Machine Learning

Exercise 1: Classification

Dzhamilia Kulikieva (12340251)
Fabian Ombui (12326236)
Ali Hassan (12448698)

April 2025

Contents

Introduction

In this project, we explore the behavior of different classification algorithms across four diverse datasets: Amazon Reviews, Congressional Voting Records, Hepatitis Domain, and Austin Animal Shelter Outcomes.

Our goal is to evaluate and compare the performance of three chosen distinct classifiers—k-Nearest Neighbors, Decision Trees, and Multi-Layer Perceptron (MLP)—on tasks with varying data structures, domains, and preprocessing challenges. The report details the datasets, preprocessing steps, classifiers used, experimental setup, and results analysis.

1 Datasets

1.1 Overview

List and name all 4 datasets. Provide a table summarizing their characteristics:

Table 1: Summary of the selected datasets

Dataset	Samples	Features	Classes	Domain	Preprocessing
Amazon Reviews	750	10000	50	E-commerce	Scaling, missing values
Congressional Voting Records	435	16	2	Politics / Social Science	Encoding, scaling
Hepatitis Domain	155	20	2	Jozef Stefan Institute	Missing values, scaling, encoding
Animal Shelter Outcomes	78256	10	2	Austin Animal Center	Encoding, date parsing, target binarization

1.2 Dataset 1: Amazon Reviews

The dataset contains Amazon product reviews and related metadata. Each entry represents a review left by a customer for a specific product. The goal is to predict the correct label based on the available review and product features.

1.3 Dataset 2: Congressional Voting Records

The dataset contains the voting records of members of the U.S. House of Representatives from 1984. Each record represents a congressman's votes on 16 key issues, along with their political affiliation. The goal is to predict a congressman's political party based on their voting behavior across the 16 issues.

1.4 Dataset 3: Hepatitis Domain

The Hepatitis Domain dataset is a clinical dataset compiled for the study of liver-related illnesses. It contains patient data used to predict clinical outcomes, with the primary goal of classifying cases as either "LIVE" or "DIE."

1.5 Dataset 4: Austin Animal Center Shelter Outcomes

The AAC Shelter Outcomes dataset contains animal-related information such as species, breed, color, age, and outcome type. The dataset supports initiatives in animal welfare by helping to analyze trends and improve shelter practices through data-driven insights. The goal of our project is to predict whether a dog or cat will be successfully adopted.

Dataset Variety and Justification We selected these datasets to ensure a diverse range of domains, data structures, and learning challenges. They differ in terms of size, feature types, number of classes, and preprocessing needs.

- The Hepatitis Domain dataset presents medical data with missing values, requiring imputation and careful handling.
- The Animal Shelter Outcomes dataset involves real-world animal rescue operations, combining categorical and temporal features, and requires target binarization.

2 Preprocessing

2.1 Amazon reviews

For the custom dataset exploration, we employed a dataset descriptor tailored to our specific needs. The dataset comprises 750 rows and an extensive 10,002 columns, with no missing values detected. Given the dataset's scale and structure, feature scaling was deemed necessary as the primary preprocessing step.

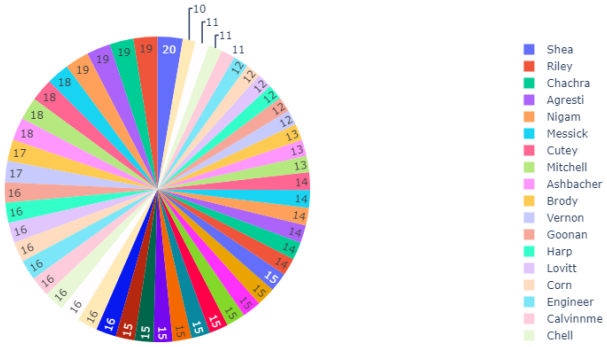

Figure 1: Distribution of labels in percentage (Unscaled)

To maintain integrity and balance during the division of the dataset into training and test sets, we utilized a stratified shuffling approach. This method ensures that each class attribute is proportionately represented in the training and test datasets, mitigating the risk of bias or skewed performance evaluation. We aim to achieve robust and unbiased model training and evaluation processes by stratifying the data based on class labels.

