Name: Bharath Ramagoni UNT ID: 11139578

As my honor as a UNT student, I have neither given nor received unauthorized assistance on this work

Honor code: Bharath Ramagoni

Title:

**CLASSIFICATION OF POSTS BELONGING TO ONE OF THE TOP 100 SUBREDDITS**

**Abstract:**

Many online communities discuss the issues of classification of posts to a Subreddit. But none of prior research has directly addressed this exact same issue as mine. There are some students from UCSD who addressed similar issue of classification using different methods (Link is given in References section). Our main goal is classify the posts so that we can analyze the factors affecting the predictions and use them to our advantage. It is will do this by learning unique and informative features of the posts of these subreddits. Features such as Nouns used in ‘titles’ and ‘selftext’ sections and authors of posts are taken as the main features to classify the posts. There are also other features such as ‘dates’ and ‘Over\_18’ used to obtain desired results. Given the accuracy I have achieved, research implies each post is about 40 % unique to one of Top100 subreddits.

**Introduction:**

The problem of a post belonging to subreddit is being addressed here. Given a post, we predict the subreddit. By this prediction, we can learn about the content of this post by looking at this prediction.

It is useful for new users to post in the right subreddit among millions of subreddits. We can recommend the subreddits by rank to which the post is revelant. Since, subreddit owners tend to delete the posts if they feel the posts do not belong there, this prediction helps in avoiding that.

**Definitions**:

Mcnemar’s test: “*In statistics,****McNemar's test****is a statistical test used on paired nominal data. It is applied to 2 × 2 contingency tables with a dichotomous trait, with matched pairs of subjects, to determine whether the row and column marginal frequencies are equal (that is, whether there is "marginal homogeneity"). An application of the test in genetics is the transmission disequilibrium test for detecting linkage disequilibrium*.” (Reference-‘wikipedia’)

**Data:**

Reddit has provided two files Submissions and Comments. It is open source and can be obtained off the internet. The file ‘Submissions’ contains all the posts to each and every subreddit. It is about 65Gigabytes in size. The file ‘Comments’ consists of all the comments to each and every post. We only use the file ‘Submissions’ to classify the posts. The data in this file gives information about each post such as creation date, author, subreddit to which it belongs etc .To put it simply, the file has features describing a post which we use to create our model. Each instance has 24 features and values of features as shown below.

Example:

null,"electionspeak",null,null,null,1199249977,1199249977,null,1,false,null,"64b3l",false,null,null,0,null,false,0,"","reddit.com","Huckabee: I did not free Dumond ",1,"http://www.electionspeak.com/USElectionVideos/298"

Order: approved\_by, author, author\_flair\_css\_class, author\_flair\_text, banned\_by, created, created\_utc, distinguished, downs, edited, gilded, id, is\_self, link\_flair\_css\_class, link\_flair\_text, num\_comments, num\_reports, over\_18, score, selftext, subreddit, title, ups, url

**Data distribution:**

I have divided the data such that 50% of posts of each subreddit go to training and 20 % to validation and rest to Test set. And this is done randomly i.e. which 50 % of posts of a subreddit go to training is decided randomly by the program. This is done to achieve independent and identically distributed datasets. Randomness ensures that we did not overfit our model.

**Approaches:**

**Hypothesis**:

Considering the features used create model, the best among those is “Nouns obtained from Titles and selftext sections”. Thus, this seemed to be the hypothesis worth stating. Remaining features implicitly state their own hypothesis such as dates,Over\_18.

1. The uniqueness of a post to a subreddit is obtained from Nouns and Adjectives used in the posts.
2. Authors are indigenous to their subreddits.

Approaches to obtain infer first hypothesis:

Using python programming language to perform the project:

1. Extraction of nouns from ‘title’ and ‘selftext’ is first step.
2. To extract, we need Parts of Speech tag of each word in the title . This done by a Package namely NLTK (Natural language tool kit). It is an open source package which can be downloaded off the net. (Link to the package is given in references). Using inbuilt methods of this package such as tokenize.word\_tokenize (), pos\_tag() we can obtain nouns from titles and selftext(if not null)
3. We store these nouns in dictionary with subreddit in which these nouns have occurred as Key.

Eg: Dict[‘Askreddit’]={Bubby:10,anurag:1,…} It is dictionary of dictionaries. It means ‘Bubby’ as occurred 10 times in subreddit “Askreddit” and ‘anurag’ occurred once in subreddit “Askreddit”.

1. Thus creating model, we calculate post’s score to each subreddit. The highest score is given as predicted Subreddit to that post.

**Datasets usage:**

*Training data*: To obtain Nouns, Authors, dates of the posts belonging to Top 100 subreddits.

*Validation data*: This data is used to check how the model is performing with given features. At first, only nouns are taken for model creation. Adding more features was advised. Thus more features are added to the model such as Authors, dates and also Over\_18 to filter out unnecessary subreddits during prediction. For example, in model creation, posts are sorted according to the time of creation of post. If the first post of a subreddit ‘y’ is on time ‘x’, if the post to be classified as created time as ‘j’ which is lower than ‘x’ then that post cannot belong to the subreddit ‘y’. Filtering out such subreddits before sending for predictions, we can save time and obtain more accuracy. We use this to tune our model. Negative example, I added ‘verbs’ in model creation which gave lower accuracy on validation. Though, it seemed illogical to me, I had to remove ‘Verbs’ from model creation since it gave bad results in practice.

