Universidad del Valle de Guatemala
Data Science
Sección 10
Lynette Garcia

Proyecto 2

Alfredo Quezada 191002
Estuardo Hernández 19202
Marco Ramirez 19588

Introducción

La minería de texto o el text mining por su nombre el inglés, es una técnica o metodología que se basa en encontrar, formar o determinar patrones o características a partir de un párrafo o incluso una base de datos para poder determinar o influenciar en una investigación o similar.

Tomando en cuenta lo anterior, el contenido de este informe se base en la utilización de la minería de texto, para determinar o clasificar elementos argumentativos en la escritura, para estudiantes, dichos argumentos pueden ser calificados como “efectivos”, “adecuados” o incluso “ineficientes”, esto resulta muy útil para cuando se quiera dar retroalimentación a los propios estudiantes para que se denote en lo que puede mejorar o con lo que puede seguir.

Y es justamente ese el problema que estamos buscando resolver, de que no todos desarrollan una calidad aceptable de elementos argumentativos, desarrollo de ideas o la organización de ideas, por lo tanto, se realizó un análisis para determinar la calidad de los argumentos, en base a su efectividad, como se mencionó al inicio se hizo todo solo con la ayuda de la minera de texto, con dos metodologías diferentes para poder tener una comparativa y así determinar de manera mucho más efectiva y acertada su calidad.

Objetivos

- Objetivos generales
    - Catalogar los argumentos en base a su efectividad.
    - Determinar el tipo de discurso con mayor efectividad.
    - Distinguir el texto utilizable y el no utilizable
    - Crear un texto limpio
- Objetivos específicos
    - Catalogar argumentos como “Ineficaz”, “Adecuado” o “Eficaz”.
    - Identificar los sentimientos obtenidos en base a los argumentos realizados.
    - Crear modelos correctos y eficaces.

Marco Teórico

El procesamiento del lenguaje natural o PNL por sus cifras, es el campo de conocimiento de la **Inteligencia Artificial** que se ocupa de investigar la manera de comunicar las máquinas con las personas mediante el uso de lenguas naturales, como el español, el inglés o el chino. Virtualmente, cualquier lengua humana puede ser tratada por los ordenadores. Lógicamente, limitaciones de interés económico o práctico hace que solo las lenguas más habladas o utilizadas en el mundo digital tengan aplicaciones en uso.

Pensemos en cuántas lenguas hablan Siri (20) o Google Assistant (8). El inglés, español, alemán, francés, portugués, chino, árabe y japonés (no necesariamente en este orden) son las que cuentan con más aplicaciones que las entienden. Google Translate es la que más lenguas trata, superando el centenar… pero hay entre 5000 y 7000 lenguas en el mundo.

(Moreno, n.d.)

Por su parte, una vez se encuentre identificado el texto con el que se desea trabajar, se deben de seguir una serie de pasos previos para poder evaluar un modelo de minería de texto, los cuales son los que listan a continuación:

- Recolección de datos: Su nombre es claramente entendible, pero es uno de los pasos, o por no decir el paso más importante, ya que se debe de tener claramente identificado el texto a trabajar y la cantidad con la que se hará para así evitar ineficiencias o muchos aspectos atípicos.
- Preprocesamiento: Esta parte forma parte del paso anterior pero se hace solamente para determinar que texto puede ser de utilidad para lo que se desea y cuál debe de considerarse como descarte.
- Limpieza: Va de la mano con el preprocesamiento pero en este paso es donde se filtra el texto, se eliminan duplicados, se eliminan caracteres no deseados, entre otros.
- Tokenización: Dicho de una forma resumida, este paso se trata de convertir el texto en lenguaje natural a un lenguaje de programación, de esta forma ya se pueden realizar los procesos de minería apropiadamente.
- Descubrimiento: Esta es la parte resultante del paso anterior, ya que esto puede ser contado como los resultados, la parte en la que se muestra lo que se obtuvo en los modelos realizados.
- Visualización: Esta parte dependiendo de lo que se quiera determinar es opcional o no, ya que esta parte como su nombre lo indica, se realizan gráficos del tipo que sean necesarios para mostrar los hallazgos o descubrimientos.

