01_data_preprocessing_and_feature_engineering

July 25, 2025

Purpose: This cell imports the necessary Python libraries for data acquisition and initial processing. It also defines the start date for data collection and loads the FRED API key from an environment variable for secure access.

Code Functionality:

- Imports **pandas** for data manipulation, **Fred API** for fetching economic data, **yfinance** for market data, and **datetime** for handling dates.
- Loads environment variables using **dotenv** to securely access the `FRED_API_KEY`.
- Sets a `START_DATE` of '1960-01-01' to ensure the data captures multiple economic cycles.
- Initializes the Fred API client with the retrieved key, raising an error if the key is not found.

Output Analysis: This cell does not produce any direct console output. Its purpose is to prepare the environment and make the necessary libraries and objects available for subsequent code execution.

[1]:
```
import pandas as pd
from fredapi import Fred
import yfinance as yf
from datetime import datetime
import os
from dotenv import load_dotenv

# Load environment variables from .env file
load_dotenv()

# --- Configuration ---
FRED_API_KEY = os.environ.get("FRED_API_KEY")
START_DATE = '1960-01-01'

# Initialize Fred API client
if not FRED_API_KEY:
    raise ValueError("FRED_API_KEY environment variable not found. Please set␣
 ↪it in your .env file.")
fred = Fred(api_key=FRED_API_KEY)

print(" Libraries imported and configuration set.")
```

```
Libraries imported and configuration set.
```

Purpose: This cell downloads the official NBER recession indicator data, which will serve as the ground truth for our model's target variable.

Code Functionality:

- Uses `fred.get_series()` to download the 'USREC' series, representing NBER-defined recessions.
- Renames the series to 'Recession' for clarity.
- Converts the series to a DataFrame and resamples it to a month-end frequency (`'ME'`). It uses `.last()` to get the last known value of the month and then forward/backward fills (`.ffill().bfill()`) to ensure there are no gaps.
- Saves the cleaned recession data to a CSV file for future use and prints the head and tail to verify the data.

Output Analysis: The output confirms the data range and shows the first and last five rows of the recession indicator DataFrame, where `1.0` signifies a month in recession and `0.0` signifies an expansion. A confirmation message indicates the file has been saved.

[2]:
```
# --- Step 1.1: Defining Recession Target Variable (USREC) ---
print("Step 1.1: Defining Recession Target Variable (USREC)")

# Download NBER Recession Indicators for the United States (monthly frequency)
usrec = fred.get_series('USREC', observation_start=START_DATE)

# Rename the series for clarity
usrec.name = 'Recession'

# Convert to DataFrame for easier merging later
df_recession = usrec.to_frame()
df_recession = df_recession.resample('ME').last().ffill().bfill()

print("USREC (Recession Indicator) head (after resampling to month-end):")
print(df_recession.head())
print("\nUSREC (Recession Indicator) tail:")
print(df_recession.tail())
print(f"\nUSREC data range: {df_recession.index.min()} to {df_recession.index.
 ↪max()}")

# Save to data folder
df_recession.to_csv('E:/Project_3/Recession_Prediction_Network_Analysis/data/
 ↪usrec_recession_indicator.csv')
print("\nRecession indicator saved to data/usrec_recession_indicator.csv")
```

```
Step 1.1: Defining Recession Target Variable (USREC)
USREC (Recession Indicator) head (after resampling to month-end):
            Recession
1960-01-31        0.0
1960-02-29        0.0
```

```
1960-03-31         0.0
1960-04-30         0.0
1960-05-31         1.0

USREC (Recession Indicator) tail:
            Recession
2025-02-28         0.0
2025-03-31         0.0
2025-04-30         0.0
2025-05-31         0.0
2025-06-30         0.0

USREC data range: 1960-01-31 00:00:00 to 2025-06-30 00:00:00

Recession indicator saved to data/usrec_recession_indicator.csv
```

Purpose: This cell downloads a curated list of key macroeconomic indicators from FRED and the S&P 500 index from Yahoo Finance. These will form the initial feature set for our model.

Code Functionality:

- Defines a list of `INDICATOR_SERIES_IDS` containing the FRED codes for various economic series (e.g., yield curve, unemployment, consumer sentiment).
- Loops through the list, downloading each series from FRED and storing it in a dictionary.
- Uses `yf.download` to fetch historical 'Close' price data for the S&P 500 index (`^GSPC`).
- Combines all downloaded series into a single pandas DataFrame, `all_indicators_df`, and saves it to a CSV file.

Output Analysis: The cell prints the download status for each economic indicator. The final output shows the head and info of the combined raw DataFrame, highlighting the different start dates and frequencies of the raw data before standardization.

