COMP5318 - Machine Learning and Data Mining
assign1 report

Shen, Chun cshe6391@uni.sydney.edu.au - 460317940
Shahrasari, Delaram dsha6256@uni.sydney.edu.au - 470183205
Zhou, Chengcheng czho9311@uni.sydney.edu.au - 460090157

May 8, 2017

Abstract

In apps market training dataset, for each app we have description represented in tf-idf and a set of labels. In test dataset, for predicting the labels for apps based on their description, we use a set of classification methods. Naïve Bayes (BN), Logistic Regression (LR) and K- Nearest Neighbor (KNN) are experimented. After an initial comparison of the three classifiers, Naïve Bayes has been allocated for this classification task which is more appropriate in terms of accuracy and runtime. After discussing the three methods, future work for enhancing the results of the classification is presented.

1 Introduction

Classification is one of the most widely used techniques in machine learning and data mining, with a broad variety of applications, such as document classification, email spam filtering, image classification that has been used to solve difficult real-world problems. Classification process is divided to two phase: 1) learning, by using the training dataset with a classification algorithm (including the data points and their labels) a classifier will be established. Then we measured the performance of our classifier to evaluate it; and 2) classification, where the class label of data points which are provided in a dataset is predicted by using the established classifier.
In this study, the purpose is to use training dataset and training label set in order to use it for predicting the labels of apps in test dataset, according to accuracy and run time we should analyse the tf-idf values and labels then Our three classifiers: NB, LR and K-NN compared and analysed.

2 Method

2.1 Naïve Bayes

Naïve Bayes algorithm based on Bayes' theorem with an assumption of independence among predictors can be used to implement classification assignment. According to bayes theorem, the posterior probability $P(c|X)$ can be calculated from $P(c)$, $P(X|c)$ and $P(X)$. Bayes' theorem can be represented as an equation below:

$$P(c|X) = \frac{P(X|c)P(c)}{P(X)} = \frac{P(c)}{P(X)} \prod_{i=1}^{d} P(x_i|c)$$

• d is the number of attributes and x_i is the value of i_{th} attribute
• $P(c|X)$ is the posterior probability of class given predictor
• $P(c)$ is the prior probability of class
• $P(X|c)$ is the likelihood which is the probability of predictor given class
• $P(X)$is the prior probability of predictor
According to the 30 different classes in this section, $P(x)$ is same. The highest probability of the category for a each specific app. So the formula below can be used to solve classification problems:

$$h_{nb}(X) = argmax(P(c) \prod_{i=1}^{d} P(x_i|c))$$

This equation is Naïve Bayes classifier[6]. Apparently, the training process for Naïve Bayes classi-

fier is to estimate the prior probability of predictor $P(c)$ and conditional probability $P(x_i|c)$ for each attribute based on training set D.
Due to the underlying limits of floating points,the Naïve Bayes classifier equation above should be optimized. Firstly, the probabilities of each attribute (the worlds for app description) should be modified as logarithms of those probabilities. Secondly, instead of multiplying the two parts $P(c)$ and $\prod_{i=1}^{d} P(x_i|c)$, these two parts should be added. After optimizing the Naïve Bayes classifier, this classifier can be modified below:

$$h_{nb}(X) = argmax(\log(P(c)) + \sum_{i=1}^{d} \log(P(x_i|c))$$

In order to avoid "zero frequency", also known as zero probability which means the prediction will be failed, the smoothing technique "Laplace correction" can be used to estimate the probability. $P(c)$ and $P(x_i|c)$ can be corrected:

$$\hat{P}(c) = \frac{|D_c + 1}{|D| + N}$$

$$\hat{P}(x_i|c) = \frac{|D_{c,x_i}| + 1}{|D_c| + N_i}$$

N is the number of categories in training set and N_i is identified as the value of i_{th} attribute.

2.2 Logistic Regression

Logistic regression is a technique for binary classification problems. More specifically, logistic regression seeks to describe data and predict the probability of a series of independent variables on a binary response variable, where two values "0" and "1" can be taken to represent two outcome categories, and classify the observations via the probability that has been estimated[2]. However, in this section, dependent variable has more than two outcome categories. Multinomial logistic regression can be generalized to solve multi-class problems.

