CS 7641 Machine Learning (Project 1): Implementation, Testing and Analysis of Commonly-Used Supervised Learning Algorithms with Two Different Classification Problems

Qisen Cheng

Computer Science Department, Georgia Institute of Technology,
Atlanta, Georgia, USA 30318, qcheng35@gatech.edu

Abstract. This work, as part of the course requirement of CS7641, demonstrates the implementation and analysis of five commonly used supervised learning algorithms in the setting of classification. The algorithms discussed in the report are: 1) K-nearest-neighbors (KNN), 2) support vector machine (SVM), 3) decision tree (DT), 4) boosting, and 5) artificial neural network (ANN). Several analyses are made in terms of data preprocessing, training process, parameter tuning, and performance comparison for each of the algorithms, respectively. The algorithms are implemented with several existing libraries (i.e. scikit-learn) in Python.

Keywords: CS7641, supervised learning, classification, implementation, analysis, python

1. Introduction of Classification Problems

1.1 Selected problems & datasets

Dataset 1-Titanic survival prediction (Titanic): In this problem, it asks to predict if certain passengers can survive the famous Titanic tragedy, based on serval given information of the passengers.

Dataset 2-Digit reorganization (MNIST): In this problem, is asks to classify given hand-writing images of 10 digits into certain numerical digits. Only 10000 rows are randomly selected to be used in this work.

The training datasets are both available on the Kaggle website. Only the training data is used as the full dataset for this work since the test data on Kaggle are unlabeled. All of the 5 classifiers are tested on both datasets. In both tests, 30% of the dataset is split to be testing data, while the remaining 70% is used for development of the models. To emphasis the comparison on algorithm performance and training process, feature engineering is minimized unless necessary (i.e. PCA). No extra features are generated in addition to the original ones.

1.2 Comparison between datasets & reasons of the choice

Table 1.1: Comparison between datasets

Dataset	Examples	Features	Classes	Type of dataset	Balance between classes
Titanic	891	12	Binary: 2	Tabular dataset (Numerical, texts, categorical)	Unbalanced
Digits	10000	785	Multiclasses: 10	Image dataset (vectorized pixel intensity)	Balanced

Several reasons are considered for the choice.

Reason 1 – "Small data VS. Big data": The two different datasets are different in size with respect to both examples and features that are available for the models to learn from. This difference may reveal some attributes of different algorithms about their ability to "handle" small and big data. For example, it can be expected that NN will outperform the others with relatively large dataset, but may suffer overfitting problems with the small dataset.

Reason 2 – "Binary VS. Multiclass": Binary classification problem appears to be easier than classification between multiple classes. The "harder" problem may need more proper design and tuning of parameters for some algorithms. This difference can lead to some comparisons of classifiers in tackling binary and multi-class classifications in terms of performance, easiness of training, etc. For example, linear SVM might be hard to use for multiclass dataset since the separation between classes is intuitively non-linear.

Reason 3 – "Balanced VS. Unbalanced": Unbalanced data intuitively contain some noise related to the difference of population between classes. This might lead to different thinking of performance tuning.

Reason 4 – "Tabular VS. Image": Different data type provides different attributes of features. For example, the features in tabular dataset may be less correlated compared to the features of image. Thus, this can lead to different thinking of data pre-processing, which is also highly specific for different algorithms.

2. K-Nearest-Neighbors Classifier

Data-preprocessing.

1) Titanic: 12 features are in the original tabular dataset. The "SibSp" and "Parch" features are combined and yield two features named "Family_Size" and "Alone". This process combines the information of number of relatives for each passenger, and emphasizes on the fact people alone may have larger survival chances since they tend to escape regardless of others. The missing data in "Age" feature is estimated based on the passengers' name initials – for example, the missing age of passenger with initial "Mr" is estimated to be the mean of all the passengers with initial "Sir". The continuous numerical features like "Age" and "Fare" are categorized by grouping. The text features like "Name", "Ticket", "Cabin" are dropped. After preprocessing, the dataset has 8 features for each example. Since the features are not highly correlated (Figure 2.1) to each other, no extra processing is used.

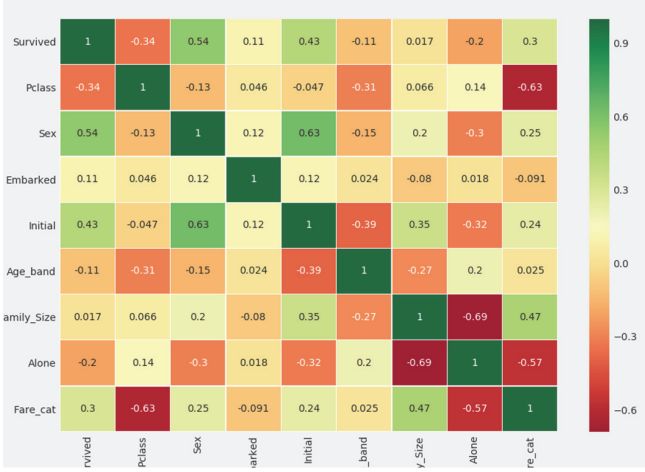


Figure 2.1: Heat map of correlation between each pair of features in Titanic dataset after preprocessing.

2) Digits: MNIST image data suffers significant correlations between features. Adjacent pixels tend to have similar intensity level and trend of intensity change. In KNN algorithm, the classification is made based on "similarity metric", or intuitively "distance" between the unseen samples and the learned samples. A large number of highly correlated features will make it hard to accurately evaluate the "distance". Thus, compared to Titanic dataset, some de-correlation process is required for MNIST image data. Here, principal component analysis (PCA) with whitening function is used to 1) de-correlate the features, 2) select a small group of representative features, and 3) normalize the selected features. The following figure (Figure 2.2) empirically shows that with certain amount of de-correlated features (=30), variance information of the dataset can be expressed sufficiently (=98.6 %). More low-variance features will degrade the performance of KNN in evaluation of similarity metrics. Thus, only 30 features from PCA is used for training of the KNN model.

