house-pricing-prediction-proposal-57.github.io

ML Project Final

Introduction/Background

A strong housing market supports the economy and increases citizen wealth through investments [4]. Given the fluctuating nature of housing markets nationally, it is essential for buyers, sellers, and policymakers to predict market trends in a neighborhood with high confidence.

ML researchers have compared how models like decision trees, linear regression, and custom regression can accurately predict the trends of the real estate sector. Literature read primarily relies on factors such as square footage and tenant quantity with price as the expected output [1][4][5]. Our research has confirmed external research, that including neighborhood-level features significantly boosts the model's accuracy [2]. The dataset chosen has 79 features describing individual homes, from lot area to year built, with a label of sale price. We believe the features range in importance, where features like overall quality, year built, while things like alley access may be less important and less contributive to overall predictions. Additionally, aspects like central air and their impact depend highly on geographical area.

Problem/Motivation

Housing prices are influenced by numerous factors like location, house size, and national economic trends, complicating informed decision making for stakeholders, including homebuyers, realtors, and policymakers. Inaccurate predictions can result in financial losses, overpriced properties, or missed investment opportunities.

Modern techniques for housing price valuation make use of traditional statistical models which struggle to capture the dynamic and complex correlations of the multiple factors determining housing values. Using ML algorithms, we plan to improve accuracy and reliability, benefitting investors, banks, policymakers, and companies like Zillow, whose products need accurate and comprehensive price predictions; this large scope motivates us.

Methods

Data Preprocessing Methods Identified

1. **One-hot encoding** - Our dataset mainly consists of numerical data; however, some fields are categorical features such as `BldgType` which could be `1Fam`, `2fmCon`, `Duplex`, `TwnhsE`. To ensure data consistency, we can convert them to numerical format using one-hot encoding [7].
2. **Feature Engineering** - Create custom features to create more meaningful data. For example, `TotalSF = TotalBsmtSF + 1stFlrSF + 2ndFlrSF` as a measure of the total square footage [8].
3. **Feature Selection** - Apply PCA or other dimensionality reduction techniques to reduce complexity.

ML Algorithms/Models Identified

1. **Ridge Regression (`sklearn.linear_model.Ridge`)** - *Supervised*
   a. A type of linear regression that applies L2 regularization, adding a penalty to large coefficients to prevent overfitting.
   b. Ridge Regression can help model the linear relationship between features such as the number of bedrooms, lot size, and year built. It can also control overfitting by reducing the impact of irrelevant or highly-correlated features [6].
2. **Random Forest Regressor (`sklearn.ensemble.RandomForestRegressor`)** - *Supervised*
   a. An ensemble learning model that creates multiple decision trees and averages their predictions to come up with a final result.
   b. It can effectively capture non-linear relationships between features such as neighborhood, house style, and condition.
   c. The model’s ability to calculate feature importance also helps in understanding which variables have the most impact on house prices [6].
3. **Gradient Boosting Machines - XGBoost (`sklearn.ensemble.GradientBoostingClassifier`)** - *Supervised*
   a. A gradient boosting algorithm that iteratively improves the prediction by focusing on correcting the errors of prior models.

b. XGBoost’s benefit is its efficiency and ability to handle complex relationships between features like the house’s quality, age, and lot characteristics [6].

Potential Results and Discussion:

We will evaluate our model using four metrics, namely MAE, RMSE, R^2 Score, and MAPE.

- **MAE** measures the average error between the predicted and actual home prices, with lower values indicating more accurate predictions.
- **RMSE** evaluates the magnitude of the errors; the lower the value, the more accurate.
- **R^2 Score** shows how well the model fits the data, with values closer to 1 representing perfect fitting.
- **MAPE** will help understand the relative error in predictions based on the magnitude of the home price.

By applying multiple ML techniques such as regression, random forests, and gradient boosting, we aim to improve accuracy and ensure fair pricing, minimizing unfair or unethical lending practices [1][3][6].

