Modeling and Forecasting U.S. Inflation

STAT410 Final Project Report

Benjamin Allen

11/20/2022

Abstract

The primary aim of this project is to model inflation in the United States using a variety of economic indicators and market prices. After modeling, attention will be focused on forecasting inflation to August 2023. A limited number of variables will be considered for this project, regardless, the relationships between these variables and the inflation rate will be discussed extensively.

Introduction

Inflation refers to a general rise in prices within an economy. Inflation is a staple of modern economies and understanding where inflation is trending is vital for policymakers' and businesses' decision-making.

Recently, national attention has shifted to the increases in inflation which has been attributed to a number of factors such as the war in Ukraine, supply chain disruptions, and monetary spending. Many of these factors were a result of the COVID-19 pandemic and subsequent government stimulus packages.

Economist point to three main drivers of inflation, the first of which is demand-pull inflation where it is theorized that excess consumer demand leads to companies raising their prices and thus increasing inflation. Next is cost-push inflation which says that when the cost for raw and intermediate goods increases then that pushes companies to raise their prices and pass the cost onto consumers. Finally, monetary policy affects inflation where increases in the money supply reduce the value of the dollar meaning consumers can buy less with the same amount of money, effectively increasing prices and inflation.

Analysis Plan

The project will use a variety of variables that correspond to the three different explanations for inflation as well as other economic indicators like unemployment. Each variable will be on a monthly interval, cleaned, and then merged to create single data set. After merging the variables, data analysis will be performed to explore the characteristics of each variable as well as their relationship with inflation. From there it is time to model the data using linear regression and then the assumptions for the model will be verified. Due to the high likelihood of autocorrelation given this project is working with time series data, two other methods will be applied as well, that being the ARIMA model and the NNAR model. Combinations of variables will be tested with these models and then they will be compared based on prediction error from a subset of the data. The best models will be ensembled and then used to forecast inflation for the next twelve months.

Project Data

Data Sources

The data was collected from two sources, with each variable described coming from its own file. The first source was the United States Bureau of Labor Statistics, and the data was taken from its website, https://bls.gov/. Due to the nature of the data retrieval tool, direct links to the data cannot be provided.

To begin with, the inflation rate was retrieved as the 12-month percent change of the inflation rate for all goods for urban consumers between 1950 and 2022. Unemployment was retrieved as the original data values from 1948 to 2022. WTI crude oil prices and all commodities price index were retrieved as original data values from 1947 to 2022 then transformed to 12-month percent change.

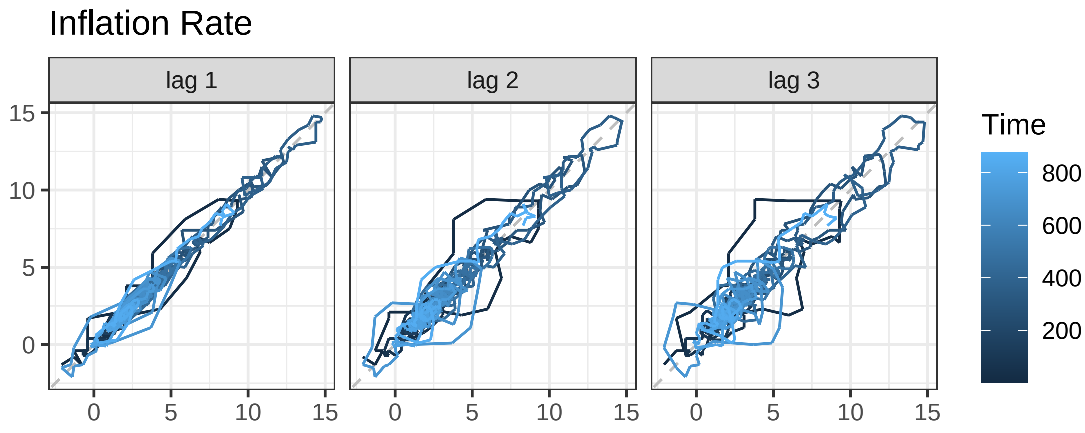
The next source of data was the St. Louis Federal Reserve's economic data at https://fred.stlouisfed.org/. The monthly federal funds rate between 1955 and 2022, the 12-month percent change in M2 money supply from 1948 to 2022, and the consumer sentiment index from 1952 to 2022 were retrieved from this source. It is worth noting that the University of Michigan created and maintains the consumer sentiment index.

The following table provides summary statistics for each variable:

Variable	Minimum	Median	Mean	Maximum	Standard Deviation
Inflation	-2.10%	2.90%	3.60%	14.80%	2.82%
Federal Funds	0.05%	4.16%	4.63%	19.10%	3.63%
Money Supply	-1.08%	7.06%	6.70%	16.43%	3.05%
Unemployment	3.40%	5.60%	5.90%	14.70%	1.66%
Sentiment	50.00	90.45	86.86	112.00	12.15
Commodities	-16.06%	2.68%	3.40%	23.44%	5.50%
Oil	-43.34%	0.00%	0.77%	134.57%	8.88%

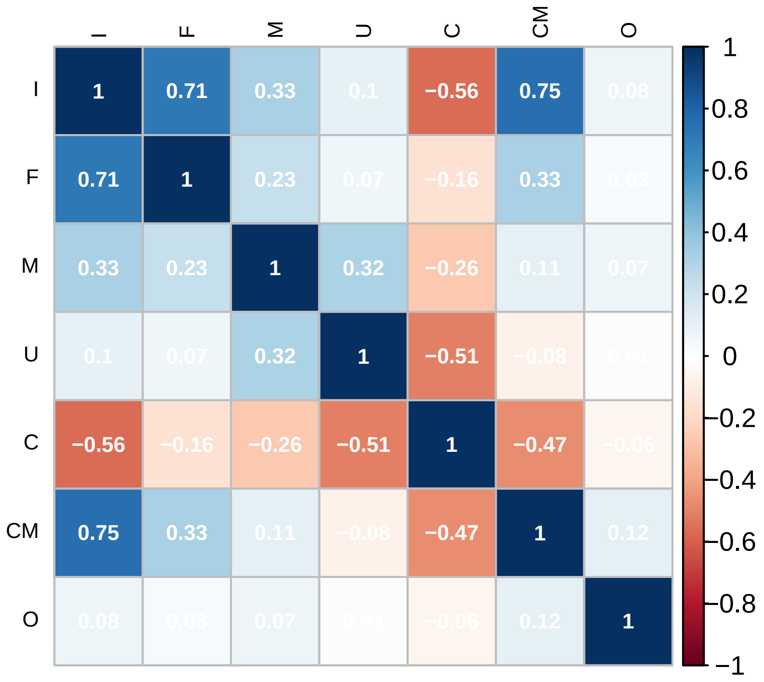
Inflation Serial Correlation

It is worth looking for serial correlation in the response variable. From the lag plot below with three lags, the inflation rate for one month tends to be quite close to the inflation rate from the next three months. This is indicated by the relatively straight diagonal line for each plot.



