Supervised Learning

Kirsten Odendaal

I. Introduction

Modern machine learning can be categorized into classification and regression problems. This paper focuses on classification, where the objective is to assign items to discrete labels based on their features. To investigate this, two unique classification problems are analyzed using various algorithms: k-Nearest Neighbours (kNN), Support Vector Classifier (SVC), and Multi-Layer Perceptron (MLP) neural networks. The NASA Near-Earth Objects and Wine Quality datasets, sourced from the Kaggle repository, were chosen for their unique characteristics and complexity. These datasets provide a testing ground for comparing performance, which offers insights into their strengths and limitations in handling real-world data scenarios.

A. Initial Hypothesis:

The general hypothesis (H) of the study is as follows;

- *k-Nearest Neighbors (kNN):* kNN will likely have the lowest predictive performance due to its simplicity and sensitivity to irrelevant features (curse of dimensionality). kNN only stores data, which is extremely fast; however, its prediction will be the slowest of all three algorithms, especially on larger datasets, due to its "lazy" [6] nature. For imbalanced datasets, kNN might struggle unless distance weighting is applied to disregard the effects of extreme outliers.
- *Support Vector Classifier (SVC):* The SVC is expected to excel on the binary classification datasets, given its internal ability to handle two-class maximal separation tasks. Training time is expected to be significant, especially with non-linear kernels and larger datasets. Furthermore, algorithm modifications are required for multi-class problems, likely worsening performance and making training times comparable to or worse than MLPs.
- *Multi-Layer Perceptron (MLP):* The MLP is anticipated to outperform the other algorithms due to its ability to learn complex smooth-continuous mappings, thus making them well suited for both binary and multi-class problems. However, this complexity will likely result in the slowest training times of all algorithms.

II. Dataset Explanation

The study investigates the use of two different datasets. The first focuses on the NASA Near-Earth Object (NEO) dataset, which provides detailed information on asteroids [3]. This source contains 4,687 instances and 40 features related to physical asteroid characteristics such as size, orbits, and orientations. The target feature, "Hazardous," is a binary indicator detailing whether the asteroid poses a potential threat to Earth.

Redundant and zero-variance features were removed from the NASA dataset, resulting in 17 features with a binary target class (safe or hazardous). Figure 1 summarizes the feature and target distributions. The NASA dataset provides an interesting foundation for model comparison due to the following challenges:

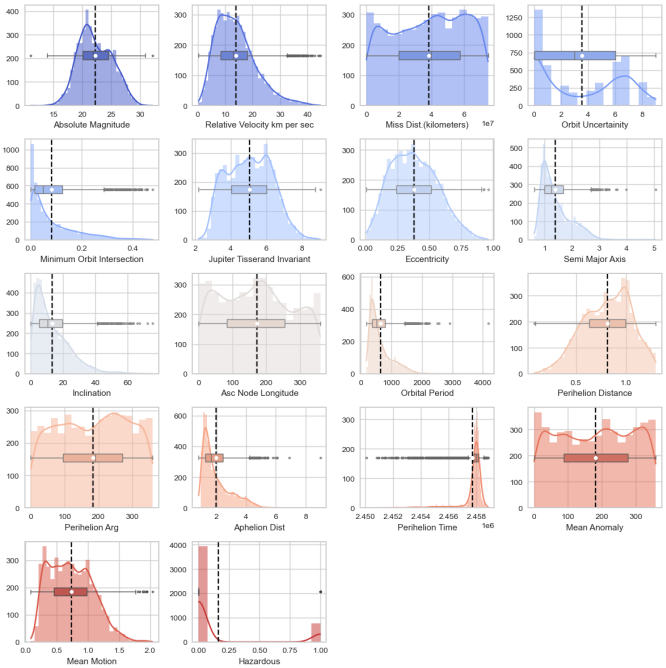


Fig. 1: Distributions of dataset 1: Nasa

1) *Variety of Features:* The dataset includes 17 distinct features detailing various aspects of asteroid properties, such as physical characteristics (e.g., absolute magnitude, estimated diameter), orbital parameters (e.g., eccentricity, semi-major axis, orbital period), and velocity metrics. This diversity allows for exploring different feature interactions and their impact on model performance.
2) *Class Imbalance:* The large class imbalance (83.89% non-hazardous and 16.10% hazardous) presents a real-world challenge for machine learning models. This imbalance requires advanced evaluation metrics (e.g., precision, recall, F1-score) to ensure reliable model performance.

3) *Critical Application:* Classifying asteroids as hazardous or non-hazardous has high stakes for planetary defence and risk assessment. This adds a layer of importance and urgency to developing accurate and reliable predictive models so as not to miss any false positives.

The second dataset, consisting of 1,143 instances and 11 features, relates to the Portuguese 'Vinho Verde' wine. It focuses on classifying wine quality based on physicochemical properties such as acidity and pH levels, with quality scores ranging from 3 (poor) to 8 (excellent) [4].

