Estimate House Price Using Machine Learning

Rishita Gadde[1], Md Amiruzzaman[2]*, Richard Burns[3], Md. Rajibul Islam[4], Rizal Mohd Nor[5]

[1]Department of Computer Science, West Chester University of PA, West Chester, USA, rg991837@wcupa.edu

[2]Department of Computer Science, West Chester University of PA, West Chester, USA, mamiruzzaman@wcupa.edu

[3]Department of Computer Science, West Chester University of PA, West Chester, USA, rburns@wcupa.edu

[4]Department of Computer Science and Engineering, Bangladesh University of Business and Technology, Dhaka 1216, Bangladesh, md.rajibul.islam@bubt.edu.bd

[5]Department of Computer Science, International Islamic University Malaysia, Kuala Lumpur, Malaysia, rizalmohdnor@iium.edu.my

Received: January 15, 2023; Accepted: April 10, 2023; Published: May 02, 2023

Abstract

The House Price Index (HPI) is commonly used to estimate the changes in housing prices. The sale price of the house is correlated with many other factors like geographical location, size of the house, age of the house, the area and population of the neighborhood etc. There has been a considerably large number of datasets released in the literature of various locations to explore the correlation of the sale price of houses with their corresponding features. However, all the features don't affect the sale price in equal proportion. Some features strongly correlate with each other and, while some features don't carry any importance or are probably redundant. As a result, to explore various impacts of features on sales prices, we performed a detailed data analysis on the original house dataset. This report also comprehensively validates multiple steps of data analysis with supporting statistics and visualizations to provide an optimistic result of various features and their impact on the sales price of houses.

Keywords: feature engineering, exploratory data analysis, cross-validation.

1 Introduction

The House Price Index(HPI) [27, 29] is a broad measure and often serves as an indicator for the house price in a given geographical area. This measure is a weighted repeated sales Index, meaning it measures the average price change based on repeated sales. The HPI gets updated either monthly or quarterly basis and basically gives an idea of estimated changes in the rates on houses in a given geographical location [26]. However, there is a problem with HPI in predicting the house price of a specific house. HPI gives an overall trend of the house prices whether they prices are high or low, however, it discards the individual house information and thus makes it hard to predict the individual house price [21, 25].

In recent years, with Big Data and Machine learning becoming ubiquitous there has been a lot of data collected for the sale price of the house with lots of attributes that give out information about the house [26]. Several methods [22, 24] explored the problem of house price prediction and proved that the

predictions are close enough to the actual price; however, in this report we will not try to predict the sale price of the house but we try to analyze the data and its attributes to explore the important features, redundant features and other correlations with analysis and visualizations.

Looking at the importance of the data analysis, this report thoroughly goes through many techniques like data cleaning, data preprocessing and feature transformation to explore the effect of each attribute to the sale price of a given house. We used the “Ames housing dataset”, a residential property in Ames, Iowa from 2006 - 2010 [27, 28]. The dataset [28] has many records with self explanatory variables for each record. The variables determine the geographical location, area, size, age and many other features of a given house. Most of the variables are exactly the type of the variables that a typical home buyer would want to know about the property.

In general, all these properties usually a potential buyer would like to know about. However, there are only a few variables that bias the buyer to put forward a price.

1.1 Problem statement

The problems of the dataset are as follows 1) Outliers - the dataset contains outliers i.e. partial sales that likely donâ€™t represent the actual market values and also unusual sales i.e. very large house price deviation relatively with its neighborhood. 2) unusual sizes - there are very few data records that are very large in size and affect the total statistics of the whole raw dataset. 3) The other major problem is that with few number of data points and many explanatory variables we need to perform data transformation and convert into a low dimensional data for better analysis. 4) The dataset also has missing values, working with missing values would mislead the analysis of the dataset. Thus, studying the core features that have high impact on predicting the sales price of houses on this challenging dataset with the help of exploratory data analysis is the goal of this report.

1.2 Purpose of the study

The goal of this study is to perform exploratory data analysis of the features affecting the sales price of houses from "Ames housing dataset" [27] . The dataset contains nearly 4000 data points and approximately 80 self explanatory variables. Out of the 80 self explanatory variables 23 are of nominal type, 23 ordinal, 14 discrete type and finally 20 continuous variables. Each variable tracks different information about a given house. These variables, which served as features of the dataset, are used to predict the price of a given house.

1.3 Contributions

The contributions of this paper is listed below:

1. The research paper contributes to exploring the correlation between the sale price of houses and various features, such as geographical location, size of the house, age of the house, the area and population of the neighborhood, etc. This can help to understand the factors that influence housing prices and to make more informed decisions about buying and selling properties.

2. The paper also contributes to identifying which features strongly correlate with each other and which features are redundant or carry little importance in determining the sale price of houses. This can help to improve the accuracy of models that predict housing prices and to reduce the complexity of these models by eliminating irrelevant features.

3. The paper provides a detailed data analysis with supporting statistics and visualizations to validate

the results, which can increase the credibility and reproducibility of the findings. This can also serve as a reference for other researchers who are interested in exploring the impact of different features on housing prices.

2 Literature review

Existing studies have explored the correlation between housing prices and various features. For example, [1, 19, 20] examined the impact of neighborhood characteristics on housing prices using a hedonic regression model. Authors of [19] found that factors such as crime rates, school quality, and proximity to amenities had a significant impact on housing prices. Another study by [1] examined the relationship between house prices and environmental amenities such as parks and open spaces. They found that these amenities had a positive impact on housing prices. Exploratory Data Analysis (EDA) is a method for examining and summarizing a dataset's key features. The mainÂ characteristics of home prices in the context of this study, such as lot size, house age, number of bedrooms and bathrooms, location, and building condition, have been identified using EDA. The relationship between home prices and other aspects can be better understood by examining these variables, which can then be used to create housing price prediction models.

Exploratory Data Analysis (EDA) has also been used extensively in previous studies to analyze housing data. For instance, [2] used EDA to identify the key factors influencing housing prices in a specific region. They found that factors such as lot size, age of the house, and number of bedrooms and bathrooms were important predictors of housing prices. Similarly, a study by [3-5] used EDA to identify the key factors influencing housing prices in Madrid. They found that factors such as location, size of the property, and condition of the building were important predictors of housing prices. Feature engineering has also been explored in previous studies. For example, a study by Wang and Li [6, 7] used principal component analysis (PCA) to reduce the dimensionality of housing data and improve the accuracy of predictive models. Similarly, an article by [8] proposed a feature selection method based on mutual information to identify the most relevant features for predicting housing prices.

Machine learning models have been widely used in previous studies to predict housing prices. Linear regression models have been used extensively, such as in the study by [9]. Other studies have explored more advanced machine learning models, such as decision trees [12], random forests [13], and neural networks [14, 17]. In addition to models, other ideas such as data preprocessing techniques [11, 15, 16] are also commonly considered in the machine learning pipeline. Finally, evaluation is another important aspect of machine learning modeling. Various evaluation metrics have been used in previous studies, such as mean absolute error, mean squared error, and coefficient of determination (R-squared) [18].

