Developing Two Classes:

Linear Regression Assumptions and Model Selection Classes in Python

By:

Tomiwa Omotesho, Iman Khazrak, Sajjad Rezvani Boroujeni, and Ying Xie

Bowling Green State University

CS 6010: Data Science Programming Project

December 2, 2022

Contents

1 Introduction

We have created two classes in this project, the first is for verifying the linear regression assumptions, and the second is for model selection. Linear regression models offer straightforward mathematical formulas for prediction purposes. They are primarily used to find the causal relationship between variables, hoping to express the relationship between variables and targets through linear combinations. Linear regression has the advantage of fast model building and high interpretability because it does not contain complex algorithmic processes. These advantages make the algorithm a popular choice for many data scientists. However, the model assumptions are often overlooked, leading to biased results. Therefore, confirming these assumptions is paramount to successfully conducting regression analysis. Through this research, data scientists can use our linear regression assumption class to check linear regression model assumptions automatically.

Nevertheless, the need for more sophisticated algorithms often arises to solve business challenges in industries. Hence, the motivation for developing a second class that can be used to identify the best regression models suitable for any preprocessed dataset.

2 Methodology

2.1 Linear Regression Assumptions Class

The class for linear regression assumption was built for the four assumptions, namely:

2.1.1 Linearity

To satisfy the linearity assumption, the relationship between response y and the regressors must be linear, or at least approximately so. The supposition can be verified by looking at the scatter plot

of the dependent variables versus the response variable. The scatter plot illustrates the correlation between the regressors and the response variables.

2.1.2 Homogeneity of Variance (Homoscedasticity)

For the assumption of equal variances, we assume that the variability in the response is constant as the value of the predictor increases. We check this assumption by plotting the residual errors against the fitted values. Such plots are referred to as residual plots. The desired properties of a residual plot are as follows:

- Zero mean (Unbiased): an average value of zero must be obtained in any thin vertical strip.
- Constant variance (Homoscedasticity): the spread of the residuals should ideally be the same in any thin vertical strip, as illustrated in Figure 1.

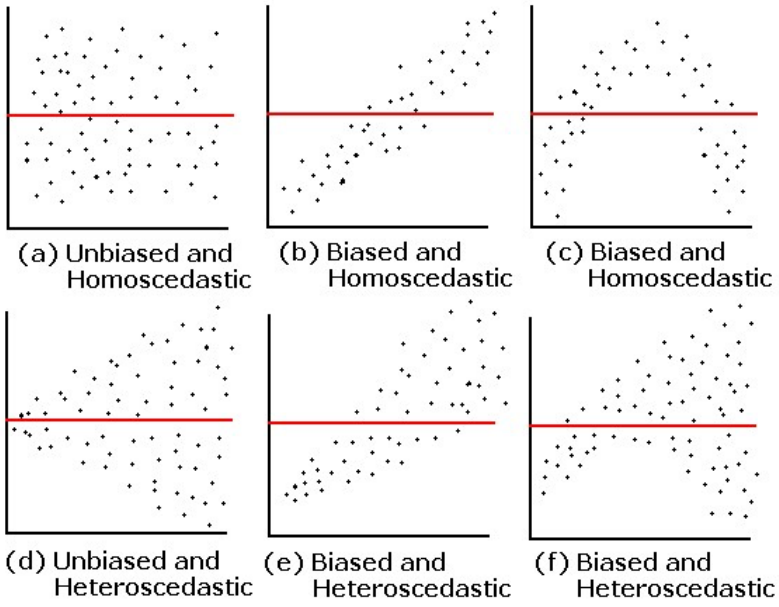


Figure 1: Examples of residual plots [1]

2.1.3 Independence of Error Terms

This assumption is most important when dealing with time series data. Ideally, we want to avoid patterns between successive residuals. The easiest way to verify this assumption is by plotting

residuals against the sorted row numbers. The residuals should be randomly and symmetrically distributed around zero under all conditions [2].

2.1.4 Normality of Error Terms

To visually check that the assumption of normality of error terms is normally distributed, we can use the normality probability plot (also called the Q-Q plot). The assumption of normality is satisfied if the points in the plot roughly form a straight diagonal. The idea is to plot the data against a theoretical normal distribution such that the points form an approximate straight line. Deviation from this straight line indicates a departure from normality. Other approaches to checking the normality assumption involve using formal statistical tests such as the Shapiro-Wilk test, the Kolmogorov-Smirnov test, or the Jarque-Bera test. However, a drawback of these tests is that they are sensitive to large sample sizes and often conclude that the residuals are abnormal.

Correlation plot and Variance Inflation Factor (VIF) functions were also built into the linear regression assumption class to detect multicollinearity between variables in a dataset.

2.2 Model Selection Class

Eleven regression algorithms were integrated into the model selection class. The two parameters needed to run the class successfully are the name of the dataset of interest and the name of the response variable. We used a five-fold cross-validation approach where 80% of the data is randomly split into five subsets and used for model training, while the remaining 20% is used as the test set. The class identifies the best models suitable for a clean dataset based on the Root Mean Square Error (RMSE) values for the models. The results are saved in a Pandas dataframe and displayed as a table when the class is executed. Likewise, a bar plot is generated to display the metrics in ascending order for easy comparison. Since this class aims to identify potential candidate models suitable for each dataset, we mainly used the default values for the

hyperparameters of the models. Appendix 1 shows a comprehensive list of the hyperparameters of each model.

