CS7641 - Assignment 1 Supervise Learning

Ashish Panchal
OMSCS Student - College of Computing
Georgia Institute of Technology
apanchal33@gatech.edu

***Abstract*—This Assignment explores 5 supervised learning techniques to understand associated behaviour in different circumstances. These behavioural differences shape the preference of one algorithm over other in solving various problems, aligning with the interests and goal of the applicant. As per instructions, the scope of the study is limited to 2 classification tasks with varying degree of complexity entailing the prediction problem, where the implementation focuses on the trade-off between variance and biases of each algorithm, levered using associated parameters. Based on the observations and analysis we aim to comment on associated reason for such behavioural differences in and across these algorithms.**

***Index Terms*—Supervised learning, Classification, Decision Tree, Boosting, Neural networks, State vector Machines, K Nearest Neighbours.**

I. Introduction

- Introduce your data. Talk about why it's interesting (from an ML POV) and what preprocessing you did (be brief).
- Talk about your experimental methodology.
- Discuss results isolated to each algorithm.
- Compare and contrast results across algorithms, across data sets.
- A solid conclusion to wrap up your journey. Tie in some stuff you learned from lecture that applied to the assignment and more!

This document is a model and instructions for LaTeX. Please observe the conference page limits.

II. Problem statement and Data

Two datasets were choosen for this assignment as detailed below.

A. Task 1 : Wine Quality classification

The Wine Quality dataset, sourced from Kaggle [1], provides psychochemical and sensor data for Portuguese "Vinho Verde" wine. After cleaning, it consists of 6,463 records and 12 features. Among these features, 11 are continuous, covering aspects like acidity, sugar content, and more, while 1 (the "type") is categorical with two categories.

The target variable, "quality," is ordinal, ranging from 3 to 9, with higher values indicating better wine quality. Notably, the dataset's quality distribution is imbalanced, with the majority falling into categories 5, 6, and 7 (32%, 46%, and 16% respectively), while the remaining four categories each represent less than 3% of the total distribution. To mitigate prediction errors on both extremes, we introduced a derived target feature called "quality_cat," indicating "average or better" quality (quality ¿ 5), resulting in two categories with a distribution of 36.7% and 63.3% for low and average-or-better quality, respectively.

Cleaning the data involved removing rows with missing values, which accounted for only 0.5% of the total data. This step did not compromise the generalization of the dataset. To simplify analysis, both the "type" and "quality_cat" features were converted into Boolean variables, where "white" and "average or above" were assigned values of 1.

Univariate distribution of all features and the target variable (Figure 1) shows 8 out of the 11 numeric features exhibit significant skewness, except for "fixed acidity," "pH," and "alcohol." Different machine learning models may respond differently to these distributions, making it an interesting aspect to explore.

(Figure 2) reveals no strong correlations in the data, except for "total sulfur dioxide" with "free sulfur dioxide" and the "type" feature, showing correlations of -0.72 and 0.7 respectively, these were found relevant in study.

Lastly, the data was divided into an 80:20 ratio for the training and test sets. The training data was further used to create a cross-validation set, distinct from the actual training set.

B. Task 2: Gender classification

The Gender Classification dataset, sourced from Kaggle, contains synthetic data representing various facial attributes used for predicting gender.

This dataset comprises 5,001 records and 7 features. Notably, 5 of these features are categorical, while the remaining 2 ('forehead_width_cm' and 'forehead_height_cm') are continuous. This dataset was chosen to explore how different models handle two distinct types of features. The target variable is categorical, with two possible values: Male and Female, evenly split at 50% each. This balanced distribution differs from the previous dataset, which had slight skewness.

With the exception of the 'long_hair' categorical feature, the distributions for other features are approximately symmetric. Additionally, two variables exhibit balanced data in the lower 75% of values, with a slight skew toward the higher end. These distributions are not entirely uniform, highlighting a more balanced dataset and reducing potential model biases caused by skewness.

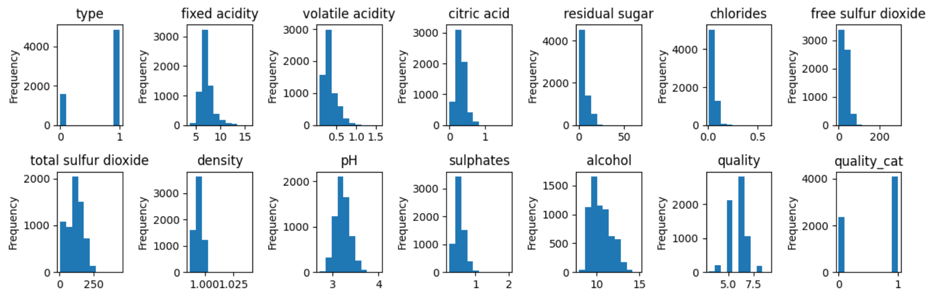


Fig. 1: Dataset 1 feature summary : univariate.

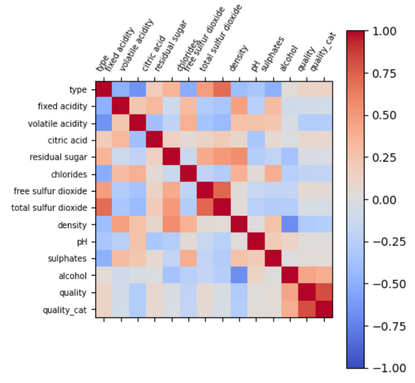


Fig. 2: Dataset 1 feature correlation summary.

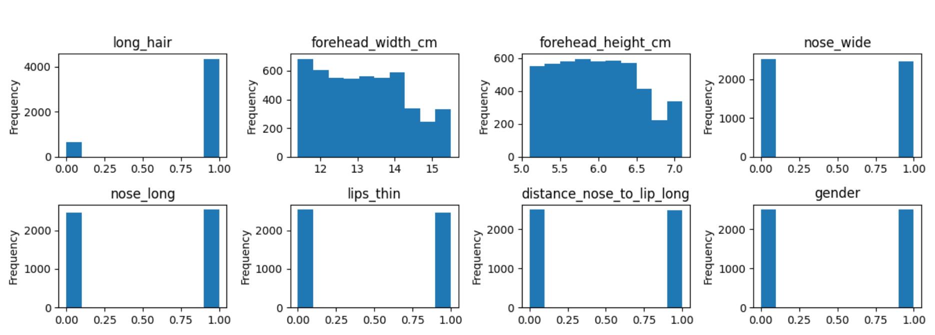


Fig. 3: Dataset 2 feature summary : univariate.

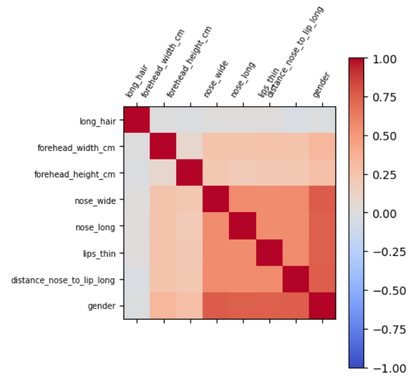


Fig. 4: Dataset 2 feature correlation summary.

Figure 2 illustrates varying degrees of correlation between features, with no clear inverse relationships. Notably, there is a significant positive correlation among features related to the nose, lips, and gender. Similar to the previous dataset, we divided this data into training, testing, and validation sets for improved model evaluation and comprehensibility.

This dataset offers an opportunity to examine how different machine learning models handle both categorical and continuous features. The balanced nature of the target variable and the variety of feature correlations make it an interesting case for model exploration and analysis.

C. *Evaluation Metric: Negative Log Loss*

In our study, we employ Negative Log Loss as our evaluation metric. Log Loss quantifies the disparity between predicted probabilities and actual class labels. Minimizing Log Loss equates to maximizing the likelihood of the true labels given the predicted probabilities.

Log Loss places significant emphasis on the accuracy of individual predictions. It penalizes strongly for confidently incorrect predictions. In simpler terms, when the model is highly confident in an incorrect prediction, Log Loss increases significantly. This sensitivity encourages the model to provide well-calibrated probability estimates, effectively addressing class imbalances within the data.

III. Modeling and Analysis Methodology

Five algorithms were modeled over each dataset, where the nature of underfitting and overfitting was studied, along with the preference and inductive bias of the model across different parameters. The best set of parameters was found using iterative validation (grid search), within a limited scope of parameter sets and ranges. The learning curve of each model for the best-found parameters and a suboptimal variant was further studied. The analysis in the following sections is segmented into three segments:

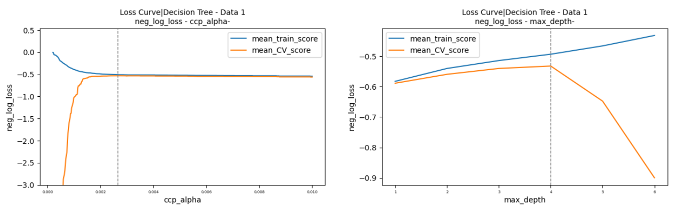
1. Analysis of each algorithm across different datasets. 2. Comparative analysis across algorithms for each dataset. 3. Summary analysis.

