Problem Chosen	2020 MCM/ICM Summary Sheet	Team Control Number
C		2014775

Use easy-to-interpretate models to find interesting pattern in e-commerce data

Summary

Firstly, we extract some important features in review include: flesch reading ease readability, number of sentences, number of positive and negative words, And use algorithm proposed by previous work of some experts to classify article sentiment to positive and negetive classes.

Then we use crosstable and chi square test to find and validate intuitive relationship between star rating,review type and helpful rate.

In order to gain more details about how these features of review influence helpful rate and star rating, we use two easy-to-interpretate machine learning models: tree classifier and regularized logistic regression which take review features as predictors. We fit each model to three kinds of product seperately to find out if these features have different influences on different products and get interesting results. Based on fitting results, we made some easy-to-track data measures based on ratings and review features to help company filter important reviews.

To inform whether the product is successful or not, rather than using subjective criterias which are proposed by many works before we use PCA as an unsupervised learning method to automatically select the features which indicates wheter the company will be successful.

To see some time-based pattern, we implement ARMA model to different products range from best-sold product to some poor-reputation product. Rather than fit the model completely we are more interested in the auto correlation analysis to determine the time period during which reviews have the most impact on a business's reputation.

As a supplement to the previous technique of filtering out effective reviews, we use chi-square test to automatically filter words to find descriptive words in reviews that have the most impact on star ratings which could help company to identify potentially important design features that would enhance product desirability.

Keywords: reviews filter; tree classifier; Logistic regression; unsupervised learning; PCA; ARMA; χ^2 test

Use easy-to-interpretate models to find interesting pattern in e-commerce data

March 10, 2020

Summary

Firstly, we extract some important features in review include: flesch reading ease readability, number of sentences, number of positive and negative words, And use algorithm proposed by previous work of some experts to classify article sentiment to positive and negetive classes.

Then we use crosstable and chi square test to find and validate intuitive relationship between star rating,review type and helpful rate.

In order to gain more details about how these features of review influence helpful rate and star rating, we use two easy-to-interpretate machine learning models: tree classifier and regularized logistic regression which take review features as predictors. We fit each model to three kinds of product seperately to find out if these features have different influences on different products and get interesting results. Based on fitting results, we made some easy-to-track data measures based on ratings and review features to help company filter important reviews.

To inform whether the product is successful or not, rather than using subjective criterias which are proposed by many works before we use PCA as an unsupervised learning method to automatically select the features which indicates wheter the company will be successful.

To see some time-based pattern, we implement ARMA model to different products range from best-sold product to some poor-reputation product. Rather than fit the model completely we are more interested in the auto correlation analysis to determine the time period during which reviews have the most impact on a business's reputation.

As a supplement to the previous technique of filtering out effective reviews, we use chi-square test to automatically filter words to find descriptive words in reviews that have the most impact on star ratings which could help company to identify potentially important design features that would enhance product desirability.

Keywords: reviews filter; tree classifier; Logistic regression; unsupervised learning; PCA; ARMA; χ^2 test

Contents

1 Introduction

1.1 Background

Amazon, one of the largest online retailers, seeks to provide customers with the most helpful reviews for a product and has provided a great platform for review sharing. Undoubtedly, It is important for our service target-Sunshine Company to identify the key patterns, relationships, measures, and parameters in past customer-supplied ratings and reviews associated with other competing products before the company introduce new products into the market.

1.2 Problem Restatement

Oue team are required to solve two big questions and then write a letter to the Marketing Director of Sunshine Company summarizing our teams analysis and results. The questions are shown as follows as far as we know about:

- Find quantitative relationships between ratings, reviews and helpfulness ratings
- Find measures which is the easiest one for the company to track after the products are put into the market, focusing on ratings and reviews.
- Find measures which reflects the reputation of a product, focusing on time
- Find measures which best indicates the potential of a product, focusing on texts and ratings
- To verify whether it is easier for customers to make some kinds of reviews after seeing a series of low or high ratings
- To verify whether some sentimental words are related to rating levels

1.3 Overview of our work

- Do exploratory data analysis using crosstable and chi-square test to find and validate intuitive relationship between ratings, reviews and helpfulness ratings
- Use tree model and regularized logistic model to find quantitative relationships b etween ratings, reviews and helpfulness ratings which can be easily interpretated.
- Implement PCA boldly on a combination of star rating ,features of reviews and features of sales volume and appoint the first component as indicator of success and interpret PCA results in a fresh way
- Implement Autocorrelation analysis to find different time-based patterns of different products and interpret them.Use ARMA model on a single reputation series and get desirable result
- Implement regularized logistic model on star rating series to predict possiblity of next review is possitive and find interesting result
- Use chi-square test to find if some descriptive words has a great impact on star rating and develope a word filter base on it.

1.4 Data source

We are provided with three data files:hairdryer.tsv, microwave.tsv, and pacifier.tsv. All the important data and data label definitions are provided in the files, which is the only data we should use for our problems.

2 Notations

Symbol	Definition
N_{pos}	number of positive words in a review
N_{neu}	number of neutral words in a review
N_{neg}	number of negative words in a review
N_{wn}	number of words in a review
R_t	product reputaion score related to time t
CV	number of groups for cross-validation
$trans\_helpful$	determined by the value of helpful rate
$trans\_rating$	determined by the value of rating

3 Model

3.1 Data preprocessing

3.1.1 Features extraction from reviews

According to the data, we can tell there is no strong connection between ratings and helpfulness because some of the high star rating do not get even one helpful votes while the others get hundreds of them. Also, we can not tell the interior rules between the low star rating and helpfulnuss, which means it is hard to find positive or negative correlations, so the research into it is definitely meaningless. Therefore, we simplify the question and just consider two groups of relationships: Helpfulness-(Reviews&Ratings) and Ratings-Reveiws. The first group aims to find how helpfulness is affected by reviews and ratings and he second group aims to find how ratings is affected by reviews. We reomove the impact brought by ratings for the second one since the ratings are created before the helpfulness is created. So the helpfulness won't function on the ratings.

To help find the methodology of this problem, we need to quantify ratings, reviews and helpfulness, which means we need to find features for each of them that have high representitaitveness. We will show the way in the following parts.[4,5,6]

- **sentiment analysis**
  Sentiment is something which exists in the words naturally and it is obvious that the positiveness or negativeness of a review would decide the ratings to some degree and would influence other reviewers, which would influence the helpfulness indirectly.

- **readability**
  Readability determines whether it is easy for other reviews to understand the words. Others would think a review as not helpful if it is too difficult to understand it.

- **reviewers**
  Reviewers are divided into vine group and not-vine group. People may hold predudice for or againest reviews from vines which influence the helpfulness. Besides, the crowd of vines shows a trend in ratings.
- **length** Actually, the length of reviews is somewhat related to the readability. Apart from that, it is also related to the credibility of words. High credibility reviews seem to be more helpful.

Then, combining with the data provided, we picked several measures for four features.

- For sentiment, we choose$\{N_{pos}, N_{neg}, N_{neu}\}$, N_{pos} is the number of positive words in a review, N_{neg} is the number of negative words in a review, N_{neu} is the number of neutral words in a review.
- For readability, we choose $FKRE$ index, which is specially created to calculate readability.
- For reveiwers, we choose $vine$ and $verified\ purchase$.
- For length, we choose N_{wn}, which is the number of words in a review.

