The University of Nottingham

UNITED KINGDOM · CHINA · MALAYSIA

| **Group Details** | |
|---|---|
| **Student Name** | **Student ID** |
| Aditya Purswani | 20596344 |
| Anasuya Dutta | 20594248 |
| Anurag Phukan | 20520078 |
| Dhruv Bhattacharjee (L) | 20592268 |
| Shashwat Sinha | 20520082 |

| **Coursework Details** | | | |
|---|---|---|---|
| **Module Name:** | **Machine Learning** | | |
| **Module Code:** | **COMP4139** | | |
| **Module Convenor:** | **Dr. Xin Chen** | **Seminar Tutor** (if applicable) | |
| **Coursework Title:** | Comparative Analysis of Wine Quality and Iris Datasets by Using Classification and Regression | | |
| **School/Dept:** | **School of Computer Science** | | |
| **Deadline date:** | **10th November 2023** | **Deadline time:** | **15:00 hrs** |
| **Group work:** (please circle) | [✓ Yes] No | **Group No:** (if known) | **C4_18** |

| **Additional information for certain work when requested by School** | | | |
|---|---|---|---|
| **Word count** | **1068** | **Page count:** | **4** (Excluding cover sheet) |
| **Electronic Receipt or Submission Number:** (eg from Moodle or Turnitin) | | | |

| **In the case of an AGREED extension:** | |
|---|---|
| **Extension date authorised by:** | **-** |
| **Agreed new submission date:** | **-** |

**Declarations:**

- I have read and understood the Academic Misconduct Policy and confirm that this submission complies with the policy
- I certify the word count is accurate
- I certify that any electronic copy I have submitted is identical to this hard copy

| OFFICE USE ONLY | | |
|---|---|---|
| **Mark** | **Penalty** | **Provisional mark** |
| | | |
| **Assessed by:** | | |

**Date/time stamp:**

# Comparative Analysis of Wine Quality and Iris Datasets by Using Classification and Regression

Anurag Phukan[1], Aditya Purswani[1], Anasuya Dutta[1], Dhruv Bhattarcharjee[1], and Shaswat Sinha[1],
[1]School of Computer Science, University of Nottingham

**Iris and wine quality datasets were used for classification and regression respectively for this coursework. The classification's goal is to categorize Iris flowers into three distinct species, while for regression we predict 0 to 10 ratings for wine quality. Four of the most used models for machine learning are experimented here: Support vector machines, decision trees, Multilayer Perceptrons, and linear (logistic) regression. For evaluations, We use the k-fold method in which Classification accuracy and mean squared error (MSE) value are used as the evaluation metric for classification and regression respectively.**

***Index Terms*—Dataset Analysis, Wine Quality Dataset, Machine Learning Models, Data Preprocessing**

## I. Introduction

A classification learning algorithm trains a model to predict the class of a data point based on its features used in spam filtering, image recognition, and medical diagnosis. Classifiers are trained on labeled data, which consists of data points with known class labels that learn to identify and predict patterns and features of classes. Hence, predicting the class of new unlabeled data.

The regression model is the association of input variables and continuous outputs of a dataset in machine learning predicting numerical outcomes using mathematical functions. This includes linear, random forest, Bayesian, and so on. In simpler words, regression is used to understand the data relationships.

## II. Literature Review

**Linear Regression** predicts a continuous value by establishing a linear relationship between the target variable and one or more predictor variables. It works towards finding a linear equation that fits the best, reducing the error that lies in the space between the predicted values and actual ones. This final equation found hence allows the user to form estimations and then predict data points that are new. It is based on the assumption that the relationship between variables is linear.

**Logistic regression** predicts a binary outcome (0 or 1) using the sigmoid function. Through this, we can model relationships between dependent values which are binary with independent ones.

**Support Vector Machine (SVM)** calculates an optimal Hyperplane which helps to maximize the margins existing between each class and to reduce the number of errors in regression methods using support vectors, data points closest to the decision boundary.

Here, Kernel functions are capable of handling linear and non-linear data. SVMs are used for their generalization and robustness. For classification, data is bifurcated into smaller classes. For regression, they predict continuous values. SVMs are mostly used in text classification and image processing.

**Decision trees** are in the form of tree-like structures. The internal nodes and leaf nodes represent features and class labels or numeric values respectively, whereas the branches signify all the decision rules.

