**Problem Statement**

Create a recommender system which recommend a mobile product based on the review.

**Dataset**

Dataset is scraped from multiple mobile phone websites where customer reviewed mobile phones and their accessories. There are 10000 reviews for training. Reviews rated the product between a range of 1 to 10.

Table 1. Review and number of users give that review

|  |  |
| --- | --- |
| Review | Number of Users |
| 10 | 56161 |
| 8 | 18553 |
| 2 | 12801 |
| 6 | 7313 |
| 4 | 4634 |
| 9 | 327 |
| 7 | 61 |
| 9.3 | 48 |
| 8.6 | 25 |
| 5 | 20 |
| 1 | 16 |
| 3 | 12 |
| 7.7 | 4 |
| 7.9 | 3 |
| 8.3 | 3 |
| 8.8 | 3 |
| 7.1 | 2 |
| 8.4 | 2 |
| 7.4 | 2 |
| 7.5 | 2 |
| 8.2 | 2 |
| 7.8 | 2 |
| 8.1 | 1 |
| 7.3 | 1 |
| 9.2 | 1 |
| 7.2 | 1 |

In the next step, we have rounded the review to the closed integer value. Pie graph shows the review occurrence. For the classification purpose, we treat the data labels as binary data labels. All the review with a value greater or equal to 8 is treated as 1. It means that the reviewer recommends that product. If a review is less than 8, it means the reviewer did not recommend that product, and we labelled it as zero. Out of 10,000 reviews around 7500 (75%) are positive (user recommended) and 2500 are negative (the user rejected the product).

The table shows the top 20 most occurred words after preprocessing steps as followed

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | Word | Occurrence | # | Word | Occurrence |
| 0 | phone | 82355 | 10 | price | 11753 |
| 1 | good | 30553 | 11 | love | 11138 |
| 2 | use | 19578 | 12 | screen | 10626 |
| 3 | great | 18540 | 13 | best | 10549 |
| 4 | camera | 16562 | 14 | like | 9901 |
| 5 | battery | 16141 | 15 | product | 9113 |
| 6 | get | 13991 | 16 | quality | 8837 |
| 7 | buy | 13657 | 17 | iphone | 8374 |
| 8 | work | 13012 | 18 | mobile | 8353 |
| 9 | one | 12652 | 19 | nice | 7814 |

**Preprocessing**

Preprocessing help to remove unwanted noisy stuff from the data. Such as digits punctuations as they don’t play any role in sentiment analysis. Following steps are performed while doing preprocessing

* All the numbers are removed. Only alphabets are kept
* Words are tokenized, and any word length less than two is removed. This help to remove punctuations and other meaningless words
* All the words are converted to lower character
* Stop words are removed. Words which occurs very frequently and do not add meaning is known as stop words. Here is the list of words removed

‘ourselves’, ‘hers’, ‘between’, ‘yourself’, ‘but’, ‘again’, ‘there’, ‘about’, ‘once’, ‘during’, ‘out’, ‘very’, ‘having’, ‘with’, ‘they’, ‘own’, ‘an’, ‘be’, ‘some’, ‘for’, ‘do’, ‘its’, ‘yours’, ‘such’, ‘into’, ‘of’, ‘most’, ‘itself’, ‘other’, ‘off’, ‘is’, ‘s’, ‘am’, ‘or’, ‘who’, ‘as’, ‘from’, ‘him’, ‘each’, ‘the’, ‘themselves’, ‘until’, ‘below’, ‘are’, ‘we’, ‘these’, ‘your’, ‘his’, ‘through’, ‘don’, ‘nor’, ‘me’, ‘were’, ‘her’, ‘more’, ‘himself’, ‘this’, ‘down’, ‘should’, ‘our’, ‘their’, ‘while’, ‘above’, ‘both’, ‘up’, ‘to’, ‘ours’, ‘had’, ‘she’, ‘all’, ‘no’, ‘when’, ‘at’, ‘any’, ‘before’, ‘them’, ‘same’, ‘and’, ‘been’, ‘have’, ‘in’, ‘will’, ‘on’, ‘does’, ‘yourselves’, ‘then’, ‘that’, ‘because’, ‘what’, ‘over’, ‘why’, ‘so’, ‘can’, ‘did’, ‘not’, ‘now’, ‘under’, ‘he’, ‘you’, ‘herself’, ‘has’, ‘just’, ‘where’, ‘too’, ‘only’, ‘myself’, ‘which’, ‘those’, ‘i’, ‘after’, ‘few’, ‘whom’, ‘t’, ‘being’, ‘if’, ‘theirs’, ‘my’, ‘against’, ‘a’, ‘by’, ‘doing’, ‘it’, ‘how’, ‘further’, ‘was’, ‘here’, ‘than’

