Final Year Project Report

Full Unit - Final Report

Comparison of Ridge Regression Algorithm against Others When Solving Boston Housing Problem

Fabio Eugenio dos Santos de Sampaio Doria

A report submitted in part fulfilment of the degree of

BSc (Hons) in Computer Science

Supervisor: Nicolo Colombo

Department of Computer Science
Royal Holloway, University of London

Fabio Eugenio
dos Santos de Sampaio Doria

March 13, 2024

Declaration

This report has been prepared on the basis of my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count:

Student Name: Fabio Eugenio dos Santos de Sampaio Doria

Date of Submission:

Signature: Fabio Eugenio dos Santos de Sampaio Doria

Table of Contents

Abstract

For most people buying a house will be one of their most important and expensive economic decisions that they will take in their lives.[1] Because of this it would be logical to say that being able to accurately predict the prices of said houses would be of extreme value to people. One possible way to make these predictions would be to create a machine learning model that, given a certain amount of features from each house, would be able to create an accurate prediction of their price.[2] There are a wide range of different machine learning algorithms that could be used to solve this problem. However, this project will focus on two: Ridge Regression and Decision Trees.

In this project I plan on implementing both of these algorithms and comparing their performance on one another to see which one is more effective at resolving the Boston housing problem. To measure their accuracy I will use two metrics specifically used for regression problems, mean absolute error (MAE) and root mean squared error (RMSE). MAE returns the average residual of the predictions that each model makes, this is useful as it can be directly compared against the other model for differences.[3] However, RMSE returns the square root of the average residual of the predictions. This highlights larger errors caused by the model which is practical as it helps to differentiate the performance of both algorithms.[4]

The dataset that will be used to judge the different algorithms is called the Boston housing problem datasets. The dataset is comprised of 506 entries, each having a total of 14 features which describe a property inside of the Boston Massachusetts area.[5] Two variations of this dataset will be used, one with full 14 features and another with a lower number 5. Two models will be created, one with each algorithm and each of these models will be trained and tested on both of these datasets. Their performance will then be analysed, compared, and finally conclusions will be drawn from the results in order to define the effectiveness both algorithms.

Chapter 1: **Introduction**

Since humans began roaming the earth there have always been three main things that we have needed to survive: food, water, and shelter. In modern developed civilisations food and water has become relatively easy to come by with affordable versions of both being available to most people. However, shelter seems to only become more expensive with time while income stays the same. This can be seen in the UK where the median price of residential houses increased by 14% while income decreased by 1% from 2020 to 2021, making the issue of housing one of the most prevalent ones in recent times.[6] Also, With the advent of the internet and its capability of delivering large scale of information to users, the options of houses available are in-numerous. Because of this it can be an extremely overwhelming task to look for a suitable place to live that both fits the budget but is also reasonably priced considering the aspects and features of the house. House prices also play a significant role in the economy, one of these is being linked to consumer spending. If a homeowner knows that the value of his home increases then he feels confident and is likely to spend more in goods or services or to pay off their debts. If they know the value has decreased then the opposite occurs, homeowners become less confident meaning they are likely to spend less and save more.[7] Taking all of these factors into consideration it can be said that having an accurate method of estimating house prices is of importance from an individual looking for a home all the way to governments trying to predict what economic measures they should take next.

The first step in order to generate accurate estimates is to gather and analyse data on past houses which have been sold and try to find a pattern regarding their value. Specifically between the price they were sold for and features that describe the houses, such as the size of the house, area it is located in, and so forth. This analysis used to be done through traditional statistical methods, however, recently a new approach has emerged called machine learning. Here a machine is responsible for processing and learning from the data it is given in order to make predictions on new data autonomously.[8] When it comes to predicting house prices the machine learning model is given data consisting of past sales of houses where each house sold is also accompanied with a number of features that describe it as mentioned before. The system can then learn how these features influenced the price of the houses and use this information to make predictions on new houses based only on its corresponding features.[9]

Chapter 2: **Theoretical Background**

Within machine learning there are two main methodologies utilised when trying to solve a problem, supervised and unsupervised learning. Unsupervised learning is used when the data being analysed does not need to be labelled, but instead needs to be sorted into groups by their features. On the other hand, supervised learning is concerned with providing labels to unlabelled data in an dataset, such as a list of houses without a price attached to them.[10] Within supervised learning there is once again two different types of problems that occur: classification and regression problems. Classification problems occur when the list of possible classifications is finite such as identifying a handwritten digit as its correct number. However, in the Boston housing problem the list of possible labels is infinite as they could be any price, i.e., any real number.[11] This is called a regression problem and will be the focus of the research paper.

