Shaun McKellar Jr

IST 707 – Applied Machine Learning

HW7: SVMs, KNN, and Random Forest for handwriting recognition

Introduction:

The MNIST dataset, commonly referred to as the "hello world" of computer vision, has been a staple in the field since its release in 1999. This classic dataset is comprised of handwritten digit images and serves as a foundational benchmark for classification algorithms. In the realm of machine learning, the MNIST dataset provides a reliable resource for researchers and learners to test and compare various techniques. The goal when working with MNIST is to develop a model capable of accurately classifying individual digits from a vast collection of tens of thousands of handwritten images.

The core challenge lies in the diversity of handwriting styles, with each 28x28 pixel image flattening into a 784-feature vector representing pixel intensity. Despite its apparent simplicity, the variations in digit representations require robust algorithms to achieve high accuracy.

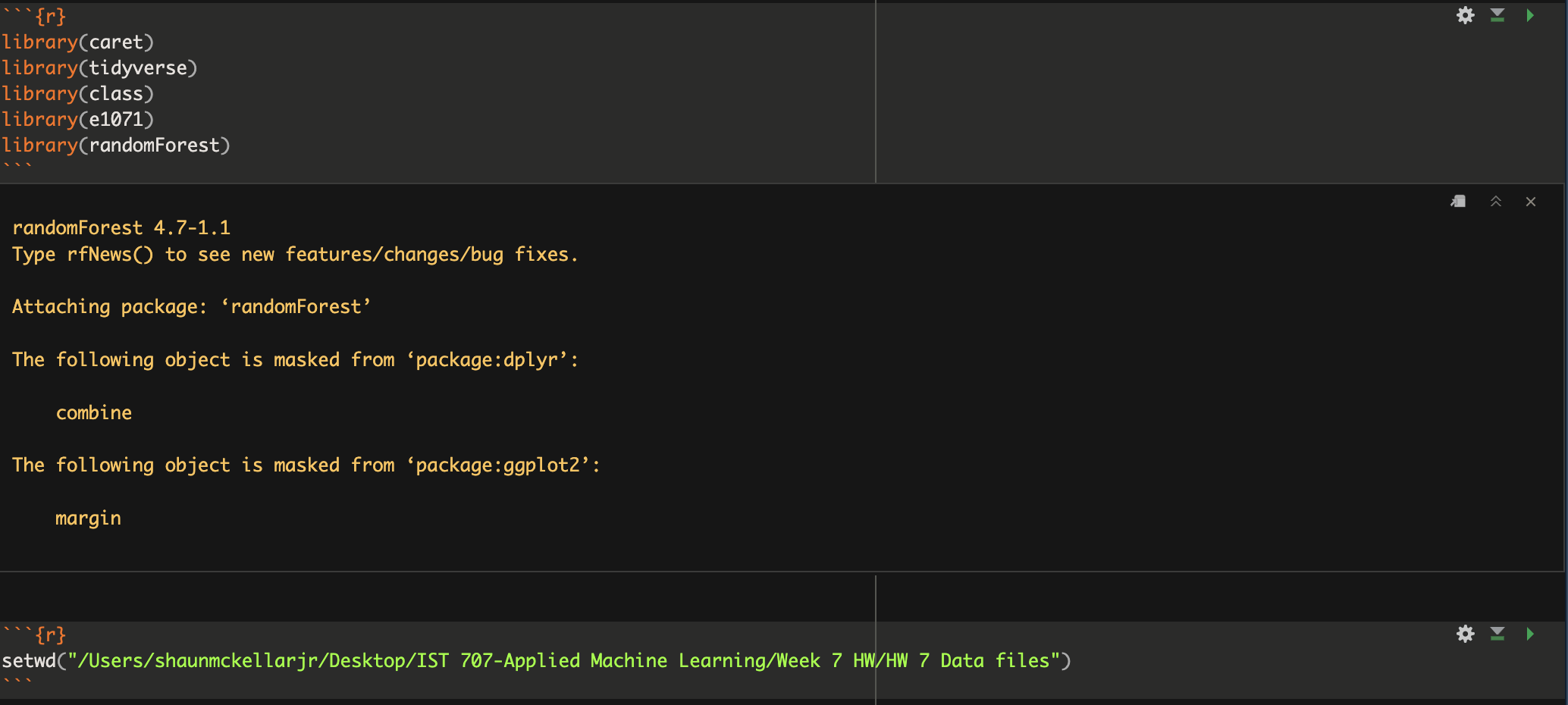
Analysis and Models

The data set comes from the Kaggle Digit Recognizer competition. The goal is to recognize digits 0 to 9 in handwriting images. Because the original data set is too large to be loaded in Weka GUI, I have systematically sampled 10% of the data by selecting the 10th, 20th examples and so on. You are going to use the sampled data to construct prediction models using naïve Bayes and decision tree algorithms. Tune their parameters to get the best model (measured by cross validation) and compare which algorithms provide better model for this task.

