Evaluation of Credit Risk

Marland Sitt, Tony Wu

Abstract

Credit ratings are of large interest to bond investors and debt issuers. Machine learning techniques have emerged as prominent ways for corporate credit ratings analysis by achieving better performance than traditional statistical ones. We report a SVM–based credit rating classifier with 70% classification accuracy when compared to standard credit ratings. Our technique uses a 5–fold cross validation to find optimal parameters for the linear kernel. The results indicate possible over or under rating of companies.

1 Introduction

Credit ratings are used by bond investors and debt issuers as a measure of riskiness of the companies and their bonds. They are important determinants of risk premiums and the marketability of bonds. Credit ratings in essence are a measure of bankruptcy risk. The prediction of credit ratings and thus default risk can have a significant impact on profitability. This was recently brought into light by the 2008 credit crunch. During this turbulent period, many companies declared bankruptcy. The traditional approach for credit risk prediction takes into account various quantitative as well subjective factors such as leverage, earnings, and reputation through a scoring system. Some banks, however, use ratings issued by standard credit rating agencies such as Moody's and Standard & Poor's. These ratings tend to be reactive rather than predictive. Therefore, there is a need to develop an accurate quantitative prediction model for corporate defaults. We demonstrate a support vector mechine based bankruptcy prediction model. The relationship between the characteristics of the company will be learned instead of explicitly modeling the underlying dynamics of the company.

We see a few immediate uses of such a classifier. The first would be to obtain credit ratings for a new company which may not have been rated by a credit rating agency. Another would be to obtain credit ratings for established companies which have recently undergone a large change, for which their current credit ratings may be out-of-date. Lastly, we can find fundamental misratings of companies. For example, Advanced Micro Devices (AMD) is currently regarded as a B company by Moody's. If our machinery predicts AMD as A or C, then this suggests the bonds issued by AMD are overpriced or underpriced, respectively.

2 Data Collection

Data was aggregated from a variety of sources. Credit rating data for each company was collected from Morningstar (`morningstar.com`). Fundamental data from income statements, balance sheets, and statements of cashflows for each company was collected from Gurufocus (`gurufocus.com`). Historical daily stock open, high, low, and closing prices as well as volume information were downloaded from Yahoo! Finance (`finance.yahoo.com`).

3 Data Preprocessing

Our collected data falls into one of the following categories: continuous, discrete categorical, or discrete ordinal.

Continuous data does not need any sort of preprocessing, and can be used directly.

Discrete categorical data needs to be converted into a format a standard machine learning algorithm can use. Assuming we have K classes, we map into K binary variables, with each element indicating whether that feature belongs to that class. For example, if we have a discrete variable taking values in $\{A, B, C, D\}$, this will get mapped into the 4 dimensional vector

$$\begin{bmatrix}\mathbf{1}_{\{\text{class } A\}} & \mathbf{1}_{\{\text{class } B\}} & \mathbf{1}_{\{\text{class } C\}} & \mathbf{1}_{\{\text{class } D\}}\end{bmatrix}$$

Lastly, we must handle ordinal data. For example, {no, low, high} gives an ordinal ranking. One possibility is to transform the ordinal scale into a numerical scale, and treat the data as continuous. Generally, we can't quantify the "distance" between two ordinal rankings. Thus, the numerical scale must be picked rather arbitrarily, which is quite undesirable. Another possibility is to treat the data as categorical data, but this is undesirable as it loses the ordering information. Because the SVM implementations we use do not handle ordinal data for regressor variables, we use one of the above approaches, depending on which seems more appropriate for the data.

Class	Credit Rating
1	{AAA, AA+}
2	{AA}
3	{AA-}
4	{A+}
5	{A}
6	{A-}
7	{BBB+}
8	{BBB}
9	{BBB-}
10	{BB+}
11	{BB}
12	{BB-}
13	{B+}
14	{B}
15	{B-}
16	{CCC, CC}

Table 1: Class Mapping

Since very few companies are given the best and worst ratings, we group the top two and bottom two ratings into one class. The mapping we use is given in Table 1.

