Problem Overview:

For this project, we used ChatGPT to build an end-to-end ML pipeline to predict inflation. Utilizing prompt engineering, we were able to get ChatGPT to produce all of the code for the pipeline, including the requirements.txt and the Dockerfile. The pipeline produces a dashboard, which incorporates ML to forecast inflation and other top economic factors, as well as provides users with an understanding of how each of these economic factors influence one another.

Introduction:

The Federal Reserve defines inflation as "an increase in the prices of goods and services over time" [1]. Inflation means that, on average, the prices of goods and services are rising, and it reduces the purchasing power of money [2]. Inflation has been a topic of concern within the United states over the past year. According to the Bureau of Labor Statistics (BLS), the Consumer Price Index (CPI) for all items increased by 7.0 percent over the 12 months ending in December 2021. This was the largest 12-month increase since June 1982 [3]. This has created a scenario where a typical American citizen is having trouble buying basic necessities like food, water, and shelter. Inflation is a multifactor issue with numerous moving parts that contribute to it.

If the rate of inflation becomes uncontrollable, the economy's status is known as hyperinflation. According to an article published in Forbes, hyperinflation is when prices rise rapidly, often at a rate of 50% or more per month [4]. This leads to a rapid loss in the purchasing power of the currency and makes it difficult for individuals, businesses, and governments to plan and budget effectively [5]. As such, hyperinflation can lead to catastrophic outcomes. Tax revenue will decrease, businesses will close, and massive food shortages occur. This also leads to general distrust in financial institutions causing the problem to worsen.

The issue of inflation needs to be approached strategically, so that policy makers stay informed and approach this issue tactfully. The intended users for this dashboard are policy makers who would like to gain further insights into the economic factors of inflation in order to decide whether or not to adjust their fiscal policies, such as government spending and taxation policies. By monitoring the dashboard regularly, policy makers can quickly respond to changes in economic conditions and adjust their policies as needed to maintain stable prices. Another use case for this dashboard could be to help policy makers identify potential risks to the economy before they become a problem. For example, if the dashboard predicts that inflation is likely to rise in the near future, policy makers could take proactive steps to address the underlying factors driving inflation, such as by reducing government spending or increasing taxes [6].

What the Dash Contains:

The first page of the dashboard allows users to select any number of economic factors they would like to learn more about. Then the users are able to select the window of months they would like to take a further look at. This allows the user to see how these different economic factors act in a general trend along with how they act with respect to one another.

Within the second page, users are able to select which factors have an effect on one another. The users are able to select two factors from the dataset. The dashboard will then display the covariance between the two factors. This effect is determined by the Vector AutoRegression Model.

The dashboard will then display the covariance with respect to each of the variables for the first selected factors, and for the second, each of the other factor's covariances with respect to each of the factors. The idea is to use a Vector auto regressive model to demonstrate how each series interacts with one another.

Pipeline Description:

The pipeline starts in the python notebooks. The read in data is explored within the Exploratory_Data_Analysis.ipynb notebook. Within the other two notebooks, the data read in, each series is standardized, the raw and scaled datasets are saved, trained, tuned, and evaluated. Within the first model forecasting notebook Final_Pipeline.ipynb, a seasonal auto fitting auto regressive integrated moving average model is trained and fit for each series within the dataset. Within the second notebook VAR_model_execution.ipynb, each column is placed within a Dickey-Fuller test to test for stationarity. Each series is differentiated until stationarity is reached. Then the first year of data is filtered from the set, and fed into the Vector Autoregressive model for training. Each model forecasts for the next 24 months. The results are then appended to the standardized dataset set and saved to the data folder. Each of the models are pickled and saved to the models folder. The dashboard then pulls from these folders to create its results.

The dashboard is created using the Streamlit framework. There are two pages within the dashboard, the time series dash which resides in the multi_select_dash.py script, and the factor effect dash which resides in the var_display.py script . Each of the pages are displayed in the full_dash.py script through a page navigator. The first page uses a multi select to allow the users to select one or more series to analyze. Two sorted select slider widgets that allow the user to select the start and end dates of the time period they would like to take a further look at. The dash will display the covariance of the first selected feature from the Vector Autoregressive model. It will then display two histograms. The first histogram shows the covariances between the first selected feature and the other features, the second histogram shows the covariances between the second selected feature and the other features.

Update functions are implemented as a part of the pipeline which resides within the utils folder. There is a function in full_dash.py that checks if it is the last day of the month. If it is the last day of the month, then the update functions run. The update functions that reside within the update_functions.py script, read in the data from FRED and check the most recent entries to ensure that there is new data. If the function detects new data to update the model with, the training and evaluation are run again, and the datasets and models are updated. The dashboard is then deployed in docker. The dockerfile contains the settings. It establishes the directory, copies the current files, installs the requirements in the requirements.txt file, then runs the full_dash file using the streamlit command.

How to Run:

1. Clone the git repository 'git clone https://github.com/AlexanderBaker44/Chat_GPT'
2. Move into the FinalPipeline folder
3. This will change depending on the scenarios:
    a. If you have Docker installed, ensure the Docker app is running (this is a workaround as the docker container is pretty large):
        i. Run the command 'docker build -t my_streamlit_app .' (include the dot at the end)
        ii. Run the command 'docker run -p 8501:8501 my_streamlit_app'
    b. If you do not have docker installed:
        i. Run the command 'pip install -r requirements.txt'
        ii. Run the command 'streamlit run full_dash.py'
4. The dashboard will then run either on your local machine or via Docker.

Sources:

[1] "Economy." Federal Reserve. https://www.federalreserve.gov/faqs/economy_14419.htm

[2] ChatGPT, response, March 11, 2023.

[3] "Consumer Price Index Summary." Bureau of Labor Statistics. https://www.bls.gov/news.release/cpi.nr0.htm

[4] Maitri Suhas, "What Is Hyperinflation? Everything You Need To Know." Forbes, December 6, 2022. https://www.forbes.com/sites/qai/2022/12/06/what-is-hyperinflation-everything-you-need-to-know/

[5] ChatGPT, response, March 11, 2023.

[6] ChatGPT, response, March 11, 2023.