Sentiment Analysis

Student Name : Kurva Ravi

Roll No. : B20CS030

Description

Sentimental analysis helps many brands to know what customer id actually feeling about their brand and it is basic way of classifying incoming text to positive, negative or neutral. One can predict sentiment using my model created in Major project. Sentiment analysis is basically mining of text to understand their customers and services they provide. It helps business to bring changes and developments in their service. With recent advancement in deep learning algorithms the ability to understand text is considerably increased. Sentiment analysis tells customer’s view and what are the services they care about. One can use Sentiment classification to classify review of any product, ecommerce website, social media etc..,

Twitter sentiment analysis is ongoing project which having a large data about 1600000 tweets. We have to train our model using right algorithm to get high accuracy score. As twitter will have large data it is necessary to classify data and we need high classification accuracy score for right classification. Therefore we need to develop an Automated Machine Learning Sentiment analysis model in order to compute customer views and perception. Here I developed a web application which detects whether a given review is positive or negative.

Objective of The Project

#Scrapping product reviews on various websites featuring various products.

#Analyze and categorize review data.

#Analyze sentiment on dataset from document level.

#Categorization or classification of opinion sentiment into-
 Positive
 Negative

Preprocessing

In our project given data file is a tab separated file. So, lets create Data Frame for this file using pandas. We can see that data is having 2 columns and 1000 rows with no null values. We can see any review we would like to see and whether that review is positive or negative from ‘liked’ column. There are 500 positive and 500 negative reviews in our data set that can seen from the graph of “liked” column.

This is a vital part of training the dataset. Here Words present in the file are accessed both as a solo word and also as pair of words.

Tfidf Vectorizer

I will use a Tfidf vectorizer to vectorize the text data in the review column (training feature for this project) and then use three different classification models from scikit-learn models. After that, to evaluate the model on this dataset find out the accuracy, confusion matrix, true positive rates, and true negative rates. Here are the steps.

1. The first step is to split the dataset into training sets and testing sets.
2. Vectorize the input feature that is out review column (both training and testing data)
3. import the model from scikit learn library.
4. Find the accuracy score
5. find the true positive and true negative rates.

I will repeat this same process for three different classifiers now. The classifiers that will be used here are Support Vector Machine, Naive bayes Classifier and pipeline. I will summarize the results towards the end of this article.

Training Data

The main chunk of code that does the whole
evaluation of sentimental analysis based on the preprocessed data is a part of this.
The following are the steps followed:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=0)

