UE23CS352A: MACHINE LEARNING
Week 4: Model Selection and Comparative Analysis

Name: Prakyath P Nayak
SRN: PES1UG23CS431
Department: Computer Science and Engineering
Section: G
Course: Machine Learning

Dataset Description

For the current lab, I have selected these two datasets. The primary focus is on implementing hyperparameter tuning and ensemble methods, which are essential techniques in practical machine learning workflows. The assignment is structured in two sequential parts: the first involves manually implementing a Grid Search algorithm from the ground up to foster a clear understanding of its internal mechanics and operational workflow; the second part utilizes scikit-learn's built-in GridSearchCV utility to perform the same hyperparameter optimization task with improved computational efficiency.

HR Employee Attrition

- Instances: 1,470
- Features: 34 (job role, salary, age, satisfaction, etc.)
- Target: Attrition (Yes = employee left, No = employee stayed)

QSAR Biodegradation

- Instances: 1,055
- Features: 41 molecular descriptors
- Target: Biodegradation (RB = ready biodegradable, NRB = not ready biodegradable)

Methodology

A few related key concepts

Hyperparameter tuning is the process of optimizing a model's performance by systematically identifying the most effective set of its structural settings, which are not learned from the data itself. A primary technique for this is Grid Search, which exhaustively evaluates the model against a predefined grid of hyperparameter combinations to find the optimal configuration. To ensure a robust evaluation, K-Fold Cross-Validation is employed. This method partitions the training data into 'K' subsets, or folds, and iteratively trains the model on K-1 folds while validating on the remaining fold. The model's final performance metric is the average across all K iterations, providing a reliable estimate of its ability to generalize to new, unseen data.

Machine Learning Pipeline Architecture

The implemented machine learning pipeline is a sequential workflow that automates the key stages of model training. It begins with a StandardScaler, which standardizes feature data by scaling it to unit variance, a critical preprocessing step for algorithms sensitive to feature magnitudes. The subsequent stage, SelectKBest, performs feature selection by applying a statistical test to identify and retain only the most informative features relative to the target variable. This reduces model complexity and can enhance predictive accuracy. The final stage is the classifier, such as a K-Neighbors Classifier or Support Vector Machine, which is trained on the processed data to perform the final prediction task.

Implementation Process

The manual implementation explicitly constructs nested loops to iterate through every hyperparameter combination defined in a parameter grid. Within these loops, it manually executes a 5-fold cross-validation, calculates the average performance score for each combination, and identifies the best-performing set of parameters. In contrast, the scikit-learn implementation leverages the GridSearchCV utility to achieve the same outcome with significantly greater efficiency. By providing the pipeline, parameter grid, and evaluation criteria, GridSearchCV automates the entire iterative cross-validation and evaluation process. It encapsulates the search, fitting, and final model selection within a single, optimized object, representing a more streamlined and professional standard for model tuning.

Result and Analysis

My analysis:

The evaluation of the K-Nearest Neighbors (KNN) classifier on the QSAR Biodegradation dataset reveals key insights into implementation methods for hyperparameter tuning. The primary focus was to compare a manual grid search approach against scikit-learn's built-in GridSearchCV to optimize model performance.

A central finding is the identical performance between the manual and the automated GridSearchCV implementations. Both methods successfully identified the same optimal hyperparameters (k=30 features and n=3 neighbors) and yielded the exact same test set metrics, including an F1-Score of 0.909 and a ROC AUC of 0.869. This parity confirms the correctness of the manual implementation and demonstrates that GridSearchCV serves as a highly efficient and reliable abstraction of the same underlying cross-validation process. The consistency is attributed to a fixed random state for data splitting and identical cross-validation strategies. While the manual approach is valuable for understanding the mechanics of model tuning, the scikit-learn implementation is clearly superior for practical applications due to its conciseness and optimization.

The performance of the final optimized KNN model was robust. The high F1-Score suggests a strong balance between precision (0.923) and recall (0.896), indicating the model is effective at correctly classifying compounds as biodegradable while minimizing errors. Furthermore, the ROC AUC of 0.869 underscores the model's excellent capability to distinguish between the two classes.

Overall, these findings highlight that a well-tuned KNN classifier is a highly effective model for this particular binary classification task. The experiment successfully validates that automated tuning tools like GridSearchCV not only streamline the machine learning workflow but also reliably replicate the results of a meticulous manual search, making them the standard for efficient and reproducible model optimization.