2.1 Simple & Multiple Linear Regression

Regression as a type of machine learning problem comes from the statistical method of regression, which has the goal of determining the relationship between a dependent and independent variable(s).[12] This is then expanded to create regression models which can be formally defined as using an independent variable 'x' to predict a dependent variable 'y'.[13] With the Boston housing problem the independent variable x are the features of each house such as, per capita crime rates per town, while the dependent variable y are the prices of the houses.[14]

2.1.1 Simple Linear Regression

There are many variations of the regression algorithm that can be used in a machine learning model, however simple linear regression is arguably the most simplistic ones and serves as a base for the others. The equation used in a simple linear regression model has the form:

$$y = \beta_0 + \beta_1 x + \epsilon$$

where y and x are the independent and dependent variable respectively. β_0 is the intercept of the line created by the equation with the y-axis, this is known as the constant term as it does not change. β_1 is the regression coefficient which defines the slope of the line. And lastly ϵ is the random error term or the random deviation which represents the difference between the predicted value and the real-life value.[15, 16]

The goal of this machine learning method is to fit the best possible line to the given data, which is achieved by finding the optimal weights of the regression coefficient β_1 and β_0, i.e., the parameters of the model. For this a cost function is used which calculates the difference between the predicted value and the expected values of every sample in the training dataset and returns it as a single real number.[17] This cost is then minimised by modifying the parameters, when the cost function cannot be further minimised then the most optimal coefficients have been achieved.[18] The most commonly used cost function for linear regression is the Sum of Squares Error (SSE) function

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

where y_i is the observed true label and $\hat{y}_i$ is the predicted label. The predicted label variable can also be substituted with the linear regression equation to get another version of the cost function.

$$SSE = \sum_{i=1}^{n} (y_i - \beta x_i)^2$$

2.1.2 Multiple Linear Regression

With the basic theoretical framework on regression being set out, it is now necessary to expand it in order to fit a practical use as a machine learning model that can be applied to real-world datasets. In reality, when creating a model their is most of the time going to be more than simply one feature that will define the outcome of the model. For this, multiple linear regression is used which has the same form as the previously seen linear regression but is extended by adding in more terms to account for the multiple features.[19]

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p + \epsilon$$

In this new equation there are now p dependent variables, for each of the features present in the data being used. Each dependent variable also has a their own β coefficient, which represents the change on y given a one unit change the coefficients x.[20] This is the equation describing the process of predicting one value y given p number(s) of dependent variables. However, this can be expanded to solve so that n number of predictions can be made given p dependent variables in a data set.

$$\begin{aligned}
y_1 &= \beta_0 + \beta_1 x_{11} + \beta_2 x_{12} + ... + \beta_p x_{1p} + \epsilon \\
y_2 &= \beta_0 + \beta_1 x_{21} + \beta_2 x_2 2 + ... + \beta_p x_2 p + \epsilon \\
&\vdots \\
y_n &= \beta_0 + \beta_1 x_{n1} + \beta_2 x_{n2} + ... + \beta_p x_{np} + \epsilon
\end{aligned}$$

During this report the matrix notation will be used primarily when it comes to manipulating and solving equations. Therefore, the system of equations above describing multiple linear regression can also be displayed in this form which is much easier to work with with and to visualise. It is written as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & ... & x_{1p} \\ 1 & x_{21} & x_{22} & ... & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & ... & x_{np} \end{bmatrix}, \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix},$$

where $\boldsymbol{y}$ is a vector of dimensions $n \times 1$ with n being the number of predictions being made. $\boldsymbol{y}$ is a matrix of dimensions $n \times p$ where p is the number of dependent variables. The row of this matrix are made up of the values of the data sample, with each column in the row being a different feature for the given sample. Its first column is filled with ones so that β_0 is always present in the equations after the matrix multiplication. $\boldsymbol{\beta}$ is a vector of dimensions $p \times 1$ that holds all the coefficients for the dependent variables. It is also important to note that

features can easily be added or removed from this model, this will be used when comparing the performance of the two algorithms on datasets with different numbers of features.

