SOFTENG 755

Bayesian Machine Learning

Assignment 1

Author: Lite Kim

Email: lkim564@aucklanduni.ac.nz

Department of Software Engineering

The University of Auckland

Assignment 1: Familiarising with machine learning models

Lite W. Kim
Department of Software Engineering
The University of Auckland
Auckland, New Zealand
lkim564@aucklanduni.ac.nz

***Abstract*—**

Keywords—machine learning, regression, ridge, classification, decision tree, nearest neightbor, naive bayes, svm, perceptron

I. Introduction

Given an assignment for the course of SOFTENG 755, 2018, we are to familiarise ourselves with the implementation of specific machine learning algorithms within a Python development environment. These algorithms include regular linear regression, ridge linear regression, decision tree classification, nearest neighbor classification, naive bayes classification, SVM classification, and perceptron classification. The recommended python libraries to use to gain practice with such algorithms were pandas, numpy, and sklearn.

II. Implementation Steps

A. Preprocessing the Data

The preprocessing of data happens at the beginning of the implementation process of machine learning algorithms. During this stage, given a set of data, the data is split into features and targets. The list of features is narrowed down, dropping any features that are perceived to be unimportant. The data may consist of cells that are not mathematically viable. For such, these cells are categorised into a binary matrix. Given n potential options of a certain feature, the matrix is n columns wide with each column associating with a potential option. Only one of the entries is filled as a 1 with that specific data entry associating to that option of the feature while the rest are 0. The data may come with invalid cells such as empty cells. During the preprocessing stage, these empty cells are filled with an appropriate value, usually the median value. Finally, the values of each cell need to be relatively balanced. This is achieved through scaling of the data. Once the data has been preprocessed, features need to be appropriately selected that will contribute to the optimisation of the machine learning algorithm.

B. Feature Extraction

During this stage, the best set of features is selected that would appropriate to the optimisation of the machine learning algorithm. Features selection needs to be repeated for each unique algorithm as there is no universal set of features that will provide the most optimal results for all algorithms.

C. Hyperparameter Tuning

Given the list of features for each machine learning algorithm, each algorithm is investigated to see which parameters can be adjusted to provide more accurate results. Of the regression algorithms, only ridge regression requires tuning of hyperparameters. The hyperparameter that needs to be tuned is the alpha value. For the classification algorithms, decision tree, nearest neighbors, SVM, and perceptron require have hyperparameters that can be tuned. The Decision tree algorithm has two hyperparameters: max_depth and min_samples_leaf; the nearest neighbors algorithm has two hyperparameters: n_neighbors and weight; the SVM algorithm has two hyperparameters: kernel and C; and the perceptron algorithm has one hyperparameter: alpha. The naive bayes algorithm does not have a hyperparameter to tune.

III. 2018 World Cup Predictions

Requiring the implementation of both regression and classification to predict the number of goals and the results respectively, all stated machine learning algorithms were implemented.

A. Preprocessing

The data is initially split into a list of features and two lists of targets. The targets were the number of goals scored and the match result. The preprocessing stage consisted of dropping the features ‘Date’, ‘Location’, ‘Phase’, and ‘Normal_Time’. The reason for such was that these features were neutral and were perceived to have no correlation to the output result. The features are further split into numerical and categorical features. The numerical features consisted of:

'Team1_Attempts', 'Team2_Attempts', 'Team1_Corners', 'Team2_Corners', 'Team1_Offsides', 'Team2_Offsides', 'Team1_Ball_Possession(%)', 'Team2_Ball_Possession(%)', 'Team1_Pass_Accuracy(%)', Team2_Pass_Accuracy(%)', 'Team1_Distance_Covered', 'Team2_Distance_Covered', 'Team1_Ball_Recovered', 'Team2_Ball_Recovered', 'Team1_Yellow_Card', 'Team2_Yellow_Card', 'Team1_Red_Card', 'Team2_Red_Card', 'Team1_Fouls', and 'Team2_Fouls'

while the categorical features consisted of:

‘Team1’, ‘Team2, ‘Team1_Continent’, and ‘Team2_Continent’.

The categorical features are converted to a mathematically analysable matrix form and joined with the numerical features. In case of empty entries, among the numerical features, the empty cells are filled with a median value.

B. Feature Extraction

For the extraction of features for the linear regression, an initial evaluation is conducted fitting a default set of features with the number of goals in a linear regression model. The default set of features were 'Team1' and 'Team2' selected on the perception that the model must at least fit for them. Pairs of features ('Team1_xxx' and 'Team2_xxx') are joined to the default set to see if the addition of the features can provide a better accurate model. This is judged by comparing the mean squared errors of the default set by themselves with the mean squared errors of the default set with the additional pair of features. If there exists at least a pair of features that does have a lower mean squared errors than the default set, then the pair of features that provides the lowest mean squared errors becomes part of the new default set for comparing. The most optimum set of features can then be found through a while loop which terminates if all features are added to the default set or if the default set has a lower mean squared errors than all set with an additional pair of features. To make more accurate results, a cross-validation method is adopted. The cross-validation method for this case was K-folds using a K-fold value of 3 due to the limited set of entries provided. This provided the features set of: Team1, Team1_Continent, Team1_Fouls, Team1_Ball_Recovered, Team1_Attempts, Team2, Team2_Continent, Team2_Fouls, Team2_Ball_Recovered, Team2_Attempts, with an average mean squared errors of 4.79.