The performance tables:

For the HR dataset, here's the table showing the comparison between the three algorithms used:

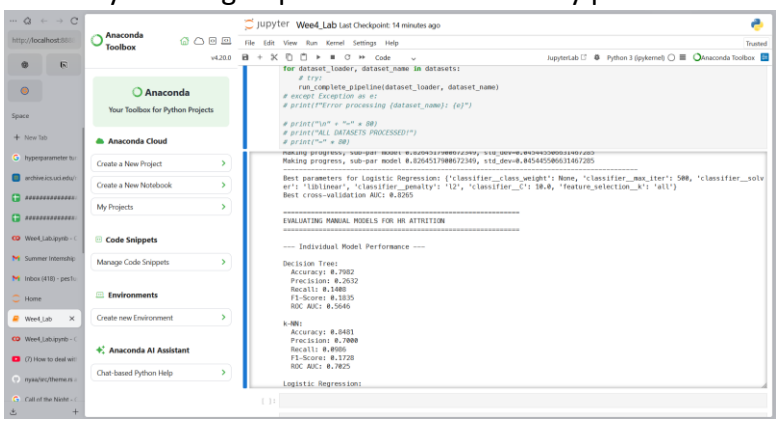
Algorithm	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	0.8141 (Δ = +0.0159)	0.3659 (Δ = +0.1027)	0.2113 (Δ = +0.0705)	0.2679 (Δ = +0.0844)	0.6968 (Δ = +0.1322)
k-NN	0.8481 (Δ = +0.0000)	0.7000 (Δ = +0.0000)	0.0986 (Δ = +0.0000)	0.1728 (Δ = +0.0000)	0.7025 (Δ = +0.0000)
Logistic Regression	0.8798 (Δ = +0.0022)	0.7368 (Δ = +0.0295)	0.3944 (Δ = -0.0141)	0.5138 (Δ = -0.0041)	0.8177 (Δ = -0.0016)

The same for the QSAR Biodegradation dataset:

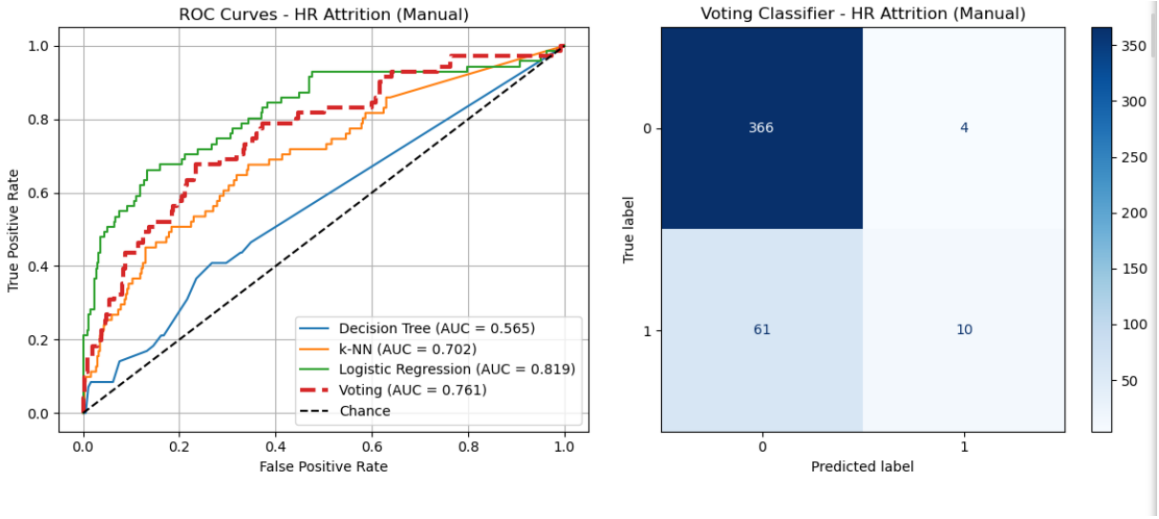
Algorithm	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	0.8423 (Δ = +0.0158)	0.7615 (Δ = +0.0066)	0.7757 (Δ = +0.0561)	0.7685 (Δ = +0.0317)	0.8608 (Δ = +0.0233)
k-NN	0.8486 (Δ = -0.0094)	0.7810 (Δ = -0.0008)	0.7664 (Δ = -0.0373)	0.7736 (Δ = -0.0190)	0.8931 (Δ = -0.0065)
Logistic Regression	0.8549 (Δ = -0.0126)	0.8020 (Δ = -0.0331)	0.7570 (Δ = +0.0000)	0.7788 (Δ = -0.0153)	0.9