The cost function that was discussed earlier can also be put into matrix notation which will make the task of minimizing it much easier. It can be written as

$$SSE = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})$$

In order to square the residuals before summing them, the vector transpose is multiplied by the vector containing the residuals. This will first multiply each residual with itself squaring them, before they are all added up.

2.2 Ridge Regression

The Ridge Regression method is a different way of estimating the parameters for a multiple linear regression model, specifically in order to deal with issues that can arise from using data that suffers from multicollinearity.[21] Before looking at how Ridge Regression addresses this issue it is first necessary to understand what multicollinearity and the effects that it can have on a linear regression model.

2.2.1 Issues Caused by Multicollinearity

As said before, Linear Regression models utilise the Sum of Squares Error method to estimate the value of the regression coefficients. However, the reliability of these coefficient estimations could be severely impacted if the data being used to train the model suffers from multicollinearity, when two or more independent variables in the model are correlated.[22] This impacts the reliability of the estimations as linear regression models define the regression coefficient to represent a mean change in the dependant variable for each unit change in the independent variable when all other independent variables remain the same, and by definition this cannot occur if two or more independent variables are correlated.[23] This multicollinearity causes the model to become unstable because a small change in one variable will suddenly effect another causing a big change in the prediction. It also makes the model harder to interpret as the regression coefficients do not necessarily reflect the significance of only that specific feature.[24] Finally, it can lead to overfitting which is when a model displays a high level of accuracy on the training dataset, but performs badly when it comes to the testing set. This happens because the model is trained to specifically to the individual data points in the training set instead of the general trend of the points.Then when it tries to predict a data sample not from that set it accuracy is poor.[25]

2.2.2 Ridge Regression Solution

The Ridge Regression algorithm solves these issues caused by multicollinearity by applying a processes to the model called regularisation, specifically L2 Regularisation. Regularisation is the process of 'regularising' the regression coefficients by trying to make them as small as they need to be.[26] The effect this has on the model is of generalisation, meaning that it wont be as accurate with its predictions in the training set, but it will work better in the long run with predictions on new unseen data. In more precise terms it can be said that a small amount of bias is added into the model which in return decreases the amount of variance

which causing the poor performance.[27]

L2 regularisation achieves this by enhancing the SSE equation by adding a penalty term to the end of it. This term is the summation of squared weights of each feature, multiplied by 'a' that defines how 'harsh' the penalty term should be. Finding the optimum value for 'a' is critical as it defines the overall performance of the equation, if it is zero then the model once again uses just SSE and the higher 'a' is the more generalised and less accurate the model will be.[28]

$$SSE + a||\beta||^2 = \sum_{i=1}^{n}(y_i - \beta x_i)^2 \ + a\sum_{j=1}^{P}\beta_j^2.$$

This equation can be represented in matrix notation aswell,

$$SSE_{ridge} = (\boldsymbol{y} - \boldsymbol{X\beta})^T(\boldsymbol{y} - \boldsymbol{X\beta}) + a(\boldsymbol{\beta^T\beta})$$

This equation is then minimised and with its β values being the most optimal coefficients that will be used to help prevent overfitting in the model.