The same was conducted on the ridge regression feature selection with unattuned hyperparameters providing the feature set of Team1, Team1_Ball_Recovered, Team2, Team2_Ball_Recovered, with an average mean squared errors of 2.96.

Similarly, the optimum features can be found for the classification machine learning algorithms. Instead of comparing the mean squared errors, you would compare the accuracy score. Fitting against untuned machine learning classifiers, you get the following features set.

decision tree algorithm with:
Team1, Team2, Team1_Red_Card, Team2_Red_Card with an average accuracy of 50.4%;

nearest neighbors algorithm with:
Team1, Team2, Team1_Ball_Possession(%), Team2_Ball_Possession(%) with an average accuracy of 53.4%;

naive bayes (gaussian) algorithm with:
Team1, Team2, Team1_Red_Card, Team2_Red_Card, Team1_Continent, Team2_Continent with an average accuracy of 47.0%;

SVM algorithm with:
Team1, Team2, Team1_Yellow_Card, Team2_Yellow_Card, Team1_Pass_Accuracy(%), Team2_Pass_Accuracy(%) with an average accuracy of 53.5%;

perceptron algorithm with:
Team1, Team2, Team1_Yellow_Card, Team2_Yellow_Card, Team1_Corners, Team2_Corners, Team1_Ball_Possession(%), Team2_Ball_Possession(%), Team1_Continent, Team2_Continent with an average accuracy of 51.8%.

C. Hyperparameters

Tuning for the prediction of goals only consisted of tuning the alpha value of the ridge regression. This is done using the GridSearchCV function. Given a list of hyperparameter ranges or options, the function finds the best parameters that provides the most accurate machine learning algorithm.

For the tuning of the ridge regression, the optimum alpha value was set as 14.5 providing a mean squared errors of 2.48. The decision tree algorithm was set with the max_depth as 8, and min_sample_leaf as 1 providing an accuracy of 54.7%. The nearest neighbors algorithm was set with the n_neighbors as 4 and weight as uniform providing an accuracy of 54.7%. Compared to the untuned results, all showed a better mean squared errors or accuracy outcome. For the following SVM and perceptron algorithms, the inclusion of hyperparameters reduced the accuracy. For the SVM algorithm, the best parameters were set with a C as 1 and kernel as rbf providing an accuracy of 53.1% down from 53.3%. For the perceptron, the alpha was set as 1.0 providing an accuracy of 51.6%, down from 51.8%. The reduced accuracy may have been due to the limited range of options provided to the GridSearchCV function, where the default values were not included within the options and where the default values provided a more optimum algorithm.

D. Results

a) Regular Linear Regression: Of the regression models, the regular linear regression managed to predict all the goal results. Such results cannot be relied upon as the model was fit using all the dataset which may imply an overfitting problem. The results of such are shown below with the blue data set as the true results and the orange data set as the predictions.

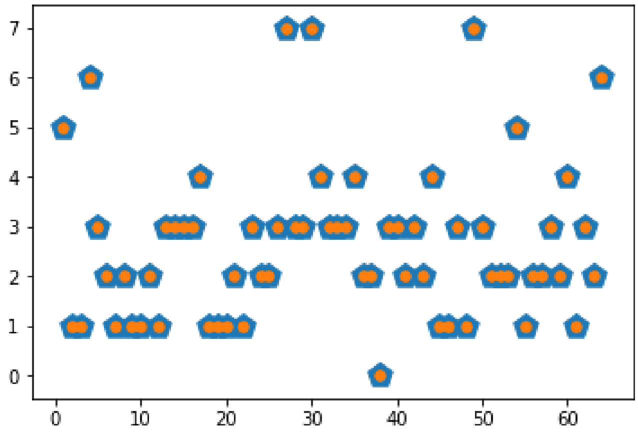

Fig. 1. Predictions of linear regression (World Cup)

b) Ridge Regression: For the ridge regression models on the other hand, using cross validating methods, An alpha value set at 15.7 was determined to provide an optimal average mean squared errors. However, when the predictions were plotted against the true results, the predictions seemed to be underfitting the data with a concentrated scatter about the mean as shown in figure 2. The R^2 score of the prediction were also found to be 0.247. When the alpha value was set as 0, figure 3 is resulted with an R^2 score of 0.977 implying that a ridge regression with an alpha set as 0 is more accurate. By comparing the two plots, the statement becomes obvious.

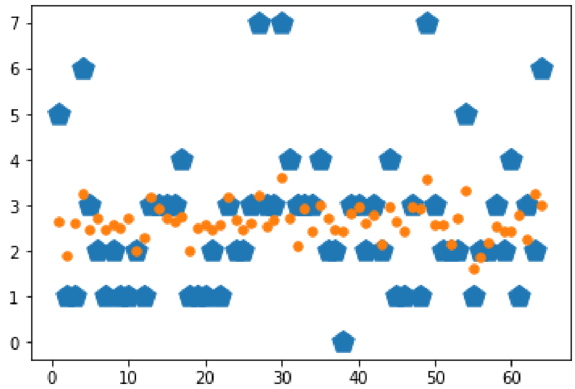

Fig. 2. Ridge regression prediction (alpha = 15.7)

