Sentiment analysis on the US drug reviews

Team 2:

Faiza Ayoun, Syama Ravi Teja Jerrypothula, Heer Bhavesh Parekh, Vamshi krushna Lakavath

Introduction

- There are more than 14000 drugs in the world.
- Choosing one drug with most curing capacity and least side effects is very difficult.
- For better treatment, there is a huge need to analyze the effectiveness of these drugs.
- We are analyzing the drug reviews and ratings of users in the US.
- Sentiment analysis on reviews of drugs by using different ML models

Objectives and Motivation

- Analysis of reviews and ratings of drugs
- Creating the sentiment score using the NLTK and TextBlob for each drug review.
- Sentiment classification of the drug reviews using the developed models.
- Providing the best drug for a medical condition.

Existing Solution

- Existing system provides sentiment analysis on product reviews and news.
- Very few researches related to health care.

Current Solution

- Deep analysis on drugs dataset.
- Provides Sentiment analysis along with the sentiment score.
- Comparative analysis of different ML models.
- Useful as a drug recommender system.

Literature Survey

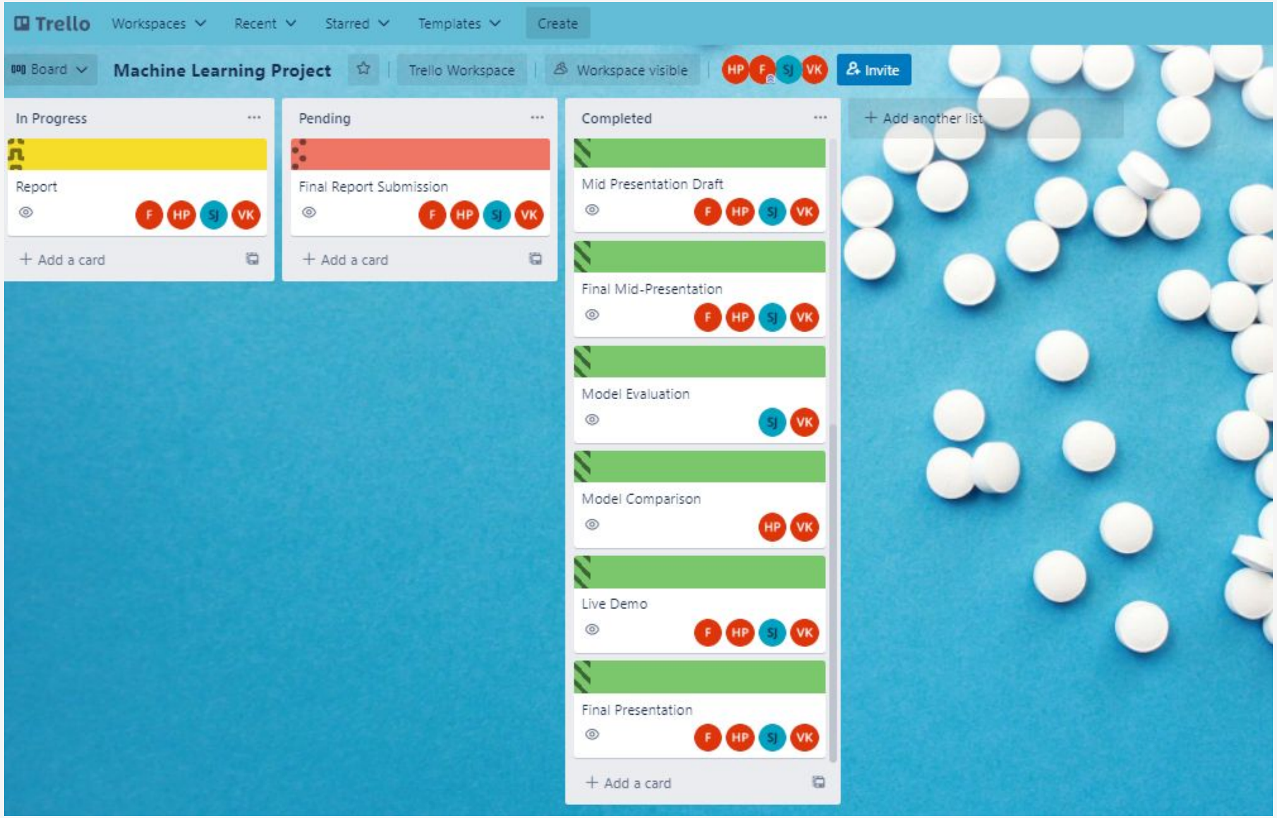
Author	Models Used	Accuracy	Description
Ting-Chou ling, Yufei Mao, Yen-Yi Wu	Linear Regression Ridge Regression Random Forest, Gradient Boosting	R squares best scores: 57.52, 45.82, 78.86, 57.50, respectively	Authors used many techniques to prepare data for sentiment analysis, then used and compared different models.
Yue Han, Meiling Liu1, and Weipeng Jing	PM-DBiGRU	78.26	Proposed a aspect level sentiment analysis dataset SentiDrugs, Built several models and compared with their PM-DBiGRU model.
Felix Gräßer, Surya Kallumad,Hagen Malberg, Sebastian Zaunseder	Linear Regression	92.24	The researchers have used Aspect-Based Sentiment Analysis of Drug Reviews by Applying Cross-Domain and Cross-Data Learning approaches

Project Requirements (Data, Tools, and Services)

