What to Turn in

• Your code : HW2.py

• a Readme ﬁle for compiling and executing your code : README.txt

• A detailed write up that reports the accuracy, precision, recall and F1 score obtained on the three test sets for the following algorithms and feature types:

– Multinomial Naive Bayes on the Bag of words model

|  |  |  |  |
| --- | --- | --- | --- |
| Multinomial Naive Bayes | Bag-of-words  Dataset 1 | Bag-of-words  Dataset 2 | Bag-of-words  Dataset 3 |
| Accuracy | 0.9476987 | 0.9342105 | 0.93738 |
| Precision | 0.9654179 | 0.9396825 | 0.90972 |
| Recall | 0.9626437 | 0.9641694 | 0.86184 |
| F1 - score | 0.9640288 | 0.9517685 | 0.88514 |

– Discrete Naive Bayes on the Bernoulli model

|  |  |  |  |
| --- | --- | --- | --- |
| Discrete Naive Bayes | Bernoulli Dataset 1 | Bernoulli Dataset 2 | Bernoulli Dataset 3 |
| Accuracy | 0.7782427 | 0.7368421 | 0.91713 |
| Precision | 0.7700893 | 0.7220903 | 1 |
| Recall | 0.9913793 | 0.990228 | 0.70395 |
| F1 - score | 0.8668342 | 0.8351648 | 0.82625 |

– Logistic Regression on both Bag of words and Bernoulli models

|  |  |  |  |
| --- | --- | --- | --- |
| Logistic Regression | Bernoulli Dataset 1 | Bernoulli Dataset 2 | Bernoulli Dataset 3 |
| Accuracy | 0.960251 | 0.9517544 | 0.979742 |
| Precision | 0.968661 | 0.970297 | 1 |
| Recall | 0.977011 | 0.9576547 | 0.927632 |
| F1 - score | 0.972818 | 0.9639344 | 0.962457 |
|  | L=0.1, n=0.01, iter=300 | L=1.0, n=0.01, iter=300 | L=0.01, n=0.01, iter=300 |

|  |  |  |  |
| --- | --- | --- | --- |
| Logistic Regression | Bag-of-words  Dataset 1 | Bag-of-words  Dataset 2 | Bag-of-words  Dataset 3 |
| Accuracy | 0.922594 | 0.9473684 | 0.959484 |
| Precision | 0.953353 | 0.9579288 | 0.882353 |
| Recall | 0.939655 | 0.9641694 | 0.986842 |
| F1 - score | 0.946454 | 0.961039 | 0.931677 |
|  | L=0.1, n=0.01, iter=300 | L=1.0, n=0.01, iter=300 | L=0.1, n=0.01, iter=300 |

– SGDClassiﬁer on both Bag of words and Bernoulli models

|  |  |  |  |
| --- | --- | --- | --- |
| SGD Classifier | Bernoulli Dataset 1 | Bernoulli Dataset 2 | Bernoulli Dataset 3 |
| Accuracy | 0.962343 | 0.949561 | 0.974217 |
| Precision | 0.976879 | 0.973333 | 0.985915 |
| Recall | 0.971264 | 0.95114 | 0.921053 |
| F1 - score | 0.974063 | 0.962109 | 0.952381 |
|  | alpha=0.001,  loss=’log’  max\_iter=300  penalty='l2'  tol=0.01 | alpha=0.01,  loss=’log’  max\_iter=300  penalty='l2'  tol=0.001 | alpha=0.001,  loss=log  max\_iter=300  penalty='l2'  tol=0.001 |

|  |  |  |  |
| --- | --- | --- | --- |
| SGD Classifier | Bag-of-words  Dataset 1 | Bag-of-words  Dataset 2 | Bag-of-words  Dataset 3 |
| Accuracy | 0.916318 | 0.899123 | 0.968692 |
| Precision | 0.958333 | 0.936455 | 0.941176 |
| Recall | 0.925287 | 0.912052 | 0.947368 |
| F1 - score | 0.94152 | 0.924092 | 0.944262 |
|  | alpha=0.0001,  loss='log'  max\_iter=300  penalty='l2'  tol=0.01 | alpha=0.1,  loss='log'  max\_iter=100  penalty='l2'  tol=0.1 | alpha=0.01,  loss=’log’  max\_iter=300  penalty='l2'  tol=0.001 |

Your report should also describe how you tuned the hyper-parameters for logistic regression and SGDClassiﬁer for each dataset (speciﬁcally values of λ used, hard limit on the number of iterations, etc.). The write up should be self contained in that we should be able to replicate your results.

