CAPSTONE WHITE PAPER (MAJOR STRATEGY CONSULTING FIRM)

PREDICTING CONSUMER DEFAULTS

Shireen Aboukhalil
Rolando Chapiro Uziel
Karoline Marum
Heewon Park
Bogdan Stovba
Indu Upadhyay
Elaine Wei
Jingjing Zhang

Contents

Project Scope & Objectives

The primary goal of this capstone project was to develop a quantitative model to address the issue of predicting consumers' default patterns and its level in the near future. It seeks to be used as an early warning system to mitigate financial risk, especially during times of economic uncertainty and recession fears, as we are witnessing at present. Clients can use this model to get an indication of what default patterns and levels they should expect given the economic conditions in the present and immediate future. The client can then adjust business strategies and invest in financial assets accordingly.

The team used data on past consumer defaults, behavioral factors, policy environment, and macroeconomic indicators to construct a quantitative model that predicts the patterns and levels of consumer defaults expected across four loan categories under projected macroeconomic conditions. The model was then tested using historical data to validate its accuracy and refine it further.

Based on the predicted default patterns, the team identified future default sequences and the extent of impact on different industries. This helped in identifying a potential client from most sensitive stakeholders to pitch the final product to. The analysis was broken down into steps that can be repeated in any time period to yield default pattern projections. This was then turned into a product prototype that can be used by various clients as an early warning system to estimate expected default sequence and prepare a best response action plan. This product provides clients with a clear understanding of the expected default patterns based on economic indicators, policy context and consumer behavior. This knowledge can be used to prepare for potential financial risks, minimize losses, and reduce the impact on their business. The generalizability and ease of application made the model more commercially viable.

Key Takeaways from Previous Research

Extensive literature on the traditional consumer default hierarchy during recessions has revealed that most commonly if consumers run into liquidity problems they first stop paying credit card bills, then auto loans and only in dire crises resort to defaulting on mortgages.

However, more recent research shows that the traditional default hierarchy did not hold true during the 2008 financial crisis. High mortgage defaults persisted while bankcard and auto loan delinquency rates were well controlled. This resulted in a significant shift in the way consumers prioritized their debt payments.

During the 2008 financial crisis, many consumers chose to pay their credit card bills to preserve liquidity, even if it meant falling behind on their mortgage payments and eventually facing foreclosure. This phenomenon is known as strategic default, where consumers make calculated decisions on which debts to pay and which to default on, based on their perceived importance and potential consequences.

Moreover, non-financial data such as utility and telecom payment history has been found to be predictive of future credit delinquencies with high predictive power, even when controlling for traditional credit scores. This suggests that non-financial data can provide valuable insights into consumer behavior and help predict default patterns better. Further, in the 2008 crisis we have also seen that policies easing the consequences of defaulting such as bailouts acted as an incentive to default more on mortgages. Therefore, newer prediction models must account for behavioral and policy factors in addition to economic and financial factors.

To understand where the consumer default sequence rests currently, we looked at a 2022 Australian consumer survey which highlighted the hierarchy of financial products that consumers prioritize when they feel the pinch. According to the survey, most consumers prioritize paying their utility bills, followed by their

mortgage or rent payments, and then their credit card bills. This suggests that consumers place greater importance on basic necessities such as utilities and shelter, over unsecured debts such as credit cards.

Hence, understanding the shifting dynamics of consumer debt payments and the impact of non-financial data on predicting default patterns can provide valuable insights to financial institutions. Additionally, understanding the hierarchy of financial products most valuable to consumers during times of financial stress can aid in developing effective strategies to mitigate risks and minimize losses.

Project Approach and Methodology

The project approach and methodology involved two main steps. In Step 1, the team used input indicators to predict expected changes in levels of default and expected sequence of defaults using best fitting lags across various loan categories. The input indicators consisted of macroeconomic factors such as inflation, unemployment, GDP growth, and consumer spending, as well as behavioral factors such as consumer sentiment, and policy factors such as loan and utility bill forgiveness. The output defaults included credit cards, auto loans, and two categories of home mortgages (first mortgage and second mortgage).

To analyze the data, the team used several techniques: OLS regression, ridge regression and non-parametric regression. Ridge regression is a statistical method used to analyze multiple regression data that display multicollinearity, a common issue when dealing with macroeconomic indicators. Nonparametric regression, on the other hand, is a flexible approach that can capture complex relationships between variables without making assumptions about the underlying functional form of the data.

In Step 2, the team predicted the expected sequence of defaults across categories based on the ridge regression model using projected macroeconomic indicators for the future time period (2023-2024). In short, the predictions were based on analyzing historical data to identify any patterns or trends in the sequence of defaults across categories corresponding to concurrent economic indicators and then extrapolating it to the future. These predictions were further qualified using behavioral indicators such as consumer sentiment and indicating how different policies if implemented in this future period could impact consumer defaults.

The project was based on a data-driven approach that used a combination of macroeconomic, behavioral, and policy factors to predict default patterns across various loan categories. By understanding the expected sequence of defaults

across categories, the team was able to develop a commercially useful product that can be used by various clients as an early warning system to estimate expected default sequence and prepare a best response action plan.

Quantitative Model

Pursuing the objective of creating a model for predicting consumer default level across auto loans, bank cards, first mortgage, and second mortgage, several quantitative models were formulated and tested. Data availability is the key constraint on the analysis that was performed and the limited predictive power of the resulting models.

Data Overview

Initially, the team attempted to get consumer level data from the credit bureaus to conduct consumer level analysis. However, due to the timelines limitation, this approach was abandoned in favor of the readily available S&P and Experian defaults data.

Default Indices

S&P and Experian produce monthly composite consumer defaults indices and make available its components:

1. Auto[1]
2. Bank Cards[2]
3. First Mortgage[3]
4. Second Mortgage[4]

The index measures new defaults only and the index is reported as an annualized three-months moving average.

$$\frac{12*100*\sum \text{Three months of newly defaulted balances}}{\sum \text{Three months of newly bad and open good balances}}$$

[1] https://www.spglobal.com/spdji/en/indices/indicators/sp-experian-auto-default-index/#overview
[2] https://www.spglobal.com/spdji/en/indices/indicators/sp-experian-bankcard-default-index/#overview
[3] https://www.spglobal.com/spdji/en/indices/indicators/sp-experian-first-mortgage-default-index/#overview
[4] https://www.spglobal.com/spdji/en/indices/indicators/sp-experian-second-mortgage-default-index/#overview

The data methodology document is publicly available[5].

Explanatory Variables

The following explanatory variables were used for the development of the model. These variables were extracted via a Bloomberg terminal, however, these are also available from original sources::

1. Real GDP[6]
2. Interest Rate (Federal Funds Effective Rate, Percent, Monthly, Not Seasonally Adjusted[7]
3. Unemployment (Labor Force Statistics from the Current Population Survey)[8]
4. CPI[9]
5. Inflation Rate[10]
6. University of Michigan - Consumer Sentiment[11]

Additional Explanatory Variables

Besides the key explanatory variables, the team analyzed the following additional explanatory variables:

1. Debt delinquency expectations (Mean probability of not being able to make minimum debt payment over the next three months; Quarterly data)[12]
2. Flow into Early Delinquency (30+ days; Quarterly data)[13]

Data Transformation

[5] https://www.spglobal.com/spdji/en/documents/methodologies/methodology-sp-experian.pdf
[6] https://www.spglobal.com/marketintelligence/en/mi/products/us-monthly-gdp-index.html
[7] https://fred.stlouisfed.org
[8] https://data.bls.gov/pdq/SurveyOutputServlet
[9] https://www.usinflationcalculator.com/inflation/consumer-price-index-and-annual-percent-changes-from-1913-to-2008/
[10] https://www.rateinflation.com/inflation-rate/usa-historical-inflation-rate/
[11] https://fred.stlouisfed.org/series/UMCSENT
[12] https://www.newyorkfed.org/microeconomics/sce#/debtexp-1
[13]https://www.newyorkfed.org/medialibrary/Interactives/householdcredit/data/xls/HHD_C_Report_2022Q4.xlsx?sc_lang=e

To assess validity and appropriateness of different model approaches the team has performed several data transformation and analysis steps[14]:

1. *3-months Moving Average*.

   To align the macroeconomic indicators with default indicators, the team created 3-months moving averages of the macroeconomic indicators (see Appendix A for code snippet).

2. *Log and Exponential Transformation*.

   To assess whether a log or exponential transformation would help with achieving linearity, the team created log and exponential versions of the variables (see Appendix A for code snippet). Visual assessment of the pairwise scatterplots suggested that while there is evidence of not linearity, no transformation has achieved a more linear relationship (see Appendix A for scatter plot sample).

3. *Creating Lags.*

   Based on the hypothesis that it takes time for macroeconomic indicators to impact default levels, lags from 0 (no lag) to 12 months were created for all indicators (see Appendix A for code snippet).

4. *Identifying Appropriate Time Splits.*

   To identify appropriate time splits the team leverages several approaches. Firstly, the team used default patterns and economic conditions analysis to create initial splits (see Predicted Sequence section). Secondly, the team used macroeconomic indicators patterns analysis to create potential time splits (see Appendix A). Thirdly, the team fitted data into a K-means classification algorithm. The data was split into 5 groups of similar characteristics to help

[14] The steps were not necessarily performed in the same sequence as members of the quant team worked in parallel.

us in the task of estimating the defaults for the 4 indexes in the current macroeconomic climate (see Appendix A).

Importantly, testing models with different combinations of time slits results in models that fitted well the given time period but had extremely low fit when applied to another time period. To increase 'generalization' and predictive power of the model, the team decided to use 2004-2019 as the 'learning' period (i.e. the period used for the regression model calculation) and 2021-2022 as the 'testing' period. 2020 was removed as Covid period presented severe outliers.

Modeling Approaches

After completing the data transformation and identifying potential time splits, the team has created several distinct econometric models.

1. OLS Regression - Basic Model

   Using a loop, the team tried every possible combination of lags to identify the best fitting combination of macroeconomic indicators (see Appendix B).