2.2 Congressional Voting Records

In the first two models (k-NN and Decision Tree), missing votes marked as '?' were assigned a placeholder value (2), while for the MLP model, missing values were explicitly set as NaN and later imputed using the most frequent value per feature.

The target variable (Class Name), representing party affiliation, was label-encoded into binary format: Democrat as 0 and Republican as 1. The dataset was then split into training and testing sets using a 70/30 ratio with a fixed random state for reproducibility.

For the MLP model, an additional preprocessing step was applied: feature standardization using StandardScaler, to ensure the features were centered and scaled properly, which is critical for neural networks.

Thus, while the core steps of handling missing data, encoding labels, and splitting data were common across all models, the need for imputing missing values and scaling was introduced specifically for MLP to optimize performance.

2.3 Hepatitis Domain

We performed visual outlier detection using boxplots for all numerical features with missing values.

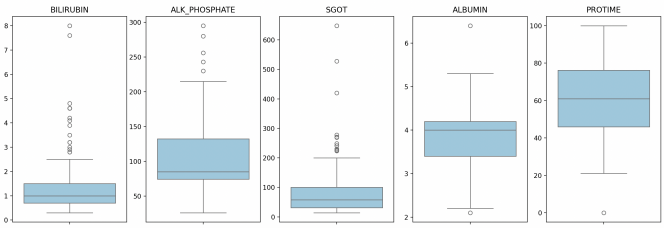

Figure 2: Boxplots of numerical values

We handled missing values using mean or median imputation, depending on the distribution of each feature. Median was used for skewed features (BILIRUBIN, ALK_PHOSPHATE, SGOT), and mean for ALBUMIN. For PROTIME, which had many missing values, we added a binary indicator (PROTIME_missing) before applying median imputation.

As both k-NN and MLP are sensitive to extreme values, we applied a logarithmic transformation using $\log(1 + x)$ to the features SGOT, ALK_PHOSPHATE, and BILIRUBIN to reduce the impact of high-value outliers and smooth right-skewed distributions. This is especially helpful for distance-based and probabilistic models. The features ALBUMIN and PROTIME showed only a small number of outliers and fairly symmetric distributions. Therefore, we decided not to apply any outlier transformation to them.

We mapped categorical string features to binary numeric values using a defined dictionary. After the mapping, several columns (STEROID, FATIGUE, MALAISE, ANOREXIA, LIVER_BIG, LIVER_FIRM, SPLEEN_PALPABLE, SPIDERS, ASCITES, VARICES) now have missing values (NaN). To handle the missing values that appeared, we imputed the missing entries using the most frequent value (mode) for each affected column.
No multicollinearity issues were detected, as no correlation exceeds ± 0.9.

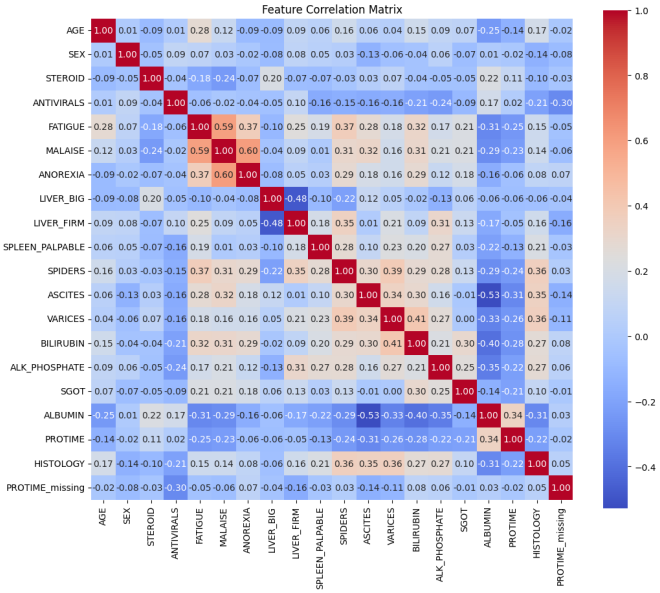

Figure 3: Correlation matrix

Some moderate positive correlations are observed between:

