Assignment 1 : CS 7641

Muhammad Haris Masood
mmasood30@gatech.edu

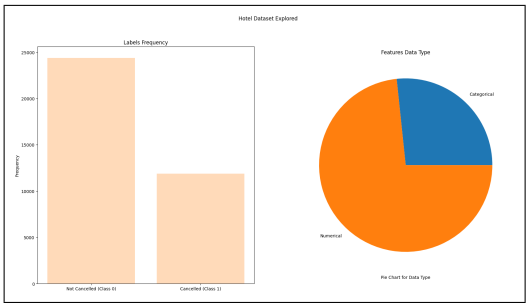

Figure 1. Hotel Reservation Dataset

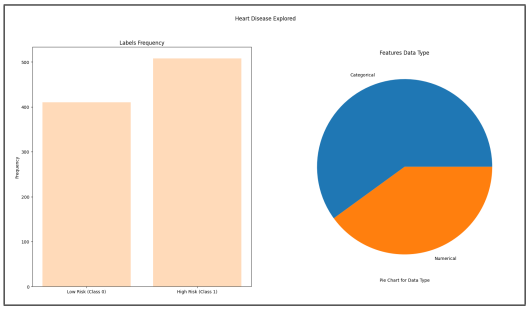

Figure 2. Heart Dataset

1. Introduction & Data

1.1. Overview

My two classification problems were; identifying if a customer would cancel their online hotel reservation [10] (hotel dataset), and, if an individual was high or low risk in terms of having a heart attack [3] (heart dataset). These were both found using Kaggle. The hotel reservation dataset provides comprehensive information about online bookings, and identifies if they were cancelled. The heart dataset is medical data, the features are several health markers for a patient, and the label identifies if the patient is at a high risk for a heart attack.

1.2. Datasets Explored

The exact characteristics of the figures can be observed from Figure 1 & 2. In some sense, the datasets are very good opposites of each other. The heart dataset is relatively balanced, whereas, the hotel reservation dataset is less so. Additionally, the hotel reservation dataset is a relatively large dataset, and has more numerical features than categorical. The heart dataset is relatively balanced, represents a smaller sample size, and there are more categorical features in the data.

PDP (Partial Dependence Plots) and Pearson R correlations revealed significant interactions among features and between features and labels in both datasets. No single feature was adequate for building a successful model; instead, relationships between multiple features and the label required consideration. Both datasets presented nontrivial challenges, making them valuable for algorithm testing. Furthermore, initial data exploration indicated that the heart attack detection dataset showed more features highly correlated with the label and generally displayed greater interaction in PDP plots.

2. Hypothesis

Based on initial data exploration we had a few hypotheses:

1) We believe that the heart dataset is more complex than the other dataset. Therefore, it is expected to exhibit poorer performance due to the greater difficulty of the problem it presents.

2) The heart dataset consists mostly of categorical variables. Therefore, it is expected to perform well with Decision Trees and Gradient Boosted Classifiers. Additionally, this characteristic may make the dataset more linearly separable.

3) The heart dataset has a relatively small sample size. Coupled with its predominantly categorical nature, which leads to a sparser representation due to one-hot encoding, this is expected to result in easier overfitting (low bias, high variance).

4) The hotel reservation dataset includes two features that are strongly correlated with the label: "lead time" and "number of special requests." Therefore, we believe that adding noise to any of these features will cause a strong degradation in model performance.

5) Due to the less balanced nature of the hotel reser-

vation dataset, we believe, ensemble methods may work quite well [4].

3. Testing Methodology

3.1. Preprocessing Data

For the heart dataset, we removed the Booking ID column, as that was just an index value. For the hotel reservation dataset, we combined the date and time columns to produce two features. The first of which identified if the booking was in the first or second half the month, the second, identified if the booking was in the summer or winter. All categorical features were one-hot encoded to avoid any incorrect ordering in the data.

3.2. Metrics

Due to the somewhat balanced nature of both datasets and the aim of this paper, which is to compare differences in datasets and algorithms, we have selected accuracy as our primary metric for most of our experimentation and hyperparameter tuning plots. We previously mentioned that the hotel reservation dataset is less balanced, which is indeed the case. However, even if the learner outputs the majority class label, the maximum achievable accuracy is only approx. 67%. Consequently we still consider accuracy to be a viable metric for our paper. We did experiment on learner performance when faced with extremely unbalanced datasets (more on this later) and for this experiment we used an F1 score. For some of our experimentation plots we used % score of the max. For example, if introducing noise dropped the accuracy to 0.9 from 0.95 the corresponding value on the graph would be 94.7%. This percentage of the max was chosen as it made it easier to compare the learners amongst themselves, and, between datasets. We also measured clock time both for learning and inference. The average of 10 iterations for the optimal learner was used. This timing metric was used as it allowed for the clearest comparison between algorithms, and, comparisons between the datasets themselves.

3.3. Procedure

To begin, each algorithm underwent hyperparameter tuning. This process involved varying one hyperparameter at a time while keeping the others constant. To accurately assess the effect of varying the hyperparameter value, 3-fold cross-validation was performed at each step. Three folds were chosen due to computational limitations, as a wide variety of hyperparameters were tested for all learners. Initial testing indicated that 3-fold cross-validation was sufficiently accurate in assessing the actual effect of the hyperparameter on the learner.

After hyperparameter tuning, a narrower range of hyperparameter values were selected and used alongside Scikit-learns GridSearch method to identify the optimal hyperparameters for the learner. The optimal learner was then used on the test set and it's score evaluated.