3.2 Exploratory data analysis(1)

To explore the relationship between helpful rate and review orientation and ratings We make a contingency table between two pairs, and use the chi-square test to determine whether the two are related. Three tests have shown that we should reject the null hypothesis, that is, these three will affect each other. In order to better understand the relationship between helpful rate and scoring, we classify both and then make the following contingency table. It's easy to see that low star reviews have a greater share of useful reviews.[Figure 1,2,3,4]

star_rating	1	2	3	4	5
rela_pos					
0	1634	819	882	676	1625
1	992	877	1677	4436	18406

(a) Crosstable

```
(6520.180419435507,
 0.0,
 4,
 array([[  462.15763178,   298.4841369 ,   450.36610042,   899.67624282,
          3525.31588808],
        [ 2163.84236822,  1397.5158631 ,  2108.63389958,  4212.32375718,
         16505.68411192]]))
```

(b) χ^2 test

Figure 1: Relationship between Star rating and review sentiment

star_rating	1	2	3	4	5
trans_helpful					
0	1542	1151	1912	3942	16260
1	1084	545	647	1170	3771

(a) Crosstable

```
(787.3435160197104,
 4.2341268545782586-169,
 4,
 array([[ 2034.19878841,  1313.78566075,  1982.29805771,  3959.94828878,
         15516.769204351,
        [  591.80121159,   382.21433925,   576.70194229,  1152.05171122,
          4514.23079565]]))
```

(b) χ^2 test

Figure 2: Relationship between Star rating and helpful rate

trans_helpful	0	1
rela_pos		
0	4159	1477
1	20648	5740

(a) Crosstable

```
(52.524625532975094,
 4.248871058382662e-13,
 1,
 array([[ 4365.85848114,  1270.14151886],
        [20441.14151886,  5946.85848114]]))
```

(b) χ^2 test

Figure 3: Relationship between review sentiment and helpful rate

trans_rating	0	1
trans_helpful		
0	0.338211	0.661789
1	0.457669	0.542331

Figure 4: Relationship between ratings and helpful rate

3.3 Regularized logistic Regression

3.3.1 Regularized Logistic Regression

Logistic regression, despite its name, is a linear model for classification rather than regression and regularization is applied by default, which is common in machine learning. It minimizes the following cost function:

$$\min_{\omega,c} \frac{1}{2}\omega^T\omega + C\Sigma_{i=1}^{n} log(exp(-y_i(X_i^T\omega + c)) + 1) \tag{1}$$

where $X = (X_1, X_2, ..., X_n)$ is provided data, ω and c are changing.[1]

3.3.2 Cross-validation based on AUC-ROC score

- **cross-validation** The basic idea of cross-validation is to group the original data in a certain way, one part as the training set and the other part as the validation set or test set. First, use the training set to train the classifer. Then, the validation set is used to test the trained classifier, the result of which is used as an evaluation index of the classifier.

- **AUC-ROC score** The Receiver Operating Characteristic (ROC) curve and Area Under Curve (AUC) are often used to evaluate the pros and cons of a binary classifier. Compared with evaluation indicators such as accuracy, recall, and F-score, the ROC curve has a very good characteristic: when the distribution of positive and negative samples in the test set changes, the ROC curve can remain unchanged. Class imbalance often occurs in actual data sets, that is, negative samples are much more than positive samples(or vice versa),

and the distribution of positive and negative samples in test data may also change over time.[2]

In our model, we adopt cross validaiton and will divide our data into two parts, one as train set and another one as test set. Besides, due to the great features of AUC-ROC score, we use it to grade our regression.

3.3.3 Helpfulness and review features

The measures are picked like following:[N_{pos},N_{neu},N_{neg},$FKRE$,N_{wn},$vine$,$verified\,purchase$,$rating$] From the coefficients of all the measures, [Figure 5]we can tell the relation between helpfulness and review features. And in the rest of paper, NR means negative related, PR means positive related, just for brief.

Here is the result for hair dryer:

- N_{pos}–NR,N_{neu}–PR,N_{neg}–PR
  This phenomenon indicates people prefer neutral and negative reviews, which they reagrd as more helpful.
- $FKRE$–PR
  It means more readable a review is, more helpful it will be. To explain this, we know that the higher $FKRE$ is, the harder it is for reviewer to understand because maybe there are lots of difficult terms.
- N_{wn}–PR
  It means people like to read longer reveiws which include more details.
- $vine$–NR
  It means consumers do not put enough trust in vines. Maybe they think the vines jump to the conclusions without long-term experience and that could be untrustable.
- $verified\,purchase$–NR
  This is explicit because consumers want to know about the part of reviews coming from unverified purchase which may show the problems lying in the product.
- $ratings$–NR
  Just like $verified\,purchase$, maybe consumers believe low star ratings are more objective and reliable.

When we moving to the results of microwave and pacifier, what surprises us is their high similarity to hair dryer. As far as the positive and negative correlation is concerned, the three products have a totally same result.

As a conclusion for helpfulness, consumers prefer negative detailed reviews from unverified purchase.

3.3.4 Ratings and review features

The measures are picked like following:[N_{pos},N_{neu},N_{neg},$FKRE$,N_{wn},$vine$,$verified\,purchase$] Here is the result for hair dryer:[Figure 6]

```
array([[-0.18037431,  0.1742199 ,  0.09097124, -0.07722475,  0.5147693 ,
        -0.1353087 , -0.1732267 , -0.00504605]])
```

(a) hair_helpful_logstic

```
array([[-0.10646686,  0.09849678,  0.04826202, -0.15224923,  0.25702749,
        -0.00032852, -0.12791423, -0.04909023]])
```

(b) mirco_helpful_logstic

```
array([[-0.18449131,  0.28166061, -0.00245761, -0.16715908,  0.56442919,
        -0.04545558, -0.21483115,  0.09577647]])
```

(c) paci_helpful_logstic

Figure 5: Coefficient of logistic regression(helpful)

- N_{pos}–RR,N_{neu}–NR,N_{neg}–NR
  This is quite trivial because a person will never give a low-star rating after showing his love for a product.
- $FKRE$–NR
  Comments with high $FKRE$ index always come from some specialized reviewers who have stricter standards for products, which may lead to low-star rating.
- N_{wn}–NR
  This shows people who comment a lot about a product are more likely to be complaining.
- $vine$–PR
  This shows $vine$s of hair dryer tend to give a higher star than the average level.
- $verified\ purchase$–PR
  This shows consumers who verify the purchase always give a high rating.

Microwave is different from hair dryer in N_{wn}:

- N_{wn}–PR This shows for microwave, consumer comment more may give a higher rate.

However, pacifier shows more differences:

- N_{pos}–RR,N_{neu}–PR,N_{neg}–NR
  What is different is the neutral reviewer tend to give a higher rating.
- $FKRE$–PR
  Comments from specialized reviewers contrarily give a higher rating.
- N_{wn}–PR
  This shows people who comment a lot about a product are more likely to be prasing.
- $vine$–NR
  This shows $vine$s of hair dryer tend to give a lower star than the average level.

As a conclusion for rating, different products have different results and we can see from several aspects: N_{neu}, $FKRE$, N_{wn} and $vine$.

```
array([[ 0.35385174, -0.2205398 , -0.41487926, -0.01186181, -0.0269585 ,
         0.03079806,  0.06640762]])
```

(a) hair_rating_logstic

```
array([[ 0.92485192,  0.0421784 , -0.49558498,  0.07229581,  0.03011448,
        -0.03327071,  0.00839038]])
```

(b) paci_rating_logstic

```
array([[ 1.15467598, -0.87848541, -0.91536825, -0.04394478,  0.02832352,
         0.28929317,  0.85042478]])
```

(c) micro_rating_logstic

Figure 6: Coefficient of logistic regression(rating)

3.3.5 Relation between previous star ratings sequence and following review type(2d)

Now we consider some more interesting problems with regularized logistic regression: are customers more likely to write some type of review after seeing a series of low star ratings.

Although this is not very helpful in developing market strategies, it is helpful for marketing directors to understand some comment behavior.

For a comment, we set parameter X_i. If the previous $i-th$ personal rating star rating is lower than the average star rating of the product, $X_i = 0$, otherwise $X_i = 1$. Another parameter d is the difference between the previous five star averages and the total star average.

We use these six variables as predictors to predict whether this review will be positive or negative.

$CV = 5$ is also used to determine the regularization parameters. The results are as follows[Figure 7,8,9]. The best score is: Then we perform regression on the entire data set. We get six coefficients as follows: This shows that each of the five previous negative reviews will increase the number of reviewers. Rate of negative reviews, but if the average of the last five stars is lower than the average star rating, the reviewer's praise rate will increase

param_C	params	split0_test_score	split1_test_score	split2_test_score	split3_test_score	split4_test_score	mean_test_score	std_test_score	rank_test_score
0.01	{'C': 0.01}	0.536845	0.601995	0.679564	0.661624	0.972848	0.690556	0.149735	7
0.1	{'C': 0.1}	0.555786	0.623949	0.723355	0.664277	0.966225	0.706700	0.140743	6
1	{'C': 1}	0.562160	0.627864	0.737947	0.665896	0.966225	0.712000	0.139248	5
10	{'C': 10}	0.562925	0.628107	0.739401	0.666220	0.966225	0.712558	0.139088	1
100	{'C': 100}	0.562929	0.628130	0.739503	0.666185	0.964901	0.712312	0.138607	4
1000	{'C': 1000}	0.562939	0.628125	0.739515	0.666169	0.964901	0.712312	0.138607	3
10000	{'C': 10000}	0.562932	0.628129	0.739522	0.666172	0.964901	0.712313	0.138608	2

Figure 7: choose $C = 10$ base on validation result

```
array([[ 0.88260493,  0.93565244,  0.26066389,  0.81599526,  0.90498935,
        -4.1220591 ]])
```

Figure 8: logistic fitted coeffcient

```
array([-0.70272061])
```

Figure 9: logistic fitted intercept

3.4 Classifier: Tree

3.4.1 Why do we choose tree?

Tree is a easy-interpretate model and similar to how reviewers make decisions.In this manner,we can use tree to understand how reviewers judge wether some review are helpful for him.Furthermore,market derector can use some similar motif in tree model to filter out useless review

3.4.2 Cross-validation based on AUC-ROC score

Such method is suitable for all classifiers and we will continue to use it for our tree model.

3.4.3 Helpfulness and review features

Looking closely at the trees of three products[Figure 10,11,12], we can find the following common points from them:

- The longer the comment is, the greater the possible help for the product This is a big feature that is clearly displayed in the trees of three products. We can clearly see from it that the short comments after classification are basically not helpful, and the helpful attributes of the tree of each product are classified into helpful categories. Focus on long comments.
- Not positive reviews are helpful for your product Except that in the tree of the microwave oven, we could not find the impact of the enthusiasm of the review on the product. In the trees of the remaining two products, we can see that the positive reviews are well classified. At the same time, we can find that positive reviews are more short reviews and usually do not help the product much.
- Sometimes relatively readable reviews will bring more help to your product From the trees of three products, we can find classified reviews with higher or lower readability. Short reviews are often more readable, but at the same time, reviews with very high readability are usually There is not much help for the product. The reviews with helpful attributes that we can find in the picture are relatively readable.

For these common points, we believe we can give a reasonable explanation:

- The users who send short comments are usually just to respond to the merchant or participate in the reward task of the platform or even write a review. Such reviews are rarely of great significance. Users who are willing to spend a lot of time posting long reviews are the customers who really have the right to speak about the product. It is not difficult to find bright spots from these reviews to help the product improve. Suggestion: Please don't be too troublesome to read long comments, often only long comments can give you the biggest help.
- Very positive reviews usually mean that you can't get what you need to improve the product. At the same time, these reviews can often be classified into short reviews.
- Some review texts are less readable because they are mixed with some obscure professional vocabulary. Although these reviews are low readability, they are more professional, and they are more helpful for product improvement.

We also take a closer look at the tree of each product to find the characteristics of each product:

- For hair dryer, the comments of users who have not purchased or used products are usually ignored. From the figure we can see that user reviews of unpurchased products are well classified, and these reviews are more short reviews and reviews of users who have not used the product. Such reviews are not helpful for the product.
- For microwave, neutral reviews are often not helpful for products. We can see from the figure that among all the emotional comments, neutral comments often occupy a large part. However, such reviews are shorter reviews, and there is no way to help the product.

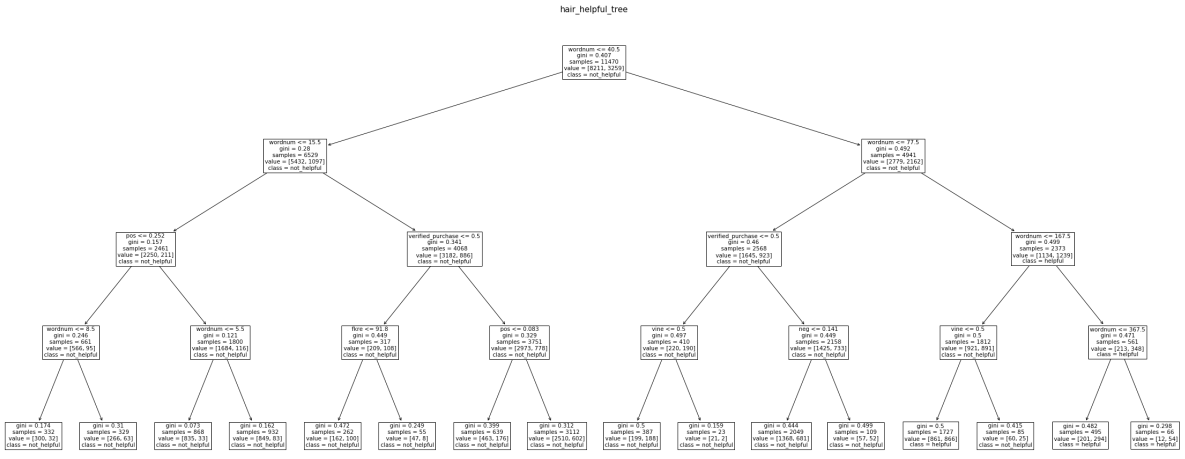


Figure 10: hair_helpfulness_tree

3.4.4 Ratings and review features

First, we still focus on the common points of three trees[Figure 13,14,15]:

- N_{pos}–PR,N_{neg}–NR
  It means, generally, the ratings are determined by the number of positive or negative words in review.

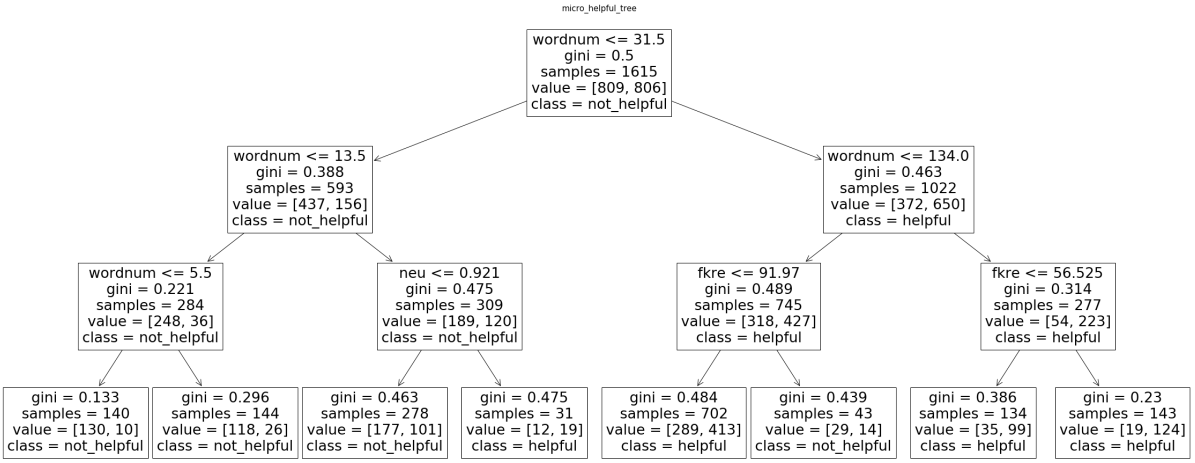


Figure 11: micro_helpfulness_tree

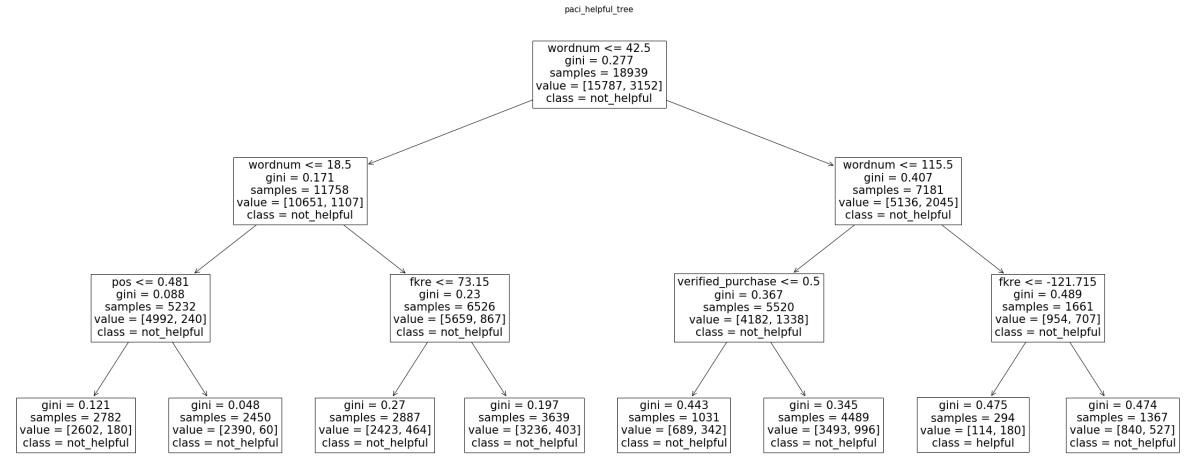


Figure 12: paci_helpfulness_tree

Now look at the different features of each product:

- For hair dryer, we found no other new features, which means we can almost predict the ratings just by N_{pos} and N_{neg}.
- For microwave, we found something through three lines: $vine–PR$ and $verified\ purchase–PR$
- For pacifier, we found no other new features, too.