- FATIGUE and MALAISE (r = 0.59)
- SPLEEN_PALPABLE and LIVER_BIG (r = 0.48)

- SPIDERS, ASCITES, and VARICES show inter-correlations around r = 0.30–0.40, suggesting potential clinical co-occurrence.

Most numerical lab features (e.g., BILIRUBIN, ALBUMIN, PROTIME) are relatively independent, making them suitable for inclusion in modeling without high risk of redundancy.

Since the dataset is imbalanced (123 LIVE vs 32 DIE), we used stratified sampling during the train-test split (stratify=y) to ensure that both training and test sets preserve the original class distribution.

```
Train shape: (124, 20), Test shape: (31, 20)
Train target distribution:
 Class
0    0.790323
1    0.209677
Name: proportion, dtype: float64
Test target distribution:
 Class
0    0.806452
1    0.193548
Name: proportion, dtype: float64
```

2.4 Animal Shelter Outcomes

Preprocessing steps for this Dataset included converting age and date columns into numerical features, encoding categorical variables (e.g., sex, breed, color), handling missing values, creating derived features (such as presence of a name), and scaling numerical data to ensure compatibility with distance-based and neural network classifiers.

All categorical features were one-hot encoded to convert them into numeric format suitable for K-NN and MLP models. We used pd.get_dummies() with drop_first=True to avoid multicollinearity and reduce dimensionality. age_upon_outcome was converted from textual format into numerical days to provide a continuous, model-friendly feature. The datetime column represents the timestamp of the animal's outcome event. We extracted time-based features such as hour, weekday, month, and year to capture potential temporal patterns that might influence adoption or other outcomes.

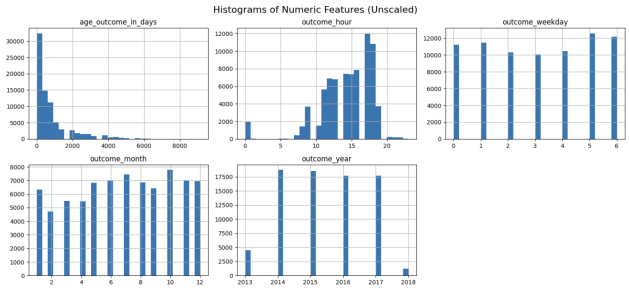

Figure 4: Histograms of Numeric Features (Unscaled)

As we see from the diagrams, most outcomes occurred during daytime and weekends. Age is strongly skewed toward younger animals. To reduce skewness, we applied a log transformation to age_outcome_in_days. All numeric features were then standardized using StandardScaler to ensure compatibility with distance-based and neural network models. While all time-related features originate from the same timestamp, they represent distinct temporal aspects (e.g., hour vs. year). No logical redundancy or high correlation was observed, so all were retained.

To predict whether an animal had a successful outcome, we converted the multiclass target outcome_type into a binary classification task. We defined Adoption and Return to Owner as positive outcomes (1), while all other types (e.g., Transfer, Euthanasia) were considered negative (0). The resulting binary target was stored in the column outcome_binary, and the original outcome_type column was dropped.

Missing values were handled by imputing the median for the numeric feature age_outcome_in_days. The name column was converted to has_name. We dropped irrelevant or redundant object columns, including IDs, dates, and subtypes. After extracting time features, only numerical columns remained, and the data was ready for modeling. Finally, we split the data into X (features) and y (binary target outcome_binary), to clearly separate input variables from the prediction target.

