Rating prediction of Drug Reviews with exclusion of text

Nguyen Hoang

1 Dataset

1.1 Description:

The dataset I used is the Drug Review Dataset (Drugs.com) from UCI Machine Learning Repository for retrieval of user reviews and ratings on drug experience. According to (1), the dataset provides user reviews on specific drugs along with related condition and a 10-star based rating system that reflects the overall customer satisfaction, which was obtained by mining the most widely visited pharmaceutical information website Drugs.com. The original dataset has 215063 reviews and then further split into training and testing sets using stratified random sampling with proportion of 75% and 25% respectively.

However, since the original dataset is too large and there are multiple drugs with different conditions that has too few interactions such that I decided to filter it out so that there must be more than 50 reviews for a drug and more than 50 reviews for a condition to appear in the dataset. After filtering, the size of training set drops to 132607 and the test size drops to 44178, which is still relatively large (Table 1). On the other hand, the number of unique drugs drops significantly from 3436 to 564 and conditions from 884 to 206. This makes sure that we are more focused on the most popular drugs and conditions. Even though that the original literature focus on the sentiment analysis of the patient's drug experience through review, my task is more focused on predicting the ratings based on a variety of other features. Therefore, I create a feature using the length of the review and remove the review texts from the features.

1.2 Descriptive statistics:

Since we use stratified random sampling to split the dataset, the distribution of the testing set attributes also resembles the distribution of the training set attributes. Table 2 shows the summary statistics of the numerical attributes, which consist of the number of people found the review useful and the length of the review text I just added. As we can see that both features have the distributions to be heavily skewed right. Using 1(a), 1(b) and 1(c), we can see that the dataset covers from year 2008 to year 2017 with more emphasis on the three most recent years. Moreover, the reviews span uniformly over all the days and months (day 31 is nearly half since only half of the months has day 31). Figure 1(d) shows that the distribution of ratings is heavily leaned towards either rating 1 or 10, kind of similar to the bimodal distribution.

Figure 2 suggests that the top 15 conditions with most counts are mostly those that don't require a drug prescription to buy, which is why there are more user experiences since it's easy for people to get their hand on. Besides, the conditions shown are easier to detect as the symptoms can express clearly without going through health inspection.

2 Predictive Task

Before going into clearly defining what are the predictive tasks for this dataset, I will be using random forest and XGBoost models in order to find the most important features. I will pre-process the dataset using standard scaler for numerical attributes, one hot encoder for categorical attributes and oridinal labeler for the days in a month. From 3(a) and 3(b), reviewLen, usefulCount and day are the three most important features. However, based on (3), the impurity-based feature importance of random forests suffers from numerical features even though they are not predictive of the targets. The limitations of this approach are that it is biased towards high cardinality features and it might not generalize well on the testing set. Therefore, I will based my feature selection on XGBoost feature importance as the algorithm is well-regarded among data science competitions. With 3(c) and 3(d), in general we can see that the feature usefulCount is still relatively important. Furthermore, some conditions and drug names coupled with three most recent years stand out on the top importances. Hence, instead of doing sentiment analysis of the reviews like (1), I will divide my tasks in 2 categories: regression and classification problems.

- First, predicting ratings based on drug names and conditions. This will be considered as a regression problem which try to predict the ratings between 1 and 10. I believe that the prediction models from (4), especially KNN, SVD and Baseline, will help.

- Second, labeling ratings using a combination of drug names, conditions, usefulCounts, time and reviewLen. This will be regarded as a classification problem where we try to predict what will the label falls into between 1 and 10. I will be using KNN and XGBoost algorithms from (2) for this predictive task.

- Last but not least, I believe that a 10-star based rating system can be too much since it is hard to solve a 10-class classification problem using subjective data such as user reviews as the same drug can have different reactions to different people's physical states. Because it is too complicated so I decide to reduce it down to a 3-class predictive task such that every ratings between 1 and 3 will be considered as negative (rated -1), between 4 and 7 will be neutral (rated 0) and between 8 and 10 will be positive (rated 1). For this task, I will only be using drug names, conditions and usefulCounts based on the previous feature importance. The same algorithms and metrics will be used like the previous classification task.

