**Fitness vs Background**

Name, [aridley@bellarmine.edu](mailto:aridley@bellarmine.edu)

**ABSTRACT**

**In this project performed logistic regression on an cardio fitness dataset using jupyter notebook in python. The dataset consists of 180 entries with 9 total columns and contains various types of variables such as categorical. I will use this dataset to test a prediction based on the independent and dependent variables I choose. I will also use the data to make a confusion matrix and a classification report to compare the validation to see if the model was accurate. I will also point out any interesting data.**

1. **INTRODUCTION**

For this project I will use logistic regression in python to predict how fit people are based on their life circumstances such as age, education and martial status. I will use 3 different experimental parameters on the train, test and validation set and compare the three to see which parameter is the most accurate. I will also predict a new validation set based on the background in the dataset and compare hoe accurate it is as well.

1. **BACKGROUND**
   1. *Data Set Description*

Because I used logistic regression, I wanted to find a dataset that had a reasonable amount of data to work with that included different variables Kaggle, I came across a fitness dataset that had exactly what I was looking for. I chose to use this dataset because it had multiple columns with categorical data, and I can also relate to this dataset because I am now starting to go to the gym myself.

* 1. *Machine Learning Model*

**Logistic regression is a machine learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable, meaning 0 or 1, that contains coded data. It can also be used as a data analysis technique that uses mathematics to find the relationship between to data factors using x and y variables.**

**In this project I used the logistic regression to analyze the dataset to see if there were any missing values and split the datatset into dependent and independent variables. This helped to split the dataset into the training and test set, where I distributed the different experimental parameters. Using this, I was able to make a prediction validation set where I can predict a value based on the information I gave.**

1. **EXPLORATORY ANALYSIS**

My datatset has 180 entries and 9 total columns. There are no missing values

**Table 1: Data Types**

|  |  |
| --- | --- |
| **Variable Name** | **Data Type** |
| **product** | **object** |
| **age** | **Int 64** |
| **gender** | **object** |
| **education** | **Int 64** |
| **Martial status** | **object** |

|  |  |
| --- | --- |
| *Variable Name* | *Data Type* |
| usage | Int 64 |
| fitness | Int 64 |
| income | Int 64 |
| miles | Int 64 |

1. **METHODS**

To prepare this data, I had to import the libraries that were needed to analyze the data and import the dataset. I then used the csv file to find any basic information and view the first 5 columns of the dataset. Next, I used the isnull( ).sum( ) function to find any missing values, but fortunately there were none. Next, I split the dataset into dependent and independent variables using x and y. Next, I split the dataset into 3 different parameters using the test and training set, then I predicated a new validation set. Finally, I made a confusion matrix and a classification report to see how accurate the model prediction was with the 3 different parameters.

* 1. *Data Preparation*

Since there were no missing values, because the dataset was balanced, all I had to do was split the dataset into dependent and independent variables and test them based on the 3 experimental parameters.

* 1. *Experimental Design*

Table X: Experiment Parameters

|  |  |
| --- | --- |
| **Experiment Number** | **Parameters** |
| 1 | All four (4) raw features with 80/10/10 split for train, validate, and test |
| 2 | All four (4) normalized features with 80/10/10 split for train, validate, and test |
| 3 | All four (4) raw features with 70/15/15 split for train, validate, and test |
| 4 | All four (4) normalized features with 70/15/15 split for train, validate, and test |

* 1. *Tools Used*

The following tools were used; Jupiter notebook, a dataset from Kaggle, numpy, matplotlib. pyplot, pandas, seaborn, isnull( ).sum( ), pd.get\_dummies, sklearn, LogisticRegression and a classification report. These were used to analyze the dataset and show a isolation to see how accurate the prediction was with each parameter.

The following tools were used for this analysis: Python v3.5.2 running the Anaconda 4.3.22 environment for Apple Macintosh computer was used for all analysis and implementation. In addition to base Python, the following libraries were also used: Pandas 0.18.1, Numpy 1.11.3, Matplotlib 1.5.3, Seaborn 0.7.1, SKLearn 0.18.1.

1. **RESULTS**
   1. *Classification Measures*
2. Experiment 1:80/10/10 :89% accuracy
3. 5 0 0 0
4. 1 20 0 0
5. 0 2 1 0
6. 0 1 0 6
7. Precision:0.93
8. Recall:0.79
9. F1-score:0.81
10. Experiment 2:70/15/15
11. Confusion matrix:84% accuracy
12. 4 0 0 0
13. 1 13 0 0
14. 0 2 0 0
15. 0 1 0 4
16. Precision:0.65
17. Recall:0.68
18. F1-score:0.66
19. Experiment 3:80/10/10
20. Confusion matrix:88% accuracy
21. 4 0 0 0
22. 0 14 0 0
23. 0 2 0 0
24. 0 1 0 4
25. Precision:0.71
26. Recall:0.70
27. F1-score:0.70
    1. *Discussion of Results*

Using the confusion matrix helped a lot with visualizing the data and seeing how accurate it really was. Each confusion matrix printed a 4x4 matrix which then had color to the similar “heatmap” we used in our seaborn notebook. For experiment 2, the classification was pretty good, displaying a 84% accuracy rate. Experiment 1 however, was the best out of the three experimental parameters displaying a 89% accuracy rate. Experiment 3 was the second highest producting a 88% accuracy rate. The closer the percentage is to 100% the better the accuracy model is. For this project Experiment 1 was the best accuracy model.

* 1. *Problems Encountered*

The few problems that occurred was finding a good dataset to make a good prediction with. The first dataset I chose had too many categorical variables causing the program to crash when I ran it.

* 1. *Limitations of Implementation*

I think I picked the best model for the project. I could have used multiple linear regression, but I personally like using and seeing the confusion matrix and classification report.

* 1. *Improvements/Future Work*

I could try to find a better dataset with more variables and less categorical data. I would also like to experiment with more data to see what I can do further to the model.

1. **CONCLUSION**

I think this was a very good model, because in the end Experiment 2 had the highest accuracy rate of 83% which is good considering the other. Splitting the test and training set using different experimental parameters really helped the model and myself, to see how python programming can work to predict validation sets. Overall it was a good experiment with a great dataset that ended up giving very accurate results.

**Other directions:**

1. 10-pt, Times New Roman, 1” margins all around (if you use this template you are already set).
2. Ensure all tables and figures are numbered appropriately and referenced in the text. See examples above and below.

|  |  |
| --- | --- |
| **Figure 1: Comparison of X/Y from dataset (single plot) (8 pt.)** | **Figure 2: (a) Function Output (b) A against B (multiple plots) (8 pt.)** |