Unlike the NASA source, this dataset contains no redundant data features; therefore, minimal pre-processing is required. Figure 2 summarizes the corresponding feature and target distributions. The wine classification dataset provides additional interesting challenges in the context of model comparison:

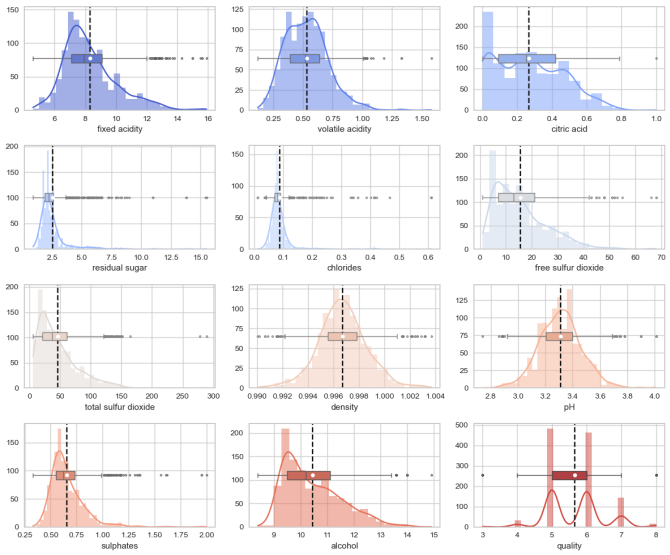


Fig. 2: Distributions of dataset 2: Wine

1) *Multiple Ordered Classes:* The dataset consists of 1,143 ordered quality scores, indicating a progression from poor to excellent wine quality with six discrete classes. Thus, appropriate methods should be selected to handle such multi-class data.
2) *Varying Class Imbalance:* The imbalanced nature of the quality scores (ranging between 0.66% and 42.32%) additionally requires specialized evaluating techniques such as using evaluation metrics beyond accuracy to ensure reliable model performance.

Given the significant imbalance in both datasets, relying only on accuracy can be misleading as it fails to account for the difference between false positives and false negatives. The F1-score, the harmonic mean of precision and recall [7], provides a more balanced evaluation by considering false positives and false negatives. In the case of the NASA dataset, where false positives carry large consequences, recall becomes crucial. Recall measures the model's ability to identify all positive instances, ensuring that critical positive cases are not missed. Therefore, the F1-score will be the primary performance metric. However, for the final test evaluation, all metrics (with a particular emphasis on recall for the NASA dataset) will be considered to ensure a comprehensive assessment that balances the importance of correctly identifying positive instances and minimizing false positives.

While additional data pre-processing steps like outlier removal, correlation analysis and feature engineering/selection exist, these ultimately fall outside the scope of the analysis but should nevertheless be considered in most machine learning pipelines to achieve the best predictive performance.

III. Methodology

In this study, we aim to investigate the performance of three different algorithms: k-Nearest Neighbors (kNN), Support Vector Classifier (SVC), and Multilayer Perceptron (MLP). Each algorithm is evaluated using consistent and standardized methodologies to ensure robust comparisons. Technical details regarding the algorithms and their detailed formulations can be found in [7], [6].

A. Training Strategy:

A hybrid training strategy combining hold-out and cross-validation methodologies ensures a reliable evaluation of each algorithm's performance. Figure 3 presents the approach schematic.

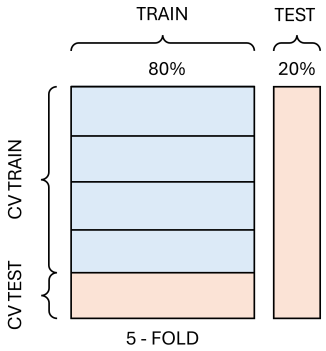
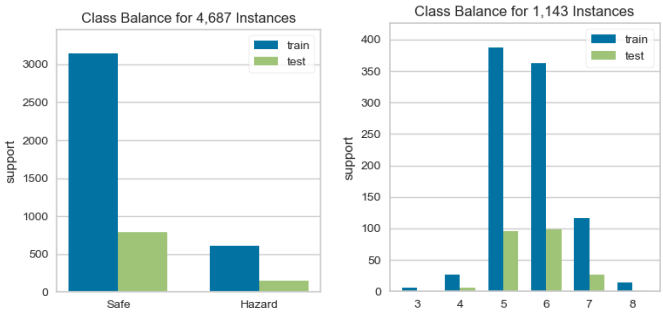


Fig. 3: Train and test strategy schematic

First, the data is partitioned into training and hold-out test sets using an 80/20 split, with 80% of the data reserved for training and the remaining 20% for final evaluation. Next, a 5-fold cross-validation approach is applied within the training set to optimize hyperparameters for the best bias-variance trade-off. Each fold maintains an 80/20 split, ensuring consistency with the overall training strategy. Given the extreme imbalance characteristics of both datasets, a stratified sampling approach is employed throughout the training and cross-validation procedures to preserve class distribution within the testing sets. The dataset target distributions for the train and test splits can be seen in Figure 4.

