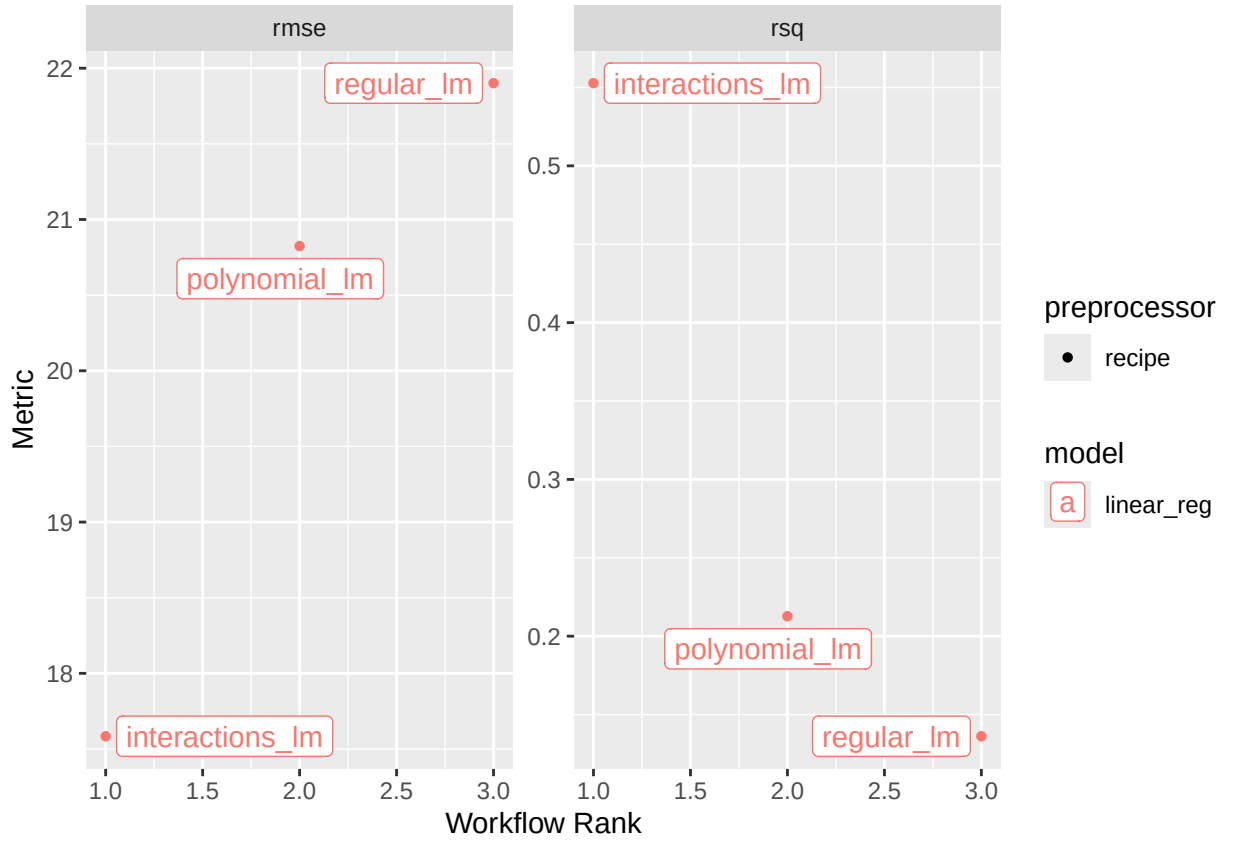
Having computed these recipes, we assigned each to a workflow and then a workflow set. We then re-sampled the results to improve the reliability of the predictions from the model. Finally, we graphed the models to see which one performed the best (the one which had the lowest RMSE and highest R^2 value), extracted it from the workflow set, finalized the model and analyzed metrics like RMSE and R^2 to augment how well the model would work under real world conditions. From our results we saw that rec_interactions was the best recipe with an RMSE of around 20 and an RSQ of around 0.64 (at best).

```
lin_model2_data <- combined_data4

lm2_split <- initial_validation_split(lin_model1_data, prop = c(0.6, 0.2))
lm2_train <- training(lm1_split)
lm2_val <- validation(lm1_split)
lin_model2_test <- testing(lm1_split)

rec_basic1 <- recipe(Unemployment_YoY ~ CPI_YoY + `SPX YoY(%)`, data = lm1_train) |>
  step_normalize(all_numeric_predictors())

rec_interactions1 <- recipe(Unemployment_YoY ~ CPI_YoY + `SPX YoY(%)`, data = lm1_train) |>
  step_interact(~ CPI_YoY * Unemployment_YoY + `SPX YoY(%)` * Unemployment_YoY + CPI_YoY * `SPX YoY(%)`
  step_normalize(all_numeric_predictors())

rec_polynomial1 <- recipe(Unemployment_YoY ~ CPI_YoY + `SPX YoY(%)`, data = lm1_train) |>
  step_poly(CPI_YoY, degree = 2) |>
  step_poly(`SPX YoY(%)`, degree = 2) |>
  step_normalize(all_numeric_predictors())

rec_all <- recipe(Unemployment_YoY ~ ., data = lm1_train)

lm1_wflow <- workflow() |>
  add_model(linear_reg()) |>
  add_recipe(rec_all)

lm1_wflow |>
  fit(lm1_train) |>
  predict(new_data = lm1_val) |>
  bind_cols(lm1_val) |>
  ggplot(aes(x = .pred, y = Unemployment_YoY)) +
  geom_point() +
  labs(
    title = "Predicted vs Actual Unemployment YoY",
    x = "Predicted Unemployment YoY",
    y = "Actual Unemployment YoY"
  ) +
  theme_minimal() +
```

```
  geom_abline(intercept = 0, slope = 1)
```




```
final_w2 <- workflow_set(
  preproc = list(
    regular = rec_basic1,
    interactions = rec_interactions1,
    polynomial = rec_polynomial1
  ),
  models = list(lm = linear_reg())
) |>
  workflow_map(
    fn = "fit_resamples",
    seed = 100,
    resamples = validation_set(lm2_split)
  )

final_w2 |>
  autoplot() +
  geom_label_repel(aes(label = wflow_id))
```




```
best_un <- final_w1 |>
  extract_workflow_set_result("interactions_lm") |>
  select_best(metric = "rmse")

final_w2 |>
  extract_workflow("interactions_lm") |>
  finalize_workflow(best_un) |>
  last_fit(lm2_split) |>
  collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric .estimator .estimate .config
##   <chr>   <chr>          <dbl> <chr>
## 1 rmse    standard      39.2   Preprocessor1_Model1
## 2 rsq     standard       0.304 Preprocessor1_Model1
```