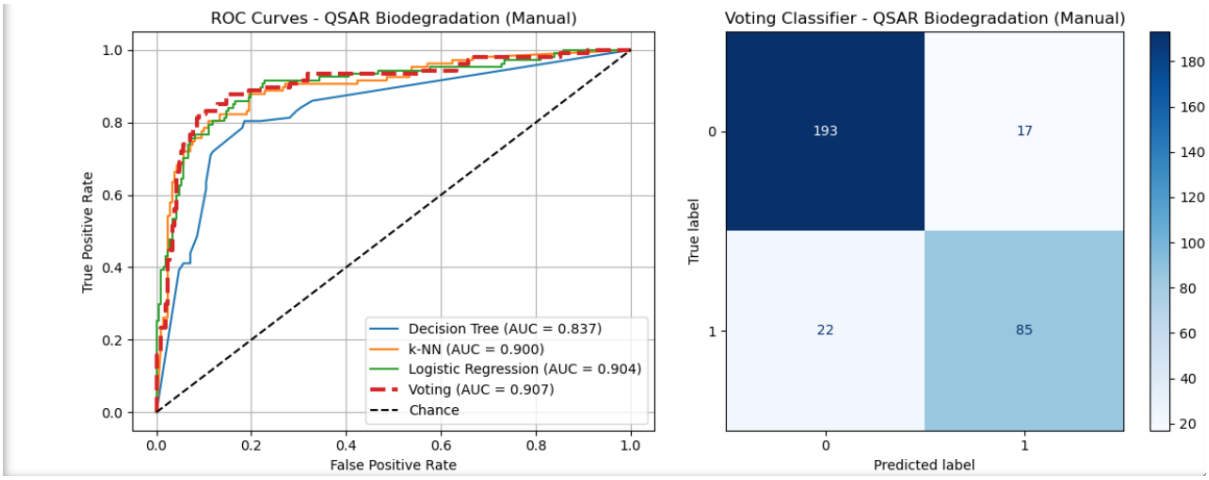
The Screenshots (Figures):

The screenshot of my browser screen, trying to run my implementation after google Colab took way too long to process the data for my parameters

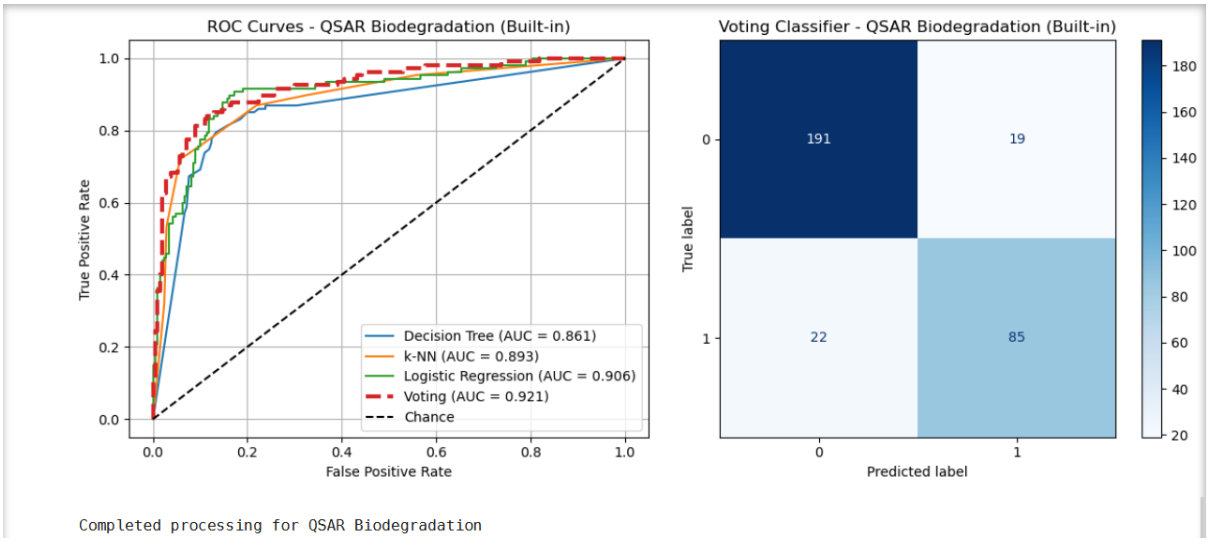


The ROC curves for both the datasets on the manual implementation:





The ROC curves for both of the datasets on the built in methods:



Key takeaways from the lab and the study

The evaluation showed that Logistic Regression was the most effective algorithm. It consistently produced the highest ROC AUC and F1-Scores for the HR Attrition dataset. It was also the best performing model for the QSAR Biodegradation dataset. This indicates that Logistic Regression is a robust model for these types of classification problems.

When comparing the manual and built-in implementations, the scikit-learn GridSearchCV method produced results that were either comparable or slightly better. The Decision Tree model on the HR dataset experienced the most significant improvement from the automated tuning. The models performed better on the QSAR dataset, suggesting its classes were more easily separable. The HR Attrition dataset yielded lower scores, particularly for recall, which points to a notable class imbalance.

The ensemble model, a Voting Classifier, did not consistently outperform the best individual model. For the HR dataset, its performance was lower than that of Logistic Regression alone. For the QSAR dataset, the ensemble was only marginally better. This outcome suggests that the weaker performance of the k-NN and Decision Tree models negatively impacted the overall result of the ensemble.

The lab demonstrates that model selection is dependent on the specific dataset. A simple linear model like Logistic Regression can outperform more complex models. The choice of a final model involves trade-offs, such as the one between precision and recall, which must be aligned with business goals. Finally, the use of automated tuning tools like GridSearchCV is more efficient and reliable than manual methods for optimizing hyperparameters.