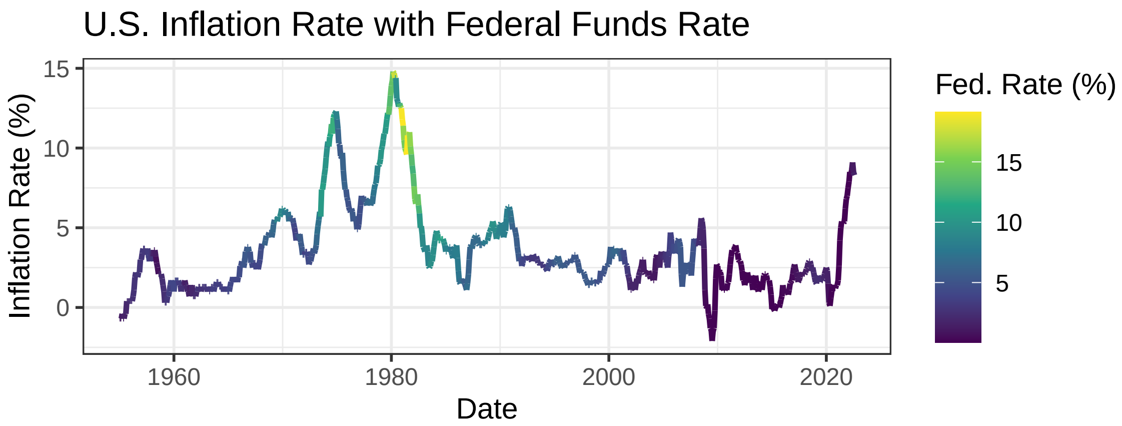
Inflation Rate Relationships

For this project, it is important to understand how inflation is related to the other variables being used. Looking at the following correlation plot, the federal funds rate and 12-month percent change in commodities prices are highly positively correlated with coefficients of 0.71 and 0.75 respectively. Consumer sentiment is moderately negatively correlated with inflation as well.



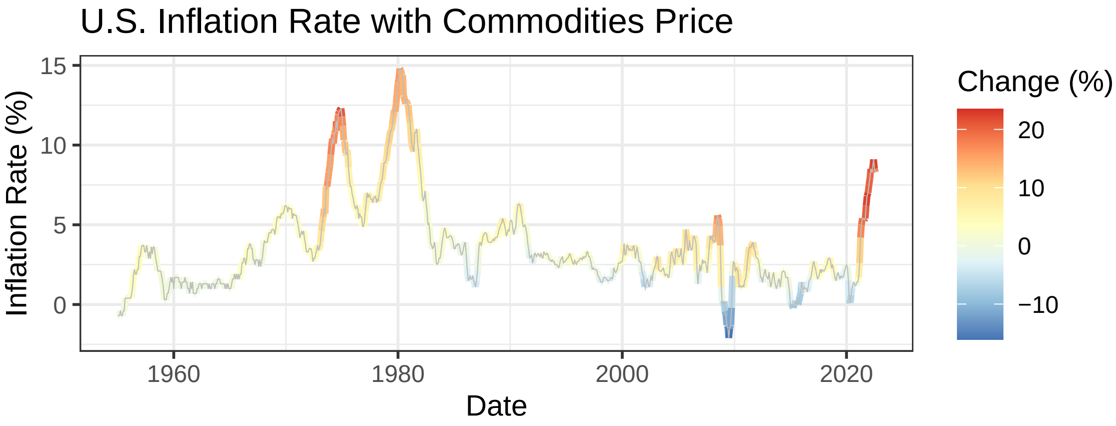
	I	F	M	U	C	CM	O
I	1	0.71	0.33	0.1	−0.56	0.75	0.08
F	0.71	1	0.23	0.07	−0.16	0.33	0.03
M	0.33	0.23	1	0.32	−0.26	0.11	0.07
U	0.1	0.07	0.32	1	−0.51	−0.08	0.01
C	−0.56	−0.16	−0.26	−0.51	1	−0.47	−0.06
CM	0.75	0.33	0.11	−0.08	−0.47	1	0.12
O	0.08	0.03	0.07	0.01	−0.06	0.12	1

The following line plot shows that the federal funds rate is distinctly lower today than it has been in the past. However, the federal funds rate does still appear to rise with inflation. This occurs because the Federal Reserve sets the federal funds rate largely in reaction to inflation and a higher federal funds rate is supposed to decrease inflation.



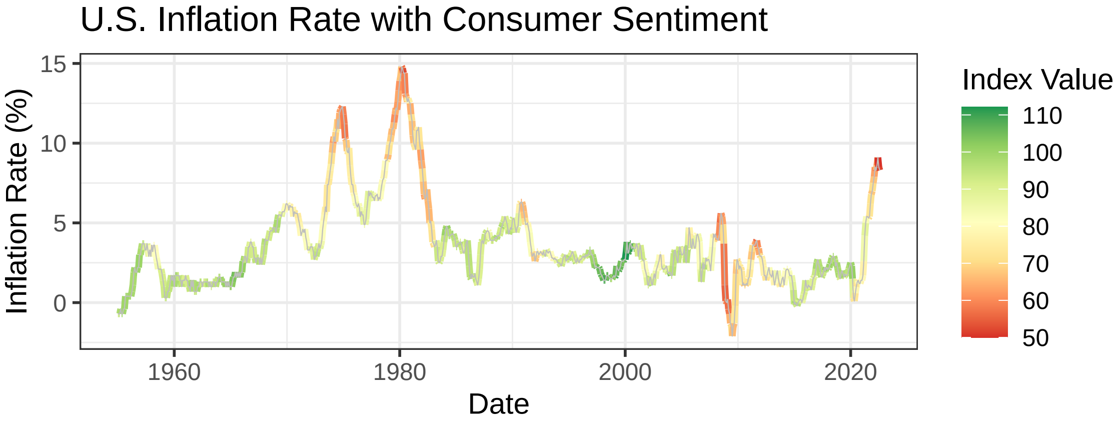
Sources: St. Louis Federal Reserve Economic Data, Bureau of Labor Statistics

Next, the commodities prices seem to simply rise and drop with the inflation rate, which given that both the inflation rate and commodities price index measure prices, this relationship makes sense.