3.5 PCA

3.5.1 Principle

PCA (principal components analysis) is the principal component analysis technology, which is the most widely used data dimensionality reduction algorithm. It aims to use the idea of dimensionality reduction to transform multiple indicators into a few comprehensive indicators.The steps are as follows:

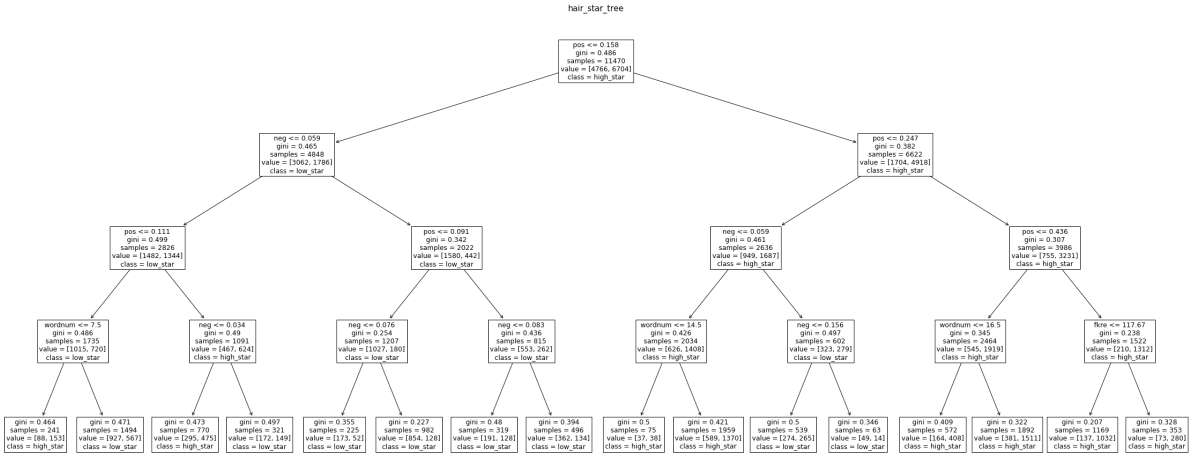


Figure 13: hair_ratings_tree

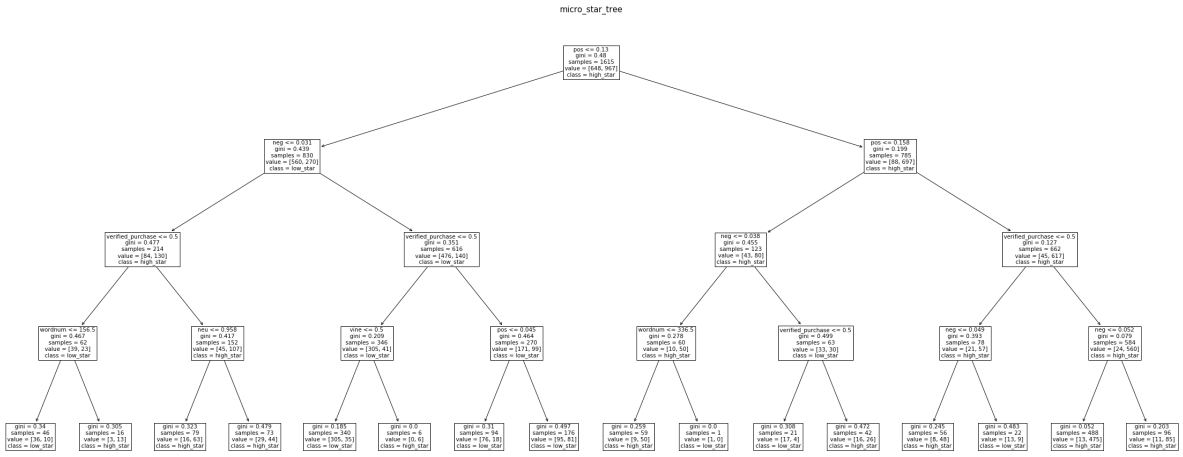


Figure 14: micro_ratings_tree

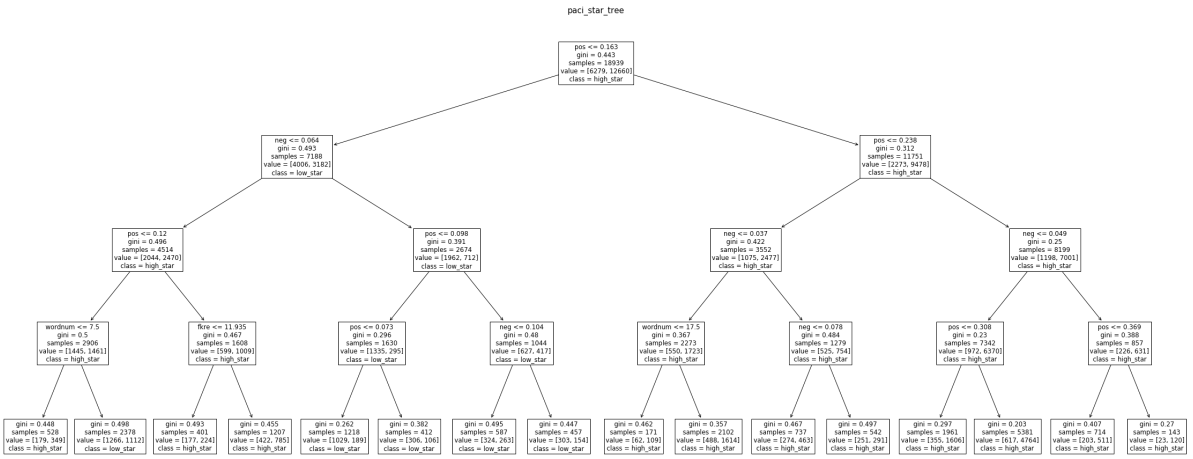


Figure 15: paci_ratings_tree

The first step is to calculate the covariance matrix S of the sample of the matrix X (this is a non-standard PCA, and the standard PCA calculates the correlation coefficient matrix C).

The second step is to calculate the eigenvectors e_1, e_2, ..., e_N and eigenvalues $t = 1, 2, ..., N$ of the covariance matrix S (or C).

The third step projects the data into the space where the feature vectors are expanded.Use the following formula,

$$newBV_{i,p} = \sum_{k=1}^{n} e_i BV_{i,k} \tag{2}$$

where the BV value is the value of the corresponding dimension in the original sample.

From this,we finally get r (r <n) new variables, make them reflect the main features of things, compress the size of the original data matrix, reduce the dimension of the feature vector, and pick out the least number of dimensions to summarize the most important features. Each new variable is a linear combination of the original variables, reflecting the comprehensive effect of the original variables, and has a certain practical meaning. These r new variables are called "principal components", they can largely reflect the influence of the original n variables, and these new variables are uncorrelated and orthogonal. Through principal component analysis, the data space is compressed, and the features of multivariate data are visually represented in a low-dimensional space.

3.5.2 Bold attempt to use PCA method to judge the potential of success or failure(2c)

We have already completed the extraction of comment features in Section 4.1, and we have corresponding scores on hand. A common aggregation method is to artificially define an indicator that uses these features as a function. Many papers have discussed various methods for scoring weights using vine and verify perchase and helpfulrate. They have their own advantages and disadvantages.

Here we propose an unsupervised learning method: let the data learn the weight itself. The specific implementation is to use principal component analysis. Our research results show that the first principal component vector is the weight that we are looking to measure the successful product.

This is a bold attempt, because our assumptions don't sound very rigorous: the direction of the vector that makes all products different is the direction of success. How can we verify that we are doing this correctly?

First, we added a validation variable: whether unit sales are in the top 25 percent of all similar products.

Then for the first principal component vectors of the three types of products, for direct verification, we do not consider their size and directly compare the coefficients of the first principal component vector. We were pleasantly surprised to find that in many well-known factors that have a positive impact on product success (such as: average rating, average review praise rate, total number of items sold, total number of items sold, and validation variables we added) are three different The coefficients of the first principal component vector elements corresponding to the products are all positive.

The corresponding symbols of these factors make us confident that the coefficient of the first principal component is the right weight, so that the value obtained by multiplying each factor by the weight is an index to measure the success of the product.

Given the value of each factor of a product, we calculate its first principal component score and compare it with the principal component score of a similar product to get the approximate position of the product in the market on which we can base on to determined whether we are successful or not.

