**Pontificia Universidad Católica del Ecuador**

**Facultad De Ingeniería**

**Carrera de Sistemas de Información**

**TEMA:**

# Análisis de la Percepción de los Clientes de la Empresa Uber en la Red Social Twitter

**AUTOR:**

María Paula Becerra Salas

TRABAJO PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERO DE SISTEMAS DE INFORMACIÓN

**QUITO, diciembre 2022**

**DEDICATORIA**

---

A mi familia, en especial a mis padres Pablo y María Fernanda que han sido mi fuente de apoyo y de confianza en mis momentos más difíciles y también de éxito, este logro es de y para ustedes.

A mi abuela Martha que me enseñó a ser la persona que soy el día de hoy, gracias a tus cualidades magnificas sigues y seguirás siendo fuente de motivación e inspiración. Gracias por creer en mí y seguir cuidando de mi desde el cielo.

Eres una de las personas más importantes de mi vida, gracias por apoyarme y motivarme en todo momento y en cada una de mis metas. Gracias José por formar parte de mi vida y este logro.

**AGRADECIMIENTO**

Mi agradecimiento dirigido para mi familia que han sido parte fundamental de mi vida inspirándome fortaleza en cada etapa de mi vida. A mis padres por su infinito apoyo para luchar por cada uno de mis sueños y metas.

A mi abuela Martha, gracias a tu guianza, amor y apoyo me has hecho crecer como persona y profesionalmente.

Al director de este trabajo Dr. Henry Roa quien con sus años de experiencia profesional y académica me ha sabido orientar a lo largo de este proceso.

A cada uno de mis profesores que formaron parte de mi trayectoria PUCE que han dejado varias enseñanzas más allá de la academia.

Y todas las personas, amigos y compañeros que han formado parte de mi vida durante estos cuatro años.

**RESUMEN**

---

El análisis de sentimientos es unas herramientas más utilizadas en la actualidad para conocer la aceptación del público ante cierto tema, o producto. Gracias a las rede sociales, este proceso se ha vuelto más accesible y fácil de implementar, especialmente Twitter, ya que su dinámica se basa en la publicación de tweets por parte de los usuarios. Este trabajo tiene como propósito analizar la percepción de los usuarios de Uber mediante un conjunto de datos previamente obtenido de la red social Twitter mediante la implementación de una de las metodologías más aceptadas en el medio para proyectos de este tipo CRISP-DM. Las fases descritas por esta metodología fueron implementadas en Python con ayuda de diferentes librerías desde la generación del conjunto de datos, la traducción de los registros para obtener mejores resultados y en el modelo de los algoritmos de aprendizaje de máquina.

El trabajo con lenguaje natural es una tarea complicada por lo que se necesita la división del conjunto de datos en datos de entrenamiento y de prueba. El conjunto de aprendizaje paso por un procesamiento de lenguaje natural para poder clasificar los registros en tres categorías: neutral, positivo y negativo, y de esta manera entrenar de manera eficiente los medios. Por otro lado, ambos subconjuntos se sometieron por procesos de vectorización para poder ser implementado en los algoritmos de clasificación. Los algoritmos seleccionados para las predicciones fueron: Regresión logística, Máquinas de soporte vectorial (SVM) y Naive -Bayes, siendo SVM el algoritmo con mejor rendimiento.

**Palabras clave:** Python, Análisis de sentimientos, Procesamiento de lenguaje natural, Uber, Twitter

**ÍNDICE**

## ÍNDICE DE FIGURAS, GRÁFICOS Y TABLAS

### ÍNDICE DE FIGURAS

## ÍNDICE DE TABLAS

# CAPÍTULO I: INTRODUCCIÓN

## 1. MARCO DE REFERENCIA

### 1.1. JUSTIFICACIÓN

El análisis de sentimientos se ha convertido en una técnica muy utilizada en los últimos años por grandes empresas alrededor del mundo para monitorizar los comentarios que los usuarios realizan a la hora de consumir un producto o servicio con el fin de recibir retroalimentación para mejorar y potenciar sus productos y su imagen corporativa.

Este proceso complejo combina diferentes áreas de conocimiento mediante la aplicación de minería de datos, procesamiento del lenguaje natural y algoritmos de aprendizaje de máquina con el fin de que la máquina sea capaz de analizar el lenguaje humano considerando variaciones gramaticales reflejadas en la jerga e inclusive en las faltas ortográficas. Además, el análisis de sentimientos no solo valora las opiniones de manera positiva, negativa o neutra, sino también mediante la detección de tendencias.

En este contexto, se utilizará la red social Twitter como fuente de datos para este estudio, ya que con el paso del tiempo se ha posicionado como uno de los medios más grandes para la difusión de contenido y opinión en el internet. En Ecuador se ha implementado el análisis de sentimientos para conocer la tendencia política en elección presidenciales, así también para la clasificación de comentarios xenófobos hacía inmigrantes en el país, pero no existen una implementación con respecto a satisfacción de usuarios de plataformas que ofrecen servicios tecnológicos enfocados al transporte como Uber.

## 1.2. Planteamiento del problema

Uber es una de las empresas tecnológicas más importantes alrededor del mundo, ya que desde el año de su lanzamiento ha revolucionado la forma en la cual sus usuarios se transportan a sus destinos día a día. En 2017, se introduce en el mercado ecuatoriano y se posiciona rápidamente como una de las opciones más usadas por los usuarios cuando necesitan un automóvil para llegar a su destino por varias razones, entre ellas: la facilidad de uso, la comodidad de pagos y de precios del servicio, etc. A pesar de que las aplicaciones que ofrecen este tipo de servicio son ilegales, a la fecha existen más de 20.000 usuarios activos en el país, mostrando cifras bastante representativas en cuanto a la movilidad privada del país.

El servicio al cliente de Uber se basa en chats en línea e inclusive en chats automatizados para atender a requerimientos comunes entre los usuarios de su plataforma, por lo que el uso de redes sociales para expresar opiniones respecto al servicio proporcionado por los socios conductores es muy común. De hecho, en los últimos años Twitter se ha catalogado como una de las redes sociales más usadas en Ecuador, según el Digital 2021 Global Review Report publicado por We are Social y Hootsuite, existen más de 1.15 millones de personas dentro de la red social y se ha convertido en la red social por excelencia para publicar opiniones referentes a política, socioambiental, inclusive de interés ciudadano como lo son las quejas públicas realizadas hacía ciertas empresas por sus productos o servicios.

Por este motivo, Twitter es considerada una de las mejores fuentes para la extracción de datos en la actualidad y mediante el uso de minería de datos se detectan patrones en el comportamiento de usuarios dentro de la red social en diferentes contextos. Mediante el análisis de sentimientos en la red social Twitter se podrá conocer la imagen corporativa de Uber en el país de manera integral implementando soluciones tecnológicas basadas en algoritmos, procesamiento de lenguaje natural y medidas de desempeño.

### 1.3. Objetivo General

Implementar técnicas basadas en el Aprendizaje de máquina y minería de datos con la finalidad de realizar análisis de sentimientos y conocer la influencia de los usuarios a la imagen corporativa de Uber.

### 1.4. Objetivos Específicos

- Extraer y construir un conjunto de datos de la red social Twitter mediante el uso de herramientas de software.
- Diseñar una estrategia para el análisis de datos extraídos referentes al tema, procurando realizar este proceso de manera óptima.
- Implementar modelos de análisis de datos para la clasificación de los registros en función de los sentimientos expresados por los usuarios de la plataforma.

### 1.5. Antecedentes

El análisis de sentimientos es un área de estudio que ha tomado relevancia en los últimos años. En primer lugar, en septiembre del 2018 un artículo es publicado por la Universidad Politécnica de Valencia “ELiRF-UPV en TASS 2018: Análisis de Sentimientos en Twitter basado en Aprendizaje Profundo” por José-Ángel Gonzáles, Lluís-f. Hurtado, Ferran Pla.

En este artículo se describe la participación del grupo de investigación ELiRF de la Universidad Politécnica de Valencia, donde se pueden identificar las aproximaciones basadas en Deep Learning utilizadas dentro del Procesamiento del Lenguaje Natural dentro de la plataforma Twitter. Por otra parte, se puede comprender a fondo la clasificación de los diferentes registros denominados tanto en la red como en la investigación como tweet en una escala de cuatro niveles de intensidad (Negativo, Neutral, No Neutral y Positivo) y las diferentes dificultades a la hora de la extracción y limpieza de datos como son: el lenguaje informal, errores ortográficos, utilización de términos especiales como los “emojis” y la multilingualidad, representando un reto para el

Congreso que el grupo de investigación formó parte en año descrito. Un aspecto importante dentro de esta investigación es la utilización de tres diferentes corpus (datos a utilizar) para validar el funcionamiento de los algoritmos en español con los corpus InterTASS-ES (España), con una partición de entrenamiento de 1008 registros, un conjunto de datos de validación de 506 muestras y una de testeo con 1920 registros, y InterTASS-CR (Costa Rica), con una partición de entrenamiento de 800 registros, una de validación de 300 registros y una de prueba de 1233 registros, en último lugar tenemos el corpus InterTASS-PE (Perú), con 1000 registro de entrenamiento, 500 de validación y 1428 registros para el testeo.

Enfocándonos al contexto del país, existe la investigación denominada “Influencia de redes sociales en el análisis de sentimiento aplicado a la situación política del Ecuador” publicado por Estevan Gómez-Torres, Roger Jaimes, Orlando Hidalgo y Sergio Luján-Mora. En la presente investigación se toma como referencia el último proceso electoral suscitado en el año 2018 en Ecuador con todas sus fases (antes, durante y después de las elecciones), este artículo presenta una metodología de trabajo basada en Stanford NLP y el uso de diccionarios de palabras para que el algoritmo considere las valoraciones como positivas o negativas. Este estudio determino que los resultados obtenidos dentro del estudio no coinciden con el porcentaje de aceptación, el resultado de las elecciones y el declive que ha tenido la popularidad de los candidatos de dichas elecciones a comparación de candidatos anteriores de su mismo partido político.

## CAPÍTULO II: FUNDAMENTACIÓN TEÓRICA

# 2. Marco Teórico

## 2.1. Generalidades

En su investigación, Rosenbrock, Trossero y Pascal (2021) afirman que el análisis de sentimientos estudia diferentes textos con el fin de obtener opiniones y sentimientos de los autores del texto analizado. Este estudio se puede enfocar a un producto, persona relevante del medio, organizaciones, tema tendencia, entre otros. Hay que considerar que el objeto d estudio tiene características y propiedades relevantes para el proceso.

El análisis de datos es parte fundamental dentro de este estudio, como se puede observar en la Figura 1 se describen los diferentes métodos que existen


*Ilustración 1 Técnicas de minería de datos.*

## 2.2. Minería de texto

Proceso por el cual se analizan textos de diferentes fuentes con el objetivo de obtener conceptos clave, relaciones y tendencias existentes sin necesidad de conocer con exactitud las palabras que han sido utilizadas para expresar dichos pensamientos. Al utilizar análisis de datos junto a minería de texto podemos realizar una búsqueda más profunda a nivel de términos, palabras inclusive temas de textos tanto estructurados como no estructurados.

En este tipo de minería es común empelar: algoritmos A priori, algoritmos de Bayes, entre otros; lo que permite la clasificación, agrupamiento y extracción de atributos de los datos.

## 2.3. Análisis de sentimientos

Permite el entendimiento del estado de ánimo de los usuarios respecto a un tema, situación o producto. Este procedimiento se basa en la construcción de modelos para clasificar las diferentes opiniones aportadas por usuarios. Gracias a esta construcción y al uso de Machine Learning y Procesamiento del Lenguaje Natural, es posible categorizar los sentimientos en tres categorías principales: positivo, neutral o negativo. Estas opiniones plasmadas en texto pueden ser divididas en: oraciones, frases y tokens los que permiten identificar el sentimiento que compone a la frase y la asignación de una categoría prevista anteriormente.

## 2.4. Conjunto de datos

De su término en inglés, data set, es considerado la materia prima de cualquier algoritmo de predicción, compuesto por instancias o en términos más sencillos, características o propiedades de dichos datos. Su función principal radica en entrenar a los sistemas para la detección de patrones.

Además, se puede categorizar en tres tipos: secuencial, este permite el almacenamiento de registros de manera consecutiva, por ejemplo, en orden alfabético. El segundo tipo, es el

particionamiento, que contiene un directorio que almacena la dirección de cada miembro del sistema operativo y, por último “VSAM secuenciado de claves de método de acceso que contiene datos de almacenamiento virtual” ()

## 2.5. Uber

Uber comienza como una empresa emergente en 2009 como la materialización de la idea de Travis Kalanick y Garrett Camp con el objetivo de llamar a un carro privado con sólo un botón. En la actualidad esta empresa se basa en el modelo de negocios multilateral, es decir, Uber conecta a un grupo de participantes desempeñando el rol de moderador para que se pueda dar con éxito el servicio. En la actualidad, Uber se encuentra en más de 80 países y se considera como una de las formas más efectivas de movilidad urbana.

Por otro lado, la imagen corporativa se define como el conjunto de creencias y percepciones que el público tiene sobre una empresa o marca. En el caso de Uber, su imagen se basa en la experiencia que ofrece en sus diferentes servicios basados en el transporte.

## 2.6. Herramientas y librerías

### 2.6.1. Google Colab

Google Colaboratory es un producto gratuito de Google Research que permite escribir y ejecutar código Python en diferentes navegadores, técnicamente, Google Colab proporciona un entorno Jupyter que no requiere configuración. Esta herramienta es adecuada para la ejecución de tares de aprendizaje automático y análisis de datos. (Google Colab, s. f.)