2.2.1 Hypothesis representation

To talk about hypothesis representation for logistic regression, hypothesis representation for linear regression should be discussed first. In linear regression, a linear function can be defined as $h_\theta(x) = (\theta^T x)$. For classification hypothesis representation, the hypothesis representation in linear regression should be modified as a different way $h_\theta(x) = g((\theta^T x))$ where the $g(z)$ is a sigmoid function and z is a real number. Sigmoid function or logistic function can be used to represent hypothesis in classification. Sigmolid function is a kind of S-shaped function returning value from 0 to 1 monotonically increasing on $\mathbb{R}$. The Sigmoid function can be represented as a formula mathematically below:

$$g(z) = \frac{1}{1 + e^{-z}}$$

With the combination of these two equations mentioned, the hypothesis can be described mathematically. For example, a training set should be given $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(m)}, y^{(m)})\}$ and if predicting a probability pertains to the "1" class against the probability that it pertains to the "0" class, the hypothesis should be learned as the formula below:

$$P(y = 1|x) = h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}} \equiv g((\theta^T x))$$

$$P(y = 0|x) = 1 - P(y = 1|x) = 1 - h_\theta(x)$$

Those functions have two parameters: x is training set variable and θ is weight matrix of the training set. After defining the hypothesis of logistic regression, the objective can be transfered to find a set of fit θ parameters used in mathematic model for future predictions.

2.2.2 Cost function

The cost function can be used to determine the parameter θ and it should be a convex cost function but not a non-convex cost function because only convex cost function can be applied for gradient descent method to reach a global minimum[3].The cost function used for logistic regression can be desicribed and simplified below (binary classification problems, y is always 0 or 1):

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} Cost(h_\theta(x^{(i)}, y^{(i)}))$$

$Cost(h_\theta(x), y) = -y \log(h_\theta(x)) - (1-y) \log(1-h_\theta(x))$
The cost function for the θ parameters can be represented below:

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log(h_\theta(x^{(i)})) - (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))$$

2.2.3 Optimization: gradient descent method

In order to search for fit θ pamamters, gradient descent method should be used for minimizing logistic regression cost function $J(\theta)$ repeatedly[4].

$$Repeat\{\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)\}$$

This code is to update each j in θ sequentially.

2.2.4 Multi-class classification

Apparently,logistic regression classifier can only solve binary classification problems.In point of fact,multi-class classification problems can also be figured out using logistic regression model which can be generalized from binary classification.In this section, 20104 apps can be classified into 30 multiple classes that is a multi-class classification problem.In oder to classify those apps based on logistic regression, one vs. all classification concept can transfer multi-class classification problems to binary classification work. For example, one of the 30 multiple classes,'photography',is defined as "1" class and the rest of 29 classes can be identified as "0" class. After training the model,30 classifiers can be output.

2.3 K-Nearest Neighbours

K nearest neighbor (K-NN) is a simple example of a supervised non-parametric classifier [7] which identifies the distance between data points in the training data set and each of the test data points. It retrieves data points in training data set that are similar to new data points which need to be classified. The simplest case is when k=1 where we find a data point that is closest (the nearest neighbor). Despite being simple, it can be powerful when we have a large number of data points in our training set [1]. Following 1-NN, for k > 1, the k most similar data points are used to classify a new data point. KNN computes similarity between data points based on a distance function. For instance, in movie recommendation, distance can be calculated based on similarity or commonality among their features such as actors, directors, genre, etc. As another example, in our assignment, each app in training data can be identified as a data point and each app can be defined as a point in test data. Afterwards, the distances between each data points in the training dataset and each test data points in the test dataset is computed by the K-Nearest Neighbors classifier where k nearest neighbor points will be selected next, and finally the label of each test data point can be predicted according to the labels of the k nearest neighbors. Depending on the application, similarity methods can be calculated in various ways. The most common methods include:[5]

• Euclidean Distance function (Vector Space):

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

• Manhattan (Vector Space):

$$\sum_{i=1}^{k}|x_i - y_i|$$

• Minkowski:

$$\left(\sqrt{\sum_{i=1}^{k})\ |x_i - y_i|}\right)^{1/q}$$

• Cosine Method (Vector Space):

$$Cosine(X,Y) = \frac{X \cdot Y}{\|X\| \times \|Y\|} = \frac{\sum_{i=1}^{n} x_i \times y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \times \sqrt{\sum_{i=1}^{n} y_i^2}}$$

• Jaccard Method (Set-Based):

$$Jaccard(X,Y) = \frac{X \cap Y}{X \cup Y}$$

3 Experiment&Result

3.1 Experiment Process

Given three classifiers mentioned above, Naïve Bayes Classifier was finally chosen to be implemented in this assignment. K-Nearest Neighbors Classifier was wiped out of option due to its huge computing. Logistic Regression Classifier, on the

other hand, required large amount of vector multiply in float mode which make the training process slow. In that, Naïve Bayes Classifier became the decent choice if the data would be processed in feasible time.

