**Machine Learning-Based Research on Prediction of Hot Topics in Online Media**

**Abstract**

**In today's digital era, online media has become a key channel for information dissemination, significantly influencing public opinion, societal discourse, and market trends. The plethora of trending topics on these platforms reflects shifting public interests, necessitating advanced methods for accurate prediction. Traditional content analysis methods, reliant on manual efforts, are no longer sufficient to meet these needs. This study employs machine learning approaches to predict emerging hot topics in online media by analyzing extensive data to identify characteristics of topics likely to garner widespread attention, thereby forecasting their popularity. The experiment includes data cleaning, feature engineering, and model building and evaluation, with parameter tuning through cross-validation and grid search to enhance prediction accuracy. This paper compares various models in terms of performance and time consumption, establishing the effectiveness of the final model in detecting hot topics in online media.**

**Key Words**

**Machine learning, hot topic prediction, data analysis, feature engineering**

1. **Introduction**

In the current digital age, the development of online media has led to unprecedented levels and speeds of topic information generation and dissemination. This vast amount of information not only enriches people's sources of information but also provides valuable resources for analyzing public sentiment and predicting societal trends. However, faced with rapidly emerging data, traditional content analysis techniques are inadequate. These traditional methods often rely on manual operations, which are not only inefficient but also prone to errors when dealing with large-scale data.

This study proposes a machine learning-based approach aimed at using technological means to identify and predict hot topics in online media that may attract widespread attention. These topics typically reflect public focus and societal concerns. Thus, predicting these topics through machine learning models not only enhances the accuracy and efficiency of predictions but also provides real-time data support for decision-makers and media workers, helping them better understand and guide public opinion.

1. **Related Work**

"Hot topic prediction in online media" is an interdisciplinary field combining computer science, data science, and social science. With the rapid development of the internet and social media, the amount of information generated and shared online has dramatically increased, making it more urgent to capture and analyze this information timely. Predicting hot topics in online media helps media and content creators better understand public interest and can have a profound impact on market trends, political events, and even disaster response. The core tasks in this field include, but are not limited to, topic detection, trend analysis, and influence modeling.

To accomplish these tasks, researchers typically use various machine learning models and algorithms, such as classification algorithms, clustering techniques, and deep learning networks, to automate the analysis and handling of large-scale datasets. These models can extract useful features from a large amount of data, identify patterns, and predict future trends. This paper aims to perform relatively fast and accurate predictions of massive online media information through feature engineering and the study of different models, including stacking techniques.

1. **Experiment**

**3.1 Description of the Experimental Dataset**

The original data is a csv file with dimensions of 38,541 \* 60, each example containing 59 features describing the text, with the last 'shares' as the data label. Features include the time difference between the article's publication date and the earliest article date in the dataset, the number of words in the title, the number of words in the article body, the proportion of unique words in the text, and the proportion of non-stop words in the article, among others, as shown in the figure below:

|  |  |
| --- | --- |
| Feature Names | Descriptions |
| timedelta  n\_tokens\_title  n\_tokens\_content  n\_unique\_tokens  n\_non\_stop\_words  n\_non\_stop\_unique\_tokens  num\_hrefs  num\_self\_hrefs  num\_imgs  num\_videos  average\_token\_length  num\_keywords  data\_channel\_is\_lifestyle  data\_channel\_is\_entertainment  data\_channel\_is\_bus  ··· | Time difference between the article's publication date and the earliest article date in the dataset.  Number of words in the title.  Number of words in the article body.  Proportion of unique words in the content.  Proportion of non-stop words in the article.  Proportion of unique non-stop words in the article.  Number of links in the article.  Number of links to other pages within the same website.  Number of images in the article.  Number of videos in the article.  Average length of words in the article.  Number of keywords tagged in the article metadata.  Whether the article belongs to the lifestyle category.  Whether the article belongs to the entertainment category.  Whether the article belongs to the business category.  ··· |

**3.2 Data Preprocessing**

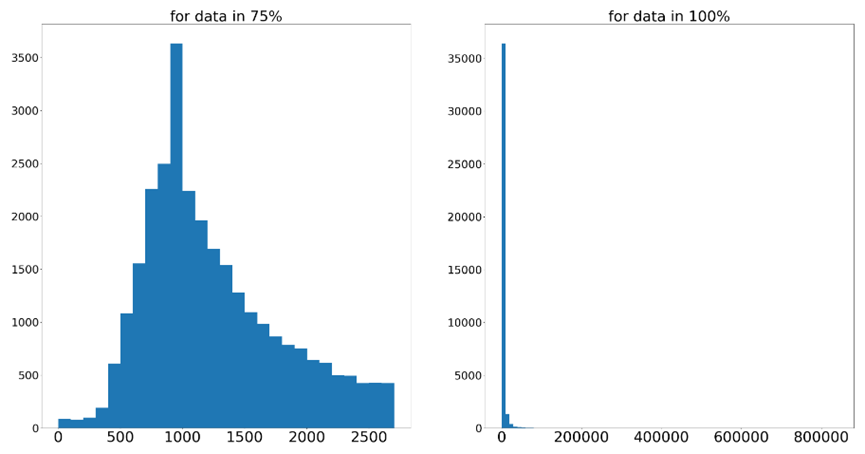
Data preprocessing includes removing irrelevant features, handling missing values, eliminating duplicate data, label transformation, and feature data conversion:

**Removing Irrelevant Features:** The feature 'timedelta' is irrelevant to our task and deemed unimportant for the classification process, hence it is removed.

**Handling Missing Values:** Managing missing values is crucial for ensuring the accuracy and integrity of data analysis and machine learning models. Ensuring data consistency and completeness helps avoid analytical biases and improves the quality of decisions and predictions. Our data quality check revealed zero missing values, indicating good data quality and no need for further steps in handling missing values.

**Removing Duplicate Data:** Removing duplicate data helps maintain the accuracy and reliability of the dataset, enhances data processing efficiency, and avoids biased analytical results, ensuring that analysis and model training outcomes are more aligned with actual conditions. Data duplication checks revealed zero duplicates, eliminating the need for further processing.

**Label Transformation:** Upon analyzing the data, we find that most 'shares' labels fall between 0 and 2500, as shown in the graph below, and 1400 serves as a threshold to classify an article as "popular" or not.

****

**Feature Data Conversion:** Feature data conversion ensures that features of different scales have the same impact during model training, thereby enhancing the performance and convergence speed of machine learning algorithms. This process also helps prevent the model from being overly sensitive to certain features during training, achieving more stable and fair prediction results. Observations indicate that some feature data are significantly larger compared to others, necessitating conversion. We use logarithmic functions to scale these values to a range similar to that of other data. Features requiring conversion are listed in the table below.

|  |
| --- |
| Features |
| n\_tokens\_content  kw\_max\_min  kw\_avg\_min  kw\_min\_max  kw\_max\_max  kw\_avg\_max  kw\_max\_avg  kw\_avg\_avg  self\_reference\_min\_shares  self\_reference\_max\_shares  self\_reference\_avg\_sharess |

**One-hot Encoding:** One-hot encoding converts categorical variables into a numerical format that machine learning models can understand by creating a separate binary variable for each category to avoid misinterpretation due to numerical size. This encoding helps the model correctly interpret unordered categorical data and improves the prediction accuracy. However, it was observed that some features that required one-hot vector encoding were already separated in the original dataset, hence no further processing steps are needed.