### 2.6.2. Snscrape

Es una herramienta “scraper” para servicios de redes sociales (SNS). Soporta diferentes redes sociales como Facebook, Instagram y Twitter. Para su implementación en Python es necesario

la instalación de librería disponible en GitHub y a diferencia de otras herramientas con el mismo funcionamiento, Snscrape no necesita la API de desarrollador de Twitter. (GitHub,2022)

### 2.6.3. Pandas

Es una librería de Python distribuida baja la licencia Berkeley Software Distribution (BSD) que se enfoca en la manipulación y análisis de datos, proporciona estructura de datos y operaciones para la manipulación de datos. (Pandas, s.f)

### 2.6.4. NLTK

De las siglas Natural Language Toolkit, desarrollado en Python para trabajar con datos de leguaje humano, gracias a sus más de 50 corpus y recursos léxicos, que permiten el procesamiento de texto mediante tokenización, clasificación, etiquetado, entro otros. (NLTK, s.f.)

### 2.6.5. Scikit-Learn

Librería de Python que nos proporciona acceso a diferentes algoritmos de aprendizaje de máquina supervisados y no supervisados en sus versiones más estables para ser utilizados por usuarios de lenguajes de alto nivel (Scikit-Learn: Descubre la biblioteca de Python dedicada al Machine Learning, 2022)

### 2.6.6. RegEx

“Es un módulo de Python que permite el manejo de expresiones regulares que contienen símbolos especiales que se refieren a la palabra que se ha relacionado con una subexpresión específica; lo que nos permite realizar procesos como la limpieza de cadenas” (Romero,2021)

### 2.6.7. MatplotLib

Librería la cual permite la visualización estática, animada e interactiva en Python. (Matplotlib — Visualization with Python, s. f.)

### 2.6.8. Googletrans

Es una librería gratuita e ilimitada de Python la cual permite implementar la API de Google Translate dentro de proyectos de programación, llamando a sus métodos de detección y traducción. (googletrans, 2020)

## 2.7. Modelo supervisado

El aprendizaje supervisado permite el modelamiento de datos y predecir un valor en particular. En este aprendizaje se tiene múltiples variables independientes que son usados como entrenamiento con el fin de poder predecir el valor correcto de la variable dependiente. La ventaja del aprendizaje supervisado es que puede ser medido mediante la comparación de los resultados con los datos originales.

Este modelo cuenta con algunos algoritmos principales los cuales son: arboles de decisión, los cuales proporcionan escalabilidad y transparencia al modelo y Naives Bayes utilizado en problemas de clasificación binaria y multiclase.

En la Figura 6 se puede observar el procesamiento supervisado.

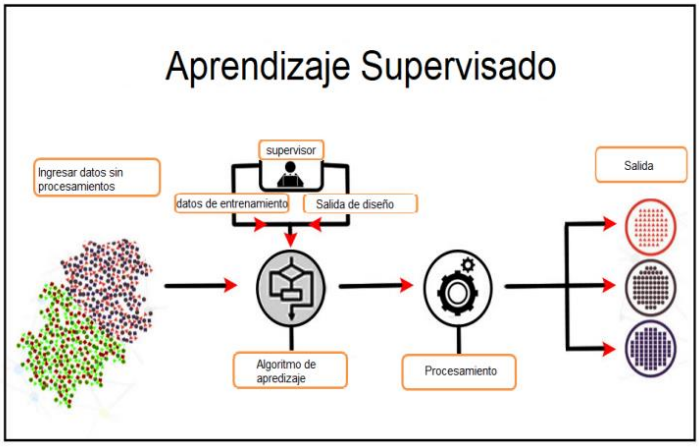



*Ilustración 2 Aprendizaje Supervisado*

## 2.8. Análisis de datos predictiva

El análisis de datos predictiva consiste en predecir el comportamiento de un algoritmo en relación con una o más variables. Esto se implementa mediante el descubrimiento de normas de clasificación o predicción basado en los resultados que se pueden llegar a tener en el futuro. En otras palabras, el análisis de datos predictivo es la extracción de información de los datos trabajados para predecir patrones o tendencias (Páez, Monroy, 2020. p 28)

## 2.9. Modelos de predicción

Los modelos predictivos son técnicas estadísticas que es útil para predecir el comportamiento futuro de las variables. Dentro del análisis de datos, los modelos de predicción analizan los datos históricos y actuales permitiendo la predicción de las variables de salida.

Hay que considerar que los modelos predictivos tienen que ser monitoreados regularmente e incorporar cambios para el correcto procesamiento de los datos. A pesar de ellos, la mayoría de estos modelos trabajan de manera eficiente y óptima mientras se aporte un conjunto de datos para su entrenamiento.

### 2.9.1. Regresión

Estos algoritmos son utilizados para predecir valores en una o más variables continuas. El algoritmo de regresión lineal tiene como objetivo generar un modelo de regresión que permita explicar la relación entre dos variables X como variable independiente e Y como variable dependiente.

### 2.9.2. Regresión logística

Algoritmo de aprendizaje supervisado que clasifica de manera binaria las probabilidades de que ocurra un evento o salida. Este modelo da como resultado una salida dicotómica, esto quiere decir que se limita a solo dos opciones: si/no, 0/1 o verdadero/ falso. Según Kanade (2022), el

funcionamiento de este algoritmo radica en el análisis de las relaciones entre una o más variables independientes y clasificar los datos en clases discretas.

La regresión logística emplea una función logística sigmoide que permite el mapeo de predicciones y sus probabilidades, su característica principal es que su forma gráfica es una curva en forma de S que convierte cualquier valor real de Y entre [0, 1].

La ecuación del algoritmo es la siguiente:

$$y = \frac{e^{b_0 + b_1 X}}{1 + e^{(b_0 + b_1 X)}}$$

Donde:

X= valor de entrada.

y= output predicho.

b0= bias o término de intersección.

b1= coeficiente de la entrada.

### 2.9.3. Naive Bayes

Este algoritmo clasificador probabilístico basado en el teorema de Bayes acepta que la presencia de una característica específica en una clase no está relacionada con la presencia de otra característica. Además de ser un algoritmo sencillo de implementar es muy usado ya que tiene un alto rendimiento a la hora de clasificar elementos de un conjunto de datos. El teorema de Bayes es útil para calcular la probabilidad mediante la siguiente formula.

$$P(f|c)\ ni(d)\ \mathrm{PNB(c|d)}\ = \frac{\mathrm{P(c)} \sum \mathrm{i = 1 \ldots m}}{\mathrm{P(d)}}$$

### 2.9.4. Máquinas de Soporte Vectorial (SVM)

Este algoritmo puede ser usado tanto para clasificación como regresión, pero es usada en su mayoría para problemas de clasificación. En SVM, se grafica cada ítem como un punto en un espacio de n- dimensiones (definido por el número de variables con las que se trabaje) en el cual para cada variable existe un valor particular de coordenadas. El funcionamiento básico del clasificador de este algoritmo se fundamenta en encontrar un hiperplano intentando que la separación de las clases sea la máxima posible. Para esto, se utiliza dos vectores de soporte denominados S1, S2 considerados como atributos en un espacio dimensional como se muestra en la figura.

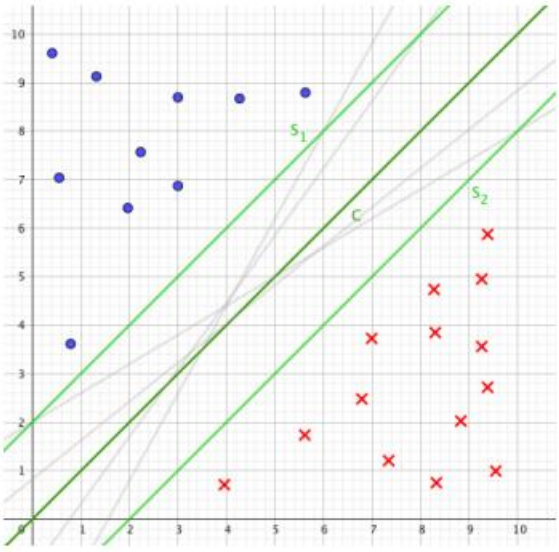



*Ilustración 3 Máquinas de soporte vectorial para dos clases*

## 2.10. Matriz de confusión

La matriz de confusión es parte fundamental dentro del aprendizaje automático ya que contiene información sobre las clases reales y predichas realizadas por el algoritmo que se esté analizando. Esta matriz posee dos dimensiones, valores actuales (verdadero y falso) y

valores predichos (positivo y negativo), que dan como resultado cuatro posibles combinaciones que se muestra en la siguiente imagen

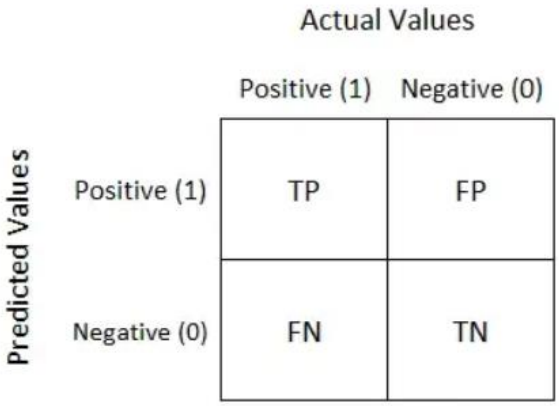



*Ilustración 4 Matriz de confusión*

A continuación, se detalla las abreviaciones de la matriz de confusión

- **True Positive (TP):** La predicción es positiva y verdadera.
- **True Negative (TN):** La predicción es negativa y es verdadera.
- **False positive (FP):** Es catalogado como error tipo 1 y la predicción es positiva, pero es falsa.
- **False negative (FN):** es catalogado como error tipo 2 y la predicción es negativa y es falsa.

### 2.11. Exactitud

Del término en inglés Accuracy, es la proporción del número total de predicciones que fueron correctas.

$$Accuracy = \frac{\sum_{i=1}^{n} N_{ii}}{\sum_{i=1}^{n} \sum_{j=1}^{n} N_{ij}}$$

### 2.12. Precisión

Permite medir la exactitud de un algoritmo siempre y cuando este haya predicho una clase en específico. (Romero,2021)

$$Precisión = \frac{TP}{TP + FP}$$

### 2.13. Recall

Es la medida que sirve para conocer el desempeño del algoritmo tomando en cuenta todas las clases positiva y cuantas fueron predichas correctamente. (Romero,2021)

$$Recall = \frac{TP}{TP + FN}$$

### 2.14. F1- score

Esta medida de desempeño utiliza recall y precisión al mismo tiempo. F1-score usa la media armónica lo que quiere decir que utiliza los valores recíprocos de las variables. Su fórmula utilizando las medidas de desempeño descritas anteriormente es

$$F - score = \frac{2 + Precisión * Recall}{Precisión_i + Recall_i}$$

## CAPÍTULO III: PROCESO DE CIENCIA DE DATOS

### 3. Metodología el análisis de datos

Cross Industry Standard Process for Data Mining o en sus siglas CRISP-DM es una metodología creada en 1996, con el objetivo de ser implementado en proyectos de Minería de Datos, su ventaja consiste en la flexibilidad del modelo y se puede personalizar con facilidad dependiendo de las necesidades del estudio. Consiste en seis fases que pueden tener iteraciones s cíclicas dependiendo de las necesidades del desarrollador. Las etapas de CRISP-DM son las siguientes: Entendimiento del negocio, Entendimiento de los datos, Preparación de los datos, Modelado, Evaluación y Despliegue. Para este proyecto se omitirán la fase del despliegue, ya que no es una fase aplicable dentro de este análisis.

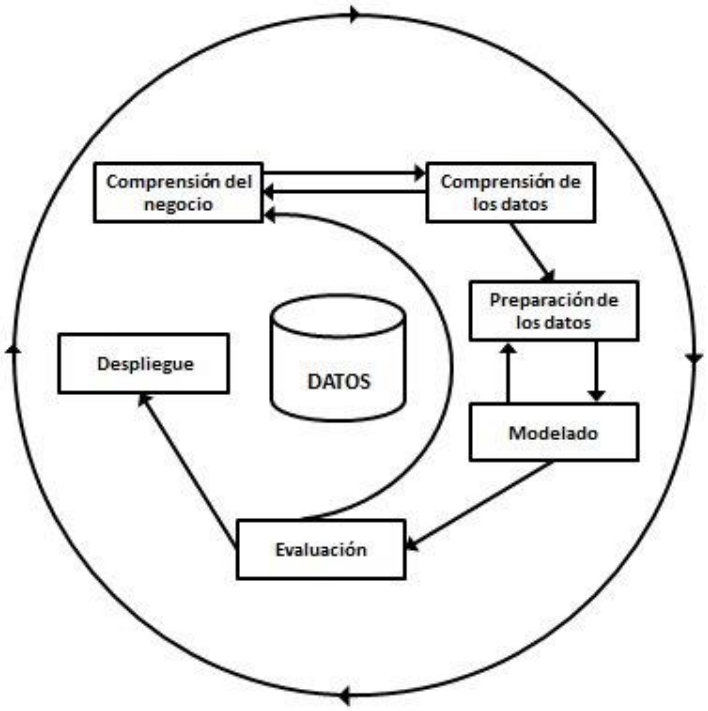



*Ilustración 5 Metodología CRISP-DM*

### 3.1. Entendimiento de Negocio

Es la fase enfocada en la compresión de los objetivos del proyecto desde la visión del negocio. Dentro de esta fase existe una serie de pasos que se deben cumplir para poder comprender en su totalidad el problema. (Galán Cortina,2015).

