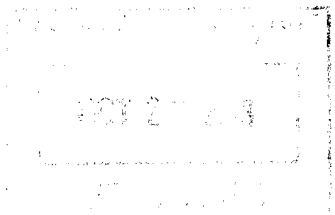
An Evaluation of Support Vector Machines in Consumer Credit Analysis



by

Benjamin A. Mattocks

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology
September 2013

Author: _____
Department of Electrical Engineering and Computer Science
September 9, 2013

Certified by: _____
Andrew W. Lo
Charles E. and Susan T. Harris Professor
Thesis Supervisor

Accepted by: _____
Professor Albert R. Meyer
Chairman, Masters of Engineering Thesis Committee

An Evaluation of Support Vector Machines in Consumer Credit Analysis

by
Benjamin A. Mattocks

Submitted to the Department of Electrical Engineering and Computer Science
on September 9, 2013, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Computer Science and Engineering

Abstract

This thesis examines a support vector machine approach for determining consumer credit. The support vector machine using a radial basis function (RBF) kernel is compared to a previous implementation of a decision tree machine learning model. The dataset used for evaluation was provided by a large bank and includes relevant consumer-level data, including transactions and credit-bureau data. The results suggest that a support vector machine offers similar performance to decision trees, but the parameters specifying the soft-margin constraint and the inverse-width used in the RBF kernel could significantly affect its performance.

Thesis Supervisor: Andrew W. Lo
Title: Charles E. and Susan T. Harris Professor

Acknowledgements

I would like to thank all of the following people for my experiences as I wrote my thesis:

I thank Andrew Lo, my thesis supervisor, for providing the vision for several possible areas of research which culminated in this thesis. His patience, wisdom, and guidance inspired me to pursue these ideas and provided a good scope for this thesis.

I thank David Fagnan for guiding me and providing the background information, data layouts, and pointing me in the right direction for code templates.

I would like to thank those in the Laboratory for Financial Engineering for setting up the server and the database which I used in my research. I also thank those who wrote code examples which were accessible on the server.

I would like to thank those at *Kx Systems* for use of their *kdb+* software,. I would also like to thank the authors of *Weka*.

I extend my thanks to my friends for putting up with me as I sat at my computer screen.

I would like to express my appreciation for my girlfriend, Kristen, who put up with me as I spent countless hours writing and organizing my thesis.

I thank my family members, Frances, Paul, and Brad, for their love and support in all my endeavors.

Contents

1 Introduction

1.1 Background

Consumer credit risk models are used to determine the feasibility of lending to consumers. The goal of these models is to use a consumer's characteristics to output a value in the form of a probability or binary value indicating if a consumer will become delinquent on their payments. Because a lot of data is involved, a model needs to incorporate statistical information about a customer. These estimates are based on a number of statistics, such as debt-to-income ratios, the number of days a consumer is delinquent on payment, current and past information about bank accounts, and transaction information. Consumer credit risk models were developed as a way to make credit determinations based on systematic objective computations instead of using subjective human judgement.

Most models currently in use today generate a score that determines the creditworthiness of a customer. These models are usually private and unique to institutions. Most of today's credit models are based on logistic regression (logit), discriminant analysis, or credit scores [1]. In general, they are a good indicator of consumer creditworthiness. However, the main drawback is that they do not incorporate the most recent market conditions and that the measures used to evaluate a consumer are long-term [12]. In particular, when conditions change rapidly, as seen in the financial crisis towards the end of 2008, these models will not capture such conditions and will lead to losses. Futhermore, due to the nature of these models, assumptions about the distribution of data must be made, often assuming that relationships between inputs and outputs are linear. In practice, there are other non-linear relationships which cannot be captured in these models [1].

In an attempt to capture non-linear relationships more accurately, machine learning has become an emerging technique in consumer credit analysis. Such models are not always restricted to making assumptions about data distributions to the extent of older statistical methods, such as logit and discriminant analysis. The most prevalent machine learning techniques in consumer credit research include decision trees, support vector machines (SVMs),

neural networks, and others [5].

1.2 Approach

Recently, various machine-learning models have been created to perform consumer credit analysis. Decision trees have been used in some studies [11, 12], and others [16, 13, 10] have used SVMs, a widely-used machine learning algorithm which will be used for this study. Machine learning techniques have the ability to detect non-linear relationships among a consumer's statistics which are ignored by common credit models which rely on data assumptions.

This study expands upon the work of Khandani et al. [12], in which decision trees, a machine learning technique, were compared to a traditional credit analysis algorithm. They examined a dataset from a commercial bank, which will be referred to as "The Bank", over the course of four years and showed that patterns pertaining to consumers' bank accounts, such as expenditures, savings, and debts, could be used to develop a model that vastly improves upon traditional consumer credit default algorithms. To further investigate the potential for machine learning in the problem of consumer credit determination, this study compares a support vector machine model to the decision tree model used by Khandani et al.

1.3 Outline

Section 2 describes the most commonly used consumer credit models, such as discriminant analysis and logistical regression, This section also provides a background of machine learning in consumer credit analysis, namely decision trees and suport vector machines. Section 3 provides a theoretical background for support vector machines and explains the training algorithm. Section 4 describes the data fields used and provides an overview of data trends. Section 5 explains the technical layout of the data and the details of the SVM model used in this study. Section 6 provides results with a comparison to the decision tree of Khandani et al. Section 7 provides an analysis of the SVM model. Section 8 discusses future work in the field of consumer credit analysis. Section 9 provides concluding remarks.

2 Consumer Credit Analysis Models

The process of determining consumer credit involves many decisions based on consumer characteristics. A computational framework provides an advantage over manual analysis in the task of generating a suitable evaluation of a consumer due to the scale of consumers' information and the complexity of the problem.

This section outlines models used in consumer credit analysis. Because different models are implemented by different organizations, it is challenging to form a direct comparison between different methods when the data and testing methods are not the same across organizations. The most common traditional methods currently used in practice are discriminant analysis and logistic regression (logit) [1]. Several machine learning methods have been proposed for this task to replace traditional methods. The machine learning methods discussed in this thesis are decision trees and support vector machines.

2.1 Traditional

For traditional statistical models, a typical lending institution uses information from within its own company and information from other companies to determine a score based on past behavior. While these models predict credit worthiness reasonably well, these models are can be slow to change, especially when market conditions vary drastically in relatively short periods of time [12]. Often, these models make use of quantitative information regarding consumers, as well as other information pertaining to regulations and actions of competitors. This information is measured at discrete intervals in the past and does not reflect the most up-to-date information. Some of the consumer information used in these models does not vary quickly enough with respect to actual outcomes.

The actual models are usually unique to particular companies or organizations, so the specific information considered in various models is not explicitly known. There is room for improvement in these models, as even a small percentage of cost savings can amount to several hundreds of millions of dollars [12]. The most commonly used models in practice today are discriminant analysis and the logistical model, also known as logit [5]. These models

make assumptions about the characteristics of data distributions, which limit their ability to analyze generalized data sets.

2.1.1 Discriminant Analysis

The goal of discriminant analysis is to find a linear function of accounting and marketing variables that best separates the good and bad groups of borrowers. Each borrower is classified into one of those two groups. When considering the variables, the between-group variance is maximized while the within-group variance is minimized. With linear discriminant analysis, it is assumed that the variables can be expressed linearly. However, this does not reflect the reality of the relationships between variables in most cases, so discriminant analysis is limited in its ability to optimally classify borrowers.

Linear discriminant analysis attempts to find a subspace lower in dimension than the original data sample's dimension such that the data points are separable. This is achieved by finding a hyperplane in order to maximize the means between the two classes and minimize variances. Suppose that the sample means are defined for classes A and B as follows [18]:

$$\overline{x}_A = \frac{1}{N_A}\sum_{x\in A} x; \overline{x}_B = \frac{1}{N_B}\sum_{x\in B} x$$

where N_i represents the number of samples in class i. Then, the scatter matrices, which define sample variability in each class, are defined as follows [18]:

$$S_A = \sum_{x\in A}(x-\overline{x}_A)(x-\overline{x}_A)^T; S_B = \sum_{x\in B}(x-\overline{x}_B)(x-\overline{x}_B)^T$$

Then, the goal is to find the optimal hyperplane vector ϕ which minimizes the data sample variance. This can be expressed as follows [18]:

$$\min_{\phi}(\phi^T S_A\phi + \phi^T S_B\phi) = \min_{\phi}\phi^T(S_A+S_B)\phi = \min_{\phi}\phi^T S\phi$$

The scatter matrix between the two classes is as follows [18]:

$$S_{AB} = (\overline{x}_A - \overline{x}_B)(\overline{x}_A - \overline{x}_B)^T$$

To solve the optimization problem, it can be shown as the maximization of Fisher's criterion [18]:

$$\max_{\phi} J(\phi) = \max_{\phi} \frac{\phi^T S_{AB} \phi}{\phi^T S \phi}$$

There are infinitely many solutions, so the denominator is set to the following equality constraint:

$$\phi^T S \phi = 1$$

Thus the Lagrangian becomes:

$$L_{LDA}(x, \lambda) = \phi^T S_{AB} \phi - \lambda(\phi^T S \phi - 1)$$

Then, the optimal hyperplane is the eigenvector corresponding to the smallest eigenvalue of the generalized eigensystem:

$$S_{AB} \phi = \lambda S \phi$$

To classify a new sample z, the following formula is used [18]:

$$\underset{n}{\text{argmin}}\{d(z\phi, \overline{x}_n \phi)\}$$

The discriminant analysis function can be estimated using a maximum likelihood estimate or the discriminant function. The discriminant function in the latter case can be expressed as follows:

$$Z_i = \sum_j d_j x_{ij}$$

There is a common misconception that the variables used must be multivariate normally distributed. When the variables are linearly distributed, this condition is not necessary unless significance tests are performed [7]. Discriminant analysis assumes that the data can

be split linearly [1].

