



VIT[®]

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering



A COMPARATIVE STUDY OF CLASSIFIERS ON SENTIMENT ANALYSIS FOR AMAZON PRODUCT REVIEWS

ABIRAMI
20MCA1024

Research Supervisor
Dr. M. Sivabalakrishnan

RBL-SET CONFERENCE -2 PPT APPROVAL Inbox x



ABIRAMI 20MCA1024 Respected Sir, I have attached the ppt for the set conference, kindly let m...  Tue, Apr 6, 4:23 PM (17 hours ago) 



Sivabalakrishnan M

to me ▼

Tue, Apr 6, 7:14 PM (14 hours ago)



Approved

Regards

Sivabalakrishnan

From: ABIRAMI 20MCA1024 <abirami.2020@vitstudent.ac.in>

Sent: Tuesday, April 6, 2021 4:23:11 PM

To: Sivabalakrishnan M

Subject: RBL-SET CONFERENCE -2 PPT APPROVAL



Outline:

- **Introduction**
 - **Objective**
 - **Proposed Solution**
 - **Dataset & Methodology**
 - **Data Preprocessing**
 - **Simple Word-Cloud**
 - **Summary of review-1**
 - **Implementation of Random Forest , Logistic Regression model**
 - **Results**
-

Introduction

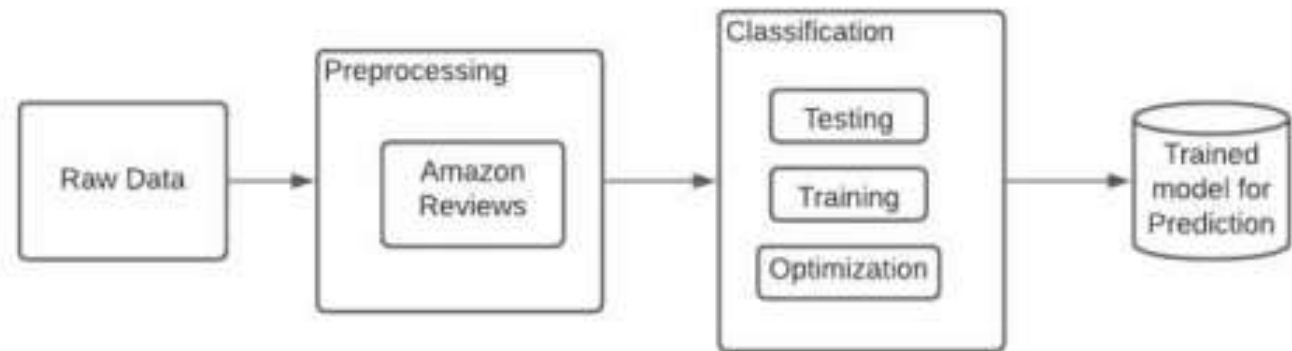
- *NLP is a broader term that helps in interactions between human language and computer ,especially in processing and analyzing large amounts of natural language data.*
- **Sentiment analysis /opinion mining /emotion AI:** refers to the use of natural language processing, text analysis, computational linguistics, and to systematically identify, extract, quantify, and study affective states and subjective information.
- Sentiment analysis is applied to the customer reviews or survey responses in order to understand the targeted audience more.

Objective

- *The objective is to make Sentiment Analysis on the customer reviews and to provide the best classification model to derive an accurate result to help the organization grow .*

Proposed work:

- Problem statement: Appropriate selection of a classification model among the observed models and proper processing of given data to show a higher accuracy than the previous papers claims.
- *My paper provides a comparative study between the classification models such as SVM, Random Forest, Logistic Regression, and a hybrid / model to determine the polarity of a product .*



Dataset:

- Amazon Review Dataset of a smart phones were taken.
- Dataset contains One-plus and Redmi reviews.
- Dataset has almost 30,000 instances with 20 columns.

Methodology

- Software: Anaconda.
- Language: Python-jupyter notebook
- *Key words*: NLTK,Textblob,sklearn-svm

Data preprocessing

- Preprocessing involved the following processes:
 - ☐ removing unnecessary columns
 - ☐ extracting rating values to numerical values
 - ☐ filling the missing values
 - ☐ removing numbers
 - ☐ trimming lower case
 - ☐ word tokenization
 - ☐ dealing with negation
 - ☐ removing punctuation
 - ☐ removing stop words
 - ☐ word stemming
 - ☐ lemmatization

Output:

```
Out[94]: 0      yea pre-ord juli got august packag nice withou...
          1      got deliv yesterday use hour tell first mid ra...
          2                                     amaz phone
          3                                     brilliant
          4      skeptic chang one plu nord still process power...
          ...
          30607      qualiti phone great perspect expect high
          30608                                     recommend
          30609      redmi amazon engag worst market tactic flash s...
          30610      face display retent problem use display minut ...
          30611      front camera qualiti wors compar note pro when...
          Name: review_text, Length: 30612, dtype: object
```