Thanks to principal component analysis, we also draw several useful conclusions.

The first principal component coefficient corresponding to the vine variable in microwave oven products is positive, while the first principal component coefficient corresponding to the other two categories of products is negative. This indicates that the invitation to participate in the review of vine has a positive impact on the success of microwave oven products, but it has a negative impact on electricity. Hair blowing and pacifiers have a negative effect.

The greater the total number of days that have been sold, the more difficult a hair dryer and microwave oven can be, but the opposite is true for a pacifier. This is easy to understand because both the hair dryer and the microwave are appliances. But we use the unsupervised learning method to get this result naturally. This common sense conclusion further proves the rationality of our model.

So if I have anything to tell the marketing director: please invite vine to comment on your microwave oven products instead of pacifiers and hair dryers. If a microwave oven or hair dryer is sold for a long time, his downhill is coming.

This is the power of unsupervised learning.[3]

	star_rating	helpful_votes	total_votes	vine	fied_pur	fkre	helpful_rate	trans_rating	trans_helpful	rela_pos	rela_neg	posnum	negnum
hair0	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE
micro0	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE
paci0	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE

Figure 16: True indicates respective first principle component coefficient is positive

pos-negnum	weighted_pos	weighted_neg	counts	days	unt_per_	top25
FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE
FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE

Figure 17: Red dot indicate interesting pos/neg relationship with successs

3.5.3 PCA method to judge product reputaion

We also applied the techniques of PCA to the evaluation of product reputation. Although the selected variables are different, they are the same as the above analysis method, and we will not explain further. The purpose presented here is to illustrate that we will use these principal components as indicators in subsequent product reputation time series autocorrelation analysis.

3.6 Analysis of time series for reputation

3.6.1 ARMA(2b)

What is product reputation? We naturally think that reputation depends entirely on the characteristics and star rating of each review. The usual mathematical approach is to combine the characteristics of these reviews with various weighted combinations. But as we said before, we are more inclined to use unsupervised learning to judge reputation, so we choose the reputation scores obtained by using the PCA method.

How to predict whether a product's reputation will rise or fall? In fact, this is equivalent to predicting whether the mathematical expectation of the next product reputation score is higher than the mathematical expectation of the current reputation score. The traditional ARMA method can help us easily solve this problem.

We let R_t be the product reputation score

$$R_t = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + Z \tag{3}$$

Where R is the observed value of the prediction object and Z is the error. As a prediction object, R_t is affected by its own changes, and its law can be reflected by the following formula,

$$R_t = \beta_1 R_{t-1} + \beta_2 R_{t-2} + \cdots + \beta_p R_{t-p} + Z_t \tag{4}$$

The error term has a dependency relationship in different periods, which is expressed by the following formula,

$$Z_t = \epsilon_t + \alpha_1 \epsilon_{t-1} + \alpha_2 \epsilon_{t-2} + \cdots + \alpha_q \epsilon_{t-q} \tag{5}$$

From this, the ARMA model expression is obtained:

$$R_t = \beta_0 + \beta_1 R_{t-1} + \beta_2 R_{t-2} + \cdots + \beta_p R_{t-p} + \epsilon_t + \alpha_1 \epsilon_{t-1} + \alpha_2 \epsilon_{t-2} + \cdots + \alpha_q \epsilon_{t-q} \tag{6}$$

The results are as follows[Figure 18].

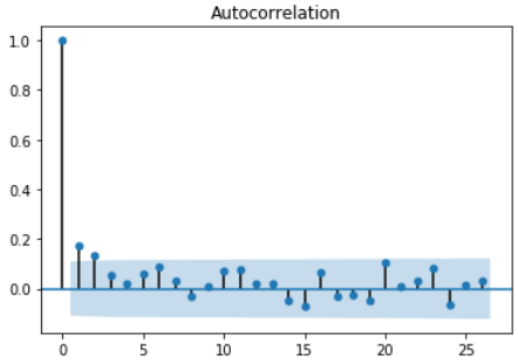
3.6.2 Autocorrelation analysis

But before we build the model, we must first think about one thing, whether the mathematical expectations of the remaining points of all products at the next moment can be obtained from the reputation scores of previous reviews.

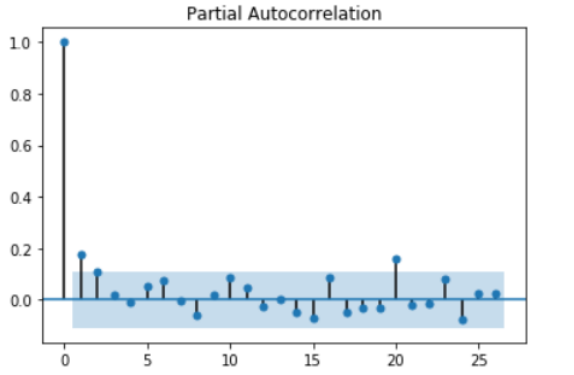
Of course not, when the reputation scores of all reviews are independent of each other, we cannot obtain any useful information from the previous reputation scores. Although this situation is not common.

So the question we are interested in is whether there are products that have a review reputation score that has nothing to do with the reputation score of previous reviews. More generally, will the reviewer s previous product evaluations have a significant impact on the reviewer s evaluation, and if so, how many recent reviews will affect him. Once we solve this problem, we can answer the marketing director, should he think that several consecutive negative comments will make the subsequent series of comments negative, so that he changes his marketing strategy, or whether the comments are positive or negative Independent so he doesnt have to worry. According to the above, we are more concerned about the shape of the autocorrelation graph than fitting the ARMA model.

We give the autocorrelation diagrams of several products below[Figure 19,20]. We can see that for some products, the time series autocorrelation is very small for each of their principal component scores, but not for some products.



(a) choose q=3 in ARMA(p,q)base on Autocorrelation score



(b) choose p=3 in ARMA(p,q)base on Partialcorrelation score

Model:	ARMA	BIC:	1284.2306
Dependent Variable:	y	Log-Likelihood:	-619.03
Date:	2020-03-10 02:23	Scale:	1.0000
No. Observations:	321	Method:	css-mle
Df Model:	7	Sample:	0
Df Residuals:	314		1
Converged:	1.0000	S.D. of innovations:	1.654
No. Iterations:	51.0000	HQIC:	1266.106
AIC:	1254.0591		

	Coef.	Std.Err.	t	P>|t|	[0.025	0.975]
const	0.0119	0.1489	0.0800	0.9363	-0.2799	0.3037
ar.L1.y	1.4864	0.1557	9.5488	0.0000	1.1813	1.7915
ar.L2.y	-1.5346	0.1105	-13.8829	0.0000	-1.7513	-1.3180
ar.L3.y	0.7773	0.1492	5.2093	0.0000	0.4848	1.0697
ma.L1.y	-1.3417	0.1847	-7.2635	0.0000	-1.7038	-0.9797
ma.L2.y	1.4501	0.1217	11.9154	0.0000	1.2115	1.6886
ma.L3.y	-0.6699	0.1832	-3.6565	0.0003	-1.0290	-0.3108

(c) ARMA(3,3) summary

Figure 18: Implement whole ARMA(p,q) on a specific product reputation score series

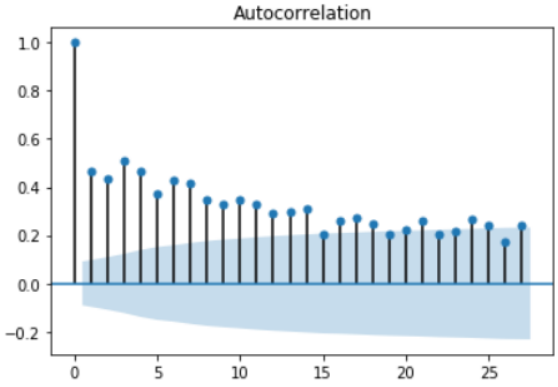


Figure 19: First kind of product :Reputation score depend highly on previous reputation score

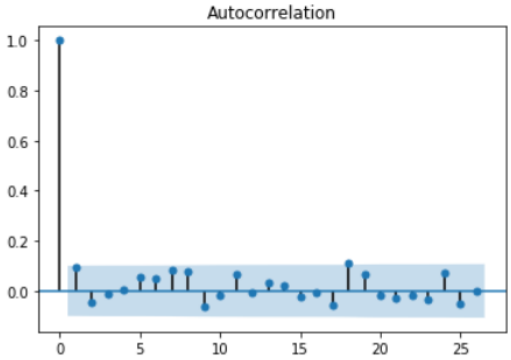


Figure 20: second kind of product :Reputation score has no relation with reputation score before

How do we interpret this result? Our explanation is that products with high time series autocorrelation should be products with very hidden defects or advantages. Only when some customers discover these defects or advantages first will later customers find them. For these time series products with high autocorrelation, the product manager should discover the problems raised by customers in time and make improvements quickly, otherwise the product reputation is likely to continue to decline. For products with low time series autocorrelation, the marketing director should not change the marketing strategy because of several consecutive negative comments. These negative comments are most likely to come from competitors, so he can think that the product reputation is not affected. I think this has solved the problem that most marketing directors hope to solve through time series. If there are certain market directors who really want to accurately predict the mathematical expectations of future reputation through the ARMA model.

