Georgia State University

Department of Computer Science

Value Visionaries

Linn Kloefta
Areeb Eshan
Bhavya Patel
Nihar Naik

April 25, 2025

Abstract

Predicting how home values change each year is important for investors, planners, and anyone involved in housing decisions. This project focuses on forecasting year-over-year (YoY) home price growth at the ZIP-code level in Georgia. We used Zillow Home Value Index (ZHVI) data along with socioeconomic and demographic information from the American Community Survey (ACS) to build a dataset that captures both housing trends and local conditions.

Each row in our dataset represents a ZIP code in a specific year, with features like volatility, average monthly growth, and previous YoY changes from the ZHVI time series. We also added county-level ACS variables like education levels, poverty rates, unemployment, and household structure to capture broader context.

To find the most useful features, we used two different selection methods: mutual information and f-regression. Then we trained three models — Decision Tree, K-Nearest Neighbors (KNN), and Random Forest — tuning them using GridSearchCV. The models were evaluated using R^2, RMSE, MAE, and SMAPE on a ZIP-based test set. Random Forest performed the best overall, while the other models showed signs of overfitting unless properly tuned.

Overall, we found that recent YoY trends and economic indicators were the most reliable signals for short-term price growth. This setup could be extended in the future to make longer-term predictions as more recent ACS data becomes available.

Keywords: housing prediction, machine learning, ZHVI, ZIP code, ACS

Report's total word count: x words

Contents

List of Figures

List of Tables

List of Abbreviations

ACS	American Community Survey
CAGR	Compound Annual Growth Rate
KNN	K-Nearest Neighbors
MAE	Mean Absolute Error
MI	Mutual Information
RF	Random Forest
RMSE	Root Mean Squared Error
SMAPE	Symmetric Mean Absolute Percentage Error
ZIP	Zone Improvement Plan (ZIP code)
ZHVI	Zillow Home Value Index
YoY	Year-over-Year

Chapter 1

Introduction

1.1 Background

Understanding housing price trends is important for economic planning, smart investing, and making housing more accessible. In the U.S., housing markets can vary a lot — not just between states or cities, but even from one ZIP code to another. Predicting growth at a local level helps everyone from city planners to individual buyers make better decisions.

Zillow's Home Value Index (ZHVI) gives monthly estimates of home values across the country and is a great starting point for analysis. But price data alone doesn't tell the full story. Things like income, education levels, household makeup, and transportation access all influence how housing prices change over time. In this project, we combined ZHVI time series data with demographic information from the American Community Survey (ACS) to build a model that can forecast local home price growth across Georgia ZIP codes.

1.2 Problem statement

The main goal of this project was to predict next-year home value growth at the ZIP-code level in Georgia. A lot of existing models work at a broader level — like metro areas or states — and often leave out important social and economic context. We wanted to build something more detailed, that looks at local trends and uses both housing and demographic data.

This isn't an easy problem. Home prices are sensitive to the economy, and there's a lot of noise at the local level. Some ZIP codes don't have much data, and there are also issues with reporting timelines and inconsistent trends from year to year. Still, our hope was that with a strong enough dataset and good feature engineering, we could make reliable short-term predictions.

1.3 Aims and objectives

Aim: Build a model that predicts one-year-ahead home value growth for ZIP codes in Georgia, using both housing trends and demographic data.

Objectives:

- Load and clean Zillow ZHVI data for Georgia ZIP codes.
- Create features from the ZHVI time series (e.g., volatility, growth rates, prior YoY).
- Add in ACS demographic data by county and year.

- Compare different models: Decision Tree, KNN, and Random Forest.
- Use feature selection (mutual information and f-regression) to reduce dimensionality.
- Tune and evaluate each model using a ZIP-based test split.

1.4 Solution approach

We built a long-format dataset where each row represents a single ZIP code in a given year. The target was defined as the YoY growth from that year to the next — letting the model learn how past trends relate to future outcomes.

From the ZHVI data, we engineered features like:

- Final ZHVI value for the year,
- Average monthly price growth,
- Volatility (standard deviation of monthly percent change),
- CAGR (Compound Annual Growth Rate),
- Number of years with negative YoY growth,
- YoY growth in the previous year.

We merged in county-level ACS data by year, including indicators like:

- Education levels,
- Poverty rate,
- Household structure,
- Vehicle ownership,
- Unemployment,
- Age breakdowns.

After preprocessing, we trained three models: Decision Tree, KNN, and Random Forest. We used ZIP-based train/test splits to prevent target leakage and tuned hyperparameters using GridSearchCV. Feature selection helped reduce noise and focus on the most informative variables.

