MLDM 2021 Coursework Group: *Random Strangers*

Arjun Krishnan AK01928@SURREY.AC.UK
Prabha Krishnamoorthy PK00580@SURREY.AC.UK
Fan Yam "Kelvin" Tsang FT00247@SURREY.AC.UK
Tsz Hin (Hingo) Chan TC01022@SURREY.AC.UK

Abstract

The Bankruptcy prediction and Rain in Australia datasets are evaluated using three supervised learning algorithms namely, Decision Tree, Support Vector Machine and Deep neural network and their performances are compared. Model evaluation experiment results prove DNN to be the best performing model. For reinforcement learning, two model-free algorithms are used to train the agent namely, Q Learning and SARSA, in an OpenAI environment, CartPole-v1. Result shows that SARSA performs better in general while Q Learning performs better with fewer training episodes.

1. Project Definition

The aim of the project is to analyze and prepare the selected datasets to apply different Machine learning and data mining algorithms to evaluate and discuss the results by following the CRISP-DM methodology. The model is evaluated using various evaluation metrics such as Accuracy, F1 score, Precision, Recall, ROC and Precision-recall curve. The best learning algorithm is determined based on these metrics.

1.1 Company Bankruptcy Prediction

Prediction of bankruptcy is the key to evaluate the financial situation of a company and can protect the business from financial distress. This dataset contains Bankruptcy data from the Taiwan Economic Journal for the years 1999–2009. The dataset contains 6.8K rows and 96 financial features. This is a classification problem to identify whether the company is bankrupt or not. The target variable is Bankrupt that contains values 1(Yes) and 0(No).

Source: https://www.kaggle.com/fedesoriano/company-bankruptcy-prediction

1.2 Rain in Australia

Predicting the future of the weather is vital for individuals and organizations for various reasons. This dataset contains about 10 years of daily weather observations from many locations across Australia. It contains 145K rows and 23 features correlated to weather conditions. This is a classification problem to predict whether it will rain in Australia the next day or not. RainTomorrow is the target variable. It has values 'Yes' and 'No'.

***Source*:** https://www.kaggle.com/jsphyg/weather-dataset-rattle-package

1.3 Reinforcement learning

The aim of this section is comparing the performance of two model free reinforcement learning algorithms, Q Learning and SARSA in the OpenAI CartPole-v1 environment. The goal is to prevent the pendulum falling over by moving the cart to the left or right.

Link: http://gym.openai.com/envs/CartPole-v1/

2. Data Preparation

Data preparation/preprocessing steps include handling missing values, feature transformation, encoding categorical features, feature scaling, feature selection and dimensionality reduction.

2.1 Company Bankruptcy Prediction

2.1.1 Data cleaning / data integration

The data cleaning or the preprocessing is the data mining process of detecting and correcting data inconsistency that includes incorrect format, duplicate or incomplete data in a dataset. Various preprocessing steps that were done on the dataset are,

- ***Handling missing values:***

The data is plotted using the missingno library to visualize the distribution of Nan values. There no missing values present across the columns in the dataset.

- ***Handling Outliers:***

Outliers are observations that lie at an abnormal distance from other values in the population sample. The outliers are identified using IQR (Inter quartile range) which is the measure of statistical dispersion calculated as the difference between 75th and 25th percentiles. The outliers once identified are replaced by their respective mean value.

2.1.2 Variable Transformation/derivation of new variables/dimensionality reduction

- ***Feature scaling:***

It is necessary to standardize the data to avoid bias towards certain variables. The input features are normalized with the use of StandardScaler() function. It transforms the data such that it has a mean of 0 and a standard deviation of 1.

- ***Dimensionality Reduction:***

It is necessary to reduce the number of dimensions(features) as a greater number of features often makes a modelling task more challenging. This dramatically impacts the performance and the predictive capability of the model by learning redundant features that are not required for the modelling task as hand.