Ahora bien, sabiendo cuales son los pasos para realizar una minería de textos, debemos enfocarnos en las herramientas o mejor dicho, en los algoritmos que podemos utilizar para realizar dicha actividad, entre ellos se encuentran:

Naive Bayes

El algoritmo clasificador Naïve-Bayes (NBC), es un clasificador probabilístico simple con fuerte suposición de independencia. Aunque la suposición de la independencia de los atributos es generalmente una suposición pobre y se viola a menudo para los conjuntos de datos verdaderos. A menudo proporciona una mejor precisión de clasificación en conjuntos de datos en tiempo real que cualquier otro clasificador. También requiere una pequeña cantidad de datos de entrenamiento. El clasificador Naïve-Bayes aprende de los datos de entrenamiento y luego predice la clase de la instancia de prueba con la mayor probabilidad posterior. También es útil para datos dimensionales altos ya que la probabilidad de cada atributo se estima independientemente (Medium, 2019).

Las ventajas del modelo Naive Bayes son:

- Fácil de implementar y aplicar a diferentes casos como la clasificación de texto.
- Naive Bayes requiere una pequeña cantidad de datos de entrenamiento para estimar los datos de las pruebas.
- Menos esfuerzo de preparación de datos.

(Medium, 2019)

Las desventajas del modelo Naive Bayes son:

- Tiene la fuerte hipótesis de independencia variable.
- Si la variable categórica tiene una categoría en el conjunto de datos de prueba, que no se observó en el conjunto de datos de entrenamiento, entonces el modelo asignará una probabilidad 0 (cero). En este caso, debe añadirse una unidad de cada conjunto de datos.

(Medium, 2019)

Gaussian Naive Bayes

Uno de los tipos de clasificadores más populares es el llamado en inglés Gaussian Naive Bayes Classifier. Este toma en cuenta la probabilidad, la verosimilitud y la probabilidad marginal. Los nombres Gaussian y Naive del algoritmo vienen de dos suposiciones: - Asumimos que las características de la verosimilitud no estan correlacionada entre ellas. Como no es siempre cierto y es una suposición ingenua es que aparece en el nombre “naive bayes”. - Asumimos que el valor de las características tendrá una distribución normal (gaussiana). Esto nos permite calcular cada parte usando la función de probabilidad de densidad normal (Aprende Machine Learning, 2017).

Las ventajas del modelo Gaussian Naive Bayes son:

- Es rápido.
- Es simple de implementar.
- Funciona bien con conjunto de datos pequeños.
- Va bien con muchas dimensiones (features).
- Llega a dar buenos resultados aún siendo “ingenuo” sin que se cumplan todas las condiciones de distribución necesarias en los datos.

(Aprende Machine Learning, 2017)

Las desventajas del modelo Gaussian Naive Bayes son:

- Requiere quitar las dimensiones con correlación.
- Para buenos resultados las entradas deberían cumplir las 2 suposiciones de distribución normal e independencia entre sí (muy difícil que sea así ó deberíamos hacer transformaciones en los datos de entrada).

(Aprende Machine Learning, 2017)

Multinomial Naive Bayes

El clasificador multinomial Naive Bayes es adecuado para la clasificación con características discretas, por ejemplo, recuento de palabras para la clasificación de texto. La distribución multinomial normalmente requiere recuentos de características de enteros. Sin embargo, en la práctica, los recuentos fraccionarios también pueden funcionar (Anguiano, 2009).

Se usa ampliamente para asignar documentos a clases en función del análisis estadístico de sus contenidos. Proporciona una alternativa al análisis semántico “pesado” basado en IA y simplifica drásticamente la clasificación de datos textuales. La clasificación tiene como objetivo asignar fragmentos de texto (es decir, documentos) a clases determinando la probabilidad de que un documento pertenezca a la clase de otros documentos que tengan el mismo asunto (Anguiano, 2009).