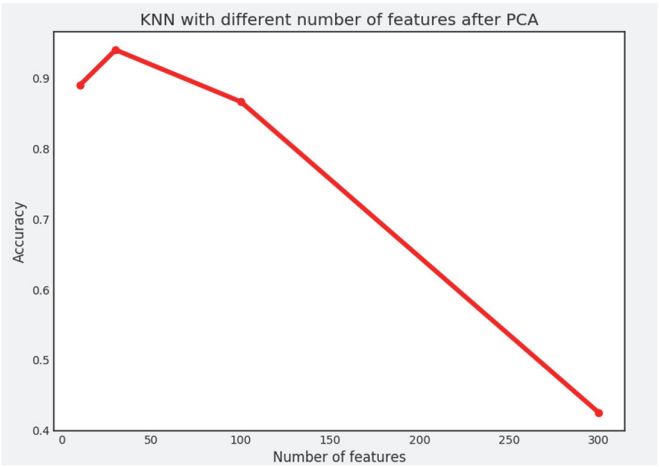


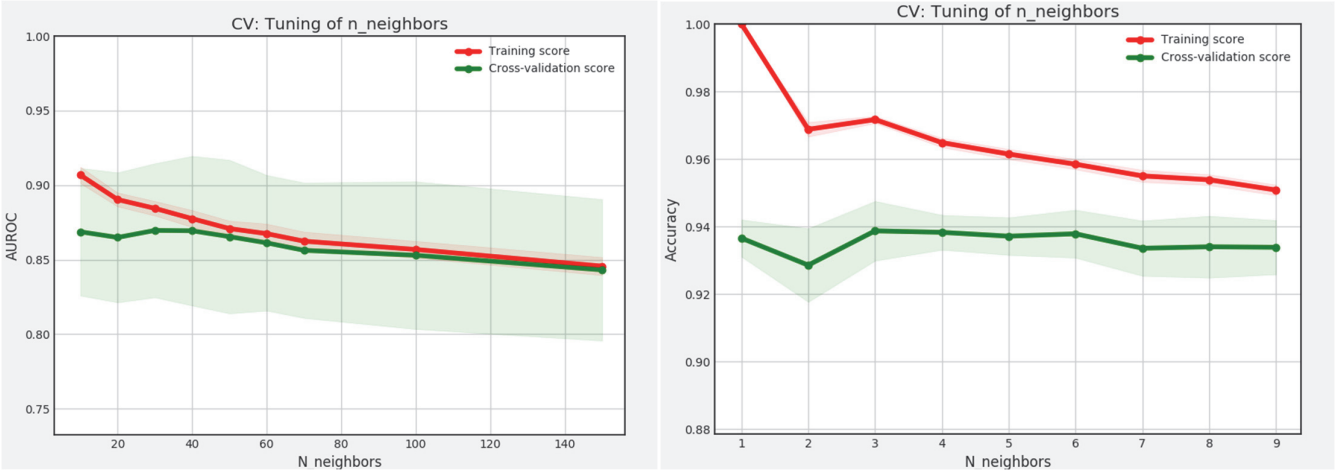
Figure 2.2: Empirical performance of KNN (with default parameters) with number of features.

Parameter tuning.

The only parameter tuned for KNN in this work is the number of neighbors considered in the majority voting.

1) Titanic: The parameter (n_neighbor) is tuned using grid search with 10-fold cross validation. Considering the fact that the populations of survived and un-survived group in the dataset are unbalanced (roughly 30 % vs. 70 %), the validation folds are drawn in a stratified sampling fashion. This diminishes the difference in population between training data and validation data, thus helps to minimize the variation in validation score. The validation score is also chosen to use AUROC instead of accuracy. AUROC is less sensitive to unbalancing between populations and still targets for good accuracy to some extent. The searched grid of n_neighbor is: 10, 30, 50, 60, 70, 100, 150. In following figure 2.3(a), it can be seen that 30 can be an optimal choice.

2) Digits: The same parameter is tuned in the testing with Digits dataset using the same approach as for Titanic dataset. One difference is the grid search is changed to 5-fold to remain relatively larger number of examples to be learned in each folds. Besides, the folds are randomly sampled since the data is balanced. The cross-validation score is changed to accuracy and search grid is 1-10. The optimal parameter is selected to be 3 (Figure 2.3(b)).



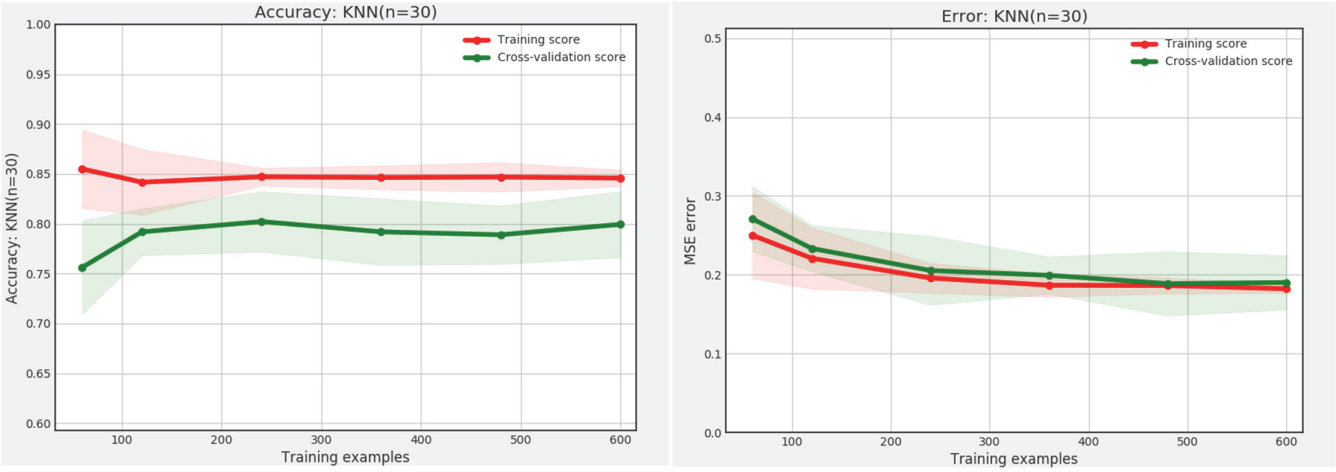
(a) Titanic dataset (b) Digits dataset

Figure 2.3: Tuning of number of neighbors for (a) Titanic and (b) Digits datasets.

Cross-validation curve of trained KNN models.

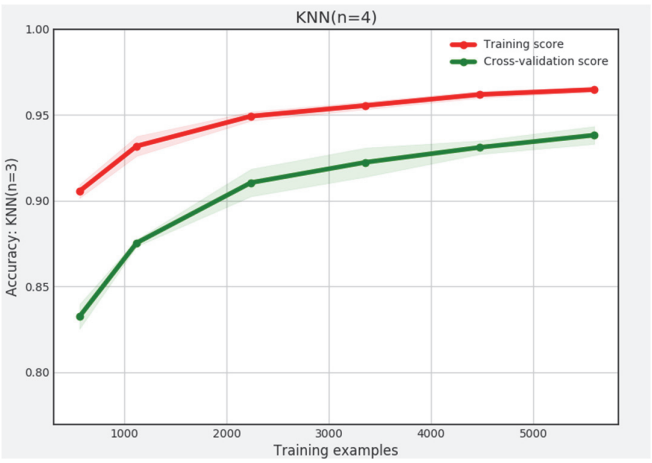
1) Titanic: The learning curves (accuracy-training size and MSE error-training size) are shown in figure 2.4. It can be seen from (a) that the training accuracy is slightly decreased as the validation accuracy increases. Besides, the error curve shows that the training error is below the validation error, and both decrease as training size increases. It can be concluded that the trained model is acceptable in terms of generalization, though some suspicions about underfitting can be made based on the gap between red and green lines in the accuracy curve.

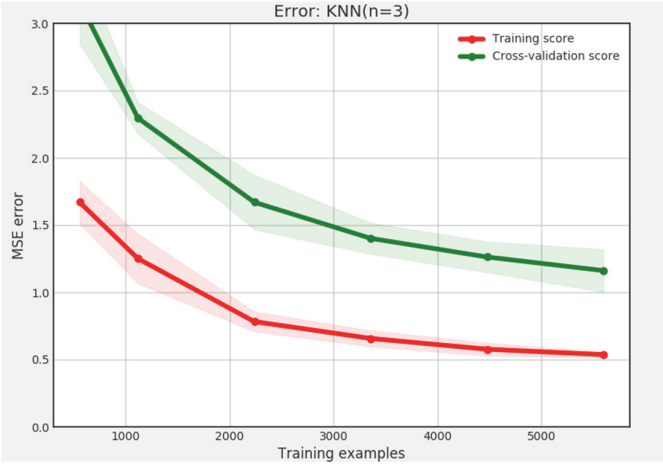
2) Digits: The learning curves are shown in figure 2.5. It can be concluded from both the accuracy curve and the error curve that the trained model is good – not underfitting since both lines are converging to good metrics; not overfitting since gaps between validation and training are getting smaller in both (a) and (b).



(a) Titanic dataset – Cross validation (accuracy) (b) Titanic dataset – Cross validation (MSE)

Figure 2.4: Cross-validation for KNN (with optimal parameter) for Titanic dataset.





(a) Digits dataset – Cross validation (accuracy) (b) Digits dataset – Cross validation (MSE)

Figure 2.5: Cross-validation for KNN (with optimal parameter) for Digits dataset.

Testing.

The trained models are tested with testing data set for both Titanic and Digits problems. The results can be found in Table 2.1. It can be concluded that the performance of KNN classifier is improved with cross-validation for parameter tuning in both testing cases. The difference of performance between Titanic KNN and Digits KNN is mainly due to difference of available training examples. Since KNN is an instance-based classifier, the performance is pretty related to the training size – the more instances it saw, the better it can classify similar observations. The time of training is of at least one magnitude smaller than the time for testing. This is mainly because KNN does not do much learning (basically memorizing the seen instances), but make efforts in measuring the similarity when encountering new observations. It also can be seen from Figure 2.6 that the unbalanced data of Titanic dataset imposes some preference to the class with larger population (un-survived group), while the balanced data of Digits dataset does not have the effect.

Table 2.1: Testing performance of trained models W. and W/O. cross validation (CV) for parameter tuning.

Dataset	**W/O. CV**	**W. CV**				
	Accuracy	**Accuracy**	**F1**	**AUROC**	**Training time**	**Testing time**
Titanic	78.03 %	80.27 %	0.802	0.775	0.00098 s	0.0038 s
Digits	93.97 %	94.2 %	0.942	0.968	0.015 s	1.33 s

	precision	recall	f1-score	support
0	0.80	0.90	0.85	137
1	0.80	0.65	0.72	86
avg / total	0.80	0.80	0.80	223

	precision	recall	f1-score	support
0	0.95	0.99	0.97	282
1	0.92	0.99	0.95	327
2	0.95	0.92	0.94	295
3	0.92	0.94	0.93	315
4	0.95	0.92	0.93	293
5	0.93	0.91	0.92	280
6	0.98	0.98	0.98	287
7	0.95	0.93	0.94	347
8	0.96	0.90	0.93	296
9	0.92	0.94	0.93	278
avg / total	0.94	0.94	0.94	3000

(a) Titanic dataset – performance scores (b) Digits dataset – Performance scores

Figure 2.6: Performance scores of different testing datasets.

3. Support Vector Machine Classifier

This section details the implementation and analyses of SVM classifier with both Titanic and Digits datasets.

Data-preprocessing.

1) Titanic: The same preprocessing procedures for KNN is used for SVM.

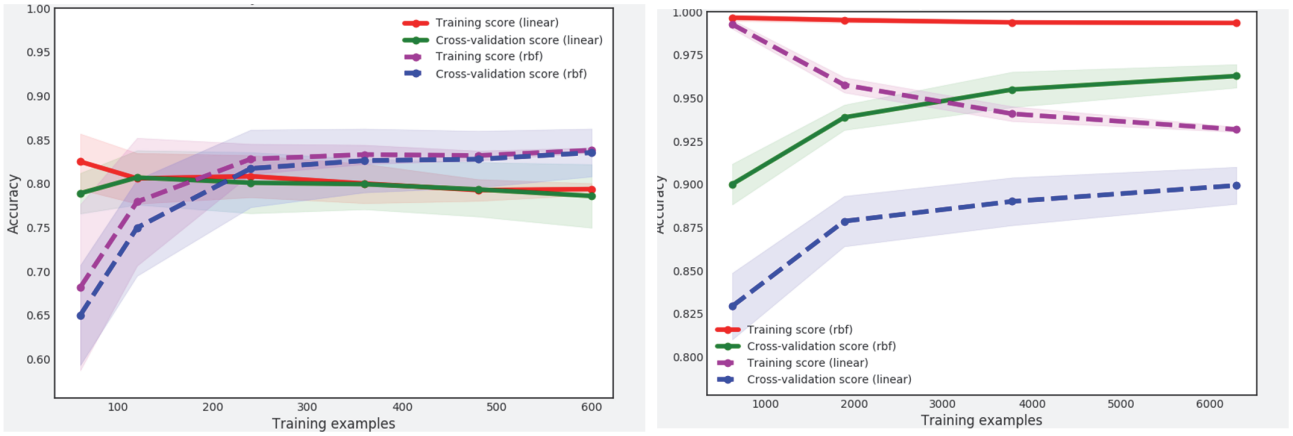
2) Digits: The same preprocessing procedures for KNN is used for SVM.

Kernel selection and parameter tuning.

Candidate kernels considered in this work include linear kernel and radial basis function (RBF) kernel. In order to have a valid comparison, some coarse tuning of parameters is performed first. The parameters considered include punishment coefficient (C) for both kernels and influence radius gamma for RBF. Parameters of selected kernel is further tuned using cross-validation for better performance.

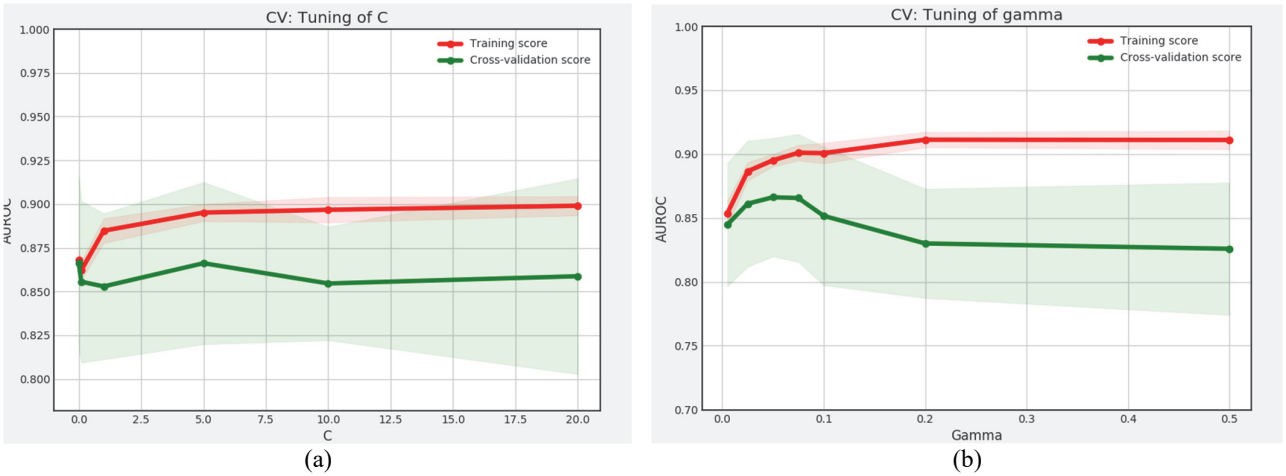
1) Titanic: The performance comparison between linear kernel and RBF is shown in figure 3.1(a). The linear model is consistently overfitting (even with relatively small C=0.1) compared to RBF kernel. Thus, RBF kernel is selected for this dataset. The parameters for RBF kernel (C and gamma) are tuned using grid search with 10-fold cross validation. The validation score is also chosen to use AUROC instead of accuracy. AUROC is less sensitive to unbalancing between populations and still targets for good accuracy to some extent. The searched grid of C is: 0.001, 0.1, 1, 5, 10; and gamma is: 0.001, 0.01, 0.05, 0.1. In following figure 3.2, it can be seen that C=5, and gamma=0.05 can be an optimal choice.

2) Digits: The kernel selection and parameters tuning are performed for Digits dataset using the same approach as for Titanic. The RBF kernel is selected for its promising cross validation curve. The optimal parameters are selected to be C= 0.1, gamma = 0.05 (Figure 3.3).



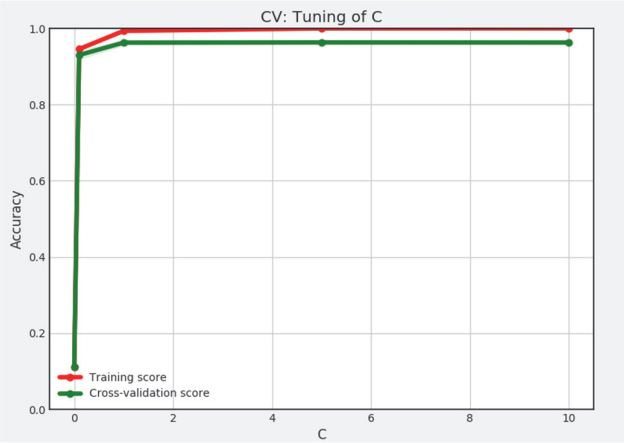
(a) Titanic dataset (b) Digits dataset

Figure 3.1: Selection of kernel (linear or RBF) for (a) Titanic and (b) Digits datasets.

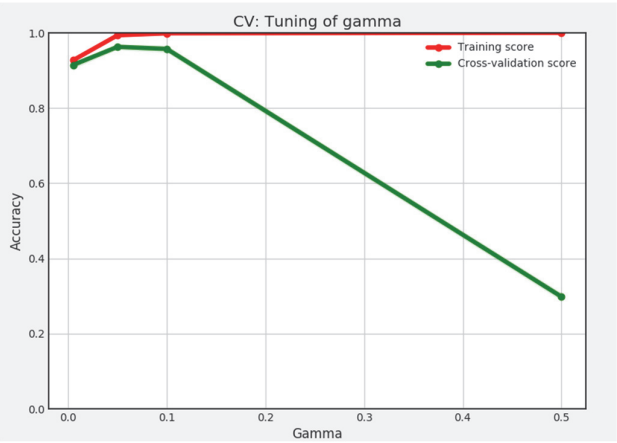


(a) (b)

Figure 3.2: Tuning of (a) C and (b) gamma for Titanic dataset



(a)



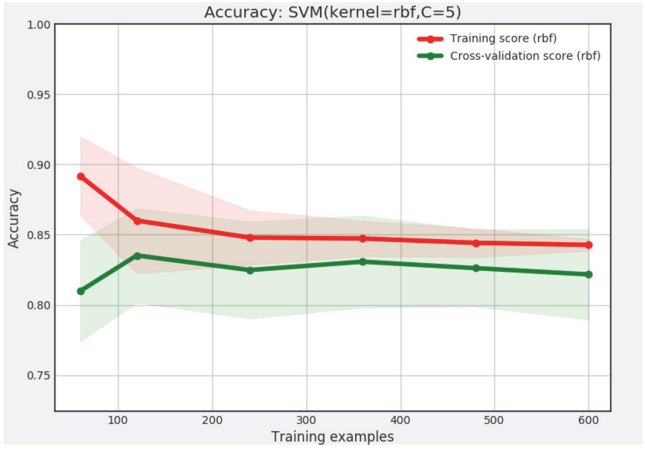
(b)

Figure 3.3: Tuning of (a) C and (b) gamma for Digits dataset

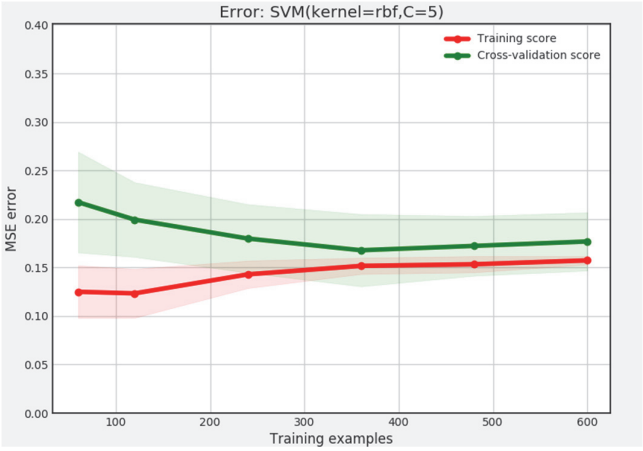
Cross-validation curve of trained SVM models.

1) Titanic: The learning curves (accuracy-training size and MSE error-training size) are shown in figure 3.4. It can be seen from both accuracy and error curves that the trained model suffers underfitting.

2) Digits: The learning curves are shown in figure 3.5. It can be concluded from both the accuracy curve and the error curve that the trained model is good – not underfitting since both lines are converging to good metrics; not overfitting since gaps between validation and training are getting smaller in both (a) and (b).

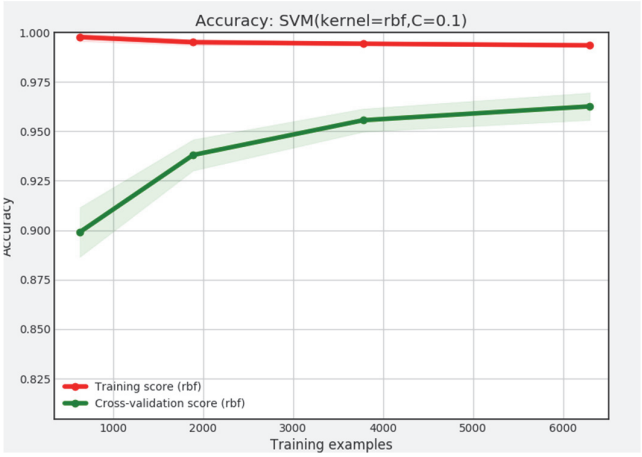


(b) Titanic dataset – Cross validation (accuracy)

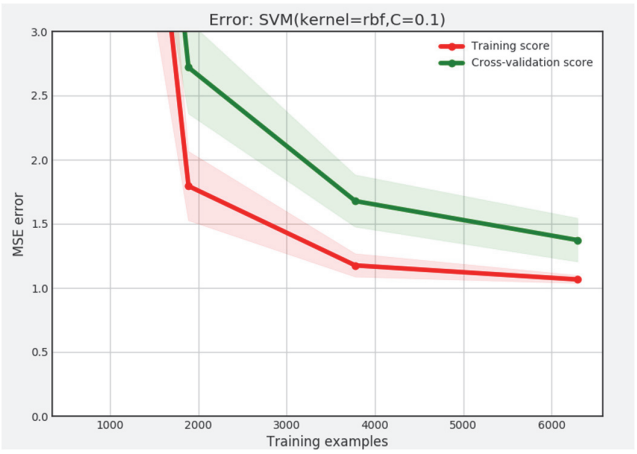


(b) Titanic dataset – Cross validation (MSE)

Figure 3.4: Cross-validation for SVM (with optimal parameter) for Titanic dataset.



(b) Digits dataset – Cross validation (accuracy)



(b) Digits dataset – Cross validation (MSE)

Figure 3.5: Cross-validation for SVM (with optimal parameter) for Digits dataset.

Testing.

The trained models are tested with testing data set for both Titanic and Digits problems. The results can be found in Table 3.1. It can be concluded that the performance of SVM classifier is improved with cross-validation for parameter tuning in Digits dataset, while not helpful in Titanic dataset. The difference of performance between Titanic SVM and Digits SVM is mainly due to difference of available training examples. The time of training for linear kernel is much larger than that for RBF kernel in the Titanic case, due to slower convergence. But the time of testing for linear kernel is much less in the Titanic case. In the Digits case, both training time and testing time of linear kernel are much less than that for RBF kernel. Thus, it can be clearly seen that linear kernel requires much less computation than RBF kernel.

Table 3.1: Testing performance of trained models W. and W/O. cross validation (CV) for parameter tuning.

Dataset	W/O. CV	W. CV					
	Accuracy	Accuracy	F1	AUROC	Iterations	Training time	Testing time
Titanic	81.61 %	81.17 %	0.811	0.782	Linear: 21685	Linear: 0.043 s	Linear: 0.0009 s
					RBF: 1433	RBF: 0.018 s	RBF: 0.004 s
Digits	93.06 %	96.20 %	0.929	0.961	Linear: 31996	Linear: 1.188 s	Linear: 0.251
					RBF: 24918	RBF: 4.413 s	RBF: 1.375 s

	precision	recall	f1-score	support
0	0.81	0.91	0.86	137
1	0.82	0.65	0.73	86
avg / total	0.81	0.81	0.81	223

(b) Titanic dataset – performance scores

	precision	recall	f1-score	support
0	0.96	0.99	0.97	282
1	0.98	0.97	0.98	327
2	0.90	0.95	0.92	295
3	0.87	0.93	0.90	315
4	0.90	0.93	0.92	293
5	0.93	0.89	0.91	280
6	0.96	0.95	0.96	287
7	0.97	0.90	0.93	347
8	0.93	0.91	0.92	296
9	0.91	0.89	0.90	278
avg / total	0.93	0.93	0.93	3000

(b) Digits dataset – Performance scores

Figure 3.6: Performance scores of different testing datasets.

4. Decision Tree (Single & Gradient-Boosting Version) Classifiers

Data-preprocessing.

1) Titanic: The same preprocessing procedures for KNN is used for DT and GB.

2) Digits: PCA is not used with DT and Boosting classifiers. The reason is that, during training, DT and GB can automatically evaluate and choose different features based on their importance (closely related to variance to some extent). PCA also selects the best dimensions (aka. features) based on variance. Thus, the repeated selection of features based on variance can cause DT and GB classifiers become overfitting.

Pruning and cross validation.

The major issue that DT and GB suffer is about overfitting. Thus, pruning of DT and GB is essential for the training of models. In this work, pre-pruning is applied to both DT and GM classifiers. Pre-pruning parameters include maximum tree depth (max_depth), minimum number of samples for splitting (min_samples_split), minimum impurity decrease for splitting (min_impurity_decrease), and maximum of features to be considered for splitting at each node (max_features). Additionly, number of trees (n_tree) is another parameter for GB classifier. All the parameters can limit the growth of trees. Due to the limitation of pages, only the tuning results are presented here. The detailed parameter tuning process is missed.

1) Titanic: DT is tuned with min_sample_split (=34), which limits the splitting as the tree grow to the leaf level. This parameter is the most effective parameter to prevent overfitting in the Titanic case. In contrast, max_depth tend to prune relatively deep subtrees (or make the depth of different subtrees more uniform). It turns out to be not effective since the training date size is not large – so that relatively deep trees are rare. If the max_depth is set to be too small, then it can even lead to underfitting due to lack of flexibility to deal the different leaf node. As for max_features, Titanic training set has only 9 features, so tuning max_features will easily lead to underfitting. The min_impurity_decrease is similar to min_sample_split to some extent. But it is hard to estimate the range of the parameter to start tuning, while min_sample_split is intuitively bounded by the population of each

class. Figure 4.1(a, b) show the learning curve of trained DT with optimal parameter. The fitting seems fine since the gap between training and validation curve is getting smaller; and both of the curves converge to similar values.

The Gradient Boosting (GB) is tuned with number of trees (=30) and max_depth (=4). Due to limited amount of training data, the GB classifier turns to be overfitting with large number of base classifiers. In fact, cross validation for tuning of n_trees tend to favor small number of trees. However, the trained model with selected parameters (Figure 4.1(c, d)) tend to be underfitting. This might be rooted in the limited size of data – the cross validation for parameter tuning is done with multiple (=5) folds, which makes each fold of data not representative enough. But in the final model training procedure, relatively more data (than the validation fold) is used for training, which leads to underfitting with the parameters selected in the tuning step.

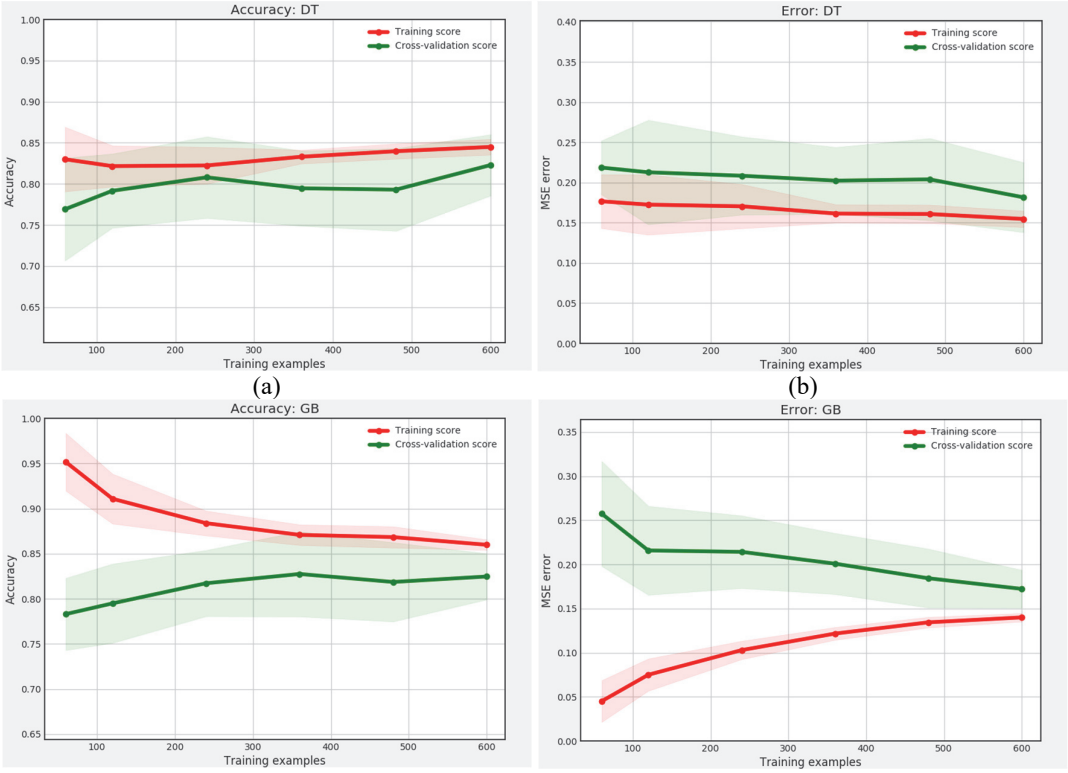


Figure 4.1: Cross validation for assessment of trained (a-b) DT and (c-d) GB with Titanic dataset.

2) Digits: DT is tuned with min_samples_split (=4) and max_depth (=10). Since there are many features (784) and they did not go through PCA before training; some randomness is also mixed into the splitting at each node. This sort of randomness also helps preventing overfitting. All the tuned parameters result in seemly good fitting (Figure 4.2), though DT, inherently, is not strong enough to handle the MNIST dataset.

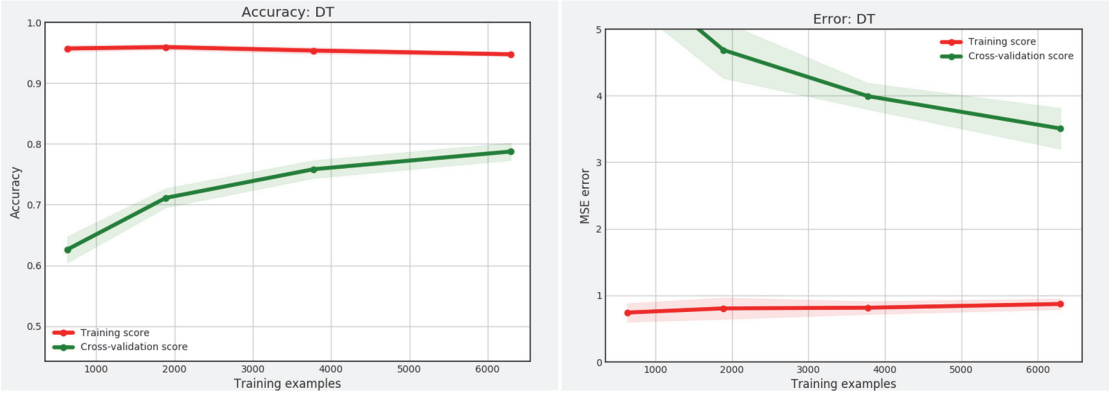
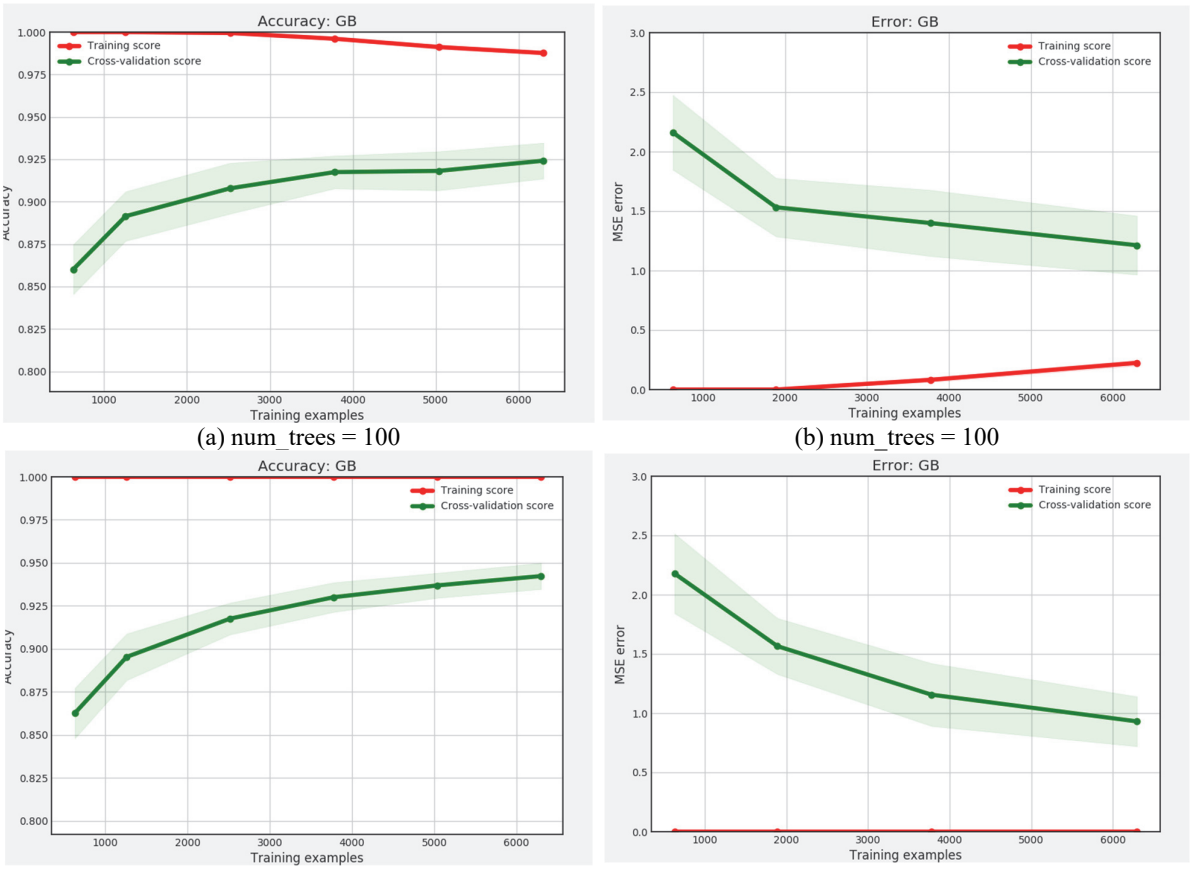


Figure 4.2: Cross validation for assessment of trained DT with Digits dataset



(a) num_trees = 100 (b) num_trees = 100

(c) num_trees = 400 (d) num_trees = 400

Figure 4.3: Cross validation for assessment of (a-b) GB with default number of trees (=100), and (c-d) GB with tuned parameter (number of trees =400) with Digits dataset

Testing.

The trained models are tested with testing data set for both Titanic and Digits problems. The results can be found in Table 4.1. The performance of GB classifiers is improved for both Titanic and Digits dataset. The performance of DT classifier is improved in Titanic case, but not improved in Digits case. Another observation is that GB classifier outperforms the DT classifier in both cases. In the titanic case, the performance tuning of GB turns to be difficult due to limited available data. In the Digits case, with sufficient data, GB can give relatively good performance. On the other hand, DT suffers even more issues in complex cases (i.e. large amount of features).

Time for training of DT classifier closely related to complexity of problem – the more (correlated) features, the longer it needs to finish the training. On the other hand, GB generally needs more time for training than DT. The amount of time for training highly related to number of base classifiers combined.

Table 4.1: Testing performance of trained models W. and W/O. cross validation (CV) for parameter tuning.

Dataset	**Classifier**	**W/O. CV**	**W. CV**					
		Accuracy	**Accuracy**	**F1**	**AUROC**	**Iterations**	**Training time**	**Testing time**
Titanic	DT	79.82 %	81.61 %	0.713	0.775	/	0.000725 s	0.000114 s
	G-Boosting	81.17 %	82.06 %	0.712	0.773	30	0.0361 s	0.000428 s
Digits	DT	78.37 %	78.23 %	0.780	0.878	/	1.122 s	0.0045 s
	G-Boosting	92.57 %	94.77 %	0.947	0.971	400	45.180 s	0.566 s

5. Artificial Neural Networks Classifier

Data-preprocessing.

1) Titanic: The same preprocessing procedures for DT is used for NN.

2) Digits: The same preprocessing procedures for DT is used for NN. Worth noting that no PCA is used.

Neural Network structures.

1) Titanic: In Titanic problem, a two-layer NN is trained. The NN has 16 neurons in the first hidden layer and 32 neurons in the second hidden layer. The numbers of neurons are chosen to be in such order that let the details are learned gradually between layers. Both the two layers use ReLU activation function, which is randomly initialized in a "glorot uniform" fashion. A dropout (=30%) is performed right after each hidden layer to prevent overfitting. The output layer uses the sigmoid activation function to predict the probability of each of the two classes. The optimizer is chosen to use Adam optimizer with learning rate optimized to 0.005.

2) Digits: A 3-layer convolutional neural networks (CNN) is built for classification of hand-writing images. The three hidden convolutional layers use 32, 64, 128 neurons in the first, second and third layer, respectively. In order to keep the model more robust (in terms of overfitting), max pooling is performed right after the first and second convolutional layers. In addition, dropout procedure is also added between each layer for better robustness. All the three hidden layers use ReLU activation function. At the last layer, softmax activation function is used to predict the probability of each class. The RMSprop optimizer is used for the Digits case.

Learning curve and cross-validation.

1) Titanic: The tuning parameters are the learning rate and the training epochs (iterations). By trying different combinations of the two parameters, the performance can be tuned. The learning curves are shown in figure 5.1. It is very interesting to see that the cross-validation (performance vs. training size, Figure 5.1a, b) curves show contradictory information about overfitting compared to the training process curves (Figure 5.1c, d). The training process curves indicate that the training curve and validation curve basically have the same trend – it is a good fitting. The difference might be rooted in the difference between training data and the testing data. Again, since the total amount of data is not sufficient enough, the data drew into each validation or testing partition may not be representative and can be quite different.

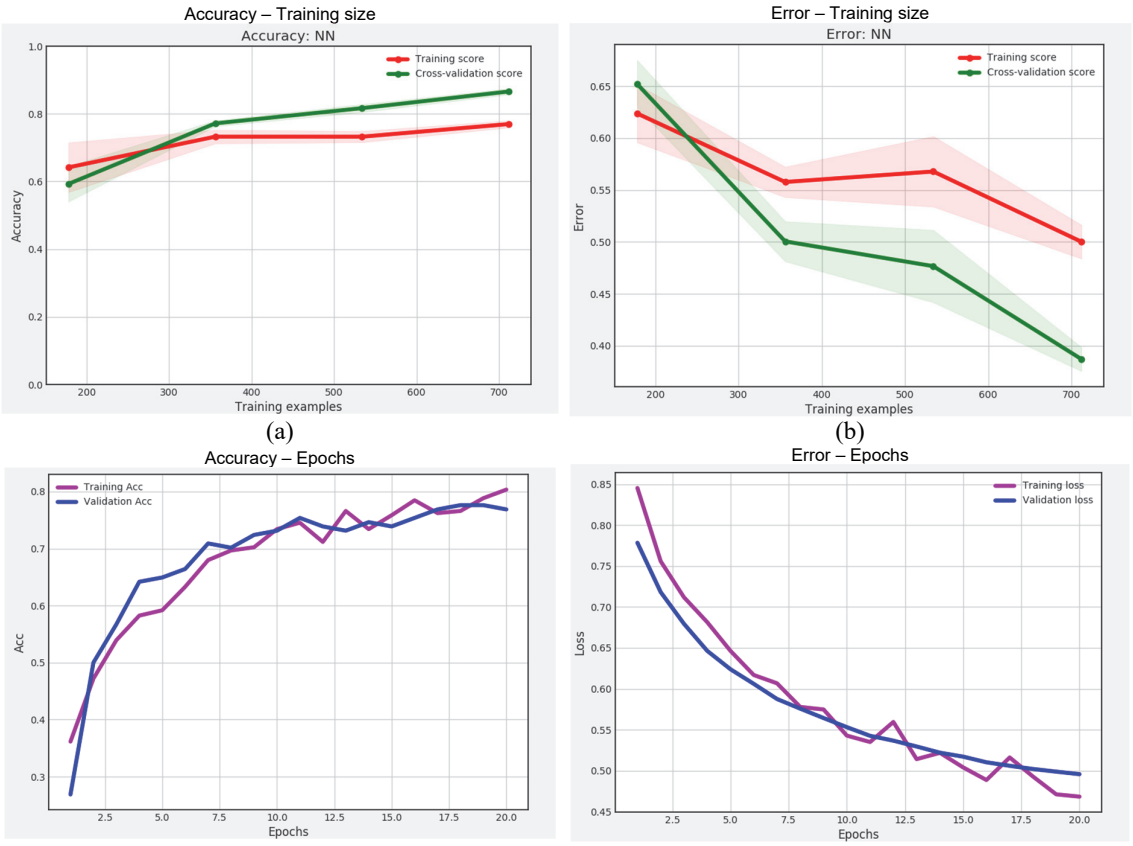
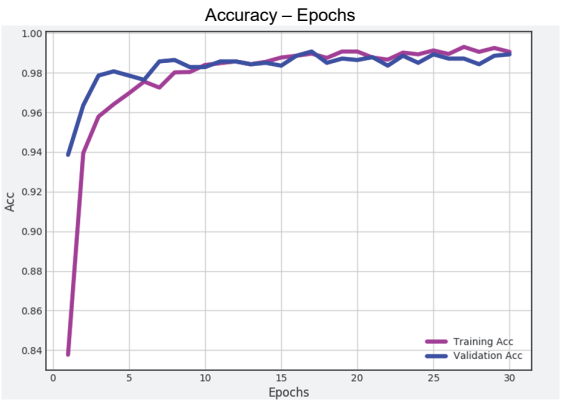
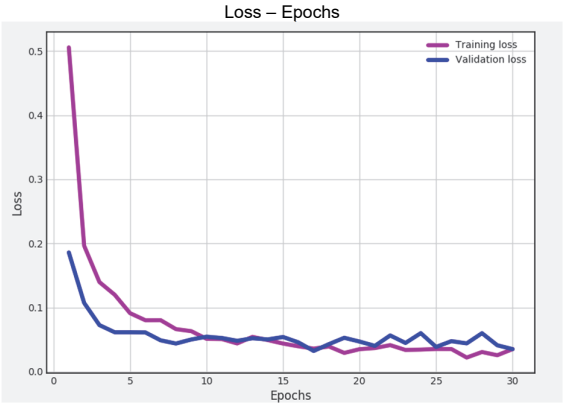