3 Method

The study started by finding a dataset with more features that can be used with various machine-learning techniques for analysis. The House Price dataset from Kaggle, which contains a variety of features relating to housing properties and their sale prices, was obtained as part of the process for conducting this study. We choose to carry out a thorough data analysis to determine which features have the strongest correlation with the sale price of a house after analyzing the dataset and interpreting the numerous attributes. This involved cleaning and processing the data, identifying any missing values and outliers, and performing various statistical analyses such as correlation analysis and regression analysis. To better comprehend the relationship between the attributes and the sale price,

we used several visualization techniques, including scatter plots and heatmaps. The objective is to determine the most important features that impact the sale price of a house and provide a comprehensive report on our findings.

3.1 Data

The dataset contains 77 characteristics of residential properties, including information on the property's zoning, lot size, construction date, and various features like the number of bedrooms, bathrooms, and fireplaces. The main target variable is SalePrice, the price at which the property was sold. This dataset is used in regression and feature engineering challenges in this paper.

3.2 Analysis

In the following section we conduct different types of analysis on the Ames Housing dataset. We also display the results and our findings both empirically and visually to support our claims on the housing dataset.

3.2.1 Exploratory Data Analysis (EDA)

In the project EDA is a really important step. The dataset has many many features that are correlated, has outliers, missing values etc. So, before fitting a machine learning data engineering is important. A clean data can give higher performance on a simple baseline model as well. Hence, In the following sections we discuss the exploration data analysis of our dataset in detail.

3.2.2 Dependent Variable Analysis

Ideally having an output variable in the form of normal distribution is good. It helps us to fit a better model as most of the data lie towards the center and there are always data points towards extreme which indicates the data points at 2 or 3 standard deviations away from data. First, letâ€™s visualize the output variable.

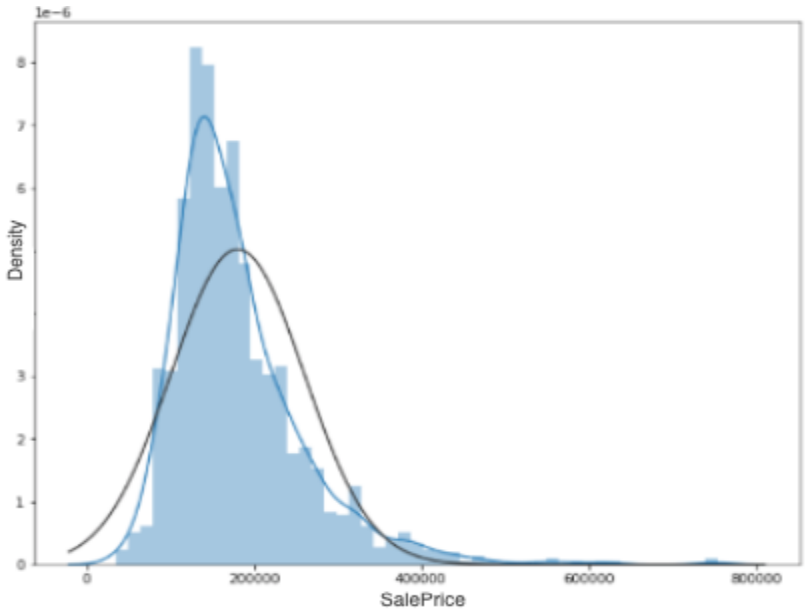


Figure 1. Output Variable Analysis. The Image displays the skewed distribution of output variable.

In the figure 1, we can see that the dependent variable is not normal and In Fact it is skewed towards right. Also, the statistics shows that the mean is relatively higher than median which also shows that the outliers towards the right are really large and having a greater impact on the mean. The above claim can also be supported by the statistics that the max value is at least 3 times larger than the 75th percentile.

Now, letâ€™s remove some outliers and re-visualize the dependent variable. In Fig 2 we can see that, though we removed some outliers to the right we still have the curve skewed to right. Removing right-leaning outliers may lead to a small improvement in the data's distribution, but the curve's general shape may still be skewed to the right. This may be a sign that some underlying variable or factor is affecting the data's distribution in this manner.

The other guess for this skewness might be the fact that the outliers are really large compared to the 50th and 75th percentile, and when they are in their natural order the spacing is high. So, let's apply a transformation and bring them to logarithmic order and see. Now, transforming the output variable with log, the dependent variable is more towards Gaussian with reduced skew and Kurt. The transformed space can be seen in figure 3:

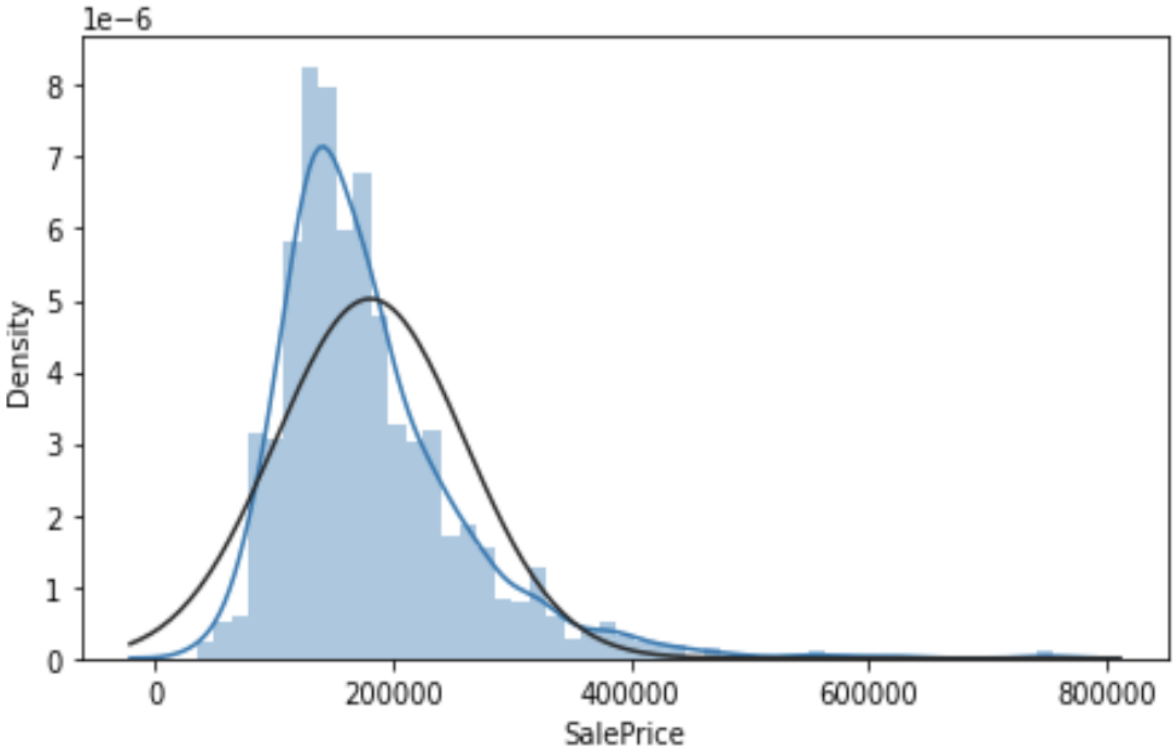


Figure 2. The output variable with outliers removed. The data distribution of output variable now looks bit towards normal distribution but still skewed towards right.