Snippet of preprocessing

```
# convert text to lower case
review_text = reviews["reviewText"].str.lower()
print("original: ",review_text[7],"\n")

# remove numbers
review_text = review_text.apply(remove_number)
print("numbers: ",review_text[7],"\n")

# words Tokenization
review_text = review_text.apply(word_tokenize)
print("tokenization: ",review_text[7],"\n")

# deal with negation
review_text = review_text.apply(n_apostrophe_t_handler)
print("negation: ",review_text[7],"\n")

# remove punctuation
punctuations = list(string.punctuation)
review_text = review_text.apply(lambda x:
    [i.strip("".join(punctuations)) for i in x if i not in punctuations])
print("punctuation: ", review_text[7],"\n")

# remove stop words
stop_words=set(stopwords.words("english"))
review_text = review_text.apply(lambda x:
    [item for item in x if item not in stop_words])
print("stop words: ",review_text[7],"\n")

# word stemming
stemmer = PorterStemmer()
review_text = review_text.apply(lambda x: [stemmer.stem(y) for y in x])
print("stemming: ",review_text[7],"\n")

# Lemmatizer
lemmatizer = WordNetLemmatizer()
review_text = review_text.apply(lambda x: [lemmatizer.lemmatize(y) for y in x])
print("lemmatizer: ",review_text[7],"\n")
```

Sentiment analysis-Textblob

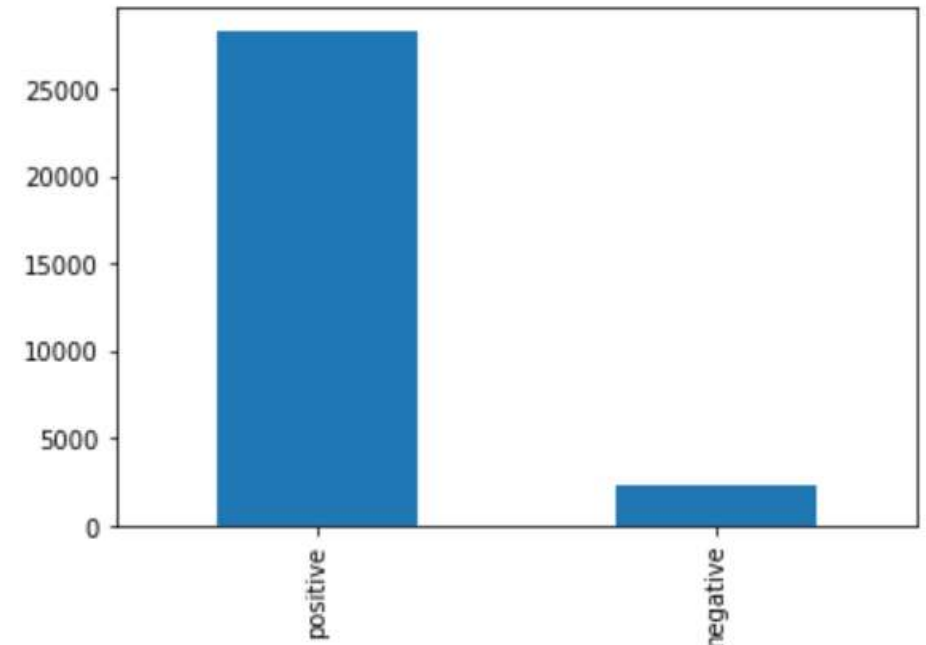
- *TextBlob* is a Python library for processing textual data.
- Provides a simple API for natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.
- Data is split into training and testing data.
- For instance : 30% of the total data was used as a testing data in my module.

```
In [111]: # Sentiment
raw_df.groupby('sentiment').review_text.count()
```

```
Out[111]: sentiment
negative      2339
positive     28273
Name: review_text, dtype: int64
```

```
In [71]: data['sentiment'].value_counts().plot(kind = 'bar')
```

```
Out[71]: <AxesSubplot:>
```



Simple Word-Cloud for the cleaned reviews:

```
In [64]: import warnings
warnings.filterwarnings("ignore")
```

```
In [65]: from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

# Get stopwords from wordcloud library
stopwords = set(STOPWORDS)
```

```
In [66]: # join all reviews
text = " ".join(review for review in data['review_clean_str'])

# Generate the image
wordcloud = WordCloud(stopwords=stopwords, background_color="white", max_words=100, min_word_length=5).generate(text)

# visualize the image
fig=plt.figure(figsize=(15, 8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title('Total Reviews Word Cloud')
plt.show()
```



Summary from Review-1

- SVM model was implemented and an accuracy of 98.56%

	precision	recall	f1-score	support
negative	0.95	0.86	0.90	691
positive	0.99	1.00	0.99	8493
accuracy			0.99	9184
macro avg	0.97	0.93	0.95	9184
weighted avg	0.99	0.99	0.99	9184

```
print(metrics.accuracy_score(y_test,predictions))
```

```
0.9856271777003485
```

Implementation of Random Forest model:

- ❖ Random forest algorithm(SML) creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting.
- ❖ Ensemble method which is better than a single decision tree as-
 - ❖ it reduces the overfitting by averaging the result.