[3]:
```python
# --- Step 1.2: Selecting & Downloading Leading Economic Indicators ---
print("Step 1.2: Selecting & Downloading Leading Economic Indicators")

INDICATOR_SERIES_IDS = [
    'T10Y3MM',
    'ICSA',
    'UNRATE',
    'PERMIT',
    'UMCSENT',
    'VIXCLS',
    'USALOLITONOSTSAM',
    'PCE',
    'CPIAUCSL',
    'INDPRO',
    'CPILFESL'
]
```

```
# Dictionary to store all downloaded series
all_indicators = {}

# Download data from FRED
for series_id in INDICATOR_SERIES_IDS:
    print(f"Downloading {series_id}...")
    try:
        data = fred.get_series(series_id, observation_start=START_DATE)
        if data is not None and not data.empty:
            all_indicators[series_id] = data
            print(f"  Downloaded {series_id}: Data from {data.index.min().
 ↪strftime('%Y-%m-%d')} to {data.index.max().strftime('%Y-%m-%d')}␣
 ↪({len(data)} entries)")
        else:
            print(f"  Warning: No data returned for {series_id}.")
    except Exception as e:
        print(f"  Error downloading {series_id}: {e}")

# --- Add S&P 500 download using yfinance ---
sp500_ticker = '^GSPC' # Standard ticker for S&P 500
yfinance_start_date = START_DATE
yfinance_end_date = datetime.now().strftime('%Y-%m-%d')

print(f"Downloading {sp500_ticker} (S&P 500) using yfinance from␣
 ↪{yfinance_start_date} to {yfinance_end_date}...")
try:
    sp500_downloaded_df = yf.download(sp500_ticker, start=yfinance_start_date,␣
 ↪end=yfinance_end_date)
    if 'Close' in sp500_downloaded_df.columns:
        sp500_data = sp500_downloaded_df['Close'].squeeze()
        if not sp500_data.empty:
            sp500_data.name = 'SP500'
            all_indicators['SP500'] = sp500_data
            print(f"  Downloaded {sp500_ticker} (S&P 500): Data from␣
 ↪{sp500_data.index.min().strftime('%Y-%m-%d')} to {sp500_data.index.max().
 ↪strftime('%Y-%m-%d')} ({len(sp500_data)} entries)")
        else:
            print(f"  Warning: No data returned for {sp500_ticker} (S&P 500)␣
 ↪from yfinance.")
    else:
        print(f"  Error: 'Close' column not found in downloaded {sp500_ticker}␣
 ↪data.")
except Exception as e:
    print(f"  Error downloading {sp500_ticker} (S&P 500) from yfinance: {e}")
```

```python
# Combine all series into a single DataFrame
all_indicators_df = pd.DataFrame(all_indicators)
all_indicators_df.to_csv('E:/Project_3/Recession_Prediction_Network_Analysis/
 ↪data/raw_economic_indicators.csv')

print("\nRaw economic indicators saved to data/raw_economic_indicators.csv")
print("\nRaw all_indicators_df head:")
print(all_indicators_df.head())
print("\nRaw all_indicators_df info:")
all_indicators_df.info()
```

```
Step 1.2: Selecting & Downloading Leading Economic Indicators
Downloading T10Y3MM…
  Downloaded T10Y3MM: Data from 1982-01-01 to 2025-06-01 (522 entries)
Downloading ICSA…
  Downloaded ICSA: Data from 1967-01-07 to 2025-07-19 (3055 entries)
Downloading UNRATE…
  Downloaded UNRATE: Data from 1960-01-01 to 2025-06-01 (786 entries)
Downloading PERMIT…
  Downloaded PERMIT: Data from 1960-01-01 to 2025-06-01 (786 entries)
Downloading UMCSENT…
  Downloaded UMCSENT: Data from 1960-01-01 to 2025-05-01 (785 entries)
Downloading VIXCLS…
  Downloaded VIXCLS: Data from 1990-01-02 to 2025-07-23 (9277 entries)
Downloading USALOLITONOSTSAM…
  Downloaded USALOLITONOSTSAM: Data from 1960-01-01 to 2024-01-01 (769 entries)
Downloading PCE…
  Downloaded PCE: Data from 1960-01-01 to 2025-05-01 (785 entries)
Downloading CPIAUCSL…
  Downloaded CPIAUCSL: Data from 1960-01-01 to 2025-06-01 (786 entries)
Downloading INDPRO…
  Downloaded INDPRO: Data from 1960-01-01 to 2025-06-01 (786 entries)
Downloading CPILFESL…
  Downloaded CPILFESL: Data from 1960-01-01 to 2025-06-01 (786 entries)
Downloading ^GSPC (S&P 500) using yfinance from 1960-01-01 to 2025-07-25…

