House Prices: Advanced Regression Techniques

Akash Jagannathan
013856531

Sreeja Madanambeti
014619930

Sudha Vijayakumar
014749488

Yashwant Khade
014531751

***Abstract*—Typically, we think interested home buyers only look at certain features of a home when buying, such as the size or number of bedrooms. However, typically much more influences the buying decision. The dataset has attributes for almost every conceivable aspect of a home to predict house prices. We have used various regression fitting techniques to be able to accurately estimate the price of a home using different dimensions of a home that are looked over. We have employed different models to solve this problem and present a comparison between the different techniques in terms of accuracy, efficiency, and usability. In our project we found that weighted ensembling of the best performing regressors XGB, StackedCV, and Voting Reg gave the best generalization. We submitted the result by predicting values for the test dataset and our submission was approximately in the top 50 percentile of the public submissions.**

***Index Terms*—advanced regression techniques, ensembling, house prices, machine learning, neural networks, prediction, random forest, xgboost**

I. Introduction

In recent times we have seen applications that can predict the values of assets or things. In this project, we are predicting house prices based on various features. Traditionally we estimate house prices based on features like the number of bedrooms, area, floors, but it is not limited to this as various amenities like swimming pools, back yards, etc also impact the estimated value of house prices significantly. We cannot formulate an equation directly with these features. Therefore, this is a good problem to solve using machine learning techniques. Kaggle released a dataset for predicting house prices named as Ames Housing dataset which can be used in data science for educational purposes. The dataset has 79 features and is collected from the residential homes in Ames, Iowa. The Kaggle competition challenges us to predict the final price of each home by using advanced regression techniques. The advanced regression techniques like Random Forest and XGBoost with other models and Neural Networks will be used in this project.

The goal of this project is to apply advanced regression techniques to increase the out of sample accuracy in predicting house prices. The performance of the models will be evaluated based on various error metrics. The best performing model will be chosen and submitted to the Kaggle competition. The Kaggle competition has defined Root Mean Squared Logarithmic Error (RMSLE) as its error metric. In this project, we have used the same error metric for evaluating our models. Finally, we will submit our predictions of the test dataset to the Kaggle competition to analyze the performance of our techniques. The flow of the research paper is as follows:

- Background - This section will discuss the literature survey and related studies for this topic.
- Technical details - This section describes the models we have used in our project. It also shows us the tuning parameters and the performance of the chosen models.
- Experimental study - This section will discuss the steps in the implementation in detail. It will describe the data analysis, preprocessing steps and the advanced regression techniques used in this project.
- Result analysis - This section will analyze the results from our implementation of the regression techniques. We will compare these results based on the error metrics used and performance of the models.
- Conclusion - In this section, we will describe our hypothesis on the results obtained from our implementation. We will discuss the future works to improve the performance of our work.

II. Background

Previous generations did not have tools we now have available to predict house prices. Economists had been able to derive theoretical formulas for house price prediction such as the hedonic model [1] and repeat sales indices [2], however they were unable to verify their theories due to a lack of data.

Prediction tasks continued to be a challenge even after the turn of the 21st century. Previously, when researchers took data-driven approaches to predicting house prices, they were only able to build simplistic models showing relatively rudimentary forms of prediction. For example, the main results presented in one such study [3] were correlative relationships between one variable and the house price; they were not able to combine multiple variables together in a meaningful way or take a majority of potentially relevant factors into account.

The current era has seen data accumulation in massive proportions and at rapid speeds, which has significantly aided the efforts of economists and researchers in their house price prediction efforts. Studies conducted in [2] found that with the increased availability of housing data, they were able to develop models that had high prediction accuracy and zero bias by simply accounting for the geographical location of the property. As a result of the massive increase in data collection, computational techniques were developed specifically to harvest this novel information source. Data mining and machine learning techniques were derived to build predictive and descriptive models based on real data. These techniques provided a number of options for finding underlying patterns in data with many factors contributing to the final prediction.

