Vasquez, C., & Chellamuthu, V. (2021). House Price Prediction With Statistical Analysis in Support Vector Machine Learning for Regression Estimation. *Curiosity: Interdisciplinary Journal of Research and Innovation*, *2*.
https://doi.org/10.36898/001c.22311

STEM

House Price Prediction With Statistical Analysis in Support Vector Machine Learning for Regression Estimation

Cesar Vasquez[1], Vinodh Chellamuthu, PhD[1] [a]

[1] Mathematics, Dixie State University

Keywords: machine learning, statistics

https://doi.org/10.36898/001c.22311

Curiosity: Interdisciplinary Journal of Research and Innovation

Vol. 2, 2021

Support vector learning machines are supervised models that analyze data for classification analysis. However, with modifications these models can be extended for regression estimation. In this paper, we use Dean De Cock's Iowa house price dataset, composed of quantitative and qualitative variables, to build a predictive house pricing model. We employ a systematic approach to preparing the data and processing the features using statistical analysis. To reduce model over-fitting, we use stratified k-fold cross-validation. We also use various metrics for measuring model performance including r-squared (R^2) and mean absolute error (MAE). Moreover, we explore various kernels for hyper-dimensional feature mapping to account for data that is not linearly separable. We compare the kernel performance and find that there is a trade-off between model complexity and performance. Furthermore, there also exists a trade-off between maximum accuracy achievable and general model performance.

Introduction

In this paper, we explore the application of machine learning in predicting house prices. Specifically, we will be constructing a support vector machine for regression estimation. We attempt to improve model performance by implementing kernels for hyper-dimensional feature mapping We also go over statistical techniques commonly used in processing data for regression estimation. We find that there exists a trade-off between model complexity and performance. Should the model be either too simple or complex, we risk under-fitting or over-fitting the model. In between optimal model complexities, we find a trade-off between level of accuracy and general model performance. For a broader study of learning machines in house price prediction see Babb (2019).

The learning problem can be said to have started with Rosenblatt's Perceptron (1962).[1] Discoveries in learning theory, such as back-propagation (Bryson et al., 1963) for solving vector coefficients simultaneously (Le Cun, 1986) and regularization techniques for the overfitting phenomenon, have advanced understanding of the learning problem significantly. Since

[a] Dr. Vinodh Chellamuthu joined Dixie State University in 2015 after completing his graduate work at University of Louisiana at Lafayette. His main research interests lie in Mathematical Modeling. A passionate proponent of undergraduate research, Dr. Chellamuthu regularly engages undergraduates in problems related to his research program, and he fosters a commitment in his students to disseminate their work through publications and national/regional presentations. His work with students has led to over 50 student presentations at various conferences. He also has a passion to recruit and train students to participate in the international mathematics competition Mathematical Contest in Modeling.

1 Now known, thanks to the artificial intelligence community, as a neural network.

Rosenblatt's digit recognition problem, the importance of statistical analysis in the learning problem has been emphasized greatly. The learning problem is that of choosing a function $f(x, \alpha)$, where α is a set of parameters, based on a set of independent and identically distributed observations that returns a value $\hat{y}$ most approximate to y. Hence, a supervised machine-learning model requires that in a set of observations (x, y), y is known a priori for training the model. A risk function $R(\alpha) = \int L(y, f(x, \alpha))dF(x, y)$, where $F(x, y)$ is the joint probability distribution function (pdf) of the observations, is used to measure the discrepancy $L(y, f(x, \alpha))$ between a learning machine's output $f(x, \alpha)$ to a given variable x and the observed value y. In this paper, we will consider the specific learning problem of regression estimation. In regression estimation, we wish to minimize the risk function $R(\alpha)$ with a loss function $L(y, f(x, \alpha)) = (y - f(x, \alpha))^2$.[2] Of course, in order to minimize the risk function $R(\alpha)$ we must know the joint pdf $F(x, y)$. This is often not the case in real-world applications. Accordingly, the empirical risk minimization principle is applied to substitute the risk function with $R_e(\alpha) = \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i, \alpha))^2$.

In 1992, Boser et al. proposed a training algorithm for optimal marginal classifiers. In 1995, Vapnik expanded this algorithm for regression estimation problems. Support vector machines (SVMs) are supervised machine-learning models that analyze data for classification problems. Such models can be modified for regression analysis with the use of logistic regression. SVMs have a complexity dependent on the number of support vectors rather than on the dimensions of the feature space (Vapnik, 1995); hence, SVMs are memory-efficient. Training data need not be linearly separable (though, this is an assumption made in the construction of optimal separating hyperplanes). We begin our inquiry by constructing a support vector machine without a hyper-dimensional mapping kernel. We will create additional models by implementing a polynomial kernel and the radial basis function (RBF). After running our dataset with each model, we will compare and contrast the results of their house price predictions.

In the following section, we introduce Vapnik and Chervonenkis' (1974; cited in Devroye et al., 1996) optimal separating hyperplanes and the support vector learning machine used for constructing such hyperplanes. In Section 3, we introduce the dataset used and techniques in preparing the dataset for optimal model performance. Section 4 will go over techniques for creating and processing features to further improve model performance. We display and discuss the results in section 5. Finally, we make some concluding remarks in section 6.

2 The least-squares method.

Model Formulation

We begin by defining our training data of m samples $(\mathbf{x}_i, y_i)$, $i = 1, 2, \ldots, m$, where $\mathbf{x} \in R^n$ and $y \in R$; in other words, the variable vector consists of n features and the output is a real value. A linear separating hyperplane has the form $\langle \mathbf{w}, \mathbf{x} \rangle - b = 0$, where the weight vector $\mathbf{w}$ and the threshold scalar b determine the position of the hyperplane. The optimal hyperplane is found by separating the variable vector $\mathbf{x}_i$ into two different classes $y \in \{-1, 1\}$ with the smallest norm $||w||$. This is done by minimizing the function

$$\phi(w) = \frac{1}{2}(w \cdot w)$$

subject to $y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1_i$ for $i = 1, 2, \ldots, m$. The solution is found at the saddle point of the Lagrange function

$$G(w, b, \alpha) = \frac{1}{2}(w \cdot w) - \sum_{i=1}^{m} \alpha_i [|(\mathbf{x}_i \cdot \mathbf{w} - b)| y_i - 1]$$

where α_i are the Lagrange multipliers. In the case that the data is not perfectly separable, we have penalties $\xi_i = max[0, 1 - y_i(\langle \mathbf{w}, \mathbf{x} \rangle + b)]$. Here, to find the optimal hyperplane, we minimize

$$\phi(w, \xi) = \frac{1}{2} + C \sum_{i=1}^{m} \xi_i$$

where C is a given regularization parameter, subject to $y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1_i - \xi_i$ and $\xi_i \geq 0$ for $i = 1, 2, \ldots, m$.

To account for non-linearly separable data, kernel tricks,[3] originally proposed by Aizerman et al. (1964), are applied to map the vector $\mathbf{x}$ to a high-dimensional feature space Z (Boser et al., 1992). A kernel function simply replaces the inner products $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ with convoluted inner products $k(\mathbf{z}_i, \mathbf{z}_j)$. The result is a hyperplane in high-dimensional space that may or may not be linear in the original variable space. Consequently, the algorithms remain fairly similar and support vector machines are thus versatile and effective in high-dimensional spaces. We will create additional models by applying the inhomogeneous polynomial kernel function of degree d, defined $k(\mathbf{z}_i, \mathbf{z}_j) = (\langle \mathbf{z}_i, \mathbf{z}_j \rangle + r)^d$, where r is a constant added to avoid problems with the inner product equalling 0 (specifically, we will consider degrees d=2,3 and 4). In addition, we will also construct a model with the (Gaussian) radial basis kernel function (RBF), defined $k(\mathbf{z}_i, \mathbf{z}_j) = exp\{-\gamma ||\mathbf{z}_i - \mathbf{z}_j||^2\}$.

3 A kernel function can be any symmetric function satisfying the Hilbert-Schmidt theory (1953). For example, a linear support vector machine for non-separable data may have a kernel function defined as $k(\mathbf{z}_i, \mathbf{z}_j) = \langle \mathbf{z}_i, \mathbf{z}_j \rangle$, where $\mathbf{z}$ is the mapped feature vector in high-dimension of the variable vector $\mathbf{x}$.

The risk function for a logistic regression is given by $R_e(t) = ln(1 - e^{-yt})$ where $t = \langle \mathbf{w}, \mathbf{x} \rangle + b$. The loss function is given by $L(t) = c_0 max(0, 1 - t)$ for some constant c_0. With penalty $\xi_i = max[0, 1 - y_i(\langle \mathbf{w}, \mathbf{x} \rangle + b)]$ where $y_i(\langle \mathbf{w}, \mathbf{x} \rangle + b) \geq 1 - \xi_i \quad \text{and} \quad \xi_i \geq 0$, we have the risk function $R_e(w, b) = \frac{1}{2} + C \sum_{i=1}^{m} \xi_i$ for a support vector machine for regression estimation (otherwise known as support vector regression or SVR).

Data

We shall now explore the Ames, Iowa Housing Data by Dean DeCock.[4] This dataset has a total of 2,919 entries with 80 variable columns that focus on the quantity or quality of different physical attributes of the respective property. Thorough descriptions of the variables can be found in Table 5 and their summary statistics in Table 8 of the Appendix. We perform data cleaning methods such as removing duplicates and outliers as well as filling in missing data values. Duplicates are removed as this causes the duplicated properties (i.e., the properties sold multiple times within the 2006-2010 period) to hold more weight when training a model. Data outliers have high potential to misinform and harm a model's training process and hence need to be removed as well. Failure to remove outliers can lead to less accurate models and, consequently, uninformative results.

To detect outliers, we calculate inner and outer fences for each numeric variable column and differentiate between suspected and certain outliers by using inner and outer fences (Equations (1) to (4); Hogg et al., n.d.).

$$InnerLowerFence = Q1 - (1.5 * IQR) \quad (1)$$

$$InnerUpperFence = Q3 + (1.5 * IQR) \quad (2)$$

$$OuterLowerFence = Q1 - (3 * IQR) \quad (3)$$

$$OuterUpperFence = Q3 + (3 * IQR) \quad (4)$$

Here, $Q1$ is the column's first quartile; $Q3$ is the third quartile; and IQR is the column's interquartile range, given by $Q3 - Q1$. Suspected outliers are the datapoints that lie between the inner and outer fences whereas certain outliers are those that exceed the outer fences. To address outliers in the target column (i.e., the *'Sale Price'* column), we apply a natural log transformation to balance the data and thus make it more suitable for the machine-learning model. Figure 1 shows the histogram for the target column before and after the natural log transformation. Missing values can be handled by looking at the data description provided with the dataset. Table 6 in the Appendix summarizes how the missing variables are handled for each column with missing data.

Categorical data can be classified as being either ordinal or nominal. Ordinal data is defined as data with a natural order (e.g., rankings, order or scaling). As a result, we can transform ordinal variables into ordered numbers. For example, the variable column *'ExterCond'* has 5 possible values: poor, fair,

4 https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data.

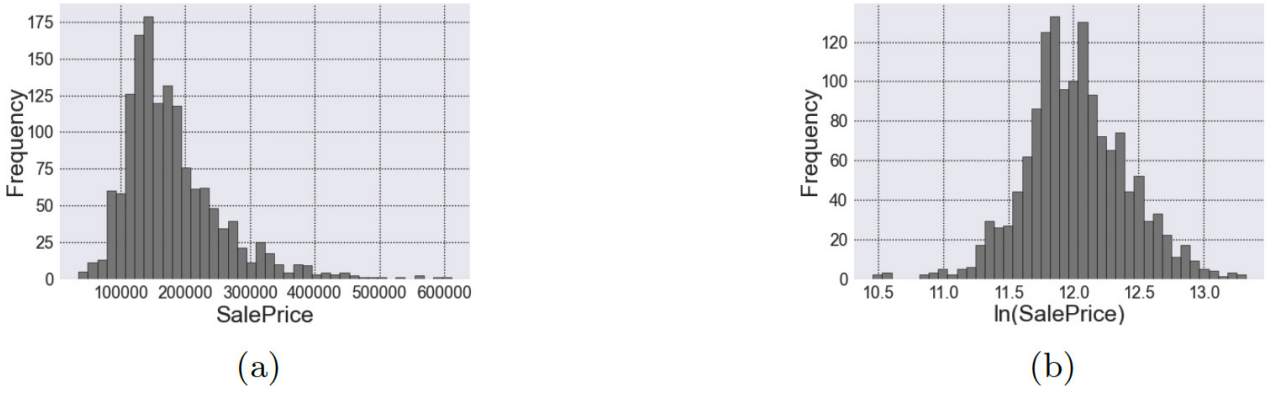


Figure 1: Histograms of the target column *'SalePrice'* before (1a) and after (1b) a natural log transformation.

Table 1: Example of One-Hot encoding for nominal data using the *'LandContour'* variable column. Notice that a new column is created for each possible value in the nominal variable's column.

PropertyID	LandContour
1	Lvl
22	Bnk
54	HLS
59	Low

(a) Before One-Hot encoding.

PropertyID	Lvl	Bnk	HLS	Low
1	1	0	0	0
22	0	1	0	0
54	0	0	0	1
59	0	0	1	0

(b) After One-Hot encoding.

average, good, and excellent. We encode these as follows: poor - 1, fair - 2, average - 3, good - 4, and excellent - 5. Table 7 in the Appendix shows the comprehensive transformations for the 20 ordinal variable columns.

On the other hand, nominal data is defined as data used for naming or labelling purposes. Consequently, nominal data does not have quantitative values or a natural order. Thus, for nominal variables we create "dummy" columns via one-hot encoding as follows: Given a nominal variable column, we create a new *dummy* column for each possible value determined by the column. Then, for a given property, we mark a '1' to the *dummy* column corresponding to the property's variable value, and a '0' for the non-corresponding *dummy* columns. This process can be further understood by examining Table 1.

