Yelp Data Challenge: What Makes a Review Useful, Funny or Cool?

Ziyang Gao, Lichun Gao, Yang Wu, Mingjie Zeng

I. Abstract

In this project, we used the Yelp Data Challenge dataset to explore the useful/funny/cool tag with the text. The goal of this project is to determine the useful/funny/cool label of a given review text. Moreover, we want to extend the aim of deciding how useful/funny/cool a review is. Thus, we propose two main methods to achieve our goal: multi-label classification for tag identification, multi-class regression for tag score prediction.

We cleaned the raw data to get the binary label, and preprocessed the review texted with tokenization, stopwords removal and lemmatizing. We did exploratory data analysis to examine the preprocessed data and processed the labels with several normalization strategies such as outlier shrinking and z-score. We used two features for the model building: TF-IDF and Word2vec. We successfully built the multi-label classification model by using Binary Relevance, Classifier Chains, and Label Powerset. We also built the multiclass regression model with linear regression, KNN, Random Forest, Decision Tree, and SVR (MultiOutput Regressor and Regressor Chain)

II. Introduction

With the emergence of the Internet era and increased user engagement, user reviews have gradually gained importance in a variety of areas. Whether it is the e-shopping platform like Amazon or the travel related platform such as TripAdvisor, for the selection of merchants and products, besides referring to the official information, users rely more on the feedback given by other users. One of the most critical reasons is that the feedback given by other users can, in a sense, directly reflect the more realistic and practical user demands. In addition to providing users with more diversified information, the platform can also get more information from users' reviews, which are important for more accurate user profiling, more precise information pushing, and even better screening of merchants, etc.

For a platform like Yelp, user reviews may have a more important role to play. In terms of the evaluation for the food or service, users often need a wider variety of information and diverse perspectives to filter out the restaurants or other businesses they need. At the same time, in order to attract more new customers and have a better reputation, merchants also need a greater number of better and more informative customer reviews to compete with other similar merchants.

However, some businesses might take advantage of the attention of review in an improper way. Merchants might give customers a certain amount of discount or even pay the ghostwriters to write fake reviews for that merchant in order to get a higher rating or to take the impact of bad reviews away. This behavior can significantly improve the merchant's rating, but creating tons of fake reviews. In most cases, those fake reviews often can only show a simple compliment attitude, but there is little substantive information. Such reviews are not helpful for users to choose a merchant, and can also cause the platform to make the wrong recommendation of the business based on the rating, further causing a bad impact on the user's satisfaction with the platform.

To solve this problem, Yelp has a feature that allows users to hit the like button of the “useful” label to the reviews they find useful, which helps to manually filter out the reviews that are genuine and have enough valid information. There are another two tags called “Funny” and “Cool” to allow users to vote for the reviews they find interesting. However, the manual vote might take a long time to validate the properties of the review, we need a more efficient way to identify the tags of a review to be whether useful/funny/cool.

To achieve this, in this paper, we propose to build a classification model to predict these attributes of reviews using the information provided by Yelp for each review of useful, funny and cool. Further, we also intend to build a regression model to determine how useful, funny, or cool the review is. We expect that this

model will help the platform to recommend more valuable reviews to users by determining these three attributes of reviews, and also further evaluate businesses and achieve better user pushing. At the same time, by judging whether reviews are useful, interesting and cool, it can also further filter out fake reviews and reduce the phenomenon of click farming.

Moreover, for other review-related analysis, filtering out valuable reviews can also provide a better user experience, even change the layout of the Yelp app. Temporarily, Yelp has a search based interface, and it ranks the review based on the date and the review writer's popularity. But sometimes, users might not have a very specific intention for searching, and the top reviews are not the most useful ones or do not fit the user's reading habit. By implementing the label identification, we can design a new feeding system which is able to learn the user's habits for preference of useful/funny/cool content. This customer specific review showing can increase the stickiness of users and enhance the enthusiasm for the users to write high-quality reviews.

III. Related work

Yelp Data Challenge. Yelp data is a classical dataset for personal, educational and academic research. It is available as JSON files for students to learn NLP. The previous works were focusing on latent features extraction and business recommendations. For example, Xu [10] used double embeddings and CNN-based sequence labeling method to extract latent business aspects. Liu [6] used a phrasal segmentation method to automatically extract high-quality phrases from the yelp text dataset. On the other hand, some related works are based on customer recommendation. Li [4] developed a recommendation system to detect the potential customers for business. Another recommendation system is focusing on solving the individuals' needs and service providers' needs by using Yelp's social network data. All the existing works make different contributions from different perspectives. Our work is to analyze the review data, and classify them into different review types.

Multi-task learning. In the machine learning field, people typically want to train and optimize one single model for more than one task. Multi-task learning works much more efficiently when the tasks are not independent. In the NLP domain, Liu [7] applied BERT embedding layers on CNN models to achieve multi-task purposes on robotic pushing, poking, and grasping. The multi-task architecture, named MT-DNN, achieved SOAT performance on eight of nine GLUE tasks. In our work, we build multi-output regression models for click count analysis on each type of review..

IV. Methodology

A. Data Preprocessing

After extracting and filtering from original review data, the dataset has 83926 rows and 8 columns. Each row represents one review. Columns indicate binary labels and click counts for each class. There is no missing value in the whole dataset. Since we will build linear models later, we set the outliers to the corresponding upper or lower range of normal data. After cleaning data, the binary label ratio (0/1) for "useful" "funny", and "cool" are 1:3, 5:3, 1:1, respectively. Except the lower range and upper range in each class, the distributions of click counts are near to normal distribution. Then, we split 80% data as training set, 20% as testing set.

In the feature engineering part, we used two kinds of features, TF-IDF and word2vec embedding. TF-IDF is one of the simple and robust methods to understand the content of text. The idea is to find the most essential words for the content of each document by decreasing the weight for commonly used words and increasing the weight for words that are not used very much in a collection of documents. Word2vec is also a classical method that creates word embedding in the field of NLP. Word2vec takes in words from a large corpus of texts as input and learns to give out their vector representation. The main learning methods are the continuous bag-of-words model (CBOW) and skip-gram model. In our work, we used pre-trained word2vec on google news. The vector dimension is 300.

B. Multi-label Classification

Our goal on this stage is to classify the comments' labels by using the appropriate model. As we are

analyzing the problems with several labels, we compare the multi-class classification with multi-label classification to find which one fits our proposed problems the best. The multi-class classification will only output one class and classify the problems which only have one answer at once. While the multi-label classification could have multiple output classes, it will help us to solve the problems which depend on more than one solution at once. Looking back at our data, we have three labels for each comment, and what we want to predict is more than just a simple label. Furthermore, J. Huang et al [3] stated that multi-label learning aims at dealing with data that have multiple class labels simultaneously. Then we come out with using multi-label classification as our proposed solution for the classification stage.

J. Huang et al [3] suggested that it is effective to have problem transformation when we are doing multi-label classification. In order to build the models in an easier way, we could convert the multi-label dataset into one or more single-label datasets. The reason why single-label datasets are helpful is that they are machine-readable and easy with building models.

G. Tsoumakas [9] suggested that Multi-label classification problems can be grouped into two main categories: 1) problem transformation methods, 2) algorithm adaptation methods. For the first portion, we have to find the techniques which help us to do the transformation. There are three techniques, which are 1) binary relevance, 2) classifier chains, and 3) label powerset. Each of them has distinct advantages and drawbacks so that they play different characters when helping us to transform.

For binary relevance, each label will be treated independently to separate the multi-label classification as single-label classification. However, the label correlation may be lost during the process. The one that does a better job on label correlation is classifier chains. The multiple classifiers are connected in a chain. The input data will build the first classifier in the chain, and the combined inputs together with previous classifiers will train the following classifiers. The accuracy for the classifier chains technique is relatively low, to solve this issue, we have the last technique introduced in our experiment: label powerset. The goal of label powerset is to find the combination of unique labels so that it could assign different values for them. By applying a label powerset, each multi-class classifier will then be trained with unique label combinations with higher accuracy.

