CS7641 Project 1 Report
Supervised Learning

Juejing Han
jhan446@gatech.edu

Abstract**—**Five supervised learning algorithms are examined with two balanced datasets for binary classification. With a high-quality dataset, the testing accuracy has minimal differences (0.96 - 0.99) among Decision Tree (DT), Gradient Boosting (GB), K-nearest Neighbors (KNN), and Support Vector Machines (SVM), while Neural Networks (NN) faces challenges. SVM and KNN achieve similar high scores, but SVM requires more training and testing time. Conversely, when dealing with a more complex dataset containing noise, there is little variation in testing accuracy (0.72 - 0.79) among five algorithms. GB achieves the highest score while maintaining significantly shorter training and testing time than NN, which has the second-highest score. KNN is sensitive to noisy data, while GB exhibits greater robustness. SVM's training time is sensitive to kernel selection, and NN demands more effort in hyperparameter tuning.

1 EXPERIMENT DESIGN

1.1 Datasets & Pre-processing

Records of diabetes (raw Data1, 8 features) and sleepiness (raw Data2, 10 features) are used. All attributes are numerical and non-negative. Both datasets have a binary target and exhibit class imbalance. The minority-to-majority class ratio is 1:1.6 for raw Data1 and 1:1.3 for raw Data2.

To emphasize characteristics beyond varying degrees of data imbalance, under-sampling techniques are applied to balance the data. Given the unknown feature distribution and the non-negative nature of the datasets, normalization is employed[1]. The pre-processed datasets are referred to as Data1 (Data_Diabetes, 1270 samples) and Data2 (Data_Sleepiness, 3200 samples).

1.2 Characteristics of Datasets & Classification Problems

There is a substantial variability in data nature. The sample ratio between Data1 and Data2 is 1:2.5. Data2 has more features, increased complexity, a larger sample size, and higher noise level.

Binary classification experiments on both datasets explore algorithm performance under diverse data characteristics, particularly variations in complexity and noise. This mirrors real-world data variability and aids in pinpointing suitable algorithms for specific applications.

[1] https://www.simplilearn.com/normalization-vs-standardization-article

1.3 Algorithms & General Settings

Five machine learning algorithms – Decision Tree (DT), Gradient Boosting (GB), K-nearest Neighbors (KNN), Neural Networks (NN), and Support Vector Machines (SVM) – are evaluated on binary classification with Python libraries (scikit-learn, etc.).

Following pre-processing, the data is divided into training (80% of the entire dataset) and testing (20%) subsets; a 5-fold cross validation approach is applied. Since the datasets are balanced, accuracy is introduced as the metric[2].

1.4 Hyperparameters

Grid search is employed to assist in hyperparameter tuning. Table 1 presents the optimal combinations for each model across both datasets.

Table 1—Optimal combinations of hyperparameters.

Model	Data1	Data2
DT	max_depth = 13, min_samples_leaf = 1	max_depth = 8, min_samples_leaf = 17

GB	max_depth = 14, min_samples_leaf = 13
	n_estimators = 10, learning_rate = 1
		max_depth = 11, min_samples_leaf = 12
		n_estimators = 20, learning_rate = 0.1

KNN	n_neighbors = 1, p = 2	n_neighbors = 11, weights = distance, p = 1

NN	hidden_layer_sizes = (20,)
	alpha = 0.001, learning_rate_init = 0.01
		hidden_layer_sizes = (62,)
		alpha = 0.0001, learning_rate_init = 0.1

SVM	kernel = rbf, C = 1, gamma = 1000	kernel = rbf, C = 10, gamma = 10

2 RESULTS & ANALYSIS

2.1 Decision Tree (DT)

2.1.1 *Data1 (Data_Diabetes)*

As model complexity increases, it fits the training data more closely (slightly overfitting) and generalizes effectively (Fig. 1). When max_depth = 13, both training and validation scores peak and stabilize. As a result, with a max_depth = 13, the model is complex enough to capture the data's characteristics.

[2] https://www.statology.org/f1-score-vs-accuracy/

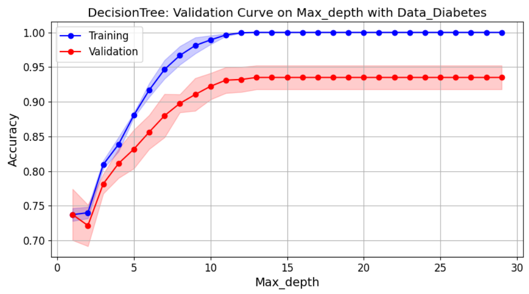

Figure 1—Validation Curve on Max_depth

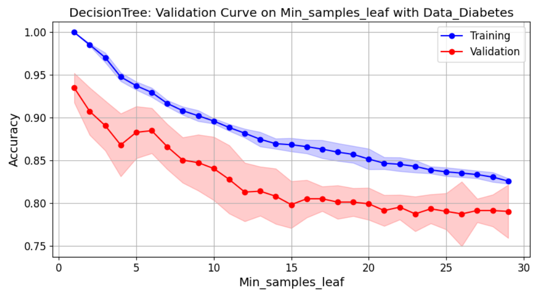

Figure 2—Validation Curve on Min_sample_leaf

Fig. 2 shows that the model achieves the best performance when min_sample_leaf = 1, allowing for fine-grained leaf nodes, which make the model more complex. It is important to note that this configuration has the potential to overfit.

