Vivek Murali
CS-4641
23rd September 2018

Analysis of Supervised Learning Techniques

Introduction:

This report explores the performance and characteristics of supervised learning techniques, which include: Decision Trees, Neural Networks, Boosting, Support Vector Machines and k-Nearest Neighbors. In order to determine the behavior of each algorithms, experiments were performed with various parameters of each algorithm.In the following pages, you will find detailed analyses of five different supervised learning algorithms. For example, I will explore the relationship between the values of various hyperparameters and their corresponding algorithms. However, there are a number of things that will remain invariant throughout the analysis in order to isolate and focus on the topic of analysis; in the previous example, that would be the hyperparameters. One invariant is the datasets used, as described below:

Dataset Description:

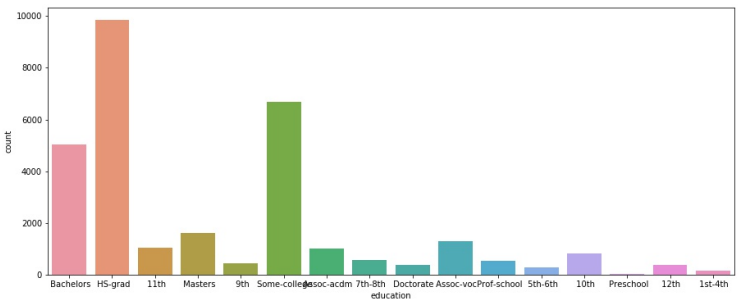
Census :-

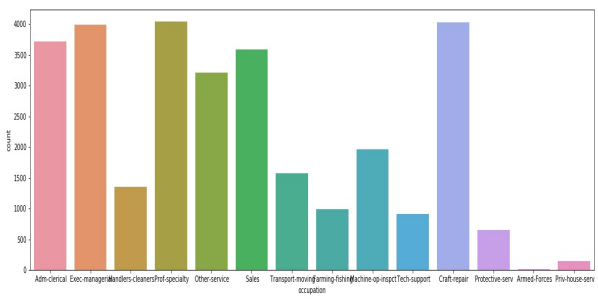
Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset.This data was extracted from the census bureau database found at http://www.census.gov/ftp/pub/DES/www/welcome.html. Prediction task is to determine whether a person makes over 50K a year. The weights on the CPS files are controlled to independent estimates of the civilian noninstitutional population of the US. These are prepared monthly for us by Population Division here at the Census Bureau. We use 3 sets of controls.

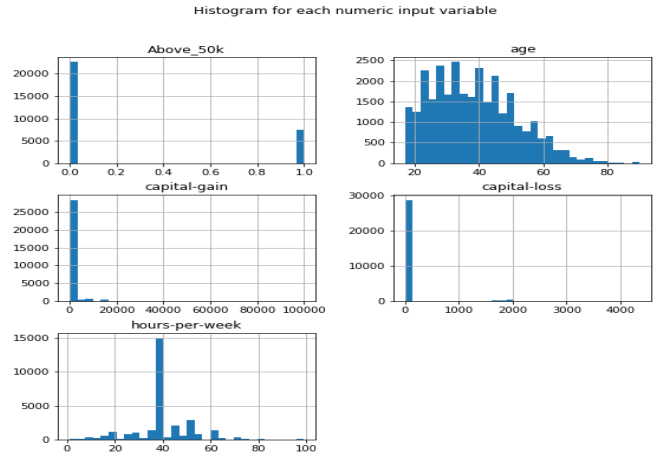
These are:

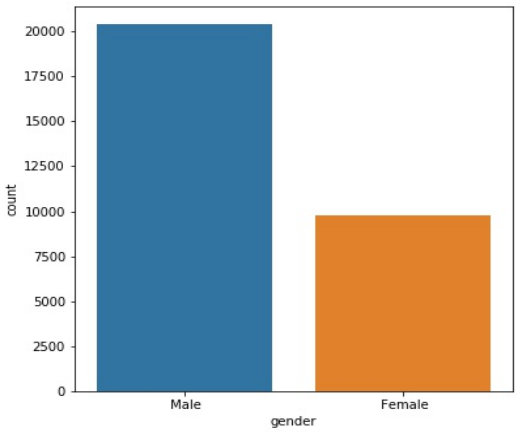
1. A single cell estimate of the population 16+ for each state.
2. Controls for Hispanic Origin by age and sex.
3. Controls by Race, age and sex.

