62_CayoPletikosicRavuril_Exercise1_Report

Cayo Ventura Cesar, Vice Pletikosic, Ravuri Charan Naveen Kumar

November 2025

Contents

1 Introduction

This report presents the results of Exercise 1: **Classification**, part of the *Machine Learning 2025W* course. The main goal of this exercise is to perform classification using multiple algorithms across several datasets, experimenting with preprocessing strategies, parameter tuning, and evaluation techniques. We aim to compare algorithm performance, investigate the effect of preprocessing, and analyze runtime differences.

All experiments were implemented in Python using the `scikit-learn` framework. The analysis followed a systematic pipeline approach to ensure reproducibility across all datasets and models.

Work distribution. The project work was shared equally among the three team members, with each focusing on different but complementary aspects of the assignment:

- **Cayo Ventura Cesar** was responsible for selecting and preparing the datasets, including data acquisition, cleaning, and preprocessing.
- **Ravuri Charan Naveen Kumar** implemented the experimental pipeline, selected the models, and performed the training and evaluation of the classifiers.
- **Vice Pletikosic** compiled the final report, verified experiment consistency, and ensured that all required tasks and deliverables were fulfilled as accurately as possible.

This division of work enabled the team to collaborate efficiently while maintaining quality and coherence across all project components.

2 Datasets

The datasets vary in sample size, dimensionality, and data type. Two datasets originate from Kaggle competitions, and two are benchmark datasets for general classification.

2.1 Dataset 1: Energy Efficiency (UCI ID 242)

The **Energy Efficiency** dataset was obtained from the UCI Machine Learning Repository (`https://archive.ics.uci.edu/dataset/242/energy+efficiency`). It originates from a study on assessing the heating and cooling load requirements of residential buildings as a function of structural parameters such as

wall area, roof area, and glazing configuration. The dataset consists of 768 samples with eight features and two continuous targets representing heating load (Y_1) and cooling load (Y_2). In our exercise we focused on the heating load (Y_1), which we discretized into three categories (*low*, *medium*, and *high*) using quantile binning, thus converting the regression task into a classification problem.

Feature Summary. The predictors X_1-X_8 include both continuous and integer variables:

- X_1-X_5, X_7: Continuous features (for example, surface area, wall area, overall height).
- X_6, X_8: Integer-coded categorical variables (orientation and glazing area distribution).

Preprocessing. No missing values were present. Categorical attributes (X_6, X_8) were one-hot encoded with `drop_first=True` to avoid collinearity. Continuous features were standardized using `StandardScaler` (mean = 0, standard deviation = 1) to ensure that all dimensions contributed equally and to improve the convergence of gradient-based classifiers such as SVM and MLP. The data were split into training and test sets using an 80/20 stratified split to preserve the proportion of the three classes.

Exploration. Class frequencies after discretization were approximately uniform across the three categories, so no class imbalance techniques were required. The final preprocessed dataset therefore consisted of $n = 768$ instances, 8 original features expanded to 12 after dummy encoding, and 3 target classes.

Remarks. This dataset serves as a compact, noise-free benchmark to test whether models capture relationships between continuous physical features and categorical energy efficiency levels. It also provides a clean environment to observe the influence of feature scaling on SVM and Neural Network performance.

2.2 Dataset 2: PhiUSIIL Phishing URL (UCI ID 967)

The **PhiUSIIL Phishing URL** dataset was obtained from the UCI Machine Learning Repository (`https://archive.ics.uci.edu/dataset/967/phiusiil+phishing+url+dataset`). It contains features derived from both

the URL string and the HTML source of each webpage, designed for detecting phishing websites. The dataset comprises 235,795 instances and 54 features of mixed types (real-valued, integer, and categorical). The target variable `label` is binary, indicating whether a URL is legitimate (0) or phishing (1).

Feature Summary. The feature set encodes structural, lexical, and content-based characteristics of each URL. Examples include character continuation rate, top level domain legitimacy probability, and URL title similarity metrics. Four features were stored as categorical attributes, while the remaining fifty were numeric. The column `FILENAME` was ignored as it represents only metadata.

Preprocessing. The dataset contained no missing values. Categorical features were one-hot encoded, and all numeric features were standardized to zero mean and unit variance. The preprocessing pipeline therefore consisted of an encoder scaler combination applied via a `ColumnTransformer`. Because of the dataset size, we used an 80/20 train test split without further resampling to ensure computational feasibility.

Exploration. The target classes were moderately imbalanced, with approximately 134,850 legitimate and 100,945 phishing samples (a 57:43 ratio). This imbalance was considered manageable and no oversampling was required. A feature correlation analysis was conducted to identify redundant or highly correlated predictors; however, the overall correlation between numeric features remained below 0.8, so all were retained for modeling.

