MA4270 Term Paper

Chia Wen Kai A0140215Y
Daniel Koh Chong Xiang A0140195L

19th April 2018

1 Introduction

Defaulting on credit cards payments has been a lingering problem for credit companies for many years. It would be beneficial for credit companies as well as consumers if there exists a technique in determining whether a consumer will would default his/her next credit payment.

In this paper, we will tackle this problem by exploring the effectiveness of various classification techniques taught in class such as SVM and AdaBoost algorithms. In addition, we will also employ techniques not taught such as using Decision Trees and Random Forest Classification.

We ran our experiments using Python 3 due to its extensive libraries and useful packages.

2 Dataset

For this problem, our dataset was obtained from UCI Machine Learning Repository [1] and contains information of consumers from Taiwan in 2005. As such, all currency denominations will be in New Taiwan Dollars (NTD).

The dataset contains 30,000 entries and information such as the amount of credit given, marital status, education and payment information of the past quarters of 2005. In total, we have 23 variables highlighted in Fig. 1 that are taken into consideration during classification.

[1] https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

Variables	Category	Information (all numeric data)
Bal	Numeric	Amount of given Credit(NTD)
Gender	Categorical	1=male, 2= female
Education Level	Categorical	1 = graduate school, 2 = university, 3 = high school; rest = others/unknown
Marital Status	Categorical	1= married, 2 = Single, 3= divorce, 0=others
Age	Numeric	Years of age
Repayment status at Sep 2005	Categorical	-2: No consumption; -1: Paid in full; 0: The use of revolving credit. x ? N: payment delay of x months
Repayment status at Aug 2005	Categorical	-2: No consumption; -1: Paid in full; 0: The use of revolving credit. x ? N: payment delay of x months
Repayment status at July 2005	Categorical	-2: No consumption; -1: Paid in full; 0: The use of revolving credit. x ? N: payment delay of x months
Repayment status at June 2005	Categorical	-2: No consumption; -1: Paid in full; 0: The use of revolving credit. x ? N: payment delay of x months
Repayment status at May 2005	Categorical	-2: No consumption; -1: Paid in full; 0: The use of revolving credit. x ? N: payment delay of x months
Repayment status at April 2005	Categorical	-2: No consumption; -1: Paid in full; 0: The use of revolving credit. x ? N: payment delay of x months
Amount of Bill at Sep 2005	Numeric	Amount of Bill of the person at September 2005
Amount of Bill at Aug 2005	Numeric	Amount of Bill of the person at August 2005
Amount of Bill at July 2005	Numeric	Amount of Bill of the person at July 2005
Amount of Bill at June 2005	Numeric	Amount of Bill of the person at June 2005
Amount of Bill at May 2005	Numeric	Amount of Bill of the person at May 2005
Amount of Bill at April 2005	Numeric	Amount of Bill of the person at April 2005
Amount paid in Sep 2005	Numeric	Amount paid by the person in September 2005
Amount paid in Aug 2005	Numeric	Amount paid by the person in August 2005
Amount paid in July 2005	Numeric	Amount paid by the person in July 2005
Amount paid in June 2005	Numeric	Amount paid by the person in June 2005
Amount paid in May 2005	Numeric	Amount paid by the person in May 2005
Amount paid in April 2005	Numeric	Amount paid by the person in April 2005
Label	Categorical	Y=0 means he is unlikely to default on payment Y=1 means he is likely to default on payment

Fig.1 The 23 features that were used to identify the label

2.1 Categorical Values

In the given dataset, some variables were given categorical values instead of numerical, eg. gender where male = 1 and female = 2. Algorithms may not be able to differentiate categorical values from numerical values thus, to handle such instances, we apply *one-hot encoding* to the dataset.

One-hot encoding maps a category that takes on k possible values into an array of k-length with binary values (0 or 1). For example, if a variable x, takes a categorical value of t, $0 \leq t \leq k$, then its corresponding array would be $(x^{(1)}, x^{(2)}, ..., x^{(k)})$ with $x^{(t)} = 1$ and 0 everywhere else.

We perform *one-hot encoding* on all categorical values, namely, gender, education, marital status and repayment status of the past six months. After *one-hot encoding*, entries in our dataset contains 81 feature vectors.

2.2 Partitioning of Dataset

We partitioned the dataset into 2 parts, 60% of the dataset was used as training data and 40% of the dataset was used as testing data. After the classifier was trained using the training data, we tested the performance of the various classifiers against the testing data.

3 Methods

We used 4 different classifiers for our dataset:

1. Decision Trees
2. Random Forest classifier
3. AdaBoost classifier
4. SVM

3.1 Decision Trees

Decision Trees are arguably the most simple and fastest classification technique which is one of the reasons we included this classifier in our experiments. Algorithms for decision trees usually work top-down, by choosing a feature at each step that best differentiates the entries in the dataset. Different algorithms uses different ways of choosing the feature. Gini impurity, entropy and variance reductions are just a few ways algorithms use in determining the best feature. The chosen features then make up the internal nodes of the trees when training the classifier.