4 Methods

As discussed in the data preprocessing section, ordinal rankings have some of the properties of both numerical and categorical values. Both classification and regression methods can be directly applied to ordinal data, but neither is particularly well suited for the task. We apply two methods to deal with the ordinal rankings. The first method makes use of an algorithm designed for ranking items. The second method uses binary SVM methods applied to the ordinal classification problem.

4.1 SVM-Rank

The SVM-Rank method uses a Cutting-Plane Algorithm for training linear SVMs [1]. This algorithm is not only designed for ordinal classification problems but can be trained in linear time ($O(sn\log(n))$) for linear kernels. The Cutting-Plane Algorithm, as outlined in [1], iteratively constructs a sufficient subset $\mathcal{W}$ of the set of constraints in the ordinal regression SVM:

$$\begin{aligned}\underset{\mathbf{w},\xi_{ij}\geq 0}{\text{minimize}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}}\xi_{ij} \\ \text{subject to} \quad & (\mathbf{w}^T\mathbf{x}^{(i)}) \geq (\mathbf{w}^T\mathbf{x}^{(j)}) + 1 - \xi_{ij}, \quad \forall\,(i,j)\in\mathcal{P}\end{aligned}$$

where $\mathcal{P} = \{(i,j) : y^{(i)} > y^{(j)}\}$

In each iteration, it first computes the optimum over the current working set $\mathcal{W}$. It then finds the most violated constraint and adds it to the working set $\mathcal{W}$. The algorithm then continues by optimizing over the new working set, unless the most violated constraint is not violated by more than the desired precision ϵ.

To pick the regularization parameter C, we perform a 10-fold cross-validation on the training set and pick the value that minimizes the misclassification rate. The misclassification rate is defined as the number of incorrectly classified companies over the total number of companies. A linear kernel is used in our experiments to utilize the time efficient training of the Cutting-Plane Algorithm.

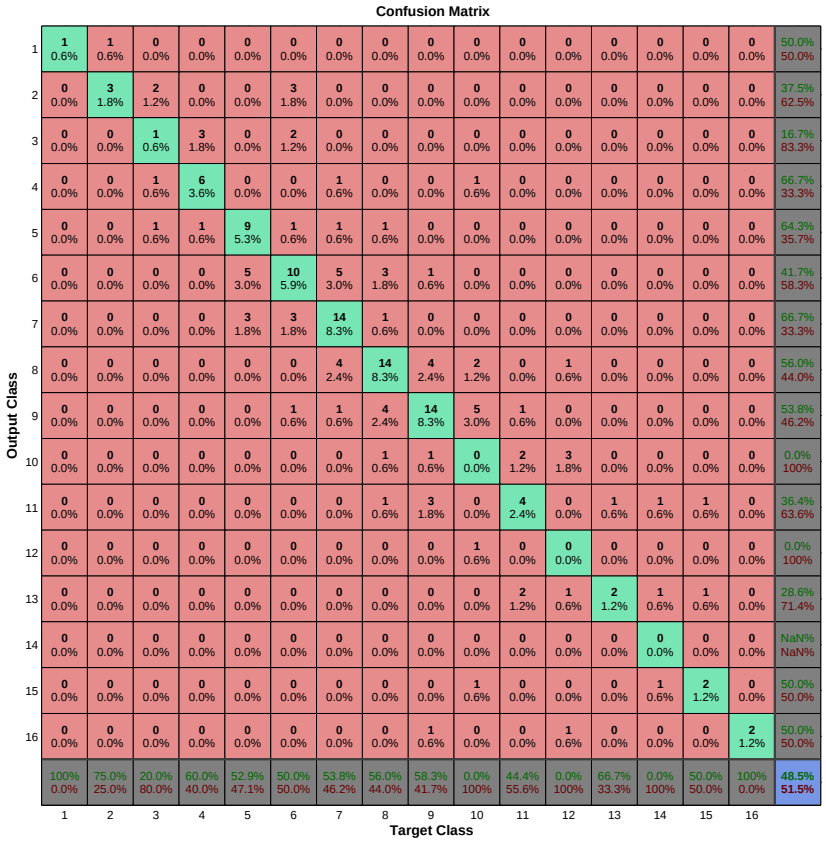

Figure 1: Confusion Matrix for SVM-Rank

4.2 Round Robin Learning

One simple idea about how to apply binary SVMs to the multi-class problem is called round-robin learning [2]. Assuming we have L classes numbered 1 to L, we train classifiers $\mathcal{M}_{ij}$ for $1 \leq i < j \leq L$. With this method we obtain $L(L-1)/2$ classifiers. In our dataset, we have $L = 16$, which results in 120 binary classifiers.