After we decide the methods to use, we begin to think about the models we want to use. A. C. Carvalho [2] has summarized several models to give us an insight into picking suitable models. The decision tree is used and modified by many other researchers for their study on multi-label classification problems. Support Vector Machines (SVM) is employed in many multi-label classification problems to minimize the ranking loss. Finally, we decided to use the three classification models: logistic regression, random forest, and SVM.

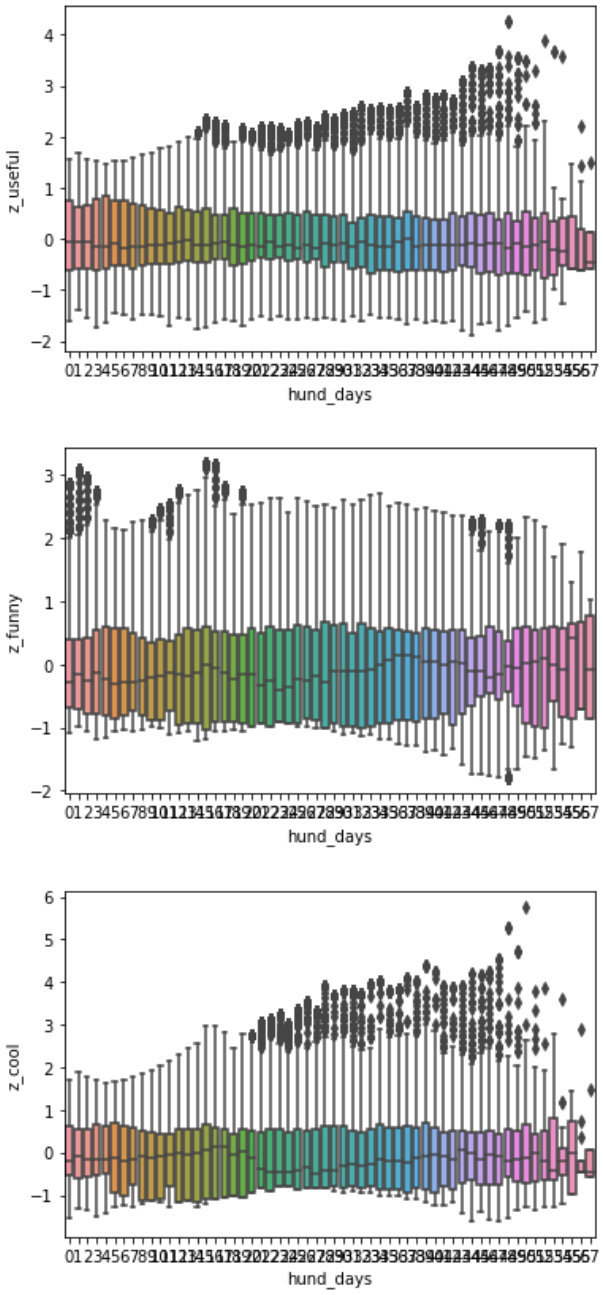
Subsequently, we want to find the metrics to measure our models and methods. This will be important because measurement could help us to understand clearly about our model choosing. R. B. Pereira [8] introduced the metrics accuracy and hamming loss. Accuracy is very often to see for the machine learning models measurement. Hamming loss is one of the very famous multi-label classification measures, it accounts on the prediction error and the missing error. The prediction error happens when incorrect labels are predicted, and missing error appears when the relevant label is not predicted. In short, hamming loss could help us to determine the fraction of incorrect predictions of the given model.

For the multi-label classification to solve Yelp's comment label problem, we have decided to use the three models: 1)logistic regression, 2) random forest, 3) SVM, with three methods: 1) binary relevance, 2) classifier chains, 3) label powerset. We will use accuracy and hamming loss to measure our solutions.