By recursively partitioning data based on informative features, they create a hierarchy of decisions. They offer clear, interpretable models for complex decision-making processes. Ensemble methods like Random Forests can enhance decision tree performance.

**Multilayer Perceptron (MLP)** a neural network that involves input, hidden, and output layers that have neurons having activation functions. During the training process, backpropagation adjusts weights and biases to lower the errors in predictions.

This makes them appropriate for relationships of data that are complex. We can apprehend intricate patterns using the non-linear activation functions along with the multiple hidden layers. In this method, the data flows in the direction that initiates from the input layer and goes through different hidden layers present till the final output layer.

## III. Parameter optimization

Parameter optimization/hyperparameter tuning helps us to calculate the most suitable collection of hyperparameters to form and attain the most optimized model for machine learning or deep learning to attain the best accuracy.

Here we used the grid search-based approach that extensively runs the algorithm in iteration for the set of parameters the user wants to tune from, then it runs all possible permutations of the hyperparameters to find the best hyperparameter.

## IV. Evaluation methods

**K-fold** cross-validation helps us in the evaluation process of classification and regression and mitigates over-fitting which helps in providing accurate estimations and predictive performance of new data. It follows the following process:

1) Split k-subsets of the dataset
2) Cross-validation iterations
3) Model Training and Evaluation

**Classification Accuracy** is used for the k-fold validation process. The value is given out between 0 and 1, where 0 no predictions are correct and 1 represents the perfect classification.

Sometimes, this process may not work for imbalanced datasets. We hence use metrics like precision, recall, F1 Score, or AUC or ROC to achieve a wide-range calculation for classification. The formula used is  given as:

$$Accuracy = \frac{Correct\ Classification\ Instances}{Total\ instances} \tag{1}$$

**Mean Squared Error** or **MSE** is an evaluation metric for measuring the average of the values found from finding the value of the square of the difference of the actual target value from the predicted value. This evaluation is adapted to find the model's accuracy by simply calculating how well it fits the given data.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2}$$

Here we find the average of the squared difference of the predicted value from the actual target value. It penalizes larger errors more severely than smaller ones, making it suitable for assessing the goodness of fit for regression models. A lesser value of MSE shows a fit which is better and eventually more accurate value.

## V. Results and Analysis

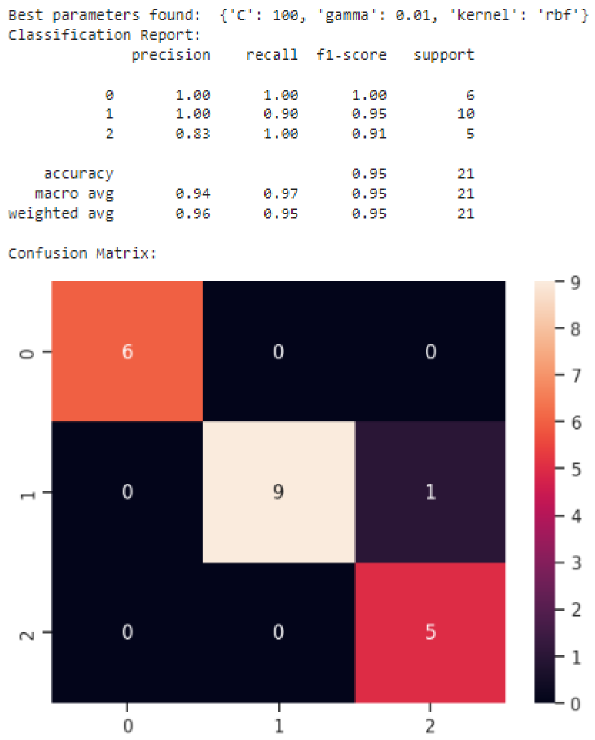
We focused on optimizing the hyperparameters for the above models:

For **SVM model** we tuned C, Gamma, and utilization of the RBF Kernel. The **Decision Tree** model used criterion and Max_Depth (For regression, criterion parameter was not used).

For **MLP** we tuned the values for activation, Alpha, hidden layer configuration, learning rate, and solver.

For **Logistic Regression** we optimized the values for C and Penalty. These selected values significantly improved the overall accuracy and predictive capabilities of the models.

