Commentary Analysis: A Machine or a Customer?

*Bojun Chen[1], Yiqi Li[1, *], Wenxuan Zhu[2], Qiqi Hong[1], Jiayin Liu[1], Yuesheng Huang[1]*

(1: Guangdong Polytechnic Normal University, Guangdong, China

2: Guangdong University of Finance, Guangdong, China)

ABSTRACT

In the field of e-commerce, shopping reviews play a significant role in influencing consumers' purchase intention. This paper utilizes natural language processing techniques to analyze the information in e-commerce reviews in detail in order to facilitate digital insights.

For Task 1, this paper utilizes Python to extract the review text, split words, and remove deactivated words to preprocess the data. After that, this paper plots the word clouds respectively, and also draws bar charts for the top 20 words in terms of frequency of occurrence in the reviews as well as the items with the most 5-point reviews in OVERALL. Finally, this paper preliminarily analyzes the importance of keywords in review texts based on the bureau TF-IDF method.

For task 2, after the same preprocessing, the fastText tool is utilized to construct word vectors combined with the weights of words obtained from TextRank for keyword extraction. After the feature extraction, a machine learning-based semantic classification model was constructed with the ASIN field of Appendix III and IV as the classification variable. By comparing different machine learning classifiers, the SVM classifier with the highest accuracy is selected as the algorithm for the final semantic analysis model in this paper, and the accuracy rate reaches 69%.

For Task 3, feature extraction based on TF-IDF Weighted Word2vec was processed for the comment text after preprocessing, and for the SVM classifier selected for Task 2, the PSO algorithm was used for optimization, and learning factors and inertia weights were added to improve it, and a text sentiment analysis model based on AIPSO-SVM was constructed. In this paper, experiments were conducted on the data of Appendix V and VI. The experimental accuracy rate reaches more than 75%.

For Task 4, this paper constructs an evaluation system through seven evaluation criteria, such as different angles of comment text, comment time, OVERALL and HELPFUL fields. And based on this, we established a machine review research and judgment model based on T-S fuzzy neural network, and combined with 2D Fourier transform to convert it into a threshold, and experimented with 6 attachments, according to the results, we wrote a letter of advice to help customers identify the categories of reviews.

The highlights of this pape are the comparison of multiple classifiers, the incorporation of an improved PSO algorithm to optimize the classifiers, and the construction of a discriminative model via T-S fuzzy neural networks.

Keywords: Text feature extraction; AIPSO algorithm; T-S fuzzy neural network; Natural Language Processing (NLP); Semantic Analysis

1 INTRODUCTION

1.1 Problem Background

In the field of e-commerce, reviews of a product by consumers who have already purchased it, whether positive or not, largely shape the purchase intentions of other consumers. Therefore, e-commerce reviews have multiple functions and values for brand merchants [1]. Nowadays, brand merchants can digitally improve consumer insights by deeply analyzing the information contained in e-commerce reviews with the help of advanced technologies, such as artificial intelligence and natural language processing, so as to better understand consumer needs and optimize products and services.

1.2 Restatement of the Problem

Problem 1: Based on appendices 1 and 2, a text analytics digital model is built to achieve word frequency statistics, drawing text cloud maps, as well as visualization and in-depth analysis of data and information in merchandise reviews.

Problem 2: Based on Appendices 3 and 4, a semantic analysis numerical model is built to achieve automatic extraction of keywords in commodity reviews, accurate prediction of commodity names, as well as visualization and in-depth analysis of the prediction results [2].

Problem 3: Based on Appendix 5 and 6, establish a mathematical model for sentiment analysis in order to realize the automatic extraction of keywords in product reviews and thus the accurate prediction of product ratings, as well as the comparative analysis of the results with the actual data.

Problem 4: Build a rating system to determine whether a product review is made by a customer or a machine, and validate the evaluation criteria. Finally, write an online shopping advice letter for the customer on the results of the model.

2 ASSUMPTIONS

Assumption I: Assuming that the data collected are true and valid.

Assumption II: Assuming that the name of the product in topic II corresponds to the "asin" key of the attachment.

Assumption III: Assuming that the lexical and syntactic structures in the text contain semantic and affective information of the text.

3 NOTATIONS

Symbols	Descriptions
TF	Frequency of occurrence of a word in a document
IDF	Indicators of the discriminatory power of a word in the corpus
WS(Vi)	denotes the weight of the node Vi
Pid、gid	Individual optimization, global optimization
ye	Output to Input Error

4 TEXT ANALYSIS MODEL

4.1 Data Preprocessing

4.1.1 Extracting Review Text Data

By examining the data provided in Appendix I and Appendix II, we can see that the data is in JSON format. Therefore, we need to first filter the review Text data. We use the Python lambda function：

lambda x: ast.literal_eval(x)['reviewText'].lower( ) to process each element (review text) x. Using ast.literal_eval(x) converts the string type review text x into a Python dictionary. We then extract the value associated with the key 'review Text' from the dictionary, which corresponds to the review text. Finally, we use the lower() method to convert the review text into lowercase [3].

4.1.2 Removal of Stopwords

The purpose of removing stopwords when calculating word frequency in reviews is to exclude words that appear frequently in the text but have no actual meaning or importance. Stopwords refer to common words in natural language such as "the", "a", "and", "is", etc. They often appear frequently in the text, but do not contribute much to the meaning and analysis of the text [4]. Removing these stopwords can reduce the noise in the text and make the analysis more accurate and meaningful. In this paper, we use the built-in stopwords list 'stopwords' in the NLTK library in Python to remove stopwords.