2.2.1 Linear Regression

Linear regression is a type of regression analysis that models the relationship between one or more independent and dependent variables using the least squares function known as the linear regression equation [3]. These are linear combinations of a single model parameter, or many parameters known as regression coefficients. Simple linear regression is a scenario with only one independent variable, while multiple linear regression is a scenario with numerous independent variables.

2.2.2 Ridge Regression

Ridge regression is a regularized regression that controls the regression coefficients, reducing the coefficient's size, and fluctuation, thereby improving the model's accuracy. Mathematically, ridge regression adds an L_2 regular term to the linear model. The penalty parameter λ makes a second-order penalty on the coefficient, also known as the L_2 penalty parameter. When λ tends to zero, ridge regression is no different from the original linear model, while when λ tends to infinity, all coefficients tend to zero. The ridge regression model pushes correlated feature variables towards each other and avoids situations where one has a large positive coefficient and the other has a large negative coefficient. A disadvantage of ridge regression is that it retains all the predictors in the final model, so it cannot be used for feature selection.

2.2.3 Lasso Regression

The full name of Lasso is the least absolute shrinkage and selection operator. As a linear model for estimating sparse parameters, lasso regression is suitable for reducing the number of parameters. Mathematically, Lasso adds an L_1 regular term to the linear model. The parameter λ controls how much the sparse parameter estimation is penalized. Since most of the parameters after Lasso regression are zero, the feature variables corresponding to those not zero can be used as the features of other models. The feature dimension can be reduced without changing the model's accuracy.

2.2.4 Elastic net

Elastic net combines the penalties from lasso and ridge regularization methods to regularize regression models. The elastic net method enhances the lasso's drawbacks, namely that it only needs a few samples for high-dimensional data. Until saturation, "n" variables may be included using the elastic net approach. When there are several highly correlated groups of variables, lasso usually picks one from each group and ignores the others.

2.2.5 Principal Component Analysis (PCA)

PCA is one of the most widely used data dimensionality reduction algorithms. When using PCA, new variables are produced that are weighted linear combinations of the original variables, and that maintains most of the information of the entire dataset. The new variables are created so that they are orthogonal to each other *i.e.,* the correlation between them is zero. PCA is designed to be used for numerical variables. Other approaches, such as correspondence analysis, are better suited for categorical data.

2.2.6 K-Nearest Neighbors (KNN)

KNN is a non-parametric, supervised learning classifier that uses proximity to classify groups of individual data points. The fundamental tenet of the KNN algorithm is that a sample belongs to a category and exhibits the traits of samples in that category if most of its k nearest neighbor samples in the feature space also fall into that category. KNN uses complex distance measures and observable data similarities to produce precise predictions. While it can be used in regression or classification problems, it is often used as a classification algorithm, assuming that similar points can be found near each other. The KNN algorithm is simple and easy to understand.

2.2.7 Decision Tree

The foundation of decision trees is dividing data into smaller groups using splits on predictors. These divisions produce logical rules that are clear and simple to comprehend. Because it does not assume the shape of the relationship between the response variable (Y) and the predictors $X_1, X_2, \ldots, X_p$, this method is non-parametric. A decision tree consists of a root, internal, and leaf nodes. The complete set of samples is contained in the root node, the internal nodes represent feature attribute tests, and the leaf nodes show decision-making outcomes.

2.2.8 Boosting

The idea of boosting is to combine multiple weak learners into strong classifiers. A weak learner is computationally simple and expected to have slightly better accuracy than random guessing. The typical example of a weak learner is a decision stump *i.e., a* decision tree with just one split. These learners are not particularly useful on their own, but when used as a group in the boosting method, they frequently produce pretty accurate predictions.

2.2.9 XGBoost

XGBoost stands for Extreme Gradient Boosting. It is a decision tree-based integrated machine-learning technique that uses the Gradient Boost framework. XGBoost is primarily used to resolve supervised learning issues, in which target variables are predicted using training data with various. A regularization item is added to the loss function of the XGBoost algorithm to help manage the model's complexity. The leaf node count and the sum of the squares of the weight of each leaf node are both included in the regularization item.

2.2.10 Neural networks

Artificial Neural Networks (ANNs), also called Neural Networks (NNs), are based on a biological model of brain activity in which neurons are connected to one another and gain knowledge via experience. Although neural networks are frequently criticized for being difficult to understand, these algorithms show exceptional prediction capability. Their structure supports capturing complex relationships between predictors and an outcome variable, which is often impossible with other predictive models.

2.2.11 Random forest

The term "random forest" describes a classifier that trains and predicts data using several trees. The algorithm's precision, simplicity, and adaptability have made it one of the most used. Given that it can be used for classification and regression tasks and is non-linear, it is quite flexible and can be applied to a wide range of data and circumstances. Random forests can be applied to various tasks, including discrete-valued classification, continuous-value regression, unsupervised learning, clustering, outlier identification, etc.

2.3 Datasets to Test Classes

We used two datasets to test the functionality of our classes. These were House Rental and Medical Insurance datasets.

2.3.1 House Rental Dataset

The dataset reports the average rent for 64 college towns from the 1980 and 1990 United States censuses. It has 128 observations with 23 variables. A detailed description of this dataset can be found in the rdrr.io repository [4]

2.3.2 Medical Insurance Dataset

The dataset which reports the Individual medical costs billed by health insurance was obtained from Kaggle [5]. It has 1338 observations with seven variables, which include *age, sex, BMI, children, smoker, region, and charges*.