In the aim of evaluation, ten-fold cross validation was applied in this experiment. Training data was divided into ten parts. Each part took turns playing role of testing data and the rest parts were combined, becoming the training data of that loop. After that, the real test dataset was introduced and the predicted labels of test data have been calculated based on this classifier. The output file predicted_labels.csv has been attached in the submitted folder.

3.2 Result evaluation

The accuracy of Naïve Bayes Classifier based on given training data is listed as Table 1 on page 4. The minimal accuracy rate happens on ninth loop, 47.16%, while the maximum rate occurs on fourth loop, 55.52%. The average accuracy rate is calculated as 52.25%, in that case, the estimated accuracy of our classifier is around 52.25%. It is noteworthy that accuracy here means the correctness rate of predicted classes. If one of the test data is correctly classified, we count one. Vice versa, if one of the test data is incorrectly classified, we count zero. The correctness rate is the percentage of correctly classified data out of all test data.

Loop	Test Data Number	Accuracy
1	1-2010	53.68 %
2	2011-4020	52.24 %
3	4021-6030	53.88 %
4	6031-8040	55.52 %
5	8041-10050	49.50 %
6	10051-12060	54.33 %
7	12061-14070	54.23 %
8	14071-16080	52.99 %
9	16081-18090	47.16 %
10	18091-20104	50.99 %
Average	1-20104	52.25%

Table 1: 10-Fold Cross Validation for Naïve Bayes

In order to perform further statistics, a confusion matrix plan was executed at the first place. The Table 2 on page 7 represents the result from first loop. As this is a multi-class classification problem, when it comes to calculate precision and recall, the estimated classes not belonging to actual classes are regard as No. On the other hand, the correct predicted tests are regraded as Yes. Through this method, this multi-class classification problem is transfered to binary classification problem. According to common definition of Cost-Sensitive Measures:

$$\begin{aligned} Precision(p): &\quad \frac{TP}{TP+FP} \\ Recall(r): &\quad \frac{TP}{TP+FN} \\ F-Measure(F): &\quad \frac{2TP}{2TP+FN+FP} \end{aligned}$$

A more detailed performance table is pictured on page 8 (Table 3). Analysis based on this table is listed as follows:

- The precision of different labels vary obviously. The lowest precision,14.04%, is located at Libraries and Demo. Conversely, the highest precision, highest precision,94.73% is located at Sports Games.
- Precision index express the probability of correctness of result when specific class is predicted. Top 3 precision labels are Sports Games, Medical, Racing. That means when predicted results are these three labels, they are highly likely correctly predicted.
- The recall values also differ ranging from lowest 27.50% to highest 81.69%.
- Recall index means how often does it predict Yes when it is actual Yes. In other words, it represent how specific class is recognized. As the table shows, Cards and Casino, Racing labels are recognized well.
- It is noteworthy that the Precision an Recall of Comics are zero. It says this classifier is unable to classify this label very well.
- Confusion matrix in Table 2 says the Travel and Local label is high likely to be classified into Transportation label. In the real world, these two labels may occur to same application because transportation guide applications are in high demand among visitors, especially overseas visitors.

After we finished all loops and averaged the values in confusion matrix, an average confusion matrix was introduced in Table 4 on page 9. Based on this table, an average cost sensitive measure table was created. New finds are as follows:

- The Precision of Comics increases to a quite high level with Recall very low. In other word, the recognition of this class is still unreliable.
- F-measure is defined as the weighted harmonic mean of the precision and recall of the test. In this average matrix, f-measure is attached. From this prospective, Cards and Casino and Finance classes are suitable data for our classifier since they get high mark on f-measure.
- Classify on Comics, Libraries and Fitness, Lifestyle performed bad on out classifier. In that, if there are too much entries belongs to these classes in the test dataset, performance of the classifier may be lower than expected one.

