Assignment 2 Chang Hong Jie A0140154W

Throughout this analysis, RTextTools library is used to perform training and evaluation. Some tm methods are also used for data pre-processing. To pre-process and clean the data with greater control, a cleaning function was used to manually clean the raw data instead of the default parameters used in creating the TDM matrix. The specific cleaning method that was started with is as follows:

```
clean.data <- function(data) {
  data <- data[!(is.na(data$review) | data$review==""), ]
  data$review <- tolower(data$review)
  data$review <- removeWords(data$review, myStopwords)
  punct <- '[]\\?!\"#$%&(){}+*/:;,._`|~\\[<=>@\\^-]'
  data$review <- removePunctuation(data$review)
  data$review <- gsub(punct, "", data$review)
  data$review <- removeNumbers(data$review)
  data$review <- stripWhitespace(data$review)
  data$review <- stemDocument(data$review)
  data$review <- iconv(data$review, "latin1", "ASCII", sub="")
  data <- data[!(is.na(data$review) | data$review=="" | data$review==" "), ]
  data<- as.matrix(data$review)
}
```

Firstly, NA and empty string values are removed from the data, then proceed to remove extraneous characters before stemming. Non-ASCII characters (i.e. foreign characters and emoticons) were removed after stemming so words do not bind together incorrectly prior to stemming. Any resulting empty strings are removed again to prevent NULL errors in analysis later.

```
#sampling
dataSize <- floor(0.8*nrow(weather.clean))
set.seed(1)
```

Sampling was fixed as 80% of weather data's size, as weather data was found to have the least number of cleaned reviews across various runs. This resulted in a consistent sample size across the classes, although sacrificing on sample size slightly.

```
#partition train and test data
train.size <- floor(0.75*dataSize)
train.inx <- sample(dataSize,size = train.size)
```

Partitioning of training and testing data was maintained at a ratio of 3:1 throughout the runs. This ensured consistent results to isolate any improvements to the various techniques applied in the assignments. Also, when sample size is small, the ratio ensures relatively sufficient data is used for training the classifiers.

No feature engineering techniques are done as the predictors are from a text vector. The default parameters are used in create_matrix method with the pre-processing parameters set to false as cleaning has already been done to the texts.

The library methods offered by RTextTools are used to train, classify and subsequently create the Precision & Recall scorings accordingly as per the R script attached. For training algorithms, SVM, Decision Tree and Max Entropy have been picked arbitrarily in the beginning for fast yet effective classification. The following runs are documented to show how the original analysis and classification models have improved with various techniques applied.

<u>Run 1:</u>

- **Cleaning**: Default
- **Process**: Sample 1,000 rows from each category, and partition 3:1 ratio for training : test data.
- **Features**: TDM of training data without any feature engineering or selection
- **Algorithms**: Trained using SVM, Decision Tree and Max Entropy to determine useful algorithms for Ensemble Learning later
- **Test results**:

	SVM_PRECISION	SVM_RECALL	SVM_FSCORE	TREE_PRECISION	TREE_RECALL	TREE_FSCORE	MAXENTROPY_PRECISION	MAXENTROPY_RECALL	MAXENTROPY_FSCORE
1	0.52	0.38	0.44	NaN	0.00	NaN	0.52	0.39	0.45
2	0.53	0.46	0.49	0.30	0.35	0.32	0.54	0.48	0.51
3	0.76	0.50	0.60	0.92	0.39	0.55	0.59	0.52	0.55
4	0.31	0.54	0.39	0.23	0.73	0.35	0.34	0.46	0.39
5	0.54	0.53	0.53	1.00	0.20	0.33	0.52	0.57	0.54

Conclusion:

Cleaning may not be optimal as there may be stopwords after stemming. Also, many short reviews were filled with junk words or useless predictors that affected accuracy. As seen in the results above, Class 3 ("Game") also has a lower accuracy compared to the other categories in terms of F-score, which offers a balanced view of Precision and Recall. This could be due to a poor training dataset with fewer defining terms for game apps than other categories. Also, the F-scores of SVM and Max Entropy are relatively better in general than Decision Tree. Since Decision Tree does not seem to work well, SVM and Max Entropy will be used as the benchmark for scoring to keep the runs short. Other algorithms such as Random Forest and Logistic Regression have been used as well but their performances were comparatively worse than SVM and MaxEnt. 2 algorithms are used to provide a more balanced view of the scores as certain algorithms may favour some types of data and skew the prediction scores.

To reduce the number of junk words found in short reviews, I removed any reviews which were 280 characters or less in length. This would increase the quality of the training data as examining the training data showed that long reviews tend to contain more good quality words that associate with the right class. The next run indicates the improvements gained from this filtering technique.

Run 2:

- **Cleaning**: Only used reviews that were longer than 280 characters
- **Process**: Sample 223 rows from each category (i.e. 0.8 times of smallest category Weather), and partition 3:1 ratio for training : test data.
- **Algorithms**: Trained using SVM and Max Entropy
- **Test results**:

	SVM_PRECISION	SVM_RECALL	SVM_FSCORE	MAXENTROPY_PRECISION	MAXENTROPY_RECALL	MAXENTROPY_FSCORE
1	0.75	0.71	0.73	0.65	0.64	0.64
2	0.81	0.84	0.82	0.69	0.79	0.74
3	0.84	0.75	0.79	0.87	0.73	0.79
4	0.67	0.71	0.69	0.71	0.64	0.67
5	0.81	0.86	0.83	0.78	0.88	0.83

Conclusion:

There is a rather significant increase in all 3 scores across the board for both algorithms with the new cleaning technique of removing shorter reviews. However, this meant sacrificing in sample size as we only got around 1000 data points to train and test with. This indicates that long reviews are a good indicator of their respective classes and are easier to classify, but the small sample size may skew the training and evaluation scores. Thus, a better training corpus had to be incorporated into the models to improve the accuracy of Class 4 ("Social") while also increasing training sample size.

To further improve the model, Feature Selection techniques were used. The following code was used to find the most common features in our trained model and remove possible stopwords that have not been removed. App descriptions were also included in the training data to improve the corpus and strength of good quality words/features in the models.

```
#find most prominent features to check for possible stopwords
freq <- as.matrix(findFreqTerms(m, 50))
common.features = data.frame(term = freq, value = tm_term_score(m, freq, FUN = slam::col_sums))
```

The words identified were “app” and “love” which had very high number of occurrences in the TDM and skews the results a lot. Intuitively, these words are irrelevant to the classification as they do are not particularly indicative of any class, and should thus be omitted from the corpus.

term	value
app	2091
love	1520
weather	1235
game	560
time	554
free	466
phone	454

Run 3:

- **Cleaning**: Most common features in the TDM matrix found in table on the right. As a result, removed “app”, “love” after stemming to completely remove any related words. App descriptions were added in and cleaned as well.
- **Process**: Sample 201 rows from each category (i.e. 0.8 times of smallest category Weather), and partition 3:1 ratio for training : test data.
- **Features**: TDM of training data including app descriptions added in without any feature engineering or selection
- **Algorithms**: Trained using SVM and Max Entropy to determine useful algorithms for Ensemble Learning later
- **Test results**:

```
> s
  SVM_PRECISION SVM_RECALL SVM_FSCORE MAXENTROPY_PRECISION MAXENTROPY_RECALL MAXENTROPY_FSCORE
1          0.74       0.69       0.71                0.68                0.70               0.69
2          0.79       0.88       0.83                0.81                0.85               0.83
3          0.89       0.85       0.87                0.94                0.87               0.90
4          0.68       0.67       0.67                0.69                0.69               0.69
5          0.87       0.88       0.87                0.88                0.88               0.88
```

Conclusion:

Removing the top stopword features had mixed results on accuracy. Class 1 (“education”) & 4 (“social”) performance did not improve substantially so more features will need to be obtained, possibly through relaxing the review length constraint. Also, the number of data points reduced even though app descriptions were added in as the common stopwords have been removed.

Run 4:

- **Cleaning**: Reduced character limit to 150 to increase the number of reviews and additionally removed the words “good” and “great” which were subsequently found in the top features.
- **Process**: Sampled around 3000 rows from each category (i.e. 0.8 times of smallest category Weather), and partition 3:1 ratio for training : test data.
- **Features**: TDM of training data including app descriptions added in without any feature engineering or selection
- **Algorithms**: Trained using SVM and Max Entropy.
- **Test results**:

```
  SVM_PRECISION SVM_RECALL SVM_FSCORE MAXENTROPY_PRECISION MAXENTROPY_RECALL MAXENTROPY_FSCORE
1          0.79       0.80       0.79                0.79                0.83               0.81
2          0.86       0.86       0.86                0.88                0.84               0.86
3          0.91       0.90       0.90                0.93                0.91               0.92
4          0.79       0.79       0.79                0.80                0.81               0.80
5          0.90       0.91       0.90                0.91                0.91               0.91
```

Conclusion:

The relaxing of the character limit to 150 gave optimal results based on the current sampling and partitioning ratios. The recall scores for all classes improved during this run, with massive improvements for Education and Social classes, which likely had poor features with very long reviews. Removing “good” and “great”, which became one of the most common features in the new TDM, may have affected precision slightly for some

classes, however the removal is justified as the features are irrelevant to this classification and recall scores still improved with the new techniques applied.

Based on the model above, the ensemble summary scores are as follows:

```
        n-ENSEMBLE COVERAGE n-ENSEMBLE RECALL
n >= 1                1.00              0.86
n >= 2                0.84              0.94
```

According to RTextTools documentation, the matrix implies that when both classifiers (SVM and Max Entropy) agree on a classification (i.e. n >= 2), 84% of the model classifications have a recall score of 0.94, which is very high for a text classification analysis. To examine the effects a larger ensemble has on the coverage and recall scores, we further include Logistic Regression algorithm to make the number of classifiers odd as per the diagram below.

```
models <- train_models(container = container.train, algorithm =  c("MAXENT", "SVM", "GLMNET"))
```

The respective precision/recall scores and ensemble summary scores are as follows:

	SVM_PRECISION	SVM_RECALL	SVM_FSCORE	GLMNET_PRECISION	GLMNET_RECALL	GLMNET_FSCORE	MAXENTROPY_PRECISION	MAXENTROPY_RECALL	MAXENTROPY_FSCORE
1	0.79	0.79	0.79	0.73	0.83	0.78	0.79	0.83	0.81
2	0.86	0.86	0.86	0.87	0.83	0.85	0.88	0.84	0.86
3	0.91	0.89	0.90	0.93	0.87	0.90	0.93	0.91	0.92
4	0.79	0.79	0.79	0.77	0.81	0.79	0.80	0.81	0.80
5	0.90	0.91	0.90	0.92	0.86	0.89	0.91	0.91	0.91

```
> ensemble
        n-ENSEMBLE COVERAGE n-ENSEMBLE RECALL
n >= 1                1.00              0.87
n >= 2                0.97              0.88
n >= 3                0.79              0.96
```

As seen here, the ensemble where the majority vote (n >= 2) of the classifiers is used in classification has very positive results. 97% of all the data can be predicted with a recall score of 0.88, while if all 3 classifiers agreed on a prediction, nearly 80% of the data can be predicted with an exceptional recall score of 0.96.

To get a more accurate evaluation of the data, 5-fold cross-validation is done on the training data.

```
> cv = cross_validate(container.train, 5)
Fold 1 Out of Sample Accuracy = 0.8261157
Fold 2 Out of Sample Accuracy = 0.8444673
Fold 3 Out of Sample Accuracy = 0.8186849
Fold 4 Out of Sample Accuracy = 0.8239079
Fold 5 Out of Sample Accuracy = 0.8365991
> cv$meanAccuracy
[1] 0.829955
```

The mean accuracy derived from cross-validation was approximately **0.83**.

Techniques that did not work out:

Unfortunately, removing the most common stopwords alone did not yield any notable increase in performance of the classifiers. This can likely be attributed to the reduction in sample data and less fitting of the trained model.

Also, bigrams could not be used in create_matrix due to an error, although it was hypothesized to increase accuracy of long reviews significantly. As such, engineering the features with bigrams was not done and its performance boost cannot be measured in this assignment.

Increasing the sample size to the order of tens of thousands also did not improve the classification performance. This can be attributed to increased noise in the data with more junk features added into the corpus and not contributing to classification accuracy. The optimal sample size was found to be between 1000 and 3000, depending on the sampling and partitioning.

References

https://journal.r-project.org/archive/2013-1/collingwood-jurka-boydstun-etal.pdf