Sources: St. Louis Federal Reserve Economic Data, Bureau of Labor Statistics

Finally, the consumer sentiment index consistently drops at the peaks and troughs of inflation. This indicates that consumer sentiment is affected by volatile inflation rather than simply just high inflation.



Sources: St. Louis Federal Reserve Economic Data, Bureau of Labor Statistics

Data Analysis and Modeling

Evaluation Criteria

To fairly evaluate the different methods tested in this project, the models were trained on data from January 1955 to July 2017. This gives five years of data to calculate prediction error for. This was determined to be a useful test set as it includes stable and volatile periods of inflation, and it is desirable that are model can handle both situations well. The following table gives more description of the evaluation criterion used.

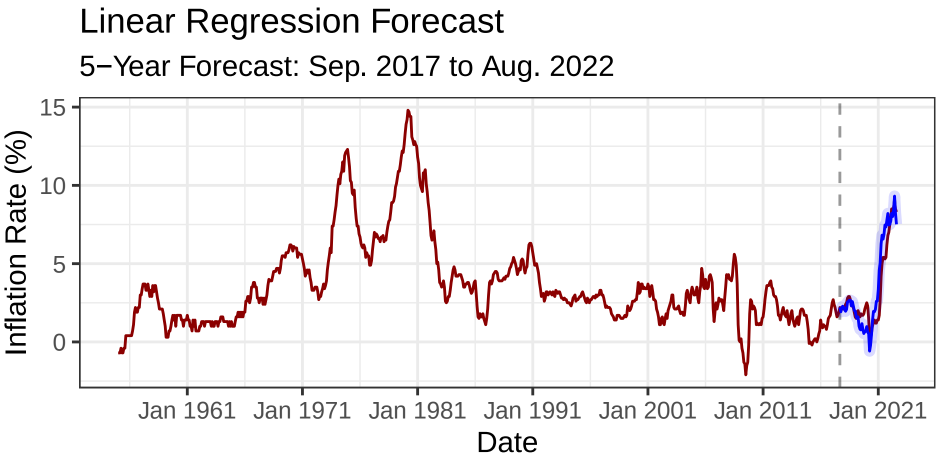
Criteria	Formula
Mean Absolute Error	$\frac{\sum_{i=1}^{n} \lvert\hat{y}_i - y_i\rvert}{n}$
Median Absolute Error	$\text{median}(\lvert\hat{y}_i - y_i\rvert)$
Max Error	$\max(\lvert\hat{y}_i - y_i\rvert)$
AIC	$2K - 2ln(L(\hat{\theta}))$
BIC	$Kln(n) - 2ln(L(\hat{\theta}))$

Linear Model

To begin modeling, a multiple linear regression is utilized with all of our variables besides year and month. The following table summarizes the results of this model:

Variable	Estimate	Standard Error	T-value	Pr(>|t|)	Significance
Intercept	6.46	0.52	12.34	$<$ 2e-16	***
Unemployment	-0.11	0.03	-3.82	0.00	***
Federal Funds	0.37	0.01	31.42	$<$ 2e-15	***
Sentiment	-0.06	0.00	-14.77	$<$ 2e-15	***
Commodities	0.24	0.01	25.32	$<$ 2e-15	***
Oil	-0.00	0.00	-0.78	0.43	
Money Supply	0.13	0.01	9.25	$<$ 2e-15	***

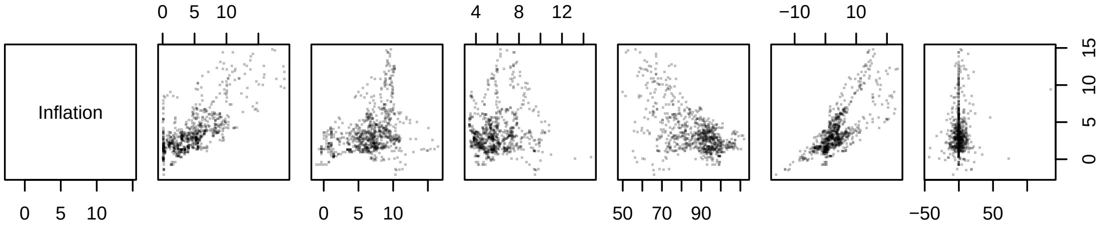
Here we see that all the variables except for crude oil prices are significant for the model. However, it will be shown later that these significance values may be spurious.



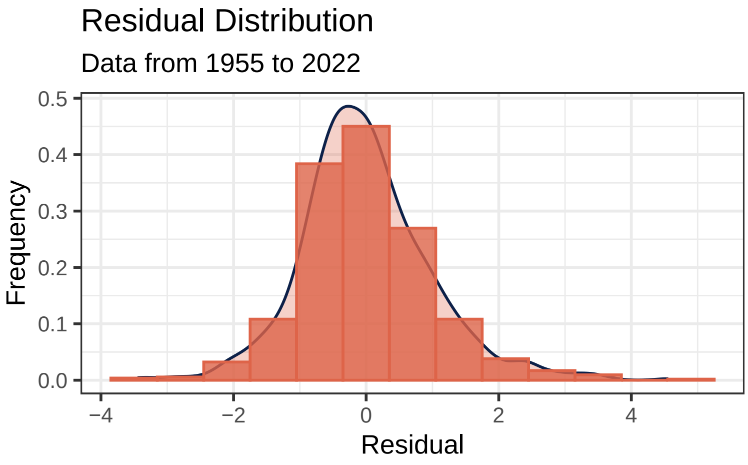
Plotting the forecasted values, it appears that the model has been able to fit the data relatively well, and this is compounded with and adjusted R^2 value of 0.8799. Regardless, we would like to verify the assumptions of this model.

Verifying Assumptions

1. Linearity The first assumption of linear regression is that a linear relationship exists between the response variable, in this case the inflation rate, and the independent variables. Looking at the scatter plots below, the federal funds rate, consumer sentiment index, and commodity price index appear to be reasonably linearly related to inflation. For the other variables, their relationship is not clearly linear.

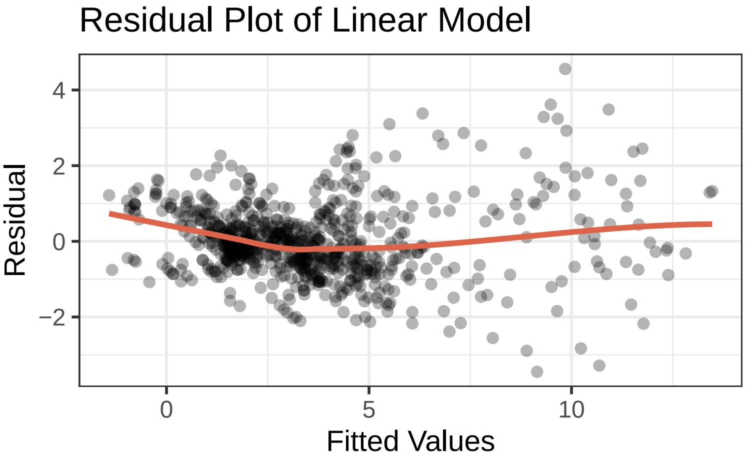


2. Normal Residuals The next assumption of linear regression is that the residuals of the model are normally distributed. This can be verified in a number of ways; first, viewing the histogram of the residuals, the residuals may be normally distributed however there does appear to be a slight positive skew in the distribution.



