CS7641 - Assignment 1: **Supervised Learning**

Phuc M. Nguyen
pnguyen333@gatech.edu

1 INTRODUCTION

1.1 Motivation

This project aims to explore, analyze and provide insights about common classes of supervised learning algorithms on 2 different **simple** datasets. Five different classification algorithms studied include: i) Decision trees with pre-pruning (DT); ii) Histogram-based gradient boosted decision trees (BoostDT); iii) Multi-layer perceptrons (MLP); iv) k - Nearest Neighbors (kNN) and v) C - Support Vector Machines (SVC). Each method's hyperparameters were optimized via Grid Search with successive halving (with cross-validation) on the training portion of each dataset before tested on the testing portion.

1.2 Selected Datasets

The first dataset selected was obtained from OpenML repository: Default vs. Non-default payments of credit-card carriers in Taiwan. There are 30,000 samples with 23 anonymized features, of which 3000 was randomly extracted for the testing set. Among the remained 27000 train samples, 5939 samples were labeled "1" (cardholders with default payment), so the labels were not equally distributed. The encoded features include information about gender, education level, marital status, age (years), history of past payment ...

The second chosen dataset was obtained from a past Kaggle competition: Two Sigma Connect - Rental Listing Inquiries, where the objective was to classify the interest level for a given rental listing. There are more than 49000 samples for the training data, this training data was used to create both training (40000 observations) and testing dataset (9352 observations) as labels for the competition's "true" test set were not released. There were originally 14 features with 3 interest levels (labels): "low" - 0, "medium" - 1, and "high" - 2. These original features include information about the number of bathrooms, bedrooms, listing created time, description, list of listings' features, latitude ... There was also a list of photo links. As the goal of the overall project was about the understanding of supervised learning algorithms, image processing and NLP were not conducted. Instead, the number of photo links and dummy binary features were created from the "photos" and listings' "features" columns. Again, the distribution among the 3 label types were not equal. For the training set, there were 27813 labeled "0", 9084 for "1" and 3103 for "2".

2 EXPERIMENTAL DESIGN

2.1 Grid-Search Hyperparameters

For each method, a validation curve was plotted for an arbitrary range of parameters values. Based on the validation curve, a narrower range of values was selected for each parameter where Grid Search with successive halving was applied. Below is the table of selected range of parameters for each supervised algorithms.

	Param 1	Param 2
DT	min_samples_leaf: [10,40]	max_depth [5, 20]
BoostDT	max_bins: [40,60]	max_iter [30,51]
MLP	hidden_layer_sizes (1 layer): [48, 64, 80, 96, 128, 192]	activation: ['logistic', 'tanh', 'relu']
kNN	n_neighbors: [1,10]	metric: ['minkowski', 'cityblock', 'cosine']
SVC	C: [1.02, 1.32, 1.69, 2.19, 2.82, 3.63, 4.68, 6.03, 7.76, 10]	kernel: ['poly', 'rbf', 'sigmoid']

Table 1—Selected parameters and corresponding search ranges for each supervised method.

2.2 Metric

For both datasets, the selected metric to evaluate performance is the F1 score, weighted by the number of instanced per label type. Weighted F1 score was chosen subjectively primarily due to the second dataset containing more than 2 classes and it was assumed that each class should be as important with respect to its probability.

2.3 Experimental Steps

For each algorithm the following procedure was applied:

1. Plot learning curve and elapsed time plot for the algorithm with all default parameters.
2. Plot validation curve by varying hyperparameter 1.
3. Plot validation curve by varying hyperparameter 2.
4. Visually inspecting the above validation curves to determine narrower ranges for hyperparameter 1 and 2 for Grid Search.
5. Find the training set - optimal hyperparameters by Grid search.
6. Construct the training set - optimal classifier based on the best parameters from step 4 above, then fit the classifier on the whole training set and make predictions on the testing set.
7. Plot confusion matrix and record the weighted F1 score.

For step 1 to 5, scores were all obtained from 5-fold cross-validation.

3 RESULTS

3.1 Default vs. Non-default Payment

model_name	metric_score	params	train_time (s)	infer_time (s)
DT	0.789	{'max_depth': 5, 'min_samples_leaf': 38}	0.155	0.000342
BoostDT	0.788	{'max_bins': 52, 'max_iter': 30}	0.142	0.002072
MLP	0.667	{'activation': 'logistic', 'hidden_layer_sizes': (96,)}	2.492	0.008032
kNN	0.702	{'metric': 'cosine', 'n_neighbors': 10}	0.002	1.99867
SVC	0.789	{'C': 1.32 , 'kernel': 'rbf'}	26.019	1.395917

Table 2—Training-optimal parameters and performance results for each supervised model on the Default vs. Non- default payments of credit-card carriers dataset.

Among the five supervised methods, decision tree with pruning (DT) and C-support vector machines (SVC) had the highest weighted F1 scores, followed by Histogram - based Boosted Decision Trees (BoostDT). However, a closer look at figure 1 showed that both DT and BoostDT models had better accuracy based on the true positive and true negative ratios. With regards to training and inference time, both DT and BoostDT significantly outsped the others, with SVC being the slowest.

