**Classification Technical Report**

**Ahmed Abdelatty**

**Ahmed.Abdelatty@utdallas.edu**

**1 Summary**

In this report I apply classification techniques to two datasets: 1st is the set of tweets about US airline companies and their sentiment, 2nd is the Turkish sign language digits. For the sentiment analysis task, I apply over sampling methods (SMOTE, ADASYN) to adjust the class distribution, because of the imbalanced nature of the dataset. Furthermore, I apply different feature extraction techniques and use PCA for dimensionality reduction. For the Turkish sign language dataset, I apply many classification techniques including convolution neural nets, which give a 98.06% test accuracy.

**2 Background**

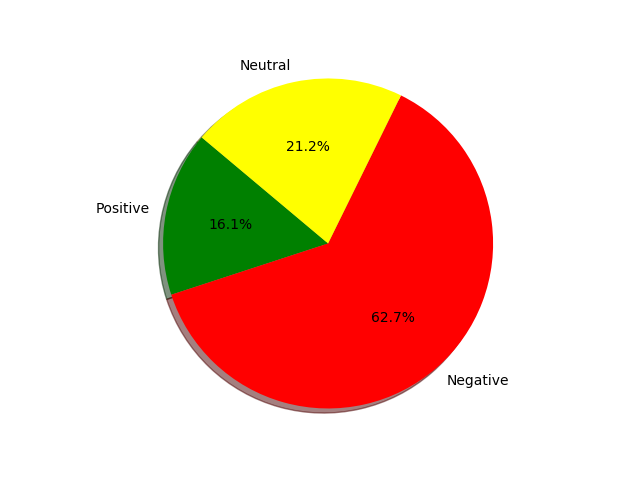
**2.1 Over Sampling**

Oversampling [1] is a bias correction technique that is used to adjust the class distribution of a given classification dataset. For example, assume that we toss a fair coin for 100 times out of which we got 70 heads. To adjust our dataset to represent the fact that tails must come up to 50% of the times we can use one of the following over sampling techniques:

1. Simple or Random oversampling: sample with replacement from the misrepresented class until we have a balanced dataset.
2. Synthetic Minority Oversampling Technique (SMOTE) [2]: instead of repeating already exiting data points, SMOTE generate new samples by sampling a datapoint from the misrepresented class then find it’s k nearest neighbor from the same class and generate their weighted average as the new datapoint.
3. The adaptive synthetic sampling approach (ADASYN) [3]: same as SMOTE, however, they are different on how they select the weights when finding the average.

**3 Task 1**

**3.1 Data description:** The dataset contains tweets about six US airline companies, each of which is labelled as: positive, negative, and neutral. The total number of tweets is 14,640 and label distribution is given as follows:

****

* 1. **Data preprocessing**

Data preprocessing was splitted into 4 levels:

**Level 1**: cleaning tweets by removing stop words, converting all words into lower case, stemming all words, and optionally removing (special characters, mentions, hashtags, and links).

**Level 2**: extract features either using term frequency (tf), tf-idf or word2vec.

**Level 3**: use Principal component analysis for dimensionality reduction.

**Level 4**: use over sampling to adjust the class distribution, because of the imbalanced nature of the dataset.

* 1. **Data Splitting**

I used two approaches classifying the dataset

**Approach 1:** As said before the dataset contains tweets about six US airline companies. In this approach, I divided the data set into 6 small datasets, every one of which contains tweets about one of the airlines. Then I trained 6 different models: one for each dataset. This approach was proposed in [4], however, I didn’t get same results as theirs.

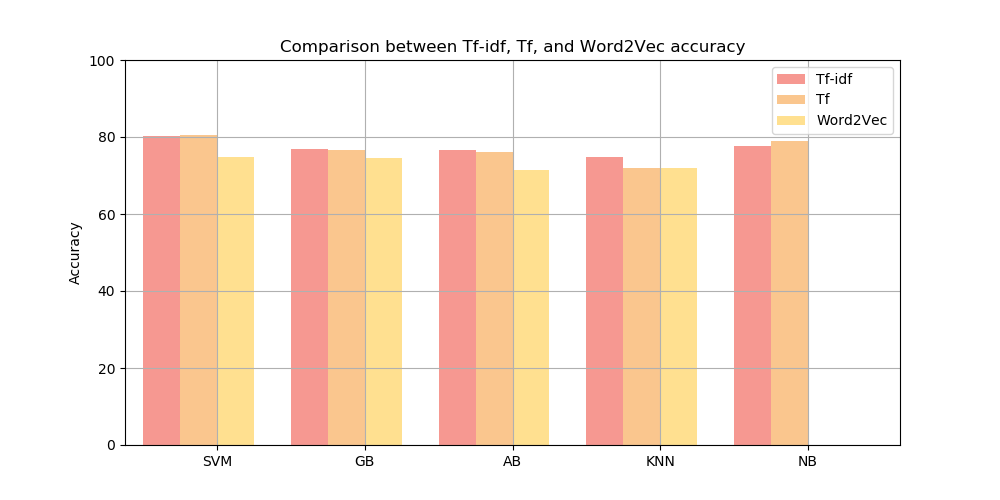
**Approach 2:** Use the data as its.

