Ranking and Prediction of Amazon Fine Food based on Costumer's rating and review

Chan Hei Nam,
Cheng Tin Chu,
Lei Wen Feng,
Ng Wing Hin

Introduction

User's rating and review is one of the explicit ways to determine product's popularity. In this project, we are trying to process the Amazon Fine Food Reviews such that the result can be used to predict the rating based on review.

Basic statistics of the dataset

Basic statistics of the dataset

	Rating	Word(Processed)
Mean	4.18	255
Minimum	1	7
Maximum	5	14425

System Structure

There are mainly three stages in our system :

Stage 1 Pre-processing the dataset. Also, splitting 70% of the dataset as training set and 30% as testing set.

Stage 2 Using MapReduce, find the first hundred k-shingles, $1 \leq k \leq 5$, of each rating, $1 \leq \text{rating} \leq 5$.

Stage 3 Do prediction of rating based on the review input to the system.

Stage 1 Pre-processing

Algorithm 1: Pre-processing

while *There exists next row inside Reviews.csv* **do**
    Extract **Rating** and **Text**;
    Lower **Text** and remove *some* stopwords and punctuation;
    Returning line with format $Rating, word_1, word_2, ...$;
end

Example result from Stage 1

Original data :

Id, ProductId, UserId, ProfileName, HelpfulnessNumerator, HelpfulnessDenominator, Score, Time, Summary, Text
1,B001E4KFG0,A3SGXH7AUHU8GW,delmartian,1,1,5,1303862400,Good Quality Dog Food,I have bought...
2,B00813GRG4,A1D87F6ZCVE5NK,dll pa,0,0,1,1346976000,Not as Advertised,"Product arrived labeled...
3,B000LQOCH0,ABXLMWJIXXAIN,"Natalia Corres ""Natalia Corres""",1,1,4,1219017600,""Delight"" says it all","This is a confection...
4,B000UA0QIQ,A395BORC6FGVXV,Karl,3,3,2,1307923200,Cough Medicine,If you are looking...

Result data :

Score, Text
5,bought vitality canned dog food products good quality product looks...
1,product arrived labeled jumbo salted peanutsthe peanuts actually...
4,confection centuriesit light pillowy citrus gelatin nutsin case filberts...
2,looking secret ingredient robitussin believe iti got addition root...

Stage 2 MapReduce

Algorithm 2: MapReduce (Mapper)

while *There exists next row inside Preprocessed.csv* **do**
  for $k \leftarrow 1$ **to** 5 **do**
    Find all k-shingles;
    for *Each shingle found* **do**
      Return $<<$ rating, k, shingle $>, 1 >$;
    end
  end
end

Stage 2 MapReduce

Algorithm 2: MapReduce (Reducer)

forall *tuples with key* $<$ *rating, k, shingle* $>$ **do**
    frequency = sum of all tuple values;
    Return $<$ rating, k, frequency, shingle $>$;
end

Exmaple result from Stage 2

Result from MapReduce :

Score	k	frequency	shingle
3	4	23	food freshly openedpi likes
5	2	1712	very nice
2	1	1601	say
3	4	27	coffeetea love organic coffee

Perform sorting and filtering to output top N (N = 100, 300, etc.) frequent shingles for each score and each k.

Stage 3 Prediction

Algorithm 3: Prediction using length-k shingles

Load the *records* from **Shingle Database**;
Find all *k-shingles* in the input text;
foreach *shingle* **do**
    Collect all records of *shingle* from database;
    Append to *RecordsFound*;
end
if *RecordsFound is not empty* **then**
    score = weighted average of *RecordsFound*;
else
    score = average of all records in database;
end
return *score*;

Stage 3 Prediction

Prediction Example

Database:

Score	k	frequency	shingle
5	1	68	good
4	1	35	good
5	1	57	really

Input: *this is really good*

Score:

$$\frac{5\times68+4\times35+5\times57}{68+35+57} = 4.78125$$

Stage 3 Prediction

Using shingles of different lengths, we got different performances:

Shingle Length	Mean Squared Error
1	1.596
2	2.160
3	2.918
4	3.089
5	3.112

But we need to respect shingles of different length. How?

Regression Model

Regression model

Regression equation :

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_5 x_5 \tag{1}$$

where x_i is the i-th shingle score, for $i = 1, ..., 5$

Regression Model

Regression model

System of linear equation :

$$\begin{aligned} y_1 &= \beta_0 + \beta_1 x_{11} + \cdots + \beta_5 x_{15} \\ y_2 &= \beta_0 + \beta_1 x_{21} + \cdots + \beta_5 x_{25} \\ &\vdots \qquad\qquad\qquad \vdots \\ y_n &= \beta_0 + \beta_1 x_{n1} + \cdots + \beta_5 x_{n5} \end{aligned} \tag{2}$$

In matrix form :

$$\mathbf{Y} = \boldsymbol{\beta_0} + \mathbf{X} \cdot \boldsymbol{\beta}$$

where
$\mathbf{Y}$ is a $n \times 1$ matrix,
$\boldsymbol{\beta_0}$ is a $n \times 1$ matrix,
$\boldsymbol{\beta}$ is a 5×1 matrix,
$\mathbf{X}$ is a $n \times 5$ matrix.

