Отчетный проект по дисциплине “Введение в технологии искусственного интеллекта”

Выполнила:
студентка 1 курса ММвЭ
Погибельная Елена Андреевна

Шаг 1. Для работы программы R 4.5.2, необходимо предоставить доступ к базе данных с отзывами. После предоставления происходит чтение

R Console

```
>
> # ========== 1. ЧТЕНИЕ ДАННЫХ ИЗ WORD-ФАЙЛА ==========
> file_path <- "C:/Users/Елена/Desktop/датасет. Погибельная Е.А..docx"
>
> # Чтение документа Word
> doc <- read_docx(file_path)
>
> # Извлечение таблицы из документа
> tables <- docx_summary(doc)
> content_table <- tables %>%
+   filter(content_type == "table cell") %>%
+   select(text) %>%
+   pull(text)
>
> # Преобразование в структурированный датафрейм
> # Предполагаем, что таблица имеет структуру: №, Текст, Ссылка, Метаданные
> # Нужно адаптировать под реальную структуру вашего файла
>
> # Альтернативный подход: если файл сохранен как CSV или другой формат
> # data <- read.csv("ваш_файл.csv", encoding = "UTF-8")
>
```

Шаг 2. После обработки на данных из документа, необходимо прописать ряд признаков для оценки отзывов

R Console

```
> # ========== 3. ПРЕДОБРАБОТКА ТЕКСТА И ИЗВЛЕЧЕНИЕ ПРИЗНАКОВ ==========
>
> # Функция для извлечения признаков из текста
> extract_features <- function(text) {
+   # Базовые признаки
+   text_length <- nchar(text)
+   word_count <- length(strsplit(text, "\\s+")[[1]])
+
+   # Эмоциональные признаки
+   exclamation_count <- str_count(text, "!")
+   question_count <- str_count(text, "\\?")
+   positive_words <- str_count(tolower(text), "отличн|хорош|доволен|рекомендую$
+   negative_words <- str_count(tolower(text), "плох|ужас|разочарован|недоволен$
+
+   # Признаки конкретности
+   has_price <- str_detect(text, "\\d+\\s*(руб|р\\.|₽)")
+   has_technical <- str_detect(tolower(text), "размер|вес|цвет|характеристик|п$
+   has_comparison <- str_detect(tolower(text), "по сравнению|лучше|хуже|чем")
+
+   # Признаки структуры
+   has_details <- word_count > 20  # Детальный отзыв обычно длиннее
+   has_rating <- str_detect(text, "\\d+/10|\\d+\\s*звезд|оценк")
+
```

Шаг 2. После обработки на данных из документа, необходимо прописать ряд признаков для оценки отзывов

```
+   return(data.frame(
+     text_length,
+     word_count,
+     exclamation_count,
+     question_count,
+     positive_words,
+     negative_words,
+     sentiment_score = positive_words - negative_words,
+     has_price = as.numeric(has_price),
+     has_technical = as.numeric(has_technical),
+     has_comparison = as.numeric(has_comparison),
+     has_details = as.numeric(has_details),
+     has_rating = as.numeric(has_rating)
+   ))
+ }
>
> # Применяем функцию к каждому тексту
> features_list <- lapply(data$Text, extract_features)
> features_df <- do.call(rbind, features_list)
>
> # Добавляем целевую переменную
> features_df$Helpful <- as.factor(data$Helpful)
```

Шаг 3. Происходит обучение классификатора на основе тестовых данных

```
> # ========== 5. ОБУЧЕНИЕ КЛАССИФИКАТОРА ==========
>
> # Обучение модели случайного леса
> rf_model <- randomForest(
+   Helpful ~ .,
+   data = train_data,
+   ntree = 100,
+   importance = TRUE
+ )
>
> # Предсказание на тестовых данных
> predictions <- predict(rf_model, test_data)
>
> # Оценка модели
> confusionMatrix(predictions, test_data$Helpful)
Confusion Matrix and Statistics

          Reference
Prediction Да Нет
       Да  10   4
       Нет  4   1
```

Шаг 3. Происходит обучение классификатора на основе тестовых данных

```
Confusion Matrix and Statistics

          Reference
Prediction Да Нет
       Да  10   4
       Нет  4   1

               Accuracy : 0.5789
                 95% CI : (0.335, 0.7975)
    No Information Rate : 0.7368
    P-Value [Acc > NIR] : 0.9606

                  Kappa : -0.0857

 Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.7143
            Specificity : 0.2000
         Pos Pred Value : 0.7143
         Neg Pred Value : 0.2000
             Prevalence : 0.7368
         Detection Rate : 0.5263
   Detection Prevalence : 0.7368
      Balanced Accuracy : 0.4571

       'Positive' Class : Да
```

Шаг 4. Происходит ввод переменных признаков важности для оценки отзыва

R Console

```
> # ========== 6. ВАЖНОСТЬ ПРИЗНАКОВ ==========
> importance_df <- as.data.frame(importance(rf_model))
> importance_df$Feature <- rownames(importance_df)
>
> # Визуализация важности признаков
> ggplot(importance_df, aes(x = reorder(Feature, MeanDecreaseGini), y = MeanDec$
+   geom_bar(stat = "identity", fill = "steelblue") +
+   coord_flip() +
+   labs(title = "Важность признаков для определения полезности отзыва",
+        x = "Признак",
+        y = "Важность (Mean Decrease Gini)") +
+   theme_minimal()
>
```

Шаг 5. Анализируем самые частые слова и создаем матрицы частоты

R Console

```
> # ========== 7. АНАЛИЗ ТЕКСТОВ ==========
>
> # Анализ самых частых слов
> corpus <- Corpus(VectorSource(data$Text))
> corpus <- tm_map(corpus, content_transformer(tolower))
Предупреждение:
В tm_map.SimpleCorpus(corpus, content_transformer(tolower)) :
  transformation drops documents
> corpus <- tm_map(corpus, removePunctuation)
Предупреждение:
В tm_map.SimpleCorpus(corpus, removePunctuation) :
  transformation drops documents
> corpus <- tm_map(corpus, removeNumbers)
Предупреждение:
В tm_map.SimpleCorpus(corpus, removeNumbers) :
  transformation drops documents
> corpus <- tm_map(corpus, removeWords, stopwords("russian"))
Предупреждение:
В tm_map.SimpleCorpus(corpus, removeWords, stopwords("russian")) :
  transformation drops documents
> corpus <- tm_map(corpus, stripWhitespace)
Предупреждение:
В tm_map.SimpleCorpus(corpus, stripWhitespace) :
  transformation drops documents
```

Шаг 5. Анализируем самые частые слова и создаем матрицы частоты

```
> # Создание матрицы частот
> dtm <- DocumentTermMatrix(corpus)
> freq <- colSums(as.matrix(dtm))
>
> # Облако слов для полезных отзывов
> helpful_texts <- data$Text[data$Helpful == "Да"]
> if(length(helpful_texts) > 0) {
+   corpus_helpful <- Corpus(VectorSource(helpful_texts))
+   corpus_helpful <- tm_map(corpus_helpful, content_transformer(tolower))
+   corpus_helpful <- tm_map(corpus_helpful, removePunctuation)
+   corpus_helpful <- tm_map(corpus_helpful, removeNumbers)
+   corpus_helpful <- tm_map(corpus_helpful, removeWords, stopwords("russian"))
+
+   wordcloud(corpus_helpful, max.words = 50,
+             colors = brewer.pal(8, "Dark2"),
+             main = "Частые слова в полезных отзывах")
+ }
```

Шаг 6. Производится статический анализ относительно эмоциональной окраски и длины отзыва по полезности

R Console

```
> # ========== 8. СТАТИСТИЧЕСКИЙ АНАЛИЗ ==========
>
> # Сравнение средних значений признаков
> stats_summary <- features_df %>%
+   group_by(Helpful) %>%
+   summarise(
+     avg_length = mean(text_length),
+     avg_words = mean(word_count),
+     avg_exclamation = mean(exclamation_count),
+     avg_positive = mean(positive_words),
+     avg_negative = mean(negative_words),
+     n = n()
+   )
>
> print(stats_summary)
# A tibble: 2 × 7
  Helpful avg_length avg_words avg_exclamation avg_positive avg_negative     n
  <fct>        <dbl>     <dbl>           <dbl>        <dbl>        <dbl> <int>
1 Да            91.3      11.4           0.356         2.14         1.05    73
2 Нет           87.1      10.8           0.222         1.85         1.19    27
>
```

Шаг 6. Производится статический анализ относительно эмоциональной окраски и длины отзыва по полезности

```
> # Визуализация различий
> ggplot(features_df, aes(x = Helpful, y = word_count, fill = Helpful)) +
+   geom_boxplot() +
+   labs(title = "Длина отзыва (количество слов) по полезности",
+        x = "Полезность",
+        y = "Количество слов") +
+   theme_minimal()
>
> ggplot(features_df, aes(x = Helpful, y = sentiment_score, fill = Helpful)) +
+   geom_boxplot() +
+   labs(title = "Эмоциональная окраска отзывов",
+        x = "Полезность",
+        y = "Сентимент-скор (положительные - отрицательные слова)") +
+   theme_minimal()
>
```

Шаг 7.Сохранение результатов с выводом сводки и построением графика

```
> # ========== 9. СОХРАНЕНИЕ РЕЗУЛЬТАТОВ ==========
>
> # Сохраняем предсказания
> results <- data.frame(
+   ID = data$ID[-train_index],
+   Actual = test_data$Helpful,
+   Predicted = predictions,
+   Probability = predict(rf_model, test_data, type = "prob")[,2]
+ )
>
> write.csv(results, "C:/Users/Елена/Desktop/результаты_анализа_отзывов.csv",
+           row.names = FALSE, fileEncoding = "UTF-8")
>
> # Сохраняем модель
> saveRDS(rf_model, "C:/Users/Елена/Desktop/модель_классификации.rds")
>
```

Шаг 7.Сохранение результатов с выводом сводки и построением графика

```
> # ========== 10. ВЫВОД СВОДКИ ==========
> cat("Анализ завершен!\n")
Анализ завершен!
> cat("Количество полезных отзывов в данных:", sum(data$Helpful == "Да"), "\n")
Количество полезных отзывов в данных: 73
> cat("Количество неполезных отзывов:", sum(data$Helpful == "Нет"), "\n")
Количество неполезных отзывов: 27
> cat("Точность модели на тестовых данных:",
+     round(mean(predictions == test_data$Helpful) * 100, 2), "%\n")
Точность модели на тестовых данных: 57.89 %
>
> # Самые важные признаки
> cat("\nТоп-5 важных признаков:\n")

Топ-5 важных признаков:
> top_features <- importance_df[order(-importance_df$MeanDecreaseGini), ][1:5, ]
> print(top_features[, c("Feature", "MeanDecreaseGini")])
                      Feature MeanDecreaseGini
text_length         text_length         4.957943
negative_words   negative_words         3.558014
word_count           word_count         3.521338
positive_words   positive_words         2.969621
sentiment_score sentiment_score         2.490542
```

Шаг 7.Сохранение результатов с выводом сводки и построением графика

На основе проведенного анализа можно рассмотреть полезность или ее отсутствие относительно датасета с оценкой на количество слов в каждом из отзывов