IV. Analysis of Each Algorithm Across Different Datasets

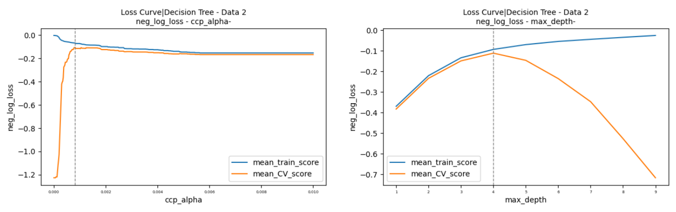
This section provides a comprehensive analysis of the implementation of the five different algorithms mentioned in Section 1. These algorithms are applied to the datasets discussed in Section 3.

A. ***Decision Tree***

The decision tree was trained and optimized through pre-pruning using the "Max_depth" parameter and post-pruning using "ccp_α"



(a) Dataset 1 : a.1 ccp_α and a.2 Max_depth



(b) Dataset 2 b.1 ccp_α and b.2 Max_depth

Fig. 5: Loss curve for DT, NLL vs ccp_α and Max_depth

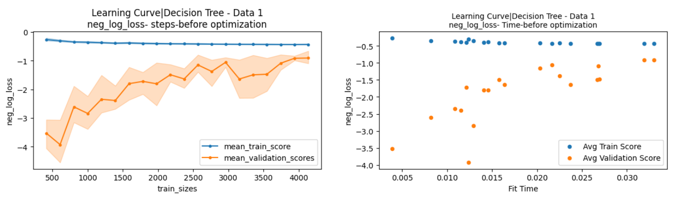
Loss curve

1) Cost Complexity Parameter - ccp_α: - **Dataset 1:** At smaller values the -ve log loss (NLL) for training data is observed very high, where the NLL for validation set was very low, representing low classification accuracy, representing high overfitting, possible reason. However, with increase in value of the parameters, both train and validation NLL move closer reaching a peak NLL value of -0.533 for CV, at 0.00265, reaching highest generalization performance. With further increase in the parameters both train and CV NLL start to gradually decrease together, moving towards underfitting, ccp_α controls the trade-off between complexity and accuracy, for DT, complexity represents number of leaf nodes, At lower ccp_α the pruning is limited resulting in deeper (more complex) tree, fitting the training instances more accurately, however due to the preference bias of DT the lower nodes constitute to very less information gain as compared to as nodes root node, resulting fitting models to training data specific intricacies, creating loss of generalization over unseen instance, as observed in low NLL for CV and high NLL for training data. As most data features for data 1 are continuous in nature, it creates opportunity for creating more defined rules/cuts over the features and a higher chance of overfitting Similarly, at high values of ccp-alpha, more number of nodes trimmed are closer to root node, corresponding to higher information gain, these nodes may represent minimum specificity required for representing part of target function and loosing which results in moving towards more general rule creating nodes, effectively decreasing both NLL training and CV.

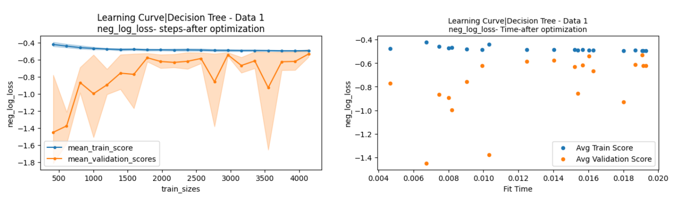
- **Dataset 2:** Like dataset 1, the decision tree model overfits at lower values of ccp_α, reaches its peak of NLL -0.1066 at 0.0008 and then declines to start underfitting as the parameter value increases. Compared to dataset 1, this set shows significantly better classification performance at the peak, this is true for the lowest and highest complexity of the model for data 2 as well, Additionally the peak performance is attained at much lower parameter value for this dataset. Because, less parameters in dataset 2 therefore max depth is less, additionally most features are categorical in nature with only two possible values further limiting the possible cuts which can be made, describing early reaching of the peak.

Additionally as the data is synthetically created, from the correlation charts, more features have higher correlation with the target feature as compared to the dataset 1, which could result in significant increase in information gain. A common observation for both the dataset is after certain value of ccp-alpha, the decent toward underfit is much gradual, meaning these parameter values values were not enough to remove any highly significant node yet, therefore these nodes would be much closed to the root, or could be the root node itself.

2) Parameter 2: Max Depth: - **Dataset 1:** - Higher max-depth represents a more complex tree and similar to ccp_α at lower values, therefore the loss curve for the model, as

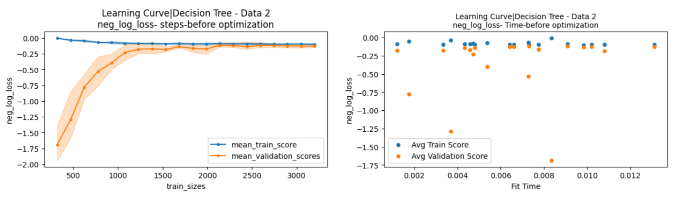


(a) Dataset 1 before optimization : a.1 Train size, a.2 Fit time

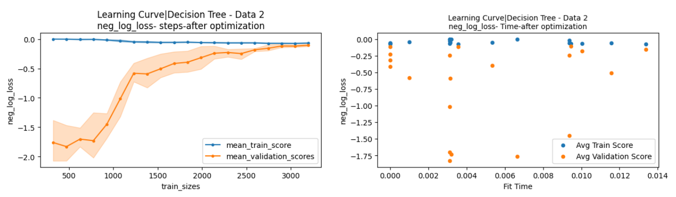


(b) Dataset 1 after optimization : b.1 Train size, b.2 Fit time

Fig. 6: Dataset 1 DT learning curve

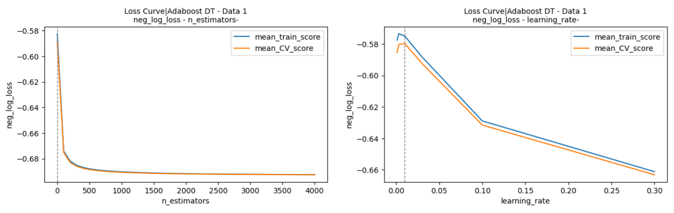


(a) Dataset 2 before optimization : a.1 Train size, a.2 Fit time

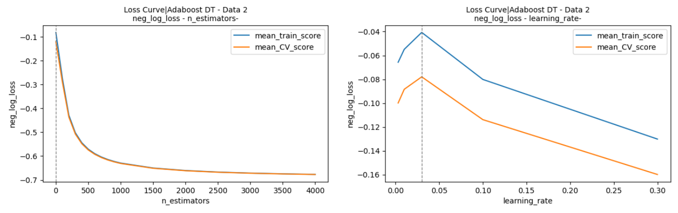


(b) Dataset 2 after optimization : b.1 Train size, b.2 Fit time

Fig. 7: Dataset 2 DT learning curve



(a) Dataset 1 : a.1 #estimators and a.2 LR



(b) Dataset 2 b.1 #estimators and LR

Fig. 8: Loss curve for Adaboost DT, NLL vs #estimators and LR

expected, shows underfit at lower depth values, that is CV and Train NLL are low, which attains a peak NLL of   -0.5319 for cv at depth 4, as postulated for ccp_α, the performance difference between depth 1 and peak performance depth 4 is not much, that is information gain attainted by root node is significantly high compared to the subsequent feature rules, supporting the slow descent to underfitting for higher values of cpp-alpha for dataset 1. This showcase the preference bias of DT. The peak performance for both the parameter are comparable, which relay both pruning mechanisms are equally good for the current classification problem data. At higher values, the model descents to overfit

- **Dataset 2:** DT max-depth parameters behaves similar to Dataset 1, underfit at low values, peak performance at NLL of -0.1116 for cv at depth 4, followed by a descents to overfitting. Compared to Dataset 1, the root node for D2 itself shows a higher performance, additionally subsequent 3 tree depth add significant information gain, unlike D1, resulting in a better peak NLL, which also support, while more features in D2 were correlated with te target, they were less correlated with each other, therefore adding more individual value in classification.

Learning curve

The performance of DT with increase in the number training instances was observed, comparing Best found DT parameters for the data and a suboptimal DT based on the final CV NLL.

- **Dataset 1:** As shown above the performance of suboptimal DT with ccp_α : 0.00001 and depth : 6, a more overfit model, was compared to the best found model with 'ccp_α': 0.0002, 'max_depth': 4, with Max NLL -0.5319, as seen from above in both case the increase in training data decreases the gap in CV and training data performance, i.e more data brings more generalization, however through out the different training size gap in NLL for CV and train data for the suboptimal model is higher because of significant overfit. Additionally the as the number of decision nodes are more with low ccp_α and high tree depth the training time for suboptimal model is higher as well,   0.035 compared to 0.020 of best model for the whole data.

- **Dataset 2:** As shown above the performance of suboptimal DT with 'ccp_α' : 0.002,'max_depth' : 9, a more overfit model, was compared to the best found model with 'ccp_α': 0.0008, 'max_depth': 9, with Max NLL -0.1058. Unlike dataset 1, the suboptimal and best models have the same pre-pruning parameter maxdepth 9, where the ccp_α for best is much lower, which would result in less trimming of latter nodes. As observed in fig [4] and [5], the suboptimal model converges faster to the to its best performance with increase in data, where as the best model which is much deeper/complex shows overfitting with less training data, unlike dataset 1, however shows a better performances with large training set, as discussed for loss curve, nodes till depth 4 showcase high info gain, however when the dataset is less this generalization is not pronounce and model tends to overfit, its only when sufficient data is present the model could find better rules for generalization. Additionally in dataset 1, unlike data 2, the best model converges faster, on the same logic as defined above, the info gain with deeper nodes is not comparatively high from root node, and even with low data the model could find the root node generalization.

B. *Adaboost Decision tree*

Adaboost was applied using significantly underfit decision tree for dataset 1 and dataset 2, which was further optimized using number of estimators and learning rate.

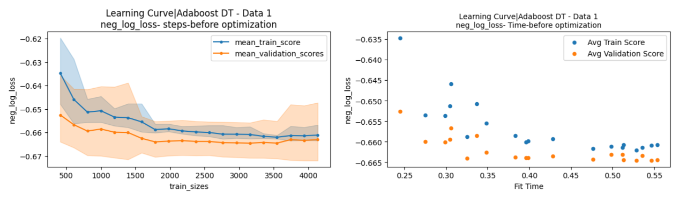
Loss curve

1) Parameter 1 : n estimators: - **Dataset 1:** A base decision tree of max_depth=1 ,ccp_α=0.0001, which would underfit the data, ccp_α is irrelevant as the node is only 1, boosting with learning_rate=0.3 which would align more weight to the best weak learner, however would still not avoid other weak learners. As observed above a single estimator was found to produce the best result, NLL   -0.5885, this corresponds to DT discussion above, till depth 4, the increase in info gain is not very significant, as the number of estimators increases, stumps which are fitted on other variables are given significant weightage, however these features may not add more value, which results in overall model underfitting. This can be seen by the drastic drop in NLL with increase in estimators. (possible high learning rate)

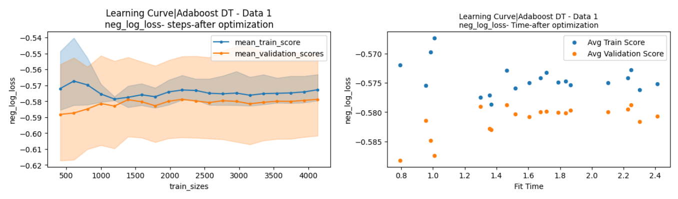
- **Dataset 2:**

Dataset 2 shows a very similar behaviour for n-estimators with highest performance at 1,NLL   -0.1209, for a base DT of max_depth=5,ccp_α=0.00005, which from the DT graphs in previous subsection shows would inherit the best tree with depth 4 however with low ccp_α would not lose information. Unlike the Dataset 1, the drop in performance with increase in estimators is significantly high, this could be because of more categorical features and limited options to create rules, which would result in significantly underfit stumps.

2) Parameter 1 : Learning Rate LR: - **Dataset 1:** The same base DT was used, with 50 estimators, as seen the NLL is high at lower learning rates, giving nearly equal importance to most weak learners, the performances slightly increases with increase in learning rate reaching peak at LL 0.01, NLL

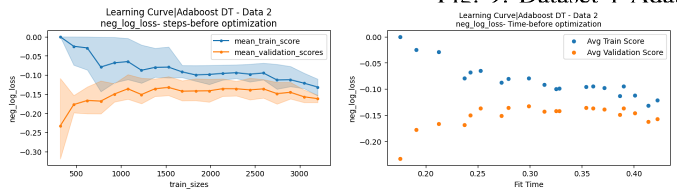


(a) Dataset 1 before optimization : a.1 Train size, a.2 Fit time

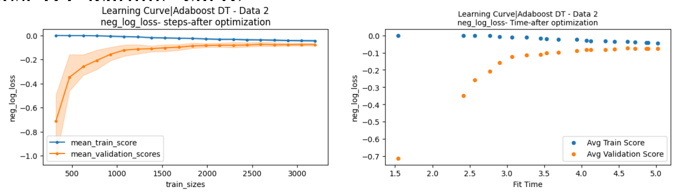


(b) Dataset 1 after optimization : a.1 Train size, a.2 Fit time

Fig. 9: Dataset 1 Adaboost DT learning curve



(a) Dataset 2 before optimization : a.1 Train size, a.2 Fit time



(b) Dataset 2 after optimization : a.1 Train size, a.2 Fit time

Fig. 10: Dataset 2 Adaboost DT learning curve

-0.5797, where it would be able to sufficiently differential best weak learner vs others. With further increase in learning rate the model starts weighing best weak learners more and prior LL starts to underfit. This could explain the single best estimator in 1st parameter study. It can be seen that the performance of boosted DT is lower than single most optimal DT because of this configuration, as even the higher weighted single stumps significantly underfit the data.

- **Dataset 2:** A very similar observation is made for dataset 2, with same base DT as for previous parameter with 50 estimators. High NLL at lower learning rates, peak at LL 0.03, NLL -0.0779, and underfit at higher values because of reliance on only 1 estimator. The difference in the performance at low and high learning rates for both the datasets is comparable for the same number of estimators. However compared to the single DT, the best boosted DT accordingly to LL show improvement, unlike dataset 1, this could be factored to distribution and association of selected features which as they add significant individual information gain unlike dataset 1.

Learning Curve:

- **Dataset 1:** A suboptimal boosted DT with max_depth=1,ccp_α=0.0001 along with 50 estimators and 0.3 learning rate was used for compared over training set size, with the best found boosted DT constrained to same max_depth and ccp_α but 'LR': 0.001, 'n_estimators': 201, with NLL -0.5788 over whole data, as seen from the fig[8],fig[9] the over all difference in NLL from 10% to whole training data is not significant, that is model performance over CV does not significantly change with increase in training data, however the average performance of best model is higher than the suboptimal model. Which associates the impact of lower learning rate accompanied with more estimators as discussed in the individual analysis. This consistency in performance at various train sizes reflect easy identification of generalization rules from the data, which reflects the observation of the high NLL with depth 1 and fast converge for DT. At smaller training set, the NLL of training data is comparatively higher than CV NLL, which indicates relative overfitting, this gap reduces with increase in data. The overall train time of best boosted tree is 5x of suboptimal model, because of processing 4x number of estimators.

- **Dataset 2:** A suboptimal boosted DT with max_depth=5,ccp_α=0.00005 along with 50 estimators and 0.3 learning rate was used for compared over training set size, with the best found boosted DT constrained to same max_depth and ccp_α but 'LR': 0.003, 'n_estimators': 601, with NLL -0.0755 over whole data, as seen from the fig[8],fig[9], similar to dataset 1. Unlike dataset 1, there is significant difference in NLL over train size for both suboptimal and best model. Where both show high overfitting behaviour at low train size, however this is more pronounce for best model because of high estimator count, as large number of trees easily adjust to intricacies of the training data. Suboptimal model reaches peak faster because of higher learning rate and low number of estimators. However this change is much more gradual in the best model, as there is more scope for larger number of trees to generalize better over data with smaller learning rate. Following dataset 1, the training time here as well for best model is high, 12x of suboptimal model over whole data, because of large number of estimators.

C. *K nearest neighbours - KNN*

KNN was trained and optimized controlling K (number of neighbours) and p (power of distance) over both the datasets, as detailed below. For this model, no prior knowledge of feature weightage is assumed and therefore to minimize the effect of curve of input features were standardized.

