Zhang, Jinhan PID A53211930
University of California, San Diego

February 28, 2017

ASSIGNMENT 1 REPORT

Problem 1. Helpfulness Prediction

In this prediction task, I firstly choose to fit $\frac{nHelpful}{outOf}$. Then I start to pick up features and build our models.

Firstly, I find out that "outOf" is a very useful feature. But if I just add a simple linear or square feature about "outOf", I do not end up with that much improvement on my validation dataset. So I just plot the histogram for all "outOf" values and find its imbalanced distribution. So I choose to add a "plank" function to "outOf" feature. That is implemented by the code in Listing 1 below:

Listing 1: plank function for "outOf" feature

```
def outOf_plank(num):
    if num<1:
        return 1.0
    elif num<2:
        return 2.0
    elif num<4:
        return 3.0
    elif num<9:
        return 4.0
    elif num<17:
        return 5.0
    elif num<25:
        return 6.0
    elif num<50:
        return 7.0
    else:
        return 8.0
```

Secondly, "rating" field is another useful feature. So I try to add linear "rating" feature to the model, and this works well. Then I furtherly try to add a "0/1" feature about "whether the rating is below average", and this improve the results on the validation set to some degree.

Thirdly, I try to analyze the content in "reviewText" field. Since uppercase characters and punctuations may be useful, I just calculate the percentage of uppercase characters and punctuations in "reviewText" and add them to the model. Furthemore, I also calculate the percentage of "?" and "!" since they are usually representitive. The code stub for this part is shown as Listing 2 below.

Listing 2: analysis function for "reviewText" feature

```
def parse_string(str):
    cnt_ques , cnt_up , cnt_punc , cnt_char  = 0 , 0 , 0 , 0
    for c in str:
```

```
        if c in string.uppercase:
            cnt_up+= 1
        if c in string.punctuation:
            cnt_punc += 1
        if c in ['?','!']:
            cnt_ques += 1
        cnt_char += 1
    if cnt_char == 0:
        return 0.0, 0.0, 0.0, 0.0, 0.0
    return cnt_ques * 1.0 , cnt_up * 1.0, cnt_ques * 1.0 , \
           len(str.strip().split()), cnt_char
```

Then I try to analyze the information in "categories" field. I analyze the strings in "categories" to find if they are for "women", "men", "boys", "girls" or "babies", and use five "0/1" feature to represent them respectively. That analysis process is like code list in Listing 3 below.

Listing 3: analysis function for "categories" feature

```
def analyze_cate(datum):
    iswomen ,ismen ,isboys ,isgirls,isbabies =  0,0,0,0,0
    for item in datum:
        if 'Women' in item:
            iswomen = 1
        if 'Men' in item:
            ismen = 1
        if 'Boys' in item:
            isboys = 1
        if 'Girls' in item:
            isgirls = 1
        if 'Babies' in item:
            isbabies =1
    return iswomen ,ismen
```

Then the final chosen model is like:

$$\begin{aligned}\frac{nHelpful}{outOf} \simeq \alpha &+ \beta_1(plank([\text{"}outOf\text{"}])) + \beta_2(uppercase \quad rate \quad in \quad review) \\ &+ \beta_3(punctuation \quad rate \quad in \quad review) + \beta_4(?\&! \quad rate \quad in \quad review) \\ &+ \beta_5(rating < average) + \beta_6(\text{"}Women\text{"} \quad in \quad categories) \\ &+ \beta_7(\text{"}Men\text{"} \quad in \quad categories) + \beta_8(\text{"}Boys\text{"} \quad in \quad categories) \\ &+ \beta_9(\text{"}Girls\text{"} \quad in \quad categories) + \beta_{10}(\text{"}Babies\text{"} \quad in \quad categories)\end{aligned}$$

After we use model to predict the "helpfulness rate", we just multiply it with "outOf". And also since we are judged by MAE, I furtherly round the prediction result.

The result on validation set (i.e. the 100001^{st} to the 200000^{th} data) in MAE is about 0.175, while on the public portion of test data is about 0.162. Unfortunately, it is ovefitted and get MAE on the unseen portion of test data to be 0.185. I've seen also in the final "mysubmissions" that if I drop the last 3 features, I get 0.16514 MAE on public data while 0.17586 on private data. So it's a sad story. :(

Problem 2. Rating Prediction

In the rating prediction task, I firstly choose to use latent-factor model like:

$$rating(user, item) = \alpha + \beta_u + \beta_i + \gamma_u \cdot \gamma_i$$

And I try to set the length of vector for every user/item in γ_u and γ_i to be 3,5,10 and 15. But all these trails seem to improve only a little and runs for a long time in my implementation. So I choose not to use γ_u and γ_i in my final model.

Then I try to add some features about user/item to the latent-factor model to deal with new users/items problem, this improves the MSE result to some degree but seems no better than the final one.

Finally I try to use the simple latent-factor model like:

$$rating(user, item) = \alpha + \beta_u + \beta_i$$

and adjust the regularization term λ to 5.6, then I end up with a pretty good MSE on validation set. And then after I get the parameter, I use 200000 data to be trainging set and use the result to be my final model. I get MSE to be 1.13562 on the seen portion of test set and 1.08731 on the unseen portion of test set. The big difference is probably because the two portions may not be randomly assigned or the test set is not big enough.

Problem 3. Kaggle Information

My Kaggle username: YaphetS

Email used for your kaggle account: jiz530@eng.ucsd.edu

Submitted by Zhang, Jinhan PID A53211930 on February 28, 2017.