Project Report

STAT 380 Section 004

Gianna DeLorenzo & Melissa Kim

1 Patient Satisfaction Analysis with Drug Reviews

Patient experiences with medications is crucial when it comes to the healthcare domain. These drug reviews give valuable insights into aspects such as side effects, drug effectiveness, and overall patient satisfaction. Our project explores the ***Drug Reviews*** dataset from the UC Irvine Repository.

In this report, we will be exploring the following **research question**:

- **Can machine learning tasks predict the overall satisfaction of the patients with a particular class of drugs for a given medical condition?**

This question explores the patterns with machine learning tasks and their effectiveness on predicting the satisfaction of patients based on a given class of drug.

We will first be presenting the background of the dataset and related work. Then, we will be describing the data and preprocessing. Next, we will be going into detail about the methodology. Then, we will be explaining the results of each machine learning model. Then, we will be explaining our interpretations of each machine learning model. Finally, we will close up the report with a discussion of our findings.

2 Background & Related Work

The ***Drug Reviews*** dataset provides a useful source of information beyond just the structured clinical trial data, offering insights into the real-world drug effectiveness, side effects, and the overall patient satisfaction. The Health and Medicine field usually leverages techniques from natural languages processing (NLP) and machine learning in order to get the information from patient's textual data and understand it. Prior research has explored the identification of adverse drug reactions using NLP and the analysis of patient sentiments regarding drug experiences.

The first piece of related work is the introductory paper for the Drug Reviews dataset is titled **Aspect-Based Sentiment Analysis of Drug Reviews Applying Cross-Domain and Cross-Data Learning** [1] by F. Gräßer, S. Kallumadi, H. Malberg, and S. Zaunseder. This research paper demonstrated the applicability of sentiment analysis to drug reviews. This includes classifying the

[1] Gräßer, F., Kallumadi, S., Malberg, H., & Zaunseder, S. (2018). Aspect-Based Sentiment Analysis of Drug Reviews Applying Cross-Domain and Cross-Data Learning. Proceedings of the 2018 International Conference on Digital Health.

overall patient satisfaction and investigating the challenges with data limitations and transferring models across various domains and data sources.

Furthermore, the application of machine learning with predicting patient sentiments/ratings and satisfaction from drug reviews is an active area. The second piece of related work is a research paper that was highly influencial to the production of the introductory paper. This paper is titled **Predicting Sentiments of Users About Medical Treatments using Pre-trained Large Language Models** [2] by A. Hassan, J. Bagherzadeh, M. Khorasani, and A. Sorayaie Azar. This research paper explores the usage of various machine learning techniques including traditional models and large language models (LLMs). These models were used to predict the ratings on drug reviews and achieve high accuracies in classification tasks.

3 Data Description & Preprocessing

3.1 Provenance

The dataset used for this analysis, the ***Drug Reviews*** dataset from the UC Irvine Repository [3], explores reviews from patients on specific drugs along with related medical conditions. The reviews and ratings are grouped into reports on the three aspects being the benefits, side effects and overall comment. The data was originally obtained by crawling online pharmaceutical review sites. This dataset is split into 2 tsv files with one being a train set (75%) and the other one being a test set (25%). It contains 4143 instances and 8 features.

Source: Drug Effects Data.

3.2 Variable Description

The response variable is the patient's overall ratings of the drugs with the negative reviews from **0-6** and the positive reviews from **7-10**.

An example of a **high review's** (Rating = 10) textual review includes the following:

- **urlDrugName**: Xanax
- **Rating**: 10
- **benefitsReview**: This simply just works fast and without any of the nasty side effects of SSRI medicines…
- **sideEffectsReview**: I really don't have any side effects other than the mild but very tolerable issue of taking it 4-5 times a day which isn't an effect but just the way you need to take this…
- **commentsReview**: I first started taking this at 3 times per day with .25 mg pills. I was advised not to take as needed with my panic attacks as that would reinforce the idea of taking a pill when it's needed and not help me work around my anxiety…

An example of a **low review's** (Rating = 1) textual review includes the following:

[2] Hassan, A., Mohasefi, J.B., & Azar, S. (2025). Predicting Sentiments of Users about Medical Treatments using Pre-trained Large Language Models. Journal of Information Systems Engineering and Management.

[3] Kallumadi, S. & Grer, F. (2018). Drug Reviews (Druglib.com) [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C55G6J.

- **urlDrugName**: Claritin
- **Rating**: 1
- **benefitsReview**: None - did nothing to help allergies. I just had a dryer, more painful version of my allergies with new allergies/irritations developing on top of my original ones...
- **sideEffectsReview**: I had some horrifying mental and physical side effects. ruined a year of my life. read this - it mentions everything I dealt with: http://www.askapatient.com/viewrating.aspdrug=196
- **commentsReview**: Took one 10 mg pill nightly...

The variables that we used in our analysis include the following:

Table 1: Variables used in Analysis

Variable	Type	Explanation
reviewID	Integer	Review ID
urlDrugName	Categorial	Name of drug
rating	Integer	Name of condition
effectiveness	Categorical	10 star patient rating
sideEffects	Categorical	5 step side effect rating
condition	Categorical	5 step effectiveness rating

Table 1

This provides a clear overview of the variable names, data types, and what each variable represents and means.

