*Jorge Basilio | Student ID: 011149553*

*NY Housing Price Predictor*

WGU - C964 – Computer Science Capstone

April 26, 2024

[Part A: Letter of Transmittal 2](#_Toc165040194)

[Part B: Project Proposal Plan 5](#_Toc165040195)

[Project Summary 5](#_Toc165040196)

[Data Summary 6](#_Toc165040197)

[Implementation 6](#_Toc165040198)

[Timeline 6](#_Toc165040199)

[Evaluation Plan 6](#_Toc165040200)

[Resources and Costs 6](#_Toc165040201)

[Part C: Application 8](#_Toc165040202)

[Part D: Post-implementation Report 9](#_Toc165040203)

[Solution Summary 9](#_Toc165040204)

[Data Summary 9](#_Toc165040205)

[Machine Learning 9](#_Toc165040206)

[Validation 9](#_Toc165040207)

[Visualizations 9](#_Toc165040208)

[User Guide 9](#_Toc165040209)

[Reference Page 11](#_Toc165040210)

# Part A: Letter of Transmittal

April 26, 2024

John Doe, CTO

Financial Investment Corp

123 Financial Rd.

NYC, New York

# Dear Mr. Doe,

# I hope this letter finds you well. Thank you for considering our proposal aimed at addressing Financial Investment Corp's recent setbacks in real estate investments within the New York area. We understand that your company has had difficulties with the real estate market volatility and have loss a substantial amount of money on recent real estate investment. That is why we are excited about the opportunity to collaborate and present a solution that we believe will be transformative for your organization.

# At the heart of our proposal is a data product meticulously crafted to leverage real-time insights from the US Federal Reserve and comprehensive house price data sourced from Zillow. Our tool is not just a solution, but a strategic asset designed to provide invaluable support to your decision-making processes, particularly in navigating the complexities of the real estate market.

# Here's how our data product benefits you and supports your decision-making process:

# Firstly, it offers enhanced insight into the current direction of the housing market, enabling you to make informed decisions regarding investment property purchases or sales. By basing profit margins on the average price our model predicts, you gain a competitive edge in optimizing your investments.

# Secondly, our product offers flexibility, allowing you to seamlessly switch between viewing price markets of all houses in New York or specific states. This versatility empowers you to tailor your strategies according to varying market dynamics.

# The foundation of our data product lies in the wealth of data we utilize:

# US Federal Reserve Data: Providing real-time economic indicators crucial for understanding broader market trends.

# Zillow House Price Data: Offering comprehensive insights into localized housing market dynamics, aiding in precise decision-making.

# Furthermore, our project is guided by clear objectives and hypotheses:

# Objectives: To deliver up-to-date and precise US real estate pricing data, facilitating informed decision-making for maximum returns on investments.

# Hypotheses: That leveraging predictive analytics on historical data will enhance accuracy in forecasting future market trends.

# In terms of methodology, our project follows a robust framework:

# Data Collection: Gathering and preprocessing relevant datasets from the US Federal Reserve and Zillow.

# Data Preparation:

# Model Development: Employing a Random Forest Classifier algorithm to analyze historical data and predict future market trends.

# Visualization and Presentation: Creating intuitive visualizations to effectively communicate insights to stakeholders.

# To bring this project to fruition, we seek an initial investment of $20,000, with an additional $1,000 per year for maintenance. Rest assured; our chosen developer possesses the requisite expertise:

# Developer Profile: Our developers boast several years of experience in building data applications employing machine-learning algorithms, complemented by a bachelor’s degree in computer science.

# Moreover, we are committed to upholding ethical and legal standards in handling sensitive data. Stringent precautions will be taken to ensure compliance with relevant regulations and safeguard the privacy of individuals.

# In conclusion, the implementation of our data product holds immense potential for your organization, promising increased returns and paving the way for expansion into new real estate markets. We are eager to embark on this journey with you and remain at your disposal to address any queries or concerns.

# Thank you once again for considering our proposal.

# Sincerely,

# Jorge Basilio

# Part B: Project Proposal Plan

# Executive Summary

**Problem Statement**

Financial Investment Corps have had many losses in their real estate investment. Due to the volatility of the market, they wish to minimize their losses by finding a tool that could assisted them better predict the housing market prices. Recently the new craze is machine learning and AI development and they have decided to make an investment, in a machine learning tool that will be assist them and help them analyze the market. This will help them decide if they should sell a property or buy a property.

Financial Investment Corps have had many companies’ approaches with idea on a machine learning tool, but the data is usually dummy data. They want a tool that uses actual data from the federal reserve and live housing market data. This will allow them to test out the tool and see if it will make a difference in their investment portfolios. The tool should be able to predict the future housing market. This will allow them to base their decisions on this data.