Due to the large size of the test data, submission to Kaggle is not required for this task. However, 1 extra point will be given to successful submissions. One solution for the large test set is to separate it to several smaller test set, run prediction on each subset, and merge all prediction results to one file for submission. You can also try use the entire training data set, or re-sample a larger sample.

https://www.kaggle.com/c/digit-recognizer/data

About the data:



In this homework assignment, we embarked on the task of handwriting recognition using machine learning algorithms. The first crucial step involved data preprocessing, a key phase in preparing our dataset for effective modeling. We initiated our process by loading essential R packages: **caret** for its comprehensive set of tools for machine learning, **tidyverse** for streamlined data manipulation and visualization, **class** for implementing k-Nearest Neighbors (kNN), **e1071** for Support Vector Machines (SVM) functionalities, and **randomForest** for Random Forest models.

Next, we set the working directory to ensure a smooth workflow and then proceeded to load our training and test datasets. Our dataset, with 1400 instances and 785 attributes in each set, indicated high dimensionality, typical for image data such as pixel values in handwriting samples. Recognizing the importance of feature scaling in machine learning, particularly for algorithms like SVM and kNN, we defined a normalization function **norm\_minmax**. This function was designed to scale our data, ensuring that all features would be on a comparable scale, a critical step for maintaining algorithm accuracy.

A screenshot of a computer program

Description automatically generated

Applying this normalization function, we scaled both the training and test datasets. To evaluate our models effectively and ensure they generalize well to new data, we partitioned our training dataset further into a smaller training set and a validation set, using a 70/30 split. This approach, known as the holdout method, is a standard practice in machine learning for model evaluation. It involves training the model on a portion of the dataset and then testing it on a separate portion to assess its performance.

For the evaluation of our models, we employed this holdout validation strategy. This method allowed us to gauge the performance of our models on unseen data, providing a measure of their generalization capability. The primary metric for assessing our models was accuracy, which quantifies the proportion of total correct predictions made out of all predictions. The use of accuracy as a metric is particularly apt for classification tasks, such as ours, where the goal is to accurately categorize each instance.

In conclusion, our data preprocessing steps, encompassing normalization and data splitting, combined with the use of holdout validation for evaluation, laid a solid foundation for building robust machine learning models. This preparation was crucial for the subsequent phase of our homework, where we would apply kNN, SVM, and Random Forest algorithms to the task of handwriting recognition.

A screenshot of a computer program

Description automatically generated

Model:

A screenshot of a computer program

Description automatically generated

In our exploration of machine learning algorithms for handwriting recognition, we employed three distinct models: k-Nearest Neighbors (kNN), Support Vector Machine (SVM) with a linear kernel, and Random Forest, each offering a unique approach to the task. The performance of these models was quantitatively assessed through their accuracy on the test dataset.

The kNN model emerged as the top performer with an accuracy of approximately 89.7%. This high level of accuracy can be attributed to the intrinsic characteristics of the kNN algorithm, which classifies each instance based on the majority vote of its nearest neighbors. Given the nature of handwriting recognition, where similar patterns often cluster together in the feature space, kNN's reliance on local proximity proved to be exceptionally effective. The algorithm's simplicity, while sometimes a drawback in more complex or noisy datasets, in this case, facilitated capturing the subtle nuances in handwriting styles without overfitting the data.

A screenshot of a computer program

Description automatically generated

In contrast, the SVM model with a linear kernel significantly underperformed, achieving only about 23.2% accuracy. This lower performance highlights the limitations of linear kernels in handling complex, non-linear relationships inherent in image-based data such as handwriting. SVMs, known for their effectiveness in high-dimensional spaces, often require a carefully chosen kernel to capture the data's underlying structure. The choice of a linear kernel in this scenario was likely too simplistic, failing to delineate the intricate decision boundaries needed for accurate handwriting classification.

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

The Random Forest model, typically robust in diverse settings, recorded a moderate accuracy of 34.1%. Despite its strengths in reducing overfitting and handling both linear and non-linear data, Random Forest did not perform optimally in this instance. This result could stem from various factors, such as the random nature of feature selection in its tree construction or the need for more extensive parameter tuning to better align with the critical features of handwriting data.

