International Journal of Computer Applications (0975 8887)
Volume 118 - No. 19, May 2015

A Two-Tier Classification Model for Financial Fraud Detection

Fazlul Hoque
Dept. of Computer Science & Engineering,
United International University,
Dhaka-1209, Bangladesh.

Md. Jahidul Islam
Dept. of Computer Science & Engineering,
United International University,
Dhaka-1209, Bangladesh.

Swakkhar Shatabda
Dept. of Computer Science & Engineering,
United International University,
Dhaka-1209, Bangladesh.

ABSTRACT

Financial fraud has become a daunting challenge for the business companies and baking organizations worldwide. The development of new technologies has provided further and more complicated ways in which criminals commit fraud that result in disastrous consequences. In this paper, we propose a Linear Discriminant Analysis-based novel financial fraud detection model which performs a two-tier classification based on three separate linear discriminant functions. Each function performs its own classification based on the training data and derives its own decision boundary for classification. Then, our two-tier model takes the final classification decision by utilizing the individual decisions of these discriminant functions. We evaluate the performance of our model using real-life datasets in terms of several standard metrics. Besides, we compare the performance of our model with that of several other models found in the literature. Our experimental results suggest that our model achieve reasonably improved classification performance compared to the state-of-the-art ones.

Keywords:

Financial Fraud Detection, Linear Discriminant Analysis, Multi-level Learning.

1. INTRODUCTION

Financial fraud has received great deal of attention in recent years [1, 2, 3]. The number of reported incidents of credit card fraud, corporate fraud, and money laundering fraud have been skyrocketing in an alarming rate. With the evolution of modern technology and the globalization of communication, the monetary loss and organizational consequences have become catastrophic. This alarming phenomena is formally defined [1] as:

> "A deliberate act that is contrary to law, rule, or policy with intent to obtain unauthorized financial benefit".

Financial fraud causes billions of dollars of loss worldwide each year [1, 3]. Consequently, Financial Fraud Detection (FFD) have become a very important area of research to prevent such devastating consequences. FFD implicates to distinguish fraudulent data from accurate data in order to enable the decision makers to develop appropriate strategies to decrease the impact of fraud in an automated way.

Statistical and data mining methods have been widely used [1, 4, 5, 6, 7, 8] to extract hidden information and patterns from very large quantities of data. The process of applying a computer based data methodology to discover knowledge from data—is called data mining. It uses statistical, mathematical, artificial intelligence, and machine learning techniques to extract and identify useful information and subsequently gain knowledge from large database. Fraud detection has become one of the leading established application [6, 7] of data mining in both industry and government. Different types of data mining techniques, such as Artificial Neural Networks [8, 9, 10], Naive Bayesian method [8, 9], Decision Tree (DT) models [11, 12, 13], Support Vector Machines (SVM) [14], Logistic Regression (LR) [8, 15], etc., have been used in recent studies. These automated classification models liberate the decision makers from daunting manual labors that include complex and time-consuming investigations.

In this paper, we adopt a Linear Discriminant Analysis (LDA) based two-tier classification model for classifying fraudulent transaction using three different linear discriminant functions. Each function finds its own linear combination of features to define a decision boundary that separates the fraudulent transactions from the good ones. Our two-tier model then utilizes the decision of these individual discriminant functions and takes the final classification decision. In addition to designing this model, we investigate its performance in terms of several standard metrics. Besides, we compare its performance with that of several state-of-the-art data mining models. Consequently, we make the following contributions in this paper:

- We propose a two-tier classification model for FFD based on LDA. LDA has been successfully used for several data mining and machine learning based applications [16], [17], [18]. However, to the best of our knowledge, a proper and thorough investigation of LDA based FFD model is still unexplored in the literature.
- We design three linear discriminant functions to accommodate our model. These three functions learns their separate decision boundary based on the given training data and make classification decision. Based on their decision, we make the final classification decision, to classify new instance of the test data.
- In addition, we evaluate the performance of our model using real-life datasets, in terms of several standard metrics. Furthermore, we compare this performance with that of several other approaches found in the literature. Our simulation results sug-

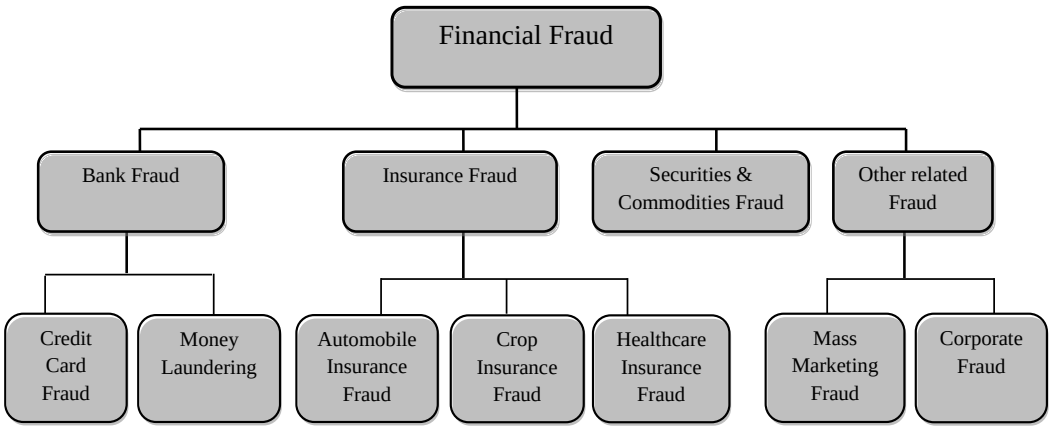

Fig. 1: Different types of financial frauds

gest that our two-tier model achieve reasonably improved performance compared to these state-of-the-art models.

The rest of the paper is organized as follows: in Section 2, we highlight different state-of-the-art data mining and machine learning techniques for different types of FFD. Then, in Section 3, we describe the working principle of our proposed two-tier classification model. Subsequently, we present the performance evaluation of our model in Section 4. Finally, in Section 5, we draw our conclusions and highlight some future work.

2. RELATED WORK

There are a number of research studies [1, 4, 5, 6, 7, 8, 19, 20, 21, 22, 23, 24] that use data mining-based classification models for FFD. The design of these models differ for different categories of FFD applications. There are a number of ways in which criminals commit fraud. We summarize the most widely reported financial frauds [1] in Fig. 1. The major categories of financial fraud are bank fraud, securities and commodities fraud, insurance fraud, and other related financial fraud. These categories can be further classified to accommodate credit card fraud, money laundering fraud, automobiles insurance fraud, health insurance fraud, marketing fraud, corporate fraud, etc.

For such a variety of FFD applications, a number of different data mining and machine learning techniques are proposed in the literature. In particular, supervised learning based classification models are the most widely used ones for FFD. These models use training data for learning the hidden patterns between fraud and good instances. Then, it uses the learned information to classify a new instance. Most commonly used classification techniques include Artificial Neural Networks [8, 9, 10], Logistic Regression (LR) [8, 15], Naive Bayesian method [8, 9], Decision Tree (DT) models [11, 12, 13], etc. Besides, K-Nearest Neighbor [25] is often applied in automobiles insurance and credit card fraud detection. In addition, Bayesian belief network [15] and Support Vector Machine (SVM) [14] are used to generate adaptive and robust FFD models. Furthermore, other regression-based statistical learning techniques [8, 25] are used mostly for credit card and corporate fraud detection.

On the other hand, clustering is the most widely used unsupervised learning for FFD. Most commonly used clustering techniques [8, 24] are based on K-means algorithm, Naive Bayesian model, self-organizing map, etc.

Although a wide range of learning algorithms and classifications models are used for FFD, a proper and thorough LDA for FFD is still unexplored in the literature,which has been successfully used for several data mining and machine learning based applications [16, 17, 18]. Besides, all the classification techniques adopt a single step learning using the whole dataset. Using the fraudulent and good instances separately to investigate hidden patterns in each case to develop multi-level classification models, is yet to be explored in the literature.

3. PROPOSED FFD MODEL

In this section, we discuss our proposed FFD model in details. We adopt a Linear Discriminant Analysis (LDA) based FFD model. At first, we discuss a basic LDA based classification model. Then, we present our two-tier classification model, its training and testing mechanisms, and corresponding algorithm.

3.1 Basic LDA Model for FFD

LDA is widely used for classification ([16, 17, 18]) in the area of machine learning and data mining. It finds a linear combination of features to define a decision boundary that separates two or more classes.

Let us consider a training dataset, D having $|D|$ instances. Each instance has d numeric attributes, i.e., features. That is, each feature vector $\vec{x}_i$ ($i \in \{1, 2, \ldots, |D|\}$) is a d-dimensional numeric vector, that corresponds to a class C_i. Here, C_i can be either 0 or 1, denoting good and fraud transactions, respectively. For this dataset, the objective LDA is to find a decision boundary $g(\vec{x})$, which is given by the following equation:

$$g(\vec{x}) = \vec{w^t} \cdot \vec{x} + w_0 = \sum_{j=0}^{d} w_j x_j \quad (1)$$

Here, $\vec{w}$ is the weight vector that we need to find and the value of x_0 is taken as 1. Based on the value of $g(\vec{x})$, the classification decision is made using the following equation:

$$C_i = \begin{cases} 0 & \text{, if } g(\vec{x}) \geq 0 \\ 1 & \text{, Otherwise} \end{cases} \quad (2)$$

3.2 Our Two-Tier Model

We design a two-tier classification model for classifying fraudulent transaction using three different linear discriminant functions. Each function finds its own linear combination of features to define a decision boundary that separates the fraudulent transactions from the good ones. Our two-tier model then utilizes the decision of these individual discriminant functions and takes the final classification decision. Now, we explain this model with mathematical and algorithmic formulation.

We present the parameters and notations that we use in our model in Table 1. Here, $\vec{Y_i^g}$ and $\vec{Y_i^f}$ are two distance vectors for the i^{th} instance of D. These two vectors numerically represent how distant the feature vector $\vec{x}$ is, from a typical (i. e, average) good and fraud instance, respectively. Values of the j^{th} element of $\vec{Y_i^g}$ and $\vec{Y_i^f}$ is 1, if the value of x_{ij} is more than one standard deviation apart from the corresponding mean feature values: m_j^g and m_j^f, respectively. Otherwise, values of the j^{th} element of $\vec{Y_i^g}$ and $\vec{Y_i^f}$ is the fraction of its distance in one standard deviation.

In a basic LDA model, we feed the feature vector $\vec{x}$ to the discriminant function so that it finds a decision boundary between good and fraud instances in the feature space. In our model, we feed the distance vectors, i.e., $\vec{Y_i^g}$ and $\vec{Y_i^f}$ to the discriminant function to generate decision boundaries between good and fraud instances in the distance vector space.

First, we consider the following discriminant function for the i^{th} instance in D:

$$g_g(\vec{Y_i^g}) = \vec{w_g^t} \cdot \vec{Y_i^g} + w_{g0} \quad (3)$$

Here, $\vec{Y_i^g}$ is the distance vector in terms of $\vec{\Delta_i^g}$. We find the weight vector $\vec{w_g}$ so that we get the decision boundary $g_g(\vec{Y_i^g})$ to correctly classify the instance using the following equation:

$$C_{gi} = \begin{cases} 0 & \text{, if } g_g(\vec{Y_i^g}) \geq 0 \\ 1 & \text{, Otherwise} \end{cases} \quad (4)$$

Similarly, using the distance vector in terms of $\vec{\Delta_i^f}$, i.e., $\vec{Y_i^f}$, we perform the following LDA:

$$g_f(\vec{Y_i^f}) = \vec{w_f^t} \cdot \vec{Y_i^f} + w_{f0} \quad (5)$$

Here, we find the weight vector $\vec{w_f}$ so that we get the decision boundary $g_f(\vec{Y_i^f})$ to correctly classify the instance using the following equation:

$$C_{fi} = \begin{cases} 0 & \text{, if } g_f(\vec{Y_i^f}) \geq 0 \\ 1 & \text{, Otherwise} \end{cases} \quad (6)$$

In addition to these two linear discriminant function, we consider the basic state-of-the-art one as well. That is, for the i^{th} instance x_i, we consider the following function:

$$g_b(\vec{x_i}) = \vec{w_b^t} \cdot \vec{x_i} + w_{b0} \quad (7)$$

Here, we find the weight vector $\vec{w_b}$ so that we get the decision boundary $g_b(\vec{x_i})$ to correctly classify the instance using the following equation:

$$C_{bi} = \begin{cases} 0 & \text{, if } g_b(\vec{x_i}) \geq 0 \\ 1 & \text{, Otherwise} \end{cases} \quad (8)$$

The discriminant function $g_b(\vec{x_i})$ finds a decision boundary based on the distribution of the instances in feature space. Whereas, $g_g(\vec{Y_i^g})$ and $g_f(\vec{Y_i^f})$ find decision boundaries based on the distribution of the two distance vectors of the instances. That is, in our model we takes three characteristics of an instance into account:

- how distant the feature values of the instance from a mean good instance,
- how distant the feature values of the instance from a mean fraud instance, and
- how the feature values of the instance are distributed in the feature space.

In the above discussion, we presented the linear discriminant functions that we use in our model. In the following subsections, we illustrate how we use our model for training and testing.

*3.2.1 **Training: Finding The Weight Vectors**.*

We train all the three linear discriminant functions separately over our training dataset D. We use the most widely used gradient descent technique [16] to find the weight vectors: $\vec{w_g}$, $\vec{w_f}$, and $\vec{w_b}$. A basic gradient descent tries to minimize a loss function $J(\vec{w})$ in an iterative manner. The iterative rule of a basic gradient descent is:

$$\vec{w} := \vec{w} - \alpha \nabla J(\vec{w})$$

Here, α is the learning rate and $\nabla J(\vec{w})$ denotes the gradient of $J(\vec{w})$.

Now, we need to design a cost function $J(\vec{w})$ for our model. The most obvious choice is to take $J(\vec{w})$ as the number of misclassified instances by $\vec{w}$. However, in that case, $J(\vec{w})$ would be piecewise constant, and therefore, would be a poor candidate for a gradient search [16]. In stead, we consider the *perceptron criterion function* as our cost function, defined as:

$$J(\vec{w}) = \sum_{x \in D_m} (-\vec{w} \cdot \vec{x})$$

Here, D_m is the set of all misclassified instances. This definition of $J(\vec{w})$ follows that,

$$\nabla J(\vec{w}) = \sum_{x \in D_m} (-x)$$

Therefore, the gradient descent rule simplifies to,

$$\vec{w} := \vec{w} + \alpha \sum_{x \in D_m} (x)$$

We present the basic perceptron algorithm in Algorithm 1. For all the three linear discriminant functions, we separately apply Algorithm 1 to find out its corresponding weights. At first, we initialize the weight vector randomly (line 4), and set the value of learning rate (line 5). Then, we start the iterative process of updating weights (line $7-14$). We classify the instances of D using the current weight vector (line 9) and find out the misclassified instances

Table 1. : Parameters used in our model

Particulars	Definition	Governed equation
$\vec{x_i}$	i^{th} instance	$\vec{x_i} =< x_{i1}, x_{i2}, \ldots, x_{id} >$
D	Training dataset	$\vec{x_i} \in D$ for $i = \{1, 2, \ldots, \|D\|\}$
D_g	Subset of D with only *good* instances	$x_i \in D_g$ if $C_i = 0$, for $\forall x_i \in D$
D_f	Subset of D with only *fraud* instances	$x_i \in D_f$ if $C_i = 1$, for $\forall x_i \in D$
$\vec{M^g}$	Mean vector for D_g	$M^g =< m_j^g >$, where $m_j^g = \frac{1}{\|D_g\|} \times \sum_{k=1}^{\|D_g\|} x_{kj}$ $(j = \{1, 2, \ldots, d\})$
$\vec{M^f}$	Mean vector for D_f	$M^f =< m_j^f >$, where $m_j^f = \frac{1}{\|D_f\|} \times \sum_{k=1}^{\|D_f\|} x_{kj}$ $(j = \{1, 2, \ldots, d\})$
$\vec{\Delta^g}$	Standard deviation vector for D_g	$\Delta^g =< \delta_j^g >$, where $\delta_j^g = \frac{1}{\|D_g\|} \times \sum_{k=1}^{\|D_g\|} (x_{kj} - m_j^g)^2$ $(j = \{1, 2, \ldots, d\})$
$\vec{\Delta^f}$	Standard deviation vector for D_f	$\delta^f =< \delta_j^f >$, where $\delta_j^f = \frac{1}{\|D_f\|} \times \sum_{k=1}^{\|D_f\|} (x_{kj} - m_j^f)^2$ $(j = \{1, 2, \ldots, d\})$
$\vec{Y_i^g}$	Distance vector for i^{th} instance of D in terms of Δ_i^g	$Y_i^g =< y_{ij}^g >$, where $y_{ij}^g = \begin{cases} 1 & \text{, if } \|x_{ij} - m_j^g\| > \delta_j^g \\ \frac{\|x_{ij} - m_j^g\|}{\delta_j^g} & \text{, Otherwise} \end{cases}$ $(j = \{1, 2, \ldots, d\})$
$\vec{Y_i^f}$	Distance vector for i^{th} instance of D in terms of Δ_i^f	$Y_i^f =< y_{ij}^f >$, where $y_{ij}^f = \begin{cases} 1 & \text{, if } \|x_{ij} - m_j^f\| > \delta_j^f \\ \frac{\|x_{ij} - m_j^f\|}{\delta_j^f} & \text{, Otherwise} \end{cases}$ $(j = \{1, 2, \ldots, d\})$

(line 10). Next to that, we calculate the gradient of the loss function (line 11). Finally we update the value of the weight vector (line 12) and learning rate (line 13). We terminate the iterative process until the value of $|\alpha \nabla J(\vec{w})|$ falls below the predefined threshold θ. We will discuss how we adopt the values of learning rate α and threshold θ , in Section 4.

Algorithm 1: Basic perceptron algorithm

1 **Input:**Feature vector $\vec{y}$, Threshold θ
2 **Output:**Weight vector $\vec{w}$
3 *begin*
4 $\vec{w} \leftarrow$ initialize randomly
5 $\alpha \leftarrow$ initialize learning rate
6 $iteration \leftarrow 0$
7 Do
8 $iteration \leftarrow iteration + 1$
9 Classify the instances of D
10 $D_m \leftarrow$ all misclassified instances of D
11 Calculate $\nabla J(\vec{w}) = \sum_{x \in D_m} (-x)$
12 $\vec{w} := \vec{w} - \alpha \nabla J(\vec{w})$
13 Adapt the value of α
14 Until($|\alpha \nabla J(\vec{w})| > \theta$)
15 **return** $\vec{w}$
16 *end*

3.2.2 ***Testing: Classifying New Instances.***
We adopt a two-tier model for classifying new instances. We illustrate this classification process in Fig. 3. In the first tier, for the input feature vector $\vec{x}$, we determine the distance vectors, i.e., $\vec{Y_x^g}$ and $\vec{Y_x^f}$. In Table 1, we defined of these distance vectors for i^{th} training instance. For testing, we calculate $\vec{Y_x^g}$ and $\vec{Y_x^f}$, for $\vec{x}$, using the following equations:

Table 2. : Classification pattern of our model

C_{gx}	C_{fx}	C_{bx}	Final decision on $\vec{x} : C_x$
0	0	–	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	–	1

$$Y_x^g =< y_{xj}^g > \text{, where } y_{xj}^g = \begin{cases} 1 & \text{, if } |x_j - m_j^g| > \delta_j^g \\ \frac{|x_j - m_j^g|}{\delta_j^g} & \text{, Otherwise} \end{cases} \quad (9)$$

$$Y_x^f =< y_{xj}^f > \text{, where } y_{xj}^f = \begin{cases} 1 & \text{, if } |x_j - m_j^f| > \delta_j^f \\ \frac{|x_j - m_j^f|}{\delta_j^f} & \text{, Otherwise} \end{cases} \quad (10)$$

Then, we feed $\vec{Y_x^g}$ and $\vec{Y_x^f}$, to g_g and g_f for classification. If they both refer to same class, i.e., $C_{gx} = C_{fx}$, then we classify the instance immediately. That is, we conclude that $\vec{x}$ belongs to class $C_x = C_{gx} = C_{fx}$. On the other hand, if $C_{gx} \neq C_{fx}$, we perform the second tier of classification by feeding $\vec{x}$ to g_b and evaluate C_{bx}. Subsequently, we classify the instance by $C_x = C_{bx}$. In Table 2, we present the classification decision made in our model in terms of different combinations of values of C_{gx}, C_{fx}, and C_{bx}.

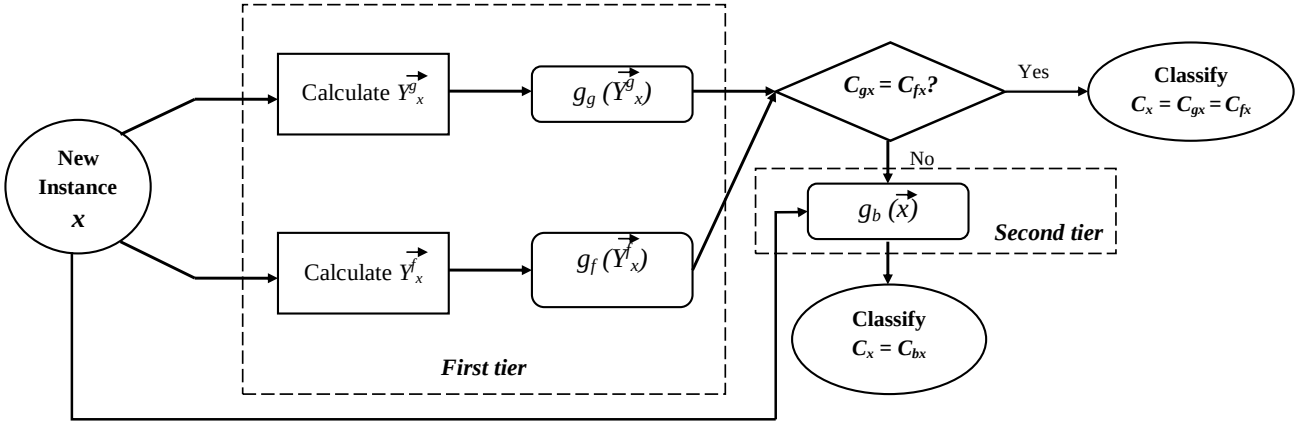

Fig. 2: Two tier classification model for new instances

We have discussed the training and testing procedure of our approach. Next, we discuss the performance evaluation of our approach.

4. PERFORMANCE EVALUATION

In this section, implement our model for performance evaluation. At first, we present the datasets that we consider for performance evaluation. Then, we illustrate the parameter settings of our model. Finally, we evaluate the performance of our model in terms of several standard metrics. Furthermore, we compare this performance with that of other state-of-the art models.

4.1 Dataset

We use two real-life datasets in our performance evaluation:

- **Dataset-1 (sales data):** our first dataset is presented and analyzed in [26], whose data refer to the transactions reported by the salespeople of some company. It has $15,772$ instances of data, each having 4 attributes.
- **Dataset-2 (credit-card data):** our second dataset is presented in [27], whose data refer to credit-card applications. It has 690 instances of data, each having 14 attributes.

In both cases, there are two classes: ok and $fraud$ (i.e., class values equal 0 and 1 respectively). In addition, we considered the nominal and numeric valued attributes only. The original datasets contain a number of instances with missing attribute values. We handle these missing attribute values using the preprocessing steps mentioned in [26]. These preprocessing or data-cleaning steps include:

- *Deleting some instances:* if the missing attribute values cannot be approximated with reasonable confidence.
- *Filling in some attribute values:* using mean values or most likely values for that attribute.

For further information regarding the data and the data-cleaning procedures, we refer to [26] and [27].

4.2 Experimental Setup

We conducted our experiments using a Windows machine with 3.3 GHz quad-core Intel Core-i5 Processor. We used R programming language to implement our model. For implementation, we used the built-in packages for machine learning in R [26, 28]. That is, the parameter settings, such as the initial weight vectors and adaptive learning rate in LDA, etc., belong to the default ones implemented in the basic built-in packages.

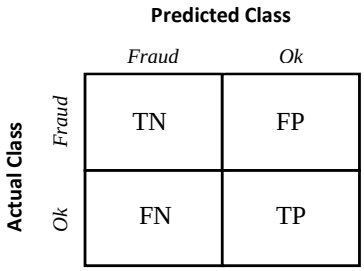

Fig. 3: A confusion matrix layout

In our performance evaluation, we used 10-fold cross validation mechanism [16] for testing. We adopt accuracy, precision, recall, specificity, and F-measure (F1) as our performance metrics. These are the most commonly used performance metrics used for evaluating classification models [16]. Values of these metrics are calculated by the confusion matrix entities. Confusion matrix is a layout (shown in Fig. 3) that allows us to visualize the performance of classification algorithms in terms of the above mentioned metrics.

Lets denote P as the total number of positive examples (i.e., class level = ok) and N as the total number of negative examples (i.e., class level = $fraud$). Therefore, $P + N$ is the size of the test set. Besides, TP (True Positives) and TN (True Negatives) denote the total number of positive and negative instances that are correctly classified, respectively. In addition, FP is the number of incorrect predictions that an instance is positive and FN is the number of incorrect of predictions that an instance is negative. Using these confusion matrix entities, we define our performance metrics in Table 3.

Table 3. : Definition of our performance metrics in terms of confusion matrix entities. Here, in our example, *positive* and *negative* instances refer to the *good* & *fraud* instances, respectively.

Metric	**Governed equation**	**Definition**
Accuracy	$\frac{TP+TN}{P+N}$	Proportion of the total number of predictions that are correct
Precision	$\frac{TP}{TP+FP}$	Proportion of the predicted positive cases that are correct
Recall/Sensitivity	$\frac{TP}{TP+FN}$	Proportion of positive cases that are correctly identified
Specificity	$\frac{TN}{FP+TN}$	Proportion of negative cases that are correctly identified
F-measure (F1)	$\frac{2\times precision\times recall}{precision+recall}$	Weighted harmonic mean of precision and recall

Table 4. : Performance evaluation of T-LDA compared to other state-of-the-art techniques (Here, $+$, $-$, and $\sim$ represent improved, declined, and similar performance of T-LDA compared to the corresponding algorithm.)

(a) For dataset-1 (*sales data*)

Models	Accuracy	Precision	Recall	Specificity	F-measure (F1)
T-LDA	**0.97**	**0.91**	**0.99**	**0.88**	**0.95**
Naive Bayes	$0.79+$	$0.84+$	$0.92+$	$0.78+$	$0.88+$
ANN	$0.92+$	$0.81+$	$0.77+$	$0.92-$	$0.79+$
SVM	$0.91+$	$0.92\sim$	$0.98\sim$	$0.81+$	$0.95\sim$
CART	$0.97\sim$	$0.90\sim$	$0.82+$	$0.99-$	$0.86+$
1-NN	$0.93+$	$0.98-$	$0.94+$	$0.66+$	$0.96\sim$
3-NN	$0.94+$	$0.98-$	$0.95+$	$0.64+$	$0.96\sim$
5-NN	$0.92+$	$0.96-$	$0.95+$	$0.48+$	$0.95\sim$
Rank	$6+, 1\sim, 0-$	$2+, 2\sim, 3-$	$6+, 1\sim, 0-$	$5+, 0\sim, 2-$	$3+, 4\sim, 0-$

(b) For dataset-2 (*credit-card data*)

Models	Accuracy	Precision	Recall	Specificity	F-measure (F1)
T-LDA	**0.91**	**0.90**	**0.96**	**0.78**	**0.93**
Naive Bayes	$0.54+$	$0.71+$	$0.55+$	$0.54+$	$0.62+$
ANN	$0.67+$	$0.73+$	$0.41+$	$0.68+$	$0.52+$
SVM	$0.61+$	$0.76+$	$0.69+$	$0.37+$	$0.73+$
CART	$0.89\sim$	$0.86+$	$0.80+$	$0.94-$	$0.83+$
1-NN	$0.54+$	$0.67+$	$0.65+$	$0.31+$	$0.66+$
3-NN	$0.58+$	$0.67+$	$0.73+$	$0.25+$	$0.70+$
5-NN	$0.59+$	$0.67+$	$0.80+$	$0.16+$	$0.73+$
Rank	$6+, 1\sim, 0-$	$7+, 0\sim, 0-$	$7+, 0\sim, 0-$	$6+, 0\sim, 1-$	$7+, 0\sim, 0-$

4.3 Results and Discussion

We name our Two-tier LDA-based classification model *T-LDA*. In Table 4, we present the performance evaluation of T-LDA in terms of the above mentioned performance metrics. In addition, we compare its performance with few most widely used state-of-the art classification models, which are based on Naive Bayes [9], Artificial Neural Networks (ANN) [10], Support Vector Machine (SVM) [14], Classification & Regression Tree (CART), and K-Nearest Neighbor (KNN) [25] algorithm. For KNN, we considered $k = 1$, 3, and 5. As we mentioned in our experimental settings (Section 4.2), we use the built-in packages of R programming language to implement these algorithms. For detailed implementation procedure and other parameter settings, we refer to [26, 28].

In Table 4(a), we present the performance evaluation for dataset-1 (*sales data*), which suggests that T-LDA performs reasonably good in terms of all the parameters. Specifically, in terms of accuracy, recall, and F-measure, T-LDA performs the best.

T-LDA produces even better performance for dataset-2 (*credit-card data*). As shown in Table 4(b), it performs the best in terms of accuracy, precision, recall, and F-measure.

Based on the performance evaluation presented in Table 4, we find that T-LDA produces the most consistent performance overall, in terms of all performance metrics. This suggests that our two-tier model is capable of detecting both fraud and good instances accurately. Besides, two datasets that we used are diverse in nature. Dataset-1 contains transaction information about customer sales data with each instance having 4 attribute values. On the other hand,

Table 5. : Improved performance of T-LDA over basic LDA

(a) For dataset-1 (*sales data*)

Models	Accuracy	Precision	Recall	Specificity	F1
T-LDA	**0.97**	**0.91**	**0.99**	**0.88**	**0.95**
Basic LDA	0.83	0.86	0.84	0.83	0.85
Improvement	17%	6%	18%	6%	12%

(b) For dataset-2 (*credit-card data*)

Models	Accuracy	Precision	Recall	Specificity	F1
T-LDA	**0.91**	**0.90**	**0.96**	**0.78**	**0.93**
Basic LDA	0.68	0.79	0.65	0.64	0.71
Improvement	34%	14%	48%	22%	31%

Dataset-2 contains transaction information about credit-card data with each instance having 14 attribute values. Therefore, our model is robust enough to perform consistently well in such diverse FFD scenarios.
Furthermore, to investigate the effectiveness of our two-tier model, we compare the performance of T-LDA with the performance of basic LDA alone. We present this performance comparison in Table 5, which shows significant performance improvement of T-LDA over the basic one. This validates our motivation of introducing the additional first tier of classification that utilizes *distance*-based feature vector in terms of good and fraud instances separately. This parallel classification improves the performance of basic LDA-based model and facilitates improved, robust, and consistent performance over varied FFD scenarios.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a novel classification model for financial fraud detection based on linear discriminate analysis. We designed a two-tier classification model based on three intuitive linear discriminant function. These functions learns their separate decision boundaries based on training data and make classification decision. Based on their decision, we make the final classification decision to classify new instances. In addition, we evaluated the performance of our model using real-life datasets and compared its performance with several state-of-the-art approaches. Our simulation results suggest that our model achieve more consistent and robust performance compared to the state-of-the-art ones.
With proper tuning, our two-tier classification model can be utilised for all types of financial fraud detection which we have mentioned in our paper. However, it is suited mostly for linearly separable cases. In our future work, we want to extend our model to accommodate linearly non-separable cases as well. In addition, investigating other learning algorithms and using more real-life datasets for performance evaluation are the two other research directions, that we want to follow in our future work.

6. REFERENCES

[1] EWT Ngai, Yong Hu, YH Wong, Yijun Chen, and Xin Sun. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision Support Systems*, 50(3):559–569, 2011.