Both kNN and MLP algorithms inherently support multi-class classification. While kNN operates discretely



(a) Dataset 1: Nasa (b) Dataset 2: Wine

Fig. 4: Class distribution between train (blue) and test (green) datasets

using a simple voting scheme, the MLPClassifier supports multi-class classification by applying the Softmax function (probability normalization) for the output layers [1]. In contrast, the SVC employs a one-vs-multi-class strategy, where either individual classifiers are trained for each pair of classes (one-vs-one), or classifiers are constructed for each class against all others (one-vs-rest) [1]. This approach aligns with the internal principles of support vector machines, which aim to maximize separation between two classes. However, these strategies can significantly slow down the training process for multi-classification problems.

B. Pipeline Evaluation:

Once the data is appropriately partitioned, a pipeline is established for consistent evaluations [1]. Data standardization is applied as a pre-processing step within the pipeline to ensure that feature parameters are scaled to approximately the same size. This allows stable model training as large-magnitude features cannot bias and dominate the learning processes. Following data transformation, model hyper-parameter tuning is conducted using a grid search approach with five-fold cross-validation. Parameter configurations vary depending on the algorithm, as summarized in Table I. It should be noted that other parameters could be tuned. However, not all configurations and parameters could be optimized due to the limited time and computing resources.

Nevertheless, to gain intuition on the intricacies and characteristics of each model algorithm, first, the model's behaviour under varying learning and validation conditions is investigated.

IV. Learning Curves

Learning curves are valuable for evaluating machine learning algorithms, showing error rates based on training size to illuminate the bias-variance trade-off. This trade-off balances bias (errors from overly simple models) and variance (errors from overly complex models sensitive to data fluctuations). Initially, training error is low with minimal data, but validation error is high due to poor generalization. Indicated in [5], as training size increases, training error rises, and validation error falls, indicating improved generalization. A narrow gap between training and validation error in simple models suggests high bias and low variance, signalling the need for more complexity or features. In contrast, complex models like neural networks might exhibit low bias but high variance, shown by a large error gap. Solutions include adding more data, increasing regularization, or simplifying the model.

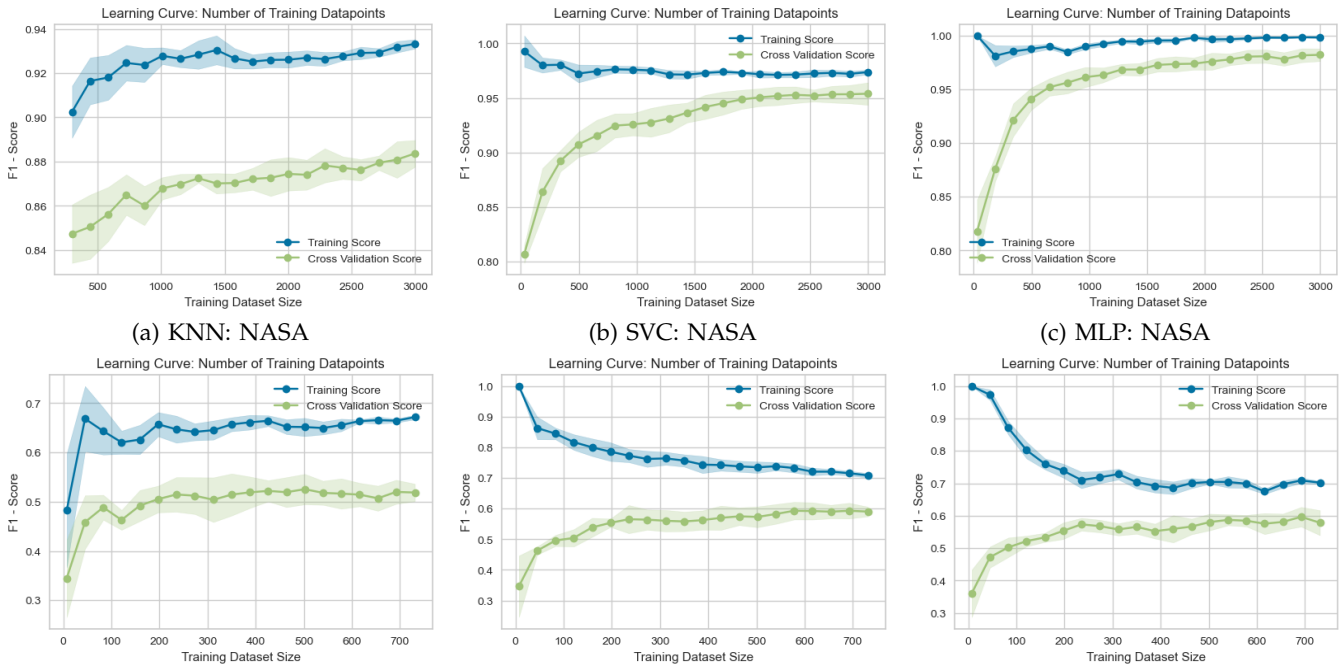
TABLE I: Summary of grid search hyper-parameters