4.2 Data Visualization

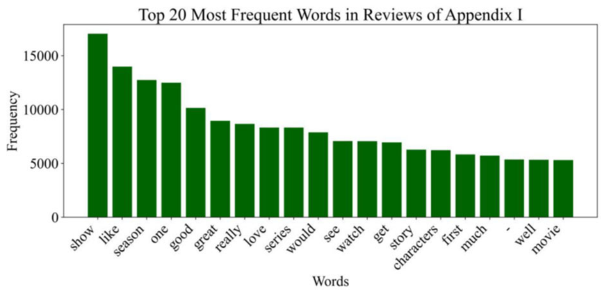
4.2.1 Word Cloud

After the aforementioned data preprocessing, we can use Python to draw the word clouds of the reviewText data in Appendix I and Appendix II, as shown below.

Figure 1: Appendix I and appendix II Word Cloud

4.2.2 Visualization of Top 20 Words by Frequency

We can use a bar chart in Python to visualize the frequency of the review text in Appendix I and Appendix II, as shown below.



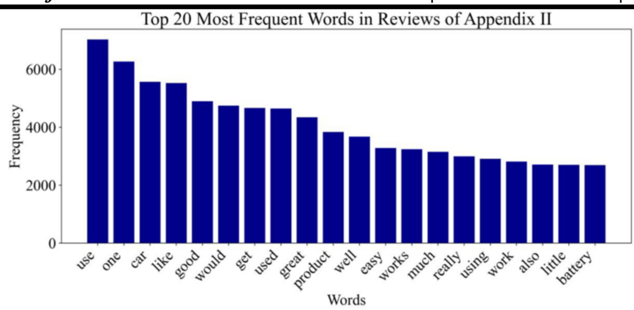


Figure 2: Top 20 words by frequency for Appendix

4.2.3 Overall Visualization

We separately analyzed the 'overall' rating for Appendix I and Appendix II, and identified the top 20 ASINs with an 'overall' rating of 5.0 for each dataset. The results are shown in the following figure.

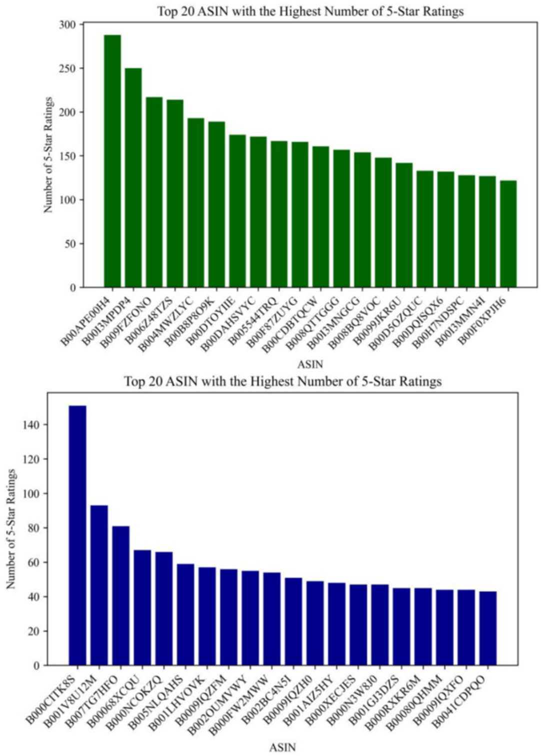


Figure 3: Top 5 Stars

4.3 Word Importance Analysis based on TF-IDF

TF refers to the frequency of a term in a particular document. After calculating th term frequency, it is usually normalized. The formula for calculating the TF is as follows:

$$TF = \frac{\omega}{W} \tag{1}$$

Where ω is the number of times a term appears in a document, and 'W' is the total number of words in the document [5].

IDF (inverse document frequency) refers to the distinguishing power of a term in a collection of documents or corpus. The formula for calculating IDF is:

$$IDF = \log\left(\frac{F_n + 1}{F_\omega + 1}\right) \tag{2}$$

Where F_n is the total number of documents in the collection or corpus, F_ω is the number of documents containing a particular word, $F_\omega + 1$ is added to avoid division by zero, and $F_n + 1$ is added to ensure that the value inside the logarithm is always greater than or equal to one, resulting in a non-negative IDF value.

For a collection of documents or corpus, the fewer the number of documents containing a particular word, the higher the IDF value, indicating that the word is more important and has greater distinguishing power [6].

Taking into account both the local information of a document based on TF and the global information of a corpus based on IDF, we can obtain the following formula.

$$TF - IDF = TF \times IDF \tag{3}$$

We used the nltk and scikit-learn libraries in Python to calculate the top 20 important words for each dataset based on TF-IDF, and the results are shown in the following figure.

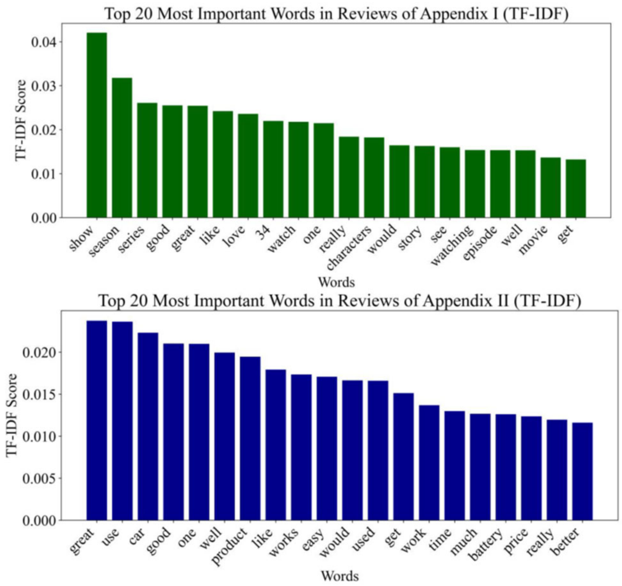


Figure 4: TF-IDF for two appendices

5 SEMANTIC ANALYSIS MODEL

The process of a semantic analysis model is shown in the following figure.