Model Selection

We performed PCA and Feature Engineering to reduce the dimensionality of the dataset and enhance the quality of data passed into the ridge regression model. We also replaced the NaN values of categorical features with the most common value using a categorical transformer pipeline. Moreover, we replaced the NaN values of numerical values with the median value (a popular strategy in the real estate industry) using a numeric transformer pipeline. We decided to start off with Ridge Regression for a baseline initial model – purely because of its simplicity. The effects of regularization in a high-dimensional dataset such as the one we are using will be immensely helpful. We have also done some feature engineering eg. square footage data in order to perform dimensionality reduction and make the data of higher quality. Ridge regression has also proved to be robust in dealing with multicollinearity – a common phenomenon in the real estate data world.

In addition to the Ridge Regressor (linear modeling), we implemented a Random Forest Regressor to account for the non-linear relationships between the complex features in the data. As it is an ensemble learning method, it captures complex non-linear interactions better than a ridge regressor can, and also avoids multicollinearity.

Lastly, we also implemented a XGBoost Regressor to further mitigate the limitations of the Random Forest model to further improve the accuracy of capturing the non-linear relationships. Since it is a gradient boosting algorithm, it better captures these non-linear relationships, uses regularization algorithms to reduce overfitting, iteratively improving its predictions by correcting the errors of its previous models, and to account for a high-dimensional dataset like the one we work with.

Results

We plotted the results of the Ridge Regression model – it showed strong initial results in predicting housing prices. For low to mid-range houses (<$400,000), it shows a very strong prediction. However, for higher-end houses, there is a deviation from the predicted line. The same trend can be seen on the testing dataset.

Predicted vs Actual Sale Price on Training Set

Predicted vs Actual Sale Price on Test Set

Metrics

We also visualized the residual plots for both the training and testing datasets. We know that a well-trained model has residuals centered around 0 and will have a narrow residual distribution. The training residual plot shows exactly that – indicating that there is no significant bias in the data that affects it by heavily over-predicting or under-predicting the prices. This is verified by the test residual plot; although the distribution is slightly greater than the training set plot, it is centered around 0. One significant insight we have from the testing dataset is that the high-end expensive properties have large residuals. This reinforces our previous insight that the model struggles to generalize for expensive houses. The wider distribution can also indicate a potential sign of overfitting to lower-end homes, thereby throwing off the higher-end data.

Residiual Distribution on Training Set

Residiual Distribution on Test Set

Random Forest Regression Implementation

We implemented an ensemble learning model in the form of Random Forest regression, a method that creates multiple decision trees during regression and averages out each tree's predictions to produce a final result. Random forest, according to the research our team conducted, captures the non-linear relationships of housing prices more effectively. After implementing the regressor, we had some key insights regarding random forest regression. It captures feature importance extremely well and understands the impact each variable has on the model. We can observe this from the side-by-side visualization of the residual plots:

Fig: Side by Side Visualization of Residual Plots for 2 Models (Training)

Metrics

As we can observe from the table, random forest performs better than ridge regression across all metrics for both, train.csv and test.csv. This improvement, we strongly believe, lies primarily in the non-linear nature of random forest.

Fig: Side-by-side visualization of the regression results with Ridge Regression (left) and RF (right)

XGBoost Implementation Implementation

We implemented the XGBoost regression model as well, which uses supervised learning techniques in order to capture a quantifying relationship between housing prices and other factors in the dataset that might be influencing expected price. XGBoost, is a gradient boosting algorithm which works by building decision trees sequentially, where each decision tree corrects the error of the previous decision trees. This ensures that the model really stresses on the most challenging predictions, and handles complex non-linear interactions, especially when the dataset in use is high dimensional with ~79 features. XGBoost also uses a loss function, in this case since our task is regression, we used the Mean Absolute Error (MAE) loss function. Since we already preprocessed the dataset due to our work in the previous models, the XGBoost model was able to work with high-quality inputs to give us an accurate prediction model. For our implementation of XGBoost to discuss briefly, we used feature engineering and hyperparameter tuning to further optimize the model performance. This was done mainly due to the issue of overfitting, since it is common for XGBoost models to suffer from overfitting due to the natural processes of XGBoost behind the scenes. To avoid this, we did feature selection where we analyzed what the top 20 important features were by calculating their individual respective F scores, which can be seen in the graph below (figure 1).

