Predicting Release Year of Song using Regression

Prakhar Dogra
Department of Computer Science,
George Mason University, Fairfax, VA
pdogra@gmu.edu

***Abstract*— With increase in the amount of publicly available digital music data there has been significant research on evolution of music over the years. And some of that research conducted involves researchers using machine learning techniques to predict release year of the song using its preprocessed timbre features (namely, the averages and covariances of the timbre vectors). This paper suggests using different regression models to solve this problem and then later comparing it to the prior work done on the same.**

I. INTRODUCTION

Over the years the music industry has changed, especially, with the introduction of digital, as opposed to analog, formats in the 1980s [1]. And during that time, technological innovations began to slowly replace the old methods of the recording industry. For the past few decades researchers have applied different preprocessing techniques to audio data in order to extract different types of features that can be later used for solving certain prediction problems [2].

Million Song Dataset is one such example, which is a collection of audio features and music meta-data for a million contemporary popular tracks. It is also a cluster of complementary datasets contributed by the community. The wide variety of features and meta-data has allowed various people in the machine learning and data mining community to do meta data analysis, artist recognition, automatic music tagging, recommendation, cover song recognition, linking lyrics to audio features, year prediction, mood classification, etc. [3][4] The work we review and demonstrate in this paper deals with release year prediction of songs using well known regression models. Later these models are compared against each other and with the results of other papers that have also dealt with the same problem of predicting the release year of the song using its respective timbre features.

This paper is organized as follows: Section 2 gives a brief explanation of related background work in the research domain of digital music and using machine learning techniques for year prediction. Section 3 discusses about our approach using different regression models. Section 4 shows the design of the experiment starting from train-test split to comparing the methods using different evaluation metrics. Section 5 describes the experiment results and provides detailed analysis on it. Section 6 concludes the research undertaken while discussing the potential applications and directions of future work.

II. BACKGROUND WORK

Year prediction is defined as estimating the year in which a song was released based on its audio features. The Million Song Dataset was released in 2011 and very limited research has been done on year prediction using the dataset since then. Mahieux, Ellis, Whitman and Lamere presented year prediction as a case study because of its practical applications in music recommendation [3]. They used the average and covariance of the timbre vectors for each song to predict its corresponding release year. Timbre (also known as tone color or tone quality) is the perceived sound quality of a musical sound. Timbre makes a particular musical sound different sound from another sound, even if they have the same pitch and loudness. And each song is comprised of segments and each segment has a timbre vector of length 12. The timbre averages were derived by averaging the timbre vector over all the segments of the song and the covariances were calculated by determining the pairwise covariances of the timbre values in the vector over all the segments. Moreover, in their research a split between train and test artists was defined and published so future results could be directly comparable. In order to avoid problems such as the producer effect the data was split among artists and not songs. This was done so that no song from a given artist ends up in both the train and test set. They used k nearest neighbor and Vowpal Wabbit [5] compared the Mean Absolute Error with the constant prediction model which always predicts the average release year from the training set (i.e, 1998.4). [3]

Mishra, Garg, Kumar and Gupta used different machine learning algorithms like Linear Regression, Random Forest and Gradient Boosted Trees available in the Apache Spark Machine Learning library (MLlib) to predict the release year of the song. They also compared the training times for single and multinode cluster environment using Apache Spark. [6]

It has been observed that people have a particular liking for different kinds of music like classic, country, rock, pop, etc. [7]. They have a particular liking for music from certain times of their lives (such as high school), thus the predicted year could be a useful basis for recommendation of songs. Moreover, a good prediction model that is based on the variation in music audio characteristics, like timbre, through the years can enlighten us on the century wide evolution of popular music [3]. It is also hard to find prior work addressing prediction of release year of songs. One of the reasons might be the lack of a large music collection that

consist of a wide variety of genres and over a significantly long period of time. However, it should be noted that many music genres are somehow associated with specific periods of time, so we can assume this problem is related to genre recognition and automatic tagging as well. [8]