2.4 Data Preprocessing

The datasets used for testing the classes required preprocessing. Seven variables were dropped for the house rental dataset due to missing values using the drop() function in the Pandas library. Likewise, the three categorical variables (*sex, smoker, and region*) in the medical insurance dataset were one-hot encoded with the pd.get_dummies() function to ensure numerical inputs were passed into the machine learning algorithms.

3. Experiment and Result Analysis

The results obtained from the experiments carried out on the classes developed are discussed in this section.

3.1 Experiment on House Rental Dataset

The performance results are summaries in Table 1

Table 1: Model Performance

Model	**RMSE**
XGBoost	4.91
Boosting	7.16
Random Forest	11.91
Decision Tree	16.99
Ridge Regression	19.78
Linear Regression	23.25
PCA	23.25
Lasso Regression	24.67
Elastic Net	31.41
Neural Network	37.72
KNN	83.83

We observed using the RMSE measure that the tree-based algorithms had the best performance. XGBoost was the best-performing algorithm, with an RMSE of 4.90, closely followed by Boosting, with an RMSE of 6.60. The two worst-performing models were KNN and NN. Their poor performance is associated with the limited number of records in the dataset. Both algorithms require many records to obtain good results. Figure [2] shows the bar chart plot for the RMSE measures of the algorithms.

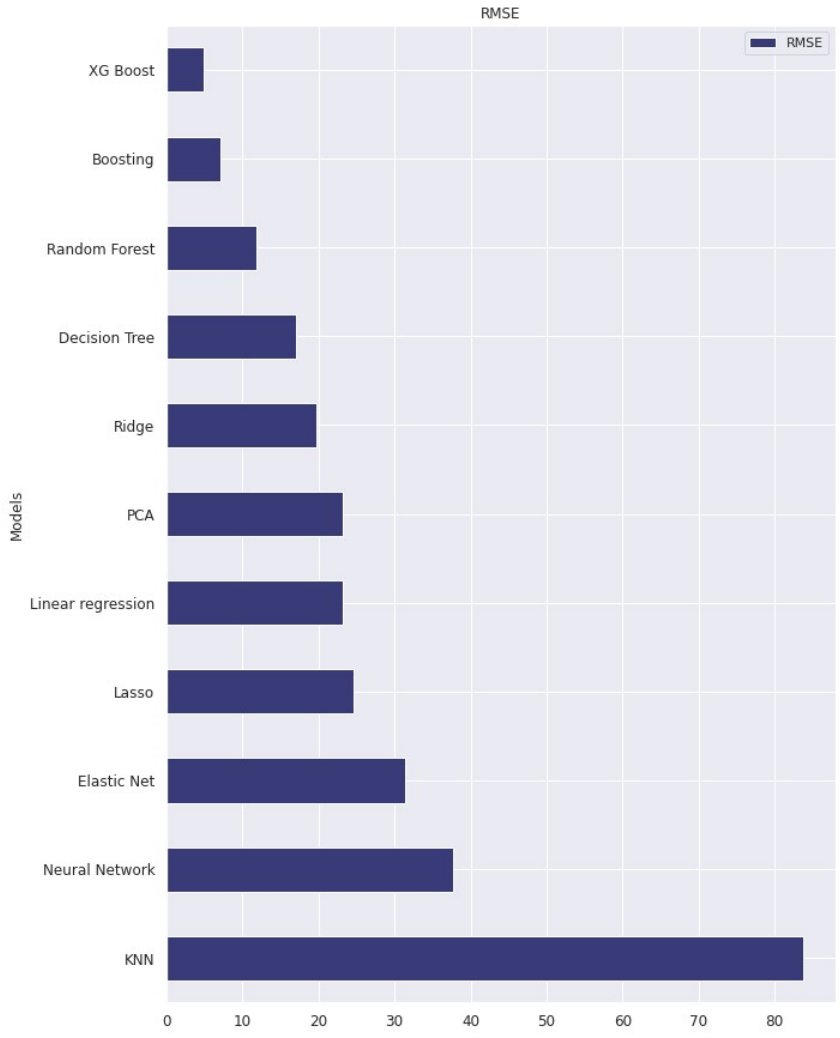


Figure 2: Model performance bar plot

A visual inspection of the linear regression assumptions plots suggested that all four assumptions were violated. We noticed non-linear relationships between the response variable, *rent* and many predictors like *city, year, pop, tothsg, and y90* as seen in Appendix II. The assumption plots for homogeneity of variance and independence of error terms in Appendix III and Appendix IV show distinct patterns and trends, which violates the assumptions. Similarly, the normality assumption was violated. See Appendix V. The violation of the regression assumptions is tied to the moderate performance of the linear regression algorithm.

3.2 Experiment on Medical Insurance Dataset

Table 2 and Figure 3 show the performance results for each model. Higher performance is also observed for the tree-based algorithms with this dataset. The best performer is boosting, while the least is NN.