2.1.2 Logistical Model

The logistical model outputs the probability of default. A dummy variable y_i is defined as follows [11]:

$$y_i = \begin{cases} 1 & \text{if default;} \\ 0 & \text{if normal.} \end{cases}$$

The logistical function can be written as follows [8]:

$$\pi(x) = \frac{\exp(\beta_0 + \beta_1 x)}{1 + \exp(\beta_0 + \beta_1 x)}$$

If the data is linearly separable, logistical models perform about the same as linear discriminant analysis. When this happens, the logistical function is essentially a straight line [7].

The logit transformation of $\pi(x)$ is defined as follows [8]:

$$\begin{aligned} g(x) &= \ln\left[\frac{\pi(x)}{1-\pi(x)}\right] \\ &= \beta_0 + \beta_1 x \end{aligned}$$

The transformation $g(x)$ has linear parameters, may be continuous, and has an infinite range.

Because $\pi(x)$ is not linearly related to the coefficients in the logistic function, the maximum likelihood estimate is used to estimate the parameters in the logistical function. The maximum likelihood estimate is used to maximize the probability of the observed data. This technique yields consistent estimates, and a unique maximum always exists.

To solve the maximum likelihood function, the likelihood function must be defined. The maximum likelihood estimators of these parameters are the values that maximize this func-

tion [8]. The contribution of the likelihood function for a given pair is as follows:

$$\pi(x_i)^{y_i}[1-\pi(x_i)]^{1-y_i}$$

Let $\boldsymbol{\beta} = (\beta_0, \beta_1)$ represent the parameters to the logistic function. If the observed data set is (x_i, y_i) and the observations are assumed to be independent, then the likelihood, or probability, of the observed data is [8]:

$$l(\boldsymbol{\beta}) = \prod_{i=l}^{n} \pi(x_i)^{y_i}[1-\pi(x_i)]^{1-y_i}$$

In practice, it is easier to use the log likelihood form to solve for the parameters. The log likelihood is [8]:

$$L(\boldsymbol{\beta}) = \ln[l(\boldsymbol{\beta})] = \sum_{i=1}^{n} \{y_i \ln[\pi(x_i)] + (1-y_i)\ln[1-\pi(x_i)]\}$$

The value of $\boldsymbol{\beta}$ is found by differentiating $L(\boldsymbol{\beta})$ with respect to β_0 and β_1. The following likelihood equations are obtained [8]:

$$\sum_{i=1}^{n} [y_i - \pi(x_i)] = 0$$

and

$$\sum_{i=1}^{n} x_i[y_i - \pi(x_i)] = 0.$$

The equations above are solved by a computer using an iterative least squares procedure [8].

The result is a function returning a value between 0 and 1, indicating the probability of the borrower being bad [1]. When the data is not linear, the logistical model may be able to adapt better than discriminant analysis. Because the output is a probability, a threshold must be set for separating good and bad borrowers [11]. A Type I error is when a bad creditor is marked as good (false positive), and a Type II error is when a good creditor is marked as bad (false negative). Type I errors are more costly than type II errors, as they result in

losses caused by consumers. The costs of each type of error can be chosen arbitrarily, and one can choose the relative proportion of the costs of the types of errors. The threshold is determined by minimizing the cost function using different model parameters, reducing the overall values of type I and type II errors [11]:

$$\text{expected cost} = \pi_{fal} C_{\text{Type I}} \text{Type I} + \pi_{lop} C_{\text{Type II}} \text{Type II},$$

where π_{fal}, π_{lop} are actual percentages of good and bad debtors;
C_i = costs of Type I and Type II errors;
Type I, Type II = number of misclassifications of the respective type.

Wiginton [17] compares the logit model to discriminant analysis and found that logit performs the same or better than discriminant analysis but that its performance is still not good enough to make fully automated decisions with regard to consumer credit. The logit model uses the cumulative probability function in the general form:

$$P_i = \frac{1}{1 + \exp(-\sum_j \beta_j x_{ij})}$$

This model can be expressed as follows to apply the maximum likelihood estimate:

$$P_i = F(\sum_j \beta_j x_{ij})$$

where $F(\cdot)$ is the cumulative logistic function.

2.2 Machine Learning Methods

Machine learning has been used in recent times in an attempt to outperform traditional consumer credit analysis techniques. The most significant advantage is a machine learning model's ability to incorporate several variables in non-linear manners that cannot be captured by traditional models. Some machine learning methods that have been used include decision trees, support vector machines, and neural networks. While numerous models based on machine learning methods have been proposed and tested on small data sets, very few are

used in practice today. The machine learning techniques described are decision trees and support vector machines.

2.2.1 Decision Trees

A decision tree can be used to perform consumer credit analysis. This was performed by Khandani et al. [12] in the form of classification and regression trees. The parameters of the decision tree are found by minimizing the least-squared error of features and parameters. These are pruned by the Gini measure to prevent the tree from becoming too large. Khandani et al. [12] found that decision trees can outperform traditional measures of credit scoring, predicting credit events 3-12 months in advance. These results indicate that a decision tree can adapt effectively to changing market conditions and that other machine learning techniques are likely to outperform standard statistical methods as well.

In classification and regression trees (CART) [3], the output is derived from a recursive calculation of independent binary calculations. In the case that the dependent variables are constant, this is known as a collection tree. When there is a linear relation to the independent variables, this is a regression tree. In such a model, the parameters (ie. bounds for regions on the X or Y axis) are found in a way that minimizes the actual distance of the dependent variable and the resulting model. The distance measured is usually mean-squared error.

CARTs have the advantage over older numerical models in that non-linear relations between input variables can be expressed more effectively. This allows more independent variables to be used in the model.

Because CART models can use high-dimensional feature spaces for the training data set, the tree must be pruned to prevent overfitting. To prune, the Gini measure is used:

$$G(\tau) \equiv \sum_{k=1}^{K} P_\tau(k)(1 - P_\tau(k))$$

where τ is a leaf node, and $P_\tau(k)$ is the proportion of training data assigned to class k.

For a CART model T, the pruning criterion is:

$$C(T) \equiv \sum_{\tau=1}^{|T|} G(\tau) + \lambda|T|$$

where $|T|$ is the number of leaf nodes, and λ is a regularization parameter. The tree is will not be expanded when $C(T)$ reaches its minimum. The tree is continually expanded, then it is pruned by taking a decision node and turning it into a leaf node.

In order to improve the performance of a CART model, a technique called boosting was used by Khandani et al. [12]. This allows different weights to be applied to each input on the training set. Each weight is modified to improve the fit of the model. For a given observation i on the n^{th} iteration, the weight is:

$$w_i^{(n)} = w_i^{(n-1)} \exp[\alpha_{n-1} I(f_{n-1}(x_i) \neq y_i)]$$

where α_{n-1} is defined as:

$$\alpha_{n-1} \equiv \ln(\frac{1 - \epsilon_{n-1}}{\epsilon_{n-1}})$$

where $I(\cdot)$ is an indicator expressing if the the model was correct in predicting y_i given an input vector x_i, and ϵ_{n-1} is the weighted average error of the model from the $(n-1)^{th}$ iteration.

2.2.2 Support Vector Machines

There have been a few implementations of support vector machines for the problem of consumer credit analysis. One previous variant is the fuzzy model, which is meant to be more general than a standard support vector machine. This model, proposed by Wang et al. [16], is comparable to traditional models when it was used with an RBF kernel.

A model proposed by Lai et al. [13] used a least squares support vector machine, which uses a linear programming problem instead of a quadratic problem. Lai et al. compared their support vector machine to linear discriminant analysis, logit analysis, artificial neural networks, and a standard support vector machine with a quadratic programming problem.