E:\temps\ipykernel_14424\1766694210.py:41: FutureWarning: YF.download() has
changed argument auto_adjust default to True
  sp500_downloaded_df = yf.download(sp500_ticker, start=yfinance_start_date,
end=yfinance_end_date)
[**********************100%**********************]  1 of 1 completed

  Downloaded ^GSPC (S&P 500): Data from 1960-01-04 to 2025-07-24 (16499 entries)

Raw economic indicators saved to data/raw_economic_indicators.csv

Raw all_indicators_df head:
            T10Y3MM  ICSA  UNRATE  PERMIT  UMCSENT  VIXCLS  USALOLITONOSTSAM  \
1960-01-01      NaN   NaN     5.2  1092.0      NaN     NaN          100.6913
```

```
1960-01-04      NaN   NaN     NaN     NaN      NaN     NaN               NaN
1960-01-05      NaN   NaN     NaN     NaN      NaN     NaN               NaN
1960-01-06      NaN   NaN     NaN     NaN      NaN     NaN               NaN
1960-01-07      NaN   NaN     NaN     NaN      NaN     NaN               NaN

              PCE  CPIAUCSL   INDPRO  CPILFESL      SP500
1960-01-01  323.6     29.37  24.1658      30.5        NaN
1960-01-04    NaN       NaN      NaN       NaN  59.910000
1960-01-05    NaN       NaN      NaN       NaN  60.389999
1960-01-06    NaN       NaN      NaN       NaN  60.130001
1960-01-07    NaN       NaN      NaN       NaN  59.689999

Raw all_indicators_df info:
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 20027 entries, 1960-01-01 to 2025-07-24
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   T10Y3MM           522 non-null    float64
 1   ICSA              3055 non-null   float64
 2   UNRATE            786 non-null    float64
 3   PERMIT            786 non-null    float64
 4   UMCSENT           641 non-null    float64
 5   VIXCLS            8978 non-null   float64
 6   USALOLITONOSTSAM  769 non-null    float64
 7   PCE               785 non-null    float64
 8   CPIAUCSL          786 non-null    float64
 9   INDPRO            786 non-null    float64
 10  CPILFESL          786 non-null    float64
 11  SP500             16499 non-null  float64
dtypes: float64(12)
memory usage: 2.0 MB

Raw economic indicators saved to data/raw_economic_indicators.csv

Raw all_indicators_df head:
            T10Y3MM  ICSA  UNRATE  PERMIT  UMCSENT  VIXCLS  USALOLITONOSTSAM  \
1960-01-01      NaN   NaN     5.2  1092.0      NaN     NaN          100.6913
1960-01-04      NaN   NaN     NaN     NaN      NaN     NaN               NaN
1960-01-05      NaN   NaN     NaN     NaN      NaN     NaN               NaN
1960-01-06      NaN   NaN     NaN     NaN      NaN     NaN               NaN
1960-01-07      NaN   NaN     NaN     NaN      NaN     NaN               NaN

              PCE  CPIAUCSL   INDPRO  CPILFESL      SP500
1960-01-01  323.6     29.37  24.1658      30.5        NaN
1960-01-04    NaN       NaN      NaN       NaN  59.910000
1960-01-05    NaN       NaN      NaN       NaN  60.389999
1960-01-06    NaN       NaN      NaN       NaN  60.130001
```

```
1960-01-07    NaN       NaN      NaN       NaN  59.689999

Raw all_indicators_df info:
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 20027 entries, 1960-01-01 to 2025-07-24
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   T10Y3MM           522 non-null    float64
 1   ICSA              3055 non-null   float64
 2   UNRATE            786 non-null    float64
 3   PERMIT            786 non-null    float64
 4   UMCSENT           641 non-null    float64
 5   VIXCLS            8978 non-null   float64
 6   USALOLITONOSTSAM  769 non-null    float64
 7   PCE               785 non-null    float64
 8   CPIAUCSL          786 non-null    float64
 9   INDPRO            786 non-null    float64
 10  CPILFESL          786 non-null    float64
 11  SP500             16499 non-null  float64
dtypes: float64(12)
memory usage: 2.0 MB
```