3 Classifiers

3.1 Overview

We selected three classifiers from distinct learning paradigms: *k-Nearest Neighbors* (instance-based), *Decision Trees* (tree-based), and *MLP* (neural network). This choice ensures diversity in algorithmic principles, making our comparison more meaningful in terms of model behavior, preprocessing sensitivity, and performance across datasets.

3.2 Classifier 1: k-Nearest Neighbors

An instance-based method relying on distance metrics to classify new samples based on nearby points.

3.3 Classifier 2: Decision Trees

A tree-based model that splits data by feature thresholds to create interpretable decision rules.

3.4 Classifier 3: MLP

A neural network model using layers of neurons and nonlinear activation functions to learn complex patterns.

4 Experimental Setup

Dataset	Validation Approach	Metrics Used	Parameter Tuning	Tools/Versions
Amazon Reviews	Holdout (80/20) & CV (5-fold)	Accuracy, Precision, Recall, F1, ROC AUC	Manual Grid Search (k, weights, metric)	scikit-learn 1.2.2, Python 3.11
Congressional Voting Records	Holdout (70/30)	Accuracy, Precision, Recall, F1	No tuning	scikit-learn 1.2.2, Python 3.11
Hepatitis Domain	Holdout (70/30) & CV (5-fold)	Accuracy, Precision, Recall, F1, ROC AUC	Manual Grid Search (depth, split)	scikit-learn 1.2.2, Python 3.11
Animal Shelter Outcomes	Holdout (80/20) & CV (5-fold)	Accuracy, Precision, Recall, F1, ROC AUC	Manual Grid Search (hidden layers)	scikit-learn 1.2.2, Python 3.11

Table 2: Experimental Setup Overview

5 Results

5.1 Performance per Dataset

5.1.1 Amazon reviews Domain

Metric	Decision Tree	MLP	KNN
Accuracy	0.367	0.640	0.293
Macro-Avg Precision	0.40	0.62	0.36
Macro-Avg Recall	0.35	0.62	0.30
Macro-Avg F1	0.36	0.59	0.30
Weighted-Avg Precision	0.42	0.62	0.36
Weighted-Avg Recall	0.37	0.64	0.29
Weighted-Avg F1	0.37	0.60	0.29

Table 3: Overall Performance on Test Set

Overall, the multilayer perception (MLP) clearly outperforms both the decision tree and the k-nearest neighbors (KNN) classifier across every metric. With an accuracy of 0.640, the MLP achieves nearly double the decision

tree's 0.367 and more than double KNN's 0.293.

In terms of balanced performance, the MLP's macro-averaged precision and recall (both 0.62) far exceed those of the decision tree (0.40 precision, 0.35 recall) and KNN (0.36 precision, 0.30 recall). Its macro-averaged F1 of 0.59 likewise outstrips the decision tree's 0.36 and KNN's 0.30, indicating that the MLP maintains strong performance even on less frequent classes.

Considering the dataset's class distribution, the weighted-average scores again favor the MLP: weighted precision of 0.62, weighted recall of 0.64, and weighted F1 of 0.60. The decision tree achieves moderate weighted averages (precision 0.42, recall 0.37, F1 0.37), while KNN lags behind (precision 0.36, recall 0.29, F1 0.29).

In summary, the MLP offers the best trade-off between overall accuracy and balanced, per-class performance. The decision tree provides a simple baseline but fails on many classes, and KNN shows limited effectiveness on this multi-class task.

5.1.2 Congressional Voting Records

In this dataset, the Decision Tree classifier achieved the best performance, with an accuracy of 0.98 and nearly perfect precision, recall, and F1-score. The MLP classifier also performed strongly, reaching an accuracy of 0.95 with balanced macro-averaged metrics. The k-NN classifier showed reasonable results with an accuracy of 0.92 but slightly lower F1-scores compared to Decision Tree and MLP.

