Weightlifting MLR
Multiple Linear Regression using R and Python

Braden Heuglin, bheuglin2@bellarmine.edu
Ben Griffis, bgriffis@bellarmine.edu

ABSTRACT

We are using Python and R Studio to apply multiple linear regression to a dataset we found. The textbook definition of multiple linear regression is a regression model that estimates the relationship between a quantitative dependent variable and two or more independent variables using a straight line. The dataset contains information about a workout program, specifically, the exercises, sets, reps, and weights used.

I. INTRODUCTION

We used a weightlifting dataset for our project. The dataset contains all the details of a man's workout history for 3 years. It includes the date, workout name, exercise name, set order, weight, reps, distance, seconds, general notes, and workout notes. The person who collected the data did not do so with a purpose in mind, he simply wanted to track his progress and asked if anyone could find any trends. We used many elements of MLR

One or two-paragraph introduction to your project in which you briefly describe the data set you are working with and the ML Regression model you chose to apply to it.

II. BACKGROUND

A. Data Set Description

We found the dataset on Kaggle, a website which hosts many various datasets for people to download and work with. The dataset we used was collected by user Joe89, who collected this data with no purpose in mind. He says, "Below are my recorded workouts going back almost 3 years, using the Strong app. Nearly every single movement I have performed in the gym is recorded here with the exception of some warmup sets. It would be interesting if anyone could find any useful patterns, surprisingly insights, or tips for getting stronger."

B. Machine Learning Model

The textbook definition of multiple linear regression is a regression model that estimates the relationship between a quantitative dependent variable and two or more independent variables using a straight line. It is the comparison between two variables to check how connected they are.

III. EXPLORATORY ANALYSIS

This data set contains 9933 samples with 10 columns with both continuous and discrete data types. None of the values of the dataset contained missing values. Due to this, plotting of the variables were not necessary to investigate.

Table 1: Data Types

Variable Name	*Data Type*
Date	Categorical: Object
Workout Name	Categorical: Object
Exercise Name	Categorical: Object
Set Order	Categorical: Int64
Weight	Quantitative: Float64
Reps	Quantitative: Int64
Distance	Quantitative: Float64
Seconds	Quantitative: Int64
Notes	Categorical: Object
Workout Notes	Categorical: Object

IV. METHODS

A. Data Preparation

The original data set had 10 columns. These columns in context were Date, Workout name, Exercise name, Set Order, Weight, Reps, Distance, Seconds, Notes, and Workout Notes. None of the columns needed to be deleted, and there appeared to be no errors in the data entry for each column. Likewise, there were no missing values so there was no need to input values using averages/mean etc.

B. Experimental Design

You will run your model several times with different parameters to see what different results you get. In a table, describe your experimental parameters. Three or four experiments are sufficient. This is where you will describe how you divided your data into train, validate and test data sets. For example:

Table X: Experiment Parameters

Experiment Number	Parameters
1	80/20 split for train and test
2	85/15 split for train and test
3	75/25 split for train and test

C. Tools Used

The following tools were used for this analysis:

- R libraries:
  - Catools allowed the dataset to be split into training and testing sets so that the data could be trained for the MLR model and then tested to investigate strength.
  - Tidy verse allowed team members to import the dataset into R studio for EDA and MLR analysis.
- Python libraries:
  - Pandas allowed team members to import the data set and to convert the categorical variables into a numeric pattern
  - Sklearn was the major tool used for MLR. It split the data into the training, testing, and validation set. It also set up the model for linear regression and helped with built in calculations for MSE and R-squared.
  - NumPy was used for stacking the validation results.

V. RESULTS

A. Mean square Error and R-Square calculation

Explain the MSE and R square and discuss the results using relevant formulas.

Mean Square Error (MSE) is a calculation that tells you how close the regression line is to a set of points. The formula is calculated by taking the difference between the actual value and the predicted value, squaring it, and taking the average of all those squares. Because this dataset is so large and the values are spread so wide, the MSE is very big. The R-squared value determines the variance in the dependent variable, but overall is another determination of fit. For the dataset, the R-squared value was about 86%, indicating that the regression was a good model for the dataset.

B. Discussion of Results

Besides having different splits for the training and test sets we found no difference between the R-Squared values and the MSE values. The R-Squared value was 0.8619 which would be considered a “good fit” for most datasets, and the MSE value was 1923.08 which doesn’t mean its not’s not a good dataset it just means it’s far from being a perfect model. This may be because of how many data points there are in this entire set being close to 10,000

C. Problems Encountered

One of the problems that was encountered came to light when exploring the dataset in R. When exploring the testing sets a warning message occur saying “Warning message: In predict.lm(MLR, newdata = testing_set) : prediction from a rank-deficient fit may be misleading”. Not exactly sure on the meaning of that message or how it affected the results, but it lowered our confidence on our final conclusions.

D. Limitations of Implementation

Our model may not be the best way to model our data because a poorly specified model or weak relationship between the variables may result in a high MSE (which we have) even with a large sample size.

E. Improvements/Future Work

For future work I’d look into comparing multiple independent variables to one another rather than looking specifically at one dependent variable like “Weight” which is what was explored in this example.

VI. CONCLUSION

In conclusion, the model deemed to be successful. Despite a large MSE, the R-squared provided confidence in that the model and was pretty fitting. We used R studio and Python to explore, split, and train the data for the model. Each experiment had different parameters with different splits for training and testing, but regardless of that the results remained the same indicating a consistently good fit.