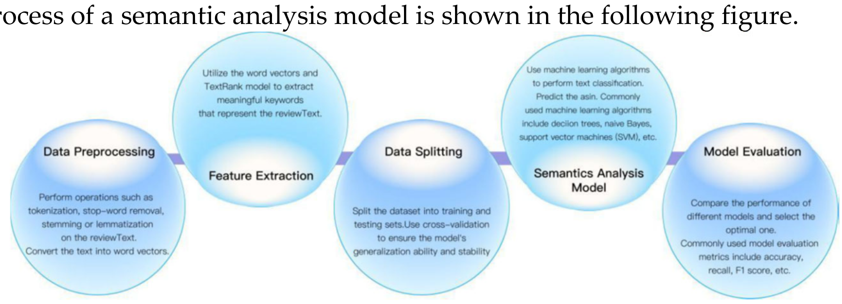


Figure 5 : Semantic Analysis Process

5.1 Semantic Classification Model Based on Machine Learning

In this paper, the extracted features and labels are combined to form a dataset. The features are word vectors extracted from the comments, and the labels are the corresponding ASINs. The dataset is then split into a training set and a test set. The paper uses random sampling to assign 70% of the dataset as the training set and 30% as the test set. The following comparison will be made using multiple classifiers.

5.1.1 Naive Bayes Classifier（NB）

Naive Bayes is a classification algorithm based on Bayes' theorem and the assumption of conditional independence between features. It assumes that all features are independent of each other, which simplifies the model. Specifically, the Naive Bayes algorithm first calculates the prior probability for each class, then calculates the conditional probability of each feature given each class, and finally uses Bayes' theorem to calculate the posterior probability and selects the class with the highest posterior probability as the prediction result. The algorithm process is shown in the following figure.

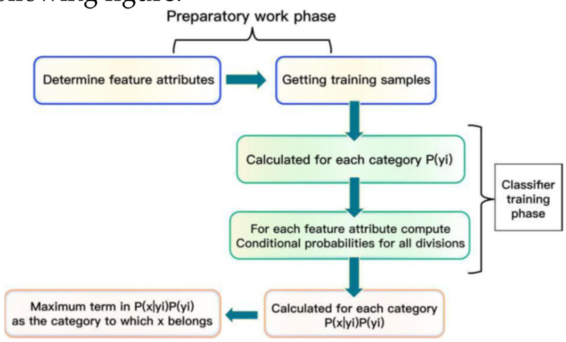


Figure 6: Naive Bayes process

5.1.2 Support Vector Machine Classifier （SVM）

Support Vector Machine (SVM) is mainly used for classification problems. The basic idea of SVM is to find an optimal hyperplane that separates the sample points of different classes and maximizes the distance (i.e., margin) between the hyperplane and the sample points of the two classes. SVM can use a kernel function to map the samples from the original space to a high-dimensional feature space, making the samples linearly separable in the high-dimensional feature space. At this point, the hyperplane becomes a line segment, with the equation $\omega x + b = 0$, and the distance between the sample points and the hyperplane is given by.

$$y_i(\omega x_i + b) \geq 1, i = 1,2,3, \ldots n \tag{4}$$

Where the support vectors are the points that are at a distance of 1 from the hyperplane. However, in many cases, it is impossible to fully separate the samples linearly, so slack variables ξi and a penalty parameter C are introduced, as well as a kernel function $K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$, which allows the SVM to be transfor and solved as a linear problem. The final expression for the decision function is:

$$f(x) = sign\left(\sum_{i=1}^{n} \alpha_i^* y_i K(x_i, x) + b^*\right) \tag{5}$$

The principle of the feature space for SVM is shown in the following figure.

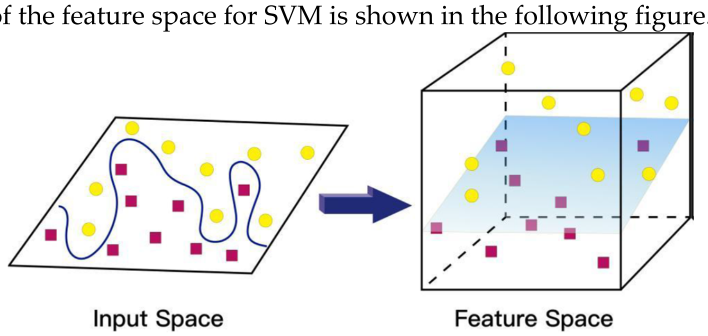


Figure 7: SVM Schematic

5.1.3 Random Forest Classifier

The random forest algorithm is a classification method that combines multiple decision trees. A random forest is composed of multiple decision trees, and each decision tree classifies the data [7]. Each decision tree is independent and is a non-parametric model. The algorithm process is as follows.

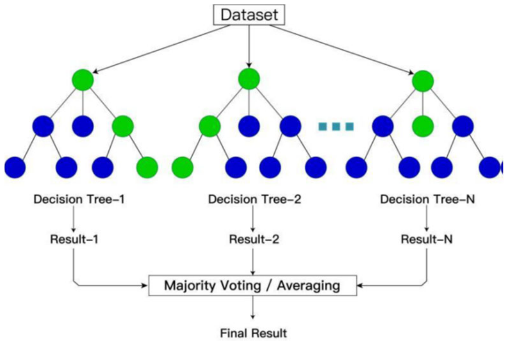


Figure 8: Random Forest Schematic

5.3 Model Results and Evaluation

Based on the machine learning classifiers described above, we used the sklearn library in Python to build the classifiers. We used the data from Appendix 3 and Appendix 4, and randomly shuffled the dataset for training. We evaluated the classifiers using metrics such as Accuracy, Precision, Recall, and F1 Score.