Remarks. This dataset offers a challenging large-scale binary classification problem with heterogeneous feature types. It is useful for testing pipeline scalability, since models must handle both categorical and continuous features efficiently. It also allows evaluating runtime performance and overfitting tendencies on data with a real-world web security origin.

2.3 Dataset 3: MONK's Problems (UCI ID 70)

The **MONK's Problems** dataset is a classical benchmark for symbolic concept learning. It was created to evaluate the generalization ability of different induction algorithms on artificially generated logical rules. The dataset consists of three related subproblems (MONK-1, MONK-2, and MONK-3), each defined by a specific logical condition on six discrete attributes.

Data Description. The dataset contains a total of 432 instances, each described by six categorical attributes (`a1-a6`) with small discrete domains (`a1`, `a2`, `a4` in $\{1, 2, 3\}$, `a3`, `a6` in $\{1, 2\}$, `a5` in $\{1, 2, 3, 4\}$). The target variable `class` is binary (0 negative, 1 positive). All three MONK problems share this attribute structure but differ in their target rule definitions. In our study, we trained models on the combined dataset to evaluate generalization on symbolic data.

Preprocessing. The unique identifier column (`ID`) was removed. No missing values were present. Although the features are nominal, they were numerically encoded as integers and subsequently standardized using `StandardScaler` to ensure compatibility with algorithms such as SVM and MLP. The data were split into an 80/20 training test partition using stratified sampling to preserve the 1:1 class balance.

Exploration. The class distribution was perfectly balanced with 216 samples in each class. The low dimensionality and clean symbolic nature of the data make MONK's Problems a useful diagnostic dataset to analyze how different classifiers handle categorical inputs and generalize on small, noise-free datasets.

Remarks. The MONK dataset highlights the differences between models that rely on feature scaling and those that do not. SVM and the Neural Network benefited from scaling, while Random Forest performed consistently due to its tree-based structure. Because of its simplicity and balance, this dataset serves as a sanity check for correct pipeline configuration and model implementation.

2.4 Dataset 4: Default of Credit Card Clients (UCI ID 350)

The **Default of Credit Card Clients** dataset was obtained from the UCI Machine Learning Repository (`https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients`). It contains demographic and financial data from 30,000 Taiwanese credit card holders and is designed to predict whether a client will default on payment in the following month. This dataset provides a realistic, moderately imbalanced binary classification problem with a mixture of continuous and categorical features.

Data Description. The dataset includes 23 explanatory variables and one binary response variable (`Y`) indicating whether a default occurred (1) or not (0). Predictors cover demographic information (gender, education, marital status, age) as well as financial and behavioral factors such as credit limit, past payment history (`PAY_0` to `PAY_6`), bill amounts (`BILL_AMT1` to `BILL_AMT6`), and payment amounts (`PAY_AMT1` to `PAY_AMT6`). All 30,000 records are complete with no missing values.

Preprocessing. The unique identifier column (`ID`) was removed as it carries no predictive information. Categorical attributes (sex, education, marital status) were label encoded to integer form. All numeric features were standardized using `StandardScaler` (mean = 0, standard deviation = 1) to reduce variance magnitude differences between credit limits and transaction amounts. The dataset exhibits a moderate class imbalance with approximately 22% defaulters and 78% non-defaulters. Therefore, stratified sampling was applied to maintain class proportions in both training and test splits (80/20). No additional oversampling or SMOTE was required for the base experiment.

Exploration. Correlation analysis showed that consecutive bill and payment amount features (`BILL_AMT` and `PAY_AMT`) were highly correlated, reflecting the temporal nature of the data. However, all features were retained to remain consistent with related literature and to test model robustness to redundant predictors. The target distribution remained stable across train and test sets.

Remarks. This dataset serves as a large-scale, real-world benchmark for assessing classifier performance on mixed-type and moderately imbalanced financial data. Its size allows reliable runtime comparison across models. Tree-based and neural methods are expected to model non-linear dependencies effectively, whereas SVM performance may be limited by scale and redundancy. The dataset complements the others by introducing both scale and noise complexity.

3 Classification Methods

We evaluated three classifiers from different algorithmic families.

- **Support Vector Machine (SVM):** Margin-based classifier that identifies the optimal separating hyperplane between classes. Both linear

and RBF kernels were tested to capture linear and non-linear relationships.

- **Random Forest (RF):** Ensemble of decision trees trained on bootstrap samples. It reduces variance through averaging and performs robustly even with limited feature scaling.

- **Neural Network (NN):** Implemented as a Multilayer Perceptron (`MLPClassifier`) with one or two hidden layers, ReLU activation, and the Adam optimizer. It is capable of modeling complex, non-linear decision boundaries.