2.2.3 Deriving Ridge Estimator

In order to find the best regression coefficients for our model, we first need to derive the Ridge Regression estimator of β. This equation can then be solved for β which will yield the most optimal coefficients for the model. This estimator will then be implemented into the program in the python programming language, where it will be used to estimate the most optimal regression coefficients for any data set given to the model. In this paper the estimator is derived from the aforementioned L2 regularisation function, specifically in its matrix notation

$$(\boldsymbol{y} - \boldsymbol{X\beta})^T(\boldsymbol{y} - \boldsymbol{X\beta}) + a(\boldsymbol{\beta^T\beta})$$

This equation first needs to be expanded before it can be solved for β

$$\boldsymbol{y^Ty} - \boldsymbol{y^TX\beta} - \boldsymbol{\beta^TX^Ty} + \boldsymbol{\beta^TX^TX\beta} + a(\boldsymbol{\beta^T\beta})$$

Here the terms $\boldsymbol{y^TX\beta}$ and $\boldsymbol{\beta^TX^Ty}$ are both equal to each other, following the rules of transpose, and also a scalar as this is the result of the dot product between its elements. Therefore,

$$\boldsymbol{y^Ty} - 2\boldsymbol{\beta^TX^Ty} + \boldsymbol{\beta^TX^TX\beta} + a(\boldsymbol{\beta^T\beta})$$

To find the point where this function has its lowest value we need to take its derivative with respect to β, as this will show us how the value output by the cost function changes with respect to β

$$\frac{\mathrm{d}(SSE_{ridge})}{\mathrm{d}\beta} = -2X^Ty + 2X^TX\beta + 2a\beta$$

This differential equation can now be set to zero and solved for β, giving us our ridge estimator for the model.

$$0 = -2X^T y + 2X^T X\beta + 2a\beta$$

$$0 = -X^T y + X^T X\beta + a\beta$$

$$X^T y = X^T X\beta + a\beta$$

$$X^T y = \beta(X^T X + a\boldsymbol{I})$$

$$\beta = (X^T X + a\boldsymbol{I})^{-1} X^T y$$

This equation can now be solved for β using the information available from a training set, giving us the optimal regression coefficients for our ridge regression model. Where the rows of matrix $\boldsymbol{X}$ are the data samples of the training set and its columns being the values of the features corresponding to each sample. And vector y containing the labels of the corresponding data samples.

Chapter 3: **Method**

3.1 Data Selection

In order to gather data on the performance of the different machine learning algorithms we first need to choose the datasets that they will be trained on. To ensure that their performance is fully analysed, each algorithm will be trained on 3 different datasets so that they can be analysed under a wide range of conditions. The idea is to check if there is a difference in performance from the algorithms when they are trained on datasets with different quantities of features. To try and observe this difference each dataset will have approximately 10, 50, and 100 features in order to capture any difference in performance. The 3 datasets chosen for this report are the Boston Housing Dataset with 13 features[29], Facebook Comment Volume Dataset with 53 features[30], and the Million Song Database with 90 features[31].

3.2 Data Analysis

3.3 Data Preprocessing

3.4 Accuracy Metrics

An important part of creating a machine learning model is having a means of objectively measuring the accuracy of the predictions made by the model. This is necessary as without this metric it would be challenging to evaluate the effectiveness of the model, making it difficult to implement meaningful modifications and observe if these changes have any positive effect. There are different metrics for measuring the accuracy of a model predicting on regression versus classification problems, as in classification problems the prediction is either correct or incorrect while in regression problems a numerical value is being predicted which can be closer or farther from the target value.

3.4.1 R^2 Score

For this report I have chosen to use the R-squared (R^2) statistical measure as a means to determine the goodness of fit of my Ridge Regression model. The R^2 score describes the variability in the labels predicted by the model, based on the features that were used to train it. This score is represented as a number between 0 and 1. A score of 0 describes a model that has not learnt any relationship between the features and the labels of the dataset, while a score of 1 means that the features perfectly describe the value of the labels. Any score x between 0 and 1 describe a variation of $(x \times 100)\%$ in the predicted labels given the features used to train the model.[32]

When evaluating the performance of the models a higher R^2 score will demonstrate a more accurate and higher performing model. It is important to note that the R^2 score will be computed on the predictions made from the test set and not the training set. This is done

in order to analyse how well the models perform on unseen data, as the main point of Ridge Regression is to generalize the model making it more effective on new data which the model has not been trained on. This score value will be the main method to objectively analyse and compare the effectiveness of the Ridge Regression and K Nearest Neighbours models on each dataset.

