# Logistic Regression Classifier and K-Fold Cross Validation Implementations

**Indumini U. Jayakody, Helen L. Lin, David A. Neudorf**
Carleton University School of Computer Science
1125 Colonel By Drive, Ottawa, Canada
{indumini.jayakody, helen.lin, davidaneudorf}@cmail.carleton.ca

## Abstract

The objective of this project was to explore and analyze the Parkinson's and Stocks datasets. By implementing the linear classification logistic regression model from scratch, we were able to predict the target values of a given dataset. To ensure valuable features were not removed, a series of runs were done, dropping each feature and calculating the average error. We observed that simply removing features individually or in groups did not increase average accuracy (observed higher average error). This was then compared with the baseline error (from the original dataset), to see which features had the most and least effect on the model. The best model accuracy achieved for the Parkinson's dataset was 0.801 and 0.832 for the Stocks dataset. It is suspected that the model performed better for the stock dataset due to the higher volume of sample data.

## 1 Introduction

Logistic regression is a common predictive analysis technique in machine learning. It is often used with dichotomous variables to explain the relationship between one dependent binary variable (called the target, herein after referred to as "Class Label") and one or more nominal, ordinal, categorical independent variables. Using sigmoid (logistic) function, the output of the equation is either 1 or 0. The two benchmark datasets used in this project are a Parkinson's dataset and a Stocks dataset. Although both datasets vary in size and features, we were able to successfully implement the logistic regression function for both datasets.

On average, we found that we could achieve an accuracy of greater than 80% on both datasets through manipulating the input data into something more useful for our algorithm. In our implementation, the learning rate was decremented by 10% every 750 iterations after the first 1000, to a minimum of 2 $\epsilon$. To maintain consistency, the seed was set to 0 for all tests, since a different seed choice can influence and alter the accuracy. Throughout the experiment, the learning rate was 0.1 and $\epsilon$ was 0.000001 (unless otherwise specified).

## 2 Datasets

### 2.1 Parkinson's Disease

The dataset for Parkinson's Disease (PD) has 180 samples, 57 features, and one target value of having a diagnosis of PD or not. The distribution of the samples showed to be balanced, as seen in Figure 1. The dataset features covered age, encoded gender, medication (Levodopa-equivalent and Clonazepam), Hoehn & Yahr scale score, Unified Parkinson Disease Rating scale score, with the rest of the features being ordinal and predominantly motor, speech, and respiratory measurements. There were no missing values in this dataset.

While exploring the data, some patterns and relationships were observed. Figure 2, where participants without Parkinson's (Class 0) appeared to have a higher duration of pause intervals. Most of the features in this dataset had ordinal values ranging from [0,3]. The dataset was preprocessed by encoding these ordinal features as 1s and 0s, we expanded the features to 137.

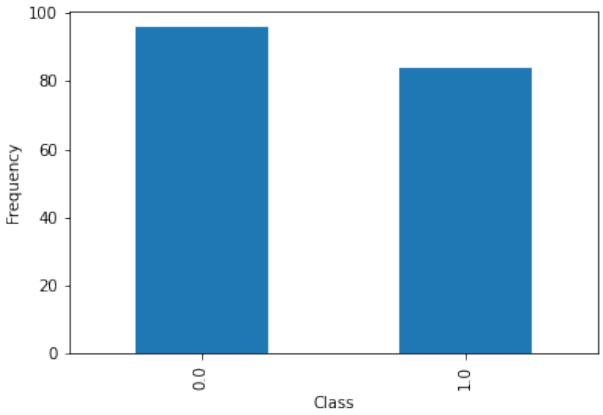


Figure 1: Matplotlib graph showing the frequency for both classes (0 - Does not have Parkinson's, 1 - Has Parkinson's)

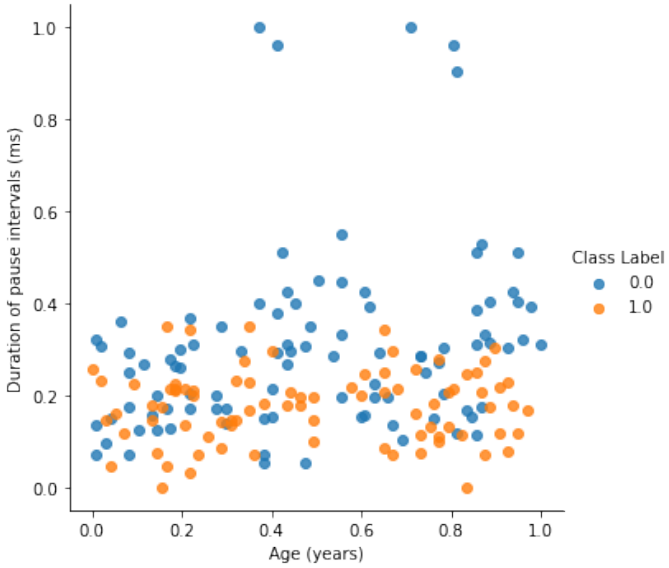


Figure 2: Matplotlib graph showing age and duration of pause intervals for both Class 1 and Class 0.

### 2.2 Stocks

The Stocks dataset was noticeably larger than the Parkinson's dataset. It had 4,907 samples, 155 features and a target label (Class 1 indicating a loss, Class 0 indicating a profit). The distribution of the classes can be seen in Figure 3. The features are comprised of ASTL, ITTEFAQ, MEBL, SMCPL, and OGDC, which represent stock ticker symbols. For each stock, there was a feature for its low, high, average, open, close, ldcp, and turnover for the day. This dataset had been preprocessed and NaN values had been set to 0. An additional preprocessing step was taken to replace values of 0 with the feature mean. This is because having values of 0 for stock price features creates outliers and would need to be discarded from the dataset.

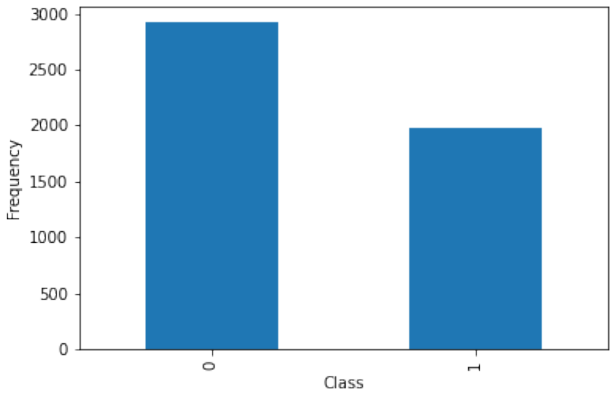


Figure 3: Matplotlib graph showing the frequency for both classes
(0 - Market ended up in profit, 1 - Market ended up in loss)

# 3 Results

## 3.1 Parkinson's Disease

Running the model on the raw data resulted in a baseline average error of 0.178. Preprocessing the data did not have any significant impact on accuracy. Notably, normalizing the data resulted in lower accuracy with an average error of 0.212 without preprocessing and an average error of 0.371 with preprocessing. For better efficiency, the normalized models were used for the remaining tests. In order to perform feature selection, the model was run repeatedly, each time with a different feature dropped. If the dropped features resulted in an accuracy higher than the baseline, they were grouped together and dropped. This resulted in an average error of 0.199 on the normalized data, and 0.186 on the raw data, which was higher than the initial raw results.

| **Experiment** | **Average Error** |
|---|---|
| Raw dataset | 0.178 |
| Preprocessed dataset | 0.178 |
| Normalized, not preprocessed | 0.212 |
| Normalized, preprocessed | 0.371 |
| Drop all columns that performed poorly on the individual tests from the normalized, not preprocessed dataset | 0.199 |
| Raw dataset with the aforementioned columns dropped | 0.186 |
| Drop all RLE from normalized & preprocessed dataset | 0.3248 |
| Drop all LLE from normalized & preprocessed dataset | 0.3183 |
| Drop all RUE from normalized & preprocessed dataset | 0.3151 |
| Drop all LUE from normalized & preprocessed dataset | 0.3403 |
| Drop all RUE and LUE from normalized & preprocessed dataset | 0.3031 |
| Drop all RLE and LLE from normalized & preprocessed dataset | 0.3025 |
| Drop all RLE and RUE from normalized & preprocessed dataset | 0.3051 |
| Drop all LLE and LUE from normalized & preprocessed dataset | 0.3015 |

Table 1: Table outlining various experiments done, and the average error produced.

Given that the majority of the features were ordinal and specific to one particular region of the body or type of motor disability, all features of one area were grouped and dropped. The features were labelled (1) "RUE" for Right-Upper Extremities, (2) "RLE" for Right-Lower Extremities, (3) "LUE" for Left-Upper Extremities, and (4) "LLE" for Left-Lower Extremities.

For context, Parkinson's Disease is a common neuro-degenerative disease that is characterized by motor, speech, and cognitive dysfunctions. Symptoms often begin on one side of the body but

eventually affect both sides (NIH). By looking into domain-specific literature reviews, it was found that there is substantial research showing that in assessing and treating PD, asymmetric symptom laterality (side in which symptoms begin) must be considered and is key in the definition of the disease (Verreyt et al, 2011; Feis et al, 2015). As such, a substantial experiment was done on dropping all features exclusively related to specific regions of the body (RUE, RLE, LUE, LLE). The differences in average error were marginal, with LUE seeming to have the most significance when dropped at an average error of 0.3403, and RUE having the least significance at 0.3151. Then, we dropped all features relating to upper extremities (RUE, LUE), lower extremities, right-side extremities (RUE, RLE), and left-side extremities (LUE, LLE). The latter half of the experiments resulted in a marginally lower error, with the lowest accuracy being from dropping left-side features. There was no feature indicating the laterality of the onset symptoms in the raw dataset. These 8 experiments increased the average error significantly and as such it can be concluded that the features in question are significant to the classification of PD.

### 3.2 Stocks

The baseline stock model average error was 0.177, when normalized it rose to 0.1885. We used the normalized dataset due to performance efficiency. When the features below were dropped, the result was a better average error (individually), while dropping all features (listed below) from the model resulted in an average error of 0.1879.

```
[MEBL_low, OGDC_low, BERG_average, BERG_high,
BERG_close, EPCL_low, AMBL_high, GATM_ldcp,
FFLNV_close, FFLNV_open, FFLNV_ldcp]
```

The following features resulted in an average error of at least 0.195 (when dropped individually):

```
[ASTL_turnover, MEBL_turnover, SMCPL_turnover,
OGDC_turnover, SERF_turnover, QUET_turnover,
BERG_turnover, EPCL_turnover, AMBL_turnover,
GATM_turnover, UBL_turnover, CPPL_turnover,
JPGL_turnover, JSML_turnover, SRSM_turnover,
JSCL_turnover, NPL_turnover, FFLNV_turnover,
YOUW_turnover]
```

The turnover features listed above were considered noteworthy as they had a significant impact on the average error. These features were added as additional columns in the dataset after being raised to the power of 1.5 showing that feature engineering increased performance. The exponent of 1.5 was chosen by experimenting with different numbers and finding that we reached a minimum average error of 0.1810. Additionally, we found that if we added columns for features whose label ended in "_average" (ex. ASTL_average) raised to the power of 0.1, our model performed even better. With both feature engineering experiments applied, our average error was reduced to 0.1678. If the log function was applied to the features ending in "_average", the average error was 0.1786. These can be seen in Table 2 rows 10-12, along with multiple other experiments conducted.

| ID | Experiment | Average Error |
|---|---|---|
| 1 | Dropping all turnover values (all values ending in '_turnover') | 0.4070 |
| 2 | Dropping all ldcp values (all values ending in '_ldcp') | 0.1960 |
| 3 | Dropping all high values (all values ending in '_high') | 0.1886 |
| 4 | Dropping all low values (all values ending in '_low') | 0.1956 |
| 5 | Dropping all average values (all values ending in '\average') | 0.1952 |
| 6 | Dropping all open values (all values ending in '_open) | 0.1920 |
| 7 | Dropping all close values (all values ending in '_close') | 0.1895 |
| 8 | Adding a feature column with '_average' squared | 0.1897 |
| 9 | Adding a feature column with '_lcdp' squared | 0.1912 |
| 10 | Adding a feature column with '_average' ^ 0.1 | **0.1682** |
| 11 | Adding a feature column with '_turnover' ^ 1.5 | **0.1810** |
| 12 | Adding a feature column with '_average' ^ 0.1 AND '_turnover' ^ 1.5 | **0.1677** |
| 13 | Adding a feature column with log('_average') | **0.1786** |

Table 2: Table outlining various experiments done, and the average error produced.

## 4 Discussion/Conclusion

In conclusion, feature selection and engineering was a crucial step in understanding how to achieve greater accuracies. We implemented binary-encoding for ordinal values, raised significant features to higher powers to provide them with more weight, and dropped features individually as well as in groups that seemed logical. Our final accuracies were 80.1% for the Parkinson's dataset and 83.2% for the Stocks dataset from running them through the same model. We suspect that the Stocks dataset had a higher accuracy because it was considerably larger in the size of samples and number of features.

We found that it was important to understand the dataset we were working with to find meaningful ways to guide the direction of feature engineering. On the other hand, as junior machine learning engineers, we will continue to learn more advanced feature engineering techniques to create more sophisticated models in the future.

## 5 Contributions

David: Data exploration, lead programmer, results section.
Indumini: Data exploration, helped with k-folds code, feature engineering, formatting, editing
Helen: Data exploration, code, reporting, formatting, editing, domain expertise

## References


[1] (n.d.). Parkinson's disease. National Institute on Aging. Retrieved October 11, 2021, from https://www.nia.nih.gov/health/parkinsons-disease.

[2] Feis, D.-L., Pelzer, E. A., Timmermann, L., & Tittgemeyer, M. (2015, October 29). Classification of symptom-side predominance in idiopathic parkinson's disease. Nature News. Retrieved October 11, 2021, from https://www.nature.com/articles/npjparkd201518.

[3] Pitsillides, Y. (2020, January 1). Python-tutorials/introduction to machine learning - logistic regression example (complete).ipynb at master · Pitsillides91/Python-Tutorials. GitHub. Retrieved October 11, 2021, from https://github.com/Pitsillides91/Python-Tutorials/blob/master/Introduction%20to%20ML%20-%20Logistic%20Regression%20Example/Introduction%20to%20Machine%20Learning%20-%20Logistic%20Regression%20Example%20(Complete).ipynb.

[4] Tokuç, A. A. (2021, January 30). Gradient descent equation in logistic regression. Baeldung on Computer Science. Retrieved October 11, 2021, from https://www.baeldung.com/cs/gradient-descent-logistic-regression).

[5] What is logistic regression? Statistics Solutions. (2021, August 11). Retrieved October 11, 2021, from https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/what-is-logistic-regression/.