After hyperparameter tuning and final scoring, experimentation on the algorithms was conducted. The experiments were conducted on a range of algorithms using a combination of hyperparameters. The hyperparameters themselves were a narrowed down range of best performing hyperparameters found from the validation plots. Due to space limitations, only results of the best-performing algorithm were graphed and added to the report. Four different experiments were conducted on each learner. The first involved adding random noise to the most important numerical feature (identified by simple correlation with the label). The second involved changing the dataset to make it unbalanced (removing Class 1 labels). The third involved adding noisy features to the dataset. The final experiment was the effect on training size on model performance. These experiments were conducted with the theme of exploration, and, as a result the entire dataset was used for each experiment.

4. Algorithm Intros & Hyper parameter Tuning

4.1. Decision Trees

For Decision Trees, we did not normalize the data. We made this choice because decision trees do not require normalization, which is a noted strength of this method. Normalizing the data could introduce additional bias, as we can only train the normalizer on the training data and assume it's a good representation of the test data. We also used the GINI criterion for assessing splits. We chose GINI as research indicated that its performance was similar to information gain, but, could require less compute.

Observing Figure 3 & Figure 4, we make several important observations. As the max depth increases, there is a risk of overfitting, and we see overfitting quite clearly in the heart attack dataset, as validation accuracy plummets after a depth of 5. We believe this is because the heart attack dataset is relatively small. We also believe that we have increased the likelihood of overfitting by one-hot encoding all the categorical variables, resulting in a higher dimensional dataset. It is possible that a combinations of these factors is leading to the heart dataset being easier to overfit to. The CCP Alpha, or amount of pruning, also conveys this message. The heart dataset works better with a higher CCP alpha, indicating more aggressive pruning. Similarly, the min_samples per split also indicates this to be the case. The higher the value, the more regularized the tree becomes, resulting in shallower depth and a simpler model. The max_features parameter was interesting; we expected the heart dataset to prefer either the sqrt or log2 methods instead of the "default". We believed this would be the case

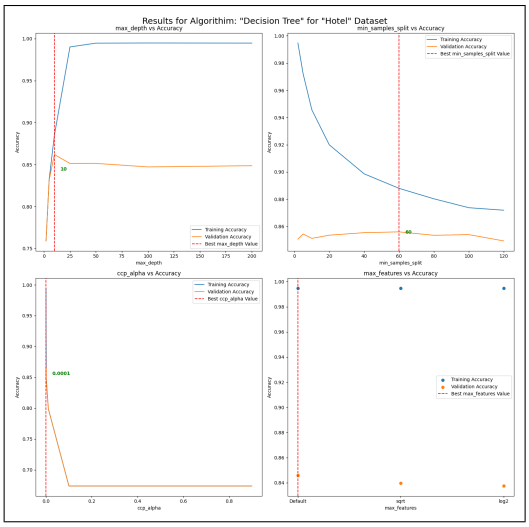

Figure 3. Hotel Dataset Decision Tree Tuning

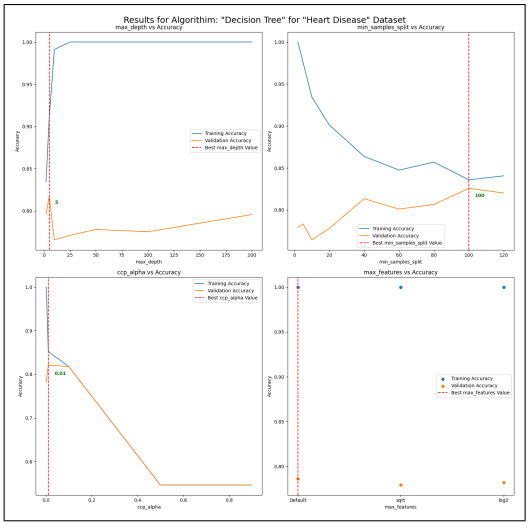

Figure 4. Heart Dataset Decision Tree Tuning

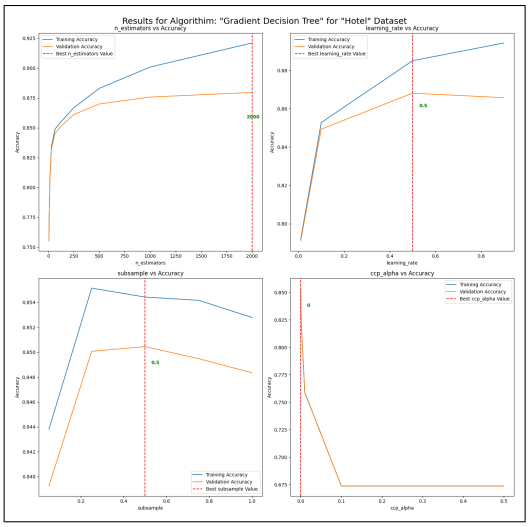

Figure 5. Hotel Dataset Gradient Boosted Classifier Tuning

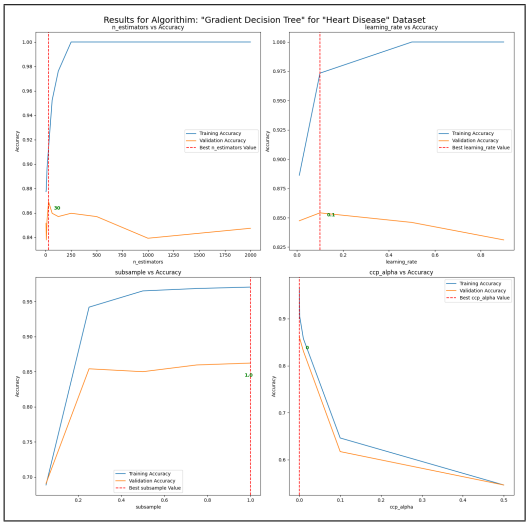

Figure 6. Heart Dataset Gradient Boosted Classifier Tuning

because these two methods regularize the tree by limiting the number of features a learner sees when performing a split. However, we now believe that these two "regularizing" methods were not preferred because the neither dataset has a very large number of features (such as genetic data), and limiting features for our tree for our relatively "simpler" datasets has too strong of a regularizing effect. Although it has to be noted that all max_feature values performed similarly to one another for both datasets.