*Test data:* This data is used to test our model predictions and also to check how reliable the model for predicting instances of real world.

**Evaluation*:***

To ensure the results I have obtained, K-fold is being done. In each fold, we obtain new training and test sets and we calculate the Fscore of each fold and average them in the end.

**Evaluation Metrics:**

Accuracy: Number of matches divided by the total number of predictions made. To obtain percentage , multiply by 100.

F1score: Since it already mentioned that there are dictionaries containing True positives, False negative, False positives . From these dictionaries, we can calculate Precision and Recall. Thus we can calculate F1score each subreddit and club them in the ratio of their Numbers in total. Fscore is helped finding which subreddit is often mistaken as some other particular subreddit .Eg: “Funny” is often mistaken as “Humor”.

**Experimental Sections:**

1. **Methods:**

Main method:

Firstly, the file is read by line and features relevant to our model are taken and a tuple is created. To get top 100 subreddits (‘reddit.com‘ is omitted . At the beginning of reddit, there are no subreddits, just posts on the home page. When they introduced the concept of subreddits, they put all the posts which are posted until then in “reddit.com”. Thus it has posts with many different topics and has bad influence on any model which tries to predict posts), we count the occurrence of each subreddit and store them in form of key,value pairs in a tuple. After doing so, we sort them according to the counts and cut the top 100 from the list of tuples. Now, we filter out the instances which do not belong to any one of the top 100 subreddits.

We then sort these instances according to time of creation, thus obtaining times of first post made in each subreddit. We also obtain the authors in subreddit. We store authors in dictionary of sets and, dates in dictionary with subreddit and date as Key and value.

Now, we divide the data into three subsets in ratio 5:2:3 for training, validation and test sets respectively. The data is divided in such a way that out of 10 instances belonging to a subreddit 5 of them go to training , 2 go to validation set and 3 go to test set. Decision of which 5 instances out 10 instances go to training is done randomly by the program (Used random module in python) in each iteration of K-fold.

We send training set to create model i.e. collecting nouns from each subreddit. We take only nouns out of all the tags because Nouns are unique and tend to repeat in the posts of a given subreddit. For example, subreddit “Harry potter” has names of characters belong to that subreddit such “Dumbledore, harry ..etc” . It is highly unlikely to find same names in other subreddits. If we consider a subreddit related to “harry potter” it is unlikely that both of them will be in top 100 subreddits. Model is a dictionary of dictionaries. After model creation, we iterate through each item in the test set and predict the subreddit by sending model, dates, authors,over\_18 as parameters.

In prediction, we filter out subreddits using dates ,over\_18 and authors and use our model to predict from possible subreddits. The reason for feature ‘over\_18’being taken is to classify subreddits such as ”nsfw” whose ‘over\_18’ column has “true ” most of the time. So while predicting, we can use the over\_18 column to see its probability of belonging to subreddits such as “nsfw”. We use another dictionary (with ids and true subreddit of an instance) to check whether the prediction made is correct or not. If it is correct, we increment the variable ‘matches’ by ‘1’ which is used to find the accuracy after predicting all the items in test set. There are other dictionaries in use to keep counts of True positives, False negatives, False positives which are used to calculate F-score.

**McNemar’s Statistical test:**

Prediction for an item is made twice using Baseline model and experimental model simultaneously. Thus, obtaining required metrics (test1-positive, test2-negative(c) of both Tests, Test1- negative and test2- positive(b) for both tests) for calculation of Chi-square value.

Formulae= (b-c)\*\*2/b+c= Chi-square

1. First test: performed on 2.1 million instances.

Baseline: Model created using only nouns.

Experiment: Model created using nouns, using Authors, dates and Over\_18 fields to filter.

Null Hypothesis: Authors, dates and Over\_18 have no influence on relation of a post to subreddit

Results: Obtained Chi-square value of about 624 which strongly suggests the Null hypothesis must be rejected. 624>>>0.2 (Took p-value as 0.2 guessing square in the numerator plays huge role)

1. Second test:

Baseline: Model created using only nouns.

Experiment: Model created using nouns and verbs.

Null hypothesis: Verbs are not unique to subreddits.

Results: Obtained 0.043 as result (Took p-value as 0.2) which suggests we cannot reject Null hypothesis.

**Results:** (Results obtained by running on 4.2 millions instances)

There are couple of experiments done in the process to obtain best results.

Experiment 1:(Baseline)

Model: Only nouns

Prediction: Parameters sent to prediction function are Model with only nouns. None of the filters are added.

Results:

In fold 1

-----------------------------------------------------------------------------------------------

Model created

Accuracy in Testing 32.425176

Fscore in Testing: 0.258401

In fold 2

-----------------------------------------------------------------------------------------------

Model created

Accuracy in Testing 32.404194

Fscore in Testing: 0.257731

Overall stats: ----------------------------------------------------

Total avg of accuracy in only Titles 32.414686

Total avg Fscore in only titles : 0.258066

Tabular representation:

|  |  |  |
| --- | --- | --- |
| Fold | Accuracy | Fscore |
| 1 | 32.425176 | 0.258401 |
| 2 | 32.404194 | 0.257731 |

Understanding results: Nouns alone can predict subreddits of post but not effectively enough. Experiments done on smaller files gave 55-60% accuracy on each fold. As we can see, there is drastic drop in accuracy. So came to conclusion that the model needs more features.

Experiment 2:

Model: Nouns , Dates, Authors ,over\_18 features are used. More features are added since we reached to conclusion that features are not enough from experiment 1.

Prediction: Now the parameters Dictofmin(dates),Over\_18(subreddits whose over\_18 has ”true”), Authors dictionary(dictionary of sets)

Results:

In fold1

----------------------------------------------------------------------------------------------

Model created

Accuracy in Testing 43.842079

Confidence in Testing: 0.394988

In fold 2

-----------------------------------------------------------------------------------------------

Model created

Accuracy in Testing 43.871230

Confidence in Testing: 0.395143

In fold 3

----------------------------------------------------------------------------------------------(Manually stopped. after 17 hours of execution)

Tabular representation:

|  |  |  |
| --- | --- | --- |
| Fold | Accuracy | Fscore |
| 1 | 43.842079 | 0.394988 |
| 2 | 43.871230 | 0.395143 |

Experiment 3: Ran on 200,000 instances

Model: Authors, Nouns, Over\_18

In fold 1

-----------------------------------------------------------------------------------------------

Model created

Accuracy in Testing 73.757455

Confidence in Testing: 0.713221

In fold 2

-----------------------------------------------------------------------------------------------

Model created

Accuracy in Testing 73.564130

Confidence in Testing: 0.712058

In fold 3

-----------------------------------------------------------------------------------------------

Model created

Accuracy in Testing 73.917142

Confidence in Testing: 0.716130

In fold 4

-----------------------------------------------------------------------------------------------

Model created

Accuracy in Testing 73.352520

Confidence in Testing: 0.710114

Overall stats: ----------------------------------------------------

Total avg of accuracy in only Titles 73.647812

Total avg confidence in only titles: 0.712880

**Learning from Results**: Studying the results by running on different size of datasets. The decrease in the accuracy and Fscore as size of dataset decreases .

Even though the file size increased from 50 mb to 1gb, the accuracy fell relatively less compared to earlier change in size(7 to 50). There will be a point where the accuracy curve becomes horizontal as we keep on increasing size.

**Statistical tests(500 mb)(Memory error for 1 gb):**

1. First test: performed on 2.1 million instances.

Baseline: Model created using only nouns.

Experiment: Model created using nouns, using Authors, dates and Over\_18 fields to filter.

Null Hypothesis: Authors, dates and Over\_18 have no influence on relation of a post to subreddit

Results: Obtained Chi-square value of about 624 which strongly suggests the Null hypothesis must be rejected. 624>>>0.2 (Took p-value as 0.2 guessing square in the numerator plays huge role)

Here, In prediction we predict subreddit to a post twice.(once on baseline model and once on Experiment) This is the cause of memory error while on 4.2 million instances.

**Future findings:**

More features from Comments file can be added. Hopefully, I get a High performance computing resource to run my program on so that I can have more confidence in the results by running on huge files.

**Conclusion:**

This will help new users to post their content in right subreddit so that it won’t be deleted by the owner of that subreddit. The main approach to classify post is saving nouns of each subreddit since each subreddit has names repeating and these nouns are unique to a subreddit.

Results obtained give the degree of how much Names in posts of different subreddits differ. It is very unlikely that prediction of a post is very far from the real one i.e. a prediction is “funny” but the real one is “humor”. Even though this is case, Prediction is still reliable in the sense of recommending subreddit to a new user (humor and funny are very similar, it wouldn’t matter that much where the post is posted).

Future scope of project:

We can add more features from “comments“ file when we are unsure about predictions made using the current features. If we do not have enough nouns from “titles” post, we can obtain some nouns from the comments to that post. For example, if title is “who wins?” and comments to that post has “obama and Trump” then we can use these to compare with model which probably gives “politics”.

**Acknowledgements:**

Name: Nishitha Guntakandla

Help: Obtained help in various ways such as feedback on design. Helped running the program on talon (No luck though in this aspect)

Name: Prof. Nielsen Rodney

Help: Suggested to add more features such as authors. Professor helped me choosing statistical test for project.

**References:**

Prior work: [chrome-extension://oemmndcbldboiebfnladdacbdfmadadm/https://cseweb.ucsd.edu/~jmcauley/cse255/reports/fa15/021.pdf](mailto:None)

Statistical test:

<https://en.wikipedia.org/wiki/McNemar's_test>

NLTK package:

<http://www.nltk.org/>

Python programming: [www.python.org](http://www.python.org)