3.3 Hardware & Runtime

The hardware specific is list below:

Processor	2.7GHz Intel Core i5
Memory	8GB 1867 MHz DDR3
Storage	128GB SSD
Interpreter	python3.6.0

Using whole training_data.csv and training_labels.csv as training dataset and test_data.csv as test dataset, the execution status of classifier is as follows:

File Loading Time	52s
Training Time	256s
Judgment Time	702s

4 Discussion

During implementation of classifier, some discussions are taken to make the program work:

- "How to apply tf-idf value to Naïve Bayes model?" comes the first question. At the beginning, we decided to do our own preprocessing from train_desc.csv. Since tf-idf looks mean nothing to this model. Later on, we found if a threshold was set to tf-idf matrix, the matrix would be transfered to Trues-False table which was familiar to us in Naïve Bayes model. We set the threshold to zero finally and regard the processed data set as Bernoulli Naïve Bayes Model in our code.
- "How to deal with floating point underflow?". Since there are about 13 thousands attributes in the dataset, if the likelihood is small, the float data type cannot store the result of multiply all of them. We add logistic function to each likelihood to solve this problem though Naïve Bayes itself does not require this operation.
- "How to deal with class which does not own enough attributes?". There are 30 classes in the dataset, if train data split into 30 classes, there is limited attributes in each split. In that case, likelihood of partial attributes will decrease into zero, which leads to mathematic error of the program. To avoid this situation, when we do statistics on likelihood, we let the numerator start with 1 and denominator start with 2. This is what did to follow the principle of Laplace smoothing.

5 Conclusions&Future Works

In this section, after discussing the results of prediction by using Naïve Bayes classifier and estimating the probable accuracy of predictions by using other classifiers such as Logistic regression and KNN classifier, the following conclusions can be drawn:
1) In this assignment, Naïve Bayes Classifier made a significant performance in handling the classification problem. The accuracy of its prediction and operating efficiency held relatively feasible results.
2) Through analysing the results of prediction and making considerable research, we try to find more feasible ways to optimize the classifier or analyse features and distribution of dataset from the root.

In the future, in order to make a better performance of the classifiers, the following parts should be emphasised:
1) In terms of improving the accuracy, term frequency should be considered instead of using the training data for tf-idf values directly. Multinomial Naïve Bayes model can be used to calculate likelihood based on count of a word/token. Furthermore, future work should be given more focus on processing it-idf values because in this section we neglected the weight meaning of tf-idf values behind.
2) In terms of holding a better computing efficency, Principal Components Analysis(PCA) algorithm is one of the optimized methods to reduce the high-dimensional training dataset to lower-dimensional dataset based on the premise of not lossing the accuracy of predictions.

A Code Usage Instructions

1. restore training dataset into the input folder
2. execute main.py under the algorithm folder with python3 interpreter
3. check output folder, the predicted_labels.csv will be created after the program is finished
4. the code of ten-fold cross validation is saved in experiment.py. if this python script is executed, an output of average confusion matrix will be printed in console.
5. this program is written in pycharm community version but same IDE is not required to execute the submitted version of program

References

[1] 15.062 data mining. spring 2003.

[2] Weiwei Cheng and Eyke Hüllermeier. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76(2):211–225, 2009.

[3] Eugene Demidenko. Sample size and optimal design for logistic regression with binary interaction. *Statistics in medicine*, 27(1):36–46, 2008.

[4] Llew Mason, Jonathan Baxter, Peter L Bartlett, and Marcus R Frean. Boosting algorithms as gradient descent. In *Advances in neural information processing systems*, pages 512–518, 2000.

[5] Punam Mulak and Nitin Talhar. Analysis of distance measures using k- nearest neighbor algorithm on kdd dataset.

[6] Kevin P Murphy. Naive bayes classifiers. *University of British Columbia*, 2006.

[7] Kevin P Murphy. *Machine learning: a probabilistic perspective*. Cambridge, MA, 2012.

	A	B	B	B	C	C	C	C	E	E	F	H	L	L	M	M	M	N	P	P	P	R	S	S	S	S	T	T	T	W
Arc	38	0	9	0	3	8	0	0	0	0	0	0	4	3	0	0	0	1	0	0	0	8	0	0	0	1	0	0	0	0
Boo	1	38	0	1	0	2	0	0	4	1	0	4	7	9	0	1	1	3	0	0	0	0	0	3	0	0	1	1	0	0
Bra	3	0	54	0	2	3	0	0	4	1	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Bus	0	0	0	29	0	0	0	3	0	0	8	4	10	8	1	0	1	6	0	0	5	0	7	6	0	0	1	1	1	0
Car	1	0	2	0	58	3	0	0	0	0	0	0	2	1	3	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Cas	6	0	17	0	3	51	0	0	1	1	0	1	4	4	1	0	1	0	1	1	0	1	0	0	0	0	0	0	0	0
Com	1	5	0	0	0	0	0	0	0	4	0	0	5	2	2	0	0	3	1	3	0	0	1	6	0	0	1	0	0	0
Com	0	0	0	2	0	0	0	27	0	0	1	0	5	4	0	0	0	1	1	0	2	0	1	12	0	0	7	1	0	0
Edu	0	8	6	1	0	2	0	0	51	2	0	0	9	8	2	0	0	2	1	0	1	0	0	0	0	0	1	0	0	1
Ent	1	0	1	0	0	0	0	1	0	15	0	0	6	3	5	0	1	3	2	4	2	0	1	4	0	0	3	0	0	0
Fin	0	0	0	2	0	0	0	1	0	0	45	0	3	3	0	0	0	4	0	0	3	0	2	0	0	0	1	0	1	0
Hea	0	0	0	0	0	0	0	0	0	1	1	38	5	11	0	4	2	1	0	0	0	0	1	1	0	0	1	0	1	0
Lib	0	0	1	0	0	0	0	0	2	3	0	1	25	6	1	0	1	0	10	1	0	0	0	0	0	0	3	0	0	0
Lif	0	4	0	0	0	2	0	0	1	0	1	0	3	21	1	0	3	1	2	0	2	0	5	6	0	0	3	2	0	0
Med	0	0	0	1	0	0	0	0	0	7	0	0	15	3	38	0	2	6	2	8	2	0	0	2	0	0	5	0	0	0
Med	0	0	0	0	1	0	0	0	0	0	0	7	7	8	1	45	1	2	1	0	1	0	1	0	0	0	0	0	0	0
Mus	0	0	0	0	0	0	0	1	2	3	0	4	5	5	10	0	51	4	0	0	1	0	0	0	0	0	1	0	0	0
News	0	0	0	0	0	0	0	2	0	0	0	0	2	1	1	0	0	25	0	0	0	0	0	1	0	0	0	0	0	1
Per	1	0	0	0	0	5	0	2	0	4	0	0	2	1	0	0	1	0	31	4	2	0	0	0	0	0	0	0	0	0
Pho	0	0	0	0	0	1	0	0	0	1	0	0	5	2	1	0	0	1	2	38	0	0	0	3	0	0	0	0	0	0
Pro	0	0	0	3	0	0	0	4	0	0	2	0	4	2	0	1	1	0	1	0	11	0	0	0	0	0	9	2	0	0
Rac	4	0	0	0	2	2	0	0	0	0	0	0	5	0	0	0	1	0	5	1	0	93	0	0	2	0	0	0	0	0
Sho	0	0	0	3	0	0	0	0	0	0	0	0	4	4	1	1	0	1	0	0	5	0	49	1	0	0	0	0	1	0
Soc	0	1	0	1	0	1	0	6	1	0	1	0	7	9	0	0	0	2	0	1	3	0	1	33	0	0	0	2	1	0
Spo	1	1	1	0	0	0	0	0	2	1	0	1	3	2	1	0	1	2	0	3	2	0	0	1	29	0	1	1	0	1
Spo	5	0	1	0	7	1	0	0	0	0	0	0	2	0	0	0	0	0	4	2	0	5	0	0	7	18	0	0	0	0
Too	0	0	0	1	0	0	0	8	1	1	1	0	11	2	5	0	2	1	2	1	6	0	0	1	0	0	33	2	0	2
Tra	0	0	0	1	0	0	0	1	0	0	1	0	8	4	0	0	0	2	0	0	0	0	1	2	0	0	4	54	4	0
Tra	0	1	0	0	0	0	0	1	2	0	3	0	4	3	0	0	0	0	0	0	0	0	2	2	0	0	0	15	17	0
Wea	0	0	0	0	0	0	0	0	0	0	0	0	2	1	0	0	0	4	1	0	0	0	0	0	0	0	0	5	0	24

