COMP30027: Project 1

Q1, Q2, Q3, Q6, Q7

What is labelled data worth to Naive Bayes?

Authors:
Chirag Rao Sahib 836011
Maleakhi Agung Wijaya 784091

March 28, 2018

Question 1

Since we're starting off with random guesses, it might be surprising that the unsupervised NB works at all. Explain what characteristics of the data cause it to work pretty well (say, within 10% Accuracy of the supervised NB) most of the time; also, explain why it utterly fails sometimes

Although the unsupervised classifier is not provided with labelled training data the classifier is still able to discover basic patterns since the inherent structure, and class-attribute relationship is maintained. It is important to note that the datasets provided were not randomly generated, and as with any real-life observation are likely to contain clusters; described by a particular probability distribution. However a key difference is that the unsupervised version is likely to invert the detected pattern due to the lack of class labels. The unsupervised NB is detecting a pattern but unsure as to which class it relates to; resulting in the need for class 'swapping'. The crux of why the unsupervised NB works at all, is that the approximate distribution of the unlabelled training data can still be modelled without the class/cluster labels.

More specifically we observe that the unsupervised accuracy is correlated with the skewness of a datasets class distribution. For example datasets with uneven class distributions such as 'breast-cancer' (70/30), 'car' (70/22/4/4) and 'hypothyroid' (95/5) tend to 'work pretty well' when compared to the supervised classifier. However the unsupervised accuracy of 'mushroom' which has a relatively even class distribution (52/48) decreases greatly in comparison. This relationship between class distribution and unsupervised accuracy is further supported since the accuracy of 'hypothyroid', the dataset with the most skewed class distribution (95/5), barely changes. It is of interest that the accuracy of the 'hypothyroid' dataset in both environments (supervised, unsupervised) is extremely close to the probability of the majority class. On the other hand the dataset with the most even class distribution (52/48), 'mushroom' experiences the greatest reduction in accuracy. In fact in many trials, accuracies as low as 0.54 were observed for the 'mushroom' dataset; in which case the classifier is no better than a coin flip. The performance cannot be guaranteed for datasets with even class distributions. Hence a scenario where the classifier may utterly fail is when the class distribution of a dataset is extremely even. See 'Class distributions of each dataset' in Question 2.

Unsupervised NB also fails when the initial class probabilities are uniformly distributed. When the initial distribution is uniform (or close to it) the probability of each class (for an instance) is set to 1 / c, where c is the number of classes. After iterating, the new class distributions will be exactly the same as the previous iteration because the classifier randomly selects either class (both equally likely).

```
Accuracy (supervised), average of 10 runs, holdout
2018S1-proj1_data/breast-cancer-dos.csv | Avg. Measure: 0.7552447552447551
2018S1-proj1_data/car-dos.csv | Avg. Measure: 0.8738425925925926
2018S1-proj1_data/hypothyroid-dos.csv | Avg. Measure: 0.9517966999775516
2018S1-proj1_data/mushroom-dos.csv | Avg. Measure: 0.991999015263417

Accuracy (unsupervised), median of 10 runs
2018S1-proj1_data/breast-cancer-dos.csv | Avg. Measure 0.7027972027972028
2018S1-proj1_data/car-dos.csv | Avg. Measure 0.7002314814814815
2018S1-proj1_data/hypothyroid-dos.csv | Avg. Measure 0.9522605121719886
2018S1-proj1_data/mushroom-dos.csv | Avg. Measure 0.6338010832102413
```

Question 2

When evaluating supervised NB across the four different datasets, you will observe some variation in effectiveness (e.g. Accuracy). Explain what causes this variation. Describe and explain any particularly suprising results.

Upon inspection it appears that datasets with fewer instances, namely breast-cancer and car, consistently have lower accuracies. However closer examination reveals that the class distributions are uneven. The 'breast-cancer' dataset contains 201 instances with class 'no-recurrence-events' and 85 with class 'recurrence-events'. While the 'car' dataset contains 1210 instances with class 'unacc', 384 instances with class 'acc', 69 instances with 'good' and 65 instances with 'vgood'.

The accuracy for the 'breast cancer' and 'car' datasets was only around 75% and 85% respectively. These datasets likely do not contain enough instances (286 and 1728, respectively) to capture the inherent variability of the data. Although Naive Bayes functions correctly for datasets with fewer instances, the accuracy suffers as the probabilistic relationship cannot be accurately described. On the other hand, a particularly surprising result is the accuracy achieved with the 'hypothyroid' and 'mushroom' datasets, which contained around 3000 and 8000 instances respectively. For the 'hypothyroid' dataset the class distribution is as follows: 4.77% (hypothyroid) and 95.23% (negative). In this case the classifier functions similarly to Zero-R and would classify most of the test instances as negative (95% accuracy); a false sense of accuracy. For the 'mushroom' dataset the classifier benefits from both a relatively even class distribution and the large number of instances available, producing the highest observed accuracy.