* 1. **Experiments and Results**

The dataset was splitted into 80% training and 20% testing. I used multiple classification techniques (Linear SVM, Naïve Bayes (NB), KNN, AdaBoost (AB), and GradientBoosting (GB)), also I used the pretrained TextBlob sentiment classifier but got 54% accuracy. To obtain the highest possible accuracy I did a lot of parameter tuning including:

1. Selecting which levels of preprocessing to apply.
2. Tuning the parameters of each classifier: for KNN K is selected between {1,3,5,7,10,15}, for NB Laplace smoothing is selected between {.5,1,1.5,2,2.5,3,5,7}, for Linear SVM the slack constant C is selected between {1,.1,.2,2,5,10,20}, for both GradientBoosting and AdaBoost number of estimators is selected between {50,100,200}.

Best performance for SVM and Naïve Bayes was obtained by applying tf only (Level 1,2 preprocessing) on the entire dataset. On the other hand, the best performance for GradientBoosting, KNN, and AdaBoost was obtained by applying tf-idf only (Level 1,2 preprocessing) on the entire dataset. Figure 1 provides a comparison between Tf-idf, Tf, and Word2Vec accuracy without using PCA or Over Sampling.



**Figure 1**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **RandomOverSampler** | | | **SMOTE** | | | **ADASYN** | | | **None** | | |
|  | **Tf** | **Tf-idf** | **W2V** | **Tf** | **Tf-idf** | **W2V** | **Tf** | **Tf-idf** | **W2V** | **Tf** | **Tf-idf** | **W2V** |
| **SVM** | 78.14 | 77.16 | 69.96 | 78.79 | 79.2 | 70.35 | 78.04 | 77.97 | 69.02 | **80.6** | 80.26 | 74.72 |
| **GB** | 73.43 | 72.82 | 69.65 | 73.5 | 75.58 | 67.79 | 75.44 | 75.00 | 70.63 | 76.74 | **76.81** | 74.52 |
| **AB** | 71.35 | 68.66 | 56.33 | 75.34 | 71.82 | 63.49 | 72.33 | 73.33 | 65.78 | 76.16 | **76.63** | 71.45 |
| **KNN** | 63.8 | 62.68 | 6466 | 41.97 | 38.18 | 48.80 | 39.03 | 37.23 | 66.19 | 72.06 | **74.76** | 71.99 |
| **NB** | 78.42 | 77.23 | ––––– | 79.51 | 78.72 | ––––– | 78.03 | 78.24 | ––––– | **79.06** | 77.56 | ––––– |

**Table 1: Accuracy of different classification techniques without using PCA**

Table 1 provides the accuracy for all the used classification techniques over the entire dataset, along with the used over sampling technique if any, and the feature extractor used. On the other hand, using PCA or splitting the data using airlines made the accuracy worse. Table 2 provides Classification Report for the best classifier in Table 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **precision** | **recall** | **f1-score** | **support** |
| **negative** | .85 | .91 | .88 | 1889 |
| **neutral** | .66 | .53 | .59 | 580 |
| **positive** | .77 | .70 | .73 | 459 |
| **avg / total** | .80 | .81 | .80 | 2928 |

**Table 2: Classification Report for the best classifier in Table 1**

**4 Task 2**

**4.1 Data description:** The dataset contains images of the Turkish sign language digits. The dataset has 10 class for digits from 0 to 9. The total number of images is 2,062 and labels are almost evenly distributed.

* 1. **Data preprocessing**

**Level 1**

All images have been converted into gray scale, then resized to 64X64 instead of 100X100. For the training of the Convolution Neural Network (CNN) purpose labels have been converted to one hot encoding. For the purpose of training other classifiers (SVM, etc) every image has been flattened into 4,096 feature vector.

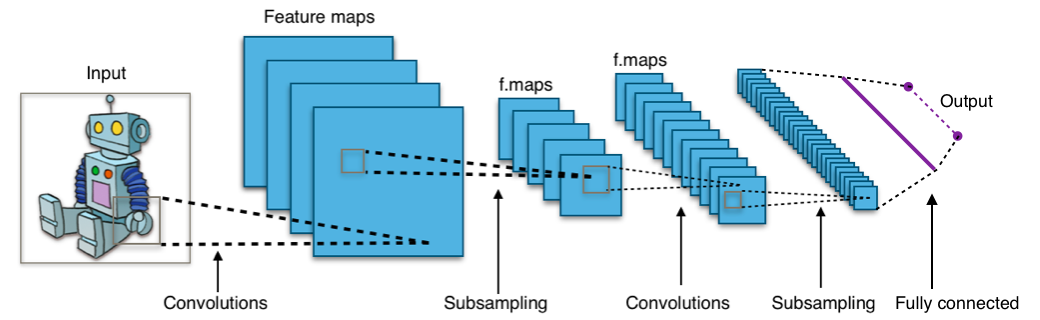
**Level 2**

Due to the high dimensionality of the data, I applied PCA for all classifiers except the CNN.

