Sentiment Analysis of U.S. Airline Twitter Posts

Huailin Tang htang73@wisc.edu
Jinghao Liu jliu975@wisc.edu
Max Zou zou47@wisc.edu

Abstract

This project aimed at predicting sentimental classification for the three class reviews (positive, negative and neutral) of airline comment tweets. We use over 16k tweets of airline comments, divided them into 80% of training set and 20% of testing set and processed them to classify the sentiments. We use Multinomial logistic regression and Ordinal logistic regression finally find Multinomial is the better accuracy for the three class airline comments analysis.

1. Introduction

In recent years there has been rapid growth in social media and review sites where a large number of comments are aggregated. Behind these comments, we can see the sentiments and thus know the overall opinion on the subject of matter. Labeling these comments would provide a clear summary of the opinion to the readers and the platform, so they could easily know what all readers think without browsing the comments one-by-one. These labels could be especially useful for websites without a rating system for individual users. It's also useful for relevant companies to improve their service. Based on the users comments, companies could give personalized service to each individual.

Twitter is one of the biggest social network media in which users usually post their positive or negative comments to the service. We can identify the users' attitude towards the service from their tweets. In this project, we will focus on identifying the sentiments behind the Twitter comments on U.S. airlines using Natural Language Process and other related machine learning knowledge. The sentiments are classified into three categories: positive, neutral, and negative. The goal of this project is to train several machine learning models, compare and choose the best one to classify sentiments.

We experiment with Multinomial logistic regression and Ordinal logistic regression to find the better algorithm for this sentiment analysis. We firstly extract the tweets and clean the text. Then we use 80% data for model training and 20% for the testing. Finally we compute the test accuracy and use confusion matrix to draw the conclusion for the sentiment analysis.

2. Related Work

Related Work There have been many works in the past that focused on sentiment analysis. In 2016, Mozetič, Grčar, and Smailovič published a paper about multilingual Twitter Sentiment Classification. Specifically, the team labeled 1.6 million tweets in 13 different languages. They trained the data and built multiple automatic sentiment classification models. Our work is similar to them, but in a smaller scale, single language, and using after-labeled dataset. Their method and training model might be helpful [5].

There are also other related natural language processing projects other than comments on twitter. In 2015, Fang and Zhan published a paper bout sentiment analysis using amazon product review data. Each review is tagged on positive or negative [4].

3. Proposed Method

3.1. Multinomial logistic regression

Multinomial Logistic Regression[3] is the regression analysis to conduct when there is a nominal dependent variable with more than two levels. Multinomial logistic regression is the extension of binary logistic regression.

3.2. Ordinal logistic regression

Ordinal regression[6] predicts labels that are discrete and ordered. Because the sentiment data has three classes with potential ordinal nature: positive, neutral, and negative, apart from the multinomial logistic regression, this paper also uses two varieties of ordinal regression. To implement this idea, the Python package mord, which is an implementation of the scikit-learn API, is used.

The two models this paper uses are the IT (Immediate-Threshold) variant and the AT (All-Threshold) variant. For the IT method, define h(z) := log(1+exp(z)). Then the objective minimizing function is

Where, are the training examples and $\{\mathrm{y}_1, ..., y_n\}$,$\mathrm{y}_i \in \{+1, 1\}$, are the labels. For the AT method, the minimizing function is

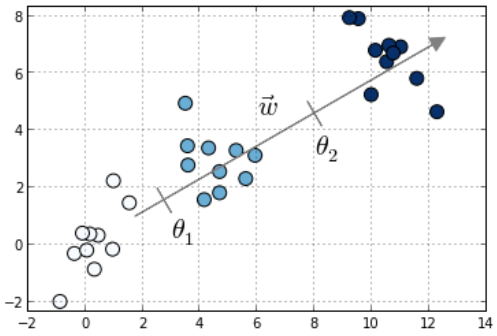

Figure 1. Ordinal logistic regression

$$J_{\text{Imm}} = \sum_{i=1}^{n} h(\theta_{y_i - 1} - \vec{x}_i^T \vec{w}) + h(\vec{x}_i^T \vec{w} - \theta_{y_i}) + \frac{\lambda}{2} \vec{w}^T \vec{w}.$$

Figure 2.

$$J_{\text{All}} = \sum_{i=1}^{n} \left[\sum_{k=1}^{y_i - 1} h(\theta_k - \vec{x}_i^T \vec{w}) + \sum_{k=y_i}^{l-1} h(\vec{x}_i^T \vec{w} - \theta_k) \right] + \frac{\lambda}{2} \vec{w}^T \vec{w}.$$

Figure 3.