from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer=TfidfVectorizer(stop_words='english')
x_train_vect=vectorizer.fit_transform(x_train)
x_test_vect=vectorizer.transform(x_test)
```

#Naives Bayes, SVC, pipeline are applied on the dataset for evaluation of sentiments.

#A review like sentence is taken as input on the console and if positive the console gives 1
as output and 0 for negative input.

Sentiment Classification Algorithm

Naive Bayesian Classification :

The Naïve Bayesian classifier works as follows: Suppose that there exist a set of training data, D, in which each tuple is represented by an n-dimensional feature vector, X=x 1,x 2,..,x n , indicating n measurements made on the tuple from n attributes or
features. Assume that there are m classes, C 1,C 2,...,C m . Given a tuple X, the classifier will
predict that X belongs to C i if and only if: P(C i |X)>P(C j |X),
where i,j∈[1,m]a n d i≠j. P(C i |X) is computed as:

$$P(C_i|X) = \prod_{k=1}^{n} P(x_k|C_i)$$

Support Vector Machine :

Support vector machine (SVM) is a method for the classification of both linear and nonlinear data. If the data is linearly separable, the SVM searches for the linear optimal separating hyperplane (the linear kernel), which is a decision boundary that separates data
of one class from another. Mathematically, a separating hyper plane can be written as: W·X+b=0, where W is a weight vector and W=w1,w2,...,w n. X is a training tuple. b is a
scalar. In order to optimize the hyperplane, the problem essentially transforms to the minimization of ∥W∥, which is eventually computed as:

$$\sum_{i=1}^{n} \alpha_i y_i x_i,$$

where αi are numeric parameters, and yi are labels based on support
vectors, Xi .
That is: if yi =1 then

$$\sum_{i=1}^{n} w_i x_i \geq 1;$$

if y i =−1 then

$$\sum_{i=1}^{n} w_i x_i \geq -1.$$

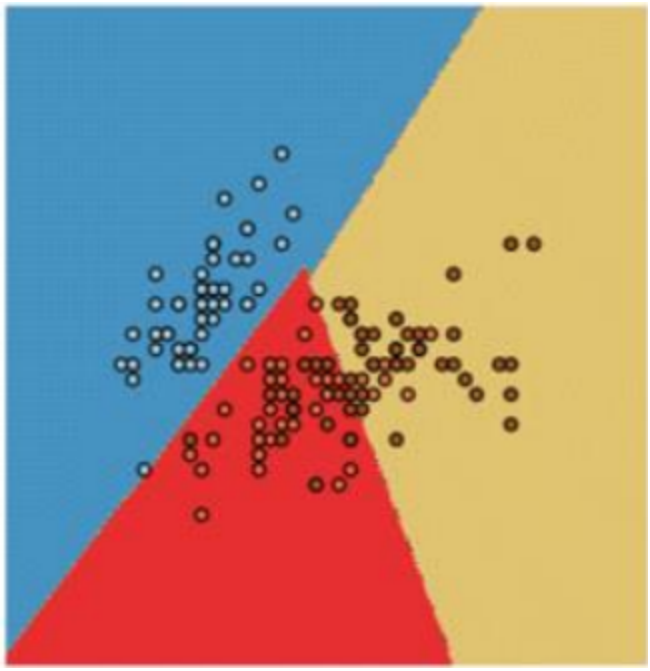
If the data is linearly inseparable, the SVM uses nonlinear mapping to transform the data
into a higher dimension. It then solve the problem by finding a linear hyperplane.
Functions

to perform such transformations are called kernel functions. The kernel function selected for
our experiment is the Gaussian Radial Basis Function (RBF):

$$K(X_i, X_j) = e^{-\gamma \|X_i - X_j\|^2/2}$$

where Xi are support vectors, X j are testing tuples, and γ is a free parameter that uses the
default value from scikit-learn in our experiment. Figure shows a classification example of
SVM based on the linear kernel and the RBF kernel on the next page

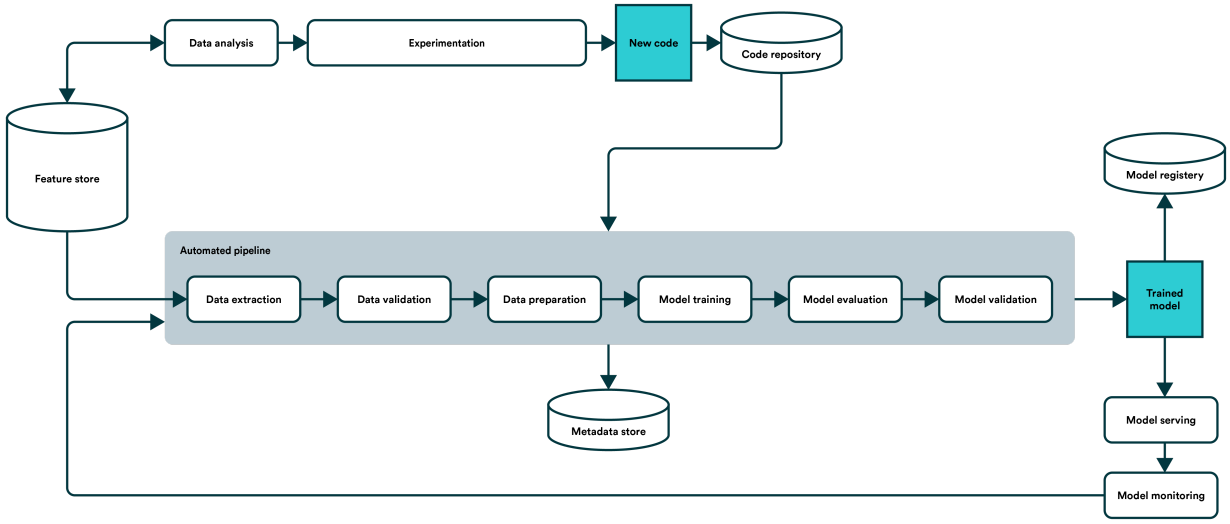
SVM with linear kernel



Pipeline :

A machine learning pipeline is **a way to codify and automate the workflow it takes to produce a machine learning model**

. Machine learning pipelines consist of multiple sequential steps that do everything from data extraction and preprocessing to model training and deployment.



Models :

Model 1 - Using Support Vector Machine

```
from sklearn.svm import SVC
model1 = SVC()
model1.fit(x_train_vect,y_train)
y_pred1=model1.predict(x_test_vect)
from sklearn.metrics import accuracy_score
accuracy_score(y_pred1,y_test)
```

Accuracy score using our model 1 is 0.74

Model 2 - Using pipeline

```
from sklearn.pipeline import make_pipeline
model2=make_pipeline(TfidfVectorizer(),SVC())
model2.fit(x_train,y_train)
y_pred2=model2.predict(x_test)
```