Most house price prediction studies with machine learning models have utilized regression techniques, which we have employed in this study. A very similar study to the present one was conducted in [4], where a researcher tested an artificial neural network on a New Zealand housing dataset containing multiple attributes and found promising results with his model.

As we were planning to implement various regression techniques on this housing dataset, we felt it was also pertinent to study what types of regression work in which situations. Our team was only familiar with more standard regression techniques such as linear and logistic regression. We found other techniques such as Ridge regression [5] and Lasso regression [6] which seemed to perform well with high-dimensional data. When we actually started working on the problem, we found more techniques to use via the Scikit-Learn library, classroom studies, and online tutorials.

III. Technical details

We have implemented a variety of models for our regression prediction in order to select the best ones.

A. Regularization Techniques

1) Ridge and LASSO regression: To reduce over-fitting, these regularization techniques will add penalty and tune the error function. In our project we have used alpha values i.e., alpha = [0.01, 0.2, 0.3, 0.4, 0.5] for initialization to both Ridge and LASSO regression. Normalization is performed on the inputs and 5-fold validation is used as it was found to give good generalization. The only difference between Ridge and LASSO regression is that Ridge performs L1 regularization whereas LASSO performs L2 regularization. RMSLE for Ridge is 0.0112 and for LASSO 0.0306.

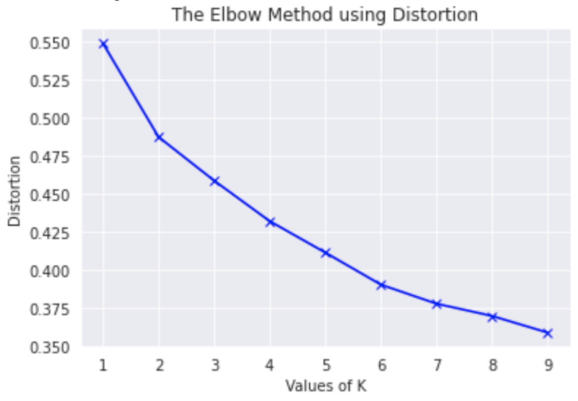
B. Boosting techniques

1) XGBoost: XGBoost implements a gradient boosting decision tree algorithm. We have used a gradient boosting tree to tune our model. Several hyper parameters are used, including estimators, max depth, base score, learning rate, jobs and regularization terms. We have tuned parameters by heuristics. This library boosts computational capabilities and it is one of the best performing libraries for our regression problem. RMSLE of this model is 0.0008.

2) CatBoost: CatBoost is another gradient boosting library based on depth factor. To grow a balanced tree, oblivious decision trees are used. It keeps on learning trees on every iteration to reduce the error made by previous tree till it builds a decent quality tree. The important factor while tuning this model is depth and number of iterations. For our regression problem we have chosen a depth of 6, number of iterations as 1000 and learning rate as 0.05. Error measure RMSLE for this boosting model is 0.0045.

C. KNN

KNN calculates distance from unknown points ’X’ to all points in the data available. For regression prediction, first we are importing K-Means to cluster and find an optimal value of ’K’ using the ”elbow method”. We created a loop that trains KNN with different k- values and an optimal K is chosen based on the elbow curve.



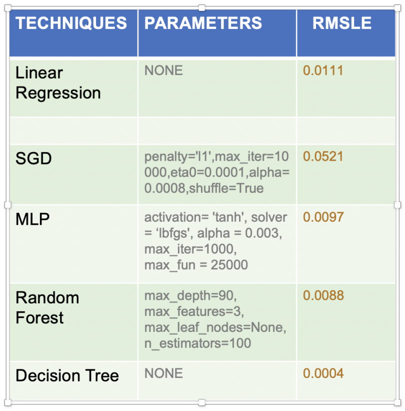
The bent position in the elbow curves gave us a optimal ’K’ value of 2. We are using this optimal K to pass through KNeighbors regressor and making predictions. With K=2, RMSLE of this model is 0.0085.