With an RBF kernel, the least squares model performed the best on a particular set of data in their tests.

Another model proposed by Huang et al. [10], explores combinations of support vector machines and genetic algorithms and found that a combination of the two algorithms works well. Genetic algorithms are types of machine learning algorithms that were inspired by nature and evolution [6].

A key difference between the support vector machine and other linear methods is that the complexity of the SVM does not scale with the size of the feature space using an RBF kernel. [5]. Support vector machines can offer similar or improved performance over other machine learning methods in classification problems, depending on the type and distribution of the data. Section 3 explains support vector machines in greater detail.

3 Support Vector Machine Theory

3.1 Overview

The SVM is a supervised machine learning algorithm which has the capability of performing function estimation. In a supervised learning method, the training data contains an input mapping to an output.

The data used can be projected into a high dimension feature space to allow the SVM to treat the data in a linear manner. The feature space can be mapped from observable quantities, which may not have a linear relationship, to a linear manner as follows [4]:

$$x = (x_1, \ldots, x_n) \rightarrow \phi(x) = (\phi_1(x), \ldots, \phi_n(x))$$

In order to perform this mapping, features from the inputs are chosen to be included in the map function. Ideally, a minimal number of features is chosen in such a way to concisely capture the original attributes.

An SVM generates a hyperplane to maximize the distance between the data which should be separated. SVMs can allow for some error in classification, and these models are known as *soft margin* SVMs. When some of the data can be misclassified in this manner, the error is minimized when finding an appropriate separator. The margin can be expressed in the following form [4]:

$$\gamma_i = y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b)$$

The function used to separate the data is called the kernel. A kernel can be expressed in the following way such that for all $\mathbf{x}, \mathbf{z} \in X$,

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle$$

where ϕ maps X to a feature space. With kernels, knowing about the underlying feature map is not necessary. A kernel allows data to be mapped into a feature space and allow linear classification within the feature space, which allows computations to be performed efficiently.

In some cases, a linear kernel is used, especially where data is linearly separable. For more complex data sets, non-linear kernels may provide advantages for the task of separating the data in feature space. Some of these include: polynomial kernels, sigmoids, and Gaussian radial basis functions (RBF). A linear kernel can also be viewed as a polynomial kernel of degree 1. Using the kernel trick, a kernel can be incorporated into the SVM estimator to allow handling infinitely many features by using Lagrange multipliers, while remaining computationally feasible. A linearly separable dataset is shown in Figure 1, and a dataset separable by an RBF is shown in Figure 2.

The formulas for the most common kernels [9] are defined below:
Polynomial: $K(\mathbf{x}, \mathbf{x}') = (\gamma \mathbf{x} \cdot \mathbf{x}' + r)^d$
RBF: $K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma ||\mathbf{x} - \mathbf{x}'||^2)$

The SVM is an optimization problem [4]. To formulate the optimization, the general form

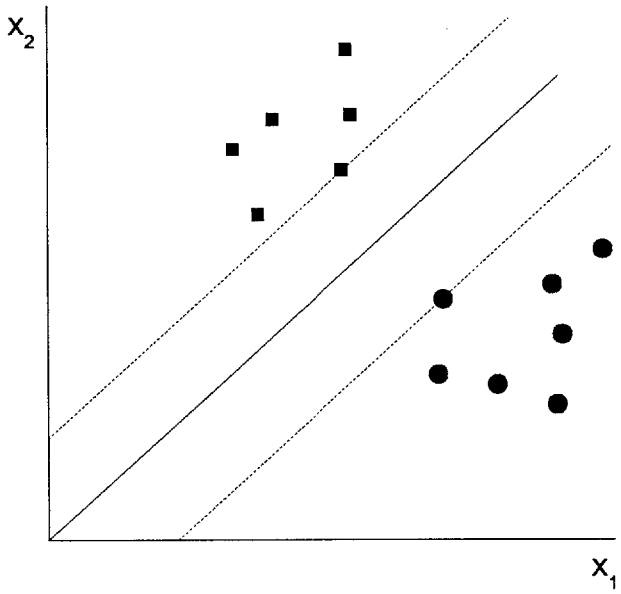


Figure 1: Linearly separable SVM data example. The solid line is the threshold used to separate one classification group from the other. The dashed lines represent the width of the separation between two groups. Data points on these lines are known as support vectors.

for the solution, where the hyperplane is $(\mathbf{w}, b)$ is:

$$\begin{aligned} &minimize_{\mathbf{w},b} && \langle \mathbf{w} \cdot \mathbf{w} \rangle \\ &subject\ to && y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b) \geq 1, \\ & && i = 1, \ldots, l \end{aligned}$$

In the soft margin case, it is possible to violate the margins by introducing a slack variable ξ [4]. The optimization problem becomes:

$$\begin{aligned} &minimize_{\mathbf{w},b} && \langle \mathbf{w} \cdot \mathbf{w} \rangle \\ &subject\ to && y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b) \geq 1 - \xi_i, \\ & && i = 1, \ldots, l \end{aligned}$$

With the introduction of slack variables, soft-margin SVMs can be used to allow some misclassifications of data and allows for better generalization of data. In practice, these are often used instead of the standard, hard-margin SVM.

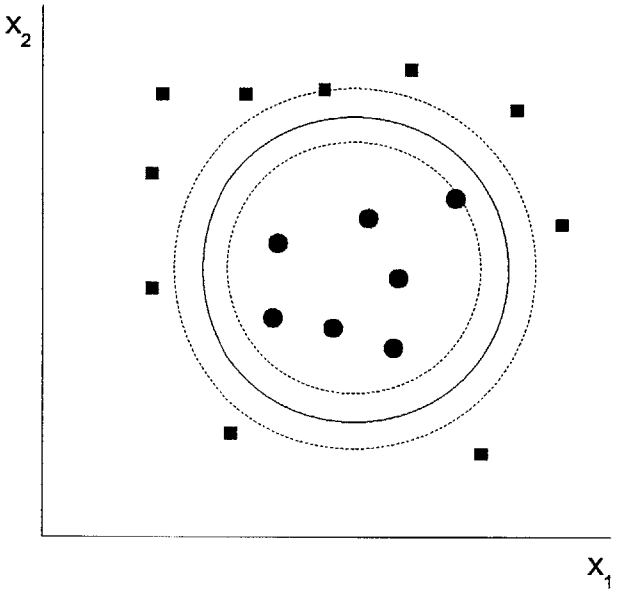


Figure 2: SVM data separable by an RBF. The solid line is the threshold used to separate one classification group from the other. The dashed lines represent the width of the separation between two groups. Data points on these lines are known as support vectors.

The maximal margin classifier is the a simple model of support vector machine classification used when the data in feature space is linearly separable [4]. While this does not apply in many real-world situations, it serves as a starting point for understanding more complex classifiers. The optimization problem is convex; that is, it minimizes a quadratic function with linear inequality constraints. With a functional margin of 1, the following is obtained:

$$\langle \mathbf{w} \cdot \mathbf{x}_+ \rangle + b = +1,$$
$$\langle \mathbf{w} \cdot \mathbf{x}_- \rangle + b = -1$$

The geometric margin is [4]:

$$\gamma = \frac{1}{||\mathbf{w}||_2}$$

where $\mathbf{w}$ is the weight vector.

The optimization problem is best solved in the dual form [4]. Normally, the Lagrangian in

the primal form is as follows:

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle - \sum_{i=1}^{l} \alpha_i [y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b) - 1]$$

To change this to the dual form by substitution, the following is obtained [4]:

$$L(\mathbf{w}, b, \alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x_i} \cdot \mathbf{x_j} \rangle$$

The dual form allows the kernel trick to be used, and the SVM is usually easier to solve in this way.

3.2 Sequential Minimal Optimization for Training

One way to train a support vector machine efficiently is to use the sequential minimal optimization technique (SMO) [15]. Normally, an SVM requires using a large quadratic programming optimization problem. SMO makes this process more efficient by breaking down the large quadratic programming problem into a series of small quadratic programming problems. Instead of running in quadratic time, it is between linear and quadratic in the size of the training set.