C. Normalization and Score Computation

To have a more scientific way for the evaluation of the useful/funny/cool level, we want to transform the tag votes into the score. We observe the trend of the tag has some bias with the dates. When we plot the useful/funny/cool with per hundred days, we find the uneven feature of the time with the votes. Thus we

performed the z-score of per hundred days to normalize and calculate the useful/funny/cool score.



D. Multi-output Regression

To accomplish our task of predicting how useful or funny or cool a review is, and also considering that each review in the yelp dataset has all these three attributes at the same time, more importantly, they are not totally independent of each other, we propose to use multi-output regression models to achieve the task goal.

Unlike the simple regression problem, which usually outputs only one predicted value, multiple output regression is a regression problem that involves predicting two or more values given an example input [10]. It's the ideal solution that this model can directly predict three scores for each of the labels at the same time just given the only input text.

So how can we build multi-output regression models? There are some algorithms that support regression for multiple outputs, such as decision trees and random forest. But there are still algorithms that do not natively support predicting more than one output. For those algorithms, there are special models that can wrap them and make them fit for the multi-output tasks. Take SVM for one example, SVM doesn't support multiple outputs, but after wrapping it, the wrapped SVM model can handle multi-output tasks.

The idea for using regression models designed for predicting one value for multi-output regression is to split a multi-output regression problem into multiple single-output regression problems. There are two main methods categories, one is inherently multi-output regression algorithms, the other is wrapper multi-output regression algorithms.

A direct approach to multi-output regression is to split the regression problem into separate problems for each target variable to be predicted. This assumes that the outputs are independent of each other, which might not be a correct assumption. Another approach to using single-output regression models for multi-output regression is to create a linear sequence of models, which develops a sequence of dependent models to match the number of numerical values to be predicted. This method considers the dependency of our three outputs but also needs different assumptions that which attribute can have a greater impact on the other attributes.

Figure 1 below shows the workflow examples for direct multi-output and chained multi-output.

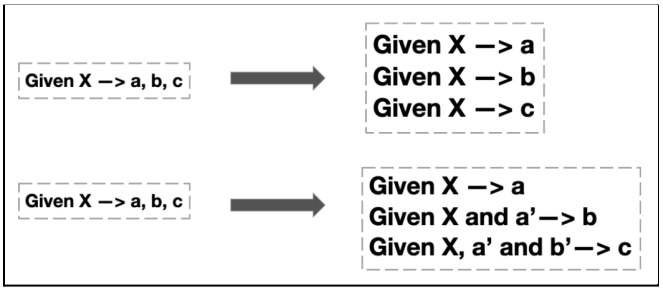


Figure 1. First row is the example for Direct Multioutput, second row is the example for Chained Multioutput.

For a better comparison between different kinds of algorithms, including multi-output inherently algorithms and wrapped models for multiple outputs, our project applied a variety of different approaches and aims to discuss the results achieved with respective methods to see the algorithm impact on the specific task: how useful, funny and cool a review is?

First, based on the normalized scores for each of the three labels, there is a score range for each label, which is essential for predicting the sentiment degree. In the experimental session, we apply 4 different inherently multi-output supported algorithms, including linear regression, k-nearest, random forest, and decision tree. For the wrapper multi-output regression part, we choose a representative algorithm SVM for our task. In order to get the comparable results of Direct Multioutput and Chianed Multioutput, we also apply different regressors on SVM to see the disparity.

For a better evaluation, we pick up 4 measuring metrics that are appropriate for regression algorithms, including Mean Squared Error(MSE), Root Mean Squared Error(RMSE), Mean Average Error(MAE), and R-Squared.

V. Results

A. Multi-label Classification

Applying sklearn, skmultilearn, and so forth packages, we have come out with different results separately for the TF-IDF and Word2Vec features. Each of our models has the result with the three techniques and two features. The accuracy and hamming score for them are as shown in the table.