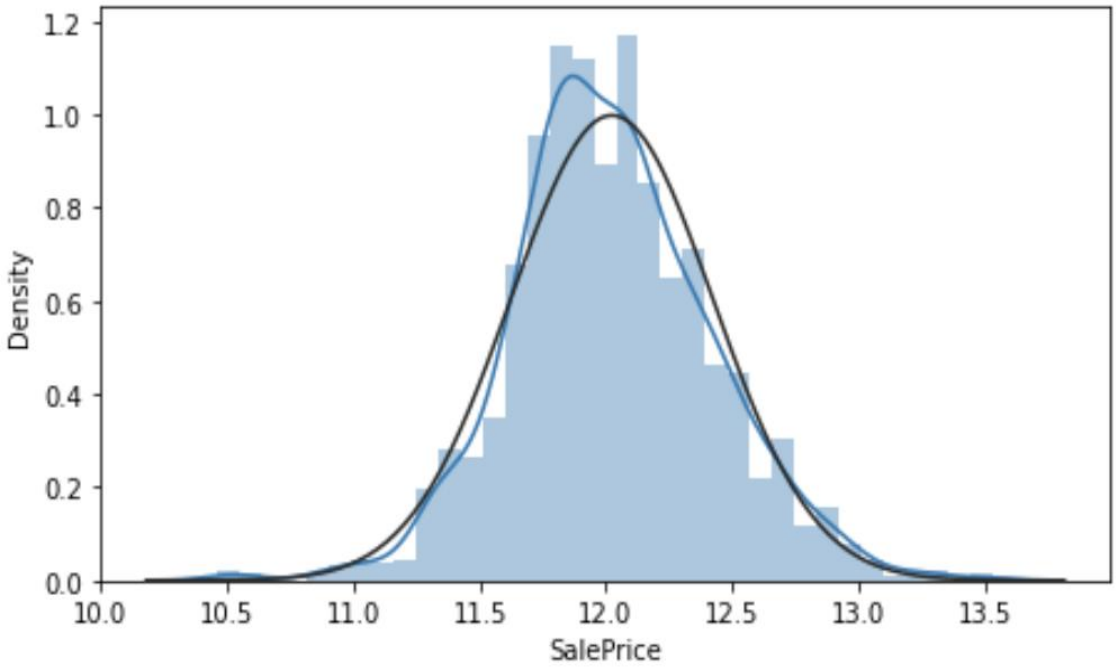
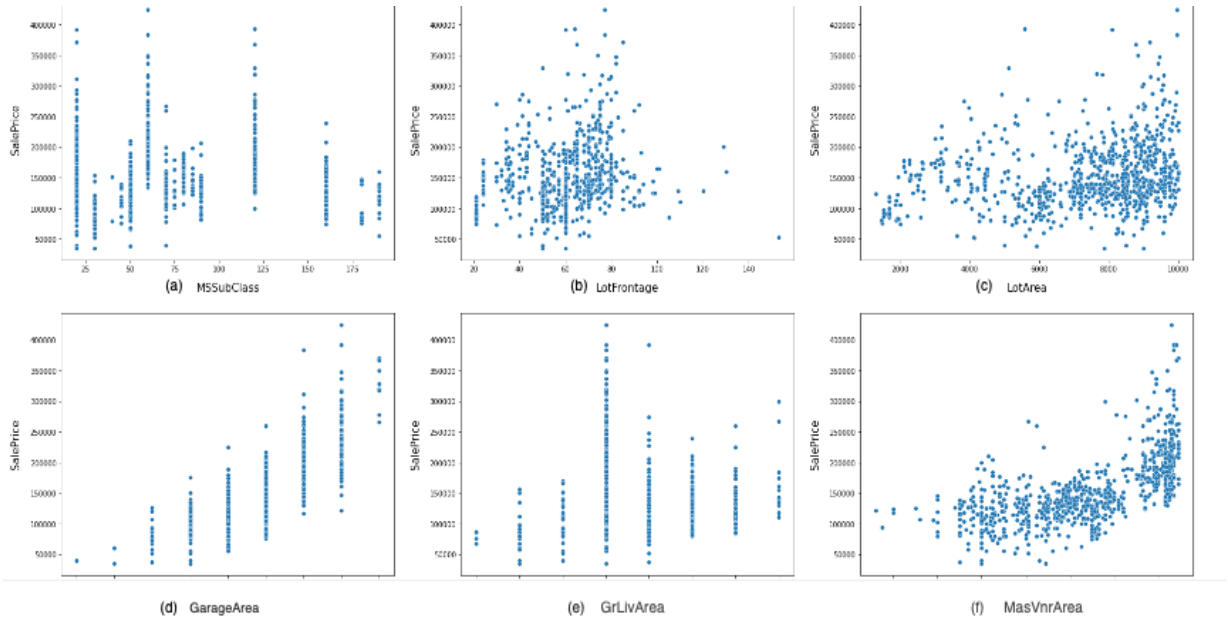


Figure 3. The data distribution of output variable after applying the Log transformation. The dependent variable this look more towards gaussian with reduced skew and Kurt.

3.2.3 Feature Analysis

There are many types of input features in the dataset. Categorical, numerical etc. Letâ€™s analyze and visualize 3.2.3 the numerical features as they are the majority nearly th of the features. And dive more into the type of numerical we can see.

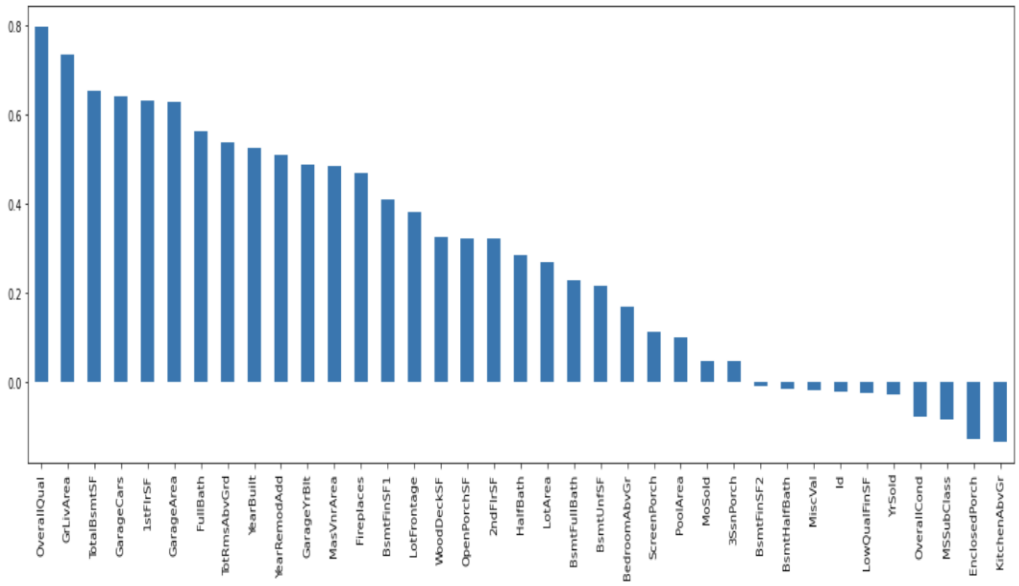


Visualizations of the numerical features in the dataset against the dependent variable SalesPrice.

In the below figure 3.2.3, we visualized different numerical features vs dependent variables. The goal is to see how they are related. In the below 3.2.3, we can see that some are continuous and spread across the whole 2d space, while some of them take a fixed set of values like 5-6 values is the total space of that feature. While some are a mix of both. Hence, we can conclude that the numerical features are both continuous and discrete. From these visualizations we can do better data engineering like scaling different features based on their space.

