Exploring relationships between real estate & financial indicators

A Shaolin AI project by Jim, Donny and Athen

Our primary goal:

To gather, plot, and compare various data from the Federal Reserve, financial markets, and real estate markets to look for correlations and trends.

Our Hypothesis:

Macroeconomic indicators like the Fed's balance sheet will correlate with asset prices.

- Federal funds rate vs Home prices
- Balance sheet vs Home prices
- S&P 500 vs Home prices

Which indicators can you use to most reliably Predict housing prices going forward?

This was the burning question at the back of our minds as we started looking for data.

Our Process

How did we do it? Each of us had separate responsibilities for the data.

- ➔ **Gather**
  Gather data, clean it and organize it for visualization
- ➔ **Plot**
  Plotting the data with python libraries to visually understand the relationships
- ➔ **Analyze**
  Analyze the plots we produced to check for correlations

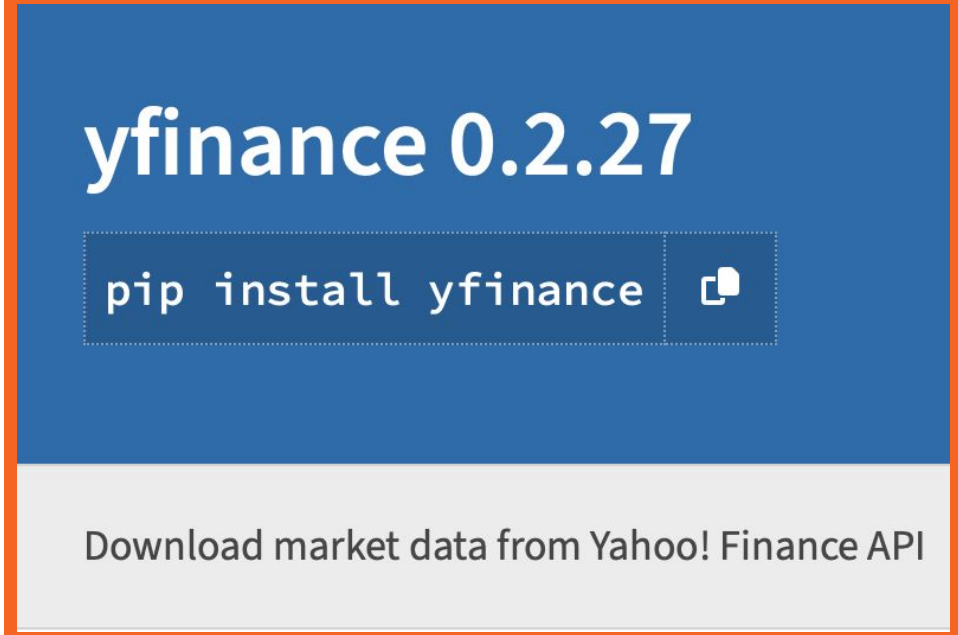
Where do we start?

We first assessed different options for pulling data from the internet.

- Yahoo Finance (yfinance)
- Zillow
- FRED (federal reserve API)
- IRS

Yahoo Finance: The first snag

- ➔ Yahoo finance no longer has an API
- ➔ Thankfully, python had our back.
- ➔ A fully functioning library that uses screen scraping to pull live data from Yahoo Finance
- ➔ Arguably easier than an API



Yfinance in a Nutshell

- **Importing libraries**
  Athen started by importing necessary libraries to make comparisons with data from yfinance

- **Making calls**
  Calling them “calls” feels a bit like cheating, but it really was as simple as creating a variable (data), using yf.download, pick a ticker symbol, choose your dates and intervals, sort it by datetime, and filter out to the last day of the month.

```
import yfinance as yf
import plotly.express as px
import pandas as pd
import hvplot.pandas
import matplotlib.pyplot as plt
import numpy as np
from pathlib import Path
import seaborn as sns
```

This was done with GOLD, SP500, bonds and the dollar

```
data = yf.download('GOLD', start="2000-01-01", end="2023-07-30", interval='1d')
data.index = pd.to_datetime(data.index)
data = data.resample("M").last()
```

```
**********100%**********]  1 of 1 completed
```

FRED and cleaning

- **Making API Calls**
  Jim used the following code to pull data from FRED
- **Organizing and cleaning data**
  This is where Jim spent most of his nights. These CSV's were monstrous (we're talking over 22,000 columns),, and Jim made them much more manageable.

```
# FRED API library and key import
from fredapi import Fred

api_key = os.getenv("FRED_API_KEY")
type(api_key)

fred = Fred(api_key=api_key)
```

Fed Total Assets (WALCL)

```
# FRED API import, "frequency = 'm'" argument to get mont
total_assets = fred.get_series('WALCL', frequency='m')
total_assets
```

```
# Remove unnecessary columns, set Region Name to index
city_condo_smoothed = city_condo_smoothed.drop(['RegionID',
                                                'SizeRank',
                                                'RegionType',
                                                'StateName',
                                                'State',
                                                'Metro',
                                                'CountyName'
                                               ],
                                              axis=1).set_index('RegionName')
# Transpose data frame to time series by rows, matching FRED data
city_condo_smoothed_by_date = city_condo_smoothed.transpose()

# reset index and rename columns to prepare to set index to 'date'
city_condo_smoothed_by_date = city_condo_smoothed_by_date.reset_index()
city_condo_smoothed_by_date = city_condo_smoothed_by_date.rename(columns = {'index': 'date'})

#convert 'date' strings to datetime objects
city_condo_smoothed_by_date['date'] = pd.to_datetime(city_condo_smoothed_by_date['date'])

#set index to 'date' and drop previous sequential index
city_condo_smoothed_by_date = city_condo_smoothed_by_date.set_index('date', drop = True)
city_condo_smoothed_by_date
```

Lots of gathering

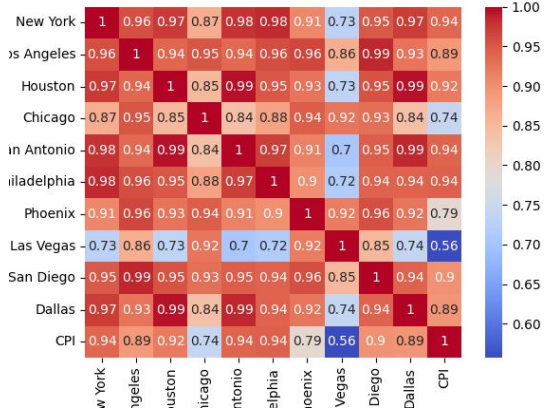
➔ **Locating the data**

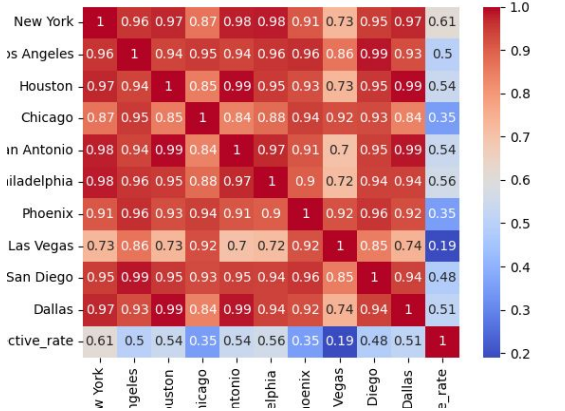
Donny spent time browsing Zillow and figuring out how to download a plethora of data on housing prices over the years.

