INSTITUTE FOR ADVANCED COMPUTING AND SOFTWARE DEVELOPMENT (IACSD), AKURDI, PUNE

Documentation On

"Recession Prediction of USA"

PG-DBDA MAR 2023

Submitted By:

Group No: 27

Roll No. :	Name :
233503	Abhishek Vats
233519	Harshal Koli
233543	Sagar Khond

Mr. Abhijit Nagargoje
Project Guide

Mr. Rohit Puranik
Centre Coordinator

ABSTRACT

This project focuses on the development of a robust model for predicting economic recessions in the United States. The model leverages a comprehensive dataset comprising key economic indicators, including the Debt-to-GDP ratio, Federal Funds rate, Inflation CPI, Manufacturing Output, Unemployment Rate, and others.

The project begins with data collection and thorough data cleaning to ensure data accuracy and reliability. Subsequently, an Exploratory Data Analysis (EDA) phase is conducted to uncover insights and patterns within the data.

Three distinct machine learning models - Logistic Regression, Random Forest, and Gradient Boost - are constructed and optimized. Hyperparameter tuning is performed using GridSearchCV to maximize predictive performance. The selected model, Gradient Boost, is then saved for future use.

To enhance accessibility, a user-friendly interface is developed using Flask, a Python web framework. The model is deployed on an Amazon EC2 instance, ensuring uninterrupted availability.

By integrating technologies such as pandas, matplotlib, seaborn, Amazon EC2, and Flask, this project demonstrates the potential of data science and machine learning in aiding economic forecasting and decision-making, ultimately contributing to a more resilient and informed economic landscape.

ACKNOWLEDGEMENT

A project usually falls short of its expectation unless aided and guided by the right persons at the right time. We avail this opportunity to express our deep sense of gratitude towards **Mr. Rohit Puranik (Centre Coordinator, IACSD, Pune)** and **Project Guide Mr. Abhijit Nagargoje.**

We are deeply indebted and grateful to them for their guidance, encouragementand deep concern for our project. Without their critical evaluation and suggestions at every stage of the project, this project could never have reached its present form.

Last but not the least we thank the entire faculty and the staff members of Institute for Advanced Computing and Software Development, Pune for their support.

Abhishek Vats (233503)
Harshal Koli (233519)
Sagar Khond (233543)

PG-DBDA Mar 2023

TABLE OF CONTENTS

ABSTRACT ---------- **2**
ACKNOWLEDGEMENT ---------- **3**

1. Introduction ---------- **5**
1.1. Introduction And Objectives ---------- 5
1.2. Why this problem needs to be solved? ---------- 5
1.3. Dataset Information ---------- 5

2. Problem Definition and Algorithm ---------- **11**
2.1 Problem Definition ---------- 11
2.2 Algorithm Definition ---------- 11

3. Experimental Evaluation ---------- **12**
3.1 Methodology/Model ---------- 12
3.2 Exploratory Data Analysis ---------- 14

4. Results and Discussion ---------- **21**

5. GUI ---------- **21**

6. Future Work and Conclusion ---------- **22**
6.1 Future Work ---------- 22
6.2 Conclusion ---------- 22

INTRODUCTION

1.1 Introduction and Objectives:

Economy is an area of the production, distribution and trade as well as consumption of goods and services. In general, it is defined as a social domain that emphasize the practices, discourses, and material expressions associated with the production, use, and management of scares resources. A given Economy is a set of processes that involve its culture, values, education, technology evolution, history, social organization, political structure, legal system, and natural resources as main factor. These factors give context, content, and set the conditions and parameters in which an economy functions. In other words, the economic domain is a social domain of interrelated human practices and transactions that does not stand alone.

1.2 Why this problem needs to be solved?

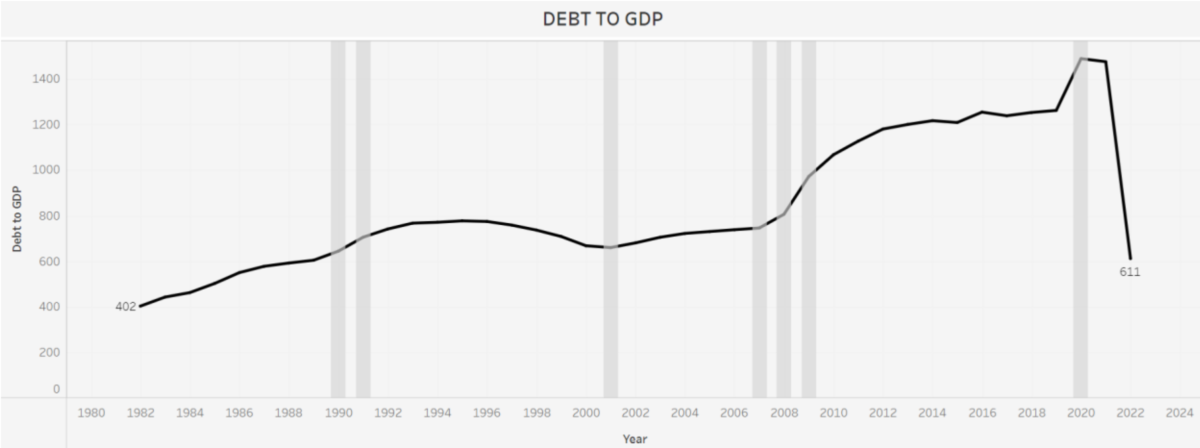
As Economy is important factor in growth of individual person as well as growth of whole nation. It indicates how well and efficiently a person can spend their money and what level of standard of living they can achieve with their income. Better the Economy condition, higher will be the standard of living people of that country can achieve. So, to have an idea in which direction country's Economy is heading and what kind of action should be taken to improve the living condition of country's people, it is necessary to know direction of economy of a country. That's why it become necessary to have some calculated prescription to always move towards better economy and have minimum possibility of having recession.