5.3.1 Accuracy

Accuracy is a basic metric used to evaluate the performance of a classification model. It represents the proportion of correctly predicted samples out of the total samples. The formula for calculating accuracy is:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{6}$$

Where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

5.3.2 Precision

Precision is the proportion of correctly classified positive samples to the total number of samples predicted as positive by the classifier. Precision measures the classifier's ability to identify positive samples [8]. The formula for calculating precision is:

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

5.3.3 Recall

Recall is the proportion of correctly classified positive samples to the total number of actual positive samples. Recall measures the classifier's ability to find positive samples. The formula for calculating recall is:

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

5.3.4 F1 Score

F1 Score is the harmonic mean of precision and recall, and is a metric that can comprehensively evaluate the performance of a classifier. The formula is：

$$F1\text{-}Score = \frac{2TP}{2TP + FP + FN} \tag{9}$$

After determining the evaluation metrics, we compared the results, as shown in the following figure.

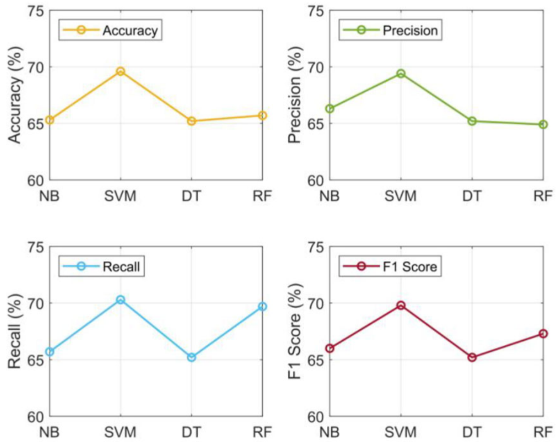


Figure 9 : Model Evaluation Comparison

It can be seen that the SVM model has better performance than the other three models, with the following metrics.

Table 1: SVM indicators

Model Name	Accuracy	Precision	Recall	F1 Score	AUC
SVM	0.696	0.694	0.703	0.698	0.746

At this point, we have completed the entire semantic analysis model. We will use word embeddings and the Text Rank algorithm for keyword filtering, and then use the SVM classifier, which has high accuracy, for predicting the asin. This is the best approach based on our evaluation of the different models.

6 AIPSO-SVM MODEL FOR TEXT SENTIMENT ANALYSIS

In the previous model, we have introduced the SVM classification model, after we have compared several machine learning classifiers, we found that for text sentiment analysis, it is still the SVM that works better, but the overall effect did not meet the expectations, so we propose to optimize the SVM classifier based on the improvement of the particle swarm algorithm [9]. The supervisory factor of the above SVM classifier is changed to OVERALL to continue the subsequent research.

6.1 Adaptive Inertia Weight Particle Swarm Optimization

The PSO algorithm was inspired by the foraging behavior of birds and is capable of searching multiple areas of the optimization target range simultaneously, greatly accelerating the search speed and enabling fast convergence. The algorithm is easy to understand: a group of particles is randomly selected and initialized, and the optimal particle location is determined by the size of the fitness value.

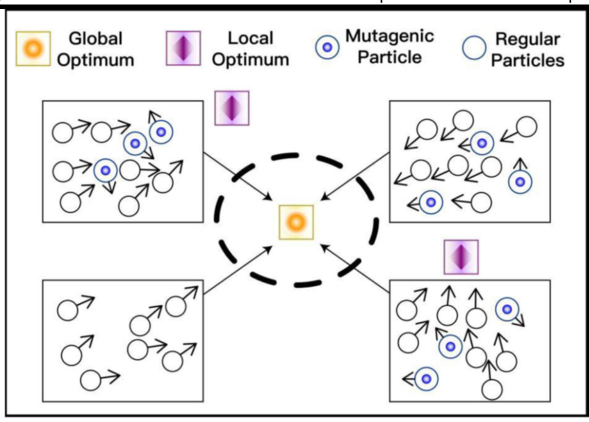


Figure 10: Schematic diagram of particle swarm algorithm

Once the optimal particle location is found, all particles start moving towards it to continuously update their speed and position. The particle's individual optimal solution (pbest) is used in combination with the global optimal solution (gbest) to update the speed and position continuously and solve the problem [10]. The algorithm stops when the constantly updating global optimal solution (gbest) reaches the maximum number of iterations or the increment of the fitness value is less than the threshold value. The equations for the velocity of the particle and the position of the particle are as follows.

$$v_{id}^{t} = \omega v_{id}^{t-1} + c_1 r_1 (p_{id} - x_{id}^{t-1}) + c_2 r_2 (g_{id} - x_{id}^{t-1}) \tag{10}$$

$$x_{id}^{t} = x_{id}^{t-1} + v_{id}^{t} \tag{11}$$

In this equation, v_{id}^{t} represents the current velocity of the particle, ω is the inertia weight, c_1 and c_2 are learning factors greater than zero. r_1 and r_2 are random data between 0 and 1, and pid and gid represent the individual best value and global best value, respectively.