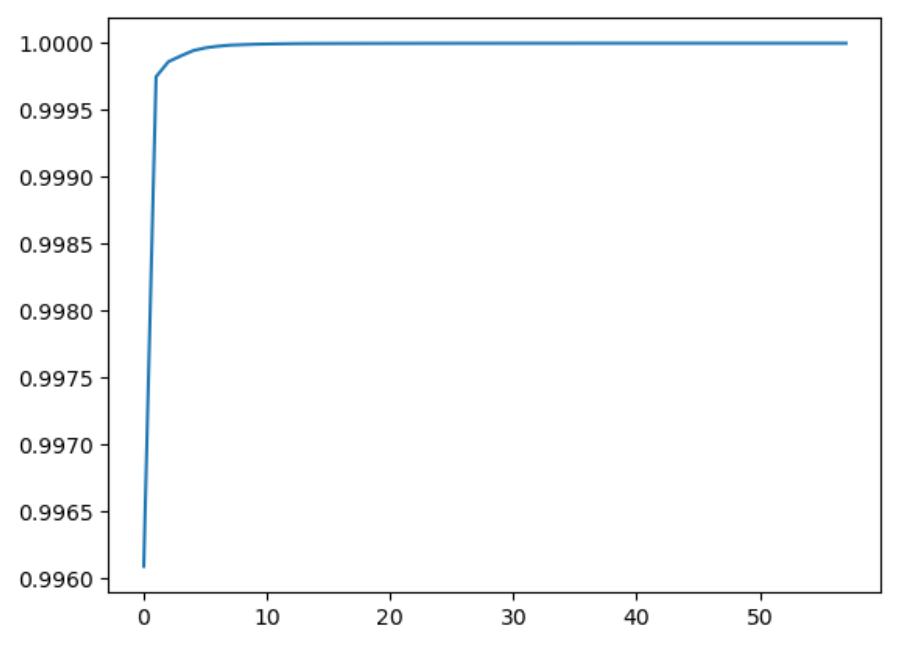
**3.3 Feature Engineering**

Feature engineering is a key step in machine learning to enhance model performance. It involves selecting, modifying, and creating new features to enhance the data's expressiveness. This process helps capture complex relationships within the data, significantly improving model accuracy and effectiveness. Our experiment's feature engineering primarily includes feature generation and selection.

Before feature generation, we separate the 'shares' label and split the dataset and labels into training and test sets in an 8:2 ratio to avoid leaking test set information during subsequent processes.

**1）Feature Generation**

PCA (Principal Component Analysis) is a statistical technique used to reduce data dimensions while retaining as much of the original data's variability as possible. It identifies the main components (directions of maximum variance) in the data and projects the data onto these components, thus reducing dimensions. PCA is particularly useful for removing noise and redundancy from data while highlighting the most important features. Apart from dimension reduction, PCA can also be used for feature generation by merging original and processed data to enhance the subsequent model's predictive capabilities. We first generate new features using PCA and add them to the original data, then use logistic regression to preliminarily assess the quality of the generated features. The following graph shows the explained\_variance\_ratio\_ attribute from sklearn.PCA, which represents the proportion of variance accounted for by each principal component after dimension reduction (summing to 1). A higher proportion indicates a more important principal component.



**2) Feature Selection**

Due to the total number of features exceeding 100 after generation, which negatively impacts the time consumption and accuracy of the classifier, important features are selected using the random forest model. The random forest model includes a feature\_importance\_ attribute, which outputs the importance of each feature. Therefore, we sort the features with an importance greater than 0.01 by their importance and select the top 50 to serve as input features for the final model. The dimensions of the final training and test sets are as shown in the table below:

|  |  |
| --- | --- |
| Description | Dimensions |
| Input Features for Training Set | (30832, 50) |
| Input Features for Testing Set | (7709, 50) |
| Labels for Training Set | (30832,) |
| Labels for Testing Set | (7709,) |

**3.3 Online Media Hot Topic Prediction Models**

1) Logistic Regression Model

Logistic regression is a multivariate analysis method primarily used for binary or multiclass dependent variables (outcomes) and their relationship with one or more independent variables (predictors). This model is a probabilistic, nonlinear regression that applies the Sigmoid function to the output of a traditional linear regression model, mapping it to the interval [0,1] for use in classification. The general form of a linear regression model is:

where are the parameters to be learned, and represents the features.

The Sigmoid function is defined as , which has an S-shaped curve. Under this mapping, the output of the Sigmoid function can be interpreted as a probability, such that when the value of exceeds 0, ，and rapidly approaches 1, allowing classification to be determined by a threshold (usually 0.5).

2) Decision Tree Model

Decision trees are a simple yet powerful method for classification and regression. Each decision tree is a tree structure composed of multiple decision nodes, each involving a test on a feature. The decision tree construction process includes:

* **Feature Selection:** Select the feature that best facilitates classification, based on criteria such as information gain, gain ratio, or Gini index.
* **Tree Splitting:** Split the dataset into two or more subsets based on the selected feature, a process that repeats at each node.
* **Recursive Building:** Recursively construct the decision tree using the same method for each generated subset until a stopping condition is met (such as a certain level of node purity, reaching a preset maximum depth, or the number of samples in a node falling below a threshold).
* **Pruning:** To prevent overfitting, prune the constructed tree by removing nodes that do not contribute statistically significant information to the final prediction.