3.3 Data Cleaning and Preprocessing

Contrary to the UC Irvine Repository stating "No Missing Values" for the dataset, there were missing values observed in the text review field including benefitsReview, sideEffectsReview, and commentsReview. We initially used glimpse() to inspect the train and test datasets to possibly find missing values. It did not show any missing values in either dataset, but it did provide us with useful information about the variables such as the data types and structure.

```
Rows: 1,036
Columns: 9
$ ...1              <dbl> 1366, 3724, 3824, 969, 696, 1380, 45, 1939, 2576, 10~
$ urlDrugName       <chr> "biaxin", "lamictal", "depakene", "sarafem", "accuta~
$ rating            <dbl> 9, 9, 4, 10, 10, 2, 8, 10, 10, 1, 3, 9, 6, 10, 5, 5,~
$ effectiveness     <chr> "Considerably Effective", "Highly Effective", "Moder~
$ sideEffects       <chr> "Mild Side Effects", "Mild Side Effects", "Severe Si~
$ condition         <chr> "sinus infection", "bipolar disorder", "bipolar diso~
$ benefitsReview    <chr> "The antibiotic may have destroyed bacteria causing ~
$ sideEffectsReview <chr> "Some back pain, some nauseau.", "Drowsiness, a bit ~
$ commentsReview    <chr> "Took the antibiotics for 14 days. Sinus infection w~
```

```
Rows: 3,107
Columns: 9
$ ...1              <dbl> 2202, 3117, 1146, 3947, 1951, 2372, 1043, 2715, 1591~
$ urlDrugName       <chr> "enalapril", "ortho-tri-cyclen", "ponstel", "prilose~
$ rating            <dbl> 4, 1, 10, 3, 2, 1, 9, 10, 10, 1, 7, 8, 8, 9, 4, 8, 6~
$ effectiveness     <chr> "Highly Effective", "Highly Effective", "Highly Effe~
$ sideEffects       <chr> "Mild Side Effects", "Severe Side Effects", "No Side~
$ condition         <chr> "management of congestive heart failure", "birth pre~
$ benefitsReview    <chr> "slowed the progression of left ventricular dysfunct~
$ sideEffectsReview <chr> "cough, hypotension , proteinuria, impotence , renal~
$ commentsReview    <chr> "monitor blood pressure , weight and asses for resol~
```

Only the drug test dataset outputted with a missing value being present, so we figured out which column contains it.

```
[1] FALSE
```

```
[1] TRUE
```

Then, we figured out that the commentsReview column contains 1 missing value, but it is not specifically an N/A column. It states "N/A currently," so it did not seem necessary to take care of this.

```
             ...1       urlDrugName            rating     effectiveness 
                0                 0                 0                 0 
      sideEffects         condition    benefitsReview sideEffectsReview 
                0                 0                 0                 0 
   commentsReview 
                1 
```

Beyond handling the missing values, preprocessing steps were tailored based on the needs of the various machine learning models.

3.4 Exploratory Data Analysis

As we explore the UCI Drug Reviews dataset, we focus on the patterns, trends, relationships between variables, and significant connections to our research question. Our goals include gaining insight on patient reviews and experiences, applying our understanding of machine learning tasks, and implementing model training and evaluation.

Below are two histograms, one for the Train dataset and one for the Test dataset.