Table 2: Confusion Matrix of First 2,000 Entries (Naïve Bayes)

No.	label	Accuracy	Precision	Recall
1	Arcade and Action	96.96%	61.29%	50.66%
2	Books and Reference	97.06%	65.51%	49.35%
3	Brain and Puzzle	97.21%	58.69%	75.00%
4	Business	96.11%	64.44%	31.86%
5	Cards and Casino	98.45%	76.31%	81.69%
6	Casual	96.41%	62.96%	54.83%
7	Comics	98.30%	0.0	0.0
8	Communication	96.66%	47.36%	42.18%
9	Education	96.81%	71.83%	53.68%
10	Entertainment	96.66%	33.33%	28.84%
11	Finance	98.05%	70.31%	69.23%
12	Health and Fitness	97.46%	63.33%	56.71%
13	Libraries and Demo	90.94%	14.04%	46.29%
14	Lifestyle	92.78%	16.15%	36.84%
15	Media and Video	95.57%	51.35%	41.75%
16	Medical	98.15%	86.53%	59.99%
17	Music and Audio	97.26%	72.85%	58.62%
18	News and Magazines	97.11%	33.33%	75.75%
19	Personalization	97.11%	46.26%	58.49%
20	Photography	97.76%	56.71%	70.37%
21	Productivity	96.71%	22.91%	27.50%
22	Racing	98.20%	86.91%	80.86%
23	Shopping	97.81%	68.05%	69.99%
24	Social	95.57%	38.82%	47.14%
25	Sports	9830%	76.31%	53.70%
26	Sports Games	98.25%	94.73%	34.61%
27	Tools	95.52%	43.42%	41.24%
28	Transportation	97.01%	62.79%	65.85%
29	Travel and Local	97.91%	65.38%	34.00%
30	Weather	99.10%	82.75%	64.86%

Table 3: Cost Sensitive Measure of First 2,000 Entries (Naïve Bayes)