Regression Model

Complexity

$$\hat{\beta} = (\mathbf{X}^{\mathsf{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathsf{T}}\mathbf{Y}$$

key computation step:

- $\mathbf{X}^T\mathbf{X}$ in $O(n \times k^2)$
- $(\mathbf{X}^T\mathbf{X})^{-1}$ in $O(k^3)$

n = number of rows from train data

k = number of regression parameters = 5 + 1

As $n >> k$, the overall complexity is $O(n)$.

Result

Shingle Match % Performance

Shingle Length	Top_100	Top_300
1	99.87%	99.97%
2	67.88%	81.60%
3	16.47%	25.83%
4	2.19%	2.89%
5	0.30%	0.05%

Result(Cont'd)

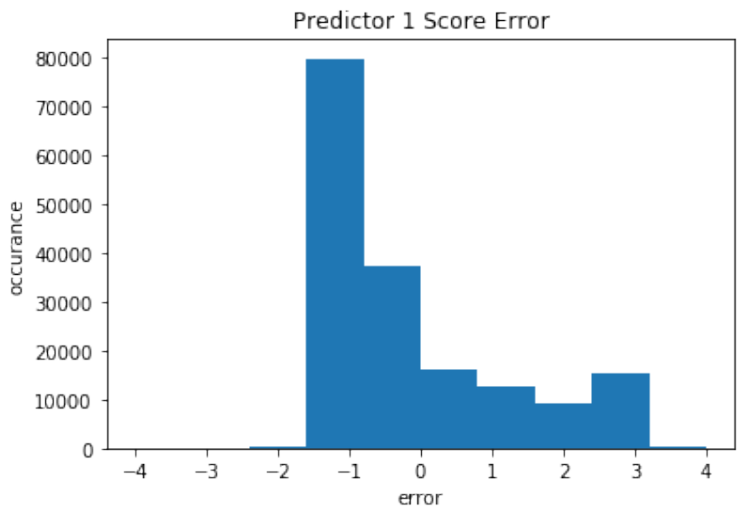
Performance - MSE comparison - Top_100

Prediction Method	Default Score	
	3	Train Mean = 4.18
Train Data Mean	1.717	1.717
1-shingle	1.596	1.596
2-shingle	2.160	1.595
3-shingle	2.918	1.667
4-shingle	3.089	1.704
5-shingle	3.112	1.710
Regression	1.449	1.369

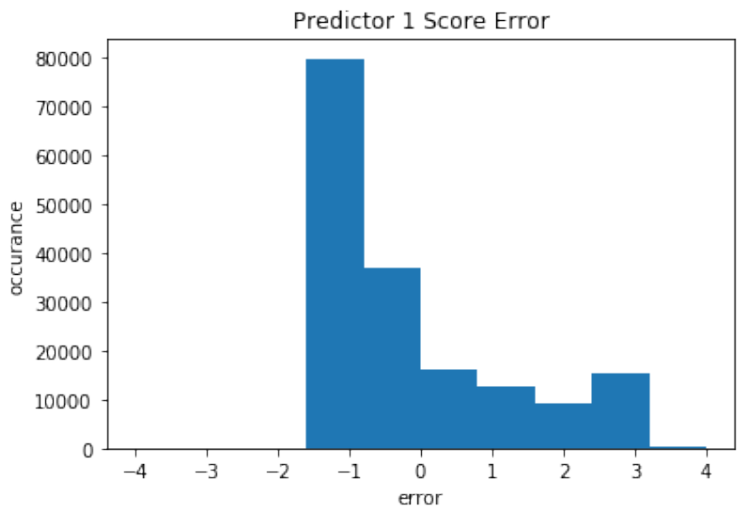
Result(Cont'd)