To be sure of this, a Shapiro test for normality and a Kolmogorov-Smirnov test were utilized with the residuals. Both tests firmly rejected the null hypothesis with p-values of effectively zero. Given that the null hypothesis for both tests is that the data is normally-distributed, these results mean that the distribution of the residuals is significantly different from the normal distribution. Therefore this assumption is violated.

3. Homoscedascity Ordinary least squares also assumes that the variance of the errors has a constant variance, and it can be seen quite clearly from the following residual plot that this is not the case. Residual variance gets larger as the predicted inflation rate gets higher. Intuitively, this means the model tends to have higher error values for more volatile inflation.



4. Uncorrelated Errors Finally, the linear model makes the assumption that the residuals are uncorrelated to each other. This is of particular concern for this project as time series data is often autocorrelated, which leads to correlated errors. Using a Durbin-Watson test for autocorrelation, the test is definitively rejected, and likewise the residuals have a high autocorrelation of 0.92.

In an effort to improve this model, step wise regression was used based on AIC. Forward and backward selection were tried, with both ways selecting the same model that used all the variables except for the crude oil price index. However, this did not seem to improve model performance substantially.

Despite the linear model seemingly being able to fit the data well, 3 out of 4 of the assumptions for linear regression are fully violated, making it unwise to go forward with this model. Therefore, the project will evaluate a few other methods that are better suited to work with autocorrelated data.

ARIMA Modeling

A natural model for autocorrelated data is the ARIMA model, and for this project, the model will be implemented using the `auto.arima` function from the `forecast` package in R.

The ARIMA model combines autoregression with differencing, where differencing refers to transforming a time series by taking the difference between consecutive observations. This leads to the ARIMA model that regresses based on past values and past forecast errors which can be complicated further to make a dynamic model that uses other time series for forecasting.

Twelve ARIMA models were tested with different combinations of variables included. The first was a model based solely on autoregression. Next, models were tested using only one of the variables. Finally, some models were tested using multiple variables. Due to the manual process of testing each model, the search was not necessarily exhaustive, however it was reasonably extensive.

The following table summarizes the results from the top three best models as well as the model without external variables:

Model #	Variables Used	Mean Absolute Error	Median Absolute Error	Max Error	AIC	BIC
13	Commodities	0.936%	0.611%	4.026%	231.238	268.209
4	All	0.948%	0.626%	3.944%	212.985	273.063
5	All w/o Oil	0.956%	0.631%	3.986%	212.326	267.783
3	None	1.742%	0.624%	7.075%	621.062	634.926

NNAR Modeling

Another useful approach is to model inflation using a neural network, in this case an NNAR model that uses autoregression. While this does drastically reduce the interpretability of the model, our primary focus is on accurate forecasting, so this is less of a concern. This will be implemented using the `nnetar` function from the `forecast` package in R.

A basic neural network consists of a set of neurons that take in input, in this case past values and the other variables of interest. Each input neuron is given a certain weight and then a non-linear function is applied to a linear combination of inputs in a hidden layer before being outputted. After this, the model continues to change the weights of each input around while minimizing some metric like mean squared error. Once this process is complete, the model is trained. The neural network used for this project is only distinguished by its use of past values of inflation to train the model, also known as autoregression.

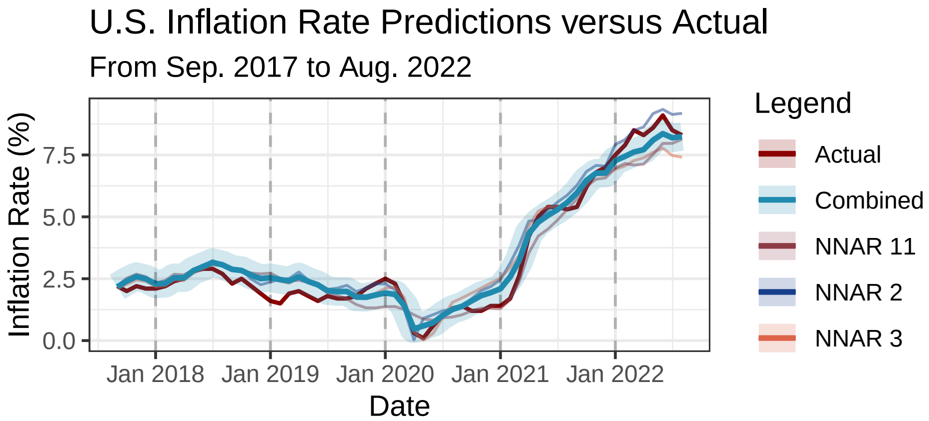
The same combinations of variables as the ARIMA models was used to train twelve different NNAR models. The following table summarizes the results from the top three best models as well as the model without external variables:

Model #	Variables Used	Mean Absolute Error	Median Absolute Error	Max Error
NNAR 3	All w/o Oil	0.406%	0.311%	1.326%
NNAR 2	All	0.408%	0.328%	1.304%
NNAR 11	Commodities	0.458%	0.310%	1.348%
NNAR 1	None	1.638%	0.855%	6.277%

Final Model Selection and Ensembling

The NNAR models performed considerably better than the linear and ARIMA models by all performance metrics. It is worth noting that the models based only on past values never performed as well as the models with external variables. This indicates that at least some of the external variables were able to improve forecasts. While testing the linear model, the oil price was not significant and a similar result is seen with the more complex models where removing oil marginally decreased or even improved performance. On the other hand, commodity prices appear to be important for forecasting inflation indicated by the variable being included in every top model.

The three best NNAR models were distinctly better than the rest of the models tested based on the prediction error, so these will be used to forecast inflation. It is common practice to average the results of multiple models to reduce forecasting error, so this project will also consider an ensemble model consisting of the average of the three models selected. Doing this, the ensemble model had a mean absolute error of 0.346% a median absolute error of 0.289%, and a max error of 0.944%. Compared to the other models, this ensemble model improves on every performance metric. The following plot shows the test forecasts of the final four models compared to the true data:



Source for Actual: St. Louis Federal Reserve Economic Data

12-Month Future Forecast

With our final models selected, it is now possible to forecast inflation into the future. The model was used to forecast 12 months into the future, however due to the data only being as recent as August 2022, this forecast will go to August 2023.

