CIS 520 Spring 2019, Project Report

Predicting Loans Future Status: Classifying Loans with High Charge Off Risk v.s. High Late Fee Rewards for Lenders

Team Members:

Mingzhi Cao (PennKey: `mingcao`; Email: `mingcao@seas.upenn.edu`)
Yanxiang Ding (PennKey: `yanxding`; Email: `yanxding@seas.upenn.edu`)
Bangxi Xiao (PennKey: `bangxi`; Email: `bangxi@sas.upenn.edu`)

Assigned Project Mentor:

Haoxian Chen

Team Member Contributions:

Team Member	Contributions
Mingzhi Cao	Data Visualization, Features Engineering, Machine Learning models, Report
Yanxiang Ding	Features Engineering, Re-sampling, Machine Learning models, Report
Bangxi Xiao	Data Cleaning, Features Engineering, Machine Learning models, Report

Code Submission: The code for this project will be submitted via the following github repository: `https://github.com/Yanxding/Predicting-Loans-Future-Status-Classifying-Loans-with-High-Default-Risk-v.s.-High-Rewards-for-Lend.git`

1 Abstract

In this study, we build machine learning models to predict multi-class loan status on Lending Club peer-to-peer lending data. We have 4 loan status to predict: fully-paid, charged-off, short late payment (1-30 days), and long-late payment (31-120 days). We divide the work-flow into 3 phases. First, we perform data cleaning on feature variables. Secondly, we use both unsupervised and supervised learning such as PCA and random forest for feature engineering. In the last phase, we explore 4 different supervised learning models with various parameters and evaluated their performance. This report defines the problem, describes the methods, and summarizes results and analysis. We achieve 98% high accuracy in predicting loan status, and we show complex performance measure on each class.

2 Introduction

The peer-to-peer lending has grown rapidly in recent years and provides loans to serve many purposes, such as consumer credit, small business and student education. The motivation behind our project is to help lenders make better lending decisions to maximize returns while minimize risks. This project provides critical application for investors to predicting future loan status (fully-paid, late, charge-off etc.) using Lending Club's loan data from 2007 to 2018. We focus on classification of future loan status trained on Principal Components built from demographic, loan types, and credit history of the borrowers.

3 Related Work

The data was obtained from Kaggle and a few previous works were done by the contributors on Kaggle. Their efforts were mainly concentrated on binary classification problem, exploratory data analysis and data structure analysis. Joe Corliss [2] made prediction on loan charge-offs from initial listing data and proposed a logistic regression with SGD parameter training algorithm. Also, he performed some other models such as random forest classifier and k-nearest-neighbor. Nathan George [7] from Kaggle made contributions on basic exploratory data analysis on the data by examining the distribution of each class. Mathew [6] explored correlatiuons between features. Luke [4] demonstrated some corrupted / not-well-formed rows in the data and he found out that these rows seem to be some sort of summary statistic or addendum referring to previous rows, which contain only a single column, providing us straight help in data cleaning process.
Previous research focuses on loan status and credit risk model. Chen and Tsai[16] studied using hybrid machine learning models for credit risk classification problem. Khandani et.al.[3] focused on using machine learning for consumer credit models.

4 Problem Formulation

All previous related works have been focused on binary classification between "fully paid" and "charged off" using limited number of manually selected features, which failed to take advantage of full information and introduced human biases. In our project, we focus on multi-classes loan status prediction utilizing full information. There are 4 status classes in our models: fully-paid, late (15-30 days), late (31-120 days) and charged-off. There are two major challenges for our project. The first is the high dimensional data set, which originally contains over 200 features and there are potential collinearity or correlation among a large fraction of them. The highly imbalanced data imposes the second major challenge.
Our approach to tackle the challenges include two stages. The first stage focus on feature engineering. We used unsupervised learning methods to remove co-linearity and select representative features in a systematic way such that our model could take advantage of full information while avoid over-fitting and multi-collinearity. In the second stage, we explored supervised learning algorithms to develop predictive models for loan status classification based on selected features from stage one.
Due to the highly imbalanced nature of our data, accuracy fails to measure true performance of learning models. Therefore, we used **precision**, **recall** and **micro-F1** for each class in an one-vs.-rest approach as performance measure[13].

