Predicting Sovereign Default

Andrew Huang [1] **Taresh Sethi** [1]

Abstract

With the financial crises ongoing in Greece and Venezuela, sovereign debt crises have become more and more prominent in the public eye. Thus, it has become important to be able to predict when nations will enter such debt crises. We collected publicly available data in order to train models to predict, given a nation's economic status in one year, whether they would be in a debt crisis the next year. We were able to predict sovereign debt crises with good accuracy, even with the elimination of a feature we initially thought was too strongly correlated with the output, with our random forest model performing the best.

1. Introduction

A sovereign default is the failure or refusal of a sovereign state to pay back its debt in full. These events have massive consequences on a global scale. Not too long ago, Greece was at the center of international spotlight for sovereign default risk, marking the first developed nation to default from a loan from the IMF with debt exceeding 130 percent of GDP. Sovereign defaults are less predictable than normal loans, since sovereign states cannot be obliged to pay back their debt, and thus present an interesting problem. With the preponderance of economic data that has been made publicly available, we propose using machine learning to predict the probability of a sovereign state defaulting on a loan. Few studies have been performed for sovereign defaults (most of the time consumer default is studied), and even fewer have been conducted recently. With recent events like Brexit and the potential Grexit contributing to much economic uncertainty in Europe and around the world, we feel that being able to predict financial crises is more important than ever, hence our motivation for the project.

[*]Equal contribution [1]Stanford University, Stanford, California. Correspondence to: Andrew Huang <ahuang98@stanford.edu>, Taresh Sethi <tksethi@stanford.edu>.

Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017. Copyright 2017 by the author(s).

At its core, predicting sovereign default risk is a binary classification problem: predicting whether or not the country is in default. Credit rating agencies, like Standard & Poors (S&P) and Moodys, currently try to do this using sovereign credit ratings. However, in addition to these ratings generally not being available to the public, recent research has indicated that their ratings of corporations may not be as accurate.

Our input to the algorithm is the economic data for a given country in a given year (e.g., GDP and Population for Algeria in 1980.) We then use one of our models to output a binary prediction on whether the country will be in a sovereign debt crisis the following year.

2. Related Work

Within the field of sovereign debt crises, researchers Reinhart and Rogoff are well-noted economic researchers, famous for writing a book *This Time is Different* on the subject (2011). While they had economic expertise, they conducted their research shortly after the economic recession in 2009, and thus did not use recent developments in machine learning in their models. However, they did an exceptional job collecting data and determining in what years countries were in sovereign debt crises.

In a separate paper, Reinhart noted that 84% of sovereign debt crises were preceded by currency crises, thus making features that would predict a currency crisis valid predictors of a debt crisis (2002).

In another paper, Hilscher and Wilson used logistic regression, and their own metric they called a failure score to predict corporate default (2017). In this, they showed that they were able to outperform S&P Credit Ratings. They were able to predict default with 77% accuracy using a logistic regression model. Rojas-Suarez also noted that S&P credit ratings performed subpar with regards to sovereign default (2002).

In 2003, Manasse, Roubini, and Schimmelpfennig attempted to predict sovereign debt crises (2003). They used logistic regression and decision trees, and were able to predict sovereign defaults with around 90% accuracy with decision trees, using specially chosen features and proprietary data. We wanted to try and exceed or replicate their results.

3. Data and Features

3.1. Data Collection

We collected data from the International Monetary Fund (IMF) and from Reinhart and Rogoff's website, specifically using the IMF's data as our features, and then Reinhard and Rogoff's data to label our examples. We originally had 1,334 samples, using economic data spanning 43 countries and 30 years, where each sample was the economic data for a given country in a given year.

Our data was unbalanced, with 21.67% positive rates. To deal with this, we used different metrics and balanced our class weights, as detailed in the Methods section.

3.2. Preprocessing

We preprocessed our data in several ways. First, we eliminated sparse columns, or features for which there was little data available. We also eliminated sparse rows, or years where countries had little data available. To fill in the few missing gaps, we use the mean of the available values of the feature. Furthermore, throughout the process we standardized features to have mean 0 and unit variance. After preprocessing, we had 1,200 samples, which we then split 70:30 into training and test sets, giving us a training set size of 840 and a test set size of 360.

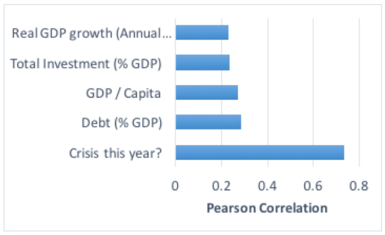
3.3. Feature Selection

In order to get a better sense of the variance and covariance in our data, we calculated the Pearson Correlation of each feature with the output, defined as:

$$\rho_{X,Y} = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y}$$

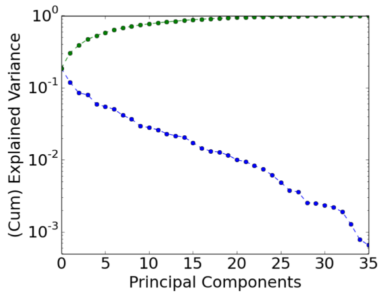
where σ_X, σ_Y are the standard deviations of X and Y, respectively, and $\text{cov}(X,Y)$ is the covariance between the two variables.

The five highest scoring features shown in the graph below.



The feature most correlated with whether a country would be in crisis the next year was whether the country was in crisis in the current year, which makes sense. However, because of this high correlation, we were concerned that this one feature might drown out the rest. We were also concerned that since many of our features were highly correlated (e.g. many features were derived from a country's GDP), we might have unnecessary noise in the data. Therefore, we performed principal components analysis (PCA) to get a better idea of the covariance between our features.

After performing PCA (shown below), we found that 30 components explained 99% of the variance in our data. Looking at the graph, we noticed that the blue curve did not drop down as suddenly as we expected to, implying that the "currently in crisis" feature was not as dominant of a feature as we initially thought. Additionally, since it took 30 principal components to account for 99% of the variance, our data was not as correlated as we thought.



Next, we performed recursive feature elimination (similar to backward search), ultimately selecting thirty features. However, during validation, the model trained with reduced features performed markedly worse than the model trained with all 49, so we ultimately opted with using all of our original features.

We also decided to try and achieve the best performance we could without the "crisis this year" feature, in order to see what economic factors influenced sovereign debt crisis the most.

4. Methods

Because of imbalance in our data, in addition to test accuracy, we also used F1 score as a metric, defined as:

$$F_1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}}$$

in order to get a better idea of how our models actually performed. This metric takes into account precision and recall and not just raw accuracy numbers, and thus is a superior way to evaluate performance on imbalanced data.

We trained four models on our training set: logistic regression, SVM, a neural network, and random forest models. Throughout the entire project, we used SKLearn to implement, train, and test our models (Pedregosa et al., 2011). For hyperparameter tuning and validation, we used k-fold cross-validation ($k = 10$).

As mentioned above, we trained and validated our data on training sets with and without the "crisis this year" feature, since we wanted to see how well we could perform without the feature.

4.1. Logistic Regression

As a baseline, we evaluated SKLearn's implementation of logistic regression with L2 regularization and balancing the classification weights to deal with the data imbalance. The algorithm learns a parameter θ for the sigmoid function $f(x) = \frac{1}{1+e^{-\theta^T x}}$, where $f(x)$ represents the probability that x is a member of the positive class. It learns this through maximum likelihood estimation, where θ is continually updated until it reaches a value that maximizes the likelihood function for the sigmoid function.

To balance the class weights, we set the class weight of each of the two classes to the inverse of the class' proportion within the training sample. We also used this to, in a way, assess the quality of our data; many of the papers we looked at used logistic regression, so it would be a relatively fair comparison.

4.2. SVM

We then chose to test an SVM model. SVM's, short for support vector machines, work as optimal margin classifiers; that is, they try to select a separating hyperplane that maximizes the minimum margin (distance from a data point to the hyperplane). More formally, it tries to solve the following optimization problem:

$$\begin{aligned} \min_{\gamma,w,b} \quad & \frac{1}{2}||w||^2 \\ \text{s.t.} \quad & y^{(i)}(w^T x^{(i)} + b) \geq 1, \quad i = 1, \ldots, m \end{aligned}$$

where γ is the minimum margin, w is the weight vector, and b is our intercept vector.

After trying numerous kernel functions and higher-dimension mapping through SKLearn and tuning these hyperparameters, the linear SVM performed best. We weight-balanced our SVM in the same fashion that we did for our logistic regression model.

4.3. Neural Network

We then chose to test a neural network model. Our primary motivation for selecting this model was that no one had tried this in the literature we read, and we were interested in applying more modern techniques. Ultimately, we chose SKLearn's fully-connected multi-layered perceptron (MLP) classifier with a sigmoid output activation function.

The neural network algorithm, as defined in class, works by creating weight and intercept matrices that represent connections between different layers of "neurons." Each neuron has an activation function, which is a nonlinear function that takes as input $Wx + b$, where W and b are the weight matrix and intercept vector, respectively, for a layer of neurons, and x is the output of the previous layer's activation function. This continues until the output layer, where the final activation function outputs the final prediction of the model. We train the model through backpropagation, where we update each weight and parameter in the model by taking derivatives with respect to the likelihood function of the output function, utilizing the chain rule.