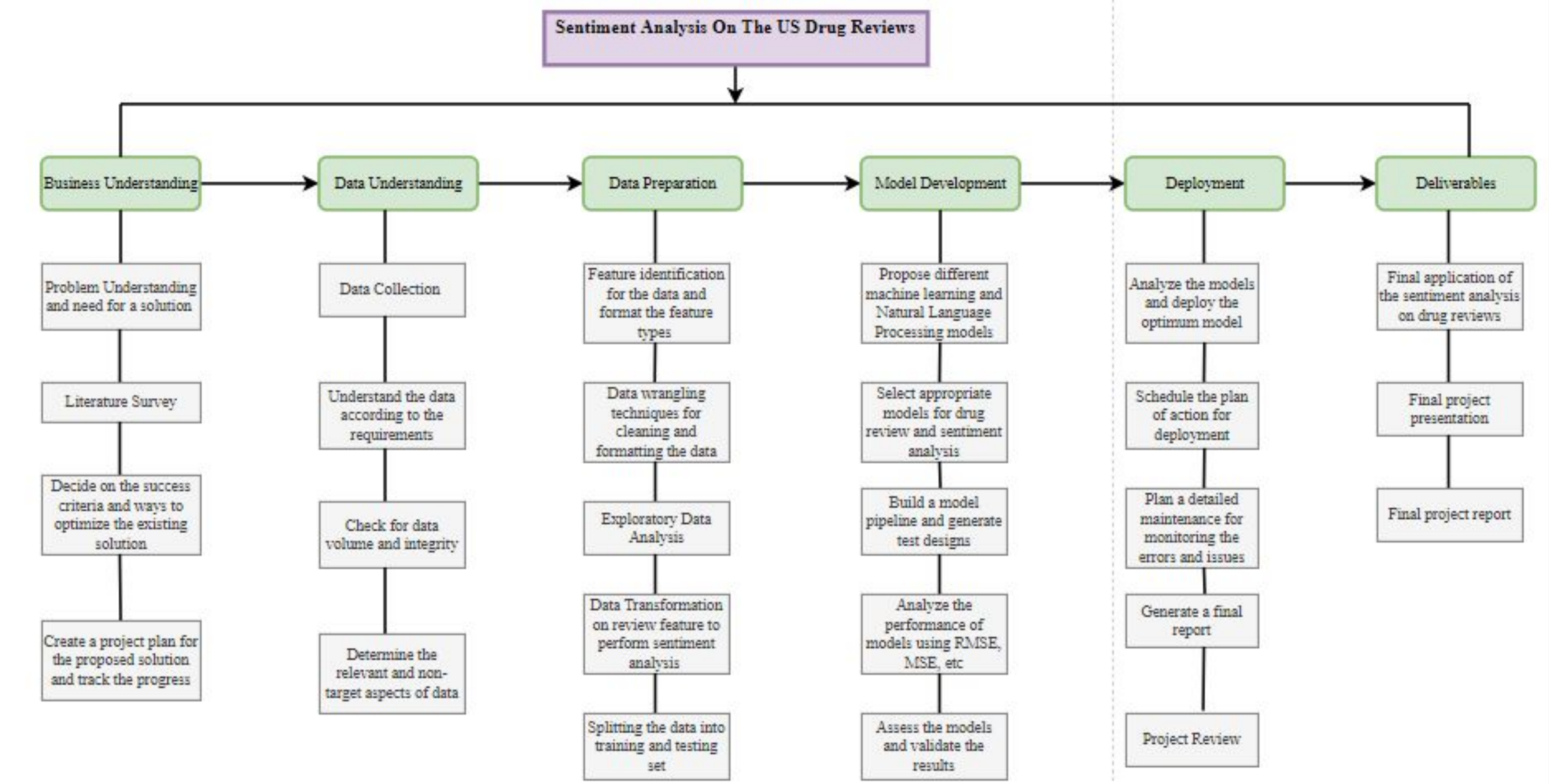
- Drug review dataset
- Computing Machine
- Data storage platform (GCP)
- Data Cleaning : Tableau Prep builder
- Collaboration & Communication : Trello, Zoom, and WhatsApp
- Environment : Google Colab Notebooks
- Language: Python 3

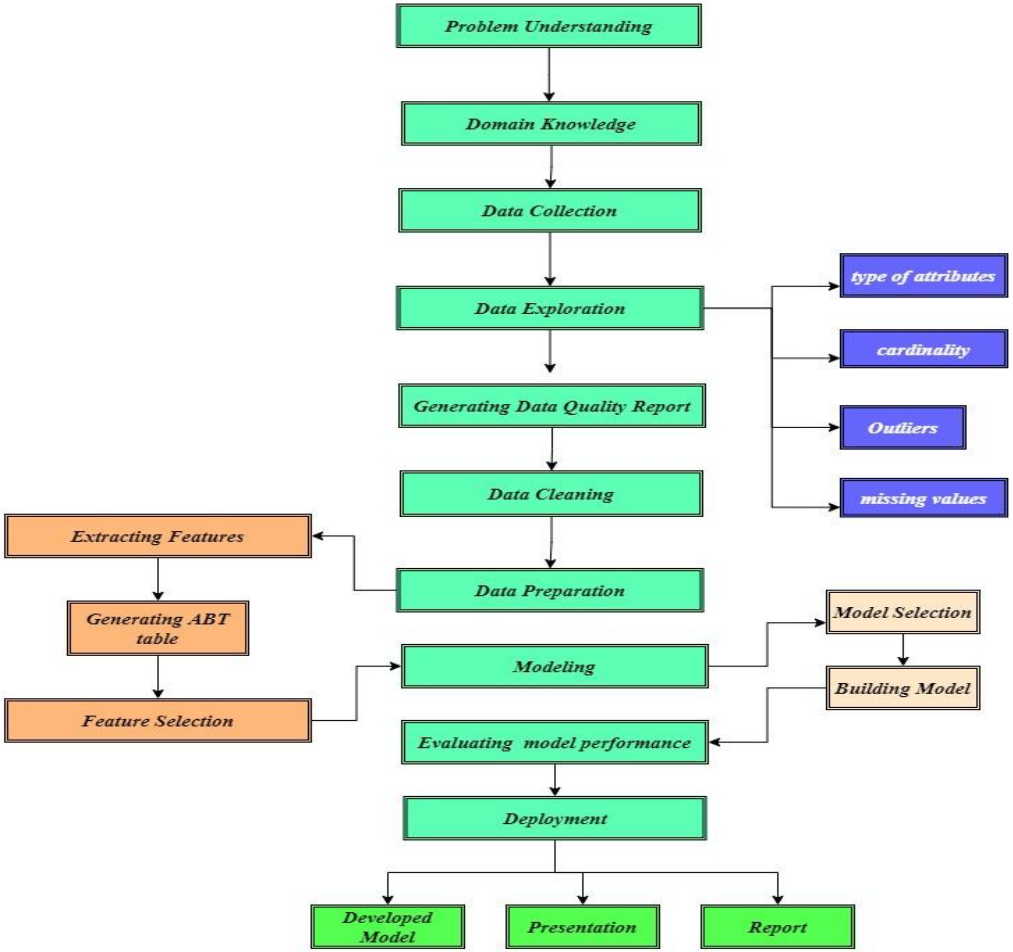
Collaboration-Project Management

Trello



Work Breakdown Structure-CRISP-DM





Flow of the Project

Data Collection

- kaggle.com
- UCI Machine Learning Repository
- Drugs.com
- The US customers

Sample of Dataset

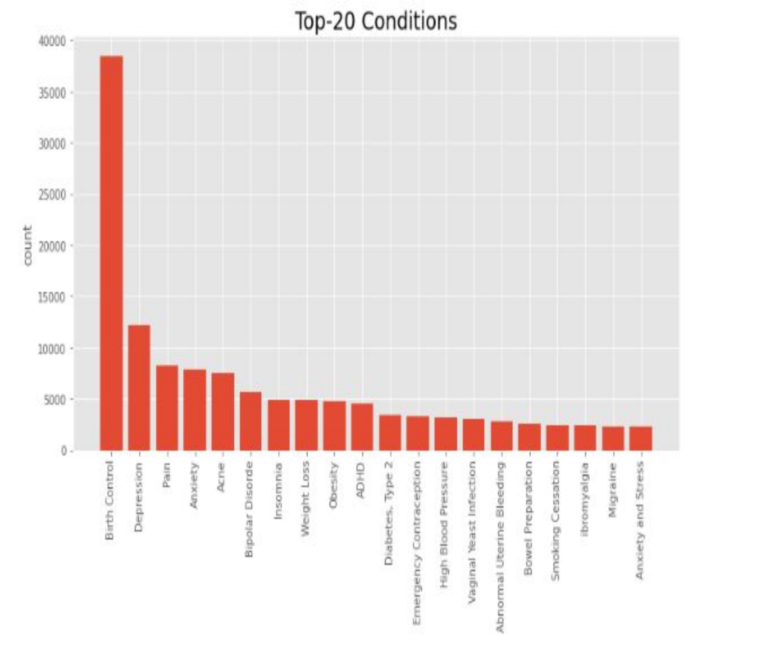
uniqueID	drugName	condition	review	rating	date	usefulCount
163740	Mirtazapine	Depression	"I've tried a few antidepressants over the	10	2/28/2012	22
206473	Mesalamine	Crohn's Disease, Maintenance	"My son has Crohn's disease and has done	8	5/17/2009	17
159672	Bactrim	Urinary Tract Infection	"Quick reduction of symptoms"	9	9/29/2017	3
39293	Contrave	Weight Loss	"Contrave combines drugs that were used for al	9	3/5/2017	35
97768	Cyclafem 1 / 35	Birth Control	"I have been on this birth control for one cycle.	9	10/22/2015	4
208087	Zyclara	Keratosis	"4 days in on first 2 weeks. Using on arms and fa	4	7/3/2014	13
215892	Copper	Birth Control	"I've had the copper coil for about 3 mont	6	6/6/2016	1