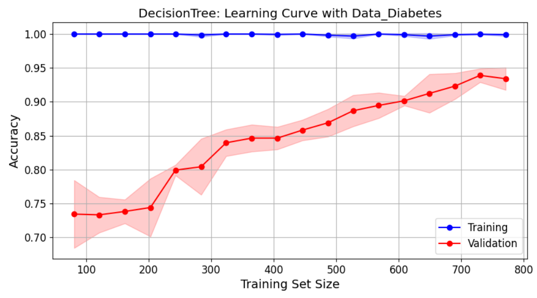

Figure 3—Learning Curve

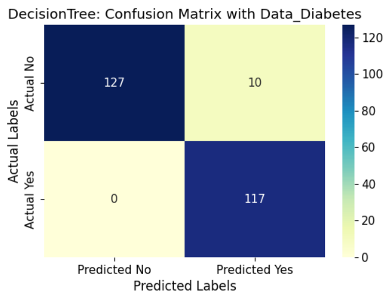

Figure 4—Confusion Matrix

As the sample size increases, the model's generalization performance improves, reducing overfitting (Fig. 3). With around 730 instances, the model achieves a validation accuracy of approximately 0.94. However, additional samples may not lead to further improvement as the validation score tends to decrease afterward. The confusion matrix (Fig. 4) reveals a recall of 1, signifying no false negatives.

2.1.2 *Data2 (Data_Sleepiness)*

Increasing tree depth improves both scores and mitigates underfitting until max_depth reaches 8 (Fig. 5). Increased variance emerges thereafter and stabilizes beyond max_depth=13, indicating no improvement with higher model complexity.

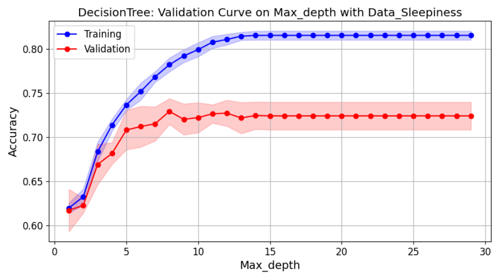

Figure 5—Validation Curve on Max_depth

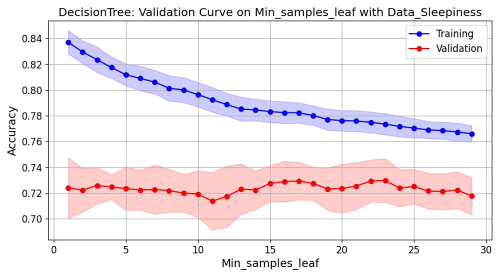

Figure 6—Validation Curve on Min_sample_leaf

The training score decreases as min_sample_leaf increases; reduced model complexity mitigates high variance (Fig. 6). Simultaneously, the validation score fluctuates and peaks when min_sample_leaf = 17, indicating the optimal value.

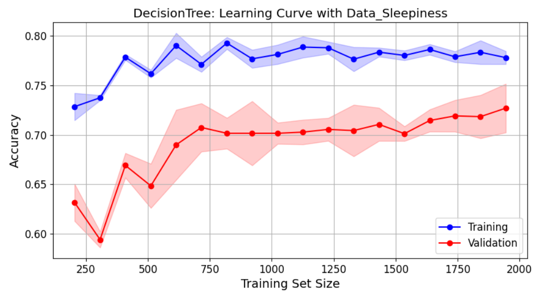

Figure 7—Learning Curve

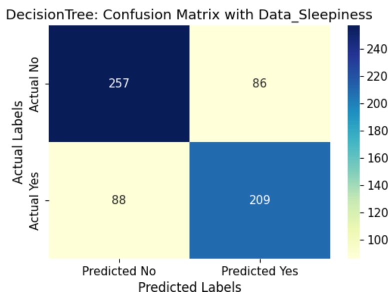

Figure 8—Confusion Matrix

The overall convergence trend suggests that additional samples would enhance model performance (Fig. 7). Model yields comparable numbers of false positives and false negatives (Fig. 8), indicating similar precision and recall (both around 0.7). It implies that DT does not exhibit significant bias towards Type I or II errors with Data2.

2.2 Gradient Boosting (GB)

2.2.1 *Data1 (Data_Diabetes)*

Both scores reach their peaks when n_estimators = 10 and stabilize thereafter. While there is a degree of overfitting, the validation score remains relatively high at 0.95 (Fig. 9). This suggests that an ensemble model with 10 weak learners is sufficient to capture the data’s characteristics and generalize to unseen data. Furthermore, the smaller variance between scores compared to DT indicates that Boosting outperforms the standalone Decision Tree.