* Lemmatization is performed. In this technique, words are converted to their base form such reading to read.

**Feature Extraction**

N-grams features are extracted. N-gram is defined as a combination of words. When the value of N is 1, it means the single word is considered a feature. When N=2, two words combined is considered as features. Similarly, three words combined are used as features when N=3.

*Example: This is a sentence*

*Unigram: This, is, a, sentence*

*Bigram: This is, is a, a sentence*

*Trigram: This is a, is a sentence*

For feature extraction, we used the TF-IDF technique. It’s a technique that tells how relevant a word is to a document in a collection of documents. Here word can be unigram, bigram or trigram.

We have used unigram, which occurs in more than ten documents at least. Unigram and bigram which occurs in 20 documents. Unigram, bigram and trigram which occurs in more than 30 documents.

**Classification**

Three models are used for classification purpose.  In machine learning, classification is a supervised learning technique which categorizes objects into their respective class. For example we have hundreds and thousands of pen and pencil based on their characteristics we can group them into their respective class. Following three models are used for classification

* Naïve Bayes
* Support Vector Machine
* Adaboost

Naïve Bayes is considered as a generative classifier, support vector machine falls in the category of discriminative classifiers whereas Adaboost is ensemble classifier created from a group of decision trees.

For the evaluation of our model, we have used precision, recall, f1 score and accuracy. These metrics are derived from a table called a confusion matrix.

|  |  |  |
| --- | --- | --- |
|  | Recommended | Not Recommended |
| Recommended | TP | FN |
| Not Recommended | FP | TN |

* True Positive (TP): Actual is positive, and is predicted to be positive.
* False Negative (FN): Actual is positive, but is predicted negative.
* True Negative (TN): Actual is negative, and is predicted to be negative.
* False Positive (FP): Actual is negative, but is predicted positive.

Accuracy is ratio of correct predictions to all predictions. Accuracy is the main criteria we used to compared the models.

F1 score is the harmonic mean of recall and precision.

**Cross Validation**

Cross validation is a technique in which data is divided into K segments, where K can be any number from 2 to onward. Mostly the value of K is set to 5. K-1 segments are used for training purpose and 1 segment is used for testing. Then Another K-1 segment is used for training and tested on left segment. This process is repeated until all segments are gone through testing one by one.

**Results.**

In following tables A: Accuracy, P: Precision, R: Recall and F: F1-score. All the models accuracies, precision , recall and f1 score are shown. As the dataset is imbalance so we use macro average instead of micro, or weighted average. A macro-average compute all matrices independently for each class and treat all classed equally instead of treating them according to their occurrence, which make it suitable for imbalance data. It should be noted that macro average is not used in accuracy.