Using subset of data

Out[257...

	review_rating	sentiment	reviewed_at	review_clean_str	dummy_y
22185	3	positive	2019-11-06	ok	1
14885	4	positive	2019-11-06	last year use mi phone nice phone budget price	1
9470	5	positive	2019-11-06	love phone purchas bank discount gb gb blue va...	1
9469	5	positive	2019-11-06	febula perform redmi note love 🥰	1
14348	5	positive	2019-11-06	camera bettari sound speed smooth superb valu ...	1
10064	5	positive	2019-11-06	exlent	1
13028	5	positive	2019-11-06	fantast phone	1
9578	5	positive	2019-11-06	best k	1
10550	5	positive	2019-11-06	quick servic genuin product	1
22003	3	positive	2019-11-06	phone googl assist work ok googl without touch...	1

Results:

RANDOM FOREST

```
In [60]: from sklearn.feature_extraction.text import CountVectorizer
X = data['review_clean_str']
y = data['sentiment']

In [61]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=101)

In [62]: from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_extraction.text import TfidfTransformer
pipeline = Pipeline([
    ('bow', CountVectorizer(stop_words='english', max_features=10000)), # strings to token integer counts
    ('tfidf', TfidfTransformer()), # integer counts to weighted TF-IDF scores
    ('classifier', RandomForestClassifier()), # train on TF-IDF vectors w/ Random Forest classifier
])

In [63]: pipeline.fit(X_train, y_train)

Out[63]: Pipeline(steps=[('bow',
                           CountVectorizer(max_features=10000, stop_words='english')),
                          ('tfidf', TfidfTransformer()),
                          ('classifier', RandomForestClassifier())])

In [64]: predictions = pipeline.predict(X_test)

In [65]: print(classification_report(y_test, predictions))
print('\n')
print(confusion_matrix(y_test, predictions))
print("Accuracy is", accuracy_score(y_test, predictions))
```

	precision	recall	f1-score	support
negative	0.92	0.74	0.82	672
positive	0.98	0.99	0.99	8512
accuracy			0.98	9184
macro avg	0.95	0.87	0.90	9184
weighted avg	0.98	0.98	0.98	9184

```
[[ 497  175]
 [  43 8469]]
Accuracy is 0.9762630662020906
```


Implementation of Logistic Regression model:

- ❖ It's a classification algorithm(SML), that is used where the response variable is categorical. The idea of **Logistic Regression** is to find a relationship between features and probability of particular outcome.
- ❖ To classify the observations using different types of data and can easily determine the most effective variables used for the classification.

Using subset of data

Out[257...

	review_rating	sentiment	reviewed_at	review_clean_str	dummy_y
22185	3	positive	2019-11-06	ok	1
14885	4	positive	2019-11-06	last year use mi phone nice phone budget price	1
9470	5	positive	2019-11-06	love phone purchas bank discount gb gb blue va...	1
9469	5	positive	2019-11-06	febula perform redmi note love 🥰	1
14348	5	positive	2019-11-06	camera bettari sound speed smooth superb valu ...	1
10064	5	positive	2019-11-06	exlent	1
13028	5	positive	2019-11-06	fantast phone	1
9578	5	positive	2019-11-06	best k	1
10550	5	positive	2019-11-06	quick servic genuin product	1
22003	3	positive	2019-11-06	phone googl assist work ok googl without touch...	1

Results:

LOGISTIC REGRESSION

```
In [66]: from sklearn.linear_model import LogisticRegression
```

```
X = data['review_clean_str']
```

```
y = data['sentiment']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=101)
```

```
In [67]: pipeline = Pipeline([
    ('bow', CountVectorizer(stop_words='english', max_features=10000)), # strings to token integer counts
    ('tfidf', TfidfTransformer()), # integer counts to weighted TF-IDF scores
    ('classifier', LogisticRegression()), # train on TF-IDF vectors w/ Logistic Regression classifier
])
```

```
In [68]: pipeline.fit(X_train, y_train)
```

```
Out[68]: Pipeline(steps=[('bow',
    CountVectorizer(max_features=10000, stop_words='english')),
    ('tfidf', TfidfTransformer()),
    ('classifier', LogisticRegression())])
```

```
In [69]: predictions = pipeline.predict(X_test)
```

```
In [70]: print(classification_report(y_test, predictions))
print('\n')
print(confusion_matrix(y_test, predictions))
print("Accuracy is", accuracy_score(y_test, predictions))
```