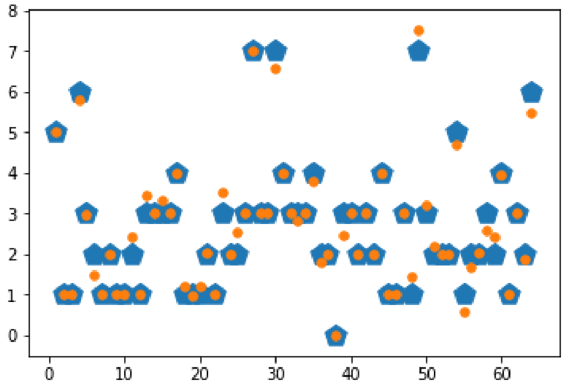

Fig. 3. Ridge regression (alpha = 0) – World Cup

c) Classification Alorithms

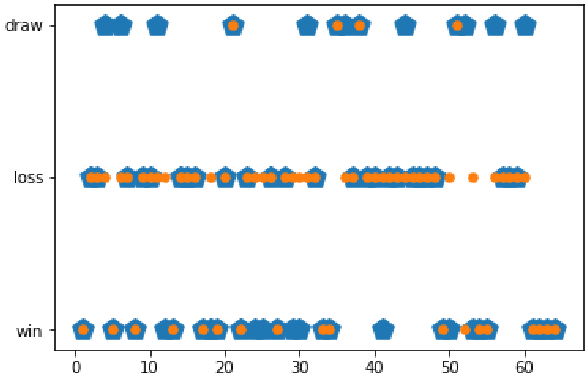

Fig. 4. Decision tree predictions (World Cup)

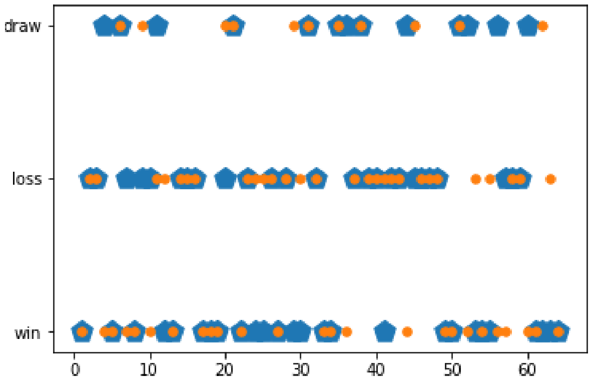

Fig. 5. Nearest neighbor predictions (World Cup)

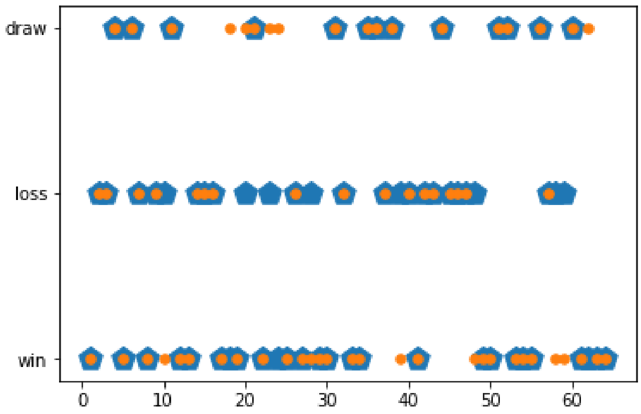

Fig. 6. Naive bayes predictions (World Cup)

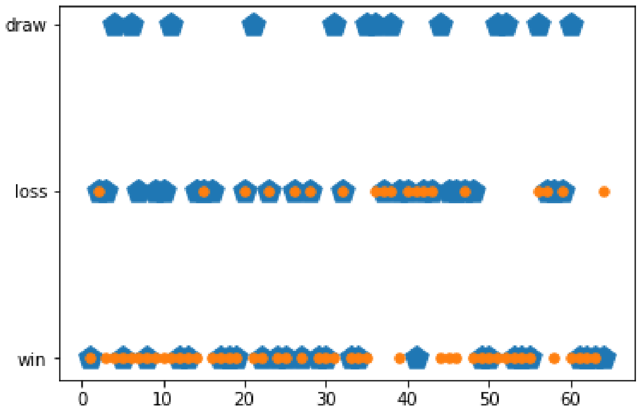

Fig. 7. SVM predictions (World Cup)

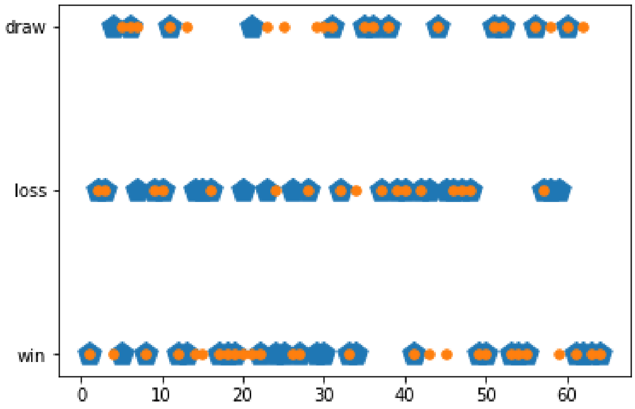

Fig. 8. Perceptron predictions (World Cup)

IV. Traffic on State Highway 1

A. Preprocessing

The data consisted of 450 potential features with data that were all numerical. There was no prejudgement on the relevance of each data. In case of missing data cells, the data is checked through and if an empty cell exists, then it is filled with the median data. The data is then scaled and becomes ready for feature extraction.