Error Spread of 1-shingle

(a) Default Score = 3



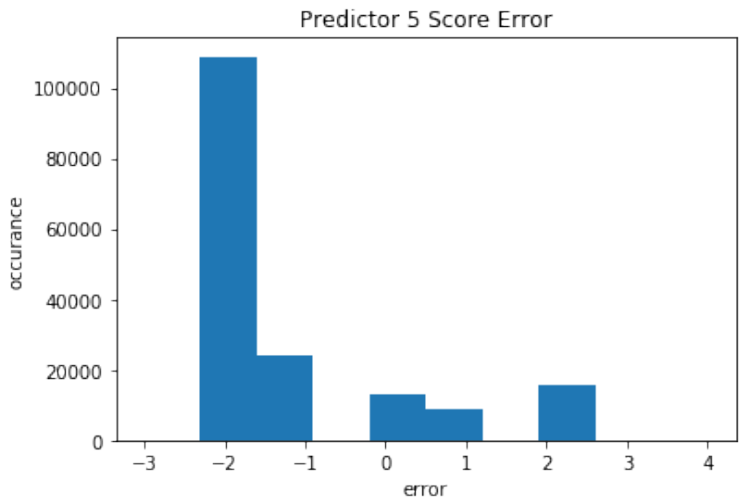
(b) Default Score = 4.18



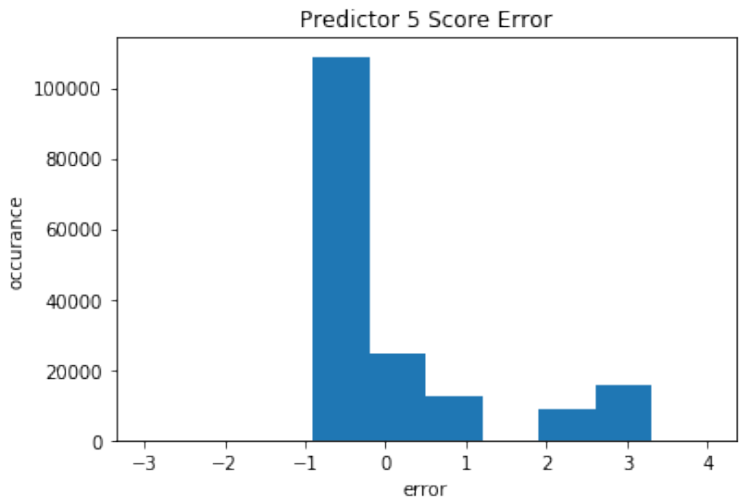
Result(Cont'd)

Error Spread of 5-shingle

(a) Default Score = 3



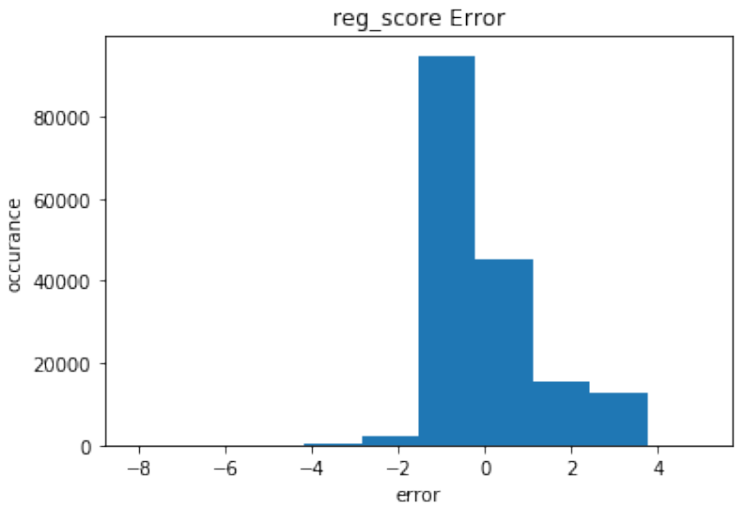
(b) Default Score = 4.18



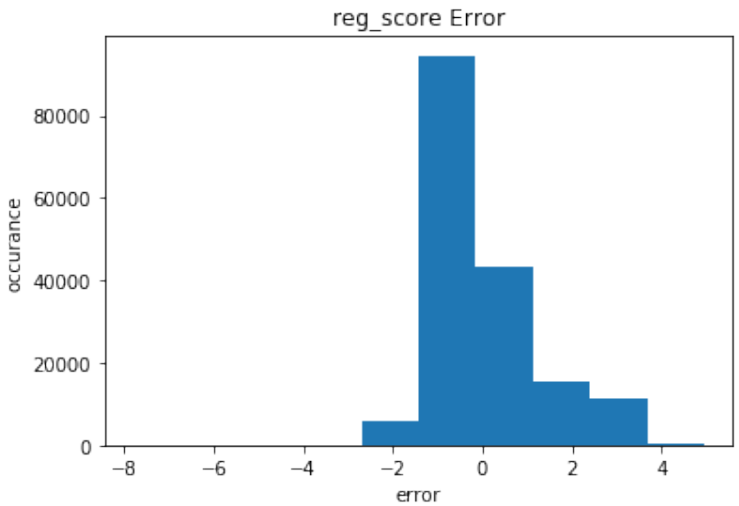
Result(Cont'd)

Error Spread of Regression Method

(a) Default Score = 3



(b) Default Score = 4.18



Result(Cont'd)

Performance - MSE comparison

Prediction Method	Default Score = Train Mean	
	Top_100	Top_300
Train Data Mean	1.717	1.717
1-shingle	1.596	1.592
2-shingle	1.595	1.531
3-shingle	1.667	1.626
4-shingle	1.704	1.701
5-shingle	1.710	1.709
Regression	1.369	1.273

Demostration

```
predict> this is good
Score = 4.287700747576452

predict> this is really good
Score = 4.5994073238152176

predict> this is quite good
Score = 4.474748081883183

predict> this is not that good
Score = 3.336314640348344

predict> this is bad
Score = 1.546573898931245

predict>
```

Video Demo: `https://goo.gl/fqaTSo`

Conclusion

- Use Train Data Mean as default score.
- The higher the match %, the more accurate prediction (for k>=2)
- 1-/2-shingle method is better than other k-shingle methods.
- Regression method is the best method.