**Solution to Customer Needs**

I believe that our Housing Price predictor will be a game changer to Financial Investment Corp who is one of the largest real estate investments agencies in New York. They mostly purchase and sell real estate based off location, square footage, and the living space. This tool aims to offer invaluable insights, enabling Financial Investment Corp to make informed decisions regarding investment property purchases or sales. By predicting housing market directions and basing profit margins on these predictions, our tool enhances decision-making processes and facilitates more successful investments. Our proposed data product harnesses real-time data from the US Federal Reserve and comprehensive housing price data sourced from Zillow. This will allow us to train and test our machine learning model on previous housing prices.

**Existing Gaps in Data**

While our tool boasts remarkable capabilities, there are areas for further enhancement. Presently, it does not extract data on individual houses or cities, limiting the granularity of analysis. Additionally, the exclusion of inflation from our model was a necessary simplification to facilitate accurate predictions solely based on housing prices. However, this aspect could be integrated in future iterations to provide a more comprehensive understanding of market dynamics. Furthermore, our tool does not currently factor in external influences such as negative news that could impact the housing market, presenting an opportunity for refinement to ensure a more holistic predictive model.

**Data**

The data required for our product includes real-time economic indicators from the US Federal Reserve and comprehensive housing price data from Zillow. These datasets are crucial for training and testing our machine learning model, enabling accurate predictions of future market trends.

US Federal Reserve Data

* + (Weekly) 30 – year fixed mortgage average in the United States – measure mortgage interest rates
  + (Monthly) Consumer Price Index for all urban consumers – measures inflation
  + (Quarterly) Rental Vacancy Rate in the United States – measures the percentage of available rental properties.

Zillow Housing Data

* + (Weekly) Median sale price (Raw, All Homes, Weekly) – the median sale price for US houses
  + (Monthly) Zillow Home Value Forecast (ZHVF) – Zillow’s home value index

All three US federal reserve data are downloadable in CSV format. I did format all the dates just in case they were in different formats. Also, since all the dates are all unique, I decided to use them as the index. Each file after making them into data frames consist of the data below.

|  |  |
| --- | --- |
| Date | Value |
| Year – Month – Day | Percentage % |

The Zillow Data is different and consist of RegionId, SizeRank, RegionName, StateName, and Dates. The dates start at date 1 and end with date 15. The median sale price has 15 weeks of data (date), and the Zillow home forecast has 15 months of data (date).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RegionID | SizeRank | RegionName | RegionType | StateName | Date 1… | … Date 15 |
| ID | Size of State | Name | Country or State | State abbreviation | Price 1 | Price 15 |

Since there are five CSV files and three different date incrementation. The federal data will have to be merged, on the date which is the index. After the data is merged, some dates will have a Null value. To fix some of the missing date values, we will use a fill forward method.

* Ex. (1, 2, Null, Null, 4, 5, Null) 🡪 (1, 2, 2, 2, 4, 5, 5)

This will allow us to fill in the missing data. Any other null. Values that are still null will be removed because the dates are too singular and too early and cannot be combined because the dates do not correlate.

The Zillow data are simple, but the first 5 columns are unnecessary. Once a specific state or the entirety of the US is chosen, all we need is the median housing prices for the given dates. Then we will have to switch the columns of the Zillow dates and make them into rows. This will allow us to merge the Zillow and the US federal reserve data.

Ethical and legal concerns regarding the data are minimal, as the datasets are publicly available and obtained from reputable sources like the US Federal Reserve and Zillow. However, it's essential to ensure compliance with data privacy regulations and usage policies outlined by these organizations. Additionally, steps are taken to handle data anomalies such as outliers or incomplete data, ensuring the integrity and accuracy of the predictive model.

**Project Methodology**

We will be using the Waterfall method. "The waterfall methodology is a project management approach that emphasizes a linear progression from beginning to end of a project.” (“Waterfall Methodology: Project Management | Adobe Workfront”). The Waterfall method is like a water fall and goes from top to bottom, phases of the project are (Requirement, Analysis, Design, Coding and Implementation, Testing, Deployment, Maintenance)

1. Requirement:
   * Define project objectives, scope, and the company’s stakeholders needs.
   * Gather and document detailed requirements through interviews, and analysis of existing systems or processes.
2. Analysis:

* Analyze gathered requirements to ensure clarity, completeness, and feasibility.
* Identify potential risks and constraints.
* Develop use cases, user stories, or functional specifications to guide the design phase.

1. Design:

* Architect the solution based on analyzed requirements.
* Create system architecture diagrams, data flow diagrams, and other relevant documentation.
* Design user interfaces, databases (if used), algorithms, and other system components.

1. Coding and Implementation:

* Translate design specifications into code.
* Develop and integrate software components according to the design.
* Follow coding standards, best practices, and version control procedures.

1. Testing:

* Conduct thorough testing to verify that the software meets requirements and functions as expected.
* Perform unit testing, integration testing, system testing, and acceptance testing.
* Identify and address defects, bugs, or discrepancies.

1. Deployment:

* Prepare the software for deployment in the production environment.
* Coordinate with the company’s stakeholders to plan and execute deployment activities.
* Monitor the deployment process and address any issues that arise.

1. Maintenance:

* Provide ongoing support and maintenance to ensure the continued functionality and performance of the deployed solution.
* Address user feedback, bugs, and enhancements through iterative updates.
* Monitor system performance and scalability, adjusting as necessary.

By adhering to the Waterfall methodology, the project team can systematically progress through each phase, ensuring that requirements are met, risks are mitigated, and the final solution aligns with stakeholders' expectations.

**Project Outcomes**

Given that the project adheres to the Waterfall methodology, each phase must be completed before proceeding to the next. The project can be delineated into three distinct sections.

Firstly, the planning phase encompasses requirements gathering, analysis, and design. This stage primarily involves data collection, strategic planning, and ideation on the project's structure. Planning activities include creating wireframes, formulating user stories, and ensuring comprehensive requirement coverage.

The subsequent phase, which consumes much of the project timeline, encompasses coding, implementation, testing, and deployment. Here, the focus shifts to active development, with teams segmented to address specific project components.

Lastly, the maintenance and post-deployment testing phase entail ongoing support and refinement. This involves addressing updates, addressing bugs identified post-deployment, and ensuring the sustained functionality and performance of the project.

**Implementation Plan**

1. Plan Appropriately - To ensure a successful software implementation, it's essential to develop a comprehensive project plan covering all stages of the release cycle. Begin by assessing company needs and mapping operational processes to meet functional and technical requirements. Install, customize, and configure the software while integrating it with existing systems. Conduct thorough testing, including usability testing, to identify and fix bugs before deployment. Train key users to ensure awareness and proficiency with the new system and prepare for the go-live phase with quality assurance checks and optimization of product functionality.
2. Define Requirements to Get a Clear Vision - Clear articulation of project goals, objectives, and expected outcomes is paramount. Conduct a project discovery phase to transform ideas into a clear vision with defined features and requirements. Develop prototypes, user navigation maps, and software requirements specifications to guide implementation effectively.
3. Assign Responsible Team Members to Track Progress - Establish clear communication channels and regular coordination meetings to track project progress. Assign roles and responsibilities to team members, ensuring accountability and effective resource management. Utilize feedback loops to address risks and bottlenecks promptly and maintain a positive work atmosphere.
4. Launch the Test Model for the Project - Initiate usability testing to identify software bugs and risks. Fix bugs and perform regression tests to ensure software stability. Set up a simulation environment to optimize product functionality before deployment.
5. Prepare a Software Release - Complete necessary changes and optimize product functionality within a simulation environment. Conduct quality assurance checks with input from colleagues outside the project team. Evaluate integrations and make necessary adjustments to improve functionality.
6. Implement the Product - Deploy the software release and perform additional testing to ensure operational readiness in a real environment. Evaluate integrations and make necessary adjustments to improve functionality. Follow best practices to avoid common mistakes in software implementation, such as lack of communication, unclear schedules, and premature go-live dates.

**Timeline and Milestones**

This project should take approximately two months to complete starting from April 1st to May 31st. The total time frame will be approximately 96 hours spent working on the project from start to finish.

|  |  |  |  |
| --- | --- | --- | --- |
| **Milestone or deliverable** | **Duration**  **(hours or days)** | **Projected start date** | **Anticipated end date** |
| Requirement Analysis | 8 hours | April 1 | April 10 |
| (Design) Wire frame and User Story | 20 hours | April 11 | April 19 |
| Coding and Implementation | 40 hours | April 20 | May 9 |
| Testing | 20 hours | May 10 | May 19 |
| Deployment | 8 hours | May 20 | May 31 |

**Evaluation Plan**

The evaluation plan for the housing price predictor encompasses a comprehensive assessment of various key performance indicators (KPIs) across different categories. Financial metrics such as Return on Investment (ROI) and Cost Performance Index (CPI) will evaluate the project's profitability and cost efficiency. Customer satisfaction will be measured through metrics like Net Promoter Score (NPS) and Customer Effort Score, ensuring that the housing price predictor meets user expectations and delivers value. Product engagement will be monitored using indicators such as User Adoption and Usage Time to gauge user interaction and satisfaction levels. Agile metrics like Lead Time and Velocity will assess team productivity and efficiency in delivering project goals. Deliverables and timeline metrics will track the quantity and timeliness of project outputs, while risk management strategies will mitigate potential risks and ensure project success.