Figure 1: Table 2

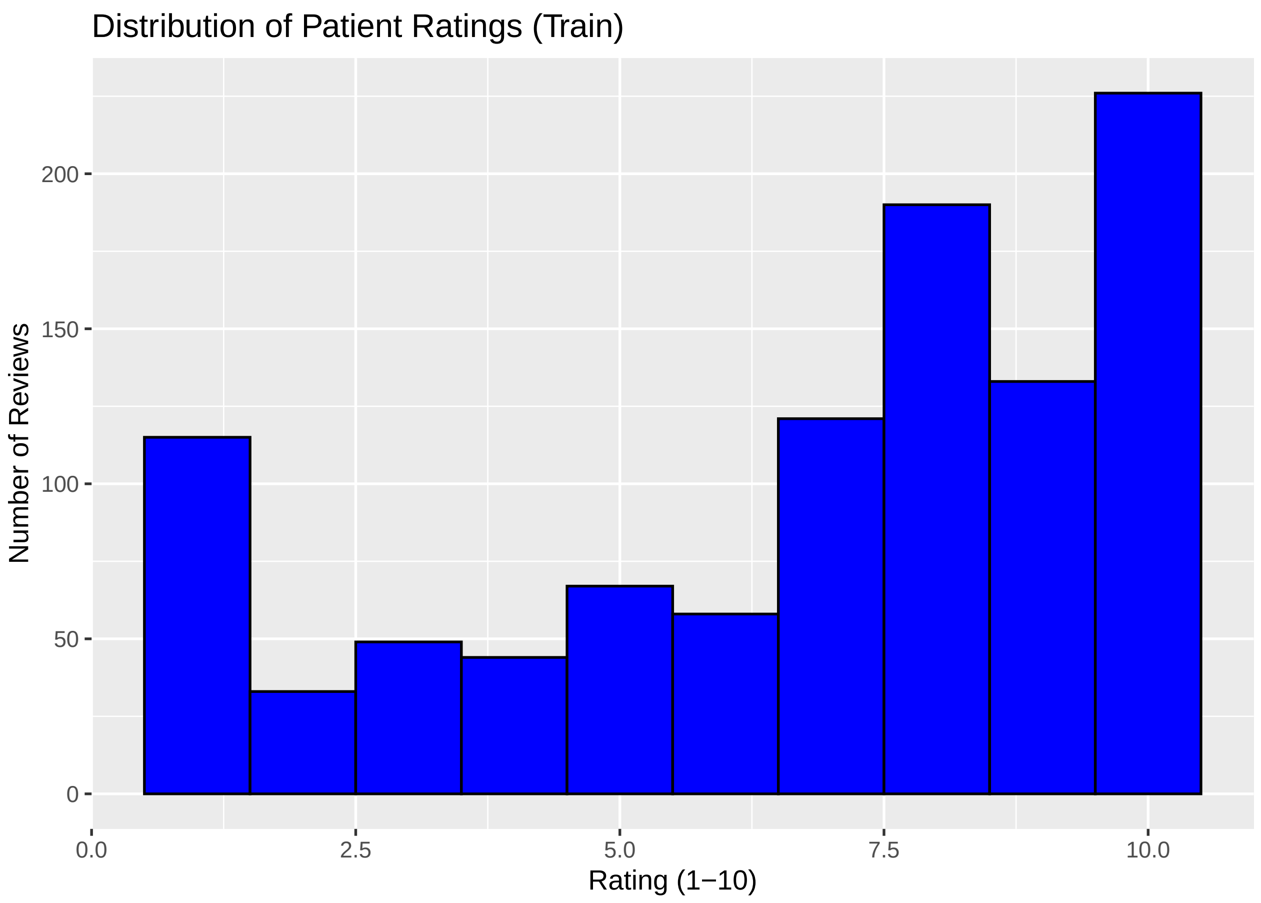
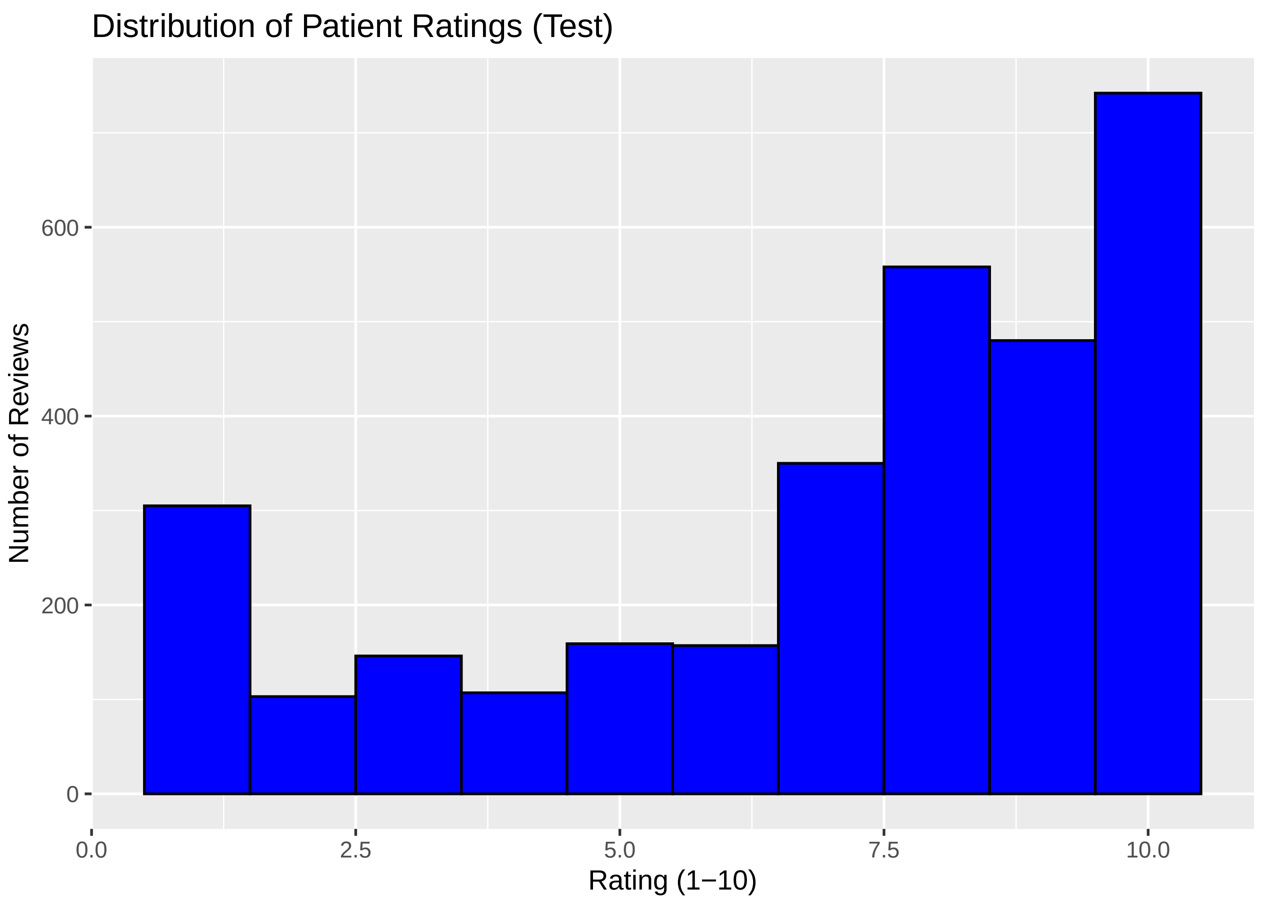


Figure 2: Table 2



Both datasets display left-skewed histograms with the distribution of patient ratings as there are more data points on the right side.

4 Methodology

We chose to use four machine learning models which are **Logistic Regression**, **Neural Networks**, **K-Nearest Neighbors (KNN)**, and **K-means Clustering**. We want to predict the overall satisfaction of the patients with a particular class of drugs for a given medical condition, so these machine learning models were the best fit.

Logistic regression is a supervised learning model that is used for binary response variables. It models the probability that the response variable will take on a specific level. In our case, the response variable is the patient's overall ratings of the drugs (positive/negative reviews) which is relevant to the satisfaction of the patients.

Neural Networks is a supervised learning model that is used to predict a more nuanced outcome. It goes beyond just the positive or negative sentiment and involves predicting the original 10-point rating scale of the drug reviews.