- **Determinar los objetivos del negocio**: En esta tarea se define el problema a resolver, la necesidad del análisis de datos y los criterios con los cuales se considera exitoso el análisis. Los problemas van desde detecciones de fraudes hasta el éxito de una campaña publicitaria. En cuanto a los criterios pueden ser de tipo cualitativo, en el que el experto del área califica el resultado del proceso, o de tipo cuantitativo, mediante medidas de rendimiento como el número de predicciones correctas de algoritmos empleados.
- **Evaluación de la situación**: En esta tarea se hace una investigación sobre la situación antes del proceso de minería de datos. Es decir, se reconocen los requisitos del problema a nivel de negocio y técnico.
- **Determinar los objetivos de la minería de datos:** Se representan los objetivos del negocio en metas para el proyecto.
- **Realizar el plan del proyecto:** Describe los pasos a seguir y las técnicas a emplear en cada una de ellas.

### 3.2. Entendimiento de datos

En esta fase se realiza la recolección de los datos con la finalidad de tener un acercamiento, verificar la calidad de los datos y establecer relaciones que permitan trabajar conforme a los objetivos del proyecto. Esta fase requiere tanto de esfuerzos como tiempo ya que este análisis puede llegar a ser complejo. Las tareas de esta fase se definen (Galán,2015)

- **Recolectar datos iniciales**: En esta tarea se realiza la recopilación de datos y su evaluación para el procesamiento en fases posteriores. En esta tarea tenemos como resultado listas de datos adquiridos y las técnicas utilizadas para su recolección.
- **Descripción de los datos:** En esta tarea tenemos que detallar a fondo las características de los datos obtenidos como el volumen de datos, identificación, formato, entre otros.
- **Exploración de datos:** Realizadas las tareas anteriores, en esta tarea buscamos una estructura por lo que se rijan los datos. Esto se realiza mediante pruebas estadísticas que dan como resultados diferentes gráficos y tablas de frecuencia.
- **Verificar la calidad de los datos**: En esta tarea determinamos la consistencia de los datos. De esta manera, garantizamos que el conjunto de datos sea idóneo para fases posteriores.

### 3.3. Preparación de los datos

Dentro de esta fase se adaptan los datos para el uso en las diferentes técnicas que van a ser seleccionadas y utilizadas posteriormente en la fase de modelado. La preparación de datos incluye: selección de datos, limpieza de datos, generación de variables adicionales y cambios de formato.

- **Selección de datos:** Creación de un subconjunto de datos considerando los criterios definidos en fases anteriores.
- **Limpieza de datos:** Es una tarea bastante demandante de suma importancia para el proceso de minería de datos, ya que existen un sin número de técnicas que pueden ser aplicadas para mejorar la calidad de los datos para su posterior uso. Algunas de las técnicas que se pueden emplear son la discretización de números, completar o eliminar datos nulos o faltantes, entre otros.

- **Construcción de datos**: En esta tarea se implementan diferentes técnicas para la creación de nuevos atributos, integración o transformación de registros ya existentes.
- **Integración de datos:** Implica la creación de estructuras con los datos seleccionados en las tareas anteriores. Por ejemplo, la fusión de tablas o campos.
- **Formateo de datos:** En esta tarea realizamos las transformaciones necesarias en cuanto a sintaxis sin afectar el significado para poder implementar los diferentes algoritmos en la fase de modelado.

### 3.4. Modelado

En esta fase de la metodología se seleccionan las técnicas adecuadas para el proyecto de análisis de datos. Según Galán Cortina (2015), se deben considerar los siguientes aspectos:

- Ajustarse con la problemática planteada.
- Disponer de datos adecuados.
- Implementación del modelo considerando los tiempos del proyecto.
- Conocimiento del funcionamiento de la técnica.

De igual modo, se deben determinar métricas de evaluación que permitan conocer el rendimiento del modelo. Una vez seleccionados los modelos se procede a la generación de los modelos, para esto se deben considerar las características de los datos y la precisión que se desea que adquiera el modelo. A continuación, se procede a detallar las tareas de esta fase:

- **Elección de técnicas de modelado**: En esta tarea se debe considerar el objetivo principal del proyecto. Por ejemplo, para un problema de clasificación un algoritmo de árboles de decisiones puedes ser viable, mientras que para un problema de predicción se puede escoger un algoritmo de redes neuronales.

- **Generar el plan de prueba:** Se genera un procedimiento en el cual se debe comprobar la calidad y eficiencia de los modelos seleccionados.
- **Construir los modelos:** Se ejecutan los algoritmos seleccionados con los datos que fueron procesados en fases anteriores. Todas las técnicas tienen un conjunto de parámetros que determinan las características del algoritmo.
- **Evaluar el modelo**: En esta fase se interpreta si las técnicas cumplen con los criterios de éxito preestablecidos.

### 3.5. Evaluación

Esta fase describe como trabajar en cuanto a la evaluación de los diferentes modelos aplicados en la fase anterior, hay que considerar que el rendimiento de los modelos aplica solamente para el conjunto de datos con el que se los implemento. A continuación, se detalla las tareas a completar en esta fase:

- **Evaluar los resultados:** Esta tarea vas más allá de los criterios de evaluación como exactitud del modelo. Aquí se evalúa el modelo y si ayuda con el cumplimiento de los objetivos establecidos en la primera fase de CRISP-DM.
- **Revisar el proyecto**: En esta tarea se califica el proceso en todas sus fases con la finalidad de buscar puntos a mejorar.
- **Determinar los próximos pasos:** En esta tarea se decide si los resultados son satisfactorios para pasar a la siguiente fase o si es necesario retrocede a alguna fase del proyecto para mejorar algunos aspectos con la finalidad de mejorar los resultados del proceso.

# CAPÍTULO IV: DESARROLLO DEL ANÁLISIS

En este capítulo se aplicará cada una de las fases de CRISP-DM detalladas en el capítulo anterior

## 4.1. Compresión del negocio

### 4.1.1. Determinar los objetivos del negocio

El objetivo de la aplicación de este proyecto es realizar predicciones fiables a partir del conjunto de datos que se obtenga de la aplicación Twitter con la finalidad de conocer a fondo la perspectiva de los usuarios de la compañía tecnológica Uber.

### 4.1.2. Evaluación de la situación

Al no contar con una base de datos proporcionada por la empresa, se debe extraer los datos de las redes sociales, en este caso, de Twitter ya que existen diversas herramientas que permiten la extracción de los tweets mediante programación. En cuanto a costes y beneficios para la organización, este proyecto no aportará de manera económica a Uber ya que se está realizando el análisis de manera independiente, pero si puede suponer de gran ayuda para conocer de mejor manera la imagen de la compañía dentro de la red social Twitter.

### 4.1.3. Determinar los objetivos de la minería de datos

Identificar los algoritmos más factibles para el análisis de sentimientos en Twitter con respecto a la empresa tecnológica Uber.

### 4.1.4. Criterios de éxito del proyecto

Dentro del análisis de datos se establece como éxito que los modelos implementados sean capaces de identificar el sentimiento de el texto analizado en un 80% de precisión y exactitud

### 4.1.5. Realizar plan del proyecto

*Tabla 1 Plan de trabajo del proyecto*

| | Agosto | | | | Septiembre | | | | Octubre | | | | Noviembre | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4. Desarrollo de trabajo | | | | | | | | | | | | | | | | |
| 4.1. Procesos de extracción y muestreo | | | | | | | | | | | | | | | | |
| 4.2. Entrenamiento | | | | | | | | | | | | | | | | |
| 4.3. Evaluación de rendimiento | | | | | | | | | | | | | | | | |
| 5. Análisis de resultados | | | | | | | | | | | | | | | | |
| 6. Conclusiones y recomendaciones | | | | | | | | | | | | | | | | |

## 4.2. Compresión de datos

### 4.2.1. Recolectar datos iniciales

Se realiza la extracción de tweets de la red social Twitter, que contengan en su contenido temas relacionados con Uber Ecuador. Para la descarga de la data se realizó un Notebook con Python en el cual se utilizó la librería snscrape.

Gracias a snscrape se obtuvieron inicialmente 5000 tweets mediante una consulta la cual tuvo los siguientes parámetros.

- **Texto dentro de la aplicación:** Uber Ecuador
- **Rango de fecha**s: 2010-01-01 a 2022-01-01
- **Límite de tweets:** 5000

Como resultado se obtiene un data set que posteriormente fue almacenado dentro de un archivo .csv a manera de evidencia de la terminación de esta tarea denominado tweets.csv y para su uso en tareas posteriores.

### 4.2.2. Descripción de datos

Los datos obtenidos mediante la consulta previamente realizada son:

- **Date:** Hace referencia a la fecha en la cual fue publicado el contenido dentro de la red social. Este dato es de tipo texto el cual contiene la fecha el formato yy/mm/dd y además la hora en formato hh:mm: ss de la zona horaria en la cual se está trabajando.
- **User:** Hace referencia al usuario que posteo el tweet que fue recolectado por el query. Es un dato de tipo texto en el cual se registra únicamente el nombre de usuario de Twitter sin ningún carácter especial
- **Tweet:** Hace referencia al contenido del tweet como tal. Este dato es de tipo texto el cual es extraído tal y como se observa dentro de la red social.

### 4.2.3. Verificar la calidad de datos

Después de la exploración inicial de los datos podemos concluir que estos son completos. En este caso, cumplen con las características para la implementación de los modelos de análisis de datos, al tratarse de datos tipo texto y en español se ha considerado su codificación en formato UTF-8 para que no existan problemas en la visualización de caracteres del idioma español en las diferentes herramientas de visualización de archivos .csv, como es el caso de Excel, bloc de notas, etc.

## 4.3. Preparación de los datos

### 4.3.1. Selección de datos

En cuanto a registros, se van a utilizar todos los registros dentro del archivo .csv previamente obtenido, ya que son datos obtenidos exclusivamente para este proyecto. El número de registros ha sido seleccionado de acuerdo con la interacción que se tiene dentro de la red social Twitter,

en promedio se detectó 3 tweets por día. Sin embargo, se ha detectado que el único campo valido de este conjunto de datos es el campo tweet ya que contiene el texto que se pretende analizar.

### 4.3.2. Limpieza de datos

Esta tarea se la ha realizado mediante un Notebook de Python en el cual se implementaron las librerías pandas para el manejo del conjunto de datos y de la librería re, las siglas de expresiones regulares, que nos permitirán manejar de mejor manera el preprocesamiento.

Para comenzar con este proceso se importa el archivo tweets.csv y se almacena en una estructura DataFrame de pandas, con la librería de expresiones regulares parametrizamos los caracteres que queremos eliminar del texto analizado. En este caso, se quiere omitir cualquier carácter que no sean letras del abecedario y números, al detectar estos caracteres por defecto se va a insertar un espacio por lo que también parametrizamos que si se encuentra más de un espacio en blanco dentro del texto se reemplace por un solo espacio. Las cadenas de texto van a ser analizadas una por una y almacenadas dentro de un data set nuevo denominado processed_tweets, una vez finalizada la limpieza de los tweets, procedemos a verificar mediante la librería de pandas los datos repetidos dentro del DataFrame y eliminarlos si es necesario. En este caso, se han eliminado 241 registros quedando como resultado 4759 registros limpios.

Por último, se almacena el resultado de este proceso en un archivo .csv denominadoo_limpia.csv

### 4.3.3. Integración de datos

En este proceso se han integrado diferentes campos que servirán para facilitar la implementación del modelo de datos. Primero, se ha realizado la traducción de los tweets al inglés, con la finalidad de poder hacer uso de la librería stopwords de Python, a pesar de que exista repositorios para el idioma español no consideran mucho las variaciones de lenguaje que puede existir entre usuarios de la plataforma por razones culturales, sociales, entre otras. Por lo que se hará uso de los

repositorios en el idioma inglés ya que tiene más repositorios de datos de lenguaje lo que nos permitirá aumentar el rendimiento de los modelos.

Reanudando el tema, se crea un Notebook de Python para hacer uso de la librería googletrans que nos permitirá realizar la traducción del texto. Esta librería además de ser importada debe ser inicializada para que se consuma la API de Google Translator dentro del notebook, una vez inicializada se importan los datos previamente procesados y se parametriza el idioma de entrada (español) y el de salida (inglés). A continuación, mediante una función vamos a definir un contador que nos permitirá conocer el número de tweets que fueron traducidos exitosamente y si por alguna razón la parametrización anterior no funciona, dentro de esta función se agregan 6 servidores de diferentes países hispanohablantes para que se pueda reintentar la traducción del texto. Por último, los registros traducidos son almacenados dentro de un archivo .csv llamado tweets_traducidos. csv

A continuación, se realiza el procesamiento del lenguaje natural, para esto se crea un diccionario mediante la librería stopwords en inglés y mediante la librería TextBlob se categorizará los registros en tres categorías: positivo, neutral y negativo. Esta categorización se da gracias a que esta librería define una polaridad al texto analizado dentro de un rango de -1 a 1, siendo cero considerado como sentimiento neutral, si el valor incrementa se considera como positivo y casi contrario, si el valor es menor que cero se considerara como negativo. Tanto la polaridad como la categorización del sentimiento serán almacenadas en nuevas columnas llamadas polarity y sentiment respectivamente, además almacenadas en un archivo csv llamado data_label.csv.

Los registros almacenados dentro de polarity son de tipo flotante dentro del rango de -1;1, mientras que los datos registrados en la columna sentiment son de tipo texto.

#### 4.3.4. Formateo de datos

Dentro de esta tarea se emplean dos técnicas de vectorización para poder trabajar de manera correcta con los modelos, ya que los modelos de predicción no aceptan como entrada registros tipo texto. Ambas técnicas provienen de la librería skcikit-learn, pero su diferencia radica en la que transformación del texto a vectores.

La primera técnica CountVectorizer nos permite transformar el texto deseado a un vector considerando la frecuencia de las palabras dentro del corpus. Por otro lado, la técnica TFIDF realiza el mismo procedimiento que la técnica y agrega importancia a las palabras, de esta manera se puede eliminar las menos relevantes reduciendo así la dimensión del vector y mejorar el rendimiento del modelo.

