Actividad 2.01

Análisis de sentimientos (Criticas de Cine)

Integrantes:

Cesar Eduardo Elías del Hoyo

José Luis Sandoval Pérez

Diego Emanuel Saucedo Ortega

Carlos Daniel Torres Macías

Universidad Autónoma de Aguascalientes

Aguascalientes, Ags, 25 de marzo, 2024

Análisis

Gaussian Naive Bayes es un tipo de método Naive Bayes en el que se consideran atributos continuos y las características de los datos siguen una distribución gaussiana a lo largo del conjunto de datos. Gaussian Naive Bayes es un tipo de algoritmo de clasificación que funciona con características continuamente distribuidas de forma normal y está basado en el algoritmo Naive Bayes. Antes de profundizar en este tema, debemos obtener una comprensión básica de los principios en los que se basa Gaussian Naive Bayes. A continuación, se presentan algunos términos que pueden ayudar a adquirir conocimientos y facilitar nuestro estudio posterior:

El algoritmo Naive Bayes Classifier se basa en un sencillo concepto de la teoría de la probabilidad llamada teorema de Bayes. Este algoritmo de clasificación funciona bien al predecir a qué clase pertenecen las características presentes. La palabra 'naive' en el nombre de este algoritmo se deriva del supuesto que realiza al realizar la predicción de la etiqueta de las características. La suposición en esta etapa es que todas las características son independientes entre sí; aunque esto no sea necesariamente cierto en situaciones del mundo real, el algoritmo sigue funcionando bien. En el proceso de entrenamiento de un clasificador Naive Bayes, el algoritmo se centra en dos factores, según el conjunto de datos que está ajustando:

El algoritmo de Naive Bayes considera la probabilidad de ocurrencia de cada clase y asigna el valor de la etiqueta a la clase con mayor probabilidad.

Este algoritmo luego compara diferentes probabilidades posteriores según el número de clases presentes en los datos, y se asigna a la clase con mayor probabilidad a la combinación de características presente.

El teorema de Bayes es un método para actualizar las probabilidades según la información nueva. La teoría es la siguiente:

Por lo tanto, P(A|B) es la probabilidad posterior y describe la probabilidad de ocurrencia de A, dado que B ha ocurrido. P(A) es la probabilidad anterior, y P(B) es la probabilidad de

ocurrencia del evento B, y P(B|A) es la probabilidad de ocurrencia de B, dado que A ya ha ocurrido.

Gaussian Naive Bayes es la aplicación de Naive Bayes en datos normalmente distribuidos. Gaussian Naive Bayes supone que la probabilidad (P()) sigue la distribución gaussiana para cada atributo para un valor dado de clase. Por lo tanto,

Para clasificar cada punto de datos nuevo "x", el algoritmo encuentra el valor máximo de la probabilidad posterior de cada clase y asigna el punto de datos a esa clase.

Implementación

La implementación para poder clasificar las críticas de las películas tiene como pasos fundamentales los siguientes:

1. **Preprocesamiento de datos:**

En primer lugar, se deben cargar un set de datos que consisten en críticas de cine etiquetadas como positivas o negativas. Cada crítica estaría asociada con una etiqueta que indica si la crítica es positiva o negativa. En nuestro caso podemos encontrar dentro de nuestro data set la reseña y un sentimiento 1 (positivo) y un sentimiento 0 (negativo).

	review	sentiment
0	In 1974, the teenager Martha Moxley (Maggie Gr...	1
1	OK... so... I really like Kris Kristofferson a...	0
2	***SPOILER*** Do not read this, if you think a...	0
3	hi for all the people who have seen this wonde...	1
4	I recently bought the DVD, forgetting just how...	0

También es de suma importancia en el preprocesamiento de datos limpiar y formatear las críticas para que sean adecuadas para el análisis. Esto puede incluir la eliminación de caracteres especiales, la conversión del texto a minúsculas, la tokenización de las palabras, la eliminación de palabras irrelevantes (stopwords) y la lematización o el stemming para reducir las palabras a su forma base.

Dentro de nuestro proceso de limpiar el texto se utiliza una librería de Python llamada “Textblob” esta de Python es utilizada para procesar datos textuales. Proporciona una API sencilla para profundizar en tareas comunes de procesamiento del lenguaje natural (PLN), como etiquetado de partes del discurso, extracción de frases sustantivas, análisis de sentimientos, clasificación y más.