K-Nearest Neighbors (KNN) is a supervised learning model that is used for classification. It can be used for regression and classification tasks. It aims to classify the sentiment of a new drug review based on similar past patient reviews. The main idea is that to predict the response for a new observation, you find the k observations in the Train dataset that are nearest to the new observation.

K-means Clustering

5 Model Results

5.1 Logistic Regression: Predicting Positive vs. Negative Reviews

Can we predict if a patient's review is positive or negative based on their feedback?

In this section, we explore the possibility of predicting the sentiment of a patient's review – whether positive or negative – based on their feedback about medications. Using a logistic regression model, a popular method in supervised machine learning for binary classification, patterns in patient reviews were analyzed in order to determine their overall sentiment. The main goal was to assess whether features like the name of the drug, condition being treated, and patient's perceived effectiveness of the medication could provide enough information to make accurate sentiment predictions.

The dataset used for this analysis consisted of 2 files: **drugLibTrain_raw.tsv** and **drugLibTest_raw.tsv**, which includes thousands of patient-generated entries about various aspects of their medication experiences. For data preparation, a new binary target variable named **sentiment** was created. Reviews with a rating of 7 or above were labeled as **positive (1)**, while those with ratings from 1 to 6 were labeled as **negative (0).** This threshold reflects a distinction between satisfied and dissatisfied users. Several categorical variables, like drug name and condition treated, were converted to factors and grouped using frequency-based lumping. For example, only the top 20 most common drug names and conditions were kept as individual factor levels, while less frequent ones were grouped under "Other". This reduced dimensionality and improved the model's ability to generalize.

To ensure consistency between training and testing datasets, factor levels in the test set were matched to those in the training set. Any rows in the test data with missing values in key features were removed to avoid complications during model prediction. Afterwards, a logistic regression model was fit using the glm() function in R with the binomial family specified. The formula for the model was:

```
sentiment \~ urlDrugName + condition + effectiveness
```

The model used these 3 features to estimate the probability that a given review was positive. It then generated predicted probabilities on the cleaned test dataset, with a threshold of 0.5. A confusion matrix was also generated to summarize how well the predicted labels matched the actual sentiment values. The model is shown below:

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0  548  121
         1  305 1716

               Accuracy : 0.8416
                 95% CI : (0.8273, 0.8552)
    No Information Rate : 0.6829
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6119

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

            Sensitivity : 0.6424
            Specificity : 0.9341
         Pos Pred Value : 0.8191
         Neg Pred Value : 0.8491
             Prevalence : 0.3171
         Detection Rate : 0.2037
   Detection Prevalence : 0.2487
      Balanced Accuracy : 0.7883

       'Positive' Class : 0
```

5.2 Neutral Network: Predicting Ratings with Nuance

Can we predict a more nuanced outcome of the review based on patient attributes, dosage, and drug features?

This section investigates whether a more detailed prediction of patient review ratings can be achieved using the neural network supervised learning model. Unlike binary sentiment classification, which only determines if a review is positive or negative, this task aims to predict the exact numerical rating on a 1-10 scale given by patients. The central question is whether structured patient attributes, like the name of the drug, treated condition, and self-reported effectiveness, contain enough information to support such predictions. Applying a neural network model may help capture the complex, nonlinear relationships that could influence how patients rate their experiences with medications.

The dataset used was the **drugLibTrain_raw.tsv** file, from which a subset of relevant variables – **rating, urlDrugName, condition,** and **effectiveness** – was selected for analysis. All rows containing missing values were dropped to ensure model integrity. The target variable, **rating**, was converted into a factor to reflect its categorical nature, and the categorical predictors were simplified by lumping less frequent values into "Other" categories. The top 20 drug names and conditions, and top 5 effectiveness levels, were retained to prevent overfitting. This preprocessing ensured a manageable number of factor levels, which is especially important in neural network models with limited hidden units.

The cleaned dataset was split into training and testing sets using an 80/20 stratified split, preserving the distribution of review scores across both sets. A feedforward neural network model was trained using the nnet package in R. The network architecture included a single hidden layer with 5 neurons, and the model was trained over 200 iterations. The predictors used were **urlDrugName, condition,** and **effectiveness**, all treated as categorical inputs. After training, the model was applied to the testing set, and predictions were evaluated against the actual review scores using a confusion matrix and class-wise performance metrics.

```
Confusion Matrix and Statistics

          Reference
Prediction  1  2  3  4  5  6  7  8  9 10
        1  12  3  4  2  1  2  1  0  0  0
        2   0  0  0  0  0  0  0  0  0  0
        3   0  0  0  0  0  0  0  0  0  0
        4   1  0  2  1  0  0  0  0  0  0
        5   0  0  0  0  0  0  0  0  2  0
        6   1  1  0  0  0  1  5  1  1  0
        7   3  2  1  1  5  4  5 10  4  1
        8   3  0  1  1  3  3  7 18  8  3
        9   0  0  0  0  0  0  0  1  2  0
        10  3  0  1  3  4  1  6  8  9 41

Overall Statistics

               Accuracy : 0.3941
                 95% CI : (0.3264, 0.4649)
    No Information Rate : 0.2217
    P-Value [Acc > NIR] : 2.428e-08

                  Kappa : 0.2724

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5 Class: 6
Sensitivity           0.52174  0.00000  0.00000 0.125000 0.000000 0.090909
Specificity           0.92778  1.00000  1.00000 0.984615 0.989474 0.953125
Pos Pred Value        0.48000      NaN      NaN 0.250000 0.000000 0.100000
Neg Pred Value        0.93820  0.97044  0.95567 0.964824 0.935323 0.948187
Prevalence            0.11330  0.02956  0.04433 0.039409 0.064039 0.054187
Detection Rate        0.05911  0.00000  0.00000 0.004926 0.000000 0.004926
Detection Prevalence  0.12315  0.00000  0.00000 0.019704 0.009852 0.049261
Balanced Accuracy     0.72476  0.50000  0.50000 0.554808 0.494737 0.522017
```