➢ ***Highly correlated features:***

Highly correlated features are removed from the data as it becomes difficult for the model to estimate the relation between the independent and dependent variables. Therefore, the features with correlation coefficients greater than 0.95 are removed.

➢ ***Principle Component Analysis (PCA):***

PCA is technique of feature extraction used to drop the features of least importance towards the predictive modelling task as hand. It works by projecting each data point onto the first few principle components to obtain lower dimensional data while preserving as much of the data's variation as possible. This ensures that no valuable information is lost from the data.

- ***One hot encoding:***

It is necessary to convert the categorical features into numerical format. Each categorical value is converted into a column of its own with values of either 0 or 1 assigned to them. Each value is represented as a binary vector, with the index marked as 1.

2.1.3 Data exploration / data visualization

- ***Distribution of labels:***

The distribution of the target variables is visualized to check whether the data is imbalanced. As seen from figure 1, the target variable is highly imbalanced, with most of the labels being "0" i.e. The company did not go Bankrupt.

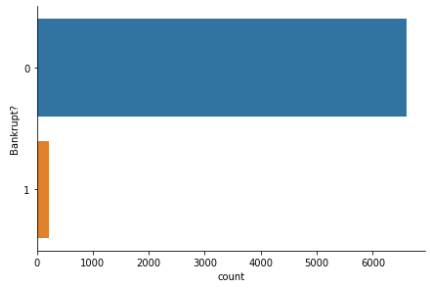


Figure 1. Distribution of labels

- ***Correlation Matrix:***

The correlation matrix is plotted to investigate the correlation between the variables in the data. The correlation matrix shows the correlation coefficients between the variables.

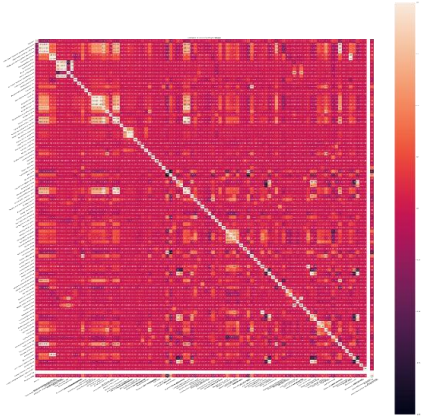


Figure 2: correlation matrix

- ***Handling Imbalance in the data (SMOTE):***

Imbalance of classes in the data often lead to skewed performance of models with a bias towards the majority class. The model would perform poorly on the minority class as the model does not have enough data points to learn. This imbalance in the data is handled by oversampling the minority class. This is achieved by duplicating the examples in the minority class. This type of data augmentation is referred as Synthetic Minority Oversampling TEchnique (SMOTE).

It is important to note that SMOTE is only used on the training data and not on the entire data before splitting to avoid any data leakage. Data leakage is when information from outside the training data is used to train the model. This outside information is otherwise not known to the model, thus producing skewed results. If the test-train split is carried over after performing SMOTE, the validation set would contain the created information from SMOTE thus causing data leakage.

- ***Splitting the data:***

Stratified test-train split is used to split the data into testing and training splits. 80% of the date is used for training and the remaining 20% is used for testing. Furthermore 20% of the training data is retained for validation.

2.2 Rain in Australia

2.2.1 Data cleaning / data integration

- ***Handling missing values:***

A graph was plotted to check for null values and the plot inferred that Cloud9am, Cloud3pm, Evaporation and Sunshine had a lot of missing values. The dataset was split into numeric and categorical values. For each of the categorical feature, the missing value is updated with the most frequent values. The null values in the numerical features were handled by replacing with their respective median values.

- ***Handling Outliers:***

It is seen that the Rainfall, Evaporation, WindGustSpeed, WindSpeed9am, WindSpeed3pm and Pressure9am columns contain outliers. After detecting, median imputation is used to take care of outliers. The extreme values are replaced with median values using IQR.