D. SVM

SVM finds a hyperplane in a multidimensional space, and using the kernel trick we can do a non-linear transform without transforming everything to the non-linear space. The parameters ’C’ and kernel are very critical to tuning. We chose a ’RBF’ kernel and ’C’ as 0.1 with other hyper parameters epsilon = 0.1, gamma = 0.1 and degree as 3. With a 5-fold validation our RMSLE for SVM is 0.0136. [7]

E. OTHER MODELS

We have implemented several other models such as linear regression, stochastic gradient descent, multi layer artificial neural networks, bagging regressors such as random forest, and a decision tree to find out the best regressor for predicting house prices. We have tuned hyper parameters for some of the models, for other models we have used the default parameters. Below table has their RMSLE scores. Up until this point, decision tree is the best performing model judging by its RMSLE score, followed by XGBoost.



TECHNIQUES	PARAMETERS	RMSLE
Linear Regression	NONE	0.0111
SGD	penalty='l1',max_iter=10000,eta0=0.0001,alpha=0.0008,shuffle=True	0.0521
MLP	activation= 'tanh', solver = 'lbfgs', alpha = 0.003, max_iter=1000, max_fun = 25000	0.0097
Random Forest	max_depth=90, max_features=3, max_leaf_nodes=None, n_estimators=100	0.0088
Decision Tree	NONE	0.0004

IV. Experimental study

All the materials related to our project is included here (https://github.com/sudha-vijayakumar/CMPE257-house-price-prediction-advancedRegression). Tools like Google collab, jupyter notebooks and libraries like scikit-learn [8], mlextend [9] has been used for the experimental study. This section is intended to discuss every step in the implementation pipeline in detail. The following figure is a high-level overview of the pipeline,

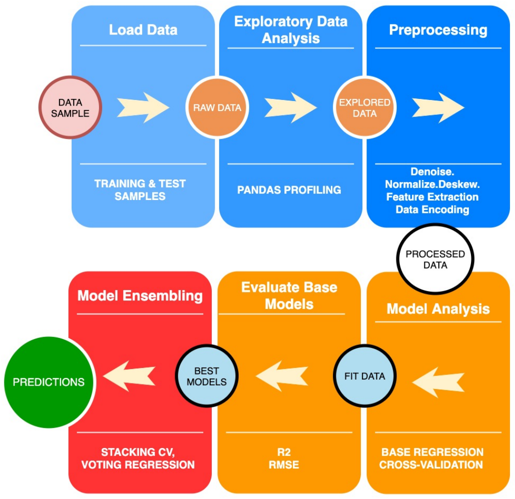


Fig. 1. Implementation pipeline

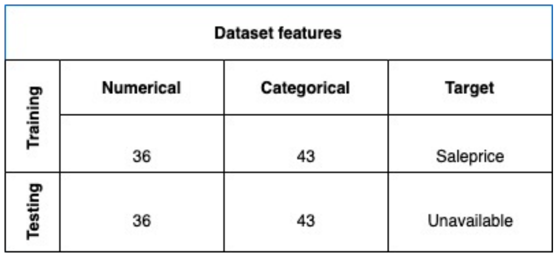
A. Load dataset

Two separate datasets, one for each training and testing, were provided separately by Kaggle. These datasets will be used for house price prediction model training and testing respectively. The training set contained 1460 labelled records and test set contained same amount of unlabelled records.

B. Exploratory data analysis

Profiling report [10] provided by Pandas library has been used for the preliminary study of the dataset characteristics. Profiling report is rendered as an interactive HTML report providing a detailed overview of the following,

- Data types, unique values.
- Minimum, maximum, range, Q1, Q3, interquartile range.
- Mean, median, mode, sum, standard deviation, sum, absolute deviation, kurtosis, skewness.
- Histogram, Correlation graphs.
- Missing values matrix, heatmap.



Dataset features

	Numerical	Categorical	Target
Training	36	43	Saleprice
Testing	36	43	Unavailable

Fig. 2. Dataset features

The above report provided a very good overall understanding of the dataset characteristics, the underlying relationship between features and the importance of features towards house price prediction. There are around 80 features describing the physical attributes of the house that would together determine the potential value of a home.