We hypothesize that datasets with lower accuracy also contain numerous dependent attributes, thus violating the assumption of conditional independence for a Naive Bayes classifier. Conversely, it is also possible that the 'mushroom' and 'hypothyroid' datasets have attributes that are less strongly correlated than those of the other datasets.

```
Using holdout, averaged over 10 runs
2018S1-proj1_data/breast-cancer-dos.csv | Avg. Measure: 0.7441493476497205
2018S1-proj1_data/car-dos.csv | Avg. Measure: 0.8604608212498983
2018S1-proj1_data/hypothyroid-dos.csv | Avg. Measure: 0.9521428931492114
2018S1-proj1_data/mushroom-dos.csv | Avg. Measure: 0.9922275854371312
```

Class distributions of each dataset.

```
data: 2018S1-proj1_data/breast-cancer-dos.csv
no-recurrence-events    201
recurrence-events        85

data: 2018S1-proj1_data/car-dos.csv
unacc    1210
acc       384
good       69
vgood      65

data: 2018S1-proj1_data/hypothyroid-dos.csv
negative       3012
hypothyroid     151

data: 2018S1-proj1_data/mushroom-dos.csv
e    4208
p    3916
```

Question 3

Evaluating the model on the same data that we use to train the model is considered to be a major mistake in Machine Learning. Implement a hold–out (hint: check out numpy.shuffle()) or cross–validation evaluation strategy. How does your estimate of Accuracy change, compared to testing on the training data? Explain why. (The result might surprise you!)

It should first be noted that the inherent randomness of holdout (selecting training and test instances randomly) will consequently produce varying accuracy on each run. The holdout method was run 10 times and the average accuracy was calculated. Implementing holdout generally results in a slight reduction in accuracy when compared to testing on the training data. The random partitioning method is likely to produce test data that is not representative of the training data especially with datasets with an uneven class distribution. (mentioned in Question 2)

Datasets with an uneven class distribution and/or fewer instances tend to experience a reduction in accuracy due to insufficient variability. The amount of data available for training is further reduced with the holdout method. Depending on the chosen ratio holdout may result in an accurate model, but the test data is still unlikely to be representative of all data (week03a.pdf). For example the 'breast cancer' dataset contains only 286 instances, randomly selecting 57 instances as test data (20%) provides no guarantee that the distribution of the test data is similar to that of the training data; hence inaccurately representing any unseen data. On the other hand training on the test data, implies the test data is most definitely representative of the training data.

Furthermore the reduction in accuracy is expected if one considers that testing on the training data can be likened to "telling the classifier what the correct answers are, and then asking whether it can come up with the correct answers". Once the correct answers are taken away we can expect the accuracy of the classifier to decrease.

```
Using holdout, averaged over 10 runs
2018S1-proj1_data/breast-cancer-dos.csv | Avg. Measure: 0.7414519588356774
2018S1-proj1_data/car-dos.csv | Avg. Measure: 0.8561387053264783
2018S1-proj1_data/hypothyroid-dos.csv | Avg. Measure: 0.9489565375703217
2018S1-proj1_data/mushroom-dos.csv | Avg. Measure: 0.9928177353617211

Training on test data
2018S1-proj1_data/breast-cancer-dos.csv | Avg. Measure: 0.7552447552447551
2018S1-proj1_data/car-dos.csv | Avg. Measure: 0.8738425925925926
2018S1-proj1_data/hypothyroid-dos.csv | Avg. Measure: 0.9522605121719886
2018S1-proj1_data/mushroom-dos.csv | Avg. Measure: 0.991999015263417
```

Question 6

Rather than evaluating the unsupervised NB classifier by assigning a class deterministically, instead calculate how far away the probabilistic estimate of the true class is from 1 (where we would be certain of the correct class), and take the average over the instances. Does this performance estimate change, as we alter the number of iterations in the method? Explain why.

Due to the nature of an unsupervised NB classifier in which classes may be 'swapped', an assumption must first be made of how to define the 'probabilistic estimate of the true class'. It would be inaccurate to simply define this as the probability of the true class after the iterations, as the probabilities may in fact be 'swapped'. Hence we assume that the class with highest probability is representative of the true class.