Method	mlb_estimator	accuracy	hamming_score
TFIDF - Logistic Regression	binary_relevance	0.3909415971394517	0.27691696466044496
TFIDF - Logistic Regression	classifier_chains	0.3909415971394517	0.27691696466044496
TFIDF - Logistic Regression	labelpowerset	0.42193087008343266	0.28843861740166865
TFIDF - Random Forest	binary_relevance	0.38498212157330153	0.278108859753675
TFIDF - Random Forest	classifier_chains	0.4028605482717521	0.2709574890742948
TFIDF - Random Forest	labelpowerset	0.42073897497020263	0.29360349622566545
TFIDF - SVM	binary_relevance	0.38974970202622167	0.288041319030592
TFIDF - SVM	classifier_chains	0.38974970202622167	0.288041319030592
TFIDF - SVM	labelpowerset	0.4302741358760429	0.2896305125148987

Table 1. Measuring scores for different models and feature selection methods with TF-IDF.

Method	mlb_estimator	accuracy	hamming_score
Word2Vec - Logistic Regression	binary_relevance	0.3766388557806913	0.2789034564958284
Word2Vec - Logistic Regression	classifier_chains	0.3766388557806913	0.2789034564958284
Word2Vec - Logistic Regression	labelpowerset	0.4195470798569726	0.2912197059992054
Word2Vec - Random Forest	binary_relevance	0.4052443384982122	0.2689709972189114
Word2Vec - Random Forest	classifier_chains	0.3909415971394517	0.27771156138259834
Word2Vec - Random Forest	labelpowerset	0.43146603098927294	0.28684942391736196
Word2Vec - SVM	binary_relevance	0.39451728247914186	0.2765196662693683
Word2Vec - SVM	classifier_chains	0.39451728247914186	0.2765196662693683
Word2Vec - SVM	labelpowerset	0.42073897497020263	0.2924116011124354

Table 2. Measuring scores for different models and feature selection methods with Word2Vec.

From Table 1 and Table 2, We could see the filter TF-IDF and Word2Vec have similar results which shows us that the feature is not a critical factor here in the multi-label classification. However, we could see there do exist slight differences between each model and metric. To show the straightforward comparison, we have the following figures to show the feature comparison in a clearer way.

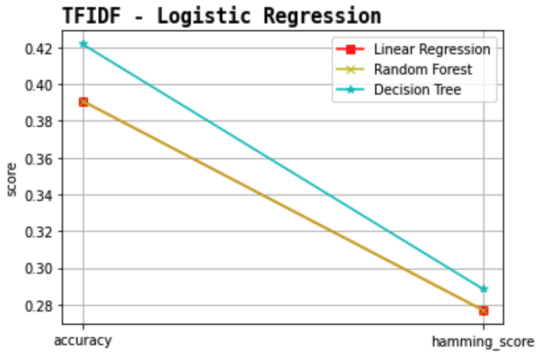


Figure 2. Measuring scores for Logistic Regression with TF-IDF.

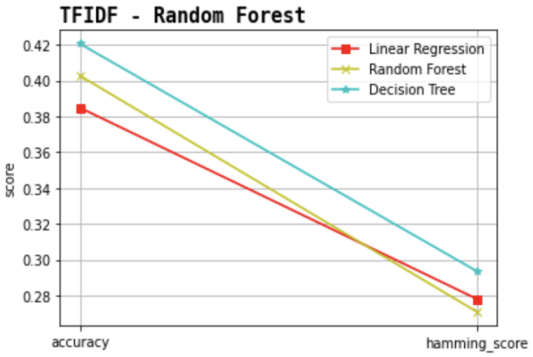


Figure 3. Measuring scores for Random Forest with TF-IDF.