Table 2: Model Performance

Model	**RMSE**
Boosting	4066.71
Random Forest	4277.06
XGBoost	4772.25
Linear Regression	5641.63
PCA	5641.63
Lasso Regression	5642.18
Ridge Regression	5644.10
Decision Tree	6699.43
Elastic Net	7303.24
KNN	9560.94
Neural Network	12619.41

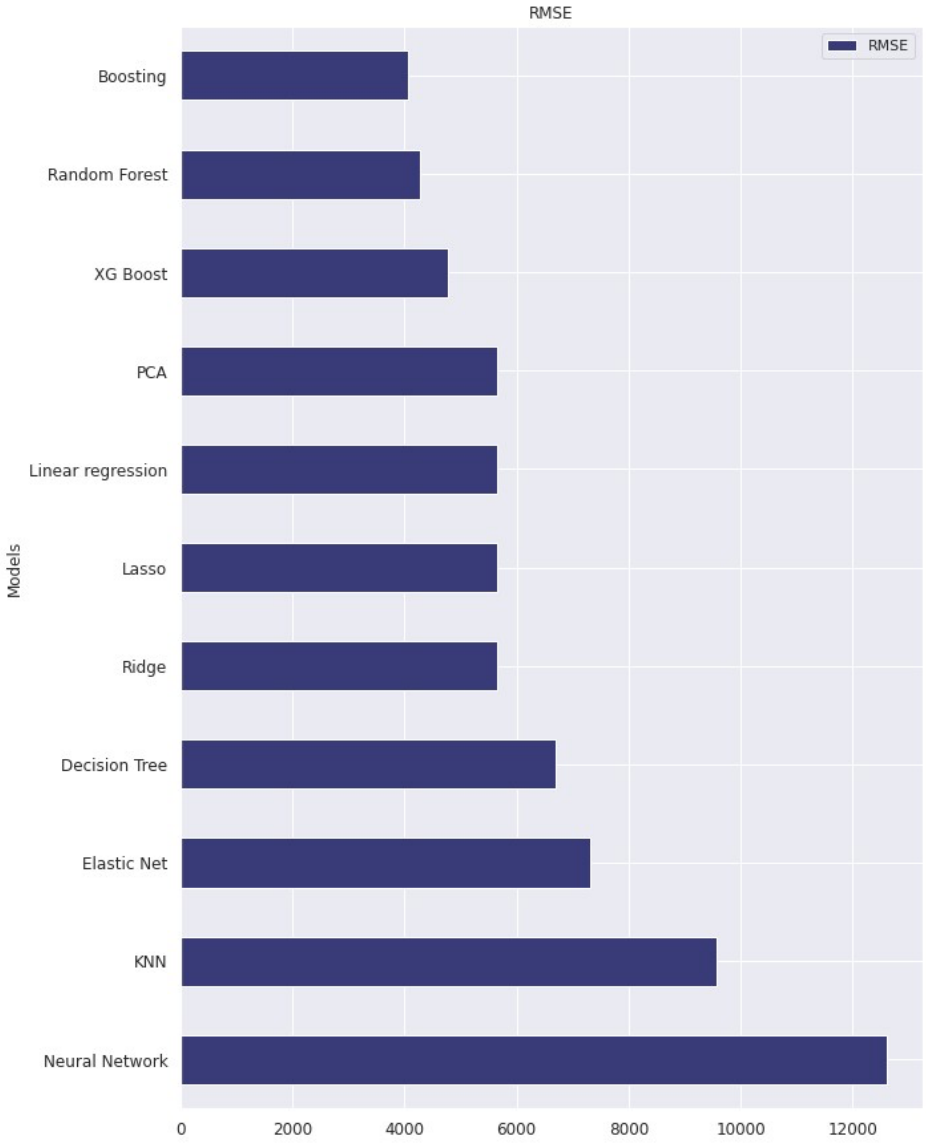


Figure 3: Model performance bar plot

The linear regression assumptions were also violated for this dataset. See the plots in Appendix VI to Appendix IX.

4. Conclusion and Future Works

It is sometimes unclear which ML model works well on a given dataset. The best model will be selected by trying different models and analyzing the results. As a result, the process is time-

consuming, and we must do everything again for the new dataset. According to the experiment section, by eliminating the manual processes, we can find the best model faster by using the model selection class. Furthermore, ML algorithms do not need to be written for every dataset. Writing code for every new dataset is also unnecessary to check linear regression assumptions. All four linearity assumptions are evaluated automatically by the class. Data analysts and data scientists can save much time using these two classes.

More popular ML models can be added to the model selection class for future work. Developing a class of functions for the classification task is also possible. Thus, all ML models can be evaluated on each dataset in the shortest time possible.

References

[1] D. Hocking, “Model validation: Interpreting residual plots: R-bloggers,” *R*, 18-Jul-2011. [Online]. Available: https://www.r-bloggers.com/model-validation-interpreting-residual-plots/. [Accessed: 29-Nov-2022].

[2] F. S. of Business, “Regression diagnostics: testing the assumptions of linear regression,” *Testing the assumptions of linear regression*. [Online]. Available: https://people.duke.edu/~rnau/testing.htm. [Accessed: 04-Dec-2022].

[3] D. Freedman, ‘Statistical Models: Theory and Practice,” *Cambrige University Press,* 2006, p.26

[4] J. Wooldridge, “Introductory Econometrics: A Modern Approach, 5th Edition,” 2013, pages 503-504.

[5] Mariapushkareva, “Medical insurance cost with linear regression,” *Kaggle*, 14-Jul-2020. [Online]. Available: https://www.kaggle.com/code/mariapushkareva/medical-insurance-cost-with-linear-regress ion/data?select=insurance.csv. [Accessed: 01-Dec-2022].