To ensure that the scores are statistically valid I will compute them 10 times on each dataset for each model. Each time the random state of the train_test_split() method will be changed, resulting in different data samples within the training and test sets. The mean of these score will then be taken along with their standard deviation in order to create a general score along with uncertainties for the results.

3.4.2 Result Visualisation

Chapter 4: **Experiment**

4.1 Ridge Regression

4.2 K Nearest Neighbours

Chapter 5: **Results**

Chapter 6: **Proof of Concept**

With this background research it is now possible to implement these algorithms as a program from scratch that can be run on a dataset to return real predictions. However, before committing to a full size project I believe it would be beneficial to create a proof of concept (POC) program where the algorithm can be tested on a smaller scale. There are many reasons for doing this, one of them is that a POC program can give insight into the feasibility of the project. If it shows very poor results than it might be better to go back to the planning stage and try to restructure the project and tackle it from another angle. Another reason is that when creating the POC program it is easier to identify any challenges that can arise during implementation. This is valuable as its easier to deal with these challenges when working on a smaller scale, solving these issues now also means that it will be less complicated to deal with them again when they appear in the full implementation. It also creates improved documentation as the issues can be clearly explained and written down, allowing others to have a better chance of re-creating the results of the report. Finally, implementing a POC before the final program is a great way of starting the process of getting to that final stage. It also makes the project aims a lot clearer as you can branch of into many different areas after having achieved the POC such as building a GUI, implementing optimisation processes, developing data visualisation capabilities. This would be much harder if the main algorithm still had not been proven to work.

6.1 Choosing the Dataset

As mentioned before the aim of the proof of concept program is to start small and simple, because of this I have chosen to not utilise the Boston housing dataset for this program. This is due to it having 13 different features and over 500 entries, not only does this mean there is a lot more space for things to go wrong but it also that it is quite unfeasible to show a written example due to the number of variables and entries. Therefore, I have decided to use a dataset that is available through the sklearn library which is called the Linnerud dataset.[33] The dataset was collected from twenty middle-aged men at a fitness club and consists of three features which are the number of reps completed for a given exercise: sit-ups, chins, and jumps. This dataset is multi-output which means in this case that it has 3 different dependent variables: weight, waist, and pulse but for this proof of concept the waist label was chosen to be the dependant variable as it seemed to have a stronger linear relationship compared to the other two features.

6.2 Implementing Ridge Regression

I would consider this to be the most challenging part of the proof of concept as not only is the understanding of the principles behind Ridge Regression needed but also the mathematical knowledge of how to optimise the coefficients and the technical ability to translate it into functioning code. After my research, I decided that the best way, at this stage of the project, to optimise the cost function is to represent the multiple linear regression line in a matrix form.

$$\hat{y} = X\hat{\beta} + \epsilon$$

Where, $\hat{y}$ is a vector containing the predictions made, X is a matrix containing the independent variables, and $\hat{\beta}$ is another vector that contains all the coefficients corresponding to

the values in X^T. By using this form it is possible to create an equation, through algebraic manipulation and calculus, where we can solve for $\hat{\beta}$ which will give us all of the coefficients where the SSE_{L2} function is the smallest.[34]

$$\hat{\beta}(a) = (X^T X + aI)^{-1} X^T y$$

This equation can be easily solved using python and the numpy library as we have all the values need. The following snippet of code is how this equations looks in my program, it sits inside of the fit method and is used when fitting the model to the training set.