Feature Processing

Feature engineering plays an important role in a learning machine model's success or failure. Feature engineering is the process of extracting new features from raw data. In our model, we create new features by calculating ages and by creating polynomials using variables with a moderate to strong correlation to *'SalePrice'*. Two new features are constructed by calculating the ages *'GarageAge'* and *'AgeAtSale'* as follows[5]:

5 For clarity, *YrSold* is the year the house was sold; *GarageYrBlt* is the year the garage was built, same as *YearBuilt* (the construction date) if the house does not have a garage; *YearRemodAdd* is the year the house was remodeled, same as the construction date if the house has not been remodeled. Refer to Tables 5 and 6 in the appendix for further details.

$$GarageAge = YrSold - GarageYrBlt \quad (5)$$
$$AgeAtSale = YrSold - YearRemodAdd \quad (6)$$

To be sure, we construct these specific variables assuming that age matters in estimating the price of a house. For example, we assume that all else equal, a 10 year old house will have a different price than a 50 year old house with the same features. The same logic applies for the age of a house's garage.

Now, to determine the features with a moderate to strong correlation to *'SalePrice'*, we calculate Spearman's rank correlation coefficient ρ (Spearman, 1910). Spearman's method is preferable as it is able to recognize strong, linear and nonlinear, monotonic relationships. We shall consider only the features with a correlation coefficient greater than 0.50. Let x represent a given feature column and let y represent the target column *'SalePrice'*. The Spearman rank correlation coefficient is calculated similar to the Pearson correlation coefficient,

$$\rho_{rg_x, rg_y} = \frac{Cov(rg_x, rg_y)}{\sigma_{rg_x}\sigma_{rg_y}}$$

where rg_{x_i} is the rank of x_i and rg_{y_i} is the rank of y_i. Here, $Cov(rg_x, rg_y)$ is the covariance between rg_x and rg_y; σ_{rg_x} and σ_{rg_y} are the standard deviations of rg_x and rg_y, respectively. Table 2 displays the features with a Spearmank rank correlation greater than 0.50. Three new features are then created for each feature[6] in Table 2 using the equations (7), (8), and (9), where x represents the respective feature:

$$x_{squared} = x^2 \quad (7)$$
$$x_{cubed} = x^3 \quad (8)$$
$$x_{sqrt} = \sqrt{x} \quad (9)$$

Since raw data values often vary widely (e.g., square footage vs number of rooms), we also perform feature normalization and standardization.[7] First, we normalize strongly skewed numerical features using the one-parameter Box-Cox transformation (Box & Cox, 1964). In order to determine which features are strongly skewed, we calculate the Fisher-Pearson coefficient of skewness (Equations (10) and (11); Pearson, 1894).

$$M_j = \frac{1}{m}\sum_{i=1}^{m}(x_i - \bar{x})^j \quad (10)$$

$$g_1 = \frac{M_3}{M_2^{\frac{3}{2}}} = \frac{\frac{1}{m}\sum_{i=1}^{m}(x_i - \bar{x})^3}{\left[\frac{1}{m}\sum_{i=1}^{m}(x_i - \bar{x})^2\right]^{\frac{3}{2}}} \quad (11)$$

A perfectly symmetric distribution will have a g_1 coefficient of 0, or close to 0 if slightly asymmetric; a distribution that is skewed left will have a negative g_1 coefficient; and a distribution that is skewed right will have a positive g_1 coefficient. The larger the magnitude of g_1, the more skewed the data is. For

6 For further details about the features, refer to Table 5 in the Appendix.

7 Normalization is the process of scaling variables to have values between 0 and 1. Standardization, on the other hand, is the process of scaling variables to have mean 0 and unit variance.

Table 2: Features with a Spearman rank correlation greater than 0.50.

Feature	Spearman's Rank Correlation ρ
OverallQual	0.810
GrLivArea	0.727
GarageCars	0.692
ExterQual	0.685
KitchenQual	0.672
YearBuilt	0.658
GarageArea	0.650
FullBath	0.634
GarageAge	0.620
TotalBsmtSF	0.597
AgeAtSale	0.580
1stFlrSF	0.567
FireplaceQu	0.531
TotRmsAbvGrd	0.530
Fireplaces	0.512

our purposes, we will define strongly skewed features as having a g_1 coefficient with an absolute value that is greater than 1. Table 9 in the Appendix displays such features. The one-parameter Box-Cox transformation is then performed as follows where, again, x denotes a given feature:

$$x_i^{(\lambda)} = \begin{cases} \frac{x_i^{\lambda}-1}{\lambda} & \text{if } \lambda \neq 0, \\ ln(x_i) & \text{if } \lambda = 0. \end{cases} \tag{12}$$

The parameter λ is chosen as the optimal parameter to approximate a normal distribution for x. Finally, we standardize our features simply as

$$x_{new} = \frac{x_{old} - \bar{x}}{s_x}$$

where $\bar{x}$ and s_X are the feature's sample mean and sample standard deviation, respectively.

Results & Discussion

To assess the performance of the various kernels, we use the stratified k-fold cross validation technique. Additionally, we use the R-squared score (R^2) and the mean absolute error (MAE) to compare the results. These metrics are found by Equations (13) and (14) where y_i is the observed *'SalePrice'*, $\hat{y}_i$ is the predicted price, and $\bar{y}$ is the average observed *'SalePrice'*.

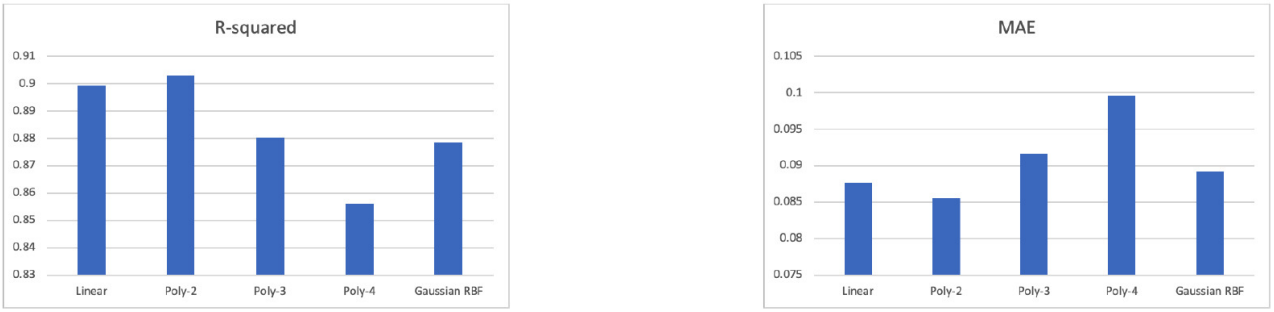
$$R^2 = 1 - \frac{\sum_{i=1}^{m}(\hat{y}_i - \bar{y}_i)^2}{\sum_{i=1}^{m}(y_i - \bar{y}_i)^2} \tag{13}$$

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|\hat{y}_i - \bar{y}_i| \tag{14}$$

Figures 3 - 7 display the results for each kernel. In each figure, we will show the observed vs. residuals using log-transformed values (a), the observed vs. residuals using the reverted values (i.e., we undo the log transformation) (b), the distribution of residuals relative to the observed values (or

Table 3: Number of relative residuals between specified bounds.