Bayesian Network

Es un modelo grafo probabilístico, un tipo de modelo estático, que representa un conjunto de variables aleatorias y sus dependencias condicionales a través de un grafo acíclico dirigido (DAG por sus siglas en inglés). Por ejemplo, una red bayesiana puede representar las relaciones probabilísticas entre enfermedades y síntomas. Dados los síntomas, la red puede ser usada para computar la probabilidad de la presencia de varias enfermedades (Gal, 2007).

Formalmente, las redes bayesianas son grafos dirigidos acíclicos cuyos nodos representan variables aleatorias en el sentido de Bayes: las mismas pueden ser cantidades observables, variables latentes, parámetros desconocidos o hipótesis. Las aristas representan dependencias condicionales; los nodos que no se encuentran conectados representan variables las cuales son condicionalmente independientes de las otras. Cada nodo tiene asociado una función de probabilidad que toma como entrada un conjunto particular de valores de las variables padres del nodo y devuelve la probabilidad de la variable representada por el nodo (Gal, 2007).

Las ventajas del modelo Bayesian Network son:

- La representación gráfica la convierte en una poderosa herramienta de comunicación, las relaciones causa-efecto se visualizan fácilmente sin la necesidad del cálculo de probabilidades.
- La posibilidad de combinar datos objetivos y subjetivos, esto es una enorme ventaja sobre todo cuando no se cuentan con suficientes datos estadísticos.
- Pueden modelar sistemas complejos.

- La red puede actualizarse rápidamente o modificarse por cambios en la información o un mal desempeño.
- Pueden utilizarse para análisis de “Qué pasa si”, para analizar la sensibilidad de las predicciones, o conclusiones respecto de los supuestos iniciales.

(Gal, 2007)

Las desventajas del modelo Bayesian Network son:

- Las Redes Bayesianas tienen la desventaja de que el modelo es bueno tanto como el que modela lo sea y la percepción que tengan los expertos de la realidad.
- Otra limitación se relaciona con el hecho de que la utilidad de las RBs está basada en la confiabilidad de la información a priori. Una expectativa demasiado optimista o pesimista de las creencias a priori pueden ya sea distorsionar la red o invalidar los resultados. Seleccionar una apropiada distribución de los datos tiene un importante efecto en la calidad de los resultados de la red.

(Gal, 2007)

Metodología

Como primer paso, para resolver el problema nos dedicamos a encontrar los algoritmos que más nos facilitarán la resolución del mismo, los cuales cabe resaltar no son muchos, pero logramos identificar dos que se acercaron bastante al resultado que deseábamos, acto seguido nos dispusimos a customizar los algoritmos para poder obtener un resultado deseado, con dichos algoritmos perfectamente acoplados a nuestras necesidades nos enfocamos en encontrar un conjunto de entrenamiento y un conjunto de prueba, mediante la frecuencia de las palabras y mediante el conteo individual de las mismas, ya que ambos modelos requieren parámetros de entrada diferentes se tuvieron que crear dos versiones de los datos de entrenamiento y prueba, eso si lo que sí llegaron a tener en común ambos modelos es que se usó un 70% de los datos para el entrenamiento y el 30% restante fue utilizado para el conjunto de pruebas.

Para el primer modelo, decidimos usar el algoritmo clasificador Naïve-Bayes (NBC), ya que este tiende a tener una facilidad para aprender de pocos datos de entrenamiento, para luego usar ese aprendizaje para predecir la instancias de prueba con la mayor probabilidad, así como también nos decantamos por este modelo, debido a que es útil para trabajar con datos de una dimensión muy grande, una cantidad de datos bastante poblada y realmente para este proyecto, pero también este modelo nos permitió una fácil implementación y una manera muy fácil de trabajar con las variables categóricas, mientras que para el segundo modelo, nos decidimos por el algoritmo de Random Forest, debido a que al igual que el algoritmo permite un aprendizaje automatizado, con base a conjuntos de entrenamiento, lo mejor es que este modelo nos permitió clasificar para el resultado y eso nos ayudó mucho, ya que en sí, la problemática era poder clasificar en tres formas diferentes el texto.