* 1. **Experiments and Results**

The dataset was splitted into 20% testing and the remaining 80% was splitted into 80% training and 20% validation. I used multiple classification techniques (Linear SVM, CNN, KNN, AB, and GB).

I followed the following structure designing our CNN



[Source](https://commons.wikimedia.org/wiki/File:Typical_cnn.png)

Where

1. Convolution layers contain 3X3 kernels.
2. Subsampling is done by 2X2 max pooling layers.
3. The fully connected layer is to flatten the all the learned features and combine them.
4. Then, flattened features are passed to a softmax to predict the label of the input data.

The final structure that obtained the highest accuracy is as follows:

1. Input layer 28X28X1.
2. Convolution layer 32- 3X3 kernel.
3. 2X2 max pooling.
4. Convolution layer 64- 3X3 kernel.
5. 2X2 max pooling.
6. Convolution layer 128- 3X3 kernel.
7. 2X2 max pooling.
8. Convolution layer 256- 3X3 kernel.
9. 2X2 max pooling.
10. Flatten layer.
11. Dense layer 256 units.
12. Softmax 10 units.

CNN training details:

The network is trained using the Adam optimizer ([A Method for Stochastic Optimization](https://arxiv.org/abs/1412.6980)) [5] with categorical cross entropy as the loss function. Training was early stopped after a specific number of iterations to prevent overfitting.

To obtain the highest possible accuracy I did a lot of parameter tuning including:

1. Selecting which levels of preprocessing to apply.
2. Tuning the parameters of each classifier: for KNN K is selected between {1,3,5,7,10,15}, for Linear SVM the slack constant C is selected between {1,.1,.2,2,5,10,20}, for both GradientBoosting and AdaBoost number of estimators is selected between {50,100,200}.
3. To prevent CNN overfitting, I used random Dropout with a varying percentage between layers also used early stopping by tuning the number of epochs.

Table 3 provides the accuracy of 8 runs of training CNN on trainining, validation, and testing data: the network was trained for 65 epochs with different drop out percentages. As shown the best accuracy obtained on the testing data was 98.06%. On the other hand, other models performed not as good as CNN: the best accuracy obtained on the testing data was 69.47%.

|  |  |  |
| --- | --- | --- |
| **Train Accuracy %** | **Validation Accuracy %** | **Test Accuracy %** |
| 99.32 | 97.58 | 96.80 |
| 99.24 | 94.85 | 95.15 |
| 99.24 | 96.76 | 97.09 |
| 99.47 | 97.58 | 97.85 |
| 99.24 | 98.18 | 98.06 |
| 99.85 | 98.48 | 97.04 |
| **99.92** | **98.48** | **98.06** |
| 99.70 | 97.27 | 97.82 |

**Table 3: CNN accuracy on train, validation, and test data after 65 epochs for 8 runs**

1. **Conclusion**

In this report, I applied multiple classification techniques including SVM, Convolution Neural Networks, Naïve Bayes, K Nearest Neighbors, GradientBoosting, and AdaBoost. Moreover, many text feature extractors have been used: Tf, Tf-idf, and Word2Vec. I used both the classification techniques to classify two datasets: 1st is the set of tweets about US airline companies and their sentiment, 2nd is the Turkish sign language digits. The sentiment analysis dataset was imbalanced, so I applied over sampling methods (SMOTE, ADASYN) to adjust the class distribution. Furthermore, both datasets were high dimensional, so I useed PCA for dimensionality reduction. For the sentiment analysis task, the best test accuracy obtained was 80.6% and was obtained by SVM. On the other hand, the best classification test accuracy I got for the Turkish sign language dataset was 98.06% and was obtained by Convolution Neural Networks.

**6 References**

[1] https://en.wikipedia.org/wiki/Oversampling\_and\_undersampling\_in\_data\_analysis

[2] <https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume16/chawla02a-html/chawla2002.html>

[3] <http://sci2s.ugr.es/keel/pdf/algorithm/congreso/2008-He-ieee.pdf>

[4]<https://www.researchgate.net/publication/315643035_Online_Social_Media-based_Sentiment_Analysis_for_US_Airline_companies>

[5] <https://arxiv.org/abs/1412.6980>