Data Mining Coursework

Outline Solution

**Abstract**

In this report we provide a solution to a customer classifying problem with the data from a marketing campaign of a Portuguese banking institution. Exploring the data set with Weka's histogram and scatter plot told us that the attributes do not have strong predictability. Applying different attribute selection techniques further justified our reason for using all the attributes for developing our models. Different classification algorithms were performed on the dataset with different parameters to find the best set up. The best model for equal cost problem is J48 which has the accuracy of 90.5% (under 10-fold cross validation). We used the same models to develop our models for unequal cost problem (misclassified a customer who will subscribe cost 10 times more than the cost of misclassified a customer who won't subscribe) and the best model is still the J48 model with a cost of 11256. Both models were then tested on the provided dataset that has 9042 instances. Their performances were 90.5% and 2838 respectively.

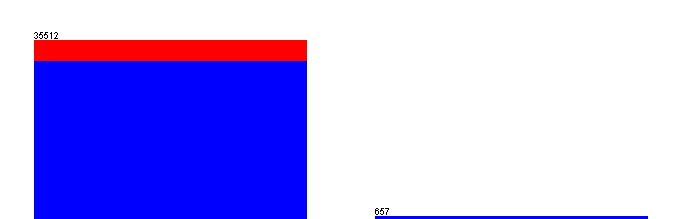
**1. Introduction**

Our task is to develop two models one for equal and one for unequal cost problem. The next parts of this report are: data exploration, data pre-processing, model development and model evaluation.

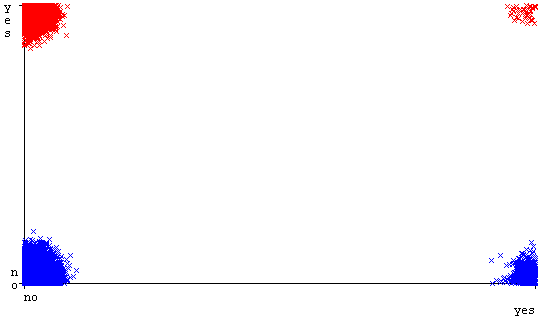
**2. Data exploration**

Exploring the data show that

* The 'no' class made up roughly 88% of the data. This will be used as the accuracy of the default classifier.
* The dataset has no missing values.
* The dataset has 17 attributes and 36,000 instances. Since we have a large amount of data, it would be beneficial to use most of the attributes for our models.
* Some attributes have imbalanced distributions of values:
  + Default: More than 95% of the data have value ‘yes’
  + Loan: Most of the data have value ‘no’
  + The numeric attributes are either skewed or non-Gaussian which could affect the naive Bayes model. We would try to overcome this by applying discretisation.
* Looking at the histograms of each attribute, there was no attribute that was strongly predictive of the class (In the histogram of the 'default' attribute, it might look like that a 'yes' label of this attribute would belong to the class 'no'. But inspecting the scatter plot showing that there are a small number of instances with label 'yes' belong to class 'yes')



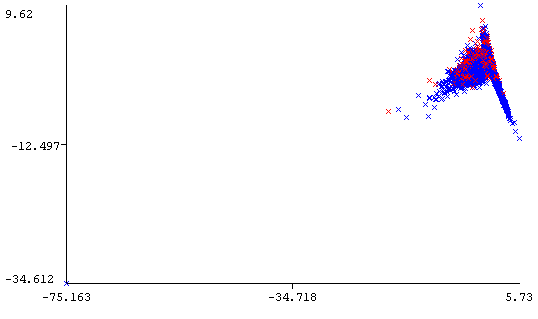
Histogram of default attribute



Scatter plot of default with jitter slide in the middle

**3. Data pre-processing**

There are 7 numeric attributes: age (a1), balance (a6), day (a10), duration (a12), campaign (a13), pdays (a14) and previous (a15).

* Since the attributes are at different scales, we applied standardisation and then PCA. First PC: *-0.663a14-0.633a15+0.286a10+0.268a13-0.067a12...* Second PC: *-0.607a13-0.496a10+0.455a12-0.305a15-0.221a14...*
* Using Weka's visualising tool to plot a scatter plot for the two PCs. Inspecting the plot showed that we do not need to use principal components as new attribute since there is almost no class separation in this space. There are some outlying instances but they are not too far away to be a problem. The only instance that is significantly far away is instance 23347 at (-34.612, -75.163) 

**4. Classification Models**

We develop our models in this section of our report. We carried out our model development following the process in the sample solution: benchmark models; attribute selection; model development; cost-based modelling.

**4.1 Benchmark models**

We applied naive Bayes, k-nearest neighbour (k=9), logistic regression and J4.8 without filtering the data and with the default parameters provided by Weka under 10-fold cross-validation. These models will be used as benchmarks for other models. The value of k for k-nearest neighbour was selected by cross-validation and the best value is 9.

|  |  |
| --- | --- |
| Model | Accuracy |
| naive Bayes | 88.1 % |
| k-nearest neighbour (k = 9) | 89.2% |
| logistic regression | 90.2% |
| J4.8 | 90.4% |

**4.2 Attribute selection**

Perform attribute selection algorithms on the data gave us the following results:

* Cfs subset evaluation gave us five attributes: marital (a3), housing (a7), loan (a8), duration (a12), poutcome (a16)
* Information gain measure: 12,16,14,11,9,1,15,7,2,10,6,13,8,4,3,5
* Symmetric uncertainty measure: 12,16,14,15,9,11,7,1,6,2,8,13,10,3,4,5

According to the results, we can pick out a12 and a16 as they stand out from other attributes. Two is too few for us since we have a large amount of instances. Because of this reason, I will use all the attributes for training our models.