1.3 Dataset Information

1.3.1 Debt_to_GDP ratio.csv

It has two columns Debt to GDP ratio and other is date column calculated Quarterly

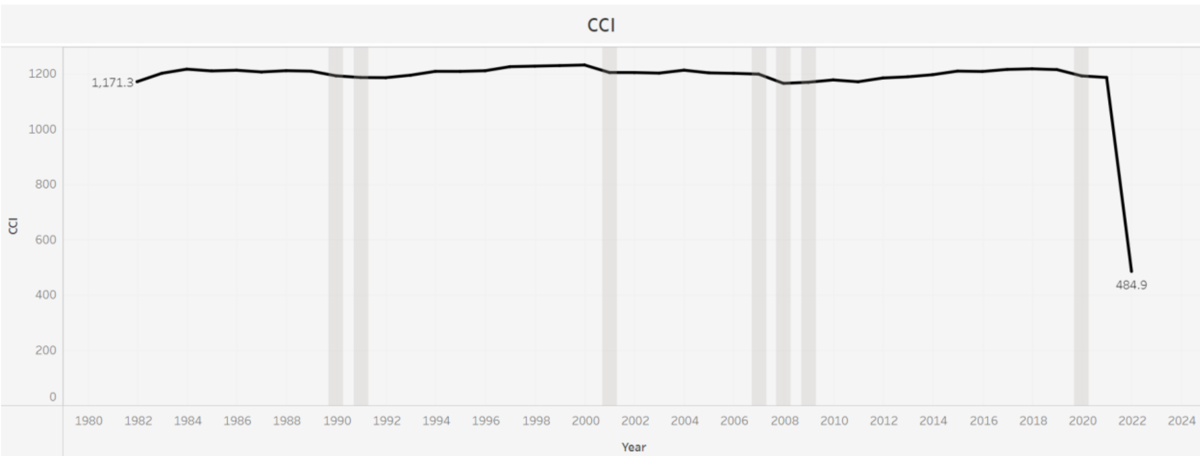
A debt-to-GDP ratio is an indicator on how much a debt a country owes and how much it produces to pay off its debts. Expressed in percentages, it is alternatively interpreted as the number of years needed in paying back the debt, in case the entire GDP has been allocated for debt repayment.



1.3.2 Consumer Confidence Index.csv

It has 8 columns- Location, Indicator, Frequency, Time, and Value.

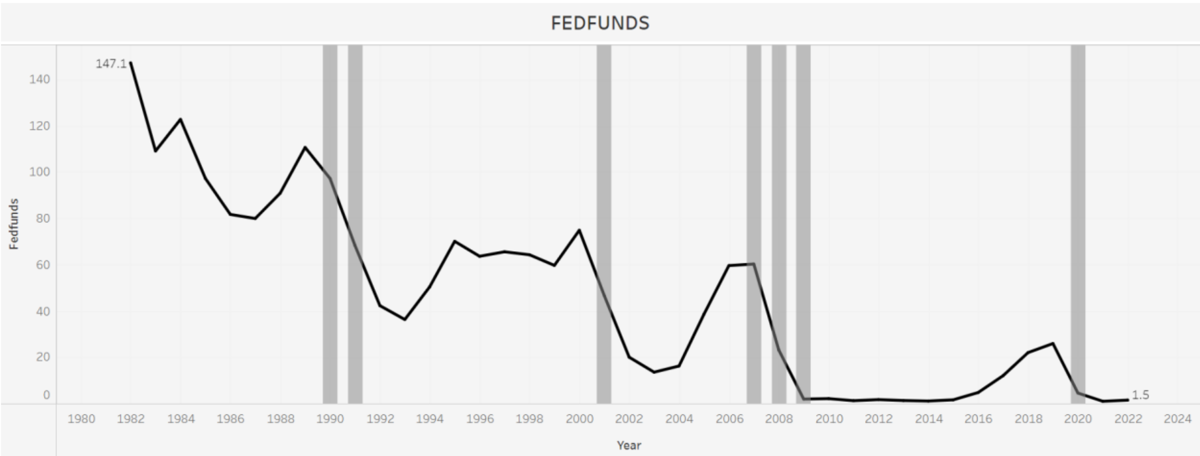
This consumer confidence indicator provides an indication of future developments of households' consumption and saving, based upon answers regarding their expected financial situation, their sentiment about the general economic situation, unemployment and capability of savings. An indicator above 100 signals a boost in the consumers' confidence towards the future economic situation, as a consequence of which they are less prone to save, and more inclined to spend money on major purchases in the next 12 months. Values below 100 indicate a pessimistic attitude towards future developments in the economy, possibly resulting in a tendency to save more and consume less.



1.3.3 Federal Funds.csv

It has 2 columns date and value, data is calculated monthly.

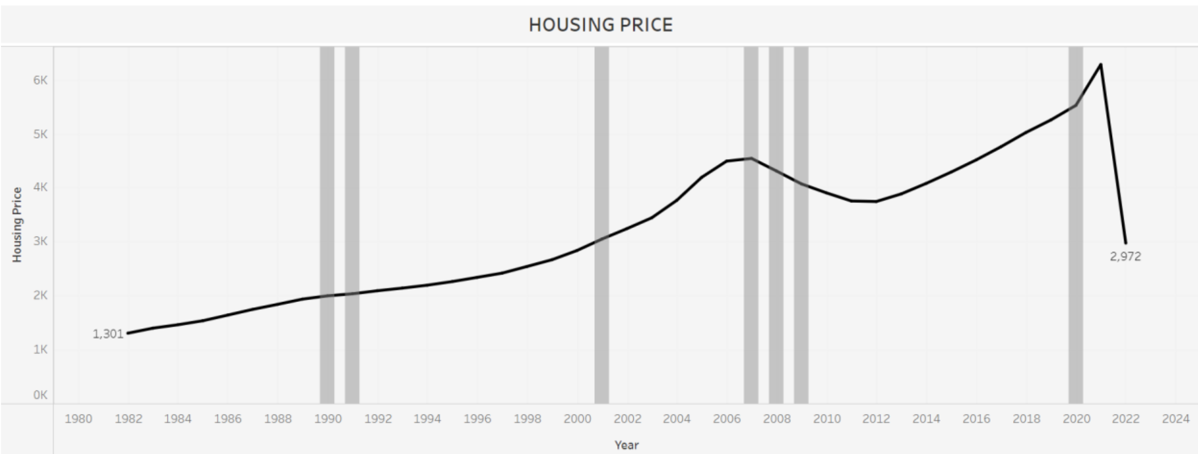
Federal funds, often referred to as fed funds, are excess reserves that commercial banks and other financial institutions deposit at regional Federal Reserve banks; these funds can be lent, then, to other market participants with insufficient cash on hand to meet their lending and reserve needs. The loans are unsecured and are made at a relatively low interest rate, called the federal funds rate or overnight rate, as that is the period for which most such loans are made.



1.3.4 Housing Price.csv

It has 2 columns date and value, data is calculated quarterly.