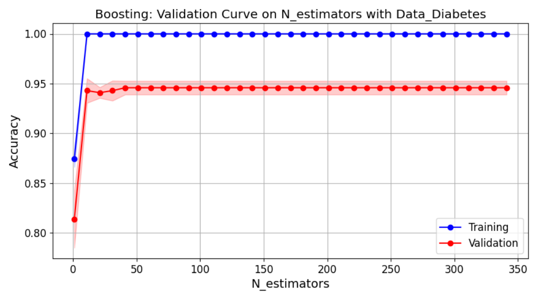

Figure 9—Validation Curve on N_estimators

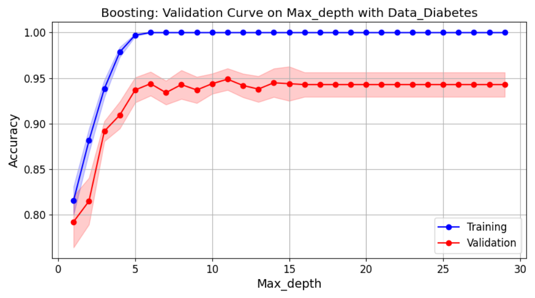

Figure 10—Validation Curve on Max_depth

Fig. 10 suggests that the optimal max_depth falls within the range of 5 to 15 – model is optimal with max_depth = 14 after tuning. As the sample size increases, generalization improves, reaching a validation score of 0.94 with 740 samples. Beyond this threshold, convergence tends to reverse, indicating diminishing returns with additional data (Fig. 11). Boosting reduces false positives and remains zero false negatives (Fig. 12), achieving higher precision compared to DT.

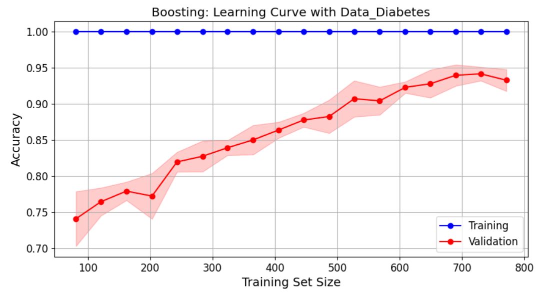

Figure 11—Learning Curve

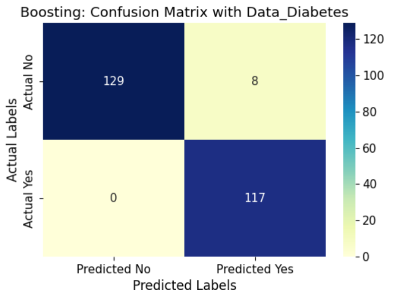

Figure 12—Confusion Matrix

2.2.2 *Data2 (Data_Sleepiniss)*

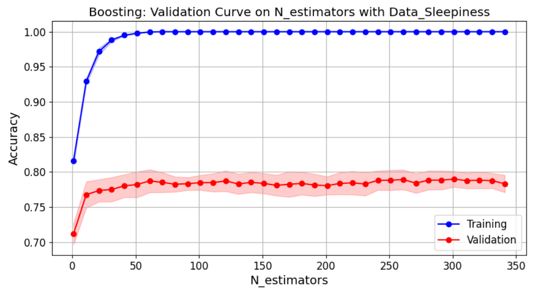

Figure 13—Validation Curve on N_estimators

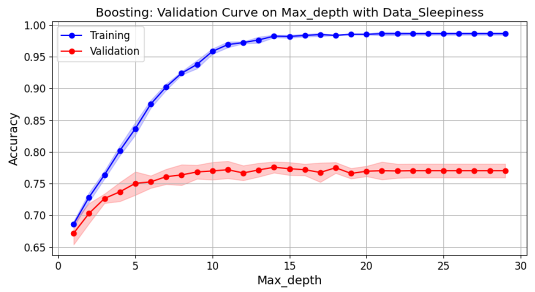

Figure 14—Validation Curve on Max_depth

Adding weak leaners (up to 60, Fig 13) or increasing max_depth (up to 18, Fig. 14) improves both training and validation but also results in increased overfitting. Beyond these thresholds, model performance levels off, suggesting that reducing model complexity would yield better performance. Tuning within these ranges identifies the optimal hyperparameters: n_estimators = 20 and max_depth = 11.