4.2. Gradient Boosted Classifier

Similar to Decision Trees we did not normalize the data as being able to work well with non-standardized data is a noted benefit of tree based methods

Upon analyzing Figures 5 & 6, we notice a trend similar to Decision Trees. The heart dataset, due to its size and it being a higher dimensional dataset, is prone to overfitting. To address this, we required fewer estimators or a lower learning rate for each learner.

There are some additional interesting observations we can make from the data. In the lecture videos, it was discussed that boosting ensemble methods are resilient to overfitting. Indeed, this is what we observe with the hotel dataset. It appears that even with a very large number of estimators, no overfitting has occurred yet. We believe this is due to a combination of the characteristics of the dataset (a large amount of data & a good mix of categorical and numerical features) and the characteristics of the learner (ensemble method, voting on results, each new tree focuses on errors of the previous tree). The same effect is not observed

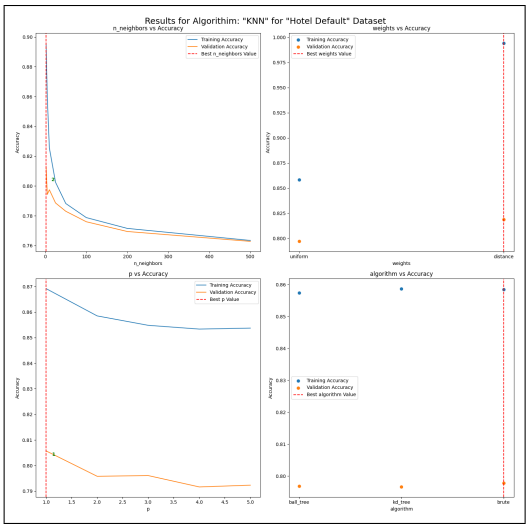

Figure 7. Default Hotel Dataset KNN Tuning

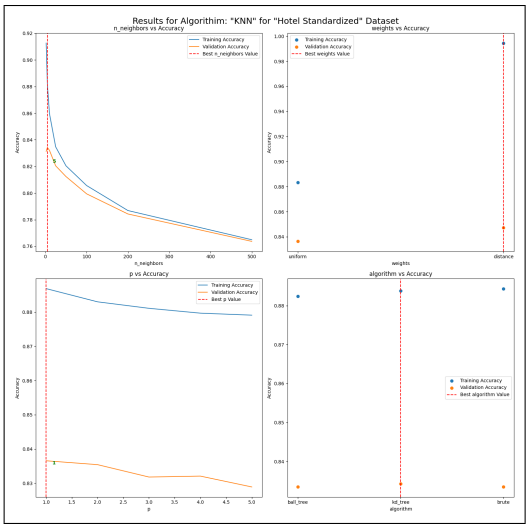

Figure 8. Standardized Hotel Dataset KNN Tuning

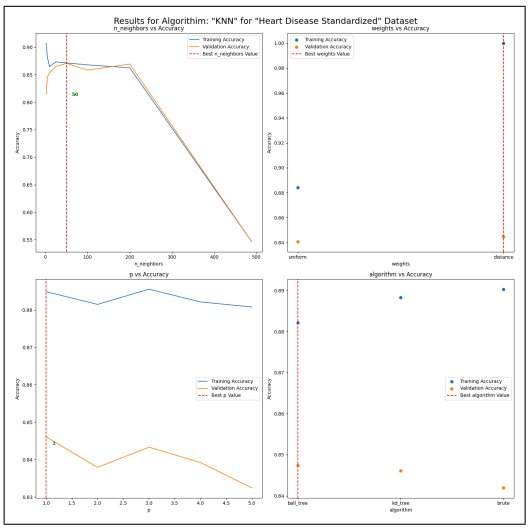

Figure 9. Standardized Heart Dataset KNN Tuning

in the heart disease risk dataset, and, that again highlights how the heart disease risk dataset is more prone to overfitting.

The differences in ease of overfitting between datasets can also be observed when we look at the graph for the learning rate's. The heart dataset prefers a lower learning rate. This likely indicates that each individual tree may be overfitting the data. Finally we note that the hotel dataset preferred a lower sub sample value then the heart dataset. This is interesting, it may be because the hotel dataset had enough data points within the dataset to benefit from the added stochasticity of not using the entire dataset to train the learner. This is likely something the heart dataset is unable to take advantage of, due to it's limited size. It must be noted that the differences in score between sub sample values within the range of 0.5-1 are quite small, so it might just be noisy patterns in the dataset that are causing the discrepancy.

The results also highlight minimal pruning for both datasets. This is deliberate, as computational constraints led to a cap on the max depth for each learner at 3, instead of allowing them to grow extensively and then pruning. Given the low max depth, pruning is essentially unnecessary for both datasets.

4.3. KNN Classifier

For KNN, we did normalize the data. We used an Sklearn Standard Scaler that was trained on the training data. We used that trained scaler to only transform the testing data.

The effect of standardization is evident in Figure 7 and Figure 8. Standardized data consistently yields higher accuracy scores. This is straightforward: KNN relies on distance for learning, and if feature distances are on very different scales, it adversely affects its learning process. A comparable outcome was noted in the heart dataset but not shown due to space limitations.