By systematically evaluating these KPIs, the housing price predictor can effectively measure its progress, identify areas for improvement, and ensure alignment with project objectives. Through ongoing monitoring and analysis, the project team can optimize processes, address challenges proactively, and deliver a high-quality product that meets the needs of stakeholders and end-users.

**Resources and Costs**

The Cost for this project is relatively minimal because it uses free data sourced from the US federal reserve and Zillow. The Visual aspect is used by Jupyter notebook and google Colabs which is also free. The main cost will be the equipment, labor, and maintenance cost.

* Programming Environment – The project software requirement’s is relatively free. There are several options when running the program. The costly one is running the program on PyCharm, which is not free. Another option is running the program on google colabs. Other than the environment, you need python 3 which is the programming language. Git to make sure our developers are collaborating and backing up the code. There are several libraries need but this is free.
* Environment Cost – The cost for the environment is around $5,000. This is for the rent on the location where their developers will be working. This also includes the electricity and internet cost. Another cost will be the computer for the developers, which will be around $1,000 each.

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Hourly Rate** | **Time** | **Total** |
| Planning & Design | $100 | 28 hours | $2,800.00 |
| Implementation and Integration | $100 | 40 hours | $4,000.00 |
| Testing | $100 | 20 hours | $2,000.00 |
| Deployment and Maintenance | $50 | 8 hours | $400.00 |

**Total Cost: $9,200.00**

**Total Time: 96 hours**

# Reference Page for (Parts A & B)

1. Olsen S. Zillow Home Value Index Methodology, 2023 Revision: What’s Changed? Zillow. Published February 11, 2023. Accessed May 4, 2024. <https://www.zillow.com/research/methodology-neural-zhvi-32128/>
2. Adobe Communications Team. Waterfall Methodology: Project Management | Adobe Workfront. Adobe.com. Published 2022. Accessed May 4, 2024. <https://business.adobe.com/blog/basics/waterfall#:~:text=The%20Waterfall%20methodology%20%E2%80%94%20also%20known,before%20the%20next%20phase%20begins.>
3. Lutkevich B, Lewis S. waterfall model. Software Quality. Published 2022. Accessed May 7, 2024. <https://www.techtarget.com/searchsoftwarequality/definition/waterfall-model>
4. The Four Phases of Project Management. Harvard Business Review. Published November 3, 2016. Accessed May 7, 2024. <https://hbr.org/2016/11/the-four-phases-of-project-management>
5. Sursaieva A. Main Steps in The Software Implementation Plan. Axon.dev. Published April 5, 2024. Accessed May 7, 2024. <https://www.axon.dev/blog/main-steps-in-the-software-implementation-plan>
6. KMS Solutions. How To Evaluate Your Software Development Project Success. Kms-solutions.asia. Published August 18, 2022. Accessed May 7, 2024. <https://blog.kms-solutions.asia/how-to-evaluate-your-software-development-project-success>

# Part C: Application

Application files will be added with this document. Steps below shows to option for running the project.

# Part D: Post-implementation Report

# Problem & Solution Summary

Financial Investment Corp (FIC) which is facing significant losses in real estate investments due to market volatility, needing a solution to predict housing market prices accurately. Our proposed solution, the Housing Price Predictor, leverages real-time insights from the US Federal Reserve and comprehensive housing price data from Zillow. By harnessing predictive analytics and machine learning algorithms, this tool offers invaluable support to FIC's decision-making processes in navigating the complexities of the real estate market.

The Housing Price Predictor provides several key benefits to FIC:

1. **Enhanced Market Insight**: By analyzing real-time economic indicators and localized housing market dynamics, the tool offers enhanced insight into the direction of the housing market, enabling informed decisions regarding investment property purchases or sales.
2. **Flexibility and Customization**: FIC can seamlessly switch between viewing price markets of all houses in New York or specific states, empowering them to tailor their strategies according to varying market dynamics.

Our solution is guided by clear objectives and hypotheses, aiming to deliver up-to-date and precise US real estate pricing data to facilitate informed decision-making for maximum returns on investments. Through a robust methodology involving data collection, preparation, model development, and visualization, we ensure the accuracy and reliability of our predictive model.

In conclusion, the Housing Price Predictor holds immense potential for FIC, promising increased returns and facilitating expansion into new real estate markets. By leveraging real-time data and advanced analytics, our solution empowers FIC to make informed decisions and minimize losses in real estate investments.