2. OLS Regression - Model incorporating Consumer Sentiment

   The OLS model without consumer sentiment did not perform well with following the default patterns during the global financial crisis. Hence, consumer sentiment measured by the University of Michigan Consumer Sentiment Index was added to the model. This greatly improved the fit. However, the model performed poorly when using it to predict default level for 2021-2022 (see Appendix C).

3. SVR Model

   This approach consisted of using the non-parametric model of Support Vector Regression (SVR). This model took the desired index as the output

variable, then, the definition of a hyperparameter search space consisting of varying values for the C, gamma, and epsilon parameters was added. GridSearchCV is then applied to identify the optimal combination of hyperparameters. Once the best hyperparameters are determined, the SVR model is fit using those values on the entire dataset. Subsequently, predictions are made, and the model's performance is evaluated using mean squared error (MSE) and root mean squared error (RMSE) as metrics. To avoid linearity problems with the data, the kernel function for the model was set to RBF (Radial Basis Function).

4. Ridge Regression

   This modeling approach consisted in taking the explanatory variable with the best number of lags for each index, and regressing them against each other. The ridge penalty was applied to the coefficients to account for multicollinearity and correct for the variance bias tradeoff of an OLS model in the same instance.

Additionally, the team considered using the delinquency prediction (survey data) and transition to early delinquency as early indicators of defaults. However, as data is available only at a quarterly level and considering relatively low r-squared results, these indicators were not included in the models.

Models - Discussion

Given the data limitation and challenges on multicollinearity and non-linearity, the Ridge Regression and SVR are the most suitable models for estimating the consumer default levels. Importantly, the SVR model has demonstrated high predictive power for auto defaults across all levels and high predictive power of other categories of defaults for low levels. The models can be greatly improved by using consumer level data and separating customers by strata (see Avenues for Model Refinement for more details).

Predicted Sequence

The given timeframe of 2004 to 2022 was divided into four different time periods based on the combination of overlapping 1) economic shocks either from a recession or pandemic, and 2) patterns observed in default indices over time.

Time period 1: Jan 2004 - Dec 2006 [Pre GFC]

Time period 2: Jan 2007 - Dec 2012 [GFC]

Time period 3: Jan 2013 - Dec 2019 [Post GFC/ Pre Pandemic]

Time period 4: Jan 2020 - Present [Pandemic & Post Pandemic]

Running correlation analysis on the four default indices (auto, first mortgage, second mortgage, bank card) against 12 months of respective lagged values, we identified the pattern of default.

Time period 1 (2004 - 2006)

Default on *bank cards* came first, followed by *auto* after 10 months. Default on *second mortgages* came 9 months after, then *first mortgage* after another 11 months.

Time period 2 (2007 - 2012)

Default on *auto* came first in the second period, followed by *bank card* after 1 month. Two months later came default on *second mortgages,* and after 1 month from this came default on *first mortgages.*

Time period 3 (2013 - 2019)

Default on *auto* came first again in the third period, followed by *bank card* after 1 month like we saw in the 2nd time period. After 5 months came default on *first mortgages*, followed by default on *second mortgages* after 7 months.

Time period 4 (2020 - Present)

In the last time period, default on *auto* came first, and 1 month later followed default on *first mortgage.* After 1 month, we saw consumers defaulting on *second mortgages,* then *bank card* after another 1 month.

Predicted Sequence (2023 - 2024)

[0 months]		[0-1 months]		[0-3 months]		[6-9 months]
Bank Card	→	Second Mortgage	→	First Mortgage	→	Auto

Based on the default patterns seen from each time period, the four sequences were ranked. Then, we combined the sequences based on the ranks to predict the overall sequence that would be expected for 2023 - 2024. The duration of the expected lags was estimated according to the average number of months based on the most common ranks.

Policy Considerations

The team also conducted an analysis of the policy landscape following the COVID-19 pandemic to understand how various programs change consumer behavior. The two identified policies are loan forbearance and cash transfer programs, both of which are expected to reduce the likelihood of defaults in the short-term periods.

Loan forbearances are temporary suspensions of payments on federally-backed loans. During the COVID-19 pandemic, federal mortgage forbearance prevented hundreds of thousands of homes from being foreclosed.[15] By preventing loans from moving into default, consumers have additional liquidity that can be used to either increase consumption or pay down other loans. As a result, we expect that a loan forbearance program will have spillover effects in reducing default rates in other loan categories.

Policies that provide additional liquidity to consumers are also expected to reduce defaults. These policies can include expansions of the social safety net, cash transfers, and increased federal loans. Consumer choices to spend or save this money varies based on the policy design and their liquidity level, but the team expects to see a net decrease in defaults in the short-term.

As new research on the effect of COVID-19 policies are published, additional literature reviews should be conducted to update these considerations.

[15] https://www.urban.org/sites/default/files/2022-07/Normalizing%20Forbearance.pdf

Commercial Application

The Product

Our product is a dynamic framework that has been carefully developed to predict consumer defaults with as high degree of accuracy as available data allows.. The framework is designed to integrate macroeconomic factors, behavioral considerations and policies to produce comprehensive forecasts. Our team of experts has spent countless hours researching and testing various metrics to ensure that our forecasts are reliable and actionable. With this framework in place, we can help our clients make informed decisions and minimize their risk exposure.

Our hypothetical client is ALPHA, a prominent hedge fund with $10-20 billion in assets under management, based in New York. They are seeking a long/short strategy that invests in mispriced assets (assets priced based on less accurate consumer defaults predictions) to maximize long-term compound annual growth rates. Our dynamic framework can provide them with valuable insights to help them identify mispriced assets and make informed investment decisions. Our team of experts will work closely with ALPHA to ensure they are able to achieve their investment objectives while minimizing their risk exposure.

The product is an essential tool for any hedge fund or investment firm seeking a competitive advantage in today's complex financial environment. Through our dynamic framework, we can help our clients achieve their investment objectives while minimizing their risk exposure. We believe that our products will be a valuable asset to ALPHA and we look forward to working with them to help them achieve their long-term investment goals.

Applying the Model

Default risk is a key factor in determining the price and volatility of a financial instrument. Our model allows you to make better informed decisions regarding the

future profitability of your product, portfolio and investment. Identifying and predicting default patterns is a challenging task as there are numerous factors to take into consideration. Our dynamic model combines historical data, policy implementations and behavioral patterns. Incorporating a default pattern model to your investment strategy allows you to gain valuable insights into the dynamics of your portfolio, and the risk of industries you are considering for investment.

We have identified five key steps to utilizing our model. The first step is to assess the current state of the economy and assess macroeconomic predictions. Based on the client's standpoint on what direction the economy is going, we match the predictions with a previous period where the macroeconomic indicators align. Our model will generate a prediction of the default pattern. The next step is to compare the relative policy environment to assess how default patterns will be impacted. This step is an important part of the process as it affects the likelihood of the default pattern realizing. The client will be able to utilize the findings of the default pattern and lags between the defaults to make better informed decisions, minimize risk and increase efficiency.

Avenues for Model Refinement and Future Advancement

There were three avenues for model refinement that we identified which include data gathering, predictive time period matching, and improving the model's predictive power using machine learning. Under data gathering one way our model can be improved is by obtaining more granular data (consumer level data) on defaults to create a more accurate model which would also allow us to split it by different. The improvement under the predictive time period matching would be to develop a statistical model for finding the best time period for each prediction based on macroeconomics, policy, and behavioral factors. This also includes finding the time period with the highest predictive power by checking fluctuations of R-square among all possible periods. And improving the model's predictive power using machine learning. The improvement would be leveraging machine learning/AI to refine the model by training on more granular data. This would assist the model with the obstacles faced including linearity, autocorrelation, and multiple covariance.

When further considering avenues for future enhancement, two key research topics that should be incorporated include quantifying behavioral and policy impacts, and exploring further granularity by strata. Today there have been ongoing research reports on how to quantify the impact of consumer behavior and policy. These are research topics that are ongoing and once released the finding should be incorporated in the model. This type of research would allow us to further understand defaults and explore further variables that affect consumers. When exploring further granularity by strata, we would want to understand how boomers, millennials, and Gen Z consumer behavior differs regarding purchasing decisions to financial management to debt repayment. This will help us understand how the future of defaults may change over time.

Appendix

Appendix A: Data Transformation Code

Creating 3-months Moving Average

```
#Create 3-months moving average (MA) of indicators
df["interest_rate_MA"]=df["interest_rate"].rolling(window=3).mean()
df["unemployment_rate_MA"]=df["unemployment_rate"].rolling(window=3).mean()
df["inflation_MA"]=df["inflation"].rolling(window=3).mean()
df["real_gdp_index_MA"]=df["real_gdp_index"].rolling(window=3).mean()
df["CPI_Level_MA"]=df["CPI_Level"].rolling(window=3).mean()
df["UMCSENT_MA"]=df["UMCSENT"].rolling(window=3).mean()
```