The leaves of the tree are class labels. In our experiment, labels are either 1 for defaulting of next payment or 0, where the next payment would not be defaulted.

During training, the decision tree classifier will model the tree such that there is no training error or minimum training error. After training the classifier, the test samples are passed into the root of the tree, in which it will traverse through its internal nodes to its leaves and output the label corresponding to the leaf.

3.2 Random Forest Classifier

A Random Forest Classifier is an extension of the Decision Tree Classifier where it uses a set of decision trees formed using a randomly selected subset of the training set. The Random Forest classifier then forms an ensemble of decision trees where it averages the votes of the decision trees.

This classifier was chosen in our experiment because a single decision tree may overfit our dataset whereas the aggregate of many decision trees reduces the effect of overfitting which improves the accuracy of our results.

3.3 AdaBoost - SAMME

In class, we were taught the AdaBoost algorithm in which it uses decision stumps to learn a dataset. In this experiment, we used a variant of AdaBoost, which is the AdaBoost - SAMME algorithm. The AdaBoost - SAMME algorithm is highlighted below.

Algorithm 1 AdaBoost-SAMME (n training samples, K labels, M base learners)

1. Initialise weights $\mathrm{W}_0(t) = \frac{1}{n}$ for $t = 1, 2, \ldots, n$
2. For $1 \leq m \leq M$ **do**
3. Fit a base learner h_m that minimises weighted training error
4. Compute error incurred by h_m given by $\epsilon_m = \sum_{t=1}^{n} \tilde{W}_{m-1}(t) \cdot 1\{y_i \neq h_m(x_i)\}$
5. Compute $\alpha_m = log\frac{1-\epsilon_m}{\epsilon_m} + log(K-1)$
6. Update weights given by $W_m(t) = \tilde{W}_{m-1}(t) \cdot exp(\alpha_m \cdot 1\{y_i \neq h_m(x_i)\})$
7. Prediction for new data point $\mathbf{x} = \underset{K}{argmax} \sum_{m=1}^{M} \alpha_m \cdot 1(h_m = K)$

3.4 SVM

In this project we used SVM which uses the Radial Basis Kernel.The standard SVM formulation with offset and slack via the hinge loss is as of follows:

$$\min_{\theta \in R^d,\ \xi_i \in R} \frac{1}{2} \|\theta\|^2 + C \sum_{i=1}^{n} \xi_i$$

subject to

$$y_t(f(x_t)) = y_t(\theta^T \phi(x_t) + \theta_0) \geq 1 - \xi_t \text{ for t=1,2,...,n}$$

$$\xi_i \geq 0$$

We used SVM to find a optimal $\hat{\theta}$ that classifies the training dataset correctly before moving on to the test samples.

4 Results

4.1 Algorithms

Below are the details of the algorithms we used.

1. For decision tree classifiers, the splitting criterion used is Gini impurity which is the default option of Python's library, sklearn. We also do not limit the maximum height of the tree.

2. For random forest classifiers, the splitting criterion used is also Gini impurity so as to ensure consistency in results. The number of decision trees used was 1000 in the experiment.

3. In AdaBoost-SAMME, 1000 decision stumps was used to classify our dataset.

4. In SVM, the kernel used was Radial-Basis Kernel (RBK) and the penalty parameter C was set to 1.

4.2 Results

The results of our experiments are shown in Fig.2.

Algorithm Used	Number of correct predictions	Accuracy in % (3.s.f)
Decision trees	8778	73.2
Random Forest	9956	83.0
AdaBoost(SAMME)	10016	83.5
SVM	9521	79.3

Fig.2 Results of the experiment

Among the algorithms, decision trees took the least time to run. However, their performance is also the worst out of the four classifiers. Using random forest classification resulted in a great improvement over the decision tree classifier. Since we used 1000 decision trees for the random forest classification, the time taken was significantly more than the decision tree classifier.

The performance for AdaBoost-SAMME was similar to the random forest classifier which is surprising since an ensemble of decision stumps were able to do as well as an ensemble of decision trees. Since AdaBoost-SAMME uses decision stumps, runtime for AdaBoost is faster as compared to random forest classifier.

SVM takes the longest to run which is to be expected since it is solving a minimisation problem. SVM also did not do as well as AdaBoost-SAMME and random forest classification.

5 Conclusion

AdaBoost-SAMME had the best performance out of all classifiers for our dataset which has only 80+ features. SVM did not do as well, this could be due to our dataset being linearly inseparable. We could also use cross-validation to find the optimal C and γ then run our SVM. Perhaps this would lead to better results. Decision trees, although simple and efficient, did not do as well as the others. This was improved upon by using random forest classification.

Overall, this project gave us a good learning opportunity to try out the techniques we have learnt during class and motivated us to explore other techniques that were not taught.

6 References

1. Ho, Tin Kam. Random Decision Forests. *Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC*, 14 August 1995. pp. 278282.