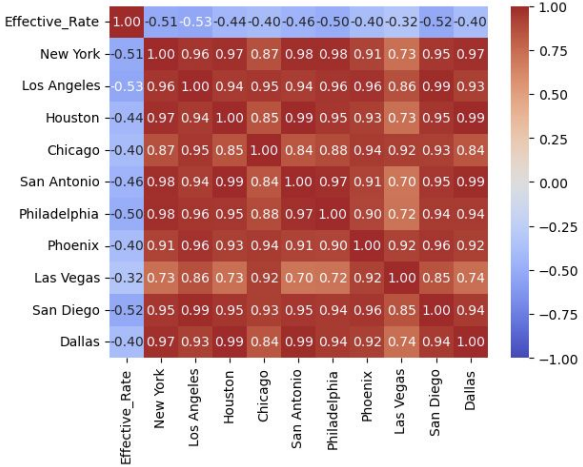
	RegionID	SizeRank	RegionName	RegionType	StateName	2000-01-31	2000-02-29	2000-03-31
1	102001	0	United States	country		116158.74119205348	116363.03351912454	116616.2319613347
2	394913	1	New York, NY	msa	NY	189174.75031153686	189979.8136852024	190791.0854275051
3	753899	2	Los Angeles, CA	msa	CA	227805.133377966	228647.76141078054	229773.6082815101
4	394463	3	Chicago, IL	msa	IL	145453.57212736708	145591.48297396602	145851.01265318235
5	394514	4	Dallas, TX	msa	TX	131047.22581238096	131112.80299049383	131186.91103907741
6	394692	5	Houston, TX	msa	TX	119115.06475485285	119134.05636326135	119048.46444145702
7	395209	6	Washington, DC	msa	VA	176874.6875789701	177018.5131136344	177283.12742728335
8	394974	7	Philadelphia, PA	msa	PA	116677.94914909739	116938.20534943193	117115.70987104376
9	394856	8	Miami, FL	msa	FL	105938.18570943388	106230.2785773084	106538.26064614777
10	394347	9	Atlanta, GA	msa	GA	143017.28118021687	143346.10832419002	143734.32659021765
11	394404	10	Boston, MA	msa	MA	215184.78442591612	216052.8429473263	217002.28130705192
12	394976	11	Phoenix, AZ	msa	AZ	145122.34306197552	145417.63787983608	145806.09553063873
13	395057	12	San Francisco, CA	msa	CA	280427.82541527745	281526.585314611	283198.6684373682
14	395025	13	Riverside, CA	msa	CA	138115.2407699303	138680.92238958646	139228.00617169283
15	394532	14	Detroit, MI	msa	MI	119708.6633039623	119834.81800344346	120155.24098048142
16	395078	15	Seattle, WA	msa	WA	223384.48898613206	224091.36376022676	224783.8708101896
17	394865	16	Minneapolis, MN	msa	MN	145151.73627189902	145631.65463814887	146061.48095970365
18	395056	17	San Diego, CA	msa	CA	219063.88976502157	219924.81179422262	220971.31285087718

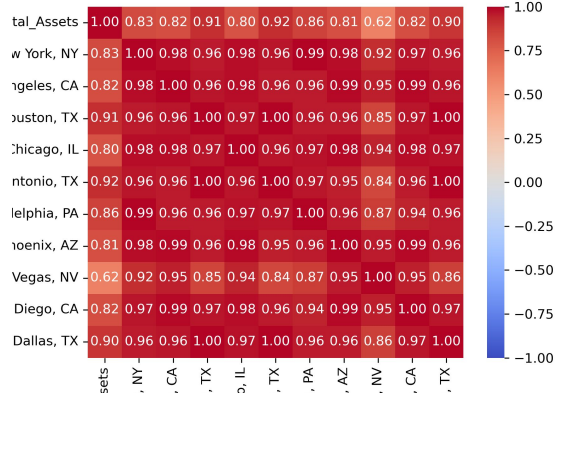
We made some heatmaps…

The first things we spent some time on were heatmaps showing correlation of multiple metrics in comparison with housing prices.