Loss Curve

1) Parameter 1: K (# neighbours): - **Dataset 1:** With base p of 2, using Euclidian distance and uniform weightage, fig[10] shows, estimating class based on less number of neighbours does not generalize well, resulting in high overfit, however, this quickly resolves in only a small increase in #neighbours for estimation, reaching a peak NLL -0.5268 at K=200. with further increase in K, the generalization increases however this also results in loss of local specificity, directing towards underfitting.

The slow descent into underfitting on the data support the inductive bias for KNN that instances of similar class have similar features, as with the increase in the k does not significantly decrease performance, therefore features of different class must be clustered separately with defined global decision boundaries

- **Dataset 2:** Same base model as dataset 1 was used for the study, which showcase as a similar trend of overfitting at low K reaching peak NLL -0.0907 at K=200 and moving towards underfitting as K increases further. However, comparatively the drift towards underfitting is more pronounced unlike dataset 1, supporting there are model local decision boundaries/ clusters compared to data 1. This is further aggregated by the limited number of possible feature values for 6 out of 7 Boolean variables. Additionally, localized impacted is further supported by low difference in lowest NLL K and peak NLL K, compared to dataset 1.

2) Parameter 2: P (power of distance): With a base K = 5 the power of distance was varied for KNN over both the dataset, a small K was taken into consideration towards preference bias of KNN towards localized commonalities.

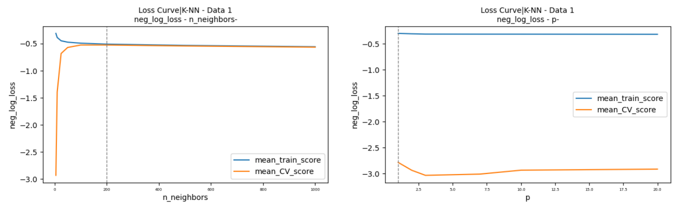
- **Dataset 1:** As shown in fig [11 b], at small k, the overall CV performance does not change drastically, however KNN attains best CV performance at p= 1, NLL -0.0907, whereas with increase in P to 2 and 3 NLL slightly decrease. As observed form Corelation chart and DT model, only one of the features showed significant association with the target variable, while KNN considers all the features for decision making, at p = 2,3 the difference in distances of irrelevant feature if high, becomes more pronounce, unlick at p =1. With further increase in p the accuracy gradually starts increasing, this could be the result of filtering effect of outliers at high p.

- **Dataset 2:** As fig [12b] shoes , the impact of the p for data 2 is similar over, i.e the change is performance is not immense. The similarity follows for p =1 and 2 as well. However performance jumps back at =3 and gradually decrease from there. Unlike dataset 1, 5 out of 7 features in dataset 2 are Boolean and therefore the difference values at any power they take up is 1 or 0, which is not the case for the 2 continuous features. As seen from fig [2] , these features are only weekly correlated with target, however the impact of the these features may continue to increase at higher powers, resulting in lower accuracy.

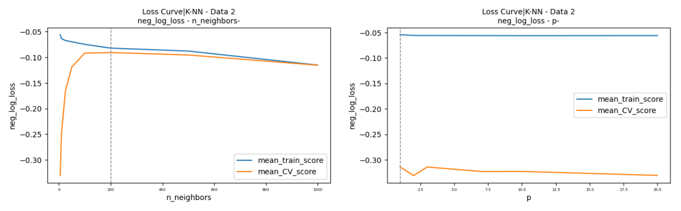
Learning curve:

A suboptimal KNN with 'k: 5,'p' : 10 was compared over training set size, with the best found KNN for both the datasets.

- **Dataset 1:** Best KNN model with 'k: 100, 'p': 3 , with NLL -0.5238 was found over whole data. As seen in the figure [14] the suboptimal model CV NLL decreases with increase in training data. At smaller as training size the instances are sparse therefore even with lower k and high p, the model is able to generalize well, however with increase in the training data the estimation is more localized resulting in high overfitting. However with the optimal model, where k is very large, the model is too generalized resulting in underfitting as seen in the fig [15], which resolves with the increase in training data. Additionally there is a drastic difference in the CV NLL of suboptimal and best model, at all training sizes, tying back to parameter analysis of k.

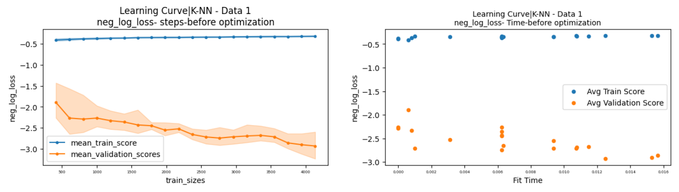


(a) Dataset 1

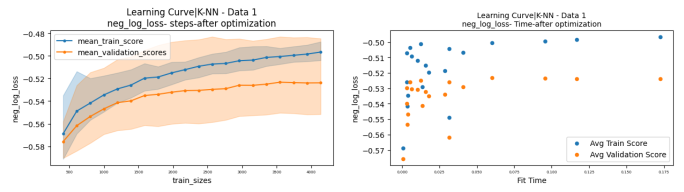


(b) Dataset 2

Fig. 11: Loss curve for KNN, NLL vs K and p

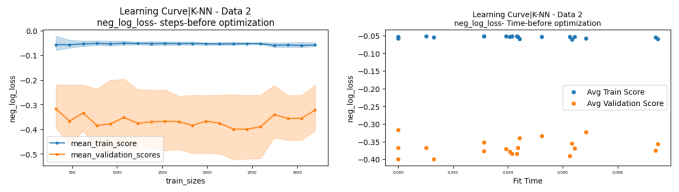


(a) Dataset 1 before optimization : a.1 Train size, a.2 Fit time

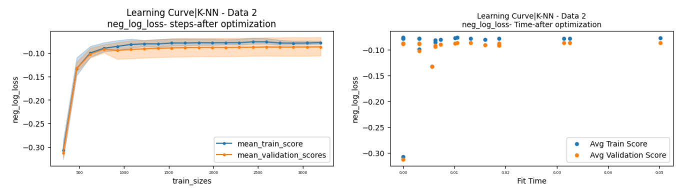


(b) Dataset 1 before optimization : a.1 Train size, a.2 Fit time

Fig. 12: Dataset 1 KNN learning curve

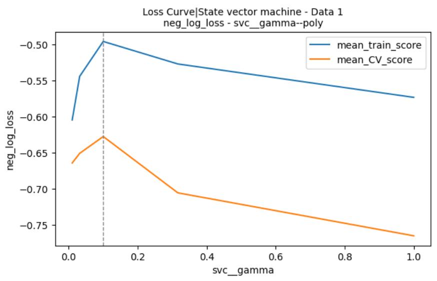


(a) Dataset 2 before optimization : a.1 Train size, a.2 Fit time

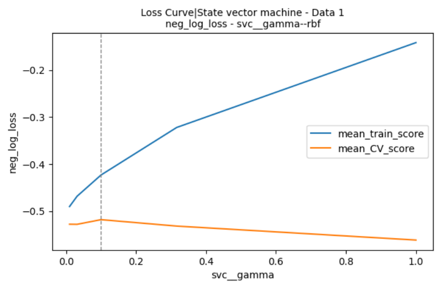


(b) Dataset 2 after optimization : a.1 Train size, a.2 Fit time

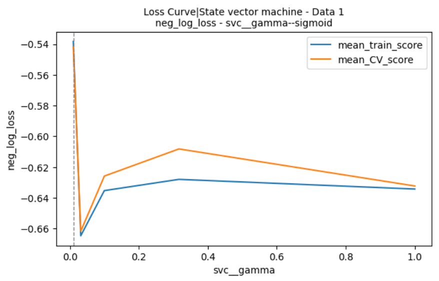
Fig. 13: Dataset 2 KNN learning curve



(a) kernel : Poly

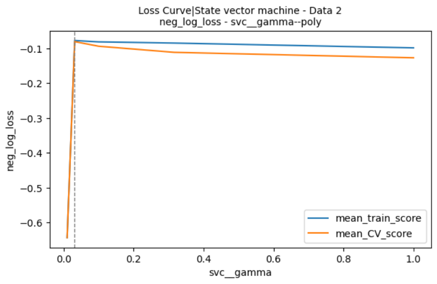


(b) kernel :RBF

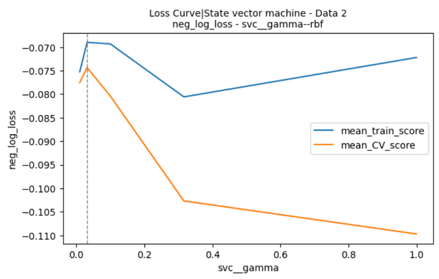


(c) kernel : Sigmoid

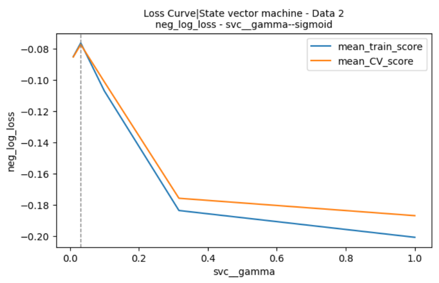
Fig. 14: Loss curve for SVC, Dataset 1 against γ