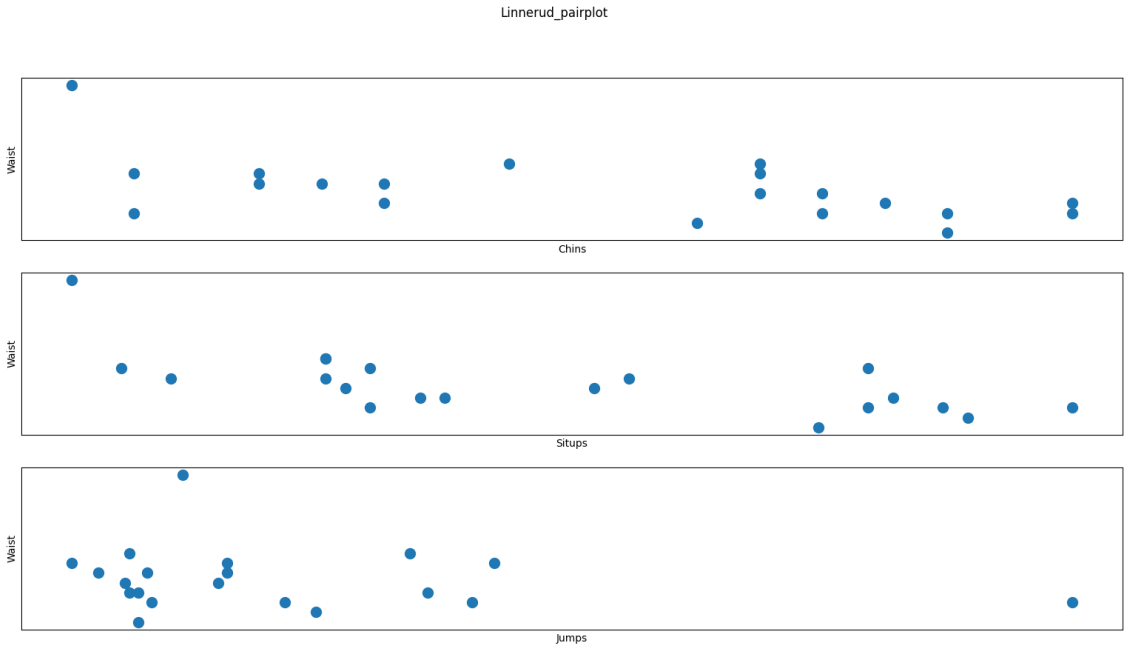
```
def fit(self, X_train, y_train):
    """Fits the regression model to the training data."""
    self.X_train = X_train
    self.y_train = y_train

    # Stores sizes of the training set matrix
    self.m = X_train.shape[0]
    self.n = X_train.shape[1]

    # Identity matrix needed for beta_ridge_hat computation
    I = np.identity(self.n)

    self.beta_ridge_hat = ((inv((self.X_train.T).dot(self.X_train)
    + self.penalty * I)).dot(self.X_train.T)).dot(y_train)
```

Now that the optimisation equation has been completed all that is needed is to create a predict method that takes a set of data points, runs the multiple regression formula with the optimised coefficients, and returns its own predictions. This part was relatively easy to do and I made sure to keep it in a general format so that the same code could be reused to work with a dataset that has a different number of features.

6.3 Data Visualisation

Even though the main focus of this report is the comparison of machine learning algorithms, data visualisation is essential in order to make smart decision regarding what data we are using and if an algorithm is even appropriate or not for a given situation. Because of this it imperative that this analysis is done first, before committing to an algorithm or dataset in order to improve the chances of success.[35]

The first thing I did was to plot out each of the independent variables against the dependent variable to have a better understanding of how the data was related to each other.

