UNIVERSIDAD DE LOS ANDES
DEPARTAMENTO DE INGENIERIA DE SISTEMAS Y COMPUTACIÓN

PROYECTO 1: ETAPA 1

ISIS 3301 – INTELIGENCIA DE NEGOCIOS

JUAN CAMILO ORTIZ

Grupo 27:

Tomas Ángel – 202020366

Raúl Rincón – 202120414

Luis Felipe Dussán – 201912308

2024-10

Tareas asignadas:

Luis Felipe Dussan

- Modelo: Bag of words
- Resultados texto
- Análisis algoritmos

Raul Santiago Rincon

- Modelo: KNN (k-nearest neighbors)
- Entendimiento negocio y analítico
- Entendimiento y preparación de datos
- Organización y análisis documental
- Video

Tomas Ángel

- Modelo: Naive Bayes
- Entendimiento del negocio
- Líder del proyecto
- Organización y análisis documental
- Mapa de actores
- Preparación de resultados

1. Entendimiento del Negocio

La oportunidad que existe en el análisis de los sentimientos en el sector turístico es invaluable para empresas relacionadas con el turismo, como agencias de viajes, hoteles, y los mismos destinos. Este análisis permite a las empresas comprender mejor la opinión y la experiencia de sus clientes, lo que les permite mejorar sus servicios y aumentar la satisfacción del cliente.
En el caso específico de las agencias de viajes, el análisis de sentimientos puede ayudar a identificar destinos populares, actividades preferidas por los turistas, y aspectos que podrían mejorarse en los paquetes turísticos. Esto les permite adaptar sus ofertas a las preferencias y necesidades de los clientes, aumentando así la probabilidad de satisfacción y fidelización.

Para los hoteles, el análisis de sentimientos puede ser fundamental para mejorar la calidad de sus servicios. Al comprender las opiniones de los clientes sobre aspectos como la limpieza, el servicio, la comida, las actividades y las instalaciones, los hoteles pueden realizar ajustes necesarios para brindar una experiencia más satisfactoria a sus huéspedes.

En los destinos turísticos, el análisis de sentimientos puede ayudar a identificar los aspectos que son atractivos para los turistas y áreas de mejora para aumentar su popularidad. Esto puede incluir la identificación de atracciones turísticas más valoradas, la percepción de la seguridad en el destino, la amabilidad de la gente, los precios en diferentes zonas, y la calidad de la infraestructura turística.

Para analizar sentimientos en el sector turístico, implementaremos un modelo de aprendizaje automático para analizar los textos proporcionados para determinar los aspectos importantes en ellos y comparar en las reseñas y determinar que afecta al turismo frente a opiniones negativas. Se planea usar algoritmos comúnmente usados para análisis de textos como ‘bag of words’, Naive Bayes y KNN. Cada algoritmo presenta características diferentes y objetivos diferentes que seran descritos más adelante.

Los estudiantes de estadística con los que se trabajó para este proyecto interdisciplinario fueron Matías Bayona y Laura Rivera.

Fechas de reuniones:

Reunión lanzamiento y planeación: 18 Marzo - 7:00 PM

Reunión de ideación: 21 marzo - 4:00 PM

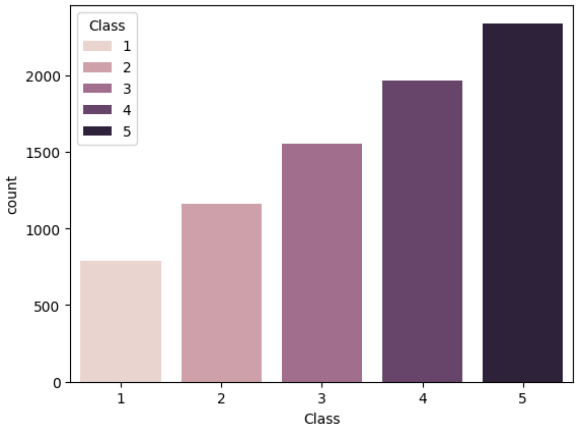
Reuniones de seguimiento: 1, 2, 3 abril - 8:00 PM

Reunión de finalización: 5 abril - 7:00 PM

Canales: Grupo Whatsapp y reunión en zoom

2. Preparación de Datos

En esta sección los datos, aunque son sencillos de interpretar debemos tener especial atención al proceso que tenemos que someterlos para posterior generar modelos de calidad y eficientes. Tenemos dos columnas las cuales hacen referencias a Reseñas que hacen las personas acerca de hoteles u hospedajes y por otro la calificación que se le dio al sitio siendo 1 la menor y 5 la mayor.