3.2.4 Correlation Analysis

From the visualizations in the figure 3.2.3, one thing we observed is that in continuous variables the features and dependent variable are proportionally related while sometimes not. And it was very difficult to find patterns with discrete values. Hence, to get an idea of how each feature affects the SalesPrice, we try to find the correlation between the numerical features and the output dependent variable.

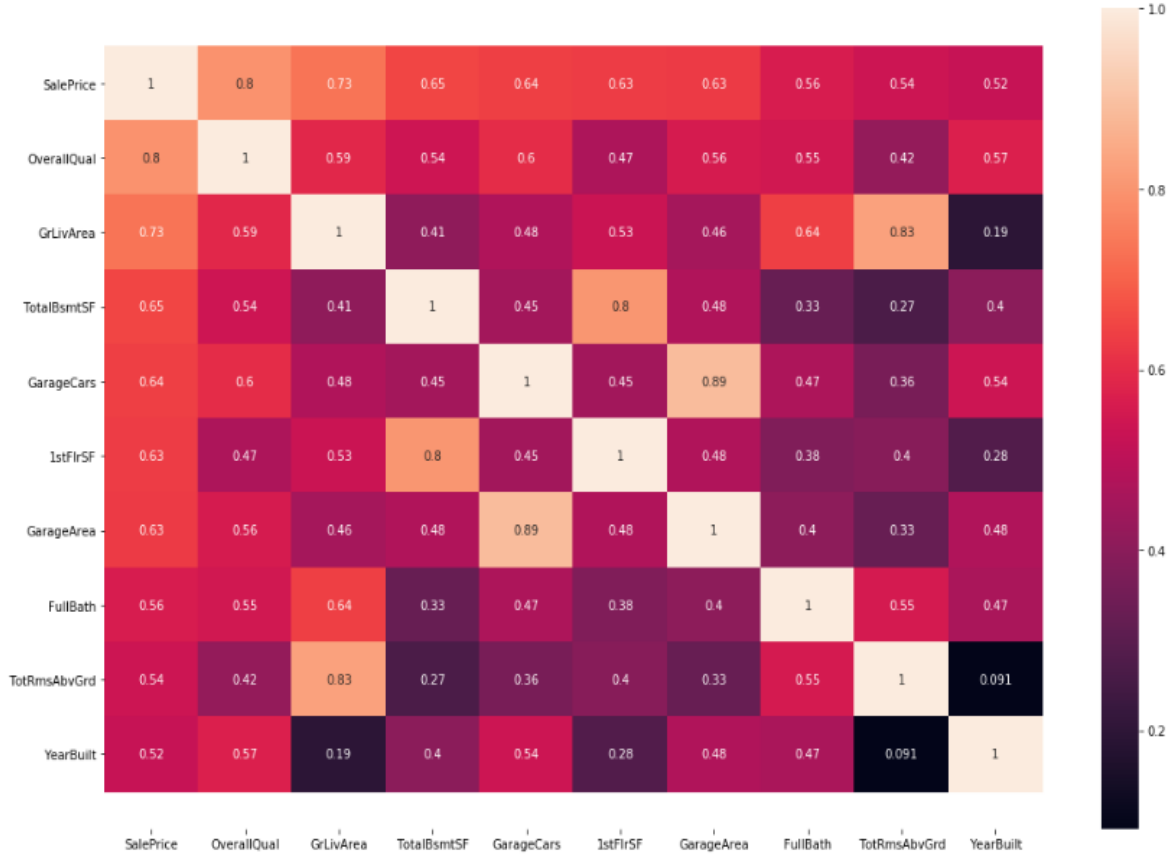


The plot shows the correlation between the output variable Sales Price against the numerical features in the dataset.

The figure 3.2.4 shown infact there is much more informative than just a bar chart. We can clearly observe that some variables are highly correlated with the output, while some have barely any effect on the output variable, and interestingly some have negative effects. The visualization shows that the â€œ Sale Price is higher if the overall condition is lowerâ€• which is against the reality. Hence, we need more data engineering for building quality features before fitting a machine learning model.

Before we go into further data engineering, let us also analyze how the features are cross correlated. Due to the high number of features we have, letâ€™s take only the top 10 from the previous plot and see the cross correlation. The goal is to see that if the cross correlation is really high, then the top 10 features are basically nothing but the features that are providing similar information. The disadvantage with these features is that though we are using more features we are again relying on the same type of information these correlated features are providing.

The following heatmap in figure 3.2.4 shows how the top 10 features are correlated with each other. (I have also added the dependent variable for better visualization.)

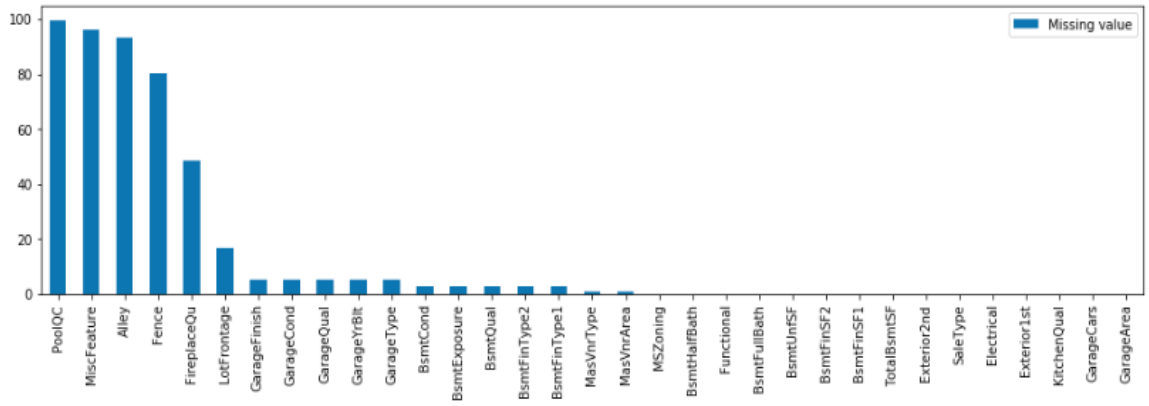


Heatmap showing the correlation among the top 10 features in the dataset. The value closer to 1 indicates the features are highly correlated and the value towards 0 are the features that have less relationship between them. The correlation can also be estimated for other features but show the visualizations only for the top 10 features.

An interesting thing to note here is that the OverallQuality feature had very high correlation while OverallCondition had negative correlation which is counter intuitive. But, our understanding and claim is that both features provide similar information, and as one feature(overall quality) is given very higher importance than required it is kind of minimized by giving negative importance to OverallCondition hence the counter intuitiveness. So, doing good data engineering like combining features can result in a single feature with logical correlation.

3.2.5 Missing Value Analysis

Another important challenge in EDA is the missing values. We show the number of missing values in each feature in the figure 3.2.5 below:



The plot demonstrates the number of missing values of individual features available in the dataset. The reading on the x-axis represents a features and the readings on the y-axis are the number of missing values. For better visualization the features with missing values are sorted on x-axis before displaying.

Building a machine learning model with missing values is not ideal as we are basically fitting with randomized data and it does not generalize well. Hence, to counter this adversary, we fill the missing values with mean and medians for continuous values, and mode for the features with limited discrete values. Better and intelligent strategies can be applied for improving the data quality, but in our report we assume that our strategy is a good start hence we limit the scope of our work here.