3) Random Forest Model

Random forests are an ensemble learning model composed of multiple decision trees, mainly used for classification and regression tasks. Its core idea is to improve the prediction accuracy and stability of the model by constructing multiple decision trees and aggregating their results (usually through a voting mechanism or averaging). The random forest construction process includes:

* **Sample Random Sampling:** Perform multiple samplings of the original dataset, potentially including repeated selections of certain samples (bootstrap sampling), to generate different training data for each tree.
* **Random Feature Selection:** At each decision point, not all possible features are considered for node splitting; instead, a subset of features is randomly selected. This introduction of randomness reduces the model's variance and enhances its generalization capability.
* **Decision Tree Construction:** Construct a decision tree using the randomly selected samples and features, continuing until a stopping condition is reached (such as the number of samples under a node falling below a minimum split number or reaching maximum depth).
* **Result Aggregation:** Aggregate the prediction results of all decision trees. In classification tasks, a voting mechanism is usually used; in regression tasks, average prediction results are used.

4) Stacking Model

Stacking (stacked generalization) is a method of combining multiple different prediction models to achieve more accurate predictions through a meta-model that synthesizes the prediction results of different models. Stacking typically involves:

* **Base Model Training:** Train multiple different models (such as random forests, support vector machines, logistic regression, etc.).
* **Creating a New Feature Set:** Use the prediction results of the aforementioned models as a new feature set, derived from predictions on the entire training set or through cross-validation.
* **Meta-Model Training:** Train a meta-model on the new feature set. This model's task is to learn how to optimally combine the predictions of the base models.
* **Final Prediction:** Use the meta-model for the final prediction. The advantage of this method is that it captures different, useful information from the base models and effectively integrates it through the meta-model.

**3.4 Model Evaluation Metrics**

This paper will use accuracy, precision, recall, F1 score, and the area under the receiver operating characteristic curve (AUC) as the main metrics for model evaluation.

* **Accuracy:** A common performance metric in classification tasks, defined as the ratio of correctly classified samples to the total number of samples. However, accuracy is not a good metric when the positive and negative samples are highly imbalanced. For example, in a click-through rate prediction scenario, if the ratio of non-click to click users is 1:10, a model that predicts all users as non-clickers would still achieve a 90% accuracy. To address this, the data preprocessing stage of this paper included balancing the positive and negative samples, making accuracy a valid evaluation metric.
* **Precision:** The ratio of true positive samples among all samples predicted as positive by the model. Precision reflects the accuracy of the model in predicting positive classes.
* **Recall:** The ratio of positive class samples correctly predicted as positive by the model out of all actual positive class samples. Recall measures the model's ability to capture positive class samples.
* **F1 Score:** The harmonic mean of precision and recall, reflecting both the precision and recall of the model. A higher F1 score indicates both high precision and recall.
* **AUC:** An important metric for evaluating the performance of binary classification models, measuring the model's ability to correctly predict positive classes (true positive rate) and its error rate in falsely predicting negative classes as positive (false positive rate). The true positive rate is a vertical measure, and the false positive rate is a horizontal measure. The ROC curve is plotted with the false positive rate on the x-axis and the true positive rate on the y-axis, with the AUC representing the area under this curve. The larger the AUC value, the better the model's performance, with an ideal value of 1, and 0.5 indicating no better than random guessing.

These evaluation metrics together form a comprehensive system for assessing model performance, reflecting the model's effectiveness in various aspects.

1. **Experimental Results**

**4.1 Impact of Different Feature Combinations on Hot Topic Detection**

The selection of appropriate features significantly affects the model's predictive outcomes. This paper classifies the features of the hot topic dataset into original features, original features enhanced by PCA, and features filtered through random forest, and compares the performance of models under different feature sets.

Initially, the model was trained using the original features, then the PCA-enhanced features were incorporated, followed by features filtered through random forest. The experimental model, a commonly used logistic regression in the industry, was evaluated based on accuracy metrics on the training and validation sets. The results, preserved to four significant digits, are displayed in the table below:

|  |  |  |
| --- | --- | --- |
| Feature Set | Training Set ACC | Test Set ACC |
| Original Features | 0.6104 | 0.6055 |
| Original + PCA Processed Features | 0.6139 | 0.6070 |
| Features After Random Forest Selection | 0.6225 | 0.6157 |

The results indicate that using original features alone achieves satisfactory results with an ACC of approximately 0.61. The performance improves upon integrating PCA-processed features. Further enhancement in ACC is noted after applying random forest feature selection.