1.5 Summary of contributions and achievements

What we were able to do in this project:

- Built a cleaned, long-format dataset combining housing and ACS data at the ZIP-year level.
- Created interpretable housing features from raw price data.
- Evaluated and tuned three models with proper train/test validation.
- Identified top predictors like recent YoY growth, volatility, and education levels.

- Made a visual tool to show predictions over time for individual ZIPs.

Our tuned Random Forest model gave the best results on the test set, while KNN and Decision Tree performed well but showed signs of overfitting without tuning. All models were able to pick up on general trends, and model performance mainly came down to how well the feature selection matched the model's complexity.

1.6 Organization of the report

This report is broken up into six chapters. Chapter 2 explains how the data was collected, cleaned, and turned into features. Chapter 3 goes over the modeling process and includes results, tuning outcomes, and visualizations. Chapter **??** discusses what the results mean and what worked well (or didn't). Chapter **??** wraps things up and suggests future directions. The appendices include any extra figures or references used in the project.

Chapter 2

Methodology

2.1 Overview

This chapter explains how the dataset was built, how features were created from the ZHVI time series and ACS data, and how the machine learning models were trained and evaluated. Everything was done in Python using libraries like `pandas`, `scikit-learn`, and `matplotlib`. The whole goal was to set things up in a way that avoided target leakage and helped each model make useful predictions based only on information available at the time.

2.2 Data collection and cleaning

The main dataset came from Zillow's Home Value Index (ZHVI), which contains monthly home value estimates for U.S. ZIP codes. We filtered it to only include rows from Georgia and cleaned the data to fix missing values in the `Metro` and `City` columns. ZIP codes with long gaps in the time series (over 12 months) were excluded.

We also pulled annual ACS 1-Year Estimates from the U.S. Census Bureau's API. These included variables related to education, poverty, transportation access, household structure, and age breakdowns, aggregated at the county level. Since ACS doesn't report at the ZIP level, each ZIP inherited the demographic data from its county and year.

2.3 Feature engineering

2.3.1 Zillow-derived features

For each ZIP and year, we generated time-based features from the historical ZHVI data leading up to that year. These included:

- **FinalZHVI**: the home value at the end of the year.
- **AvgMonthlyGrowth**: average monthly percentage growth across the year.
- **Volatility**: standard deviation of monthly growth rates.
- **CAGR**: compound annual growth rate from the earliest available data point to the cutoff.
- **NegativeGrowthYears**: total number of years that showed negative YoY growth up to that point.

- **YoY_LastYear**: percent growth from the previous year to the current one.

The target variable, `YoY_target`, was the year-over-year growth from the current year to the next — for example, a row for 2016 would have 2017's YoY as the target.

2.3.2 ACS features

ACS data was merged by `CountyName` and `Year`, and included:

- Percent of renters, households without a vehicle, and people in poverty.
- Education rates (bachelor's degree or higher).
- Unemployment rate.
- Household sizes (1-person, 4+ people).
- Percent of population in age ranges 0–17, 18–34, and 65+.

This helped give context to each ZIP's housing behavior based on local conditions.

2.4 Dataset preparation

After merging, we grouped rare counties and metro areas into an `Other` category to avoid overfitting. Categorical variables were one-hot encoded. Rows with missing target values were dropped, and the final dataset contained 3,009 rows and 139 columns.

To avoid data leakage, we did the train/test split by ZIP code. That way, if a ZIP appeared in training data, it wouldn't appear in the test set, even across years.

2.5 Feature selection

We applied two methods to rank features:

- **F-Regression:** Based on linear correlation between each feature and the target.
- **Mutual Information:** Captures any kind of dependency (not just linear).

Both methods were used to compare which features were most informative. These rankings were then used to build models with varying numbers of top features (from 1 to 50), and we compared performance using 3-fold cross-validation.