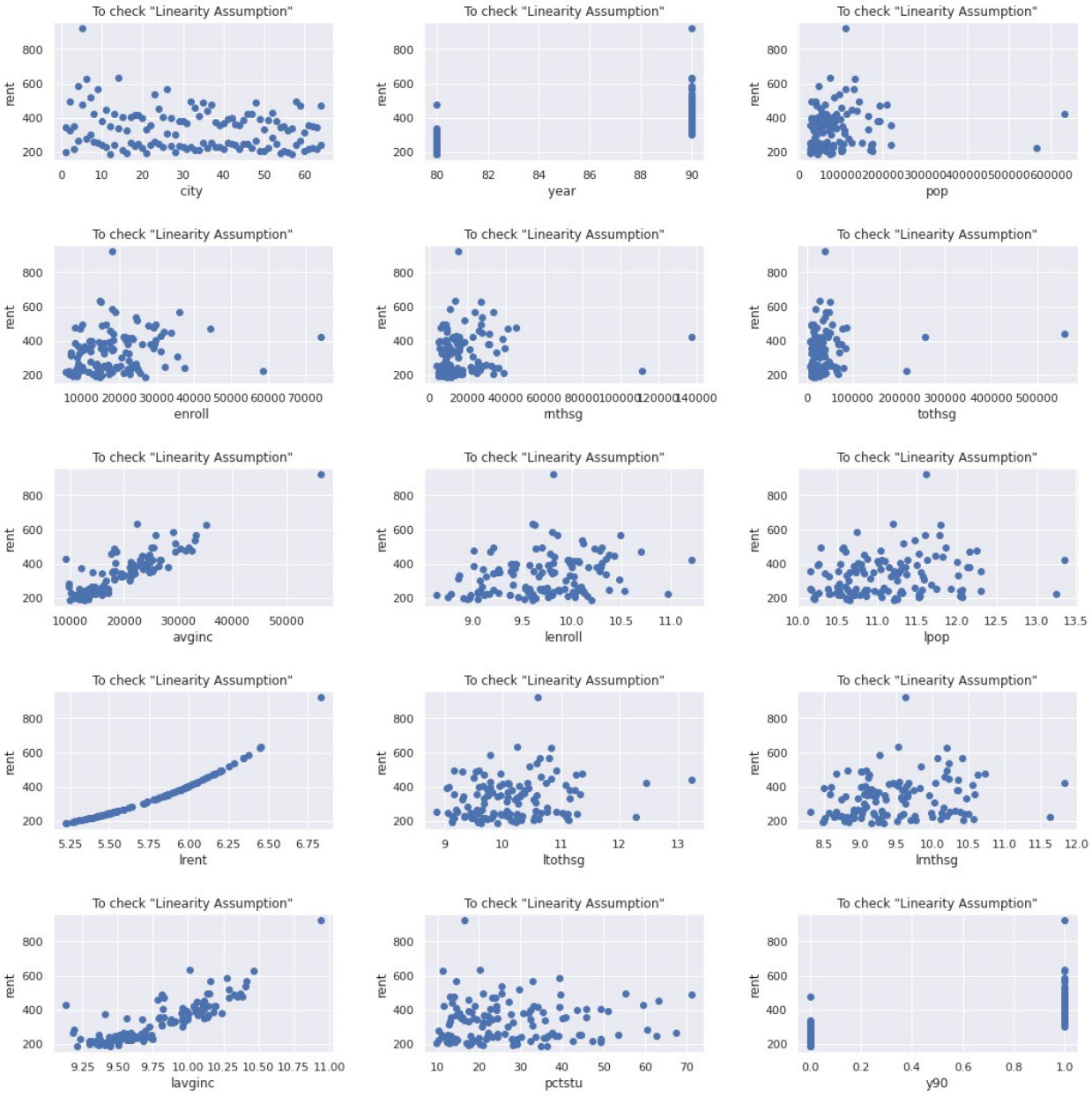
Appendices

Appendix I: Model Hyperparameters

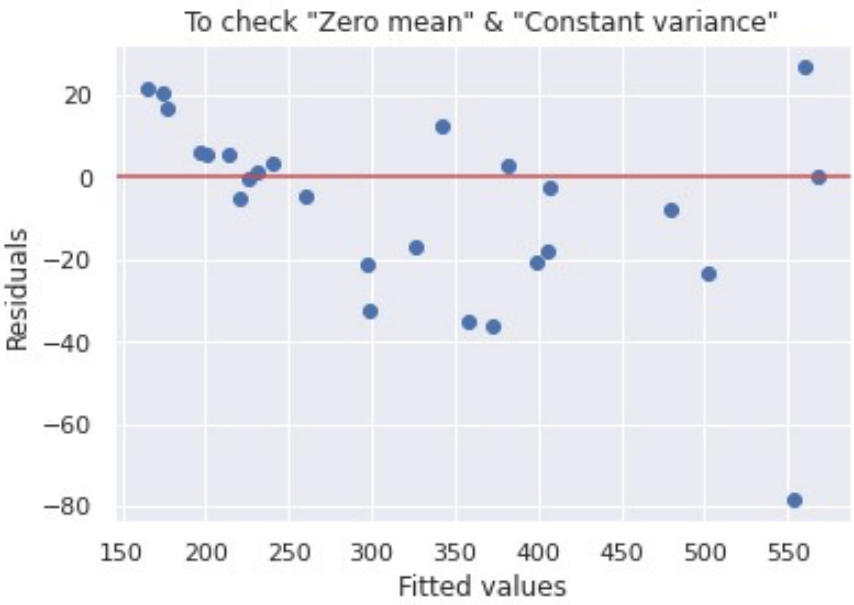
Model	**Hyperparameters**
Ridge Regression	Alpha = 1.0
Lasso Regression	Alpha = 1.0
Elastic Net	Alpha = 0.5
PCA	n_components = 4
KNN	n_neighbors = 3

