Review of Pacific Basin Financial Markets and Policies
Vol. 22, No. 3 (2019) 1950021 (27 pages)
© World Scientific Publishing Co.
and Center for Pacific Basin Business, Economics and Finance Research
DOI: 10.1142/S0219091519500218

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

Estimation Procedures of Using Five Alternative Machine Learning Methods for Predicting Credit Card Default

Huei-Wen Teng
Department of Information Management and Finance
National Chiao Tung University, Taiwan
venteng@gmail.com

Michael Lee
Georgia Institute of Technology, USA
mlee19@gatech.edu

Published 29 November 2019

Machine learning has successful applications in credit risk management, portfolio management, automatic trading, and fraud detection, to name a few, in the domain of finance technology. Reformulating and solving these topics adequately and accurately is problem specific and challenging along with the availability of complex and voluminous data. In credit risk management, one major problem is to predict the default of credit card holders using real dataset. We review five machine learning methods: the k-nearest neighbors decision trees, boosting, support vector machine, and neural networks, and apply them to the above problem. In addition, we give explicit Python scripts to conduct analysis using a dataset of 29,999 instances with 23 features collected from a major bank in Taiwan, downloadable in the UC Irvine Machine Learning Repository. We show that the decision tree performs best among others in terms of validation curves.

Keywords: Artificial intelligence; machine learning; supervised learning; k-nearest neighbors, decision tree; booting; support vector machine; neural network; Python; delinquency; default; credit card; credit risk.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

1. Introduction

Machine learning is a subset of artificial intelligence that often uses general and intuitive methodology to give computers (machines) the ability to learn with data so that the performance on a specific task is improved, without explicitly programmed (Samuel, 1959). Because of its flexibility and generality, machine learning has been successfully applied in the fields, including email filtering, detection of network intruders or malicious intruders working towards a data breach, optical character recognition, learning to rank, informatics, and computer vision (Mitchell, 1997; Mohri *et al.*, 2012). In recent years, machine learning has fruitful applications in finance technology, such as fraud prevention, risk management, portfolio management, investment predictions, customer service, digital assistants, marketing, sentiment analysis, and network security.

Machine learning is closely related to statistics (Bzdok *et al.*, 2018). Statistics is a sub-field of mathematics, whereas machine learning is a sub-field of computer science. To explore the data, statistics starts with a probability model, fits the model to the data, and verifies if this model is adequate using residual analysis. If the model is not adequate, residual analysis can be used to refine the model. Once the model is shown to be adequate, statistical inference about the parameters in the model can be furthermore used to determine if a factor of interests is significant. The ability to explain if a factor really matters makes statistics widely used in almost all disciplines.

Machine learning focuses more on prediction accuracy but not model interpretability. In fact, machine learning uses general purposes algorithms and aims at finding patterns with minimal assumption about the data-generating system. Classic statistical methods combined with machine learning techniques lead to a combined field, called statistical learning (James *et al.*, 2013).

Application domain of machine learning can be roughly divided into unsupervised learning and supervised learning (Hastie *et al.*, 2008). Unsupervised learning refers to the situations one has just predictors (also known as input, explanatory, or independent variables), and attempts to extract features that represent the most distinct and striking features in the data. Supervised learning refers to the situations that one has predictors and responses (also known as output, or dependent variables), and attempts to extract important features in the predictors that best predict responses. Using input–output pairs, supervised learning learns a function from

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

the dataset to map an input to an output using a sample (Russell and Norvig, 2010).

In financial technology (FinTech), machine learning has received extensive attention in recent years. For example, Heaton *et al.* (2017) applied deep learning for portfolio optimization. With the rapid development of high-frequency trading, intra-day algorithmic trading becomes a popular trading device and machine learning is a fundamental paralytics for predicting returns of underlying asset: Putra and Kosala (2011) used neural network and validated the validity of the associated trading strategies in Indonesia stock market; Borovykh *et al.* (2018) proposed a convolutional neural network to predict time series of the S&P 500 index.

In addition to the above applications, machine learning is also applied to other canonical problem in finance. For example, Solea *et al.* (2018) identified the next emerging countries using statistical learning techniques, and Gu *et al.* (2018) performed a comparative analysis of methods using machine learning to measure asset risk premia in empirical asset pricing.

To predict the delinquency of a credit card holder, a credit scoring model provides a model-based estimate of the default probability of a credit card customer. The predictive models for the default probability have been developed using machine learning classification algorithms for binary outcomes (Hand and Henley, 1997). There have been extensive studies examining the accuracy of alternative machine learning algorithms or classifiers. A survey can be found in Zopounidis *et al.* (1997), Thomas (2000), Baesens *et al.* (2003), Kumar and Ravi (2007), Crook *et al.* (2007), and Demyanyk and Hasan (2010). Recently, Lessmann *et al.* (2015) provided comprehensive classifier comparisons to date and divide machine learning algorithms into three divisions: individual classifiers, homogeneous ensembles, and heterogeneous ensembles.

Individual classifiers are those using a single machine learning algorithm, for example, the K-nearest neighbours, decision trees, support vector machine, and neural network. For example, Keerthi and Lin (2003) studied asymptotic behaviors of support vector machines for Gaussian kernels. Butaru *et al.* (2016) tested decision tree, regularized logistic regression, and random forest models with a unique large dataset from six larger banks. It is found that no single models applies to all banks, and suggests the need for a more customized approach to the supervision and regulation of financial institutions, in which parameters such as capital ratios and loss reserves should be specified to each bank according to its credit risk model exposures and forecasts.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

Neural networks have been used for credit scoring models as in Atiya (2001) and Bahrammirzaee (2010). Sun and Vasarhelyi (2018) demonstrated the effectiveness of a deep neural network based on clients' personal characteristics and spending behaviors over logistic regression, naïve Bayes, traditional neural networks, and decision trees in terms of better prediction performance with a dataset of size 711,397 collected in Brazil.

Novel machine learning method to incorporate complex features of the data is proposed as well. For example, Fernandes and Artes (2016) incorporated spatial dependence as inputs into the logistic regression, and Maldonado *et al.* (2017) proposed support vector machines for simultaneous classification and feature selection that explicitly incorporate attribute acquisition costs. Addo *et al.* (2018) provided a binary classifiers based on machine and deep learning models on real data in predicting loan default probability. It is observed that tree-based models are more stable than neural network-based methods.

On the other hand, the ensemble method contains two steps: model development and forecast combinations. It can be divided into homogeneous ensemble classifiers and heterogeneous ensemble classifiers. The former uses the same classification algorithm, whereas the latter uses different classification algorithms. Finlay (2011) and Paleologo *et al.* (2010) have shown that homogeneous ensemble classifiers increase predictive accuracy. Two types of homogeneous ensemble classifiers are bagging and boosting. Bagging derives independent base models from bootstrap samples of the original data (Breiman, 1996), and boosting iteratively adds base models to avoid the errors of current ensembles (Freund and Schapire, 1996).

Heterogeneous ensemble methods create these models using different classification algorithms, which have different views on the same data and may complement each other. In addition to base model developments and forecast combinations, heterogeneous ensembles need a third step to search the space of available base models. Static approaches search the base model once, and dynamic approaches repeat the selection step for every case (Ko *et al.*, 2008; Woloszynski and Kurzynski, 2011). For static approaches, the direct method maximizes predictive accuracy (Caruana *et al.*, 2006) and the indirect method optimizes the diversity among base models (Partalas *et al.*, 2010).

The rest of this paper is organized as follows. Section 2 introduces the credit card dataset. Section 3 reviews five supervised learning methods. Section 4 gives the study plan to find the optimal parameters and compares

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

the learning curves among five methods. Section 5 concludes. Python scripts are given in the appendix.

2. Description of the Data

We apply the machine learning techniques in the default of credit card clients' dataset. There are 29,999 instances in the credit card dataset. The default of credit' card clients' dataset can be found in the UC Irvine Machine Learning Repository at `http://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients` and was initially analyzed in Yeh and Lien (2009). This dataset is the payment data of credit card holders in October, 2005 from a major cash and credit card issuer in Taiwan. This dataset contains 23 different attributes to determine whether or not a person would default on their next credit card payment. It contains amount of the given credit, gender, education, marital status, age, and history of past payments, including how long it took someone to pay the bill, the amount of the bill, and how much they actually paid for the previous 6 months.

The response variable is

- Y: Default payment next month (1 = default; 0 = not default).

We use the following 23 variables as explanatory variables:

- X_1: Amount of the given credit (NT dollar)
- X_2: Gender (1 = male, 2 = female)
- X_3: Education (1 = graduate school; 2 = university; 3 = high school; 4 = others)
- X_4: Marital status (1 = married; 2 = single; 3 = others)
- X_5: Age (year).
- $X_6 - X_{11}$: History of past monthly payment traced back from September 2005, to April 2005 (-1 = pay duly; 1 = payment delay for one month; 2 = payment delay for two months; ..., 8 = payment delay for eight months; 9 = payment delay for nine months and above)
- $X_{12} - X_{17}$: Amount of past monthly bill statement (NT dollar) traced back from September 2005 to April, 2005.
- $X_{18} - X_{23}$: Amount of past payment (NT dollar) traced back from September 2005 to April, 2005.

This dataset is interesting because it contains two "sorts" of attributes. The first sort is about categorical attributes like education, marital status, and age. These attributes have a very small range of possible values, and if there

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

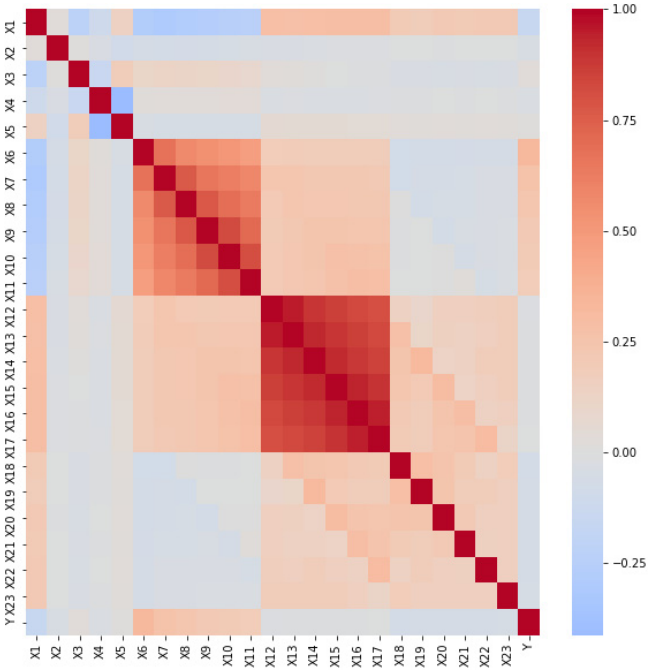

Fig. 1. The heatmap of correlations between the response variable and all predictors in the credit card dataset.

was a high correlation between these categorical attributes then the classification algorithms would be able to easily identify them and produce high accuracies. The second sort of attribute is the past payment information. These attributes are just integers without clear differentiation of categories and have much larger possible ranges of how much money was paid. Especially if there was not strong correlation between education, marital status, age, etc. and defaulting on payments, it could be more difficult to algorithmically predict the outcome from past payment details except for the extremes where someone never pays their bills or always pays their bills.

Figure 1 plots the heatmap to show pairwise correlations between attributes. It is shown that most correlations are about zeros, but high correlations exist in features of past monthly payments $(X_6, \ldots, X_{11})$ and past monthly bill statements $(X_{12}, \ldots, X_{17})$.

3. Machine Learning

Let $X = (X_1, \ldots, X_p)'$ denote the p-dimensional input vector, and let $Y = (Y_1, \ldots, Y_d)'$ denote the d-dimensional output vector. In its simplest form,

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

a learning machine is an input–output mapping, $Y = F(X)$. In statistics, $F(\cdot)$ is usually a simple function, such as a linear or polynomial function. In contrast, the form of the $F(\cdot)$ in machine learning could be more complicated.

In the following, we introduce five machine learning methods: K-nearest neighbors, decision tree, boosting, support vector machine, and neural network, with illustrative examples.

3.1. *K-Nearest neighbors*

The K-Nearest Neighbors method (KNN) is intuitive and easy to implement. First, a distance metric (such as the Euclidean distance) needs to be chosen to identify the KNNs for a sample of unknown category. Second, a weighting scheme (uniform weight or distance weight) to summarize the score of each category needs to be decided. The uniform weighting scheme gives equal weight for all neighbors regardless of its distance to the sample of unknown category, whereas the distance weighting scheme weights distant neighbors less. Third, the score for each category is summed over these k-nearest neighbors. Finally, the predicted category of this sample is the category yielding the highest score.

An example is illustrated in Fig. 2. Suppose there are two classes (category A and category B) for the output and two features (x_1 and x_2). A sample of unknown category is plotted as a solid circle. KNN predicts the category of this sample as follows. To start, we choose Euclidean distance and uniform distance weight. If $K = 3$, in the three nearest neighbors to the unknown sample, there are one sample of category A and two samples of category B. Because there are more samples of category B, KNN predicts the unknown sample to be of category B. If $K = 6$, in the six nearest neighbors to the sample of unknown category, there are four samples of class A and two samples of class B. Because class A occurs more frequently than class B, KNN predicts the sample to be of category A.

In addition to the distance metric and weighting scheme, the number of neighbors K is needed to be decided. Indeed, the performance of the KNN is highly sensitive to the size of K. There is no strict rule in selecting K. In practice, the selection of K can be done by observing the predicted accuracies for various K and select the one that reaches the highest testing and cross-validation scores. Detailed descriptions about how to calculate these scores and find the optimal parameters are given in Sec. 4.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

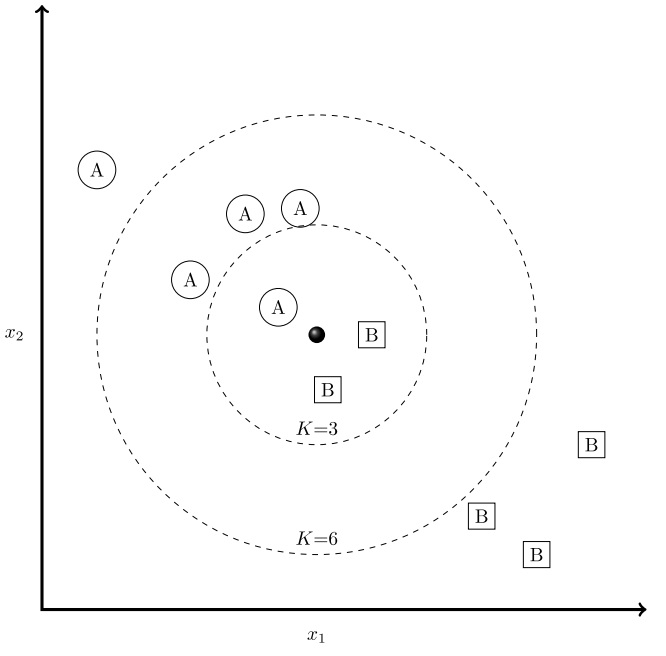

Fig. 2. Illustration of the KNNs.

3.2. *Decision trees*

A decision tree is also called a classification tree when the target output variable is categorical. For a decision tree, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision tree is usually constructed top-down, by choosing a mapping of feature variables at each step that best splits the set of items. Different algorithms choose different metrics for measuring the homogeneity of the target variables within the subsets. These metrics are applied to each candidate subset and the resulting values are combined to provide a quality of the split. Common metrics include the Information Gain and Gini Index based on the concept of entropy.

Figure 3 depicts the structure of a decision tree: the decision tree starts with a root node and consist of internal decision nodes and leaf nodes. The decision nodes and leaf nodes are stemmed from the root node and are connected by branches. Each decision node represents a test function with discrete outcomes labeling the branches. The decision tree grows along with these branches into different depths of internal decision nodes. At each step, the data is classified by a different test function of attributes leading the data

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

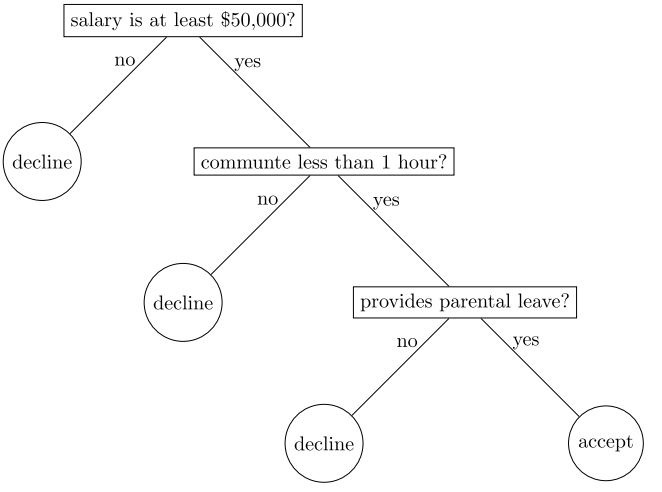

Fig. 3. Illustration of the decision tree.

either to a deeper depth of internal decision node or it finally ends up at a leaf node.

Figure 3 illustrates a simple example. Suppose an interviewer is classified as "decline offer" or "accept offer". The tree starts with a root node. The root node is a test function to check if the salary is at least $50,000. If data with answer "no" declines the offer, the branch ends up with a leaf node indicating "decline". If data with answer "yes" contains declining and accepting offer, this branch results in a second decision node to check if the interviewer needs commuting time more than 1 h and the output could be "yes" and "no". If data with answer "yes" declines the offer, this branch ends up with a leaf node indicating "decline". If data with answer "no" contains both declining and accepting the offer, the branch ends up with another decision node to check if parental leave is provided. Again, data with answer "no" declines the offer, so this branch ends up with a leaf node indicating "decline". Data with answer "yes" accepts the offer, so this branch ends up with a leaf note indicating "accept".

To apply the decision tree algorithm, we use the training dataset to build a decision tree. For a sample with unknown category, we simply employ the decision tree to figure out which leaf node the sample of unknown category will end up.

Different algorithms choose different metrics for measuring the homogeneity of the target variables within the subsets. These metrics are applied to each candidate subset and the resulting values are combined to provide a

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

quality of the split. Common metrics include the Gini Index and Information Gain. The major difference between the Information Gain and the Gini Index is that the former produces multiple nodes, whereas the latter only produces two nodes (TRUE and FALSE, or binary classification).

The representative decision tree using Gini Index (also known as the Gini Split and Gini Impurity) to generate the next lower node is the classification and regression tree (CART), which indeed allows both classification and regression. Because CART is not limited to the types of response and independent variables, it is of wide popularity.

Suppose we would like to build up a next lower node, and the possible classification label is i, for $i = 1, \ldots, c$. Let p_i represent the proportion of the number of samples in the lower node classified as i. The Gini Index is defined as

$$\text{Gini Index} = 1 - \sum_{i=1}^{c} p_i^2. \tag{1}$$

The attribute used to build the next node is the one that maximizes the Gini Index.

The Information Gain is precisely the measure used by the decision tree ID3 and C4.5 to select the best attribute or feature when building the next lower node (Mitchell, 1997). Let f denote a candidate feature, and D denote the data at current node and D_i denote the data classified as label i at the lower node, for $i = 1, \ldots, c$. $N = |D|$ is the number of the sample at current node, and $N_i = |D_i|$ is the number of sample classified as label i at the lower node. Then, the Information Gain is defined as

$$\text{IG}(D, f) = I(D) - \sum_{i=1}^{c} \frac{N_i}{N} I(D_i), \tag{2}$$

where I is an impurity measure, either the Gini Index is defined in Eq. (1) or the entropy. The entropy is defined as

$$I_e = -\sum_{i=1}^{c} p_i \log_2 p_i. \tag{3}$$

Equation (2) can be regarded as the original information at current node minus the expected value of the impurity after the data D is partitioned using attribute f. Therefore, f is selected to maximize the IG. Entropy and Gini Impurity perform similarly in general, so we can focus on the adjustment of other parameters.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

3.3. *Boosting*

In the field of computer science, weak learner is a classification rule of lower accuracy, whereas strong learner is that of higher accuracy. The term "boosting" refers to a family of algorithms which convert weak learners to strong learners. There are many boosting algorithms, such as AdaBoost (Adpative Boosting), Gradient Tree Boosting, and XGBoost. Here, we focus on AdaBoost.

In an iterative process, boosting yields a sequence of weak learners which are generated by assuming different distributions for the sample. To choose the distribution, boosting proceeds as follows:

- Step 1: The base learner (or the first learning algorithm) assigns equal weight to each observation.
- Step 2: The weights of observations which are incorrectly predicted are increased to modify the distribution of the observation, so that a second learner is obtained.
- Step 3: Iterate Step 2 until the limit of base learning algorithm is reached or higher accuracy is reached.

With the above procedures, a sequence of weak learner is obtained. The prediction of a new sample is based on the average (or weighted average) of each weak learners or that having the higher vote from all these weak learners.

3.4. *Support vector machines*

A support vector machine (SVM) is a recently developed technique originally used for pattern classification. The idea of SVM is to find a maximal margin hyperplane to separate data points of different categories. Figure 4 shows how to the support vector machine to separate the data into two categories with hyperplanes.

If the classification problem cannot be separated by a linear hyperplane, the input features have to be mapped into a higher-dimensional feature space by a mapping function, which is calculated through a prior chosen kernel function. Kernel functions include linear, polynomial, sigmoid, and the radial basis (RBF) functions. Yang (2007) and Kim and Sohn (2010) applied SVM in credit scoring problem and showed that SVM outperforms other techniques in terms of higher accuracy.

3.5. *Neural networks*

A neural network, or an artificial neural network, has the advantage of strong learning ability without any assumptions about the relationships

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

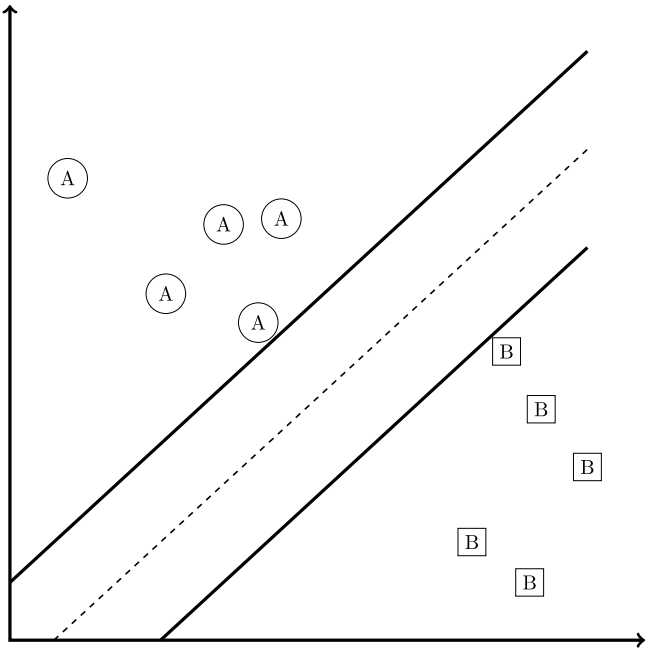

Fig. 4. Illustration of the SVM.

between input and output variables. Recent studies using a neural network or its variants in credit risk analysis can be found Desai *et al.* (1996), Malhotra and Malhotra (2002), and Abdou *et al.* (2008).

Neural network links the input–output paired variables with simple functions called activation functions. A simple standard structure for a neural network include an input layer, a hidden layer, and an output layer. If a neural network contains more than one hidden layer, it is also called as deep neural network (or deep learning neural network).

Suppose that there are unknown L layers in a neural network. The original input layer and the output layer are also called the 0th layer and $(L+1)$th layer, respectively. The name of hidden layers implies that they are originally invisible in the data and are built artificially. The number of layers L is called the *depth* of the architecture. See Fig. 5 for an illustration of a structure of a neural network.

Each layer is composed of nodes (also called neurons) representing a nonlinear transformation of information from the previous layer. The nodes in the input layer receive input features $X = (X_1, \ldots, X_p)$ of each training sample and transmit the weighted outputs to the hidden layer. The d nodes in the output layer represent the output features $Y = (Y_1, \ldots, Y_d)$.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

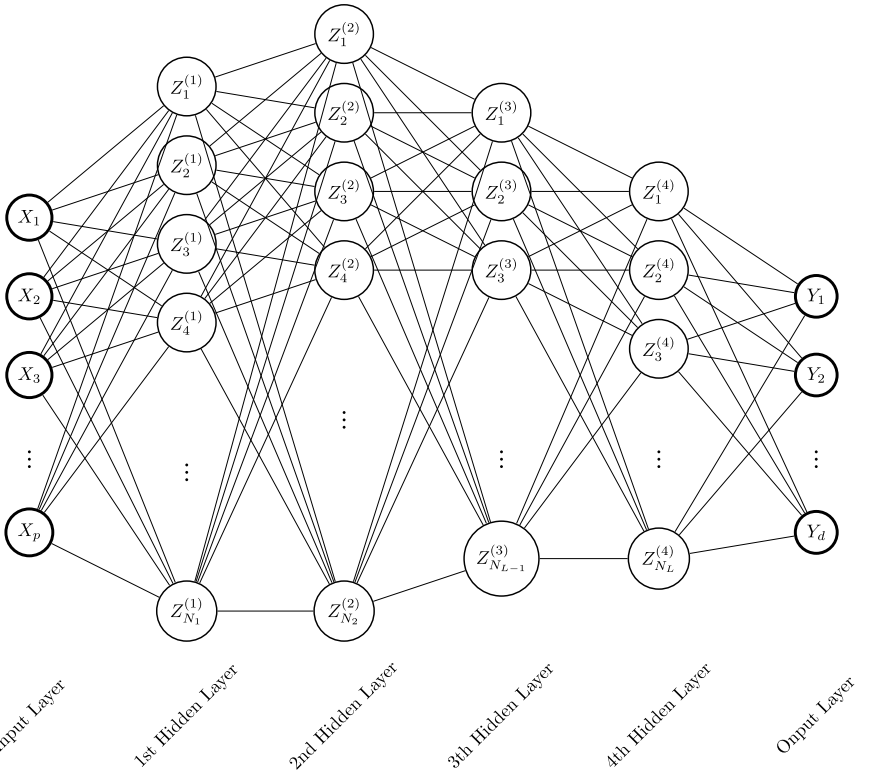

Fig. 5. Illustration of a neural network with four layers.

Let $l \in \{1, 2, \ldots, L\}$ index the layers from 1 to L. A neural network trains a model on data to make predictions by passing learned features of data through different layers via L nonlinear transformation applied to input features. We explicitly describe a deep learning architecture as follows. For a hidden layer, various activation functions, such as logistic, sigmoid, and radial basis function (RBF) functions, can be applied. We summarize some activation functions and their definitions in Table 1.

Let $f^{(0)}, f^{(1)}, \ldots, f^{(L)}$ be given univariate activation functions for these layers. For notational simplicity, if f is a given activation. Suppose

Table 1. List of activation functions.

Activation Function	Definition
The identity function	$f(x) = x$
The logistic function	$f(x) = 1/(1 + \exp(-x))$
The hyperbolic tan function	$f(x) = \tanh(x)$
The rectified linear units (ReLU) function	$f(x) = \max\{x, 0\}$

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

$U = (U_1, \ldots, U_k)'$ is a k-dimensional input. We abbreviate $f(U)$ by

$$f(U) = (f(U_1), \ldots, f(U_k))'.$$

Let N_l denote the number of nodes at the l-th layers, for $l = 1, \ldots, L$. For notational consistency, let $N_0 = p$, and $N_{(L+1)} = d$. To build the lth layer, let $W^{(l-1)} \in \Re^{N_l \times N_{l-1}}$ be the weight matrix, and $b^{(l-1)} \in \Re^{N_l}$ be the thresholds or activation levels, for $l = 1, \ldots, L+1$. Then, these N_l nodes at the lth layers $Z^{(l)} \in \Re^{N_l}$ are formed by

$$Z^{(l)} = f^{(l-1)}(W^{(l-1)} Z^{(l-1)} + b^{(l-1)}),$$

for $l = 1, \ldots, L+1$. Specifically, the deep learning neural network is constructed by the following iterations:

$$\begin{aligned}
Z^{(1)} &= f^{(0)}(W^{(0)} X + b^{(0)}),\\
Z^{(2)} &= f^{(1)}(W^{(1)} Z^{(1)} + b^{(1)}),\\
Z^{(3)} &= f^{(2)}(W^{(2)} Z^{(2)} + b^{(2)}),\\
&\vdots\\
Z^{(l)} &= f^{(l-1)}(W^{(l-1)} Z^{(l-1)} + b^{(l-1)}),\\
&\vdots\\
Z^{(L)} &= f^{(L-1)}(W^{(L-1)} Z^{(L-1)} + b^{(L-1)}),\\
\hat{Y} &= f^{(L)}(W^{(L)} Z^{(L)} + b^{(L)}).
\end{aligned}$$

Finally, the deep learning neural network predicts Y by $\hat{Y}$ using the input W and the learning parameters $W = \{W^{(0)}, W^{(1)}, \ldots, W^{(L)}\}$ and $b = \{b^{(0)}, b^{(1)}, \ldots, b^{(L)}\}$. As a result, a deep learning neural network predicts Y by

$$F^{W,b}(X) := f^{(L)}(W^{(L)} Z^{(L)} + b^{(L)}).$$

Once the architecture of the deep neural network (L and N_l for $i = 1, \ldots, L$) and activation functions $f^{(l)}$ for $l = 1, \ldots, L$ are decided, the solutions $\hat{W}$ and $\hat{b}$ for the learning parameters $W = \{W^{(0)}, W^{(1)}, \ldots, W^{(L)}\}$ and $b = \{b^{(0)}, \ldots, b^{(L)}\}$ satisfy

$$\hat{W}, \hat{b} = \arg\min_{W,b} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(Y^{(i)}, F^{W,b}(X^{(i)})).$$

Here, $\mathcal{L}$ is the loss function.

Some drawbacks of building a neural network are summarized in the following. First, the relationship between the input and output variables is mysterious because the structure of an NN could be very complicated.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

Second, how to design and optimize the NN structure is determined via a complicated experiment process. For instance, different combinations of number of hidden layers, number of nodes in each hidden layers, and activation functions in each layer, yield different classification accuracies. As a consequence, learning an NN is usually time consuming.

4. Study Plan

In Sec. 4.1, we describe how to preprocess the data and describe the python programming. We defer python scripts in the appendix. Section 4.2 provides detailed descriptions on the tuning process to decide the optimal tuning parameter, because there is no quick access in selecting the optimal tuning parameters in each method. Section 4.3 compares the performance of these five machine learning methods with the learning curves.

4.1. *Data preprocessing and Python programming*

To start with, we preprocess the data as follows. Because the dataset is quite complete, there is no missing data issue. We take log-transformation for continuous variables, such as X_{12} to X_{17} and X_{18} to X_{23}, because they are highly skewed.

Python is created by Guido van Rossum, first released in 1991, and is a high-level programming language for general purpose programming. Python has been successfully applied to machine learning techniques with a wide range of applications. See Raschka (2015) for using Python for machine learning. For simplicity, we provide Python codes in the appendix to preprocess the data and apply machine learning methods to the dataset.

4.2. *Tuning optimal parameters*

The optimal combination of parameters is decided based on criteria such as testing scores and cross-validation scores. To calculate the testing score, we split the dataset randomly into 70% training set and 30% testing set. When fitting the algorithm, we only use the training set. Then, we use the remain 30% testing set to calculate the percentage of correct classification of the method. This is the testing score, which represents the prediction accuracy.

Furthermore, to investigate if the algorithm is stable and if the overfitting problem exists, we calculate the cross-validation score. We further split the 70% training set into ten subsets, and fit the algorithm using nine of these subsets being the training data, and one set being the testing data.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

Rotating which set is the testing set, the average of these ten prediction accuracies is the cross-validation score.

Our selection rule for optimal tuning parameters goes as follows. We first plot the testing and cross-validation scores for various combinations of tuning parameters. The optimal tuning parameters are the simplest to achieve the highest testing scores, whereas the cross-validation scores are later used to check if the over-fitting problem exists.

The above procedures give a simple rule to select the optimal tuning parameters. We remark that there are other alternatives to select the optimal tuning parameters. For instance, the optimal combination of tuning parameters is selected to maximize the performance measure (such as the F1-score or AUC).

4.2.1. *K-Nearest neighbors*

Figure 6 compares testing and cross-validation scores against various combinations of tuning parameters: k ranging from $1, 21, 41, \ldots, 81$, and two weighting schemes (uniform weight and distance weight). Testing scores with uniform and distance weighting are about the same, which are also close to the two cross-validation scores. Therefore, we choose uniform weighting because it is simpler, and choose k to be 50 because all four scores appear to be stable for k larger than 50.

4.2.2. *Decision trees*

Figure 7 compares the testing and cross-validation scores for decision trees. We test both the Gini Index and entropy for the Information Gain splitting

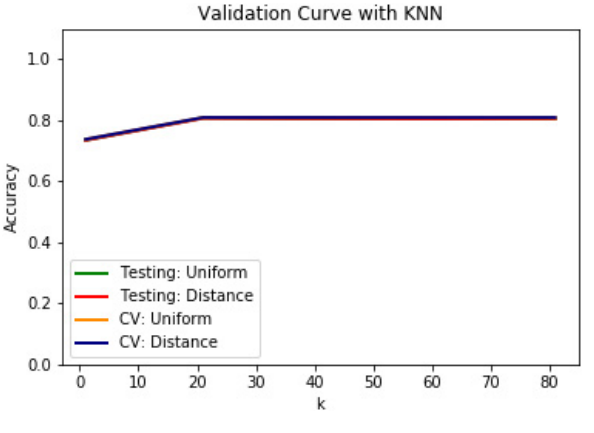

Fig. 6. Testing and cross-validation scores of the k-nearest neighbors against k with uniform weight and distance weight using training data and cross-validation of the credit card dataset.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

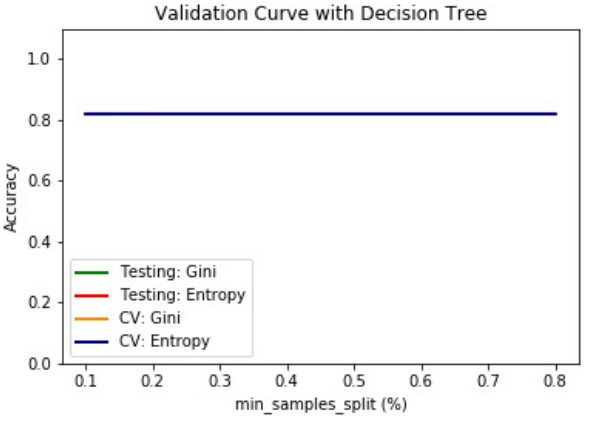

Fig. 7. Testing and cross-validation scores of decision tree against minimum samples splits with Gini Index and Information Gain using training data and cross-validation of the credit card dataset.

criteria. We vary the number of samples in a node required to split it because this effectively varies the amount of pruning done to the decision tree. A low requirement lets the decision tree split the data into small groups, increases the complexity of the tree, and corresponds to low pruning. A high requirement prevents as many nodes being created, decreases the complexity of the tree, and corresponds to higher pruning.

Because testing scores of using Gini Index and entropy are close, we choose Gini Index because it is the default criteria for splitting. On the other hand, both training and cross-validation scores are not affected by the amount of pruning. Hence, we choose 80% of samples for minimum split requirement for a decision tree with a smaller maximum depth.

4.2.3. *Boosting*

Figure 8 plots the training and cross-validation curves against the maximum depth of the decision tree (1 and 2) and the number of estimators voting on the correct hypothesis. We decide to use a maximum tree depth of one since it is more general and does not perform worse than a maximum depth of 2, and 10 estimators because it gives better performance and this data set does not benefit from having more estimators.

4.2.4. *Support vector machines*

Figure 9 compares the testing and cross-validation scores with the SVM using both the polynomial and RBF kernels with maximum iterations ranging from 1000, 1500, 2000, and 2500. Our experiments suggest to use the

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

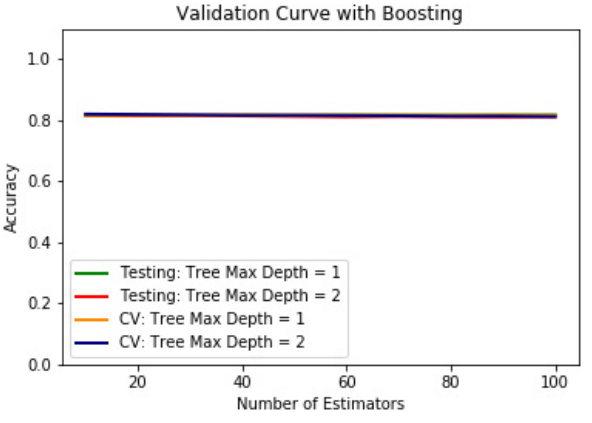

Fig. 8. Testing and cross-validation scores of boosting against number of estimators with tree maximum depth to be one and two using training data and cross-validation of the credit card dataset.

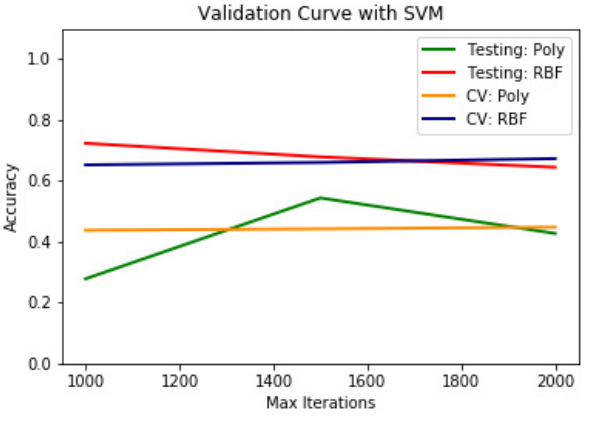

Fig. 9. Testing and cross-validation scores of the SVM against maximum iterations with polynomial and RBF kernel functions.

RBF kernel because it performs much better than the polynomial kernel, and it also runs faster than the polynomial kernel. In addition, we use a maximum iterations value of 2100, as no more improvements on the testing scores can be found with larger maximum iterations value.

4.2.5. *Neural networks*

We use the ReLU function as the activation function. For neural network, we decide to test the number of hidden layers and number of neurons in each hidden layer. Figure 10 compares the testing and cross-validation scores of

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

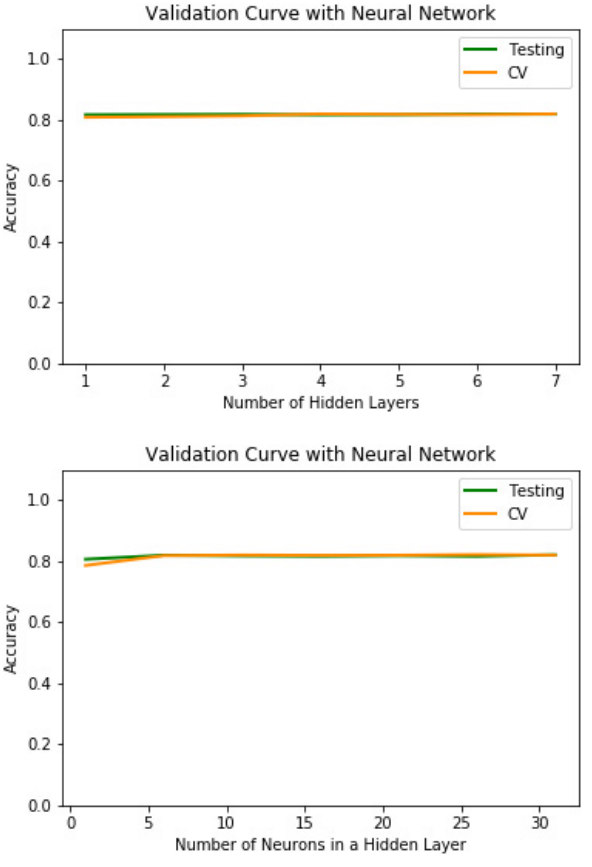

Fig. 10. Testing and cross-validation scores of neural network against number of hidden layers and number of of neurons in each hidden layer, in the upper and lower panels, respectively.

neural network. The upper panel varies the number of hidden layers, and it is suggested that we select the number of hidden layers to be three. With three hidden layers, the lower panel varies the number of hidden neurons in each layer, which suggests us to have 15 neurons as a suitable size in each layer.

4.3. *Learning curves*

Figure 11 compares the accuracy with these five machine learning methods against the number of examples (the size of training set) with the optimal tuning parameters obtained in Sec. 4.2 to see if the accuracy appears to be stable as the number if examples increases. It is shown that KNN, decision tree, and boosting, perform consistently as the number of examples

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

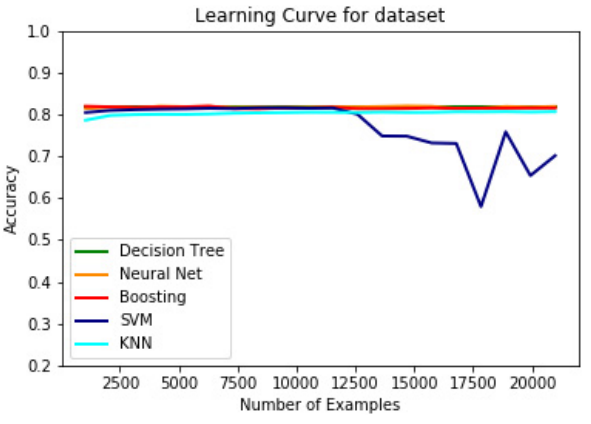

Fig. 11. Learning curves against number of examples with decision tree, neural net, boosting, SVM, and k-nearest neighbors, for the credit card dataset.

increases. But, SVMs perform worse as the number of examples increases. As a conclusion, for the credit card dataset, the decision tree algorithm performs the best. Not only does it yields highest accuracy, but it runs the quickest.

5. Conclusion

5.1. *Summary*

In this paper, we introduce five machine learning methods: KNNs, decision tree, boosting, support vector machine, and neural network, to predict the default of credit card holders. For illustration, we conduct data analysis using a dataset of 29,999 instances with 23 features and provide Python scripts for implementation. It is shown in our study that the decision tree performs best in predicting the default of credit card holders in terms of learning curves.

5.2. *Limitations and future work*

As the risk management for personal debt is of considerable importance, it is worthy of studying the following directions for future research. One limitation in this paper is that we only use one dataset. According to Butaru *et al.* (2016), multiple datasets should be used to illustrate the robustness of a machine learning algorithm, and pairwise-comparisons should be conducted to verify which machine learning algorithm outperforms the others (Demšar, 2006; García and Herrera, 2008).

This paper only uses accuracy as a measure to compare different machine learning methods. Indeed, in addition to the standard measures, such as precision, recall, F1-score, and AUC, it is interesting to consider cost-sensitive framework or profit measures to compare different machine learning algorithms as in Verbraken *et al.* (2014), Bahnsen *et al.* (2015), and Garrido *et al.* (2018).

Along with the availability of voluminous data in recent days, Moeyersoms and Martens (2015) solve high-cardinality attributes in churn prediction in the energy sector. In addition, it is also interesting to predict for longer-horizon or the default time (using survival analysis). Last but not the least, it is of considerable importance to develop a method for an extremely rare event. All the above-mentioned issues are worthy of future studies and will be investigated in separate papers.

Acknowledgments

The first author gratefully acknowledges the support by the Ministry of Education of Taiwan, R.O.C., under the Higher Education Sprout Project, and the Ministry of Science and Technology under grants MOST 108-2118-M-009-001-MY2 and MOST 108-2634-F-163-001 through Pervasive Artificial Intelligence Research Labs.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

Appendix: Python codes

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

dataset = pd.read_csv('C:\\default␣of␣credit␣card␣clients.csv')
dataset.shape
dataset.head()
dataset = dataset.iloc[0:29999,:]
colForLog = ["X1",
             "X12","X13","X14","X15","X16","X17",
             "X18", "X19", "X20", "X21", "X22", "X23"]
for col in colForLog:
    dataset.loc[:, col] = ( dataset.loc[:, col] - dataset.loc[:,col].min()
    + dataset.loc[:,col].median()).apply(np.log)

dataset = dataset.drop(['Unnamed:␣0'], axis = 1)

X = dataset.drop(['Y'],axis=1)
y = dataset['Y']

# Get 70-30 split to get the training-testing data
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.30, random_state = 42)


from sklearn import metrics # to get the accuracy
def calc_testing_score(X_train, y_train, X_test, y_test, clf):
    clf = clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    accuracy = metrics.accuracy_score(y_test, y_pred)
    return accuracy

from sklearn.neighbors import KNeighborsClassifier
```

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

```
param_range = range(1, 100, 20)
# (a) get testing score for distance = uniform
for i in range(1, 100, 20):
    clf = KNeighborsClassifier(n_neighbors = i)
    clf2 = KNeighborsClassifier(n_neighbors = i, weights = "distance")
    accuracyTest = calc_testing_score(X_train, y_train, X_test, y_test, clf)
    accuracyTest2 = calc_testing_score(X_train, y_train, X_test, y_test,
    clf2)

'''
II. Decision Tree 2/5
'''
from sklearn.tree import DecisionTreeClassifier
param_range = np.arange(.1, .9, .1)
for i in np.arange(.1, .9, .1):
    clf = DecisionTreeClassifier(min_samples_split = i)
    clf2 = DecisionTreeClassifier(criterion = "entropy",
    min_samples_split = i)
    accuracyTest = calc_testing_score(X_train, y_train, X_test, y_test, clf)
    accuracyTest2 = calc_testing_score(X_train, y_train, X_test, y_test,
    clf2)

'''
III. Boosting 3/5
'''
from sklearn.ensemble import AdaBoostClassifier
param_range = range(10, 109, 10)
for i in range(10, 109, 10):
   clf = AdaBoostClassifier(
   base_estimator = DecisionTreeClassifier(max_depth = 1),
    n_estimators = i)
   clf2 = AdaBoostClassifier(
   base_estimator = DecisionTreeClassifier(max_depth = 2),
    n_estimators = i)
   accuracyTest = calc_testing_score(X_train, y_train, X_test, y_test, clf)
   accuracyTest2 = calc_testing_score(X_train, y_train, X_test, y_test, clf2)
```

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

```
71
72  '''
73  IV. Support Vector Machine 4/5
74  '''
75  from sklearn.svm import SVC
76  param_range = range(1000, 5003, 500)
77  for i in range(1000, 10003, 500):
78      clf = SVC(kernel = "poly", max_iter = i)
79      clf2 = SVC(kernel = "rbf", max_iter = i)
80      accuracyTest = calc_testing_score(X_train, y_train, X_test, y_test, clf)
81      accuracyTest2 = calc_testing_score(X_train, y_train, X_test, y_test,
82      clf2)
83
84  '''
85  5/5 Neural networks
86  '''
87  from sklearn.neural_network import MLPClassifier
88  # (a) numHiddenLayers
89  param_range = range(1, 8, 1)
90  tup = (100,)
91  for i in range(1, 8, 1):
92      tup2 = (i, 10)
93      clf = MLPClassifier(activation = 'relu', hidden_layer_sizes = tup2)
94      accuracyTest = calc_testing_score(X_train, y_train, X_test, y_test, clf)
95
96  # (b) size of number of neurons in the
97  param_range = range(1, 36, 5)
98  for i in range (1, 36, 5):
99      clf = MLPClassifier(hidden_layer_sizes = (i, i, i))
100     accuracyTest = calc_testing_score(X_train, y_train, X_test, y_test, clf)
```

References

Abdou, H, J Pointon and AE Masry (2008). Neural nets versus conventional techniques in credit scoring in Egyptian banking. *Expert Systems with Applications*, 35(2), 1275–1292.

Addo, PM, D Guegan and B Hassani (2018). Credit risk analysis using machine and deep learning models. *Risks*, 6(2), 38.

Atiya, AF (2001). Bankruptcy prediction for credit risk using neural networks: A survey and new results. *IEEE Transactions on Neural Networks*, 12(4), 929–935.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

Baesens, B, TV Gestel, S Viaene, M Stepanova, J Suykens and J Vanthiernen (2003). Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the Operational Research Society*, 54, 627–635.

Bahnsen, AC, D Aouada, and B Ottersten (2015). A novel cost-sensitive framework for customer churn predictive modeling. *Decision Analytics*, 2(5), 1–15.

Bahrammirzaee, A (2010). A comparative survey of artificial intelligence applications in finance: Artificial neural networks, expert system and hybrid intelligent systems. *Neural Computing and Applications*, 19(8), 1165–1195.

Borovykh, A, S Bothe, and C Oosterlee (2018). Conditional time series forecasting with convolutional neural networks. Retrieved June 15, 2018, from https://arxiv.org/abs/1703.04691v4.

Breiman, L (1996). Bagging predictors. *Machine Learning*, 24(2), 123–140.

Butaru, F, Q Chen, B Clark, S Das, AW Lo and A Siddique (2016). Risk and risk management in the credit card industry. *Journal of Banking and Finance*, 72, 218–239.

Bzdok, D, N Altman, and M Krzywinski (2018). Statistics versus machine learning. *Nature Methods*, 15(4), 233–234.

Caruana, R, A Munson, and A Niculescu-Mizil (2006). Getting the most out of ensemble selection. In *Proc. of the 6th Int. Conf. Data Mining*, Hong Kong, China, pp. 828–833. IEEE Computer Society.

Crook, JN, DB Edelman, and LC Thomas (2007). Recent developments in consumer credit risk assessment. *European Journal of Operational Research*, 183(3), 1447–1465.

Demšar, J (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7, 1–30.

Demyanyk, Y and I Hasan (2010). Financial crisis and bank failures: A review of prediction methods. *Omega*, 38(5), 315–324.

Desai, VS, JN Crook, and GA Overstreet (1996). A comparison of neural networks and linear scoring models in the credit union environment. *European Journal of Operational Research*, 95(1), 24–47.

Fernandes, GB and R Artes (2016). Spatial dependence in credit risk and its improvement in credit scoring. *European Journal of Operational Research*, 249, 517–524.

Finlay, S (2011). Multiple classifier architectures and their application to credit risk assessment. *European Journal of Operational Research*, 210(2), 368–378.

Freund, Y and RE Schapire (1996). Experiments with a new boosting algorithm. In *Proc. of the 13th Int. Conf. Machine Learning*, Bari, Italy, pp. 148–156. Morgan Kaufmann.

García, S and F Herrera (2008). An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research*, 9, 2677–2694.

Garrido, F, W Verbeke, and C Bravo (2018). A robust profit measure for binary classification model evaluation. *Expert Systems with Applications*, 92, 154–160.

Gu, S, B Kelly, and D Xiu (2018). Empirical asset pricing via machine learning. Technical Report No. 18–04, Chicago Booth Research Paper.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.

Hand, DJ and WE Henley (1997). Statistical classification methods in consumer credit scoring: A review. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 160(3), 523–541.

Hastie, T, R Ribshirani and J Friedman (2008). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer.

Heaton, JB, NG Polson and JH White (2017). Deep learning for finance: Deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(3), 3–12.

James, G, D Witten, T Hastie and R Tibshirani (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer.

Keerthi, SS and C-J Lin (2003). Asymptotic behaviors of support vector machines with gaussian kernel. *Neural Computation*, 15(7), 1667–1689.

Kim, HS and SY Sohn (2010). Support vector machines for default prediction of SMEs based on technology credit. *European Journal of Operational Research*, 201(3), 838–846.

Ko, AHR, R Sabourin and JAS Britto (2008). From dynamic classifier selection to dynamic ensemble selection. *Pattern Recognition*, 41(5), 1718–1731.

Kumar, PR and V Ravi (2007). Bankruptcy prediction in banks and firms via statistical and intelligent techniques — a review. *European Journal of Operational Research*, 180(1), 1–28.

Lessmann, S, B Baesens, H-V Seow and LC Thomas (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of resesarch. *European Journal of Operational Research*, 247, 124–136.

Maldonado, S, J Pérez, and C Bravo (2017). Cost-based feature selection for support vector machines: An application in credit scoring. *European Journal of Operational Research*, 261, 656–665.

Malhotra, R and DK Malhotra (2002). Differentiating between good credits and bad credits using neuro-fuzzy systems. *European Journal of Operational Research*, 136(1), 190–211.

Mitchell, T (1997). *Machine Learning*. McGraw Hill.

Moeyersoms, J and D Martens (2015). Including high-cardinality attributes in predictive models: A case study in churn prediction in the energy sector. *Decision Support Systems*, 72, 72–81.

Mohri, M, A Rostamizadeh, and A Talwalkar (2012). *Foundations of Machine Learning*. MIT Press.

Paleologo, G, A Elisseeff, and G Antonini (2010). Subagging for credit scoring models. *European Journal of Operational Research*, 201(2), 490–499.

Partalas, I, G Tsoumakas, and I Vlahavas (2010). An ensemble uncertainty aware measure for directed hill climbing ensemble pruning. *Machine Learning*, 81(3), 257–282.

Putra, EF and R Kosala (2011). Application of artificial neural networks to predict intraday trading signals. In *Proc. of 10th WSEAS Int. Conf. on E-Activity*, Jakatar, Island of Java, pp. 174–179.

Raschka, S (2015). *Python Machine Learning*. Birmingham, UK: Packt.

Russell, S and P Norvig (2010). *Artificial Intelligence: A Modern Approach* (3 edn.). Prentice Hall.

Samuel, AL (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3), 210–229.

Solea, E, B Li and A Slavković (2018). Statistical learning on emerging ecoonomies. *Journal of Applied Statistics*, 45(3), 487–507.

Sun, T and MA Vasarhelyi (2018). Predicting credit card delinquencies: An application of deep neural network. *Intelligent Systems in Accounting, Finance and Management*, 25(4), 174–189.

Thomas, LC (2000). A survey of credit and behavioral scoring: Forecasting financial risk of lending to consumers. *International Journal of Forecasting*, 16(2), 149–172.

Verbraken, T, C Bravo, R Weber and B Baesens (2014). Development and application of consumer credit scoring models using profit-based classification measures. *European Journal of Operational Research*, 238(2), 505–513.

Woloszynski, T and M Kurzynski (2011). A probabilistic model of classifier competence for dynamic ensemble selection. *Pattern Recognition*, 44(10–11), 2656–2668.

Yang, YX (2007). Adaptive credit scoring with kernel learning methods. *European Journal of Operational Research*, 183(3), 1521–1536.

Yeh, I-C and C-H Lien (2009). The comparisons of data mining techniques for the predictive accuracy of probabiltiy of default of credit card clients. *Expert Systems with Applications*, 36, 2473–2480.

Zopounidis, C, M Doumpos, and NF Matsatsinis (1997). On the use of knowledge-based decision support systems in financial management: A survey. *Decision Support Systems*, 20(3), 259–277.

Rev. Pac. Basin Finan. Mark. Pol. 2019.22. Downloaded from www.worldscientific.com
by UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL on 10/30/22. Re-use and distribution is strictly not permitted, except for Open Access articles.