Hyperparameter	Value
kNN	
Number of Neighbors	{1,2,...,100}
Heuristic	{manhatten, Euclidean}
Weights	{Uniform, Distance}
SVC	
Regularization (C)	{0.1, 1.0, 10, 100}
Kernel	{Linear, Polynomial, RBF, Sigmoid}
Polynomial Degree	{1,2,3}
MLP	
Layer Size	{1,2,3}
Number Nodes	{5,10,15,20,25}
Epochs	{50, 100}
Activation Functions	{Tanh, Relu, Sigmoid}
Optimizer	{ADAM, SGD}

The learning curves for the three algorithms, kNN, SVC, and MLP, on the NASA and Wine datasets, are analyzed in Figure 5. Each curve shows the F1-score performance metric for training and validation datasets as a function of training dataset size. As the models are not yet tuned for optimal hyper-parameters, *Sklearn* defaults hyper-parameters are considered for the learning curve analysis to gain intuition on general data impacts.

A. Dataset 1: NASA

1) *k-Nearest Neighbors (kNN):* The training F1-score remains consistently high, progressively increasing as more data is added. The validation F1-score starts lower but continually improves, indicating initial overfitting that improves with more data. Large variance is noticed and is likely the consequence of the curves of dimensionality due to many data features. Based on the current training and validation data trajectories, more data points would help improve overall performance, thus reducing the model variance and increasing bias leading to improved performance.
2) *Support Vector Classifier (SVC):* The training F1-score remains high, while the validation F1-score increases and stabilizes around 0.95, showing good generalization with more data. However, adding more data will likely not significantly improve the performance as we approach the irreducible error region for the validation line, thus causing an overall increase in modelling bias.
3) *Multi-Layer Perceptron (MLP):* The training F1-score is very high, meaning the model can nearly provide a perfect fit with limited data perfectly. The vali-



(a) KNN: NASA (b) SVC: NASA (c) MLP: NASA

(d) KNN: Wine (e) SVC: Wine (f) MLP: Wine

Fig. 5: Learning curves for each dataset and modelling algorithm

dation curve begins low but quickly increases and stabilizes around 2000 data points. There appears to be a good balance between variance and bias, and more data may improve the model slightly.

B. Dataset 2: Wine

1) *k-Nearest Neighbors (kNN):* The training F1-score quickly stabilizes around 0.65. Additionally, the validation shares a similar trend, indicating that the model exhibits quite a high variance. Thus, more data may help increase the performance issues.
2) *Support Vector Classifier (SVC):* The training F1-score decreases with more data, indicating increasing difficulty in fitting the training data. The validation F1-score improves gradually but stabilizes around 0.60. The relatively large gap suggests the potential of high model variance. Adding more training points would likely reduce the overall modelling variance but is unlikely to significantly improve the overall performance as the model approaches a slow convergence with more data points.
3) *Multi-Layer Perceptron (MLP):* The training F1-score decreases with more data, similar to SVC on the Wine dataset. The validation F1-score improves initially and also stabilizes around 0.60, indicating a high variance problem. Again, a plateau has been reached, thus indicating that more data will likely not significantly improve the prediction performance but could help reduce risk of overfitting.

In summary, inspecting learning curves helps diagnose high bias or variance, guiding model optimization through appropriate adjustments in complexity and training strategies. Based on the two datasets, it is clear that the multi-classification problem exhibits intrinsic difficulties for all models.

V. Validation Curves

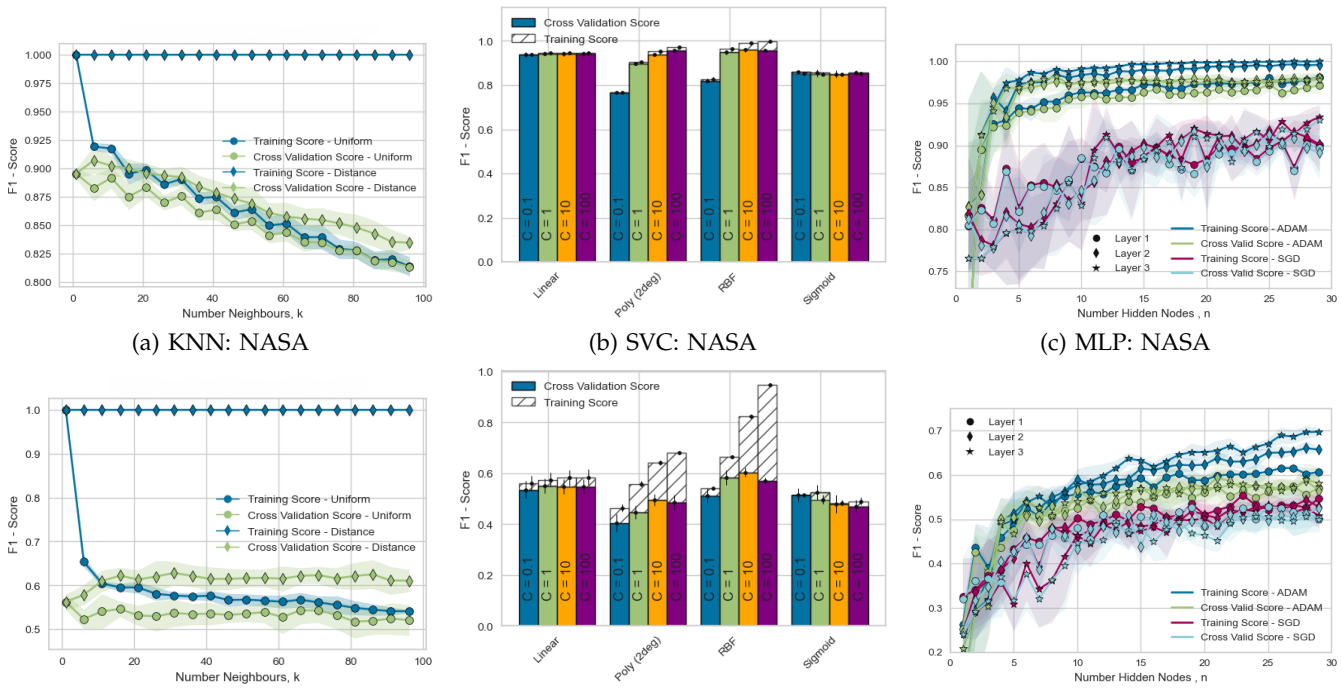
Learning curves provide a macroscopic view of how training data size affects model performance. In contrast, validation curves offer a more microscopic view as specific hyper-parameters influence the training process. Both are important for comprehensive model understanding, evaluation, and optimization in machine learning.

The validation curves for the three algorithms, kNN, SVC, and MLP, on the NASA and Wine datasets are analyzed in Figure 6. Again, each curve shows the F1-score performance metric for training and validation datasets. However, in this case, using the individual ranges defined in Table I, the model hyper-parameters are varied to investigate the impact on performance.

A. k-Nearest Neighbors (kNN)

Important hyper-parameters for the kNN are the number of neighbours (k) and the distance weighting description (Weight). These describe how many neighbouring points to consider and if the "distance" should be considered in evaluating point influence on the overall prediction.

- For both datasets, lower k values result in low bias but high variance (overfitting), with the training score significantly higher than the cross-validation score. As k increases, this results in high bias but lower variance (underfitting), with training and cross-validation scores being lower and closer.



(a) KNN: NASA (b) SVC: NASA (c) MLP: NASA

(d) KNN: Wine (e) SVC: Wine (f) MLP: Wine

Fig. 6: Validation curves for each dataset and modelling algorithm

- The distance weighting outperforms the uniform scheme in terms of cross-validation score, indicating that it is a better metric. However, after using the distance metric, the model fully fits the training set of data. This is the result of using too few data points for too many data attributes. The model overfits both datasets because there is excessive sparsity in the higher-dimensional space.

B. Support Vector Classifier (SVC)

For the SVC algorithm, two important hyper-parameters are the kernel type (kernel) and the regularization parameter (C). These parameters transform the data into a higher-dimensional space where it may become more easily separable. In comparison, the regularization parameter controls the complexity of the decision boundary.

- For both datasets, high C values lead to very high training scores but relatively lower cross-validation scores, indicating overfitting. This is more pronounced in the Wine dataset, especially with the polynomial and RBF kernels.
- The RBF kernel generally performs well across different C values in terms of cross-validation scores, indicating its robustness and good generalization ability.
- The polynomial kernel (2 degrees) can achieve high training scores but tends to overfit, particularly at higher C values, as seen in the Wine dataset.

C. Multi-Layer Perceptron (MLP)

The number of hidden layers and nodes typically governs the MLP. These parameters control the model's ability to capture complex functions. Furthermore, a look at the internal optimizer strategy, Stochastic Gradient Descent (SGD) or Adaptive Momentum (ADAM) algorithms are explored.

- For both datasets, increasing the number of hidden nodes leads to higher training scores but does not always translate into higher cross-validation scores, indicating a strong risk of overfitting.
- The cross-validation performance tends to stabilize after a certain number of hidden nodes (around 10-20 for NASA and 20 for Wine), suggesting that there is an optimal number of hidden nodes.
- Adding more layers (2 or 3) can slightly improve cross-validation performance, but the improvement is marginal and should be weighed against the increased computational complexity and risk of overfitting.
- ADAM optimizer routinely demonstrated improvements over the SGD, indicating the additional momentum improves the ability to locate more optimal and stable weights.

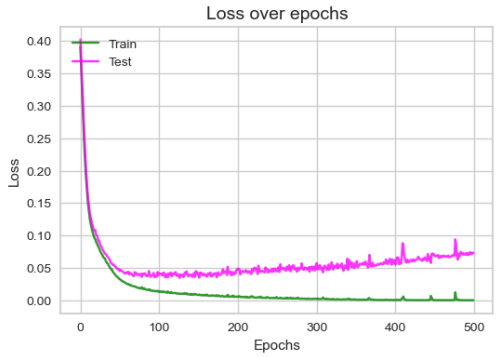
While not explicitly shown, an investigation into the Relu and Tanh activation functions was also conducted. Generally, both seem to provide very similar degrees of accuracy and computational time. However, as indicated in [7], for more complex networks, Relu can be more efficient (simpler formulation) and avoid the challenges of accumulating extremely small gradients (vanishing

gradient problem). These effects were not explicitly observed but are well-known challenges.

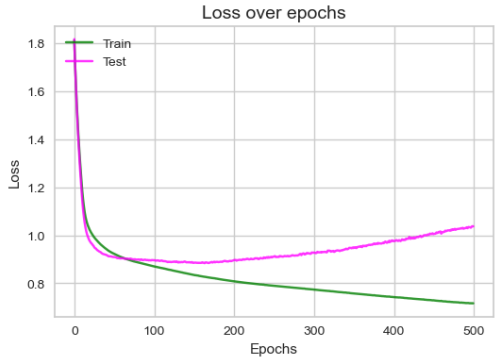
VI. Neural Networks Loss

Epoch Loss Curves are vital for monitoring neural network training. They track loss against training epochs (iterations), revealing overfitting when validation loss rises while training loss falls and reverse for underfitting. These curves identify when overfitting starts, guiding decisions like selecting the optimal amount of epochs using early stopping. They also aid in adjusting the learning rate to ensure optimal convergence.

The neural network is initially trained with a learning rate of 0.001, Nesterov momentum set to True with a value of 0.9, and cross-entropy loss for classification tasks. These settings optimize the process, with the moderate learning rate promoting steady convergence and Nesterov momentum accelerating training by considering future gradient direction. A larger momentum value helps smooth optimization and prevents oscillations [7]. Results are visualized in Figure 7 for training and internal batch validation sets.



(a) Dataset 1: NASA



(b) Dataset 2: Wine

Fig. 7: MLP epoch loss curves for each dataset

A. Dataset 1: NASA

The epoch loss curve for the NASA dataset shows that both training and test losses drop sharply during the initial epochs, indicating that the model quickly learns from the data. At around 75 epochs, the test loss increases while the training loss decreases and stabilizes at a low value. This divergence indicates the onset of overfitting, where the model performs well on the training data but poorly on the test data due to excessive fitting to the training data's noise.

B. Dataset 2: Wine

The epoch loss curve for the Wine dataset highlights different characteristics. As before, both training and test losses initially decrease sharply, showing effective learning in the early stages. After approximately 100 epochs, the test loss rises while the training loss continues to fall, similar to the NASA dataset. This pattern indicates that the model starts overfitting at this point. The test loss starts at a higher value and decreases more gradually than the NASA dataset, suggesting that the Wine dataset is more complex or noisier, making it harder for the model to generalize well.

Based on the outcomes, both datasets benefit from early stopping to prevent overfitting. For NASA, early stopping around 50 epochs is ideal, while for Wine, it is around 100 epochs. The Wine dataset appears to be more challenging for the model, as indicated by the higher initial test loss and the slower reduction in loss compared to the NASA dataset. This suggests that the model may need additional tuning, such as regularization or hyper-parameter adjustments, to perform better on the Wine dataset. The learning rate can also be modified; however, based on the general smoothness of the descent, the current value is deemed adequate. Nevertheless, this parameter is an additional hyper-parameter, which can significantly impact the overall performance.

VII. Test Results and Discussion

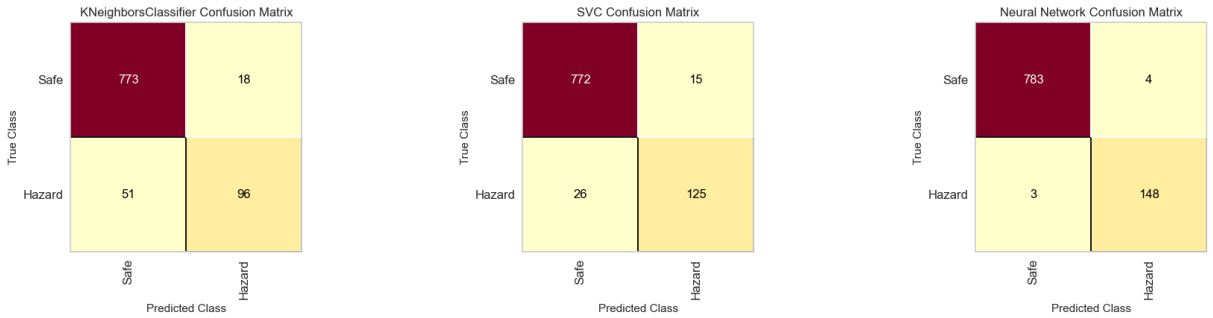
The optimal model configurations are determined via the grid search approach defined in Table I. The optimal hyper-parameter results for each evaluated model and corresponding datasets are summarized in Table II. Many outcomes determined from the learning, validation, and epoch curves are confirmed to be the optimally obtained models.