Final Linear Model

Having created two models and observing that the S&P model performed more optimally than the unemployment rate model we decided to make a final S&P model with a few more steps to see if we could make it even more accurate. We decided to

1. set a dummy variable for recession for "rec_basic" and monetary stance for "rec_interactions"
2. adding a polynomial function to see if the relationship between unemployment was a complex one of degree 2 or more.

When we ran our models, we saw that "rec_polynomial" fit the data the best. Our learning from this was that recession and monetary stance had a lesser effect on the unemployment rate as compared to CPI Inflation and the return of the S&P 500.

```
lin_model_data <- combined_data4 |>
  select(`SPX YoY(%)`, Unemployment_YoY, CPI_YoY, Recession, Monetary_stance)

lm_split <- initial_validation_split(lin_model1_data, prop = c(0.6, 0.2))
lm_train <- training(lm_split)
lm_val <- validation(lm_split)
lm_test <- testing(lm_split)

rec_basic <- recipe(`SPX YoY(%)` ~ CPI_YoY + Recession + Monetary_stance + Unemployment_YoY, data = lm1_
  step_dummy(Recession) |>
  step_interact(~ CPI_YoY * Unemployment_YoY + `SPX YoY(%)` * Unemployment_YoY + CPI_YoY * `SPX YoY(%)`

rec_interactions <- recipe(`SPX YoY(%)` ~ CPI_YoY + Recession + Monetary_stance + Unemployment_YoY, data
  step_dummy(Monetary_stance) |>
  step_interact(~ CPI_YoY * Unemployment_YoY + `SPX YoY(%)` * Unemployment_YoY + CPI_YoY * `SPX YoY(%)`
  step_normalize(all_numeric_predictors())

rec_polynomial <- recipe(`SPX YoY(%)` ~ CPI_YoY + Recession + Monetary_stance + Unemployment_YoY, data =
  step_interact(~ CPI_YoY * Unemployment_YoY + `SPX YoY(%)` * Unemployment_YoY + CPI_YoY * `SPX YoY(%)`
  step_poly(Unemployment_YoY, degree = 2) |>
  step_normalize(all_numeric_predictors())

lm1_wflow <- workflow() |>
  add_model(linear_reg()) |>
  add_recipe(rec_interactions)

final_wf <- workflow_set(
  preproc = list(
    basic = rec_basic,
    interactions = rec_interactions,
    polynomial = rec_polynomial
  ),
  models = list(lm = linear_reg())
) |>
  workflow_map(
    fn = "fit_resamples",
    seed = 100,
    resamples = validation_set(lm_split)
  )

final_wf |>
  autoplot() +
  geom_label_repel(aes(label = wflow_id))
```


```
best_SPX1 <- final_w1 |>
  extract_workflow_set_result("polynomial_lm") |>
  select_best(metric = "rmse")

final_wf |>
  extract_workflow("polynomial_lm") |>
  finalize_workflow(best_SPX1) |>
  last_fit(lm1_split) |>
  collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric .estimator .estimate .config
##   <chr>   <chr>          <dbl> <chr>
## 1 rmse    standard       6.93  Preprocessor1_Model1
## 2 rsq     standard       0.862 Preprocessor1_Model1
```