III. APPROACH

The dataset used for our research is a subset of the Million Song Dataset consisting of more than half a million data entries. By looking at the dependent variable of the dataset, i.e, release year of the song, we can say that this is a regression problem for mainly two reasons: (1) In the dataset, the release year of the song ranges from 1922 to 2011. If a classifier is trained on this dataset then the model can only predict songs from 1922 to 2011. It will not be able to predict release year of the songs that came before 1922 and the ones that came after 2011. Whereas a regression model can always predict years before 1922 and after 2011. (2) The release year can also be considered as a continuous variable, which makes it ideal for regression.

This paper aims to demonstrate seven different types of regression models in order to predict release year of a song using its timbre averages and covariances. Following are the regression models that were used for the purpose of our research:

A. Linear Regression Model

Linear regression models a linear relationship between a dependent variable (denoted as y) and independent variables (denoted as X). A simple linear model aims to minimize the cost function which is defined as the sum of the squared differences between y and the hypothesis h, which is given by

$$h = b + w^T X \tag{1}$$

where h is the hypothesized vector for the given data X, b is the intercept (constant bias term) of the linear equation and w is the weight vector of the same size as the number of independent variables in each data sample.

And the cost function is given as

$$\min \frac{1}{2} \sum_{i=1}^{m} (h_i - y_i)^2 \tag{2}$$

where m is the number of data samples, h_i is the hypothesis value for given sample X_i and y_i is the true value corresponding to X_i. [9]

B. Polynomial Regression

Polynomial regression, also called as multivariate regression, is a form of regression in which the relationship between the independent variable X and the dependent variable y is modeled as a polynomial of degree n in X. Although, the model is non linear in independent variables, it still is linear in the corresponding coefficients (i.e, the weights) that are determined from the data. Let's say that the new set of independent variables (say X') contains all the terms upto the polynomial degree n in X (these terms upto the polynomial degree n also include the product of different features with each other). Lets consider a polynomial regression model for a polynomial of degree 2. Lets assume a two dimensional feature vector X, which can be represented as $[x_1 x_2]^T$ and, therefore, the hypothesis h is given by

$$h = b + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2 \tag{3}$$

where $[w_1, w_2, w_3, w_4, w_5]^T$ is the weight vector. [9] Similar to linear regression, the hypothesis h can be written as

$$h = b + w^T X' \tag{4}$$

where w is the weight vector of the same size as the X'_i (representing one data sample of X').

C. Lasso Regression

Lasso Regression is a form of regression method that performs both variable selection and regularization in order to avoid over fitting. This type of model is able to achieve both of these goals by forcing the sum of the absolute value of the weights to be less than a fixed value, which forces certain weights to be set to zero, resulting in a model that includes only selected weights (non-zero). Here the weights are penalized based on the L1 norm (Least Absolute). Following is the cost function of the lasso regression model,

$$\min \frac{1}{2} \sum_{i=1}^{m} (h_i - y_i)^2 + \lambda ||w|| \tag{5}$$

where λ governs the relative importance of the regularization term compared with the cost term. And $||w||$ denotes the magnitude of the weight vector, w. [10]

D. Ridge Regression

Ridge Regression, similar to Lasso regression, is a form of regression method that performs regularization in order to avoid over fitting. But, unlike lasso, this type of model is unable to achieve variable selection and only shrinks the weights (i.e, it does not set any of them to zero). But, it penalizes larger weights even more than the smaller ones because the penalty (regularizer) term is square of the weights, i.e, L2 norm (Least Square). Following is the cost function of the ridge regression model,

$$\min \frac{1}{2} \sum_{i=1}^{m} (h_i - y_i)^2 + \lambda ||w||^2 \tag{6}$$

where λ governs the relative importance of the regularization term compared with the cost term. And $||w||$ denotes the magnitude of the weight vector, w. [9]