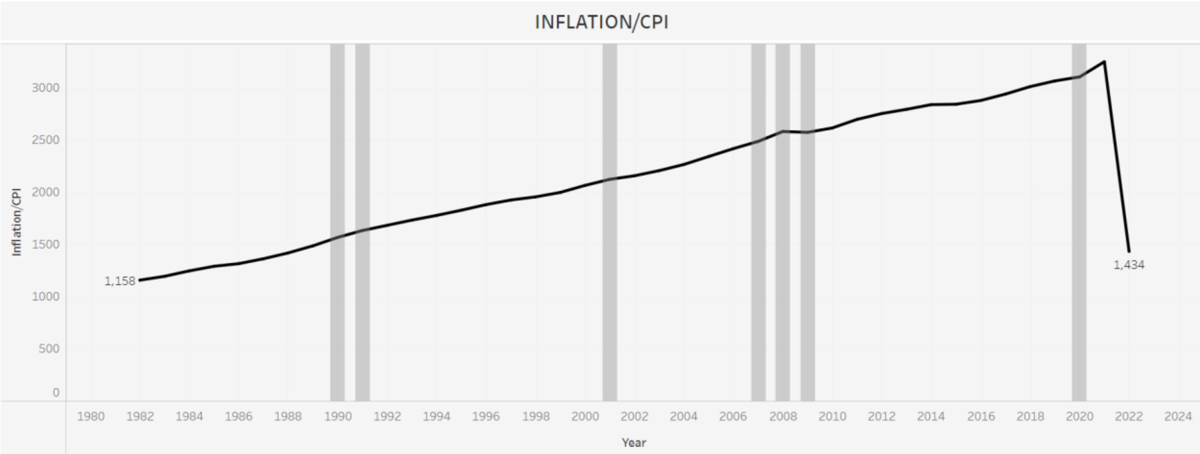
Housing prices include housing rent prices indices, real and nominal house prices indices, and ratios of price to rent and price to income. In most cases, the nominal house price index covers the sales of newly-built and existing dwellings, following the recommendations from the RPPI (Residential Property Prices Indices) manual.



1.3.5 Infliation_CPI.csv

It is a column-based data calculated monthly with a single year as row.

Inflation measured by consumer price index (CPI) is defined as the change in the prices of a basket of goods and services that are typically purchased by specific groups of households.

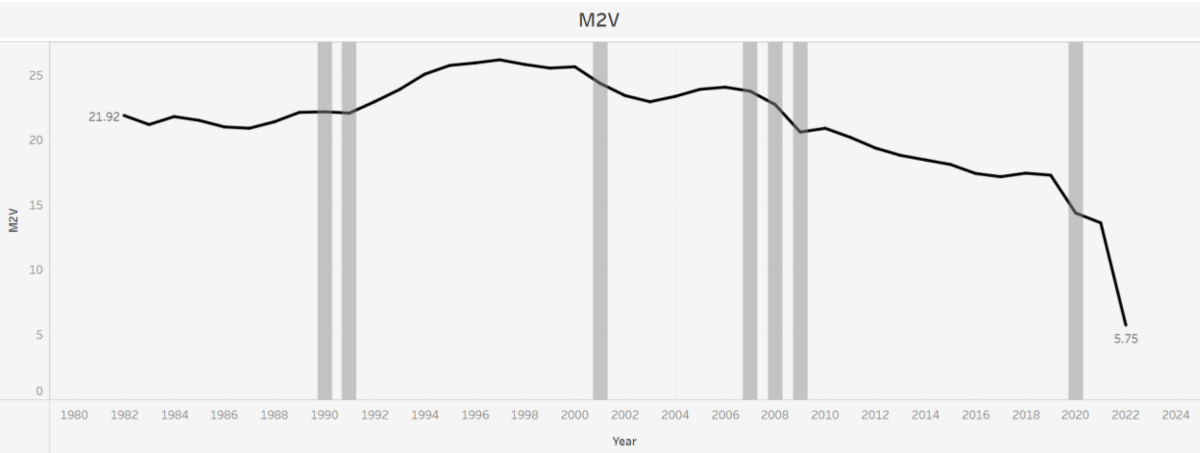


1.3.6 M2 velocity.csv

Data with velocity of M2 money stock as a column per year 1959

The velocity of money is the frequency at which one unit of currency is used to purchase domestically- produced goods and services within a given time period. In other words, it is the number of times one dollar is spent to buy goods and services per unit of time. If the velocity of money is increasing, then more transactions are occurring between individuals in an economy.

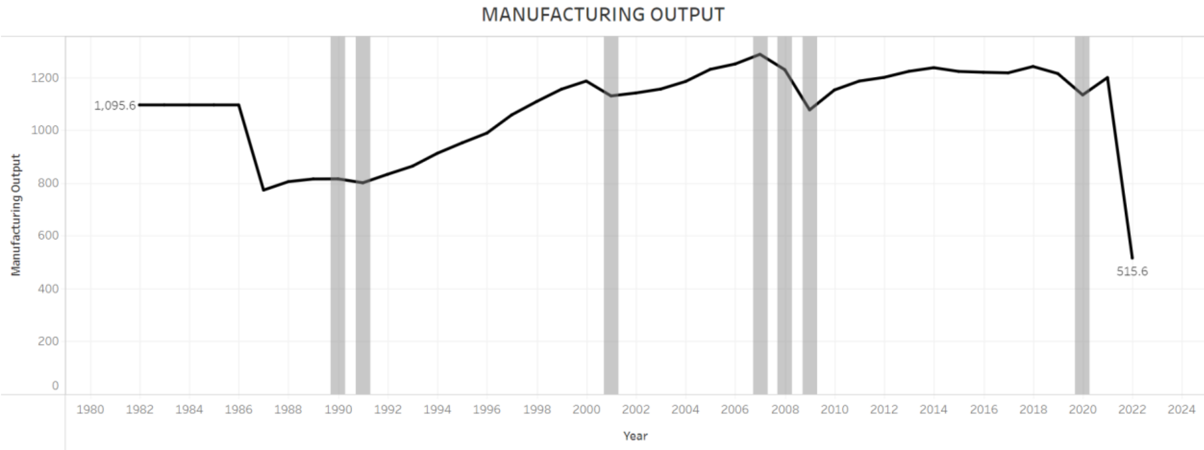
The frequency of currency exchange can be used to determine the velocity of a given component of the money supply, providing some insight into whether consumers and businesses are saving or spending their money.



1.3.7 Manufacturing Output.csv

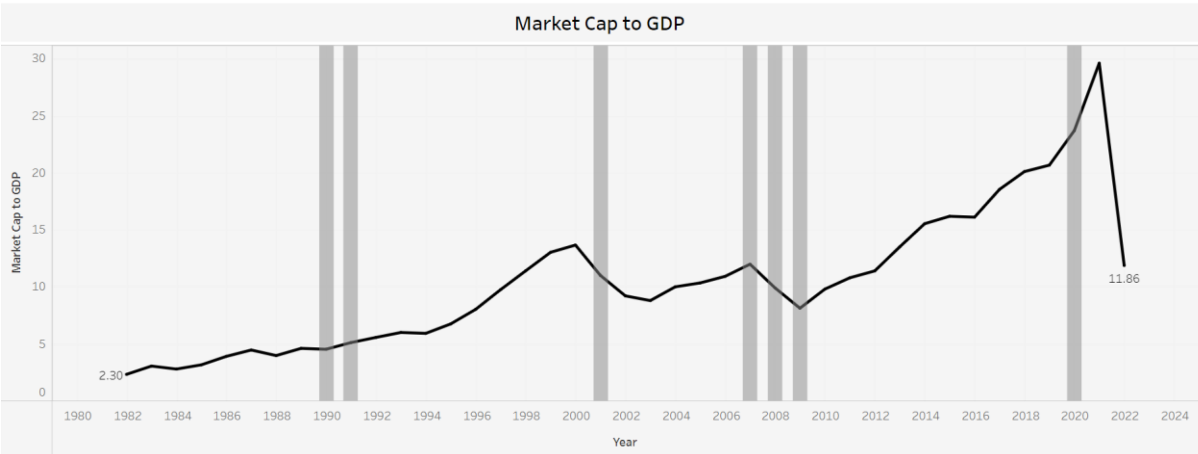
Output is the result of an economic process that has used inputs to produce a product or service that is available for sale or use somewhere else.