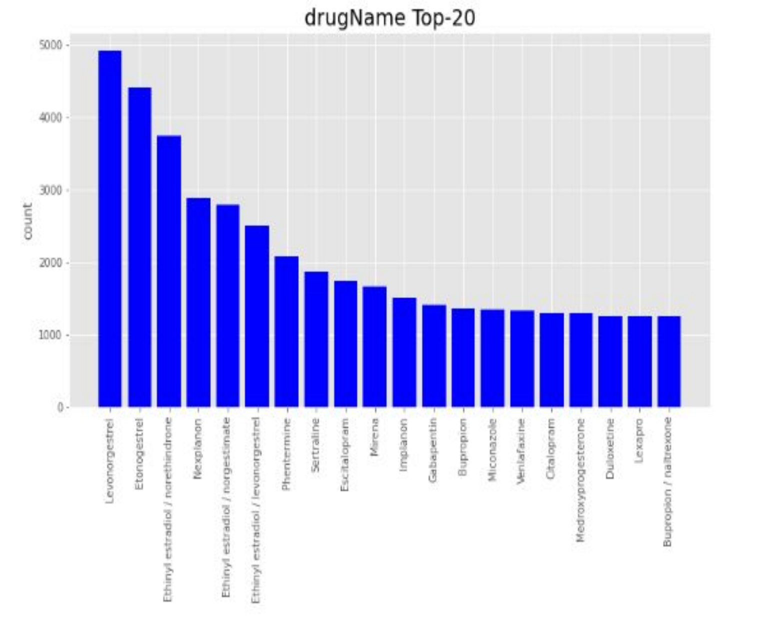
https://archive.ics.uci.edu/ml/datasets/Drug+Review+Dataset+%28Drugs.com%29

Data Description

- **UniqueID** : Unique ID for each record. There are a total of **215,063** unique IDs in the dataset.
- **DrugName**: The name of the specific dru. There are total **3671** unique drugs in the dataset.
- **Condition**: The medical condition of the user under which the drug has been consumed. There are a total of **917** unique conditions.
- **Review**: Review written by the users of the drugs used for a specified condition. There are a total of **128478** distinct reviews.
- **Rating**: The rating given by the customer from 1 to 10 to a drug for a condition.
- **Date:** Date on which the review has been entered. Date range is 02/24/2008-12/11/2017.
- **UsefulCount**: It is the number of people who found the rating and review of a customer relevant and helpful. The range of useful count is 0-1291.