El equipo computacional fue de mucha ayuda, ya que trabajamos con una máquina compuesta de una memoria RAM de 32 gigabytes, así como también equipada de un Intel I7-9700K y una tarjeta gráfica RTX 2070, con este equipo y con las siguientes librerías:

- *tm:*
- *SnowballC:*
- *wordcloud:*
- *e1071:*
- *caret:*
- *tidyverse:*
- *gmodels:*
- *udpipe*
- *tidytext:*
- *caTools:*
- *ramdomForest:*

Fuimos capaces de crear los modelos que se mostrarán y discutirán a continuación.

Resultados y Análisis

Nuestros datos crudos se encontraban en formato csv, con las siguientes columnas:

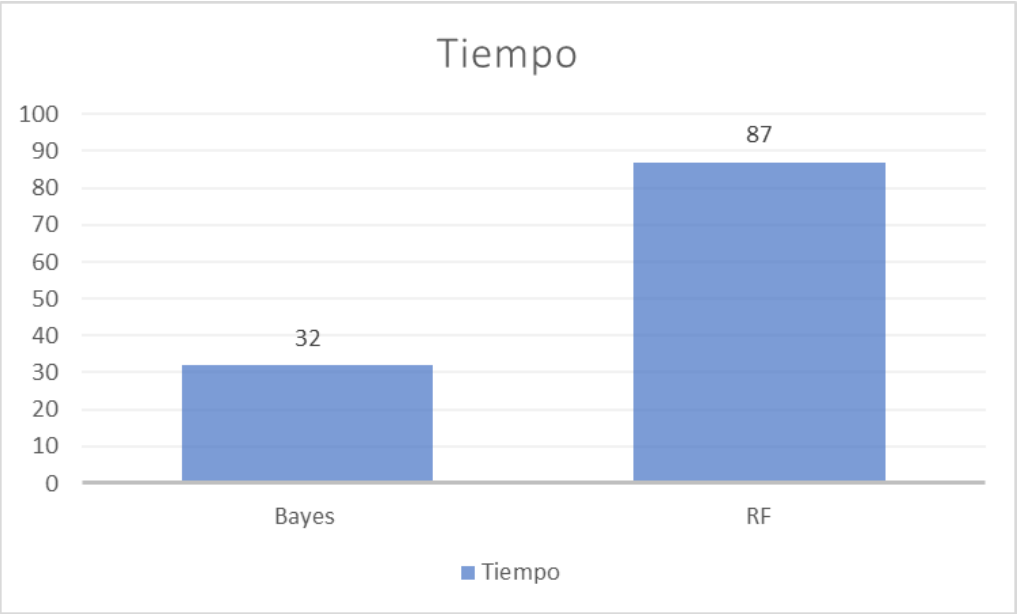
discourse_id	essay_id	discourse_text	discourse_type	discourse_effectiveness

En el cual, “discourse_id” nos daba el ID de cada uno de los discursos que se encontraban dentro del conjunto de datos, seguido del “essay_id” el cual de la misma forma nos daba el código único de los ensayos que se encontraban dentro del conjunto mientras que dentro de “discourse_text” se encontraba lo que más nos interesaba, que era el texto con el que debíamos de trabajar, el ensayo o párrafo que debíamos de analizar, seguido de “discourse_type” que nos cataloga el tipo de ensayo que teníamos para terminar con la variable de efectividad que nos indicaba que tan efectivo era el discurso.