	Linear	Poly-2	Poly-3	Poly-4	RBF
$\leq 1\%$	130	120	129	134	142
$\leq 10\%$	1144	1226	1225	1197	1234
Between 10% and 100%	294	213	214	242	205
$\geq 100\%$	1	0	0	0	0



(a) R-squared scores. (b) Mean Absolute Errors.

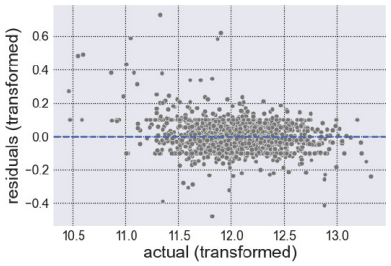
Figure 2: Resulting R^2 (2a) and MAE (2b) for the 5 kernels.

$residual/observed$) (c), and finally the actual (i.e., observed) vs. predicted values (d). Also, figures (a), (b) and (d) have a dashed line where $actual = predicted$ so that there is no residual. Moreover, figures (b) have 2 additional lines: one solid and one dotted. The solid line represents a residual value equal to the actual value; therefore anything exceeding the solid lines represents an error larger than the actual value of the house. The dotted line on the other hand, represents a residual value equal to one-tenth of the actual value; anything between the dotted lines represents an error of less than one-tenth of the actual value. Accordingly, anything between the dotted and solid lines would represent an error between 10% and 100% of the actual value. Obviously, a good model would have minimal errors that exceed the dotted line, perhaps no errors beyond the solid line, and maximal errors within the dotted lines. In other words, a good model should have errors/residuals that are small in magnitude relative to the actual value being predicted. (One should be wary of having a large proportion of errors with 1% of the actual value; while this may be a good thing, it can be a sign of an overfitted model). Table 3 summarizes the count that lay within each boundary. Moving on, Figure 2 displays the resulting R^2 and MAE of the kernels. Note that a higher R^2 and a lower MAE generally indicate a better model performance. Finally, Table 4 displays the summary statistics for the residuals of each kernel. It is important to note that Table 4 is constructed using the absolute value of the residuals.

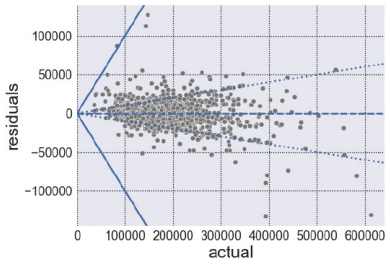
We begin by examining the results of the linear kernel (Figure 3). The linear kernel has an R^2 of 0.89920 (2nd highest) and a MAE of 0.08768 (2nd lowest), making it the second best performer in each metric. Figures 3a and 3b suggest inconsistency in the kernel's residuals. In fact, the linear kernel's residuals have the highest variation (std. dev. = \$12,022.924) of all the kernels according to

Table 4: Summary statistics for the residuals of each kernel.

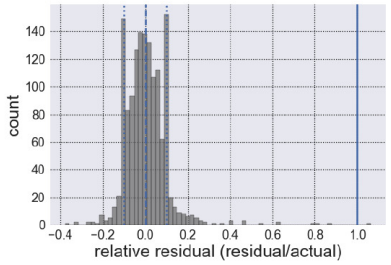
	Linear	Poly-2	Poly-3	Poly-4	RBF
Mean	11844.865	11170.330	10986.212	11429.242	11040.674
Std. Dev.	12022.924	9496.967	9387.477	9553.066	9335.920
Min.	5.605	14.470	4.975	3.054	3.174
25%	4287.944	4533.757	4266.889	4454.987	4429.951
50%	9247.746	9521.258	9107.494	9656.218	9358.401
75%	15270.623	14997.735	15038.317	15397.377	15048.800
95%	32177.842	26952.717	27650.655	29242.069	27515.704
99%	52365.139	41263.012	39854.789	40036.26	40032.341
Max.	132108.955	120667.670	107552.218	119033.523	116578.736



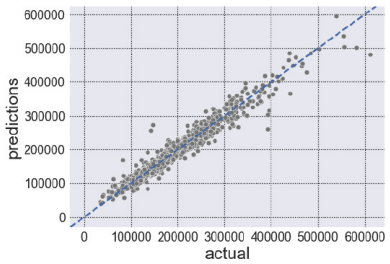
(a) Actual values vs. residuals, log-transformed.



(b) Actual values vs. residuals, non-transformed.



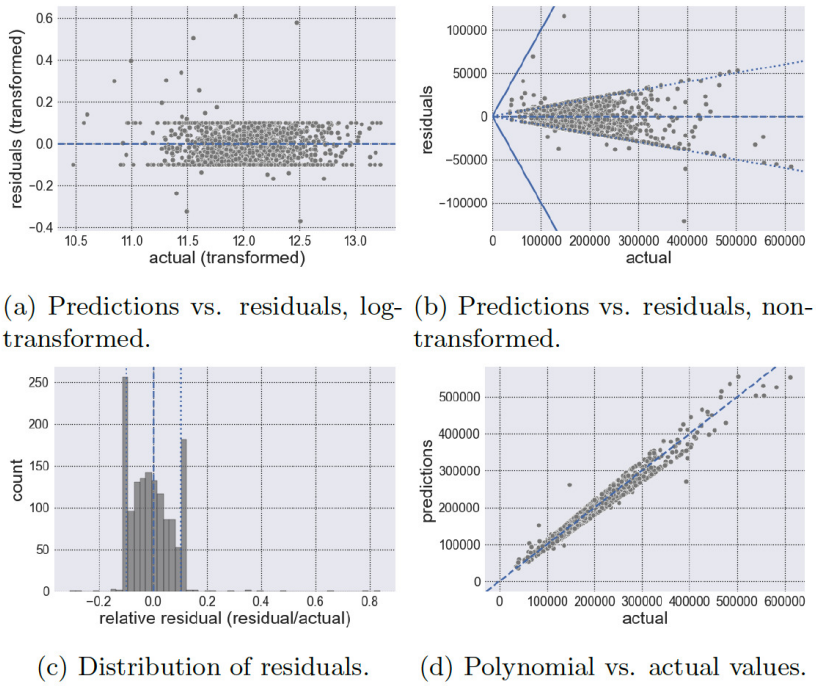
(c) Distribution of residuals relative to actual values.



(d) Actual vs. predicted values.

Figure 3: Results for SVR with a linear kernel.

Table 4. In contrast to the other kernels who have a low relative residual at lower house prices, the linear kernel shows a large relative residual even at these lower prices. Also noteworthy is that the linear kernel has the least predictions within 10% of the actual value (1144 or 79.50%); and it is the only kernel with a prediction larger in magnitude than the actual value (1 or 0.07%). In addition, the linear kernel has by far the most predictions between 10% and 100% of the actual value (294 or 20.43%). In other words, the linear kernel appears to have the largest relative errors in comparison to the other kernels. Looking at the residual summary statistics in Table 4 we see that the linear kernel has the largest residual mean ($11,844.865) as well as the largest maximum residual ($132,108.955). Due to the seemingly contradictory results, we suspect that the linear kernel suffers from model simplicity.