Create log and exponential versions of the variables

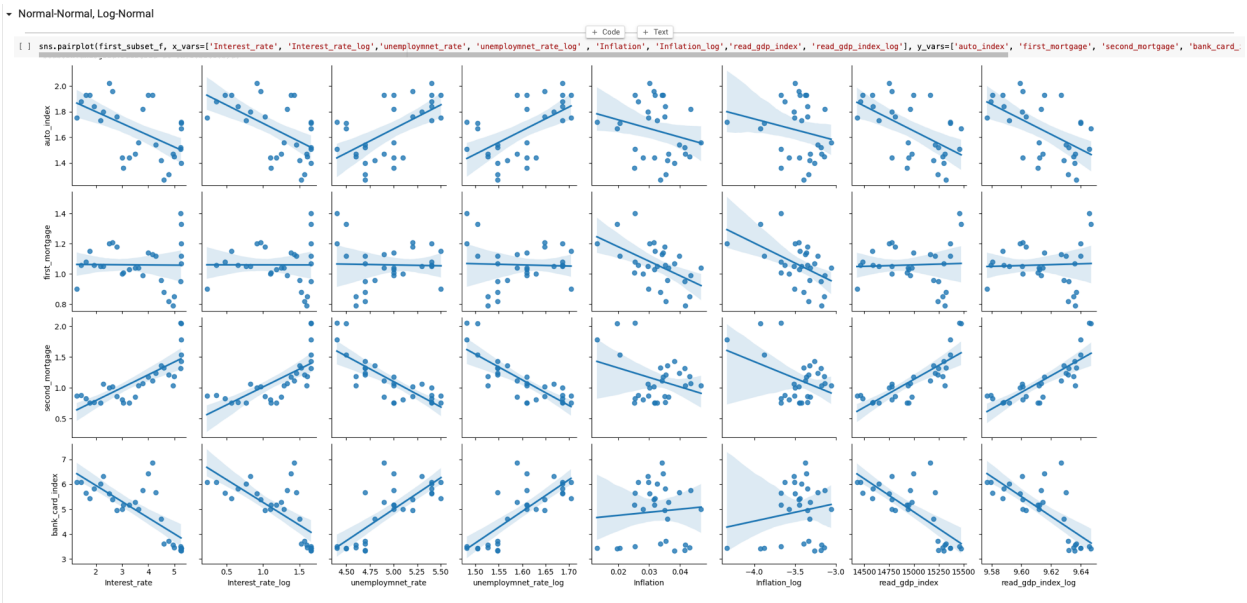
```
# Creating log transformations
df["auto_index_log"]=np.log(df["auto_index"])
df["first_mortgage_log"]=np.log(df["first_mortgage"])
df["second_mortgage_log"]=np.log(df["second_mortgage"])
df["bank_card_index_log"]=np.log(df["bank_card_index"])
df["Interest_rate_log"]=np.log(df["Interest_rate"])
df["unemploymnet_rate_log"]=np.log(df["unemploymnet_rate"])
df["Inflation_log"]=np.log(df["Inflation"])
df["read_gdp_index_log"]=np.log(df["read_gdp_index"])
df.head()
```

```
# Creating exponential transformations
df["auto_index_e"]=np.exp(df["auto_index"])
df["first_mortgage_e"]=np.exp(df["first_mortgage"])
df["second_mortgage_e"]=np.exp(df["second_mortgage"])
df["bank_card_index_e"]=np.exp(df["bank_card_index"])
df["Interest_rate_e"]=np.exp(df["Interest_rate"])
df["unemploymnet_rate_e"]=np.exp(df["unemploymnet_rate"])
df["Inflation_e"]=np.exp(df["Inflation"])
df["read_gdp_index_e"]=np.exp(df["read_gdp_index"])
df.head()
```

Estimating time splits using K-distance

```
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

df['date'] = pd.to_datetime(df['date'])

# Standardize the data
scaler = StandardScaler()
scaled_data = scaler.fit_transform(df.iloc[:, 1:])  # Assuming the first column is the date

# Choose the optimal number of clusters based on your analysis
n_clusters = 5

# Perform K-means clustering
kmeans = KMeans(n_clusters=n_clusters, init='k-means++', random_state=42)
cluster_labels = kmeans.fit_predict(scaled_data)

# Add the cluster labels to the original DataFrame
df['cluster'] = cluster_labels

# Extract the year from the 'Date' column and add it as a new column
df['year'] = df['date'].dt.year

# Create a pivot table to count the number of observations per year for each cluster
pivot_table = df.pivot_table(index='year', columns='cluster', values='date', aggfunc='count')

# Fill NaN values with 0 and convert them to integers
pivot_table = pivot_table.fillna(0).astype(int)

# Print the pivot table
print(pivot_table)

# Add the cluster labels to the original DataFrame
df['cluster'] = cluster_labels
```

Pairwise Scatterplots



Creating Lags

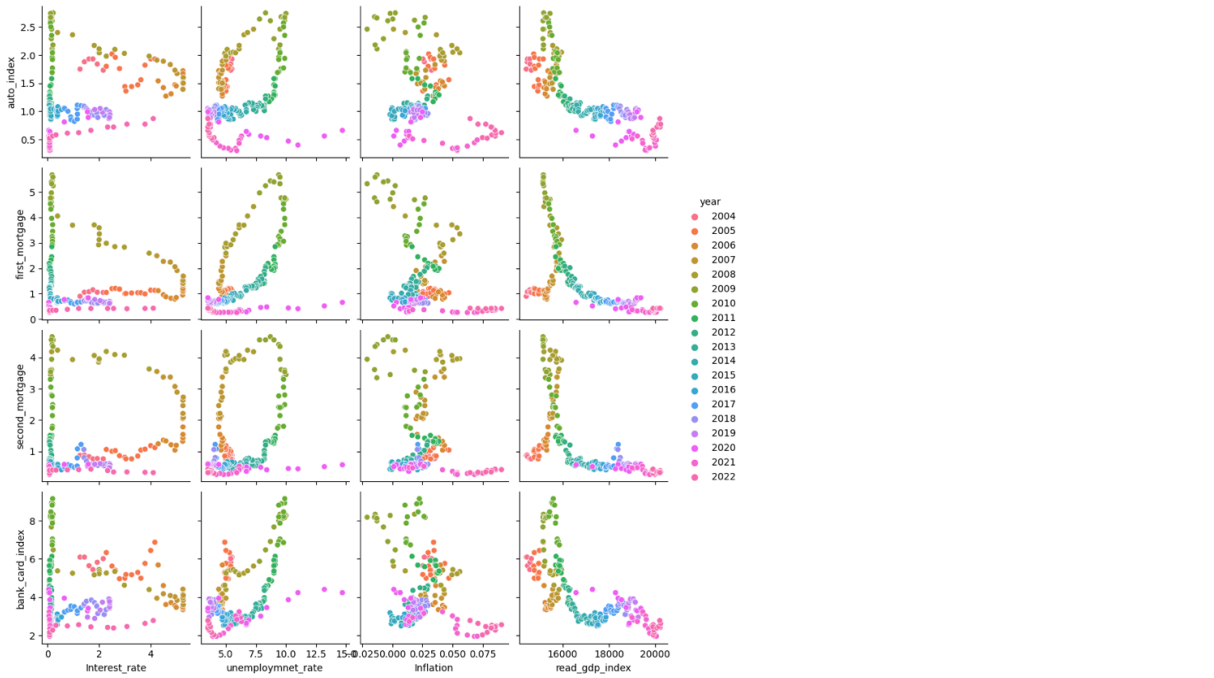
```
[ ] # Creating lags
    for i in range(0, 13):
        for col in ['interest_rate_MA', 'unemployment_rate_MA', 'inflation_MA', 'real_gdp_index_MA','CPI_Level_MA', 'UMCSENT_MA']:
            df[col+'_lag_'+str(i)] = df[col].shift(i)
```

Split based on Macroeconomic Indicators

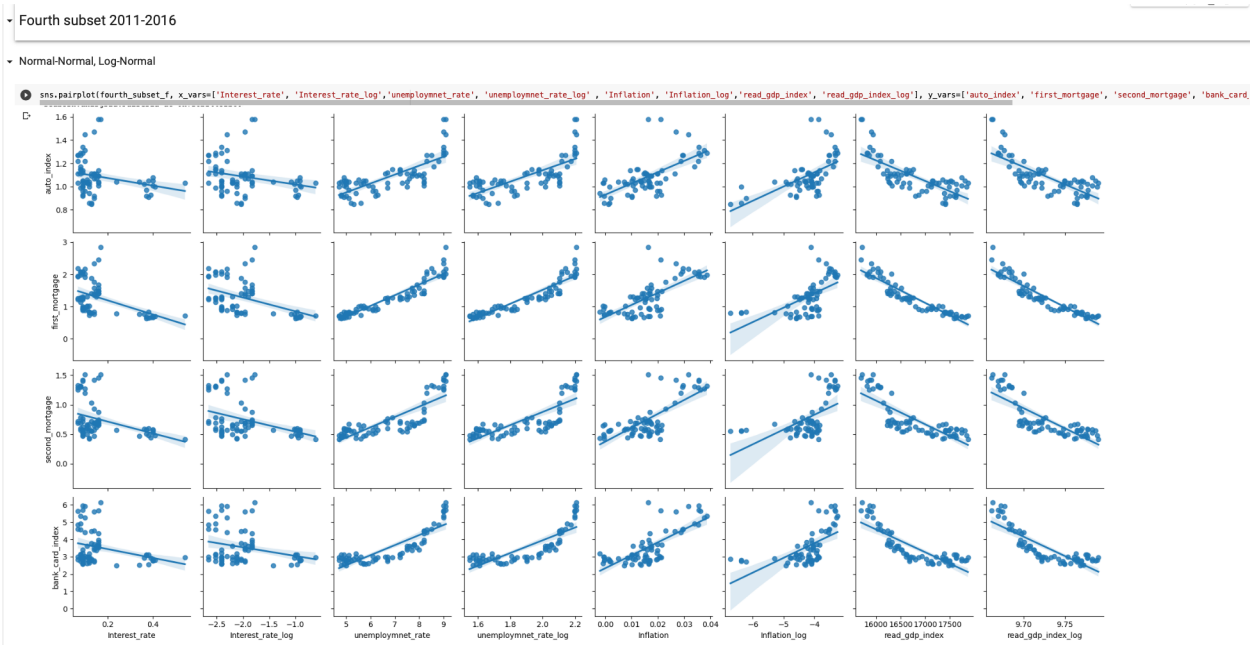
Step 1: Plot all years of macroeconomic indicators

```
sns.pairplot(df, x_vars=['Interest_rate', 'unemploymnet_rate', 'Inflation', 'read_gdp_index'], y_vars=['auto_index', 'first_mortgage', 'second_mortgage', 'bank_card_index'], hue='year', diag_kind=None)
```

```
<seaborn.axisgrid.PairGrid at 0x7f33e774aa90>
```



Step 2: Split based on similar patterns (below is the sample from subset four 2011-2016)