Forecasting Regressors

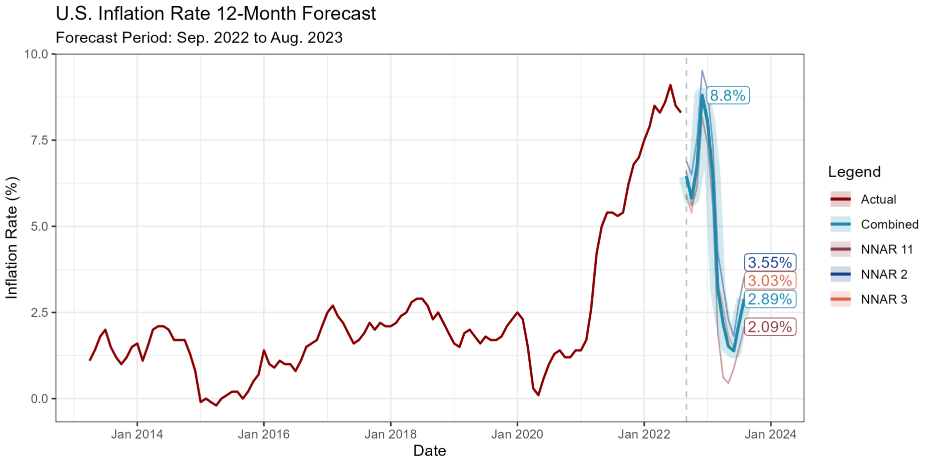
Due to our models using other variables to make predictions, we will also need to forecast these variables into the future, which adds more error into the forecast. The forecasting methodology for each variable is described below.

The federal funds rate appears to be increasing linearly right now, so we will linearly interpolate to 4.5% which is the consensus estimate of the federal funds rate by the end of 2023. Next, the M2 money supply is expected to decrease to 21310 billion so we will again linearly interpolate to this value over the course of a year and then calculate the 12-month percent change. This was computed using Excel. For unemployment, current projections by the White House for 2023 stand at 3.6% which is almost identical to the current rate of 3.5%, so again we will approximate this by linearly interpolating between these values. The EIA projects the WTI price of oil to be $88.58 in 2023, so using a similar technique to forecasting the M2 money supply, we will linearly interpolate the price and then calculate the annual percent change in price. Finally, consumer sentiment and the commodities price index do not have good third-party forecasts, so we will default to using a solely autoregressive NNAR model to forecast these variables.

Forecast Results

Finally, forecasts for inflation can be retrieved. The forecasts for August 2023 are between about 2% and 3.5%, which is close to forecasts from experts. The following table and figure summarize these forecasts:

Model #	Forecasted Inflation
NNAR 2	3.55%
NNAR 3	3.03%
NNAR 11	2.09%
Ensemble	2.89%



From this plot we also see that each model predicts a similar movement in inflation, where it is expected to drop off by March with a small rise in the following months.

Summary

The primary aim of this project of modeling inflation in the United States was accomplished with an ensemble model that averages the results of three neural network models. This model had a mean absolute error of about 0.3 meaning if the inflation rate was 1%, then the model would be expected to predict a value within 0.7% and 1.3%. The model also had a max error of just under 1%.

All variables were used to produce the final model however it appears that commodity prices and the federal funds rate were of particular importance, while oil prices were the least important for forecasting.

The initial data analysis found that commodity prices rise a drop in step with inflation while consumer sentiment drops with volatile inflation. The federal funds rate was seen to be distinctly lower today at all times however it still was positively correlated with inflation.

This project studied a limited number of variables, so future work should look at other variables that could be used to predict inflation to improve models further. A shortcoming of the model produced by this project was that the external variables also needed to be forecasted which hinges the accuracy of the model forecast on even more forecasts, adding error. Additional work could focus on finding easily forecasted variables to predict inflation as well as exploring how using autoregressive neural networks to forecast external variables impacts forecasting error for a dynamic time series model.