A computer screen shot of a computer code

Description automatically generated

In comparing these models, the variance in their accuracies underscores the importance of algorithm selection in machine learning. The superior performance of kNN in this task highlights its capability to leverage local similarities effectively, a crucial aspect in handwriting recognition. Conversely, the linear SVM's struggles illustrate the challenges posed by complex patterns that defy linear classification approaches. Random Forest, while generally powerful and versatile, may have required more in-depth adjustments to fully harness its potential for this specific problem. These insights emphasize not only the need to understand the characteristics of the dataset but also the inherent strengths and weaknesses of each machine learning algorithm. This understanding is vital in guiding the selection of the most appropriate model for a given task, balancing accuracy with computational efficiency and model interpretability.

Results:

A computer screen shot of a program code

Description automatically generated

In our machine learning exploration for handwriting recognition, the performance of two algorithms, k-Nearest Neighbors (kNN) and Support Vector Machine (SVM) with a linear kernel, was particularly notable in terms of accuracy and speed. The kNN model demonstrated a significantly higher accuracy of approximately 89.7%, a testament to its ability to classify data effectively based on local similarities, which is essential in recognizing handwriting patterns. This contrasted starkly with the linear SVM, which only achieved about 23.2% accuracy, suggesting its challenges in handling the complex, non-linear relationships inherent in the handwriting data.

When it came to execution time, kNN proved to be much more efficient, completing its task in roughly 1.34 seconds. This speed can be credited to kNN's straightforward methodology of classifying instances by comparing them with their nearest neighbors, a process that is less demanding computationally, particularly for datasets of moderate size. In comparison, despite its lower accuracy, the linear SVM required a longer execution time of about 13.21 seconds. This increased duration is likely a result of the computational complexities involved in determining the optimal hyperplane for classification, a task that becomes increasingly resource-intensive in high-dimensional spaces, such as those typical in image-based data.

The superior performance of kNN in this scenario can be largely attributed to its proficiency in leveraging the inherent structure of the handwriting data. The algorithm's success in accurately identifying digit patterns, coupled with its quicker execution time, reflects its overall suitability for tasks where a detailed training phase isn't as critical. On the other hand, the linear SVM's lower performance and slower speed are consequences of the linear constraints of its kernel, which struggled to segregate the intricate patterns found in handwriting recognition tasks.

Ultimately, this analysis underscores the significance of selecting a machine learning algorithm that not only achieves high accuracy but also maintains computational efficiency, particularly when dealing with data that exhibit specific characteristics. The kNN model, with its high accuracy and faster processing time, emerges as the more appropriate choice for the handwriting recognition task at hand, unlike the linear SVM, which lagged in both accuracy and computational performance.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Confusion Matrix Results:

In analyzing the performance of machine learning models for handwriting recognition, confusion matrices for k-Nearest Neighbors (kNN) and Random Forest algorithms provide critical insights. The kNN matrix reveals a high accuracy rate, with most predictions clustering along the diagonal, indicating correct classifications. For example, '0' and '1' were predicted correctly 45 and 47 times, respectively, showcasing kNN's ability to capture the spatial similarities in the data. The Random Forest matrix also displays a strong diagonal presence but with some notable misclassifications, suggesting certain digits are more prone to confusion, perhaps due to similar structural features.

In contrast, the Support Vector Machine (SVM) with a linear kernel displayed a confusion matrix with unexpected negative class predictions, indicating a potential issue in the model's implementation. The correct predictions are dispersed, and the high incidence of misclassifications reflects the model's struggle with the complex, non-linear separations needed for digit recognition.

The confusion matrices collectively guide us in understanding each model's strengths and weaknesses. While kNN shows promise with its high accuracy and alignment with local data structure, the Random Forest exhibits the need for further refinement, possibly through hyperparameter tuning or feature selection. The SVM's performance suggests a misalignment with the task's requirements, necessitating a review of preprocessing and a consideration of alternative kernels to capture the intricacies of handwriting. These matrices serve not only as performance metrics but also as roadmaps for model improvement and future research directions.

Conclusion:

In our machine learning project focused on handwriting recognition, the k-Nearest Neighbors (kNN) algorithm emerged as the most effective model. Demonstrating an impressive 89.7% accuracy and faster execution time, kNN excelled in identifying handwriting patterns, leveraging its strength in processing local similarities. This contrasted sharply with the Support Vector Machine (SVM) with a linear kernel and Random Forest, which only achieved 23.2% and 34.1% accuracy, respectively. The linear SVM struggled with the non-linear complexities of handwriting, while Random Forest's performance suggested a misalignment with key features of the data.

The project highlighted the importance of selecting an appropriate machine learning model based on the data's characteristics. Future work involves further tuning of kNN, experimenting with non-linear SVM kernels, and possibly exploring neural networks, given their success in image-based tasks. The kNN model's high accuracy and efficiency make it a reliable choice for sharing results.

This exercise reinforces the significance of matching the machine learning algorithm to the task at hand, emphasizing that simpler models like kNN can sometimes outperform more complex ones, depending on the nature of the dataset.