The classifier $\mathcal{M}_{ij}$ is meant to distinguish between class i and class j. Each SVM gives a "vote" for a particular class. After collecting all the "votes," the class with the majority of the votes is selected.

To pick the regularization parameter C, we perform 5–fold cross validation on the training set, and pick the value that minimizes MSE, where MSE is defined by letting the classes take the integer values 1 through L. Further, we use the linear kernel because of the computational complexity of other kernels.

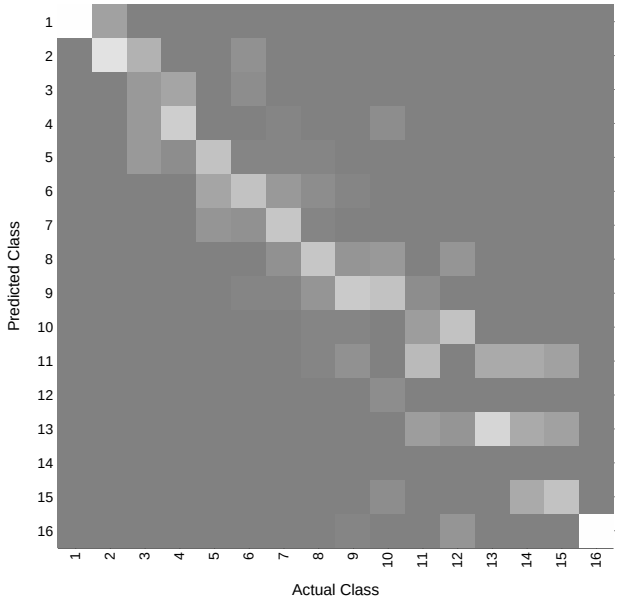

Figure 2: Confusion Matrix for SVM-Rank

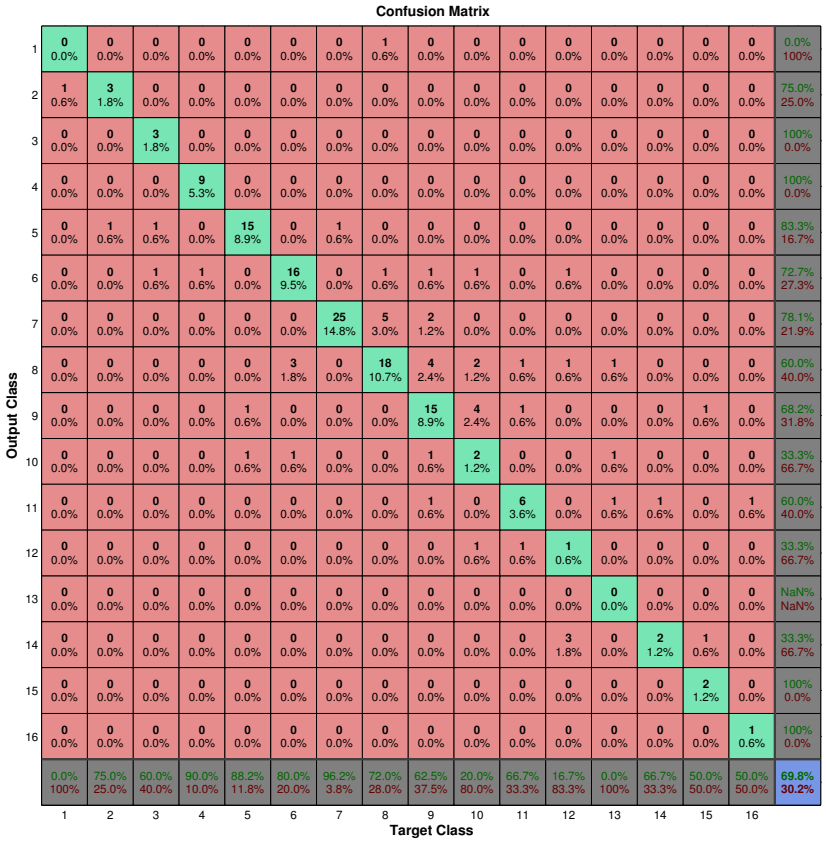

Figure 3: Confusion Matrix for Round Robin Learning

5 Results

In section 5.1 we present results when applying SVM-Rank to our ordinal classification problem. In section 5.2 we present results with round robin learning.