Purpose: This cell standardizes the raw data to a consistent monthly frequency and handles missing values that arise from different series start dates or reporting frequencies.

Code Functionality:

- Sorts the DataFrame by index to ensure chronological order before resampling.
- Resamples the entire DataFrame to a month-end frequency (`'M'`), taking the `.last()` available value in each month.
- Performs a forward fill (`.ffill()`) followed by a backward fill (`.bfill()`) to impute any remaining `NaN` values, resulting in a complete, dense dataset.
- Saves the cleaned, monthly data to a new CSV file.

Output Analysis: The output shows the head and info of the `df_monthly` DataFrame. The `.info()` summary confirms that all columns now have the same number of non-null entries, indicating that the resampling and filling operations were successful.

[4]:
```python
# --- Step 1.3: Standardizing Data Frequency and Handling Initial NaNs ---
print("\nStep 1.3: Standardizing Data Frequency and Handling Initial NaNs")

# First, ensure the index is sorted to prevent warnings with ffill/bfill
all_indicators_df = all_indicators_df.sort_index()

# Resample to month-end frequency
df_monthly = all_indicators_df.resample('M').last()
```

```python
# Fill forward and backward to handle NaNs that arise from different start dates
df_monthly = df_monthly.ffill().bfill()

# Save the monthly data
df_monthly.to_csv('E:/Project_3/Recession_Prediction_Network_Analysis/data/
 ↪monthly_economic_indicators.csv')

print("\nMonthly economic indicators saved to data/monthly_economic_indicators.
 ↪csv")
print("\nMonthly Resampled df_monthly head:")
print(df_monthly.head())
print("\nMonthly Resampled df_monthly info:")
df_monthly.info()
```

```
Step 1.3: Standardizing Data Frequency and Handling Initial NaNs

Monthly economic indicators saved to data/monthly_economic_indicators.csv

Monthly Resampled df_monthly head:
            T10Y3MM      ICSA  UNRATE  PERMIT  UMCSENT  VIXCLS  \
1960-01-31     1.67  204000.0     5.2  1092.0    100.0   25.36
1960-02-29     1.67  204000.0     4.8  1088.0    100.0   25.36
1960-03-31     1.67  204000.0     5.4   955.0    100.0   25.36
1960-04-30     1.67  204000.0     5.2  1016.0    100.0   25.36
1960-05-31     1.67  204000.0     5.1  1052.0     93.3   25.36

            USALOLITONOSTSAM    PCE  CPIAUCSL   INDPRO  CPILFESL      SP500
1960-01-31         100.69130  323.6     29.37  24.1658      30.5  55.610001
1960-02-29         100.41650  325.3     29.41  23.9508      30.6  56.119999
1960-03-31         100.11210  330.2     29.41  23.7357      30.6  55.340000
1960-04-30          99.82441  336.5     29.54  23.5476      30.6  54.369999
1960-05-31          99.57291  330.0     29.57  23.5207      30.6  55.830002

Monthly Resampled df_monthly info:
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 787 entries, 1960-01-31 to 2025-07-31
Freq: ME
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   T10Y3MM           787 non-null    float64
 1   ICSA              787 non-null    float64
 2   UNRATE            787 non-null    float64
 3   PERMIT            787 non-null    float64
 4   UMCSENT           787 non-null    float64
 5   VIXCLS            787 non-null    float64
```

```
 6   USALOLITONOSTSAM  787 non-null    float64
 7   PCE               787 non-null    float64
 8   CPIAUCSL          787 non-null    float64
 9   INDPRO            787 non-null    float64
 10  CPILFESL          787 non-null    float64
 11  SP500             787 non-null    float64
dtypes: float64(12)
memory usage: 79.9 KB
```

```
E:\temps\ipykernel_14424\337710882.py:8: FutureWarning: 'M' is deprecated and
will be removed in a future version, please use 'ME' instead.
  df_monthly = all_indicators_df.resample('M').last()
```

Purpose: This cell engineers a set of standard time-series features from the cleaned monthly data. These features are designed to capture momentum, year-over-year trends, and rolling volatility, which are common inputs for economic forecasting models.

Code Functionality:

- Creates a new DataFrame `df_features` to hold the engineered features.
- Iterates through each column of the `df_monthly` DataFrame.
- For each column, it calculates and adds four new features to `df_features`:
  - **Month-over-month percentage change (`_mom_change`)**
  - **Year-over-year percentage change (`_yoy_change`)**
  - **12-month rolling mean (`_roll12_mean`)**
  - **12-month rolling standard deviation (`_roll12_std`)**
- Saves the resulting feature set to `engineered_features.csv`.

Output Analysis: The head of the `df_features` DataFrame is printed, showing the newly created features. The initial rows contain `NaN` values, which is expected due to the 12-month rolling window calculation. The `.info()` summary shows the total number of new features created (original features x 4).