3.3 Feature Engineering

In advanced feature engineering, we discuss the importance of combining the features. In the correlation section, we saw that having multiple features which provide similar information have inverse effects on the dataset. To counter this effect we improve the data quality by combining features which provide similar information or meaning.

We can build a feature like HouseCondition by adding the features overallQuality and overallCondition. Thus, making a single variable from 2 features that provide similar information. We can build the future HouseCondition again as a continuous variable by just adding them or weighting averaging them, or again make it discrete by adding the features, then compartmentalizing them into different sets like poor, good, excellent etc. We can try out different strategies. Also another example, NumberOfBathrooms can be a new feature. There are multiple features like FullBath, HalfBath, BsmtFullBath, BsmtHalfBath. Again performing an addition, or weighted averaging based on which you think are most important will also be helpful. Thus, combining 4 different ones into 1 can give a better idea for the machine learning model

Advanced feature engineering techniques involve producing new features by performing mathematical operations or transformations on existing ones in addition to combining comparable features. For instance, to build a new feature that reflects the ratio of the basement area to the first floor area, we can compute the ratio of two features, such as TotalBsmtSF and 1stFlrSF. Similar to this, we can produce new features by subjecting the existing features to logarithmic or polynomial transformations, which can reveal non-linear correlations between the features and the target variable. Overall, sophisticated feature engineering methods can considerably increase the predictive capability of machine learning models and facilitate better decision-making in a variety of industries, including marketing, finance, and healthcare.

3.4 Model Selection

In this section we choose a set of machine learning models and try to fit them to our feature engineered data. We then observe the error metrics and performance metrics to identify our ideal model.

3.4.1 Linear and Ridge Regression

Linear regression Figure 4 is a class of machine learning models that tries to model the relationship between two different variables by fitting a linear equation to the observed data. One variable is considered to be the dependent variable and the other is supposed to be the explanatory variable. Before we try to fit the model to the data we need to identify if there is any relationship between the variables and does it make any sense in fitting the data. This can be understood by doing EDA that we discussed in the previous section.

A linear regression has an equation of the form , where is the dependent variable and is the explanatory variable. The slope of the line is and the intercept is . The most common method fitting the line is the method least squares. This method calculates the best-fitting line for the observed data by minimizing the sum of squares of the vertical deviations from each data point to the line. Observe that if the point lies exactly on the fitted line then the vertical deviations are basically 0, because the deviations are first squared, then summed and there are no cancellations between the positive and the negative values.

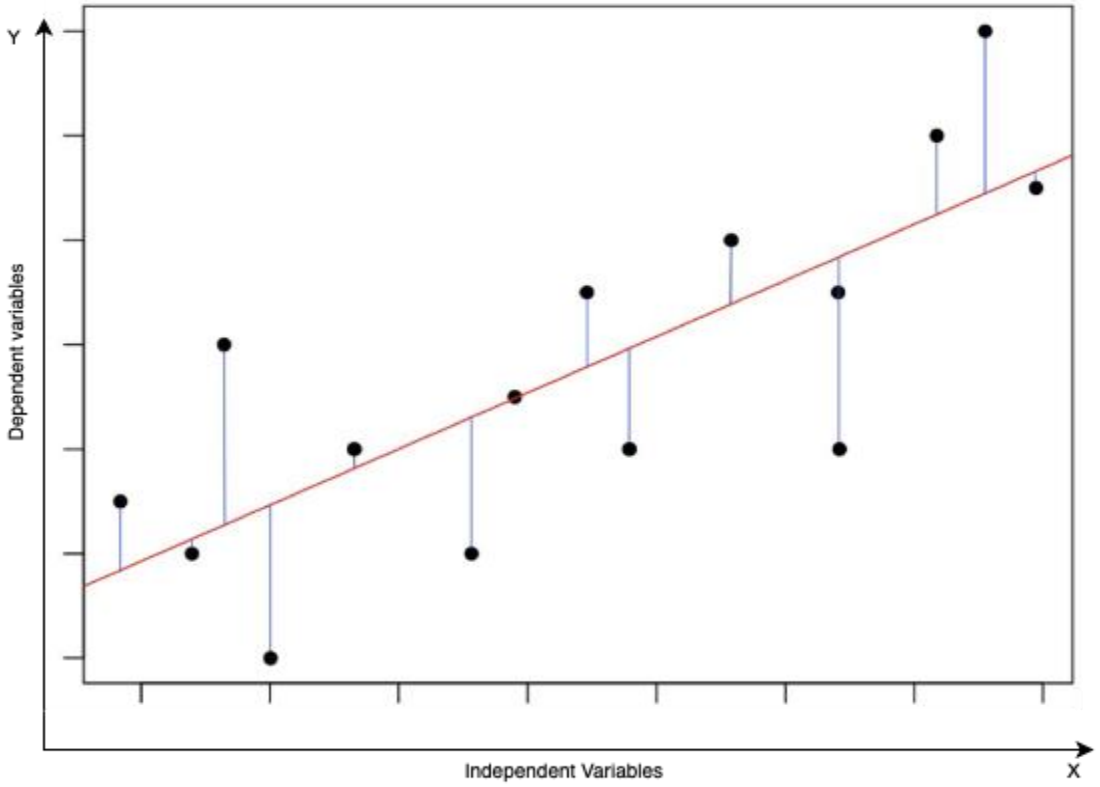


Figure 4. The figure demonstrates the general idea of linear regression. The values on -axis are the values of and the values on the -axis are the values of . The red line that goes through the points is the decision boundary of linear regression i.e. the ideal fit line that is closest to all the points. The blue dotted lines from each point to the decision boundary is the error we get for each point.

In the above figure 4 we can observe the vertical deviations because we square them summing up the deviations never result in zeros.

Now, moving into ridge regression the concept is closely similar to linear regression, however ridge regression is a technique used when the data suffers from multicollinearity i.e. independent variables are highly related (which we discussed from our EDA section). In multicollinearity, even though the linear regression estimates are good and unbiased, their variances are large and deviates the observed value far from true value. By adding a degree of bias to the linear regression, we reduce the standard error and this is ridge regression, where is the regularization parameter, is no.of features, is coefficients, is no.of samples, target variable for th sample, is Each component of this vector of predictor variables for the th sample represents the value of the th predictor variable for that sample (see Equation 1).

$$J(w) = \lambda \sum_{j=1}^{p} \beta_j^2 + \sum_{i=1}^{n} (y_i - \sum_{j=1}^{p} x_i \beta_j)^2 \quad (1)$$

In the above image, the equation with only the first term is linear regression and objective with both the terms (second term is regularization) is called ridge regression.

3.4.2 Support Vector Regression (SVR)

Support vector regression Fig. 5 is another class of machine learning models that analyze data for regression analysis. These are one of the most robust methods based on statistical learning frameworks. These models map training examples to points in space so as to maximize the width of the gap for minimizing the error. During testing, the new examples are mapped to the same space and the dependent values are estimated. These algorithms are mainly known for their kernel trick, implicitly mapping their inputs into high dimensional feature spaces for better modeling. In overall, the basic idea behind SVR is to find the best fit line i.e. the hyperplane that has maximum number of points.

