Weak Pattern Theory

Project report

COSC 6342: Machine Learning

Submitted by

S M Salah Uddin Kadir (1800503)

Rubayat Jinnah (1891217)

Let's classify any dataset with perfect accuracy! In this paper, we are going to propose a theory that can be used to classify a dataset to get the very close of perfect accuracy. We know, strong always ***dominates*** the weak by nature. When we build a classifier from a dataset, the classifier represents the maximum examples, ***strong pattern***, by dominating the other minor groups, ***weak patterns***, of examples. So, the idea is that if we can separate the weak pattern examples from the dataset and build a different classifier using them then it is possible to achieve a very good accuracy that is close to perfect.

To verify the idea, we have used the ***Census Income Data Set*** [1] where the accuracy of the dataset using a single classifier is not very good. The dataset is very famous, and lot of researchers have already worked on it. At first, we will have a survey on their work.

Relevant Work

In the paper, " An Empirical Evaluation of Supervised Learning for ROC Area", Caruana showed a comparison of the AUC performance of seven supervised learning methods. In this paper, seven algorithms were implemented and AUC was used as a performance metrics. SVM, neural network, decision trees, k-nearest neighbor, bagged trees, boosted trees and boosted stumps are the seven algorithm that were used. In the paper it is showed that Boosted trees have the best average AUC performance, followed by bagged trees, neural networks and SVMs. And finally, even better AUC performance was found in their ensemble selection method. Their Ensembles were built with forward stepwise selection. The Ensemble model that was proposed by Caruana had highest mean value and outperformed all other seven models[2].

In another paper, "Ensemble Selection from Libraries of Models" Rich Caruana did similar work. He showed a model for ensemble selection that optimized the performance. Caruana concluded that ensemble selection consistently finds ensembles that outperform all other models, including models trained with bagging, boosting, and Bayesian model averaging.[3]

Naïve Bayes algorithm had a higher accuracy on many classification task as well as decision tree but it could not scale up with larger database. In the paper "Scaling Up the Accuracy of Naïve Bayes Classifier: a Decision Tree Hybrid" written by Ron Kohavi, the NBTree is described as a hybrid of Naive Bayes and decision tree. NBTree showed better performance than individual performance of Naïve Bayes and

Decision tree specially for larger database. The error rate of NBTree was 14.01% which was one of the best .[4]

The dataset has a “Data Set Description” link that contains a list of seventeen algorithms ran on the dataset and the results. One of those algorithms is Naive Bayes with 16.12% error rate. Another study used Naive Bayes algorithm but omitted the entries with unknown values and had a higher error rate of 20.43%. In another study, it was concluded that a Log Transformed Linear Regression is best model for prediction in situations like this.[5]

In another study**, “**Predicting if income exceeds $50,000 per year based on 1994 US Census Data with Simple Classification Techniques **”**feature engineering was implemented to identify the dominant features. Then naïve bias, logistic regression and decision tree algorithms were implemented for prediction. Test error rate for decision tree was 14.78%, naïve bias 20.432% and logistic regression 38.612% which was quite high. [6]