Decision Tree	criterion = squared_error min_samples_split = 2
Boosting	n_estimators = 500 max_depth = 4 min_samples_split = 5 learning_rate = 0.01 loss function = squared_error
XGBoost	n_estimators. = 1000 max_depth = 7 eta. = 0.1
Neural Network	hidden_layer_sizes. = (5, 2) alpha = 1e-5 activation = relu
Random Forest	n_estimators = 100 criterion = squared_error min_samples_split. = 2 min_samples_leaf = 1

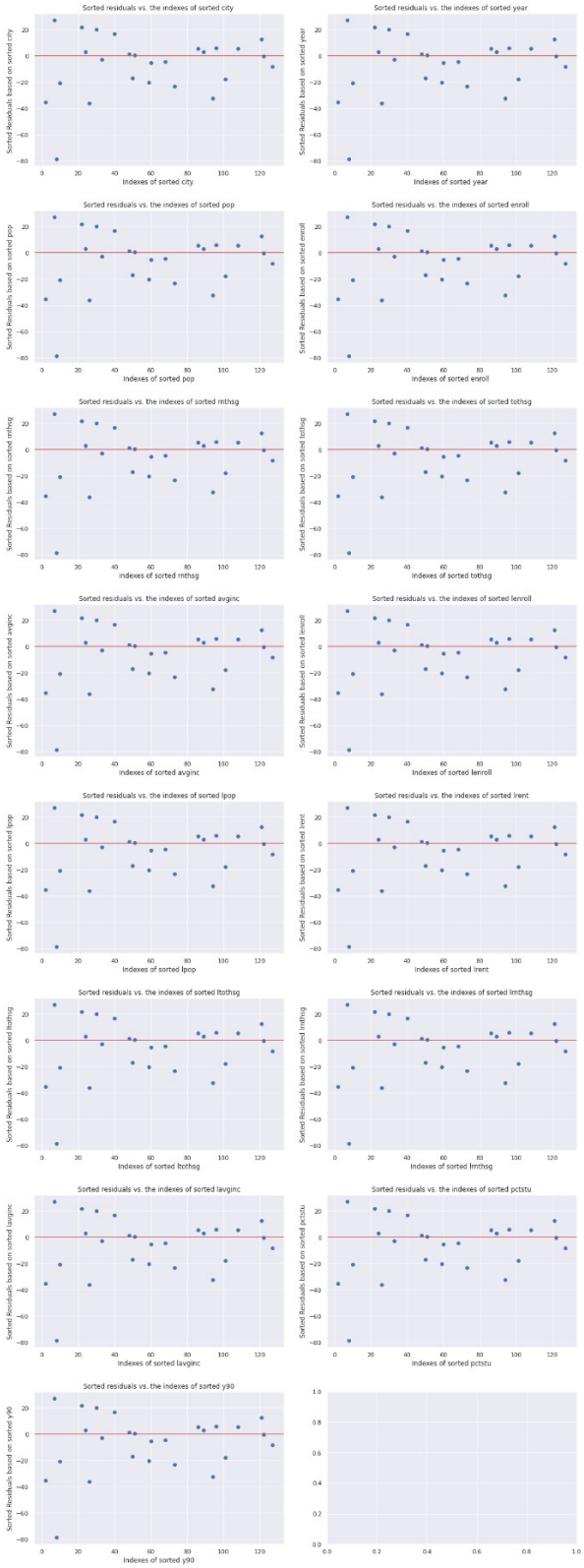
Appendix II: Linearity plot for House Rental dataset



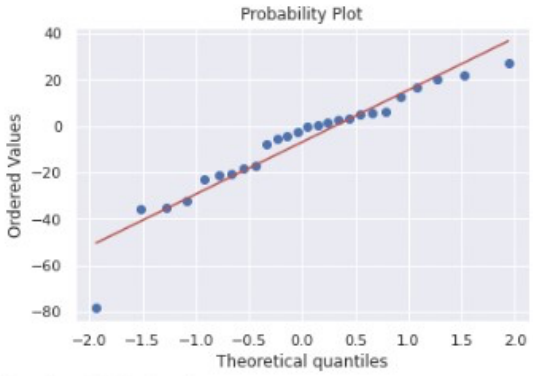
Appendix III: Homogeneity of Variance Assumption for House Rental dataset


To check "Zero mean" & "Constant variance"
Residuals
Fitted values
20
0
−20
−40
−60
−80
150
200
250
300
350
400
450
500
550

