**The Office: text classification via Machine Learning  
Mareta Masaeva**

The current project is a text classification problem, which I attempt to solve using a number of models we discussed in our Machine Learning classes.

**Dataset**

The text that will be classified comes from a dataset based on The Office, a famous American sitcom. The dataset consists of every line in the series, the character who said the line, and the season and episode number the line was spoken in. The titles of the episodes and the scene numbers are also included. All of these are in chronological order; one could read through the entire series by reading every line in the ‘line’ column. The series consists of 9 seasons and 567 episodes, and in total 54 626 lines were spoken. These lines can consist of one word, or multiple sentences; a line is just a character speaking without being interrupted. Thus, the longest line in the series consists of 232 words and 34 sentences, spoken by Pam. The shortest lines can be any word (“Yes.”, “Well.”, “Okay.”, “Oh.”, ...) and can be spoken by anyone.

The problem that I, then, attempted to solve, consists of two parts: can a model take as input a line, and predict the gender of the person who said that line? And secondly, can the model predict who exactly said a particular line? The first problem is a binary one, while the latter is a multiclass problem.

To account for the information that I would need to tackle these problems, some pre-processing of the dataset was conducted. Three columns were added: a gender column, a column with the number of words in every line, and a column with speaker ID’s. Eventually, the number of words column turned out unnecessary; it was merely used for some experimentation with different models. The gender column was used for the binary problem and consists of two classes, male and female, and the speaker ID column consists of 25 classes. There are more characters than this in The Office, but for the sake of simplicity, only the main characters were included in the analysis. This leaves us with 47 664 lines to analyse.

**Results**

For the binary problem, the following three models were used: logistic regression, naive Bayes, and a simple neural network.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Logistic Regression | | | Naive Bayes | | | Neural Network | | |
|  | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 |
| Male | .76 | .97 | .85 | .76 | 1.00 | .86 | .75 | .99 | .86 |
| female | .42 | .06 | .10 | .71 | 0.00 | .01 | .19 | 0.01 | .02 |
| Macro avg | .59 | .52 | .48 | .73 | .50 | .43 | .47 | .50 | .44 |
| Weighted avg | .68 | .75 | .67 | .75 | .75 | .65 | .62 | .75 | .65 |
| Accuracy | .75 | | | .75 | | | .75 | | |

As we can see from the above table, none of the three models performed that well. The highest accuracy in training was reported from the neural network, but as we can deduce from the recall from testing, it appears that the network classified all lines as being spoken by men. The same goes for the two other models, and this results in an overall accuracy of .75. As 17 of the 25 characters are men, and the two characters with the most lines, Michael and Dwight, are men as well, this is an obvious case of overfitting on this class.

For the neural network, during training, it appeared that the more the model learned of the data, the more it started overfitting on the male class; during its first couple runs, it classified two out of ten instances as female, but started classifying all as male after restarting the runtime and running the code again.

The model with the highest precision for the female class is naive Bayes, where .71 of positive identifications were actually correct. However, the confusion matrix shows that there were only 10 of these positive identifications. For the logistic regression model, this number is higher; there are 408 true positives for the female class. Compared to the total number of lines, however, this number is still incredibly low. For the neural network, the number is 35. In conclusion, all three models had trouble identifying the female-spoken lines, because of the class imbalance in the dataset.

For predicting the speaker from text, I used Support Vector Machines, a neural network, and a Long Short Term Memory network.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | SVM | | | Multiclass NN | | | LSTM | | |
|  | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 |
| Michael | .20 | .13 | .16 | .23 | .98 | .37 | .78 | .94 | .85 |
| Jim | .17 | .08 | .11 | .73 | .03 | .05 | .81 | .85 | .83 |
| Pam | .04 | .01 | .02 | .70 | .02 | .04 | .85 | .82 | .84 |
| Dwight | .15 | .04 | .07 | .55 | .05 | .08 | .93 | .87 | .90 |
| Jan | .17 | .10 | .12 | .70 | .03 | .05 | .94 | .78 | .86 |
| Phyllis | .26 | .27 | .27 | 1.00 | .01 | .02 | .90 | .84 | .87 |
| Stanley | .14 | .07 | .10 | .86 | .03 | .05 | .93 | .86 | .89 |
| Oscar | .11 | .04 | .05 | .92 | .02 | .05 | .95 | .80 | .87 |
| Angela | .08 | .03 | .04 | .82 | .02 | .03 | .87 | .83 | .85 |
| Kevin | .21 | .27 | .24 | .77 | .02 | .04 | .92 | .81 | .86 |
| Ryan | .15 | .05 | .08 | .65 | .03 | .05 | .93 | .79 | .86 |
| Roy | .13 | .08 | .09 | .50 | .01 | .03 | .98 | .72 | .83 |
| Toby | .09 | .04 | .05 | .85 | .04 | .07 | .93 | .79 | .86 |
| Kelly | .29 | .53 | .38 | .80 | .05 | .09 | .92 | .86 | .89 |
| Meredith | .13 | .06 | .08 | .75 | .02 | .03 | .96 | .85 | .90 |
| Darryl | .11 | .06 | .08 | 1.00 | .02 | .03 | .94 | .84 | .89 |
| Creed | .19 | .17 | .18 | 1.00 | .02 | .04 | .96 | .89 | .93 |
| Andy | .04 | .01 | .02 | .74 | .04 | .08 | .96 | .86 | .90 |
| Pete | .12 | .04 | .06 | 0.00 | 0.00 | 0.00 | 1.00 | .74 | .85 |
| Erin | .17 | .07 | .09 | .67 | .02 | .04 | .84 | .81 | .83 |
| Gabe | .00 | .00 | .00 | .67 | .02 | .04 | .96 | .89 | .92 |
| Clark | .07 | .03 | .04 | 1.00 | .05 | .10 | .99 | .88 | .93 |
| Robert | .11 | .04 | .06 | .76 | .10 | .18 | .96 | .83 | .89 |
| Robert California | .06 | .02 | .03 | 0.00 | 0.00 | 0.00 | 1.00 | 1.00 | 1.00 |
| Nellie | .15 | .05 | .08 | .82 | .05 | .10 | .94 | .87 | .90 |
| Macro avg | .13 | .09 | .10 | .70 | .07 | .07 | .93 | .84 | .88 |
| Weighted avg | .21 | .24 | .21 | .61 | .25 | .13 | .87 | .86 | .86 |
| Accuracy | .24 | | | .25 | | | .86 | | |

The table above shows that the best model for this problem is the LSTM. It returned the highest numbers overall compared to the SVM and neural network, which leads to believe that it might also have been a better fitting model for predicting gender from text.

For one character, Robert California (who is included twice in the analysis, because he is referred to as both ‘Robert’ and ‘Robert California’ in the original dataset), it even returns a perfect precision and perfect recall. The simple neural network also returns quite high precision in many cases, but the recall is extremely low in all but one class; the character that has the most lines overall, Michael Scott. It appears that the neural network was able to correctly identify almost all of Michael’s lines, but only .23 of the positive identifications were correct. This leads us to believe that this model was classifying most lines as Michael’s. This might be due to the class imbalance, as Michael had the most lines of all characters, even though he was totally absent in the last two seasons. I was able to increase the model’s accuracy by about .15, but it still remains at the low point of .25.

SVM performs slightly worse than the neural network and is not able to reach f1-scores higher than .30. This model does not seem to be suffering from the class imbalance but is merely unable to classify many lines correctly.

From this we can conclude that the LSTM is the best model for this multiclass problem, and maybe even for the binary problem.