6.2 Improvement of Inertial Weight

The inertial weight (ω) represents the influence of the previous velocity on the current velocity. A larger ω means stronger global search ability but weaker local search ability, while a smaller ω leads to the opposite. In the early stages, it's better for particles to have good global search ability to explore new areas, while in the later stages, local search ability needs to be improved for higher convergence accuracy [11]. To improve search efficiency and avoid premature convergence, a nonlinear function is proposed to dynamically adjust ω.

$$h(t) = \frac{2}{1 + e^{-\frac{2t}{t_{max}}}} - 1 \tag{12}$$

$$\omega(t) = \omega_{max} - (\omega_{max} - \omega_{min})[h(t) - 1] \tag{13}$$

Where t represents the current iteration number, t max represents the maximum iteration number, $\omega(t)$ represents the current inertial weight, ω max represents the maximum inertial weight, and ω min represents the minimum inertial weight [12]. The inertial weight decreases as the iteration number increases, further improving the particle's optimization effect.

6.3 Improvement of Learning Factor

When c_1=0, particles lack individual diversity and can have difficulty escaping from local optima. When c_2=0, particles only consider their own awareness and not the information provided by the collective, leading to delayed convergence speed [13]. To address this, a nonlinear function is proposed to dynamically adjust the learning factors c_1 and c_2.

$$c_1(t) = (c_{1e} - c_{1s})\left(\frac{t}{t_{max}}\right)^2 + c_{1s} \tag{14}$$

$$c_2(t) = (c_{2e} - c_{2s})\left(\frac{t}{t_{max}}\right)^2 + c_{2s} \tag{15}$$

Where c_1s and c1e are the initial and final values of c_1, and c_2s and c_2e are the initial and final values of c_2. The formula shows that as the iteration number increases, c_1 decreases continuously while c_2 increases continuously.

6.4 Performance Analysis of AIPSO

This paper evaluates the performance of the improved particle swarm optimization algorithm (AIPSO) by using classic test functions. The compared methods include Standard PSO and IP-SO, which is a particle swarm optimization algorithm with linearly decreasing weights. The first test function is the Quartic function, which is a unimodal function expressed as f1 $= x_1^4 + 2x_2^4 +$ rand ( ). The figure shows the 3D plot of the f1 test function.

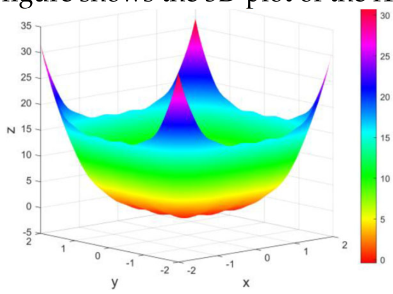


Figure 11: Quartic Function

During the test, the particle swarm size was set to 30 and the particle dimension was set to 2. The maximum number of iterations was set to 200. Based on experience, ω max was set to 0.9 and ω min was set to 0.4. The fitness function curves of the Standard PSO, IP-SO, and AIPSO algorithms were compared as the iteration number increases, as shown in the following figure.

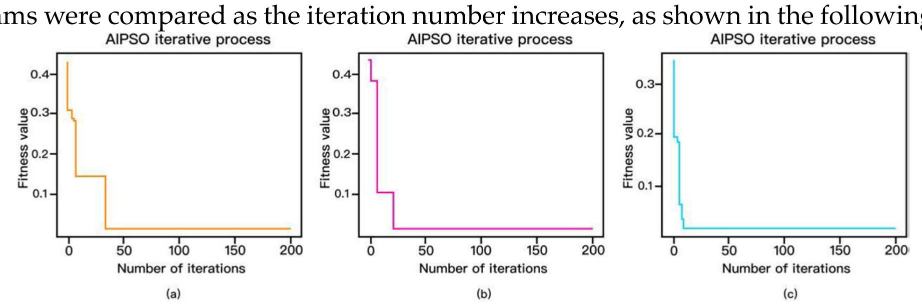


Figure 12 : Iteration curve

The second test function is the Rastrigin function, which is a complex multimodal function. The expression of the function is $f2 = 20 + x_1^2 + x_2^2 - 10(\cos(2\pi x_1) + \cos(2\pi x_2))$.The function image is shown in the following figure.

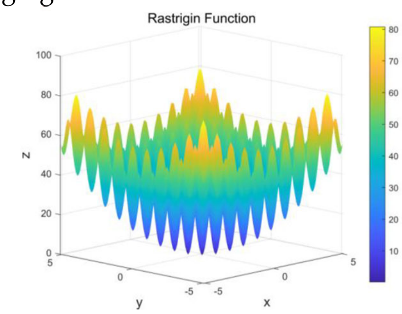


Figure 13 : Rastrigin function

Similarly the process of change in the fitness function curves of the standard PSO, IPSO, and AIPSO methods were compared. This is shown in the figure below.

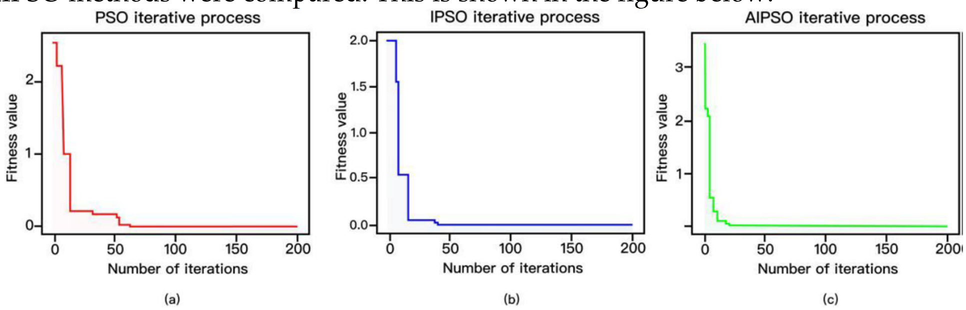


Figure 14 : Iteration curve

All three algorithms can obtain the global optimal value, but the Standard PSO algorithm takes longer to converge and is more likely to be trapped in local optima. Although the IPSO algorithm converges earlier than the Standard PSO algorithm, its convergence effect is not satisfactory. The AIPSO algorithm uses nonlinear adaptive adjustment of the inertial weight and learning factor, converges earlier than the other two algorithms, and has a smoother fitness curve. It can quickly jump out of local optima, indicating better performance than the other two algorithms.