[5]:
```python
# --- Step 1.4: Feature Engineering ---
print("\nStep 1.4: Feature Engineering")

# Create new DataFrame for engineered features
df_features = pd.DataFrame(index=df_monthly.index)

for col in df_monthly.columns:
    # Month-over-month change
    df_features[f'{col}_mom_change'] = df_monthly[col].pct_change() * 100

    # Year-over-year change
    df_features[f'{col}_yoy_change'] = df_monthly[col].pct_change(periods=12) *␣
 ↪100

    # 12-month rolling mean
```

```
    df_features[f'{col}_roll12_mean'] = df_monthly[col].rolling(window=12).
 ↪mean()

    # 12-month rolling standard deviation
    df_features[f'{col}_roll12_std'] = df_monthly[col].rolling(window=12).std()

# Save the engineered features
df_features.to_csv('E:/Project_3/Recession_Prediction_Network_Analysis/data/
 ↪engineered_features.csv')

print("\nEngineered features saved to data/engineered_features.csv")
print("\nEngineered Features df_features head:")
print(df_features.head(15))
print("\nEngineered Features df_features info:")
df_features.info()
```

```
Step 1.4: Feature Engineering

Engineered features saved to data/engineered_features.csv

Engineered Features df_features head:
            T10Y3MM_mom_change  T10Y3MM_yoy_change  T10Y3MM_roll12_mean  \
1960-01-31                 NaN                 NaN                  NaN
1960-02-29                 0.0                 NaN                  NaN
1960-03-31                 0.0                 NaN                  NaN
1960-04-30                 0.0                 NaN                  NaN
1960-05-31                 0.0                 NaN                  NaN
1960-06-30                 0.0                 NaN                  NaN
1960-07-31                 0.0                 NaN                  NaN
1960-08-31                 0.0                 NaN                  NaN
1960-09-30                 0.0                 NaN                  NaN
1960-10-31                 0.0                 NaN                  NaN
1960-11-30                 0.0                 NaN                  NaN
1960-12-31                 0.0                 NaN                 1.67
1961-01-31                 0.0                 0.0                 1.67
1961-02-28                 0.0                 0.0                 1.67
1961-03-31                 0.0                 0.0                 1.67

            T10Y3MM_roll12_std  ICSA_mom_change  ICSA_yoy_change  \
1960-01-31                 NaN              NaN              NaN
1960-02-29                 NaN              0.0              NaN
1960-03-31                 NaN              0.0              NaN
1960-04-30                 NaN              0.0              NaN
1960-05-31                 NaN              0.0              NaN
1960-06-30                 NaN              0.0              NaN
1960-07-31                 NaN              0.0              NaN
```

```
1960-08-31                NaN             0.0             NaN
1960-09-30                NaN             0.0             NaN
1960-10-31                NaN             0.0             NaN
1960-11-30                NaN             0.0             NaN
1960-12-31                0.0             0.0             NaN
1961-01-31                0.0             0.0             0.0
1961-02-28                0.0             0.0             0.0
1961-03-31                0.0             0.0             0.0

            ICSA_roll12_mean  ICSA_roll12_std  UNRATE_mom_change  \
1960-01-31               NaN              NaN                NaN
1960-02-29               NaN              NaN          -7.692308
1960-03-31               NaN              NaN          12.500000
1960-04-30               NaN              NaN          -3.703704
1960-05-31               NaN              NaN          -1.923077
1960-06-30               NaN              NaN           5.882353
1960-07-31               NaN              NaN           1.851852
1960-08-31               NaN              NaN           1.818182
1960-09-30               NaN              NaN          -1.785714
1960-10-31               NaN              NaN          10.909091
1960-11-30               NaN              NaN           0.000000
1960-12-31          204000.0              0.0           8.196721
1961-01-31          204000.0              0.0           0.000000
1961-02-28          204000.0              0.0           4.545455
1961-03-31          204000.0              0.0           0.000000

            UNRATE_yoy_change  …  INDPRO_roll12_mean  INDPRO_roll12_std  \
1960-01-31                NaN  …                 NaN                NaN
1960-02-29                NaN  …                 NaN                NaN
1960-03-31                NaN  …                 NaN                NaN
1960-04-30                NaN  …                 NaN                NaN
1960-05-31                NaN  …                 NaN                NaN
1960-06-30                NaN  …                 NaN                NaN
1960-07-31                NaN  …                 NaN                NaN
1960-08-31                NaN  …                 NaN                NaN
1960-09-30                NaN  …                 NaN                NaN
1960-10-31                NaN  …                 NaN                NaN
1960-11-30                NaN  …                 NaN                NaN
1960-12-31                NaN  …           23.229475           0.596116
1961-01-31          26.923077  …           23.059233           0.596117
1961-02-28          43.750000  …           22.904667           0.584275
1961-03-31          27.777778  …           22.779225           0.550235

            CPILFESL_mom_change  CPILFESL_yoy_change  CPILFESL_roll12_mean  \
1960-01-31                  NaN                  NaN                   NaN
1960-02-29             0.327869                  NaN                   NaN
1960-03-31             0.000000                  NaN                   NaN