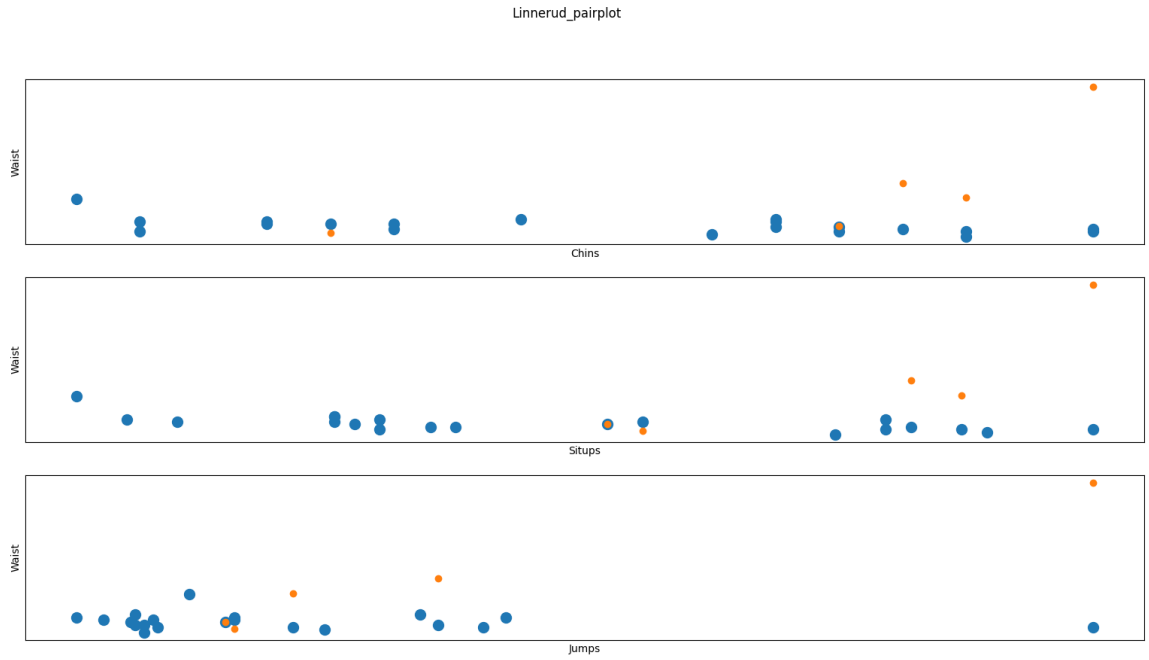


In these 3 graphs we can see that there does seem to be a linear relationship between the features and the dependent variable. However we do also see some outliers in the data which can have a negative affect on the outcome of the model. These can be reduced after applying some preprocessing methods to the data before applying our machine learning algorithms. However, for now this is sufficient information for me too continue the approach of using the Ridge Regression algorithm on this dataset for the proof of concept.

6.4 Running Program

After having written the classifier classes, importing the data, separating it, and visualising it we can finally apply the algorithm to it and see if the results appear accurate. First thing is that the data must be separated into both our training and test sets. The training set will be used to actually train the ridge regression model, i.e., contains the data points that we will fit our line to. The test set is what will be used to test our model and see how accurate its predictions are. It is very important that when training the algorithm no information from the test set is used and vice versa, this is called data snooping and can negatively impact the model.

After the separation has been made the Ridge Regression classifier is instantiated, it is then fitted to the training data and ready to make predictions. The model is then run on the test set where it makes a prediction for each sample in it. I then printed out both the predictions given by the model and the real labels of the test set. Finally, I once again plotted the features against the dependent variable but this time I plotted on top of theses graphs the predictions made by the model to get a better understanding of how the model was performing.



After taking a look at the plotted results I can say that I am reasonably satisfied with the results, especially considering that this was the proof of concept. Almost all of the predictions were reasonably close to their true values, except for the data point that is farthest away and had very poor accuracy. However, I believe that this may have happened due to not having done enough data preprocessing before applying the model. We can see from the graphs that the points tend to grow higher the more towards the right we move. With more preprocessing the data points might be closer together and more evenly spaced which could fix this issue. Also trying different values for the penalty term a might have a positive effect on the predictions as it would flatten the line out more, as it would help with the over prediction closer to the right end of the graphs.

Bibliography

[1] L. R. Weinstock, "Introduction to u.s. economy: Housing market," *Congressional Research Service*, Jan. 2023.

[2] P. Herman, "The importance of price prediction," *Future Processing*, Mar. 2023.

[3] P. Schneider and F. Xhafa, "Chapter 3 - anomaly detection: Concepts and methods," in *Anomaly Detection and Complex Event Processing over IoT Data Streams* (P. Schneider and F. Xhafa, eds.), pp. 49–66, Academic Press, 2022.