5 Data Pre-processing

Consisting of up to 170 features, the raw data contained large amount of missing values, no-meaning characters and undefined or repeated features. Thus, developing machine learning models, we pre-processed the data with RStudio. First we went through the features and deleted those that are obviously irrelevant such as "Member ID", "url" and "description". We also transformed categorical variables into binary dummy variables. The missing values were simply omitted. After cleaning, the number of observations dropped from 1.4 million to 0.5 million, which is still sufficient for model training and testing.

6 Data Set Exploration

The sample size for loan status classes are highly imbalanced, as shown in Figure 1. below. The available data samples for the grace period, late (15-30 days), late (31-120 days) classes are very small in comparison with fully paid and charge-off. Thus, we used SMOTE algorithm to over-sample these minor classes.

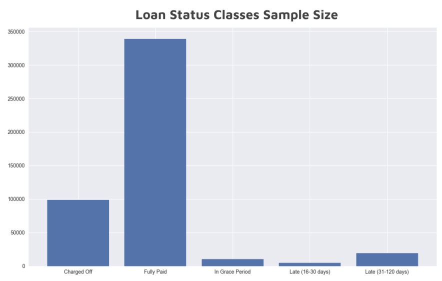

Figure 1. Sample size of 5 loan status classes.

Figure 2 shows the ratios of each class over the total sample size in each state in America. Mississippi has the third highest charge off ratio, and the highest late ratio for 15-30 days and 31-120 days. From background research, Mississippi is America's poorest state with median household income ($40,593) below the national average ($55,775), and has the largest population ratio below poverty line (22%). Arkansas has the highest charge-off rate. Online lending is a common practice for residents in Arkansas. Therefore, we recognize the importance of keeping states as categorical features.

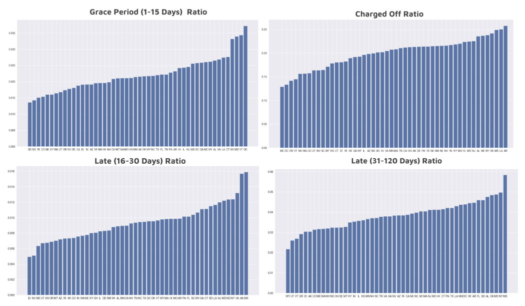

Figure 2. Class distribution over states in America.

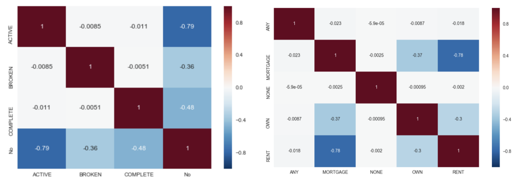

Figure 3. Left: correlation between hardship status; Right: correlation between loan purpose.

We explore interested features by visualizing their correlations and found that some features have high correlations between them. Thus, we need to remove multicollinearity and reduce confounding biases.

7 Algorithms

In stage 1 - feature engineering, **Principal component analysis (PCA)**, **Random forest** and **t-SNE** were used for dimensionality reduction, feature selection, and exploration and visualization of feature space. In stage 2 - predictive modeling, **logistic regression**, **multi-SVM**, **random forest** and **neural network** were explored for developing loan status classification model. All machine learning algorithms and visualization were implemented using Python.

7.1 Feature Engineering

7.1.1 Principal Component Analysis

PCA is a statistical procedure that uses singular value decomposition to transform a set of possibly correlated variables into mutual orthogonal variables called principal components. It is commonly used for orthogonalizing feature space and dimensionality reduction.[12] We first standardized our features to remove possible biases introduced by different feature scales. Then we applied PCA to real-value features. We used a threshold of 95% variance to select the number of principal components.

7.1.2 Random Forest Feature Selection

Random Forest is an ensemble learning method for classification that operates by constructing a multitude of decision trees and predicting new labels by "majority votes". Random Forest is also an effective method for estimating feature importance. Feature importance is computed and ranked by average weighted impurity decrease of each feature across trees in the forest.[9] The weighted impurity decrease in a tree is calculated by information gain (entropy decrease) as $IG(l, l_1, l_2) = H_l - (\frac{m_{l_1}}{m_l} H_{l_1} + \frac{m_{l_2}}{m_l} H_{l_2})$[10].