(a) kernel : Poly



(b) kernel :RBF



(c) kernel : Sigmoid

Fig. 15: Loss curve for SVC, Dataset 2 against γ

- **Dataset 2:** Best KNN model with 'k' : 200, 'p': 1 with NLL -0.0868 was found over whole data. Suboptimal model does not as show any change in the CV NLL over training size, this is because of more Boolean features, limiting the size of concept space, therefore even a relatively smaller dataset may represent the population. There are only two continuous standardised features which could drive the localization effect with increase in data, however this is not visible with p=2 where boolean feature have more weightage due to majority and standardization at a small k of 5. Fig [17] shows a similar best KNN underfitting behaviour as dataset 1 at smaller training size and high k, however this behaviour normalizes faster in dataset 2, because of only 2 continuous variables. Thereafter like suboptimal knn the best model CV NLL does not change much with increase training size, following the same reason. This is unlike dataset 1, where the CV NLL shows significant change throughout the training size, due to more continuous variables, though decelerating at larger sets. The performance of best model is significantly better than suboptimal model, because of better generalization with high k.

D. Support vector machines

SVM classifier was trained and optimized controlling 2 parameters, i.e kernel function and gamma (γ). As SVM is sensitivity to range of feature values, data was scaled before training.

Loss Curve.

Due to categorical nature of kernel function, 3 different loss curve corresponding to kernel functions ['poly', 'rbf' , 'sigmoid'] were plotted for comparison with different values of gamma for each dataset. For all models beside overall best used in studing the learning curve, regularization term C is kept to 0 to independently study γ.

*1) **Dataset 1:*** 3 kernels for dataset 1

- **Polynomial Kernel:**

As observed in fig[19 a] at the model accuracy increase while moving away from $\gamma = 0$ and reaches peak NLL of -0.6272 at $\gamma = 0.1$ for both CV and train data, with the degree of 3. This is because even if degree is high, at lower γ (near 0), impact of individual support vectors are less in calculating decision boundary w, which results in underfitting the training data. While here, similarity is a cubic function of the dot product of training instances, this underfitting may also diminish any false classification due to unrelated features. With small increase γ, the impact of individual datapoints increases creating lesser flexible boundaries, resulting in better segregation of training instances. However with further increase in γ a significant drop in CV NLL is visible, which is a result of decision boundary defined too close to training examples, overfitting in the 3rd dimensional projection of the instances, which do not generalize well, additionally the training NLL is also seen to decrease along with CV, this indicates the inability of segregation of data in the projected space, which could be due to the highly skewed nature of continuous features with a strong mix between features values of classes.

- **RBF kernel:**

Similar to polynomial kernel, the model accuracy of rbf kernel follow the same pattern of underfitting at low gamma, peak NLL of -0.5183 at $\gamma = 0.1$, best among all 3 kernels, and overfitting at higher gamma values, however is less pronounced, i.e even with narrower and more localized decision boundaries resulting from high c, this could be accounted to two reasons, 1. impact of exponential decay of similarity between instances with distance, creating more localized boundaries to resolve the issue of skewed data feature mix, 2. The projection of data in infinite dimensions, i.e while the data may not be as easy to separate at lower dimensions, at infinite dimensions plains can be found to separate the data, where impact of high γ overfitting could be normalized, the same two reasons support high growth in train NLL as well.

- Sigmoid kernel:

Compared to both Poly and RBF kernel the difference in NLL is small for both training and CV set. The peak NLL of -0.5418 at γ = 0.01 for both CV and train data. The sigmoid function uses tanh to transform the linear projection of on instance over other, which scales the relations on -1 and 1, the shape of the decision boundary is limited, unlike linear or polynomial kernel. Additionally, it is possible that the data is linearly separable, which may explain the low difference between the performance at low and high γ. Which would also direct towards possible overfit of polynomial kernel with degree 3.

2) ***Dataset 2:*** 3 kernels

- Polynomial Kernel:

Similar to Dataset 1, model accuracy drastically increases moving away from γ = 0 and reaches peak NLL of -0.0805 at γ = 0.0316 for both CV and train data, with the degree of 3. Followed by a gradual and slow decreased in the CV NLL but not enough in train data, representing slight overfitting. In addition of the explanation in dataset 1, the drastic initial NLL increase and slowed overfit could be explained by the presence of more Boolean features, that is the number of possible divisions is limited unlike dataset 1, which means decisions boundaries which can classify instances more easily and generalization could be reached faster, therefore with only a small increase in γ a complex enough decision boundary could be identified to segment the classes. Similarly, even if the γ is further increased , although it fits training examples more closely it does not diminish generalization capabilities too much, i.e the data is inherently easy to separate.

- RBF Kernel:

Like Dataset 1, model accuracy at low γ values is high and slightly increases moving away from γ = 0 and reaches peak NLL of -0.0743 at γ = 0.0316 for both CV and train data, highest among 3 kernels, and transition to overfitting behaviour. However the difference in the NLL for train and CV, throughout the experimented range is very small, therefore RBF even with large γ could segregate classes easily with limited dimensional projections. The performance compared to Polynomial kernel was observed to be better at low and similar to at medium and higher γ, similar to data 1.

- Sigmoid Kernel: KNN for sigmoid kernel showcase high NLL throughout γ range compared to dataset 1, however NLL for smaller γ show higher accuracy, reaching a peak NLL of -0.0772 at γ = 0.0316 for both CV and train data, and decreasing monotonically for higher γ for both train and CV sets. Although the range of change across range is small, it is larger than Dataset 1 observations.

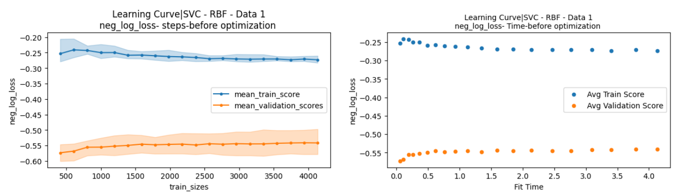
Learning curve:

The best performance SVC was identified by picking the best model from the parameter analysis based on kernel and γ. This model was further optimized using regularization parameter C for each dataset. A suboptimal SVC was compared over training set size, the best found SVC.

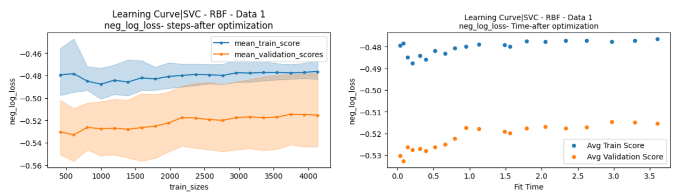
- Dataset 1 :

Fig [20 a] show the learning curve of suboptimal SVC with γ :0.5,C:0.5, kernel : rbf, compared with Fig[20 b] showing the best SVC at NLL -0.5154, with γ: 0.1, C: 0.1, kernel: rbf. As shown, Train and CV NLL for both suboptimal and best SVC does not change much with as increase in training data. The suboptimal model with high γ and C results in as more overfitted model compared, resulting in large difference in Train and CV NLL which does not minimize drastically with more data, Additionally the best model with small γ and C is more generalized SVC resulting in low and maintained Train NLL, the CV NLL in both the cases are very close throughout, directing towards more impact over segregation of instances with infinite dimension projection, this may points towards equal difficulty in classification even at different bias and variance trade-offs due to highly mixed and skewed features, as seen in the distribution. Additionally this also direct towards easier representation of data even at small samples.

- Dataset 2

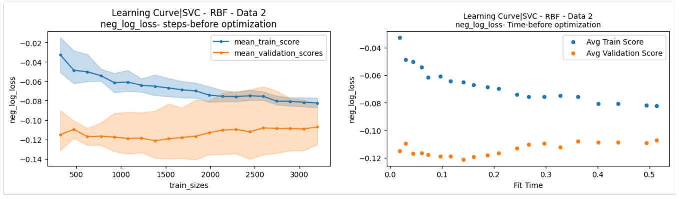


(a) Dataset 1 before optimization : a.1 Train size, a.2 Fit time

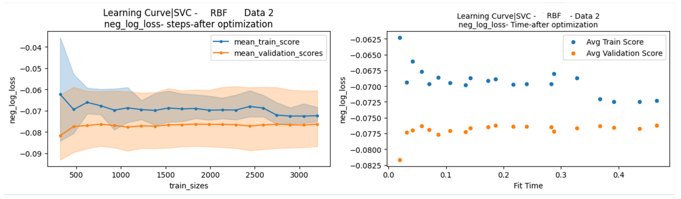


(b) Dataset 1 after optimization : a.1 Train size, a.2 Fit time

Fig. 16: Dataset 1 SVC learning curve



(a) Dataset 2 before optimization : a.1 Train size, a.2 Fit time



(b) Dataset 2 before optimization : a.1 Train size, a.2 Fit time

Fig. 17: Dataset 2 SVC learning curve

the data set suboptimal SVC (γ :0.5,C:0.8, kernel : rbf) and Best SVC (γ : 0.0316,C:1, kernel : rbf, NLL: -0.0741), both show decrease in overfitting with increase in data, only with decrease in train NLL unlike dataset 1, however, like dataset 1, the CV NLL does not value significantly, following the same argument as dataset 1. Best CV NLL compared to suboptimal SVC remains high, where low γ and C creating smoother boundaries, allows the model to avoid highly complex shapes, which may only reduce generalization to a curtain degree because of limited features values. These limited feature values also allow easier finding of separation planes, resulting in high performance in both suboptimal and best model compared to dataset 1.

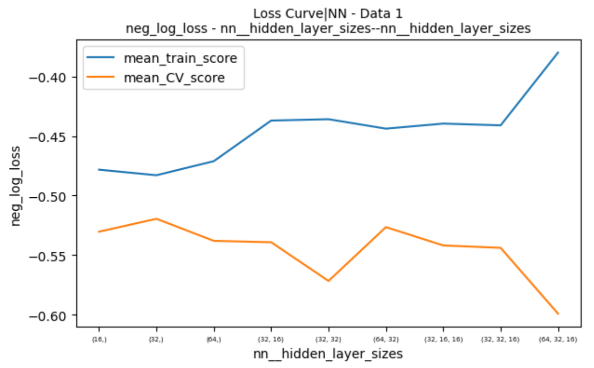
E. Neural Networks NN

Unlike previous models, multiple parameters including, hidden layer depth and size, learning rate, batch size and momentum, we studied separately for NN over both the datasets, as they all govern the model preferences and restrictions in estimating the target functions. The Loss Curve details 2 of the parameters, hidden layer depth and size, along with batch size in detail. A base NN model, with two hidden layers , 32-16 with Relu activation was used with initial learning rate 0.001, batch size 24, and momentum 0, using Adam optimizer, for both datasets individually was used varying above mentioned parameters, as detailed below. This base model was identified post iterative study of the learning curve and changing parameters. Change in momentum did not result in major performance changes, and therefore not covered in analysis. In addition to Learning curve over train size, learning curve over training epoch is also covered, only for best NN models. Lastly as the weight updates are sensitive to feature ranges, all features are scaled before training.

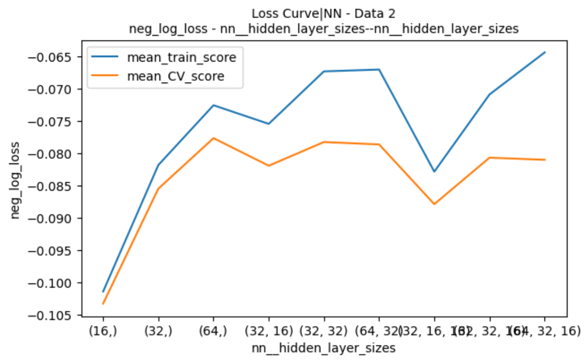
Loss curve:

1) Parameter 1: Hidden layer shape:: 9 possible hidden layer shapes where compared against, with indicative increase in complexity, 3 single HL (16,), (32,), (64,), 3 two layers HL (32,16,), (32,32,), (64,32,), and 3 3 layer HL (32,16,16,), (32,32,16,), (64,32,16,) . While the a clear order of increase in complexity cannot be accurately quoted, it can be mentioned that deeper layers make use of intermediate features and therefore are more complex, if the number of nodes between shapes are same.

- **Dataset 1:** Fig [21 a] shows a general trends, with increase in depth as well as breath of the hidden layers, gap between the CV and train NLL increases, This cause is majorly driven by more complex network being able to closely represent training data. However the power of generalization does not monotonically change with this. As seen shallow single layered networks, only capture easily available patterns, on based on the input data, and do not create too many derived intricate features, creating simple decision regions without overfitting the training data, explaining the low difference between the CV and train NLL. For our case these shallow network also have the highest NLL, indicating inherent data separability. The best NN of HL (32,) with NLL -0.5194 indicates, a minor



(a) Dataset 1

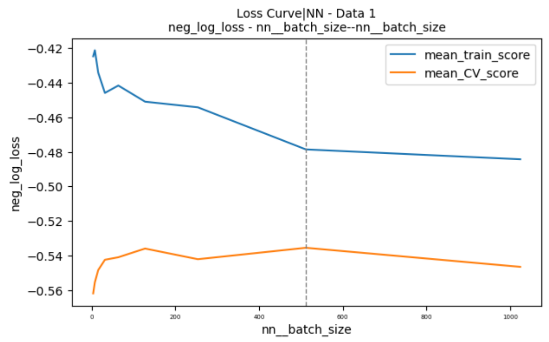


(b) Dataset 2

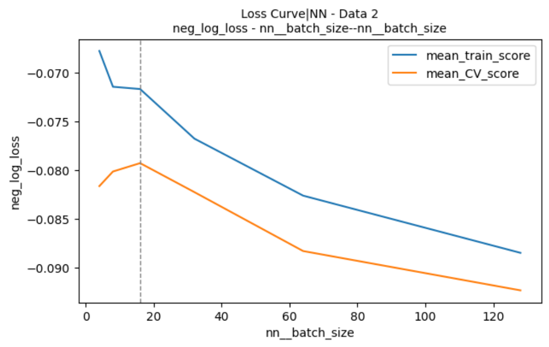
Fig. 18: Loss curve for NN, NLL vs Hidden layer size

patterns of low NLL at narrower network, because of inability to represent slightly complex relationships, and also at wider network, where more intricate identified patterns overfit the training data. This kind of distinction becomes slightly difficult when the layers are increased. More number of layers result in creation of derived features, which can be used to identify important relationships with target. (32,16) and (64,32) both perform better than (32,32), we can hypothesize that (32,16) compared to (32,32) being simpler create mode generalized classifier, at the same time (64,32) could find more valid derived features and patterns closer to the true target concept, however this may require further validation. The overfitting is more sever at deepest and widest networks. This also is an impact of continuous and skewed data features with a potential to create complex derived features.

- **Dataset 2** Fig [21 b], follows many of the explanations shared for data 1, however the nature of the highly categorical data, creates differences in the performance. As seen at narrowest and shallowest network, intricate boundaries or derived features are note created, which in this case results in relatively strong underfitting, Although NLL for dataset 2 is very high compared to the best model for data 1, which shows relatively easy class segregations. This is resolved with increasing the width, as seen from corr. Chart features in data 2 show strong association with target, and therefore this increase is enough to identify these linear relationships, where the best NN with HL (64,) with NLL -0.0776 is found. Further increase increase in the complexity via width and depth does not significantly change CV NLL but slightly represent training data better.



(a) Dataset 1



(b) Dataset 2

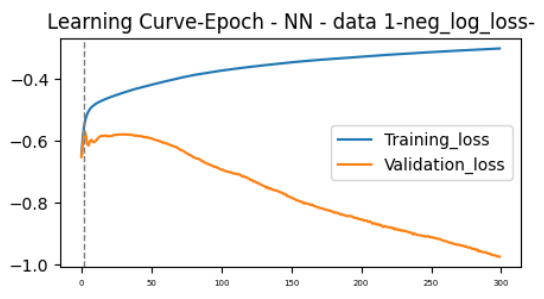
Fig. 19: Loss curve for NN, NLL vs Batch size size

This is because good relationships could be found in the first layers itself, which may just pass values from one layer to next without major transformations. However this hypothesis was not validated. Additionally more categorical features limit the creation of derived features, therefore the deeper hidden layers may not add value after a curtain important feature derivations are reached.

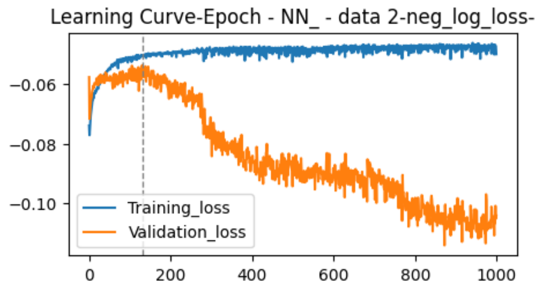
2) Parameter 2 Batch size:: Depending on the kind of features, their distributions and association of data the generalization power of the batch size varies, therefore different batch sizes were experimented for the datasets.