## 4.4. Modelado

### 4.4.1. Elección de técnicas

Existen diferentes estudios relacionados con el análisis de sentimientos en los cuales se pueden contemplar ciertos modelos usados por excelencia para la implementación de análisis de sentimientos. En esta tarea se ha considerado los trabajos de Raúl Romero publicado en el 2021, en el cual se emplea algoritmos de Regresión Logística, SVM y Naive Bayes, y el trabajo de Paéz y Monroy en el cual se emplean las técnicas Random Forest, Naive Bayes y SMV. Gracias a esta investigación, se determina que para este proyecto se empleará las siguientes técnicas: Maquinas de soporte vectorial (SVM), Naive Bayes y Regresión logística.

### 4.4.2. Generar plan de pruebas

Para probar el rendimiento y la calidad del modelo se utilizarán las siguientes métricas: precisión, recall y f1-score. De igual manera, se implementará la matriz de confusión en cada una de las técnicas con el fin de conocer los aciertos y errores del modelo a la hora de predecir el sentimiento de los datos implementados. Por otro lado, para la implementación del modelo se debe dividir el

conjunto de datos anteriormente procesado en dos grupos: por un lado, el conjunto de datos de entrenamiento que nos permite entrenar el modelo, y un segundo conjunto denominado conjunto de evaluación, el cual permite realizar las pruebas y verificar el rendimiento del modelo. Para esta división generalmente se utiliza el principio de Pareto, 80% de los datos para entrenamiento y el 20% restante de los datos para evaluación.

### 4.4.3. Construcción de los modelos

A continuación, se realiza la implementación de los modelos elegidos con los datos de entramiento. En esta sección se detallará la parametrización de los modelos, sus salidas y sus descripciones.

#### 4.4.3.1. Regresión logística

Mediante la implementación de la librería Scikit-learn se empleó el algoritmo de regresión logística para la creación del modelo. En este caso se tiene como variable a predecir el sentimiento y como variable de entrada los registros de tweets previamente vectorizados. En cuanto a parámetros, se ha trabajado con los parámetros preestablecidos por la librería. Como resultado, tenemos la siguiente tabla con las métricas de evaluación de rendimiento del modelo, una vez implementado el conjunto de datos de evaluación.

*Tabla 2 Métricas de evaluación regresión logística*

| | **Precisión** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| **Negative** | 0.00 | 0.00 | 0.00 | 1 |
| **Neutral** | 0.96 | 1 | 0.92 | 18 |
| **Positive** | 0.00 | 0.00 | 0.00 | 2 |
| **Accuracy** | | | **0.86** | **21** |
| **Macro Avg** | 0.29 | 0.33 | 0.31 | 21 |

| | | | | |
|---|---|---|---|---|
| **Weigthed Avg** | 0.73 | 0.86 | 0.79 | 21 |

De igual manera, se genera la matriz de confusión para conocer el número de predicciones de acertadas y fallidas del modelo

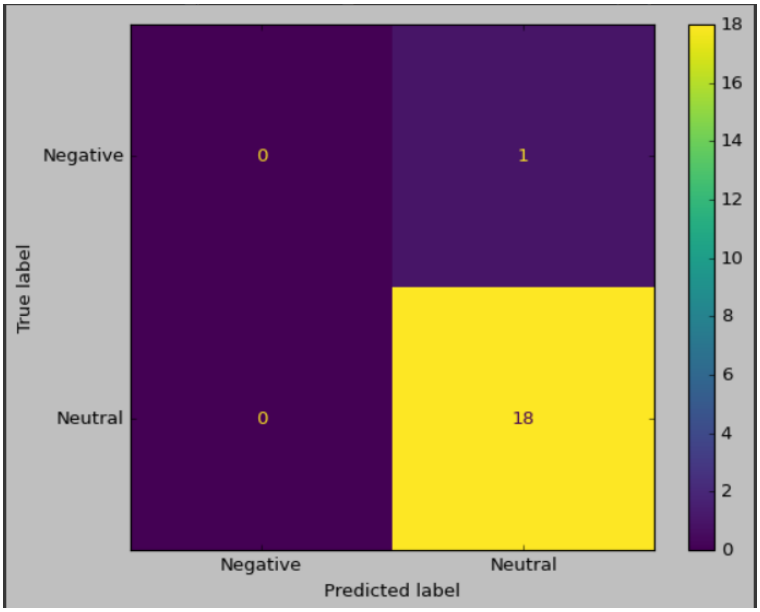



*Ilustración 6 Matriz de confusión Regresión logística*

#### 4.4.3.2. Máquinas de Soporte Vectorial

Mediante la librería Scikit-learn se empleó el algoritmo de SVM para la creación del modelo. En este caso se tiene como variable a predecir el sentimiento y como variable de entrada los registros de tweets previamente vectorizados. En cuanto a parámetros, se trabaja con una variación de este modelo, el clasificador de soporte vectorial SVC en su versión lineal. Como resultado, tenemos la siguiente tabla con las métricas de evaluación de rendimiento del modelo, una vez implementado el conjunto de datos de evaluación.

*Tabla 3 Métricas de evaluación SVM*

| | **Precisión** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| **Negative** | 0.00 | 0.00 | 0.00 | 1 |
| **Neutral** | 0.95 | 1.00 | 0.98 | 20 |
| **Accuracy** | | | 0.95 | 21 |
| **Macro Avg** | 0.48 | 0.50 | 0.49 | 21 |
| **Weigthed Avg** | 0.91 | 0.95 | 0.93 | 21 |

A continuación, se observa en la ilustración la matriz de confusión generada por el modelo el cual nos indica las predicciones del modelo

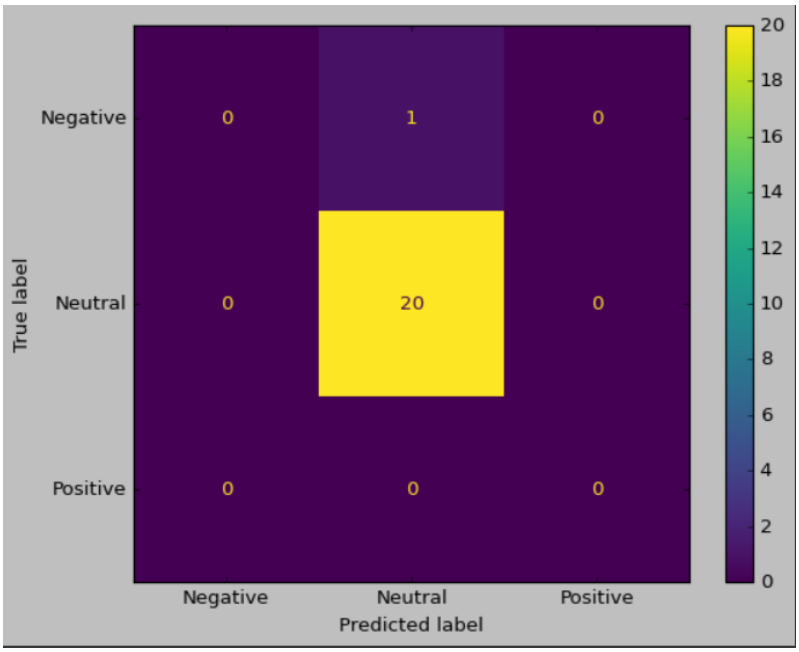



*Ilustración 7 Matriz de confusión SVM*

#### 4.4.3.3. Naive Bayes

Mediante la librería Scikit-learn se empleó el algoritmo de Naive Bayes para la creación del modelo. En este caso se tiene como variable a predecir el sentimiento y como variable de entrada los registros de tweets previamente vectorizados. En cuanto a parámetros, se trabaja con una

variación de este modelo, el Naive Bayes classifier, el cual permite la clasificación con variables discretas, ideal para este análisis. Como resultado, tenemos la siguiente tabla con las métricas de evaluación de rendimiento del modelo, una vez implementado el conjunto de datos de evaluación.

*Tabla 4 Métricas de evaluación Naive-Bayes*

| | Precisión | Recall | F1-Score | Support |
|---|---|---|---|---|
| Negative | 0.00 | 0.00 | 0.00 | 1 |
| Neutral | 0.95 | 0.84 | 0.89 | 25 |
| Positive | 0.00 | 0.00 | 0.00 | 0 |
| Accuracy | | | 0.81 | 26 |
| Macro Avg | 0.32 | 0.28 | 0.30 | 26 |
| Weigthed Avg | 0.92 | 0.81 | 0.86 | 26 |

Por último, tenemos la matriz de confusión generada por el modelo Naive Bayes el cual nos permite identificar las predicciones del modelo

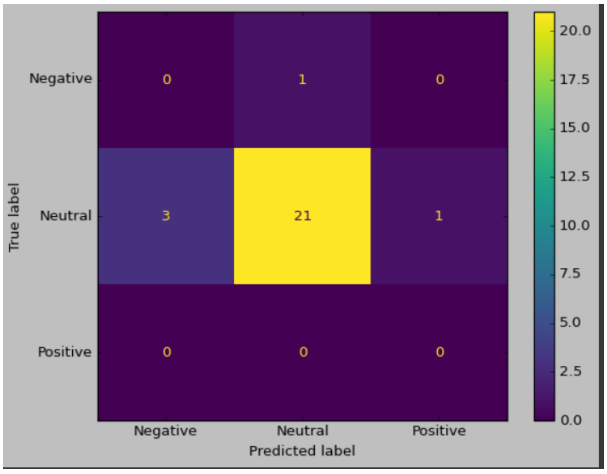



*Ilustración 8 Matriz de confusión Naive-Bayes*

### 4.4.4. Evaluar los modelos

En esta sección se va a analizar los resultados obtenidos mediante las métricas de evaluación descritas en pasos anteriores, dichas métricas son precisión, recall, f1-score. Hay que considerar que, al ser modelos multinomiales, en este caso tienen tres posibles salidas (negativo, positivo y neutro), se toma en cuenta el promedio de estas métricas en su forma balanceada para evaluar su rendimiento.

El modelo implementado de regresión logística obtuvo en su capacidad de predicción una puntuación F1 de 70% aplicando tanto el conjunto de entrenamiento (80%) como de evaluación (20%). En su métrica de precisión tenemos un 73% en su capacidad de predicción lo que permite conocer con qué porcentaje el modelo predeciría correctamente un nuevo conjunto de datos de prueba y, por último, en la métrica de exhaustividad (recall) obtuvo un 86% de rendimiento. Por otro lado, la exactitud que arroja el modelo es de 86% la cual nos indica que el modelo cumple con la métrica de evaluación propuesta en la primera fase de la metodología y puede ser usada para el cumplimiento de los objetivos establecidos.

En el segundo modelo implementado, máquinas de soporte vectorial obtuvo los siguientes resultados. En su capacidad de predicción una puntuación F1 de 93% aplicando tanto el conjunto de entrenamiento (80%) como de evaluación (20%). En su métrica de precisión tenemos un 91% en su capacidad de predicción lo que permite conocer con qué porcentaje el modelo predeciría correctamente un nuevo conjunto de datos de prueba y, por último, en exhaustividad (recall) obtuvo un 95% de rendimiento. En resumen, la métrica de exactitud que arroja el modelo es de un 95% el cual nos indica que el modelo cumple con la métrica de evaluación propuesta en la primera fase de la metodología y puede ser usada para el cumplimiento de los objetivos establecidos.

Por último, en el modelo Naive Bayes que implemento tanto el conjunto de entrenamiento como de evaluación obtuvo en su capacidad de predicción una puntuación F1 de 86%, en su métrica de precisión un 92% el cual permite conocer la precisión del modelo al ser evaluado con un nuevo conjunto de datos, en exhaustividad (recall) obtuvo un 81%. En términos generales, este modelo obtiene un 81% de exactitud lo cual cumple con la métrica de éxito establecida en la primera fase.

A continuación, se presenta la tabla con los resultados de las métricas previamente analizadas.

*Tabla 5 Comparación de rendimiento de algoritmos implementados*

| | **F1 -score** | **Recall** | **Precisión** | **Exactitud (accuracy)** |
|---|---|---|---|---|
| **Regresión logística** | 79% | 86% | 73% | 86/80 |
| **SVM** | 93% | 95% | 91% | 95/80 |
| **Naive Bayes** | 86% | 81% | 92% | 81/80 |

## 4.5. Evaluación

### 4.5.1. Evaluar los resultados

En esta fase se considera tanto las métricas obtenidas y analizadas en la fase anterior, como los objetivos planteados en la primera fase de la metodología. A pesar de que puede volver algo subjetivo la confianza de los modelos, se ha establecido que se considera un modelo fiable, al modelo que tenga dentro de sus métricas de precisión y exactitud un porcentaje mayor o igual al 80% ya que estas métricas permiten conocer el porcentaje de error del modelo con el conjunto de datos trabajado, así también en un futuro si se implementa un conjunto de datos distinto.

**Modelo 1: Regresión logística**

Este modelo no puede ser considerado factible. A pesar de que en todas sus métricas supera el 70% de aceptación, existen dos métricas que se toman para indicar la viabilidad de los modelos. En este caso, la métrica de exactitud si cumple con el porcentaje de aceptación del modelo con un 86%. Sin embargo, la métrica de precisión no cumple con dicho porcentaje ya que se obtuvo un 73%, lo cual descarta a este modelo como factible.

**Modelo 2: Máquinas de Soporte Vectorial**

Este modelo es aceptable ya que sus métricas de evaluación superan el porcentaje aceptado (80%) y de igual manera puede ser orientado tanto al objetivo del negocio ya que realiza predicciones en cuanto a los sentimientos de los tweets con una precisión 91% y de exactitud de un 95%

**Modelo 3: Naive Bayes**

Este modelo es aceptado dentro de esta fase, ya que las métricas evaluadas superan el porcentaje de evaluación aceptado (80%) teniendo como resultado una precisión de 92% y una exactitud de 81%.