3.7 Chi-square test

3.7.1 Principle

The chi-square test is the degree of deviation between the actual observed value and the theoretically inferred value of the statistical sample. The degree of deviation between the actual observed value and the theoretically inferred value determines the size of the chi-squared value. The greater the degree; on the contrary, the smaller the deviation between the two; if the two values are completely equal, the chi-square value will be 0, indicating that the theoretical values

are completely consistent.

The chi-square distribution formula is:

$$f(x) = \begin{cases} \frac{1}{2^{n/2}\Gamma(n/2)} x^{\frac{n}{2}-1} e^{-x/2}, & x > 0, \\ 0, & otherwise. \end{cases} \tag{7}$$

We introduce chi-square test statistics,

$$\chi^2 = \sum_{i=1}^{k} \frac{(f_i - np_i)^2}{np_i} \tag{8}$$

which is under the assumption that the 0 hypothesis holds, it follows a chi-square distribution with k-1 degrees of freedom.

3.7.2 Filter discriptive words(2e)

This question is so important that if we can find an automatic filtering program that filters out descriptive words of this nature, the marketing director can make targeted product improvements based on these filtered words.

Chi-square filters are the automated filtering methods we use. Here we give an example with the word "powerfulç". For all reviews, we can make the following contingency table based on whether "powerful" appears in the reviews and product ratings.[Figure 21]

star_rating	1	2	3	4	5
power					
0	947	556	840	1785	5708
1	85	83	159	311	996

(a) Crosstable

```
(36.26361333697655,
2.5539500582648775e-07,
4,
array([[ 884.98273758,  547.96896251,  856.68387097, 1797.40680035,
        5748.9576286 ],
       [ 147.01726242,   91.03103749,  142.31612903,  298.59319965,
         955.0423714 ]]))
```

(b) χ^2 test result

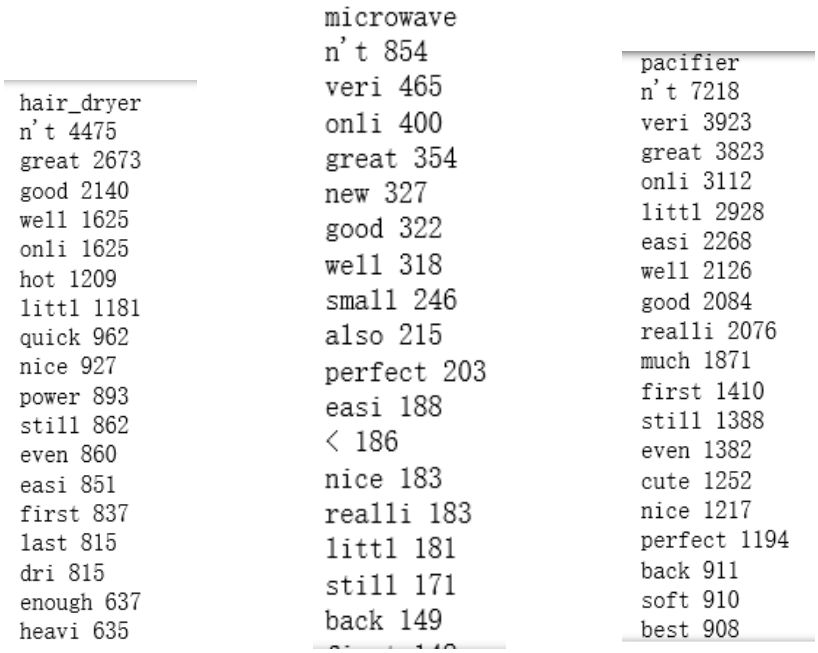
Figure 21: p-value<0.001 suggest 'power' highly associated with rating levels in hair dryer data

Then we do a hypothesis test. The original hypothesis is whether "great" has no effect on the score in the review and the alternative hypothesis has an effect. We judge by the p-value of the chi-square test. Here, in order to make the filter more powerful, the significance level is taken as 0.001. However, after we calculated the p-value, we found that it is till less than 0.001 so we reject the Assumption. This quality descriptors: great is strongly associated with rating levels.

Based on this principle, we can automatically filter out descriptive words that have a significant impact on scoring.

Here are the results we got[Figure 22].

However, there are some words it, such as "great" which does not obviously help the product improvement, but compared to the original, it is easier to pick out useful words such as: "light", "thick", "short", "big ". These words let us know which features of the product should be avoided or encouraged.



(a) hairdryer (b) microwave (c) pacifier

Figure 22: words highly associated with rating levels(number on the left indicates frequency) chosen by chi-square filter

4 Sensitivity analysis

- For seasonal goods, we need to train the model after separating the data set by season, which is difficult to accurately train on small data sets
- For seasonal goods, the ARMA model needs to be adjusted

5 Strengths and Weaknesses

- Tree model and logistic model can more clearly explain the connection between different variables instead of a black box
- Unsupervised learning method PCA avoids subjective judgments caused by artificially assigned weights
- Abandoned ensemble learning methods such as random forest, so that accurate estimates cannot be made numerically
- The PCA method ignores the interconnections between variables and only considers scores to be a linear combination of them

6 Letter

Dear Marketing Director,

Glad to share our results with you.

If your product has been on the market for some time, a time series autocorrelation analysis will tell you which category your product belongs to. If there are many high terms in the autocorrelation coefficient, this indicates that your product has some hidden advantages or disadvantages, and if it is a weakness and you don't pay attention to correction, your product reputation will continue to decline. But if the autocorrelation coefficients are all low, it means that you can't know from the past reviews whether the product's reputation will rise or fall next, so don't worry too much about the bad reviews.

If you want to know if your products are successful in the market. From a quantitative perspective, we provide three different principal component vectors for three types of products, and also provide indicators corresponding to different vectors. Based on this, the principal component score is calculated, and the ranking of the product score among all products is the position of the product in the market. And the positive and negative of the first principal component vector of the three types of products tells us, please invite VINE to participate in the review of microwave ovens instead of hair dryers and pacifiers. Microwave ovens and hair dryers will go downhill when sales are too long.

The following result are based on our research on relationship between some descriptive word and star rating:

If you want to use a computer to quickly filter out unwanted information in reviews, use Chi-Square Filtering. This is based on the fact that some descriptive words can significantly affect the distribution of ratings, which can help you quickly find words related to improving your product. It is also useful to filter comments using the tree we get. Unlike the previous chi-square filter, the tree filters out the entire comment.

This is based on our research on relationship between review features and helpful rate.

We also want to share an interesting phenomenon with you: Generally speaking, if a buyer sees that the previous five people are all bad reviews, he is more likely to make positive comments. We conclude this by regularized logistic regression.

