Assignment 1

Frank Lee

I. Description of classification problems

A. Heart disease prediction

The heart disease prediction problem involves classifying individuals based on various risk factors to determine whether they are likely to have heart disease. The dataset includes features such as age, sex, presence of chest pain, blood pressure, cholesterol, blood sugar, ECG results, heart rate, presence of angina, ST-depression, slope of ST segment, number of major vessels, and thalassemia. The target variable is binary, indicating the presence or absence of heart disease.

Why This Problem is Interesting:

- Real-World Relevance: Heart disease is one of the leading causes of death worldwide. Early detection and intervention can save lives and reduce healthcare costs. This makes the problem highly relevant and impactful.
- Diverse Feature Set: The dataset contains a mix of numerical and categorical features, which allows for the exploration of different preprocessing techniques and feature engineering strategies. This diversity also provides a rich ground for testing various machine learning algorithms. Numerical features in the set include age, blood pressure, heart rate, and number of major vessels. The rest (chest pain type, fasting blood sugar, ECG results, angina, slope of ST segment, and thalassemia are categorical variables.
- Non-Trivial Nature: While the classification task is binary, the relationships between the risk factors and the presence of heart disease are complex and non-linear. This adds a layer of complexity that makes the problem challenging and interesting to solve. For instance, the impact of age on heart disease might differ significantly based on other factors like sex and smoking status. Algorithms that can model non-linear relationships (like neural networks) can be compared against those that perform well with linear relationships (like SVM with a linear kernel).
- Algorithm Comparison: The problem is suitable for comparing different machine learning algorithms, such as neural networks, support vector machines (SVM), and k-nearest neighbors (KNN). Given such a manageable dataset size, this may be important for KNN, which may become slow as the dataset grows due to instance-based nature, and can be useful for neural networks to limit the computational resources required, and for SVM due to quadratic scaling.

B. Stroke prediction

The stroke prediction problem involves classifying individuals based on various risk factors to determine whether they are likely to experience a stroke. The dataset includes features such as gender, age, hypertension, presence of heart disease, marital status, work type, residence type, blood sugar levels, BMI, and smoking status. The target variable is binary, indicating the occurrence or non-occurrence of a stroke.

Why This Problem is Interesting:

- Critical Health Concern: Stroke is a major health issue globally, leading to significant mortality and long-term disability. Early prediction of stroke risk is crucial for preventive measures and timely medical intervention, making this problem highly relevant and impactful.
- Diverse and Rich Feature Set: The dataset includes a mix of demographic, medical, and lifestyle features. This variety allows for the exploration of how different types of data interact and contribute to stroke risk, offering a comprehensive perspective on the factors influencing stroke.
- Complex Interactions: The relationships between the features and the occurrence of stroke are intricate and non-linear. For example, lifestyle factors such as work type and residence type could interact with medical conditions to influence stroke probability.
- Opportunities for Preprocessing and Feature Engineering: The dataset offers multiple opportunities for preprocessing and feature engineering, which are crucial for enhancing model performance. For example, gender, marital status, work type, and residence type need to be encoded, and numerical features need to be normalized or standardized.
- Real-World Application: The ability to accurately predict stroke risk has significant practical implications. Effective models can be deployed in healthcare settings to identify high-risk individuals and guide preventive strategies, ultimately reducing the burden of stroke on individuals and healthcare systems.

II. Hypothesis

In the context of these two datasets, it is hypothesized that with a small training size, the model will have a low training error but a high cross-validation error, indicating overfitting. As the training size increases, the cross-validation error will decrease.

This hypothesis suggests that initially, with a smaller amount of training data, the model will perform exceptionally well on the training set, leading to a low training error. However, due to the limited data, the model is likely to overfit, resulting in poor generalization to unseen data, which is reflected in a high cross-validation error.