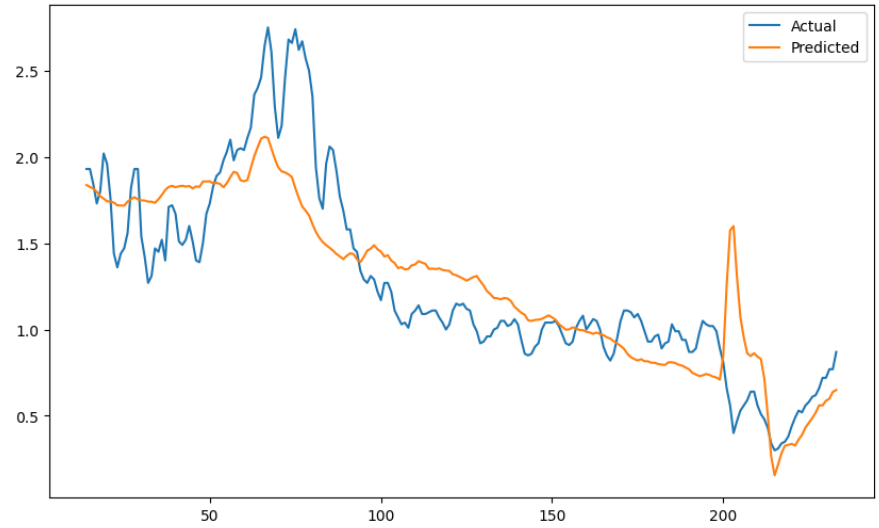
Appendix B: Basic OLS Model (without UMCSENT)

Sample Code Snippet

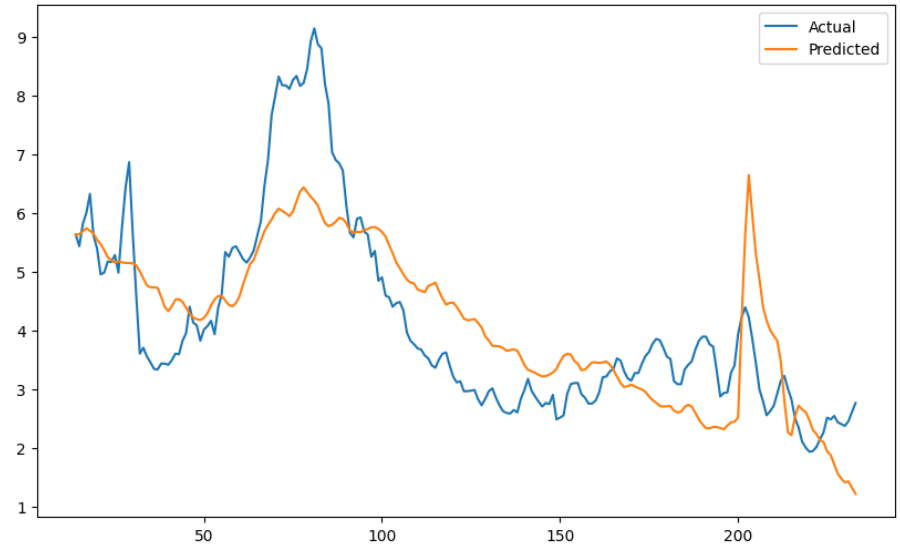
```
# Define the y variable
y = first_subset_f['auto_index']

# Define the x variables (the lags of the four variables)
x_cols = [col for col in first_subset_f.columns if 'lag' in col]
x = first_subset_f[x_cols]

# Create all possible combinations of lags for each variable
lag_combinations = list(itertools.product(range(0, 13), repeat=4))

# Fit a linear regression model for each combination of lags and record the R-squared
best_r_squared = 0
best_lags = None
r_squared_list = []
for lags in lag_combinations:
    # Select the columns with the corresponding lags for each variable
    x_lagged = x[[col+'_lag_'+str(lag) for col, lag in zip(['interest_rate_MA', 'unemployment_rate_MA', 'CPI_Level_MA', 'real_gdp_index_MA'], lags)]]

    # Fit a linear regression model
    model = LinearRegression().fit(x_lagged, y)

    # Calculate the R-squared for the model
    r_squared = model.score(x_lagged, y)
    r_squared_list.append(r_squared)

    # Update the best lags and R-squared if this model has a higher R-squared than the previous best
    if r_squared > best_r_squared:
        best_r_squared = r_squared
        best_lags = lags
        best_model = model

# Print the best combination of lags and its R-squared
print('Best combination of lags:', best_lags)
print('R-squared:', best_r_squared)

# Print the list of combinations of lags and their R-squared
#for i in range(len(lag_combinations)):
    #print(f"Lags combination: {lag_combinations[i]}, R-squared: {r_squared_list[i]}")

# Print the regression equation of the best model
x_lagged_best = x[[col+'_lag_'+str(lag) for col, lag in zip(['interest_rate_MA', 'unemployment_rate_MA', 'CPI_Level_MA', 'real_gdp_index_MA'], best_lags)]]
x_lagged_best = sm.add_constant(x_lagged_best)
model_best = sm.OLS(y, x_lagged_best).fit()
print('Regression equation:', model_best.params)

# Plot the actual auto_index and the predicted values from the best model
y_pred = model_best.predict(x_lagged_best)
fig, ax = plt.subplots(figsize=(10,6))
ax.plot(y.index, y, label='Actual')
ax.plot(y.index, y_pred, label='Predicted')
ax.legend()
plt.show()
```

Results

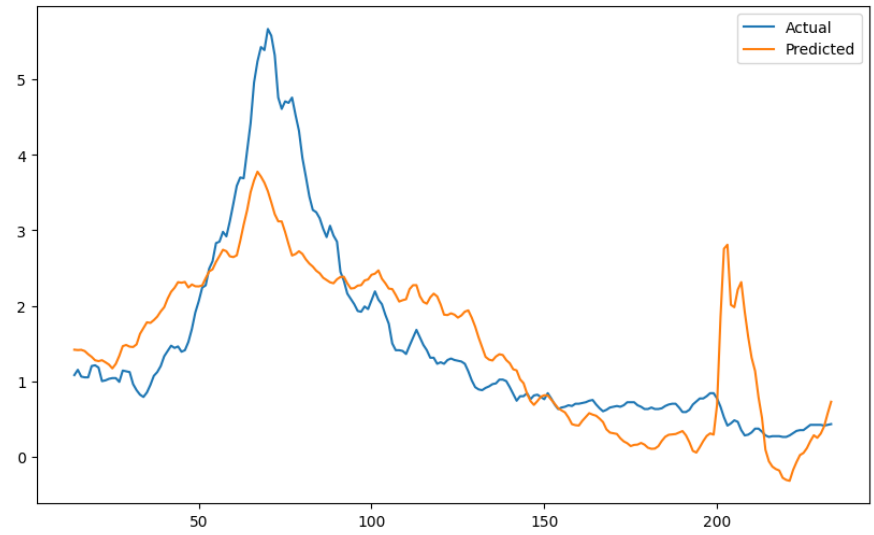
```
Best combination of lags: (12, 12, 4, 0)
R-squared: 0.6990362382475217
Regression equation: const                          6.132716
interest_rate_MA_lag_12         0.065245
unemployment_rate_MA_lag_12    -0.038225
CPI_Level_MA_lag_4              0.012963
real_gdp_index_MA_lag_0        -0.000454
dtype: float64
```



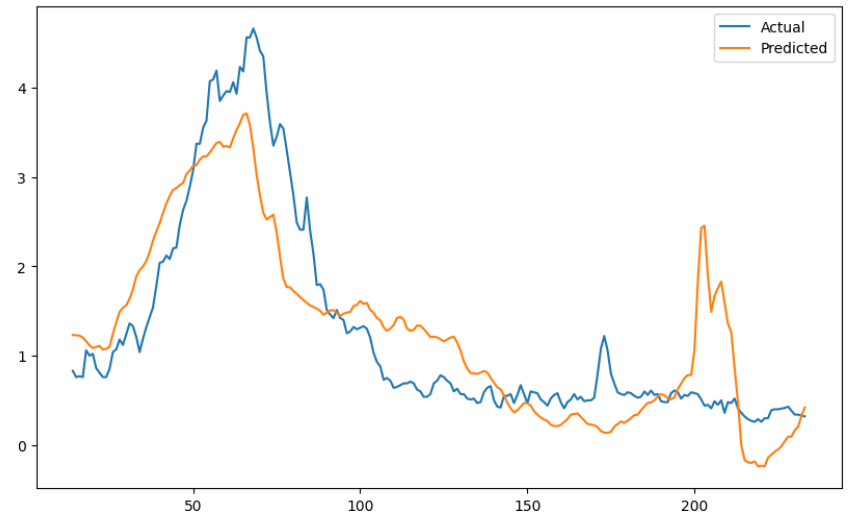

```
Best combination of lags: (5, 0, 12, 12)
R-squared: 0.6034810970390803
Regression equation: const                          8.649852
interest_rate_MA_lag_5         -0.100995
unemployment_rate_MA_lag_0      0.463842
CPI_Level_MA_lag_12            -0.088362
real_gdp_index_MA_lag_12        0.000778
dtype: float64
```



```
Best combination of lags: (12, 4, 7, 0)
R-squared: 0.6425482023000758
Regression equation: const                          5.786138
interest_rate_MA_lag_12       0.328152
unemployment_rate_MA_lag_4    0.163411
CPI_Level_MA_lag_7            0.070436
real_gdp_index_MA_lag_0      -0.001292
dtype: float64
```




```
Best combination of lags: (12, 5, 6, 0)
R-squared: 0.7218819229967379
Regression equation: const                          3.957154
interest_rate_MA_lag_12       0.536290
unemployment_rate_MA_lag_5    0.111031
CPI_Level_MA_lag_6            0.045231
real_gdp_index_MA_lag_0      -0.000853
dtype: float64
```