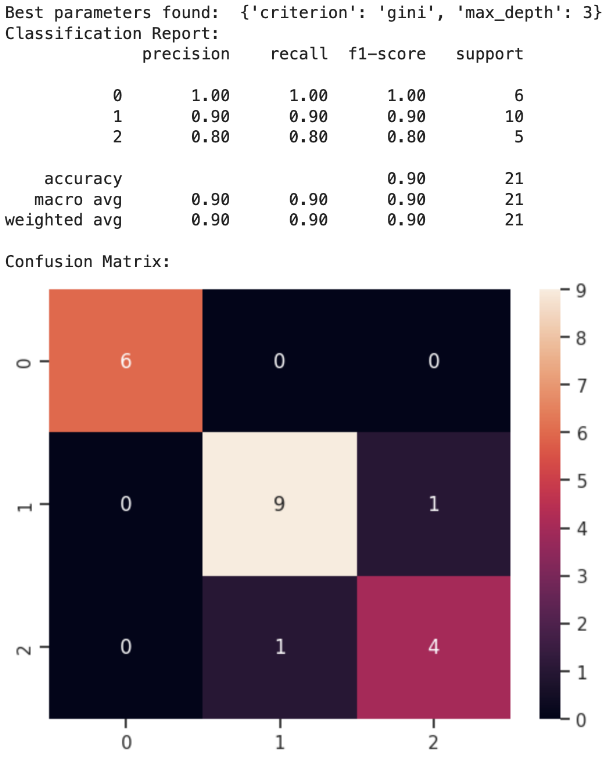
In the pursuit of optimizing the machine learning models, a comprehensive hyperparameter tuning strategy was employed, markedly refining the predictive capabilities and overall accuracy of each algorithm.



```
Best parameters found:  {'C': 100, 'gamma': 0.01, 'kernel': 'rbf'}
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       1.00      0.90      0.95        10
           2       0.83      1.00      0.91         5

    accuracy                           0.95        21
   macro avg       0.94      0.97      0.95        21
weighted avg       0.96      0.95      0.95        21

Confusion Matrix:
```

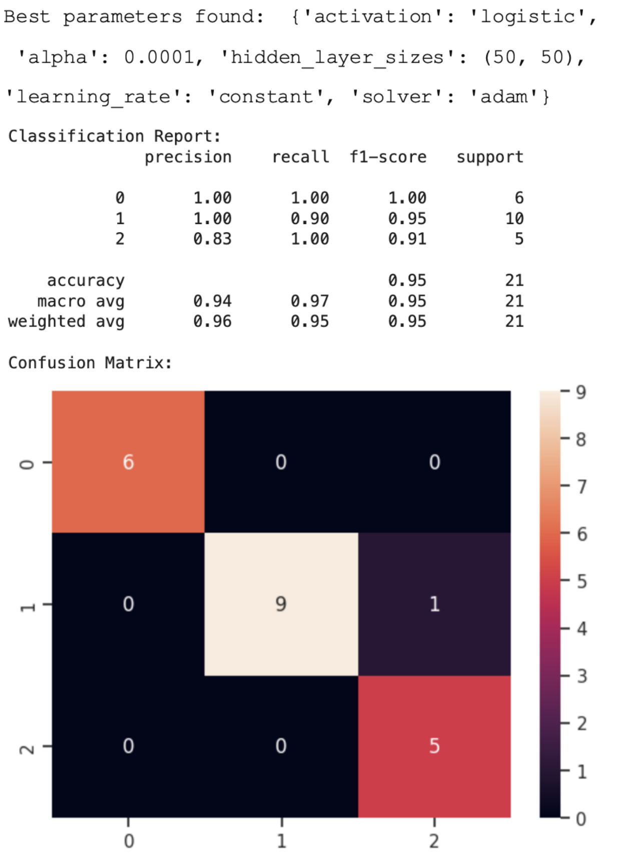
Fig. 1: Best Parameter for SVM Classifier

The judicious customization of these hyperparameters across all models was carried out.