Advanced Model

Findings from the Random Forest Model

The random forest (RF) model aimed to predict the S&P 500's year-over-year percentage change (SPX YoY%) using macroeconomic features such as unemployment, inflation, recession status, and monetary stance.

Feature Importance

- **Unemployment YoY** was the most influential variable, indicating a strong relationship between changes in unemployment and SPX YoY%.
- **CPI YoY (Inflation)** also played a significant role, reflecting the market's sensitivity to inflation trends.
- **Recession** had moderate importance, showing its indirect but notable impact on stock market returns.
- **Monetary Stance** had the lowest importance, suggesting that while policy decisions influence the market, their direct effect on SPX YoY% is less pronounced compared to other variables.

Model Performance

- **RMSE (Root Mean Square Error):** ~9.82 indicates the average prediction error in SPX YoY% is relatively high, suggesting room for improvement in the model.
- **R^2 (Coefficient of Determination):** ~0.50 indicates the model explains about 50% of the variance in SPX YoY%, showcasing moderate predictive power.
- **MAE (Mean Absolute Error):** ~7.77 reflects the average absolute prediction error, providing a more interpretable measure of accuracy.

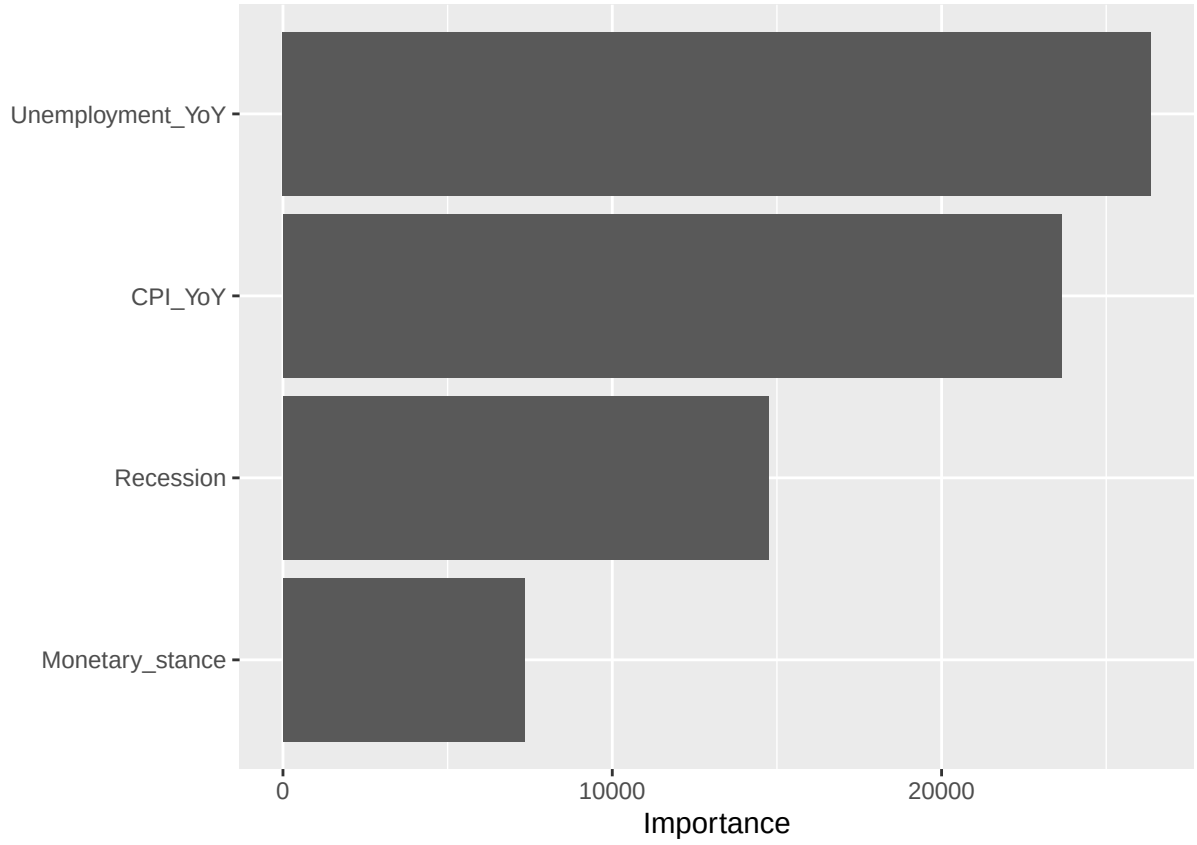
Key Takeaway The model demonstrates that unemployment and inflation are the strongest predictors of SPX YoY%, with recessions and monetary stance contributing less prominently.

```
model_data <- combined_data4 |>
  select(`SPX YoY(%)`, Unemployment_YoY, CPI_YoY, Recession, Monetary_stance)

set.seed(123)
data_split <- initial_split(model_data, prop = 0.8)
train_data <- training(data_split)
test_data <- testing(data_split)

rf_model <- rand_forest(
  mtry = 2,
  trees = 500,
  min_n = 5
) |>
  set_engine("randomForest") |>
  set_mode("regression")


rf_workflow <- workflow() |>
  add_model(rf_model) |>
  add_formula(`SPX YoY(%)` ~ Unemployment_YoY + CPI_YoY + Recession + Monetary_stance)

rf_fit <- rf_workflow |>
  fit(data = train_data)

rf_predictions <- predict(rf_fit, new_data = test_data) |>
  bind_cols(test_data)
```

```r
rf_metrics <- rf_predictions |>
  metrics(truth = `SPX YoY(%)`, estimate = .pred)

print(rf_metrics)
```

```
## # A tibble: 3 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       9.82
## 2 rsq     standard       0.498
## 3 mae     standard       7.77
```

```r
rf_importance <- rf_fit |>
  extract_fit_parsnip() |>
  vip::vip()

print(rf_importance)
```