Examining Figure 8 & Figure 9 reveals intriguing insights about the data. As anticipated, an increase in the number of neighbors results in a decline in both training and test accuracy, indicative of under-fitting. Notably, this drop is more pronounced in the heart dataset. Also to note, the heart dataset prefers a comparatively high value of K. We believe both of these are due to the characteristics of the dataset. The heart dataset is primarily categorical, this means that in essence, for most of the feature's there would be two distinct groups in the hyperspace. For this reason we believe it prefers a high K value, as it is better able to as-

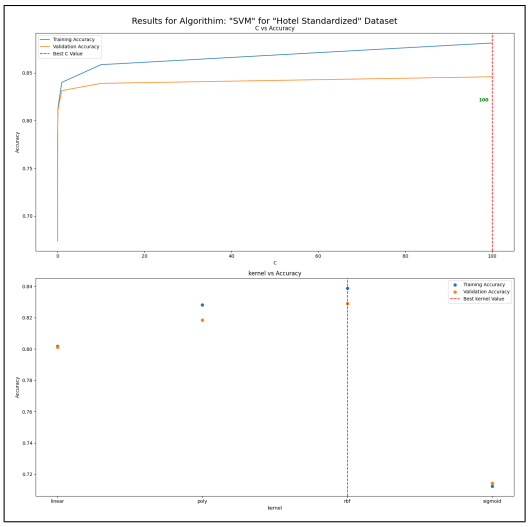

Figure 10. Standardized Hotel Dataset SVM Tuning

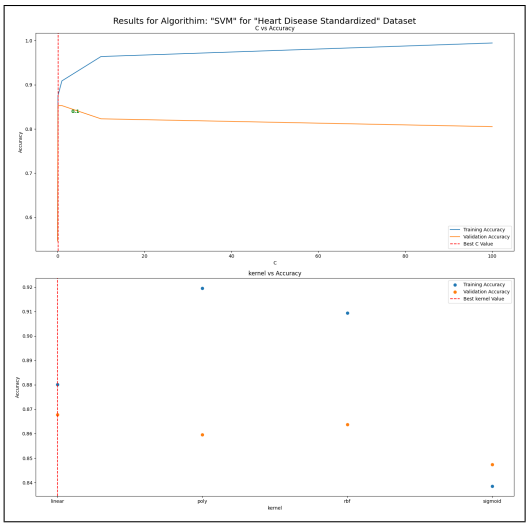

Figure 11. Standardized Heart Dataset KNN Tuning

certain the mapping between the binary features and the label. We believe the rapid degradation in the learners performance is due to the size of the dataset, as increasing learners means that the KNN learner is now essentially acting as an aggregation function.

The p value in Minkowski distance is intriguing. Increasing it leads to a notable decline in learner performance for the heart dataset, likely due to its higher dimensionality compared to the hotel dataset, favoring lower p values [1]. For the hotel dataset difference in scores between p values is very small, and this could be simply due to random patterns.

We also note that for both datasets, all algorithms performed similarly well. Although all three provide similar results, the KD tree and Ball tree are better choices here due to the higher computational and memory costs of the brute search algorithm.

4.4. SVM Classifier

For SVM, we did normalize the data. The methodology for normalization was the same as that for KNN. We are unable to include the graphs for the non standardized data due to space limitations, but, it was noted that standardization provided a significant boost to model performance. Standardization is beneficial as SVM takes distances into account when calculating a decision boundary, and features with greatly mismatched distances negatively impact model performance.

Results for our hyperparameter tuning can be seen in Figure 10 & Figure 11. For our hyperparameters, we varied the kernel and the value of C. The value of C controls regularization. In general, a low value of C causes underfitting (high bias, low variance), and a high value of C causes overfitting. In the heart dataset, we observe the overfitting portion quite clearly. It is interesting to note that the same overfitting does not take place in the hotel dataset. As suspected earlier, the fact that the heart dataset is relatively small, and, is comprised of mostly categorical variables could be the reason it is more likely to be overfit too.

Varying the kernels also led to some interesting conclusions. As hypothesized at the start of the paper, the heart dataset performs well with the linear kernel. We believe this could be attributed to its primarily categorical nature, allowing the data points to be effectively separated into two groups by a linear function in hyperspace. Although its should be noted that the poly and rbf kernel also performed well. In the hotel dataset the rbf kernel performed the best. This may be due to the data being less linearly separable by having a complex arrangement in hyperspace. The sigmoid kernel performed poorly for both datasets, but, this may be due to the fact that it requires more careful hyperparameter tuning [5].

4.5. Neural Networks

When it comes to standardization, we encountered conflicting information in literature. Some papers asserted the necessity of standardization, while others mentioned that neural networks possess self-scaling properties [12]. To reconcile this, we compared the algorithm's performance on both standardized and default datasets, the results of which can be found on Figures 12 & 13. Standardization showed a significant improvement. Consequently, we opted for standardization on both datasets. Due to space constraints, we omitted the graphs for non-standardized heart dataset.

For our Neural Network algorithm we varied many parameters which resulted in some interesting findings (Figures 13 & 14). For the heart dataset it appears a large batch size was preferred. In general, smaller batch sizes are preferred as there is research indicating that larger batch sizes

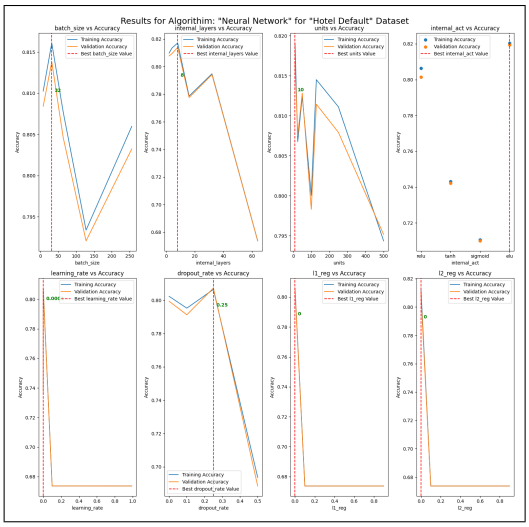

Figure 12. Standardized Hotel Default Neural Network Tuning

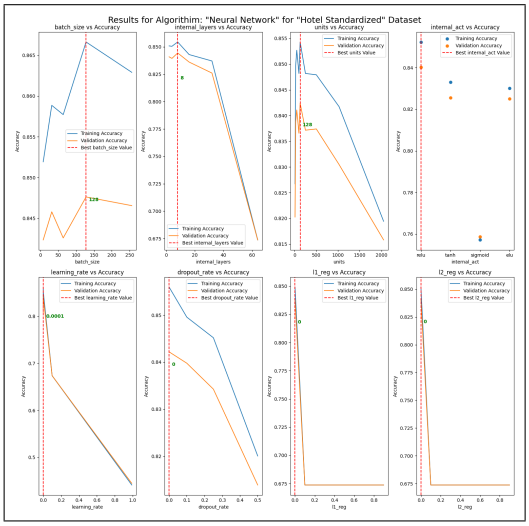

Figure 13. Standardized Hotel Dataset Neural Network Tuning

lower model reliability [8]. The fact that a batch size of 256 worked best is interesting. One theory we have is that the larger batch acts as a form of regularization, in a fashion similar to adding noise to a dataset.

We also note the effect of increasing the number of internal layers. It appears, as consistent with other research, that increasing the layers beyond a certain point leads to the degradation of model performance [7]. The exact reason seems difficult to pinpoint, but not enough data and/or too difficult of an optimization problem have been discussed as potential reasons in literature. We observe that both datasets exhibit this behavior. Too few layers lead to the model being underfit. The number of units in each layer displayed similar behavior. Initially, when we increase the number of nodes in each layer we see the model moving from a re-

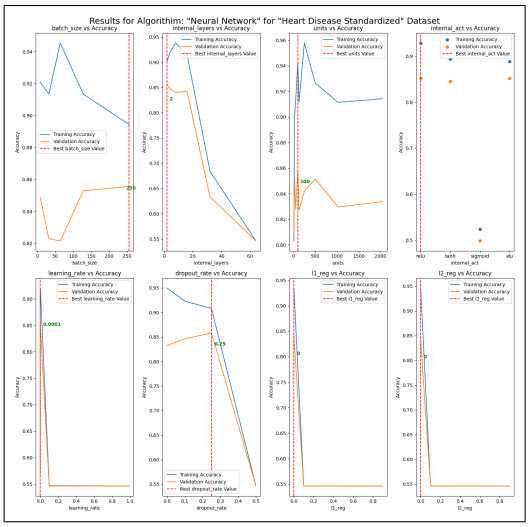

Figure 14. Standardized Heart Dataset Neural Network Tuning

gions of underfit to a region of optimal fit. However, after a certain threshold more units are actually detrimental to the model and cause degradation of performance. We see that for both layers and units the degradation begins sooner for the heart dataset. It appears that perhaps dataset size, or, optimization complexity might be the reason for the degradation we see. In terms of the differences between datasets we note that the Hotel dataset performed best with both more layers, and, more units. Consequently it does appear from these results that the hotel dataset is actually more complicated then the heart dataset. This is contrary to what we noted in the hypothesis.

From our results we also noted that the sigmoid activation performed very poorly on both datasets. From our research it appears that the sigmoid activation function suffers from output saturation and vanishing gradients, and this may be causing the poor performance [11] [6].

For learning rate, a lower learning was preferred in the heart dataset then for the hotel dataset. We believe this may be due to several factors, including, heart dataset being sparse [2], the heart dataset being more complex [9], or possibly, the heart being easier to overfit to [9].

Finally, let's examine regularization. We note that the heart dataset preferred some dropout, whereas, the hotel dataset performed best without dropout. The conclusion we can draw from this is that the heart dataset is more susceptible to overfitting, this may be due to it's more categorical nature (leading to a sparse representation) or less training data. We do however note neither dataset preferred having any L1 or L2 regularization. We suspect that this may be due to our neural net architecture being simple (a sequential model).

Finally, we observe Figure 15 to see the effect of the number of epochs on loss. Initially the model is underfit

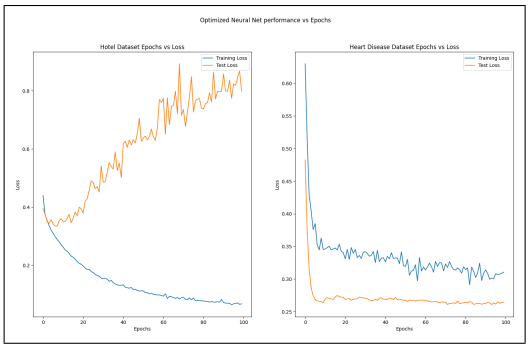

Figure 15. Epochs vs Loss

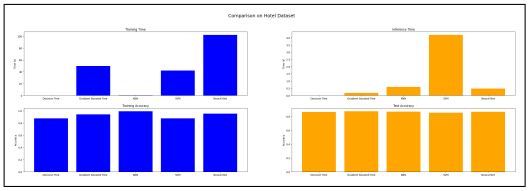

Figure 16. Hotel Dataset Final Results

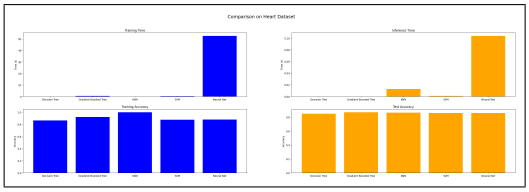

Figure 17. Heart Dataset Final Results

(high bias, low variance), then it achieves an optimal fit, and finally it enters the region of overfit (low bias, high variance). It is interesting to note that the hotel dataset appears to overfit more as the number of epochs increase. We believe this is because the optimal neural network structure for the hotel dataset is significantly more complex than that of the heart dataset.

5. Model Performance

We can compare model results using the Figures 16 & 17, as well as, the Table's 1 & 2. From the results, we can conclude that all models performed relatively well on both datasets after extensive hyperparameter tuning.