Table 4: Performance of classifiers on the new dataset (Holdout Evaluation)

Model	Accuracy	Precision	Recall	F1-Score
k-NN	0.92	0.90 (macro)	0.92 (macro)	0.91 (macro)
Decision Tree	0.98	0.98 (macro)	0.98 (macro)	0.98 (macro)
MLP	0.95	0.95 (macro)	0.95 (macro)	0.95 (macro)

Overall, the Decision Tree outperformed the other two models on this dataset, benefiting from the dataset's small size, clean structure, and categorical features.

5.1.3 Hepatitis Domain

Among the three classifiers tested, the MLP model achieved the highest performance in terms of ROC AUC (0.9133), along with solid accuracy and F1-score (both 0.8710 and 0.6667 respectively). The k-NN model performed reasonably well, with slightly lower AUC (0.7733) and significantly lower recall (0.3333), leading to a lower F1-score. The Decision Tree model showed the weakest performance overall, particularly in precision (0.4000) and ROC AUC (0.5300).

In terms of runtime, both k-NN and Decision Tree were faster (0.03–0.04 sec), while MLP required slightly more time (0.21 sec), which is expected due to its iterative nature.

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Runtime (sec)
MLP	0.8710	0.6667	0.6667	0.6667	0.9133	0.21
k-NN	0.8387	0.6667	0.3333	0.4444	0.7733	0.04
Decision Tree	0.7742	0.4000	0.3333	0.3636	0.5300	0.03

Table 5: Holdout evaluation results for three classifiers (MLP, k-NN, and Decision Tree).

While the table above reports holdout performance, all models were also evaluated via 5-fold cross-validation.

Model	CV Accuracy	CV Precision	CV Recall	CV F1 Score	CV ROC AUC
k-NN	0.7935	0.5467	0.3810	0.4403	0.6960
Decision Tree	0.7548	0.4277	0.4714	0.4422	0.6693
MLP	0.8452	0.6330	0.6619	0.6407	0.8500

Table 6: Cross-validation results (5-fold) for k-NN, Decision Tree, and MLP classifiers.

Cross-validation results confirmed that the MLP classifier outperformed k-NN and Decision Tree models across all major metrics. Compared to holdout evaluation, MLP achieved even higher stability, with a CV ROC AUC of 0.8500 and a balanced F1 score of 0.6407. k-NN showed reasonable accuracy (0.7935) but suffered from low recall (0.3810), while the Decision Tree model remained the weakest performer, with both lower precision and ROC AUC.

5.1.4 Animal Shelter Outcomes

The Decision Tree classifier with a depth of 10 and a minimum split of 5 achieved the highest F1-score of 0.8822, slightly outperforming the best k-NN and MLP models. MLP demonstrated strong performance as well, with an F1-score of 0.8674, but at the cost of significantly longer training time. k-NN achieved reasonable results, with the 5-neighbor version reaching an F1-score of 0.8673.

Table 7: Performance of classifiers (Holdout Evaluation)

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC	Runtime (s)
k-NN (k=3)	0.8190	0.8218	0.8959	0.8572	0.8531	68.65
k-NN (k=5)	0.8294	0.8209	0.9192	0.8673	0.8690	66.62
Decision Tree (depth=10, split=5)	0.8484	0.8344	0.9359	0.8822	0.8997	2.80
Decision Tree (depth=10, split=10)	0.8483	0.8341	0.9360	0.8821	0.9004	2.82
MLP (50, tanh)	0.8325	0.8344	0.9031	0.8674	0.8892	393.69

In cross-validation, Decision Trees again showed the strongest performance, achieving an F1-score of 0.8781. MLP models performed very competitively, with F1-scores around 0.854, slightly outperforming k-NN models, whose F1-scores were around 0.8370 and 0.8476 depending on the number of neighbors. Although training MLP models required significantly more time, cross-validation results were successfully obtained for the best configurations.