Limitations of the Data

1. **CPI Methodology Changes**:
   - The methodology for calculating the Consumer Price Index (CPI) has evolved over time. Changes in the basket of goods, weighting, and other adjustments may introduce inconsistencies when comparing CPI data across years.

2. **Missing October 1st Data**:

    - All October 1st data points were removed due to missing entries, resulting in only 11 months of data for each year. This impacts year-over-year (YoY) calculations, as the lag was adjusted to 11 months instead of the standard 12. This simplification might overlook some seasonal or cyclical trends that occur annually.

3. **Subjective Definition of Monetary Stance**:

    - The classification of monetary stance (Restrictive, Neutral, Accommodative) depends on subjective choices, such as using the Neutral Rate and a $\pm 0.5\%$ bandwidth. While these thresholds are informed by economic theory, they may not perfectly capture nuanced central bank intentions or real-world impacts.

4. **Use of 1-Year Forward Inflation Expectations**:

    - The calculation of the Neutral Rate relies on 1-year forward inflation expectations (`One_year_inflation (%)`), which are based on forecasts. These expectations may not accurately reflect actual future inflation, introducing potential bias in the estimation of the Neutral Rate and monetary stance.

5. **Assumptions in Recession Data**:

    - The recession indicator (`Recession`) is binary and does not account for the depth or severity of economic downturns. This simplification may obscure differences between mild slowdowns and severe recessions.

6. **Exclusion of Other Relevant Factors**:

    - Some factors influencing the S&P 500 and macroeconomic indicators, such as geopolitical events, technological innovations, or global economic conditions, are not included in the dataset. This limits the scope of the analysis.

7. **Static Model of Monetary Policy**:

    - The dataset assumes that the relationship between variables, such as interest rates and inflation, remains consistent over time. However, these relationships can evolve due to changing economic conditions, market expectations, and central bank policies.

Conclusion

Through a series of exploratory plots, linear and advanced models, we were able to understand the underlying factors that drive fluctuation in the S&P 500 and understand how macroeconomic factors influence market behavior. We learned that factors like CPI Inflation, unemployment rate and the monetary stances taken on the US Federal Reserve, were far more interconnected that we previously imagined and that a change in one of these variables would result in a domino of effects on all other macroeconomic factors.