3 Model

3.1 Learning algorithms:

For general setup, we will use the previously processed data to fit in a 4-fold cross-validation GridSearchCV in order to find the best set of hyperparameters to optimize the corresponding score metrics. In the case of XGBoost classifier, it is very time consuming to even run through a set of hyperparameter so as to reduce the training time, I will be using RandomizedSearchCV specifically for XGBoost.

1. Regression problem: For this task, Surprise (4) will be the main module that I use to run: KNNBaseline from memory-based method, BaselineOnly with both ALS and SGD from baseline estimate method and SVD from latent-factor model. Below are the hyperparameter I chose for each algorithm for tuning:
   - KNNBaseline: The number of nearest neighbors k will vary between 40, 50, 200, 500, 1000 and 1500. I will only use KNN with Pearson similarity coupled with SGD method where the learning rate is between 0.001 and 0.01, number of epochs is between 10 and 50, regularization parameter is between 0.01 and 0.1.
   - SVD: I train the model using the number of factors between 5 and 20 where the number of epochs is between 10 and 30. The learning rate for all parameters is between 0.001 and 0.01 while the regularization term for all parameters is between 0.005 and 0.08.
   - BaselineOnly: For both ALS and SGD methods, the number of epochs is between 10 and 30. However, ALS will use seperate regularization term for drugs between 5 and 20 while the regularization term for conditions is between 10 and 30. As for SGD, the only regularization term is between 0.005 and 0.1 while the learning rate is between 0.001 and 0.01.

2. Classification problem (10-class and 3-class):
   - KNN: Both problems used Euclidean distance with both uniform weighted and distance weighted. However, for 10-class, I used 10 values of K between 5 and 140 equally spaced while for the 3-class, I used 7 values of K between 5 and 29 equally spaced.
   - XGBoost: This algorithm has a lot of hyperparameter to tune. Even so, I will only choose a few to optimize like minimum sum of instance weight needed in a child varies between 1 and 5, learning rate between 0.1 and 0.3, maximum depth of a tree between 3 and 7, number of estimators between 100 and 1000. Lastly, the subsample ratio of the training instances is between 0.6, 0.8 and 1.

3.2 Metrics

1. Regression: the main metric to opt for using GridSearchCV is the RMSE. Furthermore, I will also include the MAE and R2 as reference but RMSE is already predictive.

2. Classification: the main metric that I use for GridSearchCV is the accuracy score. Moreover, for better comparisons, I will also include the classification report of precision, recall and f1-score for each class, their unweighted mean (macro) and weighted mean considering label imbalance.

3.3 Notes:

Even though the regression models I used don't require much complexity as it only need drug names and conditions to predict ratings, 10-star rating system might pose a problem since as we can see from before that most of the reviews are focused on either the lowest or the highest rating with the rest much less than both of them (you can refer to figure 1(d)). Furthermore, as I have said before that the drugs can very much depend on the physical state of each person and its effects won't be the same from people to people so it won't be very accurate to predict the ratings between 2 and 9.

As for classification models, using one hot encode for the categorial features does pump up a lot of the data attributes since we have a lot of different drug names and conditions. High-dimensional data can pose threats to both KNN and tree-based XGBoost algorithms. Since both algorithms have regularization term so I believe there won't be any overfitting problem. Also, the data doesn't have any missing entries but there is a chance for some outliers since the reviews are subjective afterall.

4 Literature

4.1 Origin:

The dataset I am using is originally from (1) where it was used for sentiment analysis of the overall user satisfaction through multiple aspects such as effectiveness and side effects to provide insights. Furthermore, they also consider transfer learning for cross-domain and cross-data sentiment analysis. They use unigram, bigram and trigrams as features to train their logistic regression models. Moreover, there are suggestions that the adoption of deep learning models can improve the achieved results.

4.2 Related works:

- K et al (6) collected review data from two different website *livewell.pk* and *kaymu.pk*. In the paper, it was sentiment analysis using lexicon-based approaches to detect reviews polarity.

- The work of (5) involves training multiple learning models like Random Forest, Naive Bayes and transformer-based neural networks with TF-IDF features as inputs. They use the same dataset as above and expand from just classifying drug reviews to also identify reviews that are inconsistent with the ratings.

- The paper (7) used the same Drugs.com dataset to predicts the class of rating using the textual review with TD-IDF and CV embeddings as inputs. Multiple supervised machine learning algorithms were used such as SVM, Random Forest, Logistic Regression, artificial NN and recurrent NN.

- Gopalakrishnan et al. (8) constructed the drug review dataset from web resource *askapatient.com* and divide it into two based on two different popular drugs cymbalta and depo-provera. The work shows interests in analyzing drug satisfaction from user reviews where the vector space representation for the drug reviews are used as inputs. As for the approaches, SVM is the baseline model while probablistic NN and radial basis function NN are the main methods to be discussed.

4.3 Summary

Most of the papers with the same dataset or similar dataset revolve around sentiment analysis to predict the ratings and their polarity. However, my work doesn't use the review text as it might not be available in real time classification, hence my algorithms focus on trying to find the pattern that there might exist outside of using text. Most, if not all, state-of-the-art methods currently employed to study this type of data are using natural language processing models coupled with deep learning and multiple different word embeddings. Since there have been a lot of good pre-trained language models for this so in the end, all you need is a good clean dataset plus a lot of time for trials and errors to find a suitable algorithm for it. Even though my models don't use the review text as its main attribute, I can learn that the useful count feature plays an important role in predicting the ratings while datetime may just be trivial.

5 Results

As mentioned in the previous sections 2 and 3, I have conducted various tasks with different combination of algorithms and preprocessed features for user rating prediction and polarity:

1. Regression task: Table 4 presents the experimental results from training various models using their optimal set of hyperparameter after GridSearchCV tuning. Even though the KNN model outperforms both the Baseline algorithm and SVD, the result isn't significantly better than the others. This suggests that building recommender systems using only drug names and conditions to deal with explicit rating data doesn't function well as it is too simple. One hypothesis of the limitation is because that given similar drug and similar condition, different people gives different sentiment toward the rating. Furthermore, due to the limit of computing resources, I didn't do a good job in tuning the parameters since the searching space might be too broad. Therefore, none of the features are predictive for these kind of models.

2. Classification task:

- For the 10-class classification task, table 5 shows the testing set performance for both KNN and XGBoost classifiers. We can see that XGBoost has a slightly better performance in accuracy but KNN excels in precision, recall and f1-score for both unweighted and weighted mean. If we look closely at both table 6 and table 7, the recall scores and f1-score for rating 1 and rating 10 are better than KNN while the rest of the ratings with both precision and recall are much worse. This can suggest that XGBoost classifier has better ability to find positive samples of rating 1 and rating 10 but at the cost that the recall scores for ratings 2 to 9 for XGBoost are close 0 since the classifier has mistaken most of them for either rating 1 or rating 10.

- Lastly, for the 3-class classification task, without including the date and the length of the review, the results from table 8 has shown that there is an increase in accuracy of 0.26 for prediction. This might mostly from the effects of changing the classification targets from the system being too wide-ranging to simpler polarity problem. On the other hand, it might also be because of the imbalance in the classes, based on table 9 and table 10, the algorithms are now excelling at predicting positive and negative that most of the misclassifications are those of neutral ratings and that their size isn't much compared to the total.

- In conclusion, if we only consider accuracy as the main metric then XGBoost is the slightly better winner here but if we consider the big picture by including the other metrics then KNN seems to be doing well overall.

3. Improvements:

   - Due to the rise of deep learning models, I believe that better results can be achieved using neural networks approaches, even with the exclusion of review text in the features used for fitting.

   - Since this is a domain-specific problem, incorporating more detailed features about the condition and the drug might help the models to predict better. Since the review text might not be available in real time, we would want to predict the user satisfaction before the user uses the drug, not after consuming it.