Figure 5.1: Cross-validation for NN (with optimal parameter) for Titanic dataset.

2) Digits: The learning curves are shown in figure 5.2. It can be concluded from both the accuracy curve and the error curve that the trained model is good – not underfitting since both lines are converging to good metrics; not overfitting since gaps between validation and training are getting smaller in both (a) and (b). Furthermore, the two learning curves basically follow the same trend.



(c) Digits dataset – Cross validation (accuracy)



(b) Digits dataset – Cross validation (loss)

Figure 5.2: Cross-validation for CNN (with optimal parameter) for Digits dataset.

Testing.

The trained models are tested with testing data set for both Titanic and Digits problems. The results can be found in Table 5.1. It can be concluded that the performance of SVM classifier is improved with cross-validation for parameter tuning in Digits dataset, while not helpful in Titanic dataset. The difference of performance between Titanic SVM and Digits SVM is mainly due to difference of available training examples. The time of training for linear kernel is much larger than that for RBF kernel in the Titanic case, due to slower convergence. But the time of testing for linear kernel is much less in the Titanic case. In the Digits case, both training time and testing time of linear kernel are much less than that for RBF kernel. Thus, it can be clearly seen that linear kernel requires much less computation than RBF kernel.

Table 3.1: Testing performance of trained NN.

Dataset	Accuracy	Epochs	Training time	Steps per epoch	Time per step	Testing time
Titanic	78.7 %	20	3.679 s	534	~200 us	0.313 s
Digits	98.90 %	30	652.647 s	5600	4 ms	2.606 s

6. Comparisons of Classifiers

Performance.

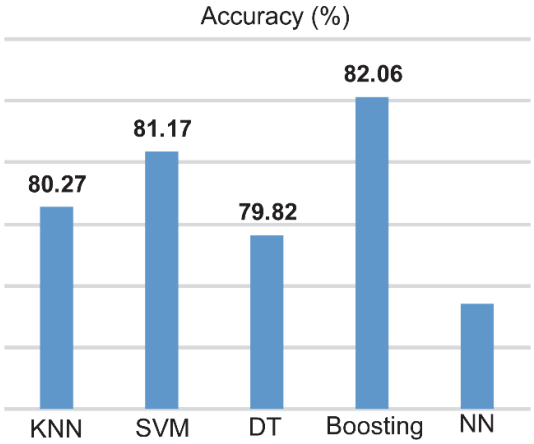


Figure 6.1: Performance comparison of classifiers for Titanic dataset.

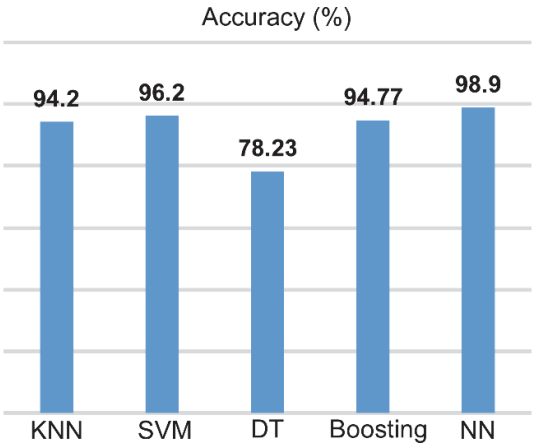


Figure 6.2: Performance comparison of classifiers for Digits dataset.

Execution time.

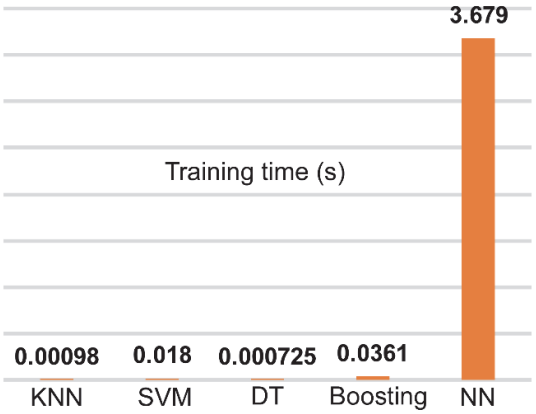


Figure 6.3: Execution time comparison of classifiers for Titanic dataset.

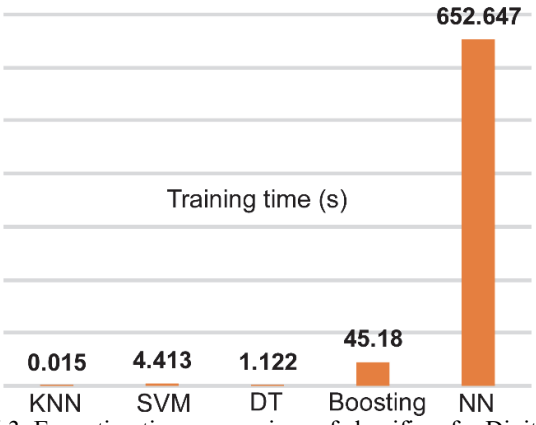


Figure 6.3: Execution time comparison of classifiers for Digit dataset.

Final comments.

1) Dataset size affects a lot. Small dataset favors less complex methods/ models.
2) Quality of data (i.e. balance between classes) also affects the training/ tuning procedures.
3) Binary classification is easier than multiclass classification. Better tuning required for multiclass classification.
4) Neural Network boost performance when sufficient data is available.