Figure 1: Features vs F-Score Graph

We then removed the least important 20 features, which had the lowest F scores and removed them from the dataset, which allowed for dimensionality reduction to improve generalization by reducing noise in the dataset. To further improve the model, hyperparameter tuning was done using a search algorithm, RandomizedSearchCV algorithm to further fine tune parameters such as n_estimators, learning_rate, max_depth, subsample, and the two regularization factors reg_alpha and reg_lambda. The search algorithm then searches through the parameter space, and fine-tunes the mentioned parameters to achieve an optimal model performance that better takes into account the bias and variance, so the model can generalize pretty strongly to further unseen data. Further, our final results (figure 2) indicate that the XGBoost model outperforms both the previous models discussed (figure 3).

Figure 2: Final Metric Results for XGBoost Model

Figure 3: Comparison between each final metric for XGBoost, Ridge Regression, and Random Forest

Figure 3 shows us a comparison of all the key evaluation metrics (MAE, RMSE, R2R^2) for both training and testing datasets. XGBoost shows a big improvement over Ridge Regression, with a 12.3% reduction in testing MAE and a 10.8% reduction in testing RMSE. Further, when compared to Random Forest, XGBoost achieves better generalization as it accommodate for overfitting and low influencing features (which Random Forest does not), showing a 6.1% improvement in testing MAE and an 8.3% improvement in testing RMSE, while slightly trailing Random Forest on the training set as this is expected due to making a tradeoff of further fine tuning the model. Even though we can notice that the training set metrics for XGBoost were lower than the previous two models, XGBoost outperformed both models for its test performance. In particular, XGBoost got scores of 16,670.58 for MAE and a RMSE score of 26,649.33, with a R^2 score of 0.9074, which are all outstanding scores and outperforming the previous two models. The R^2 score in particular gives us proof that the model is able to explain variance optimally. In conclusion, the data points of the metrics shows us that XGBoost is able to strike a balance between training and testing performance, making it the most accurate and reliable model compared to the other models.

As we can see as well from the graph below(figure 4), the residual distribution plot for XGBoost shows us a symmetric, bell curve spread centered around zero, which gives us the proof that the model has low bias and a nice distributed error across all possible price ranges.

Figure 4: XGBoost Residual Distribution Graph

In addition, XGBoost excels in further correcting what the previous two models weren't able to take into account. In particular, XGBoost, unlike Ridge Regression, was able to take into account expensive properties, since the residual errors were much reduced compared to the Ridge Regression graph. Compared to the Random Forest model as well, XGBoost with some finetuning was able to reduce the amount of overfitting of the model, since in the metrics for XGBoost's test and train metrics there was a smaller performance gap, indicating it handles overfitting much better. For example, when we take a look at the figure below, looking at the comparisons it is quite noticeable that XGBoost shows a much tighter clustering of points along the red dotted line (perfect fit line) compared to the Ridge Regression graph, which gives us a strong signal of the model performance being of higher accuracy and lower residual errors, especially in the most populated area of price. Most notably however, is its accuracy much improved for upper-range housing prices. The data point at around $700,000 housing price for XGBoost is much closer to the best fit fine compared to the Ridge Regression model, indicating higher accuracy. This can be explained since XGBoost is able to understand non-linear relationships much better than Ridge Regression, which this comparison figure proves precisely. Further, when comparing XGBoost graphs with Random Forests, we can notice that while Random Forest does show better results than Ridge Regression (further discussed previously), XGBoost slightly outperforms Random Forest, as there a few data points in the Random Forest graph that have a much higher residual than the same data points in XGBoost, taking for example the data point at $700,000, which for Random Forest sits at the lower $600,000 point and further away from the best fit line, but for XGBoost is closer towards the $700,000 sale price and much closer to the best fit line. In addition, XGBoost makes the better trade off than Random Forest since it is less prone to noise and overfitting as we can conclude from the above analysis and implicitly does regularization to reduce overfitting as well while at the same time doing much better than Random Forest at contextualizing non-linear relationships.