TABLE II: Grid search hyper-parameter optimal results using 5-Fold cross-validation

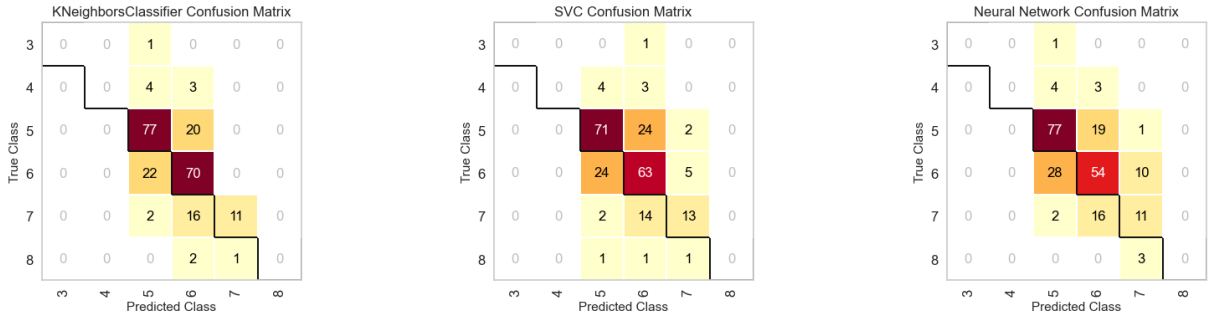
	NASA
kNN	{Neighbours: 17, Heuristic: Manhatten, Weights: Distance}
SVC	{Regularization: 10, Kernel: RBF, Poly degree: 1}
MLP	{Node (Layer): [5, 25, 10] (3), Acti: Relu, Epoch:100, Opti:ADAM}
	Wine - Imbalance
kNN	{Neighbours: 50, Heuristic: Manhatten, Weights: Distance}
SVC	{Regularization: 10, Kernel: RBF, Poly degree: 1}
MLP	{Node (Layer): [5, 20, 20] (3), Acti: Tanh, Epoch:100, Opti:ADAM}

A. Model Performance Comparison

The final evaluation of the kNN, SVC, and MLP models on the hold-out test dataset provides further insights into their performance across the NASA and Wine datasets. The key metrics considered include accuracy, precision,



(a) kNN: NASA (b) SVC: NASA (c) MLP: NASA



(d) kNN: Wine (e) SVC: Wine (f) MLP: Wine

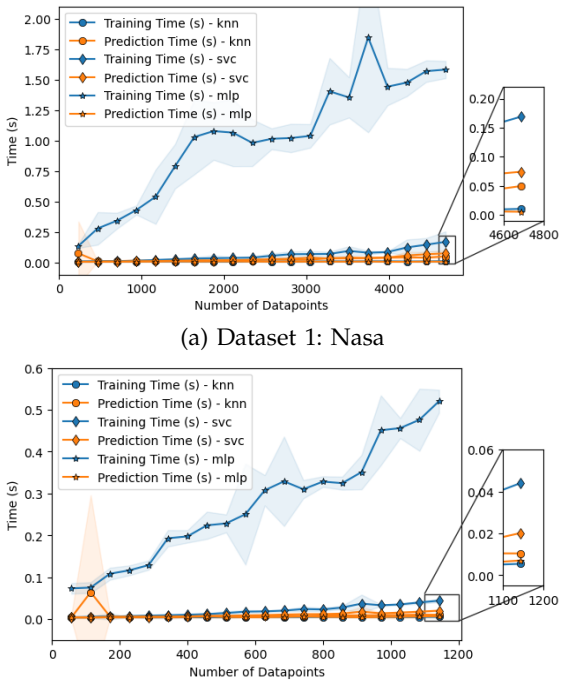
Fig. 8: Confusion matrices for each model and dataset

recall, and F1-score. Table III shows a summary of the corresponding prediction results, which can be visually confirmed in the confusion visualization shown in Figure 8. The multi-class predictions consider a weighted average, which is an appropriate metric when the dataset is imbalanced [7].

TABLE III: Final test set evaluated results

NASA			
Metrics	kNN	SVC	MLP
Accuracy	0.93	0.96	0.99
Precision	0.89	0.89	0.97
Recall	0.68	0.83	0.98
F1-score	0.71	0.86	0.98
Wine - Imbalance			
Metrics	kNN	SVC	MLP
Accuracy	0.69	0.64	0.65
Precision	0.68	0.61	0.61
Recall	0.69	0.64	0.65
F1-score	0.67	0.62	0.63

The MLP classifier's exceptional performance, outperforming both kNN and SVC across all metrics, is a significant finding. With the highest accuracy (0.99), precision (0.97), recall (0.98), and F1-score (0.98) on the NASA dataset, the MLP model demonstrates its high effectiveness in capturing underlying patterns with great accuracy and generalization. The SVC also performs well, with an accuracy of 0.96 and an F1-score of 0.86, showing strong performance but slightly lower than MLP. The kNN shows good performance with an accuracy of 0.93, but its lower recall (0.68) and F1-score (0.71) suggest sensitivity to the choice of neighbours in regards



(a) Dataset 1: Nasa

(b) Dataset 2: Wine

Fig. 9: Training and prediction time comparison

to the impacts of too many data features. The SVC model demonstrates balanced performance with precision and recall both at 0.89 and 0.83, respectively, reflected in its F1-score of 0.86. The MLP's near-perfect scores across all metrics confirm its superiority, likely due to its ability to model complex non-linear relationships.