E. Elastic Net Regression

Elastic Net is a regularized regression method that linearly combines the L1 and L2 penalties of the lasso and ridge methods, respectively. Following is the cost function of the ridge regression model,

$$\min \frac{1}{2} \sum_{i=1}^{m} (h_i - y_i)^2 + \lambda_1 ||w|| + \lambda_2 ||w||^2 \tag{7}$$

where λ_1 is the regularization coefficient for the L1 norm and λ_2 is the regularization coefficient for the L2 norm. And $||w||$ denotes the magnitude of the weight vector, w. [11]

F. Neural Network Regression

Neural Networks, also known as multilayer perceptrons, can be described as a series of functional transformations and are considered as good function approximators.[12] For the purpose of our research, a single hidden layer neural network has been considered with a logistic sigmoid activation function to introduce non-linearity in the relationship between independent variables and dependent variable. Sigmoid activation function is represented as:

$$\sigma(h) = \frac{1}{1 + \exp(-h)} \tag{8}$$

where h is the linear combination of the input features, i.e, $w^T X$ where w is the weight vector and X is the input vector. Following is a pictorial representation of a neural network with 4 feature inputs and 5 neurons in the hidden layer:

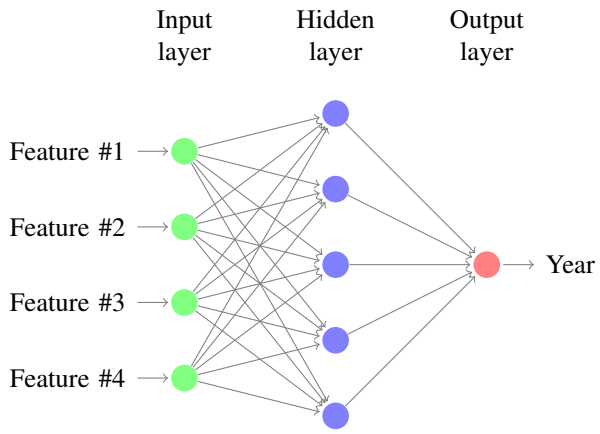

Fig. 1. Structure of a Neural Network with single hidden layer. Each circle represents a neuron in the network.

G. Step Regression

Step Regression is a wrapper selection approach in which the features are chosen in an iterative fashion based on the performance of the regression model on a certain evaluation metric (for example, adjusted R^2 score). In each step of the iteration, a feature is considered for addition to or subtraction from the set of features based on some pre-specified criterion (evaluation criteria mentioned above). The iterative procedure that has been used for our research is the Best First Forward Selection technique [13][14].

IV. EXPERIMENTAL DESIGN

This section presents the experimental design used for our research. The design steps include discussing about the training and testing dataset split, normalizing the datasets, training and validating the regression models (described in the previous section) which includes the use of batch and stochastic gradient descent algorithm to minimize the cost function followed by testing the models on a separate test set and comparing them on the basis of certain evaluation metrics. All of the steps have been explained in the following subsections:

A. Train and test set split

The data set used for our research consists of 515,345 data entries and the train-test split was provided by the source of the data set.[15] The train data set contains 463,715 entries while test data set contains 51,630 entries. This train-test split avoids the 'producer effect' by making sure no song from a given artist ends up in both the train and test set.

B. Normalization of dataset

Before training and validation for any model, data normalization is done in order to have same range of values for each of the input features and provide stable convergence of weights and biases.

C. Training and Cross Validation

Using the train dataset, each of the regression models (described in the previous section) are trained. Following are the two types of cross validation techniques that were used to check if the model is over-fitting or under-fitting:

- Hold out Cross Validation: In holdout method, the data set is divided into two parts. First part is the training set that generates the weights and biases for the model. Second part is the testing set that uses the weights and biases of the model to predict the output values for the data in the testing set. Evaluation for this method can have significant variance but is used when training specific model takes considerable time.
- K-fold Cross validation: K-fold method is an improvement over the hold out method. In this method, the data set is divided into k parts. This procedure is repeated k times and each time, one of the parts is used as the test set and the remaining k-1 parts are used as a training set. Then the average metric is calculated across all k iterations.[16] This results in much less variance during evaluation but the trade off here is that the model is trained k times which takes almost k times the time it usually takes for the hold out method.

D. Hyper parameter tuning

For each regression model, set of parameters were chosen either via manual inspection or via grid search. Learning rate, number of epochs (or iterations) and batch size were the hyper parameters that were tuned via manual inspection. Following are the regression models and their corresponding hyper parameters that were chosen via grid search:

E. Gradient Descent Optimization

Gradient descent is an iterative procedure that has been used to minimize the cost (described for the regression models in previous section). Following are the two types of gradient descent optimizers that have been used for minimizing the cost functions for each of the regression models:

TABLE I

HYPERPARAMETERS TUNED VIA GRID SEARCH

Model	Hyperparameters
Lasso	λ
Ridge	λ
Elastic Net	λ_1, λ_2
Neural Network	Number of hidden nodes

- Batch Gradient Descent: In this method, gradient of the cost function is calculated using the whole dataset. And it may end up in a local or global minimum of the cost function.
- Stochastic Gradient Descent: Also known as incremental gradient descent, is a stochastic approximation of the batch gradient descent optimization algorithm. In this method, gradient of the cost function is calculated using small number of samples (can be single sample as well). And as a result, can find multiple local minima of the cost function and may eventually find a global minimum over multiple iterations.[17] Also this method is well suited for large datasets because the whole dataset cannot be held in RAM at once. This technique is specially helpful in polynomial regression. This can be shown by taking the example of the dataset used for our research. This dataset has 90 features and more than half a million data entries. A polynomial of degree 2 will have 4196 terms and for degree 3, the polynomial will have more than a hundred thousand terms in its linear form (linear in the weights). It will require a huge amount of memory to generate and store polynomial terms. And that is why, for the purpose of our research only a polynomial of degree 2 has been considered (using stochastic gradient descent).

F. Testing the models

After careful training and cross validation, the models are tested against the initially separated test dataset (described in the subsection A). Following evaluation criteria were used for the purpose of this research:

- Mean Absolute Error : It is the mean of the absolute differences between the predicted and actual values. It can be written as:

$$MeanAbsoluteError = \frac{1}{m}\sum_{i=1}^{m}|h_i - y_i| \quad (9)$$

- Root Mean Square Error : It is the square root of the mean of the squared differences between the predicted and actual values. It can be written as:

$$RootMeanSquareError = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(h_i - y_i)^2} \quad (10)$$

- R^2 Score : It is a regression score function that represents the coefficient of determination of the model. It is calculated as the proportion of the variance in the dependent variable (X) that is predictable from the independent variable (y). It is calculated as follows:

$$SS_R = \sum_{i=1}^{m}(y - h)^2 \quad (11)$$

$$SS_T = \sum_{i=1}^{m}(y - \overline{y})^2 \quad (12)$$

$$R^2 = 1 - \frac{SS_R}{SS_T} \quad (13)$$

where SS_R is the sum of squared errors, $\overline{y}$ is the mean of the true values and SS_T is the total variance (sum of squared difference between each sample and its mean).
- Explained Variance Score : It is measured as the proportion to which a model takes in account for the variation of a given data set. It is calculated as:

$$ExplainedVariance = 1 - \frac{Var(y-h)}{Var(y)} \quad (14)$$

where $Var(y)$ represents variance of the samples in vector y.

V. EXPERIMENTAL RESULTS

Following are the results of the experiment when tested on the separate test data and compared with each other on the basis of evaluation metrics (discussed in the previous section).