C. Preprocessing

Based on the preliminary data analysis, there were 4 different data preprocessing [11] techniques performed, and the results are discussed as follows,

1) Remove noise/ outliers: There were some inconsistent high sales prices identified for GrLivArea greater than 3500 sq.ft. compared to the majority of data points around the same region. This was visually clear. Such points were identified as outliers and removed.

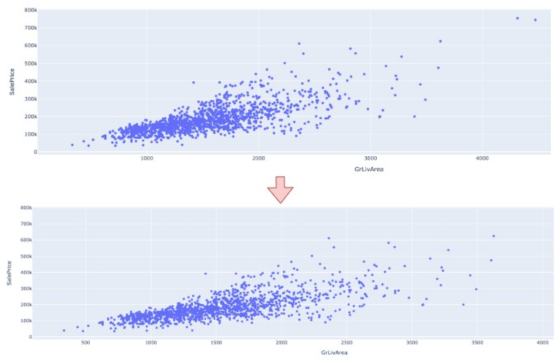


Fig. 3. Dataset features

2) Normalize target variables: The target variable sale price was found to be skewed and was normalized using MaxAbsScalar from scikit-learn library [8] as shown below,

D. Impute missing data

There were numerous missing values found in the raw dataset as shown in the below figure which could lead to incorrect prediction of the sale price of the home. Imputing

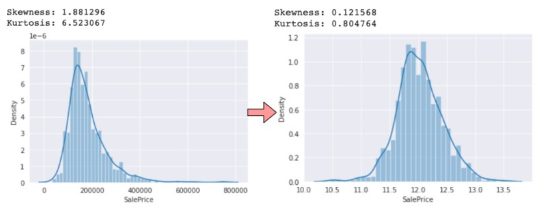


Fig. 4. Dataset features

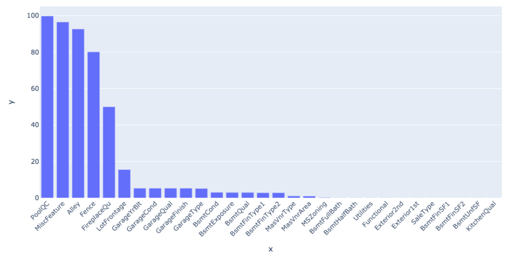


Fig. 5. Missing values

the missing values is an essential process to arrive at an accurate prediction of the sale price. The following features were imputated and the details are as follows,

- MSZoning (The general zoning classification) : replacing with mode ’RL’ zone.
- Utilities: only 3 missing values. so, it is good to drop the future.
- Electrical : replacing with mode ’SBrkr’.
- KitchenQual: replacing with mode ’TA’.
- SaleType : replacing with mode ’WD’.
- LotFrontage : imputing missing lot area with the mean of neighbouring houses square lot.
- Numerical missing values are imputed with ’0’.
- Categorical missing values are imputed with ’None’.

E. *Feature extraction*

Correlation is a measure that signifies how a dependant variable will vary according to one or more independant variables. It is imperative to identify highly correlated features (dependant variables) with the target variable (independant variable) SalePrice as they contribute more for accurate predictions. Sparse, weakly correlated features could add to unnecessary computational cost and will also affect prediction accuracy. Features with correlation value greater than 0.4 have been used for model analysis. The extracted feature map is shown in Fig. 8.

F. *Adding new feature*

One interesting finding from the dataset is that, there were 2 different fields separately to account for the total sqaure feet of the houses, TotalBsmtSF and 1stFlrSF. A new aggregate field TotalSF has been created for model analysis. The field was derived as follows:

$$TotalSF = TotalBsmtSF + 1stFlrSF$$

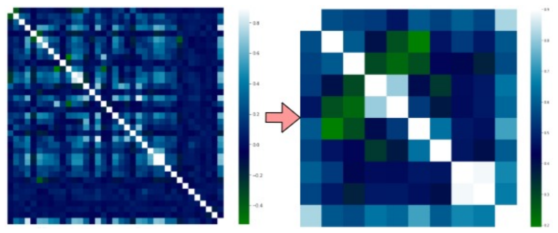
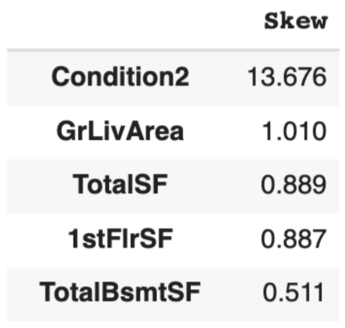


