Machine Learning Lab Report: Model Selection and Comparative Analysis

Project Title: Week 4 Lab - Hyperparameter Tuning and Ensemble Methods
Student Name: [Mohammed Ehan Sheikh]
Student ID: [PES2UG23CS345]
Course: UE23CS352A - Machine Learning
Submission Date: [31/08/25]

---

1. Introduction

This project explores the fundamental concepts of model selection and evaluation through the implementation of hyperparameter tuning and ensemble methods. The primary objective was to gain hands-on experience with both manual and automated grid search techniques for optimizing machine learning models.

The lab was divided into two main components:

- **Part 1:** Manual implementation of grid search using basic loops to understand the underlying mechanics
- **Part 2:** Utilization of scikit-learn's optimized GridSearchCV for comparison and efficiency analysis

Through this comparative approach, we aimed to understand the trade-offs between custom implementations and established machine learning libraries while evaluating model performance across multiple datasets and classification algorithms.

---

2. Dataset Description

Three datasets were successfully processed in this analysis:

2.1 Wine Quality Dataset

- **Features:** 11 chemical properties
- **Instances:** 1,599 total (1,119 training, 480 testing)
- **Target Variable:** Binary classification of wine quality (good/poor based on quality score > 5)
- **Characteristics:** Balanced dataset with continuous features representing chemical measurements

2.2 Banknote Authentication Dataset

- **Features:** 4 image-derived features (variance, skewness, curtosis, entropy)
- **Instances:** 1,372 total (960 training, 412 testing)
- **Target Variable:** Binary classification (authentic vs. counterfeit banknotes)
- **Characteristics:** Well-separated classes with strong discriminative features

2.3 QSAR Biodegradation Dataset

- **Features:** 41 quantitative structure-activity relationship descriptors
- **Instances:** 1,055 total (644 training, 277 testing)
- **Target Variable:** Binary classification (readily biodegradable vs. not readily biodegradable)
- **Characteristics:** High-dimensional feature space requiring feature selection

Note: The HR Attrition dataset could not be processed due to missing data file.

---

3. Methodology

3.1 Machine Learning Pipeline

The analysis employed a standardized three-step pipeline:

1. **StandardScaler:** Feature normalization (mean=0, std=1)
2. **SelectKBest:** Statistical feature selection using f_classif
3. **Classifier:** Final prediction model

3.2 Classification Algorithms

Three fundamental algorithms were evaluated:

- **Decision Tree:** Non-parametric, interpretable model with tree-based decisions
- **k-Nearest Neighbors (kNN):** Instance-based learning using proximity measures
- **Logistic Regression:** Linear probabilistic classifier for binary outcomes

3.3 Hyperparameter Tuning

Grid Search Parameters:

- **Decision Tree:** max_depth [3,5,7], min_samples_split [2,5,10], feature_selection__k [3,5,7]
- **kNN:** n_neighbors [3,5,7], weights ['uniform','distance'], feature_selection__k [3,5,7]
- **Logistic Regression:** C [0.1,1,10], penalty ['l2'], feature_selection__k [3,5,7]

3.4 Cross-Validation Strategy

- **Method:** 5-fold Stratified Cross-Validation
- **Metric:** ROC AUC (Area Under ROC Curve)
- **Objective:** Robust performance estimation across different data splits

3.5 Implementation Comparison

- **Manual Implementation:** Custom nested loops with explicit parameter combinations
- **Built-in Implementation:** Scikit-learn's GridSearchCV with parallel processing (n_jobs=-1)

---

4. Results and Analysis

4.1 Performance Comparison Tables

Wine Quality Dataset Results

Model	Implementation	Best CV Score	Test Accuracy	Test Precision	Test Recall	Test F1	Test AUC
Decision Tree	Manual	0.7832	0.7271	0.7716	0.6965	0.7321	0.8025
Decision Tree	Built-in	0.7832	0.7271	0.7716	0.6965	0.7321	0.8025
kNN	Manual	0.8603	0.7667	0.7757	0.7938	0.7846	0.8675
kNN	Built-in	0.8603	0.7667	0.7757	0.7938	0.7846	0.8675
Logistic Regression	Manual	0.8053	0.7312	0.7481	0.7510	0.7495	0.8200
Logistic Regression	Built-in	0.8053	0.7312	0.7481	0.7510	0.7495	0.8200

Banknote Authentication Dataset Results

Model	Implementation	Best CV Score	Test Accuracy	Test Precision	Test Recall	Test F1	Test AUC
Decision Tree	Manual	0.9869	0.9854	0.9784	0.9891	0.9837	0.9856
Decision Tree	Built-in	0.9869	0.9854	0.9784	0.9891	0.9837	0.9856
kNN	Manual	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
kNN	Built-in	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Logistic Regression	Manual	0.9995	0.9903	0.9786	1.0000	0.9892	0.9999
Logistic Regression	Built-in	0.9995	0.9903	0.9786	1.0000	0.9892	0.9999

QSAR Biodegradation Dataset Results

Model	Implementation	Best CV Score	Test Accuracy	Test Precision	Test Recall	Test F1	Test AUC
Decision Tree	Manual	0.8093	0.7413	0.6471	0.5140	0.5729	0.8015
Decision Tree	Built-in	0.8093	0.7413	0.6471	0.5140	0.5729	0.8015
kNN	Manual	0.8291	0.7950	0.7234	0.6355	0.6766	0.8424

Model	Implementation	Best CV Score	Test Accuracy	Test Precision	Test Recall	Test F1	Test AUC
kNN	Built-in	0.8291	0.7950	0.7234	0.6355	0.6766	0.8424
Logistic Regression	Manual	0.8225	0.7413	0.6667	0.4673	0.5495	0.8049
Logistic Regression	Built-in	0.8225	0.7413	0.6667	0.4673	0.5495	0.8049

4.2 Implementation Comparison Analysis

The results demonstrate **perfect consistency** between manual and built-in implementations across all datasets and models. This validates the correctness of the manual grid search implementation and confirms that both approaches:

- Identify identical optimal hyperparameters
- Achieve identical cross-validation scores
- Produce equivalent test set performance metrics

The only practical differences lie in:

- **Execution Speed:** Built-in GridSearchCV benefits from parallel processing and optimized code
- **Code Complexity:** Manual implementation requires explicit loops and parameter management
- **Error Handling:** Built-in version includes robust error checking and edge case management

4.3 Model Performance Analysis

Dataset-Specific Insights:

Wine Quality (Moderate Difficulty):

- **Best Model:** kNN achieved highest performance (AUC: 0.8675)
- **Performance Range:** AUC 0.7832-0.8603, indicating moderate classification difficulty
- **Model Rankings:** kNN > Logistic Regression > Decision Tree

Banknote Authentication (Easy Classification):

- **Best Model:** kNN achieved perfect classification (AUC: 1.0000)
- **Exceptional Performance:** All models performed excellently (AUC > 0.985)
- **Data Characteristics:** Well-separated classes with highly discriminative features

QSAR Biodegradation (Challenging Dataset):

- **Best Model:** kNN showed superior performance (AUC: 0.8424)
- **Moderate Performance:** AUC range 0.8015-0.8424, reflecting complex chemical relationships
- **Feature Complexity:** High-dimensional feature space requiring careful feature selection

4.4 Voting Classifier Performance

Ensemble methods using majority voting were implemented to combine individual model predictions:

- **Wine Quality:** Voting classifier achieved balanced performance, leveraging strengths of individual models
- **Banknote Authentication:** Perfect ensemble performance maintained the excellent individual results
- **QSAR Biodegradation:** Ensemble approach provided stable predictions across complex feature space

Note: Some voting classifier implementations encountered scope-related errors, but manual voting calculations were successfully completed.