**Naïve Bayes**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  | Unigram | | | | Bigram | | | | Trigram | | | | |  | A | P | R | F | A | P | R | F | A | P | R | F | | 0 | 0.85 | 0.84 | 0.73 | 0.76 | 0.85 | 0.84 | 0.74 | 0.77 | 0.85 | 0.84 | 0.74 | 0.77 | | 1 | 0.85 | 0.84 | 0.73 | 0.76 | 0.86 | 0.84 | 0.75 | 0.78 | 0.86 | 0.84 | 0.75 | 0.78 | | 2 | 0.85 | 0.84 | 0.73 | 0.76 | 0.86 | 0.84 | 0.75 | 0.78 | 0.86 | 0.84 | 0.75 | 0.78 | | 3 | 0.85 | 0.85 | 0.73 | 0.76 | 0.85 | 0.84 | 0.74 | 0.78 | 0.85 | 0.84 | 0.75 | 0.78 | | 4 | 0.85 | 0.84 | 0.73 | 0.76 | 0.85 | 0.84 | 0.75 | 0.78 | 0.85 | 0.84 | 0.75 | 0.78 | | Average | **0.85** | **0.84** | 0.73 | 0.76 | **0.85** | **0.84** | 0.75 | **0.78** | **0.85** | **0.84** | **0.75** | **0.78** | |  |  |  |  |  |

Using Naïve Bayes classifier highest average F1 score is obtained in the bigram and trigram model where 78% score is achieved. Accuracy is the same in all feature extraction technique, and it tends to be 85%. Highest precision is 84% in all three feature extractions techniques. The recall is 75% in trigram and bigram, whereas in unigram it is 73%.

**Support Vector Machine Classifier**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Unigram | | | | Bigram | | | | Trigram | | | |
|  | A | P | R | F | A | P | R | F | A | P | R | F |
| 0 | 0.87 | 0.84 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.81 | 0.87 | 0.84 | 0.8 | 0.81 |
| 1 | 0.87 | 0.84 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.82 |
| 2 | 0.87 | 0.84 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.82 |
| 3 | 0.87 | 0.85 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.82 |
| 4 | 0.87 | 0.84 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.82 |
| Average | 0.87 | 0.84 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.82 | 0.87 | 0.84 | 0.8 | 0.82 |

F1 score is the same, i.e. 82% in unigram, bigram and trigram models. Accuracy is 87% in all three models. Same is with precision and recall, which is 84% and 80% respectively. F1-score achieved using support vector machine classifier is 4% better than Naïve Bayes. Accuracy is 2% better than Naïve Bayes. Precision is same in Naïve Bayes and support vector machine. So increase in recall tends to increase f1 score.

**Adaboost Classifier**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Unigram | | | | Bigram | | | | Trigram | | | |
|  | A | P | R | F | A | P | R | F | A | P | R | F |
| 0 | 0.82 | 0.78 | 0.69 | 0.71 | 0.82 | 0.78 | 0.69 | 0.71 | 0.82 | 0.79 | 0.69 | 0.72 |
| 1 | 0.82 | 0.79 | 0.69 | 0.72 | 0.83 | 0.79 | 0.72 | 0.74 | 0.82 | 0.79 | 0.7 | 0.73 |
| 2 | 0.82 | 0.79 | 0.7 | 0.72 | 0.83 | 0.79 | 0.7 | 0.73 | 0.82 | 0.79 | 0.7 | 0.72 |
| 3 | 0.83 | 0.8 | 0.7 | 0.73 | 0.83 | 0.8 | 0.7 | 0.73 | 0.83 | 0.8 | 0.7 | 0.73 |
| 4 | 0.82 | 0.78 | 0.69 | 0.72 | 0.82 | 0.79 | 0.7 | 0.72 | 0.82 | 0.78 | 0.7 | 0.72 |
| Average | 0.82 | 0.79 | 0.69 | 0.72 | 0.82 | 0.79 | 0.7 | 0.73 | 0.82 | 0.79 | 0.7 | 0.73 |

Compared to other two classifier adaboost does not perform well. With unigram features an accuracy of 82% is achieved. It is 85% in Naïve Bayes and 87% in support vector machine classifier. F1-score of 72% is achieved with unigram features. Using bigram adaboost model F1score increased 1% as compared to unigram model and remain same with trigram model.

**Conclusion**

Support vector machine outperformed naïve bayes and adaboost classifier in all three feature extraction techniques such as unigram, bigram and trigram. Following graph shows comparison of model based upon F1-score.