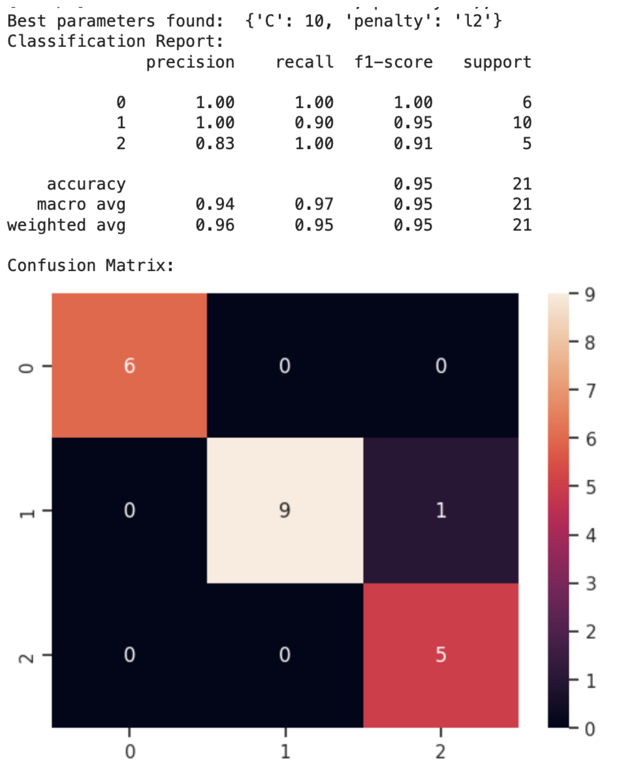
```
Best parameters found:  {'criterion': 'gini', 'max_depth': 3}
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       0.90      0.90      0.90        10
           2       0.80      0.80      0.80         5

    accuracy                           0.90        21
   macro avg       0.90      0.90      0.90        21
weighted avg       0.90      0.90      0.90        21

Confusion Matrix:
```

Fig. 2: Best Parameter for Decision Tree Classifier



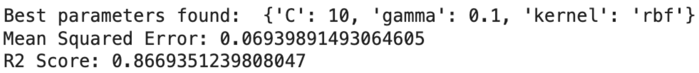
```
Best parameters found:  {'activation': 'logistic',
 'alpha': 0.0001, 'hidden_layer_sizes': (50, 50),
'learning_rate': 'constant', 'solver': 'adam'}
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       1.00      0.90      0.95        10
           2       0.83      1.00      0.91         5

    accuracy                           0.95        21
   macro avg       0.94      0.97      0.95        21
weighted avg       0.96      0.95      0.95        21

Confusion Matrix:
```

Fig. 3: Best Parameter for Multi-layer Perceptron Classifier



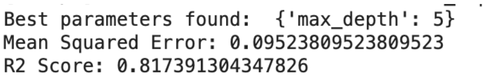
```
Best parameters found:  {'C': 10, 'penalty': 'l2'}
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       1.00      0.90      0.95        10
           2       0.83      1.00      0.91         5

    accuracy                           0.95        21
   macro avg       0.94      0.97      0.95        21
weighted avg       0.96      0.95      0.95        21

Confusion Matrix:
```

Fig. 4: Best Parameter for Logistic Regression



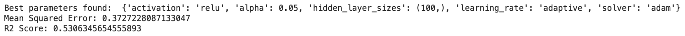
```
Best parameters found:  {'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}
Mean Squared Error: 0.06939891493064605
R2 Score: 0.8669351239808047
```

Fig. 5: Best Parameter for SVM Regressor



```
Best parameters found:  {'max_depth': 5}
Mean Squared Error: 0.09523809523809523
R2 Score: 0.817391304347826
```

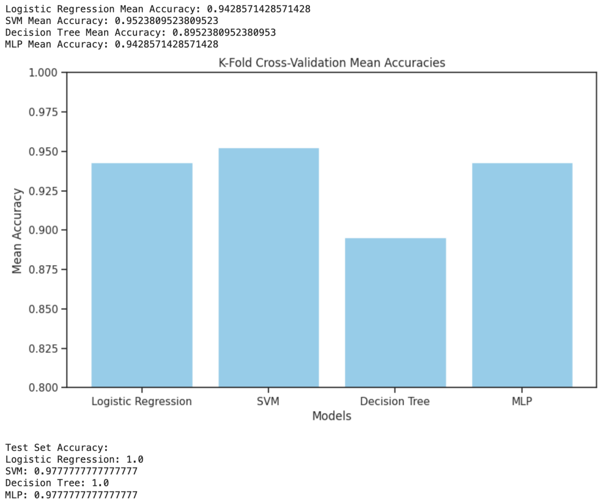
Fig. 6: Best Parameter for Decision Tree Regressor



```
Best parameters found:  {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes': (100,), 'learning_rate': 'adaptive', 'solver': 'adam'}
Mean Squared Error: 0.3727228087133047
R2 Score: 0.5306345654555893
```

Fig. 7: Best Parameter for Multi-layer Perceptron Regressor

**K-fold validation for classification -** Based on the training, the classification and test set accuracy are shown below.