```
                     Class: 7 Class: 8 Class: 9 Class: 10
Sensitivity           0.20833  0.47368 0.076923    0.9111
Specificity           0.82682  0.82424 0.994350    0.7785
Pos Pred Value        0.13889  0.38298 0.666667    0.5395
Neg Pred Value        0.88623  0.87179 0.880000    0.9685
Prevalence            0.11823  0.18719 0.128079    0.2217
Detection Rate        0.02463  0.08867 0.009852    0.2020
Detection Prevalence  0.17734  0.23153 0.014778    0.3744
Balanced Accuracy     0.51757  0.64896 0.535637    0.8448
```

5.3 K-Nearest Neighbors (KNN): Predicting Binary Classification

Can we classify the sentiment of a new review based on sentiments of similar past reviews?

In this section, we explore the sentiment of a new review based on similar past reviews with the KNN model as the supervised learning algorithm.

The dataset used was the **drugLibTrain_raw.tsv** file from which a binary sentiment variable was created. This variable assigns a value of 1 (positive sentiment) if the original rating is greater than or equal to 7, and 0 (negative sentiment) if the rating is less than 7. Then, it converts into a factor with explicit levels of 0 and 1 which is essential for classification tasks. The most frequent drugs and conditions in the train data are identified and counted. They are sorted in descending order with the top 10 values. These 2 lists are used to simplify the drug name and condition features to avoid overcrowding. It helps to reduce the dimensionality and complexity of the categorial features.

The dataset is split into training and testing sets in order to evaluate the KNN model's performance on the unseen data. It places 80% into the train set and 20% into the test set. A stratified split is then performed to attempt to keep the proportion of sentiment values around the same in the train and test sets. A design matrix is created for the train and test sets while converting the top drugs and conditions into dummy variables. Integrating -1 ensures that an intercept column is not included in the matrix. Lastly, the classification model is applied with the KNN function before outputted the confusion matrix to evaluate the predictions.

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0   7   7
         1  66 127

               Accuracy : 0.6473
                 95% CI : (0.5781, 0.7123)
    No Information Rate : 0.6473
    P-Value [Acc > NIR] : 0.5318

                  Kappa : 0.0535

 Mcnemar's Test P-Value : 1.134e-11
```

```
            Sensitivity : 0.09589
            Specificity : 0.94776
         Pos Pred Value : 0.50000
         Neg Pred Value : 0.65803
             Prevalence : 0.35266
         Detection Rate : 0.03382
   Detection Prevalence : 0.06763
      Balanced Accuracy : 0.52183

       'Positive' Class : 0
```

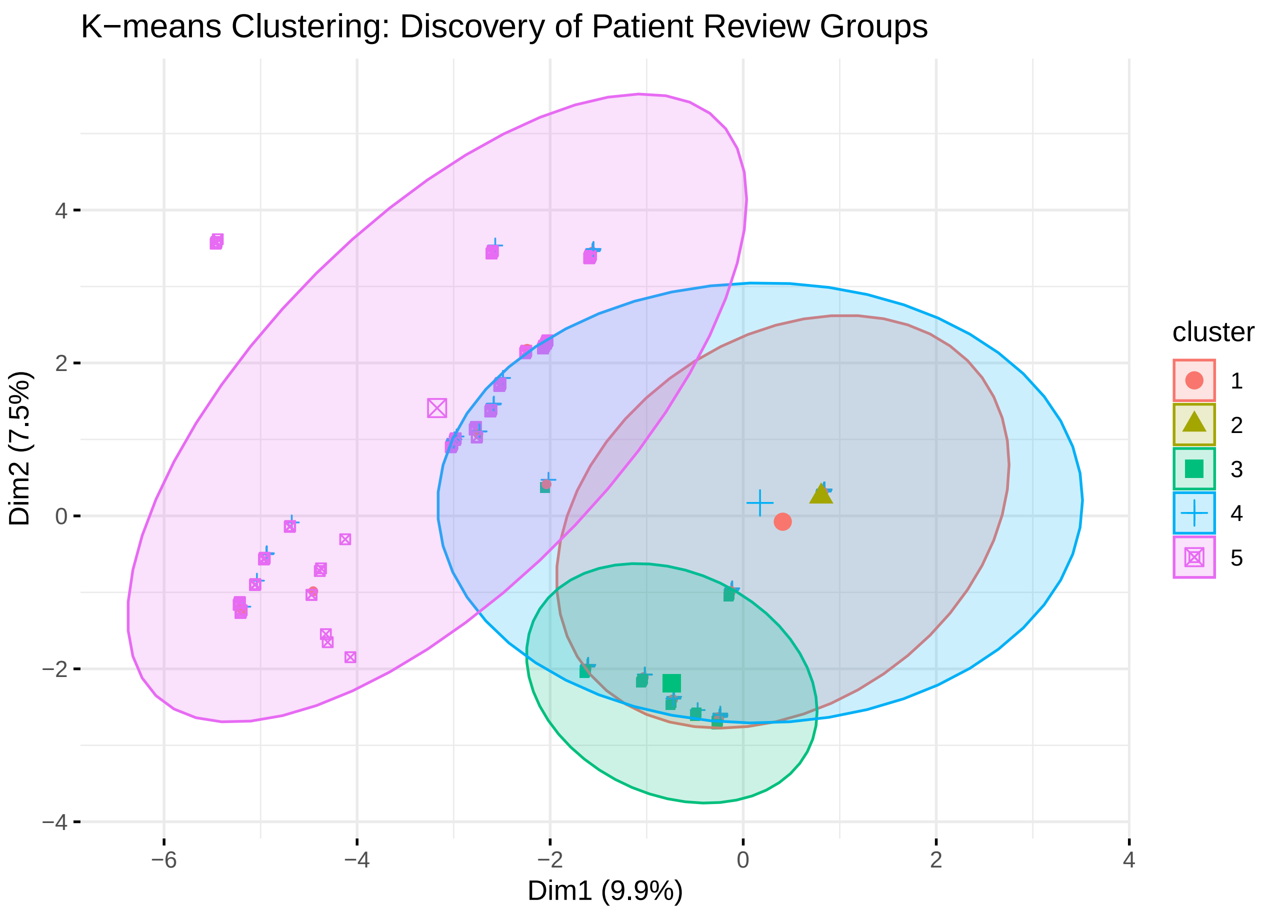
5.4 K-means Clustering: Discovery of Patient Review Groups