Fig. 6. Extract highly correlated features

G. *Scaling undesired skewed/kurtosis*

There were a number of skewed numerical values identified in the dataset and processed using theBox-Cox transform. Removing data skew is an important step as it allows us to make a uniform variance assumption for the errors and to linearize the fit as much as possible. The top five skewed variables are listed as follows:



	Skew
Condition2	13.676
GrLivArea	1.010
TotalSF	0.889
1stFlrSF	0.887
TotalBsmtSF	0.511

Fig. 7. Process skewed features

H. *Data Encoding*

LabelEncoder from Scikit-learn library [8] has been used to convert the features to numerical values for model analysis.

I. *Evaluate base models*

Base models were evaluated using different error measures shown below,

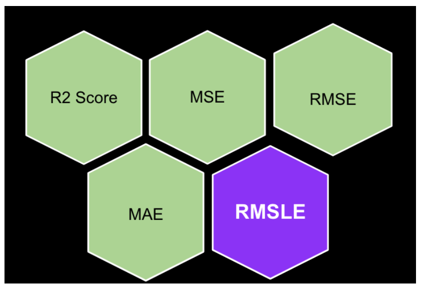


Fig. 8. Different error metrics

1) Root Mean Square Logarithmic error(RMSLE): RMSLE metric has been used as the main metric to evaluate the performance of base regressors. Fig. 10. shows the performance of base models,

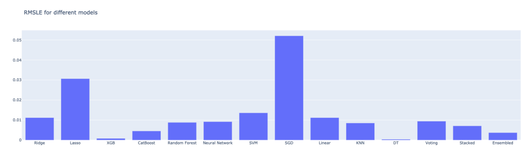


Fig. 9. Model evaluation

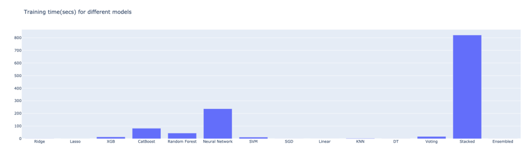


Fig. 10. Training time

2) Training time: The training time of various base models is shown below, From the above comparision, XGB has been identified to out perform rest of the base regressors.

J. Advanced regression techniques

The following advanced regression techniques [12] has been experimented and results presented,

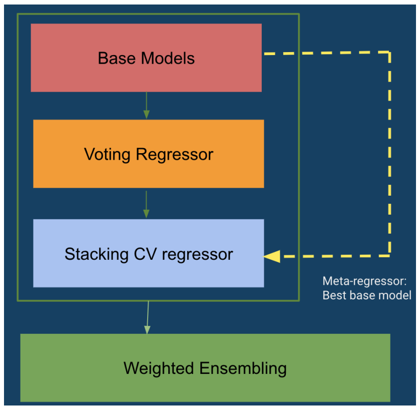


Fig. 11. Advanced regression

1) Voting regression: A voting regressor [13] ensembles several base regressors, each on the whole dataset. The predicted regression target of an input sample is computed as the mean/average of the predicted regression targets of the individual regressors in the ensemble as shown below,

2) StackingCV regression: StackingCV regression [9] implements a two level regression performing cross-validation of several base regressors in the first level and a second level regression over the results from the previous level using a meta-regressor. XGB is used as the meta-regressor for this experiment.