	A	A	B	B	B	C	C	C	C	E	E	F	H	L	L	M	M	M	N	P	P	P	R	S	S	S	S	T	T	T
Arc	36.2	0.1	7.2	0.1	2.9	5.5	0	0.1	0.1	0.6	0.	0.	3.8	0.8	0.6	0.	0.2	0.2	0.7	0.	0.1	6.7	0.	0.	0.1	0.8	0.3	0.	0.	0.
Boo	0.5	28.5	0.4	0.9	0.	1.1	0.1	0.9	6.6	1.5	0.3	1.9	9.7	9.1	1.	0.8	0.4	1.9	0.2	0.2	2.1	0.	0.3	2.5	0.	0.	0.9	0.7	0.3	0.2
Bra	5.2	0.2	43.5	0.	4.1	3.1	0.	0.	4.3	0.7	0.	0.3	4.3	0.5	0.6	0	0.5	0.2	0.2	0.2	0.2	0.8	0	0.4	0.	0.	0.5	0.1	0.	0.
Bus	0.	0.6	0.3	23.4	0.	0.1	0	2.2	1	0.	4.1	1.3	6.2	3.4	1.8	0.2	0.3	3.9	0.1	0.4	4.9	0	5	2.4	0.	0.	3.1	2.1	1.6	0.
Car	0.8	0.1	1.8	0.2	57.7	1.4	0	0	0.4	0.4	0	0	4.5	0.6	0.7	0.	0.2	0.	0.1	0.1	0.2	0.2	0	0.4	0.1	0.1	0.3	0	0	0
Cas	6.9	0.1	12.5	0	3.5	27.8	0	0.1	1.6	2.9	0	0.2	4.8	4.3	0.2	0	0.7	0	1.7	1.6	0	2.3	0	0.9	0	0.3	0.2	0	0	0
Com	1.2	2.	0	0	0	0.4	2.3	0	0.8	4.3	0.1	0	5.7	2.5	1.5	0	0.2	1.9	6.9	1.5	1.1	0	0.1	2.2	0	0	1.1	0.1	0	0
Com	0.	0.2	0.	1.9	0.	0.	0.	27.1	0.2	0.2	0.6	0.1	7.7	2.3	2.5	0.	0.	0.5	4.6	0.3	4.6	0.	0.5	11.	0.	0.	5.8	0.8	0.6	0.1
Edu	0.	7.	3.4	1.7	0.	2.6	0.	0.3	36.8	1.2	0.3	0.4	7.4	4.1	0.9	0.6	1.2	1.5	0.2	0.4	1.4	0.	0.1	1.	0.	0.	0.6	0.3	0.4	0.5
Ent	1.7	0.4	1.1	0.	0.3	2.7	0.1	0.9	0.9	16.9	0.	0.3	5.4	4.4	8.5	0.1	3.3	2.1	2.8	5.8	0.9	0.	1.1	5.5	0.8	0.	2.5	0.2	0.3	0.2
Fin	0.	0.1	0.	2.2	0.2	0.	0.	0.3	0.1	0.1	51.7	0.	3.6	2.	0.	0.1	0.	2.7	0.1	0.	1.	0.	2.6	0.3	0.	0.	1.2	1.4	0.5	0.
Hea	0.1	0.2	0.2	0.2	0.	0.5	0.	0.	0.1	0.1	0.5	43.6	6.2	9.5	0.5	4.7	1.4	0.9	0.2	0.	0.7	0.	1.6	1.4	0.5	0.	1.4	1.9	0.2	0.
Lib	0.1	0.5	0.2	0.2	0.	0.2	0.	0.8	0.8	0.9	0.	0.1	30.3	2.3	0.8	0.	0.2	0.	4.	1.7	0.7	0.	0.1	0.9	0.	0.	2.8	0.3	0.	0.
Lif	0.1	3.8	0.	1.5	0.1	1.8	0.	0.9	0.9	1.7	0.6	2.1	8.9	20.2	1.2	0.2	1.1	2.1	1.8	2.3	1.7	0.	9.5	5.1	0.2	0.	2.8	1.	1.	0.
Med	0.2	0.4	0.1	0.5	0.	0.2	0.	0.4	0.3	3.7	0.	0.1	9.	1.7	36.8	0.	3.8	4.5	0.6	4.1	2.	0.	0.2	2.	0.	0.	3.	0.	0.2	0.
Med	0.	0.2	0.2	0.7	0.1	0.	0.	0.2	0.8	0.2	0.3	7.3	6.	3.5	0.4	40.	0.8	1.5	0.2	0.2	0.7	0.	0.7	0.4	0.	0.	0.6	0.4	0.3	0.
Mus	0.2	1.	0.1	0.1	0.	0.6	0.1	0.8	1.5	2.5	0.	1.3	5.6	1.9	9.3	0.	36.8	3.7	0.9	0.	1.4	0.	0.	1.1	0.4	0.	3.	0.1	0.1	0.
News	0.1	0.6	0.	1.6	0.4	0.1	0.	0.7	0.2	0.6	0.2	0.	2.7	1.9	0.9	0.1	0.3	54.6	0.	0.3	1.4	0.1	0.2	1.3	0.6	0.	0.5	0.5	0.4	0.7
Per	0.4	0.1	0.	0.	0.	0.7	0.	0.5	0.	2.2	0.	0.2	6.5	0.9	0.7	0.	0.8	0.1	51.	1.8	1.7	0.1	0.2	0.1	0.2	0.	2.8	0.2	0.	0.3
Pho	0.1	0.2	0.2	0.2	0.	0.4	0.1	0.	0.1	1.4	0.	0.	5.5	2.	2.7	0.	0.	0.3	3.9	47.8	1.3	0.2	0.4	4.	0.	0.	1.5	0.	0.4	0.2
Pro	0.	0.1	0.	3.8	0.	0.	0.	4.	0.5	0.1	1.1	0.6	6.	2.	3.8	0.2	0.5	0.3	3.6	1.2	19.5	0.1	1.2	4.3	0.	0.	11.5	0.6	0.6	0.
Rac	4.9	0.2	1.9	0.	1.3	1.5	0.	0.	0.1	0.7	0.	0.1	2.1	0.5	0.	0.	0.3	0.1	1.5	0.7	0.	48.4	0.1	0.1	0.2	0.	0.6	0.1	0.	0.
Sho	0.	0.1	0.	2.5	0.1	0.1	0.1	1.	0.1	0.2	0.6	0.4	5.1	4.3	0.2	0.1	0.	0.9	0.2	0.1	1.6	0.1	51.7	1.1	0.	0.	0.2	0.8	0.7	0.
Soc	0.	1.2	0.	1.4	0.2	0.4	0.	4.6	0.4	0.6	0.3	0.2	6.1	5.8	1.2	0.1	0.	4.1	0.5	1.7	1.6	0.	0.3	39.6	0.2	0.	0.7	0.6	0.8	0.
Spo	0.1	0.3	0.4	0.5	0.5	0.	0.	0.	0.4	0.5	0.1	2.1	4.9	2.3	0.9	0.1	0.4	7.1	1.3	0.3	1.	0.3	0.1	0.7	34.1	0.2	0.4	1.5	0.9	0.7
Spo	3.6	0.	3.1	0.	4.2	1.6	0.	0.	0.	0.2	0.	0.	3.1	0.5	0.2	0.	0.1	0.4	0.6	0.4	0.	4.4	0.2	0.1	4.4	16.	0.	0.	0.1	0.
Too	0.	0.4	0.2	1.1	0.3	0.1	0.	4.	0.2	0.5	1.1	0.5	13.1	2.1	4.7	0.	0.4	0.3	2.4	0.9	7.4	0.	0.4	1.5	0.1	0.	26.6	1.9	0.4	0.3
Tra	0.	0.	0.	1.1	0.	0.	0.	0.3	0.	0.1	0.9	0.1	6.2	0.9	0.3	0.2	0.	1.4	0.	0.1	1.2	0.3	0.9	0.8	0.	0.	2.1	49.8	4.8	0.1
Tra	0.	1.3	0.1	0.7	0.	0.1	0.	0.7	1.1	0.1	1.	0.	5.5	1.9	0.2	0.	0.	1.3	0.	0.	0.4	0.	2.5	1.9	0.2	0.	1.5	20.6	30.	0.2
Wea	0.1	0.	0.	0.	0.	0.2	0.	0.	0.	0.	0.1	0.2	4.8	0.8	0.1	0.1	0.	6.3	2.3	0.1	1.4	0.	0.1	0.5	0.	0.	2.3	2.8	0.4	25.8