Net output, sometimes called netput is a quantity, in the context of production, that is positive if the quantity is output by the production process and negative if it is an input to the production process.



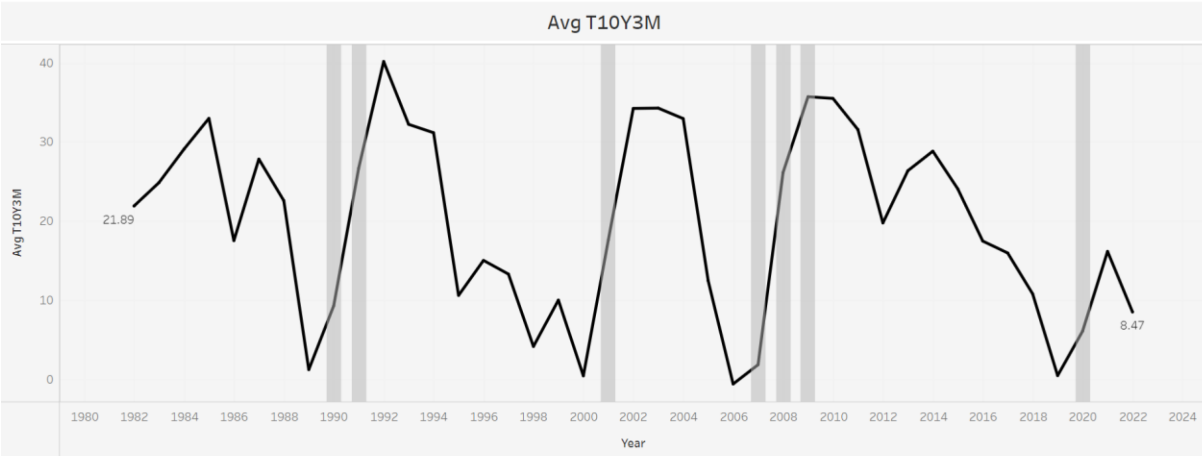
1.3.8 Market cap to GDP ratio.csv

The stock market capitalization-to-GDP ratio is a ratio used to determine whether an overall market is undervalued or overvalued compared to a historical average. The ratio can be used to focus on specific markets, such as the U.S. market, or it can be applied to the global market, depending on what values are used in the calculation. It is calculated by dividing the stock market cap by gross domestic product (GDP). The stock market capitalization-to-GDP ratio is also known as the Buffett Indicator—after investor Warren Buffett, who popularized its use.



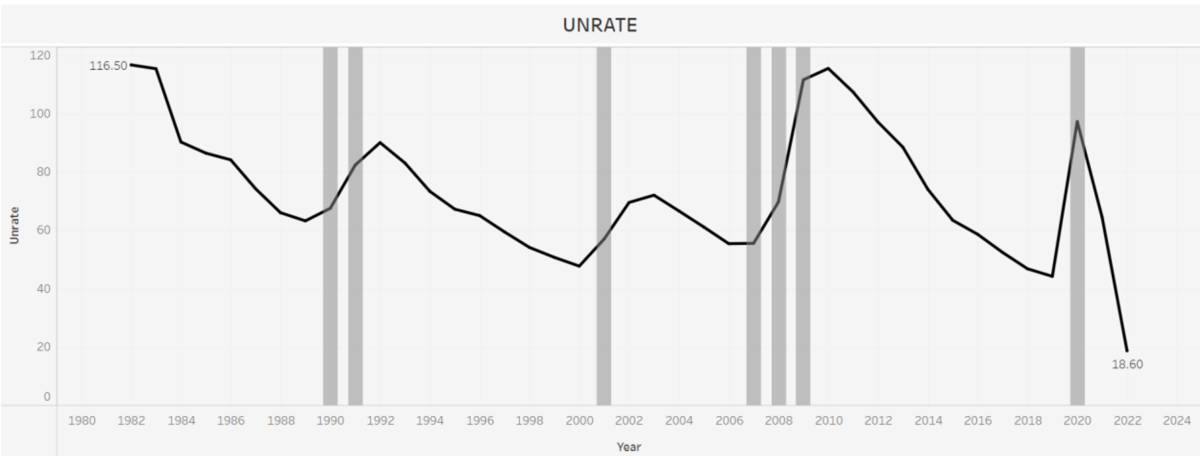
1.3.9 Treasury Yield Curve

The U.S. Treasury yield curve refers to a line chart that depicts the yields of short-term Treasury bills compared to the yields of long-term Treasury notes and bonds. The chart shows the relationship between the interest rates and the maturities of U.S. Treasury fixed-income securities. The Treasury yield curve (also referred to as the term structure of interest rates) shows yields at fixed maturities, such as one, two, three, and six months and one, two, three, five, seven, 10, 20, and 30 years. Because Treasury bills and bonds are resold daily on the secondary market, yields on the notes, bills, and bonds fluctuate.



1.3.10 Unemployment Rate

The unemployed are people of working age who are without work, are available for work, and have taken specific steps to find work. The uniform application of this definition results in estimates of unemployment rates that are more internationally comparable than estimates based on national definitions of unemployment. This indicator is measured in numbers of unemployed people as a percentage of the labor force and it is seasonally adjusted. The labor force is defined as the total number of unemployed people plus those in employment.



2. Problem Definition and Algorithm:

2.1 Problem Definition

The problem is quite straightforward. Required data sets from official website are given, it is up to us to Forecast that if country is in recession or not. The data is already split into a training and a test set, and we want to fit a model to the training data that is able to forecast recession. In fact, our metric of interest will be F1 score and Auc-Roc Score.

2.3 Algorithm Definition

Logistic Regression: This type of statistical model (also known as logit model) is often used for classification and predictive analytics. Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. In logistic regression, a logit transformation is applied on the odds—that is, the probability of success divided by the probability of failure.

Random Forest: A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is controlled with the max_samples parameter if bootstrap=True (default), otherwise the whole dataset is used to build each tree.