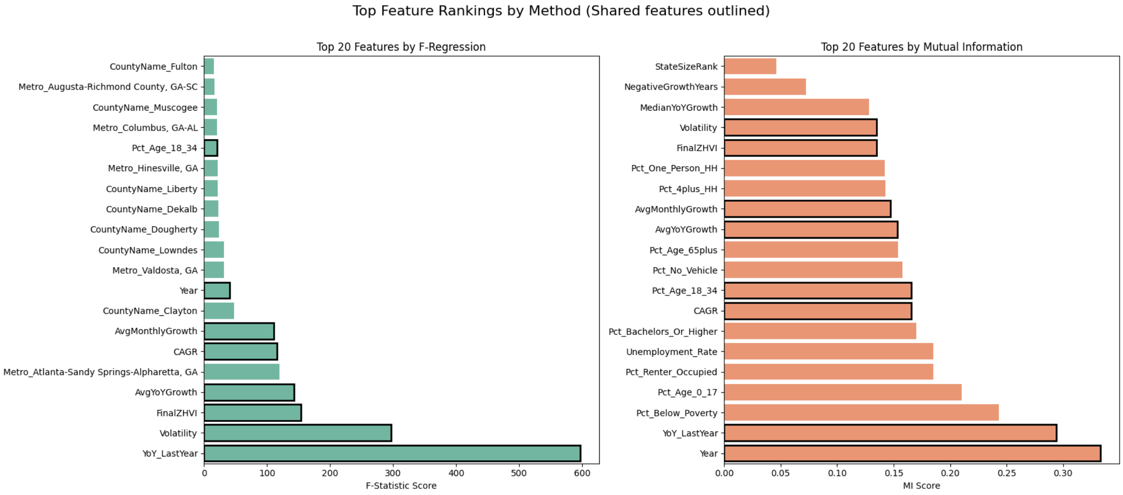

Figure 2.1: Top 20 features by F-Regression and Mutual Information.

Figure 2.1 shows the top 20 features ranked by both methods. The most influential features were consistent across methods and included `YoY_LastYear`, `Volatility`, `FinalZHVI`, and `AvgMonthlyGrowth`. F-regression emphasized time series performance metrics more strongly, while mutual information also highlighted ACS-derived features like poverty rate, vehicle access, and education level. This suggests that both price behavior and socioeconomic context contribute meaningfully to housing growth prediction.

2.6 Model training and evaluation

We tested three regression models:

- Decision Tree Regressor,
- K-Nearest Neighbors (KNN),
- Random Forest Regressor.

Each model was tuned using `GridSearchCV`. The evaluation metrics included:

- R^2 (coefficient of determination),
- RMSE (root mean squared error),
- MAE (mean absolute error),
- SMAPE (symmetric mean absolute percentage error).

Model performance was measured on a held-out test set of ZIP codes. Final evaluation included both numeric metrics and visual inspection via scatter plots and prediction-over-time charts.

2.7 Ethical considerations

This project uses only public datasets with no individual or sensitive personal data. All ACS data is aggregated at the county level, and ZHVI does not contain identifiable information. As such, no informed consent, privacy measures, or risk mitigation were needed. The model is intended for research and educational purposes and does not make real estate recommendations or financial decisions for individuals.

2.8 Summary

We started by cleaning and merging housing and demographic data. After engineering features and merging datasets, we created a ZIP–year format suitable for supervised learning. We used two feature selection methods to reduce dimensionality, and trained three different models with hyperparameter tuning and ZIP-based splits. Results were evaluated both quantitatively and visually to understand how well each model performed and where they differed.

Chapter 3

Results

This chapter presents the results of our model training and evaluation. After building and tuning Decision Tree, K-Nearest Neighbors (KNN), and Random Forest regressors using ZIP-based splits, we assessed how well each model predicted year-over-year home value growth. Results are shown across three key areas: model performance with different feature counts, test set prediction accuracy, and model behavior over time for a sample ZIP code.

3.1 Feature selection and model comparison

We first examined how the number of selected features affected each model's performance. For every value of k from 1 to 50, we used both f-regression and mutual information to rank features. The top-k features were then used to compute 3-fold cross-validated R^2 scores for each model.

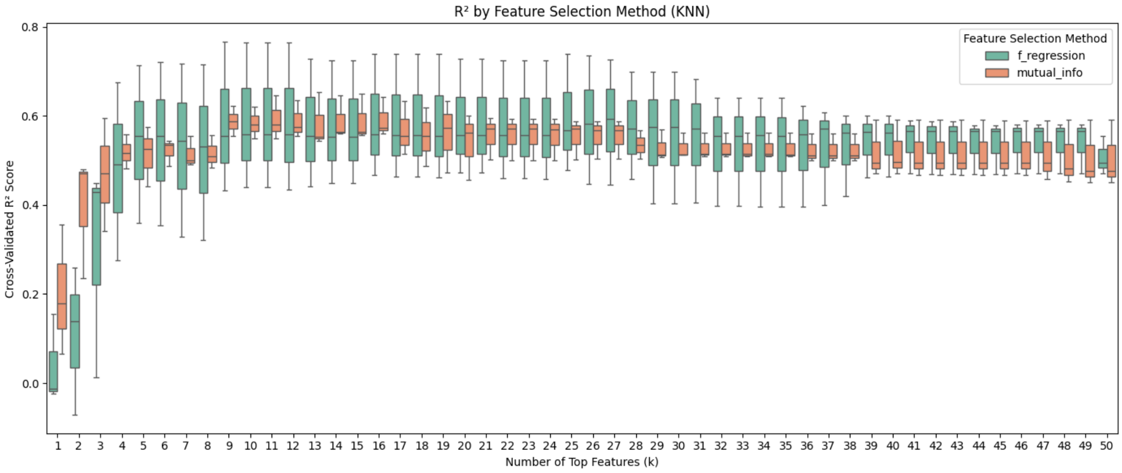

Figure 3.1: Cross-validated R^2 scores for KNN model.