The standard quadratic programming problem for training and SVM is below:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j k(\vec{x}_i, \vec{x}_j) \alpha_i \alpha_j$$

$$0 \leq \alpha_i \leq C, \forall i,$$

$$\sum_{i=1}^{l} y_i \alpha_i = 0.$$

The SMO algorithm can solve the above equation if and only if the Karush-Kuhn-Tucker (KKT) conditions are met and $Q_{ij} = y_i y_j k(\vec{x_i}, \vec{x_j})$ is positive semi-definite. The KKT con-

ditions are below:

$$\alpha_i = 0 \Rightarrow y_i f(\vec{x_i}) \geq 1,$$
$$0 < \alpha_i < C \Rightarrow y_i f(\vec{x_i}) = 1,$$
$$\alpha_i = C \Rightarrow y_i f(\vec{x_i}) \leq 1$$

SMO solves the smallest possible optimization problem at each step. These problems consist of two Lagrange multipliers which obey a linear equality constraint. The inner loop of the algorithm uses a small amount of code compared to an iterative routine involving a quadratic programming solver. Unlike the standard quadratic programming problem, which requires a matrix equal to the square of the number of training examples, large matrix storage is not necessary for SMO.

SMO provides the best speedup when the data is sparse and a linear kernel is used. This speedup is a factor of up to 1200 [15]. Even with a non-linear kernel, the speedup is a factor of 15. Chunking, the other most common algorithm for training an SVM, requires a lot of computational time to handle the quadratic programming problem. There are some cases when chunking outperforms SMO, but the overall efficiency of SMO makes it a valuable choice for training SVMs in most cases. The SVM used in this study uses an implementation of the SMO algorithm to train the classifiers.

4 Data

This section will explain the types of data that were used in this study and provide a technical description of the data used in this study. The data came from a large bank, which will be referred to as "The Bank" throughout this document. The data contains information about the Bank's customers, and three main types of consumer-level data were included in this study: transaction-level, credit bureau, and account balances. The range of the dates is from January 2005 to April 2009. The range used in this study is from February 2008 to April 2009. Table 1 shows the training and evaluation periods.

Training Period		evaluation Period	
Start	End	Start	End
Feb 08	Apr 08	May 08	Jul 08
Mar 08	May 08	Jun 08	Aug 08
Apr 08	Jun 08	Jul 08	Sep 08
May 08	Jul 08	Aug 08	Oct 08
Jun 08	Aug 08	Sep 08	Nov 08
Jul 08	Sep 08	Oct 08	Dec 08
Aug 08	Oct 08	Nov 08	Jan 09
Sep 08	Nov 08	Dec 08	Feb 09
Oct 08	Dec 08	Jan 09	Mar 09
Nov 08	Jan 09	Feb 09	Apr 09

Table 1: Training and evaluation windows for the support vector machine algorithm. Each training window is three months, and each evaluation window comprises the three months immediately following the training months. Each data window contains both delinquent and non-delinquent customers. In this study, delinquent is defined as 90 days or more of incomplete payment. The training and evaluation windows may contain the same customers.

4.1 Consumer-Level Data

The three types of consumer-level data are explained in the following subsections.

4.1.1 Transactions

Account-level transactional data was provided, indicating the amount, direction (inflow or outflow), channel, and category.

In this context, the channel indicates the method of the transaction. These are the possible channels: Automated Clearing House (ACH), Adjustments to Account (ADJ), Automated Teller Machine (ATM), Online Bill Payment (BPY), Credit Card (CC), Check (CHK), Debit Card (DC), Account Fees (FEE), Interest on Account (INT), Wire Transfer (WIR), Internal Transfer (XFR) and Miscellaneous Deposit or Withdrawal (not in another channel) (MSC). For a very small portion of the data, the channel was unidentifiable (BAD).

The category is a broad classification intended to explain what type of transaction occurred. There were 138 different categories. Some examples include restaurants, bars, and groceries. Depending on the type of channel involved in the transaction, the accuracy of the category

varied. Credit card and debit card channels proved to be more accurate than ATM and check channels in terms of assigning a category. From these categories, the types of transactions can be determined at a high level. However, this data is not completely representative of a consumer for three main reasons. First, certain channels such as ATM and check can be used for a wide range of purposes. Second, cash-only transactions are not included. Third, consumers might use multiple banks, while the dataset only represents transactions from the Bank. Despite these limitations, the data provides a relatively good picture of trends observed over time. These data items are explained in Table 2.

4.1.2 Credit Bureaus

Data from one credit bureau used by the Bank is included at the individual level. This credit score, which is called CScore, is generally an accurate measure of determining default. Information about credit or loan facilities, known as the "Trade Line", is also included. If a consumer has a checking account with the Bank and a mortgage with some other lender, the information for the consumer would include mortgage statistics in a way without identifying the actual lender itself. The six types of trade lines are as follows: Auto (AUT), Mortgage (MTG), Credit Card (CCA), Home Line of Credit (HLC), Home Loan (HLN), Other Loans (ILA), Other Lines of Credit (LOC), and Speciality Type (SPC). Credit bureau data items are shown in Table 3.

4.1.3 Account Balance Data

Data regarding transactions and credit bureau scores are matched to checking account balances and a customer's CDs within the Bank. These data items are shown in Table 4.

Transaction Data

Transaction count	**By Category (continued)**
Total inflow	
Total outflow	Hotel expenses
	Travel expenses
By Channel	Recreation
	Department stores expenses
ACH (Count, Inflow and Outflow)	Retail stores expenses
ATM (Count, Inflow and Outflow)	Clothing expenses
BPY (Count, Inflow and Outflow)	Discount store expenses
CC (Count, Inflow and Outflow)	Big box store expenses
DC (Count, Inflow and Outflow)	Education expenses
INT (Count, Inflow and Outflow)	Total food expenses
WIR (Count, Inflow and Outflow)	Grocery expenses
	Restaurant expenses
By Category	Bar expenses
	Fast food expenses
Employment inflow	Total Rest/Bars/Fast-Food
Mortgage payment	Healthcare related expenses
Credit car payment	Fitness expenses
Auto loan payment	Health insurance
Student loan payment	Gas stations expenses
All other types of loan payment	Vehicle expenses
Other line of credit payments	Car and other insurance
Brokerage net flow	Drug stores expenses
Dividends net flow	Government
Utilities payments	Treasury
TV	Pension inflow
Phone	Collection agencies
Internet	Unemployment inflow

Table 2: Inflows and outflows into accounts within the Bank. Not all fields were used due to legal restrictions.

Individual Level	Account Level
Credit score File age Bankruptcy (date & code) MSA & Zip Code	Age of account Open/Closed flag & Date of closure Type (CC, MTG, AUT, etc) Balance Limit if applicable Payment status 48-Month payment status history

Table 3: Data items provided by credit bureaus. Not all fields were used due to legal restrictions.

Deposit Data	
Checking account balance Brokerage account balance Saving account balance	CD account balance IRA account balance

Table 4: Data items relating to balances for accounts within the Bank. Not all fields were used due to legal restrictions.

4.2 Data Security

Because the data used in this study is sensitive, a few steps were taken to ensure confidentiality. The Bank first removed any personally identifying information, such as names, addresses, social security numbers, and ages. Then, the data was transferred to a remote computer where all calculations were performed. As a result of the Fair Credit Reporting Act, the following fields of information were not included in the calculations: healthcare, insurance, unemployment, government, treasury, account and file age information.

4.3 Data Trends

From January 2005 to December 2008, several trends can be observed [12]. One of the most significant trends is that the amount of credit card debt increased among the top 5% of debtors. Towards January 2005, the highest debt was around $20,190, and by December 2008, this increased to $27,612. Another notable trend is that the median credit card debt increased from $647 to $1,259. These increases might indicate that the overall amount of debt is increasing; however, it is not indicative of whether consumers have more spending power, which could increase these numbers while overall risk stays the same. Khandani et al. [12] found that the debt-to-income ratio increased from approximately 11 to 13.5, which indicates that risk increases.

Another explanation of increasing credit card debt is that credit cards were used more than other payment forms. Khandani et al. [12] found that the ratio of credit card payments to income increased until July 2007, but this trend did not hold after that point. Several other trends can be observed in the data at various levels, and most trends point to a higher debt-to-income ratio leading up to the financial crisis in 2008.

Overall, from May 2008 to April 2009, the rate of consumers becoming delinquent in three month intervals was between 2.0% and 2.6%. During the three month interval beginning in December 2008, the rate of delinquency increased from 2.1% to 2.3%. Over the next period, this rate increased to 2.6%. Table 5 shows these statistics are shown for the ten evaluation periods.

Start Period	End Period	Actual Delinquent Customers (%)
May 08	Jul 08	2.4
Jun 08	Aug 08	2.2
Jul 08	Sep 08	2.1
Aug 08	Oct 08	2.0
Sep 08	Nov 08	2.1
Oct 08	Dec 08	2.1
Nov 08	Jan 09	2.1
Dec 08	Feb 09	2.3
Jan 09	Mar 09	2.6
Feb 09	Apr 09	2.5

Table 5: Actual percent of customers going delinquent after 90 days.

5 Setup

This section describes the support vector machine framework employed in this study. The challenge here is to create a model in such a way that relying on past data will yield a model that maximizes the probability of a correct prediction. The output of the support vector machine is a binary value indicating whether the consumer will be delinquent (ie. behind on payments) after 90 days.

5.1 Technical data description

The initial data was provided in one large file, which included all variables pertaining to each consumer. Personally identifiable information, such as names, addresses, and social security numbers were excluded from the dataset. There is approximately 50GB of data total. This data was put into a database in order to facilitate efficient selection of data fields, such as all records within a certain date range or certain attributes about consumers.

From this file, database queries were run in a way that allowed specific features to be selected, such as date ranges or features. The database querying system used was *kdb+* provided by *Kx Systems*. *kdb+* is optimized for large datasets and uses its own data querying language known as *q*. The database is column-oriented as opposed to row-oriented, which allows effective feature selection. Suppose that one is attempting to select a particular field (for

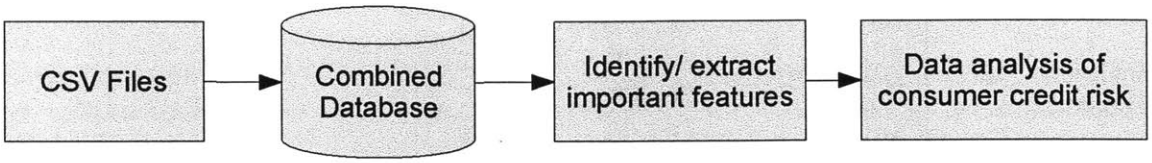


Figure 3: Overview for constructing the database for the machine learning model, as used by Khandani et al. [12].

example, all of the dates of the entries). The database stores all date entries in the same section of the file, as opposed to a field within a row entry. Thus, when entries are selected by date, the number of disk seeks is minimized, and the information is accessed faster than with a row store. Figure 3 provides a high-level overview of the steps involved in preparing a machine learning model with the provided data.

5.2 SVM Overview

Given a set of training data, the margin of separation is chosen in such a way to maximize the distance between the two possible groups. This process is known as maximal-margin separation. Classification is performed by determining which side the data point falls on. When an unknown data sample is included, the SVM classifies it as being good or bad, depending on which side of the margin it falls on.

The SVM algorithm was carried out using *Weka*, a Java-based library which supports a multitude of machine learning algorithms. The *Weka* API was called to use the support vector machine training algorithm. It supports multiple kernels, and initial testing was performed with a linear kernel. An RBF kernel was later used for experimental results.

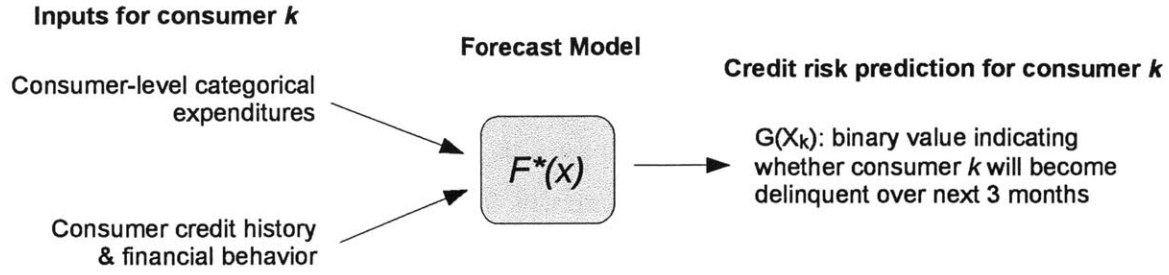


Figure 4: Overview of the SVM model setup as described by Khandani et al. [12]. Data regarding consumers is used as the input to the SVM, and the SVM outputs a binary value.

5.3 SVM Model Kernel

An SVM can be trained with several different kinds of kernels. The most common ones are linear, polynomial, and RBF (radial basis function). *Weka* provides an option to select the kernel for use in the SVM algorithm. Linear and RBF kernels were both tested in these experiments. Upon initial testing, the linear kernel performed well for some periods but was unacceptable for others. As a result, the only other choices were to use a polynomial kernel or an RBF kernel. Due to the popularity of the RBF kernel in SVM usage, its potential to adapt to most datasets, and success in prior SVM experiments involving consumer credit data, an RBF kernel was chosen for experimentation.

For the RBF kernel, two parameters can be selected: γ and C [2]. γ determines how curved the boundaries are; a larger γ value implies that the curvature in the decision boundary will increase. A γ that is too large results in overfitting, while one too small will result in a relatively straight decision boundary. C is the penalty for errors. A larger C results in a high penalty for errors and a small margin, while a small value leads to wide margins which tolerate errors. Both of these parameters must be selected properly to provide optimal results for a given dataset.

5.4 Evaluation

To test the support vector machine, a direct comparison was made to the CART models generated by Khandani et al [12]. This study uses a direct comparison to the training periods used in their study, using 3 month intervals for both training and evaluation. The first training period begins in February 2008 and ends in April 2008, and the last begins in November 2008 and ends in January 2009. The evaluation periods are the three months immediately following the training periods.

5.5 Model Inputs

Table 6 provides a list of the inputs used in the model used by Khandani et al [12] and those that will be used in this study. Of all the possible inputs, sensitive information is omitted to comply with legal requirements.

Model Inputs

Credit Bureau Data	**Transaction Data (continued)**
Total number of trade lines	Total expenses at discount stores
Number of open trade lines	Total expenses at big-box stores
Number of closed trade lines	Total recreation expenses
Number and balance of auto loans	Total clothing stores expenses
Number and balance of credit cards	Total department store expenses
Number and balance of home line of credits	Total other retail stores expenses
Number and balance of home loans	
Number and balance of all other loans	Total utilities expenses
Number and balance of all other lines of credit	Total cable TV & Internet expenses
Number and balance of all mortgages	Total telephone expenses
Balance of all auto loans to total debt	Total net flow from brokerage account
Balance of all credit cards to total debt	Total net flow from dividends and annuities
Balance of all home line of credit to total debt	
Balance of all home loans to total debt	Total gas station expenses
Balance of all other loans to total debt	Total vehicle related expenses
Balance of all other lines of credit to total debt	
Ratio of total mortgage balance to total debt	Total logging expenses
	Total travel expenses
Total credit-card balance to limits	
Total home line of credit balances to limits	Total credit-card payments
Total balance on all other lines of credit to limits	Total mortgage payments
	Total outflow to car and student loan payments
Transaction Data	
	Total education related expenses
Number of transactions	
Total inflow	**Deposit Data**
Total outflow	
Total pay inflow	Savings account balance
	Checking account balance
Total all food related expenses	CD account balance
Total grocery expenses	Brokerage account balance
Total restaurant expenses	
Total fast food expenses	
Total bar expenses	

Table 6: Inputs to the support vector machine model. Some of the fields available in the original dataset were removed from the input set due to legal restrictions.

5.6 Training Phase

The data provided was processed to include the training data fields using the *kdb+* database. Once the appropriate data was obtained, it was trained using the implementation of the SMO algorithm in *Weka* to train an SVM.

For the training phase, data was selected from the database as described above. Weka was used to train the data, using an RBF kernel as defined in its available options. The value of C was set to 1, and γ was set to 0.1. Ten intervals of three months each were selected from the database. These training windows are shown in Table 1. *Weka* was then used to generate a classifier for each three month period.

5.7 Evaluation Phase

Once all the models for the training intervals were obtained, the evaluation phase was carried out on the evaluation windows. The evaluation period for a corresponding training period was the 3 months following the evaluation period. Using *Weka*, the models trained for each month were selected from the database. *Weka* then predicted the outcomes based on the generated model, and these results were compared to the actual outcomes. In addition to evaluating the model's accuracy, other metrics were calculated. Section 6 details performance metrics and provides an evaluation of the effectiveness of the model.

6 Results

This section provides a framework for evaluating the performance of the SVM algorithm. The performance metrics used to measure the effectiveness of the model are explained. The experimental results are then presented for the ten training and evaluation periods.

6.1 Measures of Performance

The most straightforward way to evaluate the model is to examine the confusion matrix for a given test case. A confusion matrix contains four different metrics: *true positive*, *false positive*, *false negative*, and *true negative*. These can be expressed as percentages or in absolute numerical outcomes. Table 7 shows a visual representation of these definitions.

Using these metrics, the correct classification rate can be obtained by summing the true positives and true negatives. Other performance metrics that can be extracted from this confusion matrix are described in Table 8. Often, there is a tradeoff between precision and recall. High precision limits the number of false negatives, while high recall limits the number of false positives. The best model would have high precision and recall numbers. This implies that the true positive rate is high, and the false positive rate is low. While it might seem simpler on the surface to minimize the number of errors, an error classifying a bad creditor as good (false positive) is costlier than an error classifying a good creditor as bad (false negative). Thus, in this study, the false negative rate is not as costly as the false positive rate, though both of these values should be minimized in a model. These rates are highly dependent on the parameters used in a model.