4. Experiment

Multinomial Logistic regression and two types of ordinal regression/classification were employed to predict sentiment labels.The Confusion matrices were generated to visualize the model performances and evaluate their ability to make predictions of the sentiment of Twitter posts. Using scikit-learn and mord, we trained our model using processed data , then the models are tuned through altering their respectively hyperparameters settings via grid search.

4.1. dataset

The dataset uses Twitter posts about major U.S. airlines in February of 2015. This dataset consists of 14640 Twitter posts and airline sentiments (positive, neutral, and negative) classified by the contributors, with the airline information, airline sentiment confidence, and possible negative reasons. Figure 4 is the screenshot of what the data looks like:

	airline_sentiment	airline_sentiment_confidence	negativereason	airline	text
0	neutral	1.0000	NaN	Virgin America	@VirginAmerica What @dhepburn said.
1	positive	0.3486	NaN	Virgin America	@VirginAmerica plus you've added commercials t...
2	neutral	0.6837	NaN	Virgin America	@VirginAmerica I didn't today... Must mean I n...
3	negative	1.0000	Bad Flight	Virgin America	@VirginAmerica it's really aggressive to blast...
4	negative	1.0000	Can't Tell	Virgin America	@VirginAmerica and it's a really big bad thing...

Figure 4. The dataset

We visualize the number of each sentiment in the dataset. We can see most of the sentiments are negative on Figure 5, followed by positive, and neutral has the least number.

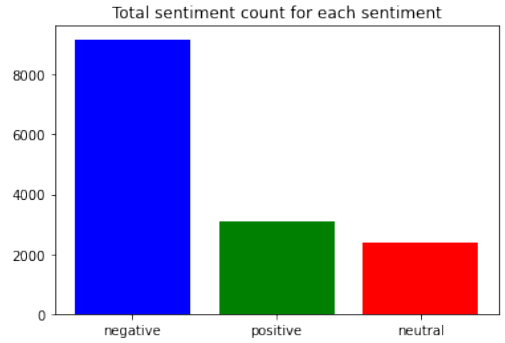

Figure 5. Sentiments counts

4.2. Language processing

4.2.1 Clean text

The raw data contains HTML markup, punctuation, and other non-letter characters. We remove all of them for simplicity. We used Python's regular expression (regex) library, re, to accomplish that.

Here is the comparison of raw text and text after prepossessing:

```
0                @VirginAmerica What @dhepburn said.
1    @VirginAmerica plus you've added commercials t...
2    @VirginAmerica I didn't today... Must mean I n...
3    @VirginAmerica it's really aggressive to blast...
4    @VirginAmerica and it's a really big bad thing...
Name: text, dtype: object
```

Figure 6. Raw text

```
0                  virginamerica what dhepburn said
1    virginamerica plus you ve added commercials t...
2    virginamerica i didn t today must mean i need...
3    virginamerica it s really aggressive to blast...
4    virginamerica and it s a really big bad thing...
Name: text, dtype: object
```

Figure 7. Pre-processed text

4.2.2 Text to tokens

We want to split the text corpora into individual elements. We simply split them into individual words by splitting the cleaned documents at their whitespace characters.