**Data Summary**

**Datasets**

The project uses Live Data from the federal reserve and Zillow. Listed below are the sites where you can get the most recent Data in CSV format.

US Federal Reserve Data

* + (Weekly) 30 – year fixed mortgage average in the United States – measure mortgage interest rates | <https://fred.stlouisfed.org/series/MORTGAGE30US>
  + (Monthly) Consumer Price Index for all urban consumers – measures inflation | <https://fred.stlouisfed.org/series/CPIAUCSL>
  + (Quarterly) Rental Vacancy Rate in the United States – measures the percentage of available rental properties. | <https://fred.stlouisfed.org/series/RRVRUSQ156N>

A graph on a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A graph on a computer screen

Description automatically generated

Zillow Housing Data

* + (Weekly) Median sale price (Raw, All Homes, Weekly) – the median sale price for US houses
  + (Monthly) Zillow Home Value Forecast (ZHVF) – Zillow’s home value index

Both Data above can be found here <https://www.zillow.com/research/data/>

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The raw data is simple, I use pandas to read the CSV files and insert them into a list. I also use the dates as the indexes because they are all unique.

# create a list containing federal documents

federal\_files = ["MORTGAGE30US.csv", "CPIAUCSL.csv", "RRVRUSQ156N.csv"]

# Use pandas read\_csv to iterate through each file.

# Read each file and parse the date and to use the date as the index.

datafs = [pd.read\_csv(file, parse\_dates=True, index\_col=0) for file in federal\_files]

## 

#Weekly Dataset from 1971 – 2024 (Measure mortgage interest rates | weekly)

datafs[0]

#Monthly Dataset from 1947 – 2024 (measures inflation | monthly)

datafs[1]

#Quarterly Dataset from 1956 – 2024 (measure the percentage of available rental property | quarterly)

datafs[2]

## A screenshot of a phone Description automatically generated A screenshot of a data table Description automatically generated A screenshot of a cell phone Description automatically generated

## I then combine all the data by concatenating all the data frames together on the index which are the dates.

federal\_data = pd.concat(datafs, asis=1)

federal\_data

## A screenshot of a graph Description automatically generated

To ensure that we retain all necessary information for my predictive model, we employ a method called forward fill due to the initial sporadic release of government data. This function fills in missing data by copying the last known value forward to subsequent dates. For instance, if there are gaps in the data series such as (1, 2, Null, Null, 4, 5, Null), the forward fill function would transform it into (1, 2, 2, 2, 4, 5, 5), ensuring continuity in the dataset.

However, if certain values remain null even after forward filling, they are deemed insufficiently reliable for analysis and are subsequently removed. This process is essential for maintaining data integrity, particularly when dealing with singular or early dates that lack correlation with the rest of the dataset.

Forward fill:

federal\_data = federal\_data.ffill()

federal\_data.head(10)

## A table with numbers and numbers Description automatically generated

## Drop NaN Value:

federal\_data = federal\_data.dropna()

federal\_data

## A screenshot of a white paper with numbers and numbers Description automatically generated

## After acquiring the raw data from the Federal Reserve, we proceeded to merge and clean it, eliminating unnecessary dates lacking relevant values. Subsequently, we imported the Zillow data, parsing it into data frames for further analysis. The data frames are then placed in a list and are displayed below.

#Metro\_median\_sale\_price\_uc\_sfrcondo\_week.csv (Zillow – median sale price for US houses)

#Metro\_zhvi\_uc\_sfrcondo\_tier\_0.33\_0.67\_month.csv (Zillow – home value)

zillow\_files = [“Metro\_median\_sale\_price\_uc\_sfrcondo\_week.csv”, “Metro\_zhvi\_uc\_sfrcondo\_tier\_0.33\_0.67\_month.csv”]

datafs = [pd.read\_csv(file) for file in zillow\_files]

#Median sale price of houses

datafs[0]

## A screenshot of a computer Description automatically generated

#zillows average estimated home value

datafs[0]

## A screenshot of a computer Description automatically generated

## To ensure consistency in the data from the Federal Reserve and Zillow, adjustments need to be made due to their differing layouts. Firstly, the Zillow data must be refined to focus solely on the information pertaining to specific dates. Additionally, a geographical region must be selected, whether it's a particular state or the entirety of the US.

## 

## The initial step involves selecting a region, with New York serving as the index at 1. Using an iloc function, the Zillow data can be segmented to isolate data related to New York. Subsequently, the first 5 columns are discarded to streamline the dataset. This process is efficiently executed by setting the start parameter to 5 [start: stop: step] when iterating through the data frame.