6.5 Experiments on Data in Appendices V and VI

6.5.1 Data Preprocessing

Similarly, the data was preprocessed by removing non-text, segmenting, and removing stop words, and then extracting 10 keywords. The preprocessing process is illustrated in the following table.

Table 2：Keyword Extraction Schematic

Original Comment	"It's hard to believe "Memory of Trees" came out 11 years ago,it has held upwell over the passage of time.It's Enya's last great album before the NewAge/pop of "Amarantine" and "Day without rain." Back in 1995...
Extracted Keywords	['enya', 'album', 'memory', 'trees', '1995', 'saddest', 'pax', 'deorum', 'new', 'age']

6.5.2 Experimental Procedure

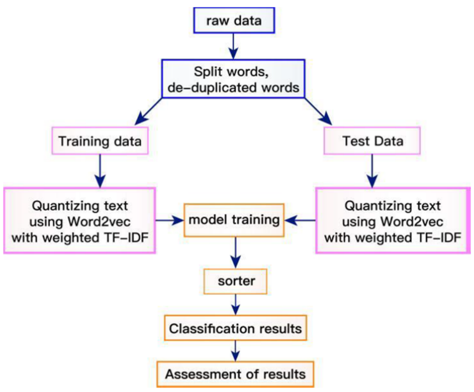


Figure 15: Experimental Procedure

6.5.3 Experimental Results

The precision P, recall R, and F1 score were used as the criteria to evaluate the experimental results. The preprocessed comments were represented as numerical vectors using the mean Word2vec model and the TF-IDF weighted Word2vec model, respectively. These vectors were then used as inputs to the support vector machine (SVM) model for classification. The results are shown in the following table.

Table 3: Algorithmic evaluation

	P/%	R/%	F1/%
Mean Word2vec	69.56	81.93	72.13
TF-IDF Weighted Word2vec	70.45	83.12	74.83

As shown in the table, the experimental results based on TF-IDF weighted Word2vec are slightly better than those based on the mean Word2vec. Different kernel functions of SVM were tested, and the results showed that the RBF kernel was the best choice. Therefore, the PSO algorithm was used to optimize the RBF-SVM. The experimental results showed that the SVM optimized by the improved particle swarm algorithm had better classification performance when compared to other classifiers. The results are summarized in the table.

Table 4: Algorithmic evaluation

Classifier	P/%	R/%	F1/%
LR	68.91	80.53	72.51
NB	6.078	77.49	69.45
SVMM	70.45	83.15	74.83
PSO-SVM	72.44	82.55	77.21
AIPSO-SVM	75.33	85.18	79.27

The SVM model outperforms the Naive Bayes (NB) model by nearly 10 percentage points in precision, and more than 5 percentage points in the other two indicators. The SVM model also performs better than the LR model by more than 2 percentage points. Among all models, the SVM optimized by the improved particle swarm algorithm performs the best in all evaluation criteria. The receiver operating characteristic (ROC) curve was plotted, and the area under the curve (AUC) value was used as a criterion to evaluate the model's effectiveness. The ROC curve of AIPSO + SVM is shown in the figure.

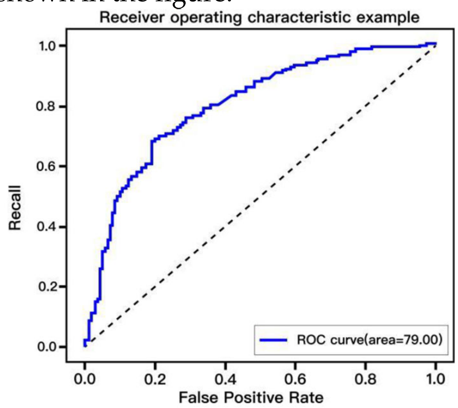


Figure 16: ROC curve

7 COMMENT SCREENING MODEL

7.1 Selection of evaluation indicators

To identify whether a product review is a user review or a machine review, we establish the following evaluation criteria to synthesize the nature of the review as an input to the subsequent T-S fuzzy neural network.

- **Language Quality:** We can differentiate between customer reviews and machine-generated comments by analyzing the language quality of comments, such as grammar, spelling, punctuation, etc.
- **Language Pattern:** We can distinguish between customer reviews and machine-generated comments by comparing the language patterns of comments, such as the vocabulary used, phrases, sentence structures, etc.
- **Comment Length:** We can differentiate between customer reviews and machine-generated comments by comparing the length of comments.
- **Comment Sentiment:** We can distinguish between customer reviews and machine-generated comments by analyzing the sentiment polarity of comments using sentiment analysis techniques.
- **UnixReview Time field:** We can differentiate between customer reviews and machine-generated comments by analyzing the timestamps of comments.
- **Overall field:** We can distinguish between customer reviews and machine-generated comments by using the overall rating given by the reviewer.
- **Helpful field:** We can distinguish between customer reviews and machine-generated comments by using the votes received on the comment, including useful and useless votes.

7.2 T-S fuzzy neural network threshold modeling

We will use a threshold model to classify the influence and evaluate each comment's category. Then we will perform descriptive statistics for each category and draw a conclusion.