For logistic regression:

(Tuning parameters present in “Lambda, Learning rate and Iteration Tuning.txt”)

For hyper parameter tuning I split the training dataset into 70% training and 30% validation examples.

First I tuned the learning rate (n) by starting with a value of 0.0001 and set the number of iterations (iter) to 100. After training on the training dataset I tested on the validation dataset and noted the accuracy and F1 score. Then I tripled the number of iterations and repeated the same process again noting the accuracy and F1 score. With each increase in the number of iterations the F1 score on the validation set also increased. Once the number of iterations crossed 1000 I increased the learning rate to 0.001 and reset the number of iterations to 100 and started the whole process again.

I repeated this procedure till the overfitting started and F1 score on validation set started decreasing. This happened at a learning rate of 0.01 and number of iterations at 300. Then I introduced the regularization parameter Lambda (L) and tested for multiple values of L from the set {0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3, 10} and found the value with the best F1 score.

• Answer the following questions:

1. Which data representation and algorithm combination yields the best performance (measured in terms of the accuracy, precision, recall and F1 score) and why?

According to above table containing values from all the algorithms on different data models, I can say that, in general, Logistic Regression gives the best performance in terms of accuracy, precision, recall and F1 score, especially on the Bernoulli data model.

I think this is because Logistic Regression doesn’t make any assumptions on conditional independence of features and it is possible to train them to a high degree of accuracy by maintaining a balance between overfitting and regularization. In the real world data is seldom conditionally independent so the basic assumption of Naive Bayes is falsified.

Bernoulli data model works better in this case because the classification between spam and ham depended more on the existence of certain words in the documents rather than their respective frequency in the particular document.

2. Does Multinomial Naive Bayes performs better (again performance is measured in terms of the accuracy, precision, recall and F1 score) than LR and SGDClassiﬁer on the Bag of words representation? Explain your yes/no answer.

According to the above tables Multinomial Naive Bayes only performs better than LR and SGDClassiﬁer if the size of the dataset is small. In all the other cases LR beats Multinomial Naive Bayes and SGDClassiﬁer with quite a margin. In general, LR works works better than all other algorithms. In the last dataset SGDClassiﬁer performs better than LR, but I think this is because the number of iterations of LR were fixed so it didn’t get sufficient iterations to converge properly and reach its maximum value.

I think this is because LR doesn’t make any assumptions on the conditional independence of the data whereas Multinomial Naive Bayes assumes conditional independence of the data. In the real world data is seldom conditionally independent and logistic regression is able to find the minute dependence between different features. So it is able to give a better performance on real data. Also it performs better when the size of the data becomes larger.

3. Does Discrete Naive Bayes performs better (again performance is measured in terms of the accuracy, precision, recall and F1 score) than LR and SGDClassiﬁer on the Bernoulli representation? Explain your yes/no answer.

According to the above tables, in all the cases LR beats Discrete Naive Bayes. The performance of LR and SGDClassiﬁer is almost similar on Bernoulli dataset. In general, LR works works better than all other algorithms. In the last dataset LR performs better than even SGDClassiﬁer. I think this is because LR considers the entire dataset on each iteration but SGDClassiﬁer is guided by a single example in each step.

I think this is because LR doesn’t make any assumptions on the conditional independence of the data whereas Discrete Naive Bayes assumes conditional independence of the data. In the real world data is rarely conditionally independent and logistic regression is able to find the relation between different features.

4. Does your LR implementation outperform the SGDClassiﬁer (again performance is measured in terms of the accuracy, precision, recall and F1 score) or is the diﬀerence in performance minor? Explain your yes/no answer.

In Bernoulli dataset LR outperforms SGDClassiﬁer in the last case whereas in multinomial dataset LR outperforms SGDClassiﬁer in the first two datasets. In all the cases where SGDClassiﬁer beats LR implementation, the difference is very minor. In general I think LR performs better than SGDClassiﬁer but in this cases due to a hard limit on the number of iterations LR was not able to converge fully.