Contents

1 Introduction

Linear regression is widely used in many applications to model linear relationships between a set of predictors and a response. One of the more common and simpler methods of fitting such a model is known as the ordinary least squares method (OLS), which is found to be a Best Linear Unbiased Estimator (BLUE) by the Gauss-Markov theorem, that is, it gives the unbiased estimates of each regression coefficient with the lowest possible variance, provided that assumptions are satisfied. Furthermore, it has a main advantage of it being easily interpretable as a parametric model as opposed to non-parametric models.

The motivation for this project arises from one of these assumptions, that is, the data matrix, $\boldsymbol{X}$, is full rank, i.e. none of the predictors are (nearly) perfectly correlated with each other. In cases where this is not true, we find that OLS fails to acquire any results. In such cases, there are several methods we can employ to address this issue, whether to drop one of the correlated variables, or to combine them, etc., based on the discretion of the data analyst. In this project, we investigate how two of the commonly used software for regression analysis, Python (`sklearn` and `statsmodel`) and R each deal with the collinearity problem.

Furthermore, we embrace this opportunity to explore further on the topic of multicollinearity, such as its impact on the accuracy of model prediction and model interpretation. We have also identified several known methods of detecting collinearity in the data, whether through hypothesis testing, or by using measures such as the variance inflation factor (VIF) and condition numbers/indices. In addition to aforementioned methods of addressing collinearity, shrinkage methods such as ridge regression and lasso were also considered to alleviate the impact of overfitting (due to collinearity) on variance at the cost of negligible bias in order to obtain an improved model for prediction. Another potential pitfall in regression analysis, that is confounding variables, were also briefly explored by examining examples of Simpson's paradox.

It should be noted that there are different conventions to what collinearity and multicollinearity mean according to varying sources. In this report, we will use these two terms interchangably.

2 Objectives

There are three main objectives for this project:

2.1 Explore issues that arise from singularity.

In practical settings, it is unlikely to obtain a set of perfectly correlated data due to inevitable noise. However, even near perfect collinearity can lead to singularity and subsequently the failure of OLS when employing the use of software. This project aims to explore how a rank-deficient data matrix can lead to singularity, how (nearly) perfect collinearity can lead to the failure of OLS, and the pitfalls caused by moderate to high multicollinearity in general in terms of model interpretation, model prediction, and model validity. Subsequently, we look into commonly used methods to handle situations where regressors are highly correlated depending on purpose of analysis.

2.2 Understand singular value decomposition (SVD).

The least squares method estimates the regression coefficients of a model with p predictors by fitting it to a dataset with n observations using the following equation

$$\beta = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y},$$

where β is a $p \times 1$ column matrix, $\boldsymbol{X}$ is a $n \times p$ data matrix, and $\boldsymbol{y}$ is a $n \times 1$ column matrix containing the observed response values. With the presence of perfect collinearity, the matrix $\boldsymbol{X}^T\boldsymbol{X}$ is singular and hence $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$ does not exist. However, the least squares solver of Python uses the Moore-Penrose inverse (or pseudoinverse) which gives the best approximation of $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$, and can be easily computed using the SVD of $\boldsymbol{X}^T\boldsymbol{X}$.

2.3 Investigate how software deals with singularity when building linear models.

It is observed that there exists discrepancy between how different software handles highly correlated predictors when fitting a multiple linear regression model via OLS. In this project, we aim to understand the reason behind this difference in behaviour by referring to standard documentation and examining program output.

3 Methodology

We narrow our focus to three least squares solvers of interests: Python's `scikit-learn` library, Python's `statsmodel` library, and R's `lm` function. In each case, we have randomly generated several sets of artificial data: one set using two correlated regressors and one set using three regressors where only one pair is correlated, each with 100 samples. The degree of collinearity is varied between high and perfect to observe differences in output. The pseudoinverse of the data matrix is also computed and multiplied with the generated response to give the theoretical minimum norm solution for the coefficient estimates and the results are compared. The use of artificial data has the advantage where the true relationship between the response and predictors is known and self-determined and that the degree of correlation and noise in the data can be manually specified.

We also performed regression analysis on built-in datasets such as California housing and diabetes dataset in `scikit-learn` and tried fitting highly correlated models. Subsequently, ridge regression models and lasso models were also used to examine its effectiveness in increasing the R^2 score of these models.

4 Results and discussion

4.1 Python's `scikit-learn` library

4.2 Python's `statsmodel` library

(include screenshot of results)

- Looks like it evenly distributes weights throughout the correlated predictors.
- Apparently uses pseudoinverse, and that the results obtained don't carry meaning.

4.3 R's `lm` function

(include screenshot of results)

- Looks like it dropped one collinear column and add the two coefficients together.
- Seems to combine collinear columns.
- Gives a warning when highly correlated variables are detected
- Option available to abort process upon detection of high mutlicollinearity.