CSCI 620 - Introduction to Big Data

Project Report (Final Phase)

21 April, 2019

Vaibhav Joshi
vj3470@rit.edu

Akash Desai
ad3059@rit.edu

Karshit Shah
ks6675@rit.edu

Nihal Parchand
np9603@rit.edu

Rohit Kunjilikattil
rk4447@rit.edu

ABSTRACT

Since the advent of the internet, we have come a long way with how we read and process various news articles. It is much more easier and we are afforded much more freedom to document our opinions be it via news websites, personal blogs, social networking sites like Reddit, or through dedicated article publishing sites such as Mashable, Medium etc. To post articles is one thing, but to find avid readers is a tough ask.

The popularity of an article is of great importance. Hence, for this project we have made use of an Online News Popularity Data Set which contains the compilation of 61 attributes for a total of 39644 articles posted on Mashable. We aim to find out the popularity of articles using Classification and Regression Trees(CART) and Naive Bayes Classifier. We analyse the accuracy of each to determine the popularity of various article by generating confusion matrices and plotting ROC curves.

1. ABOUT THE DATASET - ONLINE NEWS POPULARITY

This is a multivariate dataset which summarizes a set of features about articles published by Mashable in a period of two years.
In total, there are 39644 articles where there are attributes such as URL, number of images, number of videos, category etc. for each article. There are in total 61 attributes for each article.

Out of the 61 attributes, we have used the following attributes:

1. n_tokens_title: Number of words in the title
2. n_tokens_content: Number of words in the content
3. n_unique_tokens: Rate of unique words in the content
4. n_unique_tokens: Rate of unique words in the content
5. n_non_stop_words: Rate of non-stop words in the content
6. n_non_stop_unique_tokens: Rate of unique non-stop words in the content
7. num_hrefs: Number of links
8. num_imgs: Number of images
9. num_videos: Number of videos
10. average_token_length: Average length of the words in the content
11. num_keywords: Number of keywords in the metadata
12. data_channel_is_lifestyle: Is data channel 'Lifestyle'?
13. data_channel_is_entertainment: Is data channel 'Entertainment'?
14. data_channel_is_bus: Is data channel 'Business'?
15. data_channel_is_socmed: Is data channel 'Social Media'?
16. data_channel_is_tech: Is data channel 'Tech'?
17. data_channel_is_world: Is data channel 'World'?
18. kw_min_min: Worst keyword (min. shares)
19. kw_max_min: Worst keyword (max. shares)
20. kw_avg_min: Worst keyword (avg. shares)
21. kw_min_max: Best keyword (min. shares)
22. kw_max_max: Best keyword (max. shares)
23. kw_avg_max: Best keyword (avg. shares)
24. kw_min_avg: Avg. keyword (min. shares)
25. kw_max_avg: Avg. keyword (max. shares)
26. kw_avg_avg: Avg. keyword (avg. shares)
27. self_reference_min_shares: Min. shares of referenced articles in Mashable

.

28. self_reference_max_shares: Max. shares of referenced articles in Mashable
29. self_reference_avg_sharess: Avg. shares of referenced articles in Mashable
30. weekday_is_monday: Was the article published on a Monday?
31. weekday_is_tuesday: Was the article published on a Tuesday?
32. weekday_is_wednesday: Was the article published on a Wednesday?
33. weekday_is_thursday: Was the article published on a Thursday?
34. weekday_is_friday: Was the article published on a Friday?
35. weekday_is_saturday: Was the article published on a Saturday?
36. weekday_is_sunday: Was the article published on a Sunday?
37. is_weekend: Was the article published on the weekend?
38. shares: Number of shares (target)

Some of the attributes listed below have been discarded because either they are non predictive .i.e, they don't have any useful information for prediction. Also, there are some attribute which are useful for Natural Language Processing (NLP), Sentiment Analysis. The main objective of this project is predictive analysis and hence these attributes bear little or no relevance so they have been discarded.

1. url: URL of the article (non-predictive)
2. timedelta: Days between the article publication and the dataset acquisition (non-predictive)

All attributes below are for NLP purpose hence they are discarded.

1. LDA_00: Closeness to LDA topic 0
2. LDA_01: Closeness to LDA topic 1
3. LDA_02: Closeness to LDA topic 2
4. LDA_03: Closeness to LDA topic 3
5. LDA_04: Closeness to LDA topic 4
6. global_subjectivity: Text subjectivity
7. global_sentiment_polarity: Text sentiment polarity
8. global_rate_positive_words: Rate of positive words in the content
9. global_rate_negative_words: Rate of negative words in the content
10. rate_positive_words: Rate of positive words among non-neutral tokens
11. rate_negative_words: Rate of negative words among non-neutral tokens
12. avg_positive_polarity: Avg. polarity of positive words
13. min_positive_polarity: Min. polarity of positive words
14. max_positive_polarity: Max. polarity of positive words
15. avg_negative_polarity: Avg. polarity of negative words
16. min_negative_polarity: Min. polarity of negative words
17. max_negative_polarity: Max. polarity of negative words
18. title_subjectivity: Title subjectivity
19. title_sentiment_polarity: Title polarity
20. abs_title_subjectivity: Absolute subjectivity level
21. abs_title_sentiment_polarity: Absolute polarity level

2. DATA PREPROCESSING

- **Importing Libraries:**
  library(caTools)
  library(rpart)
  library(compare)
  library(caret)
  library(lattice)
  library(ggplot2)
- **Importing Dataset:**
  # Read data
  > dataset<- read.csv("OnlineNewsPopularity.csv")
- **Cleaning the Dataset:**
  # Removing NA values
  > dataset <- na.omit(dataset)
  # Removing unnecessary columns
  >dataset <- dataset[, -c(1,2,40:60)]
- **Factoring Categorical Attributes:**
  # Here there are multiple columns for each weekday, which are categorical attribute. The code below factors all these attributes into a single column.

  >dataset$Day <- ifelse(dataset$weekday_is_monday == 1,"monday", ifelse((dataset$weekday_is_tuesday == 1), "tuesday", ifelse((dataset$weekday_is_wednesday == 1), "wednesday", ifelse((dataset$weekday_is_thursday == 1), "thursday", ifelse((dataset$weekday_is_friday == 1), "friday", ifelse((dataset$weekday_is_saturday == 1), "saturday", "sunday"))))))

  >dataset$Day <- factor(dataset$Day, levels=c('monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'sunday'), labels=c(1,2,3,4,5,6,7))

  dataset$dataChannel <- ifelse(dataset$data_channel_is_lifestyle == 1,"lifestyle", ifelse(dataset$data_channel_is_entertainment == 1,"entertainment", ifelse(dataset$data_channel_is_bus == 1,"business", ifelse(dataset$data_channel_is_socmed == 1,"Social", ifelse(dataset$data_channel_is_tech == 1,"tech", "world")))))

  dataset$dataChannel <- factor(dataset$dataChannel, levels=c('lifestyle', 'entertainment', 'business', 'Social', 'tech', 'world'), labels=c(1,2,3,4,5,6))

  >dataset <- dataset[,-c(12:17,30:36)]

- **Splitting the dataset into Training Set and Test Set:** # Splitting the data into training and test sets in
  75% : 25% proportion
  >split = sample.split(dataset$shares, SplitRatio = 0.75)
  >training_set = subset(dataset, split == TRUE)
  >test_set = subset(dataset, split == FALSE)

- **Classifying articles:**
  Dividing articles into two categories based on the median of shares i.e 1400
  >dataset$popularity <- ifelse(dataset$shares <=1400, "low","high")

- **Finding Significant attributes using Correlation Matrix:**
  Correlation between each attribute and shares attribute:
  correlation<-as.data.frame(as.table(cor(dataset[, -c(dataset$shares)], dataset$shares)))
  names(correlation)<-c("Attribute","Shares", "Correlation with shares attribute")
  correlation<-correlation[-c(2)]

	Attribute	Correlation with shares attribute
34	shares	1.0000000000
21	kw_max_avg	0.0643058638
24	self_reference_avg_sharess	0.0577888974
22	self_reference_min_shares	0.0559575751
23	self_reference_max_shares	0.0471152233
19	kw_avg_max	0.0446858448
20	kw_min_avg	0.0395506945
6	num_imgs	0.0393875978
17	kw_max_min	0.0301139367
7	num_videos	0.0239360695
9	num_keywords	0.0218182272
30	is_weekend	0.0169581853
28	weekday_is_saturday	0.0150822494
25	weekday_is_monday	0.0097264351
2	n_tokens_title	0.0087831188
1	timedelta	0.0086622877
29	weekday_is_sunday	0.0082295387
18	kw_max_max	0.0078625693
10	data_channel_is_lifestyle	0.0058312673
13	data_channel_is_socmed	0.0050212163
32	global_sentiment_polarity	0.0041629291
3	n_non_stop_words	0.0004429416
4	n_non_stop_unique_tokens	0.0001141719
16	kw_min_min	-0.0010509877
5	num_self_hrefs	-0.0019004034
31	LDA_00	-0.0037930631
27	weekday_is_wednesday	-0.0038006719
26	weekday_is_tuesday	-0.0079406519
12	data_channel_is_bus	-0.0123761662
14	data_channel_is_tech	-0.0132528744

Fig. Correlation matrix of shares vs other attributes

3. DATA CLASSIFICATION AND MINING

The dataset is now devoid of unnecessary columns. Further, it has been split into training and test sets. Article will be classified on the basis of popularity. This will be done by data mining using the significant attributes of the dataset. This cannot be done using data management as it is not possible to predict anything for a new data apart from the available data. Classification will be used here as the articles will be classified into two categories depending on the number of shares of the article. For classification, the algorithms used on this dataset have been described below:

- **CART:**

  Classification and Regression Trees or CART for short is a term introduced by Leo Breiman to refer to Decision Tree algorithms that can be used for classification or regression predictive modeling problems.

  Classically, this algorithm is referred to as "decision trees", but on some platforms like R they are referred to by the more modern term CART.

  Decision tree is a classification technique in which a model is created that anticipates the value of target variable depends on input values. CART and C4.5 are commonly used decision tree algorithms.

  # R code for CART implementation
  >analysis<-rpart(dataset$ popularity kw_max_avg + self_reference_avg_sharess + self_reference_min_shares + self_reference_max_shares + kw_avg_max + kw_min_avg + num_imgs + kw_max_min + num_videos + num_keywords + is_weekend + weekday_is_saturday, data=training_set)

  > plot(analysis, uniform = TRUE, margin = 0.2)
  > text(analysis)

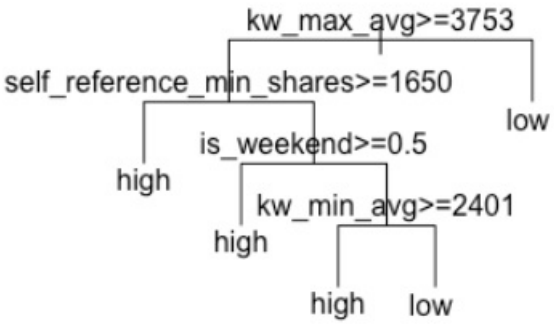

Fig. Decision Tree

- **Naive Bayes:**

  Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where

the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable.

R code for Naive Bayes

> news_naive_bayes = train(popularity $\tilde{}$ kw_max_avg + self_reference_avg_sharess + self_reference_min_shares + self_reference_max_shares + kw_avg_max+kw_min_avg + num_imgs + kw_max_min + num_videos + num_keywords + is_weekend + weekday_is_saturday, data=training_set , method= "nb", trControl=trainControl(method="cv", number=10))

```
Naive Bayes

29898 samples
   12 predictor
    2 classes: 'high', 'low'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 26909, 26908, 26908, 26908, 26908, 26908, ...
Resampling results across tuning parameters:

  usekernel  Accuracy   Kappa
  FALSE      0.5614757  0.11595805
   TRUE      0.5113721  0.01051924

Tuning parameter 'fL' was held constant at a value of 0
Tuning parameter 'adjust' was held constant at
 a value of 1
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were fL = 0, usekernel = FALSE and adjust = 1.
```

Fig. Naive Bayes Classifier

- **Confusion Matrix**

```
Confusion Matrix and Statistics

          low high
low  3388 2151
high 1556 2651

               Accuracy : 0.6196
                 95% CI : (0.6099, 0.6293)
    No Information Rate : 0.5073
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.2378

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6853
            Specificity : 0.5521
         Pos Pred Value : 0.6117
         Neg Pred Value : 0.6301
             Prevalence : 0.5073
         Detection Rate : 0.3476
   Detection Prevalence : 0.5683
      Balanced Accuracy : 0.6187

       'Positive' Class : low
```

Fig. CART - Test Set

4. DATA VISUALIZATION

By training both the models on the training set and using them on the test set to predict the results, the accuracy of Decision Tree came out as a better one. Accuracy's of both the models were:

Accuracy	Training Set	Test Set
CART	61.69%	61.96%
Naive Bayes	56.23%	56.33%

```
Confusion Matrix and Statistics

          low high
low  4337 3649
high  607 1153

               Accuracy : 0.5633
                 95% CI : (0.5534, 0.5732)
    No Information Rate : 0.5073
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.1184

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8772
            Specificity : 0.2401
         Pos Pred Value : 0.5431
         Neg Pred Value : 0.6551
             Prevalence : 0.5073
         Detection Rate : 0.4450
   Detection Prevalence : 0.8194
      Balanced Accuracy : 0.5587

       'Positive' Class : low
```

Fig. Naive Bayes - Test Set

```
Confusion Matrix and Statistics

          low  high
low     10846  7161
high     4292  7599

               Accuracy : 0.6169
                 95% CI : (0.6114, 0.6224)
    No Information Rate : 0.5063
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.2319

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.7165
            Specificity : 0.5148
         Pos Pred Value : 0.6023
         Neg Pred Value : 0.6391
             Prevalence : 0.5063
         Detection Rate : 0.3628
   Detection Prevalence : 0.6023
      Balanced Accuracy : 0.6157

       'Positive' Class : low
```

Fig. CART - Training Set

```
Confusion Matrix and Statistics

          low  high
low     13364 11311
high     1774  3449

               Accuracy : 0.5623
                 95% CI : (0.5567, 0.568)
    No Information Rate : 0.5063
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.1174

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8828
            Specificity : 0.2337
         Pos Pred Value : 0.5416
         Neg Pred Value : 0.6603
             Prevalence : 0.5063
         Detection Rate : 0.4470
   Detection Prevalence : 0.8253
      Balanced Accuracy : 0.5582

       'Positive' Class : low
```

Fig. Naive Bayes - Training Set

- **ROC Curve**

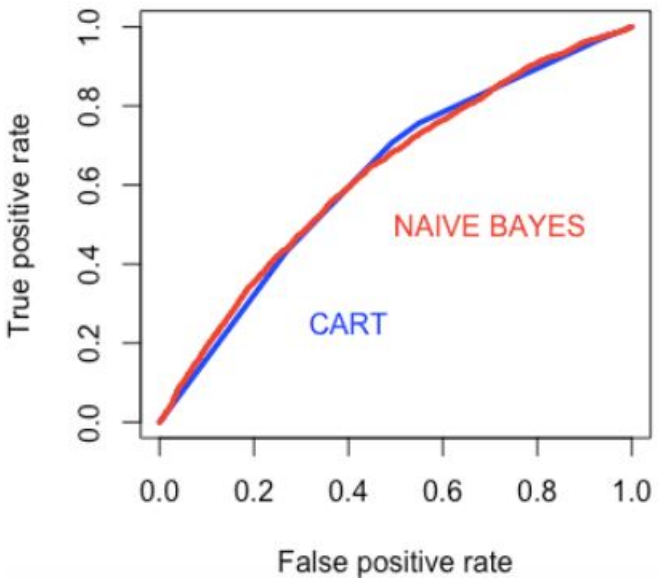

Fig. ROC Curve

5. CONCLUSION

By performing the analysis on the Online News Popularity Dataset we note the following observations :

- Having plotted the confusion matrices for both the Naive Bayes and CART classification techniques, we observed that the accuracy for both come out as :
  - Test Set :
    CART : Total - (False Positive + False Negative)/ Total = 0.6196
    Naive Bayes : Total - (False Positive + False Negative)/ Total = 0.5633
  - Training Set :
    CART : Total - (False Positive + False Negative)/ Total = 0.6169
    Naive Bayes : Total - (False Positive + False Negative)/ Total = 0.5623
- ROC(Receiver Operating Characteristic) curves are commonly used to characterize the sensitivity/specificity tradeoffs for a binary classifier such as Bayes. Different threshold values give different levels of sensitivity and specificity.The ROC curve plots true positive rate against false positive rate.
- The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the test.
- The closer the curve comes to the 45-degree diagonal of the ROC space, the less accurate the test.
- By plotting the ROC curve for Naive Bayes and CART, we observe that the curve is above the 45 diagonal for both. This means that they both achieve more than average accuracy with CART performing slightly better.

6. MISCELLANEOUS

- **Challenges Faced:**
  - While trying to divide the popularity into three classes - High, Medium and Low according to the number of shares, the decision tree was not able to classify it in three categories. Also, the tree could not classify into low and high for some partition values. This is a limitation of Binary Classifiers.
- **Workload Distribution:**

  All the five of us were actively involved in the development of the project. Below is the breakdown:-
  - Data Preprocessing - Vaibhav / Karshit
  - Data mining technique (CART)- Nihal / Rohit
  - Data mining technique(Naive Bayes)- Vaibhav / Akash
  - Confusion matrix for both models- Nihal/Rohit
  - ROC curves for both models - Karshit / Akash
  - Documentation Report - Everyone

7. REFERENCES

- https://machinelearningmastery.com/classification-and-regression-trees-for-machine-learning/
- https://en.wikipedia.org/wiki/Naive_Bayes_classifier
- https://archive.ics.uci.edu/ml/datasets/online+news+popularity
- https://en.wikipedia.org/wiki/Decision_tree
- https://www.medcalc.org/manual/roc-curves.php