Al considerar el objetivo principal de minería de datos, se tiene como resultado que el modelo más factible dentro de este trabajo es Maquinas de soporte vectorial, ya que tanto en sus métricas como evaluación previa es el que mejor rendimiento y eficiencia ha demostrado de los tres modelos implementados. A pesar de que el modelo Naive Bayes tiene una precisión superior de 92%, también se considera la exactitud de los modelos, Naive Bayes obtiene un 81% a diferencia de SVM que lo supera con un 95% de exactitud.

### 4.5.2. Revisión del proceso

El proceso se ha ejecutado exitosamente como se tenía previsto. Sin embargo, ha existido ciertas dificultades dentro de la fase 2, ya que se debe hacer una investigación exhaustiva del funcionamiento de las librerías para poder usarlas correctamente, hay que recalcar de que algunas herramientas con el tiempo han cambiado su sintaxis como funcionamiento, por lo que se debe trabajar con las versiones estables de cada herramienta para que proporcionen un rendimiento adecuado dentro del proyecto. De igual manera, a la hora de la extracción de datas se puede considerar más parámetros para realizar la búsqueda de lo tweets, en este caso al realizar un sondeo se consideró únicamente las palabras: Uber y UberEcuador, y las fechas de inicio y fin para la búsqueda. Sin embargo, se podría considerar más parámetros como incluir el usuario de la cuenta oficial de Uber Ecuador.

Adicionalmente, se recomienda extraer un número mayor de registros parametrizando la búsqueda con un rango de fechas mayor e incrementar el límite de búsqueda de la herramienta, para poder implementar conjunto de entrenamiento y prueba más grandes para identificar si existe variaciones en las métricas de evaluación de los modelos.

### 4.5.3. Determinar los próximos pasos

Esta sección no es considera dentro del proceso, ya que se excluyo la fase de despliegue de este trabajo.

## CONCLUSIONES Y RECOMENDACIONES

### Conclusiones

- Se logro identificar y clasificar los sentimientos de los usuarios de Twitter mediante los algoritmos de aprendizaje supervisado tales como: Regresión logística, Maquinas de Soporte Vectorial (SVM) y Naive-Bayes midiendo el rendimiento de cada uno de ellos mediante las métricas de F1-score, recall, precisión y exactitud.
- La extracción de datos es una de las tareas más importantes dentro del análisis de sentimientos. Por lo que se utiliza diferentes herramientas del lenguaje Python para la recopilación y creación del conjunto de datos.
- Se implemento modelos de Machine Learning y minería de datos, en el que se implemento procesamiento de lenguaje natural, limpieza y transformación de datos, y muestreo, lo que permitió la clasificación de sentimientos de los usuarios para lograr identificar el impacto de las opiniones a la imagen corporativa de Uber.
- Se evaluó el rendimiento de las diferentes métricas para medir el rendimiento de los algoritmos implementados en el trabajo. Dando como resultado que el algoritmo más factible con el conjunto de datos implementado es Maquinas de Soporte Vectorial.
- La metodología CRISP-DM es una de las más utilizadas a nivel mundial, por su capacidad de adaptación a diferentes proyectos de análisis de datos. En esta investigación se lo ha implementado y adaptado de acuerdo con las necesidades que se reconocen.

**Recomendaciones**

- Implementar conjuntos de datos más grandes y con diferentes porcentajes de muestreo para evaluar el desempeño de los modelos implementados en el trabajo.
- Es necesario conocer con precisión el funcionamiento y parámetros de los diferentes algoritmos implementados, ya que la partición del conjunto de datos varia por este criterio.
- Como trabajo futuro se recomienda extraer datos de diferentes redes sociales como Facebook para implementarlos en los algoritmos trabajados en este trabajo.

**BIBLIOGRAFÍA**


Ali, R. (2022, 18 marzo). *Predictive Modeling: Types, Benefits, and Algorithms*. Oracle NetSuite. Recuperado 1 de septiembre de 2022, de https://www.netsuite.com/portal/resource/articles/financial-management/predictive-modeling.shtml

E. (2021a, octubre 15). *¿Qué es la imagen corporativa de una empresa y para qué sirve?* NeoAttack. Recuperado 1 de septiembre de 2022, de https://neoattack.com/blog/que-es-la-imagen-corporativa/

F. (2021b, marzo 11). *Association Rules in Data Mining*. EDUCBA. Recuperado 31 de agosto de 2022, de https://www.educba.com/association-rules-in-data-mining/

Galán Cortina, V. (2015). Aplicación de la metodología CRISP-DM a un proyecto de minería de

datos en el entorno universitario [Proyecto de fin de carrera]. Universidad Carlos II de Madrid.

Gartner. (s. f.). *Definition of Predictive Modeling - Gartner Information Technology Glossary*. Recuperado 1 de septiembre de 2022, de https://www.gartner.com/en/information-technology/glossary/predictive-modeling

Google Colab. (s. f.). https://research.google.com/colaboratory/intl/es/faq.html googletrans. (2020, 14 junio). PyPI. https://pypi.org/project/googletrans/

González, B. A. (s. f.). *Conceptos básicos de Machine Learning – Cleverdata*. Cleverdata. Recuperado 31 de agosto de 2022, de https://cleverdata.io/conceptos-basicos-machine-learning/

IBM. (s. f.-a). *Acerca de la minería de textos*. © Copyright IBM Corp. 2003, 2017. Recuperado 31 de agosto de 2022, de https://www.ibm.com/docs/es/spss-modeler/18.1.1?topic=analytics-about-text-mining

IBM. (s. f.-b). *El modelo de redes neuronales*. Recuperado 31 de agosto de 2022, de https://www.ibm.com/docs/es/spss-modeler/saas?topic=networks-neural-model

IBM. (s. f.-c). *¿Qué es un árbol de decisión?* | *IBM*. Recuperado 31 de agosto de 2022, de https://www.ibm.com/es-es/topics/decision-trees

Matplotlib — Visualization with Python. (s. f.). https://matplotlib.org

NLTK: Natural Language Toolkit. (s. f.). https://www.nltk.org

Oracle. (s. f.-a). *Predictive Analysis*. Oracle Docs. Recuperado 31 de agosto de 2022, de https://docs.oracle.com/cd/E28280_01/admin.1111/e14568/predict.htm#AAMAD6433

Oracle. (s. f.-b). *What Is Data Mining?* Oracle Docs. Recuperado 31 de agosto de 2022, de https://docs.oracle.com/cd/B28359_01/datamine.111/b28129/process.htm#DMCON046

Rodrigo, J. A. (2016). *Correlación y Regresión lineales simple*. GitHub. Recuperado 31 de agosto de 2022, de https://www.cienciadedatos.net/documentos/24_correlacion_y_regresion_lineal

Rosenbrock, G., Trossero, S., & Pascal, A. (2021). *Técnicas de Análisis de Sentiminetos Aplicadas a la Valoración de Opiniones en el Lenguaje Español*. ResearchGate. Recuperado 31 de agosto de 2022, de https://www.researchgate.net/publication/355887680_Tecnicas_de_Analisis_de_Sentimientos_Aplicadas_a_la_Valoracion_de_Opiniones_en_el_Lenguaje_Espanol

*Scikit-Learn: Descubre la biblioteca de Python dedicada al Machine Learning. (2022, 1 septiembre). Formation Data Science | DataScientest.com. https://datascientest.com/es/scikit-learn-decubre-la-biblioteca-python*

*SUPERVISED MODELS | Data Vedas*. (s. f.). DataVedas. Recuperado 31 de agosto de 2022, de https://www.datavedas.com/supervised-models/

Uber. (2022, 11 enero). *Qué es Uber y qué oportunidades ofrece | Ecuador*. Uber Blog. Recuperado 1 de septiembre de 2022, de https://www.uber.com/es-EC/blog/que-es-uber-ecuador/

Urrutia, D. (2021, 5 agosto). *Qué es Análisis de sentimiento - Definición, significado y ejemplos*. Arimetrics. Recuperado 31 de agosto de 2022, de https://www.arimetrics.com/glosario-digital/analisis-de-sentimiento

## ANEXOS

### Anexo A: Conjuntos de datos

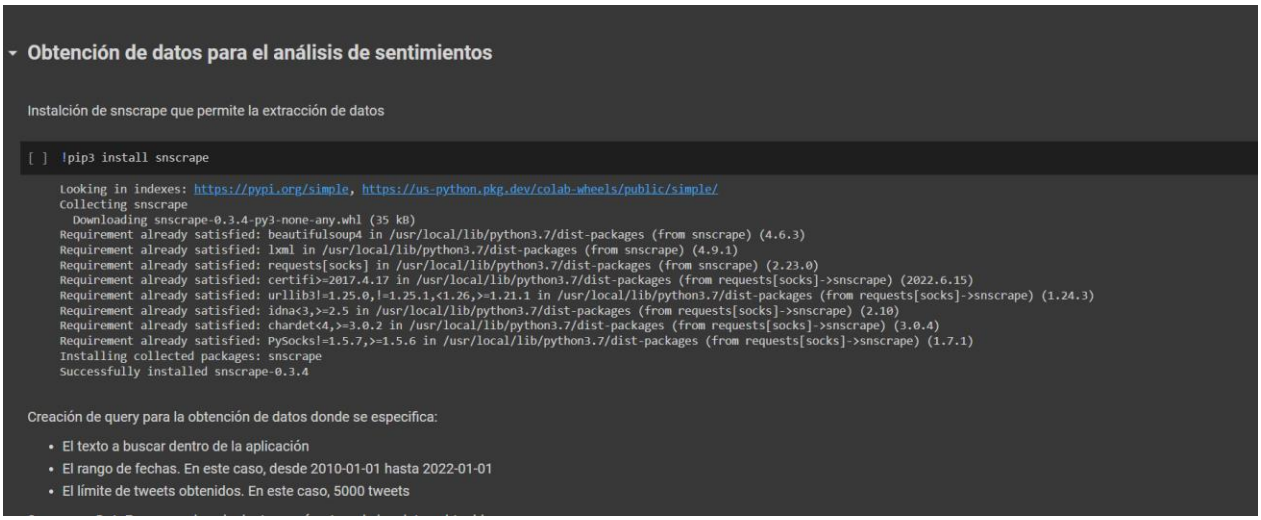
URL: https://github.com/mapaulabecerra/Titulacion-2022/tree/master/DATA

### Anexo B: Repositorio del código

Notebooks ejecutados de las diferentes etapas del análisis de datos, extracción de datos, preprocesamiento, traducción, vectorización y clasificación

URL: https://github.com/mapaulabecerra/Titulacion-2022

### Anexo C: Código de recopilación de datos iniciales



Se crea un DataFrame con los siguientes parámetros de los datos obtenidos:

- Fecha.
- Usuario.
- Tweet (texto obtenido y en lo que nos vamos a enfocar)

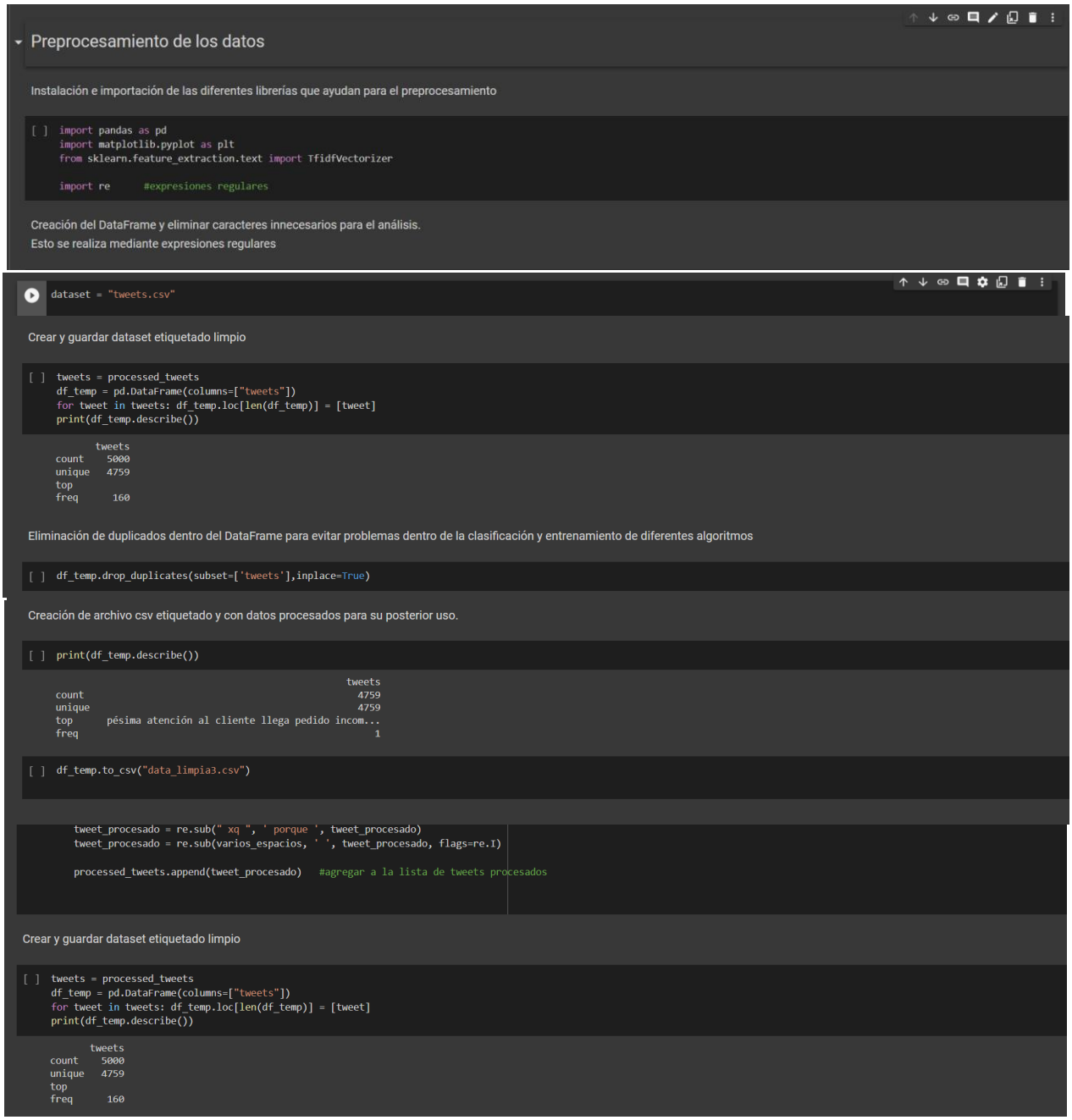
```
[ ] import snscrape.modules.twitter as sntwitter
    import pandas as pd

    query = "(Uber Ecuador until:2022-01-01 since:2010-01-01"
    tweets = []
    limit = 5000


    for tweet in sntwitter.TwitterSearchScraper(query).get_items():

        # print(vars(tweet))
        # break
        if len(tweets) == limit:
            break
        else:
            tweets.append([tweet.date, tweet.username, tweet.content])

    df = pd.DataFrame(tweets, columns=['Date', 'User', 'Tweet'])
    print(df)
```

```
                          Date             User  \
0    2021-12-31 18:12:04+00:00      AdriPabArt1
1    2021-12-31 00:10:17+00:00        juanppino
2    2021-12-30 16:13:23+00:00        alonso390
3    2021-12-30 14:30:37+00:00            elyex
4    2021-12-30 05:38:01+00:00       shane70707
...                        ...              ...
4995 2020-07-20 15:42:00+00:00  ProgressiveNova

                                                  Tweet
0     @UberEcuador @GrupoKfcEcuador @kfcecuador  Pés...
1     @Uber_Ecuador asi como me cobran un fee cada q...
2     @Uber_Ecuador sigo esperando respuesta de @Ube...
3     #InDriver #Ecuador ¿Qué es y cómo trabajar de ...
4     @cr7roprhymes @brilliantbusi Lol. Honestly win...
...                                                 ...
4995  Hace 2 meses aproximadamente mi cuenta fue cer...
4996  @Uber_Support @UberEcuador Uber Eats Ecuador C...
4997  Este es el #Ecuador de siempre: con más edific...
4998  @Uber_Support Ya he enviado DM hace más de dos...
4999  @Uber_Ecuador Ayer yo pedo una carrera en ning...

[5000 rows x 3 columns]
```