Under this assumption, in any dataset with no more than 2 classes we can observe a reduction in delta (how far away the probabilistic estimate of the true class is from 1) as the number of iterations increases; i.e. an inverse relationship exists between delta and the number of iterations. The observed increase in performance (lower delta) can be attributed to the convergence of the probabilistic class estimates as the number of iterations increase. This process maximises the likelihood through repeated estimations of the posterior probabilities, until convergence. (Further discussed in Question 7)

On the other hand the delta value for the dataset 'car' which contains 4 classes is consistently observed around 0.73. This can most likely be explained by the more complex decision boundaries (and class 'swapping') for a 4-class dataset, the uneven class distribution, and additionally the assumption above may not hold in this case. The classifier is only provided with the fact that there are 4-classes and hence its predictions converge to the uniform distribution because the instances from different class are likely to overlap one another.

```
unsupervised delta testing 5 iterations
2018S1-proj1_data/breast-cancer-dos.csv | Delta Avg.: 0.14704164732757008
2018S1-proj1_data/car-dos.csv | Delta Avg.: 0.7350380661994881
2018S1-proj1_data/hypothyroid-dos.csv | Delta Avg.: 0.05463283416580999
2018S1-proj1_data/mushroom-dos.csv | Delta Avg.: 0.0012226083886554336

unsupervised delta testing 10 iterations
2018S1-proj1_data/breast-cancer-dos.csv | Delta Avg.: 0.04936186522295407
2018S1-proj1_data/car-dos.csv | Delta Avg.: 0.7334206524625354
2018S1-proj1_data/hypothyroid-dos.csv | Delta Avg.: 0.0012487003184797482
2018S1-proj1_data/mushroom-dos.csv | Delta Avg.: 0.00036122349489423777

unsupervised delta testing 15 iterations
2018S1-proj1_data/breast-cancer-dos.csv | Delta Avg.: 0.034543243039314116
2018S1-proj1_data/car-dos.csv | Delta Avg.: 0.7339012545186191
2018S1-proj1_data/hypothyroid-dos.csv | Delta Avg.: 4.6463412735792215e-14
2018S1-proj1_data/mushroom-dos.csv | Delta Avg.: 0.0
```

Question 7

Explore what causes the unsupervised NB classifier to converge: what proportion of instances change their prediction from the random assignment, to the first iteration? From the first to the second? What is the latest iteration where you observe a prediction change? Make some conjecture(s) as to what is occurring here.

By printing out the confusion matrix for every iteration we observe that the proportion of instances changing prediction decreases as the number of iterations increase. This follows from question 6 as the delta performance measure also decreases as the number of iterations increase. Both observations assist in illustrating the convergence of the unsupervised NB classifier. Below is a summary of our results for each dataset:

Dataset	Proportion change (initial to first)	Proportion change (first to second)	Latest iteration of prediction change
breast-cancer	$34/286 = 0.119$	$24/286 = 0.084$	9
car	$520/1728 = 0.301$	$2/1728 = 0.001$	2
hypothyroid	$302/3163 = 0.095$	$150/3163 = 0.047$	9
mushroom	$4192/8124 = 0.516$	$784/8124 = 0.097$	9

For most of the datasets the latest iteration in which we observe a prediction change is the 9th iteration. As with every other question 'car' is once again an outlier which can mostly be explained by its uneven class distribution, and since it has 4 classes versus 2. The minimal convergence of 'car' is also explored in Question 6 where it's delta remains relatively constant. It is worth noting that with 'car' the predictions actually continue changing slightly, after around 6 iterations.

Through each iteration the classifier refines the prediction as it leans more and more towards a particular class; in each subsequent iteration this 'leaning' is represented by an increase in the corresponding class probability. Towards the end of the convergence process, class probabilities that are comparatively large will continue increasing while the smaller probabilities decrease proportionately. For example the gap between $P(instance|class1) = 0.99$ and $P(instance|class2) = 0.01$ will keep increasing. However the confusion matrix is unlikely to change from the 'latest iteration' where a prediction change is observed, since the difference between the two values is sufficiently large. This behaviour is similar to that of the Expectation-maximization algorithm.[1]

[1] http://cs229.stanford.edu/notes/cs229-notes8.pdf

Example depicting the change in class distributions between the initial and final iteration.

```
initial class distributions (random)
   0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17  hypothyroid  negative
0  M  f  f  f  f  f  f  f  f  f  f  f  y  y  y  y  y  n     0.123867  0.876133
1  F  t  f  f  f  f  f  f  f  f  f  f  y  y  y  y  y  n     0.533972  0.466028
2  M  f  f  f  f  f  f  f  f  f  f  f  y  y  y  y  y  n     0.435870  0.564130
3  F  f  f  f  f  f  f  f  f  f  f  f  y  y  y  y  y  n     0.395154  0.604846
4  M  f  f  f  f  f  f  f  f  f  f  f  y  y  y  y  y  n     0.785055  0.214945

final iteration (nine)
   0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17   hypothyroid  negative
0  M  f  f  f  f  f  f  f  f  f  f  f  y  y  y  y  y  n  3.338978e-13  1.0
1  F  t  f  f  f  f  f  f  f  f  f  f  y  y  y  y  y  n  3.683326e-13  1.0
2  M  f  f  f  f  f  f  f  f  f  f  f  y  y  y  y  y  n  3.338978e-13  1.0
3  F  f  f  f  f  f  f  f  f  f  f  f  y  y  y  y  y  n  8.439631e-13  1.0
4  M  f  f  f  f  f  f  f  f  f  f  f  y  y  y  y  y  n  3.338978e-13  1.0
```