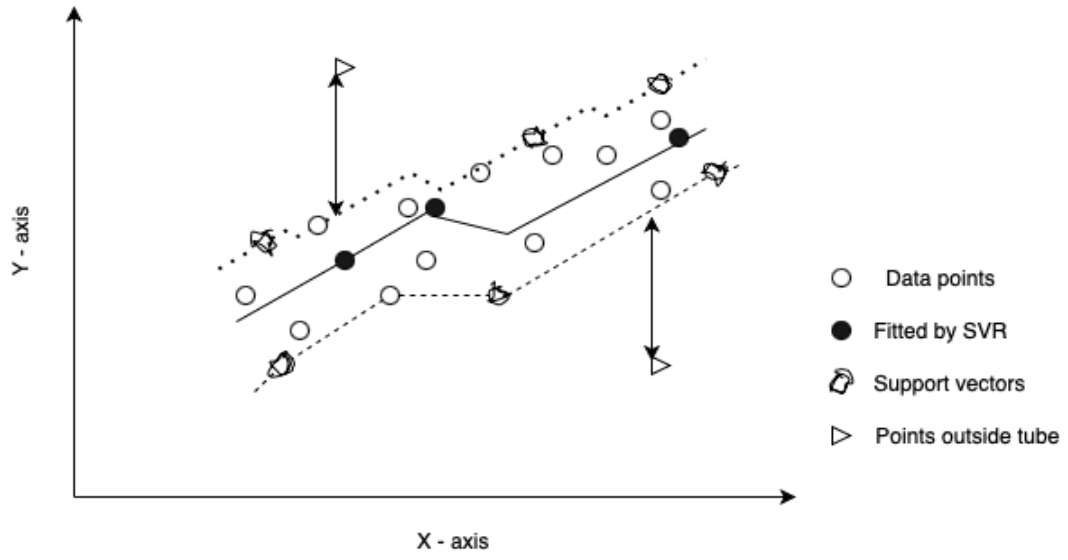


Figure 5. The image demonstrates the basic idea of Support Vector Regression (SVR). The dark lined curve at the center is the decision boundary and the dotted lines around it at a distance of are the soft margin decision boundaries.

In the equation, the dark lined curve is the hyper place we try to model, and the dotted lines around are the boundaries with a margin epsilon wide on either side. And the goal is to fit a hyperplane that maximizes the number of points that we can fit inside the hyperplane. One major thing to observe here is that unlike other regression models that try to minimize the error between the real and predicted value, the SVR tries to fit the best line within a threshold value. The threshold value is the distance between the hyperplane and the boundary line. The cost function for SVR is as follows:

$$\min \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}|\xi_i| \; s.t. \quad |y_i - w_i x_i| \leq \varepsilon + |\xi_i| \tag{2}$$

Let's see the objective in detail. Unlike other problems here we are trying to fit the hyperplane to contain as many samples as possible, hence we try to minimize the (the slope of the line) to fit the data, and we also add the bias margin because, if we have infinite margin then we can fit all data points but the margin around hyperplane is really big, hence the goal is to have a limited margin with which can achieve the maximum accuracy. Also, we minimize the objective under the constraint that absolute error less than or equal to specified margin, called maximum error. And this a hyper parameter that we can tune to gain the desired accuracy of the model and it has its own limitations. Overall, one has to understand that SVR is an advanced algorithm that allows us to choose how tolerant we are of errors by defining an acceptable margin and tolerance of falling outside acceptable error rate. the slack variable that allows for points to be on the wrong side of the boundary line but within the margin, the number of data points in the training set, the input features for the -th data point, the target output value for the -th data point, the margin or distance between the hyperplane and the boundary line.

3.5 Decision Tree Regressor

Decision Tree Regressor is another machine learning algorithm used to predict the value of a continuous target variable based on a set of input features. It's an iterative and top-down way of constructing the hypothesis. In Decision trees, we iteratively divide the dataset subspace to make active decisions i.e. once the dataset sub spaces are created, when a new data point of interest comes in, we return the decision based on which subspace the new data point belongs to or until a stopping criterion is reached. In Decision trees, each subspaces are kind of some boolean functions and we greedily search through the boolean functions by making a series of locally optimal choices to actively return a decision. In simple terms, a decision tree can be understood as a combination of nested If-else rules where each rule can have multiple constraints in it.

In Decision Tree algorithm, we first pick a feature that gives us the largest information gain of the dataset as set it as root node. If the feature is able to perfectly divide the dataset into 2 pure subspaces where an active decision can be made then we stop constructing the tree, else in cases where the split subspaces based on are not pure then we pick another feature from the remaining set of features and further build the decision tree until we get pure leaf nodes.

The Decision tree is a really strong algorithm that built properly gives outstanding results on many levels. The decision tree regressor i also easy to interpret and understand. The resulting tree can be inspecte and visualized to understand its predictions. Decision trees can also handle a mix of categorical and numerical features and select the most relevant features to include in the model. Thus Decision trees are a very good choice for EDA and feature selection. However, Decision tree algorithm also has a few practical limitations like how do we start selecting the features to build tree or sub-tree i.e. selecting criterion that explains how good are we doing in splitting the dataset into subspaces, again when do we stop building the tree i.e do we go all the way utilizing all input features in the data and hard condition to split the subspace or do we setup a stopping criterion and stop

progressively constructing the tree after using a number of features.

In Decision trees, a criterion is often required to select features at each node. This is the critical aspect of the algorithm that determines the quality and performance of the resulting model. Decision tree algorithm has several criterion that can be used to select the features. Some of them are:

3.5.1 Information Gain

Information Gain is a widely use criterion for selecting features in the decision trees. It measures the reduction in entropy or impurity achieved by splitting the data on a particular selected feature. Thus, features with high information gain are preferred to split the data at a given node because they lead to more significant reduction in impurity or otherwise . In other words, dividing data on this feature gives out the most information from the dataset.

Let D be the dataset and be a feature then the information gain of the dataset w.r.t to the feature is given by

$$\mathrm{IG}(D,F) = \mathrm{H}(D) - \mathrm{H}(D \mid F) \tag{3}$$

The entropy of the target variable $\mathrm{H}(D)$ is defined as:

$$\mathrm{H}(D) = -\sum_{i=1}^{n} p_i log_2(p_i) \tag{4}$$

where n is the number of classes and p_i is the proportion of samples in class i . The conditional entropy of the target variable given feature F is define as:

$$\mathrm{H}(D \mid F) = \sum_{j=1}^{m} \frac{|D_j|}{|D|} H(D_j) \tag{5}$$

where m is the number of values that feature F can take and $|D_j|$ is the number of samples in Dataset D that have feature F equal to j .

3.5.2 Gini Index

Gini Index is another popular criterion for selecting the features or diving the data at a given node. It measures the purity of the data set by calculating the probability of misclassification if a random sample is taken from the set. Hence, the features with lower Gini Index are preferred because they lead to more pure subsets.

$$Gini(D) = 1 - \sum_{i=1}^{n} p_i^2 \tag{6}$$

where $\mathrm{Gini}(D)$ is the Gini index of the dataset D , n is the number of classes, and p_i is the proportion of samples in class i .

There are other criterion exists in the literature but we discussed the ones that are most popular and often used by researchers and other practitioners. The choice of criterion depends on the problem and characteristics of the data. In practise, it is recommended to experiment both the criterion's and choose the one with best performance on the validation data.

Decision trees can suffer from overfitting, especially when the tree gets deep and complex. This can lead to poor generalization of performance on new and unseen data. However, to mitigate these limitations techniques like regularization, pruning and ensemble methods like Random Forest can be applied to reduce overfitting and improve generalization to the new unseen data.

3.6 Random Forest

Random Forest Regressor is another popular machine learning algorithm used for regression tasks. The algorithm is a ensemble learning method that combines the above discussed Decision Trees to make accurate predictions. In Random Forest, the algorithm creates a forest of decision trees and makes the final prediction by aggregating the predictions from each of the decision trees of the forest. Unlike decision tree the trees in Random Forest are not trained on the whole dataset, instead each tree in the forest are trained on a random subset of the input data and a random subset of the input features of the selected random data.

Given a dataset $D = (x_1, y_1), (x_2, y_2), ..., (x_m, y_m)$, where x_i represents the i^{th} input feature vector and y_i represents the corresponding output label, the Random Forest Regressor algorithm can be represented as follows:

1. sample a bootstrap data $D_{t'}$ from D with replacement. i.e. $D_{t'} : D$.
2. Select a random subset of features F_t from the set of input features: $F_t \subseteq f_1, f_2, ..., f_n$
3. Train a decision tree regressor $h_t(x;\theta_t)$ on the dataset $D_{t'}$ using the features in F_t, where θ_t represents the hyperparameters of the decision tree regressor:

$$h_t(x;\theta_t) = \text{DecisionTreeRegressor}(D_{t'}, F_t, \theta_t) \quad (7)$$

Random Forest Regressor has the ability to handle non-linear relationships and interactions between the input variables of the data. This makes the RF Regressor to be a superior algorithm when the relationship between the input and output variables is complex and difficult to be modeled using traditional algorithms like linear regression. Random Forest also has a very low risk of overfitting which is an added advantage. Since the trees in the Random Forest are trained only with a random subset of samples and a random subset of features, the chance of model learning from the noise in the data is minimized. Random Forest Regressor along with its high predictive accuracy is also a highly interpretable algorithm. The importance of each input feature can be easily determined by examining the contribution of each feature to the predictions of the individual trees in the forest. This feature importance measure can be used to gain insight to the underlying relationships between the input and output variables and can help us to identify important predictors in the forest for further investigation.

Random Forest also has the capability to handle both categorical and numerical features making it more versatile. Random Forest is also relatively robust to missing data and high-dimensional data with many input features. Despite all the advantages, Random Forest has some limitations as well. The model can become computationally intensive and slow to train when the number of input features, size of dataset or the max depth set is high. As the results of Random Forest are a combination of many decision trees it can get difficult to interpret the results sometimes. Finally, Random Forest is also not reliable and well suited when the relationship between input and output variables is highly non-linear. It is because the Random Forest is an ensemble of many Decision Trees and each decision tree is inherently a linear model.

3.7 Cross Validation

Once we choose the models, then the next step is to build a proxy test set. The goal is that we need a set of data to evaluate our performance and tweak the models in hand. By using a test set we face the

problem of data leakage and donâ€™t generalize well to the new data samples. Hence, the idea is to build a proxy test set called a validation set that can behave as a test set and also help us in tuning our model. There are many techniques to create validation sets, but for the scope of this project we limit our discussion only to K-Fold cross validation set as shown in figure 6. This idea is mainly helpful when we have a limited set of data.

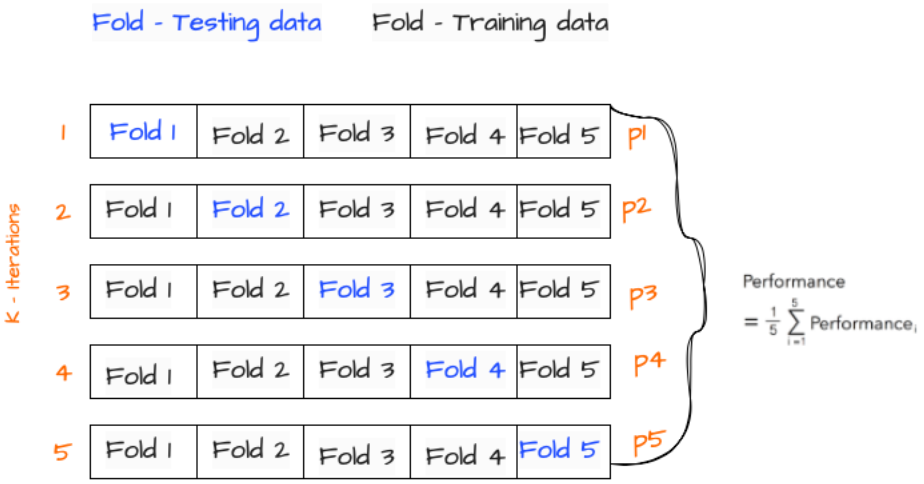


Figure 6. The idea of K-fold cross validation. Assuming the total horizontal bar is the total dataset each individual block is an equal division of dataset. The cross validation is performed the same number of blocks the dataset is divided. For each block considered as a hold out the performed is estimated.

Basically, the idea is that we dive the dataset in K-folds (i.e. k partitions) and in each iteration we take all partitions of data except 1, the remaining (k-1) partitions are used to train the data, while the left out partitions acts as the test set. Because we can leave single partitions k different ways, we perform the above process in k iterations every time leaving a different set as a test set and average the performance across k iterations. The following image demonstrates the idea visually.

4 Results

After we choose the model and build a validation set, the next step is to define evaluation metrics. Evaluation metrics are useful to estimate how well the chosen machine learning models are performing on the hidden test set and generalizing. There are many metrics we can use for regression analysis like Mean Absolute Error (MAE), Root Mean Square Error (RMSE). In our work, we mainly focus on rmse and report the rmse of different models on the test set, where n is number of observations in the dataset, y_i is the true value of the i-th observation, $\hat{y}_i$ is the predicted value of the i-th observation.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{8}$$

RMSE provides us an estimate of how far or close our predictions are to the true value on the given data point. In the above equation, y is the true value, $\hat{y}$ is the predicted value, we square the residual between true and predicted to avoid the canceling of positive and negative residuals. We then estimate the average and finally take a square root because of the squaring we initially did. This gives us the average mean error our model is making in its predictions across the whole test set. Evaluation metric and comparision of different model is presented in 1.

Table 1. Comparison of different model

Model	RMSE
Ridge Regression	0.1079
SVR	0.1081
Decision Tree	0.1926
Random Forest	0.1442

Though we see that our advanced model is doing better, data engineering helped quite a lot and the baseline model performance is quite close to our SVR.

5 Discussion

The study highlights the importance of data cleaning and feature engineering in improving the performance of machine learning models. In our ongoing research, it has become evident that dedicating adequate time on data cleaning such as finding the missing values and filling them with imputation methods and sometimes removing them altogether. The data cleaning had a significant impact on the quality of the data and also the effectiveness of the associated trained model. This approach is commonly employed by both researchers and practitioners, as datasets often contain a surplus of features relative to the available number of data points.

Another crucial issue that our study has brought to the forefront is the impact of outliers on machine learning model performance. These Outliers are deviate significantly from the rest of the data and can potentially distort the results of the analysis. In our study our dataset has handful of data points and with different scale of outliers from different features without cleaning them it would be extremely difficult to come up with a model that performs well on our dataset. We performed some visual tests and conducted some statistical analysis to extract the outliers and remove them. As an extension of our work, we recommend that researchers and practitioners employ advanced machine learning algorithms or utilize outlier imputation techniques to address the presence of outliers in datasets. Although outliers can significantly distort model performance, they may also contain valuable information that can contribute to meaningful insights. It is essential to exercise caution when handling outliers to ensure that the original dataset's representativeness and generalizability are not compromised.