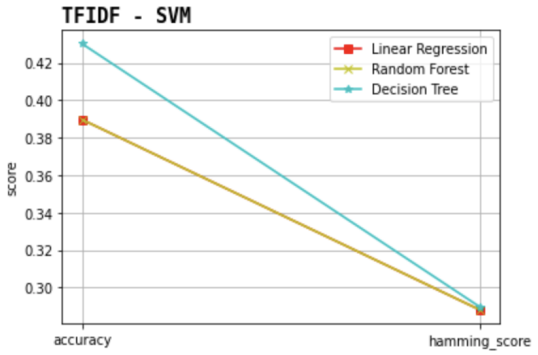


Figure 4. Measuring scores for SVM with TF-IDF.

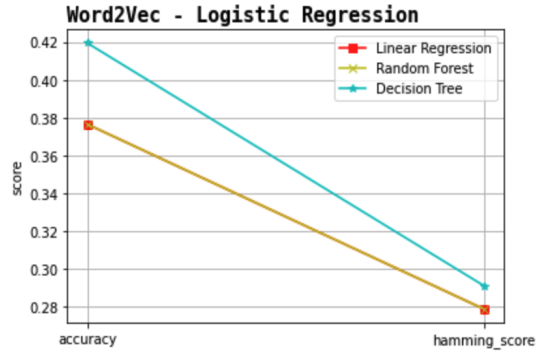


Figure 5. Measuring scores for Logistic Regression with Word2Vec.

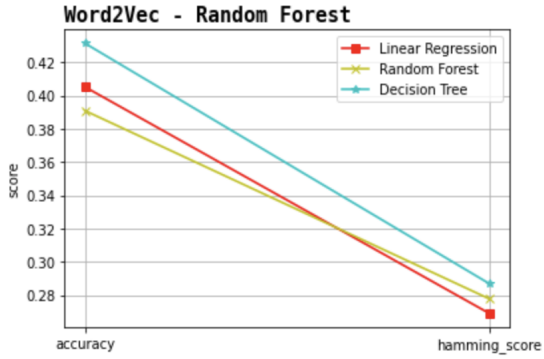


Figure 6. Measuring scores for Random Forest with Word2Vec.

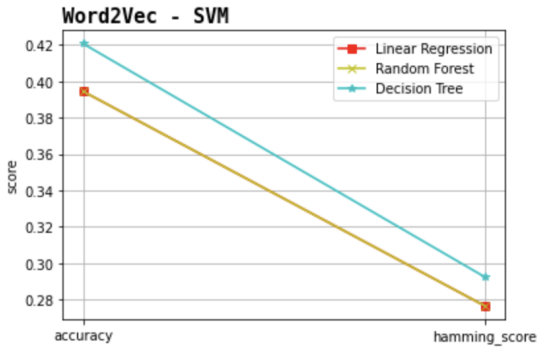


Figure 7. Measuring scores for SVM with Word2Vec.

From the figures shown above, we can see that random forest does the best job with Word2Vec. SVM with TF-IDF also does a better job compared with others.

The overall accuracy score is not that high, from my perspective, the reason may be that the way that people give labels to comments may vary from many factors. For example, two viewers may give very distinct labels to the same comments. Under this condition, our accuracy score may not be so satisfying. And we have a relatively better hamming score, this represents that our model is capable of predicting to a large extent.

B. Multi-output Regression

We apply 2 methods for feature selection, TF-IDF and Word2Vec, and separately feed the features to the models we choose before. Figure 2 below shows the entire measuring scores for all the models we experiment. We are aiming to get smaller scores for the better prediction results. We can see from the scores that expect for the differences of R-Squared scores between linear regression and SVR with TF-IDF and Word2Vec, there are no significant gaps in other indicators.