B. *Feature Extraction*

Given the 450 features, the same technique for finding the features for the World Cup data was conducted. At the end of the feature selection, 122 features were extracted to provide the most optimal feature set that minimised the mean squared errors. When fitted on the linear regression model, the R^2 value comes out to be 0.9670.

C. *Hyperparameters*

The alpha value for the ridge regression required tuning. The tuning consisted of using cross validation scoring with a K-fold of 10. This resulted an alpha value of 8.5 with the associated R^2 value of 0.9667, slightly less than that of the regular linear regression model.

D. *Results*

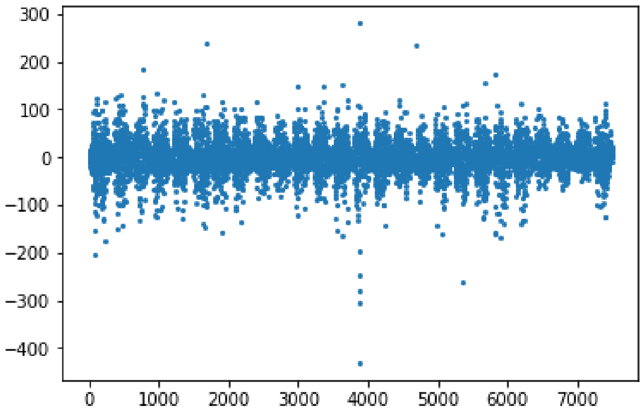

Fig. 9. Ridge regression prediction errors (Traffic flow)

Although the R^2 value of the ridge regression resulted in a decrease from the regular regression, the ridge regression was cross validated which included the alpha value of 0. The results of the cross validation showed that the average of mean squared errors must have been less with an alpha value of 8.5 than an alpha value of 0. By using an alpha value, the overfitting of the regression is negated making the ridge regression model preferable over the regular regression model.

V. Occupancy Data

A. *Preprocessing*

The data consisted of 6 features: date, Temperature, Humidity, Light, CO2, and HumidityRatio. Given these features, the best set of features need to be extracted to provide the most optimal models for each machine learning algorithm. During the preprocessing, the date feature was dropped as each entry was unique producing a date classification matrix of 17120 columns. This would result in a computationally expensive task. Because each date is unique, the effect that it would have on the predictions would overwhelm all other features and would produce a model that would be problematic to future predictions not provided. The rest of the features were numerical. The dataset is then checked for empty cells and filled with median values then is scaled.

B. *Feature Extraction*

Each machine learning model requires a feature set that will specifically optimise their own machine learning model. There is no feature set that is considered universally viable. Thus, the following feature sets were extracted for each machine learning model:

- Decision tree: Light, CO2, and HumidityRatio
- Nearest neighbor: Light, Humidity, Temperature, and HumidityRatio
- Naive bayes: Light, Humidity, HumidityRatio
- SVM: Light, Humidity, CO2
- Perceptron: Light, CO2

Among all machine learning models, Light was a common feature that influenced each model. Light was also determined to be the most dependable appearing first in each feature extraction case.

C. *Hyperparameters*

Setting the K-fold to 10 and using cross validation, the following were tuned:

- Decision tree: max_depth = 1, min_samples_leaf = 1 providing an average accuracy of 98.7%
- Nearest neighbor: n_neighbors = 49, weight = ‘uniform’ providing an average accuracy of 95.8%
- SVM: kernel = ‘linear’, C = 2 providing an average accuracy of 98.5%
- Perceptron: alpha = 1.0 providing an average accuracy of 92.7%

Naive bayes did not have a hyperparameter requiring tuning with the average accuracy outcome being 96.6%.

D. *Results*

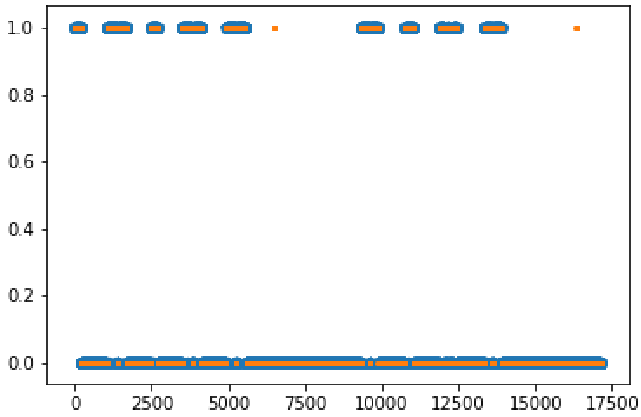

Fig. 10. Decision tree prediction (Occupancy)

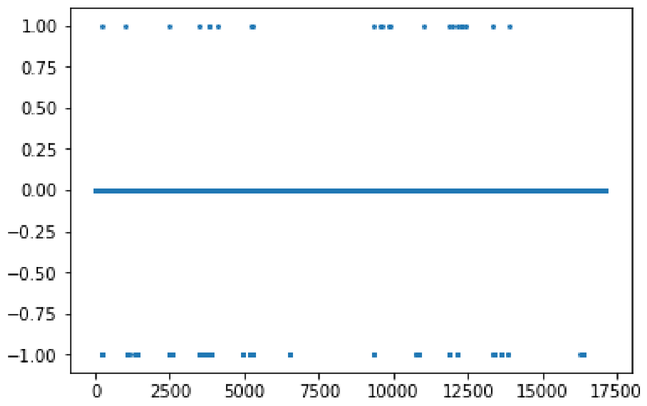

Fig. 11. Decision tree errors (Occupancy)