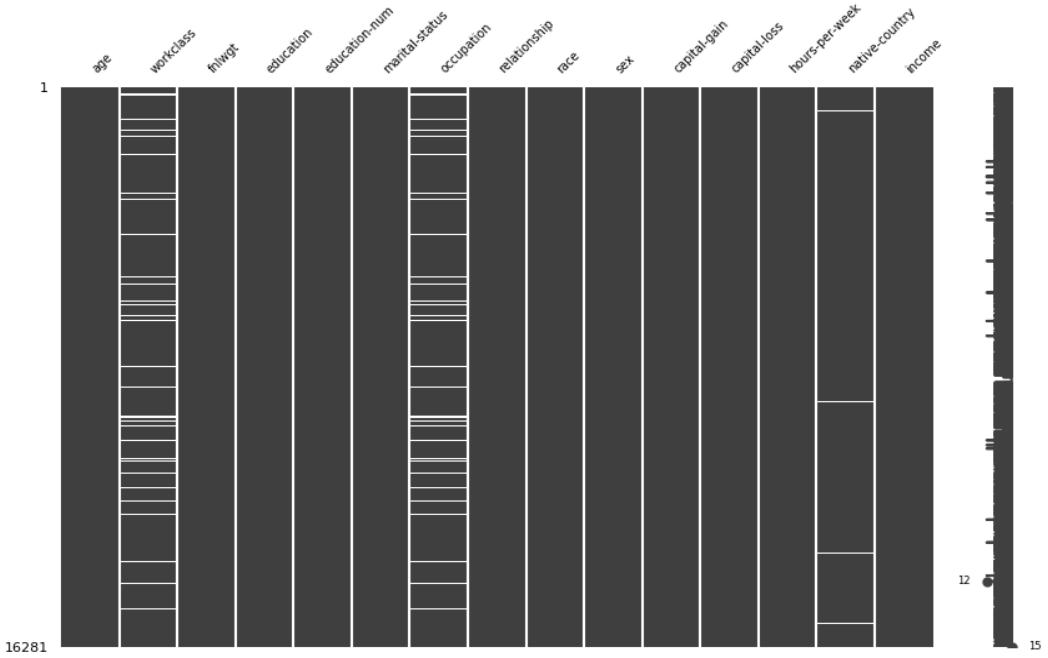
Some more similar study were conduct on the same data set. Multiple approach were taken with different algorithms to enhance accuracy. Not only that but also the parameters were also used for better performance. In our project we are trying to increase accuracy by removing weak learners and working on strong learners only.

For a quick overview, a comparison between some significant papers we studied, that used the same data set, but different algorithms and different implementation process and their performance is given below in the table.

Paper name	Algorithm	Performance	Comment
An Empirical Evaluation of Supervised Learning for ROC Area.	SVM, Neural network, Decision trees, k-nearest neighbor, Bagged trees, Boosted trees , Boosted stumps, Ensemble model.	Before using Ensemble model, Boosted trees have the best average AUC performance, followed by bagged trees, neural networks and SVMs. But Ensemble model outperformed all other models.	The performance was measured based on AUC performance metrics.
Ensemble Selection from Libraries of Models.	Bagging, Boosting, Bayesian model, Ensemble learning.	Ensemble selection outperformed all other models.	Performance measured on accuracy.
Scaling Up the Accuracy of Naïve Bayes Classifier: a Decision Tree Hybrid.	Naïve Bayes Decision Tree NBTree	NBTree outperformed the performance of individual Naïve Bayes and Decision Tree model.	The error rate of decision NBTree was 14.01% which was one of the best performance for this type of situation.
Predicting if income exceeds $50,000 per year based on 1994 US Census Data with Simple Classification Techniques.	**Naïve bias ,** **Logistic regression,** **Decision tree**	**Error rate for decision tree was** 14.78% , naïve bias 20.432% and logistic regression 38.612%	Performance was not so high in this approach.

Data set preprocessing

Our analysis of the dataset started with the preprocessing steps. I had a good effort to make the dataset suitable to use. The dataset has in total 48,842 (16281 test examples, 32561 training examples) instances with 14 different types of categorical and continuous attributes to predict the result that whether the income is greater than 50K or not. There are lot of examples in both training and testing dataset with missing values. Here, I have a plot on testing dataset where the white line represents the missing value of that example.



From the plot, we see that all the missing values are categorical type of attributes. We treated them in two ways,

- Removing the entire example, or
- Filling with the most frequent value of that attribute

We found that filling the missing value with the most frequent value gives the better result than removing the entire example. So, we used the later preprocessing option for the rest of the analysis.

The Analysis of the dataset using different estimators

Our used algorithms to analyze the dataset,

- Naïve Bayes,
- Logistic Regression

- k-nearest Neighbors
- Support Vector Machine
- Decision Tree and
- Neural Network

We also used,

- Bagging
- Boosting

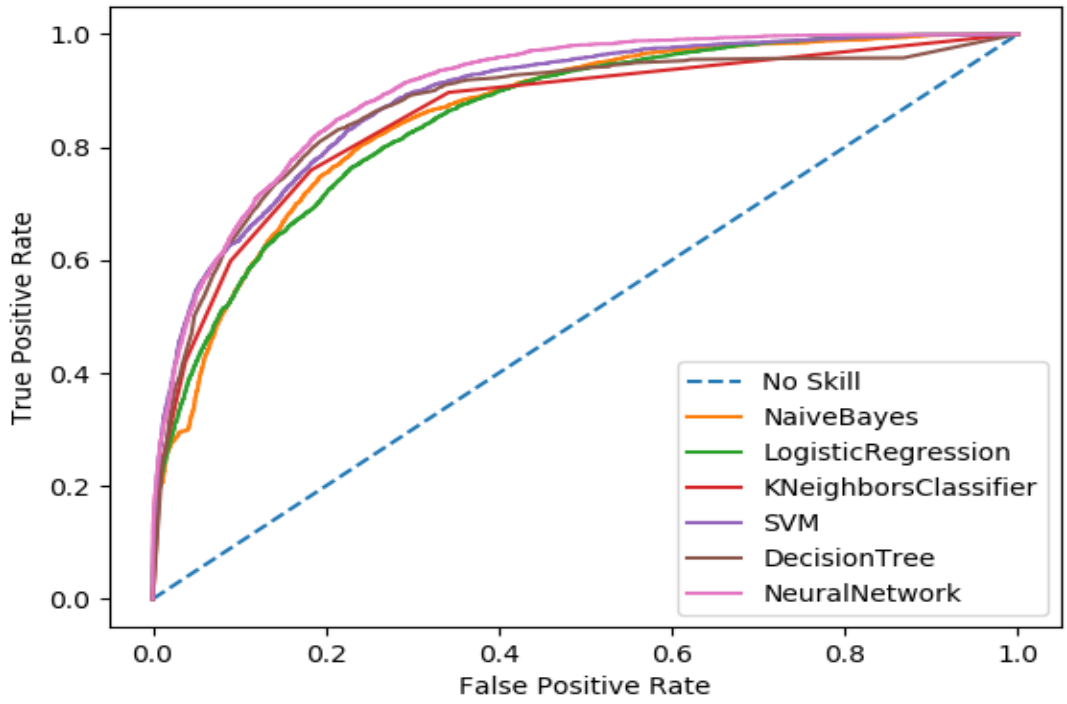
First, we will have a comparison table of performance applying those algorithms,

(*Please zoom in to read the table, sorry for the inconvenience. We made this to keep all things together for the easy comparison.*)

Comparison among all classifiers:

Classifier	TrainAccuracy	TestAccuracy	TrueNegative(00) <=50K	FalsePositive(01)	FalseNegative(10)	TruePositive(11) >50K	ROC_Score
NaiveBayes	0.8186480759190442	0.8215711565628647	11516	919	1986	1860	0.8554770192416061
LogisticRegression	0.8235312183286754	0.8248264848596524	11742	693	2159	1687	0.8513861575773847
KNeighborsClassifier	0.8838180645557568	0.8364351084085744	11318	1117	1546	2300	0.8558068780330625
SVM	0.8575289456712017	0.8534488053559364	11849	586	1800	2046	0.886259908779946
DecisionTree	0.8822210620066951	0.845771144278607	11551	884	1627	2219	0.8700676068860206
NeuralNetwork	0.8508645311876171	0.8520361157177078	11686	749	1660	2186	0.9012935595831554

And the corresponding ROC curve comparison,



From the table, we see that we have obtained the maximum test accuracy from SVM (85.34%) and the maximum ROC score from Neural Network that covers 90.13% of the plot.

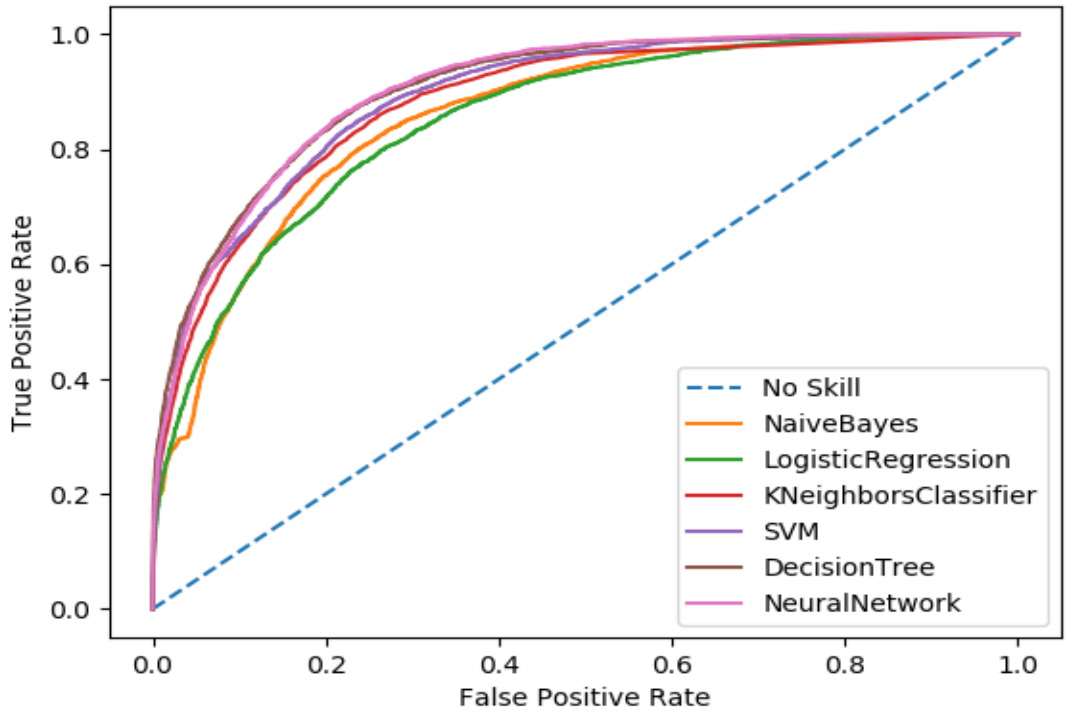
We wanted to check whether we can get a better result using Ensemble algorithms. We used Bagging and Boosting techniques,

The same comparison table using Bagging,

Comparison among all classifiers using Bagging:

Classifier	TrainAccuracy	TestAccuracy	TrueNegative(00) <=50K	FalsePositive(01)	FalseNegative(10)	TruePositive(11) >50K	ROC_Score
NaiveBayes	0.8189244802063819	0.8214483139856275	11521	914	1993	1853	0.8566135271064239
LogisticRegression	0.8233776603912656	0.8246422209937965	11734	701	2154	1692	0.8513036484467018
KNeighborsClassifier	0.8773686311845459	0.8414716540753026	11380	1055	1526	2320	0.8806335848126325
SVM	0.8575596572586837	0.8541244395307414	11847	588	1787	2059	0.8917647795578088
DecisionTree	0.8884862258530143	0.8554142865917327	11762	673	1681	2165	0.9040755245006744
NeuralNetwork	0.8508031080126531	0.85283459246975	11717	718	1678	2168	0.9033689381350888

And the ROC plot comparison,



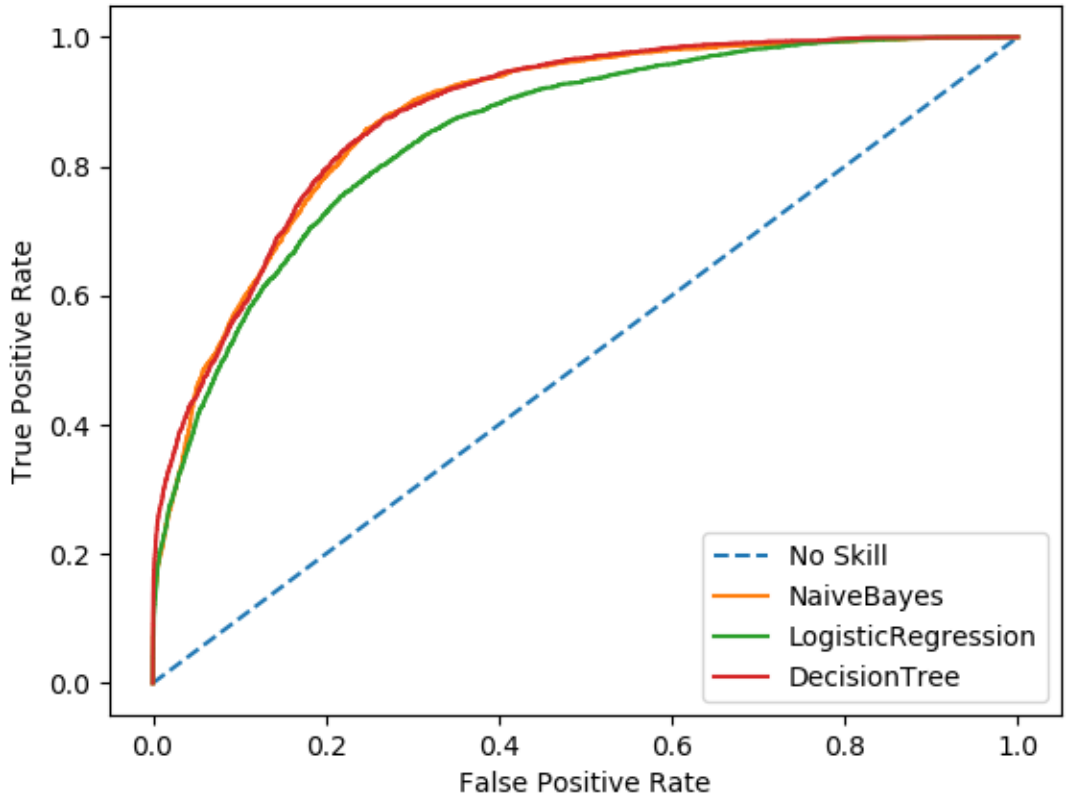
Here, we have got the best result from the decision tree with 85.54% accuracy on the test dataset where the ROC score is 90.49%.

And using **Boosting**,

Comparison among classifiers using Boosting:

Classifier	TrainAccuracy	TestAccuracy	TrueNegative(00) <=50K	FalsePositive(01)	FalseNegative(10)	TruePositive(11) >50K	ROC_Score
NaiveBayes	0.8307484413869353	0.8335483078434985	11557	878	1832	2014	0.8767109405727253
LogisticRegression	0.8060870366389239	0.8083041582212395	11486	949	2172	1674	0.8502477155781044
DecisionTree	0.9229753385952519	0.8369264787175235	11407	1028	1627	2219	0.8808825236000997

With the corresponding ROC curve,



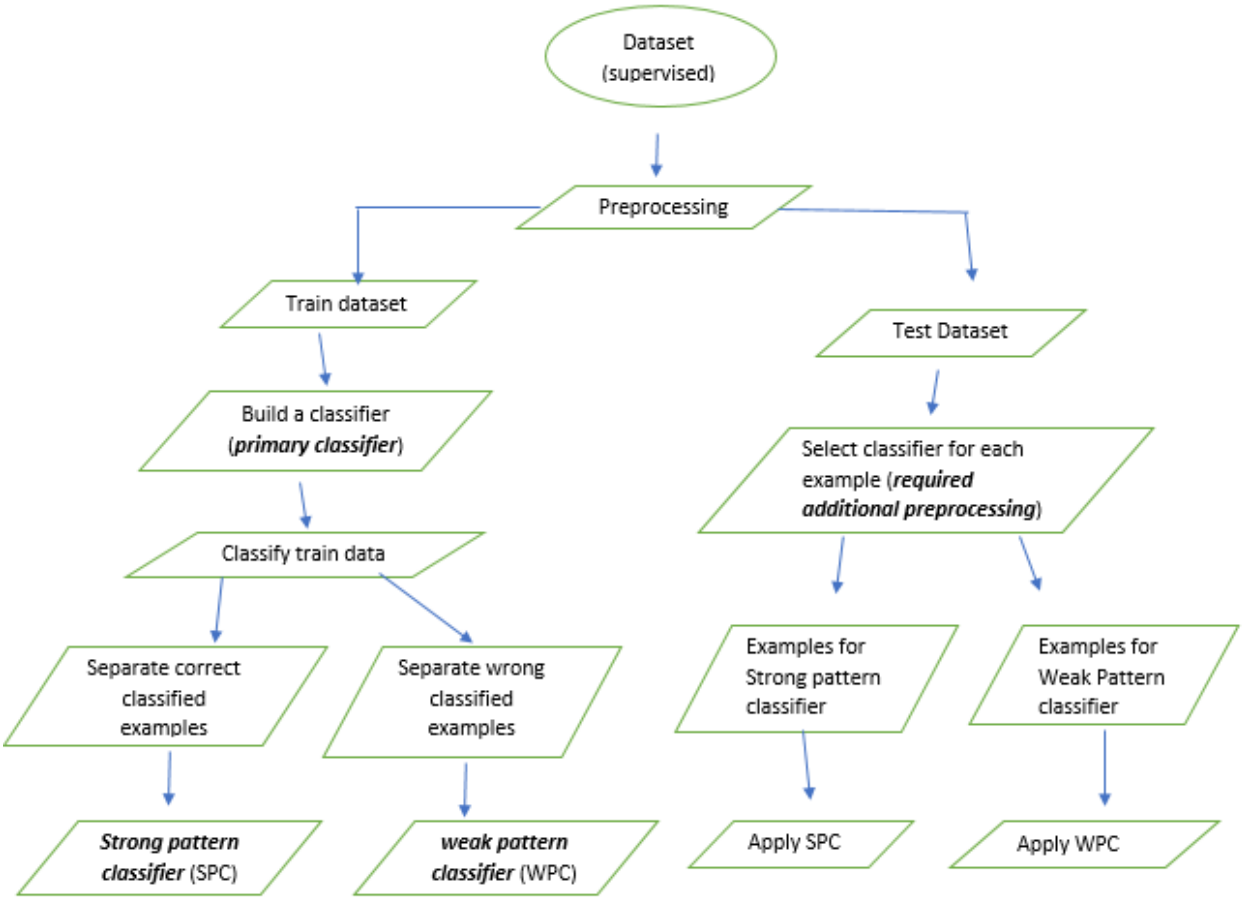
Using Boosting, we have got the best test accuracy (83.69%) and the ROC (88.89%) score from the decision tree, no better than the previous scores.

A different approach

So, we can conclude here that ***the obtained result is not good enough***. To find out the reason, I analyzed separately the correctly classified examples and the examples that are not. We discovered a pattern using the wrong classified examples that is ***very different*** from the pattern using correctly classified examples. So, the pattern from the wrongly classified examples is ***the weak pattern***, and the

pattern from the correctly classified examples is ***the strong pattern***. Now, if we can add one more ***preprocessing step*** to separate the weak pattern examples from the strong pattern examples and classify them using different classifier, then it is ***possible to achieve a very high accuracy*** close to perfect.

The ***flow chart*** for the weak pattern theory,



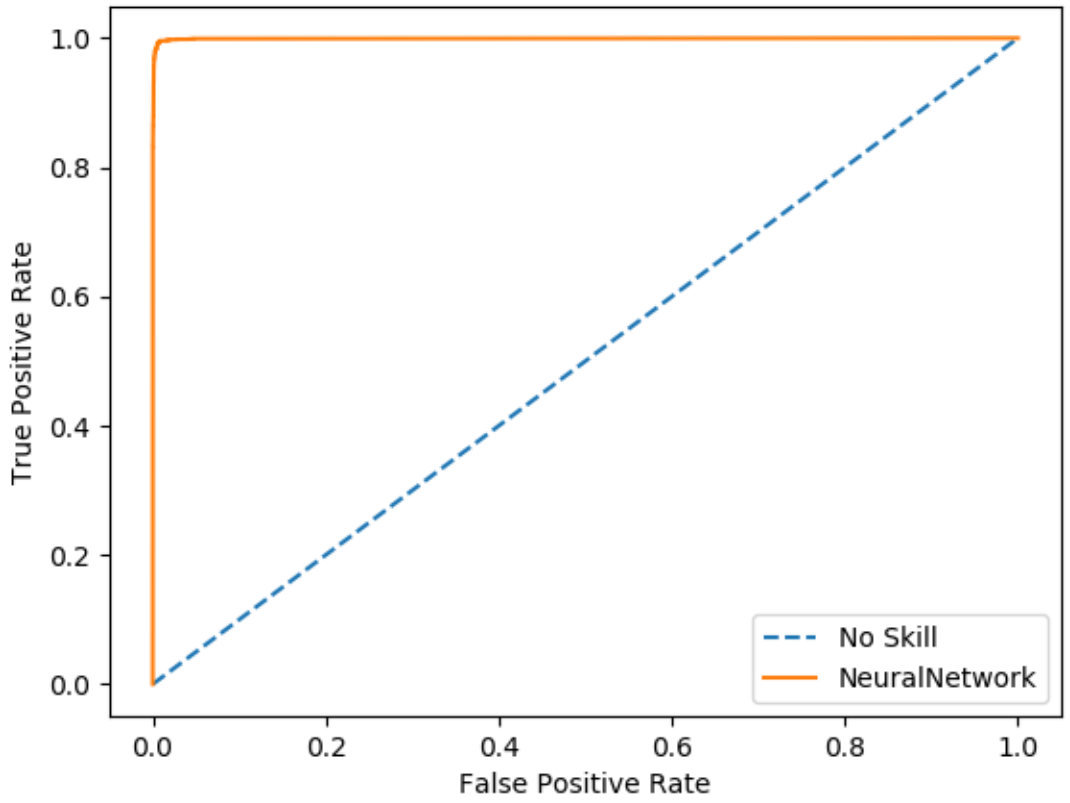
Let's assume, ***somehow***, we can separate the weak pattern examples from the strong pattern examples in the test dataset so that we can ***verify our weak pattern idea***,

Now, let's check the ***test result using weak pattern and strong pattern classifier***,

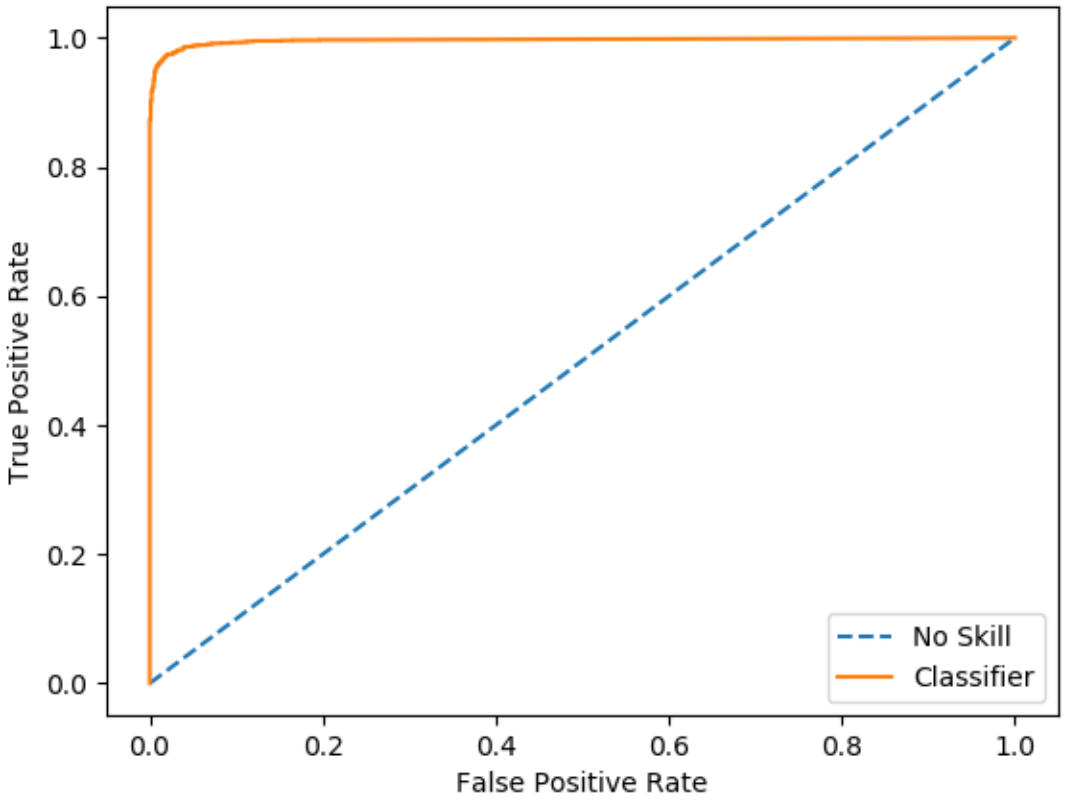
Performance comparison using Weak pattern and Strong pattern classifier:

Classifier	TrainAccuracy	TestAccuracy	TrueNegative(00) <=50K	FalsePositive(01)	FalseNegative(10)	TruePositive(11) >50K	ROC_Score
NeuralNetwork (Primary Classifier)	0.85086	0.85203	11686	749	1660	2186	0.90129
NeuralNetwork (Only strong pattern examples)	0.9961	0.99364	11141	49	36	2150	0.99915
NeuralNetwork (Only Weak pattern examples)	0.98255	0.97693	879	40	27	1959	0.99623

Corresponding ROC plot for ***strong pattern classifier***,



And the ROC plot for the ***weak pattern classifier***,



From the table, we see that we have **obtained a very high accuracy** on the test data for both strong pattern classifier (99.36%) and the weak pattern classifier (97.69%). Originally, there are in total 152 (85 using strong pattern classifier, and 67 using weak pattern classifier) misclassified examples. So, the ***original acquired accuracy*** is 99.07% which is pretty good compared to the previous results. The ROC score is also very high, for strong pattern classifier is 99.92%, and the weak pattern classifier is 99.62%.

Conclusion

We want to ensure every example the equal opportunity to ***show their own identity*** so that the dominant classifier (primary classifier) cannot misclassify any weak pattern examples. There may have several weak patterns, so we may need several weak pattern classifiers to classify them correctly. I assume it may require a huge research to select the right classifier, the additional preprocessing step, during the prediction of an example. Initially, I thought it can be possible to select the right classifier of an example based on the ***prediction probability*** of primary classifier, but it did not show the promising result. I have another thought using ***clustering algorithms*** to separate the weak pattern examples from

the strong pattern examples but could not verify yet. Finally, we can say that if we can overcome the additional preprocessing step then it is ***possible to obtain a very high accuracy close to perfect accuracy*** for any dataset.

References:

1. The Dataset (http://archive.ics.uci.edu/ml/datasets/Adult)
2. http://www.niculescu-mizil.org/papers/rocws04.crc.rev2.pdf
3. http://www.cs.cornell.edu/~alexn/papers/shotgun.icml04.revised.rev2.pdf
4. https://www.aaai.org/Papers/KDD/1996/KDD96-033.pdf
5. https://medium.com/alien-status/using-regression-models-to-predict-per-capita-and-median-household-income-in-nyc-91d5ca899509
6. http://cseweb.ucsd.edu/classes/sp15/cse190-c/reports/sp15/048.pdf