Method	MSE	RMSE	MAE	R-Squared
TFIDF - Linear Regression	0.6853857979599436	0.8278803041261119	0.6411115414057668	0.3021841300063293
Word2Vec - Linear Regression	0.8325556307210368	0.9124448644828008	0.6979680816051165	0.1524191078393177
TFIDF - K-Nearest	0.7446548296979877	0.8629338501287267	0.6702153622706692	0.2419159337890197 7
Word2Vec - K-Nearest	0.7639216277423456	0.8740261024376478	0.6792615413720968	0.2222711409044614 3
TFIDF - Random Forest	0.3566786241916813	0.59722577321452	0.4096955829560201	0.6367505112711895
Word2Vec - Random Forest	0.3599642341720213	0.5999701944030398	0.4122638377017 42	0.6334021816911375
TFIDF - Decision Tree	0.3821546642850764	0.6181865934194203	0.2730110347877886	0.6108191139659797
Word2Vec - Decision Tree	0.4215603233905168	0.6492767694831818	0.2849095897344767	0.5706002679045 9
TFIDF - SVR(MultiOutputRegressor)	0.7527118874029148	0.8675896999174868	0.6004562673472121	0.2336111367267351 4
Word2Vec - SVR(MultiOutputRegressor)	0.8654279432961328	0.9302837971802652	0.6791484176277084	0.1189964463978385 2
TFIDF - SVR(RegressorChain)	0.7526402373572249	0.8675484063481558	0.600448450867 26	0.2336829272385209
Word2Vec - SVR(RegressorChain)	0.8651992748144194	0.9301608865214767	0.6790205290254425	0.1192408441776252 2

Table 3. Measuring scores for different models and feature selection methods.

For a more straightforward comparison between different multi-output models, we perform a comparison of the results according to the respective feature extraction method. Figure 3 and Figure 4 below visualize the magnitude of these measuring metrics and the prediction accuracy of each model for the degree of useful, funny and cool of a review.

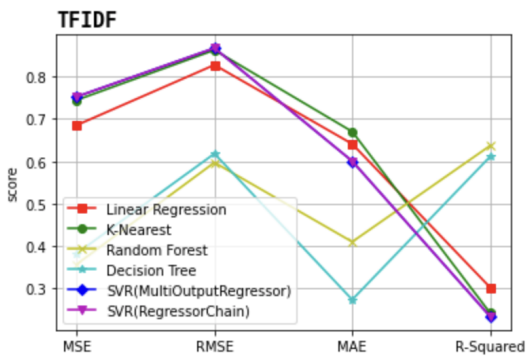


Figure 8. Measuring scores for different models with TF-IDF.

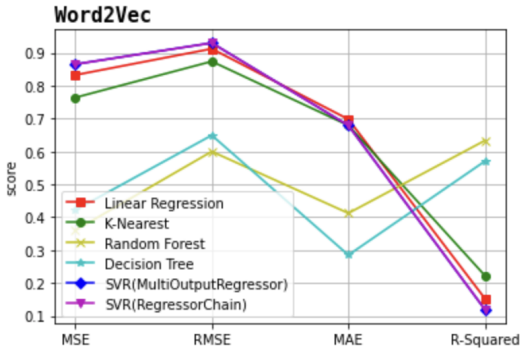


Figure 9. Measuring scores for different models with Word2Vec.

For TF-IDF, we can notice from the Figure 3 that for all the measuring metrics MSE, RMSE, MAE, and R-Squared, random forest and decision tree gain the best scores, Actually, from the formulas of these measuring metrics it is clear that for MSE, RMSE, and MAE, scores will be very different depends on the order of magnitude of the data. For example, if our model is predicting house prices, the error unit may be at the $10,000 level, and we will get an integer like 3 or 4 for the scores. But if we are predicting height, the unit of error might be centimeters, and we would get a decimal number like 0.3 for the scores. The readability of the results is not very high. However, R-Squared is very similar to the accuracy metric used in measurement classification algorithms and its score ranges from 0 to 1. So for this R-Squared indicator, the higher the score, the better the prediction result. Therefore the best performing models are still random forest and decision tree. And we can tell that the prediction results of direct multi-output method and chained multi-output method are basically similar. For Word2Vec, the results are pretty much the same as for the TF-IDF.

To explore whether the prediction ability of each model differs for these three labels, useful, funny, and cool, and whether the three labels affect each other, we respectively list the measurement scores of some models for the prediction results of different labels. The results are as follows.