For the hotel cancellation dataset, the Gradient Boosted learner performed the best, while the SVM learner performed the worst. SVM's performance may be a result of what we discussed earlier. In our hyperparameter tuning phase, we noted that the hotel dataset was not particularly linearly separable. It appears that this may be causing the relatively poor performance by the SVM learner. It is possible that the dataset is fairly inseparable even in higher order dimensions, and, the kernel trick is rendered ineffective.

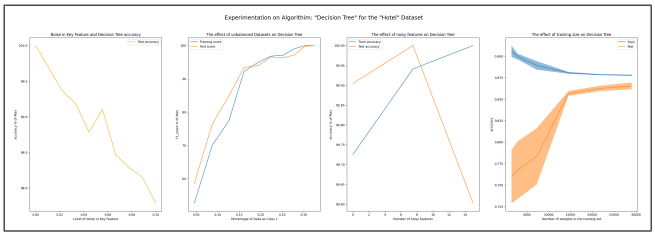

Figure 18. Hotel Dataset Decision Tree Experimentation

We attribute the success of the Gradient Boosted Classifier to its ensemble learning approach, which proved effective in handling the dataset's mix of categorical and numerical features. Additionally, its ability to address the dataset's imbalance likely contributed to its superior performance.

For the heart dataset the Gradient Boosted learner again performed the best, and surprisingly, the Decision Tree performed the worst. In our hypothesis we stated that we believed that the tree based methods would preform well based due to the primarily categorical nature of the dataset. However, it appears that that is not completely the case. The Boosted tree did indeed perform well, but, the simple Decision Tree performed poorly. Two potential hypothesis we have is that a simple Decision Tree is unable to accurately model the interaction between the features and the labels, or more likely, suffers from overfitting due to the nature of the dataset. The Gradient Boosted learner is better able to handle overfitting as it is able to train many trees, each focusing on the errors of the last tree.

In terms of training & inference times the pattern was the same across both datasets. Neural networks took the longest time to train, whereas the Decision Tree took the shortest time. The lengthy training time of neural networks aligns with findings in the literature. Training the entire network requires learning over several epochs, with each epoch iterating through the entire training set. The significant difference in magnitude between training and inference times for the Gradient Boosted learner also corresponds with existing literature. During training, trees are sequentially added and trained, which is time-consuming. However, during inference, the trees can be evaluated in parallel. It is also interesting to note that the KNN learner had a faster training than inference time. This helps show why the KNN learner is known as a "lazy" learner.

We also note that for both datasets the KNN learner appeared to have significantly overfitted the data. It appears that more regularization may be needed for the KNN, or, an ensemble KNN method is required.

Learner	Training Score	Test Score	Training Time (s)	Inference Time (s)
Decision Tree	0.8764	0.8666	0.109	0.001
Gradient Boosted Tree	0.9430	0.8759	49.762	0.181
KNN	0.9939	0.8699	0.352	0.594
SVM	0.8780	0.8557	42.310	4.201
Neural Net	0.9556	0.8680	102.492	0.487

Table 1. Hotel Cancellation Dataset Final Results

Learner	Training Score	Test Score	Training Time (s)	Inference Time (s)
Decision Tree	0.8638	0.8478	0.005	0.000
Gradient Boosted Tree	0.9223	0.8696	0.456	0.000
KNN	1.0000	0.8641	0.007	0.013
SVM	0.8774	0.8587	0.370	0.001
Neural Net	0.8787	0.8587	52.674	0.104