```
from sklearn.metrics import accuracy_score
accuracy_score(y_pred2,y_test)
```

Accuracy score using model 2 is 0.82 which is better than our 1st model.

Model 3 - Using Naive Bayes

```
from sklearn.naive_bayes import MultinomialNB
model3=MultinomialNB()
model3.fit(x_train_vect,y_train)
y_pred3=model3.predict(x_test_vect)
from sklearn.metrics import accuracy_score
accuracy_score(y_pred3,y_test)
```

This shows accuracy score of model 3 is 0.744 which is better than model 1 but not model 2

Model 4 - Using NB + Tfidf Vectorizer pipeline

```
from sklearn.pipeline import make_pipeline
model4=make_pipeline(TfidfVectorizer(),SVC())
model4.fit(x_train,y_train)
y_pred4=model4.predict(x_test)
from sklearn.metrics import accuracy_score
accuracy_score(y_pred4,y_test)
```

Accuracy score of our model4 is 0.82 which is same as our model2 and it is highest accuracy among all our models. So, we shall use this model our web app.

Web app using model 4

```
import joblib
joblib.dump(model4,'Review-Liked')
reload_model=joblib.load('Review-Liked')
```

STREAMLIT - Used to create web app

```
!pip install streamlit --quiet
```

```
%%writefile app.py
import streamlit as st
import joblib
st.title("REVIEW CLASSIFICATION")
reload_model=joblib.load("Review-Liked")

ip=st.text_input("Enter Review: ")
op=reload_model.predict([ip])

if st.button("PREDICT"):
  st.title(op[0])
```

To run STREAMLIT web application

```
!streamlit run app.py & npx localtunnel --port 8501
```

2022-07-08 07:52:46.401 INFO numexpr.utils: NumExpr defaulting to 2 threads.

You can now view your Streamlit app in your browser.

Network URL:

http://172.28.0.2:8501

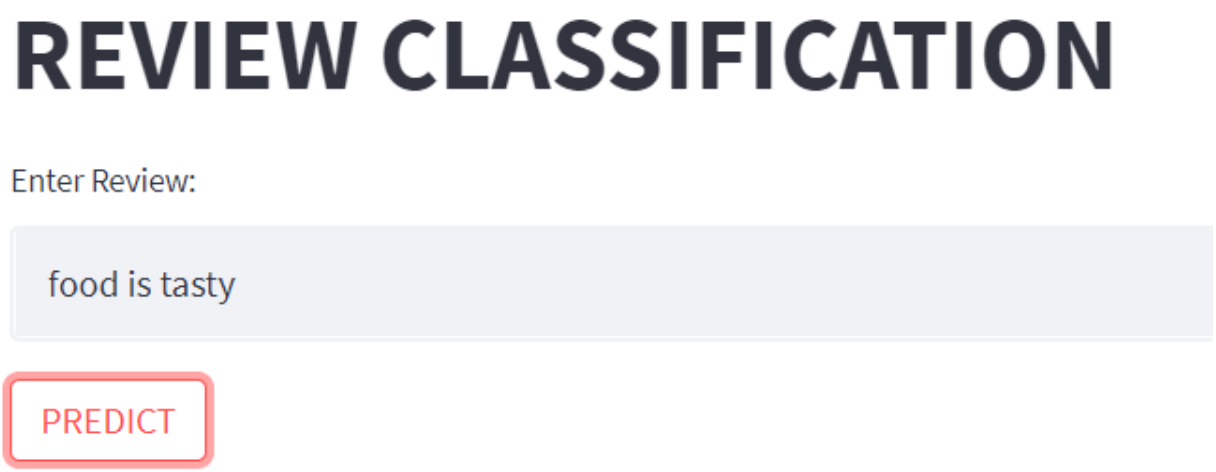
External URL:

http://35.224.250.145:8501

npx: installed 22 in 4.934s
your url is:

https://twenty-worms-refuse-35-224-250-145.loca.lt

Here the last link is a link for tunnel web application where we can check whether a review is positive or negative by entering that review.



Conclusion

Sentiment analysis deals with the classification of texts based on the sentiments they contain. This article focuses on a typical sentiment analysis model consisting of three core steps, namely data preparation, review analysis and sentiment classification, and describes representative techniques involved in those steps.

Sentiment analysis is an emerging research area in text mining and computational linguistics, and has attracted considerable research attention in the past few years. Future research shall explore sophisticated methods for opinion and product feature extraction, as well as new classification models that can address the ordered labels property in rating inference. Applications that utilize results from sentiment analysis is also expected to emerge in the near future.