For our neural network architecture, we had one hidden layer with 25 hidden ReLU units, with the ReLU activation function being defined as:

$$f(x) = \max(0, x)$$

We chose one hidden layer and 25 hidden units after reading that generally one layer is all that suffices for most problems, and that the number of hidden units should be in between the number of features and the size of the output layer. For the remaining hyperparameters, we used the default values that SKLearn provided.

4.4. Random Forest

Lastly, we evaluated the random forest model (Liaw et al., 2002). Our rationale behind this decision was that decision trees worked well in previous studies, so we wanted to try a tree-based model. Additionally, random forest tends not to be affected by unbalanced data as much as other models.

Random forests work by building a "forest" of decision trees. Decision trees tend to overfit their data, so by averaging multiple decision trees that were trained on different parts of the data, in an attempt to reduce the low bias and high variance of the individual trees. For each tree, a random subset of bootstrap aggregated, or "bagged", features is selected, and then the tree is fit on part of the training data. For classification problems, a prediction is made by taking a majority vote among the trees. The key part of this algorithm is the random subset of features; random forests are better at avoiding the problem of the decision trees being too similar, which is especially a problem for datasets like ours, where there is a single highly correlated feature.

Since it is generally advised for classification problems with p features to choose $\sqrt{p}$ as the max numbers of features per split, we set that parameter to 7. After using k-fold cross-validation to tune the hyperparameters, we ended up using the default SKLearn for the rest of the parameters.

5. Results and Discussion

The results are shown in the tables below. The first table displays the performance of our models with the ”Crisis this year” feature, and the second table displays the performance without. As a note, the testing set contained 78.3% negative samples (not in crisis). While our evaluation metric was primarily the F_1 metric as explained previously, we thought it would be interesting to display our training and test accuracy, especially to look out for overfitting/variance concerns.

Table 1. Classification Results with ”Crisis this year?” feature

Model	K-fold score	F_1 (train)	F_1 (test)	Training Acc.	Test Acc.
Logistic regr.	0.90	0.82	0.71	0.92	0.87
Linear SVM	0.92	0.82	0.74	0.93	0.89
Neural net.	0.92	0.84	0.71	0.94	0.88
Random Forest	0.93	1.00	0.76	1.00	0.91

Table 2. Classification Results without ”Crisis this year?” feature

Model	K-fold score	F_1 (train)	F_1 (test)	Training Acc.	Test Acc.
Logistic regr.	0.77	0.64	0.65	0.80	0.81
Linear SVM	0.85	0.63	0.62	0.87	0.86
Neural net.	0.87	0.76	0.63	0.91	0.85
Random Forest	0.91	1.00	0.71	1.00	0.91

5.1. Performance with ”Crisis this year?” Feature

We first analyze the performance of our models with and without the ”crisis this year” feature separately. Our logistic regression performed a little worse than previous studies, which we expected given that we did not use proprietary data. With the ”crisis this year” feature, Random Forest performed best out of all the models, achieving an F_1 score of 0.76 on the test set and achieving the highest K-fold validation score. Random Forest also outperformed the decision trees used by Manasse et al., without the proprietary data they used (2003).

At a first glance, we thought perhaps the Random Forest model may have been over-fitting greatly, especially as it achieved an $F1$ score of 1.00 on the training test and 100% accuracy on the training set. In comparison, the other models seemed to be much more generalized, with a less wide gap between the train and test score on the various metrics. However, as described in our methods, we have found that the structure of Random Forest allows it to fit the training set very well while still maintaining its generality to unobserved data.

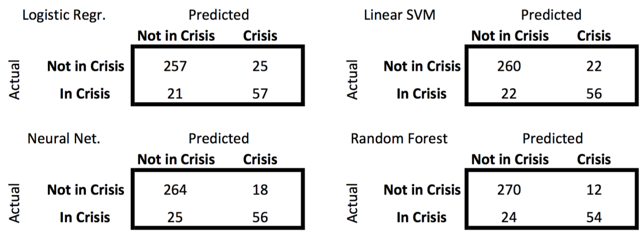
5.2. Performance without ”Crisis this year?” Feature

Without the ”crisis this year” feature, we found that that much of the analysis was similar. Random Forest performed best of all of the models, achieving an F_1 score of 0.71 and the highest K-fold validation score.

5.3. Confusion Matrices

To further evaluate the performances, as well as to delve into the F_1 metric results for each of the models, we generated a confusion matrix for each model (**Figures 1,2**). Again, we separate the models that trained with the ”Crisis this year” feature from the ones that trained without. The analysis is again very similar. For both scenarios, from the confusion matrices, we find that the biggest distinction between Random Forest and the other models is that Random Forest was able to achieve a much lower false positive rate than any of the other models. If we recall the formulation of the F_1 metric, it then makes sense that Random Forest achieved the highest F_1 score. Further, we now gain the insight that Random Forest predicts more efficiently if a country is going to go into crisis, and by doing so, it makes ”smarter” and more careful predictions of when a country will be transitioning into crisis.

