Homework 4

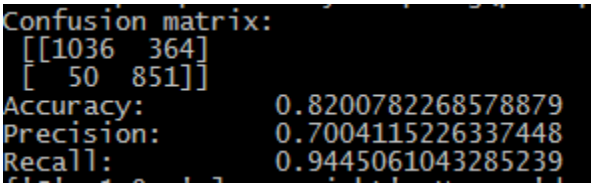
Submitted by Akanksha

Part 1-

Description:

In this experiment we have used spambase dataset which has rows having information about the spam mails(1 class) and non spam mails(0 class), we have calculated standard deviation, mean on the training data(50% of the data with 40% spam and 60 % nor spam) and then have tested 50% of data using the calculated standard deviation and mean of the training data, for testing we have calculated the probabilities and based on those probabilities we have classified the classes of the data(using logarithmic sums of probabilities) and then finally calculated accuracy ,recall ,precision and confusion matrix on the test data.

Results-



About Result-

The accuracy of the naïve bayes classifier is not that good when compared to the other classifiers we have done. The confusion matrix gives all the data which is correctly and incorrectly classified and precision and recall gives relevance and sensitivity values of the data.

How did Naïve Bayes do compared with your SVM from Homework 3?

Naïve Bayes does not do well on this problem as we can see that from the results we are obtaining, the accuracy is quite low as compared to svm on the same dataset. There is a huge gap of 10% between the accuracies of both the classifiers.

Do you think the attributes here are independent, as assumed by Naïve Bayes?

We have assumed that the attributes are independent but it can be the case that presence of one word in the spam mail means that the other word is present most of the times say 90% of times that can lead to some dependence (eg. Say spam mail is about some jackpot then there is high probability of having word money in it similarly for project,discussion and meeting). So it may not be that independent depending on the words in the dataset.

Does Naïve Bayes do well on this problem in spite of the independence assumption?

No, Naïve Bayes does not do well on this problem as we can see that from the results we are obtaining, the accuracy is quite low as compared to svm on the same dataset.

Speculate on other reasons Naïve Bayes might do well or poorly on this problem

Naïve Bayes may be will perform good if we consider only the important features as we have done in case of svm in the last Homework. We can use feature selection methods(like tf-idf) and can observe what attributes are statistically meaningful and have a dense presence over the entire data and then can use only those to classify which would probably increase the performance.

Part 2-

Description-

In ths experiment we have imported the library with available logistic regression package, using that we have trained our model on the training data(50% of the data with 40% spam and 60 % nor spam) using the fit function and then have tested on remaining 50% of data and calculated accuracy ,recall ,precision and confusion matrix.

Describe what library you used and what parameter values you used in running logistic regression.

I have used scikit_learn in Python library(sklearn.linear model's Logistic Regression), This class implements regularized logistic regression using the 'liblinear' and used the default parameters which are as below-

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

penalty : str, 'l1' or 'l2', default: 'l2'Used to specify the norm used in the penalization. **dual** : bool, default: False,Dual or primal formulation. Dual formulation is only implemented for l2 penalty with liblinear solver. Prefer dual=False when n_samples > n_features.**C** : float, default: 1.0,Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization.**fit_intercept** : bool, default: True,Specifies if a constant (a.k.a. bias or intercept) should be added to the decision function.**intercept_scaling** : float, default 1.Useful only when the solver 'liblinear' is used and self.fit_intercept is set to True**. class_weight** : dict or 'balanced', default: NoneWeights.**max_iter** : int, default: 100,Useful only for the newton-cg, sag and lbfgs solvers. Maximum number of iterations taken for the solvers to converge.**random_state** : int seed, RandomState instance, default: None,The seed of the pseudo random number generator to use when shuffling the data. Used only in solvers 'sag' and 'liblinear'.solver : {'newton-cg', 'lbfgs', 'liblinear', 'sag'}, default: 'liblinear'Algorithm to use in the optimization problem.For small datasets, 'liblinear' is a good choice, whereas 'sag' is,For multiclass problems, only 'newton-cg', 'sag' and 'lbfgs' handle,**tol** : float, default: 1e-4.Tolerance for stopping criteria.**multi_class** : str, {'ovr', 'multinomial'}, default: 'ovr',Multiclass option can be either 'ovr' or 'multinomial'. If the option chosen is 'ovr', then a binary problem is fit for each label. Else the loss minimised is the multinomial loss fit across the entire probability distribution. Works only for the 'newton-cg', 'sag' and 'lbfgs' solver**.verbose** : int, default: 0,For the liblinear and lbfgs solvers set verbose to any positive number for verbosity.**warm_start** : bool, default: False.When set to True, reuse the solution of the previous call to fit as initialization, otherwise, just erase the previous solution. Useless for liblinear solver.**n_jobs** : int, default: 1,Number of CPU cores used during the cross-validation loop. If given a value of -1, all cores are used.

Give the accuracy, precision, and recall of your learned model on the test set, as well as a confusion matrix.

```
Accuracy:         0.922207735767
Precision:        0.924705882353
Recall:           0.872364039956
Confusion Matrix:
 [[1336   64]
 [ 115  786]]
```

Write a few sentences comparing the results of logistic regression to those you obtained from Naïve Bayes and from your SVM from Homework 3.

As seen from the results , the logistic regression and SVM performs almost comparable(with accuracies 92% approx. in both the cases, SVM is a hard classifier but LR is a probabilistic one and LR can break in case of linearly separable problems due to numerical calculations issues) ,where as Naive Bayes does not perform that well as the Logistic regression as the accuracy is much lower than the Logistic regression, There is almost 10% difference between both (Both Naive Bayes and Logistic regression are linear classifiers, Logistic Regression makes a prediction for the probability using a direct functional form where as Naive Bayes figures out how the data was generated given the results.). Also Naïve Bayes performed poorer than the SVM of the Homework 3 with again an accuracy difference of 10%.