1960-04-30             0.000000                  NaN                   NaN
```

```
1960-05-31             0.000000                NaN                NaN
1960-06-30             0.326797                NaN                NaN
1960-07-31            -0.325733                NaN                NaN
1960-08-31             0.000000                NaN                NaN
1960-09-30             0.000000                NaN                NaN
1960-10-31             0.653595                NaN                NaN
1960-11-30             0.000000                NaN                NaN
1960-12-31            -0.324675                NaN          30.641667
1961-01-31             0.325733           0.983607          30.666667
1961-02-28             0.000000           0.653595          30.683333
1961-03-31             0.324675           0.980392          30.708333

            CPILFESL_roll12_std  SP500_mom_change  SP500_yoy_change  \
1960-01-31                  NaN               NaN               NaN
1960-02-29                  NaN          0.917098               NaN
1960-03-31                  NaN         -1.389877               NaN
1960-04-30                  NaN         -1.752803               NaN
1960-05-31                  NaN          2.685310               NaN
1960-06-30                  NaN          1.952349               NaN
1960-07-31                  NaN         -2.477161               NaN
1960-08-31                  NaN          2.612143               NaN
1960-09-30                  NaN         -6.039324               NaN
1960-10-31                  NaN         -0.242902               NaN
1960-11-30                  NaN          4.026974               NaN
1960-12-31             0.090034          4.627295               NaN
1961-01-31             0.088763          6.315605         11.095123
1961-02-28             0.093744          2.686954         13.043478
1961-03-31             0.108362          2.553592         17.564144

            SP500_roll12_mean  SP500_roll12_std
1960-01-31                NaN               NaN
1960-02-29                NaN               NaN
1960-03-31                NaN               NaN
1960-04-30                NaN               NaN
1960-05-31                NaN               NaN
1960-06-30                NaN               NaN
1960-07-31                NaN               NaN
1960-08-31                NaN               NaN
1960-09-30                NaN               NaN
1960-10-31                NaN               NaN
1960-11-30                NaN               NaN
1960-12-31          55.601666          1.382296
1961-01-31          56.115833          2.256656
1961-02-28          56.725833          3.092449
1961-03-31          57.535833          3.871343

[15 rows x 48 columns]
```

```
Engineered Features df_features info:
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 787 entries, 1960-01-31 to 2025-07-31
Freq: ME
Data columns (total 48 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   T10Y3MM_mom_change              786 non-null    float64
 1   T10Y3MM_yoy_change              775 non-null    float64
 2   T10Y3MM_roll12_mean             776 non-null    float64
 3   T10Y3MM_roll12_std              776 non-null    float64
 4   ICSA_mom_change                 786 non-null    float64
 5   ICSA_yoy_change                 775 non-null    float64
 6   ICSA_roll12_mean                776 non-null    float64
 7   ICSA_roll12_std                 776 non-null    float64
 8   UNRATE_mom_change               786 non-null    float64
 9   UNRATE_yoy_change               775 non-null    float64
 10  UNRATE_roll12_mean              776 non-null    float64
 11  UNRATE_roll12_std               776 non-null    float64
 12  PERMIT_mom_change               786 non-null    float64
 13  PERMIT_yoy_change               775 non-null    float64
 14  PERMIT_roll12_mean              776 non-null    float64
 15  PERMIT_roll12_std               776 non-null    float64
 16  UMCSENT_mom_change              786 non-null    float64
 17  UMCSENT_yoy_change              775 non-null    float64
 18  UMCSENT_roll12_mean             776 non-null    float64
 19  UMCSENT_roll12_std              776 non-null    float64
 20  VIXCLS_mom_change               786 non-null    float64
 21  VIXCLS_yoy_change               775 non-null    float64
 22  VIXCLS_roll12_mean              776 non-null    float64
 23  VIXCLS_roll12_std               776 non-null    float64
 24  USALOLITONOSTSAM_mom_change     786 non-null    float64
 25  USALOLITONOSTSAM_yoy_change     775 non-null    float64
 26  USALOLITONOSTSAM_roll12_mean    776 non-null    float64
 27  USALOLITONOSTSAM_roll12_std     776 non-null    float64
 28  PCE_mom_change                  786 non-null    float64
 29  PCE_yoy_change                  775 non-null    float64
 30  PCE_roll12_mean                 776 non-null    float64
 31  PCE_roll12_std                  776 non-null    float64
 32  CPIAUCSL_mom_change             786 non-null    float64
 33  CPIAUCSL_yoy_change             775 non-null    float64
 34  CPIAUCSL_roll12_mean            776 non-null    float64
 35  CPIAUCSL_roll12_std             776 non-null    float64
 36  INDPRO_mom_change               786 non-null    float64
 37  INDPRO_yoy_change               775 non-null    float64
 38  INDPRO_roll12_mean              776 non-null    float64
 39  INDPRO_roll12_std               776 non-null    float64
 40  CPILFESL_mom_change             786 non-null    float64
```

```
 41  CPILFESL_yoy_change           775 non-null    float64
 42  CPILFESL_roll12_mean          776 non-null    float64
 43  CPILFESL_roll12_std           776 non-null    float64
 44  SP500_mom_change              786 non-null    float64
 45  SP500_yoy_change              775 non-null    float64
 46  SP500_roll12_mean             776 non-null    float64
 47  SP500_roll12_std              776 non-null    float64
dtypes: float64(48)
memory usage: 301.3 KB
```