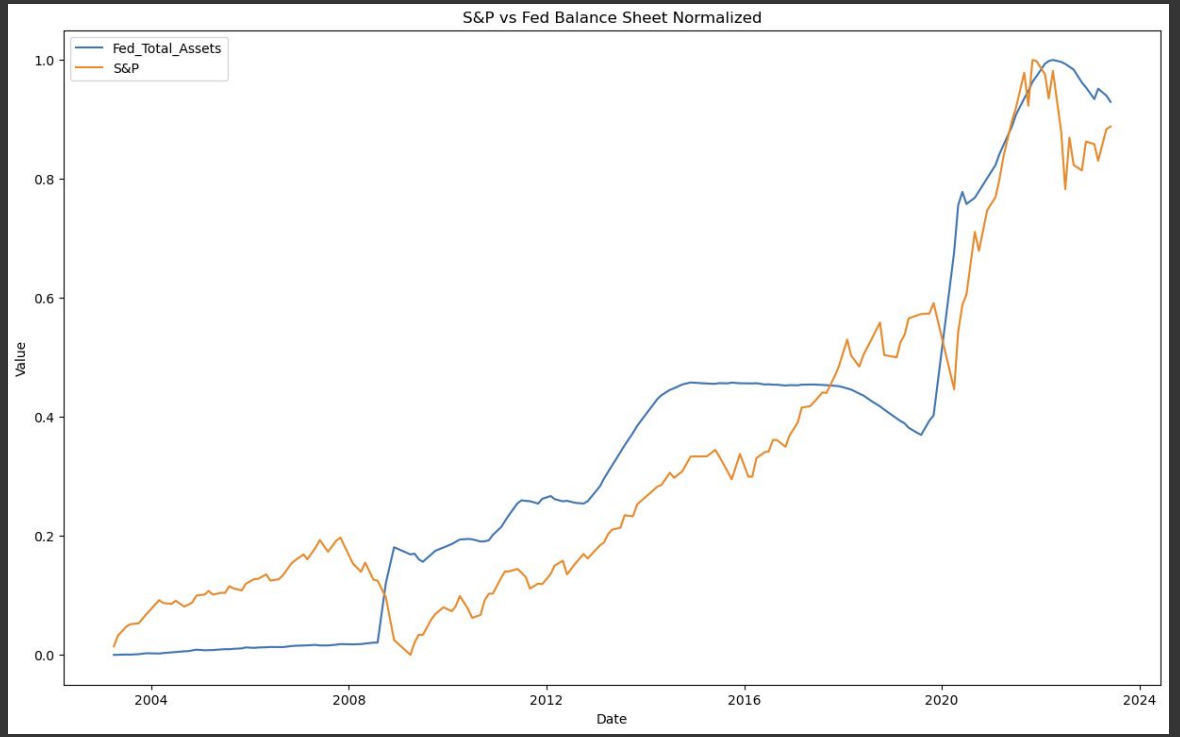




The First Observation:
S&P 500 vs the Fed Balance Sheet.

We found a correlation between the two after normalization.

The Fed balance sheet depicts Quantitative Easing or what is often called “money printing”

