Machine Learning A1 Write-up – James Nakos 1462474

Note: Throughout the report I refer to Class 0 as Non-Malicious and Class 1 as Malicious instead of 1 & 2. (Due to compatibility with code outputs)

Part 1:

* 1. Prior Probabilities:

The model has learned that the prior probabilities (assuming we know nothing about an instance) of it being in class 0 or 1 are:

* P(c = 0) = 0.8
* P(c=1) = 0.2

These are the proportions of messages of each class. Indicating that most of the training set is comprised of non-malicious messages, this may bias the model towards non-malicious predictions.

* 1. Most probable words:

|  |  |  |  |
| --- | --- | --- | --- |
| Class 0 Words | Class 0 Probabilities | Class 1 Words | Class 1 Probabilities |
| . | 0.079304 | . | 0.056522 |
| , | 0.026024 | ! | 0.024348 |
| ? | 0.025576 | , | 0.023478 |
| u | 0.018916 | call | 0.020543 |
| ... | 0.018749 | £ | 0.013913 |
| ! | 0.017182 | free | 0.010543 |
| .. | 0.014943 | / | 0.00913 |
| ; | 0.013152 | 2 | 0.008804 |
| & | 0.013096 | & | 0.008696 |
| go | 0.011137 | ? | 0.008478 |

The most probable words for both classes are mainly punctuation and short words like “u” and “call”. The punctuation mostly overlaps and are therefore not suggestive of class which is something I expected. Aside from the punctuation which wont heavily affect our model due to the nature of Naïve Bayes (considers the predictive power of words), the most frequent words in the Malicious class are “call” & “free”. These are words commonly associated with malicious offers that offer free things and are facilitated over call; hence these may prove to be informative to the model.

* 1. Most predictive words

|  |  |  |  |
| --- | --- | --- | --- |
| Most Predictive Words (Non-Malicious) | Probability Ratios (Non-Malicious) | Most Predictive Words (Malicious) | Probability Ratios (Malicious)  R1/R0 |
| ; | 60.49922 | prize | 99.05086957 |
| ... | 57.49571 | tone | 64.09173913 |
| gt | 54.06313 | £ | 49.71965217 |
| lt | 53.54824 | select | 46.61217391 |
| :) | 47.88449 | claim | 45.96478261 |
| ü | 31.92299 | paytm | 36.90130435 |
| lor | 28.83367 | code | 34.95913043 |
| hope | 24.71457 | award | 32.04586957 |
| ok | 24.71457 | won | 31.07478261 |
| d | 21.11036 | 18 | 29.1326087 |

Our most predictive words appear much more indicative of the message’s intent than our frequent words which is to be expected. The malicious word “prize” and “£” clearly indicate intent of monetary exploitation and given high probability ratios as opposed to “:)” or “ok” which frequent non malicious messages and have high inverse ratios. The presence of these high ratio (predictive) ‘words’, will aid the model to classify messages by giving it information which helps the Bayes model to create probabilities for each class. Therefore, since we have a wide range of probability ratios, it should be possible to distinguish the classes

Part 2:

2.1:

Accuracy: 0.975

Confusion Matrix

|  |  |  |
| --- | --- | --- |
|  | Class 0 Predicted | Class 1 Predicted |
| Class 0 True | 785 | 15 |
| Class 1 True | 10 | 190 |

Precision: 0.93

Recall: 0.95

These metrics indicate that the model performed well, classifying 97.5% of instances correctly and this was achieved without relying on the majority class label to classify, (we only had 10 false negatives). Our recall was hence large (0.95), and we also had a small number of False Positives (15). Therefore, the model generalises well.

2.2:

|  |  |
| --- | --- |
| Proportion of unique words in vocab | 0.9427178549664839 |
| Proportion of unique words not in vocab | 0.05728214503351615 |
| Proportion of test words in vocab | 0.9836797957695113 |
| Proportion of test words not in vocab | 0.016320204230488744 |

We observed no test instances without a word in our training vocabulary, so we could classify every item. To see how often our model saw out of vocab words w calculated the proportion of all test data words not in vocab ~0.016. Indicating that most instances should be easily classified.