In Figure 3.1, the KNN model shows a clear performance increase up to about 8 or 9 features, after which the R^2 scores plateau. f-regression consistently led to slightly higher performance at low values of k, with mutual information catching up as more features were added. Beyond 12–15 features, there was no significant improvement, and some increase in variance was observed, especially with mutual information.

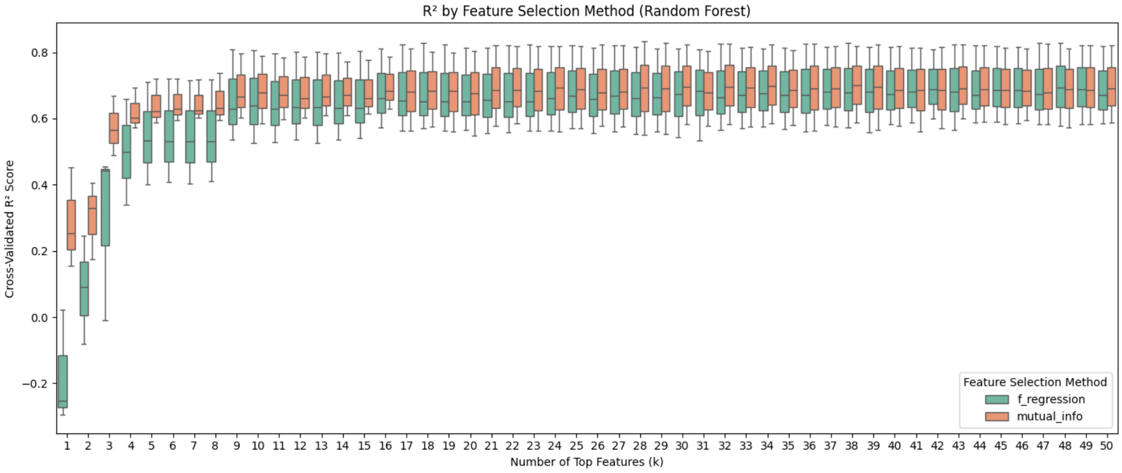

Figure 3.2: Cross-validated R^2 scores for Random Forest model.

Random Forest results in Figure 3.2 show a steep improvement in R^2 from 1 to around 15 features, peaking between 15–20 features. After that, performance remains mostly stable. The model is more robust to the number of features used, with mutual information providing slightly more consistent results at higher k. The box plots are compact throughout, indicating stable behavior across folds.

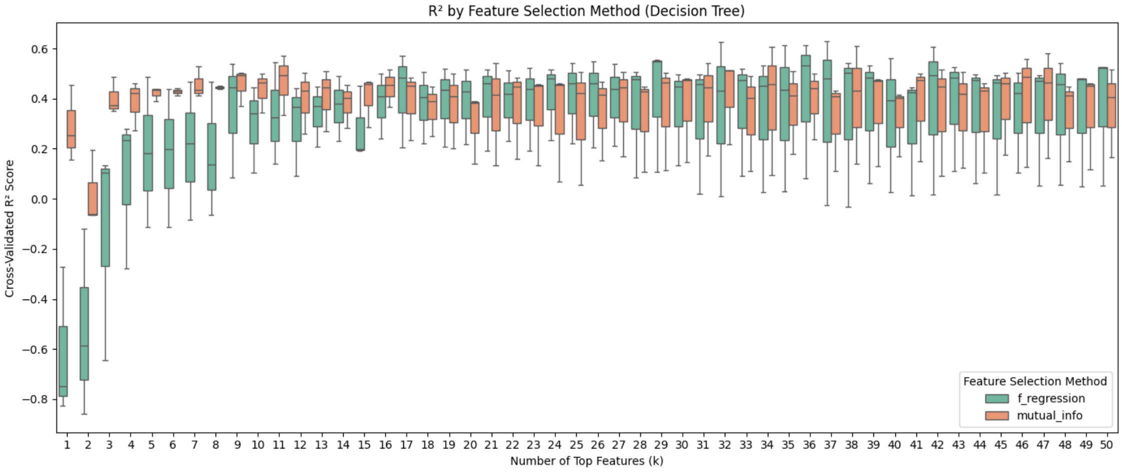

Figure 3.3: Cross-validated R^2 scores for Decision Tree model.