- **Dataset 1:** Fig [22 a] shows at smaller batch sizes the model tends to overfit, this is because these result in more number of weight updates with high variance, which could possibly result in larger weights or even reach local minima with suits training data better, which fits the train data better creating more complex decision regions. This also is against the preference bias of simpler networks. This effect reduce with increase in batch size, therefore less w updates. Reaching a peak generalization at 512, NLL -0.5355, a further increase in batch size, can cause high normalization wight changes which slows down the convergence and with insufficient training data or epochs, this may result in underfitting. For our network this is less pronounce, additionally, the CV NLL does not change much throughout Batch size range, possibly due to localized separation of the classes or high mix of classes as seen in the other algorithms.

- **Dataset 2:** The impact of underfitting is more pronounce for dataset 2, unlike data 1, at relatively smaller values. Although the patterns remains same of underfitting at smaller values, peak at 16 with NLL -0.0792, and shift towards



(a) Dataset 1



(b) Dataset 2

Fig. 20: Learning Curve : NN , number of iterations

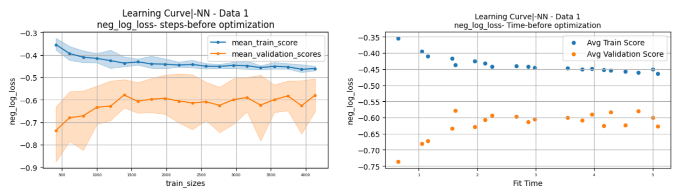
underfitting. However, the over-all change in performance is not significant, meaning the segregation function can easily be identified, irrespective to batch size. This is the direct result of limited categorical and balanced variables with significant correlation with target function which also limits the amount of variance which could be introduced with change in batch size.

Learning Curve – epochs

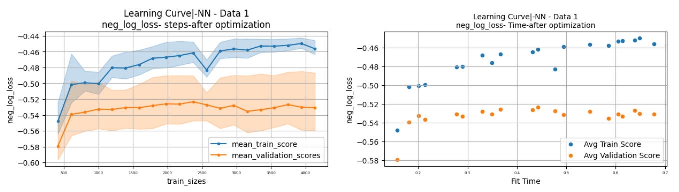
Best model was identified for each dataset by performing grid search over all the variables at the same time, in addition with early stopping, and 5 validation sets. A single run of learning curve was created to understand the epoch, it is expected that this may only be indicative of general performance of the model and may vary in case we perform and average this test over different cross validation sets. For this example, the validation NLL was calculated over test split as explained in the data section.

- **Dataset 1:** Fig [23 a] shows, at less iteration the performance over train and test is similar, for the best NN with 'batch_size: 512, hidden_layer_sizes: (32, 32), LR_init: 0.00215, momentum: 0, additionally NLL for both these sets is slightly low, which also indicates slight underfitting, however with only 1 more iteration the test NLL peaks to -0.5676, depicting easier identification a model which generalize well, with further iterations, the model fits the training data closely and subsequently follows a slow increase in train NLL, while the test NLL drastically decrease, this indicates fitting to noisy patterns in the training data, coupled with skewed continuous features, adding high variance and very complex and tight decision boundries.

- **Dataset 2** Although dataset 2 follows a similar pattern, with, 'batch_size: 16, hidden_layer_sizes: (64,), LR_init: 0.03162, momentum:1e-04, with increase in epochs, underfit at less iterations, peak at 130 epochs, NLL -0.0536 and a

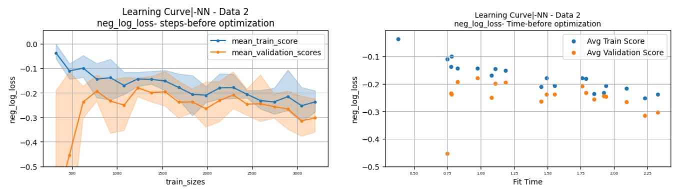


(a) Dataset 1 before optimization : a.1 Train size, a.2 Fit time

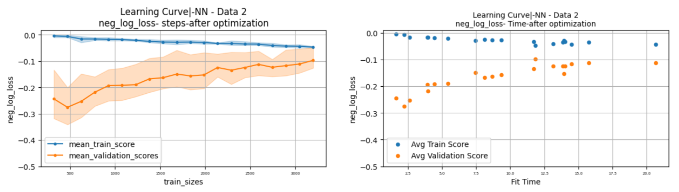


(b) Dataset 1 after optimization : a.1 Train size, a.2 Fit time

Fig. 21: Dataset 1 Learning curve :NN



(a) Dataset 2 before optimization : a.1 Train size, a.2 Fit time



(b) Dataset 2 after optimization : a.1 Train size, a.2 Fit time

Fig. 22: Dataset 2 Learning curve :NN

overfitting phase, the difference change in the performance in not much even at 1000 epochs, this is because of better representation of the target concept in the training data, only two continuous variables, balanced distributions and inherent class segregation, therefore the outliers and the noise in the training data is comparatively low.

Learning Curve – training size The best found NN for each dataset was compared against a suboptimal NN model, over different train sizes and 5 cross validations, similar to other 4 algorithms.

- Dataset 1

Fig [24 a] show the learning curve of suboptimal NN with 'batch_size': 16, 'hidden_layer_sizes': (16,), 'LR_init': 0.03, 'momentum': 1e-04, compared with Fig[24 b] showing the best NN as specified in the Learning Curve – epochs. Both models were limited to 25 max iterations better comparison. Overall there is a drastic change in performance of suboptimal NN CV NLL, where it remains more stable for the relatively stable for the best NN. This also indicates the training data is representive enough at smaller train size, however, the simpler suboptim network cannot identify these patterns very well, and due to smaller batch size results in overfitting, where best model shows slight indication of underfitting because of very large batch size for the training set and lower learning rate. Which also helps the best NN learn more patterns, given a deeper and wider structure, with slight increase in data, eventually fitting better with the training set as well. More data for suboptimal model also results in better generalization, and decrease in overfitting, resulting increased CV NLL and decreased Train NLL. Large number of updates for suboptimal model also result in higher computational time.

- Dataset 2:

Fig [25 a] show the learning curve of suboptimal NN with 'batch_size': 16, 'hidden_layer_sizes': (16,), 'LR_init': 0.5, 'momentum': 1e-04, compared with Fig[25 b] showing the best NN as specified in the Learning Curve – epochs. The major difference between the both the networks Hidden layer width and learning rate. With increase in the train size both Train and CV NLL for both models move towards each other, in suboptimal this process is more drastic, i.e while with less training example both model overfit, high LR for suboptim result in bigger changes in weights and faster convergence, this also means with further increase in training examples the model is effected by training noises more and also may fail to converge into global minima. Along with narrow hidden layer, it may fail to identify better segregating features, which could result in underfit. On the contrary, smaller learning rate and wider network lets the best model find more intricate features with gradual convergence with small updates. Although this increases training time by 10x

V. Comparison Summary

This Sections compares inter data performance and also points to overall differences in learning algorithms between the two datasets.

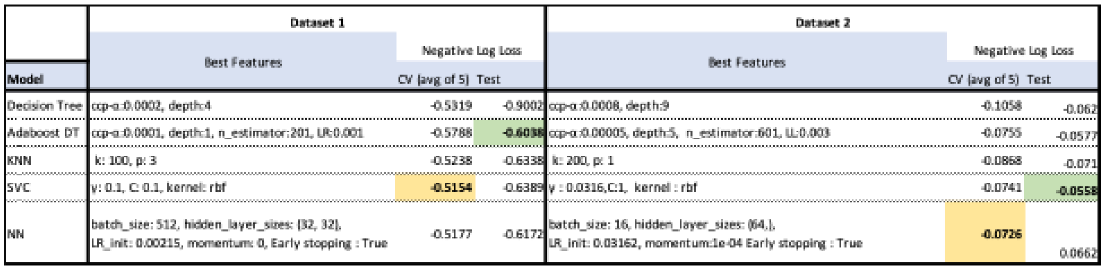
A. Table

CV accuracies were noted while identifying the best learners per data and algorithms, which were further used for calculating accuracies for test sets.