Can we discover hidden patterns in drug reviews without using labels?

In this section, we explore the underlying patterns in patient satisfaction. We applied K-means clustering to a subset of structured features including the rating, drug name, and condition.

The dataset used was the **drugLibTrain_raw.tsv** file from which we wrangled the data and selected features for the K-means model. Missing values were filtered out of the raw train data. We grouped less frequent occurrences into a category titled "Other" utilizing the top drugs and conditions defined prior. It selects the rating, top_drugs, and top_conditions variables that will be used as the input features for the k-means clustering algorithm. Model.matrix is used to perform one-hot encoding and creating dummy variables for each level of top_drugs and top_conditions. The numerical rating variable is then processed using scale() and performing standardization. After scaling, its combined (cbind) with the encoded categorical variables to create kmeans_input. Setting the seed then ensures the initialization of the algorithm being reproducible. The core k-means algorithm is run and the cluster assignments are added as a new column. The code then visualizes the clustering results.

Figure 3: Table 8



6 Model Interpretations

6.1 Logistic Regression Model

According to the confusion matrix, out of 3107 test reviews, 1716 were true positives (correctly predicted positive reviews), and 548 were true negatives (correctly predicted negative reviews). However, there were also 305 false positives and 121 false negatives. The model's overall **accuracy** was 82.85%, indicating that it correctly predicted sentiment in over 4 out of 5 cases. This accuracy significantly outperformed the **No Information Rate** of 64.71%, which reflects the accuracy that results from predicting the majority class (positive sentiment) every time.

Additional performance metrics provided further insight into the model's strengths and limitations. The **specificity**, or true positive rate, for classifying positive reviews was particularly high at 91.79%, demonstrating that the model excelled at recognizing satisfied patients. The **precision** for the negative class was also strong at 81.54%, meaning most reviews predicted as negative were actually negative. However, the **sensitivity** for negative reviews was lower at 66.46%, indicating that about of actual negative reviews were missed, often misclassified as positive. This imbalance suggests that the model may be biased toward the more frequent positive class, a common issue in sentiment classification where one class dominates the dataset. Nonetheless, the **balanced**

accuracy, or average of sensitivity and specificity, was 79.13%, indicating good performance across both classes.

The logistic regression model achieved an overall accuracy of 82.85%, which significantly outperforms the no-information rate of 64.71% (p < 2.2e-16). It performs well in identifying positive reviews (specificity = 91.8%) but is less sensitive to negative reviews (sensitivity = 66.5%). The balanced accuracy of 79.1% confirms that the model maintains reasonable performance across both classes. The confusion matrix and McNemar's test (p = 3.18e-06) further indicate that the model is not biased toward one class and captures meaningful patterns in patient review sentiment. This result, coupled with high specificity and balanced accuracy, supports the conclusion that machine learning methods can be used to reliably classify the sentiment of patient reviews, making them valuable tools for summarizing large-scale patient feedback in clinical settings.

6.2 Neural Networks Model

The neural network model achieved an overall **accuracy** of 39.8%, indicating that it correctly predicted the exact review rating in just under four out of ten cases. While this result is significantly better than the **No Information Rate** (23.95%), it still reflects modest predictive performance for a ten-class classification problem. The model performed particularly well on the most frequent class—rating 10—correctly identifying nearly 90% of these cases, which heavily influenced the overall accuracy. However, such strong performance on a single class also highlights the model's reliance on class frequency over a balanced understanding of all outcomes.

A closer look at class-specific metrics reveals that the model struggled to generalize beyond the most common ratings. Classes such as **2, 5, and 9** had extremely low or zero recall, meaning they were rarely, if ever, predicted. Other mid-range scores like **3, 4, and 6** had recall rates below 15%, paired with low precision, indicating that the model could not effectively learn the distinctions between similar ratings. Only a few classes—**rating 1** (sensitivity = 50.8%)and **rating 8** (sensitivity = 48.6%)—showed moderate performance. This skewed behavior is typical in highly imbalanced datasets where dominant classes overwhelm signals for underrepresented ones.

Although the model's architecture was likely enough to avoid overfitting, its limited depth and capacity may have constrained its ability to capture relationships between inputs and outputs. Additionally, the structured features—drug name, condition, and effectiveness—do not provide enough detail to distinguish between numerically close ratings (e.g., a 5 vs. a 6). Without richer data or more informative inputs, the neural network had little basis for making nuanced predictions across a wide range of review scores.

In conclusion, the neural network model demonstrated partial effectiveness, especially in predicting high-frequency ratings like 10, but performed poorly across the rest of the rating scale. The low recall and precision for most classes suggest that the model failed to learn adequate class-specific patterns, likely due to a combination of class imbalance, limited feature diversity, and insufficient complexity in the model. Despite achieving better-than-random performance, the model's real-world utility is limited if it cannot reliably predict ratings across the full spectrum.