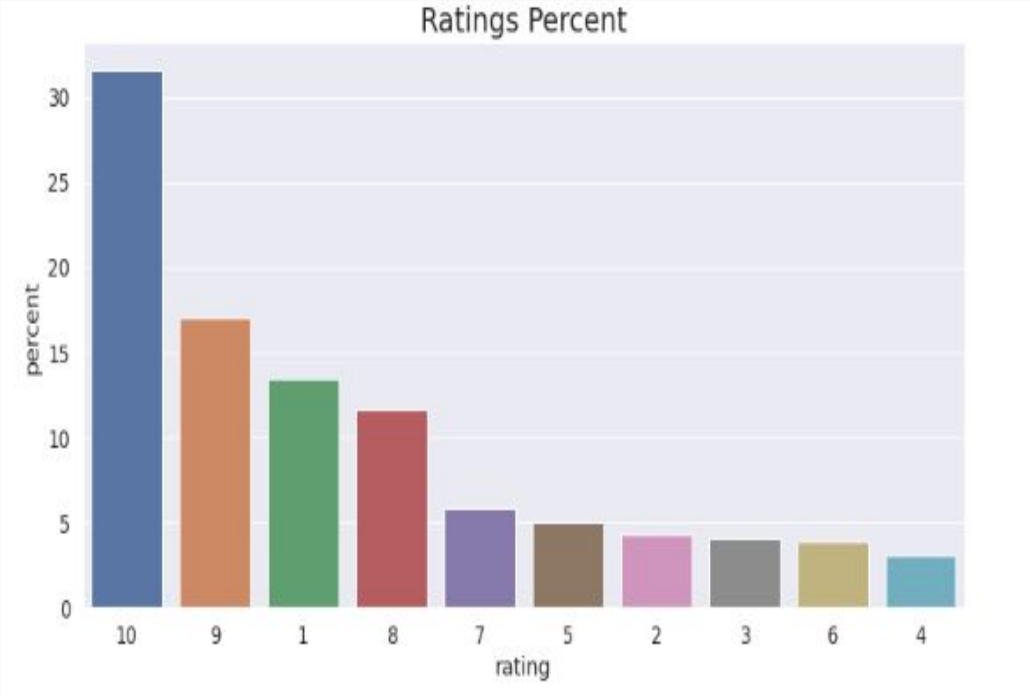
Dataset Insights



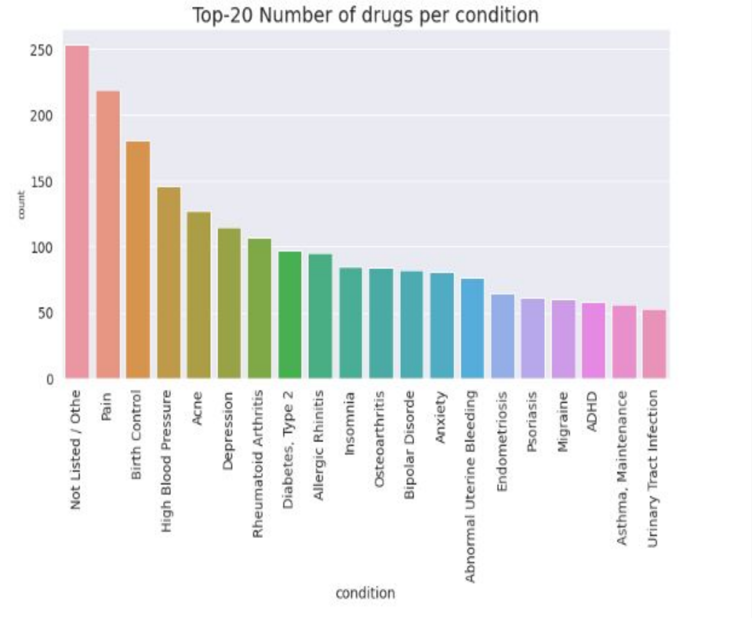


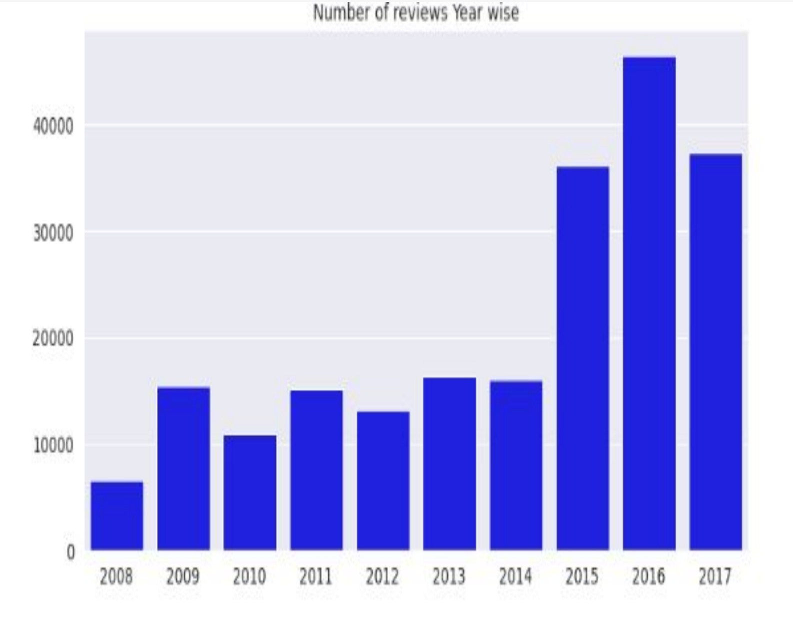
Dataset Insights (cont..)

	rating	counts	percent
0	10	68005	31.620967
1	9	36708	17.068487
2	1	28918	13.446292
3	8	25046	11.645890
4	7	12547	5.834104
5	5	10723	4.985981
6	2	9265	4.308040
7	3	8718	4.053696
8	6	8462	3.934661
9	4	6671	3.101882



Dataset Insights (cont..)





Exploratory Data Analysis

Data Quality Report

Continuous Features

	rating	usefulCount
count	215063.000000	215063.000000
mean	6.990008	28.001004
std	3.275554	36.346069
min	1.000000	0.000000
25%	5.000000	6.000000
50%	8.000000	16.000000
75%	10.000000	36.000000
max	10.000000	1291.000000

Values Threshold

	rating	usefulCount
Q1	5.0	6.0
Q3	10.0	36.0
IQR	5.0	30.0
min	1	0
max	10	1291
lower treshold	-5.0	-54.0
upper treshold	20.0	96.0

Categorical Features

	drugName	condition	review	date
count	215063	215063	215063	215063
nulls	0	1194	0	0
%miss	0.0	0.005552	0.0	0.0
cardinality	3671	917	128478	3579
mode	Levonorgestrel	Birth Control	"Good"	1-Mar-16
mode freq	4930	38436	39	185
mode%	0.022924	0.17872	0.000181	0.00086
2nd mode	Etonogestrel	Depression	"Good."	31-Mar-16
2nd mode freq	4421	12164	26	183
2nd mode%	0.020557	0.05656	0.000121	0.000851

Exploratory Data Analysis

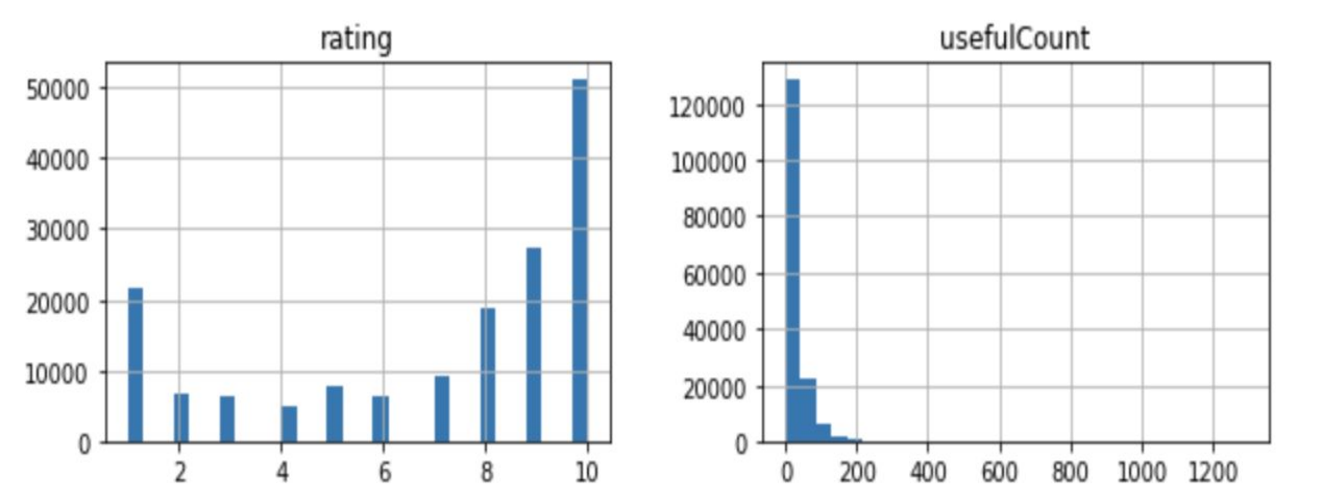
Data Types

```
In [111]: data.info() # data.dtypes

<class 'pandas.core.frame.DataFrame'>
Int64Index: 215063 entries, 206461 to 113712
Data columns (total 6 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   drugName     215063 non-null  object
 1   condition    213869 non-null  object
 2   review       215063 non-null  object
 3   rating       215063 non-null  int64
 4   date         215063 non-null  object
 5   usefulCount  215063 non-null  int64
dtypes: int64(2), object(4)
memory usage: 11.5+ MB
```