En este grafico para empezar se hace una relación entre la clasificación y el conteo de las reseñas que tiene dicha calificación, en este caso vemos un comportamiento lineal respecto a estos datos ya que la mayoría poseen 5 en clasificación mientras que la minoría el 1 de clasificación. Esto es útil de analizar ya que al ver la diferencia de cantidad de datos podemos hacer suposiciones como la calidad de los modelos frente a la etiqueta de 5 con la de 1.

Observamos los datos de las reseñas para comprender que procesa era necesario realizar frente a su limpiado como, por ejemplo; palabras muy extensas sin sentido, o reseñas extremadamente largas. Esto nos llevó a realizar un procesamiento de los datos para poder tener la forma más precisa de cada reseña, quitando palabras sin significado alguno, quitando caracteres que estén en ASCII, haciendo todas las palabras minúsculas, convertir los números en texto y por último eliminar la puntuación de las palabras. Después de realizar el preprocesamiento de los datos, generamos Tokens para cada palabra y guardarlos en una columna para obtenerlas separadas de la reseña original y generar una sin ruido o datos que no nos interesen en los modelos.

3. Modelado y evaluación

Para el modelado se utilizaron dos tipos de vectorización; count vectorizer y TFIDF (Term Frequency and Inverse Document Frequency) sobre 3 algoritmos diferentes para realizar el aprendizaje automatizado. Para algunos casos, el count vectorizer fue más eficiente que el TFIDF, mientras que en otros casos fue el revés.

Por un lado, CountVectorizer cuenta cuántas veces aparece cada palabra en una reseña, lo que puede ser útil para ver qué palabras son más frecuentes en cada reseña en particular. Por otro lado, TfidfVectorizer también considera cuán importante es una palabra en todo el conjunto de reseñas, no solo en una reseña específica.

Al realizar ambas formas de vectorización en un mismo algoritmo podemos tener una mejor idea de qué palabras son importantes tanto localmente (en una reseña específica) como globalmente (en todas las reseñas). Esto nos ayuda a entender mejor cómo las palabras afectan las opiniones de las personas sobre destinos turísticos, y muestra como ambas formas de vectorizar las palabras generas resultados diferentes en los modelos realizados.

El éxito de cada modelo se midió con la precisión obtenida, el f1 score, y la matriz de confusión utilizando las 5 clases encontradas en los datos para hacerla. La precisión del modelo indica la proporción de casos positivos correctamente identificados entre todas las instancias que el modelo ha etiquetado como positivas sobre aquellos identificados como falsos positivos. El F1 score es la media armónica de precisión y recall (es la proporción de casos positivos correctamente identificados respecto a todos los casos positivos reales). Por último, la matriz de confusión permite dar una visión detallada de como el modelo está clasificando las instancias de cada clase y midiendo el rendimiento de este.

Bag of words(Luis Felipe Dussán)

El algoritmo *Bag of Words (BoW)* es una técnica utilizada en procesamiento de lenguaje natural (NLP) para representar documentos de texto como vectores numéricos. La idea principal detrás de BoW es tratar cada documento como un "saco" (bag) de palabras, sin tener en cuenta el orden o la estructura gramatical, y simplemente contar la frecuencia de ocurrencia de cada palabra en el documento.

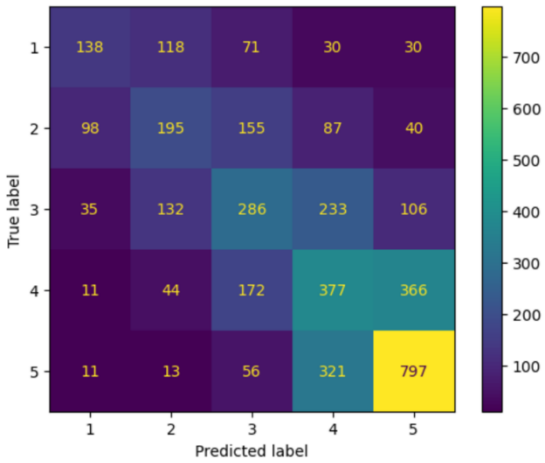
Uso de diferentes técnicas de vectorización dentro de este algoritmo.

i. CountVectorizer:

CountVectorizer es una herramienta en Scikit-learn que convierte una colección de documentos de texto en una matriz de recuentos de tokens (palabras).

Para cada documento, CountVectorizer construye un vector donde cada elemento del vector representa el recuento de ocurrencias de una palabra específica en el documento.

Por ejemplo, si tenemos un conjunto de documentos: "El cielo es azul", "La hierba es verde", "El cielo es azul y la hierba es verde", CountVectorizer convertirá estos documentos en vectores contando las palabras únicas: `[1, 1, 1, 0, 0]`, `[0, 0, 0, 1, 1]`, `[1, 1, 1, 1, 1]`, respectivamente.



Ilustracion 1: Matriz de confusion de los resultados sacados por Bag of words

Usando CountVectorizer obtuvimos un f1 score de 0,452, y un test accuracy de 0,457. Con esto podemos decir que el modelo construido utilizando CountVectorizer logró un rendimiento aceptable en la tarea de clasificación. El valor de F1-score de 0.452 indica que el modelo tiene un buen equilibrio entre precisión y recall, mientras que el accuracy de 0.457 sugiere que el modelo fue capaz de clasificar correctamente aproximadamente el 45.7% de las muestras en el conjunto de prueba.

Hay que revisar las otras técnicas y analizar sus rendimientos para hacer una comparación.

ii. TfidfVectorizer:

TfidfVectorizer es similar a CountVectorizer, pero en lugar de contar las ocurrencias de cada palabra en los documentos, calcula la puntuación TF-IDF (Term Frequency-Inverse Document Frequency) de cada palabra.

TF-IDF es una medida que combina la frecuencia de una palabra en un documento (TF) con la rareza de la palabra en el corpus completo (IDF). Esto ayuda a reducir el peso de las palabras comunes que aparecen en muchos documentos y aumenta el peso de las palabras que son específicas de un documento.

Los vectores resultantes contienen las puntuaciones TF-IDF de cada palabra en vez de los recuentos de palabras.

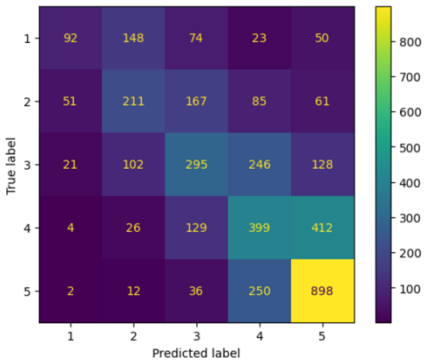


Ilustración 2: 1Matriz de confusión de los resultados obtenidos usando TF IDF Vectorizer en Bag of Words

Usando TF-IDF Vectorizer, obtuvimos un F1-score de 0.469 y un test accuracy de 0.4831. Con esto, podemos decir que el modelo construido con TF-IDF Vectorizer ha mostrado una mejora en comparación con el modelo anterior construido con CountVectorizer.

El valor de F1-score de 0.469 indica un mejor equilibrio entre precisión y recall en comparación con el modelo anterior. Esto sugiere que el modelo construido con TF-IDF Vectorizer puede ser más eficaz para clasificar correctamente las muestras de prueba, considerando tanto la precisión como la exhaustividad de las predicciones.

Además, el accuracy de 0.4831 indica que el modelo pudo clasificar correctamente aproximadamente el 48.31% de las muestras en el conjunto de prueba, lo que representa una mejora respecto al modelo anterior.

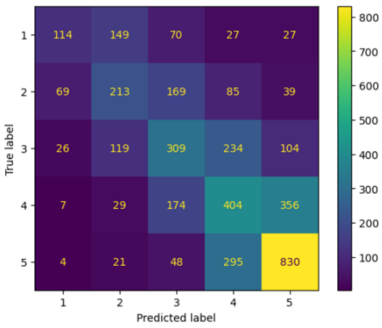
iii. TfidfTransformer:

TfidfTransformer se utiliza junto con CountVectorizer para convertir los recuentos de palabras generados por CountVectorizer en puntuaciones TF-IDF.

A diferencia de TfidfVectorizer, TfidfTransformer no realiza la tokenización inicial de los documentos, ni construye un vocabulario. Simplemente transforma las matrices de recuentos generadas por CountVectorizer en matrices de puntuaciones TF-IDF.

Esto significa que debes utilizar primero CountVectorizer para generar las matrices de recuentos, y luego usar TfidfTransformer para convertir esas matrices en matrices de puntuaciones TF-IDF.

Usando TF-IDF Transformer, obtuvimos un F1-score de 0.4711 y un test accuracy de 0.476. Con esto, podemos decir que el modelo construido con TF-IDF Transformer se encuentra en un punto intermedio en comparación con los dos modelos anteriores.



Ilustracion 3: Matriz de confusion usando bag of words y TFIDF transformer

El valor de F1-score de 0.4711 muestra una mejora en comparación con el modelo construido con CountVectorizer (F1-score de 0.452), pero es ligeramente inferior al F1-score del modelo construido con TF-IDF Vectorizer (F1-score de 0.469). Esto sugiere que el modelo con TF-IDF Transformer logra un buen equilibrio entre precisión y recall, pero no supera significativamente al modelo con TF-IDF Vectorizer en términos de rendimiento.

Además, el accuracy de 0.476 indica que el modelo pudo clasificar correctamente aproximadamente el 47.6% de las muestras en el conjunto de prueba, lo que también representa una mejora en comparación con el modelo construido con CountVectorizer (accuracy de 0.457), pero es ligeramente inferior al accuracy del modelo construido con TF-IDF Vectorizer (accuracy de 0.4831).

Naive Bayes (Tomas Angel)

El algoritmo Naive Bayes es una técnica de aprendizaje automático que se puede utilizar para analizar sentimientos en un conjunto de datos. En el caso de un data frame que contiene una columna de reseñas y una columna de clase (5 siento una buena resena y 1 siendo una mala), Naive Bayes puede ser utilizado para predecir el sentimiento de una nueva reseña.

Una vez que se ha preparado el conjunto de datos y se han creado las características relevantes, se puede entrenar el modelo Naive Bayes para predecir el sentimiento de las reseñas en función de las características de entrada. El modelo analiza las características y encuentra patrones para determinar si una reseña tiene un sentimiento positivo o negativo.

Una de las ventajas del algoritmo Naive Bayes es que es relativamente rápido de entrenar y puede manejar grandes conjuntos de datos con una alta dimensionalidad. Además, puede manejar datos faltantes y funciona bien con datos desequilibrados. Sin embargo, una limitación de Naive Bayes es que se basa en la suposición de que todas las características son independientes entre sí, lo que puede no ser cierto en todos los casos.

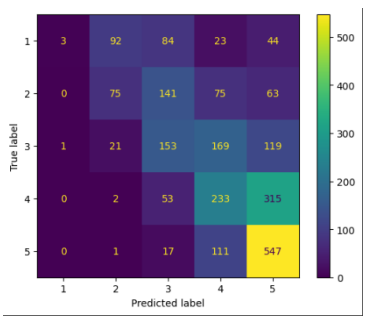


Ilustración 4: Matriz de confusion de los resultados obtenidos usando TFIDF en Naive Bayes

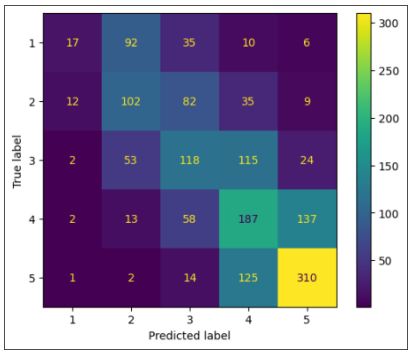


Ilustración 5: Matriz de confusion de los resultados obtenidos usando TFIDF en Naive Bayes

Los resultados obtenidos indican que tanto CountVectorizer como TFIDF no son métodos tan efectivos para el análisis de sentimientos en el conjunto de datos utilizado.

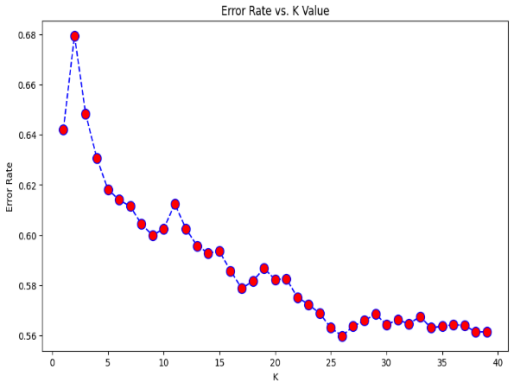
En particular, el modelo que utilizó CountVectorizer obtuvo una precisión mayor que el modelo que utilizó TFIDF, con un valor de 0.4702 frente a 0.4316. El modelo con CountVectorizer también tuvo un valor de recall menor (0.41 frente a 0.35) lo que indica que tiene una mayor probabilidad de clasificar las reseñas como negativas cuando en realidad son positivas.