#Pick the row at index 1, ehich is NY and drop the first 5 columns

datafs = [pd.DataFrame(df.iloc[1, 5:]) for df in datafs]

## By adjusting the index from 1 to 0, you gain the ability to expand the data scope from solely New York to encompass the entire country. Furthermore, this adjustment facilitates the selection of alternative cities and states based on the index. To identify specific regions, you can easily filter the data frame and verify the Region Name column.

## 

## A screenshot of a computer Description automatically generated

## Once the Zillow data is chosen and cleaned the data frames will look more like the federal reserve data.

## A screenshot of a computer Description automatically generated A screenshot of a cell phone Description automatically generated

## Although the Zillow data frames may look the federal reserve. The index is in string format, and we need it in a datetime format.

## A screenshot of a phone Description automatically generated

## A white rectangular object with a black border Description automatically generated

## To accomplish this, we will iterate through the data frame and convert the string values into a datetime format. Additionally, I will incorporate a month column to facilitate the merging of both data frames later. This step is crucial due to the differing increments in the Zillow value and price data frames. While the Zillow price data is updated weekly, the value data is updated monthly.

for df in datafs:

df.index = pd.to\_datetime(df.index) #convert index (string) to datetime

df["month"] = df.index.to\_period("M")

## A screenshot of a cell phone Description automatically generated A screenshot of a computer Description automatically generated

## Since both price and value are updated differently, by adding the column of month to both data frames. This is an alternative to filling in data that was missing for both price (1\_x) and value (1\_y).

#Merge data on added month column

price\_data = datafs[0].merge(datafs[1], on=”month”)

price\_data.index = datafs[0].index

price\_data

## A screenshot of a computer Description automatically generated

## Then all I will have to do is remove the month column and rename the 1\_x and 1\_y columns to price and value and this will be the complete Zillow data frame.

## A screenshot of a computer Description automatically generated

## To combine the federal reserve data and Zillow data. I would have to add two days to the federal reserve because the federal reserve data comes out two days before the Zillow data.

## A screenshot of a computer Description automatically generated

## Since the data is now aligned, I can now merge both the Zillow and federal reserve data.

## A screenshot of a computer Description automatically generated

## Then I just rename the columns because that will make it easier for me to work with.

## A screenshot of a computer screen Description automatically generated

## Data Life cycle:

1. Initially, raw data is obtained in CSV format.
2. The raw data is converted into structured data frames for further processing.
3. Data cleaning procedures are implemented to remove unnecessary values and ensure data integrity.
4. The datasets from Zillow and the Federal Reserve are merged to create a comprehensive dataset.
5. Additional columns, such as next quarter values, are incorporated into the data frames to enhance predictive capabilities.
6. The dataset is partitioned into training and testing sets to evaluate model performance.
7. The prepared data is fed into a machine learning model for analysis.
8. Utilizing a Random Forest Classifier, the model predicts target data based on the provided features.
9. Finally, visualizations of the predicted data are generated to provide insights and aid decision-making processes.

## Machine Learning Summary

## For our machine learning approach, we implemented a supervised learning model, specifically the Random Forest Classifier. This decision stemmed from the model's versatility and efficient handling of various tasks, including feature selection, regression, and classification. The Random Forest algorithm constructs an ensemble of decision trees during training, leveraging their collective predictions to enhance accuracy and robustness. Despite its computational complexity and potential for overfitting, Random Forest offers distinct advantages, such as high accuracy, robustness to noise, and the ability to handle both numerical and categorical data without preprocessing. This choice aligns perfectly with our objective of developing a solution that delivers accurate predictions and adapts seamlessly to diverse datasets and problem areas, ensuring informed decision-making processes.

## Lightbox

Prompt 1: Random Forest Algorithm

## Before initiating the machine learning process, I made a strategic decision to refine the dataset to enhance the accuracy of our predictions. Given the complexity of forecasting housing prices alongside inflation, I opted for a pragmatic approach. Thus, I chose to streamline the focus of our prediction model by excluding the inflation factor from the housing price data. Instead, I concentrated solely on predicting the potential changes in the next quarter's housing values. This targeted adjustment allowed us to maintain realism while optimizing the predictive capabilities of our model, aligning more closely with the project's objectives and the scope of my expertise.

price\_data[“Adj\_price”] = price\_data[“Price”] / price\_data[“CPI”] \* 100

price\_data[“Adj\_value”] = price\_data[“Value”] / price\_data[“CPI”] \* 100

# Target - next quater price change

# Use pandas Shift method to pull data 13 weeks ahead and assign to next\_quater

price\_data[“Next\_quater”] = price\_data[“Adj\_price”].shift(-13)

price\_data

First, the code above computes the adjusted price of homes sold, aiming to facilitate the prediction process by removing the inflation factor. This adjustment is made by dividing the original price data by the Consumer Price Index (CPI), a measure of inflation, and then multiplying it by 100 to convert the CPI from a percentage to a usable factor. Similarly, the Zillow-computed average value of all houses in the US is adjusted using the same method and stored in a new column named "Adj\_Value."

Furthermore, the code prepares the target variable for prediction, which is the next quarter's price change. This is accomplished by using the Pandas shift method to shift the adjusted price data 13 weeks ahead, effectively aligning it with the target prediction timeframe. The shifted data is then assigned to a new column named "Next\_quarter," allowing the model to learn and predict future price changes based on historical data.

A screenshot of a table

Description automatically generated

I then remove all the None values for next quarter because this will mess up the model when training it.

A screenshot of a computer

Description automatically generated

I then add one more column to track if there was a change from the house price to next quarter price.

#True (1) or False (0)

#Check if next quarter price changed increase or not. (1 – increase | 0 – no change)

price\_data[“Change”] = (price\_data[“Next\_quater”] > price\_data[“Adj\_price”]).astype(int)

## A screenshot of a computer screen Description automatically generated

## Now that the "next\_quarter" and "Change" columns have been incorporated into the dataset, I am equipped to proceed with selecting the specific features for model training and identifying the target variable for prediction. This pivotal step allows me to strategically tailor the input variables utilized by the model to generate accurate predictions while defining the precise outcome that the model will forecast.

# list of predictors to make prediction (change)

predictors = ["Interest Rate", "Rental Vacancy", "Adj\_price", "Adj\_Value"]

target = "Change"

## Using the predictors mentioned above in conjunction with the target variable, I utilized a library imported from scikit-learn to facilitate the selection of an appropriate machine learning algorithm. In this instance, I opted for the Random Forest Classifier. Furthermore, I incorporated functionality from scikit-learn metrics to evaluate the accuracy score of the model, ensuring a thorough assessment of its predictive performance.

## In this segment of code below, the goal is to generate predictions using the Random Forest Classifier. The function, named "prediction," is designed to take in training and testing datasets, along with predictor variables and the target variable. The Random Forest Classifier is initialized with specific parameters, such as setting the minimum samples split to ten, which serves to prevent overfitting by limiting the depth of the decision tree nodes. Overfitting occurs when the model captures noise in the training data, leading to poor generalization on new data. By controlling the minimum sample split, the algorithm ensures a more balanced and generalized model. After initializing the classifier, it is trained using the training data, where the predictors are used to predict the target variable. Finally, the model generates predictions for the test dataset, which are returned as the output of the function. This process enables the algorithm to learn patterns from the training data and apply them to make predictions on unseen test data, thereby evaluating its predictive performance.

#function to make predictions – takes in training and testing set, predictor, and target

def prediction (train, test, predictors, target):

#initialize model

#min\_sample\_split – protects against overfitting by preventing the nods from

#splitting too deeply in the random forest tree

rf = RandomForestClassifier(min\_samples\_split=10, random\_state=1)

rf.fit(train[predictors], train[target]) #fit model by passing in data

preds=rf.predict(test[predictors]) #generate predictions using test set

return preds

## Validation:

## Given the historical nature of my dataset, employing cross-validation would not have been suitable due to its reliance on future data to predict the past. Considering this, I opted for a different methodology: backtesting. Backtesting involves validating a model's robustness by iteratively training it on historical training data, ranging from older to recent values. This iterative process allows us to assess the model's performance across various time spans, with each iteration utilizing a subset of the training data as test data. By leveraging backtesting, we can iteratively refine the model's accuracy and assess its predictive capabilities with respect to different time periods, thereby ensuring its reliability and effectiveness in real-world scenarios.

## The code snippet below is how I executed backtesting. In this approach, the dataset is iteratively split into training and testing sets, with each iteration predicting the next year. The process begins with a training period spanning five years, starting from 2008 to 2013, to predict the outcome for the following year, 2014. Subsequently, the training period is extended by one year, encompassing data from 2008 to 2014, to forecast the outcome for 2015. This iterative process continues until the entire dataset has been processed, allowing for a comprehensive evaluation of the model's predictive capabilities across different time spans.

Start = 260 #5 years (52 weeks \* 5)

Step = 52 #1 Year (52 weeks)

def backtest(data, predictors, target):

all\_preds = [] #initialize list (for all predictions)

for i in range(Start, data.shape[0], Step):

train = price\_data.iloc[:i] #train set – all data up until i

test = price\_data.iloc[i: (i+Step)] #test set – year following i

all\_preds.append(prediction(train, test, predictors, target)) #adds prediction to list

preds = np.concatenate(all\_preds) #concatenate all predictions

return preds, accuracy\_score(data.iloc[Start:][target], preds) #return predictions and accuracy

## Once the validation and predictions are made an accuracy score for our model can be determined.

## A screenshot of a computer Description automatically generated

## Although the accuracy is decent, I believe we can improve the accuracy score by adding more variables to the model. For example, we can add the recent trend of the home prices. This can be achieved by taking a ratio of current sales price, value, interest rate, rental vacancy and the average over the past year.

## A screenshot of a computer screen Description automatically generated

## The bottom shows when the average was added.

## A screenshot of a computer Description automatically generated

## After integrating the additional columns representing yearly averages into the predictive model, we observed an increase in accuracy. Notably, while the accuracy for New York remained consistent, there was a marked improvement in accuracy for other states or the entire country.

## Examples of accuracy improvements after more data is added to predictive model.

|  |  |  |
| --- | --- | --- |
| Country / State | Before Accuracy | After Accuracy |
| USA | 53% | 62% |
| New York, NY | 66% | 66% |
| Los Angeles, CA | 52% | 56% |
| Chicago, IL | 55% | 63% |
| Dallas, TX | 59% | 59% |

## Visualization:

## The visualization I used were line graph showing the home prices before and after I removed inflation. The adjusted price was only to predict the home prices.

## Home prices with inflations:

## A graph showing the growth of a stock market Description automatically generated

## Home prices after inflation was removed (“Adj\_price”)

## A graph showing a line of growth Description automatically generated with medium confidence

## I also used a scatter plot to show if the model predicted the changes in the real estate market correctly.

## Green = correct

## Red = incorrect

## A graph showing the growth of a stock market Description automatically generated

## The last visualization is a bar graph that shows the percentage of importance the model placed on the predictor to make its predictions. A graph of blue rectangular bars Description automatically generated with medium confidence

## Application Files

## Project files:

## Jupyter notebook files

## Housing\_Price\_Predictor.ipynb

## Federal Reserve CSV files

## RRVRUSQ156N.csv

## MORTGAGE30US.csv

## CPIAUCSL.csv

## Zillow CSV files

## Metro\_median\_sale\_price\_uc\_sfrcondo\_week.csv

## Metro\_zhvi\_uc\_sfrcondo\_tier\_0.33\_0.67\_month.csv

## User Guide

## Option 1: Google colab (View Only option)

## Open google chrome.

## Copy this hyper link: <https://colab.research.google.com/drive/1ovPHRRY90GdzvMtdoKQjk_kGCDvoUMKT?usp=sharing>

## Paste link into search bar.

## Review analysis.

## Option 2: Pycharm (View/Edit Access)

## Download Python 3 <https://www.python.org/downloads/release/python-3123/>

## Download Pycharm <https://www.jetbrains.com/pycharm/>

## Open C964\_Capstone file with PyCharm

## A screenshot of a computer Description automatically generated

## Import Libraries: Pandas, Numpy, Matplotlib, Scikitlearn

## On navigation bar at the top click run all

## If you wish to check different location change number from 0 - 236

## 

# Reference Page (Part D)

1. pandas.DataFrame.iloc — pandas 2.2.2 documentation. Pydata.org. Published 2024. Accessed May 8, 2024. <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.iloc.html>
2. pandas.DataFrame.merge — pandas 2.2.2 documentation. Pydata.org. Published 2024. Accessed May 9, 2024. <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.merge.html>
3. GeeksforGeeks. What are the Advantages and Disadvantages of Random Forest? GeeksforGeeks. Published February 15, 2024. Accessed May 9, 2024. <https://www.geeksforgeeks.org/what-are-the-advantages-and-disadvantages-of-random-forest/>
4. Backtesting: Definition, How It Works, and Downsides. Investopedia. Published 2024. Accessed May 10, 2024. <https://www.investopedia.com/terms/b/backtesting.asp>
5. Backtesting | H2O Model Validation. Docs.h2o.ai. Published 2024. Accessed May 10, 2024. <https://docs.h2o.ai/wave-apps/h2o-model-validation/v0.15.0/guide/tests/supported-validation-tests/backtesting/backtesting>
6. GeeksforGeeks. ML Underfitting and Overfitting. GeeksforGeeks. Published November 23, 2017. Accessed May 10, 2024. <https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/>
7. pandas.DataFrame.rolling — pandas 2.2.2 documentation. Pydata.org. Published 2020. Accessed May 10, 2024. <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rolling.html>