As the training size increases, it is expected that the model will start to generalize better, leading to a decrease in the cross-validation error. The training error may increase slightly as the model becomes less overfit to the training data, but the overall generalization performance will improve, indicated by the convergence of training and cross-validation error rates.

III. Learning curves

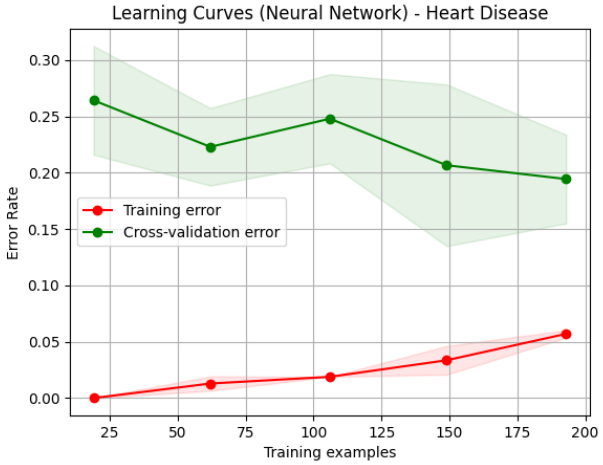
By generating learning curves, we can visualize the difference between training and cross-validation errors at different training sizes, providing insights into the model's ability to generalize as more data becomes available. This analysis will help us understand the impact of training size on the model's performance and validate the hypothesis.

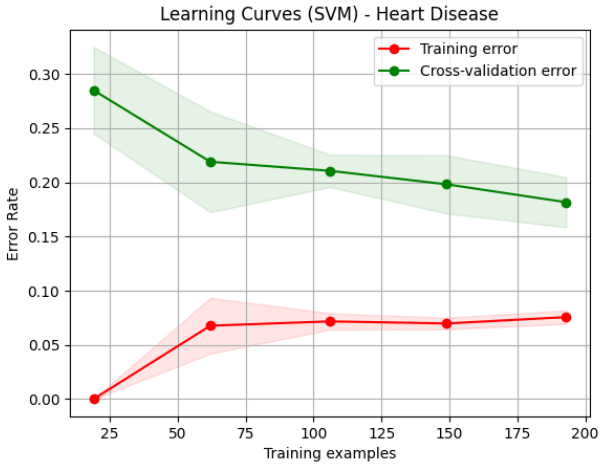
Learning curves based on training size were created for each algorithm (NN, SVM, KNN) for both datasets using scikit-learn with a test size of 0.2, default hyperparameter settings for the MLPClassifier, SVC, and KNeighborsClassifier, and a seed of 42. These curves illustrate how the model's performance evolves as the size of the training data increases. Specifically, they show the training error and cross-validation error at various training sizes, providing insights into the model's learning behavior and its ability to generalize to unseen data.

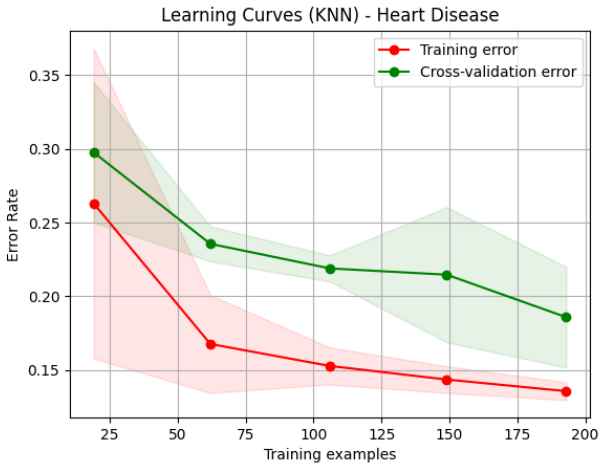
- Neural Network (NN): The learning curves for the NN model indicate how the model's performance changes with increasing amounts of training data. This helps in understanding whether the model is overfitting or underfitting and whether adding more data would benefit the model.
- Support Vector Machine (SVM): The SVM learning curves provide insights into how well the model generalizes as the training size increases. This is useful for assessing the effectiveness of the SVM in handling different amounts of training data.
- K-Nearest Neighbors (KNN): For the KNN model, the learning curves based on training size help in understanding the model's sensitivity to the size of the training data and its performance stability.

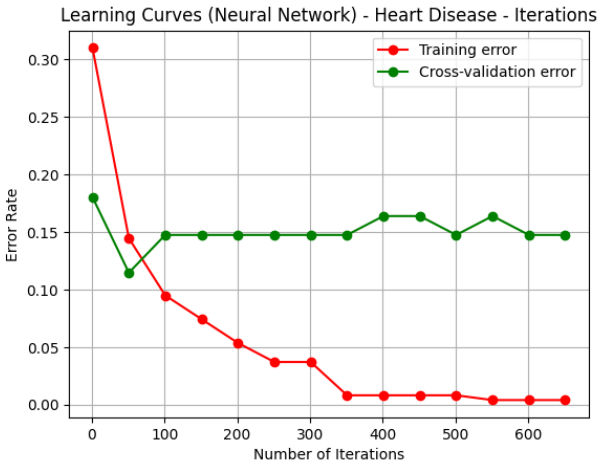
For the iterative algorithms (NN and SVM), additional learning curves based on the number of iterations were generated. These curves show how the training and cross-validation errors change as the number of iterations increases during the training process.

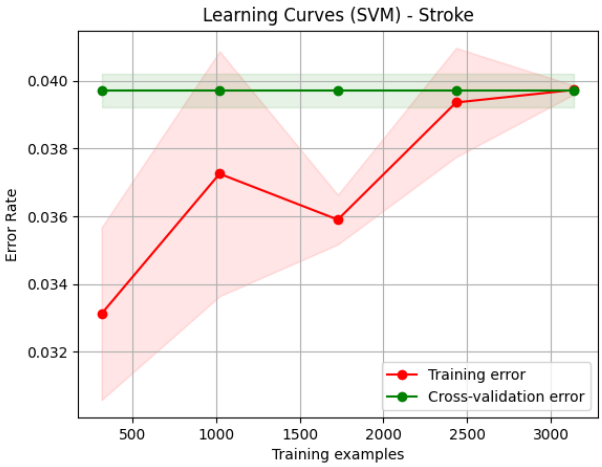
- Neural Network (NN): The iteration-based learning curves for the NN model highlight the convergence behavior of the neural network. By plotting the error rates against the number of iterations, it is possible to observe how quickly the model converges and whether it reaches a stable state.
- Support Vector Machine (SVM): The iteration-based learning curves for the SVM model provide insights into the optimization process and the rate of convergence. This helps in understanding the efficiency of the SVM solver and its ability to find the optimal decision boundary.

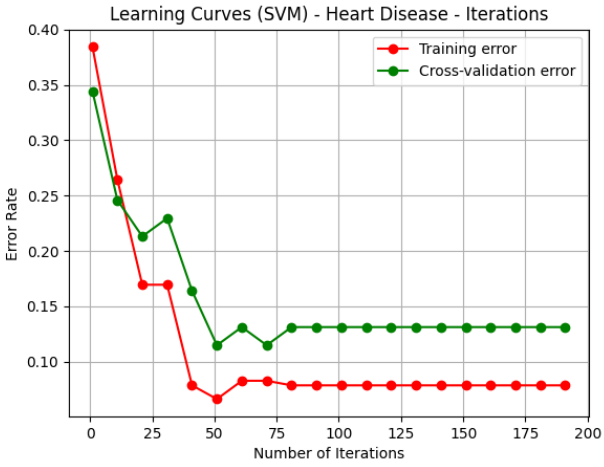


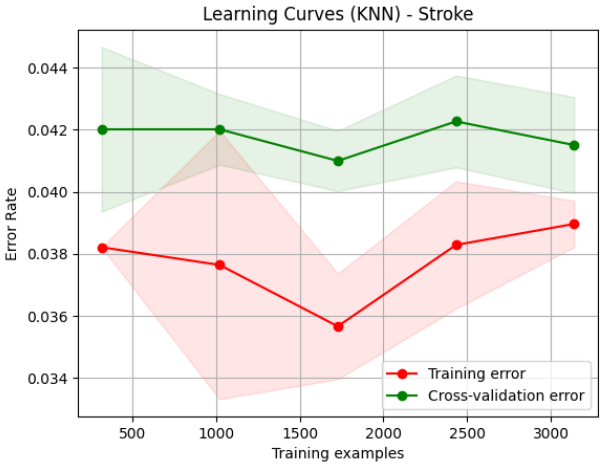


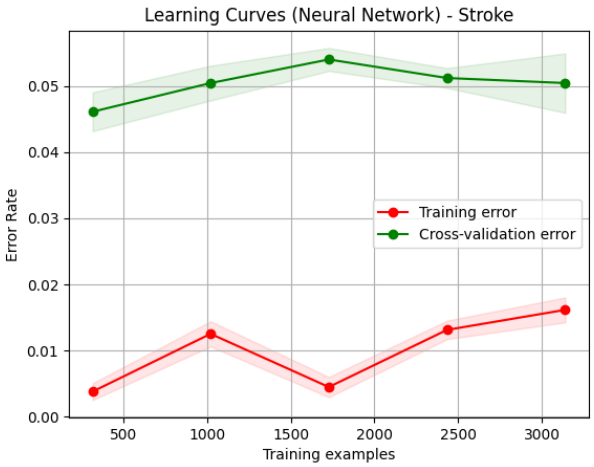


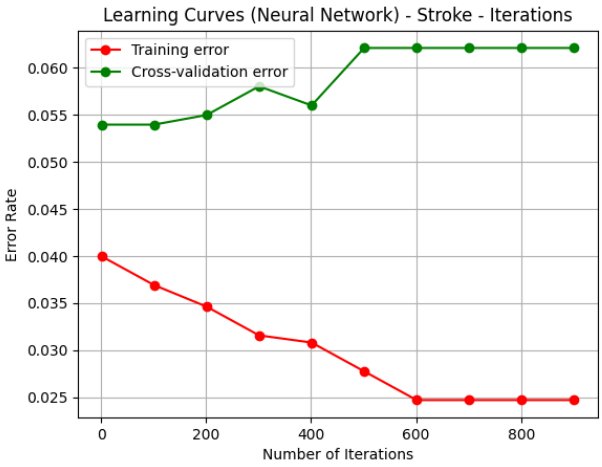

Learning Curves (Neural Network) - Heart Disease - Iterations
Training error
Cross-validation error
Error Rate
Number of Iterations

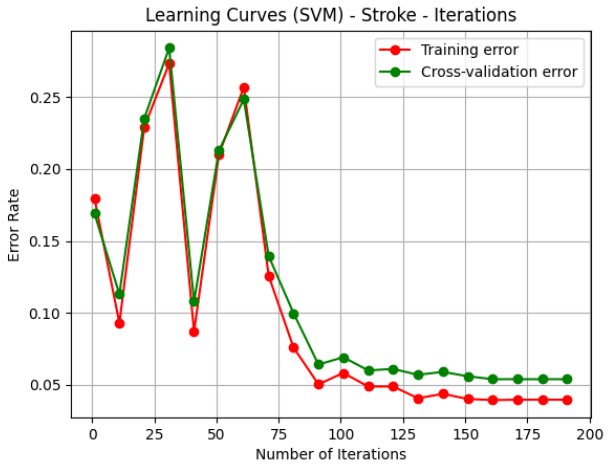

Learning Curves (SVM) - Stroke
Training error
Cross-validation error
Error Rate
Training examples