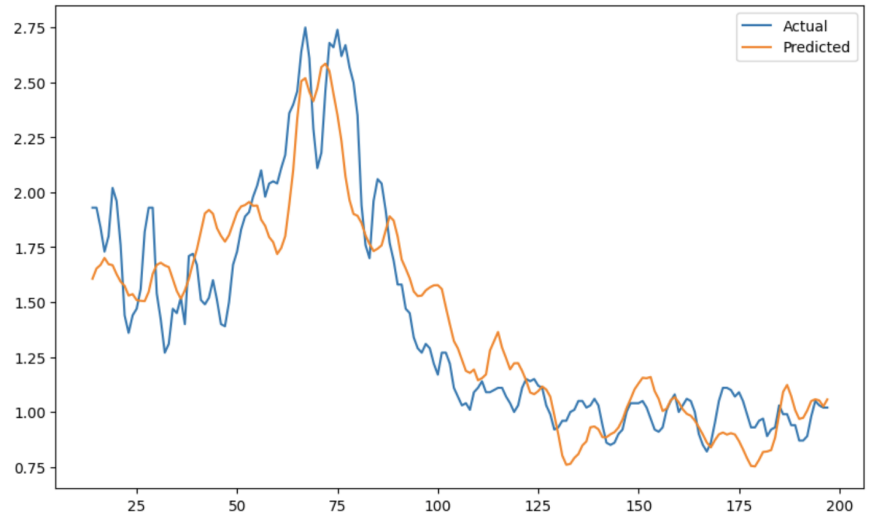
Appendix C: OLS Model with UMCSENT

Sample Code Snippet

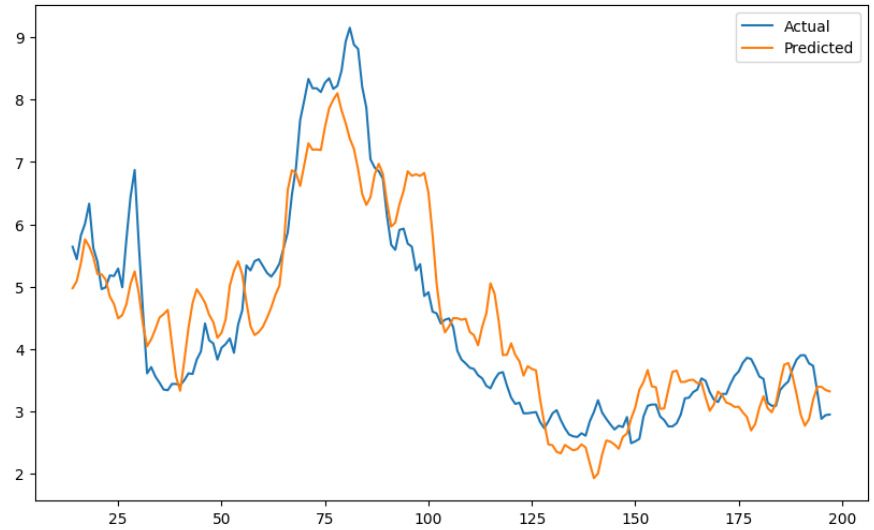
```
# Define the y variable
y = second_subset_f['auto_index']

# Define the x variables (the lags of the four variables)
x_cols = [col for col in second_subset_f.columns if 'lag' in col]
x = second_subset_f[x_cols]

# Create all possible combinations of lags for each variable
lag_combinations = list(itertools.product(range(0, 13), repeat=5))

# Fit a linear regression model for each combination of lags and record the R-squared
best_r_squared = 0
best_lags = None
r_squared_list = []
for lags in lag_combinations:
    # Select the columns with the corresponding lags for each variable
    x_lagged = x[[col+'_lag_'+str(lag) for col, lag in zip(['interest_rate_MA', 'unemployment_rate_MA', 'CPI_Level_MA', 'real_gdp_index_MA','UMCSENT_MA'], lags)]]

    # Fit a linear regression model
    model = LinearRegression().fit(x_lagged, y)

    # Calculate the R-squared for the model
    r_squared = model.score(x_lagged, y)
    r_squared_list.append(r_squared)

    # Update the best lags and R-squared if this model has a higher R-squared than the previous best
    if r_squared > best_r_squared:
        best_r_squared = r_squared
        best_lags = lags
        best_model = model

# Print the best combination of lags and its R-squared
print('Best combination of lags:', best_lags)
print('R-squared:', best_r_squared)

# Print the list of combinations of lags and their R-squared
#for i in range(len(lag_combinations)):
    #print(f"Lags combination: {lag_combinations[i]}, R-squared: {r_squared_list[i]}")

# Print the regression equation of the best model
x_lagged_best = x[[col+'_lag_'+str(lag) for col, lag in zip(['interest_rate_MA', 'unemployment_rate_MA', 'CPI_Level_MA', 'real_gdp_index_MA', 'UMCSENT_MA'], best_lags)]]
x_lagged_best = sm.add_constant(x_lagged_best)
model_best = sm.OLS(y, x_lagged_best).fit()
print('Regression equation:', model_best.params)

# Plot the actual auto_index and the predicted values from the best model
y_pred = model_best.predict(x_lagged_best)
fig, ax = plt.subplots(figsize=(10,6))
ax.plot(y.index, y, label='Actual')
ax.plot(y.index, y_pred, label='Predicted')
ax.legend()
plt.show()
```

Results

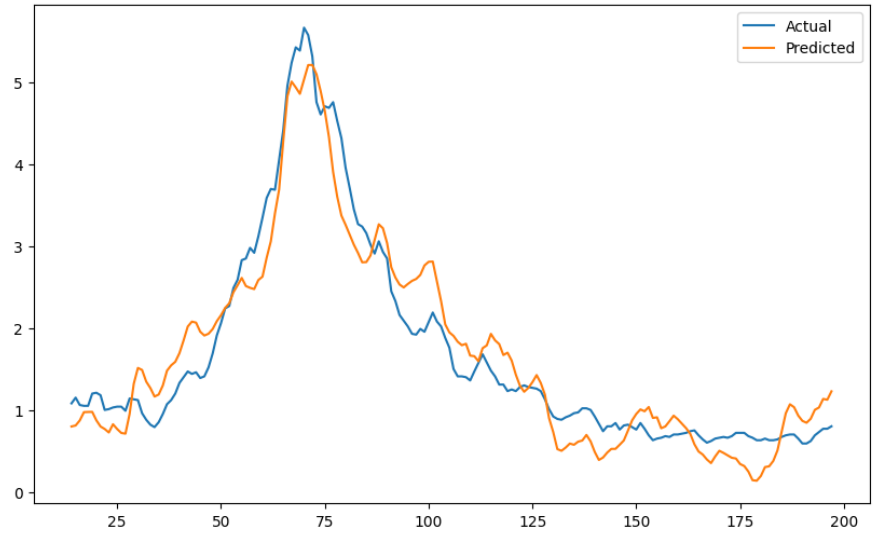
```
Best combination of lags: (12, 0, 0, 12, 9)
R-squared: 0.8299898427541652
Regression equation: const                        -0.402653
interest_rate_MA_lag_12       0.130653
unemployment_rate_MA_lag_0    0.267581
CPI_Level_MA_lag_0           -0.054992
real_gdp_index_MA_lag_12      0.000731
UMCSENT_MA_lag_9              0.006899
dtype: float64
```



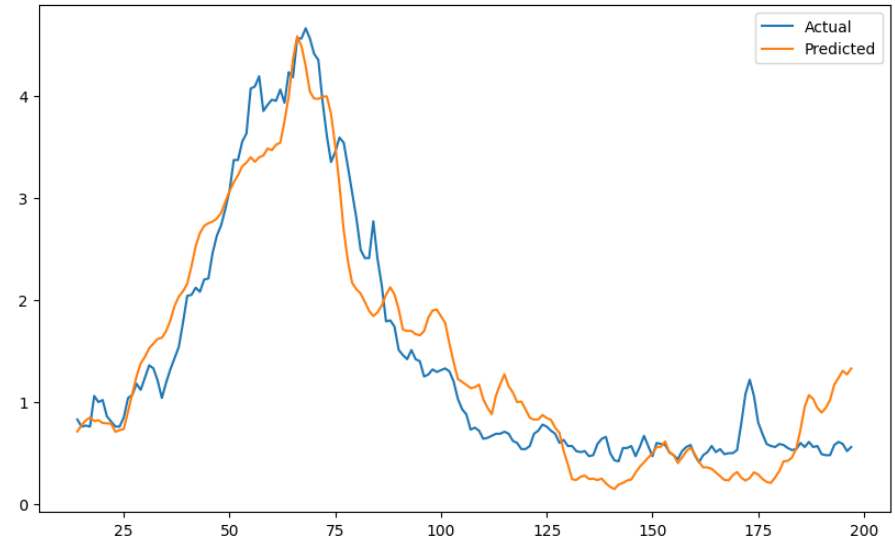

```
Best combination of lags: (12, 0, 12, 12, 4)
R-squared: 0.8012640052171578
Regression equation: const                        -16.328861
interest_rate_MA_lag_12        0.149535
unemployment_rate_MA_lag_0     1.544111
CPI_Level_MA_lag_12           -0.247772
real_gdp_index_MA_lag_12       0.003687
UMCSENT_MA_lag_4               0.074777
dtype: float64
```