At this stage, no explicit hyperparameter tuning such as `GridSearchCV` was implemented. All models were trained using `scikit-learn` default configurations to establish baseline performance and to focus on comparing the intrinsic behavior of the classifiers across different datasets. In future extensions, hyperparameter optimization using `GridSearchCV` or `RandomizedSearchCV` could be applied within the same unified `Pipeline` framework to systematically explore parameter combinations and improve performance consistency. All experiments were executed with fixed random seeds to ensure reproducibility.

4 Experimental Setup

Each model was trained and evaluated using both 80/20 holdout and 5-fold cross validation. For each dataset, we measured accuracy, macro F1, and runtime to balance predictive and computational performance. Pipelines combined preprocessing and classification into a single training process, ensuring no data leakage between training and test folds.

5 Evaluation Metrics

Model performance was evaluated using both a single holdout split and k-fold cross validation. We reported a combination of overall and class wise performance measures, as well as training and inference runtime.

Holdout Evaluation. For the 80/20 train test split, the following metrics were computed:

- **Accuracy:** fraction of correctly predicted labels.

- **F1 (macro):** harmonic mean of precision and recall, averaged across all classes.
- **Train_time_s:** total model training time in seconds.
- **Infer_time_s:** average prediction time on the test set in seconds.

The corresponding output format was:

```
{"Accuracy": acc, "F1_macro": f1, "Train_time_s": train_time,
                "Infer_time_s": infer_time}
```

Cross Validation Evaluation. To assess model stability and generalization, we applied k-fold cross validation (typically $k = 5$). The mean and standard deviation across folds were computed for accuracy and F1 score:

```
{"CV_Accuracy_mean": acc.mean(), "CV_Accuracy_std": acc.std(),
       "CV_F1_mean": f1.mean(), "CV_F1_std": f1.std()}
```

Combined Output Format. The final summarized results were organized per dataset and classifier as shown below:

Table 1: Structure of reported evaluation metrics.

Dataset	Model	Accuracy	F1_macro	Train_time	Infer_time	CV_Acc_mean	C
					filled by results pipeline		

Metric Definition. The F1 score is defined as:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{1}$$

Runtime Measurement. Training and inference durations were measured using the `time.perf_counter()` function and averaged over repeated runs for consistency.

6 Results and Analysis

6.1 Performance per Dataset

6.1.1 Dataset 1: Energy Efficiency

Table 2: Performance Comparison on Dataset 1 (Energy Efficiency)

Classifier	Accuracy	F1-score	CV_Acc_mean	CV_F1_mean
SVM (RBF)	0.701	0.681	0.686	0.671
Random Forest	**0.896**	**0.896**	**0.889**	**0.890**
Neural Network	0.734	0.728	0.732	0.728

Trend: Random Forest achieved the best overall performance with accuracy and F1 near 0.89. SVM underperformed slightly, while the Neural Network required more time to train but remained competitive.

6.1.2 Dataset 2: PhiUSIIL Phishing URL

Table 3: Performance Comparison on Dataset 2 (PhiUSIIL Phishing URL)

Classifier	Accuracy	F1-score	CV_Acc_mean	CV_F1_mean
SVM (RBF)	0.9997	0.9997	0.9998	0.9998
Random Forest	**1.0000**	**1.0000**	**1.0000**	**1.0000**
Neural Network	0.99996	0.99996	0.99995	0.99994

Trend: All models achieved near perfect accuracy due to clear separability and engineered URL features. Random Forest was marginally best.

6.1.3 Dataset 3: MONK's Problems

Table 4: Performance Comparison on Dataset 3 (MONK's Problems)

Classifier	Accuracy	F1-score	CV_Acc_mean	CV_F1_mean
SVM (RBF)	0.885	0.883	0.914	0.914
Random Forest	**1.000**	**1.000**	0.993	0.993
Neural Network	**1.000**	**1.000**	**1.000**	**1.000**

Trend: Neural Network and Random Forest achieved perfect scores; SVM remained slightly lower. Runtime differences were negligible due to the small dataset.

6.1.4 Dataset 4: Default of Credit Card Clients

Table 5: Performance Comparison on Dataset 4 (Default of Credit Card Clients)

Classifier	Accuracy	F1-score	CV_Acc_mean	CV_F1_mean
SVM (RBF)	**0.816**	0.669	**0.819**	0.670
Random Forest	0.812	0.671	0.816	0.680
Neural Network	0.810	**0.676**	0.814	**0.676**

Trend: SVM reached the highest accuracy, while the Neural Network slightly surpassed others in F1, indicating stronger class balance handling.

6.2 Cross-Dataset Summary and Analysis

To obtain an overall comparison across datasets, we computed the mean metrics per classifier, aggregating accuracy, F1 score, and runtime performance over all four datasets.