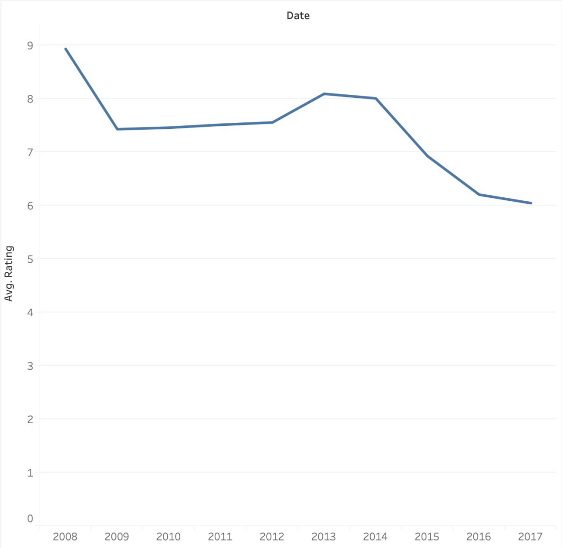
EDA- Data Distributions

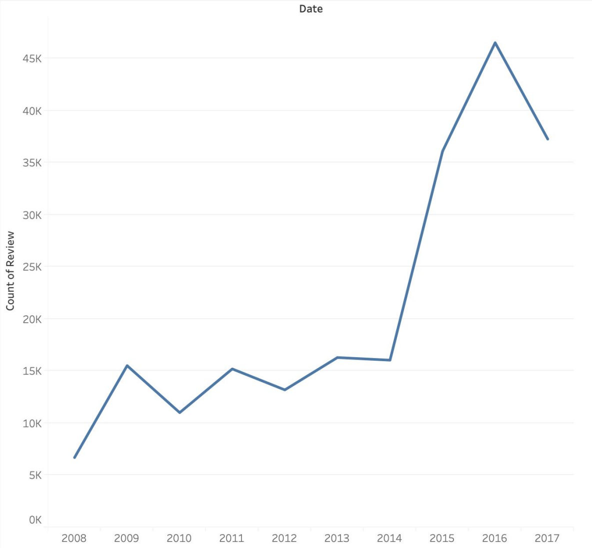
- Bimodal Distribution of “rating” feature confirming that we have two groups of ratings (positive and negative) as shown in the bar plot.
- Exponential distribution of “usefulCout” with most values between 0 and 100 (about 95%)

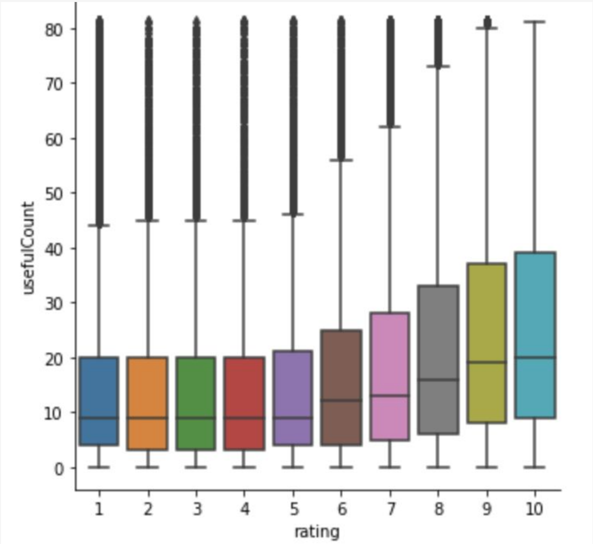


rating	
1	28770
2	9203
3	8663
4	6623
5	10652
6	8403
7	12471
8	24912
9	36500
10	67695

EDA-Relations among Attributes







EDA Summary

- 1194 **missing values** for condition
- 7580 rows has usefulCount feature higher than 2 IQR threshold from the third quartile which pull the distribution to the left. These are **outliers**. (95% usefulCount more than "100", 5% values from 100 to 1291)
- Rating feature should be a categorical feature with max Rating = 10 and min Rating =1. (This is **irregular cardinality**)
- More positive reviews than negative, There are more ratings of 10,9,and 8 than rating from 7 to 1. Mean is 7 which explains it !
- Many HTML code left from scraping.
- 1171 rows with meaningless review "N users found this comment helpful." With N a random number.