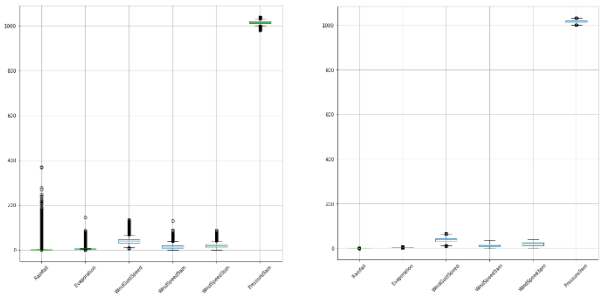


Figure 3: Before and After handling Outliers

2.2.2 Variable Transformation/derivation of new variables:

- ***Feature Expansion:***

The Date column is converted into datetime and then split into Day, Month and Year columns. These expanded features are used for other preprocessing steps. The original Date variable is dropped from the dataset.

- ***Feature scaling:***

Some of the features in the dataset varied in range and magnitude. It is critical to maintain same level of magnitude for all the features. This was achieved in the dataset by using MinMax scalar.

- ***Label encoding:***

The categorical variables Location, WindGustDir, WindDir9am, WindDir3pm are encoded using dummy feature for each unique value and the RainToday is encoded using BinaryEncoder.

2.2.3 Data exploration / data visualization

Exploratory Data Analysis is performed on the Rain in Australia dataset. It helps to validate the raw data and check for any inconsistency. Most critical part of EDA is to discover patterns and relationships between variables in the dataset.

- ***Heatmap for Correlation matrix:***

The Correlation matrix is plotted to understand the interactions between the different features in the dataset. From the matrix we could infer that Pressure9am, Pressure3pm, MaxTemp and Temp3pm features are highly correlated.

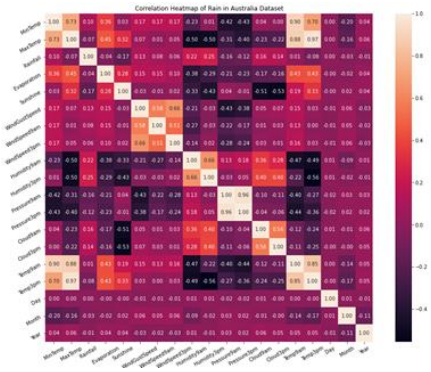


Figure 4: Heatmap for Correlation Matrix

- ***Pair plot:***

The pair plots are also plotted between these features to find the correlation.

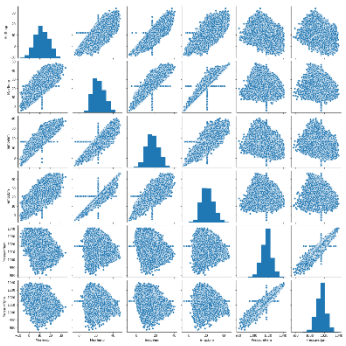


Figure 5: Pair plot

- ***Imbalanced dataset:***

The data is then visualized to check the distribution of the target variables “RainTomorrow”, As seen from figure 6, the target variable is highly imbalanced, with most of the labels being “No” i.e. It will not rain tomorrow in Australia.

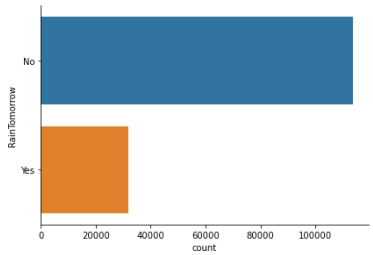


Figure 6: Distribution of labels

2.3 Reinforcement Learning

The environment used for this project is the CartPole-v1 environment provided in OpenAI gym. The environment can be loaded from the gym library in Python. The state space is continuous for this environment. Therefore, an additional function is defined to convert the state space into discrete with bin size = 20. The reward for this environment is 1 for all actions including the termination step. The maximum reward for each episode is 500.

3. Model development

Supervised ML techniques are applied to predict the correct labels for classification problem. Non-linear classifier algorithms mentioned below are used for both the datasets. The datasets are split into train and test data in the ration of 80:20. The train data is further split into train and validation data. So overall the split ratio is 64:16:20.