(a) Predictions vs. residuals, log-transformed.

(b) Predictions vs. residuals, non-transformed.

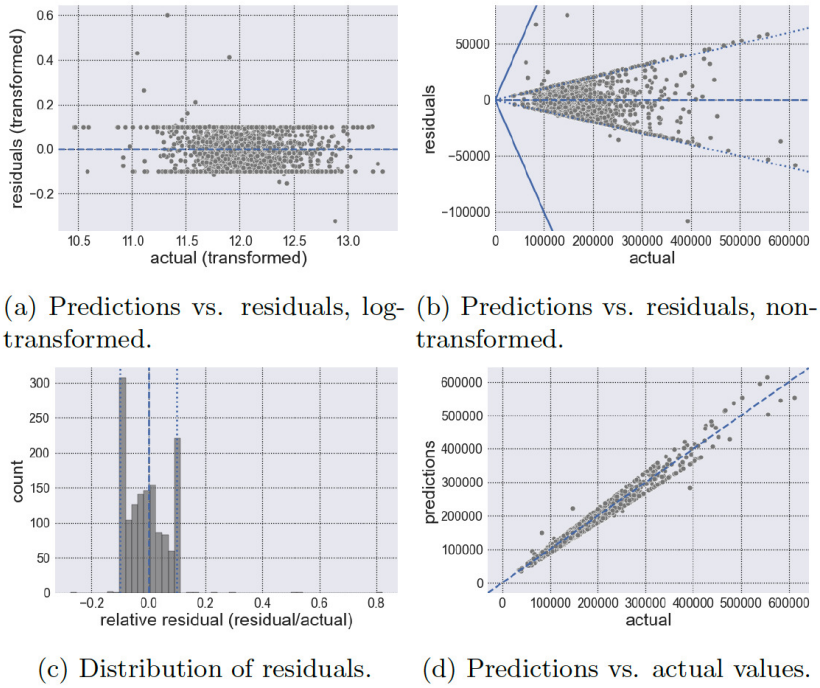
(c) Distribution of residuals.

(d) Polynomial vs. actual values.

Figure 4: Results for SVR with a polynomial kernel of degree $d = 2$ (poly-2).

Next, we look at the polynomial kernel of degree $d = 2$ (Figure 4). The polynomial of degree $d = 2$ (poly-2) kernel performed the best in each metric, having an R^2 of 0.90293 and a MAE of 0.08577. Examining Table 3, we find that the poly-2 kernel has the lowest number of predictions within 1% of the actual value. However, we also find that 1226 (or 85.20%) of predictions are within 10% of the actual value, a significant improvement from the linear kernel. Although the poly-2 kernel performs well overall, it also shows an inability in achieving extreme accuracy. Table 3 reveals that the poly-2 kernel had the lowest number of predictions within 1% of the actual value (120 or 8.34%); Table 4 reveals that the poly-2 kernel has the highest minimum error (14.470). Still, though it fails in achieving extreme accuracy in comparison to the other kernels, the poly-2 kernel displays very strong prediction capabilities within a relatively small bound.

Moving forward, we examine the polynomial of degree $d = 3$ kernel (Figure 5). The polynomial of degree $d = 3$ (poly-3) kernel has an R^2 of 0.88023 (3rd highest) and a MAE of 0.09166 (4th lowest), making it a close competitor with the gaussian RBF kernel. The residuals of the kernel appear to behave similar to the residuals from the poly-2 kernel; in fact, the poly-3 kernel has a similar count of residual predictions within 10% (1225 or 85.13%) and between 10% and 100% (213 or 14.80%) of the actual value. However, the poly-3 kernel does appear to be capable of achieving high accuracy; with a count of 129 predictions (8.96%) being within 1% of the actual value; the lowest maximum error ($107,552.218); and a minimum residual ($4.975) significantly lower than that of the poly-2 kernel, though comparable to the other kernels.



(a) Predictions vs. residuals, log-transformed.

(b) Predictions vs. residuals, non-transformed.

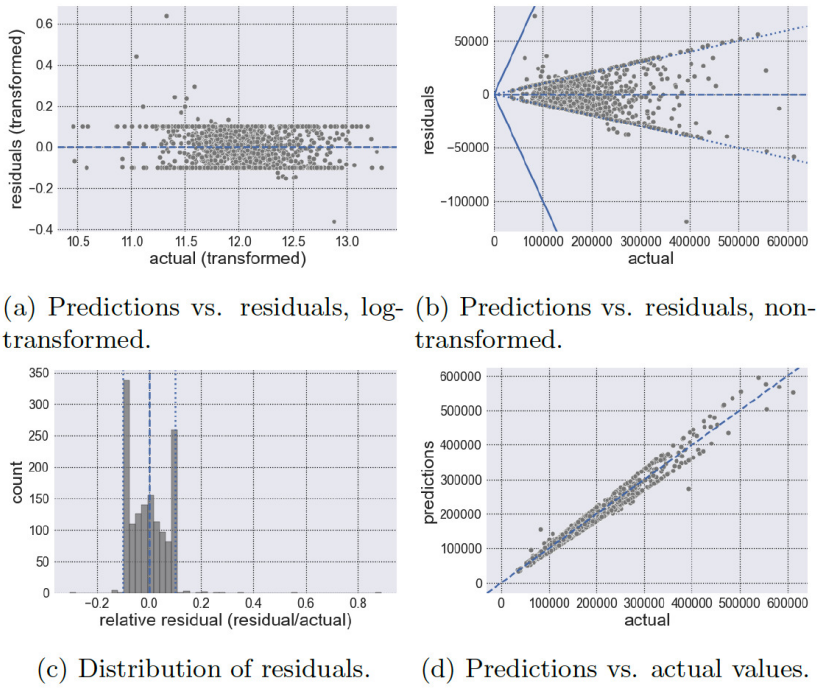
(c) Distribution of residuals.

(d) Predictions vs. actual values.

Figure 5: Results for SVR with a polynomial kernel of degree $d = 3$ (poly-3).

Now we examine the polynomial kernel of degree $d = 4$ (Figure 6). This kernel has an R^2 of 0.85604 (the lowest score) and a MAE of 0.09957 (the highest error), making it the poorest performing kernel according to these metrics. This leads us to suspect that the polynomial of degree $d = 4$ (poly-4) kernel is trading off performance for complexity. Indeed, the poly-4 kernel achieves the second highest count of predictions within 1% of the actual value (134 or 9.31%); though it also has the second highest count of predictions between 10% and 100% (242 or 16.82%), behind the linear kernel. Another contradiction is that the poly-4 kernel has the lowest minimum error ($3.054) but also the second highest residual mean ($11,429.242) and the second highest maximum error ($119,033.523), behind the linear kernel. We thus suspect that the poly-4 kernel is suffers from model complexity.