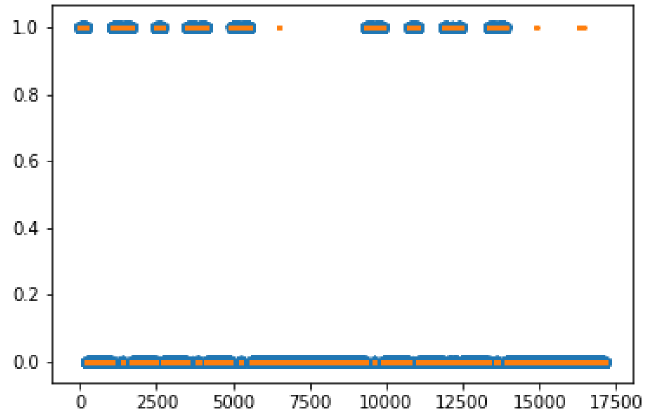

Fig. 12. Nearest neighbor predictions (Occupancy)

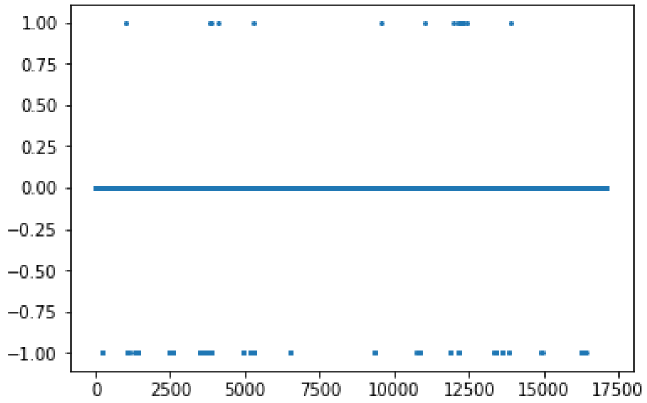

Fig. 13. Nearest neighbor errors

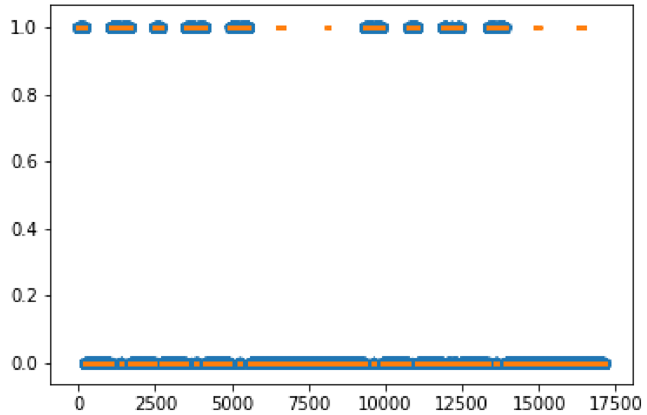

Fig. 14. Naive bayes predictions (Occupancy)

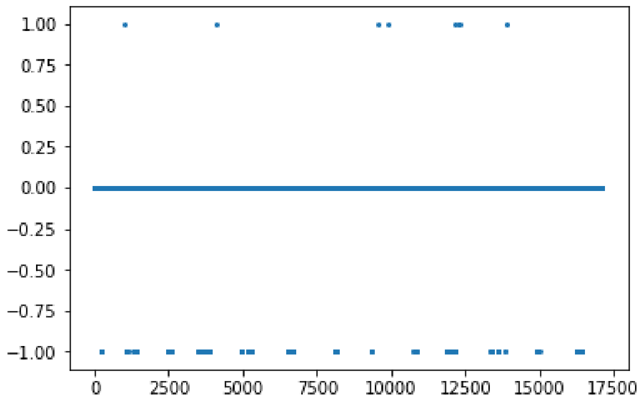

Fig. 15. Naive bayes errors (Occupancy)

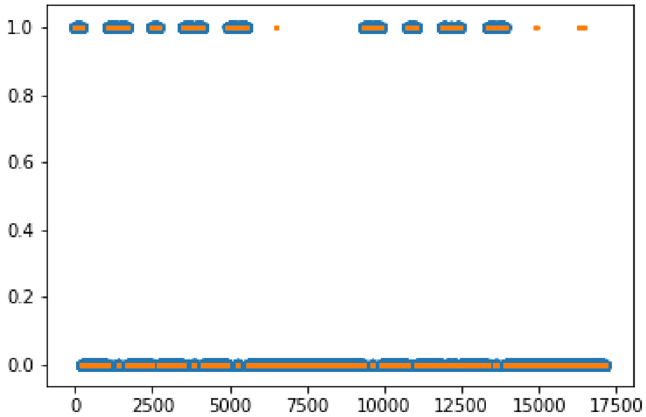

Fig. 16. SVM predictions (Occupancy)