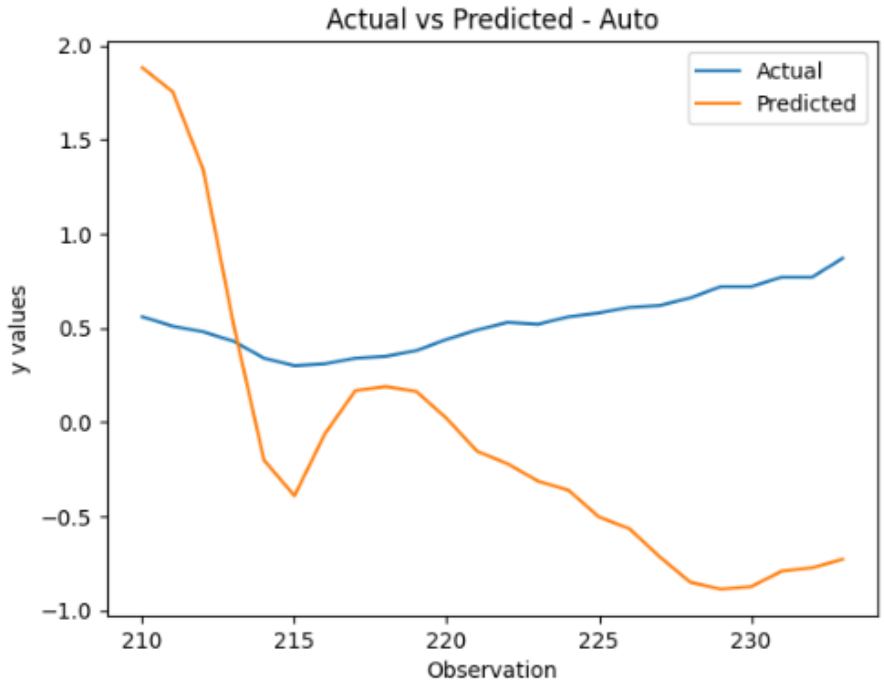
```
Best combination of lags: (12, 0, 0, 12, 2)
R-squared: 0.9158655343283544
Regression equation: const                      -6.261286
interest_rate_MA_lag_12       0.325004
unemployment_rate_MA_lag_0    0.705503
CPI_Level_MA_lag_0           -0.114815
real_gdp_index_MA_lag_12      0.001903
UMCSENT_MA_lag_2             -0.020366
dtype: float64
```

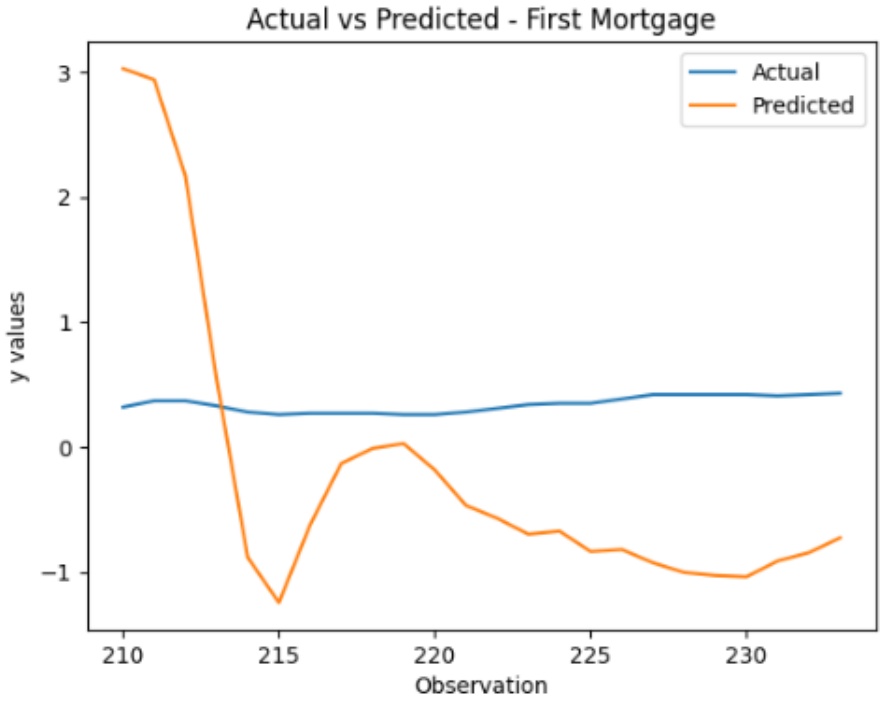


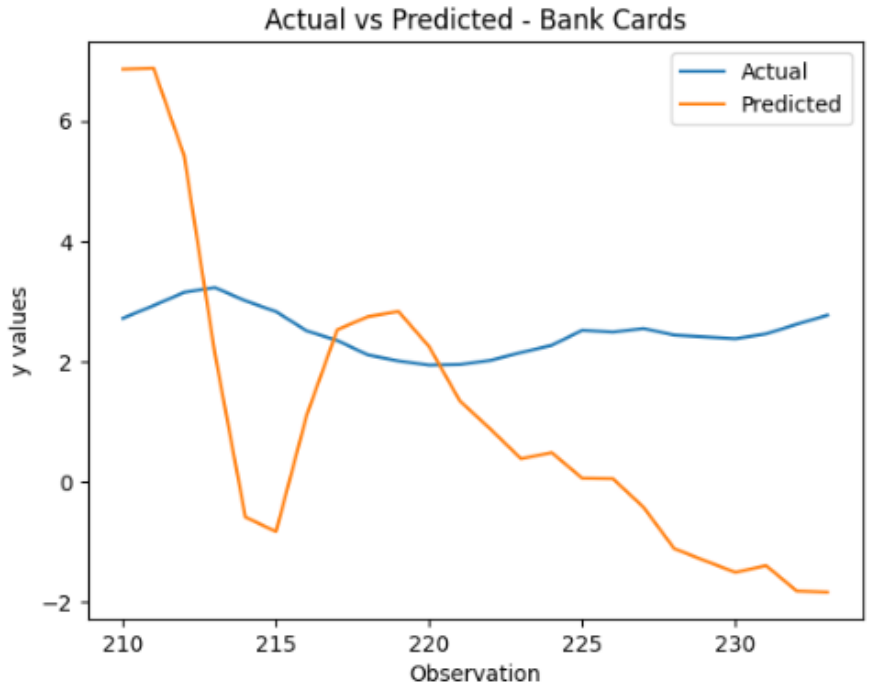

```
Best combination of lags: (12, 0, 0, 12, 0)
R-squared: 0.9014816051789531
Regression equation: const                      -4.051488
interest_rate_MA_lag_12       0.542157
unemployment_rate_MA_lag_0    0.471014
CPI_Level_MA_lag_0           -0.078011
real_gdp_index_MA_lag_12      0.001286
UMCSENT_MA_lag_0             -0.015896
dtype: float64
```

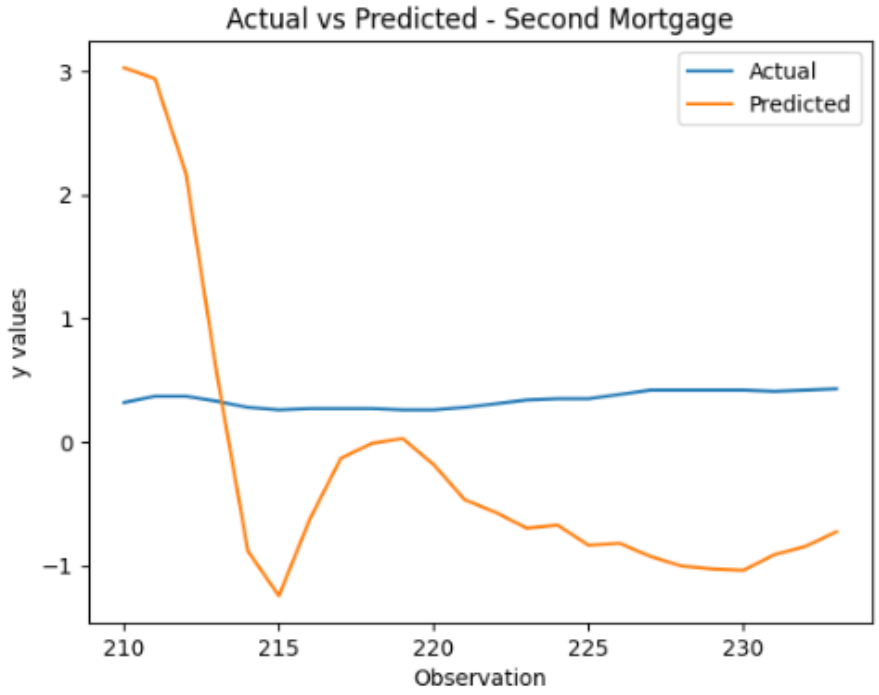


Using the models to predict defaults levels in 2021-2022


Actual vs Predicted - Auto
Actual
Predicted
y values
Observation
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
210
215
220
225
230


Actual vs Predicted - First Mortgage
Actual
Predicted
y values
Observation
3
2
1
0
−1
210
215
220
225
230


Actual vs Predicted - Bank Cards
Actual
Predicted
y values
Observation
6
4
2
0
−2
210
215
220
225
230


Actual vs Predicted - Second Mortgage
Actual
Predicted
y values
Observation
3
2
1
0
−1
210
215
220
225
230

Appendix D: SVR Model

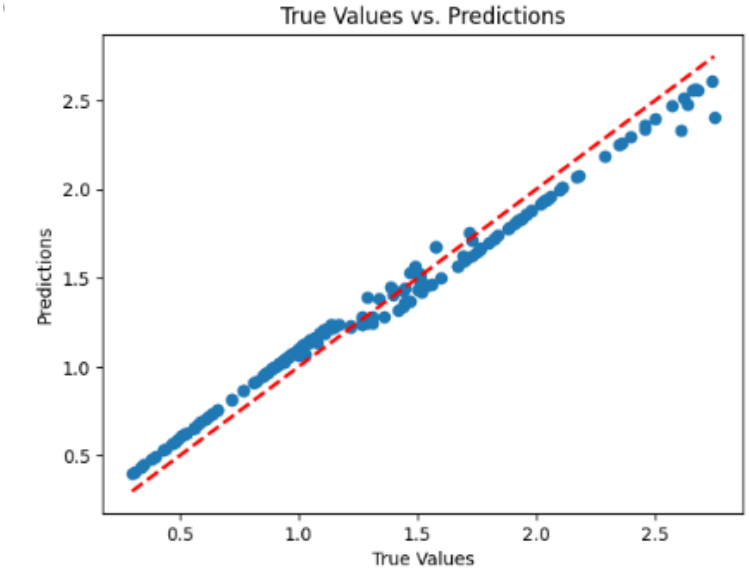
Code Snippet

```
from sklearn.metrics import mean_squared_error

svr = SVR()

# tuning the model to find optimal parameters
param_grid = {'C': [1, 5, 10, 50],
              'gamma': [0.0001, 0.0005, 0.001, 0.005],
              'epsilon': [0.1, 0.2, 0.5, 1]
              }
svm_grid = GridSearchCV(svr, param_grid).fit(X_train_scaled, y_train)
print("Model's accuracy after scaling the data")
print("Best Parameter: {}".format(svm_grid.best_params_))
print("Test set Score: {:.3f}".format(svm_grid.score(X_test_scaled, y_test)))
```