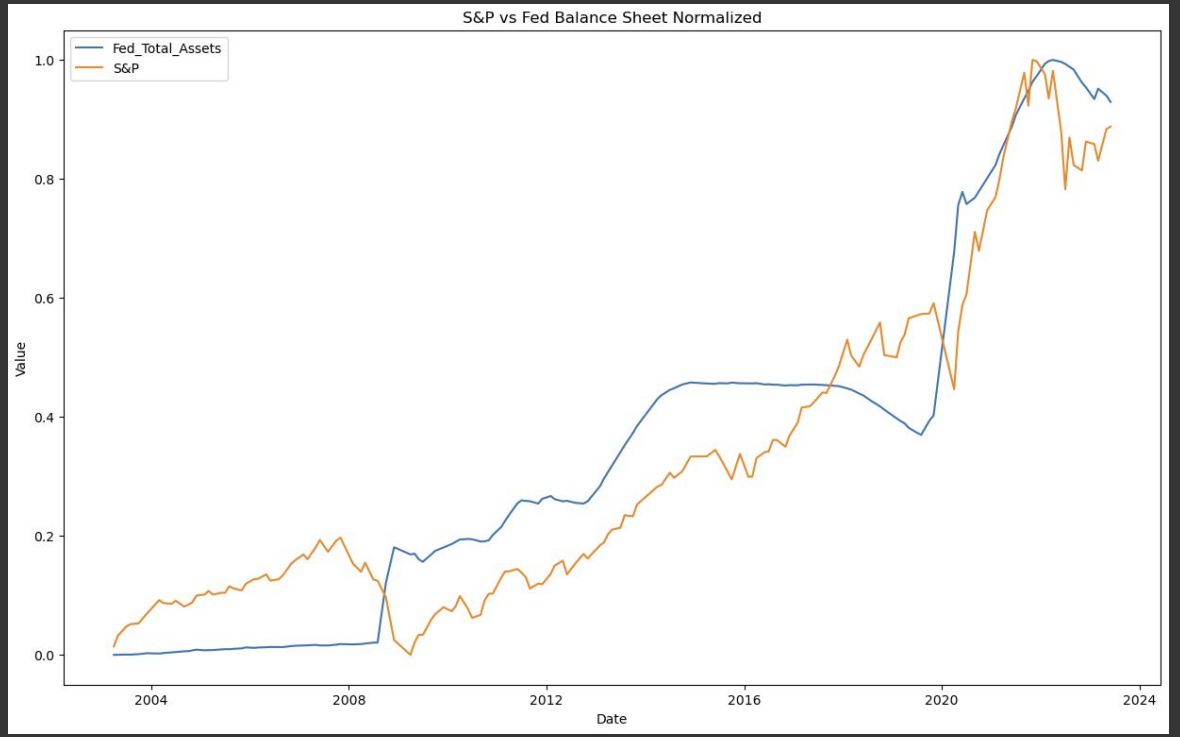


What is Quantitative Easing (QE)?

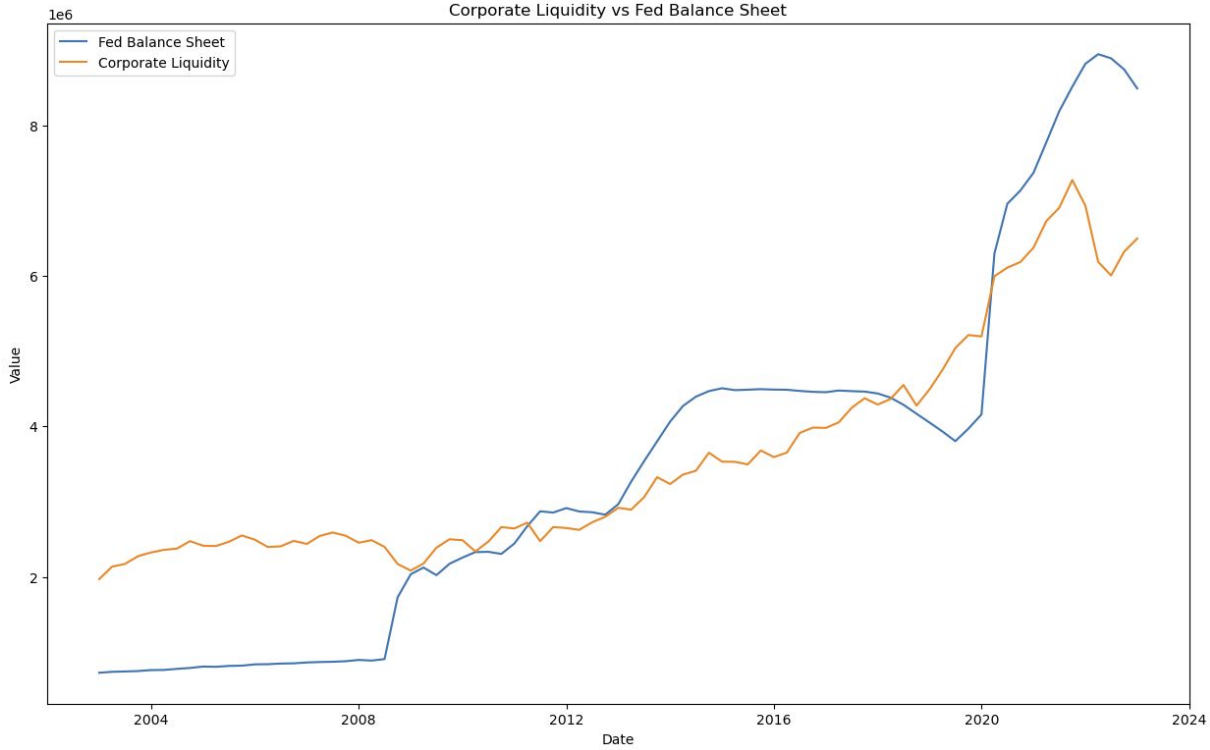
A monetary policy where a central bank, such as the Federal Reserve, commits to purchasing a set quantity of bonds or other assets to stimulate the economy. **These bonds get added to the Fed's Balance Shee**t.