Table 8: Performance of classifiers (Cross-Validation Evaluation)

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
k-NN (k=3)	0.7979	0.8155	0.8611	0.8370	0.8339
k-NN (k=5)	0.8082	0.8145	0.8849	0.8476	0.8506
Decision Tree (depth=10, split=10)	0.8431	0.8302	0.9319	0.8781	0.8930
Decision Tree (depth=10, split=2)	0.8428	0.8303	0.9311	0.8778	0.8909
Decision Tree (depth=10, split=5)	0.8427	0.8301	0.9313	0.8778	0.8911
MLP (50,) tanh	0.8201	0.8362	0.8753	0.8541	0.8821
MLP (100,) tanh	0.8198	0.8362	0.8746	0.8541	0.8816
MLP (50,) relu	0.8177	0.8371	0.8691	0.8515	0.8784

Overall, Decision Trees consistently delivered the best balance between predictive performance and computational efficiency across both evaluation strategies.

5.2 Effect of Preprocessing

Preprocessing had a noticeable impact on model performance, particularly for MLP. Feature scaling improved the convergence and predictive power of the MLP classifier, which relies on gradient-based optimization. Imputing missing values in the Congressional Voting Records dataset also helped improve stability and consistency across

models. For Decision Trees, preprocessing such as scaling had little to no effect, as trees are inherently insensitive to feature magnitudes.

As expected, k-NN benefited from feature scaling, which was explicitly applied during preprocessing for the Amazon Reviews, Animal Shelter Outcomes, and Hepatitis datasets. This scaling contributed to achieving reasonable and stable performance across these datasets.

5.3 Cross-validation vs Holdout

Cross-validation generally resulted in slightly lower evaluation metrics compared to Holdout validation. This is expected, as cross-validation provides a more realistic estimate of model generalization by averaging performance over multiple folds, reducing the chance of overfitting to a single train-test split.

In particular, the Decision Tree model showed very close results between Holdout and Cross-Validation, indicating stable behavior. k-NN models had slightly larger differences, which may be due to variability in local neighborhood structures across different folds. MLP cross-validation could not be fully evaluated due to extremely long computation times.

5.4 Summary Table

Table 9 summarizes the best-performing models and their evaluation metrics across all datasets. For each dataset, the classifier that achieved the highest performance based on F1-score and overall balance of metrics was selected. Metrics such as accuracy, macro-averaged precision, recall, and F1-score are reported, and ROC AUC is provided when applicable. In multi-class settings like Amazon Reviews, ROC AUC was not computed and is therefore omitted.

Table 9: Best results per dataset and classifier (Holdout Evaluation)

Dataset	**Model**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**ROC AUC**
Amazon Reviews	MLP	0.640	0.62 (macro)	0.62 (macro)	0.59 (macro)	–
Congressional Voting Records	Decision Tree	0.980	0.98 (macro)	0.98 (macro)	0.98 (macro)	–
Hepatitis Domain	MLP	0.8710	0.6667	0.6667	0.6667	0.9133
Animal Shelter Outcomes	Decision Tree	0.8484	0.8344	0.9359	0.8822	0.8997

6 Conclusion

Through our experiments, we found that model performance varied significantly depending on the dataset characteristics and preprocessing strategies. Decision Trees demonstrated consistent robustness, achieving the highest F1-score (0.8822) and ROC AUC (0.8997) on the Animal Shelter Outcomes dataset. MLP models showed superior performance on more complex tasks, notably outperforming others in the Amazon Reviews (Accuracy 0.640) and Hepatitis datasets (ROC AUC 0.9133), but required careful feature scaling and longer training times.

k-NN models performed reasonably well but were more sensitive to scaling and less effective on datasets with high feature dimensionality.

Overall, proper preprocessing and model selection based on dataset properties proved crucial to achieving strong classification performance.