Creación de un archivo csv con el DataFrame creado para poder trabajar posteriormente con los datos obtenidos. Considerar que al referirse a texto en el idioma español y por su contenido de caracteres especiales como tildes, etc se procede a especificar la codificación UTF-8

```
[ ] # to save to csv
    df.to_csv('tweets.csv',encoding="utf-8")
```

## Anexo D: Limpieza y preprocesamiento de datos



Preprocesamiento de los datos

Instalación e importación de las diferentes librerías que ayudan para el preprocesamiento

```
[ ] import pandas as pd
    import matplotlib.pyplot as plt
    from sklearn.feature_extraction.text import TfidfVectorizer

    import re      #expresiones regulares
```

Creación del DataFrame y eliminar caracteres innecesarios para el análisis.
Esto se realiza mediante expresiones regulares

```
dataset = "tweets.csv"
```

Crear y guardar dataset etiquetado limpio

```
[ ] tweets = processed_tweets
    df_temp = pd.DataFrame(columns=["tweets"])
    for tweet in tweets: df_temp.loc[len(df_temp)] = [tweet]
    print(df_temp.describe())
```

```
          tweets
count       5000
unique      4759
top
freq         160
```

Eliminación de duplicados dentro del DataFrame para evitar problemas dentro de la clasificación y entrenamiento de diferentes algoritmos

```
[ ] df_temp.drop_duplicates(subset=['tweets'],inplace=True)
```

Creación de archivo csv etiquetado y con datos procesados para su posterior uso.

```
[ ] print(df_temp.describe())
```

```
                                                tweets
count                                             4759
unique                                            4759
top       pésima atención al cliente llega pedido incom...
freq                                                 1
```

```
[ ] df_temp.to_csv("data_limpia3.csv")
```

```
        tweet_procesado = re.sub(" xq ", ' porque ', tweet_procesado)
        tweet_procesado = re.sub(varios_espacios, ' ', tweet_procesado, flags=re.I)

        processed_tweets.append(tweet_procesado)   #agregar a la lista de tweets procesados
```

Crear y guardar dataset etiquetado limpio

```
[ ] tweets = processed_tweets
    df_temp = pd.DataFrame(columns=["tweets"])
    for tweet in tweets: df_temp.loc[len(df_temp)] = [tweet]
    print(df_temp.describe())
```

```
          tweets
count       5000
unique      4759
top
freq         160
```

## Anexo E: Traducción del conjunto de datos

### Traducción de tweets

Instalación e importación de la librería googletrans que nos permite realizar la traducción de datos

```
[ ] !pip install googletrans==4.0.0rc1
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting googletrans==4.0.0rc1
  Downloading googletrans-4.0.0rc1.tar.gz (20 kB)
Collecting httpx==0.13.3
  Downloading httpx-0.13.3-py3-none-any.whl (55 kB)
     |████████████████████████████████| 55 kB 4.4 MB/s
Collecting hstspreload
  Downloading hstspreload-2022.9.1-py3-none-any.whl (1.4 MB)
     |████████████████████████████████| 1.4 MB 64.0 MB/s
Requirement already satisfied: idna==2.* in /usr/local/lib/python3.7/dist-packages (from httpx==0.13.3->googletrans==4.0.0rc1) (2.10)
Collecting httpcore==0.9.*
  Downloading httpcore-0.9.1-py3-none-any.whl (42 kB)
     |████████████████████████████████| 42 kB 1.5 MB/s
Collecting sniffio
  Downloading sniffio-1.3.0-py3-none-any.whl (10 kB)
Collecting rfc3986<2,>=1.3
  Downloading rfc3986-1.5.0-py2.py3-none-any.whl (31 kB)
Requirement already satisfied: chardet==3.* in /usr/local/lib/python3.7/dist-packages (from httpx==0.13.3->googletrans==4.0.0rc1) (3.0.4)
Requirement already satisfied: certifi in /usr/local/lib/python3.7/dist-packages (from httpx==0.13.3->googletrans==4.0.0rc1) (2022.6.15)
Collecting h11<0.10,>=0.8
  Downloading h11-0.9.0-py2.py3-none-any.whl (53 kB)
     |████████████████████████████████| 53 kB 2.7 MB/s
Collecting h2==3.*
  Downloading h2-3.2.0-py2.py3-none-any.whl (65 kB)
     |████████████████████████████████| 65 kB 4.0 MB/s
Collecting hyperframe<6,>=5.2.0
  Downloading hyperframe-5.2.0-py2.py3-none-any.whl (12 kB)
Collecting hpack<4,>=3.0
  Downloading hpack-3.0.0-py2.py3-none-any.whl (38 kB)
Building wheels for collected packages: googletrans
```

```
[ ] Building wheels for collected packages: googletrans
    Building wheel for googletrans (setup.py) ... done
    Created wheel for googletrans: filename=googletrans-4.0.0rc1-py3-none-any.whl size=17416 sha256=36348efaa37d00cc8b8b46d9971e66680e0fb9a3eb796f6641f5b5097b6e0e51
    Stored in directory: /root/.cache/pip/wheels/43/34/00/4fe71786ea6d12314b29037620c36d857e5d104ac2748bf82a
  Successfully built googletrans
  Installing collected packages: hyperframe, hpack, sniffio, h2, h11, rfc3986, httpcore, hstspreload, httpx, googletrans
  Successfully installed googletrans-4.0.0rc1 h11-0.9.0 h2-3.2.0 hpack-3.0.0 hstspreload-2022.9.1 httpcore-0.9.1 httpx-0.13.3 hyperframe-5.2.0 rfc3986-1.5.0 sniffio-1.3.0
```

```
[ ] import googletrans
    import pandas as pd
    from googletrans import Translator
```

Se inicializa la API para la traudcción

```
[ ] translator=Translator()
```

Creación de una prueba para comprobar la funcionalidad de la API

```
[ ] translated=translator.translate('Prueba hola mundo')
```

```
[ ] print(translated)
```

```
Translated(src=es, dest=en, text=Try hello world, pronunciation=None, extra_data="{'confiden...")
```

```
[ ] translated.text
```

```
'Try hello world'
```

Creación de DataFrame para trabajar la traducción de los tweets obtenidos y procesados

```
[ ] df_es= pd.read_csv('data_limpia4.csv',sep=',',encoding='utf-8')
```

Visualización de DataFrame

```
[ ] df_es.head()
```

| | Unnamed: 0 | tweets |
|---|---|---|
| 0 | 0 | pésima atención al cliente llega pedido incomp... |
| 1 | 1 | así como me cobran un fee cada que cancelo una... |
| 2 | 2 | sigo esperando respuesta de de por el viaje se... |
| 3 | 3 | qué es cómo trabajar de |
| 4 | 4 | lol honestly winning voting contests is what v... |

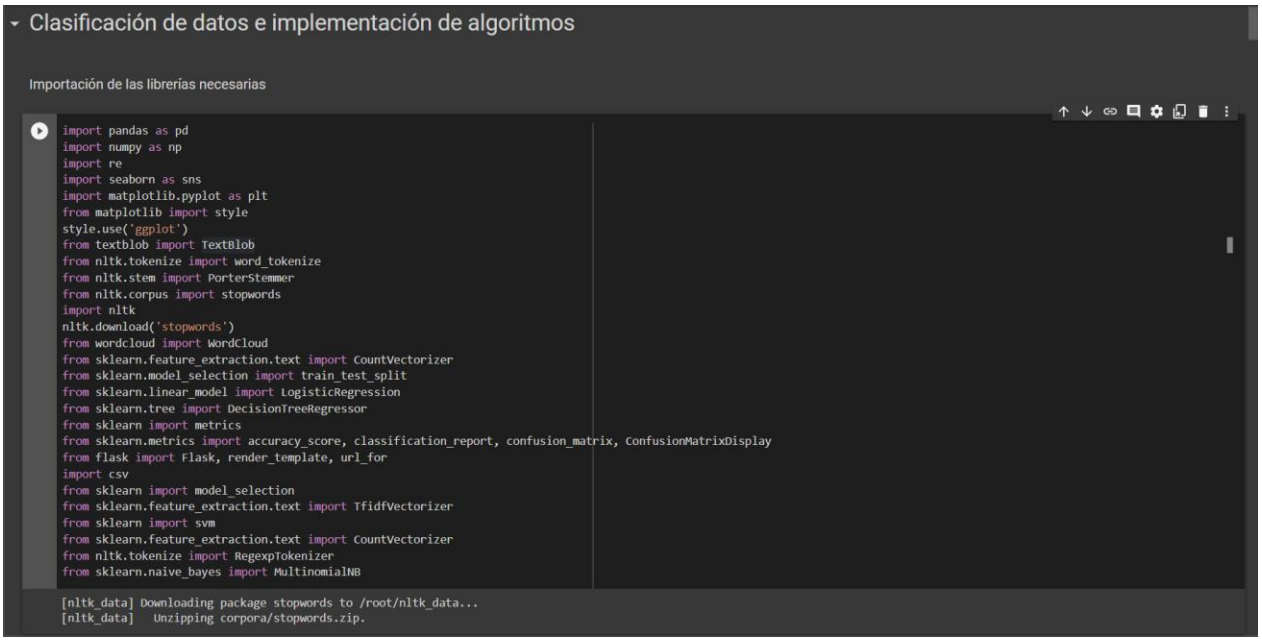
Función que permite la traducción del conjunto de datos, registro por registro. se considera la funcionalidad de la librería implementada y se crea condicionales por si el sevidor principal utilizado no traduce el texto diferentes servidores de habla hispana sean implementados.

```
df_Transl = pd.DataFrame(columns=["text_original", "text_traducido"])

#Método de traducción recorriendo la lista de tweets
print("Se ha iniciado la traducción...")
for i, tweet in enumerate(tweets):

    if (str(tweet) == 'nan'):
        print("Lectura de tweets completa")
        break
    traduccion = translator.translate(tweet, src='es')

    cc=1
    #En caso de no traducirse, lo reintenta varias veces
    while (traduccion.origin == traduccion.text):
        print("No se pudo traducir la fila "+str(i)+", reintento "+ str(cc))
        translator = Translator(service_urls=[
              'translate.google.com',
              'translate.google.co.kr',
              'translate.google.com.ec',
              'translate.google.com.mx',
              'translate.google.com.uy',
              'translate.google.cn',
            ])
        traduccion = translator.translate(tweet, src='es')

        cc+=1
        if(cc > 5): break
    #Verifica si el texto fue traducido
    if(traduccion.origin == traduccion.text):
        print("Se intentó traducir:"+str(traduccion.origin) +" pero devolvió: " +str(traduccion.text))
        print("Todo el proceso se detuvo en el índice: "+str(i)+" y el tweet: "+str(tweet))
        break

    #Agrega texto traducido a un dataframe
    df_Transl.loc[i] = [traduccion.origin, traduccion.text]
```

```
#Guardar DataFrame a un archivo csv
print("Traducción finalizada.")
df_Transl.to_csv("/content/tweets_traducidos.csv", index=False)
print("Archivos exportados correctamente.")
```

```
Se ha iniciado la traducción...
Lectura de tweets completa
Traducción finalizada.
Archivos exportados correctamente.
```