7.1.3 t-SNE

t-SNE (t-distributed stochastic neighbor embedding) is a nonlinear dimensionality reduction technique well-suited for embedding high-dimensional data for visualization in low-dimensional space. t-SNE constructs probability distribution in high dimension capturing similarities between instances. Then it maps each data point to a low dimensional distribution with minimized difference in conditional class probabilities between high and low dimensional distributions.[5]

7.2 Predictive Models

7.2.1 Multi-Class Logistic Regression

Multi-nomial logistic regression is the most basic classification algorithm. The algorithm minimizes a multinomial loss function $L = -\sum_{i=0}^{k} y_i ln(\widehat{p}_i)$, where k is the number of categories, p_i is the class probabilities and $\sum_i p_i = 1$. We adapted the Stochastic Average Gradient algorithm as the solver. L-1 regularization is commonly used to avoid over-fitting and assist in feature selection by promoting sparsity in coefficients. L2 regularization is another common penalty for reducing over-fitting.

7.2.2 Multi-SVM

Multi-SVM is a multi-class extension of SVM, in which a linear model is learned for each class producing a "score". The learning algorithm aims to maximize the margin between the score of the correct class and of any other class.[14] For non-linearly separable feature space, kernels are often used to implicitly map data to high dimensional space in which they are linearly separable. Most common kernels including polynomial and RBF kernals. RBF kernel implicitly computes dot products in infinite dimensional space using the function $K(x, x') = exp(\frac{-||x-x'||_2^2}{2\sigma^2})$. [11]

7.2.3 Neural Network

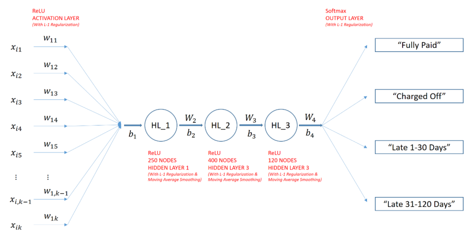
ReLU
ACTIVATION LAYER
(With L-1 Regularization)
x_{i1} x_{i2} x_{i3} x_{i4} x_{i5} $x_{i,k-1}$ x_{ik}
w_{11} w_{12} w_{13} w_{14} w_{15} $w_{1,k-1}$ w_{1k}
b_1
HL_1
W_2 b_2
HL_2
W_3 b_3
HL_3
W_4 b_4
ReLU
250 NODES
HIDDEN LAYER 1
(With L-1 Regularization & Moving Average Smoothing)
ReLU
400 NODES
HIDDEN LAYER 3
(With L-1 Regularization & Moving Average Smoothing)
ReLU
120 NODES
HIDDEN LAYER 3
(With L-1 Regularization & Moving Average Smoothing)
Softmax
OUTPUT LAYER
(With L-1 Regularization)
"Fully Paid"
"Charged Off"
"Late 1-30 Days"
"Late 31-120 Days"

Figure 4. Backpropagation neural network with three hidden layer, ReLU activation and Softmax output.

We build the backpropagation neural network with three hidden layer, ReLU activation function and Softmax output function. **ReLU**[8] function, first introduced by Hahnloser et al. in 2000, is widely used in deep learning and has the form of $f(x) = x^+ = max(0, x)$. ReLU is more computational efficient and has a better property for gradient propagation algorithm: fewer vanishing gradient problems compared to sigmoidal activation functions that saturate in both directions. **Softmax**[15] function is also commonly used in multi-class classification problems, which takes as input a vector of n real number and normalizes it into a probability distribution. Each output can be treated as a probability of being classified into the corresponding class. The softmax function is defined as $\sigma(z)_i = \frac{e^{\beta z_i}}{\sum_{j=1}^{K} e^{\beta z_j}}$.

8 Experimental Design & Results

Our total data set contains around 500,000 observations of Lending Club loan data from 2007 to 2018. 70% out of the total data were randomly selected as training set and the rest as test set. The random split followed class distribution of the total data such that training and test data sets have the same class distributions.

8.1 Dimensionality Reduction & Feature Selection

By PCA, as shown in Figure 5. below, 37 principal components were able to explain over 95% of variance of real-value features. PCA removes multi-collinearities in our feature space, which help to reduce confounding biases in our model. PCA alao allowed us to reduce from 64 original real-value features to 37 PCA-transformed features without loss of information.

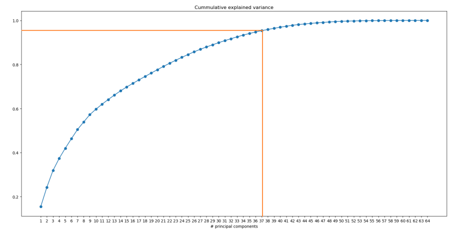

Figure 5. Explained variance vs. number of principal components (The orange lines mark 95% variance).