TABLE II

COMPARING OUR REGRESSION MODELS

Regression Model	MAE	RMSE	R^2	EV
Linear	6.84225	9.52570	0.22948	0.22971
Linear (L1)	6.83531	9.52708	0.22933	0.22926
Linear (L2)	6.85309	9.59703	0.23410	0.23410
Linear(L1 and L2)	6.83393	9.52653	0.22934	0.22942
Polynomial	6.61332	9.35797	0.25638	0.25642
Polynomial (L1)	6.67173	9.29383	0.27299	0.27301
Polynomial (L2)	6.66768	9.29647	0.27257	0.27261
Polynomial(L1 and L2)	6.74396	9.51341	0.23147	0.23149
Neural Network	6.88925	9.68270	0.21433	0.21387
Stepwise	6.84396	9.53179	0.22850	0.22872

Here, L1 represents Lasso regression, L2 represents Ridge regression, L1 and L2 represents Elastic Net regression, MAE represents Mean Absolute Error, RMSE represents Root Mean Square Error and EV represents Explained Variance.

From the above table, it can be concluded that polynomial regression without regularization produces the smallest Mean Absolute Error. And it can be also be observed that using L1 regularization gives least Root Mean Square. Although, the difference is very small, it is observed that using L1 regularization improves the result by a very tiny margin and using L2 regularization doesn't. In general, we can say that the models are under-fitting because adding more complexity in case of polynomial regression certainly improves the results. And adding regularization terms or doing feature

TABLE III

HYPERPARAMETERS TUNED VIA GRID SEARCH

Model	Hyperparameters
Lasso(Linear)	λ = 7.29 X 10^-4
Ridge(Linear)	λ = 8.1 X 10^-5
ElasticNet(Linear)	λ_1 = 0.00027, λ_2 = 0.00027
Lasso(Polynomial)	λ = 2.7 X 10^-4
Ridge(Polynomial)	λ = 1 X 10^-5
ElasticNet(Polynomial)	λ_1 = 0.00003, $\lambda_2 = 0.00009$
Neural Network	Number of hidden nodes = 180

selection (reducing number of features) in case of step regression doesn't improve the results.

Now lets compare our results with the results from [3].

TABLE IV

COMPARING WITH OTHER MODELS

Model	MAE	RMSE
Polynomial	6.61	9.35
Constant Prediction	8.13	10.80
1-NN	9.81	13.99
50-NN	7.58	10.20
Vowpal Wabbit	6.14	8.76

From the above table, it is clear that our best model (i.e, Polynomial (degree = 2) regression) is not better than Vowpal Wabbit model but it is significantly better than the constant prediction and nearest neighbor models in [3]. And actually all the regression models in table II are better than the nearest neighbor models.

VI. CONCLUSION AND FUTURE WORK

The work presented in this paper highlighted the use of different kinds of regression models to predict the release year of a song based on its timbre features. Each model was trained and validated on the training data via cross validation in order to tune the hyper parameters. After tuning the hyper-parameters, the models were tested on test data and compared with each other on the basis of certain evaluation metrics. Our results were then compared with other models (prior work on this problem). It was concluded after comparisons that regression models are indeed useful for this type of prediction problem. And adding more complexity to those models also improves their prediction power like in case of polynomial regression.

This certainly is an interesting problem, and is not limited to predicting release year of the song, but has applications like song recommendation based on users age, genre detection or doing analysis on evolution of music over the past century. And there are certainly additional avenues like Gradient Boosting and Random Forest that can be explored to solve the problem of predicting the release year of the song using its timbre features. It is possible that these models can provide more complexity similar to what polynomial regression models were able to.

ACKNOWLEDGEMENT

I would like to thank Professor Carlotta Domeniconi for helping me through the research undertaken. She was extremely helpful in providing published resource on wrapper selection method and by discussing the details of polynomial (multivariate) regression with me.

REFERENCES

[1] E. Hitters and M. Kamp, The Music Industries: Changing Practices and New Research Directions. Proceedings of the IASPM Benelux conference, Popular Music: Theory and Practice in the Lowlands. Haarlem, the Netherlands, Apr. 2011, pp 210–229.

[2] S. N. Tran, D. Wolff, T. Weyde and A. Garcez. Feature Preprocessing with RBMs for Music Similarity Learning. AES 53RD INTERNATIONAL CONFERENCE, London, UK, Jan. 2014

[3] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The Million Song Dataset. In Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011), 2011.

[4] Humberto Corona, Michael P. OMahony, An Exploration of Mood Classification in the Million Songs Dataset. Open-access article.

[5] J. Langford, L. Li, and A. L. Strehl. Vowpal wabbit (fast online learning), 2007. http://hunch.net/vw/.

[6] P. Mishra, R. Garg, A. Kumar, A. Gupta and P. Kumar, "Song year prediction using Apache Spark," 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Jaipur, 2016, pp. 1590–1594.

[7] Schfer T (2016) The Goals and Effects of Music Listening and Their Relationship to the Strength of Music Preference. PLOS ONE 11(3): e0151634.

[8] T. Bertin-Mahieux, D. Eck, and M. Mandel. Automatic tagging of audio: The state-of-the-art. In Wenwu Wang, editor, Machine Audition: Principles, Algorithms and Systems, pages 334-352. IGI Publishing, 2010

[9] C. M. Bishop. Pattern Recognition and Machine Learning. p. 4–11.

[10] Tibshirani, Robert. Regression Shrinkage and Selection via the Lasso. Journal of the Royal Statistical Society. Series B (Methodological), vol. 58, no. 1, 1996, pp. 267-288. JSTOR, JSTOR, www.jstor.org/stable/2346178.

[11] Zou, Hui, and Trevor Hastie. Regularization and Variable Selection via the Elastic Net. Journal of the Royal Statistical Society. Series B (Statistical Methodology), vol. 67, no. 2, 2005, pp. 301-320. JSTOR, JSTOR, www.jstor.org/stable/3647580.

[12] Balzs Csand Csji (2001) Approximation with Artificial Neural Networks; Faculty of Sciences; Etvs Lornd University, Hungary.

[13] M. Karagiannopoulos, D. Anyfantis, S. B. Kotsiantis, P. E. Pintelas. Feature Selection for Regression Problems.

[14] Guyon, Isabelle and Elisseeff, André. An Introduction to Variable and Feature Selection. Journal of Machine Learning Research. Mar. 2003, vol 3. p. 1157–1182

[15] Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[16] C. M. Bishop. Pattern Recognition and Machine Learning. p. 32–33.

[17] C. M. Bishop. Pattern Recognition and Machine Learning. p. 143–144.

APPENDIX

Github repository weblink

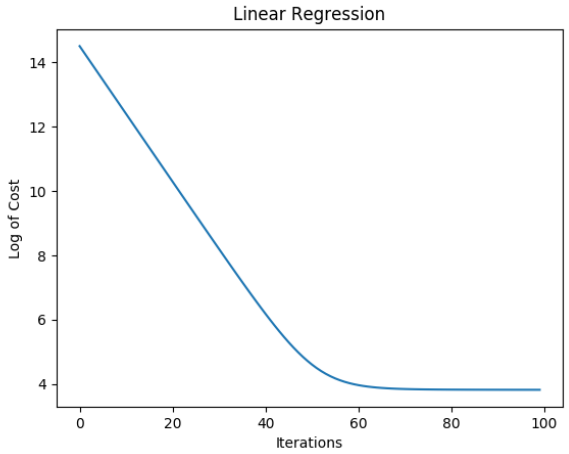

Fig. 2. Cost vs Number of iterations graph for Linear Regression - Batch Gradient Descent