To enhance predictive accuracy, future work should address the class imbalance using resampling or different learning techniques. Additionally, the feature space should include more descriptive variables—such as dosage, side effects, or even natural language processing (NLP) features extracted from written reviews. With a richer dataset and more comprehensive model architecture, the system could better capture the subtle factors that influence patient ratings, leading to more reliable and informative predictions.

6.3 K-Nearest Neighbors (KNN) Model

According to the confusion matrix, out of 207 test reviews, 127 of them were true negatives and 7 of them were true positives. 7 of them were also false positives and 66 of them were false negatives. This means that 127 reviews were predicted negative and actually were negative, 7 reviews were predicted positive but actually were negative, 66 reviews were predicted negative but were actually positive, and 7 reviews were predicted positive and actually were positive.

The overall **accuracy** of the model is 0.6722 which means that it correctly classified about 64.73% of the reviews in the test set. There is a **No Information Rate** (NIR) of 0.6473 which represents the accuracy achieved predicting the majority class. On the other hand, the Kappa is very low at 0.0535 which represents only slight agreement between the KNN model's predictions and the actual sentiment. The sensitivity is low at 9.6%, but the specificity is very high at 94.78% meaning that it correctly identified most of the positive class. The model also has a 95% **Confidence Interval** for the accuracy meaning that if the experiment is repeated multiple times, the true accuracy would most likely fall within this range 95% of the time.

The KNN Model represents a highly imbalanced performance. While it is good at correctly identifying positive reviews with a high specificity, it performs poorly at identifying negative reviews with a low sensitivity.

6.4 K-means Clustering Model

The k-means clustering model displays 5 clusters that have reasonable separation between at least 3 of them. The ellipses around each of the clusters indicate the density and spread. Cluster 1-5 can be identified on the right-side legend. Clusters 1 and 2 are relatively compact while Clusters 3 and 4 display some overlap.

7 Discussion

Logistic Regression: excellent at identifying positive reviews, reasonably good at identifying negative reviews

Neural Networks: excellent at predicting common scores (ex. 10); struggled with rarer or mid-range ratings

KNN: correctly classified about 64.73% of the reviews in the test set. excellent at identifying the positive class, struggled with identifying negative reviews

K-means Clustering:

8 Conclusion

9 References

Gräßer, F., Kallumadi, S., Malberg, H., & Zaunseder, S. (2018). Aspect-Based Sentiment Analysis of Drug Reviews Applying Cross-Domain and Cross-Data Learning. Proceedings of the 2018 International Conference on Digital Health.

Hassan, A., Mohasefi, J.B., & Azar, S. (2025). Predicting Sentiments of Users about Medical Treatments using Pre-trained Large Language Models. Journal of Information Systems Engineering and Management.

Kallumadi, S. & Grer, F. (2018). Drug Reviews (Druglib.com) [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C55G6J.

10 Code Appendix

```
#Load necessary libraries
library(dplyr) #data manipulation
library(knitr) #table formats
library(readr)
library(kableExtra)
library(caret)
library(forcats)
library(nnet)
library(kknn)
library(tidyr)
library(broom)
library(ggcorrplot)

#Import UC Irvine test tsv files. (Initial Definition for All Code)
drug_train_tsv <- read_tsv("https://raw.githubusercontent.com/gkd5216/STAT380Project/refs/heads
drug_test_tsv <- read_tsv("https://raw.githubusercontent.com/gkd5216/STAT380Project/refs/heads/

# Summary Statistics grouping by Geographic Area.
variable_analysis <- data.frame(
  Variable = c("reviewID", "urlDrugName", "rating", "effectiveness", "sideEffects", "condition
  Type = c("Integer", "Categorial", "Integer", "Categorical", "Categorical", "Categorical"),
  Explanation = c(
    "Review ID",
    "Name of drug",
    "Name of condition",
    "10 star patient rating",
    "5 step side effect rating",
    "5 step effectiveness rating"
```

```r
  )
)

# Outputs Formatted Summary Table with Kable Styling tools
kable(
  variable_analysis,
  caption = "Variables used in Analysis"
  )

glimpse(drug_train_tsv)
glimpse(drug_test_tsv)
any(is.na(drug_train_tsv))
any(is.na(drug_test_tsv))
colSums(is.na(drug_test_tsv))
ggplot(drug_train_tsv, aes(x = rating)) +
  geom_histogram(binwidth = 1, fill = "blue", color = "black") +
  labs(title = "Distribution of Patient Ratings (Train)", x = "Rating (1-10)", y = "Number of I

ggplot(drug_test_tsv, aes(x = rating)) +
  geom_histogram(binwidth = 1, fill = "blue", color = "black") +
  labs(title = "Distribution of Patient Ratings (Test)", x = "Rating (1-10)", y = "Number of Re
# Create binary sentiment variable: 1 (positive w/ rating 7-10), 0 (negative w/ rating 1-6)

library(tidyverse)
library(readr)
library(caret)
library(forcats)

# Create binary sentiment variable: 1 (positive w/ rating 7-10), 0 (negative w/ rating 1-6)
train_data <- drug_train_tsv %>%
  mutate(sentiment = ifelse(rating >= 7, 1, 0))

test_data <- drug_test_tsv %>%
  mutate(sentiment = ifelse(rating >= 7, 1, 0))

train_data <- train_data %>%
  mutate(across(c(urlDrugName, condition, effectiveness), as.factor))

test_data <- test_data %>%
  mutate(across(c(urlDrugName, condition, effectiveness), as.factor))

train_data <- train_data %>%
  mutate(urlDrugName = fct_lump(urlDrugName, n = 20),
         condition = fct_lump(condition, n = 20),
         effectiveness = fct_lump(effectiveness, n = 5))

# Match levels in test data to training data
```

```r
test_data <- test_data %>%
  mutate(urlDrugName = fct_lump(urlDrugName, n = 20),
         condition = fct_lump(condition, n = 20),
         effectiveness = fct_lump(effectiveness, n = 5))

# Align factor levels
test_data$urlDrugName <- factor(test_data$urlDrugName, levels = levels(train_data$urlDrugName))
test_data$condition <- factor(test_data$condition, levels = levels(train_data$condition))
test_data$effectiveness <- factor(test_data$effectiveness, levels = levels(train_data$effectiv

# Remove rows with NAs
test_data_clean <- test_data %>%
  filter(!is.na(urlDrugName) & !is.na(condition) & !is.na(effectiveness))

# Logistic regression model
logit_model <- glm(sentiment ~ urlDrugName + condition + effectiveness,
                   data = train_data, family = "binomial")

# Predict on clean test data
test_pred <- predict(logit_model, newdata = test_data_clean, type = "response")
test_pred_class <- ifelse(test_pred > 0.5, 1, 0)

# Confusion matrix
confusionMatrix(factor(test_pred_class), factor(test_data_clean$sentiment))
library(tidyverse)
library(readr)
library(nnet)
library(caret)
library(dplyr)
library(forcats)

train_data <- drug_train_tsv %>%
  select(rating, urlDrugName, condition, effectiveness) %>%
  drop_na()

# Treat rating as factor
train_data$rating <- as.factor(train_data$rating)

train_data <- train_data %>%
  mutate(
    urlDrugName = fct_lump(urlDrugName, n = 20),
    condition = fct_lump(condition, n = 20),
    effectiveness = fct_lump(effectiveness, n = 5)
  )

# Split into training and validation sets
set.seed(42)
```

```r
split_index <- createDataPartition(train_data$rating, p = 0.8, list = FALSE)
train_set <- train_data[split_index, ]
valid_set <- train_data[-split_index, ]

# Train the neural network model
nn_model <- nnet(rating ~ urlDrugName + condition + effectiveness,
                 data = train_set,
                 size = 5,
                 maxit = 200,
                 trace = FALSE)

# Predict on validation set
nn_pred <- predict(nn_model, newdata = valid_set, type = "class")

nn_pred <- factor(nn_pred, levels = levels(valid_set$rating))
valid_rating <- factor(valid_set$rating, levels = levels(valid_set$rating))

# Evaluate model
confusionMatrix(nn_pred, valid_rating)

library(class)
library(FactoMineR)

drug_train <- drug_train_tsv %>%
  filter(!is.na(rating), !is.na(urlDrugName), !is.na(condition)) %>%
  mutate(
    sentiment = factor(ifelse(rating >= 7, 1, 0), levels = c(0, 1)))

# Select top 30 drugs and conditions
top_drugs <- drug_train %>%
  count(urlDrugName, sort = TRUE) %>%
  slice_head(n = 30) %>%
  pull(urlDrugName)

top_conditions <- drug_train %>%
  count(condition, sort = TRUE) %>%
  slice_head(n = 30) %>%
  pull(condition)

# Recode less common ones as "Other"
drug_model <- drug_train %>%
  mutate(
    drug_group = ifelse(urlDrugName %in% top_drugs, urlDrugName, "Other"),
    condition_group = ifelse(condition %in% top_conditions, condition, "Other")
  ) %>%
  select(sentiment, drug_group, condition_group)
```

```r
# One-hot encode
encoded_data <- model.matrix(~ drug_group + condition_group - 1, data = drug_model)

# Combine features and labels
encoded_df <- as.data.frame(encoded_data)
encoded_df$sentiment <- drug_model$sentiment

# Train/test split
set.seed(42)
split_ind <- createDataPartition(encoded_df$sentiment, p = 0.8, list = FALSE)
train_set <- encoded_df[split_ind, ]
test_set <- encoded_df[-split_ind, ]

train_x <- train_set %>% select(-sentiment)
test_x <- test_set %>% select(-sentiment)
train_y <- train_set$sentiment
test_y <- test_set$sentiment

# Run KNN
knn_pred <- knn(train = train_x, test = test_x, cl = train_y, k = 5)
confusionMatrix(knn_pred, test_y)
library(ggplot2)
library(factoextra)

drug_kmeans <- drug_train_tsv %>%
  filter(!is.na(rating), !is.na(urlDrugName), !is.na(condition)) %>%
  mutate(
    top_drugs = ifelse(urlDrugName %in% top_drugs, urlDrugName, "Other"),
    top_conditions = ifelse(condition %in% top_conditions, condition, "Other")
  ) %>%
  select(rating, top_drugs, top_conditions)

train_labels <- as.factor(unlist(train_set$sentiment))
test_labels <- as.factor(unlist(test_set$sentiment))

# Encodes categorial variables
kmeans_encoded <- model.matrix(~ top_drugs + top_conditions - 1, data = drug_kmeans)
kmeans_input <- cbind(scale(drug_kmeans$rating), kmeans_encoded)

# K-means clustering (k = 5 clusters)
set.seed(123)
kmeans_result <- kmeans(kmeans_input, centers = 5, nstart = 25)
drug_kmeans$cluster <- as.factor(kmeans_result$cluster)

# Cluster Plot
fviz_cluster(list(data = kmeans_input,
                  cluster = kmeans_result$cluster),
             geom = "point",
             ellipse.type = "norm",
             ggtheme = theme_minimal()) +
  labs(title = "K-means Clustering: Discovery of Patient Review Groups")
```