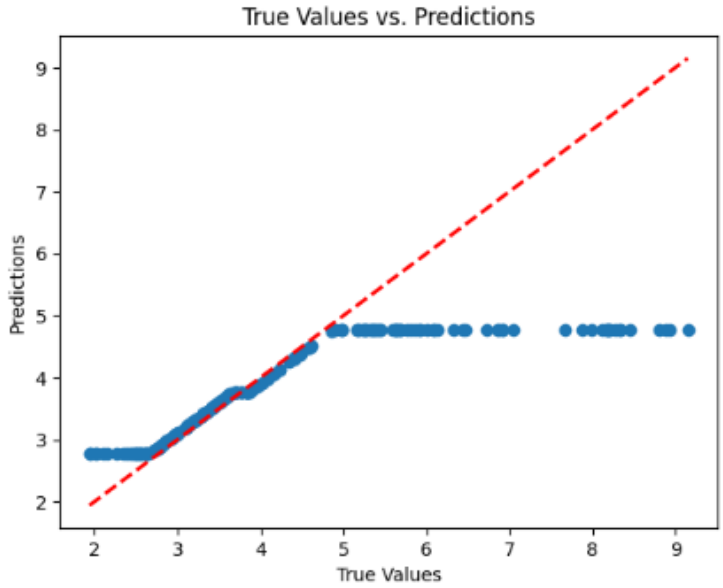
```
# Define the SVR model and hyperparameter search space
svr = SVR()
param_grid = {'C': [1, 5, 10, 50],
              'gamma': [0.0001, 0.0005, 0.001, 0.005],
              'epsilon': [0.1, 0.2, 0.5, 1]
             }

# Apply GridSearchCV
grid_search = GridSearchCV(svr, param_grid).fit(X_train_scaled, y_train)
grid_search.fit(X, Y)

# Print the best hyperparameters
print("Best hyperparameters:", grid_search.best_params_)

# Fit the SVR model with the best hyperparameters on the entire dataset
best_svr = grid_search.best_estimator_
best_svr.fit(X, Y)
rmse = sqrt(mse)

# Make predictions and evaluate the model
predictions = best_svr.predict(X)
mse = mean_squared_error(Y, predictions)
print("Mean squared error:", mse)
print("Root Mean Squared Error: ", rmse)
```

Results

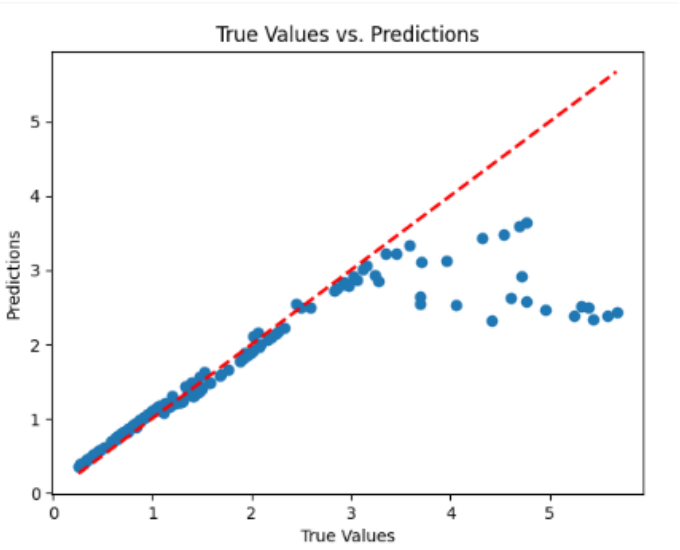
Auto Index



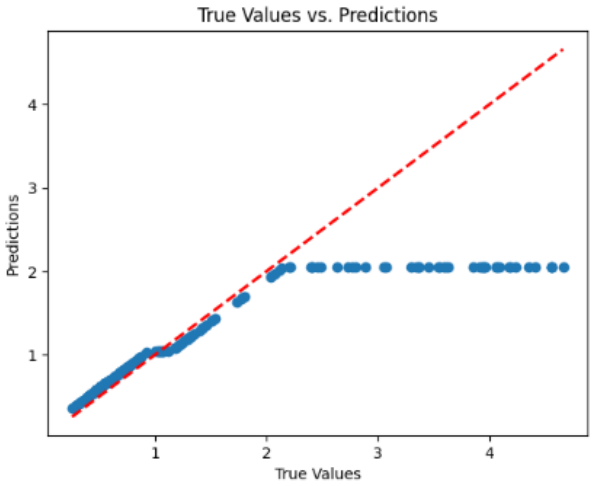
Bank Card Index



First Mortgage Index

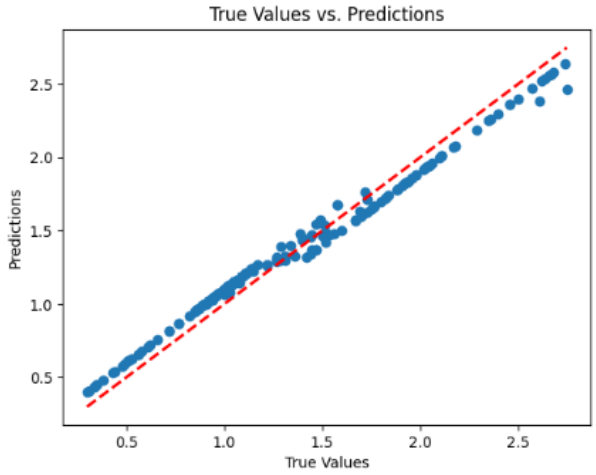


Second Mortgage Index

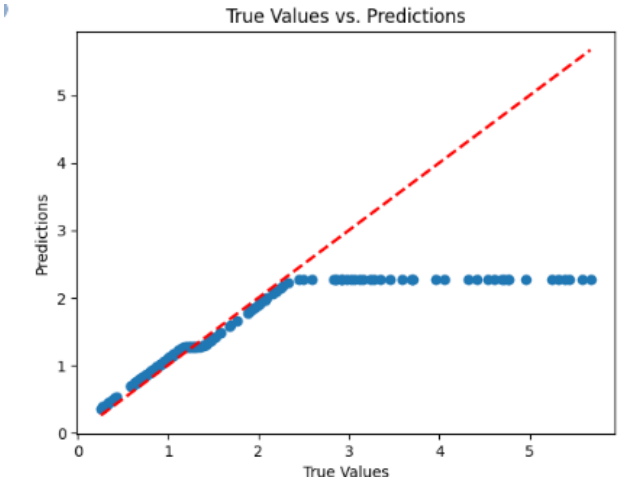


Using the models to predict defaults levels in 2021-2022

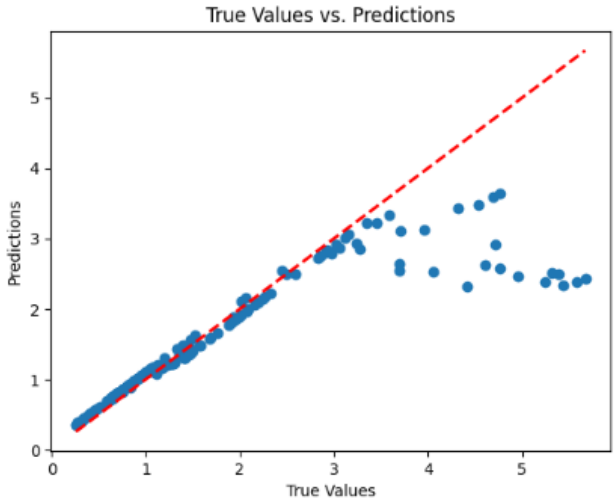
Auto Index



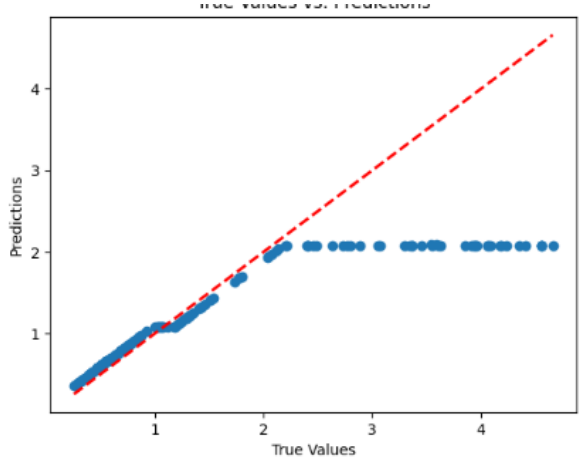
Bank Card Index



First Mortgage Index



Second Mortgage Index



Appendix E: Ridge Regression

Code Snippet

```
# Train test split the data
#X_train, X_test, y_train, y_test = train_test_split(X, Y, random_state=42)

#scale the data
scaler = preprocessing.StandardScaler()
scaler.fit(X_train)

# defineda our scaled X train and test data
X_train_scaled = scaler.transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

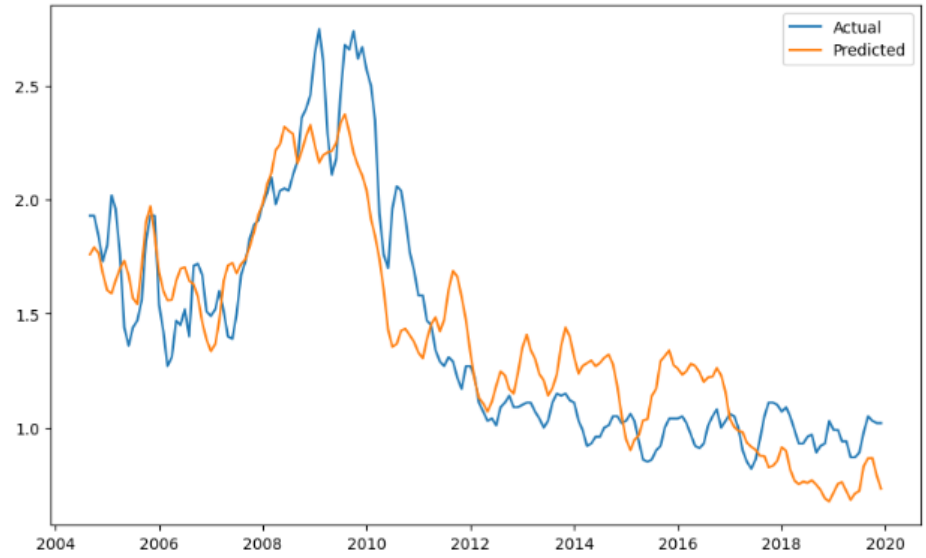
+ Code + Text