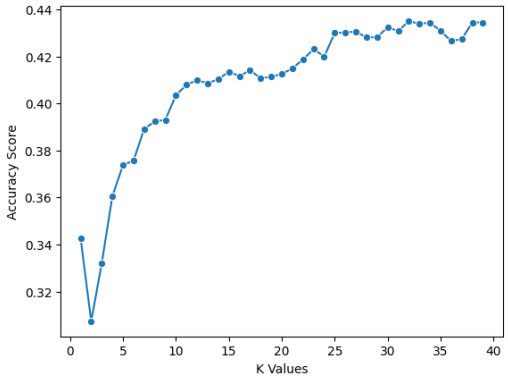
Sin embargo, el valor F1, que es una medida combinada de precisión y recall, fue mayor para el modelo con CountVectorizer (0.4829) en comparación con el modelo con TFIDF (0.3749). En general, estos resultados sugieren que el modelo con CountVectorizer es mejor para predecir sentimientos en el conjunto de datos usado.
Es importante tener en cuenta que los resultados obtenidos pueden depender del conjunto de datos utilizado y de los parámetros seleccionados para cada método. Por ello, se recomienda explorar opciones y evaluar los resultados en diferentes conjuntos de datos para determinar la técnica de procesamiento de lenguaje natural y el modelo de aprendizaje automático adecuado para análisis de sentimientos. En este caso, el modelo con CountVectorizer muestra una mejora en la precisión en general del modelo, a costa de una reducción en el recall. Es importante evaluar en qué casos es más importante la precisión o el recall para determinar el mejor enfoque.

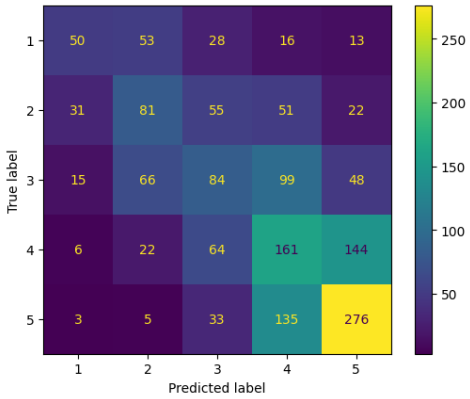
KNN (Raúl Rincón)

KNN es un algoritmo de aprendizaje supervisado que normalmente se utiliza para clasificación y regresión, su funcionamiento consta de instancias, esto significa que no aprende explícitamente un modelo, sino que clasifica cada dato nuevo en función de las características de sus vecinos mas cercanos en el conjunto de datos de entrenamiento.
La “distancia” Entre los datos se mide generalmente utilizando métricas como la distancia euclidiana o de Manhattan. Sin embargo, el parámetro más importante dentro del algoritmo es “K”, esto es un factor determinante de como clasificará un nuevo dato. Un k pequeño significa que la clasificación se basa en un pequeño número de vecinos más cercanos, lo que puede llevará a una alta varianza y aun ajuste excesivo. En cambio, un K grande nos puede dar una región de decisión mas amplia y tener un alto sesgo.

Por la parte cuantitativa se calculó una estimación del error generado respecto al K value para de esa manera escoger aquel mas adecuado para el modelo y de esa manera se pudo determinar también con los K Values el acurracy score y determinar cual K genera el mas alto para posteriormente usarlos como parámetros y generar la mejor clasificación posible







	precision	recall	f1-score	support
1	0.48	0.31	0.38	160
2	0.36	0.34	0.35	240
3	0.32	0.27	0.29	312
4	0.35	0.41	0.37	397
5	0.55	0.61	0.58	452
accuracy			0.42	1561
macro avg	0.41	0.39	0.39	1561
weighted avg	0.41	0.42	0.41	1561