Table 2. Heart Dataset Final Results

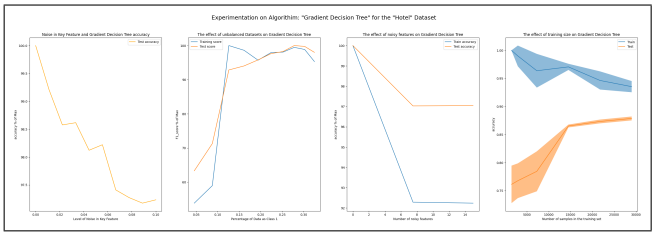

Figure 19. Hotel Dataset Gradient Boosted Tree Experimentation

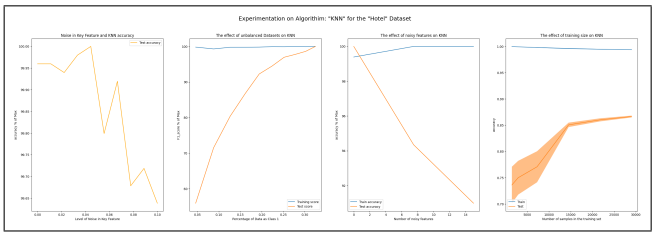

Figure 20. Hotel Dataset KNN Experimentation

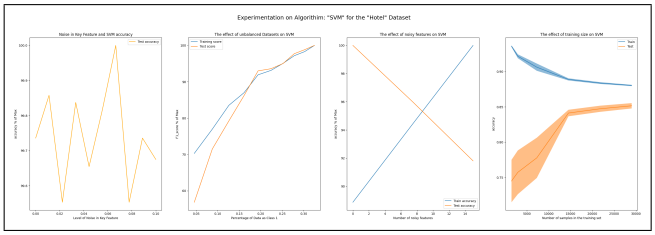

Figure 21. Hotel Dataset SVM Experimentation

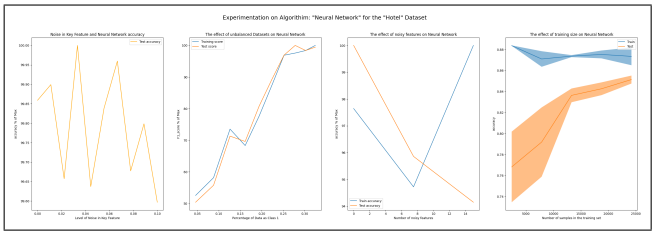

Figure 22. Hotel Dataset Neural Net Experimentation

6. Algorithm Experimentation

6.1. Hotel Dataset

From our graphs it appears that both tree based methods were the most susceptible to noise for the hotel dataset. While it is challenging to pinpoint the exact reason for this, we have a leading theory. The noise was added to the feature most correlated with the label. For the hotel dataset the feature that had the highest correlation had a correlation value much higher than the other features. We believe, that for this dataset correlation could be used to infer predictive power. Consequently, we believe that both tree based methods use this feature extensively to preform splits, as tree based methods are a top down greedy algorithm (performing the best split only taking the next step into account). The other learning methods do not show this greedy behaviour, instead focusing on a globally optimal solution. For this reason, we believe that SVM and Neural net-based learners can perform well even with minor noise added to the most correlated feature. This is because their weights have been optimized with all features in mind, striving for a globally optimal solution. KNN learners, by principle, take all features into account and again this is likely why the KNN learner was not effected by the noise.

For the unbalanced dataset experiment we note that the Gradient Decision Classifier performed significantly better then the other learners. This conclusion is in line with the literature indicating that ensemble based methods preform well on unbalanced data [4]. This highlights a particular strength of ensemble methods, notably, their ability to preform well on unbalanced datasets by utilizing many weak learners and then aggregating their predictions.

For our experiment on adding noisy features to the dataset, we observed that Decision Trees performed remarkably well. We believe this occurred for two reasons. Firstly, the GINI coefficient was utilized at each step, and it's likely

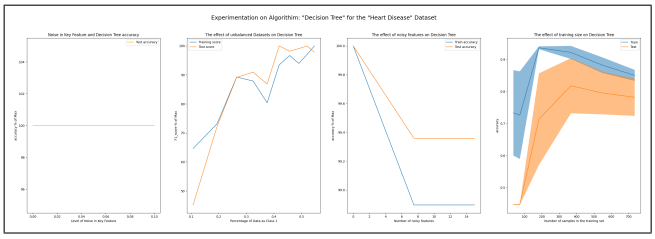

Figure 23. Heart Dataset Decision Tree Experimentation

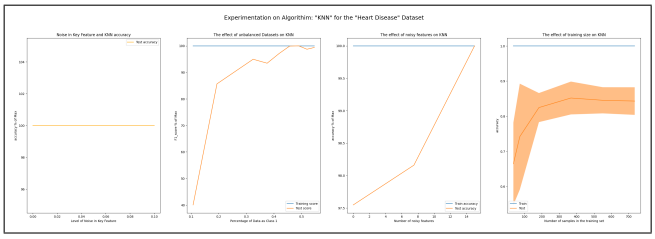

Figure 25. Heart Dataset KNN Experimentation

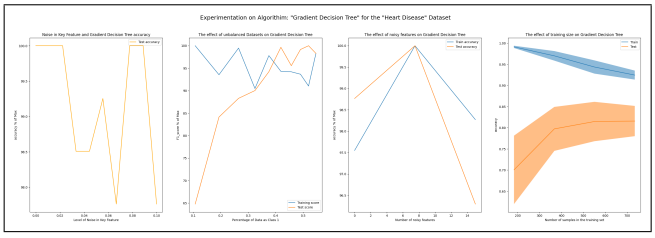

Figure 24. Heart Dataset Gradient Boosted Tree Experimentation

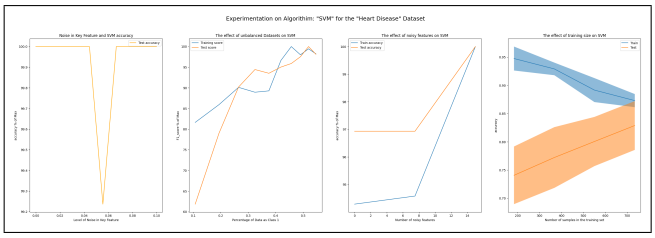

Figure 26. Heart Dataset SVM Experimentation

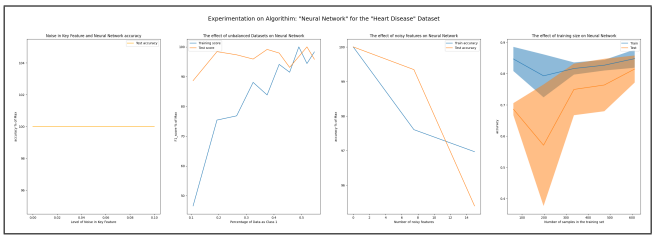

Figure 27. Heart Dataset Neural Net Experimentation

that in most of the splits, the algorithm detected that the noisy features did not work well in splitting the data, and as a result, these were ignored. Secondly, our use of post-pruning likely meant that splits that may have used noisy features in training would have been removed. As a result, the Decision Tree appears to be very effective in ignoring the noisy features. Gradient Boosted Trees also performed well, but not as well. We believe that if we had utilized post pruning for the Gradient Boosted Classifier, we would have noted similar performance to that of the Decision Tree. The other learners did not fare as well, and unfortunately, began fitting to the noisy features, leading to increasing training accuracy but degrading test accuracy (as initially hypothesised). These graphs help illustrate the importance of proper feature selection.

The final experiment was that of varying training size and measuring model performance. We note that all learners behaved similarly, if the training size is too small the learner overfits the data, it has low bias but has very high variance. The result of this is that it generalizes very poorly to test data as due to the limited training data has become overtly sensitive to noise. Increasing the training size helps remedy this problem, as training size increases model generalizability greatly improves, indicating that the model retains low bias but now also lowers its variance.