Purpose: This cell creates the final, analysis-ready dataset by combining the engineered features with the recession target variable. It performs a final cleaning step to ensure the data is ready for machine learning.

Code Functionality:

- Merges the `df_features` DataFrame (containing engineered features) with the `df_recession` DataFrame (containing the target) using an 'outer' join on their datetime index.
- Drops all rows containing any `NaN` values using `.dropna()`. This removes the initial 11-12 rows where rolling features could not be calculated, ensuring the dataset is complete.
- Saves the final, fully prepared dataset to `final_prepared_data.csv`.

Output Analysis: The `.info()` summary for the final DataFrame, `df_final_clean`, shows the total number of features plus the target column. The number of entries is reduced from the previous step, reflecting the removal of rows with `NaNs`. All columns now have an equal number of non-null entries, confirming the dataset is clean and ready for the next phase.

[6]:
```
# --- Step 1.5: Final Data Preparation ---
print("\nStep 1.5: Final Data Preparation")

# Merge features with the recession target variable
df_final = pd.merge(df_features, df_recession, left_index=True,␣
 ↪right_index=True, how='outer')

# Drop rows with NaN values. These are typically the first 11 or 12 rows
# due to the rolling window and percentage change calculations.
df_final_clean = df_final.dropna()

# Save the final prepared dataset
df_final_clean.to_csv('E:/Project_3/Recession_Prediction_Network_Analysis/data/
 ↪final_prepared_data.csv')

print("\nFinal prepared data saved to data/final_prepared_data.csv")
print("\nCleaned df_final head (after dropping NaNs):")
print(df_final_clean.head())
print("\nCleaned df_final info:")
df_final_clean.info()
```

```
Step 1.5: Final Data Preparation

Final prepared data saved to data/final_prepared_data.csv

Cleaned df_final head (after dropping NaNs):
            T10Y3MM_mom_change  T10Y3MM_yoy_change  T10Y3MM_roll12_mean  \
1961-01-31                 0.0                 0.0                 1.67
1961-02-28                 0.0                 0.0                 1.67
1961-03-31                 0.0                 0.0                 1.67
1961-04-30                 0.0                 0.0                 1.67
1961-05-31                 0.0                 0.0                 1.67

            T10Y3MM_roll12_std  ICSA_mom_change  ICSA_yoy_change  \
1961-01-31                 0.0              0.0              0.0
1961-02-28                 0.0              0.0              0.0
1961-03-31                 0.0              0.0              0.0
1961-04-30                 0.0              0.0              0.0
1961-05-31                 0.0              0.0              0.0

            ICSA_roll12_mean  ICSA_roll12_std  UNRATE_mom_change  \
1961-01-31          204000.0              0.0           0.000000
1961-02-28          204000.0              0.0           4.545455
1961-03-31          204000.0              0.0           0.000000
1961-04-30          204000.0              0.0           1.449275
1961-05-31          204000.0              0.0           1.428571

            UNRATE_yoy_change  …  INDPRO_roll12_std  CPILFESL_mom_change  \
1961-01-31          26.923077  …           0.596117             0.325733
1961-02-28          43.750000  …           0.584275             0.000000
1961-03-31          27.777778  …           0.550235             0.324675
1961-04-30          34.615385  …           0.494213             0.000000
1961-05-31          39.215686  …           0.438425             0.000000

            CPILFESL_yoy_change  CPILFESL_roll12_mean  CPILFESL_roll12_std  \
1961-01-31             0.983607             30.666667             0.088763
1961-02-28             0.653595             30.683333             0.093744
1961-03-31             0.980392             30.708333             0.108362
1961-04-30             0.980392             30.733333             0.115470
1961-05-31             0.980392             30.758333             0.116450

            SP500_mom_change  SP500_yoy_change  SP500_roll12_mean  \
1961-01-31          6.315605         11.095123          56.115833
1961-02-28          2.686954         13.043478          56.725833
1961-03-31          2.553592         17.564144          57.535833
1961-04-30          0.384261         20.121388          58.447499
1961-05-31          1.913949         19.219050          59.341666
```

```
            SP500_roll12_std  Recession
1961-01-31          2.256656        1.0
1961-02-28          3.092449        1.0
1961-03-31          3.871343        0.0
1961-04-30          4.320161        0.0
1961-05-31          4.811621        0.0

[5 rows x 49 columns]

Cleaned df_final info:
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 774 entries, 1961-01-31 to 2025-06-30
Freq: ME
Data columns (total 49 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   T10Y3MM_mom_change              774 non-null    float64
 1   T10Y3MM_yoy_change              774 non-null    float64
 2   T10Y3MM_roll12_mean             774 non-null    float64
 3   T10Y3MM_roll12_std              774 non-null    float64
 4   ICSA_mom_change                 774 non-null    float64
 5   ICSA_yoy_change                 774 non-null    float64
 6   ICSA_roll12_mean                774 non-null    float64
 7   ICSA_roll12_std                 774 non-null    float64
 8   UNRATE_mom_change               774 non-null    float64
 9   UNRATE_yoy_change               774 non-null    float64