Appendix

Note: Error values and forecast results for NNAR models are likely to differ from those presented in this report due to the in-built randomness of the model.

```r
library(readr)
library(dplyr)
library(tidyr)
library(stringr)
library(lubridate)
library(purrr)
library(ggplot2)
library(car)
library(forecast)
library(vars)
library(gganimate)
library(psych)
library(corrplot)
library(ragg)

# Loading Data
Monthly_Inflation <- read_csv("Monthly Inflation Rate All.csv")
WTI_Oil_Prices <- read_csv("WTISPLC.csv")
FEDFUNDS <- read_csv("FEDFUNDS.csv")
Monthly_Unemployment <- read_csv("Monthly Unemployment.csv")
AllCommodities_Index <- read_csv("AllCommodities_Change.csv")
UMCSENT <- read_csv("UMCSENT.csv")
M2SL <- read_csv("CURRSL.csv")

M2SL$DATE <- mdy(M2SL$DATE)
M2SL$Year <- year(M2SL$DATE)
M2SL$Month <- month(M2SL$DATE)
M2SL <- subset(M2SL, select=-c(DATE, CURRSL))
M2SL <- drop_na(M2SL)
M2SL$M1_Change <- ts(M2SL$M1_Change * 100, start=1948, frequency = 12)

unemployment_long <- Monthly_Unemployment %>%
  gather(Month, Unemployment, Jan:Dec)
unemployment_long$Month <- match(unemployment_long$Month, month.abb)
unemployment_long <- unemployment_long %>%
  group_by(Year) %>%
  arrange(Year, Month)
unemployment_long$Unemployment <- ts(unemployment_long$Unemployment, start=1948,
↪  frequency=12)

UMCSENT$DATE <- ymd(UMCSENT$DATE)
UMCSENT$Year <- year(UMCSENT$DATE)
UMCSENT$Month <- month(UMCSENT$DATE)
UMCSENT <- subset(UMCSENT, select=-c(DATE))
UMCSENT$UMCSENT <- as.numeric(ifelse(UMCSENT$UMCSENT==".",NaN,UMCSENT$UMCSENT))
UMCSENT <- UMCSENT %>%
  mutate(UMCSENT = na.approx(UMCSENT))
UMCSENT$UMCSENT <- ts(UMCSENT$UMCSENT, start=1958, frequency = 12)
```

```r
commodities_long <- AllCommodities_Index %>%
  gather(Month, Commodities_Price, Jan:Dec)
commodities_long$Month <- match(commodities_long$Month, month.abb)
commodities_long <- commodities_long %>%
  group_by(Year) %>%
  arrange(Year, Month)
commodities_long$Commodities_Price <- ts(commodities_long$Commodities_Price * 100,
↪  start=1947, frequency=12)

FEDFUNDS$DATE <- ymd(FEDFUNDS$DATE)
FEDFUNDS$Year <- year(FEDFUNDS$DATE)
FEDFUNDS$Month <- month(FEDFUNDS$DATE)
FEDFUNDS <- subset(FEDFUNDS[FEDFUNDS$Year>1954,], select=-c(DATE))
FEDFUNDS$FEDFUNDS <- ts(FEDFUNDS$FEDFUNDS, start=1955, frequency=12)

WTI_Oil_Prices$DATE <- mdy(WTI_Oil_Prices$DATE)
WTI_Oil_Prices$Year <- year(WTI_Oil_Prices$DATE)
WTI_Oil_Prices$Month <- month(WTI_Oil_Prices$DATE)
WTI_Oil_Prices <- subset(WTI_Oil_Prices, select=-c(DATE, WTISPLC))
WTI_Oil_Prices <- WTI_Oil_Prices %>%
  group_by(Year) %>%
  arrange(Year, Month)
WTI_Oil_Prices$Oil_Change <- ts(WTI_Oil_Prices$Oil_Change * 100, start=1946,
↪  frequency=12)

inflation_long <- gather(Monthly_Inflation, Month, Rate, Jan:Dec)
inflation_long$Month <- match(inflation_long$Month, month.abb)
inflation_long <- inflation_long %>%
  group_by(Year) %>%
  arrange(Year, Month)
inflation_long$Rate <- ts(inflation_long$Rate, start=1950, frequency=12)

cpi_full <- merge(inflation_long, FEDFUNDS, by=c("Year", "Month")) %>%
  merge(M2SL, by=c("Year", "Month")) %>%
  merge(unemployment_long, by=c("Year", "Month")) %>%
  merge(UMCSENT, by=c("Year", "Month")) %>%
  merge(commodities_long, by=c("Year", "Month")) %>%
  merge(WTI_Oil_Prices, by=c("Year", "Month")) %>%
  group_by(Year) %>%
  arrange(Year, Month)

colnames(cpi_full) <- c("Year", "Month", "Inf_Rate", "Fed_Rate", "MS_Pct_Change",
↪  "Unemployment", "Consumer_Sentiment", "Oil_Price_Index", "Commodities_Index")

cpi_full$Date <- mdy(paste(cpi_full$Month, "1", cpi_full$Year, sep="/"))

# Training and Testing Sets
cpi_full_train <- cpi_full[1:(length(cpi_full$Year)-60),]
cpi_full_test <- cpi_full[(length(cpi_full$Year)-60+1):length(cpi_full$Year),]

# Full model
full_model <- lm(data=cpi_full_train,
↪  Inf_Rate~Unemployment+Fed_Rate+Consumer_Sentiment+Oil_Price_Index+Commodities_Index+MS_Pct_Change)
```

```r
summary(full_model)

lin_forecast1 <- predict(full_model, h=60, newdata=cpi_full_test)
full_model_MAE <- mean(abs(lin_forecast1-cpi_full_test$Inf_Rate))
full_model_MedAE <- median(abs(lin_forecast1-cpi_full_test$Inf_Rate))
full_model_Max_Error <- max(abs(lin_forecast1-cpi_full_test$Inf_Rate))
full_model_AIC <- AIC(full_model)
full_model_BIC <- BIC(full_model)

# Normality of Errors
shapiro.test(full_model$residuals)
ks.test(full_model$residuals, "pnorm")

durbinWatsonTest(full_model) # Highly autocorrelated

# Step Model
null_lin_fit <- lm(data=cpi_full_train, Inf_Rate~1)
step_model <- step(null_lin_fit, scope=full_model$terms, direction="forward", k=2)
summary(step_model)
lin_forecast3 <- predict(step_model, h=60, newdata=cpi_full_test)
step_model_MAE <- mean(abs(lin_forecast3-cpi_full_test$Inf_Rate))
step_model_MedAE <- median(abs(lin_forecast3-cpi_full_test$Inf_Rate))
step_model_Max_Error <- max(abs(lin_forecast3-cpi_full_test$Inf_Rate))
step_model_AIC <- AIC(step_model)
step_model_BIC <- BIC(step_model)

# ARIMA Models
arima_eval <- function(x_train=NULL, y_train, x_test=NULL, y_test, for_period=60,
↪  variables="FMUCOR"){
  if (is.null(x_train)){fit <- auto.arima(y_train)} else {fit <- auto.arima(y_train,
   ↪  xreg=as.matrix(x_train))}
  if (is.null(x_train)){fit_forecast <- forecast(fit, h=for_period)} else {fit_forecast
   ↪  <- forecast(fit, h=for_period, xreg=as.matrix(x_test))}
  print(autoplot(fit_forecast))
  checkresiduals(fit)
  pred_error <- mean(abs(fit_forecast$mean-y_test))
  med_pred_error <- median(abs(fit_forecast$mean-y_test))
  max_pred_error <- max(abs(fit_forecast$mean-y_test))
  metrics <- c("arima", variables, pred_error, med_pred_error, max_pred_error, AIC(fit),
↪  BIC(fit))
  ret_list <- list(model=fit, forecast=fit_forecast, metrics=metrics,
↪  pred_residuals=y_test-fit_forecast$mean)
  return(ret_list)
}

base_fit <- auto.arima(cpi_full_train$Inf_Rate)
summary(base_fit)
base_fit_forecast <- forecast(base_fit, h=60)
base_fit_MAE <- mean(abs(base_fit_forecast$mean-cpi_full_test$Inf_Rate))
base_fit_MedAE <- median(abs(base_fit_forecast$mean-cpi_full_test$Inf_Rate))
base_fit_MaxError <- max(abs(base_fit_forecast$mean-cpi_full_test$Inf_Rate))
base_fit_AIC <- AIC(base_fit)
base_fit_BIC <- BIC(base_fit)
```

```
# Fit 2
fit2 <- arima_eval(x_train=cpi_full_train[,4:9], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,4:9], y_test=cpi_full_test$Inf_Rate)

# Fit 3
fit3 <- arima_eval(x_train=cpi_full_train[,4:8], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,4:8], y_test=cpi_full_test$Inf_Rate, variables="FMUCO")

# Fit 4
fit4 <- arima_eval(x_train=cpi_full_train[,4:7], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,4:7], y_test=cpi_full_test$Inf_Rate, variables="FMUC")

# Fit 5
fit5 <- arima_eval(x_train=cpi_full_train[,4:6], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,4:6], y_test=cpi_full_test$Inf_Rate, variables="FMU")

# Fit 6
fit6 <- arima_eval(x_train=cpi_full_train[,4:5], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,4:5], y_test=cpi_full_test$Inf_Rate, variables="FM")

# Fit 7
fit7 <- arima_eval(x_train=cpi_full_train[,4], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,4], y_test=cpi_full_test$Inf_Rate, variables="F")

# Fit 8
fit8 <- arima_eval(x_train=cpi_full_train[,5], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,5], y_test=cpi_full_test$Inf_Rate, variables="M")

# Fit 9
fit9 <- arima_eval(x_train=cpi_full_train[,6], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,6], y_test=cpi_full_test$Inf_Rate, variables="U")

# Fit 10
fit10 <- arima_eval(x_train=cpi_full_train[,7], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,7], y_test=cpi_full_test$Inf_Rate, variables="C")

# Fit 11
fit11 <- arima_eval(x_train=cpi_full_train[,8], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,8], y_test=cpi_full_test$Inf_Rate, variables="O")

# Fit 12
fit12 <- arima_eval(x_train=cpi_full_train[,9], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,9], y_test=cpi_full_test$Inf_Rate, variables="R")