Pero para poder trabajar con la variable del texto, primero hicimos una limpieza de caracteres que no nos representaban ninguna utilidad, por dar un ejemplo nos topamos con textos que contenían números, símbolos e incluso lo que parecen ser emojis no cargados correctamente, todos estos no deseados fueron eliminados antes de poder trabajar con ellos para que en el modelo no se encontraran esos datos atípicos o incluso nos dieran problemas para poder operar los modelos, habiendo terminado la eliminación de esos, nos dispusimos a poner en un mismo formato todo el texto, es decir tratamos de que no hubieran más de una mayúscula en todos los textos y que todos se encontraran de cierta forma en un texto plano, para facilitar la operación del modelo, por último, nos dispusimos a crear una nueva tabla con los solo las columnas que eran esenciales para los modelos, como lo pueden llegar a ser el texto, el tipo y su ID.

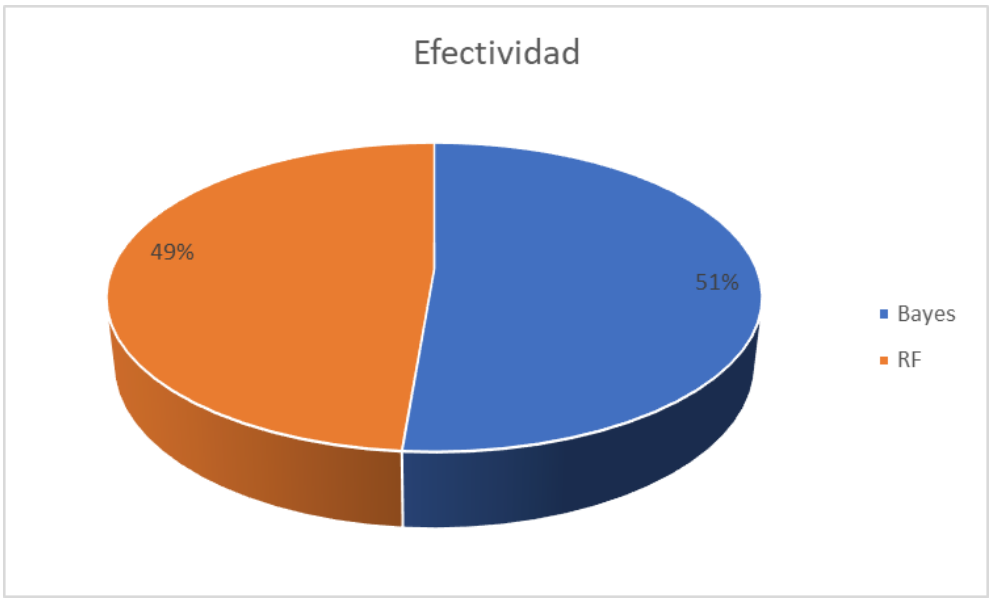
A nivel de ajustes, realmente no se tuvo que realizar muchos ajustes a los algoritmos para su utilización, lo más difícil de esta parte fue lograr crear un pre procesamiento adecuado, ya que al tener una cantidad de datos de aproximadamente 36700 datos, entonces fue dificultoso crear una manera en la que pudiéramos comprimir los datos para que el pre procesamiento fuera no tan extenuante y muchos más rápido, pero luego de poder conseguir el pre procesamiento correcto para ambos modelos, pudimos comprimir los 36700 datos en un subconjunto que fue mucho más fácil de interpretar y ejecutar, porque no tenia que hacer la “verificación” de todos los datos, es más bien como una forma en la que no recorre individualmente cada una de las palabras, sino que hace un recorrido general del conjunto que se le proporciona, agiliza todo, por lo cual reduce en gran medida la cantidad de tiempo, dicho esto, ahora podemos representar de manera visual lo que nosotros determinamos en ambos modelos, es decir, podemos representar de forma gráfica las diferencias entre ambos métodos y cual puede llegar a ser el mejor de estos dos.