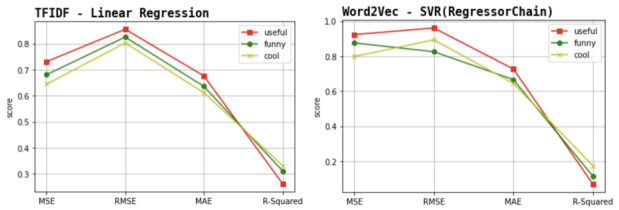


Figure 10. Measuring scores for useful, funny, and cool.

We speculate that there are more distinctive words or phrases in the reviews labeled as both “cool” and “funny”, while the attribute of “useful” is more difficult to judge because it may not be distinctive enough in terms of word usage. However, if we want to further investigate the specific reasons for this result, we may need to try more feature extraction methods to figure out what features make a review have these attributes.

VI. Discussion

We successfully built two models for predicting the useful/funny/cool labels based on the review text, in both classification and regression angle. In terms of predicting the label classification, we achieved a decent accuracy in all methods. Also, the individual label hits higher accuracy than the multi classification result, and the useful tag has the highest accuracy among all three tags. The highest accuracy result can achieve about 70% for each individual label, however, only 40% overall accuracy for the multi-labels. One important thing in the multi-label classification is the independence, however, we find our labels dependent on each other. Most funny and cool labels are co-occurring with useful, which might cause bias to our prediction result.

The regression model has some limitations in terms of the multi-output for the useful/funny/cool score. We observe the highest accuracy in the random-forest model in both tfidf and word2vec embedding. Also, the useful label has the best RMSE among all the labels. From the current regression result, we still have a long way to go to tell how useful/funny/cool a review is based on the given text.

VII. References

[1] Borchani, H., Varando, G., Bielza, C., & Larranaga, P. (2015). A survey on multi-output regression. *Wiley*

Interdisciplinary Reviews: Data Mining and Knowledge Discovery, *5*(5), 216-233.

[2] de Carvalho, A. C., & Freitas, A. A. (2009). A tutorial on multi-label classification techniques. *Foundations of computational intelligence volume 5*, 177-195.

[3] Huang, J., Li, G., Huang, Q., & Wu, X. (2016). Learning label-specific features and class-dependent labels for multi-label classification. *IEEE transactions on knowledge and data engineering*, *28*(12), 3309-3323.

[4] Li, R., Jiang, J. Y., Ju, C. J. T., & Wang, W. (2019, January). CORALS: Who Are My Potential New Customers? Tapping into the Wisdom of Customers' Decisions. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* (pp. 69-77).

[5] Liu, G., Lin, Z., & Yu, Y. (2009). Multi-output regression on the output manifold. *Pattern Recognition*, *42*(11), 2737-2743.

[6] Liu, J., Shang, J., Wang, C., Ren, X., & Han, J. (2015, May). Mining quality phrases from massive text corpora. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data* (pp. 1729-1744).

[7] Liu, X., He, P., Chen, W., & Gao, J. (2019). Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*.

[8] Pereira, R. B., Plastino, A., Zadrozny, B., & Merschmann, L. H. (2018). Correlation analysis of performance measures for multi-label classification. *Information Processing & Management*, *54*(3), 359-369.

[9] Tsoumakas, G., Katakis, I., & Vlahavas, I. (2006, September). A review of multi-label classification methods. In *Proceedings of the 2nd ADBIS workshop on data mining and knowledge discovery (ADMKD 2006)* (pp. 99-109).

[10] Xu, D., Shi, Y., Tsang, I. W., Ong, Y. S., Gong, C., Shen, X. (2019). Survey on multi-output learning. *IEEE transactions on neural networks and learning systems*, *31*(7), 2409-2429.

[11] Xu, H., Liu, B., Shu, L., & Yu, P. S. (2018). Double embeddings and cnn-based sequence labeling for aspect extraction. *arXiv preprint arXiv:1805.04601*.