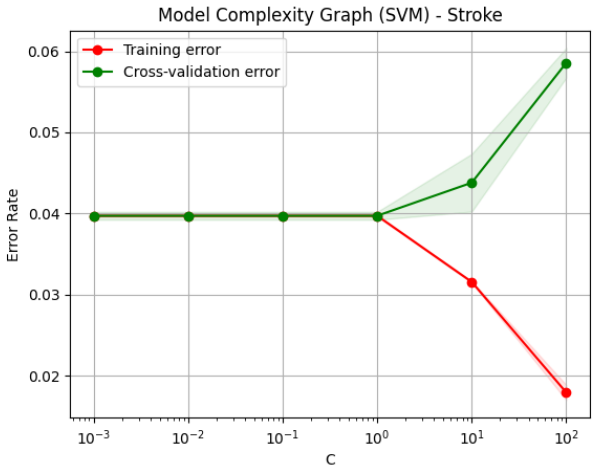

Learning Curves (SVM) - Heart Disease - Iterations
Training error
Cross-validation error
Error Rate
Number of Iterations

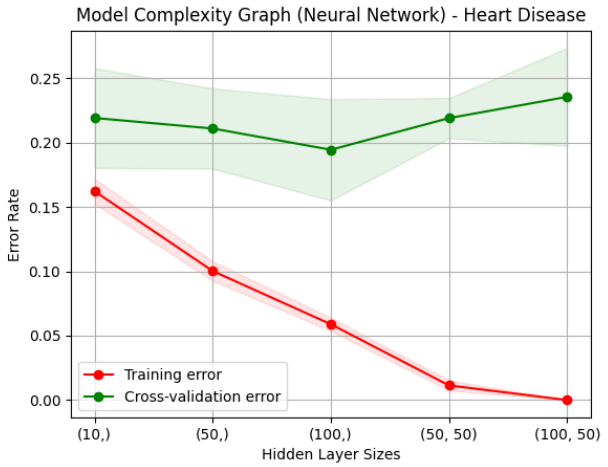

Learning Curves (KNN) - Stroke
Training error
Cross-validation error
Error Rate
Training examples


Learning Curves (Neural Network) - Stroke
Training error
Cross-validation error
Error Rate
Training examples


Learning Curves (Neural Network) - Stroke - Iterations
Training error
Cross-validation error
Error Rate
Number of Iterations







IV. Hyperparameter Tuning

For the heart disease dataset, we focused on tuning the hidden_layer_sizes hyperparameter of the MLPClassifier to analyze its impact on the model's performance. The hidden_layer_sizes parameter determines the number of neurons in each hidden layer and the number of hidden layers in the neural network. The validation curves for the hidden_layer_sizes parameter were plotted, showing the training and cross-validation error rates for each configuration. This provided insights into how the complexity of the neural network affects its performance on the heart disease dataset.

For the stroke dataset, we focused on tuning the C hyperparameter of the SVC to analyze its impact on the model's performance. The C parameter controls the trade-off between achieving a low training error and minimizing the model complexity, effectively determining the strength of regularization. The validation curves for the C parameter were plotted, showing the training and cross-validation error rates for each value. This provided insights into how the regularization strength affects the performance of the SVM on the stroke dataset.

V. Analyses of Results

A. Hypothesis

Based on the learning curves, the hypothesis was confirmed for the Neural Network and SVM for the heart disease dataset, but it did not hold true for the other cases.

For the neural network and SVM in heart disease, the model showed a low training error and high cross-validation error with a small training size, indicating overfitting. As the training size increased, the cross-validation error decreased, confirming the hypothesis. For KNN, both training and cross-validation errors were higher initially and then fell as the training size increased. An explanation might be that KNN relies heavily on the number of neighbors and the distance metric. With a small training size, the model had insufficient data to make accurate predictions, leading to high error rates. As the training size increased, both errors decreased because more data points provided a better basis for nearest neighbor calculations.