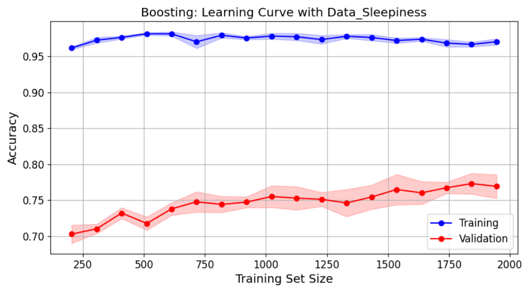

Figure 15—Learning Curve

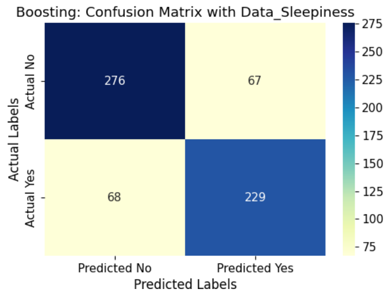

Figure 16—Confusion Matrix

The model achieves a high training score with small sample sizes, but additional instances provide limited improvement (Fig. 15). Larger training samples gradually enhance the validation score, yet overfitting is apparent, possibly due to model complexity, noisy data, inadequate features, etc. Boosting enhances precision and recall compared to DT by reducing false positives and false negatives (Fig. 16).

2.3 K-nearest Neighbors (KNN)

2.3.1 *Data1 (Data_Diabetes)*

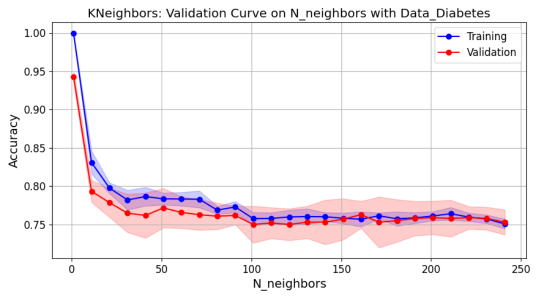

Figure 17—Validation Curve on N_neighbors

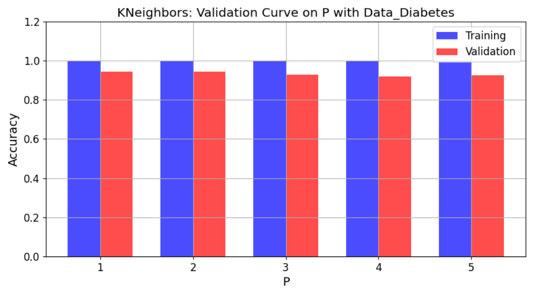

Figure 18—Validation Curve on Power Parameter

The model performs best with n_neighbors = 1 (Fig. 17). Both scores decrease as n_neighbors increases up to 100, beyond which additional neighbors have limited impact. For Data1, the weights parameter (uniform or distance) does not affect prediction since it is based on the closest single neighbor. Power parameters show minimal differences (Fig. 18), and grid search picks p=2 (Euclidean distance) as the optimal choice.

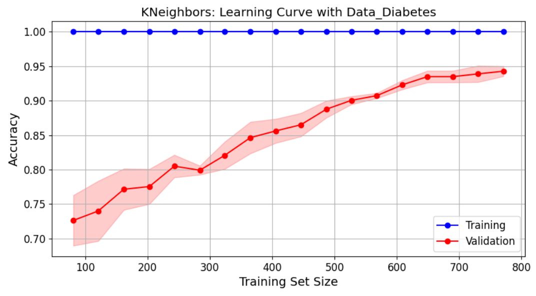

Figure 19—Learning Curve

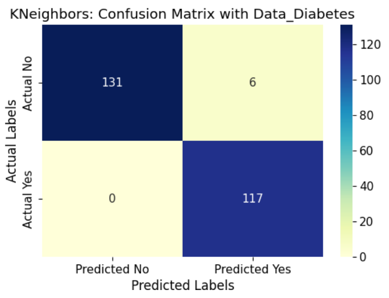

Figure 20—Confusion Matrix

Adding more samples mitigates overfitting and improves generalization. Beyond 750 instances, further increases in the sample size can lead to enhanced model performance (Fig. 19). Like DT and GB, KNN exhibits zero false negatives but fewer false positives (Fig. 20).

2.3.2 *Data2 (Data_Sleepiness)*

Validation score initially rises with increasing n_neighbors (up to 10) and declines (Fig. 21). KNN exhibits more pronounced overfitting compared to GB. The power parameters demonstrate minimal differences, with p=1 (Manhattan distance) slightly outperforming (Fig. 22). As the sample size increases, validation performance gradually improves. Nevertheless, overfitting remains noticeable (Fig. 23). KNN reduces false negatives but increases false positives compared to DT and GB, indicating significant bias towards Type I error (Fig. 24).

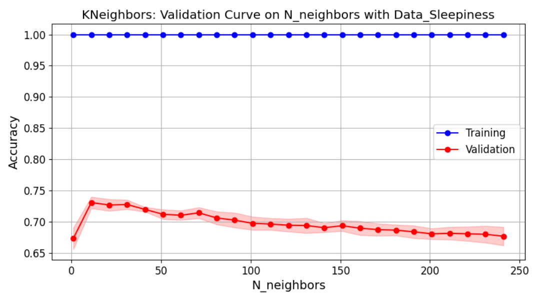

Figure 21—Validation Curve on N_neighbors

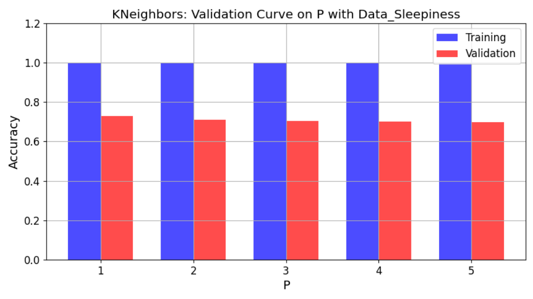

Figure 22—Validation Curve on Power Parameter

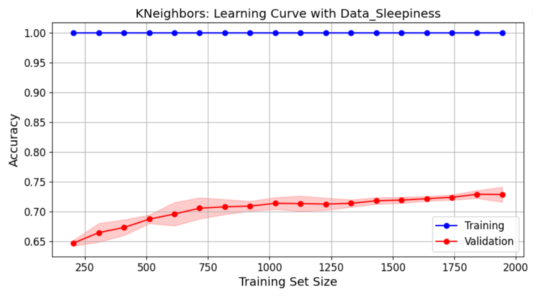

Figure 23—Learning Curve

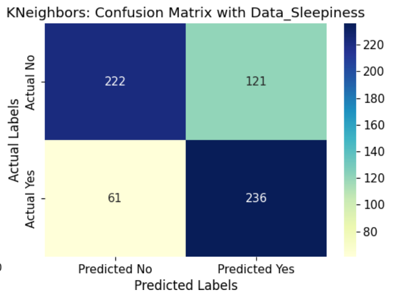

Figure 24—Confusion Matrix

2.4 Neural Networks (NN)

2.4.1 *Data1 (Data_Diabetes)*

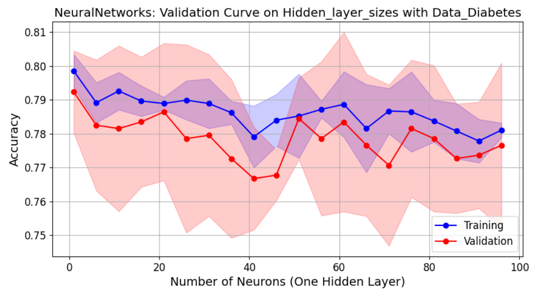

Figure 25—Validation Curve on Neuron No.

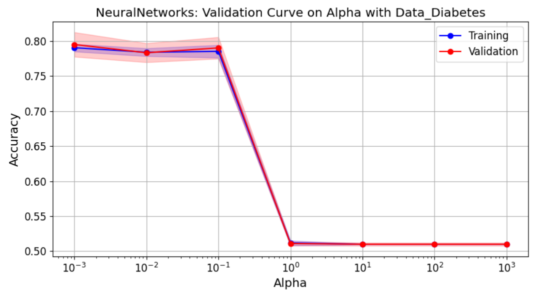

Figure 26—Validation Curve on Alpha

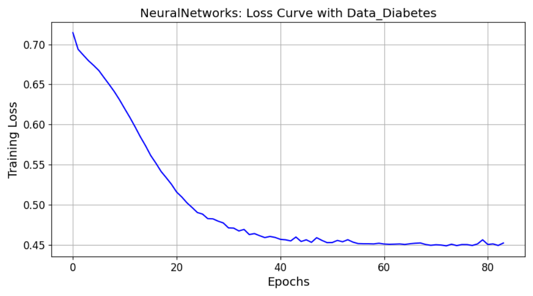

Figure 27—Loss Curve.

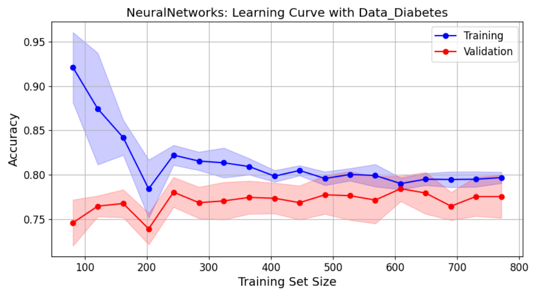

Figure 28—Learning Curve

With a single hidden layer, scores maintain low variance and high bias, which indicates underfitting (Fig. 25). To address this issue, increasing model complexity is recommended. However, in the hyperparameter tuning experiments conducted for this report, more hidden layers or neurons did not lead to significant improvement. Considering computational efficiency, a single hidden layer with 20 neurons is preferred.

Weak L2 regularization (small alpha) results in better performance (Fig. 26), with an optimal value of 0.001. The loss function steadily decreases with increasing epochs, indicating effective model learning. Convergence is signaled by the loss curve leveling off, which occurs after 60 epochs (Fig. 27). Increasing the sample size up to 200 generally reduces variance. However, underfitting is noticeable. Training and validation scores converge up to 600 instances (Fig. 28), with no performance gain beyond this point. NN shows higher levels of false positives and false negatives compared to DT, GB, and KNN (figure not shown but generated by code).