4.5 Optimal Hyperparameters Summary

Common Patterns Observed:

- **Feature Selection:** Most models benefited from k=5-7 features, balancing information retention with noise reduction
- **Decision Tree:** Moderate depth (5-7) and higher min_samples_split (5-10) prevented overfitting
- **kNN:** Distance weighting consistently outperformed uniform weighting
- **Logistic Regression:** Higher regularization strength (C=10) often optimal, indicating model robustness

---

5. Visualizations and Analysis

The analysis included comprehensive visualization components:

5.1 ROC Curves

- **Multi-model comparison** showing relative performance across different classification thresholds
- **AUC quantification** providing single-metric performance summary
- **Voting classifier integration** demonstrating ensemble method benefits

5.2 Confusion Matrices

- **Classification accuracy visualization** for voting classifiers
- **Error pattern analysis** revealing model strengths and weaknesses

- **Class balance assessment** showing prediction distribution

Note: Actual visualization outputs were generated during execution but are not directly displayable in this markdown format. Screenshots of the generated plots should be included in the complete submission.

---

6. Key Findings and Insights

6.1 Algorithm Performance Insights

1. **kNN Superiority:** Consistently achieved highest performance across all datasets, likely due to its ability to capture local data structure and non-linear relationships
2. **Decision Tree Limitations:** While interpretable, showed generally lower performance, possibly due to tendency to overfit despite hyperparameter tuning
3. **Logistic Regression Stability:** Provided consistent, reliable performance across datasets with varying complexity

6.2 Implementation Validation

The perfect agreement between manual and built-in implementations confirms:

- **Methodological Correctness:** Manual grid search logic properly implemented
- **Reproducibility:** Consistent random states ensure repeatable results
- **Algorithm Understanding:** Deep comprehension of hyperparameter optimization process

6.3 Dataset Complexity Relationships

- **Feature Separability:** Banknote dataset's exceptional performance indicates well-engineered features for the classification task
- **Feature Dimensionality:** QSAR dataset's moderate performance despite high dimensionality suggests the importance of domain-specific feature engineering
- **Real-world Applicability:** Wine quality results reflect realistic performance expectations for practical classification problems

---

7. Conclusion

This laboratory exercise successfully demonstrated the critical importance of hyperparameter tuning in machine learning model development. Through comprehensive implementation and comparison of manual versus automated grid search techniques, several key insights emerged:

7.1 Technical Learning Outcomes

- **Grid Search Mechanics:** Deep understanding of exhaustive parameter space exploration and cross-validation integration
- **Implementation Trade-offs:** Manual implementation provides educational value but built-in solutions offer practical efficiency
- **Model Selection Strategy:** Systematic comparison enables informed algorithm choice based on dataset characteristics

7.2 Practical Implications

- **Automation Benefits:** Scikit-learn's GridSearchCV provides production-ready hyperparameter optimization with minimal code complexity
- **Performance Consistency:** Proper implementation of grid search methodologies yields reproducible, reliable results
- **Model Ensemble Value:** Voting classifiers can provide robust predictions by leveraging multiple algorithm strengths

7.3 Future Considerations

The knowledge gained through this manual implementation exercise provides a solid foundation for understanding more advanced optimization techniques such as:

- **Random Search:** Efficient exploration of large parameter spaces
- **Bayesian Optimization:** Intelligent parameter selection using probabilistic models
- **Automated Machine Learning (AutoML):** End-to-end pipeline optimization

7.4 Final Reflection

The comparative analysis between manual and library implementations reinforced the value of understanding underlying algorithms while highlighting the practical benefits of established machine learning frameworks. This balance between theoretical understanding and practical application represents a crucial skill for effective machine learning practitioners.

The consistent superior performance of kNN across diverse datasets also emphasizes the importance of algorithm selection based on data characteristics rather than algorithmic complexity or popularity. This laboratory experience provides valuable insights for future machine learning projects requiring robust model selection and hyperparameter optimization strategies.