4.2.3 Word stemming

Word stemming is the process of transforming a word into its root form, which allows us to map related words to the same stem. For example, "running" and "run" have the same meaning in the text, so we want to lemmatize "running" to run in this project. We use the Natural Language

```
['virginamerica',
 'is',
 'anyone',
 'doing',
 'anything',
 'there',
 'today',
 'website',
 'is',
 'useless',
 'and',
 'no',
 'one',
 'is',
 'answering',
 'the',
 'phone']
```

Figure 8. splitted data

Toolkit for Python to implement the Porter stemming algorithm to accomplish this goal.

```
['virginamerica',
 'is',
 'anyon',
 'do',
 'anyth',
 'there',
 'today',
 'websit',
 'is',
 'useless',
 'and',
 'no',
 'one',
 'is',
 'answer',
 'the',
 'phone']
```

Figure 9. stemming text

4.2.4 Stop words

Stop-words are simply those words that are extremely common in all sorts of texts and probably bear no (or only a little) useful information that can be used to distinguish between different classes of documents. Examples of stop-words are is, and, has, and like. In order to remove stop-words from the Twitter text, we will use the set of 127 English stop-words that are available from the NLTK library.

4.3. Model training

4.3.1 Multinomial logistic regression

For multinomial logistic regression, we used the class LogisticRegression from scikit-learn. we used a GridSearchCV object to find the optimal set of parameters for our Multinomial Logistic Regression model using 5-fold stratified cross-validation.

Here are the hyperparameters and values we select for grid search:

- vect_ngram_range[1]: The lower and upper boundary of the range of n-values for different n-grams to be extracted. We select (1, 2) for unigrams and bigrams.
- vect_stop_words: Pass the string to the stop list to check for the stop word, then return the list of stop words. We select [stop, None] for checking or not checking the stop word.
- vect_tokenizer: Override the string tokenization step while preserving the preprocessing and n-grams generation steps. We select [tokenizer, tokenizer_porter] for simply splitting word or Porter stemmer algorithm.
- clf_C[2]: Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization. We select [1.0, 10.0, 100.0] for regularization strength.
- clf_solver: Algorithm to use in the optimization problem. We use 'newton-cg', 'sag', 'saga', and 'lbfgs' to handle multinomial loss for multiclass problems.
- clf_penalty: none, L1, or L2. We choose [L2, none] to use L2 penalty or no penalty based on the solvers we have.

We also use two dictionaries in the grid search. In the first dictionary, we used TfidfVectorizer with its default settings (use_idf=True, smooth_idf=True, and norm='l2') to calculate the tf-idfs; in the second dictionary, we set those parameters to use_idf=False and norm=None in order to train a model based on raw term frequencies.

4.3.2 Ordinal Logistic Regression

For the two ordinal regression models, the same language processing criteria are used, and same hyperparameters were examined during the grid search. The parameters of the ordinal regression models, however, were different as both the mord.LogisticIT and LogisticAT only have the following three parameters: alpha=1.0, verbose=0, maxiter=10000. As a result of that, only the parameter alpha is examined in the grid search.

5. Result

5.1. Multinomial logistic regression

The Multinomial logistic regression achieves a test accuracy of 80.46%, which is the highest score we achieved among the three models.

The best parameters set to achieve the test accuracy are: 'clf__C': 10.0, 'clf__penalty': 'l2', 'clf__solver': 'saga', 'vect__ngram_range': (1, 2), 'vect__stop_words': None, 'vect__tokenizer'.

The confusion matrix is used to visualize the test result. From the confusion matrix, we can see among 1836 actual negative comments, 1733 (94.38%) are predicted as negative; among 620 actual neutral comments, 343 (55.32%) are predicted as neutral; among 472 actual positive comments, 307 (65.04%) are predicted as positive. The negative comments have the highest prediction accuracy, and the neutral comments have the lowest prediction accuracy.