Comparativa entre Bayes y Random Forest en base al tiempo:



Las cantidades que se encuentran en la gráfica están ingresadas en minutos, es decir, nuestro modelo de Bayes se consumió 32 minutos mientras que para el modelo de Random Forest, tenemos una hora y 27 minutos, con lo que claramente podemos ver que en relación al tiempo, el modelo de Bayes es mucho mas optimo.

Comparativa entre efectividad de ambos modelos:



Como podemos observar, el modelo de bayes a nivel global tienen un poco más de efectividad en comparación al Random Forest, pero eso solo aplica para la efectividad global, pero ahora si lo vemos a nivel individual es curioso que el Random Forest tiene mayor efectividad.

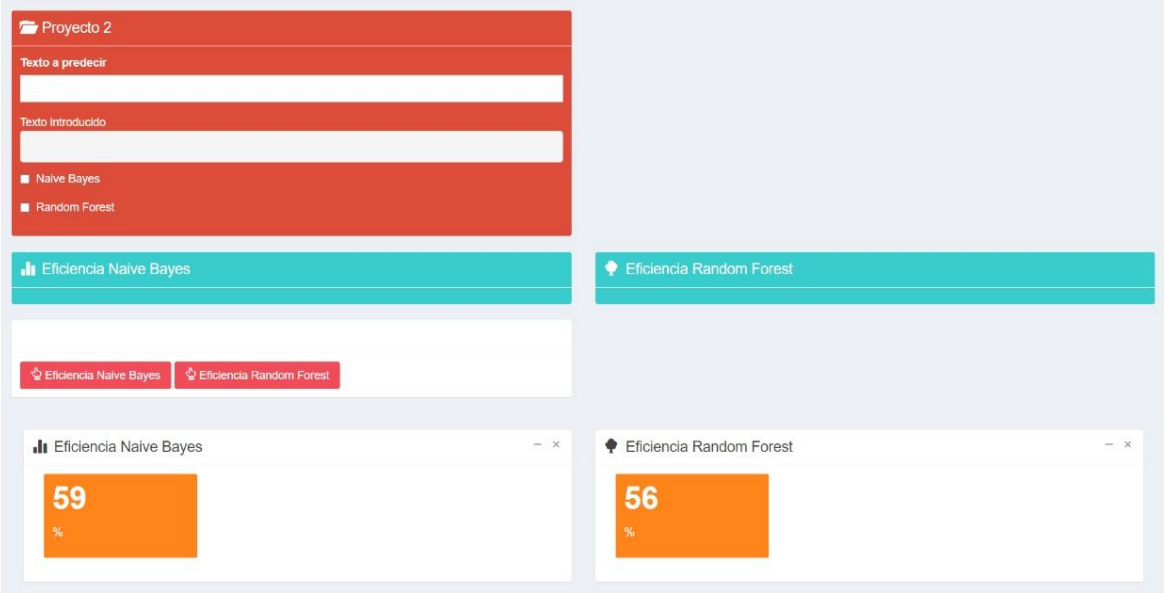
```
Confusion Matrix and Statistics

             Reference
Prediction    Adequate Effective Ineffective
  Adequate        4603      1280        1186
  Effective        722      1316         134
  Ineffective      953       214         622

Overall Statistics

               Accuracy : 0.593
                 95% CI : (0.5838, 0.6022)
    No Information Rate : 0.5692
    P-Value [Acc > NIR] : 2.104e-07

                  Kappa : 0.2687

 Mcnemar's Test P-Value : < 2.2e-16
```

Aca podemos ver las estadísticas completas y la matriz de confusión del modelo al igual que su precisión respecto al algoritmo de Bayes.

```
Confusion Matrix and Statistics

             Reference
Prediction    Adequate Effective Ineffective
  Adequate        6144       133          10
  Effective       2732        61           1
  Ineffective     1910        26           1

Overall Statistics

               Accuracy : 0.5633
                 95% CI : (0.5539, 0.5725)
    No Information Rate : 0.9789
    P-Value [Acc > NIR] : 1

                  Kappa : -0.0014

 Mcnemar's Test P-Value : <2e-16
```

Mientras que acá podemos ver las estadísticas completas y la matriz de confusión del algoritmo realizado con el algoritmo de Random Forest.