In feature engineering, random forest was used as a descriptive tool of the data instead of a predictive model for feature selection. By using the impurity decrease (information gain) measure, we ranked importance of both PCA-transformed real variables and categorical features, as shown in Figure 6. below. We decided not to use the last 6 features for our predictive model due to their marginal importance.

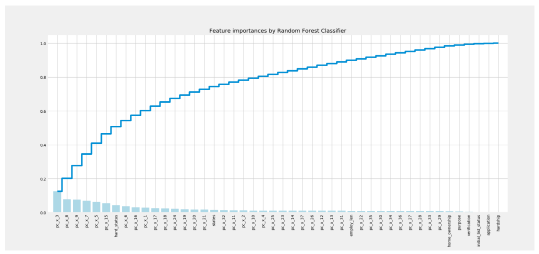

Figure 6. Feature importance by random forest (Bar plot: feature importance in descendent order; Blue line: cumulative feature importance).

By feature engineering, we finalized 37 PCA-transformed real-value features and 3 categorical features (expanded as 66 binary features) for predictive modeling.

8.2 Confusion in Feature Space

By using t-SNE, we visualized our 103 dimensional instance space in 3D space, as in Figure 7. below. Class 1 (charged-off) and 2 (fully paid) were well separated. However, class 3 (grace period), 4 (late 15-30 days) and 5 (late 31-120 days), especially class 3 and 4, were highly mixed in space.

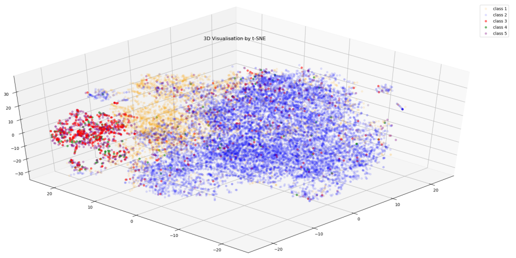

Figure 7. 3D visualization of instance space by t-SNE (parameter: perplexity = 30).

This explained why we were not able to separate class 3 and 4 in our first cycle of predictive modeling and the strong trade-off effect in performance between these classes. It also made intuitive sense such that borrowers in late periods might not have significant difference. Therefore, we decided to combine class 3 and 4 as a new "late 1-30 days" class in our second iteration of modeling.

8.3 Status Prediction

We applied 4 supervised learning models, multi-class logistic regression, multi-class SVM, random forest, and neural network for predictive modeling of loan status. We tuned hyper-parameters of each model by cross validation and compare results.

8.3.1 Multi-SVM

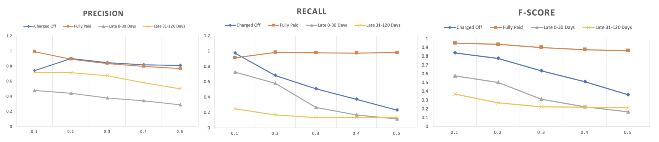

Figure 8. Performance vs. γ in SVM with RBF Kernel

We attempted SVM with polynomial kernel with degree from 1 to 5. However, they all have very poor performance, especially in minority classes (<30% F1 measure). SVM with RBF Kernel has much better performance. Figure X. shows the performance under different γ, and $\gamma = 0.1$ yields the best performance.

8.3.2 Logistic Regression

Regularization in logistic regression have only marginal benefit on performance. Accuracy on test data was improved from 0.97 in simple logistic regression to 0,98 in regularized logistic regression. However, there is no difference between $L1$ and $L2$ regularization as well as penalty parameter λ. This could be explained as collinearities were removed by PCA and unimportant features were filtered by random forest feature selection.

Logistic	Charged Off	Fully Paid	Late 0-30 Days	Late 31-120 Days
Precision	0.9862	0.9962	0.6592	0.7490
Recall	0.9901	0.9978	0.6626	0.7091
F-1	0.9881	0.9970	0.6609	0.7285

Logistic + L1	Charged Off	Fully Paid	Late 0-30 Days	Late 31-120 Days
Precision	0.9895	0.9968	0.6645	0.7470
Recall	0.9916	0.9984	0.6603	0.7210
F-1	0.9906	0.9976	0.6624	0.7339

Logistic + L2	Charged Off	Fully Paid	Late 0-30 Days	Late 31-120 Days
Precision	0.9898	0.9968	0.6642	0.7463
Recall	0.9916	0.9984	0.6599	0.7214
F-1	0.9907	0.9976	0.6620	0.7337

Table 1. Logistic Classification Performance Measure

Based on performance of simple and regularized logistic regressions, as shown in the table above, the best model is selected as $L1$ logistic regression with $\lambda = 0.1$.