## Anexo F: Procesamiento de lenguaje natural e implementación de algoritmos.

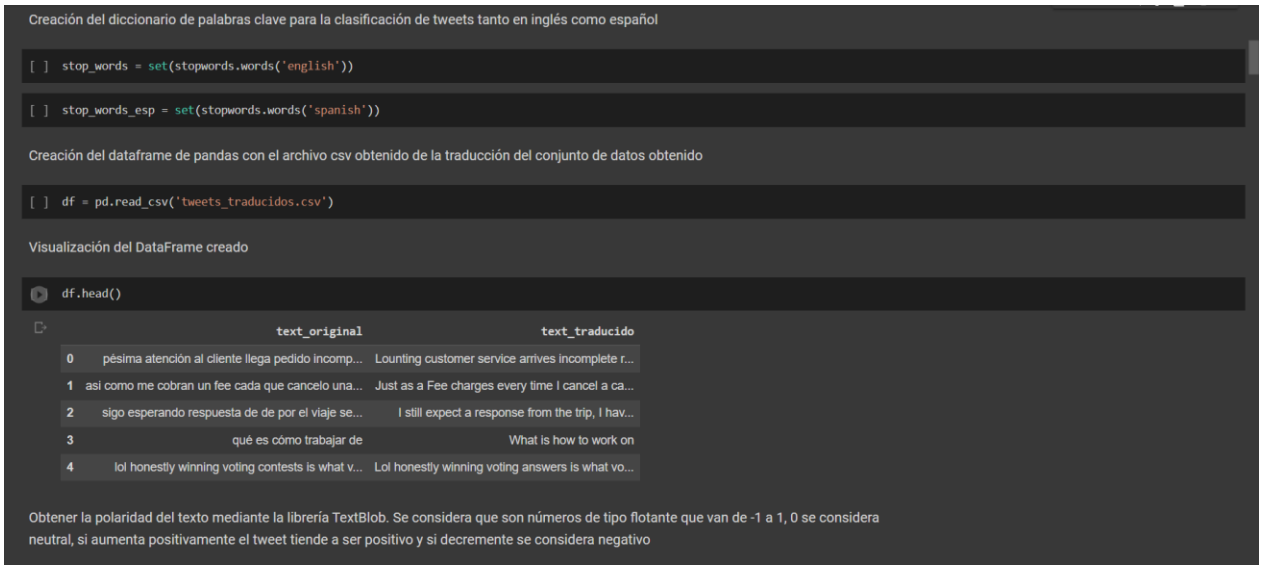


Clasificación de datos e implementación de algoritmos

Importación de las librerías necesarias

```
import pandas as pd
import numpy as np
import re
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import style
style.use('ggplot')
from textblob import TextBlob
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
from nltk.corpus import stopwords
import nltk
nltk.download('stopwords')
from wordcloud import WordCloud
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn import metrics
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, ConfusionMatrixDisplay
from flask import Flask, render_template, url_for
import csv
from sklearn import model_selection
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import svm
from sklearn.feature_extraction.text import CountVectorizer
from nltk.tokenize import RegexpTokenizer
from sklearn.naive_bayes import MultinomialNB
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```



Creación del diccionario de palabras clave para la clasificación de tweets tanto en inglés como español

```
[ ] stop_words = set(stopwords.words('english'))
```

```
[ ] stop_words_esp = set(stopwords.words('spanish'))
```

Creación del dataframe de pandas con el archivo csv obtenido de la traducción del conjunto de datos obtenido

```
[ ] df = pd.read_csv('tweets_traducidos.csv')
```

Visualización del DataFrame creado

```
df.head()
```

| | text_original | text_traducido |
|---|---|---|
| 0 | pésima atención al cliente llega pedido incomp... | Lounting customer service arrives incomplete r... |
| 1 | asi como me cobran un fee cada que cancelo una... | Just as a Fee charges every time I cancel a ca... |
| 2 | sigo esperando respuesta de de por el viaje se... | I still expect a response from the trip, I hav... |
| 3 | qué es cómo trabajar de | What is how to work on |
| 4 | lol honestly winning voting contests is what v... | Lol honestly winning voting answers is what vo... |

Obtener la polaridad del texto mediante la librería TextBlob. Se considera que son números de tipo flotante que van de -1 a 1, 0 se considera neutral, si aumenta positivamente el tweet tiende a ser positivo y si decremente se considera negativo

```
def polarity(text):
    return TextBlob(text).sentiment.polarity
```

```
df['polarity'] = df['text_traducido'].apply(polarity)
```

```
df['polarity'] = df['text_original'].apply(polarity)
```

Clasificación de los tweets en categorías: positivo, neutral y negativo. Mediante la librería de TextBlob y el dicionario de stop words creados previamente

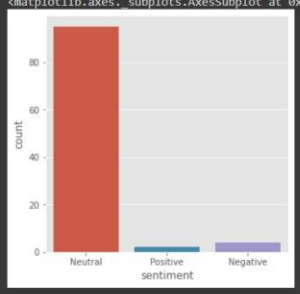
```
def sentiment(label):
    if label <0:
        return "Negative"
    elif label ==0:
        return "Neutral"
    elif label>0:
        return "Positive"
```

```
df['sentiment'] = df['polarity'].apply(sentiment)
```

Visualización de la clasificación de los tweet

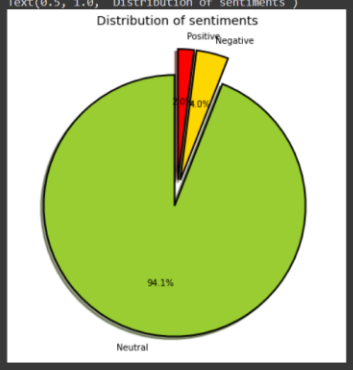
```
fig = plt.figure(figsize=(5,5))
sns.countplot(x='sentiment', data = df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f7ca5e23290>
```




```
fig = plt.figure(figsize=(7,7))
colors = ("yellowgreen", "gold", "red")
wp = {'linewidth':2, 'edgecolor':"black"}
tags = df['sentiment'].value_counts()
explode = (0.1,0.1,0.1)
tags.plot(kind='pie', autopct='%1.1f%%', shadow=True, colors = colors,
          startangle=90, wedgeprops = wp, explode = explode, label='')
plt.title('Distribution of sentiments')
```

```
Text(0.5, 1.0, 'Distribution of sentiments')
```



Creación de un archivo csv con los datos previamente clasificados

```
[ ] df.to_csv("data_label.csv", index=False)
    print("Archivos exportados correctamente.")
```

```
Archivos exportados correctamente.
```

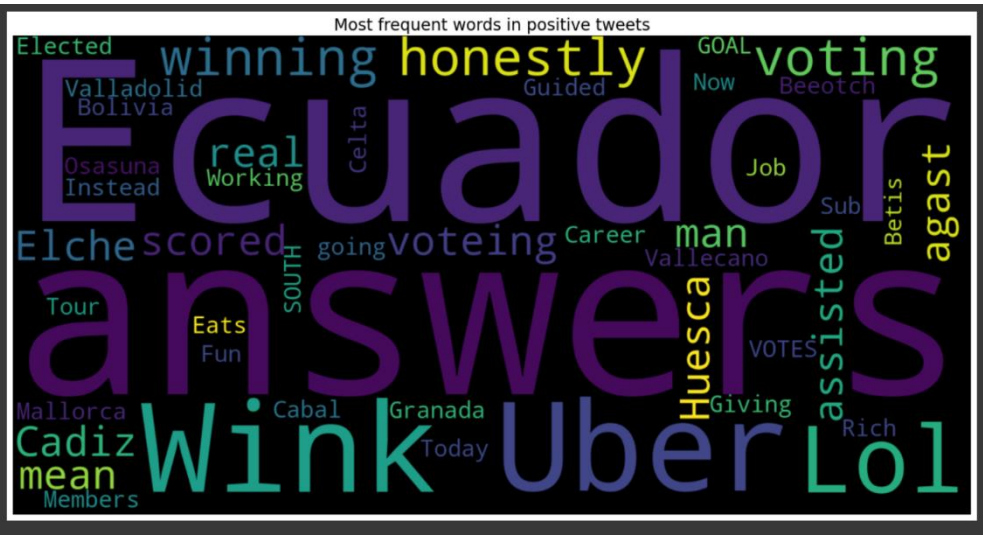
Creación de DataFrame con twwwts clasificados positivamente con su respectiva polaridad

```
[ ] pos_tweets = df[df.sentiment == 'Positive']
    pos_tweets = pos_tweets.sort_values(['polarity'], ascending= False)
    pos_tweets.head()
```

| | text_original | text_traducido | polarity | sentiment |
|---|---|---|---|---|
| 4 | lol honestly winning voting contests is what v... | Lol honestly winning voting answers is what vo... | 0.3375 | Positive |
| 45 | so why are you going to ecuador today instead ... | So Why are you going to Ecuador Today Instead ... | 0.3375 | Positive |

Nube de palabras con las palabras más frecuentes dentro del DataFrame de tweets positivos

```
text_traducido = ' '.join([word for word in pos_tweets['text_traducido']])
plt.figure(figsize=(20,15), facecolor='None')
wordcloud = WordCloud(max_words=500, width=1600, height=800).generate(text_traducido)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title('Most frequent words in positive tweets', fontsize=19)
plt.show()
```



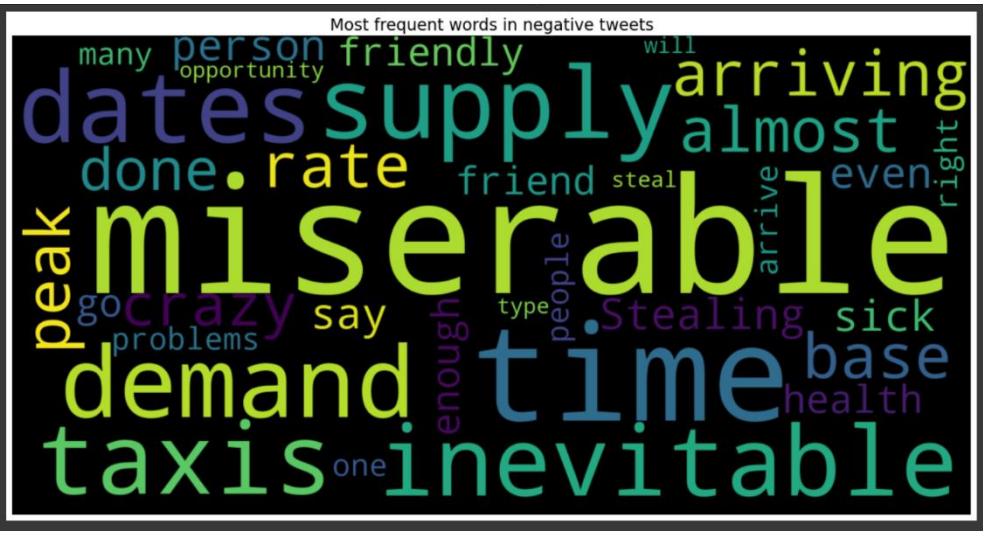
Creación de DataFrame con tweets clasificados como negativos con su polaridad

```
[ ] neg_tweets = df[df.sentiment == 'Negative']
    neg_tweets = neg_tweets.sort_values(['polarity'], ascending= False)
    neg_tweets.head()
```

| | text_original | text_traducido | polarity | sentiment |
|---|---|---|---|---|
| 39 | la oferta demanda en taxis esta fechas es inev... | The supply demand in taxis This dates is inevi... | -0.4 | Negative |
| 90 | es un miserable eso no se hace | It is a miserable that is not done | -1.0 | Negative |
| 92 | robarle una persona enferma ni siquiera digo a... | Stealing a sick person does not even say frien... | -1.0 | Negative |
| 94 | que ser para miserable el tipo | what to be miserable the type | -1.0 | Negative |

Nube de palabras con las palabras más frecuentes dentro del DataFrame con tweets negativos.