Fig: Side-by-side visualization of the regression results with Ridge Regression (upper left), Random Forest (upper right), XGBoost (bottom middle)

Overall, when putting all three models to the test and taking into account the various factors mentioned in the paragraph, all things considered XGBoost is the best performing model as it gives the best accurate results for all price ranges, doesn't account into noisy data from the dataset, and accounts for overfitting beating out the other two model's limitations.

Next Steps

Our next steps aim to improve model accuracy and interpretability to better understand the data:

1. **Hyperparameter Fine-Tuning**: Even though Random Forest and XGBoost both show strong performance, we could fine-tune potentially even more by using other search algorithms and experimentation to improve both model's performance.

2. **Additional Feature Engineering**: While for XGBoost, we did rudimentary feature engineering, we can further extend this to analyze the top feature predictors to further improve the model performance.

3. **Additional Research**: While we did research on three models, we can use an advanced dataset with more data points and features to improve the model's performance. We can also potentially combine multiple models and come up with a better model that captures the relationships across different property types and neighborhoods.

References:

1. Y. Chen, J. Jiao, and A. Farahi, "Disparities in affecting factors of housing price: A machine learning approach to the effects of housing status, public transit, and density factors on single-family housing price," Community and Regional Planning, University of Texas at Austin, United States of America, Jun. 2023.
2. L. D. Long, "Machine Learning-Based Feature Mapping for Enhanced Understanding of the Housing Market," Alexandria Engineering Journal, vol. 79, pp. 480–501, 2023, doi: 10.1007/978-3-030-62008-0_35.
3. K.-C. Chiu, "A long short-term memory model for forecasting housing prices in Taiwan in the post-epidemic era through big data analytics," Department of Finance, Shih Chien University, Taiwan, and Center for Innovative FinTech Business Models, National Cheng Kung University, Taiwan, 2023.
4. J. Avanijaa, G. Sunitha, K. R. Madhavi, P. Korad, and R. H. S. Vittale, "Prediction of House Price Using XGBoost Regression Algorithm," Sree Vidyanikethan Engineering College, India, Apr. 2021.
5. G.-Z. Fan, S. E. Ong, and H. C. Koh, "Determinants of House Price: A Decision Tree Approach," Urban Studies, vol. 43, no. 12, pp. 2301-2316, Nov. 2006, doi: 10.1080/00420980600990928.
6. I. H. Sarker, "Machine Learning: Algorithms, Real-World Applications and Research Directions," SN Computer Science, vol. 2, no. 3, pp. 1–21, 2021, doi: https://doi.org/10.1007/s42979-021-00592-x.
7. J. Samuels, "One-Hot Encoding and Two-Hot Encoding: An Introduction," 2024, doi: 10.13140/RG.2.2.21459.76327.
8. J. Heaton, "An empirical analysis of feature engineering for predictive modeling," SoutheastCon 2016, Norfolk, VA, USA, 2016, pp. 1-6, doi: 10.1109/SECON.2016.7506650.

Midterm Contributions

- **Sohum Gaitonde**
  Worked on enhancing Random Forest and XGBoost implementations in code, in addition to creating and populating reports with data visualizations to supplement technical findings. Ensured group communication and progress throughout the project.
- **Krish Asknani**
  Worked on enhancing XGBoost implementation, updating various aspects of report qualitatively, in addition to brainstorming, creating, and filming for the final video presentation. Always focused on ensuring group progress throughout the project.
- **Ayush Pai**
  Worked on enhancing Random Forest and XGBoost implementations, in addition to working extensively on qualitative portions of Final Report and updating any existing portions. Focused on ensuring group progress throughout the project and activating work sessions.
- **Farzad Ashfak**
  Worked extensively on the Random Forest technical implementation and made continuous updates to it. Worked on updating the qualitative aspects of the report accordingly and ensured timely completion and progress. Ensured group progress throughout the project and adequate timeline and completion of tasks.
- **Sidharth Subbarao**
  Worked on the XGBoost implementation and came up with the final metrics analysis. Also worked on potential next steps on the qualitative report and Random Forest implementation and fine tuning both models.

Gantt Chart

See above link

Also image here: https://drive.google.com/file/d/1LZ98S0DJ1cgVPWaltWDvGGiwCvUcl112/view?usp=sharing