# NNAR Fit
nnetar_eval <- function(x_train=NULL, y_train, x_test=NULL, y_test, for_period=60,
↪  variables="FMUCOR"){
  if (is.null(x_train)){fit <- nnetar(y_train)} else {fit <- nnetar(y_train,
  ↪  xreg=as.matrix(x_train))}
  if (is.null(x_train)){fit_forecast <- forecast(fit, h=for_period, PI=F)} else
  ↪  {fit_forecast <- forecast(fit, h=for_period, xreg=as.matrix(x_test), PI=F)}
  print(autoplot(fit_forecast))
```

```r
  checkresiduals(fit)
  pred_error <- mean(abs(fit_forecast$mean-y_test))
  med_pred_error <- median(abs(fit_forecast$mean-y_test))
  max_pred_error <- max(abs(fit_forecast$mean-y_test))
  metrics <- c("nnetar", variables, pred_error, med_pred_error, max_pred_error,
               NA, NA)
  ret_list <- list(model=fit, forecast=fit_forecast,
                   prediction_error=c(pred_error, med_pred_error),
                   internal_error=accuracy(fit),
                   pred_residuals=y_test-fit_forecast$mean,
                   metrics=metrics)
  return(ret_list)
}

# Base neural net fit
base_fit_neu <- nnetar(cpi_full_train$Inf_Rate)
base_fit_forecast_neu <- forecast(base_fit_neu, 60, PI=F)
checkresiduals(base_fit_neu)
base_fit_neu_MAE <- mean(abs(base_fit_forecast_neu$mean-cpi_full_test$Inf_Rate))
base_fit_neu_MedAE <- median(abs(base_fit_forecast_neu$mean-cpi_full_test$Inf_Rate))
base_fit_neu_MaxError <- max(abs(base_fit_forecast_neu$mean-cpi_full_test$Inf_Rate))

# Fit 2
nfit2 <- nnetar_eval(x_train=cpi_full_train[4:9], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[4:9], y_test=cpi_full_test$Inf_Rate)

# Fit 3
nfit3 <- nnetar_eval(x_train=cpi_full_train[4:8], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[4:8], y_test=cpi_full_test$Inf_Rate, variables="FMUCO")

# Fit 4
nfit4 <- nnetar_eval(x_train=cpi_full_train[4:7], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[4:7], y_test=cpi_full_test$Inf_Rate, variables="FMUC")

# Fit 5
nfit5 <- nnetar_eval(x_train=cpi_full_train[4:6], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[4:6], y_test=cpi_full_test$Inf_Rate, variables="FMU")

# Fit 6
nfit6 <- nnetar_eval(x_train=cpi_full_train[4:5], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[4:5], y_test=cpi_full_test$Inf_Rate, variables="FM")

# Fit 7
nfit7 <- nnetar_eval(x_train=cpi_full_train[,4], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,4], y_test=cpi_full_test$Inf_Rate, variables="F")

# Fit 8
nfit8 <- nnetar_eval(x_train=cpi_full_train[,5], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,5], y_test=cpi_full_test$Inf_Rate, variables="M")

# Fit 9
nfit9 <- nnetar_eval(x_train=cpi_full_train[,6], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,6], y_test=cpi_full_test$Inf_Rate, variables="U")
```

```
# Fit 10
nfit10 <- nnetar_eval(x_train=cpi_full_train[,7], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,7], y_test=cpi_full_test$Inf_Rate, variables="C")

# Fit 11
nfit11 <- nnetar_eval(x_train=cpi_full_train[,8], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,8], y_test=cpi_full_test$Inf_Rate, variables="O")

# Fit 12
nfit12 <- nnetar_eval(x_train=cpi_full_train[,9], y_train=cpi_full_train$Inf_Rate,
↪  x_test=cpi_full_test[,9], y_test=cpi_full_test$Inf_Rate, variables="R")

# Model Scoring
model_scores_df <- data.frame(method=c("linear", "linear", "arima"),
↪  variables=c("FMUCOR", "FCOR", "none"), predicted_MAE=c(full_model_MAE,
↪  step_model_MAE, base_fit_MAE), predicted_MedAE=c(full_model_MedAE, step_model_MedAE,
↪  base_fit_MedAE), predicted_MaxError=c(full_model_Max_Error, step_model_Max_Error,
↪  base_fit_MaxError), AIC=c(full_model_AIC, step_model_AIC, base_fit_AIC),
↪  BIC=c(full_model_BIC, step_model_BIC, base_fit_BIC))

# ARIMA Fits
model_scores_df[4,] <- fit2$metrics
model_scores_df[5,] <- fit3$metrics
model_scores_df[6,] <- fit4$metrics
model_scores_df[7,] <- fit5$metrics
model_scores_df[8,] <- fit6$metrics
model_scores_df[9,] <- fit7$metrics
model_scores_df[10,] <- fit8$metrics
model_scores_df[11,] <- fit9$metrics
model_scores_df[12,] <- fit10$metrics
model_scores_df[13,] <- fit11$metrics
model_scores_df[14,] <- fit12$metrics

# NNAR Fits
model_scores_df[15,] <- c("nnetar", "none", base_fit_neu_MAE, base_fit_neu_MedAE,
↪  base_fit_neu_MaxError, NA, NA)
model_scores_df[16,] <- nfit2$metrics
checkresiduals(nfit2$model)
checkresiduals(nfit3$model)
checkresiduals(nfit11$model)
model_scores_df[17,] <- nfit3$metrics
model_scores_df[18,] <- nfit4$metrics
model_scores_df[19,] <- nfit5$metrics
model_scores_df[20,] <- nfit6$metrics
model_scores_df[21,] <- nfit7$metrics
model_scores_df[22,] <- nfit8$metrics
model_scores_df[23,] <- nfit9$metrics
model_scores_df[24,] <- nfit10$metrics
model_scores_df[25,] <- nfit11$metrics
model_scores_df[26,] <- nfit12$metrics

# Combination Forecasting
```

```r
comb_forecast <- (nfit3$forecast$mean + nfit11$forecast$mean + nfit2$forecast$mean)/3
comb_predicted_MAE <- mean(abs(comb_forecast-cpi_full_test$Inf_Rate))
comb_predicted_MedAE <- median(abs(comb_forecast-cpi_full_test$Inf_Rate))
comb_predicted_MaxError <- max(abs(comb_forecast-cpi_full_test$Inf_Rate))

model_scores_df[27,] <- c("combination", "average", comb_predicted_MAE,
↪  comb_predicted_MedAE, comb_predicted_MaxError, NA, NA)

# Test forecasts
forecasts <- data.frame(Date=cpi_full_test$Date,
                        nfit2=nfit2$forecast$mean, nfit3=nfit3$forecast$mean,
                        nfit11=nfit11$forecast$mean,
                        comb=comb_forecast, actual=cpi_full_test$Inf_Rate)

# Forecasting variables
future_fedfunds <- seq(from=FEDFUNDS$FEDFUNDS[813], to=4.5, length.out=12)
M2_Forecast <- read.csv("M2Forecast.csv")
future_M2 <- M2_Forecast$YoY.Percent.Change
future_unemployment <- seq(from=unemployment_long$Unemployment[897], to=3.6,
↪  length.out=12)
Oil_Forecast <- read.csv("WTISPLCForecast.csv")
future_oil <- Oil_Forecast$Oil_Change

sentiment_model <- nnetar(cpi_full$Consumer_Sentiment)
sentiment_forecast <- forecast(sentiment_model, h=12)
commodities_model <- nnetar(cpi_full$Commodities_Index)
commodities_forecast <- forecast(commodities_model, h=12)

# Constructing xreg data frame
future_xreg <- data.frame(Fed_Rate=future_fedfunds, MS_Pct_Change=future_M2,
                          Unemployment=future_unemployment,
                          Consumer_Sentiment=sentiment_forecast$mean,
                          Oil_Price_Index=future_oil,
                          Commodities_Index=commodities_forecast$mean)

# Train best models with all data except future values
# NNAR fit 2
final_nfit2 <- nnetar(cpi_full$Inf_Rate, xreg=as.matrix(cpi_full[,4:9]))
accuracy(final_nfit2)
final_nfit2_forecast <- forecast(final_nfit2, h=12, xreg=as.matrix(future_xreg))

# NNAR fit 3
final_nfit3 <- nnetar(cpi_full$Inf_Rate, xreg=as.matrix(cpi_full[,4:8]))
accuracy(final_nfit3)
final_nfit3_forecast <- forecast(final_nfit3, h=12, xreg=as.matrix(future_xreg[,1:5]))

# NNAR fit 11
final_nfit11 <- nnetar(cpi_full$Inf_Rate, xreg=as.matrix(cpi_full[,8]))
accuracy(final_nfit11)
final_nfit11_forecast <- forecast(final_nfit11, h=12,
↪  xreg=as.matrix(future_xreg$Oil_Price_Index))

# Combination Forecast
```

```r
final_comb_forecast <- (final_nfit2_forecast$mean + final_nfit3_forecast$mean +
↪  final_nfit11_forecast$mean)/3

future_inf_rate_forecasts <- data.frame(date=mdy(c("9/1/2022", "10/1/2022", "11/1/2022",
                                               "12/1/2022", "1/1/2023", "2/1/2023",
                                               "3/1/2023", "4/1/2023", "5/1/2023",
                                               "6/1/2023", "7/1/2023", "8/1/2023")),
                                        nfit2=final_nfit2_forecast$mean,
                                        nfit3=final_nfit3_forecast$mean,
                                        nfit11=final_nfit11_forecast$mean,
                                        combination=final_comb_forecast)
```