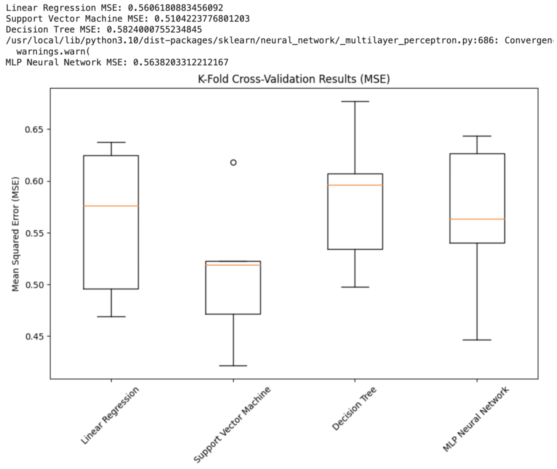


```
Logistic Regression Mean Accuracy: 0.9428571428571428
SVM Mean Accuracy: 0.9523809523809523
Decision Tree Mean Accuracy: 0.8952380952380953
MLP Mean Accuracy: 0.9428571428571428
```

```
Test Set Accuracy:
Logistic Regression: 1.0
SVM: 0.9777777777777777
Decision Tree: 1.0
MLP: 0.9777777777777777
```

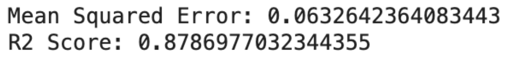
Fig. 8: K-fold for Classification

**K-fold validation for regression -** We can see the Mean Squared Errors for all 4 algorithms. These values can help us in predicting the best model (which has the least MSE value).



```
Linear Regression MSE: 0.5606180883456092
Support Vector Machine MSE: 0.5104223776801203
Decision Tree MSE: 0.5824000755234845
/usr/local/lib/python3.10/dist-packages/sklearn/neural_network/_multilayer_perceptron.py:686: Convergen
  warnings.warn(
MLP Neural Network MSE: 0.5638203312212167
```

Fig. 9: K-fold for Regression



```
Mean Squared Error: 0.0632642364083443
R2 Score: 0.8786977032344355
```

Fig. 10: Linear Regression

## VI. Conclusion

There can be various machine learning models that can yield better results as well. This project demonstrates a comparison of 4 algorithms (SVM, Decision Tree, Multilayer Perceptron, and Linear/Logistic Regression) to create machine learning models based on two datasets Iris and Wine Quality with performance validation for both classification and regression using the K-fold technique and hyperparameter optimization for better results.

## References

[1] Cortes, C., Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

[2] Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C. J. (1984). Classification and regression trees. CRC press.

[3] Dua, D., Graff, C. (2019). UCI Machine Learning Repository. http://archive.ics.uci.edu/ml. Irvine, CA: University of California, School of Information and Computer Science.

[4] Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533-536.

[5] Fisher, R. A. (1936). ”The use of multiple measurements in taxonomic problems.” Annals of Eugenics, 7(2), 179-188.

[6] Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (2009). ”Modeling wine preferences by data mining from physicochemical properties.” Decision Support Systems, 47(4), 547-553.

[7] Montgomery, D. C., Peck, E. A., Vining, G. G. (2012). Introduction to linear regression analysis (5th ed.). Wiley.

[8] Cortes, C., & Vapnik, V. (1995). ”Support-vector networks.” Machine learning, 20(3), 273-297.

[9] Breiman, L. (2001). ”Random forests.” Machine learning, 45(1), 5-32.

[10] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). ”Learning representations by back-propagating errors.” Nature, 323(6088), 533-536.

[11] Hastie, T., Tibshirani, R., & Friedman, J. (2009). ”The elements of statistical learning.” Springer.

[12] Bergstra, J., & Bengio, Y. (2012). ”Random search for hyper-parameter optimization.” Journal of Machine Learning Research, 13(Feb), 281-305.