Appendix IV: Independence of Error Terms Assumption for House Rental dataset



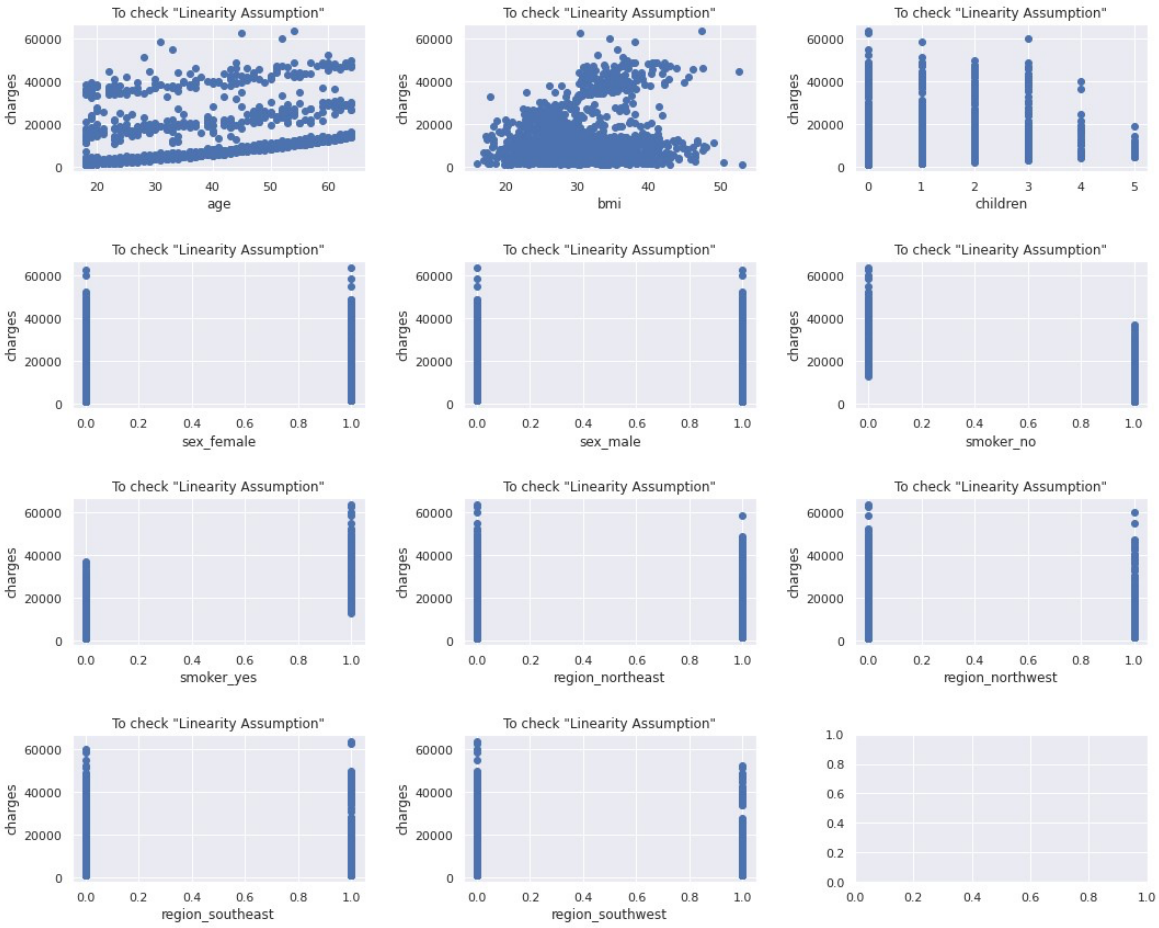
Appendix V: Normality probability for House Rental dataset



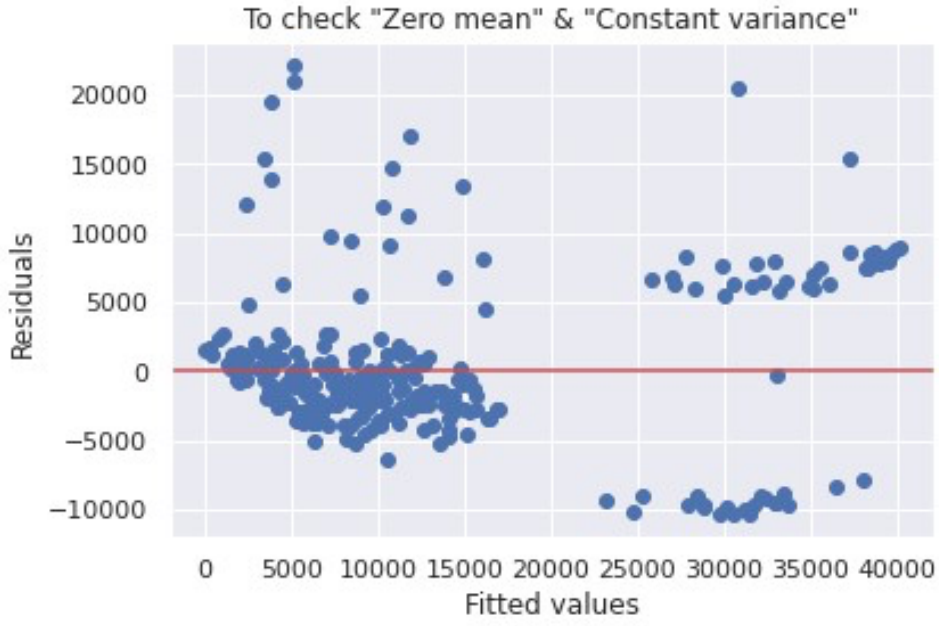
Shapiro–Wilk test:

1. The Shapiro–Wilk test is more appropriate method for small sample sizes (n<50).
2. ShapiroResult(statistic=0.9165416955947876, pvalue=0.0373322032392025)
3. As **p-value = 0.04** which is less than 0.05, **the normality assumption is rejected.**