Dentro de esta librería se nos facilita determinar el sentimiento de la crítica. En este caso con 2 parámetros “Polaridad” y “Subjetividad”. La polaridad nos indica que tan positivo es la crítica, esto se encuentra en un valor de -1 a 1. La subjetividad califica cuantas de las palabras en una oración están relacionadas con creencias, suposiciones o experiencias del

autor, por ejemplo, palabras como “yo creo”, “en mi experiencia” que restan credibilidad a lo que se expresa pues es más difícil replicarlo, este índice es representado con un numero decimal entre el rango [0,1] siendo 1, la subjetividad máxima.

2. **División de datos:**

Después de preprocesar los datos, se divide el conjunto de datos en un conjunto de entrenamiento y un conjunto de prueba. El conjunto de entrenamiento se utilizará para entrenar el modelo, mientras que el conjunto de prueba se utilizará para evaluar su rendimiento.

3. **Entrenamiento del modelo:**

Al tener vez los datos divididos, tanto para entrenamiento y prueba, se debe entrenar el modelo en este caso el clasificador Naive Bayes utilizando el conjunto de entrenamiento.

4. **Evaluación del modelo:**

Después de entrenar el modelo, lo evaluarías utilizando el conjunto de entrenamiento, obteniendo el score y el fit. Después evaluamos el modelo, pero ahora con los datos de prueba y sacamos la matriz de confusión.

Evaluación

Una vez realizada la implementación en código de Jupyter notebook, realizamos una prueba (corrida) para evaluar los resultados obtenidos del clasificador Naive Bayes Gaussiano y se entrena con los datos de entrenamiento. Se evalúa el modelo calculando la precisión en los conjuntos de entrenamiento y prueba. Además, se realiza una predicción en el conjunto de pruebas y e calcula la matriz de confusión para visualizar el rendimiento del problema

Ahora, implementamos el modelo de K-Naive Bayes

```
In [53]: clf = GaussianNB()
         clf

Out[53]: GaussianNB
         GaussianNB()
```

Una vez implementamos el modelo Gaussiano de Naive Bayes, se realiza el entrenamiento del modelo de clasificación de la siguiente manera:

```
In [54]: # entrenamos el modelo con los set de entrenamiento
         clf.fit(X_train,Y_train)
         clf.score(X_train,Y_train)

Out[54]: 0.7647428571428572
```

Podemos observar que, realizamos el entrenamiento y la evaluación del modelo de clasificación. De esta manera, aplicamos el modelo de clasificación utilizando el método fit() para clasificar el conjunto de características de entrenamiento (X_train) que contiene las polaridades y subjetividades calculadas a partir de las críticas de cine y la variable objetivo de entrenamiento (Y_train) que contiene las etiquetas de sentimiento correspondientes a cada crítica. Durante este entrenamiento, el modelo ajusta sus parámetros utilizando los datos de entrenamiento para aprender la relación entre las características y las etiquetas de sentimiento.

Una vez realizada la función fit(), pasamos a evaluar mediante el score() el rendimiento del modelo en los datos del entrenamiento. Este método devuelve la precisión del modelo en los

datos de entrenamiento, que es la fracción de muestras de entrenamiento correctamente clasificadas por el modelo.

Para este caso, observamos que el valor obtenido es de 0.7647428571428572, lo que indica que alrededor del 76.47% de las críticas en el conjunto de entrenamiento fueron clasificadas correctamente por el modelo, en otras palabras, esto sugiere que el modelo tiene cierta capacidad para aprender de los datos de entrenamiento y generalizar a instancias similares

Ahora bien, una vez entrenado el modelo, lo pusimos a prueba usando los valores del dataset de prueba que usamos para la práctica para obtener un rendimiento del modelo de los datos del dataset

```
In [55]: # probamos con los set de prueba
         clf.score(X_test,Y_test)

Out[55]: 0.7634666666666666
```

Podemos observar que el valor devuelto ha sido de 0.763466666666 lo cual no indica que el 76.34% de las críticas en el conjunto de prueba fueron clasificadas correctamente por el modelo, dándonos un valor prácticamente similar al de los datos de entrenamiento

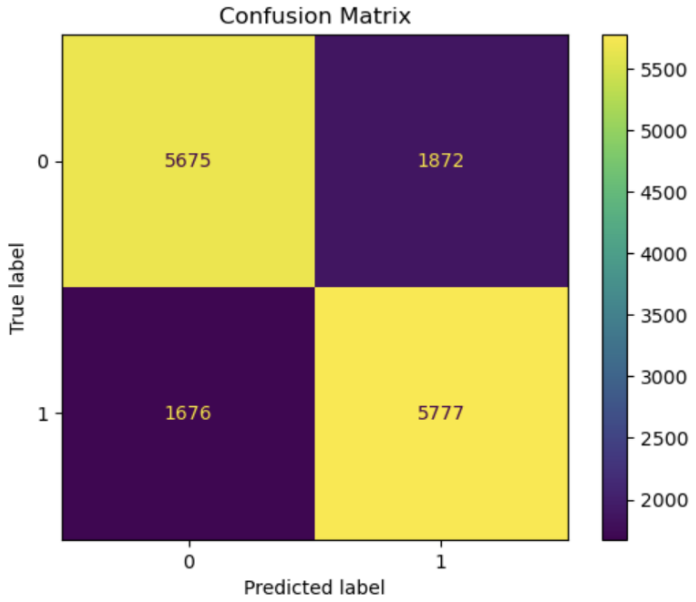
Con esto, podemos confirmar que el modelo esta correctamente generalizado para clasificar y evaluar datos no vistos que permiten obtener un resultado a las críticas de cine de los valores dados, nuestro modelo no se encuentra sobre ajustado y tiene una precisión en el conjunto cercana a la precisión en el conjunto del entrenamiento

Por último, mostramos de manera visual los resultados obtenidos mediante una matriz de confusión que nos permitió observar la tabla con la clasificación de modelo y visualizar su matriz de una manera más estética

Gaussian Naive Bayes

Matriz de confusion

```
In [57]: from sklearn.metrics import confusion_matrix,ConfusionMatrixDisplay
cm = confusion_matrix(Y_test,y_hat)
ConfusionMatrixDisplay(cm).plot()
plt.title("Confusion Matrix")
plt.show()
```



Podemos observar el modelo que muestra el número de verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos

Aquí observamos que el análisis muestra que los valores más altos, esto es, los recuadros amarillos, representan las críticas Verdadero positivo (5675) y Verdadero falso (5777), esto nos da a entender, que se representaron más altas las críticas correctamente clasificas, mientras que los recuadros morados representan las críticas Falsas negativas (1872) y Falsas positivas (1676) dándonos valores mucho más bajo

Analizando esta matriz de confusión, buscamos una alta calidad de verdaderos positivos y negativos, y así fue como se obtuvo, indicándonos un buen rendimiento en el modelo de la clasificación

Conclusiones

En análisis de texto es gracias al cúmulo de información, elementos, conocimientos y experiencias que se han recolectado a través de décadas. Es una herramienta importante en nuestros días, pues, cualquier tipo de ventaja en el conocer la reacción de las personas es un hincapié para el desarrollo de mejores métodos de comunicación, de redacción, acercamiento. Mediante esta actividad, pudimos encontrar diversas herramientas que permitieran analizar el texto, principalmente, conocer el tratamiento y el cómo se puede transformar.

Naive Bayes y sus diversos tipos de modelos, aportan herramientas para el conocimiento de modelos de aprendizaje supervisado, poder complementarlo con aplicaciones abstractas como lo es un sentimiento, nos muestra que el aprendizaje de máquinas es el presente.

Referencias

- GfG. (2023, 13 noviembre). Gaussian naive bayes. GeeksforGeeks. https://www.geeksforgeeks.org/gaussian-naive-bayes/
- Play Strands, our newest word-search game with a twist. (s. f.). NYT Puzzles - Strands. https://www.nytimes.com/games/strands
- *sklearn.preprocessing.MinMaxScaler*. (s. f.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html
- Pythonology. (2021, 27 julio). *Intro to TextBlob for Text Analysis and Processing | Python Tutorial* [Vídeo]. YouTube. https://www.youtube.com/watch?v=pkdmcsyYvb4Lis