[4] S. Olumide, "Root mean square error (rmse): What you need to know," *Arize*, Aug 2023.

[5] V. Roman, "Root mean square error (rmse): What you need to know," *Towards Data Science*, Jan. 2019.

[6] C. Smith, "Housing affordability in england and wales: 2021," *Census 2021*, Mar. 2022.

[7] "Housing affordability in england and wales: 2021," *Bank of England*, Mar. 2020.

[8] M. Chatterjeeh, "Data science vs machine learning and artificial intelligence: The difference explained (2024)," *Great Learning*, Nov. 2023.

[9] I. Ake, "Combining machine learning models to predict house prices," *Solent University*, Sep. 2022.

[10] V. Kanade, "What is machine learning? definition, types, applications, and trends for 2022," *Spice Works*, Aug. 2022.

[11] V. Vovk, "Chapter 2: Introduction to machine learning and nearest neighbours." `https://moodle.royalholloway.ac.uk/pluginfile.php/188746/mod_resource/content/27/02_1.pdf`, Sep. 2023.

[12] B. Beers, "What is regression? definition, calculation, and example," *investopedia*, Mar. 2023.

[13] D. Maulud and A. M. Abdulazeez, "A review on linear regression comprehensive in machine learning," *Journal of Applied Science and Technology Trends*, vol. 1, pp. 140–147, Dec. 2020.

[14] S. Gupta, "Boston house price prediction based using support vector regressor," *enjoy algorithms*.

[15] S. Rong and Z. Bao-Wen, "The research of regression model in machine learning field," in *MATEC Web of Conferences*, vol. 176, p. 01033, EDP Sciences, 2018.

[16] S. Glen, "Error term: Definition and examples," *Statistics How To*, Nov. 2020.

[17] K. Krzyk, "Cost function of linear regression: Deep learning for beginners," *Built In*, Jul. 2022.

[18] S.-J. Kim, S.-J. Bae, and M.-W. Jang, "Linear regression machine learning algorithms for estimating reference evapotranspiration using limited climate data," *Sustainability*, vol. 14, no. 18, p. 11674, 2022.

[19] M. Tranmer and M. Elliot, "Multiple linear regression," *The Cathie Marsh Centre for Census and Survey Research (CCSR)*, vol. 5, no. 5, pp. 10–11, 2008.

[20] S. Taylor, "Multiple linear regression," *Data Science*, Apr. 2020.

[21] G. C. McDonald, "Ridge regression," *WIREs Computational Statistics*, vol. 1, no. 1, pp. 93–100, 2009.

[22] A. Alin, "Multicollinearity," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 3, pp. 370–374, 2010.

[23] J. Frost, "Multicollinearity in regression analysis: Problems, detection, and solutions," *Statistics By Jim*, 2017.

[24] S. Wu, "Multicollinearity in regression," *towardsdatascience*, May. 2020.

[25] J. Nagidi, "How to handle overfitting in deep learning models," *Dataaspirant*, Aug. 2020.

[26] P. Gupta, "Regularization in machine learning," *Towards Data Science*, Nov. 2017.

[27] C. Maklin, "Machine learning algorithms part 11: Ridge regression, lasso regression and elastic-net regression," *Medium*, Dec. 2018.

[28] K. Kargin, "Ridge regression fundamentals and modeling in python," *Medium*, Apr. 2021.

[29] D. Harrison and D. L. Rubinfeld, "Hedonic housing prices and the demand for clean air," *Journal of Environmental Economics and Management*, vol. 5, no. 1, pp. 81–102, 1978.

[30] K. Singh, D. Kumar, and R. Kaur, "Comment volume prediction using neural networks and decision trees," 03 2015.

[31] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[32] D. Jain, "R-squared in regression analysis in machine learning," *Geeks for Geeks*, May. 2023.

[33] A. Bajwa, "What is datasets load$_l innerud()insklearn?,''$ *educative*.

[34] M. Levine, "Statistics 512: Applied linear models," *Purdue University*, 2017.

[35] E. B. Kate Brush, "Data visualization," *Tech Target*, Dec. 2022.