Lastly, we examine the gaussian RBF kernel (Figure 7) which has an R^2 of 0.87840 (4th highest) and a MAE of 0.08920 (3rd lowest), resulting in scores comparable to that of the poly-3 kernel. Like the poly-3 and poly-4 kernels, the gaussian RBF kernel is capable of achieving high accuracy, having a count of 142 (9.87%) of predictions with 1% of the actual value. Further, it has the lowest count (205 or 14.25%) of predictions between 10% and 100% of the actual value. Table 4 shows that the kernel has a performance between the poly-3 and poly-4 kernels. Namely, looking at the residual mean ($11,040.674), minimum ($3.174) and maximum ($116,578.736) residuals, the gaussian RBF kernel has metrics higher than the poly-3 kernel though lower than the poly-4 kernel. It is doubtful whether or not this kernel is also suffering from model complexity.



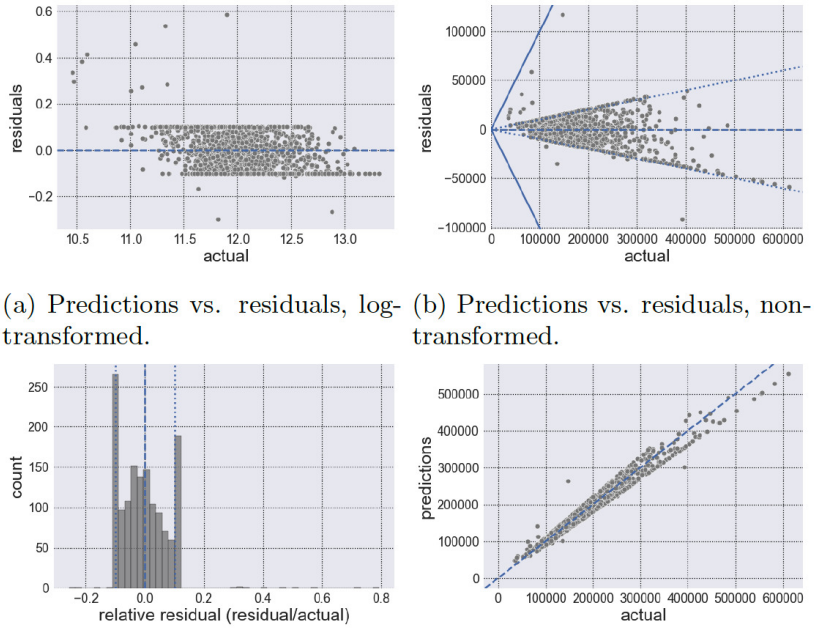
(a) Predictions vs. residuals, log-transformed.

(b) Predictions vs. residuals, non-transformed.

(c) Distribution of residuals.

(d) Predictions vs. actual values.

Figure 6: Results for SVR with a polynomial kernel and degree $d = 4$ (poly-4).



(a) Predictions vs. residuals, log-transformed.

(b) Predictions vs. residuals, non-transformed.

(c) Distribution of residuals.

(d) Predictions vs. actual values.

Figure 7: Results for SVR with an Gaussian RBF kernel.

In summary, each kernel achieved favorable results overall. We suspect the linear kernel of under performing due to model simplicity; and the poly-4 kernel (and perhaps the gaussian RBF kernel) of poor generalization due to model complexity. The result being that the poly-2 and poly-3 kernels would be preferable for future house price prediction models. The poly-2 kernel does a good job of predicting house prices overall within a certain boundary, though it fails in achieving a high level of accuracy in comparison to the poly-3 and

the more complex models. This could be a generalization issue and should be examined further. On the other hand, the poly-3 kernel does achieve a high level of accuracy but tends to perform slightly worse than the poly-2 kernel overall. Indeed, when comparing the simpler and more complex models, there does appear to exist a trade off between model complexity and performance. Examining figures (a) for each kernel, we see that the log-transformation succeeded in penalizing errors uniformly. That is, the log-transformation's goal was to maintain relatively equal penalties for both small and large house prices; otherwise, the penalties from large houses would dominate model fitting (since they would tend to be larger in magnitude).

Conclusion

In this paper, we employed a supervised machine-learning model for house price prediction, viz., the SVM for regression estimation (or support vector regression, SVR) model. We also explored 5 different hyper-dimensional mapping kernels: the linear kernel; the polynomial kernel with degrees $d = 2, 3, 4$; and the gaussian radial basis function (RBF) kernel. We performed statistical techniques (data cleaning, feature engineering, normalization and standardization) to prepare our data for the SVR model. To reduce over-fitting, we utilized the stratified k-fold cross-validation technique in training and testing our model. Model performance was evaluated using the r-squared score (R^2) and the mean absolute error (MAE). Our findings show that the polynomial kernels of degree d=2 and 3 (poly-2 and poly-3) performed the best overall. This may be due to a balance of simplicity and complexity which results in better generalization. Even with limited data on more expensive houses, our models appear to perform relatively well at predicting such house prices. Still, we would like to collect more data at those prices to test for performance improvement.

When constructing machine-learning models, data preparation is just as important as the amount of data available. Even more, the application of statistical analysis in data processing are of paramount importance. In order to improve our models, we might consider exploring additional or even different statistical techniques. As mentioned, we may also want to collect additional data on more expensive houses. (With machine-learning models, however, more data does not always improve performance.) To further test the generalization of our model, we would like to apply it to other house price datasets.

Submitted: December 30, 2020 MST, Accepted: April 12, 2021 MST

References

Aizerman, M. A., Braverman, E. M., & Rozonoer, L. I. (1964). *Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning, Automation and Remote Control* (pp. 821–837). Springer.

Babb, O. (2019). A Comparison of Machine Learning Approaches to Housing Value Estimation. *SIAM Undergraduate Research Online*, *12*. https://doi.org/10.1137/18s017296

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A Training Algorithm for Optimal Margin Classifier. *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*. https://doi.org/10.1145/130385.130401

Box, G. E. P., & Cox, D. R. (1964). An Analysis of Transformations. *Journal of the Royal Statistical Society: Series B (Methodological)*, *26*(2), 211–252. https://doi.org/10.1111/j.2517-6161.1964.tb00553.x

Bryson, A. E., Jr., Denham, W. F., & Dreyfus, S. E. (1963). Optimal Programming Problems with Inequality Constraints, I: Necessary Conditions for Extremal Solutions. *AIAA Journal*, *1*(11), 2544–2550. https://doi.org/10.2514/3.2107

Devroye, L., Györfi, L., & Lugosi, G. (1996). Vapnik-Chervonenkis Theory. In *A Probabilistic Theory of Pattern Recognition* (Vol. 31, pp. 187–213). Springer. https://doi.org/10.1007/978-1-4612-0711-5_12

Hogg, R. V., Tanis, E. A., & Zimmerman, D. L. (n.d.). Point Estimation: Exploratory Data Analysis. In *Probability and Statistical Inference* (9th ed., pp. 267–273). Pearson.

Le Cun, Y. (1986). Learning Process in an Asymmetric Threshold Network. In *Disordered Systems and Biological Organization* (pp. 233–240). Springer-Verlag. https://doi.org/10.1007/978-3-642-82657-3_24

Pearson, K. (1894). Mathematical Contributions to the Theory of Evolution. II. Skew Variation in Homogeneous Material. *Proceedings of the Royal Society of London*, *57*(340–346), 257–260. https://doi.org/10.1098/rspl.1894.0147

Renardy, M., & Rogers, R. C. (2004). Eigenfunction Expansions. In *An Introduction to Partial Differential Equations* (2nd ed., pp. 300–303). Springer.

Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons And the Theory of Brain Mechanisms*. Spartan Books. https://doi.org/10.21236/ad0256582

Spearman, C. (1910). Correlation Calculated From Faulty Data. *British Journal of Psychology*, *3*(3), 271–295. https://doi.org/10.1111/j.2044-8295.1910.tb00206.x

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory* (2nd ed.). Springer.

Appendices

Table 5: Variable descriptions for the Ames, Iowa housing dataset.

Variable Type	Variable	Description
Numeric - Continuous	LotFrontage	Linear feet of street connected to property
	LotArea	Lot size in square feet.
	MasVnrArea	Masonry veneer are in square feet.
	BsmtFinSF1	Type 1 finished square feet, see'BsmtFinType1'.
	BstmFinSF2	Type 2 finished square feet, see 'BsmtFinType2'.
	BsmtUnfSF	Unfinished square feet of basement area.
	TotalBsmtSF	Total square feet of basement area. $(BsmtFinSF1 + BsmtFinSF2 + BsmtUnfSF)$
	1stFlrSF	First floor square feet.
	2ndFlrSF	Second floor square feet.
	LowQualFinSF	Low quality finished square feet (all floors).
	GrLivArea	Above grade (ground) living area square feet. $(1stFlrSF + 2ndFlrSF + LowQualFinSF)$
	GarageArea	Size of garage in square feet.
	WoodDeckSF	Wood deck area in square feet.
	OpenPorchSF	Open porch area in square feet.
	EnclosedPorch	Enclosed porch area in square feet.
	3SsnPorch	Three season porch area in square feet.
	ScreenPorch	Screen porch area in square feet.
	PoolArea	Pool area in square feet.
	MiscVal	Dollar value of miscellaneous feature(s).
	SalePrice (target)	Sale price of the property.
Numeric - Discrete	OverallQual	Rates the overall material and finish of the house.
	OverallCond	Rates the overall condition of the house.
	YearBuilt	Original construction date.
	YearRemodAdd	Remodel date (same as construction date if no remodeling or additions).
	BsmtFullBath	Basement full bathrooms.
	BsmtHalfBath	Basement half bathrooms.
	FullBath	Full bathrooms above grade.
	HalfBath	Half bathrooms above grade.
	BedroomAbvGr	Bedrooms above grade (does not include basement bedrooms).
	KitchenAbvGr	Kitchens above grade.
	TotRmsAbvGrd	Total rooms above grade (does not include bathrooms).
	Fireplaces	Number of fireplaces.
	GarageYrBlt	Year garage was built.
	GarageCars	Size of garage in car capacity.
	MoSold	Month sold (MM).
	YrSold	Year sold (YYYY).
Categorical - Nominal	MSSubClass	Identifies the type of dwelling involved in the sale.
	MSZoning	Identifies the general zoning classification of the sale.
	LandContour	Flatness of the property.
	LotConfig	Lot configuration.
	Neighborhood	Physical locations within Ames city limits.
	Condition1	Proximity to various conditions.

Variable Type	Variable	Description
	Condition2	Proximity to various conditions (if more than one is present).
	BldgType	Type of dwelling.
	HouseStyle	Style of dwelling.
	RoofStyle	Type of roof.
	RoofMatl	Roof material.
	Exterior1st	Exterior covering on house.
	Exterior2nd	Exterior covering on house (if more than one material).
	MasVnrType	Masonry veneer type.
	Foundation	Type of foundation.
	Heating	Type of heating.
	CentralAir	Central air conditioning.
	Electrical	Electrical system.
	Functional	Home functionality.
	GarageType	Garage location.
	GarageFinish	Interior finish of the garage.
	Fence	Fence quality.
	MiscFeature	Miscellaneous feature not covered in other categories.
	SaleType	Type of sale.
	SaleCondition	Condition of sale.
Categorical - Ordinal	Street	Type of road access to property.
	Alley	Type of alley access to property.
	LotShape	General shape of property.
	Utilities	Type of utilities available.
	LandSlope	Slope of the property.
	ExterQual	Evaluates the quality of the material on the exterior.
	ExterCond	Evaluates the present condition of the material on the exterior.
	BsmtQual	Evaluates the height of the basement.
	BsmtCond	Evaluates the general condition of the basement.
	BsmtExposure	Refers to walkout or garden level walls.
	BsmtFinType1	Rating of basement finished area.
	BsmtFinType2	Rating of basement finished area (if multiple types).
	HeatingQC	Heating quality and condition.
	KitchenQual	Kitchen quality.
	FireplaceQu	Fireplace quality.
	GarageQual	Garage quality.
	GarageCond	Garage condition.
	PavedDrive	Paved driveway.
	PoolQC	Pool quality and condition.

Table 6: Columns with missing data and how they are handled.

Variable	Missing Value Count	Method
LotFrontage	252	Median 'LotFrontage' grouped by 'Neighborhood'.
Alley	1349	No alley accesses.
MasVnrType	8	None.
MasVnrArea	8	0.
BsmtQual	37	No basement.
BsmtCond	37	No basement.
BsmtExposure	38	No basement.
BsmtFinType1	37	No basement.
BsmtFinType2	38	No basement.
Electrical	1	Sbrkr (Standard Circuit Breakers & Romex).
FireplaceQu	688	No fireplace.
GarageType	81	No garage.
GarageYrBlt	81	YearBuilt.
GarageFinish	81	No garage.
GarageQual	81	No garage.
GarageCond	81	No garage.
PoolQC	1435	No pool.
Fence	1161	No fence.
MiscFeature	1390	None.

Table 7: Ordinal variable transformations.

Ordinal Variable	Transformation
Street	Grvl - 1
	Pave - 2
Alley	None - 0
	Grvl - 1
	Pave - 2
LotShape	IR3 - 1
	IR2 - 2
	IR1 - 3
	Reg - 4
Utilities	ELO - 1
	NoSeWa - 2
	NoSewr - 3
	AllPub - 4
LandSlope	Sev - 1
	Mod - 2
	Gtl - 3
ExterQual	Po - 1
	Fa - 2
	TA - 3
	Gd - 4
	Ex - 5
ExterCond	Po - 1
	Fa - 2
	TA - 3
	Gd - 4
	Ex - 5
BsmtCual	None - 0
	Po - 1
	Fa - 2
	TA - 3
	Gd - 4
	Ex - 5

Ordinal Variable	Transformation
BsmtCond	None - 0
	Po - 1
	Fa - 2
	TA - 3
	Gd - 4
	Ex - 5
BsmtExposure	None - 0
	Mn - 1
	Av - 2
	Gd - 3
BsmtFinType1	None - 0
	Unf - 1
	LwQ - 2
	Rec - 3
	BLQ - 4
	ALQ - 5
	GLQ - 6
BsmtFinType2	None - 0
	Unf - 1
	LwQ - 2
	Rec - 3
	BLQ - 4
	ALQ - 5
	GLQ - 6
HeatingQC	Po - 1
	Fa - 2
	TA - 3
	Gd - 4
	Ex - 5
KitchenQual	Po - 1
	Fa - 2
	TA - 3
	Gd - 4
	Ex - 5

Ordinal Variable	Transformation
Functional	Sal - 1
	Sev - 2
	Maj2 - 3
	Maj1 - 4
	Mod - 5
	Min2 - 6
	Min1 - 7
	Typ - 8
FireplaceQu	None - 0
	Po - 1
	Fa - 2
	TA - 3
	Gd - 4
	Ex - 5
GarageQual	None - 0
	Po - 1
	Fa - 2
	TA - 3
	Gd - 4
	Ex - 5
GarageCond	None - 0
	Po - 1
	Fa - 2
	TA - 3
	Gd - 4
	Ex - 5
PavedDrive	N - 0
	P - 1
	Y - 2
FireplaceQu	None - 0
	Fa - 1
	TA - 2
	Gd - 3
	Ex - 4

Table 8: Summary statistics for the Ames, Iowa Housing Data.