```
#Fit the regression with the parammetres and the scaled x train data
ridge_param_grid = {'alpha': [0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000]}
ridge_grid = GridSearchCV(Ridge(), ridge_param_grid).fit(X_train_scaled, y_train)
r_squared = ridge_grid.score(X_test_scaled, y_test)

#
print("RIDGE REGRESSION ")
print("Best Parameter: {}".format(ridge_grid.best_params_))
print("Best Cross-Validation Score: {:.3f}".format(ridge_grid.best_score_))
print("Test set Score: {:.3f}".format(ridge_grid.score(X_test_scaled, y_test)))
print("R-squared: {:.3f}".format(r_squared))
```

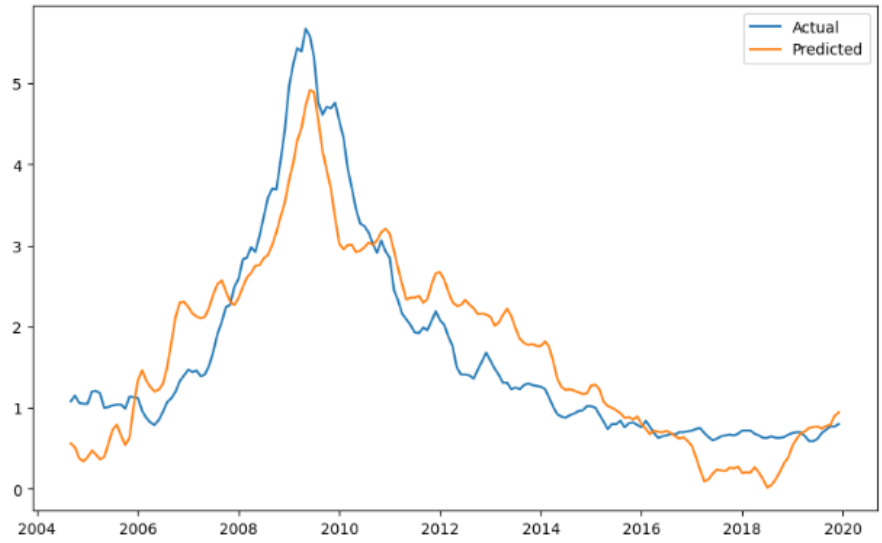
```
ridge = Ridge(alpha=10).fit(X_train_scaled, y_train) # here we manually change the alpha to 10 as it the best value obtained form the ones tested above
print("Ridge test set score: {:.3f}".format(ridge.score(X_test_scaled, y_test)))
best_ridge = np.mean(cross_val_score(ridge, X_train_scaled, y_train))
```

Results

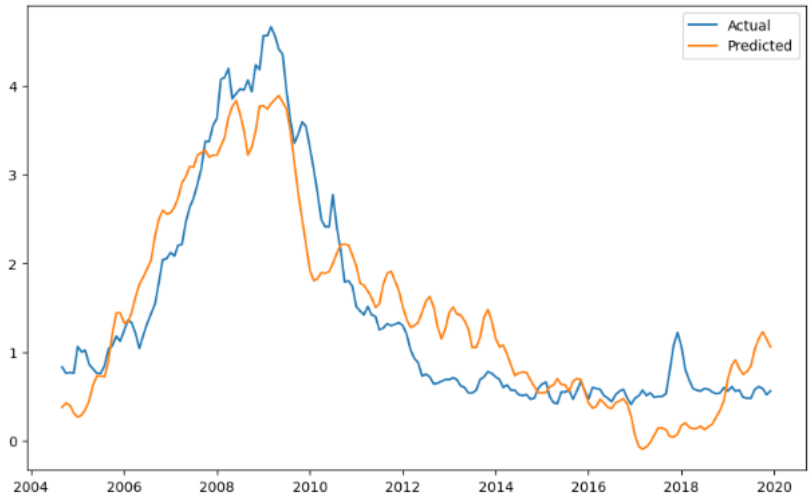
```
Best combination of lags: (0, 9, 6, 6, 0)
R-squared: 0.7251370309474902
Regression equation: const                          9.762100
interest_rate_MA_lag_0       -0.038329
unemployment_rate_MA_lag_9   -0.183372
inflation_MA_lag_6           -9.772384
real_gdp_index_MA_lag_6      -0.000281
UMCSENT_MA_lag_0             -0.028406
dtype: float64
```



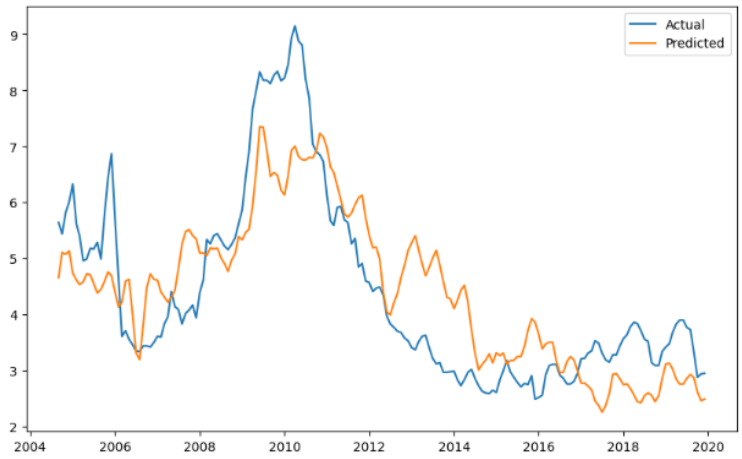
```
Best combination of lags: (9, 0, 0, 9, 3)
R-squared: 0.7773428874199755
Regression equation: const                          0.644924
interest_rate_MA_lag_9          0.451519
unemployment_rate_MA_lag_0      0.419749
inflation_MA_lag_0            -25.337159
real_gdp_index_MA_lag_9         0.000076
UMCSENT_MA_lag_3               -0.034308
dtype: float64
```



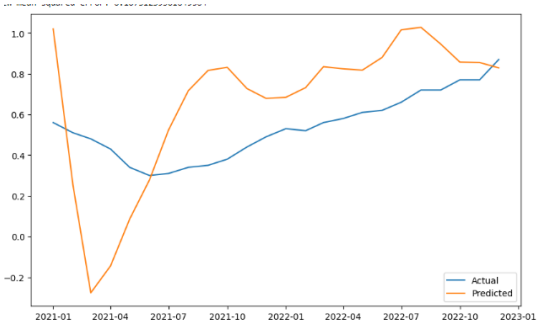

```
Best combination of lags: (9, 0, 0, 9, 0)
R-squared: 0.7894100056699481
Regression equation: const                          2.456877
interest_rate_MA_lag_9          0.594312
unemployment_rate_MA_lag_0      0.221643
inflation_MA_lag_0            -23.483163
real_gdp_index_MA_lag_9         0.000032
UMCSENT_MA_lag_0               -0.038538
dtype: float64
```

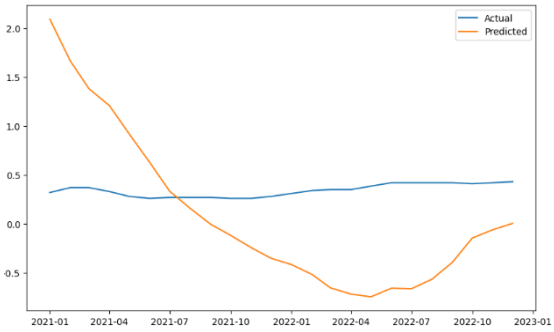


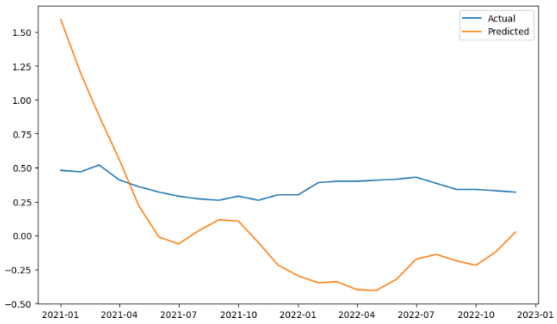
```
Best combination of lags: (9, 0, 0, 0, 9)
R-squared: 0.6310286529794872
Regression equation: const                      -7.667350
interest_rate_MA_lag_9          0.438687
unemployment_rate_MA_lag_0      1.163590
inflation_MA_lag_0            -15.332514
real_gdp_index_MA_lag_0        -0.000212
UMCSENT_MA_lag_9                0.098427
dtype: float64
```

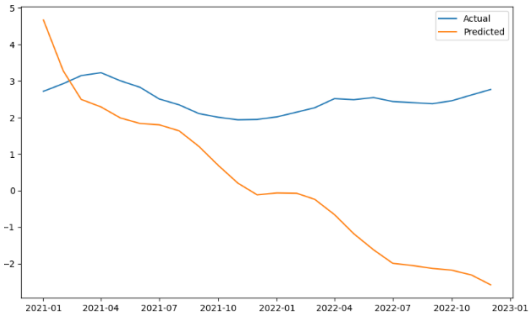


Using the models to predict defaults levels in 2021-2022






Actual
Predicted
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
2021-01
2021-04
2021-07
2021-10
2022-01
2022-04
2022-07
2022-10
2023-01


Actual
Predicted
5
4
3
2
1
0
−1
−2
2021-01
2021-04
2021-07
2021-10
2022-01
2022-04
2022-07
2022-10
2023-01