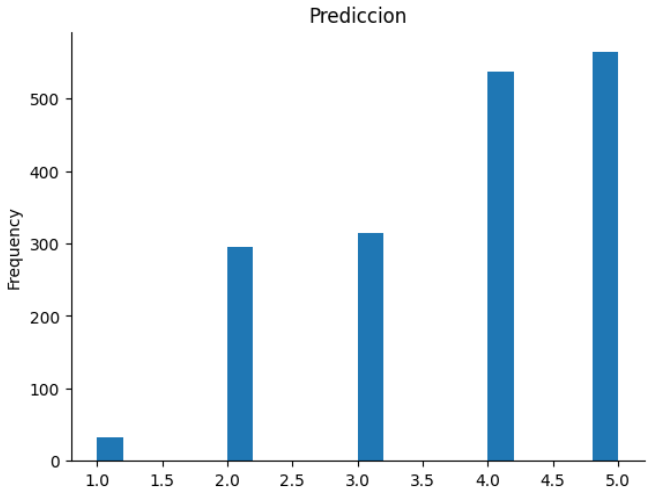
Como podemos ver la matriz de confusión y los valores obtenidos de precisión, recall, f1-score y support, podemos concluir con este modelo de clasificación que, aunque para la mayoría de las calificaciones altas como 5 y 4 produce buenas métricas y podemos determinar que el modelo podría fácilmente llegar a clasificar estas reseñas con facilidad, aquellas que tienen de 1 a 3 en reseñas el modelo difícilmente las llega a concluir.

La razón de escoger el algoritmo es que puede ser útil para determinar las características que hacen que una reseña sea mala puesto que clasifica las reseñas basándose en la similitud de sus características con las de otras reseñas en el conjunto de datos de entrenamiento. Si las reseñas malas tienden a tener características similares, KNN puede identificar estos patrones y predecir la calificación de una nueva reseña. Sin embargo, KNN puede ser computacionalmente costoso para conjuntos de datos grandes, esta es la principal razón por la cual se utilizó TFIDF y no CountVectorizer como vectorizador ya que el segundo generaba demasiadas columnas las cuales el algoritmo ni siquiera haciendo uso de técnicas como PCA podía llegar a calcular puesto que saturaba la memoria. Además, puede ser sensible a la escala de las características, por lo que lo recomendado es normalizar muy bien los datos antes para que sea efectivo en la clasificación.

4. Resultados

Para poder concluir, analizamos profundamente cual algoritmo podría ser de mayor conveniencia, esto se hizo mediante las métricas de acurracy y F1 score. Estas métricas, sobre todo el f1 nos ayudan a determinar qué tan completo y preciso es el modelo facilitando el cálculo de falsos positivos y

falsos negativos en el modelo. Un F1 alto significa un algoritmo que hace bien su trabajo clasificando o haciendo predicciones. Nosotros escogemos el algoritmo de Naive Bayes que obtuvo un f1 de 0.48 y podemos observar que las predicciones realizadas se ven del siguiente modo:



La predicción para este algoritmo es la mejor para las clasificaciones 4 y 5, pero al modelo le cuesta encontrar una predicción entre las reseñas y las clasificaciones de 1.
Para el negocio podríamos decir que la mayoría de las clasificaciones están dentro de 5 y 4, pero todavía hay críticas donde las calificaciones no son muy buenas. Para las reseñas de 1, no hubo mucha frecuencia según las predicciones sacadas, lo cual puede indicar que las calificaciones de 1 no son muy frecuentes o que el modelo se equivoca al momento de encontrar dichas reseñas. Las recomendaciones que le damos al negocio y los actores involucrados es ver particularmente aquellas reseñas que sean menores a 4 y revisar los comentarios para mejorar en los aspectos mencionados.

5. Trabajo individual

Consideramos que cada integrante trabajó igual que el resto porque cada uno se encargó de realizar un modelo y realizar el análisis de este. Las demás actividades se realizaron en conjunto, aunque no todos aportamos en los mismos aspectos del proyecto, cada uno aportó en una parte diferente del documento y el notebook, pero realizando un buen trabajo. Considerando lo anterior, consideramos que Tomas Angel fue el líder del proyecto y un líder de negocio; Raul Rincon fue otro líder de negocio y líder de datos; Luis Felipe Dussán fue el líder analítico.

Además, todos los integrantes asistieron a todas las reuniones y estuvieron pendientes del estado del proyecto, por lo que consideramos que, si tuviéramos que distribuir 100 puntos entre los tres integrantes, cada integrante tendría 33 puntos.

6. Trabajo interdisciplinario