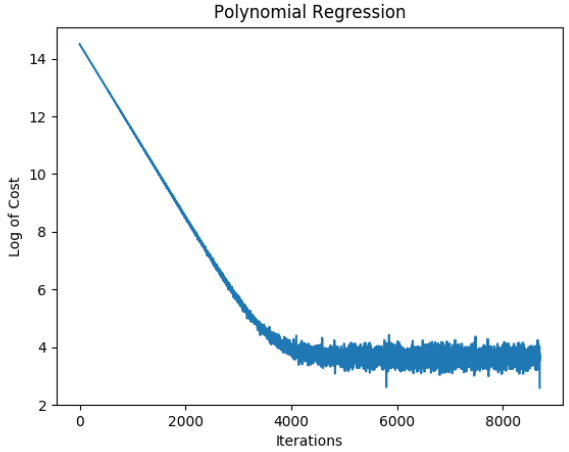

Fig. 3. Cost vs Number of iterations graph for Polynomial Regression - Stochastic Gradient Descent

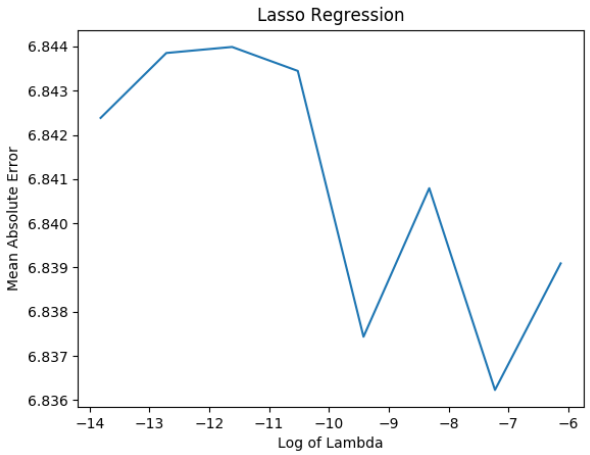

Fig. 4. Mean Absolute Error vs Log of Lambda for Lasso (Linear) Regression

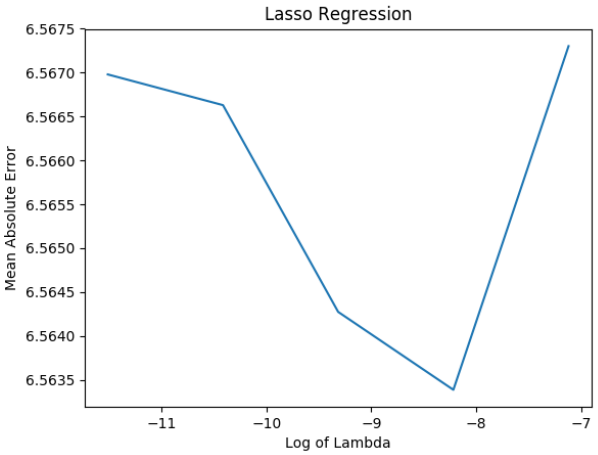

Fig. 5. Mean Absolute Error vs Log of Lambda for Lasso (Polynomial) Regression

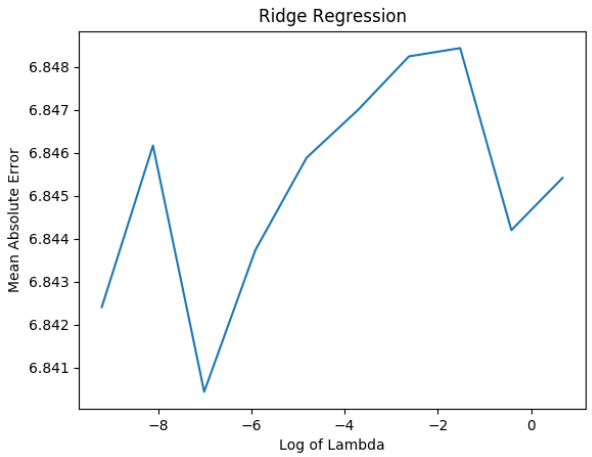

Fig. 6. Mean Absolute Error vs Log of Lambda for Ridge (Linear) Regression

Fig. 7. Mean Absolute Error vs Log of Lambda for Lasso (Polynomial) Regression

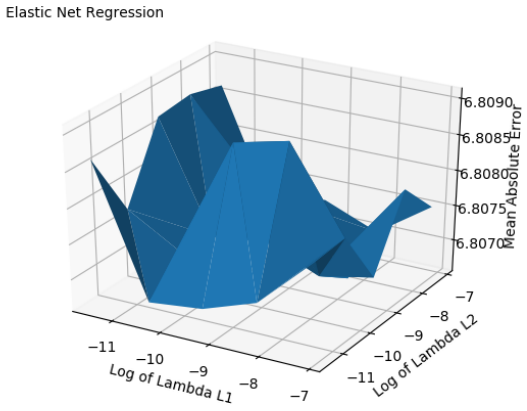

Fig. 8. Mean Absolute Error vs Log of Lambda for Elastic Net (Linear) Regression

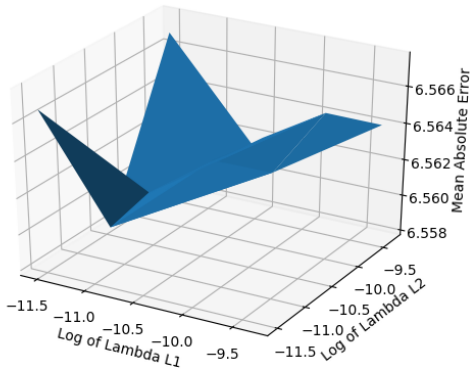

Fig. 9. Mean Absolute Error vs Log of Lambda for Elastic Net (Polynomial) Regression

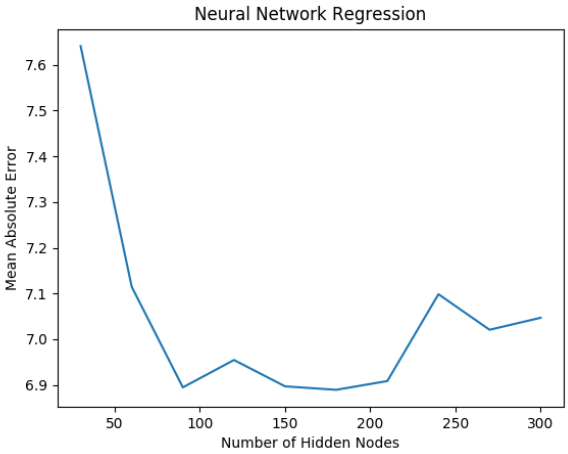

Fig. 10. Mean Absolute Error vs Number of neurons in hidden layer for Neural Network Regression

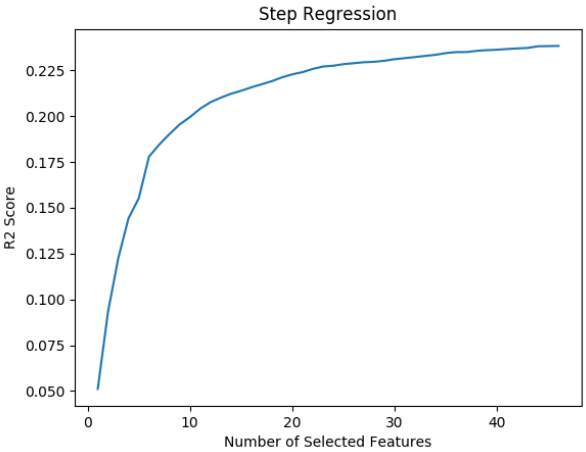

Fig. 11. Adjusted R^2 Score vs the number of selected features for Step Regression