6.2. Heart Dataset

From our experiment on adding noise to the most correlated feature we note that there is virtually no effect on model performance. We believe there are two potential reasons for this. The first is that in the heart dataset correlation cannot be used to infer predicting power. The second is that unlike the hotel dataset, there are many features that have a strong correlation with the label, hence, it is possible that the learner is able to utilize the other features to overcome the noise.

For our unbalanced dataset experiment we note that the ensemble method (Gradient Boosted Classifier) performed well and this further increases our confidence when we state that we believe that ensemble learners are well suited for extremely unbalanced data. However, the main surprise was the stellar performance of Neural nets. We are not exactly certain what is causing this outlier result (the test score is nearly double the training score) and this requires further exploration.

Our experiment involving adding noisy features again showed similar results to that of the hotel dataset, with Decision Tree's showing great invariance. We believe the reasoning for this is similar to what we discussed in the last section. Interestingly, it appears that performance of the SVM and KNN learners actually improved when we added noisy features. One potential hypothesis is that the inclusion of noisy features helps mitigate the detrimental effects of data sparseness. Although an interesting theory, more in-depth experimentation/evaluation on this is needed.

The experiment on the effect of training size yield simi-

lar results across all learners, indicating that the model goes from a region of low bias and high variance to a region where low bias is maintained but the variance is reduced. Based on hyperparameter performance and the large confidence lines in our validation scores, it does appear we need to increase the training data for our heart dataset (collect more samples).

6.3. Comparison Between Datasets

From the graphs it can be observed that when we add random noise to the the most correlated numerical feature (with the label) the heart dataset shows no effect. There are two possible conclusions we can draw from this. The first is that in the heart dataset correlation is not a good indicator of predictive ability. The second is that even though we have added noise to the most correlated numerical feature the other features, with their collective predictive power, are able to overcome the added noise. This is in contrast to the hotel cancellation dataset, where even small additions of noise degrades model performance.

Also to note is that in the heart dataset the SVM & KNN learner performance actually improved when we added noisy features. This phenomenon was not observed at all in the hotel dataset. This may be due to the sparser nature of the heart dataset.

7. Conclusions & Next Steps

We noted several hypothesis at the start that we had developed while performing initial data exploration. Through hyperparameter tuning, model evaluation and model experimentation we can evaluate if our initial hypothesis.

Our first hypothesis was that the heart attack dataset was more complex. Model performance on test sets indicated lower scores than the hotel dataset. This might be due to the problem being more complex, or, it could be due to the models overfitting the training data. Our Neural net exploration did cast doubt on this hypothesis as a relatively simple structure was chosen as the optimal learner. Consequently, we cannot conclude if the heart dataset is indeed more complex. We now believe that the heart disease prediction problem may actually be the simpler of the two. Further experimentation is needed to confirm.

Our second hypotheses was that tree based methods would work well on the heart attack dataset due to it's more categorical nature. Unfortunately we did not see this conclusively. Decision Trees performed poorly, and, although the Gradient Boosted Classifier worked well, that could have been due to it being an ensemble method. Lack of training data, a higher dimensional dataset, and, a sparse representation may have resulted in this poor performance.

Our hypothesis that the heart attack dataset was more prone to overfitting was supported by our experimentation. Nearly all learners faced overfitting issues with this dataset, necessitating constant counteraction strategies. The reasons could be that the heart dataset was higher dimensional dataset, had sparse representation, or, just lacked enough training data.

We also hypothesised that in the hotel dataset adding noise to the most correlated feature would have a strong negative effect on model performance. We were able to confirm that our initial hypothesis was indeed correct. Experimentation revealed rapid degradation in model performance when even a small amount of noise was added to the most correlated feature (with the label).

Finally, we note that ensemble methods performed exceptionally well for the hotel dataset, aligning with our initial hypothesis given its unbalanced nature. Our experimentation further demonstrated that ensemble methods are well-suited to handle unbalanced datasets.

Although we did perform detailed and in-depth experimentation, not all questions we had were answered, and new ones arose when we evaluated our results. The increasing accuracy for the KNN & SVM learner as noisy features are added to the data and the surprisingly good performance of the Neural net on unbalanced heart data are all questions that can be explored more in depth.

References

[1] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. In Jan Van den Bussche and Victor Vianu, editors, *Database Theory — ICDT 2001*, pages 420–434, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. 5

[2] Utku Evci, Fabian Pedregosa, Aidan N. Gomez, and Erich Elsen. The difficulty of training sparse neural networks. *ArXiv*, abs/1906.10732, 2019. 6

[3] Fedesoriano. Heart failure prediction dataset, 2021. 1

[4] Mikel Galar, Alberto Fernández, Edurne Barrenechea, Humberto Sola, and Francisco Herrera. A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42:463 – 484, 07 2012. 2, 8

[5] Sourish Ghosh, Anasuya Dasgupta, and Aleena Swetapadma. A study on support vector machine based linear and non-linear pattern classification. In *2019 International Conference on Intelligent Sustainable Systems (ICISS)*, pages 24–28, 2019. 5

[6] D Greig, T Siegelmann, and M Zibulevsky. A new class of sigmoid activation functions that don't saturate. 1997. 6

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 6

[8] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *CoRR*, abs/1609.04836, 2016. 6

[9] Haidong Li, Jiongcheng Li, Xiaoming Guan, Binghao Liang, Yuting Lai, and Xinglong Luo. Research on overfitting of deep learning. In *2019 15th International Conference on Computational Intelligence and Security (CIS)*, pages 78–81, 2019. 6

[10] Ahsan Raza. Hotel reservations dataset, 2023. 1

[11] Matías Roodschild, Jorge Gotay, and Adrián Will. A new approach for the vanishing gradient problem on sigmoid activation. *Progress in Artificial Intelligence*, 9:351–360, 12 2020. 6

[12] M. Shanker, M.Y. Hu, and M.S. Hung. Effect of data standardization on neural network training. *Omega*, 24(4):385–397, 1996. 5