Gradient Boost: Gradient boosting re-defines boosting as a numerical optimization problem where the objective is to minimize the loss function of the model by adding weak learners using gradient descent. Gradient descent is a first-order iterative optimization algorithm for finding a local minimum of a differentiable function. As gradient boosting is based on minimizing a loss function, different types of loss functions can be used resulting in a flexible technique that can be applied to regression, multi-class classification, etc.

3. Experiment Evaluation

3.1 Methodology:

The objective of this project is to predict if the country is in recession or not. The data set is contained from Official website of Government of USA and have 10 csv files namely Consumer Confidence Index, Debt to GDP ratio, Federal Funds, Housing Price, Inflation (CPI), M2 Velocity, Manufacturing output, Market Cap to GDP ratio also known as Buffett Indicator), Treasury Yield Curve and Unemployment Rate.

The data is cleaned individually and then merged to obtain one master datafile and the data evaluation analysis is carried out.

Loading Raw Data

```
yc_df = pd.read_csv("Treasery_yield_curve.csv")
cci_df = pd.read_csv("Consumer Confidence Index Data (1960 - 2022 May)-monthly.csv")
fedfunds_df = pd.read_csv("FEDFUNDS (1954 - 2022)- monthly.csv")
inflation_cpi_df = pd.read_excel("Inflation_CPI Data for whole US - monthly.xlsx")
buffett_df = pd.read_excel("Market Cap to GDP Ratio (Buffett Indicator) - quaterly.xlsx")
unrate_df = pd.read_excel("Unemployment Rate Data - monthly.xlsx")
m2_velocity = pd.read_csv("m2 velocity- quaterly.csv")
manufacturing_output = pd.read_csv("manufacturing output- quaterly.csv")
debt_to_gdp = pd.read_csv("debt to gdp- quaterly.csv")
housing_price = pd.read_csv("housing price- quaterly.csv")
yield_df = pd.read_csv("T10Y3M.csv")
```

Processing:

All the data in individual Dataframe is provided either monthly or quarterly, splitting the date to get month and year separately and also converting whole data into single format i.e., every Month per year

Giving a example of above process using one dataframe (using federal funds)

Initial structure of our dataset-

```
#Effective Federal Funds Rate Data
fedfunds_df #from July 1954 and has months and 816x2
```

	DATE	FEDFUNDS
0	1954-07-01	0.80
1	1954-08-01	1.22
2	1954-09-01	1.07
3	1954-10-01	0.85
4	1954-11-01	0.83
...	...	...
811	2022-02-01	0.08
812	2022-03-01	0.20
813	2022-04-01	0.33
814	2022-05-01	0.77
815	2022-06-01	1.21

816 rows × 2 columns

Getting month and year column separately

```
#Splitting the 'DATE' column into 'Month' and 'Year' Columns
fedfunds_df['Month'] = fedfunds_df['DATE'].apply(lambda x:x.split('-')[1])
fedfunds_df['Year'] = fedfunds_df['DATE'].apply(lambda x:x.split('-')[0])
fedfunds_df = fedfunds_df.drop(['DATE'], axis = 1)
```

```
fedfunds_df
```

	FEDFUNDS	Month	Year
0	0.80	07	1954
1	1.22	08	1954
2	1.07	09	1954
3	0.85	10	1954
4	0.83	11	1954
...	...	...	...
811	0.08	02	2022
812	0.20	03	2022
813	0.33	04	2022
814	0.77	05	2022
815	1.21	06	2022

Some of the data is quarterly available, we converted it into monthly data. (filling these missing values with previous month value) Example of that is shown below

```
#Handling Buffet Indicator, which has values after 3 month gaps. Giving those '3 month gaps' the value of the month prior to that
for i in range(288, len(df)): #because all indices less than 288 are NaN values for the Buffett Indicator.
  if math.isnan(df['Market_Cap_to_GDP'][i]):
    df['Market_Cap_to_GDP'][i] = df['Market_Cap_to_GDP'][i-1]
  if math.isnan(df['M2V'][i]):
    df['M2V'][i] = df['M2V'][i-1]
  if math.isnan(df['Manufacturing Output'][i]):
    df['Manufacturing Output'][i] = df['Manufacturing Output'][i-1]
  if math.isnan(df['Debt_to_GDP'][i]):
    df['Debt_to_GDP'][i] = df['Debt_to_GDP'][i-1]
  if math.isnan(df['Housing Price'][i]):
    df['Housing Price'][i] = df['Housing Price'][i-1]
```

Handling NaN values

As in our current data most of the values before 1987 are Nan values, so we decided to drop those values (as filling these values with mean will change structure of our data)

And for missing values after 1987, we filled those values with mean of that particular column

Final DataFrame

In our Final dataframe we added a column manually (targeted column) namely “Recession_US”. This column is a categorical column which contains 0 for the year’s in which USA has not seen any recession and 1 for those years in which USA was in recession.

We get the data for this column from Wikipedia page for USA Economy.

```
final_df
```

	Year	Month	Inflation/CPI	UNRATE	FEDFUNDS	CCI	Market_Cap_to_GDP	Avg T10Y3M	Housing Price	Debt_to_GDP	Manufacturing Output	M2V	Recession_US
0	1982	1	94.400	8.6	13.22	97.50	0.184275	1.676000	106.20	32.41298	91.302757	1.842	0
1	1982	2	94.700	8.9	14.78	97.41	0.184275	0.131500	106.20	32.41298	91.302757	1.842	0
2	1982	3	94.700	9.0	14.68	97.27	0.184275	0.546522	106.20	32.41298	91.302757	1.842	0
3	1982	4	95.000	9.3	14.94	97.31	0.182363	0.503182	108.84	32.40213	91.302757	1.836	0
4	1982	5	95.900	9.4	14.45	97.38	0.182363	0.865238	108.84	32.40213	91.302757	1.836	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
480	2022	1	281.933	4.0	0.08	97.46	2.371155	1.541429	578.58	122.87943	102.768000	1.140	0
481	2022	2	284.182	3.8	0.08	97.13	2.371155	1.538500	578.58	122.87943	102.768000	1.140	0
482	2022	3	287.708	3.6	0.20	96.91	2.371155	1.680435	578.58	122.87943	102.768000	1.140	0
483	2022	4	288.663	3.6	0.33	96.84	2.371155	1.889524	617.89	121.07100	103.670000	1.167	0
484	2022	5	291.474	3.6	0.77	96.60	2.371155	1.818636	617.89	121.07100	103.670000	1.167	0

485 rows × 13 columns

3.2 Exploratory Data Analysis

Describing the basic structure of or data, i.e., Count, Sum, Mean, Max, Min, 25%,50% and 75% Quartile.

```
df.describe()
```