Data Cleaning : Data Quality plan

Implementation of manual data Cleaning using Python

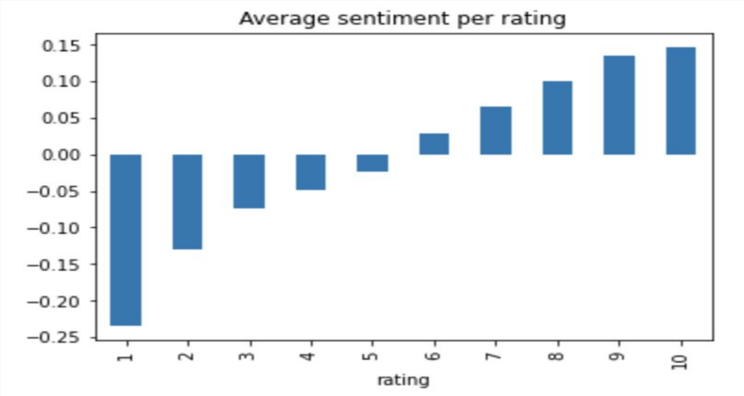
Feature	Data Quality Issue	Potential Handling Strategies
Condition	1171 rows with non-sense review “Users found this comment helpful”	Replace with null values to be treated with other null values
Condition	1194 null values + 1171 from above.	Replace with the mode of conditions corresponding to the same drug name
Review	HTML code left from scraping.	Replace code with adequate characters
usefulCount	Outliers	Drop rows with "usefulCount">200

Data Preparation

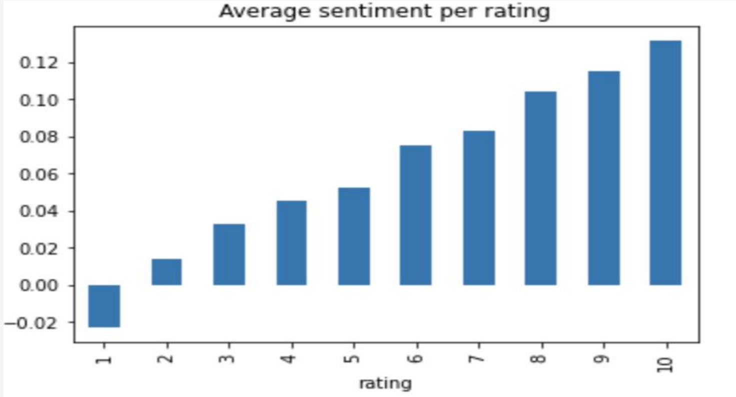
Natural Language Processing before the sentiment analysis :

- Tokenization: Splitting sentences into split words.
- Remove stop-words and punctuations
- Stemming: Reduce a word to its word stem or root
- Encoding (TF-IDF)
- Change Rating to positive (1), Negative (-1), and neutral (0)
- Split the data using train_test_split to 80% data for training 20% for Testing and Evaluating

NLTK



TextBlob



TF-IDF

- Stands for Text Frequency - inverse Document Frequency
- Two Algorithms Multiplied together

TF(t) = Nbre of times term t appears in a doc / Nbre of all terms in the doc

IDF(t) = log e (Number of Documents/ Number of Documents with term t in it)

TF-IDF = TF(t) * IDF(t)

Main Steps :

- Create a term Frequency matrix where rows are "reviews" and columns are different terms (3000 in our Case).

  Each value TF in the document using Formula
- Calculate IDF for each term using Formula
- Multiply TF matrix with IDF respectively
- Feed the Final matrix to machine learning Models

Modeling

- Naive Bayes
- Logistic Regression
- K Nearest Neighbor
- Random Forest
- Ensemble Techniques
    - Random Forest
    - Extreme Gradient Boosting (XGBOOST)
    - Max Voting

Naive Bayes

- Probabilistic Algorithm works on Bayes Theorem

Naive Bayes

```
MNB_model = MultinomialNB()
result = MNB_model.fit(xtrain, ytrain)
y_pred_MultinomialNB = MNB_model.predict(xtest)
model_metrics(ytest, y_pred_MultinomialNB)
```

```
      Mean Squared Error: 0.7902371460123159
 Root Mean Squared Error: 0.8889528367761227
   Accuracy of the model: 0.7116439253560933
```

Accuracy: 70%

Precision: 70%

Recall: 71%

F1-Score: 63%

Logistic Regression

Basic regression Technique (Supervised)

Logistic Regression

```
logreg = LogisticRegression()
logreg.fit(xtrain, ytrain)
y_pred_LR = logreg.predict(xtest)
model_metrics(ytest, y_pred_LR)
```

```
     Mean Squared Error: 0.5413180601568496
Root Mean Squared Error: 0.7357432025896329
  Accuracy of the model: 0.769479851011660 7
```

Accuracy: 77%

Precision: 73%

Recall: 77%

F1-Score: 64%

K- Nearest Neighbour

- Clustering Algorithm works on calculating distance between data points.

KNN

```
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(xtrain, ytrain)
y_pred_knn = knn.predict(xtest)
model_metrics(ytest, y_pred_knn)
```

```
     Mean Squared Error: 0.38430007299679936
Root Mean Squared Error: 0.6199194084691972
  Accuracy of the model: 0.863720590712561
```

Accuracy: 86%

Precision: 88%

Recall: 86%

F1-Score: 85%

Decision Tree

Entropy

Information Gain (IG)

Decision Tree

```
decision_tree = DecisionTreeClassifier()
decision_tree.fit(xtrain,ytrain)
y_pred_dt=decision_tree.predict(xtest)
model_metrics(ytest, y_pred_dt)
```

```
     Mean Squared Error: 0.38214760327175396
Root Mean Squared Error: 0.6181808823247076
  Accuracy of the model: 0.8463885301439347
```