In addition to data cleaning, our study also emphasizes the model performance on the Ames dataset, ranging from a simpler linear model to a more complex ensemble method. While more complex models may appear to yield better results theoretically as they seem to get past many limitations, it is not always the case, and advised to adopt a systematic and empirical approach in selecting the final model. Often times it is ideal to choose a model that has competitive performance and better interpretatbility and our study conducts experiments on just those chosen models where we interpretability rather than black box neural networks.

Our current study can be extended to further study in detail about techniques to effectively combine various features and minimize the number of independent features in the dataset. This approach can be beneficial in situations where there are many meaningful features available, but with high levels of interdependence. By reducing the number of interdependent features and enhancing their combinations, researchers can improve the model's performance and interpretability. Furthermore, the implications of our study are far-reaching, as accurate and reliable predictions with less number of data points are essential in many domains, including healthcare, finance, and climate science. Adopting the methodology and experimental approach of our study can serve as a valuable starting point for addressing many of the challenges faced in various fields where collecting large datasets is prohibitively expensive or logistically challenging. Moreover, the techniques and strategies we

discussed would be a good starting point and provide a framework for small and imbalanced datasets.

6 Conclusion and Future work

In this study, we investigated the significance of data cleaning prior to implementing a machine learning model on a given dataset. We found that the presence of outliers, missing values, and highly correlated features posed significant challenges to the model's performance. Our results highlight the importance of dedicating sufficient time and resources to data cleaning, which can significantly improve the performance of both baseline and advanced machine learning models. Additionally, we demonstrate the benefits of feature engineering, including grouping features and removing highly correlated features, in enhancing model performance. Lastly, we emphasize the role of a validation set, a proxy test set, in ensuring model generalization and robustness.

In this report, we applied basic feature engineering techniques to clean the dataset. However, there is room for further improvement through advanced data engineering. We noted that many of the features are highly correlated, presenting an opportunity to reduce the number of features while increasing their quality. To achieve better performance, we recommend exploring multiple strategies for creating an improved validation set and evaluating the model using it. Another effective technique in the field of machine learning is using ensemble models, where multiple models are utilized to make a single prediction, as each model may handle different scenarios more effectively. Furthermore, we suggest the application of feature transformation techniques, such as Principal Component Analysis (PCA) [23], AutoEncoder [11], or Linear Discriminant Analysis (LDA) [10], to reduce the dimensionality of the dataset while preserving as much of its variance as possible. This can lead to improved performance, as it is generally ideal to work with lower dimensional data.

References

[1] Kanit Wongsuphasawat; Yang Liu; Jeffrey Heer. Goals, process, and challenges of exploratory data analysis: An interview study. arXiv, 2019.

[2] Sabrina Enriquez; Fushing Hsieh. Categorical exploratory data analysis on goodness-of-fit issues. arXiv, 2020.

[3] Ana Kostovska; Diederick Vermetten; SaÅ¡o DÅ¾eroski; Carola Doerr; Peter KoroÅ¡ec; Tome Eftimov. The importance of landscape features for performance prediction of modular cma-es variants. arXiv, 2022.

[4] Amulya Agarwal; Nitin Gupta. Comparison of outlier detection techniques for structured data. arXiv, 2021.

[5] Oluwasegun Taiwo Ojo; Antonio FernÃ¡ndez Anta; Rosa E. Lillo; Carlo Sguera. Detecting and classifying outliers in big functional data. arXiv, 2019.

[6] Franz M. Rohrhofer; Santanu Saha; Simone Di Cataldo; Bernhard C. Geiger; Wolfgang von der Linden; Lilia Boeri. Importance of feature engineering and database selection in a machine learning model: A case study on carbon crystal structures. arXiv, 2021.

[7] Jeff Heaton. An empirical analysis of feature engineering for predictive modeling. arXiv, 2017.

[8] Emre. Fundamental techniques of feature engineering for machine learning. Medium, 1 Apr. 2019.

[9] Jason Brownlee. Discover feature engineering, how to engineer features and how to get good at it. MachineLearningMastery, 15 Aug. 2020.

[10] Alaa Tharwat, Tarek Gaber, Abdelhameed Ibrahim, and Aboul Ella. Hassanien. Linear discriminant analysis: A detailed tutorial. Ai Communications. 30., pages 169–190, 2017.

[11] Raja Giryes Dor Bank, Noam Koenigstein. Autoencoders. arXiv, 3 Apr 2021.

[12] Adnan Mohsin Abdulazeez Dastan Hussen Maulud1. A review on linear regression comprehensive in machine learning. Journal of Applied Science and Technology Trends Vol. 01, No. 04,, pages 140–147, 2020.

[13] Donald W. Marquardt and Ronald D. Snee. Ridge regression in practice. The American Statistician, 29:1,, pages 3–20, 1975.
[14] Debasish Basak, Srimanta Pal, and Dipak. Patranabis. Support vector regression. Neural Information Processing â€“ Letters and Reviews. 11., 2007.
[15] Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-validation). Encyclopedia of Database Systems, pages 532â€“–538, 2009.
[16] Prajwal Kudale. K-fold cross validation. Analytics Vidhya, 23 Dec. 2020.
[17] Wei-Yin. Loh. Classification and regression trees. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 1., pages 14–23, 2011.
[18] Jason. Brownlee. Regression metrics for machine learning. MachineLearningMastery, 15 Feb. 2021.
[19] Marco Cavallo; Mishal Dholakia; Matous Havlena; Kenneth Ocheltree; Mark Podlaseck. Immersive insights: A hybrid analytics system for collaborative exploratory data analysis. arXiv, 2019.
[20] Exploratory data analysis. Wikipedia, 16 Dec. 2022.
[21] Wikipedia. House price index. wikipedia.org, 30 Dec. 2022.
[22] Mu J; Wu F; Zhang A. Housing value forecasting based on machine learning methods. Abstract and Applied Analysis, page 1â€“7, 2014.
[23] Sidharth Mishra and Uttam Sarkar. Principal component analysis. International Journal of Livestock Research. 1., 2017.
[24] T. D Phan. Housing price prediction using machine learning algorithms: The case of melbourne city, australia. 2018 International Conference on Machine Learning and Data Engineering ICMLDE, 2018.
[25] D Zanzalari. House price index (hpi), the balance. thebalancemoney, 2022.
[26] Abigail Bola Adetunji, Oluwatobi Noah Akande, Funmilola Alaba Ajala, Ololade Oyewo, Yetunde Faith Akande, and Gbenle Oluwadara. House price prediction using random forest machine learning technique. Procedia Computer Science, 199:806–813, 2022.
[27] Giorgio Canarella, Stephen Miller, and Stephen Pollard. Unit roots and structural change: an application to us house price indices. Urban Studies, 49(4):757–776, 2012.
[28] Kaggle. House prices - advanced regression techniques. Kaggle.
[29] Ying Yu, Shuangbao Song, Tianle Zhou, Hanaki Yachi, and Shangce Gao. Forecasting house price index of china using dendritic neuron model. In 2016 International Conference on Progress in Informatics and Computing (PIC), pages 37–41. IEEE, 2016.