Table 6: Cross-Dataset Summary of Classifier Performance

Classifier	Mean Acc.	Mean F1	Train (s)	Infer (s)	CV Acc. Mean	CV
Random Forest	**0.927**	**0.892**	11.978	0.162	0.925	
Neural Network	0.886	0.851	20.585	0.029	0.886	
SVM (RBF)	0.851	0.808	46.011	7.963	0.855	

Table 7: Accuracy comparison per dataset.

Dataset	NeuralNet	RandomForest	SVM (RBF)
Dataset 1 (Energy Efficiency)	0.734	**0.896**	0.701
Dataset 2 (Phishing URL)	**1.000**	**1.000**	**1.000**
Dataset 3 (MONK's Problems)	**1.000**	**1.000**	0.885
Dataset 4 (Credit Default)	0.810	0.812	**0.816**

Accuracy per Dataset.

Table 8: Macro F1 comparison per dataset.

Dataset	NeuralNet	RandomForest	SVM (RBF)
Dataset 1 (Energy Efficiency)	0.728	**0.896**	0.681
Dataset 2 (Phishing URL)	**1.000**	**1.000**	**1.000**
Dataset 3 (MONK's Problems)	**1.000**	**1.000**	0.883
Dataset 4 (Credit Default)	**0.676**	0.671	0.669

F1 per Dataset.

Note on perfect scores. Datasets 2 (Phishing URL) and 3 (MONK's Problems) yielded perfect accuracy and F1 scores (1.000) across all classifiers. For MONK's Problems, this reflects the dataset's deterministic rule based nature. For the Phishing URL dataset, the perfect performance likely indicates that the engineered URL based features are highly discriminative, though potential data leakage or redundancy between train and test splits cannot be ruled out. These results should be interpreted with caution when assessing model generalization.

7 Discussion

The comparative analysis shows that each classifier demonstrates distinct strengths and weaknesses depending on the data characteristics and preprocessing steps applied. In general, model performance was consistent with theoretical expectations.

Random Forest. Random Forest achieved the highest overall accuracy and F1 across datasets. Its robustness can be attributed to ensemble averaging, which reduces variance and handles mixed feature types effectively. It performed reliably without requiring feature scaling and was the most stable model across different dataset sizes and domains. The moderate training time and low inference time make it a strong default choice.

Neural Network. The Neural Network (MLP) produced competitive results and outperformed other models on the imbalanced Credit Default dataset in terms of F1. This indicates that the model captured complex, nonlinear

relations between variables. However, the longer training times and sensitivity to initialization and hyperparameters highlight its higher computational cost. Since no grid search was used, further optimization could improve its performance.

Support Vector Machine. SVM performed well on smaller and well scaled datasets such as Energy Efficiency. For high dimensional or large datasets like Phishing URL, it was significantly slower to train and showed limited scalability.

Effect of preprocessing and evaluation. Feature scaling, encoding, and stratified splitting had a noticeable impact on performance. SVM required normalization to achieve stable convergence, while Random Forest was largely insensitive to scaling. Cross validation produced more reliable and less variable performance estimates than a single holdout split, especially for small datasets. Although we did not apply GridSearchCV, the use of unified pipelines allowed for consistent preprocessing and reproducible results.

Interpretation of perfect scores. Perfect scores on Datasets 2 and 3 are plausible due to separability and deterministic rules. Nevertheless, they warrant caution and careful validation of preprocessing to avoid leakage.

Summary. Overall, Random Forest provided the best trade off between performance, robustness, and runtime efficiency. Neural Networks showed potential for improvement with more hyperparameter tuning, while SVMs remain valuable for smaller datasets with clean, continuous features.

8 Conclusion

This exercise provided practical experience in applying and comparing different classification algorithms across multiple datasets with diverse properties. By implementing Support Vector Machines, Random Forests, and Neural Networks within unified preprocessing pipelines, we observed how data characteristics and feature preparation influence model behavior and outcomes.

Random Forest emerged as the most reliable and generalizable classifier, combining high accuracy with low variance and efficient runtime. The Neural Network demonstrated strong potential for handling nonlinear and imbalanced data, though it required more computational resources and could

benefit from further tuning. SVM achieved strong results on smaller, well structured datasets but struggled with scalability on large or high dimensional inputs.

The exercise highlighted the importance of preprocessing and systematic evaluation using cross validation. Perfect scores observed in some datasets reinforced the need for cautious interpretation and validation of model generalization.

Future work could include extending hyperparameter optimization with `GridSearchCV`, exploring regularization techniques, and testing additional ensemble or deep learning architectures for improved performance on more complex datasets.

A Appendix

- Zip file with all source code and notebooks used in the experiments.