The kappa statistic measures the amount of agreement between the actual and predicted outcomes. Landis and Koch [14] explain that a kappa statistic between 0.6 and 0.8 indicates substantial agreement, while values > 0.8 indicate almost perfect agreement between the

two sets of values. The kappa statistic is defined as follows:

$$\text{kappa statistic} \equiv \frac{(P_a - P_e)}{(1 - P_e)},$$
$$P_a \equiv \frac{TP + TN}{N},$$
$$P_e \equiv \frac{TP + FN}{N} \cdot \frac{TP + FN}{N}$$

The F-measure is a harmonic mean of the model's precision and recall, and a high value indicates a good tradeoff between precision and recall. This is defined as:

$$F\text{-Measure} \equiv \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}$$

		Predicted outcome: Good	Predicted outcome: Bad
Actual Outcome	Good	True Positive (TP)	False Negative (FN)
	Bad	False Positive (FP)	True Negative (TN)

Table 7: Visual representation of a confusion matrix.

Metric	Formula	Description
Precision	TN/(TN+FN)	Accuracy of the model when it classified an entry as bad
Recall	TN/(TN+FP)	Percent of bad entries the model detected
True Positive Rate	TP/(TP+FN)	Percent of good entries classified as good
False Positive Rate	FP/(FP+TN)	Percent of bad entries classified as good

Table 8: Formulas and descriptions of performance metrics extracted from the confusion matrix.

6.2 Outcomes

Table 9 and Table 10 show experimental results of performance metrics comparing the linear SVM to the CART model used by Khandani et al. for the ten evaluation periods outlined in Table 1. The prediction date is the last month of the training period, or equivalently the month before the evaluation period. For example, a prediction date of Apr-08 used training data from Feb-08 to Apr-08 and evaluation data from May-08 to Jul-08.

Table 11 shows the average statistics for both models. Figure 5 shows a graph of correct classification rates. Figure 6 shows a graph of the false positive rate. Figure 7 shows a graph of the root mean squared error values. Figure 8 plots the F-Measure values.

Model for Prediction Date	Apr-08		May-08		Jun-08	
Machine learning algorithm	CART	SVM	CART	SVM	CART	SVM
Correctly Classified Instance rate	0.989	0.997	0.991	0.996	0.991	0.994
Incorrectly Classified Instance rate	0.011	0.003	0.009	0.004	0.009	0.006
Kappa Statistic	0.751	0.900	0.753	0.897	0.735	0.852
Mean Absolute Error	0.006	0.003	0.004	0.004	0.004	0.006
Root Mean Squared Error	0.075	0.057	0.061	0.061	0.062	0.077
TP Rate	0.997	0.999	0.998	0.999	0.998	0.997
FP Rate	0.312	0.155	0.345	0.161	0.348	0.160
Precision	0.839	0.968	0.896	0.968	0.856	0.870
Recall	0.688	0.845	0.656	0.839	0.652	0.840
F-Measure	0.756	0.902	0.757	0.899	0.740	0.855
Model for Prediction Date	Jul-08		Aug-08		Sep-08	
Machine learning algorithm	CART	SVM	CART	SVM	CART	SVM
Correctly Classified Instance rate	0.991	0.996	0.991	0.985	0.991	0.985
Incorrectly Classified Instance rate	0.009	0.004	0.009	0.015	0.009	0.015
Kappa Statistic	0.751	0.892	0.751	0.694	0.749	0.695
Mean Absolute Error	0.004	0.004	0.004	0.015	0.004	0.015
Root Mean Squared Error	0.061	0.067	0.062	0.104	0.065	0.107
TP Rate	0.998	0.999	0.997	0.993	0.997	0.992
FP Rate	0.322	0.168	0.319	0.332	0.309	0.334
Precision	0.853	0.968	0.849	0.742	0.828	0.746
Recall	0.678	0.832	0.682	0.668	0.692	0.666
F-Measure	0.755	0.895	0.756	0.702	0.754	0.705

Table 9: Performance metric results for the linear SVM model for prediction dates ranging from Apr-08 to Sep-08.

Model for Prediction Date	Oct-08		Nov-08		Dec-08	
Machine learning algorithm	CART	SVM	CART	SVM	CART	SVM
Correctly Classified Instance rate	0.990	0.984	0.989	0.982	0.988	0.983
Incorrectly Classified Instance rate	0.010	0.016	0.011	0.018	0.012	0.017
Kappa Statistic	0.741	0.691	0.726	0.679	0.735	0.690
Mean Absolute Error	0.004	0.016	0.005	0.018	0.005	0.017
Root Mean Squared Error	0.065	0.112	0.067	0.115	0.069	0.114
TP Rate	0.997	0.992	0.997	0.991	0.997	0.993
FP Rate	0.320	0.336	0.356	0.352	0.346	0.346
Precision	0.827	0.745	0.848	0.734	0.854	0.751
Recall	0.680	0.664	0.644	0.648	0.654	0.654
F-Measure	0.746	0.700	0.732	0.688	0.741	0.699

Model for Prediction Date	Jan-09	
Machine learning algorithm	CART	SVM
Correctly Classified Instance rate	0.989	0.981
Incorrectly Classified Instance rate	0.011	0.019
Kappa Statistic	0.745	0.690
Mean Absolute Error	0.005	0.019
Root Mean Squared Error	0.070	0.118
TP Rate	0.997	0.992
FP Rate	0.320	0.342
Precision	0.839	0.743
Recall	0.680	0.658
F-Measure	0.751	0.698

Table 10: Performance metric results for the linear SVM model for prediction dates ranging from Oct-08 to Jan-09.

Average statistics

Machine learning algorithm	CART	SVM
Correctly Classified Instance rate	0.989	0.988
Incorrectly Classified Instance rate	0.011	0.012
Kappa Statistic	0.745	0.768
Mean Absolute Error	0.005	0.012
Root Mean Squared Error	0.070	0.093
TP Rate	0.997	0.995
FP Rate	0.320	0.269
Precision	0.839	0.823
Recall	0.680	0.731
F-Measure	0.751	0.774

Table 11: Average performance metric results for the CART and linear SVM models for all prediction dates.

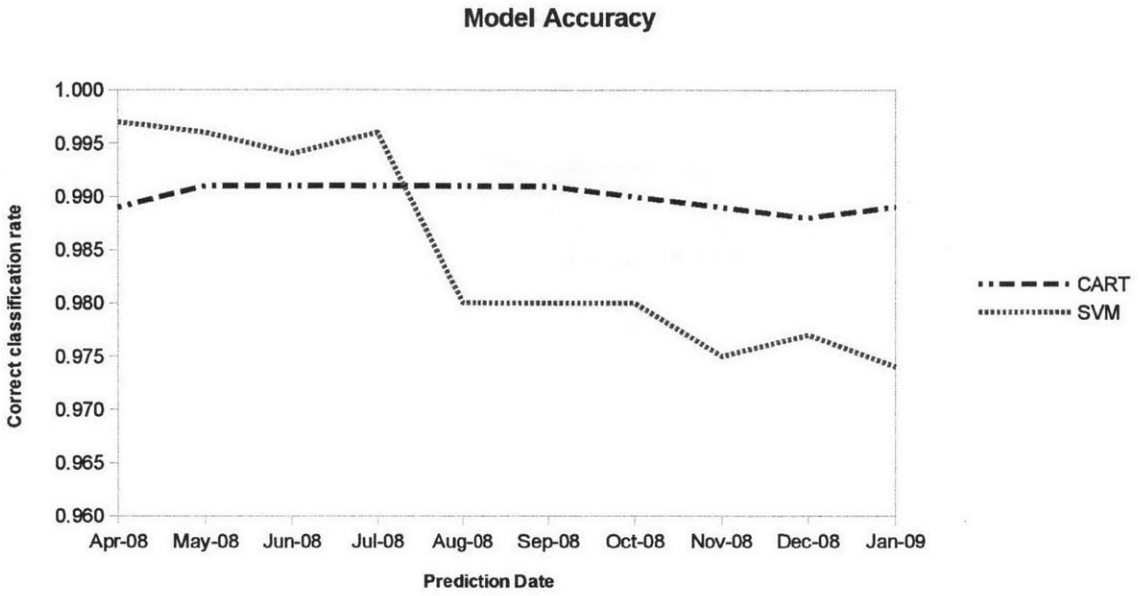


Figure 5: Plot of correctly classified instances for CART and SVM machine learning algorithms.

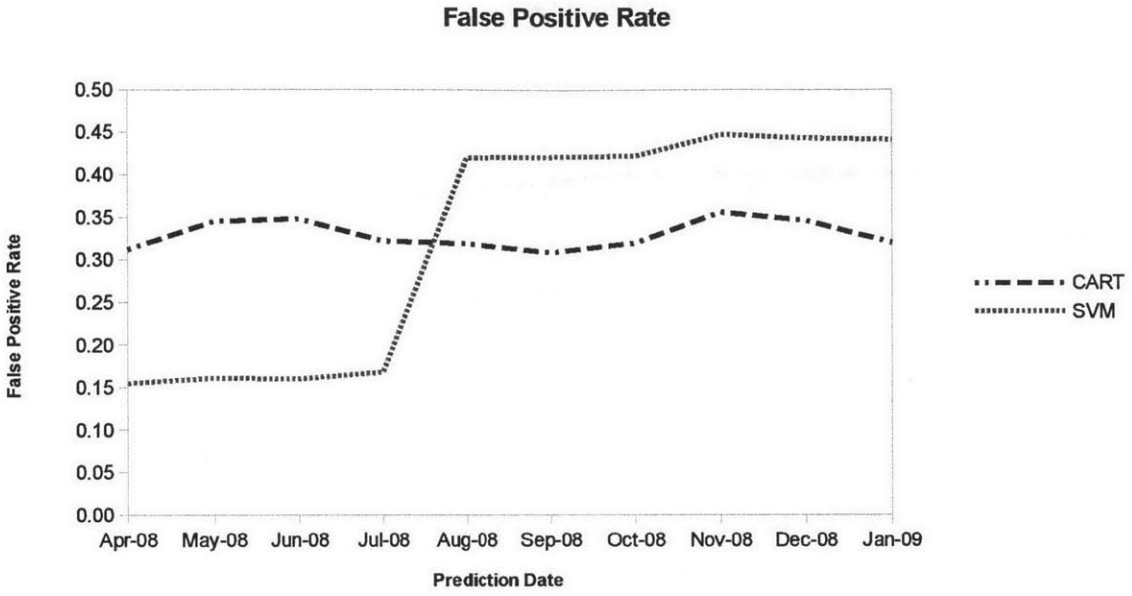


Figure 6: Plot of false positive rates for CART and SVM machine learning algorithms.

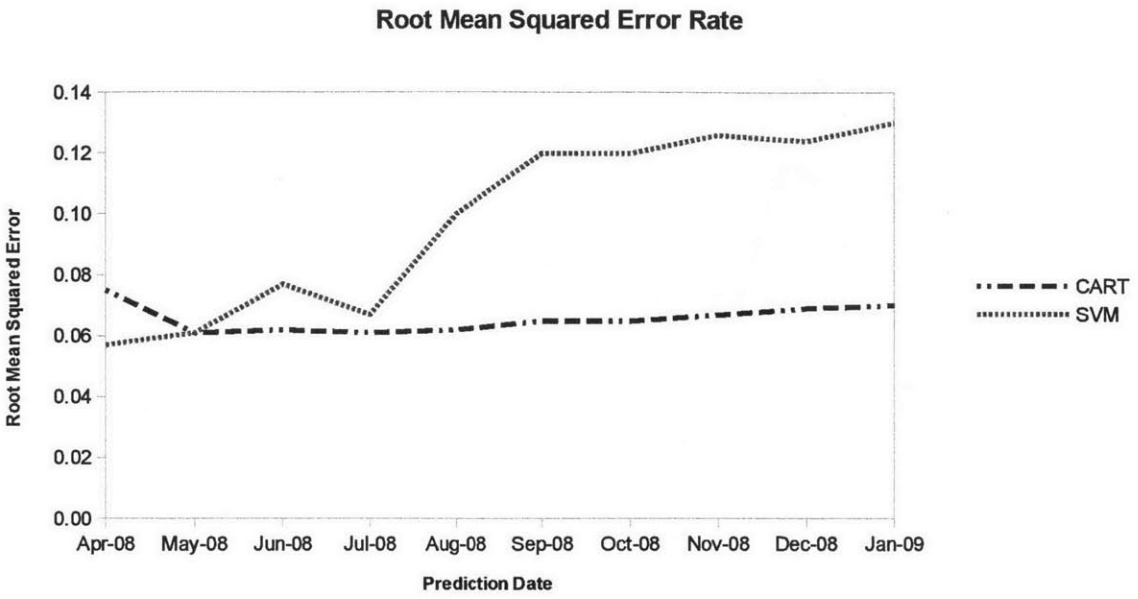


Figure 7: Plot of root mean squared error values for CART and SVM machine learning algorithms.

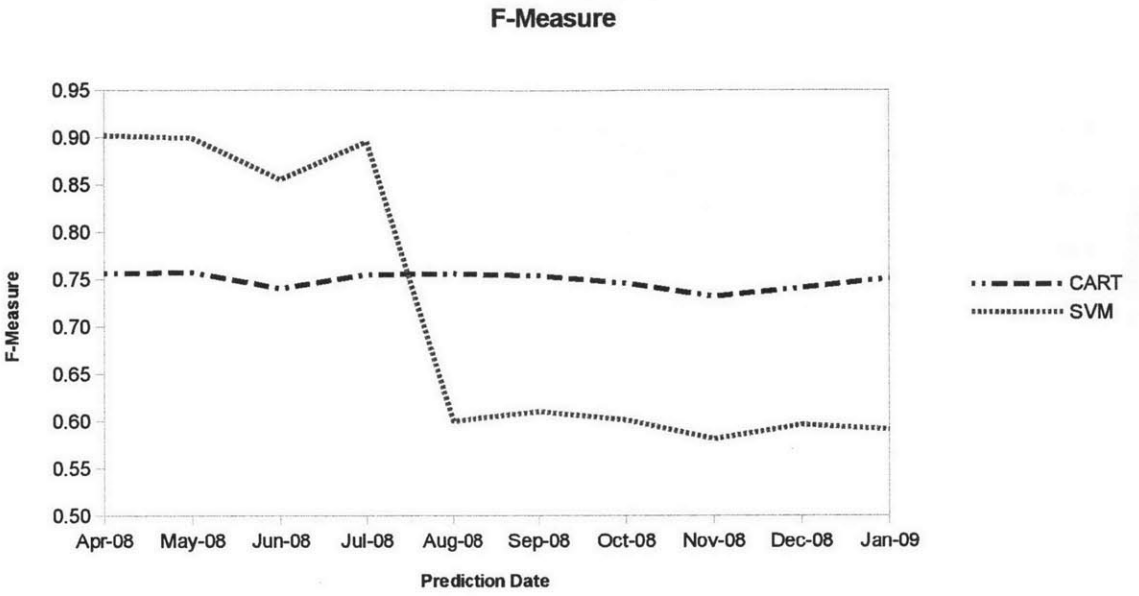


Figure 8: Plot of F-measure values for CART and SVM machine learning algorithms.

7 Analysis

These results show that the SVM with an RBF kernel and the CART model offer similar performance, but the CART model outperformed the SVM towards the end of the period. An interesting observation is that the mean absolute error rates and root mean squared error rates for the SVM model are higher than those for the CART model. A larger difference between the MAE and the RMSE implies that there is higher variance of errors within the model. This means that the model is more flexible and that the bias, or the difference between the expected values, is lower.

In order to tune the SVM model differently, the biggest change that can be made is to modify the kernel. While the RBF kernel generally works well, this all depends on its parameters, C and γ. The only way to determine the appropriate values is to test them on the dataset in use. There are no "perfect" acceptable values that would work for every possible dataset. The RBF parameters were chosen from the first couple training periods using 2-fold cross validation. However, this method was not the optimal solution for determining the best overall parameters for all further training periods.

7.1 Overfitting

One potential problem with any model is overfitting. This happens when the model too closely resembles the training data. In this case, the problem is that any data points outside of a narrowly defined model's boundaries will likely be incorrect. On the training data, an overfitted model would perfectly predict the data points, but it would be too specific in practice. While RBFs can adapt to different data distributions, they are still prone to overfitting. This can be prevented by setting the RBF parameters such that the model does not resemble the training data too closely yet provides enough separation power. Generally, a model that makes too many assumptions about future data (eg. a high-degree polynomial with a very specific fit) will fail to provide enough real-world use due to a large possible number of cases. To gain a sense of whether the model overfitted the data, cross-validation was used to evaluate it with missing information.

7.2 Cross-Validation

To measure the ability of the SVM model to predict consumer credit in the face of missing information (ie. feature vectors), 10-fold cross-validation was tested on the month of October 2008. The data is divided into 10 partitions. 9 partitions are used to train the classifier, and the last partition is used for evaluation. This process is repeated a total of 10 times, each using different partitions for evaluation and training. Overall, performance remained the same. This suggests that although the SVM had lower accuracy than the CART model for this particular month, it is able to generalize results well and is not significantly overfit. Thus, it is likely that the model will not suffer drastically under varying data conditions. With 10-fold cross-validaton, the overall accuracy of the cross-validation trials was 98.1%, compared to 98.4% for the support vector model.