	Count	Mean	STD	MIN	25%	50%	75%	MAX
ID	2919.0	1460.000000000	842.787043090	1.0	730.5	1460.0	2189.5	2919.0
MSSUBCLASS	2919.0	57.137718397	42.517627829	20.0	20.0	50.0	70.0	190.0
LOTFRONTAGE	2433.0	69.305795314	23.344904707	21.0	59.0	68.0	80.0	313.0
LOTAREA	2919.0	10168.114080164	7886.996359106	1300.0	7478.0	9453.0	11570.0	215245.0
OVERALLQUAL	2919.0	6.089071600	1.409947207	1.0	5.0	6.0	7.0	10.0
OVERALLCOND	2919.0	5.564576910	1.113130747	1.0	5.0	5.0	6.0	9.0
YEARBUILT	2919.0	1971.312778349	30.291441534	1872.0	1953.5	1973.0	2001.0	2010.0
YEARREMODADD	2919.0	1984.264474135	20.894344234	1950.0	1965.0	1993.0	2004.0	2010.0
MASVNRAREA	2896.0	102.201312155	179.334253038	0.0	0.0	0.0	164.0	1600.0
BSMTFINSF1	2918.0	441.423235093	455.610825870	0.0	0.0	368.5	733.0	5644.0
BSMTFINSF2	2918.0	49.582248115	169.205611100	0.0	0.0	0.0	0.0	1526.0
BSMTUNFSF	2918.0	560.772104181	439.543659423	0.0	220.0	467.0	805.5	2336.0
TOTALBSMTSF	2918.0	1051.777587389	440.766258116	0.0	793.0	989.5	1302.0	6110.0
1STFLRSF	2919.0	1159.581706064	392.362078667	334.0	876.0	1082.0	1387.5	5095.0
2NDFLRSF	2919.0	336.483727304	428.701455518	0.0	0.0	0.0	704.0	2065.0
LOWQUALFINSF	2919.0	4.694415896	46.396824517	0.0	0.0	0.0	0.0	1064.0
GRLIVAREA	2919.0	1500.759849263	506.051045118	334.0	1126.0	1444.0	1743.5	5642.0
BSMTFULLBATH	2917.0	0.429893726	0.524735634	0.0	0.0	0.0	1.0	3.0
BSMTHALFBATH	2917.0	0.061364415	0.245686916	0.0	0.0	0.0	0.0	2.0
FULLBATH	2919.0	1.568002741	0.552969260	0.0	1.0	2.0	2.0	4.0
HALFBATH	2919.0	0.380267215	0.502871600	0.0	0.0	0.0	1.0	2.0
BEDROOMABVGR	2919.0	2.860226105	0.822693101	0.0	2.0	3.0	3.0	8.0
KITCHENABVGR	2919.0	1.044535800	0.214462001	0.0	1.0	1.0	1.0	3.0
TOTRMSABVGRD	2919.0	6.451524495	1.569379144	2.0	5.0	6.0	7.0	15.0
FIREPLACES	2919.0	0.597122302	0.646129359	0.0	0.0	1.0	1.0	4.0
GARAGEYRBLT	2760.0	1978.113405797	25.574284724	1895.0	1960.0	1979.0	2002.0	2207.0
GARAGECARS	2918.0	1.766620973	0.761624323	0.0	1.0	2.0	2.0	5.0
GARAGEAREA	2918.0	472.874571624	215.394814994	0.0	320.0	480.0	576.0	1488.0

	Count	Mean	STD	MIN	25%	50%	75%	MAX
WOODDECKSF	2919.0	93.709832134	126.526589310	0.0	0.0	0.0	168.0	1424.0
OPENPORCHSF	2919.0	47.486810552	67.575493392	0.0	0.0	26.0	70.0	742.0
ENCLOSEDPORCH	2919.0	23.098321343	64.244245593	0.0	0.0	0.0	0.0	1012.0
3SSNPORCH	2919.0	2.602261048	25.188169331	0.0	0.0	0.0	0.0	508.0
SCREENPORCH	2919.0	16.062350120	56.184365111	0.0	0.0	0.0	0.0	576.0
POOLAREA	2919.0	2.251798561	35.663945965	0.0	0.0	0.0	0.0	800.0
MISCVAL	2919.0	50.825967797	567.402210550	0.0	0.0	0.0	0.0	17000.0
MOSOLD	2919.0	6.213086674	2.714761774	1.0	4.0	6.0	8.0	12.0
YRSOLD	2919.0	2007.792737239	1.314964489	2006.0	2007.0	2008.0	2009.0	2010.0
SALEPRICE	1460.0	180921.195890411	79442.502882887	34900.0	129975.0	163000.0	214000.0	755000.0

Table 9: Features with a Fisher-Pearson coefficient of skewness (g_1) greater than 1.

Variable	g_1
LotArea	4.029
Street	-18.888
Alley	4.283
LotShape	-1.278
Utilities	-37.895
LandSlope	-4.910
MasVnrArea	2.301
ExterCond	1.391
BsmtFinSF2	4.132
LowQualFinSF	9.200
BsmtHalfBath	4.162
KitchenAbvGr	4.487
Functional	-4.868
GarageQual	-3.196
GarageCond	-3.298
PavedDrive	-3.295
WoodDeckSF	1.443
OpenPorchSF	2.334
EnclosedPorch	2.866
3SsnPorch	10.217
ScreenPorch	4.112
PoolArea	19.382
PoolQC	23.688
MiscVal	10.679

Variable	g_1
OverallQual-3	1.383
GrLivArea-2	1.845
GrLivArea-3	3.140
GarageCars-3	2.051
GarageCars-sqrt	-1.839
ExterQual-2	1.156
ExterQual-3	1.539
KitchenQual-3	1.233
GarageArea-2	1.430
GarageArea-3	2.478
GarageArea-sqrt	-1.672
FullBath-3	1.817
GarageAge-2	2.133
GarageAge-3	4.002
TotalBsmtSF-2	2.297
TotalBsmtSF-3	5.310
TotalBsmtSF-sqrt	-1.580
AgeAtSale-3	1.318
1stFlrSF-2	1.722
1stFlrSF-3	3.067
TotRmsAbvGrd-2	1.534
TotRmsAbvGrd-3	2.556
Fireplaces-2	2.665
Fireplaces-3	4.841