Figure 1. Confusion Matrix with ”Crisis this year?” feature



Logistic Regr.

Actual \ Predicted	Not in Crisis	Crisis
Not in Crisis	257	25
In Crisis	21	57

Linear SVM

Actual \ Predicted	Not in Crisis	Crisis
Not in Crisis	260	22
In Crisis	22	56

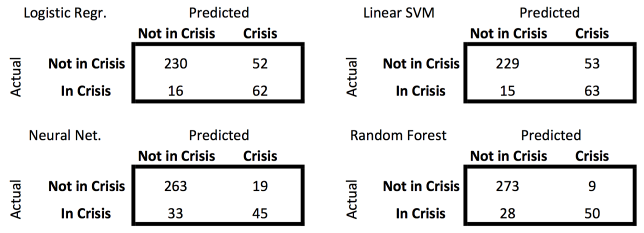
Neural Net.

Actual \ Predicted	Not in Crisis	Crisis
Not in Crisis	264	18
In Crisis	25	56

Random Forest

Actual \ Predicted	Not in Crisis	Crisis
Not in Crisis	270	12
In Crisis	24	54

Figure 2. Confusion Matrix without ”Crisis this year?” feature



Logistic Regr.

Actual \ Predicted	Not in Crisis	Crisis
Not in Crisis	230	52
In Crisis	16	62

Linear SVM

Actual \ Predicted	Not in Crisis	Crisis
Not in Crisis	229	53
In Crisis	15	63

Neural Net.

Actual \ Predicted	Not in Crisis	Crisis
Not in Crisis	263	19
In Crisis	33	45

Random Forest

Actual \ Predicted	Not in Crisis	Crisis
Not in Crisis	273	9
In Crisis	28	50

At this point, we note that strictly for the models trained without the ”Crisis this year” feature, the false-negative rates are higher for Random Forest and the Neural Network than for Logistic Regression and the Linear SVM (For the models trained with the ”Crisis this year” feature, the false-negative rates are all very similar, so this analysis does not apply). Therefore, we need to determine what we care about more - false-positives or false-negatives. In our problem, we would rather get a false-positive than a false-negative. That is, we'd rather mark that a country will go into crisis and it ends up not doing so, than overlook a country in financial crisis and find that the year after the country went into crisis. Therefore, strictly for the case when the models are trained without the ”Crisis this year” feature, we cannot clearly determine if Random Forest is the best model.

5.4. Comparing results with and without ”Crisis this year?” feature

Finally, we compare the performances with and without the ”Crisis this year” feature. Our results are fairly surprising. Initially, we suspected that our model would heavily weight the ”Crisis this Year” feature in calculating whether a country would go into crisis the following year, since that feature is so correlated with the output. However, we found that the performance levels were very comparable. While the models trained with the ”Crisis this year” feature still seemed to do better than the models trained without the ”Crisis this year” feature, the gap was not that wide. We speculate that our choice to not feature-select and have as many features as possible allowed our models to down-weight the ”Crisis this year” feature and allow for smarter predictions.

5.5. Analyzing transitions

As a last discussion, we thought it would be interesting to see how our best model - Random Forest with the ”Crisis this year” feature - would perform solely on data that represented when a country transitioned into a crisis or when it transitioned out of a crisis, given its trained parameters on the whole training set. Therefore, we generated a new test set, and ran our model on the test set. The model did not perform well. In hindsight, we realize that this was a very different problem than what we trained for, and label it as a future step.

6. Future Steps

One of our strong motivations for the project was to develop a model that only had to rely on publicly accessible data in order to predict if a country were to default the following year. With that said, we think it would be very interesting to explore more diverse features, especially from proprietary sources and socio-political implications. We also would like to expand on the idea of predicting a transition, and perhaps work to tune our parameters so that the model can be used in anomaly-detection applications.

7. Contributions

Both members contributed equally in all aspects of the project.

References

Hilscher, Jens and Wilson, Mungo. Credit ratings and credit risk: Is one measure enough? *Management Science*, 63(10):3414–3437, 2017.

Liaw, Andy, Wiener, Matthew, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

Manasse, P., Roubini, N., and Schimmelpfennig, P. Predicting sovereign debt crises. *IMF Working Papers*, 2003.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Reinhart, C. Default, currency crises, and sovereign credit ratings. *the world bank economic review*, 16(2):151–170, 2002.

Reinhart, C. and Rogoff, K. *This Time is Different*. Princeton University Press, 2011.

Rojas-Suarez, L. Rating banks in emerging markets: What credit rating agencies should learn from financial indicators. pp. 177–201. Springer, 2002.