7.3 Choice of Kernel

The RBF kernel used in this study showed potential to be close to the decision tree algorithm. It is a general kernel which can classify points well if its parameters are tuned correctly. To compare its performance to the linear kernel, the sample evaluation date of October 2008 was used. The confusion matrices were calculated for the linear kernel, which can be found in Table 12 and the RBF kernel, which can be found in Table 13. A comparison to the logit model has also been provided in Table 14. It outperforms the linear kernel in accuracy and provides a better false positive rate. Initial tests from months prior show that the linear kernel was very accurate, with performance roughly equivalent to the RBF kernel and both kernels outperforming the decision tree. However, this performance declined when the data was not as linearly separable. By comparison, the decision tree outperformed both of these in the same period.

For this particular evaluation case, market conditions were changing rapidly as the financial crisis was underway. The behavior of the linear kernel model indicates that the data patterns in this period would best be explained by a non-linear model. The RBF improves upon this by adapting to a non-linear data relationship effectively.

7.4 Comparison to Decision Trees

While both SVMs and decision trees perform similarly in this study, the real-world pros and cons of each classifier should be considered when choosing what to use on the data.

One advantage of the SVM is that it will find the optimal separation hyperplane for a given dataset and specified parameters. Another is that the kernel used has the potential to adapt to highly complex datasets. However, its major disadvantage is finding the best kernel for the type of data involved. While a linear or polynomial kernel is relatively easy to choose because it requires no parameters (other than the degree), such kernels are limited in their classification ability. Thus, the SVM would not be performing optimally for a given dataset. On the other hand, a complex kernel, such as an RBF, can adapt well to most datasets but requires more effort to find the best parameters.

The advantage of a decision tree is that the model itself is easily interpreted by examining each decision at every level. The SVM model cannot be examined in this manner; it is best viewed as a black-box model which cannot be easily interpreted. Thus, it is difficult to determine exactly which features in a model are the most significant. Another advantage of the decision tree over the SVM is that the result is a probability, as opposed to a binary classification in the SVM. Instead of training an entire SVM again to test new parameters, the decision tree's threshold can easily be changed to tune the model without retraining. The training time was also an advantage over the SVM. Although the SMO algorithm for the SVM provides some speedup compared to a standard quadratic programming solver, it still required a substantial amount of computational resources not required by a decision tree.

Based on these interpretations, an SVM with an RBF kernel has the potential to outperform other analysis techniques for this dataset assuming the optimal parameters are chosen. However, this process can be tricky, and computational resources required are relatively high. Decision trees can be trained more quickly and do not need extensive parameter tuning in the training phase. Given current computing power and the relatively similar performances of each classifier, the decision tree might make more sense for real-world applications.

		Predicted outcome: Good	Predicted outcome: Bad
Actual Outcome	Good	95.01%	2.20%
	Bad	1.89%	0.90%

Table 12: Confusion matrix for Oct-08 using a linear kernel.

		Predicted outcome: Good	Predicted outcome: Bad
Actual Outcome	Good	96.79%	0.77%
	Bad	0.92%	1.52%

Table 13: Confusion matrix for Oct-08 using an RBF kernel.

		Predicted outcome: Good	Predicted outcome: Bad
Actual Outcome	Good	94.88%	2.33%
	Bad	1.58%	1.22%

Table 14: Confusion matrix for Oct-08 using a logistical model.

8 Future Work

There is no perfect technique to estimate consumer credit scoring, and this thesis still leaves many questions about consumer credit scoring unanswered.

Other variations of the support vector machine could be used (ie. changing the kernel). While the linear kernel is a reasonable choice in many applications, the RBF kernel is the most commonly used kernel. For this particular data, the RBF kernel did not consistently outperform the decision tree. This could likely be improved by experimenting with the kernel further, allowing the optimal values of C and γ to be chosen for this dataset. An RBF kernel is likely to perform better than a linear kernel assuming the RBF parameters are set properly.

Other potential improvements in consumer credit analysis could result from completely different models or by changing the information used in modeling consumer credit. Some information about consumers or economic conditions may prove to be more helpful than others. To improve credit scores, the addition or deletion of certain features in models might provide additional relevant information, or might remove irrelevant information which a model might mistakenly use. Less noise in the model will allow a model to more clearly evaluate consumer credit data.

The observation window for training data may also have an influence on data outcomes. A short window will likely change quickly with new conditions and provide a small amount of information, but a long window is inclusive of more overall information and might not change quickly enough. A larger training and evaluation window period (eg. 6 or 12 months) has the potential to indicate a consumer's credit strength over the long run.

9 Conclusion

This study presented a support vector machine classifier with an RBF kernel for consumer credit data. Using a set of standard performance metrics related to classifying data, a comparison is drawn to previous work by Khandani et al. [12] involving decision trees,

another machine learning technique. This study shows that the support vector machine provides comparable performance to the decision tree.

Machine learning models for consumer credit analysis show great potential for real-world credit forecasting and cost savings. One point to note is that the particular outcomes of this study are not likely to work for all datasets. This depends on the available data for use in the study. Many other consumer credit evaluation models do not employ as much consumer-level data as used in this study.

Although the SVM model tends to be as good or better than other machine learning algorithms, it proved to be worse in some cases. This could be attributed to two main reasons. One, the decision tree returns a probability of default, while the SVM only returns a binary value. The probability allows for a threshold to be readily set, and this probability contains more information than a binary value. Two, machine learning algorithms can have different results, depending on the datasets involved. While the SVM is generally regarded as being more accurate than other machine learning algorithms, most machine learning algorithms offer similar performance on a variety of datasets. These findings suggest that the SVM with an RBF kernel might not be the best for consumer-credit analysis given this specific dataset, but this highly depends on the RBF kernel's parameters. The lengthy training time on this large dataset proved to be a large drawback. While it provided generally good results, the question of which combination of machine learning (or non-machine learning) algorithm and model parameters to use still remains unanswered.

10 References

[1] Edward I Altman and Anthony Saunders. Credit risk measurement: Developments over the last 20 years. *Journal of Banking & Finance*, 21(11):1721–1742, 1997.

[2] Asa Ben-Hur and Jason Weston. A users guide to support vector machines. In *Data mining techniques for the life sciences*, pages 223–239. Springer, 2010.

[3] Leo Breiman. *Classification and regression trees.* CRC press, 1993.

[4] Nello Cristianini and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods [electronic resource] / Nello Cristianini and John Shawe-Taylor.* Cambridge ; New York : Cambridge University Press, 2000 (Norwood, Mass. : Books24x7.com [generator]), 2000.

[5] Jonathan N Crook, David B Edelman, and Lyn C Thomas. Recent developments in consumer credit risk assessment. *European Journal of Operational Research*, 183(3):1447–1465, 2007.

[6] David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[7] David J Hand and William E Henley. Statistical classification methods in consumer credit scoring: a review. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 160(3):523–541, 1997.

[8] D Hosmer and Stanley Lemeshow. Applied logistic regression. new york, ny: A wiley-interscience publication, 2000.

[9] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification, 2003.

[10] Cheng-Lung Huang, Mu-Chen Chen, and Chieh-Jen Wang. Credit scoring with a data mining approach based on support vector machines. *Expert Systems with Applications*,

33(4):847–856, 2007.

[11] Philip Joos, Koen Vanhoof, Hubert Ooghe, and Nathalie Sierens. Credit classification: A comparison of logit models and decision trees. In *Proceedings of the Workshop on Application of Machine Learning and Data Mining in Finance, 10th European Conference on Machine Learning, Chemnitz, Germany*, pages 59–72, 1998.

[12] Amir E Khandani, Adlar J Kim, and Andrew W Lo. Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance*, 34(11):2767–2787, 2010.

[13] Kin Keung Lai, Lean Yu, Ligang Zhou, and Shouyang Wang. Credit risk evaluation with least square support vector machine. In *Rough Sets and Knowledge Technology*, pages 490–495. Springer, 2006.

[14] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

[15] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schoelkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998.

[16] Yongqiao Wang, Shouyang Wang, and Kin Keung Lai. A new fuzzy support vector machine to evaluate credit risk. *Fuzzy Systems, IEEE Transactions on*, 13(6):820–831, 2005.

[17] John C Wiginton. A note on the comparison of logit and discriminant models of consumer credit behavior. *Journal of Financial and Quantitative Analysis*, 15(03):757–770, 1980.

[18] Petros Xanthopoulos, P Pardalos, Panos M, and Theodore B Trafalis. *Robust data mining*. Springer, 2013.