Por último, creamos una aplicación la cual se muestra a continuación:



Como podemos observar, lo primero que nos pide la aplicación es que ingresemos un texto, un argumento que queremos evaluar, seguido de eso podemos determinar cuál algoritmo queremos usar, o bien si queremos usar los dos algoritmos (solo con la salvedad que se lleva más tiempo) y una vez escojamos cual queremos solo damos enter y el programa empieza a calcular su efectividad, como extras podemos cerrar la eficiencia de cada uno y podemos minimizar cada uno de ellos, decidimos crear ese modelo porque como se debe de mostrar un porcentaje y catalogarlo, debíamos de hacer algo de cierta forma minimalista.

Decimos usar la herramienta de RShiny porque realmente si bien conllevaba un trabajo más grande en comparación a alguna de las otras herramientas que hubiéramos podido usar, la cantidad de datos que contenía el conjunto nos limita de cierta manera a usar una herramienta que aseguraba usar R, porque sino es muy difícil que con esta complejidad y cantidad de datos pudiéramos hacer que la aplicación pudiera operar en la forma que necesitamos.

Conclusiones

- Los argumentos con mejor escritura, uso de verbos y sobre todo uso correcto de las palabras, tienden a expresar más sentimientos en su escritura, facilitando a los lectores comprender al autor.
- Un argumento demasiado positivo no significa que sea eficiente.
- Que un argumento tenga demasiadas palabras no significa que sea eficiente o de mejor comprensión.
- Para la obtención de una lectura correcta se deben de limpiar los datos lo mejor posible, ya que incluso los excesos de caracteres especiales pueden llegar a provocar particularidades.
- No siempre se puede llegar a tener un hallazgo positivo.
- En nuestros modelos, el algoritmo de Bayes fue mucho más rápido que el modelo de Random Forest, en este caso creemos que para la aplicación y problemática que queremos solucionar es muy importante que sea lo mas rápida posible.

Bibliografía

Cardellino, F. (2021, April 28). *Cómo funcionan los clasificadores Naive Bayes: con ejemplos de código de Python*. freeCodeCamp. Retrieved November 14, 2022, from https://www.freecodecamp.org/espanol/news/como-funcionan-los-clasificadores-naive-bayes-con-ejemplos-de-codigo-de-python/

Medium. (2017, November 4). *Principales Algoritmos utilizados*. Aprende Machine Learning. Retrieved November 14, 2022, from https://www.aprendemachinelearning.com/principales-algoritmos-usados-en-machine-learning/

Moreno, A. (n.d.). *Procesamiento del lenguaje natural ¿qué es? - IIC*. Iic.uam.es. Retrieved November 14, 2022, from https://www.iic.uam.es/inteligencia/que-es-procesamiento-del-lenguaje-natural/

Roman, V. (2019, April 25). *Algoritmos Naive Bayes: Fundamentos e Implementación | by Victor Roman | Ciencia y Datos*. Medium. Retrieved November 14, 2022, from https://medium.com/datos-y-ciencia/algoritmos-naive-bayes-fudamentos-e-implementaci%C3%B3n-4bcb24b307f

Anguiano, E. (29 de Abril de 2009). Naive Bayes Multinomial para Clasificación de Texto Usando un Esquema de Pesado por Clases. México.

Ben Gal I (2007). Redes bayesianas. En Ruggeri F, Kennett RS, Faltin FW (eds.). Página de soporte. Enciclopedia de Estadísticas en Calidad y Confiabilidad. John Wiley & Sons.

Borgelt C, Kruse R (marzo de 2002). Modelos gráficos: métodos de análisis y minería de datos. Chichester, Reino Unido : Wiley.