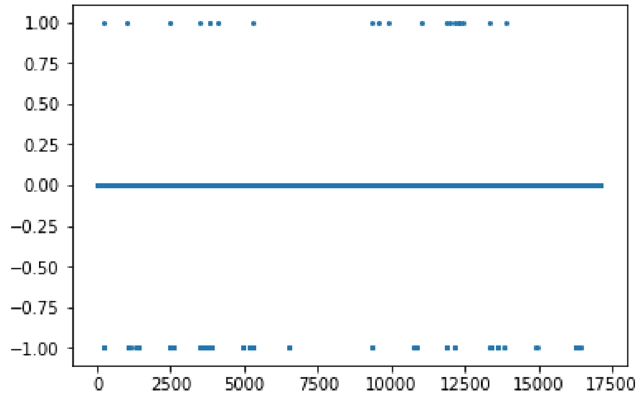

Fig. 17. SVM errors

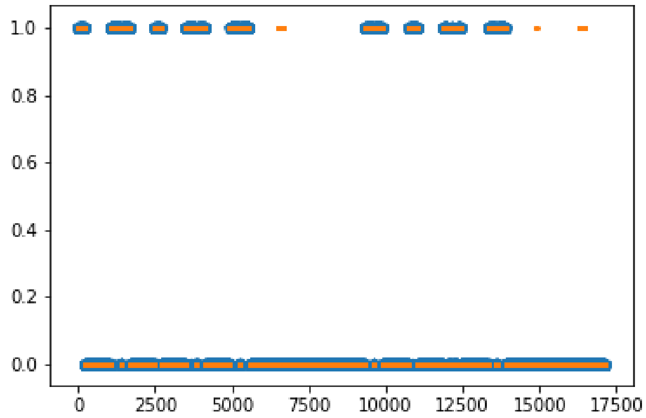

Fig. 18. Perceptron predictions (Occupancy)

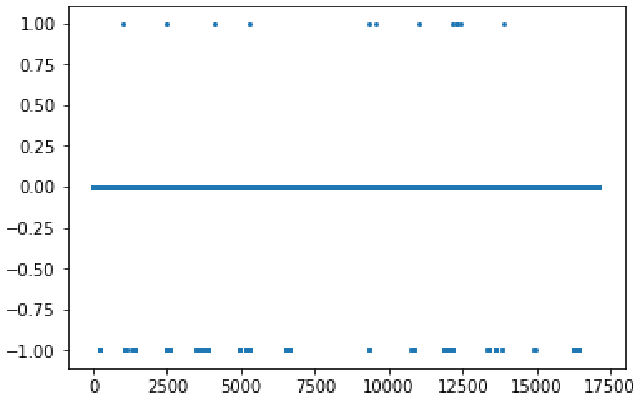

Fig. 19. Perceptron errors (Occupancy)

VI. Land Satellite Data

A. Preprocessing

The dataset consisted of 36 potential features. Like the occupancy data machine learning model implementation, all data was numerical. The data was checked for missing data cells, filling them with the median value if determined to be missing and then scaled.

B. Feature Extraction

Likewise, each feature set of land satellite data must be specifically identified for each learning model. The following was determined:

- Decision tree: Features 17, 16, 19, 22
- Nearest neighbor: Features 17, 16, 19, 22, 2, 15, 13, 0
- Naive bayes: Features 17, 12, 19, 27
- SVM: Features 17, 12, 15, 22, 5, 1, 20, 26, 0, 7, 10, 4, 2, 3, 21, 6
- Perceptron: Features 20, 22, 34

Among all machine learning models, feature 17 was the most common shared among four of the five machine learning models and being selected first. Features 16, 19, and 22 were common between the decision tree and nearest neighbor models with the features in the same priority. Naïve bayes also shared feature 19 in the same priority level however unlike the decision tree and nearest neighbor models, feature 12 was prioritised second along with the SVM. The SVM, decision tree and nearest neighbor models prioritised feature 22 in the same position as well. The perceptron algorithm took a unique set of features although sharing some features but at different priorites.

C. Hyperparameters

Setting the K-fold to 10 and using cross validation, the following were tuned:

- Decision tree: max_depth = 10, min_samples_leaf = 11 providing an average accuracy of 84.7%
- Nearest neighbor: n_neighbors = 5, weight = ‘distance’ providing an average accuracy of 87.0%
- SVM: kernel = ‘rbf’, C = 4 providing an average accuracy of 88.2%
- Perceptron: alpha = 1.0 providing an average accuracy of 51.5%

Naive bayes did not have a hyperparameter requiring tuning with the average accuracy outcome being 79.8%.

D. Results

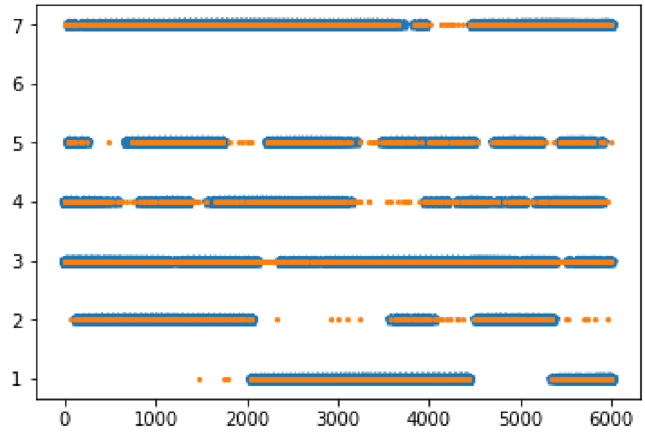

Fig. 20. Decision tree prediction (Landsat)