Table 4: Average Confusion Matrix (Naïve Bayes)

No.	label	Accuracy	Precision	Recall	F-measure
1	Arcade and Action	97.15%	57.92%	53.94%	55.86%
2	Books and Reference	96.72%	57.11%	39.04%	46.38%
3	Brain and Puzzle	97.00%	56.56%	62.23%	59.26%
4	Business	96.61%	50.32%	34.21%	40.73%
5	Cards and Casino	98.46%	76.02%	82.07%	78.93%
6	Causal	96.50%	52.25%	38.29%	44.20%
7	Comics	98.30%	82.14%	06.40%	11.89%
8	Communication	96.60%	53.34%	37.84%	44.28%
9	Education	96.96%	61.02%	49.52%	54.68%
10	Entertainment	95.99%	37.47%	24.42%	29.57%
11	Finance	98.47%	80.90%	73.64%	77.11%
12	Health and Fitness	97.37%	68.76%	56.91%	62.29%
13	Libraries and Fitness	90.64%	15.09%	63.25%	24.38%
14	Lifestyle	93.47%	20.40%	27.82%	23.54%
15	Media and Video	95.85%	44.23%	49.86%	46.88%
16	Medical	98.34%	84.03%	60.88%	70.61%
17	Music and Audio	97.37%	68.27%	50.75%	58.23%
18	News and Magazines	96.68%	52.09%	76.90%	62.11%
19	Personalization	96.91%	55.07%	71.32%	62.15%
20	Photography	97.43%	64.42%	65.56%	64.99%
21	Productivity	95.58%	31.35%	29.11%	30.51%
22	Racing	98.37%	75.62%	74.00%	74.81%
23	Shopping	97.56%	64.54%	71.50%	67.85%
24	Social	95.67%	42.35%	54.54%	47.68%
25	Sport	98.20%	80.99%	54.91%	65.45%
26	Sports Games	98.57%	91.95%	37.03%	52.81%
27	Tools	95.10%	32.92%	37.51%	35.07%
28	Transportation	96.97%	56.08%	69.55%	62.09%
29	Travel	97.19%	66.66%	42.07%	51.59%
30	Weather	98.70%	88.05%	53.30%	66.41%

Table 5: Average Cost Sensitive Measure (Naïve Bayes)