[2] Zabihollah Rezaee. *Financial statement fraud: prevention and detection*. John Wiley & Sons, 2002.

[3] G Robert Blakey. Organized crime: The rise and fall of the mob. *Notre Dame Legal Studies Paper*, (09-46), 2009.

[4] Mehmed Kantardzic. *Data mining: concepts, models, methods, and algorithms*. John Wiley & Sons, 2011.

[5] Stijn Viaene, Richard A Derrig, Bart Baesens, and Guido Dedene. A comparison of state-of-the-art classification techniques for expert automobile insurance claim fraud detection. *Journal of Risk and Insurance*, 69(3):373–421, 2002.

[6] William J Frawley, Gregory Piatetsky-Shapiro, and Christopher J Matheus. Knowledge discovery in databases: An overview. *AI magazine*, 13(3):57, 1992.

[7] EWT Ngai, Yong Hu, YH Wong, Yijun Chen, and Xin Sun. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision Support Systems*, 50(3):559–569, 2011.

[8] Efstathios Kirkos, Charalambos Spathis, and Yannis Manolopoulos. Data mining techniques for the detection of fraudulent financial statements. *Expert Systems with Applications*, 32(4):995–1003, 2007.

[9] Sam Maes, Karl Tuyls, Bram Vanschoenwinkel, and Bernard Manderick. Credit card fraud detection using bayesian and neural networks. In *Proceedings of the 1st international naiso congress on neuro fuzzy technologies*, 2002.

[10] Michael J Cerullo and Virginia Cerullo. Using neural networks to predict financial reporting fraud: Part 2. *Computer Fraud & Security*, 1999(6):14–17, 1999.

[11] Efstathios Kirkos, Charalambos Spathis, and Yannis Manolopoulos. Data mining techniques for the detection of fraudulent financial statements. *Expert Systems with Applications*, 32(4):995–1003, 2007.

[12] S Kotsiantis, E Koumanakos, D Tzelepis, and V Tampakas. Forecasting fraudulent financial statements using data mining. *International Journal of Computational Intelligence*, 3(2):104–110, 2006.

[13] Belinna Bai, Jerome Yen, and Xiaoguang Yang. False financial statements: characteristics of china's listed companies and cart detecting approach. *International journal of information technology & decision making*, 7(02):339–359, 2008.

[14] Rong-Chang Chen, Tung-Shou Chen, and Chih-Chiang Lin. A new binary support vector system for increasing detection rate of credit card fraud. *International Journal of Pattern Recognition and Artificial Intelligence*, 20(02):227–239, 2006.

[15] Timothy B Bell and Joseph V Carcello. A decision aid for assessing the likelihood of fraudulent financial reporting. *Auditing: A Journal of Practice & Theory*, 19(1):169–184, 2000.

[16] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification*. John Wiley & Sons, 2012.

[17] Suresh Balakrishnama and Aravind Ganapathiraju. Linear discriminant analysis-a brief tutorial. *Institute for Signal and information Processing*, 1998.

[18] Kamran Etemad and Rama Chellappa. Discriminant analysis for recognition of human face images. *JOSA A*, 14(8):1724–1733, 1997.

[19] Richard J Bolton and David J Hand. Statistical fraud detection: A review. *Statistical science*, pages 235–249, 2002.

[20] Zabihollah Rezaee. *Financial statement fraud: prevention and detection*. John Wiley & Sons, 2002.

[21] Hatice Uzun, Samuel H Szewczyk, and Raj Varma. Board composition and corporate fraud. *Financial Analysts Journal*, 60(3):33–43, 2004.

[22] EWT Ngai, Yong Hu, YH Wong, Yijun Chen, and Xin Sun. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision Support Systems*, 50(3):559–569, 2011.

[23] Jose R Dorronsoro, Francisco Ginel, C Sgnchez, and CS Cruz. Neural fraud detection in credit card operations. *Neural Networks, IEEE Transactions on*, 8(4):827–834, 1997.

[24] Stijn Viaene, Richard A Derrig, Bart Baesens, and Guido Dedene. A comparison of state-of-the-art classification techniques for expert automobile insurance claim fraud detection. *Journal of Risk and Insurance*, 69(3):373–421, 2002.

[25] I-Cheng Yeh and Che-hui Lien. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2):2473–2480, 2009.

[26] Luis Torgo. *Data mining with R: learning with case studies*. Chapman & Hall/CRC, 2010.

[27] Catherine Blake and Christopher J Merz. {UCI} repository of machine learning databases. 1998.

[28] Tobias Sing, Oliver Sander, Niko Beerenwinkel, and Thomas Lengauer. Rocr: visualizing classifier performance in r. *Bioinformatics*, 21(20):3940–3941, 2005.