8.3.3 Random Forest

Cross validation was applied for model selection in random forest classifier with three parameters: # of trees = 10, 50, 100, 200, 400, max tree depth = 5, 10, 20, 30, 50 and min sample leaf = 5, 10, 15, 25, 40, 70. Cross validation results are in the figures below.

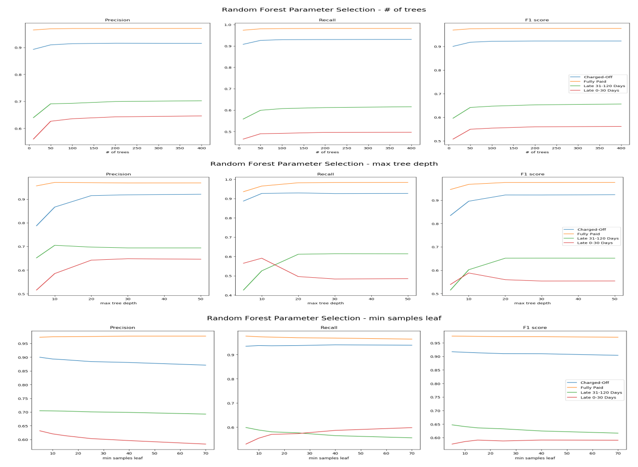

Figure 9. Cross validation of random forest.

With various maximum tree depth and minimum sample leaf, there are trade-off effect in recall and $F1$ between two late classes. The best performance was achieved with # of trees = 150, max tree depth = 15, and min sample leaf = 15.

8.3.4 Neural Network

To find the optimal number of nodes and hidden layers, we first start from 1 hidden layer and set a series number of nodes. By iteratively fitting the model, we compare the performance and pick the best one. Then based on the chosen nodes of first layer, we explored and selected number of nodes for the second layer in a similar manner to the first one. We continue this process until reach optimal number of layers and nodes.

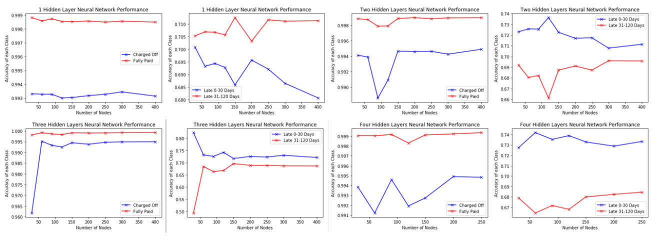
Figure 10. Model selection of neural network (# nodes in hidden layers).

The optimal number of nodes is given by $[120, 400, 150, 150]$.

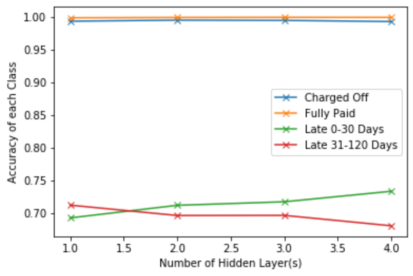

Figure 11. Neural network performance vs. # of layer(s).

The accuracy starts to go down in category Late 31-120 Days when the number of hidden layers is greater than three. So, we pick three as the optimal value of hidden layers.

8.3.5 Model Comparison

We compare both performance and computational cost (training time) of all 4 models under optimal parameters. The table below summarize the results. Multi-SVM has the worst performance among the models and its performance on two late classes were far from satisfying. Although random forest has the least computational cost, its performance was significantly lower than both logistic regression and neural network. $L1$-logistic regression yields close performance to neural network with negligible difference. With both models provide consistent and robust performance, $L1$-logistic regression is more preferable for lower model complexity and training cost.