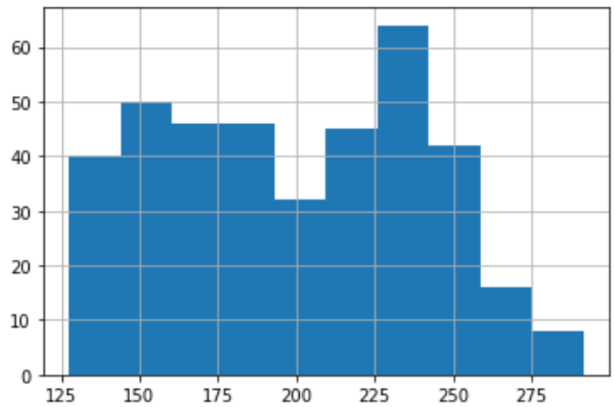
	Unnamed: 0	Year	Month	Inflation/CPI	UNRATE	FEDFUNDS	CCI	Market_Cap_to_GDP	Yield_Curve_Diff	Housing Price	Debt_to_GDP	Manufacturing Output	M2V	Recession_US
count	389.000000	389.000000	389.000000	389.000000	389.000000	389.000000	389.000000	389.000000	389.000000	389.000000	389.000000	389.000000	389.000000	389.000000
mean	194.000000	2005.712082	6.455013	199.144905	5.868895	2.705758	100.035116	1.021436	-1.685931	304.611260	78.122631	93.369797	1.800069	0.102828
std	112.438872	9.372002	3.460552	41.498502	1.723185	2.384876	1.547333	0.491998	1.112371	99.532631	22.389443	11.631069	0.299006	0.304125
min	0.000000	1990.000000	1.000000	127.500000	3.500000	0.050000	96.440000	0.350984	-3.684545	165.240000	51.968560	65.727000	1.112000	0.000000
25%	97.000000	1998.000000	3.000000	162.000000	4.600000	0.180000	98.940000	0.688687	-2.603636	207.960000	60.586930	88.872000	1.542000	0.000000
50%	194.000000	2006.000000	6.000000	199.700000	5.500000	2.090000	100.390000	0.900446	-1.618500	315.170000	64.254690	97.966000	1.905000	0.000000
75%	291.000000	2014.000000	9.000000	235.547000	6.800000	5.220000	101.080000	1.262211	-0.785000	372.380000	100.455630	101.809000	2.016000	0.000000
max	388.000000	2022.000000	12.000000	291.474000	14.700000	8.290000	102.990000	2.532661	0.696500	617.890000	134.835280	108.149000	2.192000	1.000000

Drawing the Basic Histogram for every Feature in our data to get the idea of its structure and skewness. So that we can understand how data is tending towards right or left.

a. Histogram for the Inflation/CPI Data

```
#Histogram for the Inflation/CPI Data
df['Inflation/CPI'].hist()
```

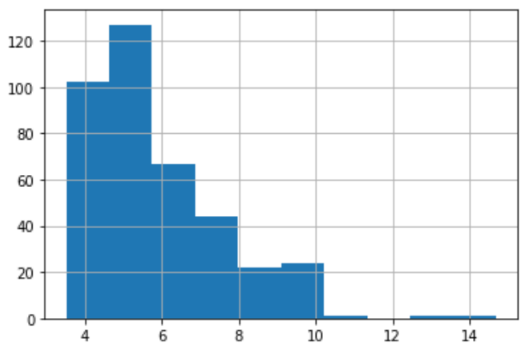
<AxesSubplot:>



b. Histogram for Unemployment Rate Data

```
#Histogram for the Unemployment Rate Data
df['UNRATE'].hist()
```

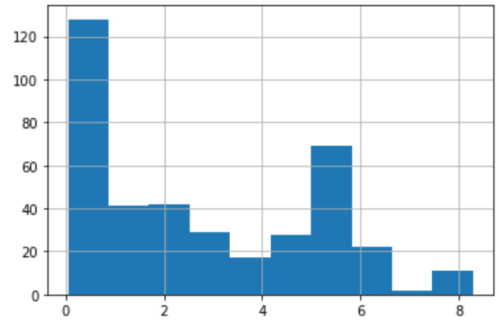
<AxesSubplot:>



c. Histogram of Effective Federal Fund Rate

```
#Histogram for the Effective Federal Funds Rate Data
df['FEDFUNDS'].hist()
```

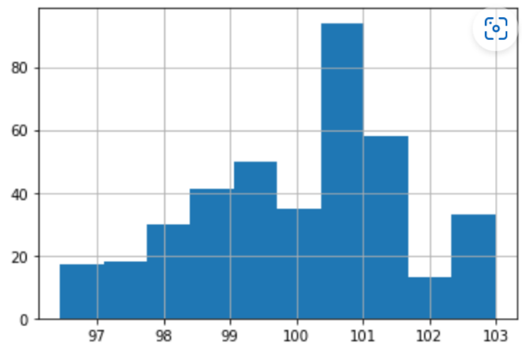
```
<AxesSubplot:>
```



d. Histogram for Consumer Confidence Index

```
#Histogram for the Consumer Confidence Index Data
df['CCI'].hist()
```

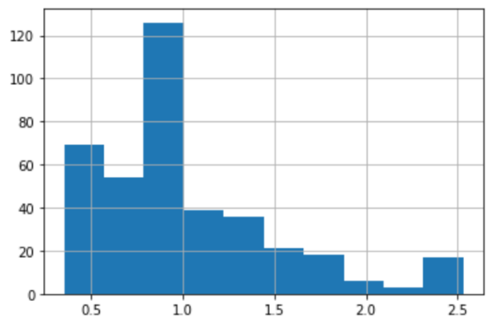
```
<AxesSubplot:>
```



e. Histogram for Market Cap/GDP (Buffet Indicator)

```
#Histogram for the Market Cap/GDP (Buffett Indicator) Data
df['Market_Cap_to_GDP'].hist()
```

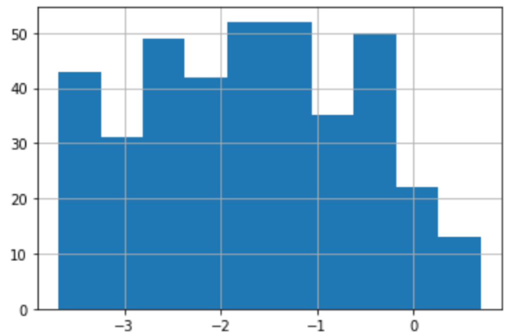
```
<AxesSubplot:>
```



f. Histogram for Yield Curve Data

```
#Histogram for the Yield Curve Data
df['Yield_Curve_Diff'].hist()
```

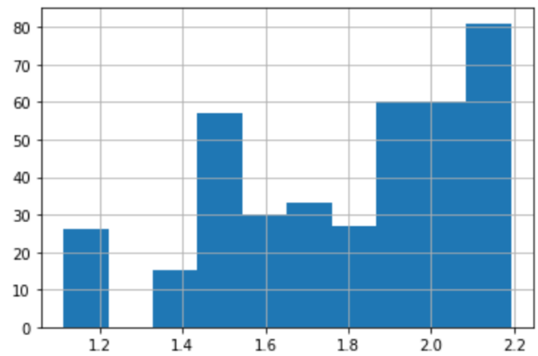
```
<AxesSubplot:>
```



g. Histogram for M2 Velocity

```
# Histogram for the M2 Velocity
df['M2V'].hist()
```

```
<AxesSubplot:>
```

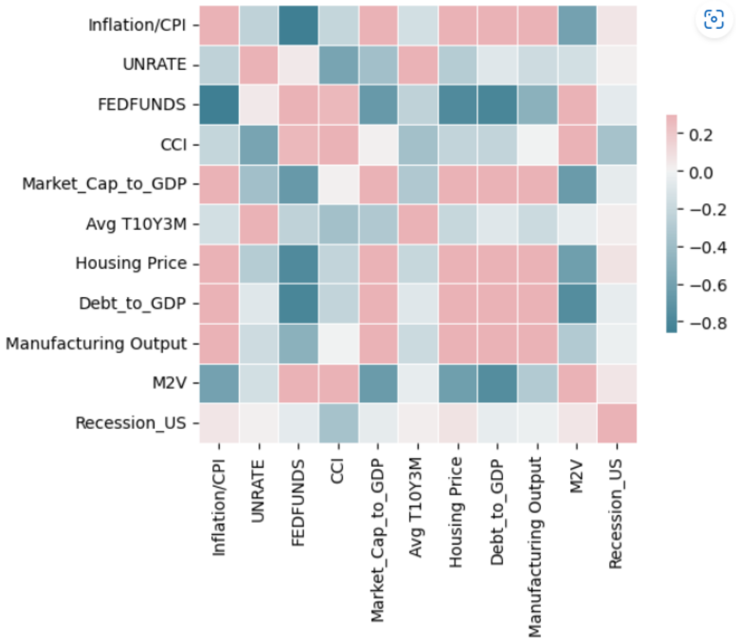


h. Correlation of each Feature with each other

```
df[['Inflation/CPI', 'UNRATE', 'FEDFUNDS',
       'CCI', 'Market_Cap_to_GDP', 'Avg T10Y3M', 'Housing Price',
       'Debt_to_GDP', 'Manufacturing Output', 'M2V', 'Recession_US']].corr()
```

	Inflation/CPI	UNRATE	FEDFUNDS	CCI	Market_Cap_to_GDP	Avg T10Y3M	Housing Price	Debt_to_GDP	Manufacturing Output	M2V	Recession_US
Inflation/CPI	1.000000	-0.237280	-0.862150	-0.212979	0.887889	-0.146520	0.961051	0.918863	0.681310	-0.601832	0.056172
UNRATE	-0.237280	1.000000	0.044088	-0.578681	-0.384517	0.534637	-0.286409	-0.090145	-0.176026	-0.142681	0.009955
FEDFUNDS	-0.862150	0.044088	1.000000	0.270903	-0.664503	-0.237928	-0.773377	-0.811555	-0.485627	0.464613	-0.055961
CCI	-0.212979	-0.578681	0.270903	1.000000	0.005079	-0.373346	-0.224949	-0.227259	-0.006500	0.322755	-0.359720
et_Cap_to_GDP	0.887889	-0.384517	-0.664503	0.005079	1.000000	-0.315151	0.901188	0.867876	0.601269	-0.644412	-0.050179
Avg T10Y3M	-0.146520	0.534637	-0.237928	-0.373346	-0.315151	1.000000	-0.207492	-0.088878	-0.179438	-0.042872	0.026427
Housing Price	0.961051	-0.286409	-0.773377	-0.224949	0.901188	-0.207492	1.000000	0.848834	0.683274	-0.622106	0.071737
Debt_to_GDP	0.918863	-0.090145	-0.811555	-0.227259	0.867876	-0.088878	0.848834	1.000000	0.463569	-0.755154	-0.049085
Manufacturing Output	0.681310	-0.176026	-0.485627	-0.006500	0.601269	-0.179438	0.683274	0.463569	1.000000	-0.298510	-0.025636
M2V	-0.601832	-0.142681	0.464613	0.322755	-0.644412	-0.042872	-0.622106	-0.755154	-0.298510	1.000000	0.050971
Recession_US	0.056172	0.009955	-0.055961	-0.359720	-0.050179	0.026427	0.071737	-0.049085	-0.025636	0.050971	1.000000

i. Correlation Heat Map



1. Result and Discussion

After Cleaning the data and Doing EDA on our data (as shown above) we made some machine learning models namely, Logistic regression, Random Forest and Gradient Boost

Code for these 3 models is shown below:

a. Logistic Regression

we have used GridSearchCV to find the Hyperparameters and on bases of these Hyperparameter we have trained our model from Training Data (divided total Data into Training and Testing Data)

```
param3 = {}
param3['classifier__C'] = [10**-2, 10**-1, 10**0, 10**1, 10**2]
param3['classifier__penalty'] = ['l1', 'l2']
param3['classifier__class_weight'] = [None, {0:1,1:5}, {0:1,1:10}, {0:1,1:25}]
param3['classifier'] = [clf3]
```

```
pipeline3 = Pipeline([('classifier', clf3)])
params3 = [param3]
```

```
gs3 = GridSearchCV(pipeline3, params3, cv=3, n_jobs=-1, scoring='roc_auc').fit(x_train, y_train)
```

```
# Best performing model and its corresponding hyperparameters
gs3.best_params_
```

```
{'classifier': LogisticRegression(C=100, class_weight={0: 1, 1: 25}, random_state=42),
 'classifier__C': 100,
 'classifier__class_weight': {0: 1, 1: 25},
 'classifier__penalty': 'l2'}
```

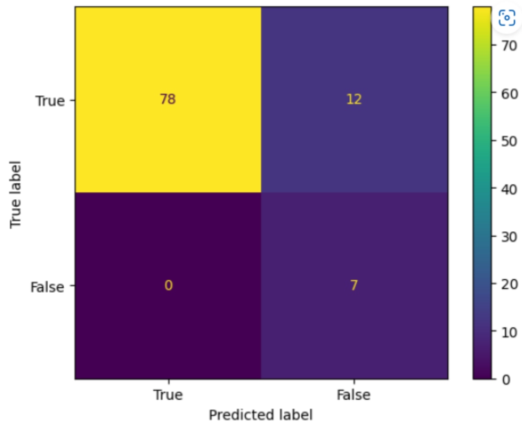
Test performance parameters of this model are shown below:

```
# Test data performance
print("Test Precision:",precision_score(gs3.predict(x_test), y_test))
print("Test Recall:",recall_score(gs3.predict(x_test), y_test))
print("Test F1 Score:",f1_score(gs3.predict(x_test), y_test))
print("Test ROC AUC Score:",roc_auc_score(gs3.predict(x_test), y_test))
```

```
Test Precision: 1.0
Test Recall: 0.3684210526315789
Test F1 Score: 0.5384615384615384
Test ROC AUC Score: 0.6842105263157895
```

From here we can say that Logistic regression model is giving accuracy of about 53.846%. which is too low to consider this model as final model.