2.4.2 *Data2 (Data_Sleepiness)*

With one hidden layer, both scores exhibit low variance and high bias, signifying underfitting. Increasing complexity (number of neurons up to around 65) improves model performance but does not notably alleviate underfitting (Fig. 29). Like Data1, weak regularization (which allows the model to fit the training data closely) leads to better performance (Fig. 30).

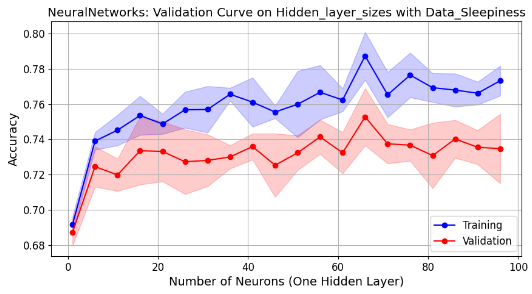

Figure 29—Validation Curve on Neuron No.

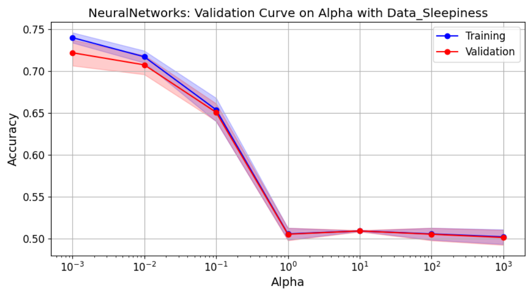

Figure 30—Validation Curve on Alpha

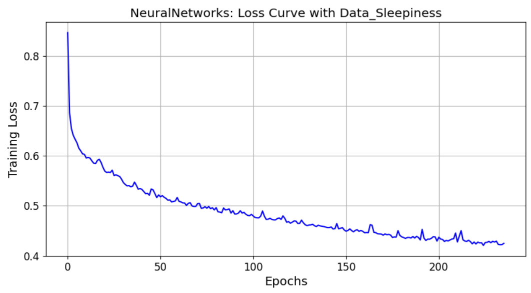

Figure 31—Loss Curve

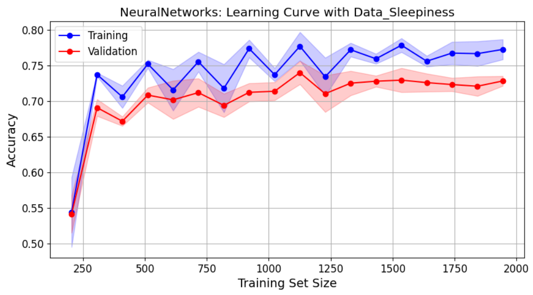

Figure 32— Learning Curve

As epochs increase, training loss decreases, indicating model is learning from data (Fig. 31) with more periodic variations than in Data1, possibly due to Data2's inherent noise. Notably, Data2 takes more epochs than Data1 to converge, illustrating Data2 presents greater complexity. Generally, increasing sample size leads to gradual improvement, but underfitting persists (Fig. 32). NN shows similar false positives as GB (figure not shown but generated by code).

2.5 Support Vector Machines (SVM)

2.5.1 *Data1 (Data_Diabetes)*

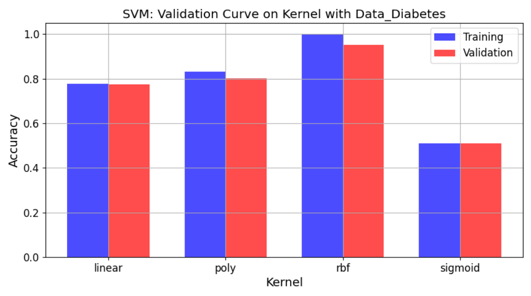

Figure 33—Validation Curve on Kernel Type

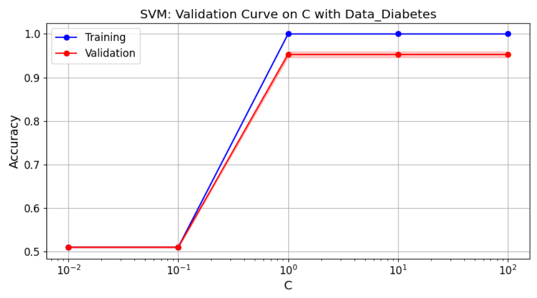

Figure 34— Validation Curve on C

For Data1, the RBF kernel yields the best performance (Fig. 33). Fig. 34 illustrates that lower regularization parameters (C) result in underfitting, whereas higher ones (C ≥ 1) achieve high scores. Incorporating more instances mitigates overfitting and further improves model performance (Fig. 35). Among all algorithms, SVM achieves zero false positives (Fig. 36), while DT, GB, and KNN exhibit zero false negatives. This highlights trade-offs in model performance: One algorithm might excel in one aspect (such as precision) but lag in another (such as recall).