Sincerely yours,

A group of modelers who are enthusiastic about E-commerce

References

[1] https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

[2] https://www.cnblogs.com/hit-joseph/p/11448792.html

[3] https://blog.csdn.net/program_developer/article/details/80632779

[4] Abhilasha Singh Rathor, Amit Agarwal, Preeti Dimri Comparative. Study of Machine Learning Approaches for Amazon Reviews, Procedia Computer Science, 132:1552-1561, 2018

[5] Sara Ashour Aljuhani, Norah Saleh Alghamdi, A Comparison of Sentiment Analysis Methods on Amazon Reviews, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol.10, No.6, 2019

[6] Shadi AlZubi, Abdalraheem Alsmadiv, Sokyna AlQatawneh, Mahmoud Al-Ayyoub, Bilal Hawashin, Yaser Jararweh, A Brief Analysis of Amazon Online Reviews, 2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)

[7] Yitian Zhang, Study on Online Review helpfulness Classification Based on Product Feature Mining[D], Dalian University of Technology, 2016

[8] Junkui Wang, Research on the Usefulness of E-commerce Website Online Reviews[D], Xidian University, 2014

[9] Umar Farooq, Antoine Nongaillard, Yacine Ouzrout, Muhammad Abdul Qadir, A multi source product reputation model, Computers in Industry, 83 (2016) 5567, 2016

[10] Marios Kokkodis, Learning from Positive and Unlabeled Amazon Reviews: Towards Identifying Trustworthy Reviewers, Leonard N. Stern School of Business, New York University, 2012

[11] Huiying Li, The Influence of Online Reviews on Consumer Perceptions and Product Sales[D], Harbin Institute of Technology, 2013

[12] Yaqi Chen, Research on the Relationship between Online Reviews and Consumer Purchase Intention[D], Jiangsu University of Science and Technology, 2014

[13] lshrif Ibrahim Elmurngi, Abdelouahed GherbiUnfair, Reviews Detection on Amazon Reviews using Sentiment Analysis with Supervised Learning Techniques[J], Journal of Computer Science, 14 (5): 714.726, 2018