3) Weighted ensembling: Weighted ensembling [14] is an approach to boost predictive accuracy by combining the predictions of multiple strong machine learning models as per the below constraints:

$$w(M1) + w(M2) + ... + w(Mn) = 1$$

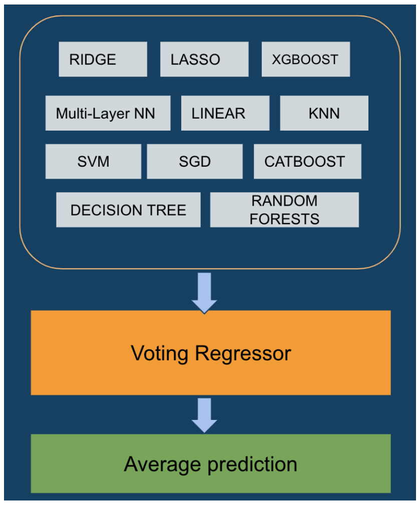


Fig. 12. Voting regression

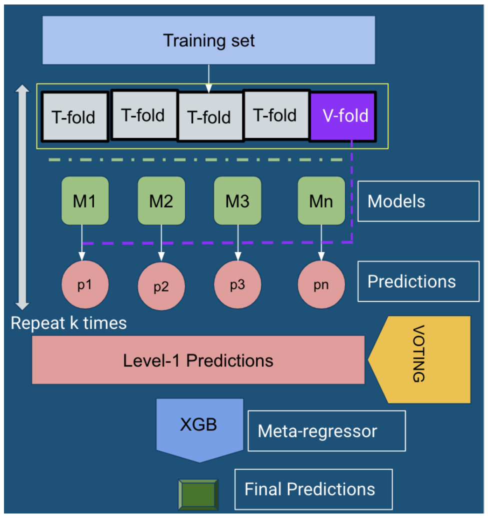


Fig. 13. StackingCV regression

$$w(M1) * p(M1) + w(M2) * p(M2) + ... + w(Mn) * p(Mn)$$

where 'w(Mn)' is the weightage allocated for regression model prediction 'p(Mn)'.

V. Result Analysis

1) Training and Testing error: Training error using the hybrid weighted ensembling approach and test error obtained using the Kaggle submission interface are presented below, Based on the above results, the above hybrid regression [14] approach provided the optimal bias-variance tradeoff compared to single regression models.

2) OLS Regression fit: The below regression fit obtained as a result of the hybrid weighted ensembling approach also provides a visual indication of a good regression result of the target variable with respect to the multiple independant variables used in the experiment.

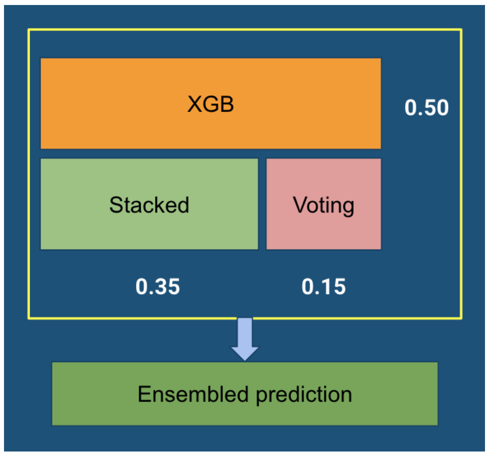
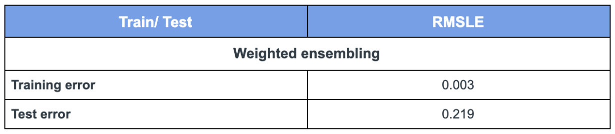


Fig. 14. Weighted ensembling



Train/ Test	RMSLE
Weighted ensembling	
Training error	0.003
Test error	0.219

Fig. 15. Training vs Test error

VI. Conclusion

In this study, we exhausted many of the currently available options for regression. Among base models, we found the decision tree gave us the lowest cross-validation error. However, when continuing on to more advanced techniques such as stacking and weighted ensembling, we noticed the decision tree model was performing worse than a boosted version of the decision tree (XGBoost) as a meta-regressor. Since ensembled methods such as XGBoost are known to be more stable than a single decision tree, we proceeded with this model as a meta-regressor for our stacking procedure.

By combining our advanced techniques using weighted ensembling, we achieved minimum error scores in the cross-validation portion of our experiments. An interesting conclusion one could derive from these results is that in the realm of regression, advanced techniques seem to perform better while also avoiding overfitting.

When submitting our results to Kaggle, we achieved a test RMSLE error of 0.21932, ranking us in about the 50th percentile of all submissions to the competition. While these results are somewhat promising given that we are newcomers to machine learning and regression, we have some ideas on how we can improve in the future. Foremost of these is deep diving into our ensemble method to find the optimal weights for this model.

After fine-tuning our model, we would like to deploy it as a web application or mobile application allowing interested buyers to predict the price of their homes in real-time. A robust model could revolutionize the real-estate business! Predicting house prices, which was considered a pinnacle of achievement

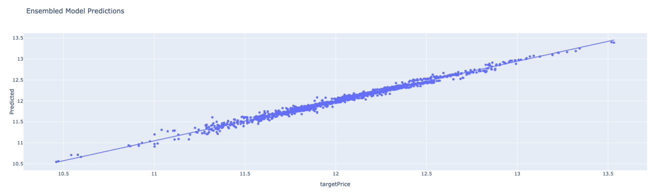


Fig. 16. OLS regression fit

for economists a few decades ago, is now feasible thanks to machine learning and data availability.

VII. Contribution

We all contributed to finding the problem and dataset, writing proposal, preprocessing, researching regression techniques, implementing them, writing the report, and preparing the presentation. Akash was involved in the preprocessing (standardization, normalization, and removing lowly correlated features), and implementing Neural Network, Ridge, and LASSO models. Sreeja was part of EDA and worked on KNN, Stochastic GD, Linear Regression, and SVM. Sudha worked on imputations as part of preprocessing, cross validation, voting regression, stacking regression, weighted ensembling and generated model evaluation reports. Yashwant was involved in the EDA and worked on the models Decision Tree, Random Forest, and Boosting techniques.

References

[1] R. A. Dubin, "Predicting house prices using multiple listings data," *The Journal of Real Estate Finance and Economics*, vol. 17, no. 1, pp. 35–59, 1998.
[2] A. Caplin, S. Chopra, J. V. Leahy, Y. LeCun, and T. Thampy, "Machine learning and the spatial structure of house prices and housing returns," *Available at SSRN 1316046*, 2008.
[3] G. D. Sutton *et al.*, "Explaining changes in house prices," *BIS quarterly review*, vol. 32, no. 1, p. 47, 2002.
[4] V. Limsombunchai, "House price prediction: hedonic price model vs. artificial neural network," in *New Zealand agricultural and resource economics society conference*, pp. 25–26, 2004.
[5] R. A. Maronna, "Robust ridge regression for high-dimensional data," *Technometrics*, vol. 53, no. 1, pp. 44–53, 2011.
[6] J. Huang, S. Ma, and C.-H. Zhang, "Adaptive lasso for sparse high-dimensional regression models," *Statistica Sinica*, pp. 1603–1618, 2008.
[7] H. Yu and J. Wu, "Real estate price prediction with regression and classification," *CS229 (Machine Learning) Final Project Reports*, 2016.
[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.
[9] S. Raschka, "Mlxtend," 2016.
[10] S. Brugman and I. Eaves, "Visions: An open-source library for semantic data," *Journal of Open Source Software*, vol. 5, no. 48, p. 2145, 2020.
[11] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. Springer, 2015.
[12] J. Mendes-Moreira, C. Soares, A. M. Jorge, and J. F. D. Sousa, "Ensemble approaches for regression: A survey," *Acm computing surveys (csur)*, vol. 45, no. 1, pp. 1–40, 2012.
[13] K. An and J. Meng, "Voting-averaged combination method for regressor ensemble," in *International Conference on Intelligent Computing*, pp. 540–546, Springer, 2010.
[14] S. Lu, Z. Li, Z. Qin, X. Yang, and R. S. M. Goh, "A hybrid regression technique for house prices prediction," in *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pp. 319–323, IEEE, 2017.