The performance across all models was notably lower for the Wine dataset, highlighting the challenges introduced by class imbalance and dataset complexity. All three models showed similar performance levels with marginal differences, indicating none significantly outperformed the others in handling the imbalance effectively. This is evidenced in confusion plots, in which no model can adequately predict the boundary classes. kNN achieved an accuracy of 0.69 and an F1-score of 0.67, slightly better in balancing precision (0.68) and recall (0.69) than SVC and MLP. SVC had the lowest accuracy (0.64) and F1-score (0.62), with lower precision (0.61) and recall (0.64), suggesting it struggled more with the imbalanced data. MLP showed modest performance with accuracy (0.65), precision (0.61), recall (0.65), and F1-score (0.63). Although it balanced the metrics slightly better than SVC, the simpler kNN model outperformed the MLP classifier. Several potential reasons for this include:

- *Local Learning:* kNN, as a local learner, makes decisions based on nearby data points, which can be advantageous for datasets with locally smooth decision boundaries [6]. MLP, aiming to learn global patterns, might be less effective if the decision boundaries are complex or the data is limited.
- *Hyper-parameter Sensitivity:* MLP's performance highly depends on hyper-parameters like the number of layers, neurons, optimizers, learning rate, and regularization strength. Without optimal tuning, MLP might underperform. Due to computational constraints, only a few hyper-parameters were explored, leaving learning rate and regularization strength largely unexplored.

B. Model Time Complexity Comparison

In addition to prediction performance, time performance is a crucial metric when comparing the algorithms. The training and prediction times, summarized for both datasets in Figure 9, provide valuable insights into the efficiency of the models.

Based on the outcomes, the following generalities can be drawn: The kNN is fast in training (it only stores data) but slower during prediction due to its inherent "lazy" nature; this drawback could be computed, especially with large datasets. The SVC is computationally expensive both in training and prediction, particularly with large datasets and non-linear kernels. The MLP has extremely high computational costs due to multiple interconnected layers being optimized and requiring significant training time; however, its prediction time is extremely fast through parallelism.

VIII. Conclusions

The evaluation of kNN, SVC, and MLP models on the NASA and Wine datasets highlights significant differences in performance and computational efficiency. A general summary and hypothesis confirmation (✓) or rejection (×) is indicated in Table IV.

TABLE IV: Model result and hypothesis summary

Models	Pros	Cons
kNN	✓ Application of weighting distance improved performance. × Effective for complex datasets where local representation is simpler than global representation (Ⱨ: best performing on Wine). ✓ Essentially zero training time required	✓ Performance degrades with high dimensionality as all features are required for prediction. × Suffers from expensive prediction times (Ⱨ: better than SVC). • Very sensitive to the choice of k and distance metrics.
SVC	× Effective in high-dimensional spaces due to non-linear kernel mappings. (Ⱨ: worse than MLP on binary class). • Robust to overfitting with appropriate regularization.	✓ Computationally expensive training for larger datasets. × Inherently binary, making multi-class strategies inefficient. (Ⱨ: Training time much better than MLP)
MLP	× Capable of capturing complex patterns through deep architecture. (Ⱨ: Lower prediction performance on multi-class than KNN) ✓ Fast prediction time.	✓ Computationally intensive training. • Prone to overfitting without regularization and hyper-parameter tuning • Requires large amounts of data and tuning.

While the MLP model demonstrated superior performance across all metrics for the NASA dataset, all models struggled with the Wine dataset, indicating challenges related to class imbalance and dataset complexity. The target variable in wine classification is inherently subjective and heavily influenced by human factors and preferences. This subjectivity suggests that there may not be a systematic way to quantify wine quality, thus limiting the ability to capture consistent patterns.

Future research should focus on integrating advanced techniques with qualitative assessments to develop more comprehensive and robust models, acknowledging the complexity and variability intrinsic to wine quality evaluation. Potential improvements could enhance model performance, such as further hyper-parameter tuning and advanced algorithms like boosting and ensemble methods. Additionally, feature engineering and data augmentation, including oversampling and synthetic data generation, offer potential solutions to overcome imbalance-related issues.

IX. Resources

[1] *API Reference.* Scikit-learn. https://scikit-learn.org.
[2] *Machine Learning LaTeX Template*. Nakamura, K. (2023).
[3] *Nasa Asteroid Dataset*. Kaggle. https://www.kaggle.com/datasets/shrutimehta/nasa-asteroids-classification.
[4] *Wine Quality Dataset*. Kaggle. https://www.kaggle.com/datasets/yasserh/wine-quality-dataset.
[5] *Learning Curves for Machine Learning.* DataQuest. https://www.dataquest.io/blog/learning-curves-machine-learning/.
[6] *Machine Learning.* Mitchell, T. M. vol. 1, (1997).
[7] *Hands-on machine learning with Scikit-Learn, Keras and TensorFlow: concepts, tools, and techniques to build intelligent systems.* Geron, A. 2nd ed, (2019)