Figure 3.3 shows that the Decision Tree model is the most sensitive to the number of features. Performance improves up to around 8–10 features, but quickly deteriorates beyond that, especially with mutual information. This is likely due to overfitting. The most compact and stable results occurred when using 8–12 features, with f-regression performing slightly better overall.

Based on these observations, we selected the following number of features for each model:

- **Decision Tree:** 8 features
- **KNN:** 9 features
- **Random Forest:** 16 features

3.2 Test set prediction accuracy

We evaluated each model on a held-out test set of ZIP codes that were not used during training. Figures 3.4, 3.5, and 3.6 show the predicted vs actual year-over-year growth values, along with a regression line, an ideal line ($y = x$), and ±1 RMSE bands.

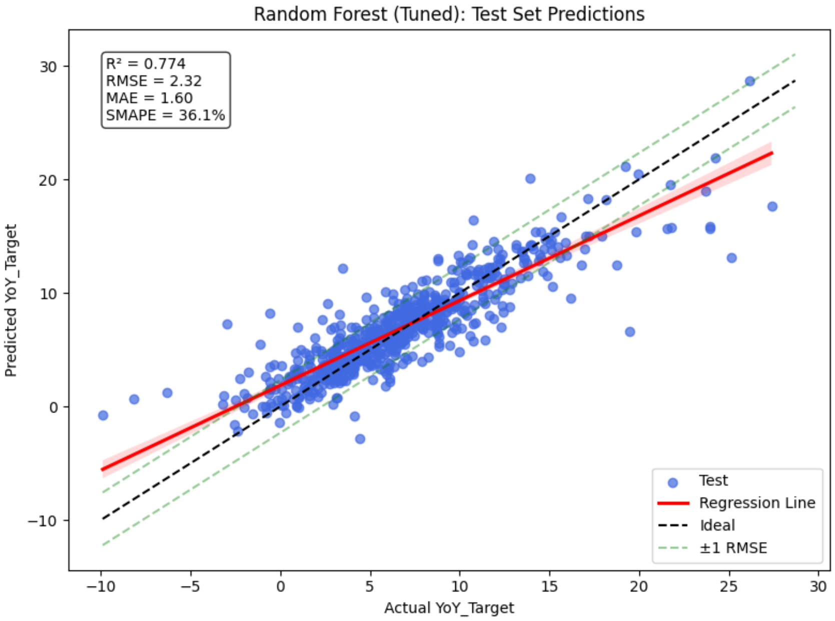

Figure 3.4: Random Forest (Tuned): Test set predictions.

In Figure 3.4, the Random Forest model shows the strongest test performance. Most predictions fall close to the ideal line, and the regression line closely matches it with a steep slope. The green dashed bands (±1 RMSE) are relatively tight, covering most of the distribution. The model achieved an R^2 score of 0.774 on the test set.

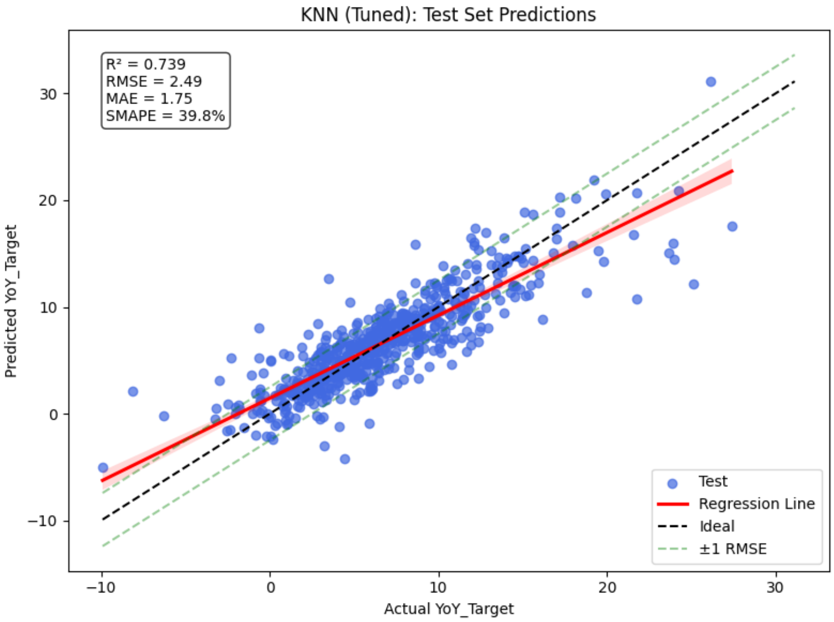

Figure 3.5: KNN (Tuned): Test set predictions.

The KNN model in Figure 3.5 also performed well, achieving a test R^2 of 0.739. Its predictions were slightly more spread out compared to Random Forest, and the regression line was less steep. A few outliers at the higher end of the target range were underpredicted, but the overall alignment to the ideal trend was solid.

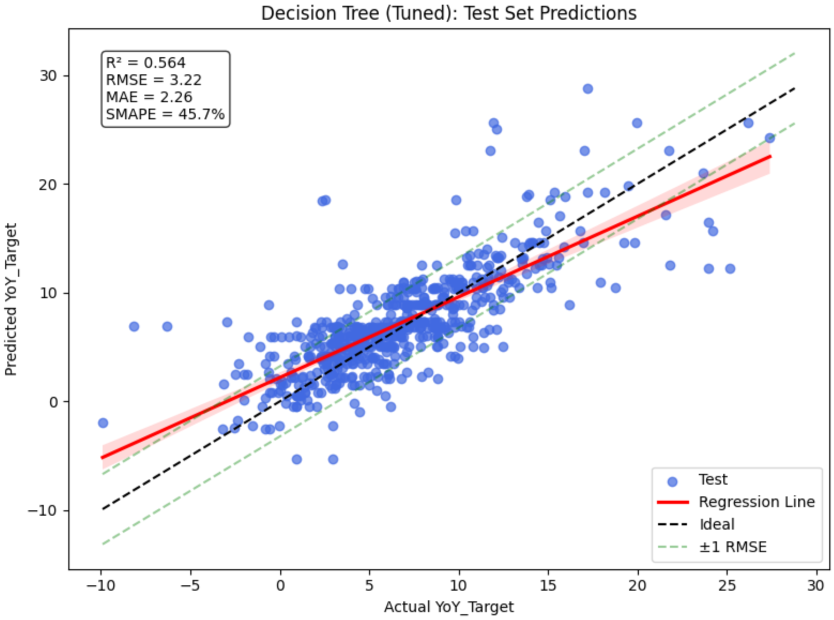

Figure 3.6: Decision Tree (Tuned): Test set predictions.

Figure 3.6 shows the weakest test performance, from the Decision Tree model. Its regression line is noticeably flatter than the ideal, and predictions show more deviation. It underpredicts many higher values and compresses the range of outputs. The test R^2 was 0.564, indicating limited generalization.

3.3 Train vs test comparison

To understand how well each model generalizes, we compared train vs test predictions side by side. Figures 3.7, 3.8, and 3.9 show both sets of predictions in one plot.

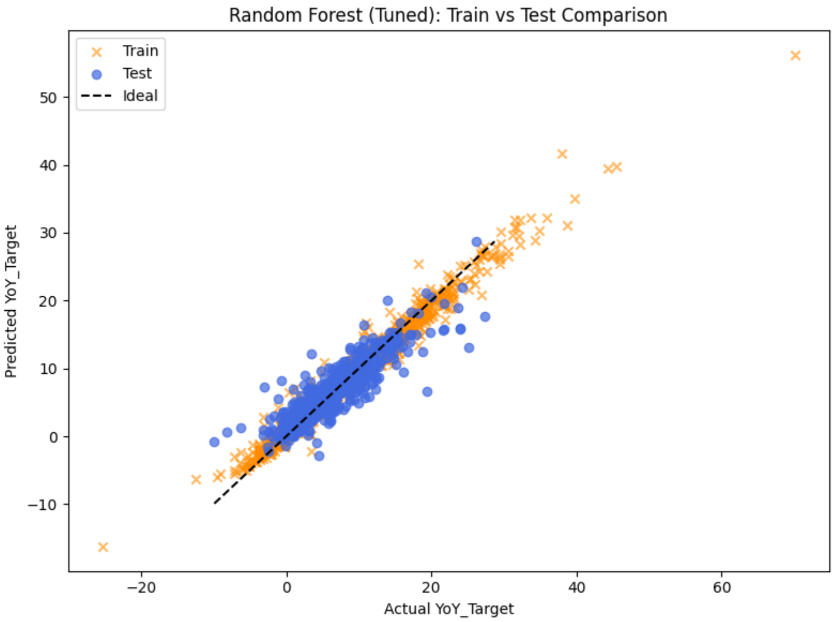

Figure 3.7: Random Forest: Train vs test predictions.

Figure 3.7 shows a well-balanced fit for Random Forest. Predictions on both train and test sets are tightly clustered around the ideal line, suggesting the model generalizes well without overfitting.

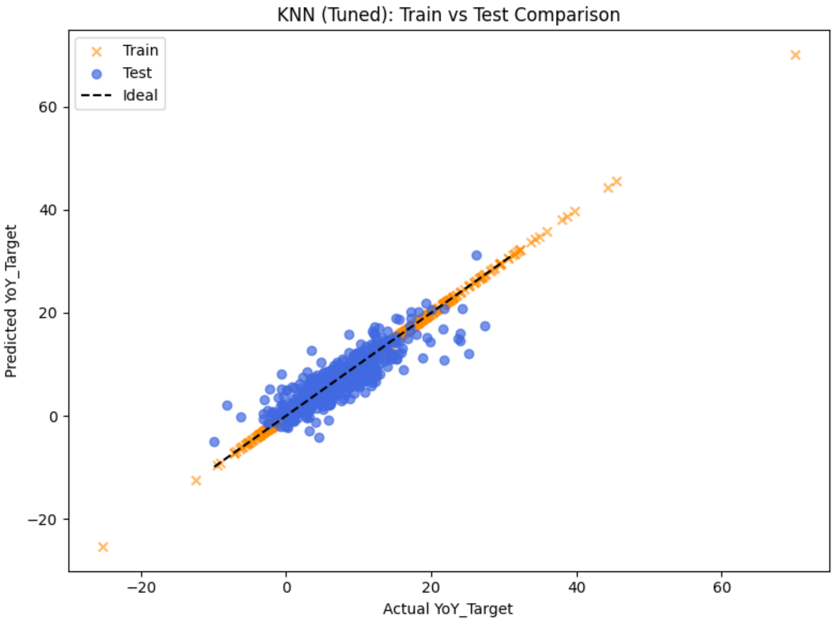

Figure 3.8: KNN: Train vs test predictions.

KNN (Figure 3.8) also performs consistently across train and test splits, though train predictions are slightly tighter. The test distribution shows slightly more spread, but no major signs of overfitting.

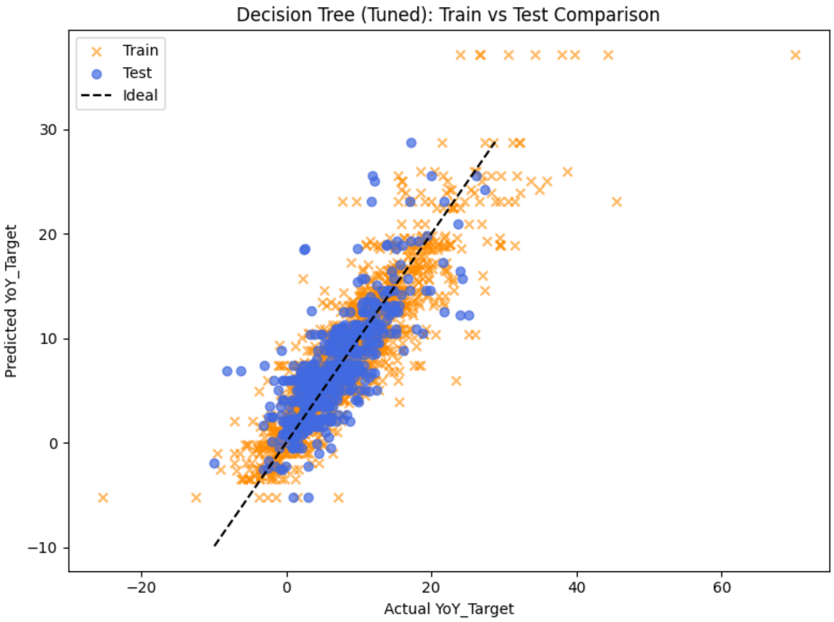

Figure 3.9: Decision Tree: Train vs test predictions.

The Decision Tree (Figure 3.9) shows a stronger fit on the train set than on the test set, with the test predictions exhibiting more variance. This difference suggests some overfitting, consistent with its lower R^2 on unseen ZIPs.

3.4 Prediction trend over time

To visualize how each model performs across years for the same location, we selected a ZIP code (31052) and plotted predictions alongside actual YoY values over time.