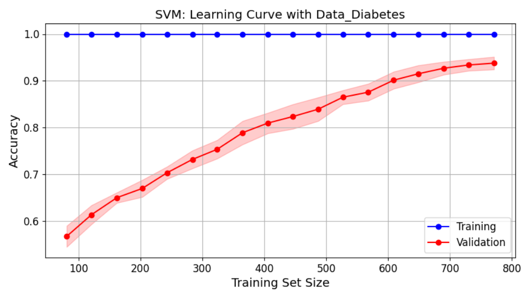

Figure 35—Learning Curve

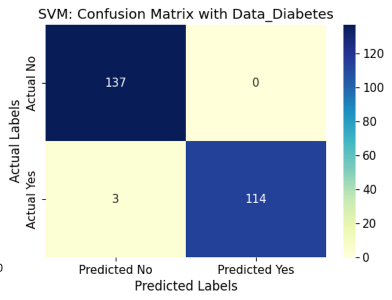

Figure 36—Confusion Matrix

2.5.2 *Data2 (Data_Sleepiness)*

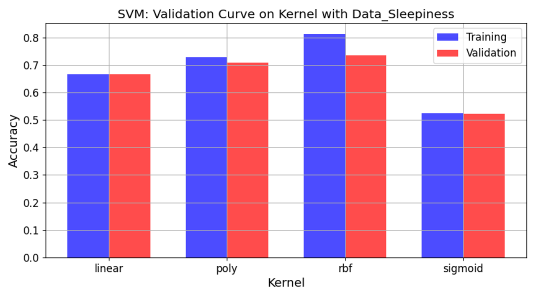

Figure 37—Validation Curve on Kernel Type

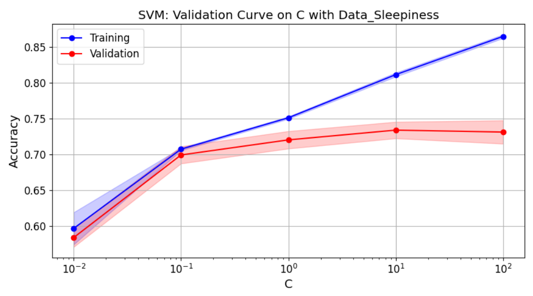

Figure 38— Learning Curve

For data2, RBF kernel yields the best results (Fig. 37). Lower C result in underfitting, whereas higher ones (C ≥ 10) tend to overfit (Fig. 38) – it aligns with the fundamental function of C, which balances the tread-off between model complexity and training error.

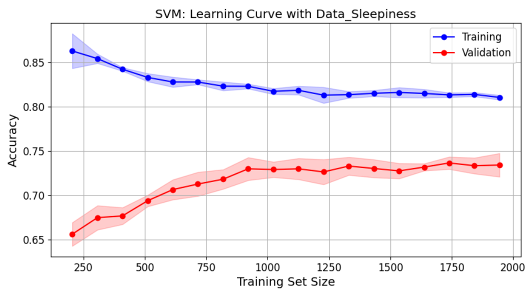

Figure 39—Learning Curve

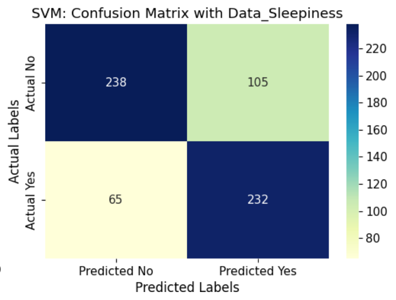

Figure 40—Confusion Matrix

Increasing samples up to 1000 results in convergence of training and validation scores; adding more instances afterward does not lead to further improvement (Fig. 39). SVM yields more false positives than false negatives and exhibits significant bias towards Type I error (Fig. 40).

2.6 Model Comparison

2.6.1 Accuracy for Testing Data

Fig.41 shows that Data1 accuracy ranges from 0.8 (NN) to 0.99 (SVM). NN is highly dependent on data suitability and often benefits from large datasets; its hyperparameter tuning is also crucial[3]. SVM can be more effective at handling outliers[4], especially when using kernel functions like the Radial Basis Function (RBF)[5].

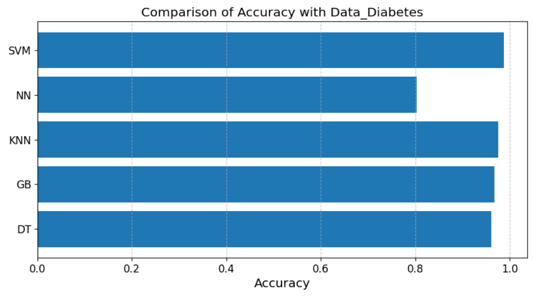

Figure 41—Comparison Across Models with Data1

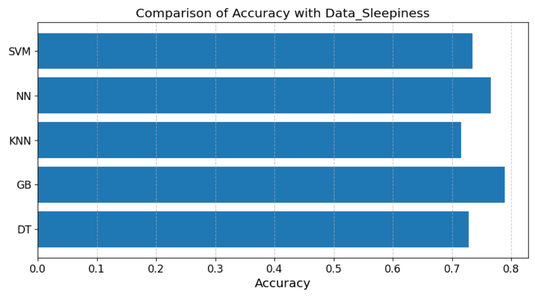

Figure 42—Comparison Across Models with Data2

Fig. 42 shows Data2 accuracy varies from 0.72 (KNN) to 0.79 (GB). Data2 is inherently complex and may contain significant noise. KNN is "very sensitive to noisy predictors since they cause

[3] https://www.analyticssteps.com/blogs/advantages-and-disadvantages-neural-networks

[4] https://www.geeksforgeeks.org/support-vector-machine-algorithm/

[5] https://towardsdatascience.com/comparative-study-on-classic-machine-learning-algorithms

similar samples to have larger magnitudes and variability in distance values."[6] Tree-based models, known for their robustness to noisy data (Ljunggren, 2021), are further enhanced when used in ensemble models (Li, 2017) such as GB.

For both datasets, GB demonstrates superior performance compared to standalone DT.

2.6.2 Clock Time

In Table 2, training time exceeds testing time for algorithms, but KNN differs by not learning during training, resulting in the shortest training time. KNN's time complexity for a single query point is O(nd)[7], where n is the sample count and d is the feature count. Data2 has more samples and more features than Data1, resulting in a longer testing time.

Table 2—Clock time across models.

Model	**Data1**		**Data2**	
	Training Time (s)	**Testing Time (s)**	**Training Time (s)**	**Testing Time (s)**
DT	0.00700	0.00041	0.01284	0.00012
GB	0.03225	0.00142	0.34581	0.00073
KNN	0.00392	0.00616	0.00135	0.01695
NN	0.07795	0.00013	1.22625	0.00138
SVM	0.02594	0.00892	0.16810	0.05580

For Data1: KNN and SVM achieve similar testing accuracy (Fig. 41). However, SVM requires more training and testing time with an RBF kernel. SVM's sensitivity to kernel choice is evident: training with a polynomial kernel of degree 2 takes considerably more time (198.76 s) than using an RBF kernel (0.078 s). NN has the longest training time and the lowest accuracy. While hyperparameter tuning to increase model complexity might improve NN's performance, given the computational expense and the high accuracies achieved by other algorithms, NN is not the optimal choice in this case.

For Data2: Due to its larger samples compared to Data1, Data2 requires more training time (except for KNN) and testing time (except for DT and GB). Besides data size, other factors like

[6] https://bradleyboehmke.github.io/HOML/knn

[7] https://arize.com/blog-course/knn-algorithm-k-nearest-neighbor

model complexity, algorithm choice, and hardware also impact training and testing time. Using one hidden layer with 20 neurons in NN is faster (0.49 s) than using 65 neurons (1.22 s). GB achieves the highest accuracy (0.79) with significantly shorter training and testing time than NN (the second highest accuracy, 0.77), making GB the optimal choice in this case.

3 CONCLUSIONS

Based on consistently positive model outcomes, Data1 (8 features and 1270 samples) appears to be of high quality. However, NN fails to perform well. NN usually requires large datasets. This may stem from model bias raised from inadequate architecture, hyperparameter tuning, or complexity mismatch. It could also be the data suitability, such as inadequate training samples.

According to the consistent model performance with Data2 (a more complex dataset with 10 features, 3200 samples), it is likely that Data2 contains significant noise, which introduces systemic bias and consequently results in low performance across models.

Different characteristics are observed among models: KNN requires more testing time and is sensitive to noise, while GB exhibits greater robustness on noisy data. SVM is sensitive to kernel selection, and NN demands more effort in hyperparameter tuning.

With Data1, SVM achieves zero false positives, while DT, GB, and KNN exhibit zero false negatives. With Data2, both KNN and SVM show a bias towards Type I error. In real-world scenarios, the significance of false negatives or false positives can vary depending on the application (e.g., cancer prediction or water quality assessment). Model selection should consider these performance trade-offs.

Interestingly, adding irrelevant features affects the models differently. When an ID column (a unique identifier for column number) is included as a feature in Data1, NN maintains the same testing score (0.8). GB and DT experience a slight decrease in accuracy, with a drop of 0.01 and 0.02, respectively. KNN's score drops from 0.98 to 0.93. It has the most significant impact on SVM (using a polynomial kernel as the best choice in this case), causing a drop of 0.14. This demonstrates how typical noise can have varying impacts on each algorithm, and GB still showcases its great noise tolerance.

4 REFERENCES

1. D. Ljunggren, S. Ishii (2021). A Comparative Analysis of Robustness to Noise in Machine Learning Classifiers. KTH Royal Institute of Technology. Stockholm, Sweden.
2. J. Li, L. Liu, J. Liu, R. Green (2017). Building Diversified Multiple Trees for classification in high dimensional noisy biomedical data. Health Inf Sci Syst. 5(1): 5. doi: 10.1007/s13755-017-0025-x.