List of Tables

Table 1: Description of the dataset

Feature	Description	Type
drugName	The name of the drug to cure the condition	str
condition	The type of condition the user experienced	str
usefulCount	The number of people that found this review useful	int
review	The content of the review	str
date	The date of the review submitted	str
rating	The rating for user satisfaction of the drug for this condition	int $\in [1, 10]$

Dataset	Train size	Test size	No.drugs	No.conditions
Original	161297	53766	3436	884
After filter	132607	44178	564	206

Table 2: Training set numerical atrributes

Statistics	Count of number found useful	Length of review
Mean	28.4528	476.4809
Std	37.2771	239.9271
Min	0	3
25%	6	285
50%	16	481
75%	37	706
Max	1291	10787

Table 3: Training set categorical atrributes

Statistics	Drug	Condition	Month	Year
Unique	564	206	12	10
Top	Levonorgestrel	Birth Control	August	2016
Frequency	3631	27815	11891	29850

Table 4: Testing set performance using Surprise model for regression task

Model	RMSE	MAE	R2
SVD	3.068976	2.547659	0.128873
BaselineALS	3.066863	2.554697	0.130072
BaselineSGD	3.074845	2.559574	0.125538
KNN	3.064171	2.542272	0.131598

Table 5: Testing set performance using models for 10-class classification task

Model	Acc	Precision(m)	Precision(w)	Recall(m)	Recall(w)	F1(m)	F1(w)
KNN	0.35914	0.26	**0.34**	**0.24**	**0.36**	**0.25**	**0.35**
XGBoost	**0.35925**	**0.27**	0.3	0.17	**0.36**	0.15	0.27

Table 6: 10-class KNN Classification report

Model	Precision	Recall	F1-score
Rating 1	0.37	0.41	0.39
Rating 2	0.20	0.16	0.17
Rating 3	0.20	0.15	0.17
Rating 4	0.16	0.12	0.13
Rating 5	0.21	0.16	0.18
Rating 6	0.19	0.13	0.16
Rating 7	0.20	0.15	0.17
Rating 8	0.27	0.24	0.26
Rating 9	0.34	0.33	0.34
Rating 10	**0.48**	**0.58**	**0.52**

Table 7: 10-class XGBoost Classification report

Model	Precision	Recall	F1-score
Rating 1	0.32	0.55	0.41
Rating 2	0.23	0.01	0.03
Rating 3	0.25	0.02	0.04
Rating 4	0.23	0.01	0.03
Rating 5	0.21	0.03	0.05
Rating 6	0.30	0.03	0.05
Rating 7	0.23	0.03	0.05
Rating 8	0.23	0.07	0.11
Rating 9	0.25	0.15	0.19
Rating 10	**0.40**	**0.80**	**0.54**

Table 8: Testing set performance using models for 3-class classification task

Model	Acc	Precision(m)	Precision(w)	Recall(m)	Recall(w)	F1(m)	F1(w)
KNN	0.61791	0.48	**0.56**	**0.44**	**0.62**	**0.43**	**0.57**
XGBoost	**0.6201**	**0.49**	0.55	0.42	**0.62**	0.40	0.55

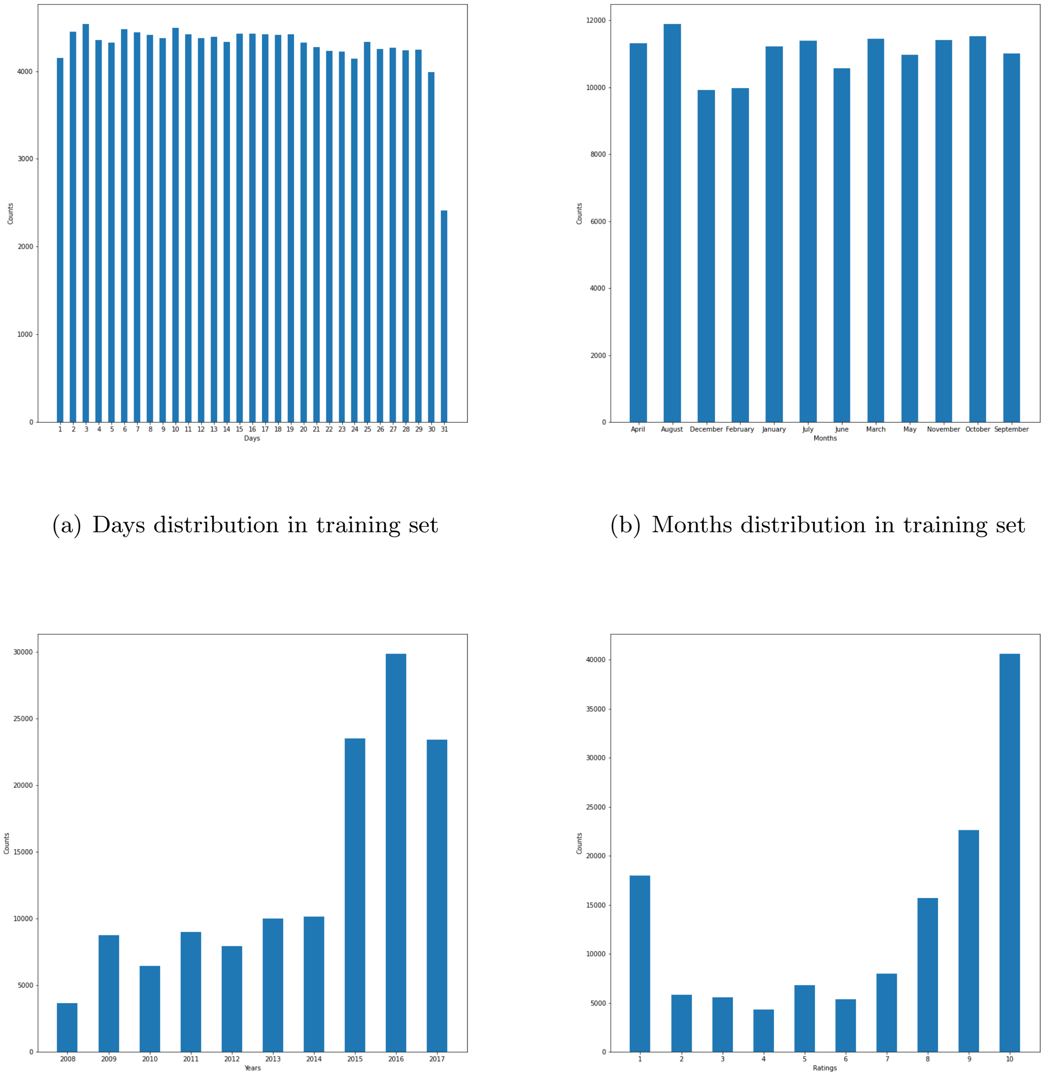
Table 9: 3-class KNN Classification report

Model	Precision	Recall	F1-score
Rating negative	0.45	0.39	0.42
Rating neutral	0.32	0.07	0.11
Rating positive	**0.67**	**0.87**	**0.76**

Table 10: 3-class XGBoost Classification report

Model	Precision	Recall	F1-score
Rating negative	0.49	0.29	0.37
Rating neutral	0.33	0.03	0.06
Rating positive	**0.65**	**0.92**	**0.76**

List of Figures



(a) Days distribution in training set

(b) Months distribution in training set

(c) Years distribution in training set

(d) Ratings distribution in training set

Figure 1: Training set descriptive statistics distribution

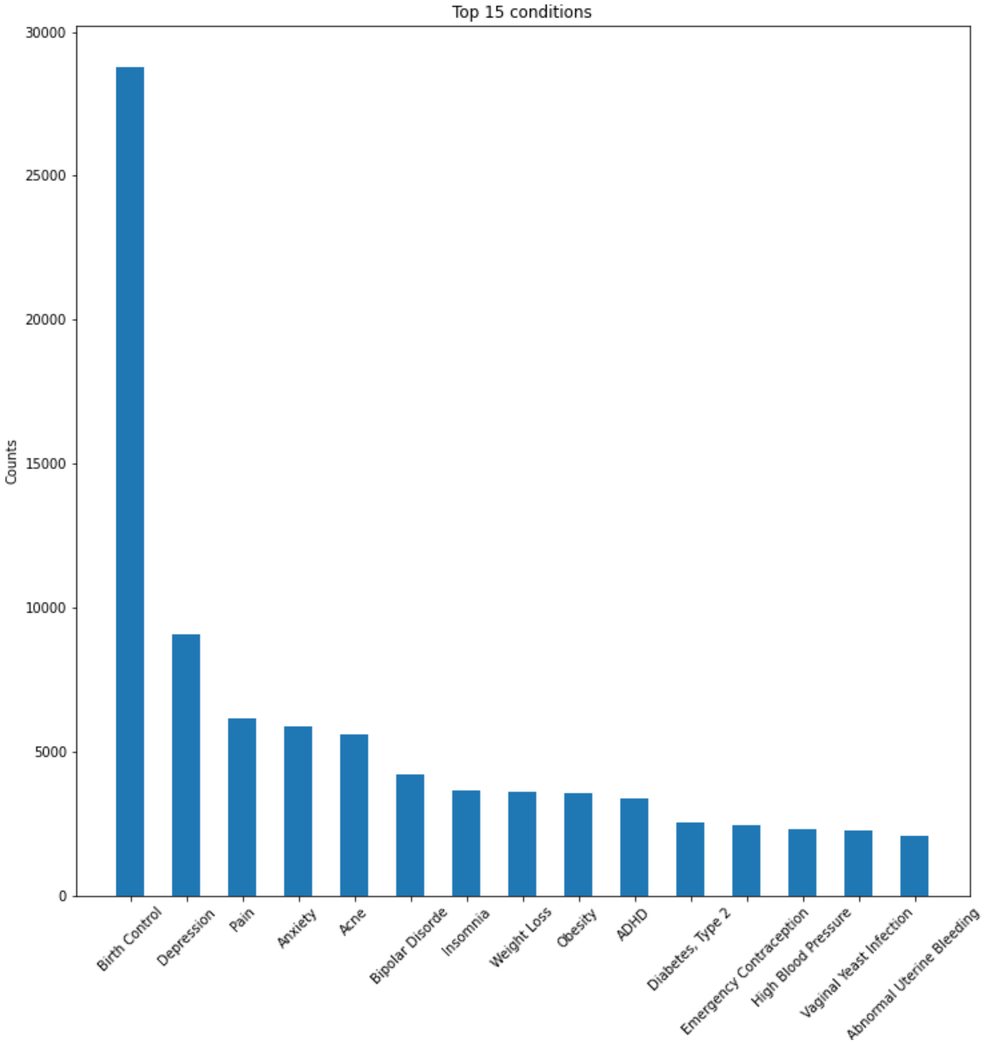
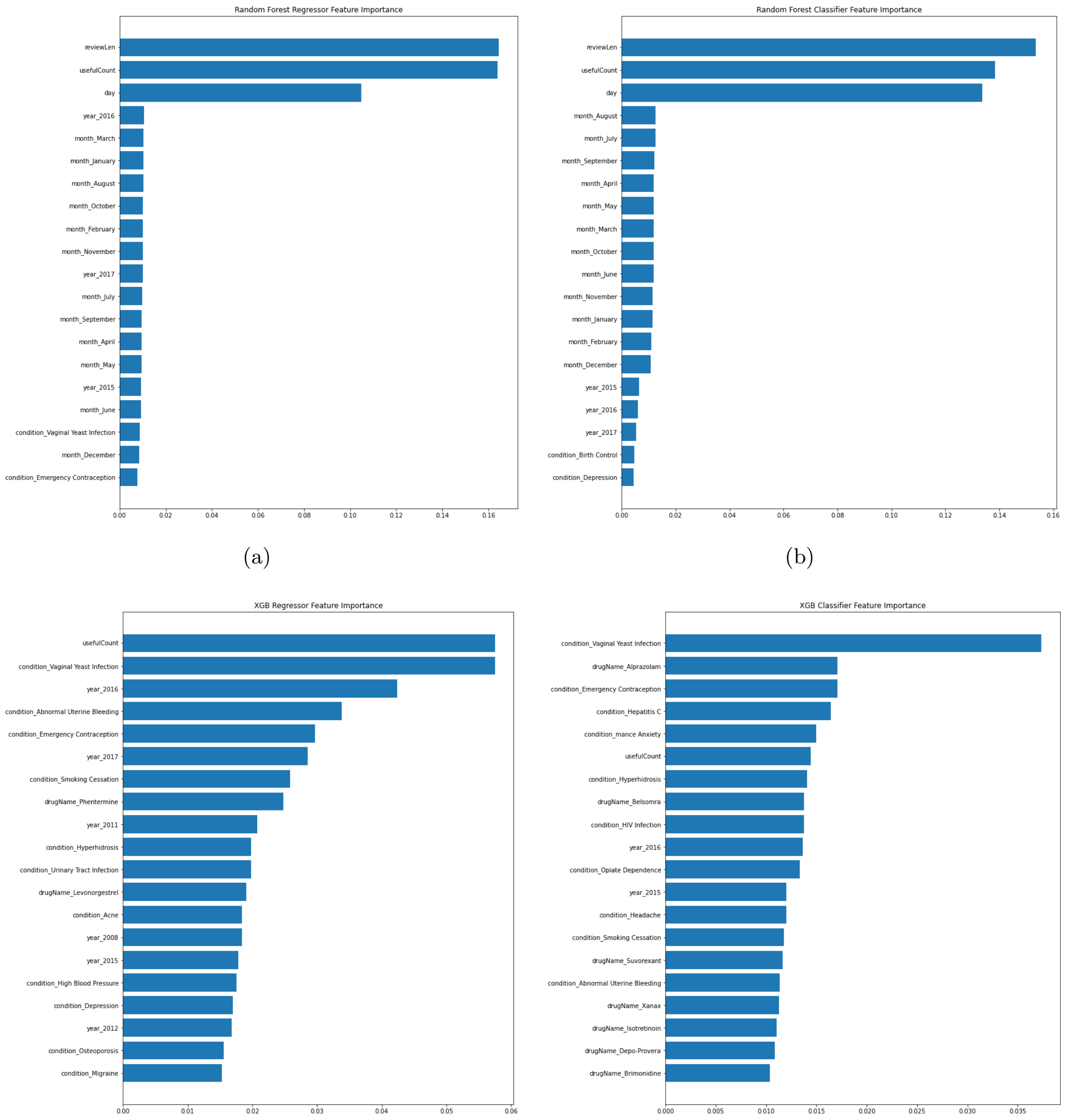


Figure 2: Top 15 conditions with the most counts



(a)

(b)

(c)

(d)

Figure 3: Feature importance using Random Forest and XGBoost for regression and classification

References

[1] Felix Gräßer, Surya Kallumadi, Hagen Malberg, and Sebastian Zaunseder. 2018. Aspect-Based Sentiment Analysis of Drug Reviews Applying Cross-Domain and Cross-Data Learning. In Proceedings of the 2018 International Conference on Digital Health (DH '18). ACM, New York, NY, USA, 121-125. DOI: https://dl.acm.org/doi/10.1145/3194658.3194677

[2] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[3] L. Breiman, "Random Forests", Machine Learning, 45(1), 5-32, 2001. DOI: https://doi.org/10.1023/A:1010933404324

[4] Hug, N., (2020). Surprise: A Python library for recommender systems. Journal of Open Source Software, 5(52), 2174, https://doi.org/10.21105/joss.02174

[5] Akhil Shiju, Zhe He, "Classifying Drug Ratings Using User Reviews with Transformer-Based Language Models", medRxiv 2021.04.15.21255573; doi: https://doi.org/10.1101/2021.04.15.21255573

[6] Mahboob, Khalid & S., Hina. (2018). Sentiment Analysis of Pharmaceutical Products Evaluation Based on Customer Review Mining. Journal of Computer Science & Systems Biology. 11. 10.4172/jcsb.1000271.

[7] Vijayaraghavan, Sairamvinay, and Debraj Basu. "Sentiment analysis in drug reviews using supervised machine learning algorithms." arXiv preprint arXiv:2003.11643 (2020).

[8] Gopalakrishnan, V., & Ramaswamy, C. (2019). Patient opinion mining to analyze drugs satisfaction using supervised learning. Journal of Applied Research and Technology, 15(4). https://doi.org/10.1016/j.jart.2017.02.005