2022

Análisis De Sentimiento de Tweets en español

PROCESAMIENTO DEL LENGUAJE NATURAL

JOSE RODRIGO CALLES VILLAESCUSA

Tabla de Contenidos

Introducción

El proyecto consiste en hacer un análisis de sentimiento de una base de datos de tweets de México. El conjunto de datos se consiguió en la página del TASS (Taller de análisis semántico de la SEPLN).

Sección 1 Preparación de los datos

1.1 Conversión de datos a un diccionario

Los datos se descargaron en formato XML y no se pudieron convertir en un data frame de forma inmediata, por lo que se tuvo que primero convertir a un diccionario que después fue aplanado por una función llamada 'flatten_dict' y posteriormente le aplicamos la función a los datos de entrenamiento y de prueba.

1.2 Conversión a un Data frame

Después de tener los datos en formato de diccionario, lo pasamos a un data frame utilizando la función 'DataFrame' de la librería sklearn. Para poder partir los datos a manera que nuestro gusto, se concatenaron los datos de entrenamiento y de prueba para posteriormente partirlos de la manera que fuera más convierte.

Sección 2 Limpieza de datos y división de los datos

En general los datos venían bastantes limpios, pero había ciertos valores que necesitaban una limpieza.

2.1

Se quitaron los datos con valores 'NONE' de las etiquetas del conjunto de datos para no ocasionar problemas a la hora del entrenamiento. Esto redujo el número de filas de 1499 a 896, lo que nos dejó con una cantidad un poco reducida de datos con que trabajar.

2.2

Se decidió mapear los valores de las etiquetas para trabajar solamente con datos numéricos. Se utilizó la función 'map' para lograr esto.

2.3 Limpiando texto

Se limpiaron los tweets utilizando la función 'LimpiaTexto', donde se quitaron los caracteres de "@" y "#" que corresponden a las menciones y "hashtags", además de remplazar los espacios en blanco y quitar los híper textos. Esta función se aplicó a la columna "content" que corresponde a todos los tweets que tenemos y se creó una columna nueva llamada "tweets_limpios" donde están los tweets ya con la limpieza aplicada.

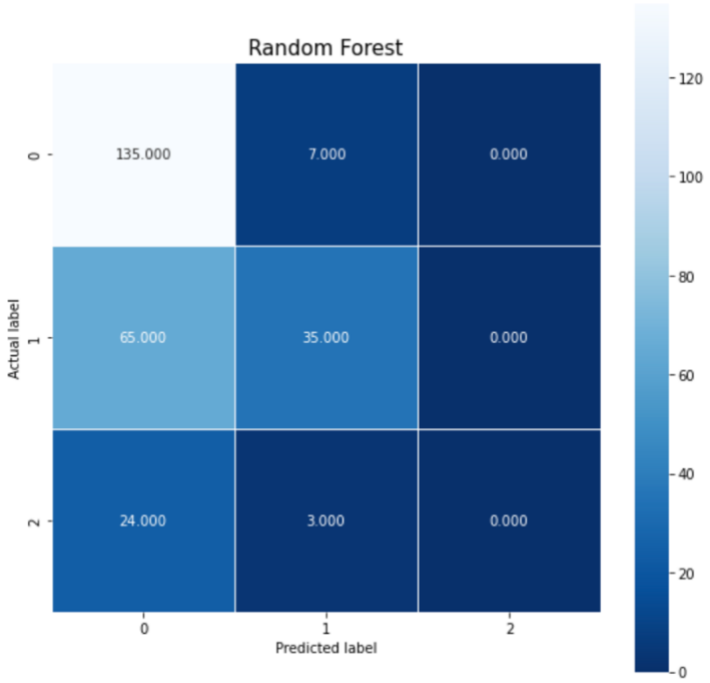
2.4 División de los datos

Se agarró a la columna con los tweets limpios como las características que se alimentarán a los modelos y la columna de 'sentiment_polarity_value', que tienen el sentimiento de cada tweet como nuestras etiquetas. Posteriormente se dividieron los datos utilizando la función de 'train_test_split' de sklearn, donde se dejó un 70% para el entrenamiento y un 30% para pruebas.

Sección 3. Modelos

3.1 Modelo Random Forest

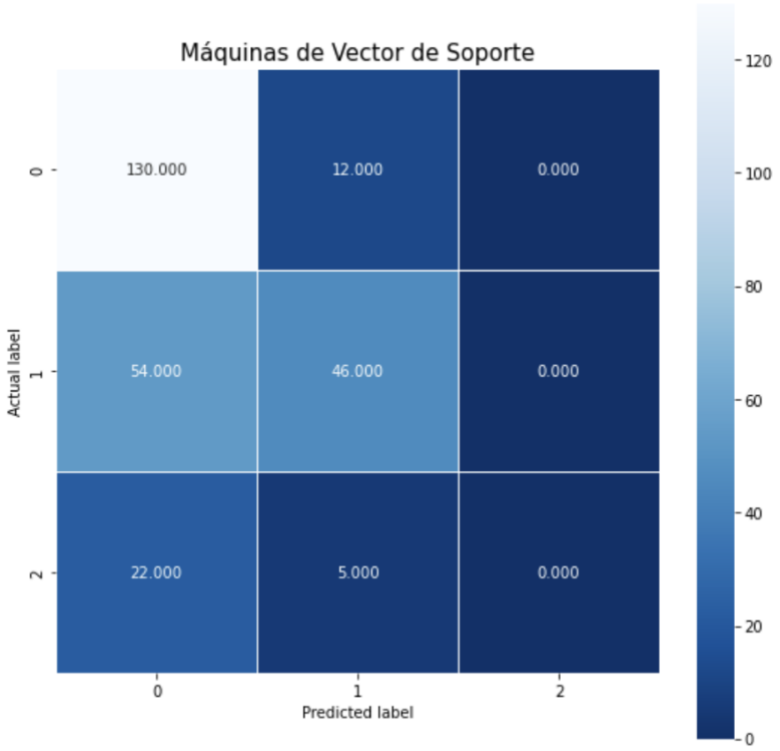
El primer modelo que se decidió utilizar fue el de Random Forest. Se usó la función de 'RandomForestClassifier' de sklearn para llevar a cabo esta tarea, además de la función 'GridSearchCV' para aplicar un ajuste de híper parámetros. Los híper parámetros que se consideraron para el ajuste fueron: El número de estimadores, la profundidad máxima, el número de muestras para que un nodo se pueda partir, el número de muestras para que un nodo pueda ser considerada un nodo hoja y el Bootstrap.
Después del entrenamiento y las predicciones se obtuvo una precisión del 63.20% y se obtuvo la siguiente matriz de confusión:



Basado en esta matriz podemos ver que el modelo trabajó decentemente en calificar los sentimientos negativos verdaderos, pero en lo demás batalló más, especialmente en predecir los sentimientos neutrales.

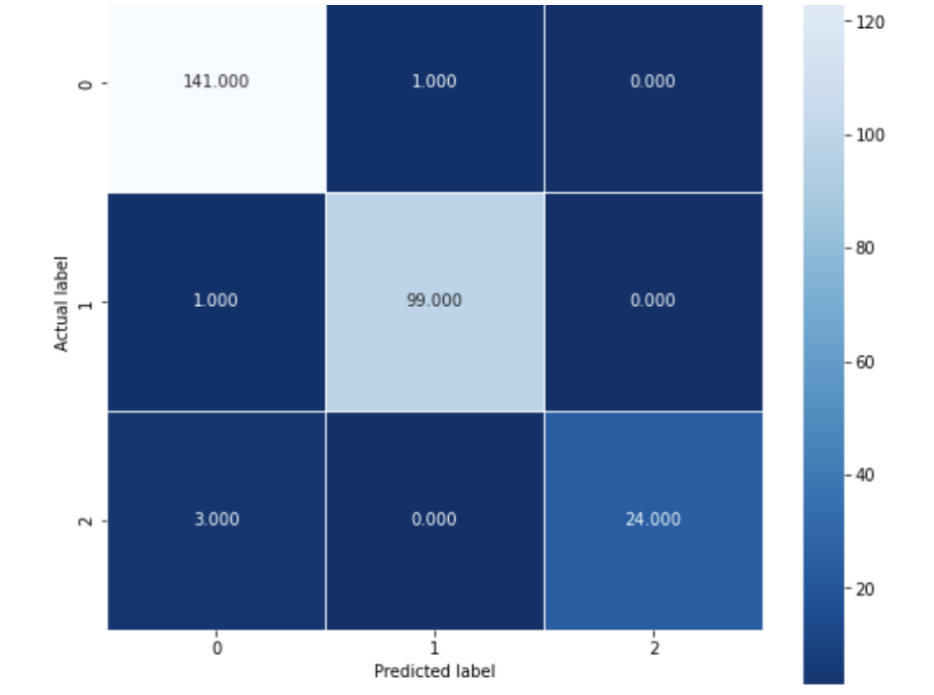
3.2 Modelo Máquinas de Vector de Soporte

De igual manera que en el modelo de random forest, se utilizó la función 'GridSearchCV' para el ajuste de híper parámetros y entrenamiento. Los híper parámetros que se consideraron fueron el parámetro de regularización "C", la gamma y el kernel que utilizará el modelo.
Después de entrenar y hacer las predicciones, obtuvimos una precisión del 65.43%, lo que representa una mejora en comparación al modelo de random forest, pero sigue estando lejos de una precisión deseada. La siguiente imagen es la matriz de confusión resultante del modelo:



3.3 Modelo Regresión Logística

En este modelo también se utilizó el método de búsqueda en grid para encontrar los mejores híper parámetros para nuestro modelo. Los híper parámetros que se eligieron para hace el ajuste fueron: El “solver” que es el algoritmo que se usa para el problema de optimización, el parámetro de penalización y el valor de “C”. Después de obtener el diccionario de parámetros, entrenamos el modelo con la combinación de esos parámetros para obtener el mejor modelo posible. Posteriormente se hicieron las predicciones y se obtuvo una precisión del 98.14%, lo que es claramente superior a los otros dos modelos y también es una precisión muy buena para lo que queremos. La siguiente imagen es la matriz de confusión generada por el modelo:



Sección 4. Conclusiones

Inicialmente solo se esperaba trabajar con los modelos de random forest y de máquinas de vector de soporte, pero al no conseguir resultados favorables se decidió intentar con regresión logística, lo cual dio muy buenos resultados.

El trabajar con tan pocos datos puede ser un factor del por qué los primeros dos modelos no trabajaron bien, también es posible que se pueda mejorar la limpieza de los datos para obtener mejores resultados con esos modelos, sin embargo, el modelo de regresión logística es satisfactorio para llevar a cabo la tarea en mano, por lo que, con los resultados obtenidos en este proyecto, lo preferible es quedarse con ese modelo.