References

Data Sources:

- CUUR0000SA0, *12-Month Percent Change CPI for All Urban Consumers, U.S. City Average, All Goods, 1950-2022*, retrieved from data.bls.gov

- LNS14000000, *Seasonally Adjusted Unemployment Rate, 1948-2022*, retrieved from data.bls.gov

- WPU00000000, *12-Month Percent Change PPI Commodity data for All commodities, not seasonally adjusted, 1947-2022*, retrieved from data.bls.gov

- WPU0561, *PPI Commodity data for Fuels and related products and power-Crude petroleum (domestic production), not seasonally adjusted, 1946-2022*, retrieved from data.bls.gov

- Board of Governors of the Federal Reserve System (US), Federal Funds Effective Rate [FEDFUNDS], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/FEDFUNDS, November 28, 2022.

- Board of Governors of the Federal Reserve System (US), M2 [M2SL], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/M2SL, November 28, 2022.

- University of Michigan, University of Michigan: Consumer Sentiment [UMCSENT], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/UMCSENT, November 28, 2022.

Forecast Sources:

- Federal Funds Rate Forecast, https://www.bankrate.com/banking/federal-reserve/economic-indicator-survey-interest-rates-october-2022/

- Money Supply Forecast, https://www.forecasts.org/economic-indicator/m2-money-supply.htm

- Unemployment Forecast, https://www.whitehouse.gov/wp-content/uploads/2022/03/ap_2_assumptions_fy2023.pdf

- WTI Crude Oil Price Forecast, https://www.eia.gov/outlooks/steo/

Self-reflection

I devoted a significant amount of time to this project, and I really felt intrigued by the topic I chose. I wanted to explore it as thoroughly as I could with the skills and time that I had. The primary thing I learned about during this project was how to properly understand, model, and forecast time series data. I had only briefly worked with data like this, so it was a major learning experience. I had a lot of setbacks due largely to lack of background with some of the methods I implemented. I occasionally had to rework my data pipeline, and there were a lot of decisions that I switched around throughout the project. One thing I would do differently next time is to focus on getting the best data I possibly could at the start of the project rather than working with data I had easily found and just trying to fit the same the methods to better quality data. For future students, the thing that helped me the most with completing this project was getting started early. It's easy to underestimate how long a data analysis project wll take, so definitely plan for it to take more time then you might otherwise anticipate.