2.3

A,

|  |  |
| --- | --- |
| Confidence Ratios (Class 0) | Text Instances (Class 0) |
| 9.1349944516292e+37 | time : rs. transaction number & & & & & & & & & & ; ; ; ; ; ; ; ; ; ; lt lt lt lt lt # # # gt gt gt gt gt credit account reference decimal |
| 2.6903818585609323e+29 | ? ? ? ? .. .. u u u u , , ... ... ... ... say person yes ! f : hello hello hello o o wen knw knw girl girl mean @ " " " " t name name g g n d d d d d d lift bt real dat h girlfrnd girlfrnd moral |
| 3.1829245411853877e+25 | . every & & & & & & ; ; ; ; ; ; lt lt lt # # # gt gt gt big hr |

B,

|  |  |
| --- | --- |
| Confidence Ratios (Class 1) | Text Instances (Class 1) |
| 1.3538895972837586e+20 | . 4 + call Â£ - \* holiday & urgent 18 t landline 150ppm cash cs await collection po box sae complimentary 10,000 ibiza |
| 1.2870905388142676e+20 | . 3 4 + ! call : Â£ offer \* holiday & urgent 18 t landline 150ppm cash cs await collection po box sae tenerife 10,000 |
| 1.1491239652937078e+20 | . . . , please order text call / : customer tone number [ [ service mobile ] ] colour colour thanks ringtone reference charge 4.50 arrive = red x49 09065989182 |

A,B Discussion

The high confidence class 0 predictions do not appear to be “normal” text messages, they are unsurprisingly filled with punctuation which were also our most probable words. The model is therefore highly confident that instances like these are non-malicious, this could potentially be exploitable (an area to improve on). On the other hand, messages with high class 1 confidence are as expected, incorporating monetary language like “cash” which are commonly associated with malicious intent. This outlines that class 1 is likely easier to identify due to its more distinctive vocabulary while non malicious messages were more difficult.

C,

|  |  |
| --- | --- |
| R Values Closest to 1 | Text Instances |
| 1.0170981352199624 | . call dear |
| 1.0441124738285466 | . reply glad |
| 0.9297658433847452 | . . tell return re order |

The model is rightfully uncertain about these messages as they are short and hence do not give adequate information to the model to help classify them. We also have the presence of words that can be associated with both categories like “call”, leaving the model uncertain.

Part 3 (Option 2):

To improve my model from Q1, I chose to employ **Active Learning** which involved adding 200 records from the unlabelled dataset to the training data to enhance & diversify the training set. At first, I selected the 200 records at random and added them to the training data, then retrained the model. I firstly evaluated this on a random validation sample (20%) of the data.

Here were our results:

Accuracy: 0.965

Precision: 0.90

Recall: 0.94

Each of these were noticeable lower than how our original model performed. I decided that this was likely due to the fact that the random sampling does not ensure an equal distribution of classes in both the training and validation set. This could lead to the training set overfitting to the majority class and hence a loss of performance.

To combat this, I reevaluated this on a stratified validation set using the sklearn train\_test\_split function, this kept the proportions of the classes in both samples close to the same as the full dataset.

Results:

Accuracy: 0.975

Confusion Matrix:

|  |  |  |
| --- | --- | --- |
|  | Class 0 Predicted | Class 1 Predicted |
| Class 0 True | 315 | 5 |
| Class 1 True | 5 | 75 |

Precision: 0.94

Recall: 0.94

These results are extremely close to that of our first Naïve Bayes model and hence I needed to reselect 200 instances that were more informative to the model to see an improvement.

To do this I utilised the original model to examine all the unlabelled instances and select the 200 it was most uncertain about classifying. This was done by calculating the ratio of the posterior probability of each class given an instance. These were hence instances that the original model was unconfident about labelling, so they should be more informative and help the model improve on similar inputs.

After retraining these were our validation evaluation results:

Accuracy: 0.9825

Confusion Matrix:

|  |  |  |
| --- | --- | --- |
|  | Class 0 Predicted | Class 1 Predicted |
| Class 0 True | 318 | 2 |
| Class 1 True | 5 | 75 |

Precision: 0.97

Recall: 0.94.

These results outline that selecting more informative instances had a positive effect and reduced the number of False Positives, subsequently increasing our precision.

To further improve our model, I attempted to utilise decision entropies rather than R values to select my 200 instances as they better capture uncertainty. However, the model still performed the same. This is likely attributed to the data being relatively easy to separate into classes and our model already being confident in predicting them, leaving most instances with low entropy and no room for entropy to improve our previous uncertainty measures.

I also attempted other improvements to no avail, such as adjusting the Laplace smoothing parameter alpha (which we left as 1), I tried smaller values to increase the model’s reliance on rare words, however the model only sli

PUT GRAPH

and the training validation split. However, these simply improved the model’s performance on

Lastly, I evaluated the model on our test data:

Accuracy: 0.977

Confusion Matrix:

|  |  |  |
| --- | --- | --- |
|  | Class 0 Predicted | Class 1 Predicted |
| Class 0 True | 785 | 15 |
| Class 1 True | 8 | 192 |

Precision: 0.93

Recall: 0.96

**In your write-up, explain your approach and give the reasoning behind your implementation choices. Show results on the training and/or validation set to support your discussion. Your response should be no more than 500 words.**

Part 4: