~~~
---
title: "Naive Bayes Classification"
author: "Erin Tafarshiku"
date: "`r Sys.Date()`"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)

knitr::opts_chunk$set(echo = TRUE)

userReviews <- merge( review_data_small, user_data_small, by="user_id")

install.packages("tm")
install.packages("SnowballC")
install.packages("tidytext")
install.packages("dplyr")
install.packages("caret")
install.packages("e1071")
install.packages("ggplot2")

library(tm)
library(SnowballC)
library(tidytext)
library(dplyr)
library(caret)
library(e1071)
library(ggplot2)

# Sample a subset of userReviews for computational efficiency
set.seed(1)
sampledReviews <- userReviews[sample(nrow(userReviews), 90000), ]

# Text processing
corpus <- Corpus(VectorSource(sampledReviews$text))
corpus <- tm_map(corpus, content_transformer(tolower))
corpus <- tm_map(corpus, removePunctuation)
corpus <- tm_map(corpus, removeNumbers)
corpus <- tm_map(corpus, removeWords, stopwords("english"))
corpus <- tm_map(corpus, stripWhitespace)

# Create and filter the Document-Term Matrix
dtm <- DocumentTermMatrix(corpus)
sparseDtm <- removeSparseTerms(dtm, 0.95)

# Convert to a data frame
reviews_df <- as.data.frame(as.matrix(sparseDtm))
colnames(reviews_df) <- make.names(colnames(reviews_df))
reviews_df$stars <- factor(sampledReviews$stars, levels = 1:5)

# Splitting the data into training and test sets
set.seed(1)
trainingIndex <- createDataPartition(reviews_df$stars, p = .8999, list = FALSE)
trainingData <- reviews_df[trainingIndex, ]
testData <- reviews_df[-trainingIndex, ]

# Training the Naive Bayes model
model <- naiveBayes(stars ~ ., data = trainingData)

# Making predictions
predictions <- predict(model, testData)

# Evaluate the model
confMat <- confusionMatrix(predictions, testData$stars)
print(confMat)

# Compute and print the accuracy
~~~

~~~
accuracy <- sum(diag(confMat$table)) / sum(confMat$table)
print(paste("Accuracy:", accuracy))


# Visualisation
visualization_df <- data.frame(Actual = testData$stars, Predicted = predictions)
ggplot(visualization_df, aes(x = Actual, fill = Predicted)) +
  geom_bar(position = "dodge") +
  labs(title = "Distribution of Actual vs. Predicted Star Ratings", x = "Actual Star
Ratings", y = "Count") +
  scale_fill_discrete(name = "Predicted Stars") +
  theme_minimal()


# Load necessary libraries
install.packages("tidytext")
install.packages("dplyr")
install.packages("tibble")

library(tidytext)
library(dplyr)
library(tibble)

# Unnest tokens to separate words
word_sentiments <- sampledReviews %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("bing"))

# Count the number of positive and negative words
word_sentiment_counts <- word_sentiments %>%
  group_by(sentiment) %>%
  count(word, sort = TRUE)

# Display the sentiment counts for both positive and negative words
print(word_sentiment_counts)

# Separate positive and negative words
positive_words <- word_sentiments %>%
  filter(sentiment == "positive") %>%
  count(word, sort = TRUE)

negative_words <- word_sentiments %>%
  filter(sentiment == "negative") %>%
  count(word, sort = TRUE)

# Print out the top positive and negative words
print(positive_words)
print(negative_words)

```
~~~

```
---
title: "Markdown"
author: "Erin Tafarshiku"
date: "`r Sys.Date()`"
output: pdf_document
---
```

EC349 Assignment 1

DS Methodology

The CRISP-DM methodology is a well-established framework, highly regarded in the data science community for its effectiveness in managing complex projects. I began by outlining my business understanding to define the project's objectives. The data preparation phase aided in handling the vast and messy text data. During the modelling phase, I used a Naive Bayes classifier, while the evaluation and deployment phases ensured my results were both statistically valid and aligned with the business objectives.

I chose CRISP-DM for its structured approach, suitable for navigating the project’s lack of initial domain expertise and uncertain modelling path. Unlike the KDD methodology, CRISP-DM's fewer stages reduced complexity without losing detail, allowing for agile adaptation as the project evolved. This was particularly beneficial for the text-heavy Yelp data, where iterative refinement was necessary. Furthermore, CRISP-DM's emphasis on business understanding and deployment also provided a clear start and end point, aligning the technical process with business objectives, important phases which are not covered in the SEMMA approach.

Business Understanding

A growing body of literature has established the importance of user reviews and ratings as tools to support purchasing decisions of consumers, while proving valuable for businesses who use rating systems to build trust and reputation (Lackermair et. al., 2013), thus having a significant impact on product sales and business revenues. Due to the large nature of review data, such as the Yelp dataset, businesses will find it difficult to extract valuable information, especially without the expertise. To improve business operations, businesses must find meaningful patterns in data to better understand how to provide the best possible experience for their customers as this is proven to play a significant role in motivating customer loyalty (Ching et. al., 2019). Thus, the project aims to apply data science techniques to offer a repeatable methodology to help businesses draw insights directly from customers, ultimately improving operations and profitability in the long run.

Data Understanding

In the Data Understanding phase I used Yelp Open Data, comprising five sections: Business Information, User Check-Ins, User Information, User Reviews, User Tips, and featuring data on 150,346 businesses and 6,990,280 reviews. I focused on a subset containing 1,398,056 user reviews and 279,878 user data entries for practicality. Our variables of interest are ‘stars’, representing user ratings on a 1-5 scale, and ‘text’, detailing user sentiment.

Data Preparation

To begin, I merged two subsets of the Yelp Open Dataset - user information and user reviews - using the common ‘user_id’ field to create a unified dataset named 'userReviews', allowing me to associate review text with user ratings. For computational purposes, I used R's ‘sample()’ function with a set seed to randomly select 90,000 observations from userReviews.

In the text preprocessing stage, I converted all textual data to lowercase to maintain consistency and removed punctuation, numbers, and common English stopwords to standardise the text and remove irrelevant text that could disrupt the sentiment analysis. This was done using the 'tm' and 'SnowballC' - libraries tailored for text manipulation tasks.

I then transformed cleaned text into a Document-Term Matrix (DTM), where I refined the data by applying the 'removeSparseTerms()' function to the DTM to focus on terms with the highest relevance across the corpus. The sparse DTM was then converted into a data frame 'reviews_df', where each term represented a feature, and the user ratings ('stars') were formatted as a categorical outcome variable. For model training and testing, 'reviews_df' was split into two sets using the 'createDataPartition()' function from the 'caret' package, guided again by a set seed for reproducibility. The resulting training and test sets provided the foundation for building and assessing the Naive Bayes sentiment classification model.

Modelling

Literature implementing this method tend to produce accurate results (Dey et. al., 2016; Reddy et. al., 2017; Shah et. al, 2018). In this project, Naive Bayes uses Bayes' theorem to classify user reviews on a scale of 1 to 5, by calculating the conditional probabilities of features in a review belonging to a specific rating. An ideal feature of Naive Bayes is the feature independence assumption: treating features as non-influential makes it less sensitive to unimportant components in each document such as stop words.

Evaluation

The Naive Bayes model achieved an accuracy of 52.45%, outperforming the baseline chance of 20% for a five-class classification problem. The 95% confidence interval for the model's accuracy, ranging from 51.41% to 53.48%, suggests that similar results could be expected if the model were applied to different subsets of the data.

The statistics by class in our model's performance give us detailed insights into its predictive capabilities across the 1 to 5-star rating spectrum. Notably, the model excelled at recognising 5-star reviews, as evidenced by the high sensitivity score of 0.7858, indicating a strong ability to detect true positive 5-star ratings. Conversely, the model demonstrated a high specificity for class 1 with a score of 0.92402, reflecting its effectiveness in accurately identifying reviews that are not 1-star rated.

The model's positive predictive value (precision) for class 5 stood at 0.6158, suggesting a reasonable accuracy in predicting 5-star ratings when such a rating was forecasted. The prevalence metric shows that 5-star ratings are the most common in the dataset, appearing 46.50% of the time, which could contribute to the model's propensity to predict 5-stars more often.

The model's accuracy slightly surpasses the No Information Rate—what we'd expect if we only guessed the most common rating (5-stars), which appears in 46.50% of the data. The Kappa statistic of 0.285 supports this slight edge, indicating that the model does a bit better than just guessing the most frequent outcome, though the improvement is not substantial. Furthermore, McNemar's test, with a p-value below 0.05, validates that the model's predictions are statistically significantly different from random guessing. This affirms that the model's performance, while not highly accurate, still holds statistical significance.

Assuming independence of document features to classifications, Naive Bayes can be a robust method to estimate user ratings based on probabilities. However, some caveats present while estimating this model. The distribution of one-to-five-star reviews is heavily skewed towards 4 and 5 stars, thus making the prediction of these ratings far more accurate as shown by the confusion matrix relative to the other ratings. Also, the independence condition does not account for negation handling cases, especially when tokenising documents to unigrams, impeding the effectiveness of the model. Thus, the accuracy count may be improved using POS (part of speech) techniques, and higher n-grams.

Deployment

In the deployment phase, the end goal would be to integrate the predictive model into business workflows. A user-friendly dashboard should be created for stakeholders to easily access and interpret sentiment analysis results. A routine for monitoring and updating the model should be established to ensure the model's accuracy and relevance. Training materials should be provided for relevant teams, enabling effective use of the model's insights. Regular reviews and feedback collection would be key in aligning the model with business-specific objectives and refining its application for proper use.

Reflection

A significant challenge I faced in the project related to the vectorization of the text corpus, which produced errors due to memory constraints. While I was able to manipulate the corpus in the pre-processing stage, I struggled in further steps before I partitioned data for training and testing which I found daunting, as completing this task was essential for converting large sets of documents into a numerical format. To navigate this challenge, I adopted an iterative approach where I searched GitHub community forums to solve the problem and corrected subsequent errors regarding vectorisation on a feedback-loop basis. I solved the issue by restricting the dataset to a sample containing 90,000 observations, which quickened the prediction process. Although the trial-and-error stage was a lengthy process, I eventually managed to completely run the modified version of the code with the aid of programming contributors.

References

Ching, M.R.D. and de Dios Bulos, R., 2019, June. Improving restaurants' business performance using Yelp data sets through sentiment analysis. In Proceedings of the 3rd International Conference on E-commerce, E-Business and E-Government (pp. 62-67).

Dey, L., Chakraborty, S., Biswas, A., Bose, B. and Tiwari, S., 2016. Sentiment analysis of review datasets using naive bayes and k-nn classifier.

Lackermair, G., Kailer, D. and Kanmaz, K., 2013. Importance of online product reviews from a
consumer’s perspective. Advances in economics and business, 1(1), pp.1-5.

Reddy, C.S.C., Kumar, K.U., Keshav, J.D., Prasad, B.R. and Agarwal, S., 2017, November. Prediction of star ratings from online reviews. In TENCON 2017-2017 IEEE Region 10 Conference (pp. 1857-1861). IEEE.

Shah, A., Kothari, K., Thakkar, U. and Khara, S., 2020. User review classification and star rating prediction by sentimental analysis and machine learning classifiers. In Information and Communication Technology for Sustainable Development: Proceedings of ICT4SD 2018 (pp. 279-288). Springer Singapore.