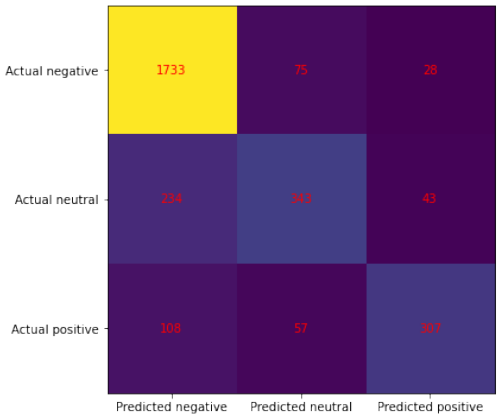

Figure 10. Confusion matrix of the test set of multinomial logistic regression

Observe that most of the error are caused by misprediction of the neutral and positive classes, in particular, the model has the tendency of predicting the sentiment of negative even though the actual sentiment is neutral or positive. Following by a examine of the confusion matrix of the training data set,

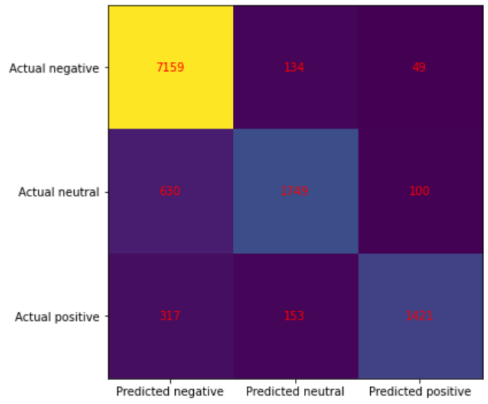

Figure 11. Confusion matrix of the training set of multinomial logistic regression

5.2. Ordinal Logistic Regression

The IT variant and the AT variant obtains very similar results, achieving 77.2% and 76.7% CV accuracy; 77.3% and 76.2% testing accuracy respectively. Comparing to the traditional multinomial logistic regression, we can see that although the ordinal methods have slightly lower accuracies, if we examine the confusion matrices,

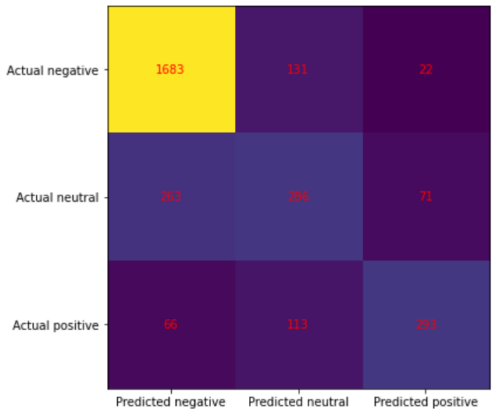

Figure 12. Confusion matrix of the testing data of ordinal logisticIT regression model

There are far less miss-classified actual positive-predicted negative cases (bottom left corner). Which is a notable improvement. This improvement is likely caused by the ordinal nature of the model since the positive class and negative class are further apart than either of their distance to neutral class.

One of the reasons that the two ordinal models had a slightly lower accuracy is that these two models have limited hyperparameter turnabilities. Because of the way the mord package is developed, there was essentially only one parameter, alpha, to pass to the regression, which limits the

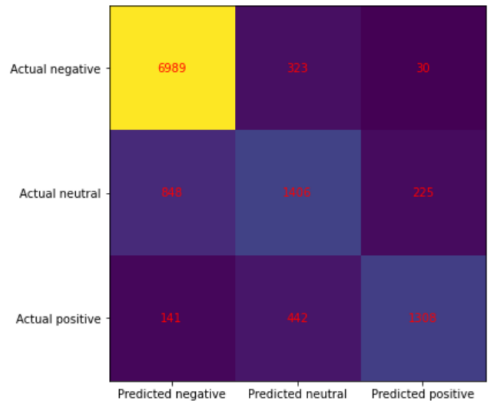

Figure 13. Confusion matrix of the training data of ordinal logisticIT regression model

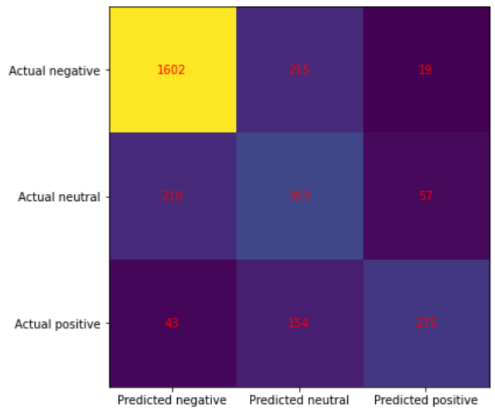

Figure 14. Confusion matrix of the testing data of ordinal logisticAT regression model

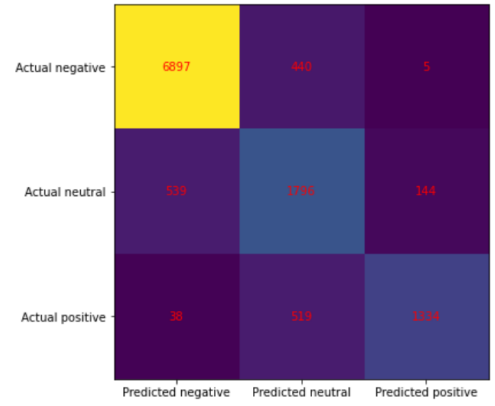

Figure 15. Confusion matrix of the training data of ordinal logisticAT regression model

potential of a better fit of the models. Another reason is that by applying the restriction of ordinal labels, we lost some flexibility of the model.

Also a similarity between the three models is that most of the error comes from miss-classifications relating to the neutral class. This is expected because of the nature of the data. By intuition the correlation between words in twitter review and neutral sentiment is less strong comparing to positive or negative sentiment. For example, reviews that contains more negative words could still yield a neutral label. A solution to this problem we propose is to study and includes punctuation marks like the exclamation point as additional features since by intuition, a comment with more exclamation marks or question marks are more likely to express the more extreme sentiments, rather than a neutral sentiment, which could contains more period marks.

6. Conclusion

This report presented three models for sentiment analysis of twitter comments about US airlines. The three classification models used are logistic regression, immediate-threshold variation of the ordinal logistic regression, and all-threshold variation of the ordinal logistic regression. The data were cleaned and processed with multiple language processing toolkit. The best fitting parameters of the three models were determined by grid search method. Then the model performance was evaluated through test set accuracy scores and confusion matrices. The implementation is done through python and its packages.

The traditional logistic regression performed the best among the three models, the two ordinal regression model had a slightly lower accuracy rate. This is mainly because of the limit of the python package and the restriction applied by the ordinal approach. However, benefiting from the ordinal nature, the ordinal methods provided far less miss-classified actual positive-predicted negative cases.

We propose a few ways that could potentially improve and expand our analysis. To extract more information from the data, punctuation marks from the original comments should be included as a feature. Other ordinal regression models could be fitted to improve the tunability of the algorithm. Finally, combination of words could be considered when constructing features because in some cases, combination of words could provide better significance.

7. Contribution

As a group, we decide to split the computational and writing task evenly.

- Jinghao Liu coded the data processing part and gave some suggestion for model selection and pipeline.
- Huailin Tang coded and tuned the Multinomial logistic regression.

- Max Zou coded and tuned the Ordinal logistic regression.

For the writing task,

- Jinghao worked on overall organization and related parts of the report, also the paper style.
- Huailin drafted the method and model training part.
- Max finished the rest of the model training, result and conclusion parts.

8. Code

All of the code for model or to generate the plots and result is on the GitHub:`https://github.com/LIU1202/STAT451_project`

References

[1] Sklearn.feature_extraction.text.countvectorizer.

[2] Sklearn.linear_model.logisticregression.

[3] J. Chen, B. Lewis, A. Marathe, M. Marathe, S. Swarup, and A. K. Vullikanti. Chapter 12 - individual and collective behavior in public health epidemiology. In A. S. Srinivasa Rao, S. Pyne, and C. Rao, editors, *Disease Modelling and Public Health, Part A*, volume 36 of *Handbook of Statistics*, pages 329–365. Elsevier, 2017.

[4] X. Fang and J. Zhan. Sentiment analysis using product review data. *Journal of Big Data*, 2(1):5, Jun 2015.

[5] I. Mozetic, M. Grcar, and J. Smailovic. Multilingual twitter sentiment classification: The role of human annotators. *PLOS ONE*, 11, 02 2016.

[6] F. Pedregosa-Izquierdo. Feature extraction and supervised learning on fmri : From practice to theory, Jan 2016.