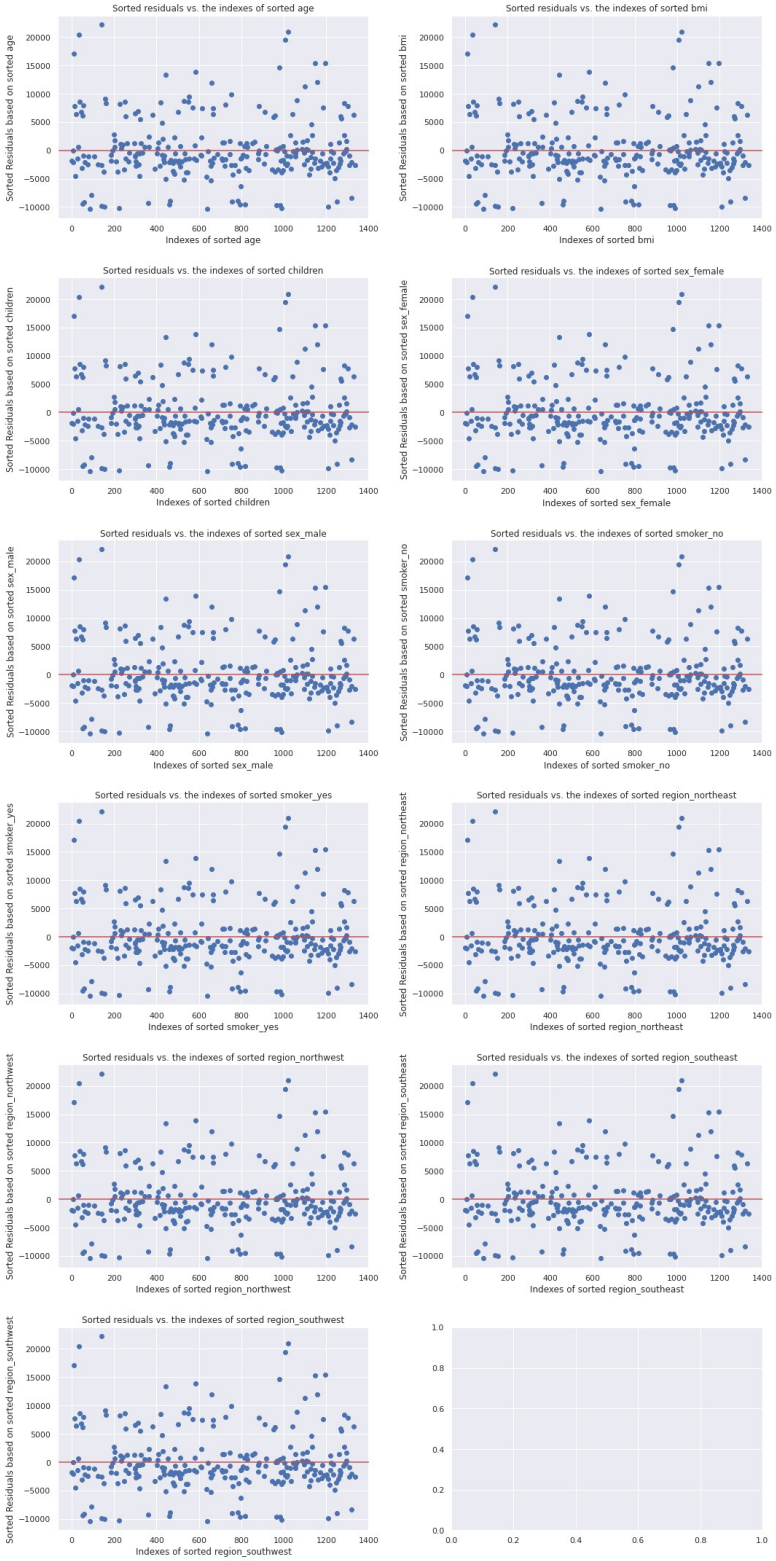
Appendix VI: Linearity plot for Medical Insurance dataset



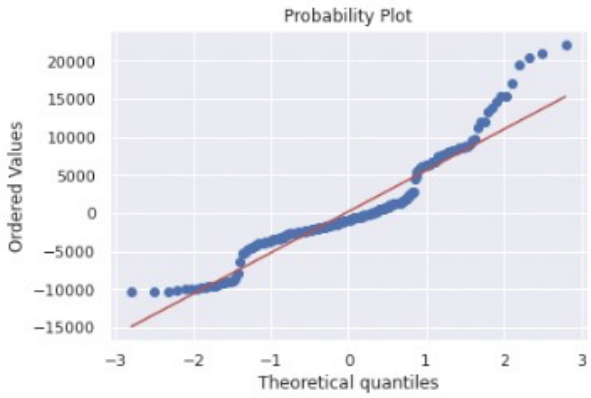
Appendix VII: Homogeneity of Variance Assumption for Medical Insurance dataset



Appendix VIII: Independence of Error Terms Assumption for Medical Insurance dataset



Appendix IX: Normality probability for Medical Insurance dataset



Shapiro–Wilk test:

1. The Shapiro–Wilk test is more appropriate method for small sample sizes (n<50).
2. ShapiroResult(statistic=0.9032657146453857, pvalue=4.221275672761093e-12)
3. As **p-value = 0.00** which is less than 0.05, **the normality assumption is rejected.**