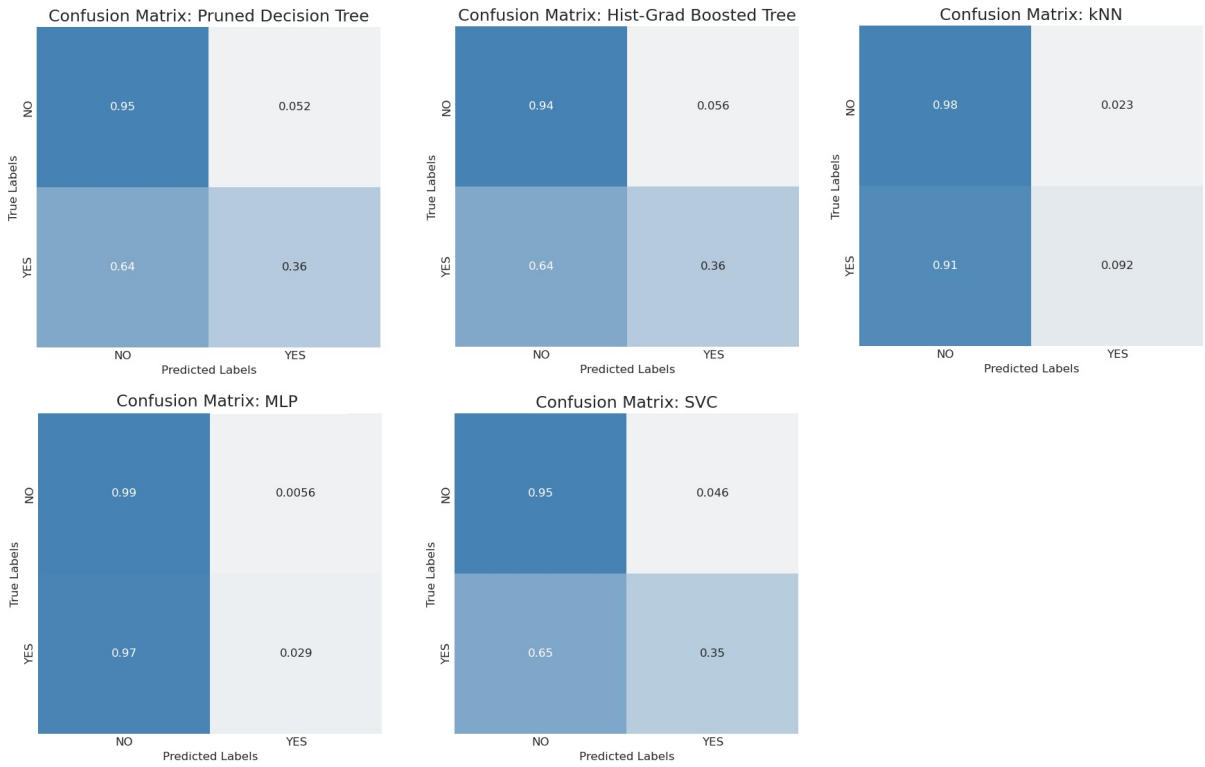

Figure 1—Scaled confusion matrices of fine-tuned supervised models on Default Payment Dataset. Within each cell of the matrix, the predicted value is divided by the true value; when the value is 1 the number of predicted equals the number of true value.

3.2 Rental Listings

model_name	metric_score	params	train_time (s)	infer_time (s)
DT	0.668	{'max_depth': 8, 'min_samples_leaf': 39}	0.138	0.001048
BoostDT	0.710	{'max_bins': 56, 'max_iter': 41}	0.480	0.008730
MLP	0.566	{'activation': 'logistic', 'hidden_layer_sizes': (64,)}	1.249	0.009362
kNN	0.628	{'metric': 'cityblock', 'n_neighbors': 10}	0.005	0.952476
SVC	0.566	{'C': 2.19, 'kernel': 'rbf'}	62.483	6.355503

Table 3—Training-optimal parameters and performance results for each supervised model on the Rental Listing Inquiries dataset.

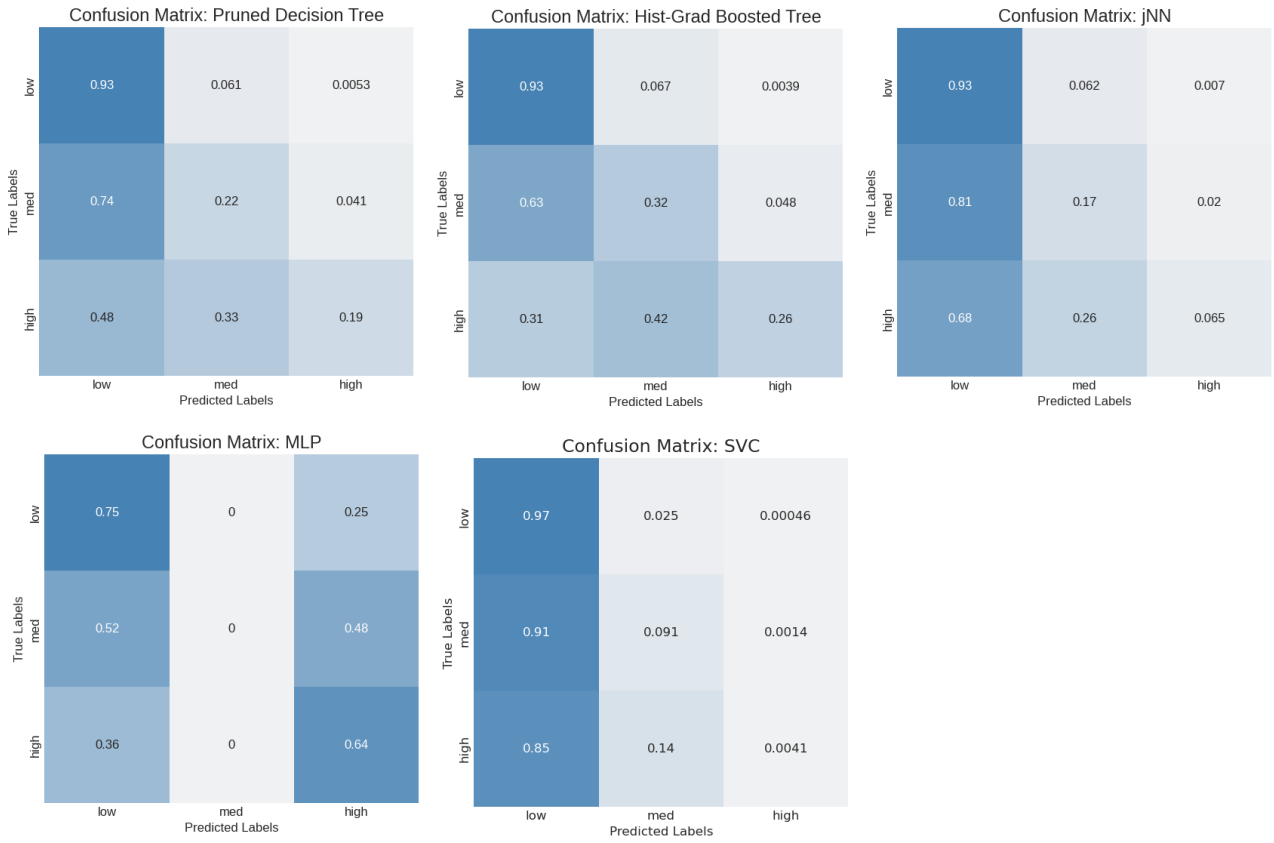

Figure 2—Scaled confusion matrices of fine-tuned supervised models on Rental Listing Dataset. Within each cell of the matrix, the predicted value is divided by the true value; when the value is 1 the number of predicted equals the number of true value.

For the Rental Listings classification task, BoostDT had the highest F1 score. BoostDT also had the highest accuracy when considering all three label types. At figure 2 showed that although MLP predicted significantly more accurate high-interest level observations than BoostDT, MLP almost didn't classify any instance as medium-interest level observation. In terms of training and inference time, SVC the slowest at both training and inferencing. All other algorithms took nore more than 1.25 seconds (MLP) for fitting the whole training set but SVC took more than

62 seconds.

4 DISCUSSION

4.1 Decision Tree with Pruning (DT)

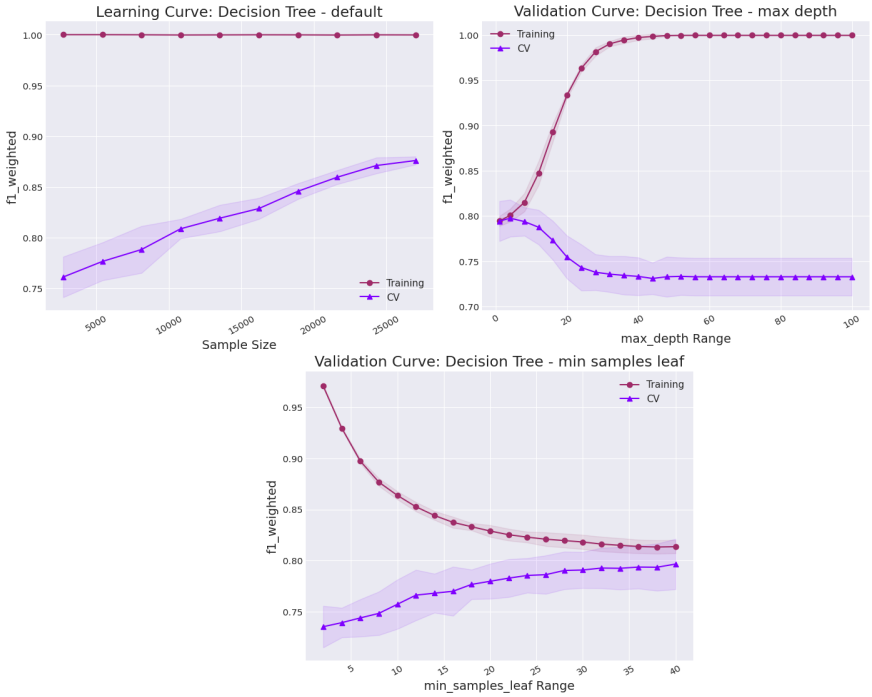

Figure 3—Default learning curve and experiment validations curves for DT on Default payment classification task. The chosen hyperparameters were max depth and minimum samples in leaf nodes.

The two learning curves both showed that F1 weighted scores improved as more samples were added. This was trivial as with more data it became easier for a DT to approximate the attribute distributions that were closer to the validation data. Experiments with the DT's maximum depth demonstrated that by increasing the maximum depth the validation performance the performance would gain a little before dropping offto a plateau on the validation set while the training score kept increasing, showing DTs were overfitted on the training data. Increasing the minimum number of samples in the leaf nodes helped rectifying this issue, at the cost of the DTs becoming more underfitted. A good balance between maximum depth and leaf nodes' minimum sample numbers helped achieving a good optimal balance. DT with pruning suffered from high bias due to its relative simplicity.

4.2 Histogram - based Gradient Boosted Decision Trees (BoostDT)

On both tasks, the learning curves showed that by increasing sample size the model performance on the validation sets increased while the score on the training set decreased, meaning that more training data helped correcting the overfitting behavior of boosted DT ensemble. For the Default vs Non-default payment classification task, increasing the number of histogram

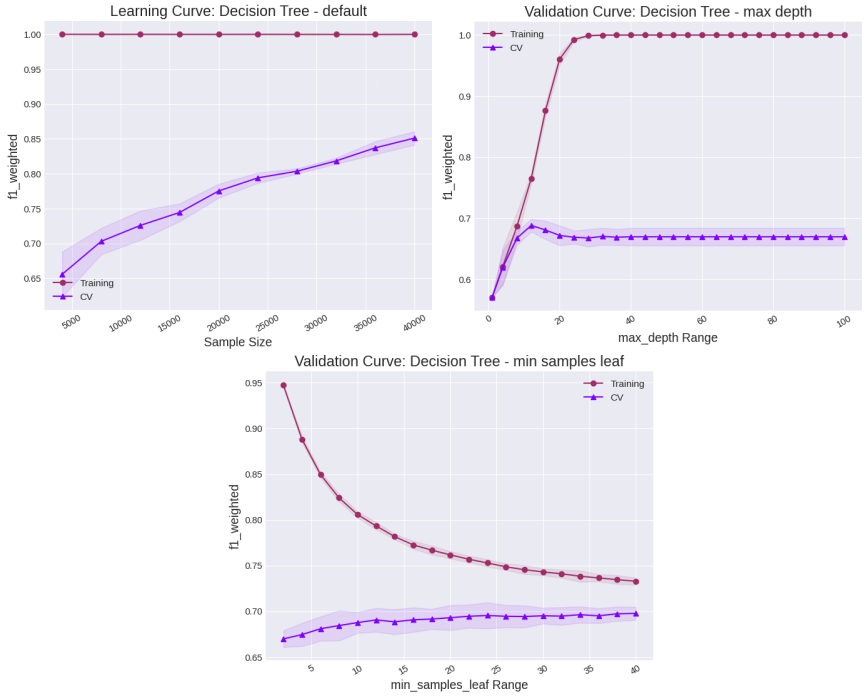

Figure 4—Default learning curve and experiment validations curves for DT on Rental Listing Inquiries classification task. The chosen hyperparameters were max depth and minimum samples in leaf nodes.

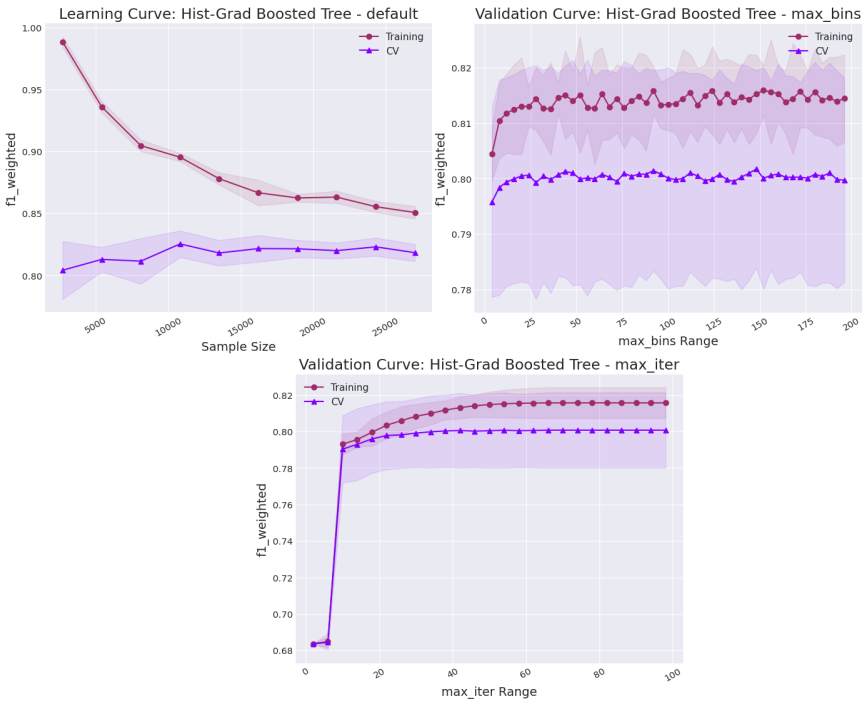

Figure 5—Default learning curve and experiment validations curves for BoostDT on Default payment classification task. The chosen hyperparameters were maximum bin and maximum boosting iteration

bins for feature value binning did not show any significant result in comparison to the Rental Listing Inquiries task where there was an initial jump in weighted F1 score and then a plateau.z In addition varying max bins in Default payment dataset showed a much higher variation in score standard deviation in comparison to the other task, this was probably due to the fact the Default payment feature values had a wider range of values. On both tasks, especially the Rental Listing Inquiries task, increasing the maximum number of boosting iterations could lead to more overfitting on the training dataset. In comparison to DT with pruning, BoostDT did not suffer from high bias.

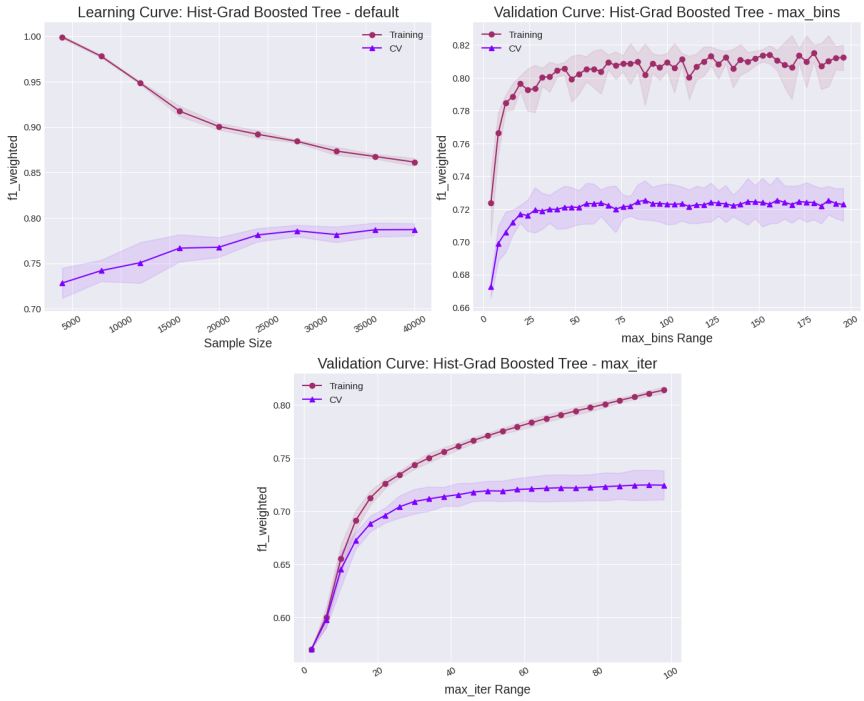

Figure 6—Default learning curve and experiment validations curves for boostDT on Rental Listing Inquiries classification task. The chosen hyperparameters were maximum bin and maximum boosting iteration

4.3 Multi-layer Perceptrons (MLP)

For both taks, the learning curves showed that mean and standard deviation of weighted F1 score of the default MLP (1 hidden layer with 100 nodes) was similar across both the training and validation sets. With more training samples the moddel didn't achieve visibly higher scores. Logistic and tanh activation functions gave similar mean results across both tasks, with very low standard deviation; ReLu, however, yielded a much higher standard deviation. Looking at the confusion matrices (fig 1 and 2) and the fact that the optimal activation function chosen by HalvingGridSearchCV, this might be explained that with logistic and tanh functions the model was stuck in a local optima or that the units in the hidden layer were saturated with very large negative and positive weight values. In addition, for the second task there were 3 different classes of labels and this might explain how ReLU performed worse than the other 2 activation functions, as the left half of ReLU function was simply 0 and the right half behaved just like

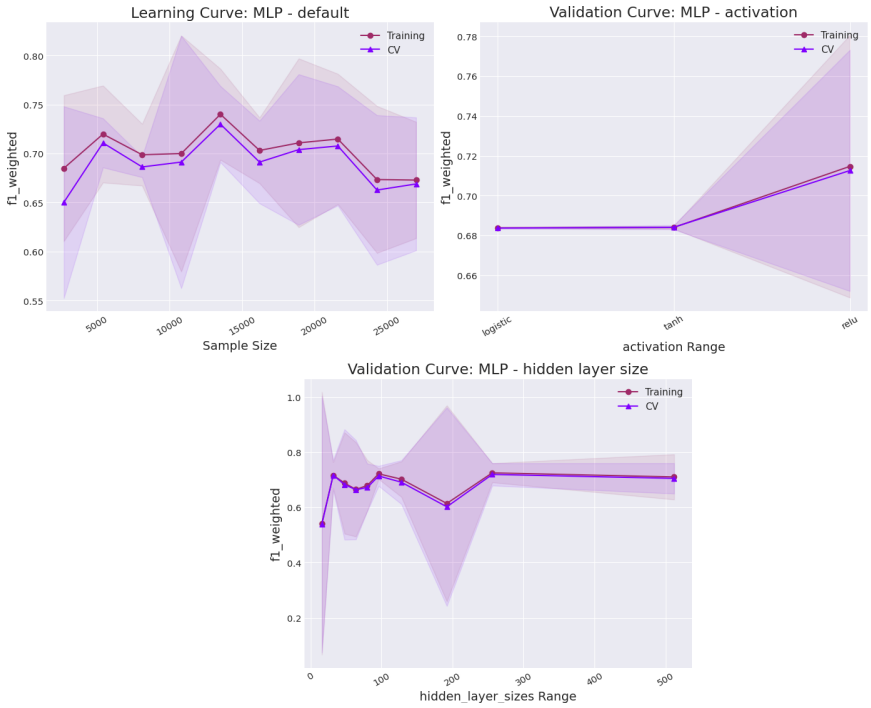

Figure 7—Default learning curve and experiment validations curves for MLP (1 hidden layer) on Default payment classification task. The chosen hyperparameters were number of nodes in hidden layer and the type of activation function.

a linear function. Varying the number of nodes in the single hidden layer did not produce any significant improvement for both tasks. Finally, with respect bias-variance trade-off MLP suffered more from high variance based on the broad band of standard deviation from the learning curves. When logistic and tanh were selected as the activation function of choice, the model from high bias, while with ReLU the model were more prone to variance error.

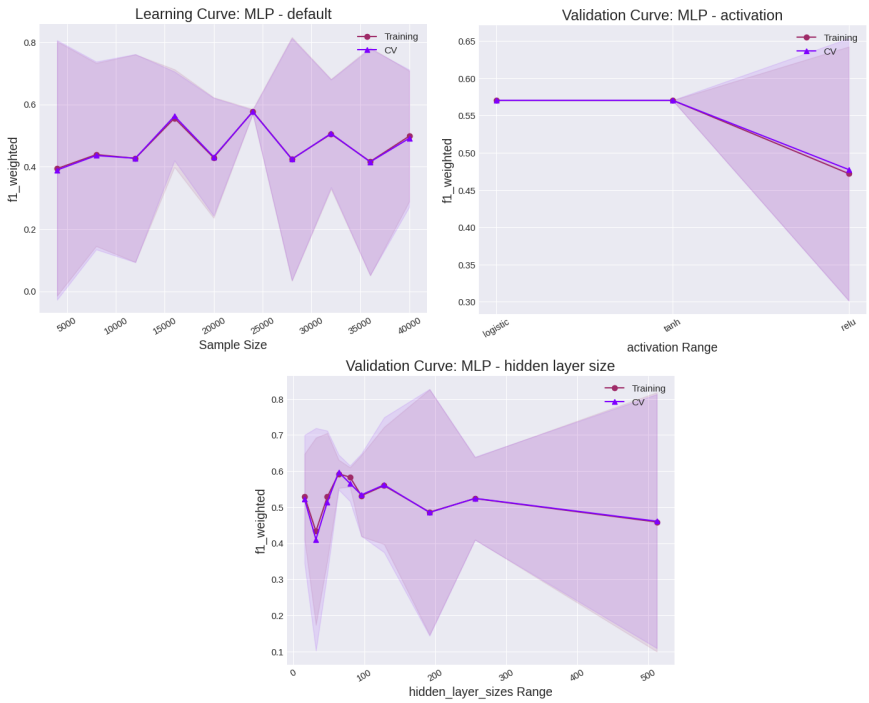

Figure 8—Default learning curve and experiment validations curves for MLP (1 hidden layer) on Rental Listing Interest classification task. The chosen hyperparameters were number of nodes in hidden layer and the type of activation function.

4.4 k-Nearest Neighbors (kNN)

The two learning curves both showed that kNN's F1 weighted scores on both training and validation improved as more samples were added. This was trivial as with more data it became easier for a kNN to approximate more correct labels based as there were potentially more nearby samples where majority voting rules applied. Experiments with the kNN's number of neighbors on both tasks showed that by increasing the number of neighbors considered for voting the model became less overfitted on the training data. There was no significant difference when selecting different distance metrics for either dataset. Among all the algorithms kNN was very fast at training time but had very long inference time, though not as long as SVC.

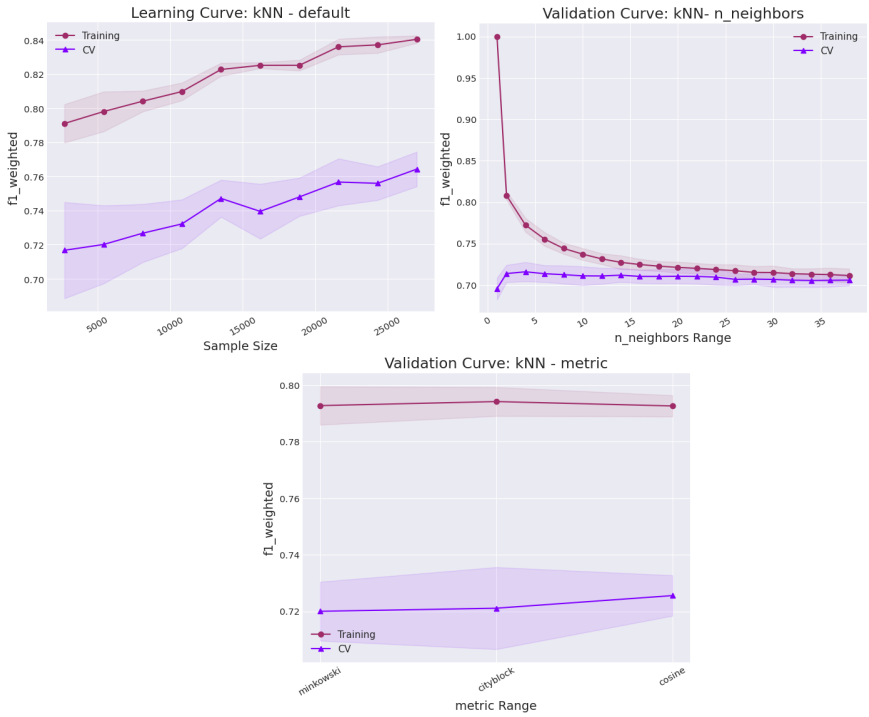

Figure 9—Default learning curve and experiment validations curves for kNN on Default payment classification task. The chosen hyperparameters were number of neighbors and the type of distance metric.

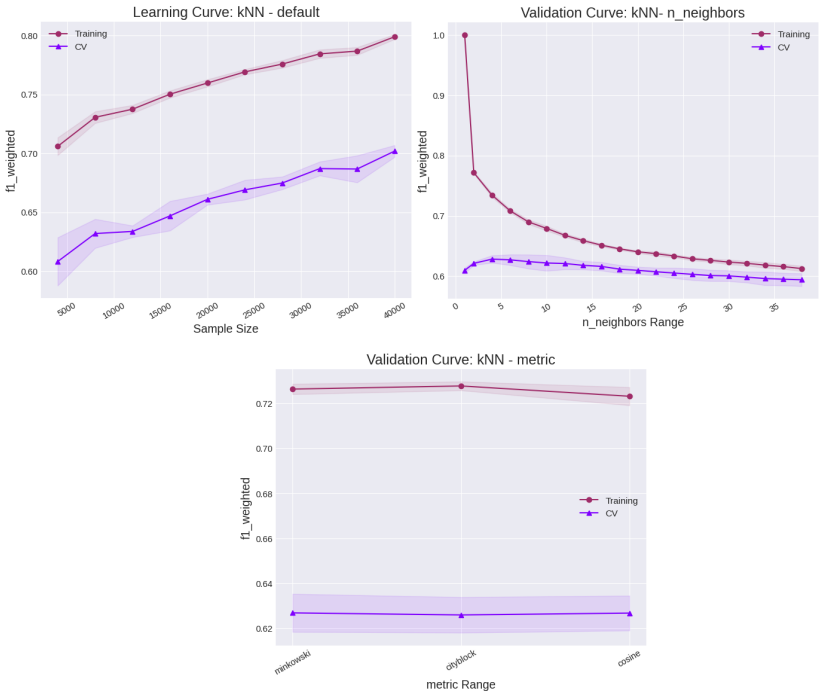

Figure 10—Default learning curve and experiment validations curves for kNN on Rental Listing Interest classification task. The chosen hyperparameters were number of neighbors and the type of distance metric.

4.5 C-Support Vector Machines (SVC)

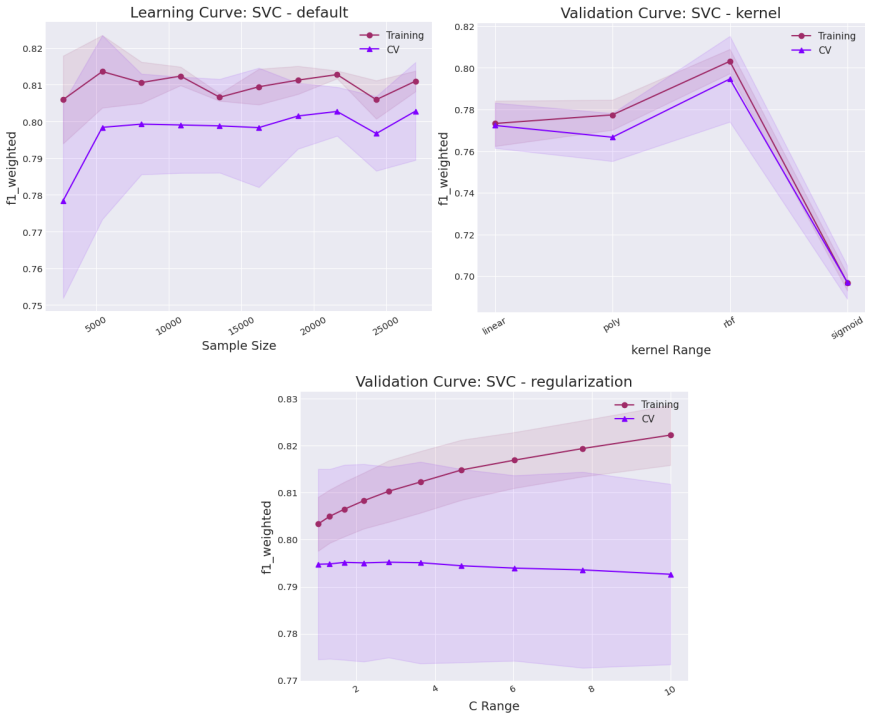

Figure 11—Default learning curve and experiment validations curves for SVC on Default payment classification task. The chosen hyperparameters were types of kernel and C value (regularization).

For SVC, the two learning plots were signifcantly different from each other. For the Default vs. Non-default payment task, as sample sizes increased, the model performance increased initially before almost plateuing. In comparison, for the second task the model's training and cross-validation scores were almost exactly similar. For both tasks, the radial basis function (rbf) kernel were selected to be the optimal kernel; on the 2nd task specifcally there was no visible difference between poly and rbf. For the second task, both poly and rbf kernel had very low validation standard deviation when comparing to sigmoid, this implied that when using poly or rbf for this particular task the model suffered from high bias while with sigmoid the model had high variance error. Looking at confusion matrices from figures 1 and 2, it could be observed that SVC performed very poorly with very low accuracy for the 2nd task. This could be explained by the fact that the first task had many more continuous numerical features than the second task and the mapping hyperplane constructed by SVM with categorical features could not separate observations from different labels well. With respect to regularization, by increasing the C value (which is inverse to regularization strength) we could observe an increase in training weighted F1 score and a decrease in the validation score, implying that the model was overfitting in the first task. On the second task, the model had significant variation in performance for all C values.

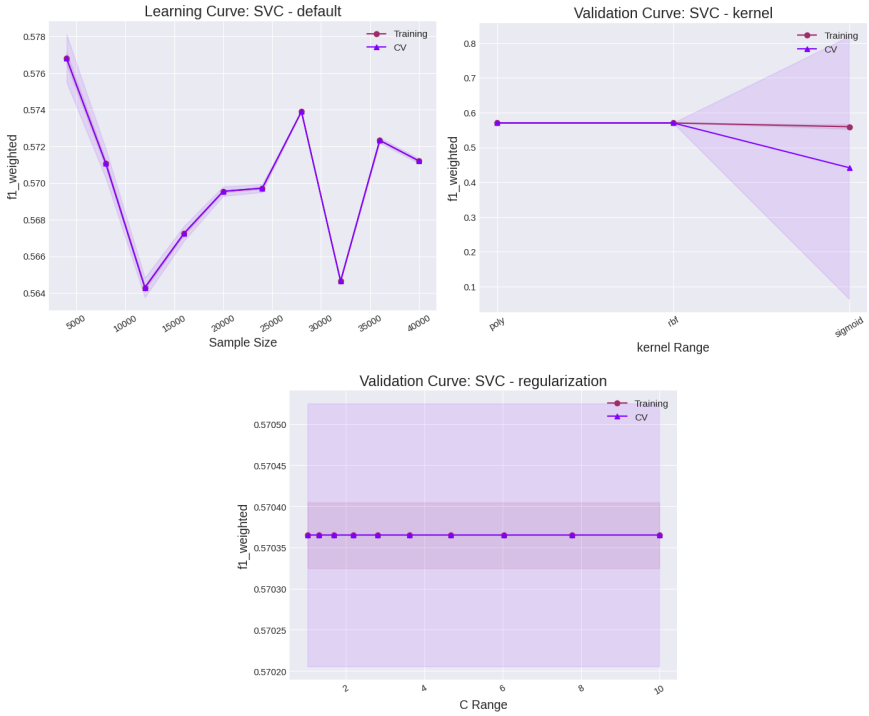

Figure 12—Default learning curve and experiment validations curves for SVC on Rental Listing Interest classification task. The chosen hyperparameters were types of kernel and C value (regularization).