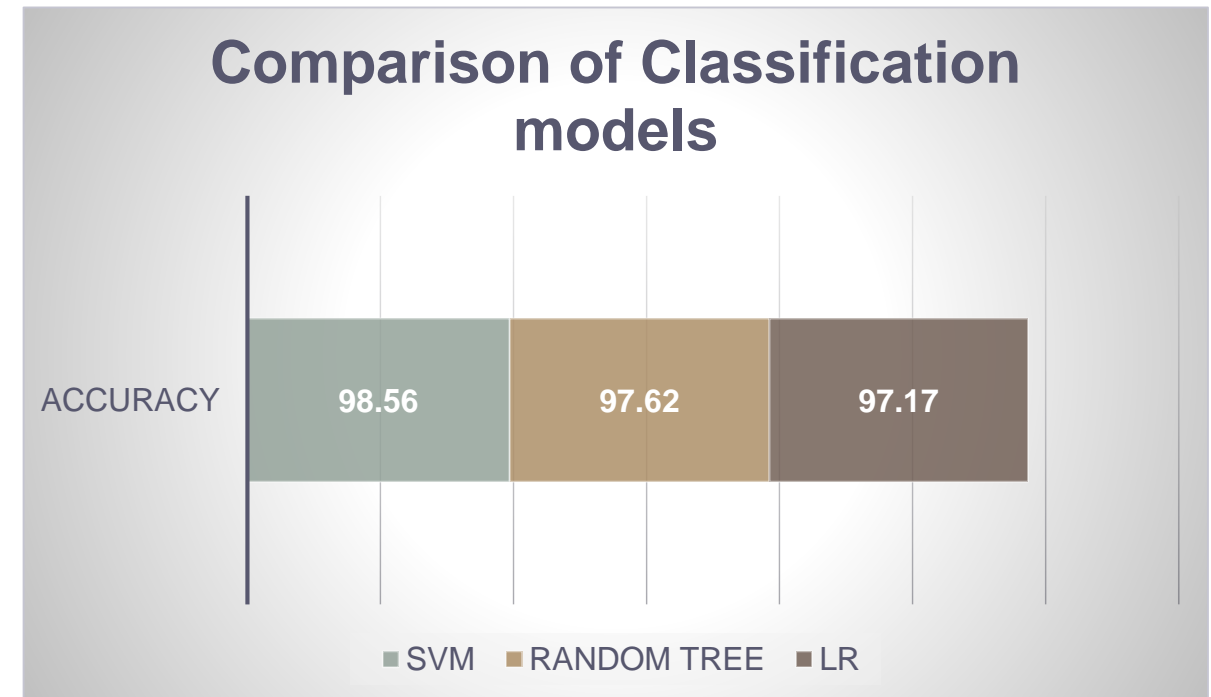
	precision	recall	f1-score	support
negative	0.93	0.66	0.77	672
positive	0.97	1.00	0.98	8512
accuracy			0.97	9184
macro avg	0.95	0.83	0.88	9184
weighted avg	0.97	0.97	0.97	9184

```
[[ 446  226]
 [  33 8479]]
```

Accuracy is 0.9717987804878049

Comparison Implemented models:

S.NO	SVM	RANDOM TREE	LR
Accuracy	98.56%	97.62%	97.17%



Future Implementation

- Inclusion a hybrid model or another model
 - To satisfy the novelty criteria and
 - To provide a better Accuracy score from the classifier produced.

References

- <https://ieeexplore.ieee.org/document/8321910>
- <https://ieeexplore.ieee.org/document/8603585>
- <https://ieeexplore.ieee.org/document/8850982>
- <https://ieeexplore.ieee.org/document/8389689>
- <https://doi.org/10.1109/ICNSC.2019.8743331>
- <https://doi.org/10.1109/CSITSS.2017.8447660>
- <https://doi.org/10.1145/3340997.3341012>
- <https://doi.org/10.1109/IC3A48958.2020.233300>
- <https://doi.org/10.1109/ICCIC.2016.7919584>
- <https://doi.org/10.1016/j.procs.2018.01.150>
- Other references:
- Business reviews classification using sentiment analysis-<https://doi.org/10.1109/SYNASC.2015.46>
- Sentiment Analysis in TripAdvisor-<https://doi.org/10.1109/MIS.2017.3121555>
- Using Objective Words in SentiWordNet to Improve Word-ofMouth Sentiment Classification-<https://doi.org/10.1109/MIS.2013.1>

Thank You