B. Intra-data

:

1) Dataset 1: While SVC creates the best average estimator with highest CV NLL of -0.5154, the difference between performance of algorithms is not very significant, that is all models performance relatively poor on the data with very low NLL, this points depicts how unbalanced and skewed distributions, along with irrelevant features fail identify a good estimate of target concept. Additionally, models with similar or best CV accuracies may not equally generalize test data, that is it is possible to overfit CV data. The worst performing model over CV, Adaboost, generalized better than all other additionally over test set, NLL -0.6038, which also indicates the how it decrease possible overfit over DT, which performs the worst over test set.



	Dataset 1			Dataset 2		
	Best Features	Negative Log Loss		Best Features	Negative Log Loss	
Model		CV (avg of 5)	Test		CV (avg of 5)	Test
Decision Tree	ccp-α:0.0002, depth:4	-0.5319	-0.9002	ccp-α:0.0008, depth:9	-0.1058	-0.062
Adaboost DT	ccp-α:0.0001, depth:1, n_estimator:201, LR:0.001	-0.5788	-0.6038	ccp-α:0.00005, depth:5, n_estimator:601, LL:0.003	-0.0755	-0.0577
KNN	k: 100, p: 3	-0.5238	-0.6338	k: 200, p: 1	-0.0868	-0.071
SVC	γ: 0.1, C: 0.1, kernel: rbf	-0.5154	-0.6389	γ : 0.0316,C:1, kernel : rbf	-0.0741	-0.0558
NN	batch_size: 512, hidden_layer_sizes: (32, 32), LR_init: 0.00215, momentum: 0, Early stopping : True	-0.5177	-0.6172	batch_size: 16, hidden_layer_sizes: (64,), LR_init: 0.03162, momentum:1e-04 Early stopping : True	-0.0726	0.0662

Fig. 23: Overall performance summary

2) Dataset 2: All estimators perform relatively well on CV as well as test data, with best being NN with CV NLL -0.0726, and worst DT at CV NLL -0.1058, which indicates much complex feature can could segregate data better. This also indicates, when the data is well balanced, show associated with target function, and creates limited concept space, then any method when tuned can segregate instances well. Following the observation from dataset 1, when the models have similar training or cv performances, the expected performance on unseen data is more driven by characteristics of data, even though DT had lower CV NLL, it performed better than NN on test. And SVC performed better on test than any other model, while performing comparable to NN on CV, that might indicate support of inherent separate of instances based on features, which NN might not have captured so well. Additionally the performance of models across test was found better than CV. Given the good overall performance of the models and sufficient training data, this may indicate there could be specific CV sets with high noise. Therefore it would be beneficial to validate the performance of all CV sets.

C. Performance comparison across Data

From the above analysis the importance of balanced and relevant feature set is visible, therefore it is imperative to analyse the data before training a model. This fact is supported by all the algorithms, which performed better on dataset 2. Additionally, it can be seen that NN and SVC could identify patterns regardless description of data, using higher dimensions or derived features, while giving acceptable performance on unseen test data.

VI. Conclusion

This Study explores the working of different algorithms on different type of data, highlighting key driving factors of each. A major insight from showcase the importance of description of dataset itself before application of any algorithm, which can limit performance throughout the spectrum of algorithms. Additionally the study highlights the adaptability of SVM and Neural networks to different kind of data distributions and prediction problems.

- Use either SI (MKS) or CGS as primary units. (SI units are encouraged.) English units may be used as secondary units (in parentheses). An exception would be the use of English units as identifiers in trade, such as "3.5-inch disk drive".

A. Equations

Number equations consecutively. To make your equations more compact, you may use the solidus (/), the exp function, or appropriate exponents. Italicize Roman symbols for quantities and variables, but not Greek symbols. Use a long dash rather than a hyphen for a minus sign. Punctuate equations with commas or periods when they are part of a sentence, as in:

$$a + b = \gamma \tag{1}$$

Be sure that the symbols in your equation have been defined before or immediately following the equation. Use "(1)", not "Eq. (1)" or "equation (1)", except at the beginning of a sentence: "Equation (1) is . . ."

B. LaTeX-Specific Advice

Please use "soft" (e.g., `\eqref{Eq}`) cross references instead of "hard" references (e.g., `(1)`). That will make it possible to combine sections, add equations, or change the order of figures or citations without having to go through the file line by line.

Please don't use the `{eqnarray}` equation environment. Use `{align}` or `{IEEEeqnarray}` instead. The `{eqnarray}` environment leaves unsightly spaces around relation symbols.

Please note that the `{subequations}` environment in LaTeX will increment the main equation counter even when there are no equation numbers displayed. If you forget that, you might write an article in which the equation numbers skip from (17) to (20), causing the copy editors to wonder if you've discovered a new method of counting.

BibTeX does not work by magic. It doesn't get the bibliographic data from thin air but from .bib files. If you use BibTeX to produce a bibliography you must send the .bib files.

LaTeX can't read your mind. If you assign the same label to a subsubsection and a table, you might find that Table I has been cross referenced as Table IV-B3.

LaTeX does not have precognitive abilities. If you put a `\label` command before the command that updates the counter it's supposed to be using, the label will pick up the last counter to be cross referenced instead. In particular, a `\label` command should not go before the caption of a figure or a table.

Do not use `\nonumber` inside the `{array}` environment. It will not stop equation numbers inside `{array}` (there won't be any anyway) and it might stop a wanted equation number in the surrounding equation.

C. Some Common Mistakes

- The word “data” is plural, not singular.
- The subscript for the permeability of vacuum μ_0, and other common scientific constants, is zero with subscript formatting, not a lowercase letter “o”.
- In American English, commas, semicolons, periods, question and exclamation marks are located within quotation marks only when a complete thought or name is cited, such as a title or full quotation. When quotation marks are used, instead of a bold or italic typeface, to highlight a word or phrase, punctuation should appear outside of the quotation marks. A parenthetical phrase or statement at the end of a sentence is punctuated outside of the closing parenthesis (like this). (A parenthetical sentence is punctuated within the parentheses.)
- A graph within a graph is an “inset”, not an “insert”. The word alternatively is preferred to the word “alternately” (unless you really mean something that alternates).
- Do not use the word “essentially” to mean “approximately” or “effectively”.
- In your paper title, if the words “that uses” can accurately replace the word “using”, capitalize the “u”; if not, keep using lower-cased.
- Be aware of the different meanings of the homophones “affect” and “effect”, “complement” and “compliment”, “discreet” and “discrete”, “principal” and “principle”.
- Do not confuse “imply” and “infer”.
- The prefix “non” is not a word; it should be joined to the word it modifies, usually without a hyphen.
- There is no period after the “et” in the Latin abbreviation “et al.”.
- The abbreviation “i.e.” means “that is”, and the abbreviation “e.g.” means “for example”.

An excellent style manual for science writers is [7].

D. Authors and Affiliations

The class file is designed for, but not limited to, six authors. A minimum of one author is required for all conference articles. Author names should be listed starting from left to right and then moving down to the next line. This is the author sequence that will be used in future citations and by indexing services. Names should not be listed in columns nor group by affiliation. Please keep your affiliations as succinct as possible (for example, do not differentiate among departments of the same organization).

E. Identify the Headings

Headings, or heads, are organizational devices that guide the reader through your paper. There are two types: component heads and text heads.

Component heads identify the different components of your paper and are not topically subordinate to each other. Examples include Acknowledgments and References and, for these, the correct style to use is “Heading 5”. Use “figure caption” for your Figure captions, and “table head” for your table title. Run-in heads, such as “Abstract”, will require you to apply a style (in this case, italic) in addition to the style provided by the drop down menu to differentiate the head from the text.

Text heads organize the topics on a relational, hierarchical basis. For example, the paper title is the primary text head because all subsequent material relates and elaborates on this one topic. If there are two or more sub-topics, the next level head (uppercase Roman numerals) should be used and, conversely, if there are not at least two sub-topics, then no subheads should be introduced.

F. Figures and Tables

a) Positioning Figures and Tables: Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation “Fig. 24”, even at the beginning of a sentence.

TABLE I: Table Type Styles

Table Head	**Table Column Head**		
	Table column subhead	***Subhead***	***Subhead***
copy	More table copy[a]		

[a]Sample of a Table footnote.

Fig. 24: Example of a figure caption.

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity “Magnetization”, or “Magnetization, M”, not just “M”. If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write “Magnetization (A/m)” or “Magnetization {A[m(1)]}”, not just “A/m”. Do not label axes with a ratio of quantities and units. For example, write “Temperature (K)”, not “Temperature/K”.

Acknowledgment

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

References

Please number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] was the first . . ."

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors' names; do not use "et al.". Papers that have not been published, even if they have been submitted for publication, should be cited as "unpublished" [4]. Papers that have been accepted for publication should be cited as "in press" [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

References

[1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[3] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[4] K. Elissa, "Title of paper if known," unpublished.

[5] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.