**4.3 Model development**

**4.3.1 Naïve Bayes**

The naive Bayes benchmark model has the accuracy of 88.1% which is slightly better than the default classifier. The fact that most of the numeric attributes are skew could be the reason for this. To address this, we discretised numeric attributes into 10 bins. Experimenting with equal frequency parameter enabled and not enabled gave us:

|  |  |
| --- | --- |
| equal frequency | accuracy |
| true | 88.1% |
| false | 88.4% |

With equal frequency set to false, naive Bayes gave a slightly better model with 88.4% accuracy

**4.3.2 k-nearest Neighbour**

For our k-nearest neighbour model, we will be experimenting with weight distance. For algorithm such as k-nearest neighbour, it is important to normalise our data before applying the algorithm. Experimenting with 1/distance and 1-distance gave us:

|  |  |
| --- | --- |
| weighting | accuracy |
| 1/distance | 89% |
| 1-distance | 89.2% |

Both models gave better accuracy than the default classifier (88%). Using 1-distance weighting gave us a small increase in performance compared to 1/distance weighting.

**4.3.3Logistic regression**

We tried changing the ridge estimator parameter. The results are as following:

|  |  |
| --- | --- |
| Ridge parameter | Accuracy |
| 1 × 10−8 | 90.18% |
| 1 × 10−4 | 90.18% |
| 1 | 90.18% |
| 10 | 90.19% |

There is a negligible improvement when we set the ridge parameter to 10.

**4.3.4 Decision Trees**

We experiment with the parameters that affect the model's complexity.

|  |  |  |
| --- | --- | --- |
| Complexity Control | Parameter Value | Accuracy |
| Post-pruning | 0.35 | 90.1% |
| Post-pruning | 0.30 | 90.3% |
| Post-pruning | 0.25 | 90.4%% |
| Post-pruning | 0.20 | 90.5% |
| Post-pruning | 0.15 | 90.5% |
| Post-pruning | 0.10 | 90.4% |
| Reduced error pruning | ---- | 90.2% |

Again the differences in results are still very little. The model uses post-pruning at 0.15 gave the best accuracy at 90.5%

**4.3.5 Model Selection**

All models perform very close to each other. J48 model with post-pruning at 0.15 gave us the best accuracy so we will choose this model.

**4.4 Unequal cost**

The cost of misclassifying a client who will subscribe to a term deposit is 10 times that of misclassifying a client who will not. The cost matrix would be:

*0.0 1.0*

*10.0 0.0*

If we assign every instance to 'no' class then the cost would be 31,981. This would be our benchmark.

**4.4.1 Naive Bayes**

We discretised the data as above with equal frequency set to false. Then we ran a Costsensitiveclassifer model and set minimizeExpectectedCost to true. The confusion matrix we get:

*23981 8000*

*826 3362*

The cost of this matrix is 826 × 10 + 8000 × 1 = 16260. This is significantly lower than the default classifier

**4.4.2 K-nearest neighbourhood**

We ran k-nearest neighbourhood with weighting 1-distance with costSensitiveClassfier and minimizeExpectedCost set to true.

*21371 10610*

*916 3272*

The cost of this matrix is 916 × 10 + 10610 × 1 = 19770 which is notably higher than the cost of naive Bayes

**4.4.3 Logistic Regression**

We ran logistic regression with 10 as the ridge parameter. Again with costSensitiveClassifier and minimizeExpectedCost set to true.

*25347 6634*

*514 3674*

The cost of this matrix is *514* × 10 + *6634* × 1 = 11774 which is significantly smaller than both naive Bayes and k-nearest neighbourhood

***4.4.4* Decision Trees**

We ran costSensitive Classifier and minimizeExpectedCost set to true with J.48 post-pruning at 0.15.

*26408 5573*

*794 3394*

The cost of this matrix is *794* × 10 + *5573* × 1 = 13513

One interesting I found while experimenting with the J48 algorithm is that setting minimizeExpectedCost to false gave a surprisingly better cost matrix.

*26965 5016*

*624 3564*

The cost is only *624* × 10 + *5016* × 1 = 11256 which is lowest we have obtained.

**4.5.5 Model Selection**

From the cost we have obtained, it is obvious that J48 model is the model that performs the best for unequal cost problem. But it is also worth noticing that with all the models, the cost of misclassifying customers that will subscribe and the cost of misclassifying customers that will not subscribe are quite different. This is not a good property for an unequal cost model to have.

**6. Evaluation and Conclusions**

For the equal cost problem, the performance of the J48 model on our test data was 90.5% which is also the best accuracy achieved by all models. The default accuracy for our test data is 7941 / 9042 = 87.8% (classify all instances as 'no')

For our unequal cost problem, the performance of our J48 model is as following:

*6633 1308*

*153 948*

The cost of this matrix is *153* × 10 + *1308* × 1 = 2838. The default classifier for the test dataset has a cost of 7941. This is significantly higher than what our models achieved.

In conclusion, the J48 model performed the best on both equal and unequal problem. For the unequal cost problem, our model performs significantly better when the 'minimizeExpectedCost' is set to false.