Accuracy: 85%

Precision: 85%

Recall: 85%

F1-Score: 85%

Random Forest

- Ensemble learning method, Create multiple DTs as a forest

Random Forest

```
#kfold = model_selection.KFold(n_splits=10, random_state=seed)
random_forest = RandomForestClassifier(n_estimators=100, max_features=100)
random_forest.fit(xtrain, ytrain)
y_pred_rf = random_forest.predict(xtest)
model_metrics(ytest, y_pred_rf)
#results = model_selection.cross_val_score(model, X, Y, cv=kfold)
#print(results.mean())
```

```
      Mean Squared Error: 0.27881033934153143
 Root Mean Squared Error: 0.5280249419691568
   Accuracy of the model: 0.8936305613266701
```

Accuracy: 89%

Precision: 90%

Recall: 89%

F1-Score: 89%

XGBoost

Gradient Boosting Technique, highly flexible

XGBoost

```
xgb_model = XGBClassifier(random_state = 0 )
xgb_model.fit(xtrain, ytrain)
y_pred_xgb = xgb_model.predict(xtest)
model_metrics(ytest, y_pred_xgb)
```

```
      Mean Squared Error: 0.572032867276845
 Root Mean Squared Error: 0.7563285445339512
   Accuracy of the model: 0.769592153779924
```

Accuracy: 77%

Precision: 75%

Recall: 77%

F1-Score: 73%

Voting Classifier

- Ensemble learning method, Create multiple DTs as a forest

```
In [100]: Max_Vot_pred = np.array([])
          for i in range(0,len(xtest)):
              Max_Vot_pred = np.append(Max_Vot_pred, stats.mode([y_pred_xgb[i], y_pred_dt[i], y_pred_rf[i]]).mode)

          model_metrics(ytest, Max_Vot_pred)

                Mean Squared Error: 0.261927489845958O4
           Root Mean Squared Error: 0.511788520627376
             Accuracy of the model: 0.8978231980084975
```

```
In [160]: Max_Vot_pred = np.array([])
          for i in range(0,len(xtest)):
              Max_Vot_pred = np.append(Max_Vot_pred, stats.mode([y_pred_LR[i], y_pred_dt[i], y_pred_knn[i]]).mode)

          model_metrics(ytest, Max_Vot_pred)

                Mean Squared Error: 0.27188500196529847
           Root Mean Squared Error: 0.5214259314277517
             Accuracy of the model: 0.8946038519849514
```

Accuracy: 89%

Precision: 90%

Recall: 89%

F1-Score: 89%

Model Result Table

Model	Accuracy	Precision	Recall	F1-Score
Naive Bayes	70%	70%	71%	63%
Logistic Regression	77%	73%	77%	64%
KNN	86%	88%	86%	85%
Decision Tree	85%	85%	85%	85%
Random Forest	**89%**	**90%**	**89%**	**89%**
XGBoost	77%	75%	77%	73%
Voting Classifier	**89%**	**90%**	**89%**	**89%**

Conclusion

- Reviews are greatly useful to analyze the drug effectiveness.
- Each medicine and conditions are different with its corresponding side effects
- As per the evaluation matrix, Random Forest is the best model with 89% accuracy.

Future Scope

- Try and increase the accuracy of models above 90%
- Use Deep Learning models for classification
- Create an end to end pipeline for drug classification and review in real time
- Create an interface to help the doctors in selecting the best drugs based on the reviews and ratings from the patients.

Code

References

- Y. Han, M. Liu and W. Jing, "Aspect-Level Drug Reviews Sentiment Analysis Based on Double BiGRU and Knowledge Transfer," in IEEE Access, vol. 8, pp. 21314-21325, 2020, doi: 10.1109/ACCESS.2020.2969473.
- Saha, J., Chowdhury, C., & Biswas, S. (2020). Review of machine learning and deep learning based recommender systems for health informatics. In Deep Learning Techniques for Biomedical and Health Informatics (pp. 101-126). Springer, Cham.
- Tran, T. N. T., Felfernig, A., Trattner, C., & Holzinger, A. (2021). Recommender systems in the healthcare domain: state-of-the-art and research issues. Journal of Intelligent Information Systems, 57(1), 171-201.
- *Statistics*. Statistics | DrugBank Online. (n.d.). Retrieved May 10, 2022, from https://go.drugbank.com/stats


thank you
danke
謝謝
ngiyabonga
شكراً جزيلاً
рахмат
teşekkür ederim
Баярлалаа
спасибо
faafetai lava
mersi
barka
welalin
tack
spas
kia ora
dank je
misaotra
vinaka
спасибі
blagodaram
matondo
paldies
grazzi
mahalo
tapadh leat
gracias
хвала
asante
manana
obrigada
tenki
kiitos
dankie
nandri
nanni
dhanyavad
maururu
köszönöm
hvala
grácje
bayarlalaa
bedankt
enkosi
dziękuję
akun dankon
aciú
djiere dieuf
tau
mochchakkeram
murakoze
дякую
mamnun
chnorakaloutioun
gratias ago
gràcies
sulpáy
go raibh maith agat
chokrane
taiku
sobodi
dėkuji
sagolun
sukriya
kop khun krap
arigatô
takk
dakujem
trugarez
obrigado
mèsi
didi madloba
ありがとう
tanemirt
rahmet
grazie
najis tuke
kam sah hamnida
rahmat
terima kasih
diolch
dhanyavadagalu
shukriya
merce
мерси
তোমাকে ধন্যবাদ
감사합니다
xièxie
ευχαριστώ
merci