7.2.1 Threshold optimization scheme based dynamic programming

In order to build a more detailed classification case, we first consider that the threshold conditions for the planar parameters have changed to some extent, so we need to readjust the planning objective to solve for a truly suitable threshold model. We define the objective function describing the size of the loss as.

$$J(x_N, u_N) = \phi_f(x_n) + \sum_{K=1}^{N-1} L(x_k, u_k) \tag{16}$$

where J denotes the error case in the process of calculating the relationship between the threshold and the product review coefficients, and L denotes a summed case. For the objective function of the system, we reduce it to:

$$J = \frac{1}{2}x_N^T W_N x_N + w_N x_N + \sum_{k=1}^{N-1}\left(\frac{1}{2}x_k^T W_k x_k + R_k u_k\right) + w_k x_k + r_k u_k \tag{17}$$

Which contains the positive definite and positive definite transit cost weighting matrices.V denotes the best forecast model and Q represents the virtual forecast volume of the iterative process.

When conforming to the computational description of the optimal policy, using the optimality principle, the recurrence relation can be expressed as:

$$J = minQ_k^* = min\left(\sum_{k=1}^{N-1}\frac{1}{2}x_k^T W_k x_k + w_k x_k + \frac{1}{2}u_k^t R_K u_k + r_k u_k + V_k^*\big(f(x,u)\big)\right) \tag{18}$$

$$\begin{cases} \dfrac{\tau}{m+n+\tau} + 1 - \varsigma^2 + \beta & i = 0 \\ \dfrac{1}{2(m+n+\tau)} & i = 1, \dots, 2(m+n) \end{cases} \tag{19}$$

$$\tau = \varsigma^2(m+n+\kappa) - (m+n) \tag{20}$$

The λ in the above equation is defined as the influence factor parameter, which often takes a very small value and K follows the principle of greater than or equal to zero. β takes 2 for Gaussian distribution and generally takes other values when in anon-Gaussian situation. The matrix is represented as:

$$\begin{bmatrix} Q_{xx} & Q_{ux} \\ Q_{xu} & Q_{uu} \end{bmatrix} = \left\{ \frac{1}{2\lambda^2} \sum_{i=1}^{m+n} \left\{ \begin{bmatrix} x_i^{S-} \\ u_i^S \end{bmatrix} - \begin{bmatrix} x_k \\ u_k \end{bmatrix} \right\} \left\{ \begin{bmatrix} x_i^{S-} \\ u_i^S \end{bmatrix} - \begin{bmatrix} x_k \\ u_k \end{bmatrix} \right\}^T \right\}^{-1} + \begin{bmatrix} W_K & 0 \\ 0 & 0 \end{bmatrix} \tag{21}$$

Projected onto the sample point used to calculate the equilibrium value, this can be calculated.

$$V_1^s = V_{x|k+1}^T x_i^s \tag{22}$$

The system of linear equations to be solved is

$$\begin{bmatrix} x_1^{s-1} - x_{n+m+1}^{s-} \cdots & x_{n+m}^{s-} + x_{2(n+m)}^{s-} \\ u_1^s - u_{n+m+1}^s \cdots & u_{n+m}^s - u_{2(n+m)}^s \end{bmatrix} \begin{bmatrix} f_x^T V_{x|k+1} \\ f_u^T V_{x|k+1} \end{bmatrix} = \begin{bmatrix} V_1^s - V_{n+m+1}^s \\ \cdots \\ V_{n+m}^s - V_{2(n+m)}^s \end{bmatrix} \tag{23}$$

When the iterative operation reaches moment k, threshold is expected to change to:

$$\delta V_k = -l_k^T Q_{uu} l_k - Q_u^T l_k \tag{24}$$

The forward recursion process stops once the operation is initiated. The feedback correction of the new state control U is brought into the algorithmic model, and a new state X can be calculated. The above process is repeated alternatively until the termination condition is met, leading to convergence to a local optimum.

7.2.2 Solution based on T-S fuzzy neural network modeling

We developed a T-S fuzzy neural network to address the threshold problem and incorporated an evaluation algorithm based on dynamic programming. The network consists of an input layer, a fuzzification layer, a fuzzy rule computation layer, and an output layer. The fuzzification layer uses membership functions to obtain fuzzy membership values, and the fuzzy rule computation layer calculates the fuzzy value ∂ using a fuzzy multiplication formula. The error is calculated during the learning process of the fuzzy neural network.

$$e = \frac{1}{2}(y_d - y_c)^2 \tag{25}$$

In the above equation, y_c is the expected output of the network, $y_d$is the actual output of the network, and e is the error between the expected and actual outputs.

$$p_j^i(k) = p_j^i(k-1) - \alpha \frac{\partial e}{\partial c_j^i} \tag{26}$$

The parameter is amended to

$$c_j^i(k) = c_j^i(k-1) - \beta \frac{\partial e}{\partial c_j^i} \tag{27}$$

For input the $y = [y_2, y_3, \dots, y_n, y_1]$ degree of affiliation y_k of each input variable is calculated according to the fuzzy rule as

$$uA_j^i = \frac{exp\left[-\left(x_j - c_j^i\right)^2\right]}{b_j^i} \quad j = 1,2, \dots, k; i = 1,2, \dots, n \tag{28}$$

Where c and b are the center and width of the affiliation function, respectively; k is the input parameter; and n is the number of fuzzy subsets. The fuzzy operator used in the fuzzy calculation of each affiliation is the concatenation operator, i.e.

$$\omega^i = uA_j^1(x_1) * uA_j^2(x_2) * \dots * uA_j^k(x_k) \tag{29}$$

Based on the results obtained from the fuzzy calculation then the output value y^i of the fuzzy model is calculated as

$$y^i = \frac{\sum_{i=1}^n \omega^i \left(p_0^i + p_1^i x_1 + \cdots + p_k^j x_k\right)}{\sum_{i=1}^n \omega^i} \tag{30}$$