3.1 Decision Tree:

Decision trees (DT) are simple and very popular algorithm for predictive modeling. This algorithm performs both classification and regression and are highly interpretable. It is a binary tree that recursively splits the dataset until it is left with the pure leaf nodes (one set of class). This algorithm is sensitive to unbalanced classes and can result in less accuracy when one of the classes are dominant. To avoid this problem, either resampling methods or class weights needs to be added so that the minority class will be equally treated against the majority class.

The decision tree pruning is done by tuning the various hyper parameters. By doing this the problem of overfitting can be avoided. When the depth of the tree increases, the complexity increases thereby leading to overfitting problems. We below mentioned hyper parameters are tuned using GridSearchCV().

Table 1: DT Hyper parameters

Hyperparameter	***Dataset 1***	***Dataset 2***
criterion	entropy	entropy
ccp_alpha	0.015	0.015
splitter	random	random
max_depth	25	35
max_features	30	30
max_leaf_nodes	20	20
min_samples_leaf	5	15
min_samples_split	15	15

3.2 Support Vector Machine:

SVM is a supervised machine learning algorithm available in scikit-learn library that be used for classification and regression tasks. SVM works by plotting each data point in a n-dimensional space where n is the number of dimensions in the data, with the value of each feature being the value of a certain coordinate. The data is then classified by finding the hyperplane that divides the two different classes (in case of binary classification).

The advantage of SVM is its ability capture complex relations between the data points with a clear margin of separation between the classes. It is also highly effective in high dimensional spaces. Although SVM works very well, the complex data transformation and the boundary plane cannot be visualized hence the name Black box is associated with it. SVM is also computationally expensive on large data sets with considerable higher training times compared to other algorithms.

GridSearchCV() is used for hyper-parameter tuning. Grid search function works by building a model for every combination of the hyperparameters as specified and evaluates each model based on the evaluation metric specified by using the scoring parameter (F1 score in our case).

Table 2: SVM Hyper parameters

Hyper parameter	***Company Bankruptcy Prediction***	***Rain in Australia***
C(Regularization parameter)	1	100
Gamma(kernel coefficient)	0.1	0.01
Kernel	rbf	rbf

3.3 Deep Neural Network:

Neural network with more than one hidden layer is deep neural network. A deeper network can potentially learn more complex problem. However, gradient vanishing problem might occur if the network is too deep. The backpropagated error update decreases exponentially as the hidden layer is further from the output layer. Therefore, a total of 4 layers are used in the experiment, including one input layer, 2 hidden layers and one output layer.

Hyperband is used to tune the hyperparameters of the neural network. Number of neurons in a hidden layer, degree of regularization, type of regularization (L1 or L2), dropout rate of each hidden layer, as well as optimizer are the hyperparameters to be tuned in the neural network.

Grid search, random search, Bayesian optimization and hyperband are the potential options to tune the model. Grid search and random search are researched to be less efficient and effective than Bayesian optimization and hyperband, as the former two are exhaustive approaches that do not learn the search space when optimizing. Hyperband is an

improved version of random search by balancing the exploration and exploitation to find the best time allocation for all potential configurations. [1] The best parameters are shown in the table below.

Table 3: DNN Hyper parameters

Hyperparameter	***Dataset 1***	***Dataset 2***
Hidden layer 1 neurons	40	80
Hidden layer 2 neurons	20	40
Optimizer	Adamax	Adam
L1 regularization penalty	1e-8	1e-8
L2 regularization penalty	1e-6	1e-7
Dropout rate in hidden layer 1	0.5	0.0
Dropout rate in hidden layer 2	0.0	0.2

3.4 Reinforcement Learning:

Two model-free algorithms are used to train the agent, Q Learning and SARSA. Q learning directly learns the optimal policy, which may learn the optimal solution faster than SARSA, but may suffer problems in convergence. SARSA is more conservative than Q Learning, it learns near the optimal policy and avoids the paths that near termination state (including the optimal path). This makes SARSA a safer algorithm but slower to converge. In general, SARSA performs better than Q Learning.