Confusion Matrix for Logistic regressor model



b. Random Forest Model

Finding Hyperparameters for Random forest and the fitting the model with Training data

```
param1 = {}
param1['classifier__n_estimators'] = [10, 50, 100, 250]
param1['classifier__max_depth'] = [5, 10, 20]
param1['classifier__class_weight'] = [None, {0:1,1:5}, {0:1,1:10}, {0:1,1:25}]
param1['classifier'] = [clf1]
```

```
# creating Pipeline
pipeline = Pipeline([('classifier', clf1)])
params = [param1]
```

```
# finding best parameters and fitting that model
gs = GridSearchCV(pipeline, params, cv=3, n_jobs=-1, scoring='roc_auc').fit(x_train, y_train)
```

```
# Best performing model and its corresponding hyperparameters
gs.best_params_
```

```
{'classifier': RandomForestClassifier(class_weight={0: 1, 1: 10}, max_depth=10,
                        random_state=42),
 'classifier__class_weight': {0: 1, 1: 10},
 'classifier__max_depth': 10,
 'classifier__n_estimators': 100}
```

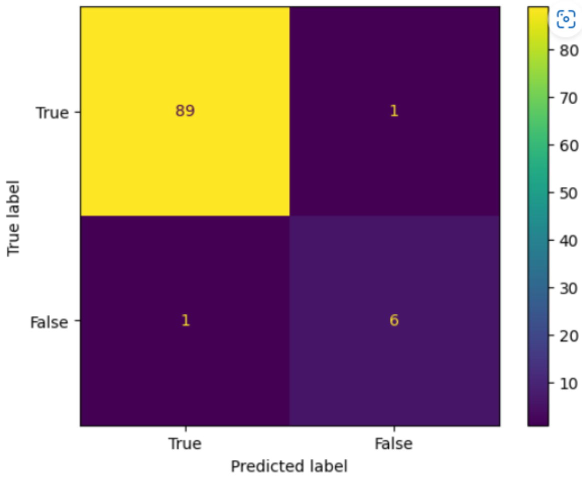
Test performance parameters of this model are shown below:

```
# Test data performance
from sklearn.metrics import *
print("Test Precision:",precision_score(gs.predict(x_test), y_test))
print("Test Recall:",recall_score(gs.predict(x_test), y_test))
print("Test F1 Score:",f1_score(gs.predict(x_test), y_test))
print("Test ROC AUC Score:",roc_auc_score(gs.predict(x_test), y_test))
```

```
Test Precision: 0.8571428571428571
Test Recall: 0.8571428571428571
Test F1 Score: 0.8571428571428571
Test ROC AUC Score: 0.9230158730158731
```

From above calculations, we can say that our model is having an accuracy of about 85.714% (which is enough to have this model in confidence)

Confusion Matrix of Random Forest Model



c. Gradient Boost Model

Finding Hyperparameters for Gradient Boost Model and the fitting the model with Training data

```
param4 = {}
param4['classifier__n_estimators'] = [10, 50, 100, 250]
param4['classifier__max_depth'] = [5, 10, 20]
param4['classifier'] = [clf4]
```

```
pipeline4 = Pipeline([('classifier', clf4)])
params4 = [param4]
```

```
gs4 = GridSearchCV(pipeline4, params4, cv=3, n_jobs=-1, scoring='roc_auc').fit(x_train, y_train)
```

```
# Best performing model and its corresponding hyperparameters
gs4.best_params_
```

```
{'classifier': GradientBoostingClassifier(max_depth=5, random_state=42),
 'classifier__max_depth': 5,
 'classifier__n_estimators': 100}
```

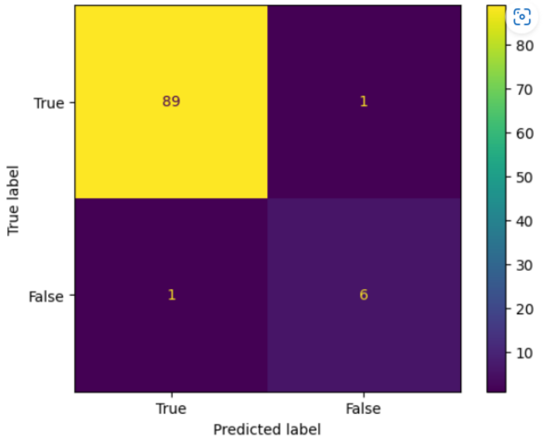
Test performance parameters of this model are shown below:

```
# Test data performance
print("Test Precision:",precision_score(gs4.predict(x_test), y_test))
print("Test Recall:",recall_score(gs4.predict(x_test), y_test))
print("Test F1 Score:",f1_score(gs4.predict(x_test), y_test))
print("Test ROC AUC Score:",roc_auc_score(gs4.predict(x_test), y_test))
```

```
Test Precision: 0.8571428571428571
Test Recall: 0.8571428571428571
Test F1 Score: 0.8571428571428571
Test ROC AUC Score: 0.9230158730158731
```

from this model, we can say that it is giving an accuracy of about 85.714%, which is equal to the accuracy of Random Forest.

Confusion Matrix for Gradient Boost model:



Discussion:

From above calculation we can see that Random forest and Gradient Boost are giving same accuracy i.e., 85.714%. This is why we decided to save the model build by Gradient Boost Algorithm and later we will deploy it on cloud.

2. GUI

We have Made a local server using Flask in python, which makes the user interaction more attractive and easier for user to make predictions using Model that we have built and saved. Later we have deployed our model on Amazon EC2 instance which will make server run indefinitely.

Front Page

Page when Country is in Recession

Page when Country is Not in Recession

3. Future Work and Conclusion

6.1 Future Work

we can enhance our model building of data by gathering economic data more frequently, which will help us to understand the situation of economy much better and in faster way.

6.2Conclusion

Helping the policy makers in improving the condition of citizens of those country will affect the life of everyone. Machine learning can help us to make better decisions by studying the previous data and giving us indication of future outcomes.

REFERENCES

1. Dataset Link :

https://fredhelp.stlouisfed.org/#fred-data-how-can-i-download-data-from-fred

2. SKlearn documentation (Version 1.2) :

https://scikit-learn.org/1.2/modules/classes.html