We adjust the control quantity of the transportation algorithm over time steps to ensure it stays within the threshold range. The control quantity of the two algorithms is the same at identical time, and their convergence trajectory and trend are also the same. To ensure accurate forecasting, we use the two-dimensional Fourier transform for more precise information.

$$G(x, y; x_0, y_0) = \frac{1}{4\pi^2} exp(\beta y^0) \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \left\{exp\left[i\left(\xi(x - x_0) + \eta(y - y_0)\right)\right]/L^\wedge\right\} d\xi d\eta \tag{31}$$

The variable G represents the threshold value. Using the model described above, we established rules and a comment evaluation model, and provided specific classification criteria for the comments.

Table 5: threshold distribution

Thresholds	Discriminate
G≥0.50	Machine Reviews
G<0.50	Customer Reviews

By inputting the data from attachments I, II, III, IV, V, and VI into the model and calculating the threshold values, the results of the discrimination are shown in the following table

Table 6: Discriminatory results

Appendix	Sum	Machine	Customer
I	37126	21584	15542
II	20473	10869	9604
III	53258	30487	22771
IV	10261	6358	3903
V	64706	10000	54706
VI	13272	8410	4862

8 MODEL PROS & CONS

In this study, we systematically explored the application of advanced machine learning and natural language processing techniques to analyze and interpret e-commerce reviews. By employing a combination of text analysis, semantic analysis, sentiment analysis, and comment screening models, we achieved a nuanced understanding of consumer feedback. Our research demonstrated that through data preprocessing, visualization, and the use of sophisticated algorithms such as SVM and AIPSO, it is possible to extract significant insights from unstructured text data.

The text analysis model laid the foundation for our study by efficiently preprocessing and visualizing the data, enabling the identification of key terms and overall sentiment within the reviews. Subsequently, the semantic analysis model, employing fastText and TextRank algorithms, facilitated the extraction of meaningful keywords, which proved essential for accurate semantic classification. The incorporation of the SVM classifier, optimized by the AIPSO algorithm, significantly enhanced the model's performance, achieving an accuracy rate of over 75% in sentiment analysis. This optimization indicates the potential for machine learning techniques to refine the accuracy of sentiment analysis in e-commerce settings.

Furthermore, the development of a comment screening model using the T-S fuzzy neural network represents an innovative approach to discerning the authenticity of reviews. This model, supported by seven evaluation criteria, highlights the capability to distinguish between genuine customer feedback and machine-generated content, a critical aspect in the era of information overload.

However, despite these advancements, the study acknowledges certain limitations, such as the reliance on specific machine learning models and the challenge of implementing more complex deep learning techniques due to the technical expertise required. Future research could explore the integration of transformer models, such as BERT, to further enhance the analysis of e-commerce reviews.

REFERENCES

[1] Peng, J. L., Wang, S. H., Lu, Z. Q., et al. (2023). Based on LDA-TF-IDF and Word 2vec document representation. Journal of Zhejiang Textile and Garment Vocational and Technical College, 22(02), 91-96.

[2] Liu, Y., Zhang, H., Xu, J., Ren, Y., & Tang, J. (2023). Improved SVM regression m odel for PSO and its application in temperature prediction. Computer Systems Ap plications, 32(9), 203-210.

[3] Wu, C. (2023). An M-TextRank text summarization model based on Word2Vec. Inf ormation Technology and Informatization, (05), 125-128.

[4] Li, C., Zhao, Y., Yu, J., et al. (2023). Research on key vocabulary extraction metho d for policy text based on word vector and TextRank. Modern Computer, 29(02), 68-72.

[5] Su, Y., Lv, S., Xie, W., et al. (2023). Analysis of risk factors for the development of type 2 diabetes mellitus based on LASSO regression and random forest algorith m. Journal of Environmental Health, 13(07), 485-495.

[6] Lai, F. F., & Cui, G. M. (2022). Grinding particle size prediction based on PSO op timized T-S fuzzy neural network. Sinter Pellet, 47(04), 64-70.

[7] Yuan, Y., & Li, Z. (2021). Solving Black-Scholes equation based on T-S fuzzy neur al network model. Journal of Minjiang College, 42(02), 13-17.

[8] Vishwanathan, S. V. M., & Murty, M. N. (2002, May). SSVM: a simple SVM algor ithm. In Proceedings of the 2002 International Joint Conference on Neural Networ ks. IJCNN'02 (Cat. No. 02CH37290) (Vol. 3, pp. 2393-2398). IEEE.

[9] Wang, H., & Hu, D. (2005, October). Comparison of SVM and LS-SVM for regress ion. In 2005 International conference on neural networks and brain (Vol. 1, pp. 27 9-283). IEEE.

[10] Cherkassky, V., & Ma, Y. (2004). Practical selection of SVM parameters and noise estimation for SVM regression. Neural networks, 17(1), 113-126.

[11] Marini, F., & Walczak, B. (2015). Particle swarm optimization (PSO). A tutorial. C hemometrics and Intelligent Laboratory Systems, 149, 153-165.

[12] Fernández Martínez, J. L., & García Gonzalo, E. (2008). The generalized PSO: a ne w door to PSO evolution. Journal of Artificial Evolution and Applications, 2008.

[13] Moazen, H., Molaei, S., Farzinvash, L., & Sabaei, M. (2023). PSO-ELPM: PSO with elite learning, enhanced parameter updating, and exponential mutation operator. I nformation Sciences, 628, 70-91.