```
[ ] text_traducido = ' '.join([word for word in neg_tweets['text_traducido']])
    plt.figure(figsize=(20,15), facecolor='None')
    wordcloud = WordCloud(max_words=500, width=1600, height=800).generate(text_traducido)
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off")
    plt.title('Most frequent words in negative tweets', fontsize=19)
    plt.show()
```



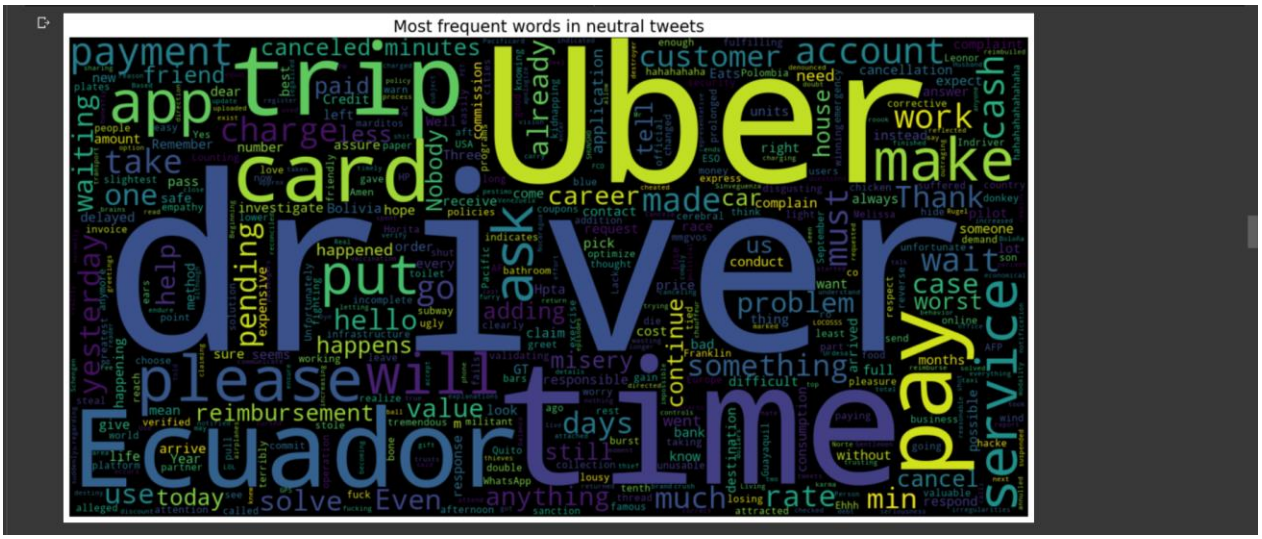

Cración de DataFrame con tweets clasificados como neutrales con su respectiva polaridad

```
[ ] neutral_tweets = df[df.sentiment == 'Neutral']
    neutral_tweets = neutral_tweets.sort_values(['polarity'], ascending= False)
    neutral_tweets.head()
```

| | text_original | text_traducido | polarity | sentiment |
|---|---|---|---|---|
| 0 | pésima atención al cliente llega pedido incomp... | Lounting customer service arrives incomplete r... | 0.0 | Neutral |
| 63 | osea en vez de ganar pierdo dinero que no es n... | I mean instead of winning, I lose money that i... | 0.0 | Neutral |
| 72 | se pasan de hp | They pass from HP | 0.0 | Neutral |
| 71 | dónde pongo una queja con | Where do I put a complaint with | 0.0 | Neutral |
| 70 | no pida tenga en cuenta algo melissa del preci... | Do not ask to take into account something Meli... | 0.0 | Neutral |

Nube de palabras con palabras más frecuentes dentro del DataFrame de tweets neutrales

```
[ ] text_traducido = ' '.join([word for word in neutral_tweets['text_traducido']])
    plt.figure(figsize=(20,15), facecolor='None')
    wordcloud = WordCloud(max_words=500, width=1600, height=800).generate(text_traducido)
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off")
    plt.title('Most frequent words in neutral tweets', fontsize=19)
    plt.show()
```




Vectorización de los tweets tanto traducidos al idioma inglés como originales

```
[ ] vect = CountVectorizer(ngram_range=(1,2)).fit(df['text_traducido'])
```

```
[ ] vect_esp = CountVectorizer(ngram_range=(1,2)).fit(df['text_original'])
```

Creación del conjunto de features con los tweets vectorizados por palabras

```
[ ] feature_names = vect.get_feature_names_out()
    print("Number of features: {}\n".format(len(feature_names)))
    print("First 20 features:\n {}".format(feature_names[:20]))
```

```
Number of features: 2185

First 20 features:
 ['about' 'about his' 'about the' 'ac' 'accept' 'accept card' 'account'
 'account cancel' 'account card' 'account something' 'account was'
 'accounts' 'accounts are' 'adding' 'adding credit' 'adding that'
 'adding uber' 'addition' 'addition the' 'af']
```

```
[ ] feature_names_esp = vect_esp.get_feature_names_out()
    print("Number of features: {}\n".format(len(feature_names_esp)))
    print("First 20 features:\n {}".format(feature_names_esp[:20]))
```

```
Number of features: 2555

First 20 features:
 ['abusan' 'abusan de' 'ac' 'acabo' 'acabo de' 'aceptar' 'aceptar el'
 'actualizan' 'actualizan su' 'adding' 'adding credit' 'además'
 'además la' 'adjunto' 'adjunto detalles' 'af' 'afirmación'
 'afirmación conducta' 'after' 'after few']
```

```
[ ] X = df['text_traducido']
    Y = df['sentiment']
    X = vect.transform(X)
```

```
[ ] X = df['text_original']
    Y = df['sentiment']
    X = vect_esp.transform(X)
```

Creación de los sets de entrenamiento como de prueba tomando en cuenta la proporción 80 20

```
[ ] x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
```

```
[ ] xe_train, xe_test, ye_train, ye_test = train_test_split(X, Y, test_size=0.2, random_state=42)
```

Visualización de los tamñanos de los diferentes sets de entrenamiento y de prueba

```
[ ] print("Size of x_train:", (x_train.shape))
    print("Size of y_train:", (y_train.shape))
    print("Size of x_test:", (x_test.shape))
    print("Size of y_test:", (y_test.shape))
```

```
Size of x_train: (80, 2555)
Size of y_train: (80,)
Size of x_test: (21, 2555)
Size of y_test: (21,)
```

```
[ ] import warnings
    warnings.filterwarnings('ignore')
```

Creación de diferentes algoritmos de clasificación para el entrenamiento de los mismos con los sets de datos creados anteriormente

Algoritmo de regresión logística

```
[ ] logreg = LogisticRegression()
    logreg.fit(xe_train, ye_train)
    logreg_pred = logreg.predict(xe_test)
    logreg_acc = accuracy_score(logreg_pred, ye_test)
    print("Test accuracy: {:.2f}%".format(logreg_acc*100))
```

```
Test accuracy: 85.71%
```

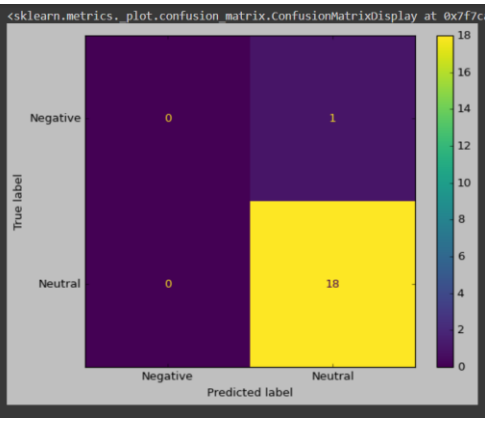
```
[ ] print(classification_report(ye_test, logreg_pred))
```

```
                precision    recall  f1-score   support

      Negative       0.00      0.00      0.00         1
       Neutral       0.86      1.00      0.92        18
      Positive       0.00      0.00      0.00         2

      accuracy                           0.86        21
     macro avg       0.29      0.33      0.31        21
  weighted avg       0.73      0.86      0.79        21
```

Matriz de confusión del algoritmo de regresión logística

```
[ ] style.use('classic')
    cm = confusion_matrix(y_test, logreg_pred, labels=logreg.classes_)
    disp = ConfusionMatrixDisplay(confusion_matrix = cm, display_labels=logreg.classes_)
    disp.plot()
```

```
[ ] <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f7ca5eca250>
```




```
[ ] from sklearn.model_selection import GridSearchCV

[ ] param_grid={'C':[0.001, 0.01, 0.1, 1, 10]}
    grid = GridSearchCV(LogisticRegression(), param_grid)
    grid.fit(x_train, y_train)

    GridSearchCV(estimator=LogisticRegression(),
                 param_grid={'C': [0.001, 0.01, 0.1, 1, 10]})

[ ] print("Best parameters:", grid.best_params_)

    Best parameters: {'C': 0.001}

[ ] y_pred = grid.predict(x_test)

[ ] logreg_acc = accuracy_score(y_pred, y_test)
    print("Test accuracy: {:.2f}%".format(logreg_acc*100))

    Test accuracy: 85.71%

[ ] print(confusion_matrix(y_test, y_pred))
    print("\n")
    print(classification_report(y_test, y_pred))

    [[ 0  1  0]
     [ 0 18  0]
     [ 0  2  0]]
```

```
[ ]              precision    recall  f1-score   support

    Negative       0.00      0.00      0.00         1
     Neutral       0.86      1.00      0.92        18
    Positive       0.00      0.00      0.00         2

    accuracy                           0.86        21
   macro avg       0.29      0.33      0.31        21
weighted avg       0.73      0.86      0.79        21

[ ] logreg = LogisticRegression()
    logreg.fit(xe_train, ye_train)
    logreg_pred = logreg.predict(xe_test)
    logreg_acc = accuracy_score(logreg_pred, ye_test)
    print("Test accuracy: {:.2f}%".format(logreg_acc*100))

    Test accuracy: 85.71%
```

SVM

```
[ ] train_X, test_X, train_Y, test_Y = model_selection.train_test_split(df['text_traducido'], df['sentiment'], test_size = 0.2, random_state = 0)

[ ] df_train80 = pd.DataFrame()
    df_train80['Sentiment'] = train_X
    df_train80['Label'] = train_Y

    df_test20 = pd.DataFrame()
    df_test20['Sentiment'] = test_X
    df_test20['Label'] = test_Y

[ ] df_train80.to_csv(r"df_train80.csv")
    df_test20.to_csv(r"df_test20.csv")
```

```
[ ] # TF-IDF

    from sklearn.feature_extraction.text import TfidfVectorizer

    tfidf_vect_8020 = TfidfVectorizer(max_features = 5000)
    tfidf_vect_8020.fit(df['text_traducido'])
    train_X_tfidf_8020 = tfidf_vect_8020.transform(df_train80['Sentiment'])
    test_X_tfidf_8020 = tfidf_vect_8020.transform(df_test20['Sentiment'])
```

```
[ ] print(train_X_tfidf_8020.shape)
    print(test_X_tfidf_8020.shape)

    (80, 662)
    (21, 662)
```

```
[ ]
    # You can use the below syntax to see the vocabulary that it has learned from the corpus
    print(tfidf_vect_8020.vocabulary_)

    {'lounting': 334, 'customer': 152, 'service': 502, 'arrives': 45, 'incomplete': 289, 'request': 472, 'no': 373, 'one': 390, 'is': 302, 'responsible': 478, 'for': 225, 'their': 55
```

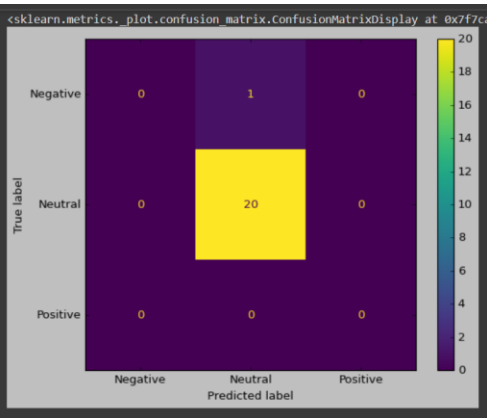
```
[ ] from sklearn.svm import SVC

    model = SVC(kernel='linear')
    model.fit(train_X_tfidf_8020,train_Y)

    SVC(kernel='linear')
```

```
[ ] from sklearn.metrics import accuracy_score

    predictions_SVM_8020 = model.predict(test_X_tfidf_8020)
    test_prediction_8020 = pd.DataFrame()
    test_prediction_8020['Sentiment'] = test_X
    test_prediction_8020['Label'] = predictions_SVM_8020
    SVM_accuracy_8020 = accuracy_score(predictions_SVM_8020, test_Y)*100
    SVM_accuracy_8020 = round(SVM_accuracy_8020,1)
```

```
[ ] SVM_accuracy_8020

    95.2
```

```
    style.use('classic')
    cm = confusion_matrix(test_Y, predictions_SVM_8020, labels=model.classes_)
    disp = ConfusionMatrixDisplay(confusion_matrix = cm, display_labels=model.classes_)
    disp.plot()
```

```
    <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f7ca311e850>
```




```
[ ] # Accuracy, Precision, Recall, f1-score

    from sklearn.metrics import classification_report

    print (classification_report(test_Y, predictions_SVM_8020))
```

```
[ ]              precision    recall  f1-score   support

    Negative       0.00      0.00      0.00         1
     Neutral       0.95      1.00      0.98        20

    accuracy                           0.95        21
   macro avg       0.48      0.50      0.49        21
weighted avg       0.91      0.95      0.93        21
```

Naive Bayes

```
[ ] from sklearn.feature_extraction.text import CountVectorizer
    from nltk.tokenize import RegexpTokenizer
    token = RegexpTokenizer(r'[a-zA-Z0-9]+')
    cv = CountVectorizer(stop_words='english',ngram_range = (1,1),tokenizer = token.tokenize)
    text_counts = cv.fit_transform(df['text_traducido'])
```

```
[ ] X_train, X_test, Y_train, Y_test = train_test_split(text_counts, df['sentiment'], test_size=0.25, random_state=5)
```

```
[ ] MNB = MultinomialNB()
    MNB.fit(X_train, Y_train)
```

```
    MultinomialNB()
```

```
[ ] predicted = MNB.predict(X_test)
    accuracy_score = metrics.accuracy_score(predicted, Y_test)
```

```
[ ] print(predicted.shape)
```

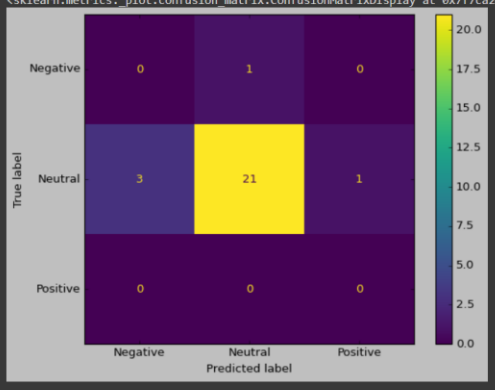
```
    (26,)
```

```
[ ] print(str('{:04.2f}'.format(accuracy_score*100))+'%')
```

```
    80.77%
```

```
[ ] style.use('classic')
    cm = confusion_matrix(Y_test, predicted, labels=MNB.classes_)
    disp = ConfusionMatrixDisplay(confusion_matrix = cm, display_labels=MNB.classes_)
    disp.plot()
```

```
[ ] <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f7ca2d39210>
```




```
[ ] print (classification_report(Y_test, predicted))
```

```
              precision    recall  f1-score   support

    Negative       0.00      0.00      0.00         1
     Neutral       0.95      0.84      0.89        25
    Positive       0.00      0.00      0.00         0

    accuracy                           0.81        26
   macro avg       0.32      0.28      0.30        26
weighted avg       0.92      0.81      0.86        26
```