5.1 SVM-Rank

We achieve 48.5% classification accuracy with the test data using SVM-Rank. This is at 7.8× improvement over the baseline probability of correct classification of 6.25%. If we expand the tolerance for error to ±1 classes, the classification accuracy increases to 76%. A confusion matrix for the results is shown in Figure 1. As shown, we see most of the predicted classes fall within a class of distance 1 of the target class.

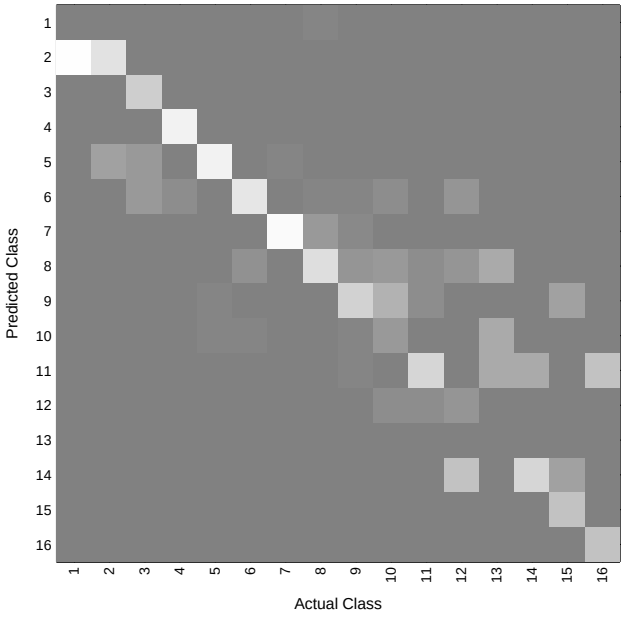

Figure 4: Confusion Matrix for Round Robin Learning

5.2 Round Robin Learning

Performing round robin learning, we can achieve 69.8% classification accuracy on the test set, an 11× improvement over the baseline probability. If we expand the tolerance for error to ±1 classes, the classification accuracy increases to 79%. See Figure 3 for the detailed confusion matrix.

6 Discussion

Figures 2 and 4 show color coded confusion matrices for both methods. Data points lying outside of the tolerance band around the diagonal indicate companies of possible concern. Companies located above or below the diagonal indicate a possible under or over rating of credit, respectively. Consequently, this could indicate that the bond prices are over or under valued.

7 Summary

We showed that classification of corporate credit ratings based on fundamental and technical financial data can be done using SVM-based methods. Ordinal regression SVMs can achieve up to 48.5% classification accuracy while Binary SVMs using round-robin learning with a linear kernel can achieve 69.8% accuracy. Our model provides insight to predicting credit ratings for a new company which may not have been rated by a credit rating agency. Moreover, a high error of classification could indicate a fundamental misrating of a company by the credit rating agency.

References

[1] T. Joachims, *Training Linear SVMs in Linear Time*, Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD), 2006.

[2] Furnkranz, J., Round Robin Classification. Journal of Machine Learning Research 2 (2002) 721-747.

[3] Cardoso, J., da Costa, J.P.: Learning to classify ordinal data: The data replication method. JMLR 8, 13931429 (2007).

[4] T. Joachims, Making large-Scale SVM Learning Practical. Advances in Kernel Methods – Support Vector Learning, B. Scholkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.

[5] O. Chapelle, Support Vector Machines in the primal. 2006. 10 Nov. 2012. `http://olivier.chapelle.cc/primal/`.

[6] R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9(2008), 1871-1874.

[7] Chang, Chih-Chung & Lin, Chih-Jen (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2, 27:1-27:27.