Appendices

```
#!/usr/bin/env python
# coding: utf-8

# In[126]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import tree
from sklearn.model_selection import GridSearchCV
from datetime import datetime
import nltk.classify.util
from nltk.classify import NaiveBayesClassifier
from nltk.corpus import movie_reviews
import textstat
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.tokenize import PunktSentenceTokenizer
from nltk.stem import WordNetLemmatizer
from nltk.stem.snowball import SnowballStemmer
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
from sklearn.linear_model import LogisticRegression
from sklearn import preprocessing
from sklearn.decomposition import PCA
from  scipy.stats import chi2_contingency


# In[191]:


#import data
hair_dryer_data=pd.read_csv('hair_dryer.tsv', sep='\t')
microwave_data=pd.read_csv('microwave.tsv', sep='\t')
pacifier_data=pd.read_csv('pacifier.tsv', sep='\t')
alldata=[]
alldata.append(hair_dryer_data)
alldata.append(microwave_data)
alldata.append(pacifier_data)


# In[192]:


#data preprocessing
for i in range(3):
    alldata[i]['verified_purchase']=alldata[i]['verified_purchase'].str.upper()
    alldata[i]['verified_purchase']=(alldata[i]['verified_purchase']=='Y')+0
    alldata[i]['vine']=alldata[i]['vine'].str.upper()
    alldata[i]['vine']=(alldata[i]['vine']=='Y')+0
# review feature extraction
    alldata[i]['fkre']=[textstat.flesch_reading_ease(review)if type(review)==str else
    0 for review in alldata[i]['review_body']]
    alldata[i]['syllablenum']=[textstat.syllable_count(review)if type(review)==str else
```

```
    0 for review in alldata[i]['review_body']]
    alldata[i]['wordnum']=[textstat.lexicon_count(review)if type(review)==str else
    0 for review in alldata[i]['review_body']]
    alldata[i]['sentencenum']=[textstat.sentence_count(review)if type(review)==str else
    0 for review in alldata[i]['review_body']]
    alldata[i]['helpful_rate']=alldata[i]['helpful_votes']/alldata[i]['total_votes']


# In[193]:


analyzer=SentimentIntensityAnalyzer()
for i in range(3):
    print('loading...')
    alldata[i]['pos']=[analyzer.polarity_scores(text)['pos'] if type(text)==str else
    0.33 for text in alldata[i]['review_body']]
    alldata[i]['neu']=[analyzer.polarity_scores(text)['neu'] if type(text)==str else
    0.33 for text in alldata[i]['review_body']]
    alldata[i]['neg']=[analyzer.polarity_scores(text)['neg'] if type(text)==str else
    0.33 for text in alldata[i]['review_body']]


# In[194]:


#feature categorize
for i in range(3):
    if i==0:
        alldata[i]['trans_rating']=[1 if num==5 else 0 for num in alldata[i]['star_ratin
    else:
        if i==1:
            alldata[i]['trans_rating']=[1 if num>3 else 0 for num in alldata[i]['star_ra
        else:
            alldata[i]['trans_rating']=[1 if num==5 else 0 for num in alldata[i]['star_r
    alldata[i]['trans_helpful']=[1 if num>0.5 else 0 for num in alldata[i]['helpful_rate


# In[195]:


#set onfounding parametres
for i in range(3):
    alldata[i]['rela_pos']=(alldata[i]['pos']>alldata[i]['neg'])+0
    alldata[i]['rela_neg']=1-alldata[i]['rela_pos']
    alldata[i]['helpful*neg']=alldata[i]['trans_helpful']*(1-alldata[i]['rela_pos'])
    alldata[i]['helpful*pos']=alldata[i]['trans_helpful']*alldata[i]['rela_pos']
    alldata[i]['nothelp*neg']=(1-alldata[i]['trans_helpful'])*(1-alldata[i]['rela_pos'])
    alldata[i]['nothelp*pos']=(1-alldata[i]['trans_helpful'])*alldata[i]['rela_pos']


# In[7]:


#data preprocessing
for data in alldata:
    poslist=[]
    neglist=[]
    for review in data['review_body']:
```

```
        posnum=0
        negnum=0
        if type(review)==str:
            for word in nltk.word_tokenize(review):
                a=analyzer.polarity_scores(word)
                if a['pos']==1:
                    posnum+=1
                if a['neg']==1:
                    negnum+=1
        poslist.append(posnum)
        neglist.append(negnum)
    data['posnum']=poslist
    data['negnum']=neglist
    data['pos-negnum']=data['posnum']-data['negnum']


# In[8]:


#data preprocessing
for i in range(3):
    alldata[i].fillna(0,inplace=True)
    #alldata[i].drop(columns=['marketplace', 'customer_id', 'review_id', 'product_id',
     #  'product_parent', 'product_title', 'product_category'],inplace=True)
    alldata[i]['weighted_pos']=alldata[i]['helpful_rate']*alldata[i]['rela_pos']
    alldata[i]['weighted_neg']=alldata[i]['helpful_rate']*alldata[i]['rela_neg']


# In[9]:


Xtrain=alldata[2][['pos','neu','neg','fkre','wordnum','vine','verified_purchase','trans_
ytrain=alldata[2]['trans_helpful']
scaler = preprocessing.StandardScaler().fit(Xtrain)
Xtrain=scaler.transform(Xtrain)


# In[204]:


allall=pd.DataFrame()
for data in alldata:
    allall=pd.concat([allall,data])


# In[205]:


pd.crosstab(allall['rela_pos'],allall['star_rating'])


# In[208]:


chi2_contingency(pd.crosstab(allall['rela_pos'],allall['star_rating']))


# In[213]:
```

```
pd.crosstab(allall['trans_helpful'],allall['star_rating']).div(pd.crosstab(allall['trans

# In[214]:

pd.crosstab(allall['trans_helpful'],allall['trans_rating']).div(pd.crosstab(allall['tran

# In[209]:

chi2_contingency(pd.crosstab(allall['trans_helpful'],allall['star_rating']))

# In[207]:

pd.crosstab(allall['rela_pos'],allall['trans_helpful'])

# In[212]:

chi2_contingency(pd.crosstab(allall['rela_pos'],allall['trans_helpful']))

# In[10]:

param_grid = { 'C': [0.001,0.01,0.1,1,10,100,1000,10000]}
mytree=LogisticRegression()
clf=GridSearchCV(mytree, param_grid, cv=5,scoring='roc_auc')
clf.fit(Xtrain,ytrain)
pd.DataFrame(clf.cv_results_)

# In[11]:

clf3=LogisticRegression(C=0.001).fit(Xtrain,ytrain)
clf3.coef_

# In[12]:

Xtrain=alldata[2][['pos','neu','neg','fkre','wordnum','vine','verified_purchase','trans_
ytrain=alldata[2]['trans_helpful']

# In[13]:

param_grid2 = { 'max_depth': [1,2,3,4,5,6]}
mytree2=tree.DecisionTreeClassifier()
```

```
clf2=GridSearchCV(mytree2, param_grid2, cv=5,scoring='roc_auc')
clf2.fit(Xtrain,ytrain)
pd.DataFrame(clf2.cv_results_)


# In[14]:


clf3=tree.DecisionTreeClassifier(max_depth=3)
clf3.fit(Xtrain,ytrain)
fig=plt.figure(figsize=(40,16))
ax=fig.add_subplot(1,1,1)
tree.plot_tree(clf3,ax=ax,feature_names=['pos','neu','neg','fkre','wordnum','vine','veri
               class_names=['not_helpful','helpful'])
plt.title('paci_helpful_tree',fontsize=15)


# In[15]:


#groupby product id
all_group=[]
all_group.append(hair_dryer_data.groupby('product_id'))
all_group.append(microwave_data.groupby('product_id'))
all_group.append(pacifier_data.groupby('product_id'))


# In[16]:


all_group_detail=[[],[],[]]
for i in range(3):
    for _,group in all_group[i]:
        all_group_detail[i].append(group)
    all_group_detail[i].sort(key=len,reverse=True)


# In[17]:


for products in all_group_detail:
    for product in products:
        product['days']=[(datetime.strptime(date,'%m/%d/%Y')-datetime.strptime(product['
            for date in product['review_date']]


# In[18]:


all_info=[]
for i,group in enumerate(all_group):
    product_count=all_group[i].count().sort_values(by=['star_rating'],ascending=False)
    products_info=all_group[i].mean().loc[product_count.index]
    products_info['counts']=product_count['star_rating']
    products_info=products_info[products_info['counts']>=10]
    products_info['days']=[product['days'].max() for product in all_group_detail[i][0:le
    products_info['count_per_day']=products_info['counts']/products_info['days']
    products_info['helpful_rate']=products_info['helpful_votes']/products_info['total_vo
```

```
    products_info['successful']= (products_info['count_per_day']>=products_info['count_p
    all_info.append(products_info)


# In[19]:


arima_group=[[],[],[]]
for i in range(3):
    for group in all_group_detail[i]:
        arima_group[i].append(group[['star_rating','helpful_rate','trans_rating','trans_


# In[20]:


pcares=[[],[],[]]
post_group=[[],[],[]]
for i,groups in enumerate(arima_group):
    for group in groups:
        if group.shape[0]>4:
            Xtrain=group.values
            pca_digits = PCA(n_components=4)
            scaler = preprocessing.StandardScaler().fit(Xtrain)
            Xtrain=scaler.transform(Xtrain)
            Xtrain = pca_digits.fit_transform(Xtrain)
            100*pca_digits.explained_variance_ratio_
            pcares[i].append(pca_digits)
            post_group[i].append(Xtrain)
        else:
            pcares[i].append(0)
            post_group[i].append(0)


# In[180]:


regar=post_group[0][4][:,0]


# In[113]:


post_group[0][3]


# In[181]:


from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
plot_acf(regar)
plot_pacf(regar)


# In[108]:


from statsmodels.tsa.stattools import adfuller
```

```
adfuller(regar)
from statsmodels.stats.diagnostic import acorr_ljungbox
print(u'',acorr_ljungbox(regar, lags= 1))


# In[183]:


import statsmodels.api as sm
model=sm.tsa.ARIMA(regar,(3,0,3)).fit()
model.summary2()


# In[25]:


for data in alldata:
    Xtrain=data[['star_rating', 'helpful_votes', 'total_votes', 'vine',
       'verified_purchase', 'fkre','helpful_rate', 'pos', 'neu', 'neg', 'trans_rating',
        'trans_helpful','rela_pos', 'helpful*neg', 'helpful*pos', 'nothelp*neg',
        'nothelp*pos','rela_neg', 'posnum', 'negnum', 'pos-negnum'
       ]]


# In[33]:


pcadf=pd.DataFrame(columns=all_info[0].columns,index=['hair0','micro0','paci0','hair1','


# In[35]:


all_info[2].fillna(0,inplace=True)
pcares=[]
for info in all_info:
    Xtrain=info.values
    pca_digits = PCA(n_components=16)
    scaler = preprocessing.StandardScaler().fit(Xtrain)
    Xtrain=scaler.transform(Xtrain)
    Xtrain = pca_digits.fit_transform(Xtrain)
    100*pca_digits.explained_variance_ratio_
    pcares.append(pca_digits)


# In[36]:


for i in range(4):
    pcadf.loc['hair{}'.format(i)]=pcares[0].components_[i]<0
    pcadf.loc['micro{}'.format(i)]=pcares[1].components_[i]<0
    pcadf.loc['paci{}'.format(i)]=pcares[2].components_[i]<0


# In[37]:


pcadf.to_excel('2c.xlsx')
```

```
# In[38]:


all_star_review=[]
for groups in all_group_detail:
    part_star_review=[]
    for group in groups:
        if group.shape[0]>5:
            star_review=group[['pos','neu','neg','rela_pos']][:-5]
            for i in range(5):
                if i!=4:
                    star_review['back{}'.format(i+1)]=group['trans_rating'].values[i+1:i
                else:
                    star_review['back{}'.format(i+1)]=group['trans_rating'].values[i+1:]
            star_review['deviation']=(star_review['back1']+star_review['back2']+star_rev
-group['trans_rating'].mean()
        part_star_review.append(star_review)
    all_star_review.append(part_star_review)


# In[39]:


star_review_all=pd.DataFrame()
for part_star_review in all_star_review:
    for star_review in part_star_review:
        star_review_all=pd.concat([star_review_all,star_review])


# In[40]:


Xtrain=star_review_all[['back1', 'back2', 'back3', 'back4', 'back5','deviation']]
ytrain=star_review_all['rela_pos']


# In[41]:


scaler = preprocessing.StandardScaler().fit(Xtrain)
Xtrain=scaler.transform(Xtrain)


# In[42]:


param_grid6 = {'max_depth': [1,2,3,4,5,6]}
mytree6=tree.DecisionTreeClassifier()
clf6=GridSearchCV(mytree6, param_grid6, cv=5,scoring='roc_auc')
clf6.fit(Xtrain,ytrain)
pd.DataFrame(clf6.cv_results_)


# In[59]:
```

```
clf7=tree.DecisionTreeClassifier(max_depth=5)
clf7.fit(Xtrain,ytrain)
fig=plt.figure(figsize=(60,25))
ax=fig.add_subplot(1,1,1)
tree.plot_tree(clf7,ax=ax,feature_names=['back1', 'back2', 'back3', 'back4', 'back5','de


# In[44]:


Xtrain=star_review_all[['back1', 'back2', 'back3', 'back4', 'back5','deviation']]
ytrain=star_review_all['rela_pos']


# In[60]:


param_grid8={'C': [0.01,0.1,1,10,100,1000,10000]}
mytree8=LogisticRegression()
clf8=GridSearchCV(mytree8, param_grid8, cv=5,scoring='roc_auc')
clf8.fit(Xtrain,ytrain)
pd.DataFrame(clf8.cv_results_)


# In[46]:


mylog9=LogisticRegression(C=10).fit(Xtrain,ytrain)
mylog9.coef_


# In[149]:


mylog9.coef_.sum()+mylog9.intercept_


# In[143]:


mylog9.intercept_


# In[47]:


import re
regex=re.compile('\w*')
worddicts=[]
lenlist=[]
stemmer_snowball = SnowballStemmer('english')
list_stopWords=list(set(stopwords.words('english')))
for i in range(3):
    print('loading')
    wordlist=[]
    worddict={}
    for reviews in alldata[i]['review_body']:
        if(type(reviews)==str):
```

```
            tagword=nltk.pos_tag(nltk.word_tokenize(reviews))
            tagwords=[]
            for word,po in tagword:
                if (po=='JJ')|(po=='JJR')|(po=='JJS')|(po=='RB'):
                    tagwords.append(word)
            stemwords=[stemmer_snowball.stem(word) for word in tagwords]
            wordlist.extend(stemwords)
    cleanword=[word for word in wordlist if word not in  list_stopWords]
    for word in cleanword:
        if word in worddict:
            worddict[word]+=1
        else:
            worddict[word]=1
    worddicts.append(sorted(worddict.items(),key=lambda x:x[1],reverse=True))


# In[135]:


def discriptive(given_word,val=[0,1,2],data=alldata):
    for i in val:
        haveword=[]
        for reviews in alldata[i]['review_body']:
            if(type(reviews)==str):
                if given_word in [stemmer_snowball.stem(word) for word in nltk.word_toke
                    haveword.append(1)
                else:
                    haveword.append(0)
            else:
                haveword.append(0)
        data[i][given_word]=haveword


# In[136]:


prodtype=['hair_dryer','microwave','pacifier']


# In[141]:


pd.crosstab(alldata[0]['power'],alldata[0].star_rating)


# In[142]:


chi2_contingency(pd.crosstab(alldata[0]['power'],alldata[0].star_rating))


# In[137]:


for i,worddict in enumerate(worddicts):
    print(prodtype[i])
    for word,freq in worddict:
        if freq>50:
```

```
            discriptive(word,[i])
            see_if_related=pd.crosstab(alldata[i][word],alldata[i].star_rating)
            if chi2_contingency(see_if_related)[1]<0.001:
                print(word,freq)


# In[53]:


worddicts[0]


# In[54]:


worddicts[1]


# In[55]:


worddicts[2]
```