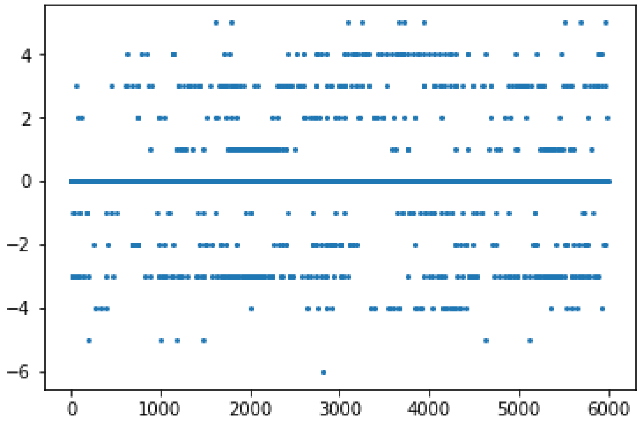

Fig. 21. Decision tree error (Landsat)

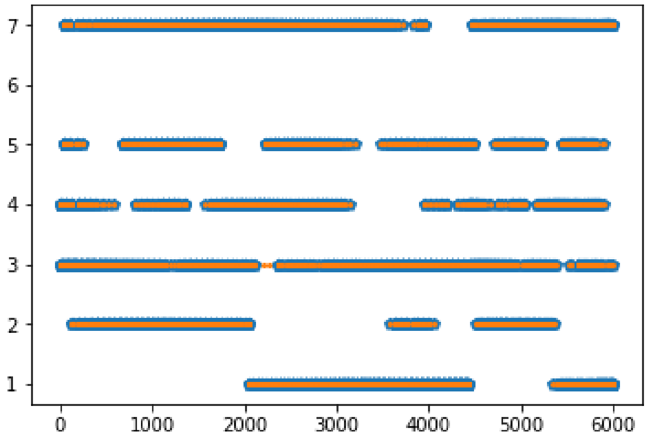

Fig. 22. Nearest neighbor prediction (Landsat)

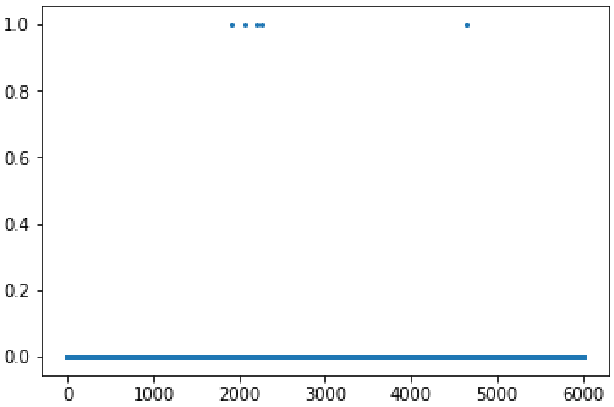

Fig. 23. Nearest neighbor errors (Landsat)

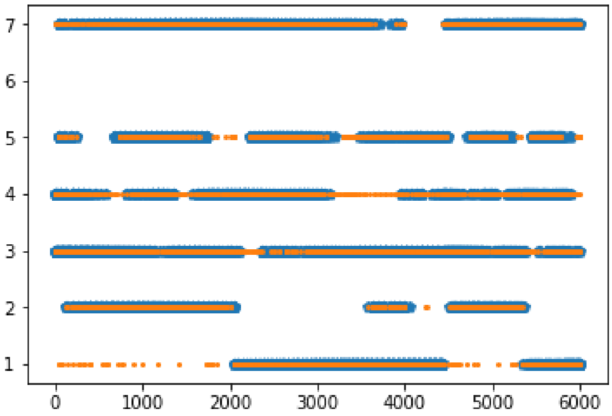

Fig. 24. Naive bayes predictions (Landsat)

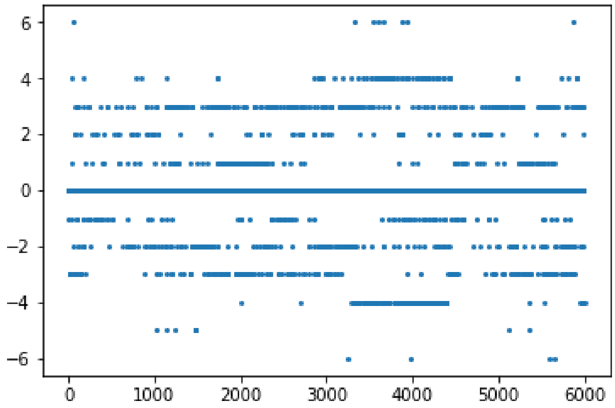

Fig. 25. Naive bayes errors (Landsat)

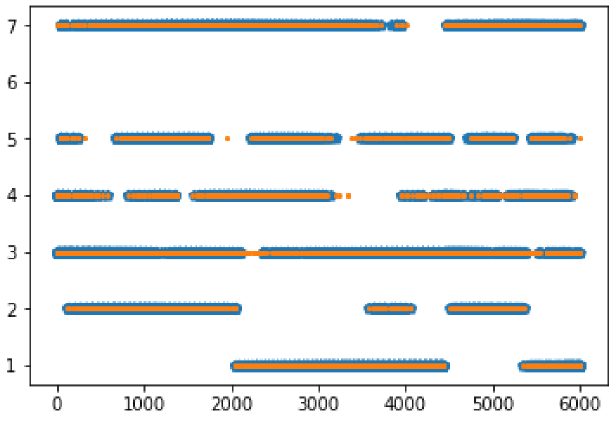

Fig. 26. SVM prediction (Landsat)

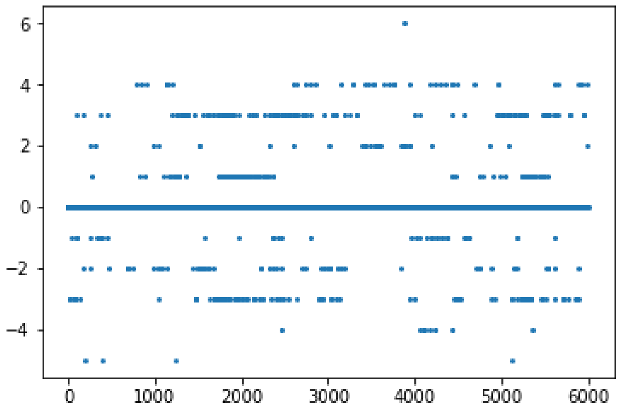

Fig. 27. SVM errors (Landsat)

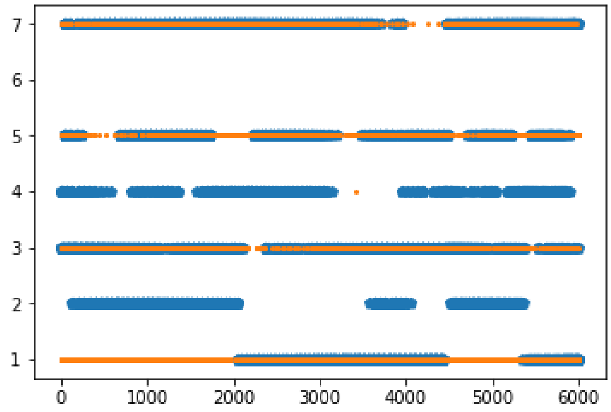

Fig. 28. Perceptron predictions (Landsat)

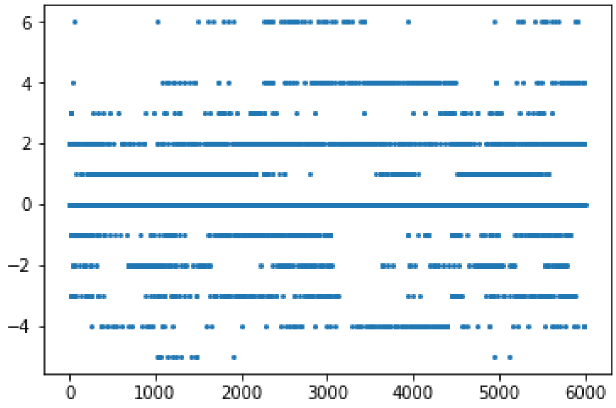

Fig. 29. Perceptron errors (Landsat)

Of all the machine learning algorithms, with regards to the Land Satellite data, the predictions had an accuracy of around 80% except for the perceptron algorithm providing an unfavorable prediction of 51.5%. Comparing all algorithms, the nearest neighbors algorithm provided the most accurate model.

VII. Discussion

A. *World Cup Machine Learning Models*

Given only 64 results to determine a machine learning model provides a limitation to the accuracy of the learned model. The model becomes more susceptible to over fitting and due to the relative small feature set provided, the model becomes less adoptable. As observed from the implementation, the linear regression showed perfect predictions however this is very sceptical and should be assumed that the model is overfitting the data. The Ridge regression on the other hand shows that an alpha value set to 0 is more desirable than an optimised alpha value of 15.7 shown through the different plots despite the average mean squared errors generated through cross validation being smaller. This may imply that the implementation of the cross validation may be incorrect as the R^2 value of the tuned ridge regression was smaller than the untuned model.

B. *Traffic Flow Machine Learning Models*

The tuned ridge regression model provided a slightly small R^2 value than the untuned model however, this does not imply that the untuned model is better as the results of the cross validation showed that the average mean squared errors were smaller for the tuned model than it was for the untuned model. Introducing the alpha value also implies a restriction on the fit of the model negating the effects of overfitting. Overfitting may be a larger problem with this data set as 122 of the 450 features were used to determine the model. But the overfitting problem becomes less evident as there are comparatively large amounts of data points to consider.

C. *Occupancy Machine Learning Models*

The decision tree machine learning algorithm was determined to be the most accurate model with an average accuracy of 98.7% determined from a 10-fold cross validation method. This was closely followed by the SVM machine learning algorithm with an average accuracy of 98.5%. The most common feature used in all models was the Light feature which is logically true as if there is no light, then it is highly unlikely that the room would be occupied.

D. *Land Satellite Machine Learning Models*

The SVM machine learning model was the most accurate with an average accuracy of 88.2% determined by 1 10-fold cross validation method. The most common features used were feature 17, 19, and 22.

Acknowledgment

Acknowledgements given to the tutor Mr. Li Tian for the introduction to machine learning using Scikit learn, pandas, and Numpy on Python, the lecturer Dr. Xuyun Zhang for the background information into understanding machine learning models, Youtube for the tutorials and the official documentation page for Scikitlearn.