For the neural network in the stroke dataset, the training error was very low throughout, and the cross-validation error remained about the same. For SVM, the training error was always lower than the cross-validation error but eventually converged to be the same. For KNN, the training error was consistently lower and did not converge with the cross-validation error.

It is possible that the model complexity for heart disease matched the dataset size better, allowing for improved generalization with more data, while the model complexity for the stroke dataset might have been too high leading to persistent overfitting. This was also likely for the KNN model, where the model struggled to generalize, possibly due to the dataset or suboptimal hyperparameters.

The hypothesis was validated for the Neural Network and SVM models on the heart disease dataset, where increased training size led to reduced cross-validation error, indicating better generalization. For the other cases, different behaviors were observed, indicating persistent overfitting or inadequate model complexity for the given data. These results highlight

the importance of matching model complexity to dataset characteristics and the need for careful hyperparameter tuning and model selection to achieve optimal performance.

B. Hidden layer size

Based on the model complexity graph for the heart disease neural network, training error decreased as we increased the size of the hidden layer, while cross-validation remained about the same. This suggests that the increased model capacity does not translate to better generalization performance. Instead, the model is likely overfitting the training data. To combat overfitting, techniques such as regularization (e.g., dropout, L2 regularization) can be applied to penalize overly complex models. Increasing the amount of training data (data augmentation) can also help improve the generalization performance of the model.

C. C hyperparameter

With a relatively low C value, the SVM model for stroke maintains a balance between margin maximization and classification error minimization. Both training and cross-validation errors are equal and reasonably low, indicating that the model is neither underfitting nor overfitting at this point. As the C value increases, the model prioritizes minimizing the classification error on the training data, leading to a lower training error. However, this comes at the cost of a higher cross-validation error, indicating overfitting. The model fits the training data very closely, but its performance on unseen data deteriorates. The results indicate that a moderate C value strikes a balance between bias and variance, ensuring both training and cross-validation errors are low and similar. For the stroke dataset, a C value around 10^0 seems optimal, minimizing both training and cross-validation errors effectively.

D. Iterations

The analysis of iteration learning curves across different models and datasets reveals distinct patterns. For both heart disease and stroke datasets, the neural network model's training error decreases while cross-validation error remains stable, indicating potential overfitting. Regularization, early stopping, and model simplification are recommended to improve generalization. The SVM iteration learning curve shows that both training and cross-validation errors decrease, indicating effective learning and good generalization. The current model settings appear effective, but continued monitoring is advised to maintain performance.

E. Algorithm comparisons

Given the results, SVM seems to be a good model for heart disease given that it shows a clear pattern of decreasing both training and cross-validation errors, indicating effective generalization and better performance compared to NN and KNN. KNN improves with more data, but its initial high errors suggest it may need more data than currently available to perform optimally. NN overfits the training data, as indicated by the stable cross-validation error despite decreasing training error.

For the stroke dataset, SVM shows the lowest consistent cross-validation error, indicating it generalizes better to unseen data compared to the other models. The KNN model has consistent training and cross-validation errors, indicating good performance and generalization. However, its error rate is slightly higher than that of the SVM. The neural network, despite having a very low training error, shows a higher cross-validation error of 0.05, indicating overfitting and less effective generalization compared to the SVM and KNN. Based on the learning curves and cross-validation errors, the Support Vector Machine (SVM) is the best-performing algorithm for the stroke dataset, demonstrating the best generalization performance. The K-Nearest Neighbors (KNN) also performs well, with consistent and low error rates. The Neural Network (NN), while effective in fitting the training data, overfits and fails to generalize as well as the SVM and KNN models. To improve the neural network's performance, regularization techniques should be considered to mitigate overfitting.

REFERENCES

[1] Scikit-learn. (2024). Scikit-learn: Machine Learning in Python. Retrieved from https://scikit-learn.org/stable/