Universidad Nacional Autónoma De México

Facultad de Ciencias

Clasificación de reseñas de productos

Proyecto Final

Autores:
Molina Bis Victor Hugo
Rivas León Alexis
Sánchez Segura Cristian Alonso

Junio 2020

1. Introducción

El comercio por internet consiste en la compra y venta de productos o servicio a través de internet. Este tipo de comercio ha crecido en los últimos años debido a la exploción del internet, cada vez es más común realizar las compras por medio de plataformas digitales de distintas empresas a realizarlas en su tienda física. Ya sea por comodidad, precios o tiempo, estas plataformas cada vez cuentan con más usuarios y es posible que unos años sustituyan a las tiendas físicas casi en su totalidad.

Las plataformas cuentan con un sistema de reseñas de productos, en el cual los usuarios pueden escribir comentarios acerca de su experiencia con ellos; los productos buenos regularmente reciben muy buenas reseñas, caso contrario con los productos muy malos. Normalmente los comentarios muy buenos o malos predominan en las reseñas, pero los comentarios que realmente ayudan a las empresas a mejorar sus producto o servicios no se encuentran en este tipo de comentarios, se encuentran en un intermedio y estos no son los que se muestran al inicio de la sección. Es por esto que decidimos crear una herramienta para poder clasificar los comentarios que las empresas reciben acerca de sus productos en aquellos que contienen retroalimentación y aquellos que no lo tienen, con esto se podrá facilitar el trabajo de encontrar las inquietudes que tienen los usuarios y los factores que pueden ayudar a mejorar la experiencia y funcionamiento de sus productos o servicios.

2. Objetivo

Crear una herramienta que permita clasificar los comentarios de una serie de productos dependiendo de su contenido, la clasificación consiste en dividir los comentarios en dos grupos, los que contienen retroalimentación acerca de los productos y los que no la contienen. Con esto se pretende ayudar a los vendedores a conocer las recomendaciones que los usuarios tienen acerca de sus productos y poder mejorar la calidad de ellos.

3. Métodos

- Obtención de datos
  El dataset que se utilizó es el siguiente Amazon Musical instruments Reviews. Consta de los ID's de los usuarios, ID's de los productos, nombre de usuario, utilidad del comentario, comentario del producto, valoración, resumen del comentario y las fechas en las que fueron realizados. Dada la tarea que se deaseaba realizar y el dataset con el que se contaba, había columnas que no brindaban información relevante, por lo que se redujeron las columnas de la tabla a dos, la valoración y el comentario. La columna del comentario consta de la unión de dos campos, la opinión del usuario y el resumen de éste, con esto el comentario también contiene las palabras

clave que se encuentran en el resumen.
Para asignarles a los comentarios una clase (con feedback y sin feedback), se optó por definirla por medio del rating hacia el producto. Por lo general los comentarios con una valoración de 5 no contienen feedback, pero al momento del desarrollo nos percatamos que se obtenían mejores resultados al no contar con los de una valoración de 4 y solo tomar como feedback aquellos que son menores a 4.

- Construcción del clasificador
  Por lo general los textos cuentan con ruido en ellos que complican la clasificación, es decir, cuentan con signos de puntuación, stopwords, etc., por lo que se decidió hacer una limpieza del texto como primer paso. Esta limpieza consiste en convertir todo el texto a minúsculas, eliminar números y signos de puntuación. Se tomó en consideración la eliminación de las stopwords (the, a, and), solo que esto implicaba eliminar la palabra *but*, la cual es una palabra clave en los comentarios que incluyen feedback, por lo que se mantuvo el texto con este tipo de palabras.

  Los textos son una serie de palabras y para poder ejecutar los algoritmos en ellos se necesita convertir esos textos en vectores. Para realizar esta acción utilizamos **CountVectorizer**, con esto se aprende el vocabulario y se obtiene una matriz.
  Con **TF** se hace un score de las palabras dentro del documento, mientras más se use una palabra más grande será su TF. **IDF** es otro coeficiente que debería decrementar siguiendo el número de ocurrencias donde el término se repita. Después de aplicar el CountVectorizer aplicamos **TF-IDF** y proseguimos con el algoritmo que nos ayudará en la clasificación. Existen varios algoritmos que ayudan a la clasificación de textos, nosotros probamos 3:

  - Regresión logística
  - SVM (Support Vector Machine)
  - Naive Bayes

  Después de hacer las pruebas con ellos y nuestro conjunto de datos, se optó por intentar conocer su mejor accuracy y los parámetros que lo lograban. Al comparar todos los resultados obtenidos, el de mejor rendimiento se obtuvo con SVM, por lo que se utilizó ésta como el algoritmo de clasificación.

4. Resultados

Se obtuvieron distintos resultados con cada uno de los algoritmos de clasificación:

- Regresión Logística

El desempeño con **Logistic Regression**

```
[131] predicted_LR = text_clf_LR.predict(resume_test["resume"])
      np.mean(predicted_LR == resume_test["overall"])
```

```
0.8904910366328916
```

- Naive Bayes

El desempeño del **Naive Bayes**

```
[129] predicted_NB = text_clf.predict(resume_test["resume"])
      np.mean(predicted_NB == resume_test["overall"])
```

```
0.8803585346843336
```

- SVM

El desempeño con **Support Vector Machines**

```
[133] predicted_SVM = text_clf_SVM.predict(resume_test["resume"])
      np.mean(predicted_SVM == resume_test["overall"])
```

```
0.8803585346843336
```

Se buscaron los parámetros con los que se podrían obtener lo mejores resultados de cada clasificador. SVM tuvo un mejor desempeño, alcanzando un 90 % de accuracy el test, así que usamos este para seguir con la clasificación de comentarios:

- Regresión Logística

El mejor score y parámetros de **Logistic Regression**

```
print(gs_clf_LR.best_score_)
print(gs_clf_LR.best_params_)
```

```
0.8778427550357375
{'clf__C': 0.1, 'tfidf__use_idf': True, 'vect__ngram_range': (1, 1)}
```

- Naive Bayes

El mejor score y parámetros para **Naive Bayes**

```
[136] print(gs_clf.best_score_)
      print(gs_clf.best_params_)

      0.8810916179337231
      {'clf__alpha': 1e-05, 'tfidf__use_idf': True, 'vect__ngram_range': [1, 3]}
```

- SVM

El mejor socre y parámetros para **Support Vector Machine**

```
[ ] print(gs_clf_SVM.best_score_)
    print(gs_clf_SVM.best_params_)

    0.8990253411306043
    {'clf-svm__alpha': 1e-05, 'clf-svm__epsilon': 0.01, 'tfidf__use_idf': True, 'vect__ngram_range': (1, 2)}
```

- SVM con el cojunto test

```
[ ] text_clf_SVM = Pipeline([
        ('vect', CountVectorizer(ngram_range=(1,2))),
        ('tfidf', TfidfTransformer()),
        ('clf-svm', SGDClassifier(alpha=1e-05, epsilon=0.01)),
    ])
    text_clf_SVM = text_clf_SVM.fit(resume_train["resume"], resume_train["overall"])
```

Con los nuevos parámetros, se alcanza un 90% de accuracy en el test

```
[ ] predicted_SVM = text_clf_SVM.predict(resume_test["resume"])
    np.mean(predicted_SVM == resume_test["overall"])

    0.9029618082618862
```

La función que clasifica comentarios entre *feedback* y *no feedback* los comentarios que le introducimos arrojó buenos resultados:

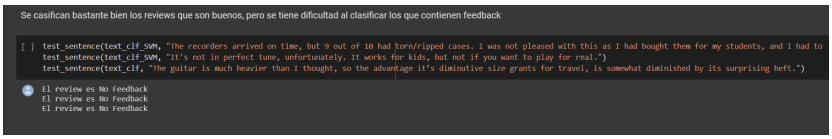
Se crea una función para clasificar reviews

```
def test_sentence(model,sentence):
    sentence = [sentence]
    result = model.predict(sentence)
    res = ['No Feedback','Feedback']
    print("El review es %s" % (res[int(result)]))

test_sentence(text_clf_SVM, "It's a perfect starter pack. And the price is right.")
test_sentence(text_clf_SVM, "The trumpet is hard to blow and the keys stick")
test_sentence(text_clf_SVM, "This instrument has a good sound and is easy to play. It is good for beginners, especially at its price level.")
test_sentence(text_clf_SVM, "Not worth the money. It isn't the easiest to tune.")
test_sentence(text_clf_SVM, "Lovely recorder. Perfect for my 5 year old who is starting lessons.")
test_sentence(text_clf_SVM, "These instruments (the ones made in Italy, not China) have been great for my preschoolers. We originally bought a trumpet for my son, it held up to
test_sentence(text_clf_SVM, "I wish I would not have bought this. This sound so horrible, and my kid keeps playing it. I guess you get what you pay for.")
test_sentence(text_clf_SVM, "The piano is very entertaining for kids. It has so many options to play like different instruments . It's very good. In built music is nice for young
test_sentence(text_clf_SVM, "Piano stopped working even after replacing the batteries few times .The sound quality was ok. Ended up returning")
```

```
El review es No Feedback
El review es Feedback
El review es No Feedback
El review es Feedback
El review es No Feedback
El review es No Feedback
El review es No Feedback
El review es Feedback
El review es No Feedback
El review es Feedback
```

Sin embargo, hay comentarios que resultaron más complicados de clasificar y se cometieron errores:



5. Conclusiones

Los clasificadores de texto resultan de gran utilidad para una gran cantidad de actividades y es un trabajo que puede automatizarse sin dejar de obtener buenos resultados.

La calidad del conjunto de datos con el que se trabaja juega un papel muy importante en la clasificación de textos, ya que gran parte de los resultados que se obtienen depende de este. Con la ayuda de TF-IDF se pueden construir clasificadores de forma sencilla, además, al tener una variedad de algoritmos que ayudan a la clasificación, se cuenta con una gran cantidad de opciones que se adaptan al tipo de datos con los que se trabajan y los resultados finales pueden ser buenos. Sin embargo, para construir un clasificador con un gran desempeño y que cometa pocos errores, es posible que necesiten utilizar una mayor cantidad de herramientas y un conjunto de datos que contenga información que ayude a mejorar el desempeño.

Referencias

[1] Shaikh, J. (2017, July 23). *Machine Learning, NLP: Text Classification using scikit-learn, python and NLTK.* Retrieved from https://towardsdatascience.com/machine-learning-nlp-text-classification-using-scikit-learn-python-and-nltk-c52b92a7c73a

[2] Villa, J. G. (2019, February 13). *TF IDF: herramientas para mejorar la relevancia de tus contenidos.* Retrieved from https://useo.es/tf-idf-relevancia/#Que_es_el_TF_IDF

[3] Singh, A. (2019, November 4). *Sentiment Classifier using Tfidf - DataSeries.* Retrieved from https://medium.com/dataseries/sentiment-classifier-using-tfidf-3ffce3f1cbd5

[4] https://scikit-learn.org/stable/