 10  UNRATE_roll12_mean              774 non-null    float64
 11  UNRATE_roll12_std               774 non-null    float64
 12  PERMIT_mom_change               774 non-null    float64
 13  PERMIT_yoy_change               774 non-null    float64
 14  PERMIT_roll12_mean              774 non-null    float64
 15  PERMIT_roll12_std               774 non-null    float64
 16  UMCSENT_mom_change              774 non-null    float64
 17  UMCSENT_yoy_change              774 non-null    float64
 18  UMCSENT_roll12_mean             774 non-null    float64
 19  UMCSENT_roll12_std              774 non-null    float64
 20  VIXCLS_mom_change               774 non-null    float64
 21  VIXCLS_yoy_change               774 non-null    float64
 22  VIXCLS_roll12_mean              774 non-null    float64
 23  VIXCLS_roll12_std               774 non-null    float64
 24  USALOLITONOSTSAM_mom_change     774 non-null    float64
 25  USALOLITONOSTSAM_yoy_change     774 non-null    float64
 26  USALOLITONOSTSAM_roll12_mean    774 non-null    float64
 27  USALOLITONOSTSAM_roll12_std     774 non-null    float64
 28  PCE_mom_change                  774 non-null    float64
 29  PCE_yoy_change                  774 non-null    float64
 30  PCE_roll12_mean                 774 non-null    float64
 31  PCE_roll12_std                  774 non-null    float64
```

```
 32  CPIAUCSL_mom_change           774 non-null    float64
 33  CPIAUCSL_yoy_change           774 non-null    float64
 34  CPIAUCSL_roll12_mean          774 non-null    float64
 35  CPIAUCSL_roll12_std           774 non-null    float64
 36  INDPRO_mom_change             774 non-null    float64
 37  INDPRO_yoy_change             774 non-null    float64
 38  INDPRO_roll12_mean            774 non-null    float64
 39  INDPRO_roll12_std             774 non-null    float64
 40  CPILFESL_mom_change           774 non-null    float64
 41  CPILFESL_yoy_change           774 non-null    float64
 42  CPILFESL_roll12_mean          774 non-null    float64
 43  CPILFESL_roll12_std           774 non-null    float64
 44  SP500_mom_change              774 non-null    float64
 45  SP500_yoy_change              774 non-null    float64
 46  SP500_roll12_mean             774 non-null    float64
 47  SP500_roll12_std              774 non-null    float64
 48  Recession                     774 non-null    float64
dtypes: float64(49)
memory usage: 302.3 KB
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 774 entries, 1961-01-31 to 2025-06-30
Freq: ME
Data columns (total 49 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   T10Y3MM_mom_change            774 non-null    float64
 1   T10Y3MM_yoy_change            774 non-null    float64
 2   T10Y3MM_roll12_mean           774 non-null    float64
 3   T10Y3MM_roll12_std            774 non-null    float64
 4   ICSA_mom_change               774 non-null    float64
 5   ICSA_yoy_change               774 non-null    float64
 6   ICSA_roll12_mean              774 non-null    float64
 7   ICSA_roll12_std               774 non-null    float64
 8   UNRATE_mom_change             774 non-null    float64
 9   UNRATE_yoy_change             774 non-null    float64
 10  UNRATE_roll12_mean            774 non-null    float64
 11  UNRATE_roll12_std             774 non-null    float64
 12  PERMIT_mom_change             774 non-null    float64
 13  PERMIT_yoy_change             774 non-null    float64
 14  PERMIT_roll12_mean            774 non-null    float64
 15  PERMIT_roll12_std             774 non-null    float64
 16  UMCSENT_mom_change            774 non-null    float64
 17  UMCSENT_yoy_change            774 non-null    float64
 18  UMCSENT_roll12_mean           774 non-null    float64
 19  UMCSENT_roll12_std            774 non-null    float64
 20  VIXCLS_mom_change             774 non-null    float64
 21  VIXCLS_yoy_change             774 non-null    float64
 22  VIXCLS_roll12_mean            774 non-null    float64
```

```
 23  VIXCLS_roll12_std             774 non-null    float64
 24  USALOLITONOSTSAM_mom_change   774 non-null    float64
 25  USALOLITONOSTSAM_yoy_change   774 non-null    float64
 26  USALOLITONOSTSAM_roll12_mean  774 non-null    float64
 27  USALOLITONOSTSAM_roll12_std   774 non-null    float64
 28  PCE_mom_change                774 non-null    float64
 29  PCE_yoy_change                774 non-null    float64
 30  PCE_roll12_mean               774 non-null    float64
 31  PCE_roll12_std                774 non-null    float64
 32  CPIAUCSL_mom_change           774 non-null    float64
 33  CPIAUCSL_yoy_change           774 non-null    float64
 34  CPIAUCSL_roll12_mean          774 non-null    float64
 35  CPIAUCSL_roll12_std           774 non-null    float64
 36  INDPRO_mom_change             774 non-null    float64
 37  INDPRO_yoy_change             774 non-null    float64
 38  INDPRO_roll12_mean            774 non-null    float64
 39  INDPRO_roll12_std             774 non-null    float64
 40  CPILFESL_mom_change           774 non-null    float64
 41  CPILFESL_yoy_change           774 non-null    float64
 42  CPILFESL_roll12_mean          774 non-null    float64
 43  CPILFESL_roll12_std           774 non-null    float64
 44  SP500_mom_change              774 non-null    float64
 45  SP500_yoy_change              774 non-null    float64
 46  SP500_roll12_mean             774 non-null    float64
 47  SP500_roll12_std              774 non-null    float64
 48  Recession                     774 non-null    float64
dtypes: float64(49)
memory usage: 302.3 KB
```