**4.2 Comparative Analysis of Different Machine Learning Models**

The performance of different machine learning models applied to hot topic prediction is shown below. Logistic regression, decision trees, and random forests were evaluated using grid search to tune hyperparameters, with five-fold cross-validation to find the optimal parameter combinations for the best ACC results.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model Category | Training Set Accuracy | Training Set AUC | Test Set Accuracy | Test Set AUC |
| Logistic Regression | 0.6226 | 0.6637 | 0.6155 | 0.6588 |
| Decision Tree | 0.6567 | 0.7142 | 0.6444 | 0.6944 |
| Random Forest | 0.7752 | 0.8629 | 0.6601 | 0.7283 |

As evident, the performance of the models ranks from highest to lowest as follows: random forest, decision tree, and logistic regression. A random model was also tested for comparison, with an AUC of 0.5116 and ACC of 0.5088, which are lower than any of the other models.

Precision, recall, and F1 score values for the models are also measured:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model Category | Training Set Precision | Training Set Recall | Training Set F1 Score | Test Set Precision | Test Set Recall | Test Set F1 Score |
| Logistic Regression | 0.6284 | 0.7198 | 0.6710 | 0.6171 | 0.7163 | 0.6631 |
| Decision Tree | 0.6793 | 0.6782 | 0.6788 | 0.6600 | 0.6728 | 0.6664 |
| Random Forest | 0.7697 | 0.8270 | 0.7973 | 0.6591 | 0.7375 | 0.6961 |

Differences in runtime for these models were also noted, showing each model's total training time relative to that of logistic regression:

|  |  |  |
| --- | --- | --- |
| Machine Learning Model | Average Training Time (seconds, 10-folds) | Relative Time |
| Logistic Regression | 0.359 | 1.00 |
| Decision Tree | 1.17 | 3.26 |
| Random Forest | 56.36 | 155.98 |

Random forest model takes significantly more time than other models but is still within an acceptable range, while logistic regression and decision tree have training times under two seconds, indicating faster response times.

Furthermore, a stacking approach was used to combine these three models, adding a meta-classifier to output the final prediction results. The performance is shown in the table below:

|  |  |  |
| --- | --- | --- |
| Metric | Results | Improvement Over Currently Best Model |
| Training Set Accuracy | 0.6671 | -0.1081 |
| Training Set AUC | 0.6914 | -0.1715 |
| Test Set Accuracy | 0.6667 | 0.0076 |
| Test Set AUC | 0.7133 | -0.0242 |
| Test Set Precision | 0.6657 | 0.0066 |
| Test Set Recall | 0.7404 | 0.0029 |
| Test Set F1 Score | 0.7011 | 0.0050 |

Although the random forest model performs better on the training set than the Stacking model, indicating high training accuracy and AUC, its performance on the test set is slightly lacking, suggesting potential overfitting. In contrast, the Stacking model, despite its slightly lower performance on the training set, shows slight improvements in accuracy, precision, recall, and F1 score on the test set, suggesting better generalization and adaptability to new data. Considering the model's generalization ability and stability, choosing the Stacking model may be a better decision, especially when dealing with unknown data, as it may offer more reliable predictive performance.

1. **Conclusion**

This paper explores hot topic prediction from two perspectives: appropriate feature selection and model evaluation. Feature data effectively reflect the category and popularity of topics, making feature processing crucial. Initially, logistic regression, decision trees, and random forests are considered based on actual environmental conditions, followed by stacking integration to select the best-performing model. Among these, the random forest model shows the best predictive results but is time-consuming and poses a risk of delayed predictions causing errors. Although the decision tree model has shorter run times, it faces accuracy issues. Logistic regression, although slightly inferior to the random forest and decision tree in performance, has the shortest runtime and strong interpretability, making it a viable option. Combining these three models through stacking improves the overall prediction effectiveness while mitigating their individual drawbacks. However, the overall time taken might impact real-time predictions.