We use all three sets of controls in our weighting program and "rake" through them 6 times so that by the end we come back to all the controls we used. The term estimate refers to population totals derived from CPS by creating "weighted tallies" of any specified socio-economic characteristics of the population. People with similar demographic characteristics should have similar weights. There is one important caveat to remember about this statement. That is that since the CPS sample is actually a collection of 51 state samples, each with its own probability of selection, the statement only applies within state. 15 set of attributes and more than 32,000 instances. With 20 % test and 80% training split.





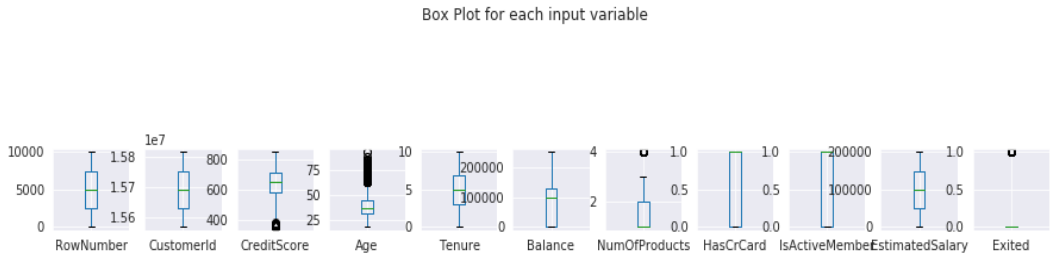


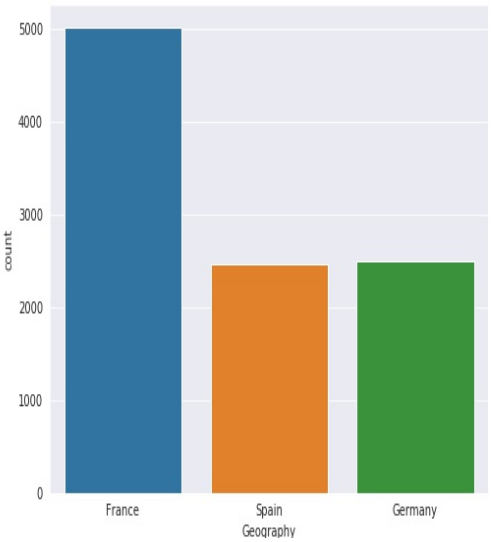


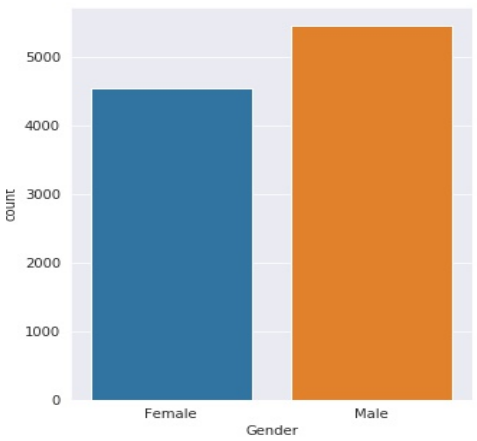
Churn Modeling :-

Predict whether customer exits the bank based on bank chun-modeling data. Also known as "Bank Churn-Modeling" dataset. This data was extracted from the Kaggle database found at https://www.kaggle.com/hj5992/churn-modelling/data This dataset in very interesting. Successful detection. Churn Modeling dataset consist of 13 attributes and 10000 instances. With 20% test set and 80% training set .

In contrast to the Census dataset, The Churn-Modeling dataset only has 10,000 instances and two are categorical and have enum values. Comparison between experiments performed on Churn-Modeling dataset and Census dataset can demonstrate how algorithms behave differently on different types of dataset. Also, the difference in size can also demonstrate how size can influence the performance of classifiers.







Decission Trees:

Decision tree learning is a highly intuitive supervised learning algorithm that attempts to take a dataset and infer a number of rules or questions, corresponding to if then else constructs, wherein the root node is an initial question, which based upon a result allows one to progress farther down the tree, asking more and more questions and coming closer to an answer. The algorithm is fairly simple to understand, but can be unstable, creating completely different trees for even the most minute changes to the data.

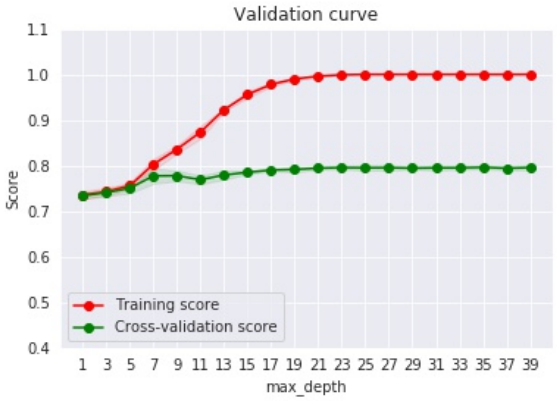


fig 1.1(A)Churn Modeling

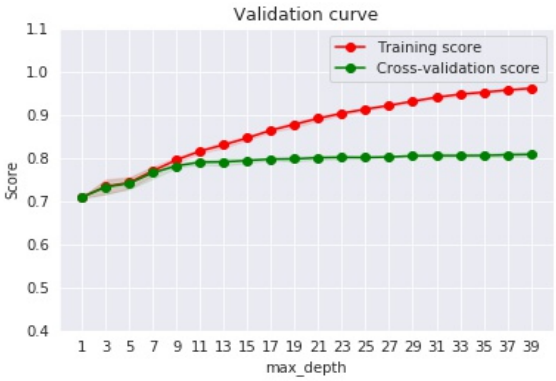


fig 1.1(B)Census

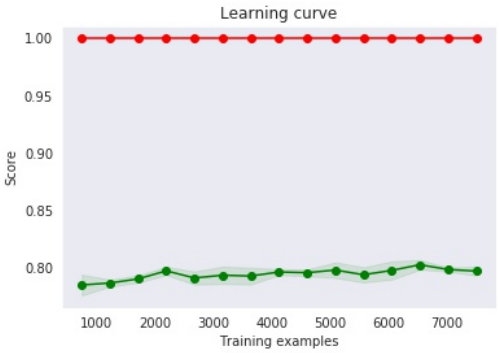


fig 1.2(A) Churn Modeling

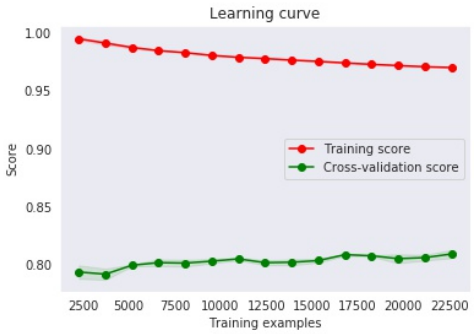


fig 1.2(B)Census

Dataset Name	Cross Validation Accuracy on Training Set	Leaves	Tree Size	Training Time	Predection Accuracy Score	Cross Validation Accuracy on Test Set
Churn-Modeling	86	5	5	0.025808s	84.95	85
Census	84	5	5	0.126222s	84.20	84

For implementing Decision trees for the given datasets the DecisionTreeClassifier from sklearn library was used the parameter what was used was the same given above max depth is 5 and max sample leaves is also 5 was some form of pruning(We implement pruning for a decision tree algorithm in order to reduce the complexity of the tree and therefore overfitting. Pruning for the below analysis was done by enforcing a max depth) and by default the information gain in the DecisionTreeClassifier class is gini index. Here in the fig 1.1 it shows the validation curve of both datasets the curve starts with the max_depth of 1 till 39 on the x axis and on the y axis there is the scores where are from 0.4-1.1 this are in terms of percentage and fig 1.2 above consist of learning curves of both the datasets which consist of the training examples on the x-axis and accuracy scores on the y-axis. Above we can see the effect of pruning on the accuracy of our decision tree algorithm, which seems to have a same effect on each of our datasets. One will quickly note that while the algorithm's accuracy on both dataset are incereased until to a certain max depth and remain constant as max depth increases.

Neural Networks

Neural networks represent a very powerful supervised learning algorithm, with incredible complexity and predictive power. For the purposes of our analysis, we use Keras library to create a Artifical Neural Netwok , a feedforward neural network model through a weighted directed acyclic graph, wherein the edges are reinforced or reduced by each piece of data that is added to be learned from. Where keras library what i used Tensor flow at backend to create this neural network, The model creation were carried out with 100 epoches each to test the accuracy of the model. The activation functions such as relu, sigmoid functions were used, the relu activation function were used for the input layer and each hidden layer were added to the input layer, 3 hidden layers were used to train both the models And sigmoid functions were used in the output layer. The use of this particular activation fuctions in this models as relu works best for hidden layers and sigmoid works

best for output layer, best in terms of accuracy and time. The rest of the hyperparameters are kept default to train the model.

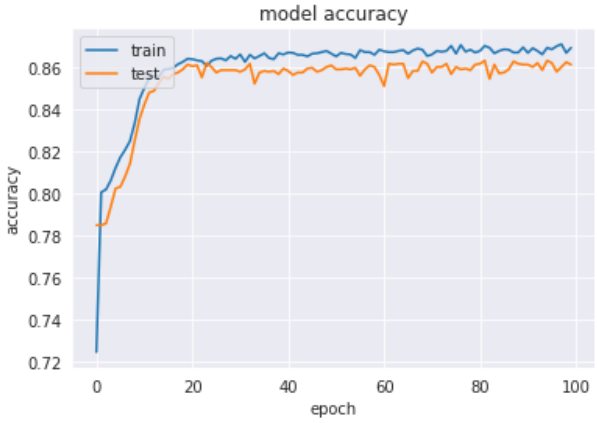


fig 2.1(A) Churn Modeling

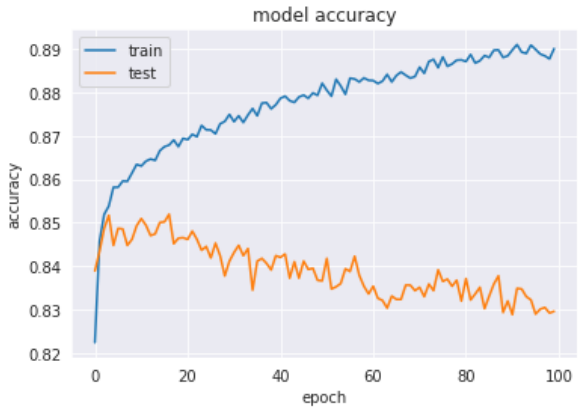


fig 2.1(B)Census

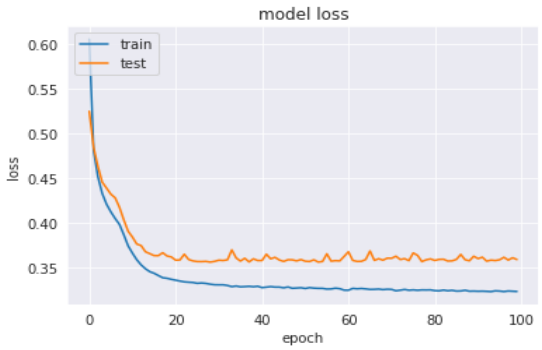


fig 2.2(A) Churn Modeling

fig 2.1(B)Census

Dataset Name	No of Epoches	No of Hidden Layers	Input Dimenssions	Training Time	Predection Accuracy Score	Accuracy
Churn-Modeling	100	3	61	191.467468s	85.65	86.01
Census	100	3	12	666.537623s	83.22	87.48

In fig 2.1 above shows the model accuracy of neural network in respective datasets. In Churn Modeling dataset as the number of epoches increases the model accuracy gradually increases whereas in Census dataset as the number of epoches increases the accuracy of the model decreases, Perphas due to Census dataset consist of more categorical data than Churn-modeling and also consisting noise in the data. In fig 2.2 its a comparision of both datasets showing the model loss in increasing epoches in Churn-Modeling dataset the loss decreases as the increasing epoches whereas in Census dataset the loss increases with the increasesing of epoches this may due to less number of consisting nodes and hidden layers in the Artifical neural network.

Adaptive boosting

We now experiment with boosted decision trees, in particular adaptive boosting, or AdaBoost, as included in the scikit library. AdaBoost, a modified decision tree algorithm, has a hyperparameter in

the the number of estimators, which limits the maximum number of estimators allowed before boosting ends. We experiment with this number below, at the 5 max depth.

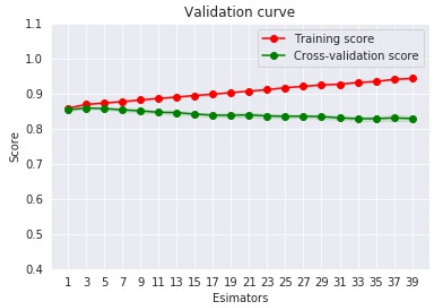


fig 3.1(A) Churn Modeling

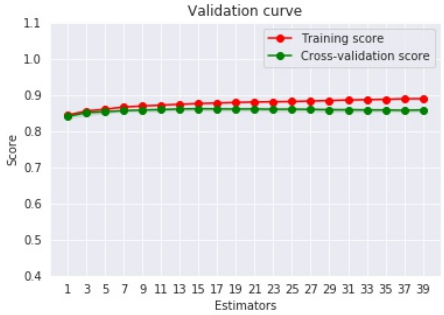


fig 3.1(B)Census

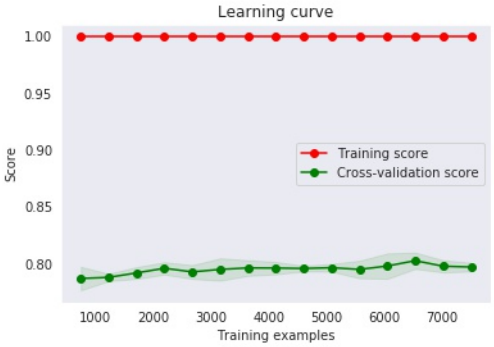


fig 3.2(A) Churn Modeling

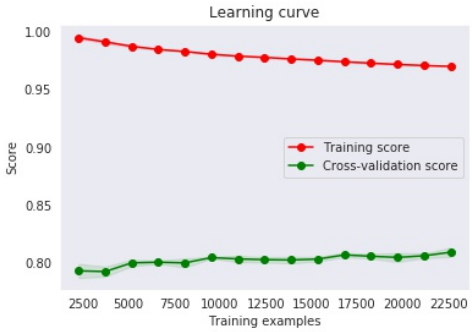


fig 3.2(B)Census

Dataset Name	Cross Validation Accuracy on Training Set	Estimators	Learning rate	Training Time	Predection Accuracy Score	Cross Validation Accuracy on Test Set
Churn-Modeling	96	50	1	1.220123s	82.1	82
Census	89	50	1	7.557173s	85.64	86

Analyzing the effect of the number of estimators on our accuracy yielded no useful insights; it seemed to have almost no effect on the number of estimators. Therefore, we can only recommend the default value for the number of estimators, which is 50. We thus move on below to analysis of pruning on an adaptively boosted decision tree. For the same reasons as for our nonboosted decision tree algorithm, and to maintain consistency, we again use max_depth as our hyperparameter of choice to implement pruning. In fig 3.1 shows the effects of increasing estimators in the Adaboost Classifiers it has the same effect on both the datasets with learning rate to be 1 and rest of the hyperparameters kept to defaults for the same Decission Tree.With the above, we can clearly see asymptotic trends on accuracy as we increase our maximum depth. Since ideally, we'd like to minimize overfitting, and therefore our model's susceptibility to unseen data, we would ideally pick maximum depth values which correspond to the inflection points in the graph where the curves begin to become asymptotic. For the both the dataset, this corresponds to a max_depth of 5.

k-Nearest Neighbors

K-Nearest Neighbors (kNN) is a non-parametric method used for classification. kNN is also instance-based learning where the function is only approximated locally and all computation is deferred until classification. For each row (observation) in the testing dataset, the distance between its position and each row in training dataset is calculated.a comparatively lazy algorithm compared to the others discussed in this paper, but performs excellently in a number of problem domains for its simplicity. The algorithm makes predictions for a point of given input by looking at the k nearest data points and making a decision on what to predict based off of those k nearest points. The potential for tuning, then, and thus the hyperparameter, lies in the value of k that is used for a particular dataset.

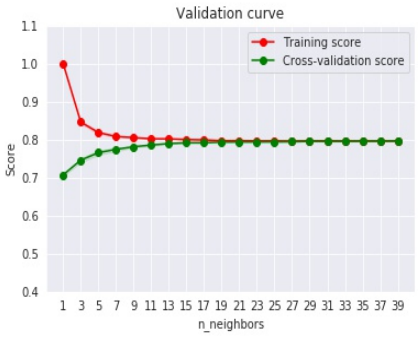


fig 4.1(A) Churn Modeling

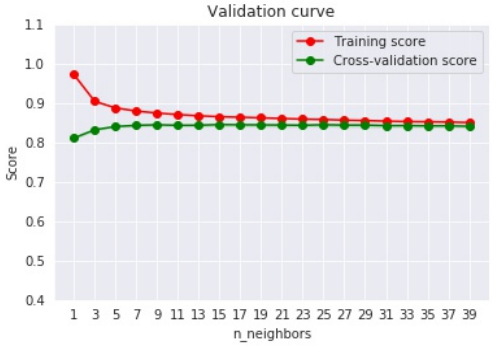


fig 4.1(B)Census

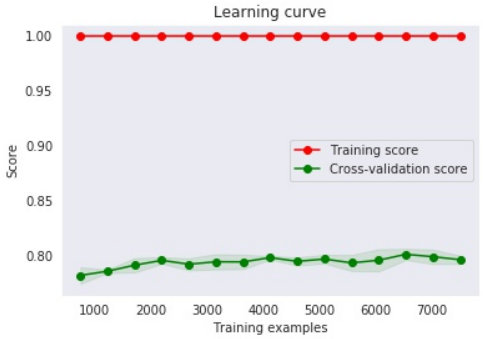


fig 4.2(A) Churn Modeling

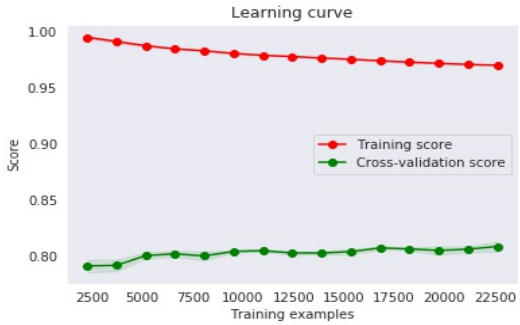


fig 4.2(B)Census

Dataset Name	Cross Validation Accuracy on Training Set	K	Training Time	Predection Accuracy Score	Cross Validation Accuracy on Test Set
Churn-Modeling	82	5	0.040148s	76.3	76
Census	89	5	1.550702s	84.25	84
Churn-Modeling	85	3	0.025859s	75.4	75
Census	90	3	1.427757s	83.05	83

The above fig 4.1 demonstrates the relationship of k to the accuracy of our predictions. For the Churn-Modeling, accuracy scores in every category converged to about 82%, so the point at which they became near-indistinguishable from each other determined my value of k for that one in particular. Thus, the value of k I picked for the coding survey was 5 . On the other hand, for our Census dataset, the values do converge, but at a lower accuracy than is potentially possible. In fact, there is a small local maximum at around k=3, after which overfitting increases accuracy asymptotically to about 90%. This local maximum, however, gets us about 83%, a substantial improvement compared to that overfitted 75%. Thusly, the value of k I selected for the Census dataset was 5.

Support Vector Machines

Support Vector Machine (SVM) is a discriminative classifier defined by a separating hyperplane. Given the labeled training data, SVM outputs an optimal hyperplane that categorizes new examples for the final supervised learning algorithm the support vector machine,As that categorizes the training data by finding the best fit parameters for a specifically chosen kernel function that will try to map various spaces of the data as belonging to a certain category. The support vector machine we will be using is scikit's SVC implementation, of which we have picked 2 different kernel functions to test, the radial basis function kernel (RBF) and the poly kernel. This choice of kernel will be our hyperparameter to test with our datasets.

RBF Kernel

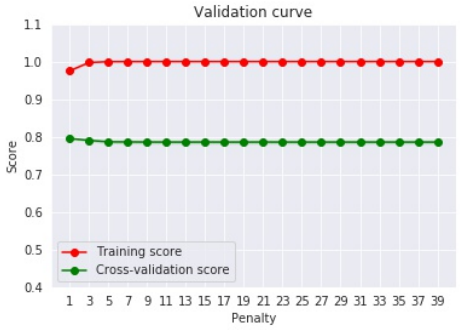


fig 5.1(A) Churn Modeling

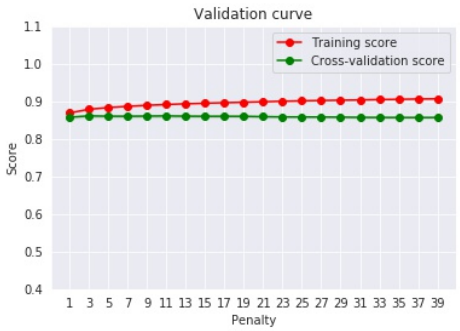


fig 5.1(B)Census

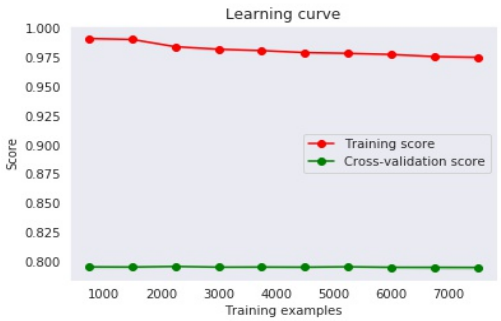


fig 5.2(A) Churn Modeling

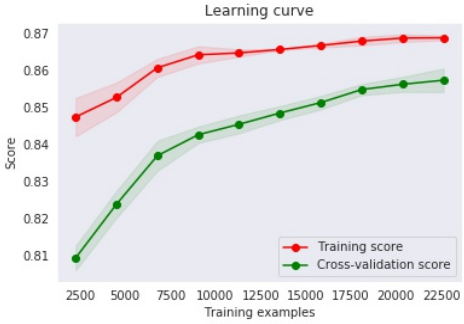


fig 5.2(B)Census

Poly Kernel

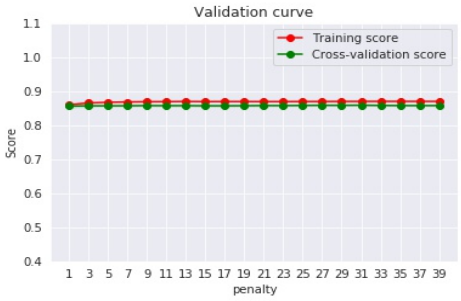


fig 5.3(A) Churn Modeling

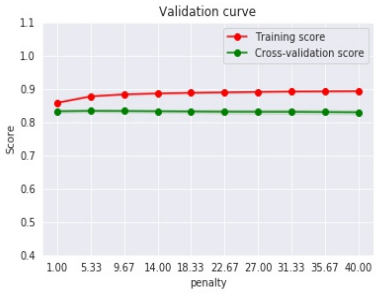


fig 5.3(B)Census

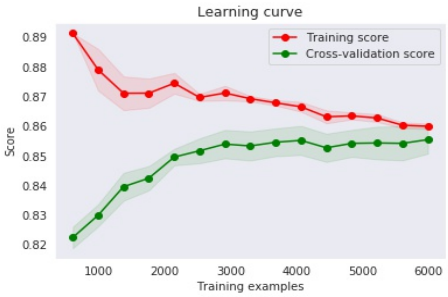


fig 5.4(A) Churn Modeling

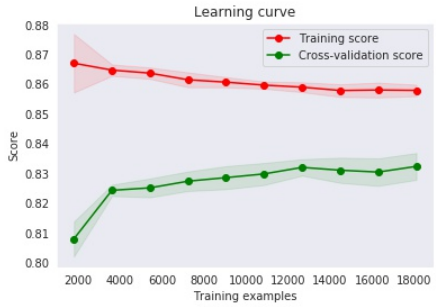


fig 5.4(B)Census

Dataset Name	Cross Validation Accuracy on Training Set	Kernel	Training Time	Predection Accuracy Score	Cross Validation Accuracy on Test Set
Churn-Modeling	97	RBF	5.318115s	79.9	80
Census	87	RBF	77.353230s	85.91	86
Churn-Modeling	86	Poly	2.023063s	85	85
Census	86	Poly	44.928418s	83.40	83

From the graphs on the above, we can see that the kernel choice seems to have a variety of effects on the dataset. To an untrained observer, it may be difficult to pick between the poly and RBF kernels, as they seem to have around the same accuracy, with the RBF having a much higher training score. However, a larger training score does not offer the same benefit, as say, a higher cross validation score would, since a larger training score may simply imply a large amount of overfitting. It is for this reason that the poly kernel was my kernel of choice for the Churn-Modeling dataset. For the Census dataset however, it is much more clear, and RBF is picked handily over the other kernels, reliably performing substantially better in all scores.

Conclusion

From the analysis above, we can see that the performance of classifier is determined by the selection of hyper-parameters and the nature of dataset. In Census dataset, there is always some level of fluctuation in the performance curve that suggests overfitting while in Churn Modeling dataset, the performance curves are much smoother. This could also attribute to the nature of the datasets: Census contains more noise since the data is derived from surveys, while in Churn Modeling, since all the attributes are enums, there may exist some contracting samples that confuses the algorithms.Perhaps a change in choice of attributes was in order, or what we were predicting, but it seemed like there was so much noise and interference and bias and whatnot that it was at risk of clouding the entire analysis. Fortunately, it did not, and while there were definitely some issues, I was still able to glean a number of insights about the data and about the algorithms themselves. In the future I will be able to take all these insights and the potential shortcuts that I've now devised in picking better data sets and better methodology for analysis of algorithms.

Now allow us to aggregate all of our results for each of the algorithms in the table below:

Algorithms	Decision Trees		Support Vector Machines		Neural Networks		kNN		Adaptive boosting	
	Training Time	Cross Validation Accuracy on Test Set	Training Time	Cross Validation Accuracy on Test Set	Training Time	Cross Validation Accuracy on Test Set	Training Time	Cross Validation Accuracy on Test Set	Training Time	Cross Validation Accuracy on Test Set
Churn-Modeling	0.025808s	85	2.023063s	85	191.467468s	86.01	0.025859s	75	1.220123s	82
Census	0.126222s	84	44.928418s	83	666.537623s	87.48	1.427757s	83	7.557173s	86

One thing we will note right away about the results of values in the table is that our dataset clearly exhibits a large bias. That is to say, the accuracies for all of our algorithms seems to converge towards a single asymptote at around 80%; adding more data will not improve the accuracy of our algorithms very much. Neural Networks accuracy seems to be the best compared to others in the both datasets with over 85% of accuracy though it takes longer time to train compared to others. Decision Tree for Churn-Modeling dataset seems to be the best algorithm in terms of training time and accuracy with 85% of accuracy and 0.02 seconds of train time where in Census Adaboosting would be the best in terms of training time and accuracy with 86% of accuracy and 7 seconds of training time. However, the training time of Neural Network is much higher than all the other algorithms. Thus, if in the future, the Churn-Modeling wants to apply the system to larger dataset, Decission Tree may be a better option. If accuracy is more important than the time cost, then Neural Network is a better option. It goes the same with Census if accuracy is more important than the time time cost preferably Neural Network would be the best choice, if time is also considerd the Adaboost will be a better option.