Student Number: 215816

1. Introduction

Binary Classification is essential for everyday tasks such as classifying e-mails into spam and not spam. However, this task can become difficult when handling data with high dimensionality and missing features. In this report we discuss some methods we used to reduce the dimensionality and fill the missing features, as well as our own prediction model, with over 70% accuracy.

2. Methods

2.1. Filling in missing values

There are many ways to deal with missing data: using mean, median or mode to fill in the missing values, K nearest neighbours, logistic regression etc. As a first step of pre-processing our data, we decided to exclude all features which had more than 20% of the variables missing. We believed that if a feature contained many missing values, it might have been because it is not an important feature to consider [1], thus we removed them.

For the remaining missing data, we decided to implement k-nearest neighbours, where the missing value would pick the most common value within the 10 nearest neighbours. We then concatenated our training1 and training2 data.

2.2. Reducing dimensionality

Before we reduced the high dimensionality of our training set, we used standardization and variance scaling, in order to acquire 0 mean and unit variance for each variable. We then normalised our data in order for our next, Principal Component Analysis (PCA), to work correctly.

In PCA we let the algorithm from sklearn [3] decide to choose the number of components for us, we specified it to preserve 95% of the variability in the data. We succesfully reduced the features from 4608 to 1479.

2.3. Classification Approach & Model Selection

After thoroughly visualising our data, we decided to solve this task using a Support Vector Machine (SVM) [2]. We chose this model, because it handles well with high dimensional data. The SVM constructs a hyperplane in a high dimensional space in order to separate the data.

To train the SVM, we did K-fold validation to see with which hyper parameters out model would give the best accuracy when testing on the validation set. From sklearn [3] we imported the SVM, and we changed the hyper parameters for: the regularization parameter, kernel and kernel coefficient [4]. For the model weights, we used the confidence our labels had.

3. Results & Discussions

We discovered that our model performs best when given 1.0 regularization parameter,rbf "kernel" and 0.0 coefficient, which are the default values for when we create the SVM, as we can see in Figure 1.

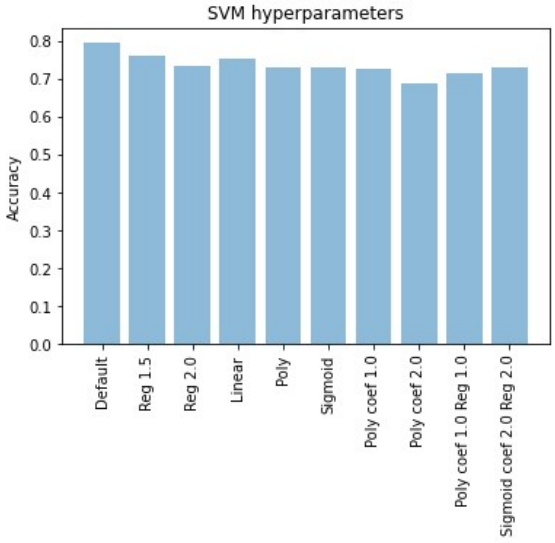

Figure 1. Bar plot for highest accuracy for different hyper parameters in Support Vector machines, modifying the parameters of regularization parameter, kernel and kernel coefficient.

We also changed the hyper parameter of gamma [4], to find the "sweet spot" of our model, in other words observe when the model is neither under fitting or over fitting, see Figure 2.

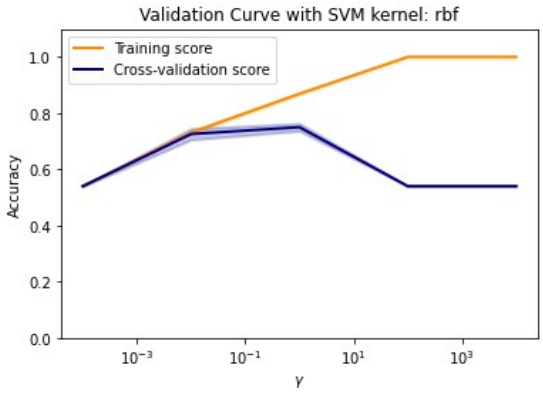

Figure 2. Validation curve for Support Vector Machine with 1.0 regularization parameter,rbf "kernel" and 0.0 coefficient

We also discovered that there is no difference between

the accuracy when training the model with all features of the training data and removed features from the training data, we also discovered the same results when adding and not adding confidence weights to the model, see Figure 3

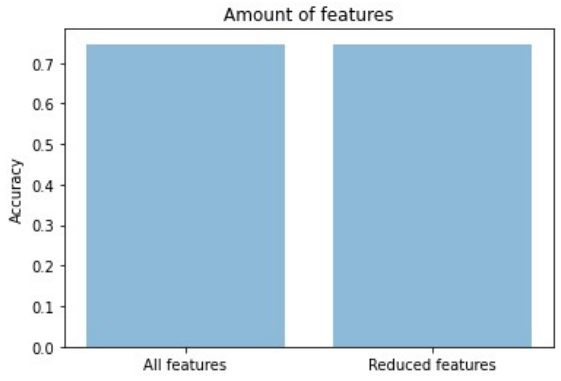

Figure 3. Performance of our model with all and reduced features; and with and without confidence weights. We can clearly observe that our model gives the same results

Here we provide the confusion matrix for our final classifier, Figure 4

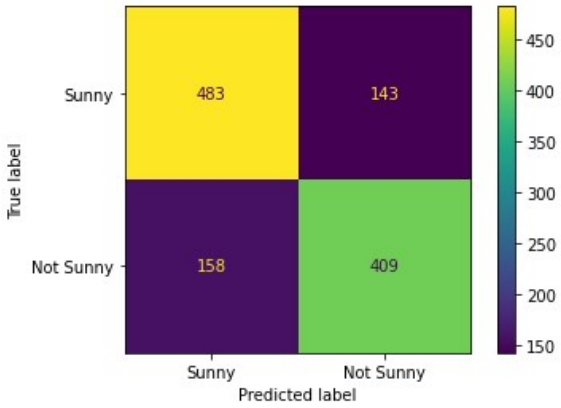

Figure 4. Confusion matrix for the final classifier

In terms of improving the performance of the model, we believe that we could have removed more features considering we had information about the features we were given. Moreover, we suggest trying different hyper parameters for the Support Vector Machine (Degree of polynomial function, shrinking heuristic, tolerance etc.) [4].

We also believe we could have performed more plots to better visualise the evaluation of our model, for example visualising the ground truth function vs our predicted function for the task. For future work, we recommend visualising the data in a plot instead of inside a table, to better visualise the data we are given.

References

[1] G. E. Batista and M. C. Monard. An analysis of four missing data treatment methods for supervised learning. *Applied artificial intelligence*, 17(5-6):519–533, 2003. 1

[2] W. S. Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006. 1

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 1

[4] Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E. Scikit-learn: Machine learning in Python support vector machine, 2011. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html, Last accessed on 2021-05-19. 1, 2

4. Appendix

Link to colab:
https://colab.research.google.com/drive/1iAFWAek5mbkuXGg9DGCGhueUVWCJ4ShN?usp=sharing

I have also appended my code as part of the submission