This **increases the supply of money** and encourages the market to hold long term or risky assets instead of cash.

QE began in 2008 after the Great Financial Crisis and has mostly continued nonstop.



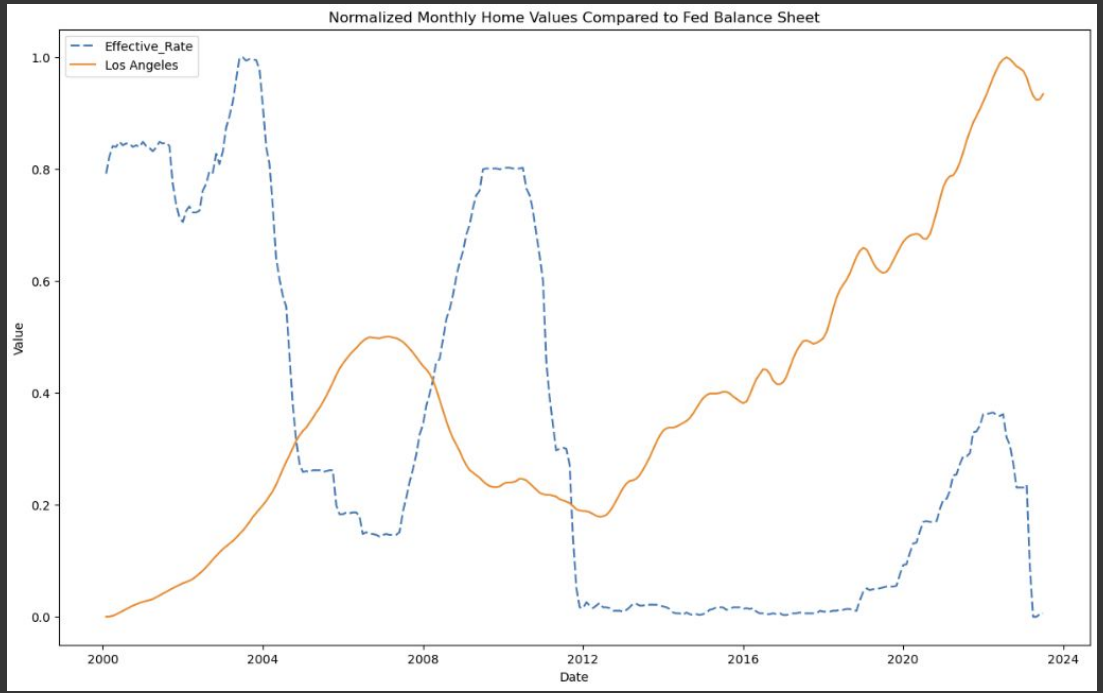
Corporate Assets and the Fed



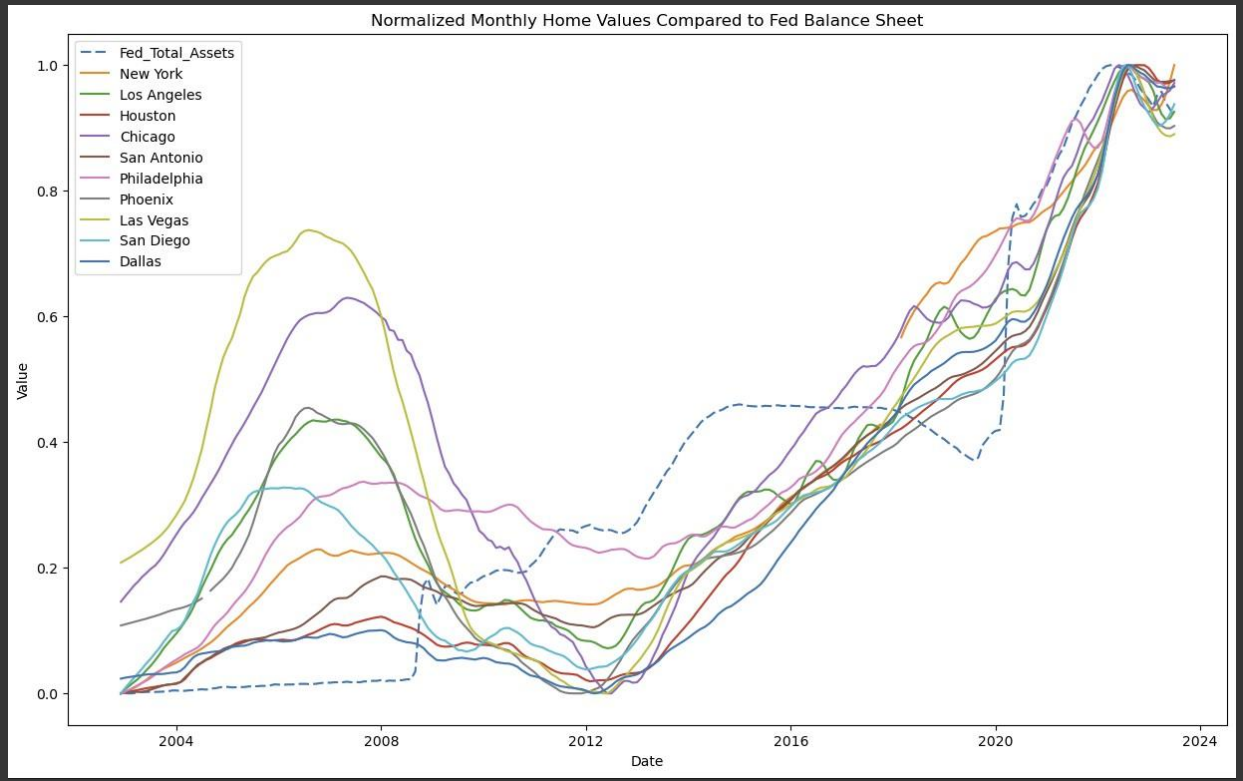
Corporate liquid assets are also correlated with the fed's balance sheet.

Interest Rates are Semi-correlated.

When delayed by three years, interest rates show an inverse correlation with asset prices before the QE era (which is normal), but after QE began the inverse correlation was suppressed.

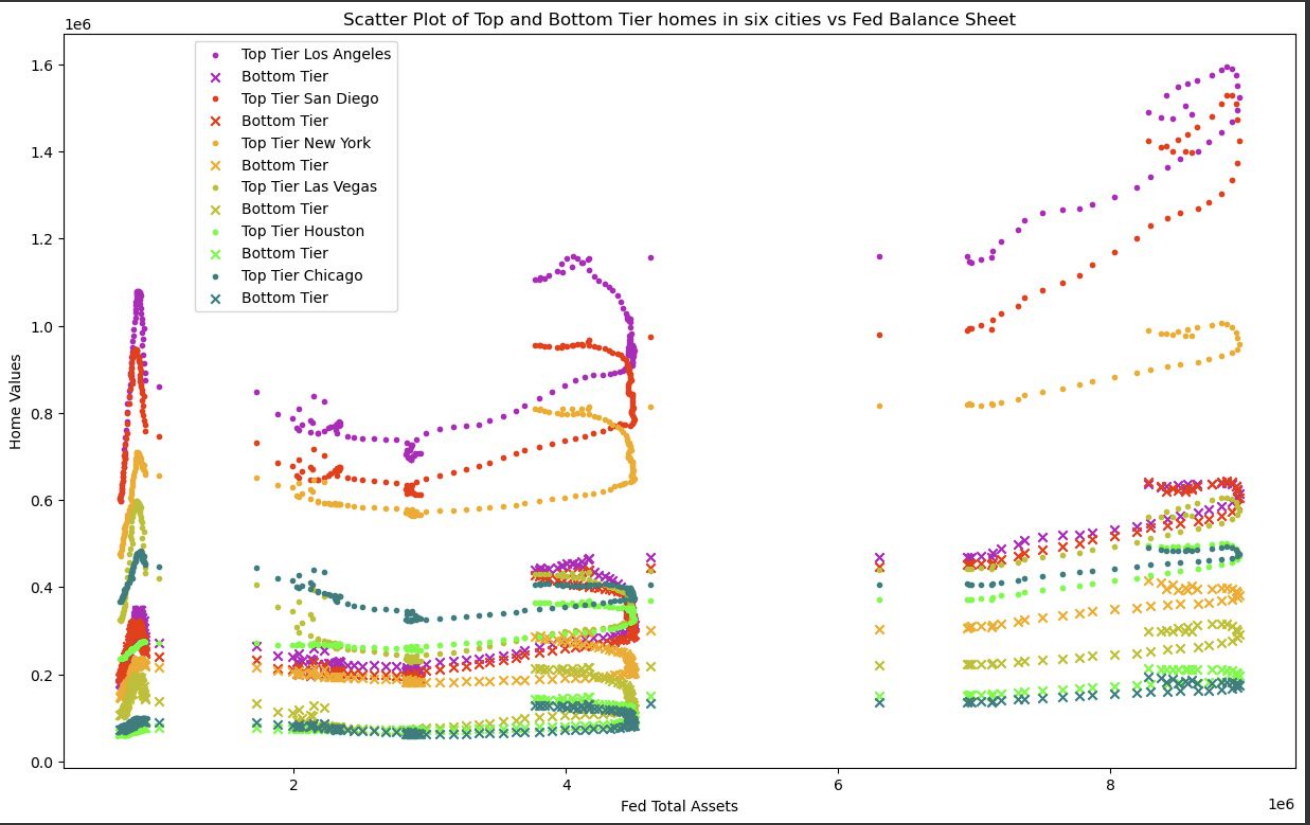


We plotted average home prices in ten major cities and compared them to the fed balance sheet.



Starting in 2012, home values became highly correlated with the fed balance sheet.

Second Question: can we predict which homes gain more than others?



We charted the top & bottom tiers of housing of six major cities against the Fed balance sheet.

This chart shows how some cities were more correlated to the Fed than others, especially in the top tier homes of certain cities.

For each city, Top Tier contains homes within the 65th and 95th percentile range, while Bottom Tier contains those within the 5th and 35th percentile.