	F-1 Score				Training Time
	Charged-Off	**Fully Paid**	**Late 0-30 Days**	**Late 31-120 Days**	
L1 - Logistic Regression	0.9906	0.9976	0.6624	0.7339	2,939 s
SVM (rbf)	0.8405	0.9498	0.5757	0.3709	1,997 s
Random Forest	0.9131	0.9736	0.5910	0.6361	356 s
Neural Network	0.9950	0.9986	0.6852	0.7252	14,500 s

Table 2. Model performance and training time summary.

In multi-SVM, random forest and neural network, we all observe trade-off effects in performance (one class increase while the other decrease, and vice versa) between late 0-30 days and late 31-120 days during model selection. This confirms our hypothesis that there is significant confusion between these two classes in the feature space. The consistent ceiling on performance across different algorithms suggest that we are reaching the limit, or the optimal error given the data set.

9 Conclusion & Discussion

We use features engineering and PCA to establish highly independent features, reduce confounding bias, and identify important features. We adapted SMOTE oversampling algorithm to improve the highly imbalanced data set. Our parameters tuning improve our model performance across the performance measure. The conclusion is that our selected multi-class logistic classification model can make high accuracy prediction on the loan status. However, if our original data set could contain more data on the late payments, the model might be more accurate.
The implication of this model can help lenders to make better lending decision based on the loan features. The lenders can achieve excessive returns by releasing their risk constraints and lending more capital to loans that are classified as late (borrowers will pay extra late fees to lenders). This model also helps lenders prevent losses by classifying loans that could be charge-off. Therefore, the machine learning model can help lenders make high accuracy prediction on what the future loan status and build loans portfolios based on the classification.

Acknowledgments

We would also like to acknowledge the Prof Shivani Agarwal for teaching this course, and our project mentor Haoxian Chen for helping us with the project.

References

[1] Hecht-Nielsen, R. (1992). Theory of the backpropagation neural network. In Neural networks for perception (pp. 65-93). Academic Press.

[2] Joe Corliss. Predicting Charge-off from Initial Listing Data,
https://www.kaggle.com/pileatedperch/predicting-charge-off-from-initial-listing-data

[3] Khandani, A. E., Kim, A. J., Lo, A. W. (2010). Consumer credit-risk models via machine-learning algorithms. Journal of Banking Finance, 34(11), 2767-2787.

[4] Luke. Demonstrating Corrupted/Mal-formed Rows,
https://www.kaggle.com/lukemerrick/demonstrating-corrupted-mal-formed-rows

[5] Maaten, L. V. D., Hinton, G. (2008). Visualizing data using t-SNE. Journal of machine learning research, 9(Nov), 2579-2605.

[6] Mathew. Some basic EDA and a how-to guide,
https://www.kaggle.com/mrdweebtastic/eda-with-python

[7] Nathan George. Some basic EDA in R and demo how to load the data,
https://www.kaggle.com/wordsforthewise/eda-in-r-arggghh

[8] Rectifier (neural networks), Wikipedia.
https://en.wikipedia.org/wiki/Rectifier_(neural_networks)

[9] Selecting good features Part III: random forests, Dec 2014.
https://blog.datadive.net/selecting-good-features-part-iii-random-forests/

[10] Shivani Agarwal. Decision Trees and Nearest Neighbor Methods, Feb 2019.
http://www.shivani-agarwal.net/Teaching/CIS-520/Spring-2019/Lectures/Reading/08-decision-trees-nearest-neighbor.pdf

[11] Shivani Agarwal.Kernel Methods, Feb 2019.
http://www.shivani-agarwal.net/Teaching/CIS-520/Spring-2019/Lectures/Reading/06-kernels.pdf

[12] Shivani Agarwal. Principal Component Analysis, Mar 2019.
http://www.shivani-agarwal.net/Teaching/CIS-520/Spring-2019/Lectures/Reading/12-pca.pdf

[13] Shivani Agarwal. Performance Measures, Feb 2019.
http://www.shivani-agarwal.net/Teaching/CIS-520/Spring-2019/Lectures/Reading/10-performance-measures.pdf

[14] Shivani Agarwal. Structured Prediction, Apr 2019.
http://www.shivani-agarwal.net/Teaching/CIS-520/Spring-2019/Lectures/Reading/22-structured-prediction.pdf

[15] Softmax function, Wikipedia.
https://en.wikipedia.org/wiki/Softmax_function

[16] Tsai, Chih-Fong. Chen, Ming-Lun. (2010). Credit Rating by Hybrid Machine Learning Techniques. Applied Soft Computing. 10 (2010) 374-380.