The gamma parameter is set to be 1 in updating the action value function. The epsilon parameter is set to be 0.1. It should be a small value between 0 and 1 but should not be too small to ensure that the agent can balance between exploration and exploitation. Different number of episodes and learning rate (step size) are tested to determine the best one for the agent. As the reward is 1 for all actions, the evaluation method is to sum the total rewards received, this indicates how many episodes the agent can survive. The maximum reward is 500.

50000 episodes are experimented to limit the training time, with learning rate 0.1, 0.3, 0.5, 0.7. The best parameters are shown below.

Table 4: Reinforcement Hyper parameters

Hyper parameter	***Q Learning***	***SARSA***
Episodes	50000	50000
Learning Rate	0.3	0.3

4. Model evaluation / Experiments

Evaluating the model using various evaluation metrics is a crucial step to determine whether the model works well on unseen data. The evaluation metrics are needed to quantify the model performance. Accuracy, F1 score, Precision, Recall, ROC and Precision-recall curve are used to evaluate the performance of various learning algorithms used for the different experiments.

4.1 Experiment 1

In Experiment 1, the Company bankruptcy dataset is used for the model evaluation. The learning algorithms are compared with the imbalanced (original) dataset and the balanced (SMOTE) dataset.

4.1.1 Null Hypothesis 1

The learning algorithms perform better on balanced dataset (SMOTE) compared to imbalanced (original) dataset thus producing better model prediction. The Null hypothesis is true when the different learning algorithms show a better model prediction based on evaluation metrics.

4.1.2 Material & Methods 1

The dataset is split into train and test data in the ration of 80:20. The train data is further split into train and validation data. So overall the split ratio is 64:16:20. The learning algorithms Decision tree, SVM and DNN are trained using balanced dataset and imbalanced dataset. The Decision tree and SVM are tuned using GridSearchCV() to get the best parameters. The DNN is also hyper parameter tuned to get the best model. The Table 1, 2 and 3 in Section 3 provides details of the hyper parameters.

4.1.3 Results & Discussion 1

The experiment 1 result shows that each of the learning algorithm has performed better with balanced dataset than the imbalanced dataset. Hence the null hypothesis is accepted. The evidence of the results is shown below:

SVM:

Model	***Accuracy***	***F1 Score***	***Precision***	***Recall***
SVM with SMOTE	0.967742	0.513536	0.5	0.022727
SVM with imbalanced data	0.967742	0.491803	0.0	0.000000

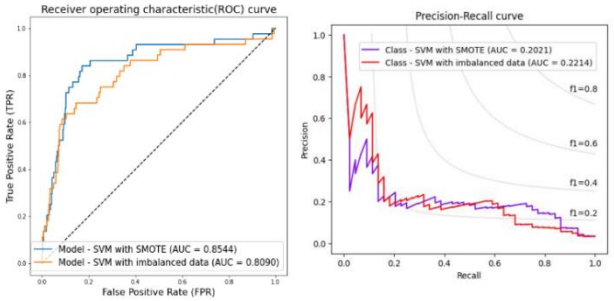


Figure 7: ROC and Precision-Recall curve for SVM

DT:

Model	*Accuracy*	*F1 Score*	*Precision*	*Recall*
DT with SMOTE	0.905425	0.597582	0.165354	0.477273
DT with imbalanced data	0.967742	0.491803	0.000000	0.000000

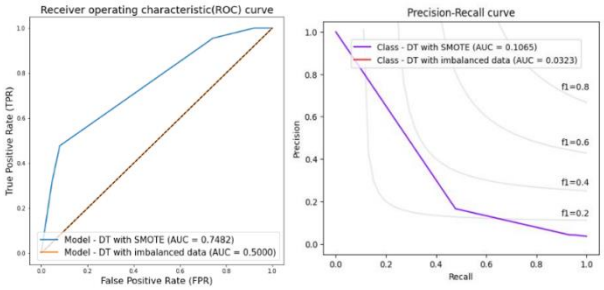


Figure 8: ROC and Precision-Recall curve for DT

DNN:

Model	*Accuracy*	*F1 Score*	*Precision*	*Recall*
DNN with SMOTE	0.956012	0.631511	0.300000	0.272727
DNN with imbalanced data	0.964809	0.577218	0.357143	0.113636

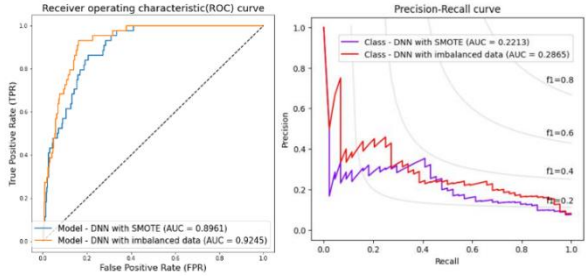


Figure 9: ROC and Precision-Recall curve for DNN

4.2 Experiment 2

In Experiment 2, the evaluation metrics of different algorithms are compared between different size of datasets.

4.2.1 Null Hypothesis 2

The size of the data does have an impact on the performance of different machine learning algorithms. It is expected that SVM performs better DNN when the data size is not large enough. The NULL hypothesis is true when different learning algorithm performs the best on the datasets.

4.2.2 Material & Methods 2

The performance of Decision Tree, SVM and DNN on datasets of two different sizes are compared. The imbalance of both the datasets have been dealt with by using SMOTE. The Hyperparameters of the learning algorithms are tuned with respect to both the datasets as seen in table 1, 2 & 3.

To effectively compare different models, Gaussian naïve bayes with default hyperparameters in scikit-learn sets a basis to compare the metrics of the supervised learning algorithms. A baseline model is usually a simple model that can be trained in a minimum amount of time, and it is easy to deploy in production environment. A baseline result could suggest a trivially attainable performance, which could be beat by any fine-tuned model. Therefore, if the model performs worse than the baseline, the algorithm might be not suitable for that dataset.

4.2.3 Results & Discussion 2

	Dataset 1 (bankruptcy)				*Dataset 2 (rain)*			
Metrics	*Base*	*DT*	*SVM*	*DNN*	*Base*	*DT*	*SVM*	*DNN*
Accuracy	0.89	+0.02	+0.08	+0.07	0.59	+0.16	+0.15	+0.21
F1 score	0.60	-0.01	-0.09	+0.03	0.56	+0.12	+0.14	+0.16
ROC AUC	0.88	-0.13	-0.02	+0.02	0.70	+0.04	+0.10	+0.12
PR AUC	0.21	-0.10	-0.01	+0.01	0.42	-0.02	+0.16	+0.18

F1 score, area under the receiver operating characteristic curve (ROC-AUC) and area under the precision-recall curve (PR-AUC) are the three chosen metrices for comparing the performances of different algorithms in different sizes of data. PR-AUC and f1-score represent a good tradeoff between precision and recall scores, while the ROC curve represents the tradeoff between specificity and sensitivity.

For dataset 1, we can infer from the table above that DT & SVM perform worse than the baseline model in all 3 metrics. DNN, the only model performs better than the baseline model, is the best model in handling the highly biased medium size dataset. For dataset 2, SVM and DNN dominate the baseline model in all metrices, and DT also

performs better than the baseline in general, with a slight lacking in PR AUC.

DNN has outperformed all models in both the datasets. Thus, the null hypothesis is rejected. However, the comparison might be limited by the fact that the nature of the two datasets is different, for example there are more categorical data in dataset 2 and there are more features in dataset 1.

4.3 Experiment 3

Reinforcement Learning

4.3.1 Null Hypothesis 3

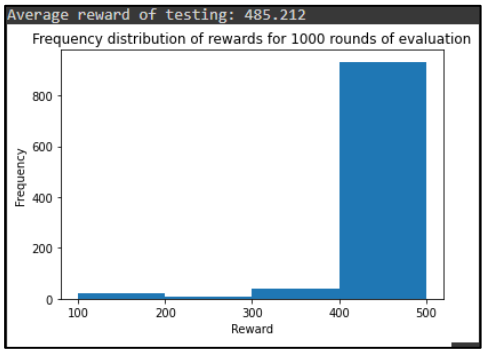
Generally, SARSA performs better than Q Learning. The null hypothesis is true when SARSA performs better than Q Learning in the CartPole-v1 environment.

4.3.2 Material & Methods 3

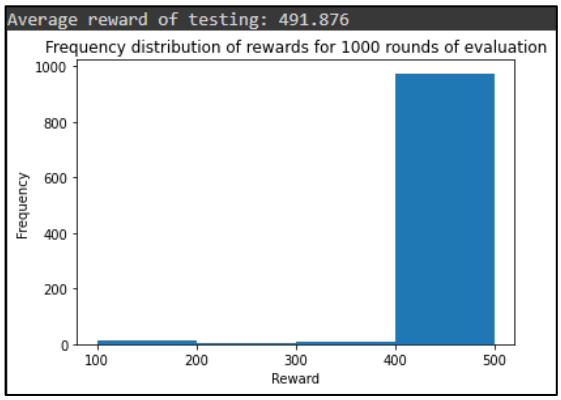
1000 rounds of test are performed for each algorithm. The average of 5 rounds of training is used for comparison. The higher average number of rewards received indicates the better performance achieved. The maximum reward for each round is 500.

4.3.3 Results & Discussion 3

Result of Q Learning (50000 episodes, 0.3 learning rate):



Result of SARSA (50000 episodes, 0.3 learning rate):



From the above experiment, both algorithms perform great, while SARSA performs slightly better than Q Learning. The average reward of SARSA is around 492 while Q Learning is around 485. This matches the null hypothesis which states that SARSA performs better than Q Learning. Assuming the mean of Q Learning is 485, the standard deviation is 4.69, by conducting a statistical test, the probability of the mean of Q Learning is equal to SARSA is smaller than the 5% significant level ($z>3.27$). Therefore, the mean of SARSA is greater than Q Learning at 5% significant level and the null hypothesis stated in 4.3.1 is true.

4.4 Experiment 4

4.4.1 Null Hypothesis 4

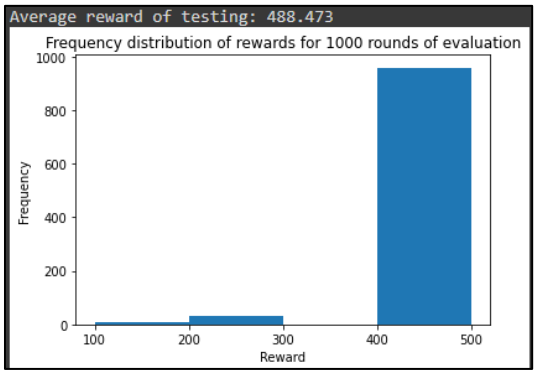
The performance of Q Learning does not drop significantly but the performance of SARSA drops significantly when the number of episodes decreases.

4.4.2 Material & Methods 4

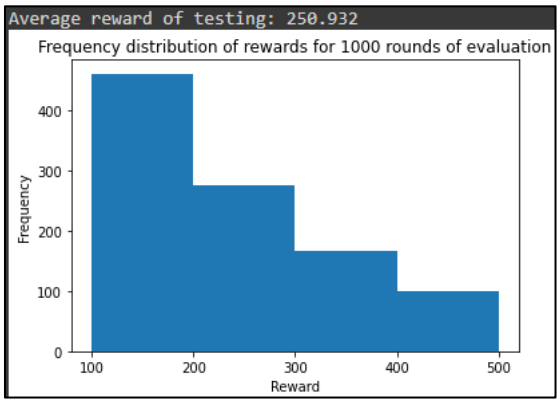
The same testing method in above section is used but with 25000 training episodes only.