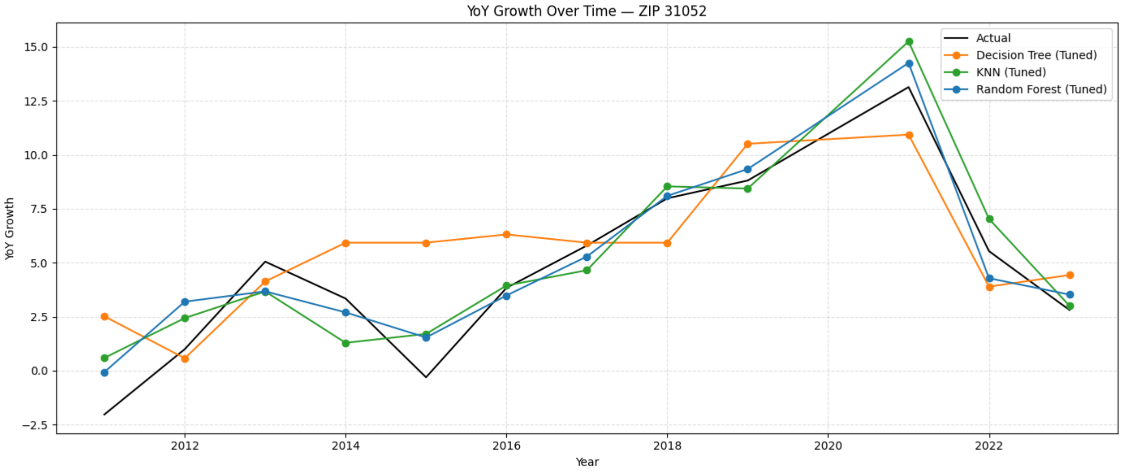

Figure 3.10: Predicted vs actual YoY growth over time for ZIP 31052.

Figure 3.10 shows that all models followed the general trend of actual YoY growth, but with different levels of accuracy. Random Forest consistently tracked peaks and dips, while Decision

Tree predictions were flatter and missed several changes. KNN captured sharp increases fairly well but overshot in some years. All three models produced usable forecasts, though Random Forest offered the most precise temporal behavior.

3.5 Summary of test performance

Table 3.1 summarizes final performance on the test set using the selected number of features.

Table 3.1: Model performance on test set (selected features)

Model	**R^2**	**RMSE**	**MAE**	**SMAPE**
Random Forest (16 features)	0.774	2.32	1.60	36.1%
KNN (9 features)	0.739	2.49	1.75	39.8%
Decision Tree (8 features)	0.564	3.22	2.26	45.7%

3.6 Summary

All three models captured short-term housing value growth with varying levels of accuracy. Random Forest performed the best across the board, showing strong generalization and tight error margins. KNN followed closely and was relatively stable, but more sensitive to feature selection. Decision Tree required careful tuning and showed signs of overfitting at higher feature counts.

Using both f-regression and mutual information helped ensure that selected features were consistently informative. The ZIP-based split offered a realistic test scenario, ensuring that models were evaluated on truly unseen locations.

Chapter 4

Discussion and Analysis

Depending on the type of project you are doing, this chapter can be merged with “Results” Chapter as “ Results and Discussion” as suggested by your supervisor.

In the case of software development and the standalone applications, describe the significance of the obtained results/performance of the system.

4.1 A section

The Discussion and Analysis chapter evaluates and analyses the results. It interprets the obtained results.

4.2 Significance of the findings

In this chapter, you should also try to discuss the significance of the results and key findings, in order to enhance the reader’s understanding of the investigated problem

4.3 Limitations

Discuss the key limitations and potential implications or improvements of the findings.

4.4 Summary

Write a summary of this chapter.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

Typically a conclusions chapter first summarizes the investigated problem and its aims and objectives. It summaries the critical/significant/major findings/results about the aims and objectives that have been obtained by applying the key methods/implementations/experiment set-ups. A conclusions chapter draws a picture/outline of your project's central and the most signification contributions and achievements.

A good conclusions summary could be approximately 300–500 words long, but this is just a recommendation.

A conclusions chapter followed by an abstract is the last things you write in your project report.

5.2 Future work

This section should refer to Chapter 3 where the author has reflected their criticality about their own solution. Concepts for future work are then sensibly proposed in this section.

Guidance on writing future work: While working on a project, you gain experience and learn the potential of your project and its future works. Discuss the future work of the project in technical terms. This has to be based on what has not been yet achieved in comparison to what you had initially planned and what you have learned from the project. Describe to a reader what future work(s) can be started from the things you have completed. This includes identifying what has not been achieved and what could be achieved.

A good future work summary could be approximately 300–500 words long, but this is just a recommendation.

Appendix A

An Appendix Chapter (Optional)

Some lengthy tables, codes, raw data, length proofs, etc. which are **very important but not essential part** of the project report goes into an Appendix. An appendix is something a reader would consult if he/she needs extra information and a more comprehensive understating of the report. Also, note that you should use one appendix for one idea.

An appendix is optional. If you feel you do not need to include an appendix in your report, avoid including it. Sometime including irrelevant and unnecessary materials in the Appendices may unreasonably increase the total number of pages in your report and distract the reader.

Appendix B

An Appendix Chapter (Optional)

…