4.4.3 Results & Discussion 4

Result of Q Learning (25000 episodes, 0.1 learning rate):



Result of SARSA (25000 episodes, 0.3 learning rate):



The results above show that Q Learning outperforms SARSA when the number of training episode reduces. This does not affect the performance of Q Learning, but the performance of SARSA drops significantly, the average reward achieved is 250 only, which has a huge difference compared to the reward achieved in the model trained with 50000 episodes. Therefore, the hypothesis is true. A possible reason for this is Q Learning always learn the optimal policy while SARSA does not. Q Learning can find the optimal solution in a shorter period while SARSA requires a longer period to learn. Therefore, if time is limited or cost is expensive, Q Learning is a better choice.

5. Discussion of the results, interpretation and critical assessment

The model evaluation experiments 1 and 2 prove that the DNN model to be the best model and it outperforms the other learning algorithms used. This algorithm performs better on balanced and different size datasets. Although the decision tree does not outperform the other algorithms, it has the advantage of interpretability by humans unlike SVM and DNN.

Hypothesis testing is not well-implemented in the experimental section. Hypothesis testing is achieved by finding the confidence interval of the statistical problem and reject the null hypothesis if it falls out of the confidence interval. No confidence interval could be obtained as there is only one set of testing data. To avoid using seen data in the testing data, k-fold cross validation is only implemented in the hyperparameter tuning but not in the model comparison. There, there is only one set of testing data in each dataset. Further research should be done on how to implement hypothesis testing correctly in machine learning.

In reinforcement learning, we have assumed that the results in the 5 trials are following the normal distribution when calculating confidence interval. In general, a sample size of 30 is sufficient to form a normal distribution. However, it is too computationally expensive to run over 30 trials of Q-learning and SARSA. It can be solved in the future if there is better equipment or sufficient time.

For reinforcement learning, the performance of SARSA is better than Q Learning in general. In the CartPole-v1 environment, the difference between the two algorithms is not huge while the number of training episodes is large enough for SARSA to converge. However, when the number of training episode decreases, the performance of Q Learning is not affected while the performance of SARSA reduced significantly. Therefore, Q Learning is a better algorithm when the cost of training is expensive, otherwise, SARSA would be better.

6. Conclusions

The classification problems – Bankruptcy prediction and Rain tomorrow prediction was performed using three learning algorithms with tuned hyper parameters. Various experiments were performed on imbalanced and balanced datasets and the algorithms were compared based on various evaluation metrics on different size of datasets. The results of the experiments proved that Deep neural networks (DNN) performed better than the other algorithms.

For reinforcement learning, the performance of both algorithms is similar, while SARSA performs slightly better than Q Learning in general. Q Learning can retain good performance when the number of training episodes is reduced but SARSA does not. Q Learning is a good option for the CartPole-v1 environment if time can cost are limited. Otherwise, SARSA is a better option in general.

Contributions

AK01928 – Dataset 1 – visualization**,** Dataset 2 – Data preparation, SVM model for both the datasets, Model evaluation, Report – Abstract, Section 2.1, 3.2, 4.2, Proof-reading the final report.

PK00580 – Dataset 2 – Data preparation, Decision tree model for both the datasets, Model evaluation function (Confusion matrix, metrics table), Report – Section 1, 2.2, 3.1, 4.1, 6. Formatting and final proof reading.

FT00247 – Dataset 1 – Data preparation, DNN model for both the datasets, Model evaluation function for Experiment 1 and 2, Report – Section 3.3, 4.2, 5. Proof-reading the final report.

TC01022 – Reinforcement learning – Coding and testing, Report – Section 1.3, 2.3, 3.4, 4.3, 4.4, 5. Proof-reading the final report.

References

[1] Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization