Proyecto 1

Kevin Andres Babativa Santos
Nicolas Angarita Moreno
Nicolas Arturo Alvarado Vargas

Comprensión del negocio y enfoque analítico

La empresa Yelp quiere poder predecir cuantas estrellas va a tener un comentario según su contenido. Se quiere poder tener un modelo con más del 50% de precisión para garantizar que valga la pena usarlo.

Oportunidad/problema Negocio	Teniendo una gran muestra de comentarios con sus respectivas estrellas, es posible entrenar un modelo que pueda predecir el número de estrellas según una reseña de un producto. Yelp podría usar este modelo para analizar textos en redes sociales, donde no hay calificaciones, para tener una medida cuantitativa de la satisfacción del público, por otra parte, un tercero ajeno a Yelp podría usar estos datos para realizar sus propios modelos y poder calificar sus calificaciones que no están etiquedas.
Enfoque analítico (Descripción del requerimiento desde el punto de vista de aprendizaje de máquina)	Se requiere entrenar un modelo de procesamiento de lenguaje natural que pueda predecir cuantas estrellas va a tener un comentario con una precisión de más del 50%.
Organización y rol dentro de ella que se beneficia con la oportunidad definida	Todos los comentarios que se van a usar de entrenamiento son de la empresa Yelp, sin embargo, al Yelp agrupar reseñas de sectores distintos puede ser valioso para empresas de cualquier sector. La posibilidad de clasificar de manera cuantitativa los comentarios de forma automatizada le da a las

	empresas una oportunidad de analizar rápidamente los comentarios de sus clientes.
Técnicas y algoritmos a utilizar	Para la vectorización de los textos se utilizará CountVectorizer y TfidfVectorizer de Scikit-learn. Para la tarea de clasificación se utilizarán los algoritmos RandomForest, DecisionTree y Naive Bayes, todos de Scikit-learn. Para Naive Bayes es necesario utilizar una estrategia de OneVsRest para que sea posible clasificar utilizando Bayes, lo anteriormente descrito es una transformación del problema. Finalmente para la tokenización se usa la función word_tokenize de la librería NLTK.

Entendimiento y preparación de los datos

Se cuenta con 314562 datos para el entrenamiento del modelo. Cada dato consiste de un texto con su respectivo número de estrellas. De la totalidad de los datos usamos 251649 (80%) para el entrenamiento. Para la tokenización de los datos se usa la librería NLTK con su función word_tokenize, utilizamos esta librería dado que es la que mejor se acomoda a los datos usados. Los stop words son los de la librería NLTK para el idioma inglés, dado que los datos están en este idioma. Se definió una propia función de tokenización basada en word_tokenize de NLTK, en esta función se eliminan caracteres especiales como \n, se *lematizan* (proceso por el cual se obtiene la raíz de un verbo) y el *stemming* (proceso en el cual las palabras en plurar se convierten en singular), por último, se eliminan las palabras menores o iguales en tamaño a 2, dado que estas no aportan información al modelo. Para la tarea de vectorización se usa CountVectorizer (BoW) y TfidfVectorizer, y se compararan los resiltados para determianr cual es mejor.

Modelado y evaluación

Los algoritmos que se aplicaron fueron:

DecisionTree: Este es un algoritmo que usa árboles de decisión para hacer tareas de clasificación. Las ramas de estos árboles representan las clases en las que se clasifican los datos. Este algoritmo fue escogido por su simplicidad y por la valiosa información que otorga, este algoritmo nos permite saber las importancias de cada token dentro de un texto, con dicha información es posible tener una visión general sobre que valores son los que determinan que una reseña tenga cierta puntuación.

RandomForest: Este algoritmo consiste en crear varios árboles de decisión y ver cómo se comportan los datos en cada uno para asignarles una clase. Dado que este algoritmo es una mejora del algoritmo de Decision Tree realizaer ambos conjuntamente proveen información valiosa (como lo es la importancia de un token dentro de un texto), así como una mejoría en sus métricas.

Los dos algoritmos anteriores son algoritmos adaptables por sí mismos a problemas de clasificación, por lo cual su implementación es de una complejidad menor, estos algoritmos sirven naturalmente para problemas *multi-label*. Por otra parte, Naive Bayes es un algoritmo que tiene que ser transformado para responder a un problema *multi-label,* esta transformación se basa en la transformación de un problema con múltiples label a varias tareas con un solo label. Para lograr esta transformación se usa la estrategia OneVsRest, esta estrategia se basa en entrenar un solo clasificador para cada clase en el cual se trate a la clase en cuestión como positiva mientras las otras como negativas. Esta estrategia implementa el método de la relevación binaria, es decir cada clase es independiente de las otras clases.

Naive Bayes: Este es un algoritmo que implementa el teorema de Bayes para hacer clasificación según características. Este algoritmo usa fórmulas de probabilidad condicional para determinar en qué clase poner una unidad de información. Como se explicó anteriormente, para ejecutar este algoritmo es necesario realizar una tarea de transformación del problema multi clase, esto se logró utilizando la función OneVsRestClassifier de scikit learn. Se escogió este algoritmo dado que presenta unas métricas decentes en un tiempo prudencial (<20 minutos), dado que los problemas de transformación representan una mayor complejidad computacional, por lo tanto, demorarán un tiempo mucho mayor (>2 horas).

Resultados

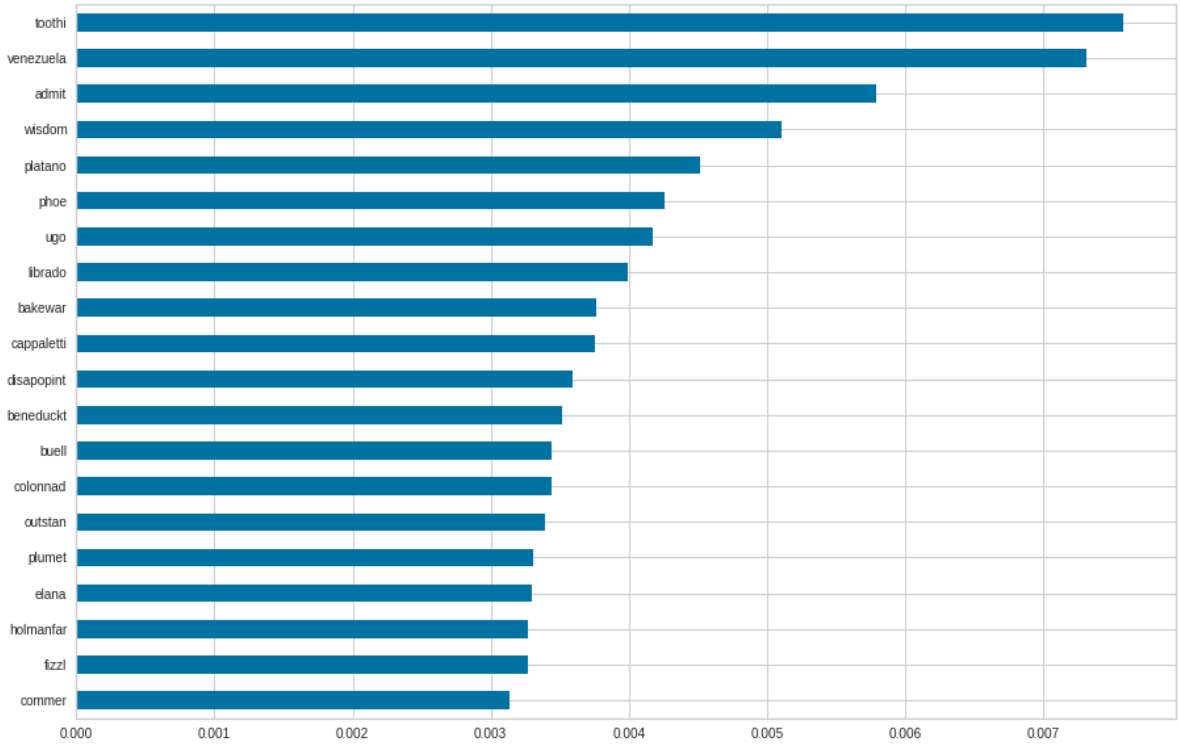
En cuestión de métricas se encontraron los siguientes resulltados para CountVectorizer:

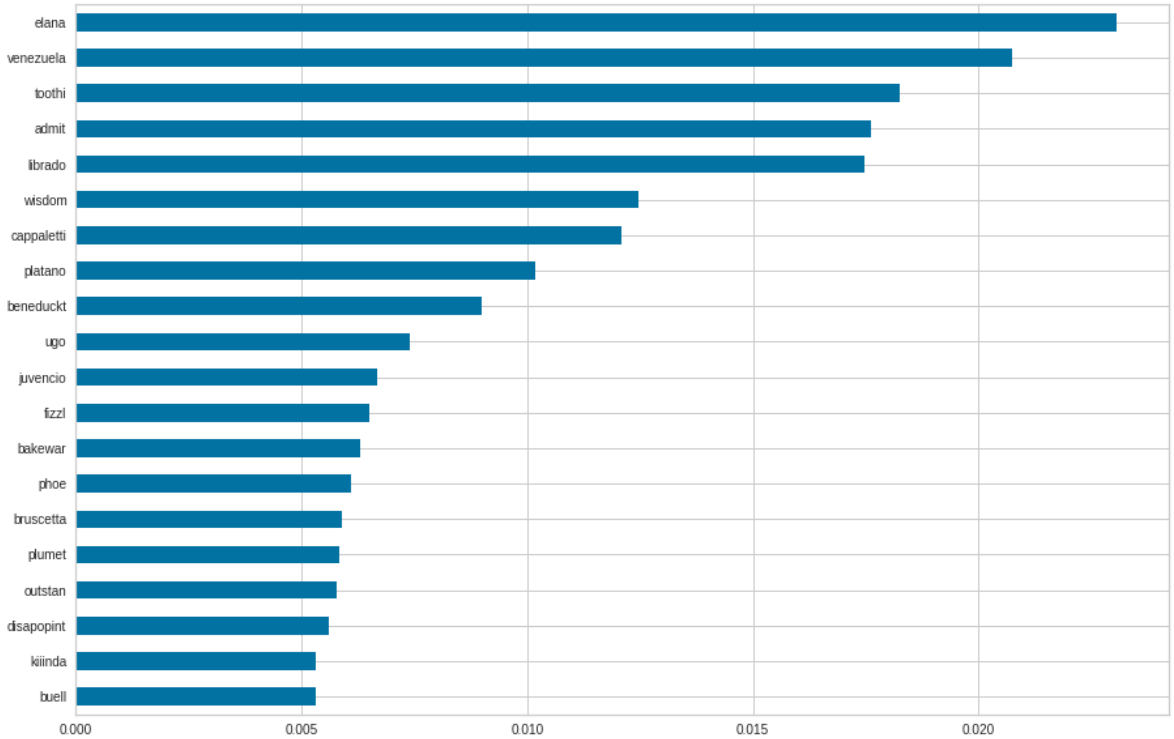
	Naive Bayes	**DecisionTree**	**RandomForest**
Micro F1 Score	0.63	0.52	0.60
Macro F1 Score	0.50	0.40	0.36
Hamming Loss	0.37	0.48	0.40
F1 Score weighted	0.62	0.51	0.51

En cuestión de métricas se encontraron los siguientes resulltados para TfidfVectorizer:

	Naive Bayes	**DecisionTree**	**RandomForest**
Micro F1 Score	0.58	0.51	0.60
Macro F1 Score	0.31	0.39	0.35
Hamming Loss	0.42	0.49	0.40
F1 Score weighted	0.47	0.51	0.51

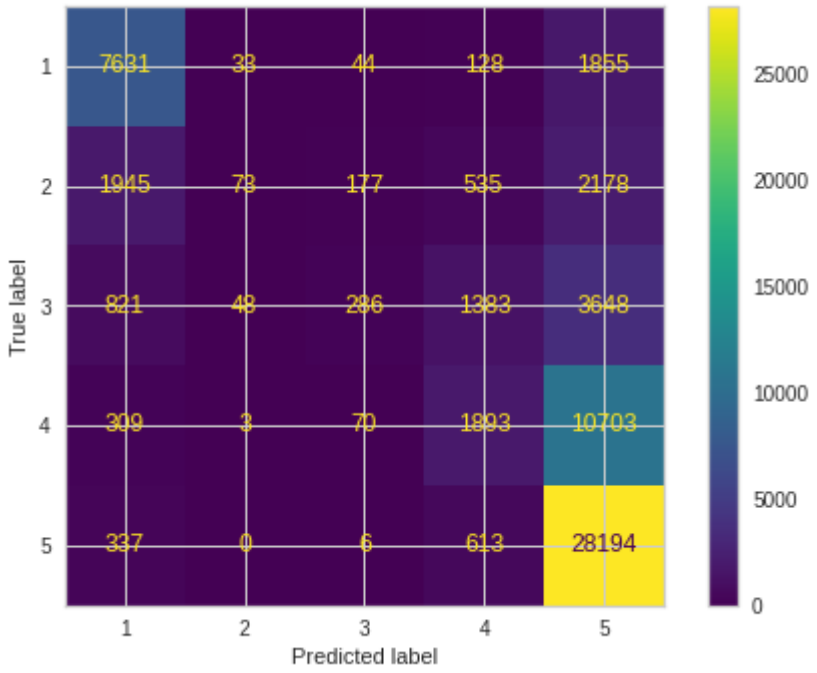
Tal como se esperaba, el algoritmo de Naive Bayes es el que presenta mejores métricas, RandomForest y DecisionTree presentan peor desempeño, sin embargo, la utilidad de estos algoritmos va en las conclusiones que se pueden obtener de ellos.

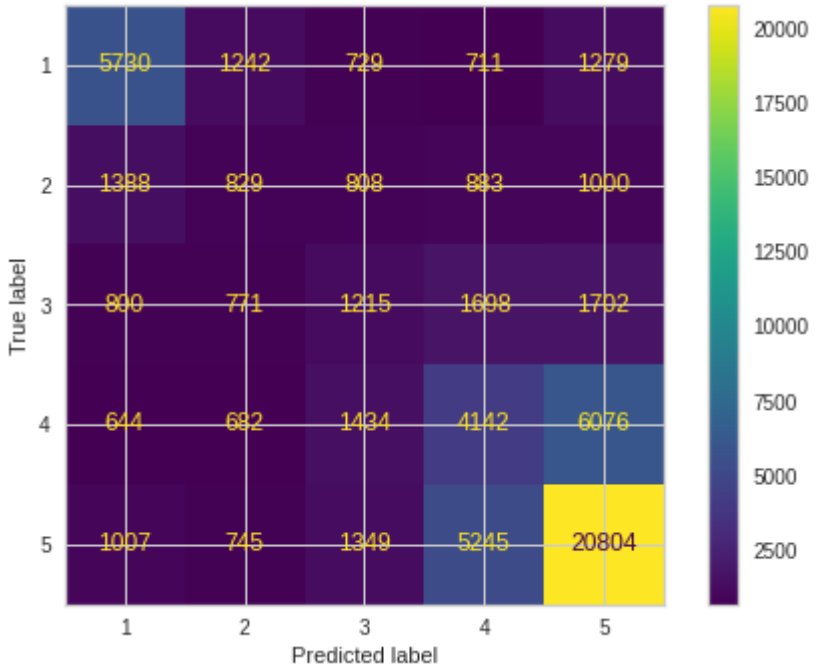




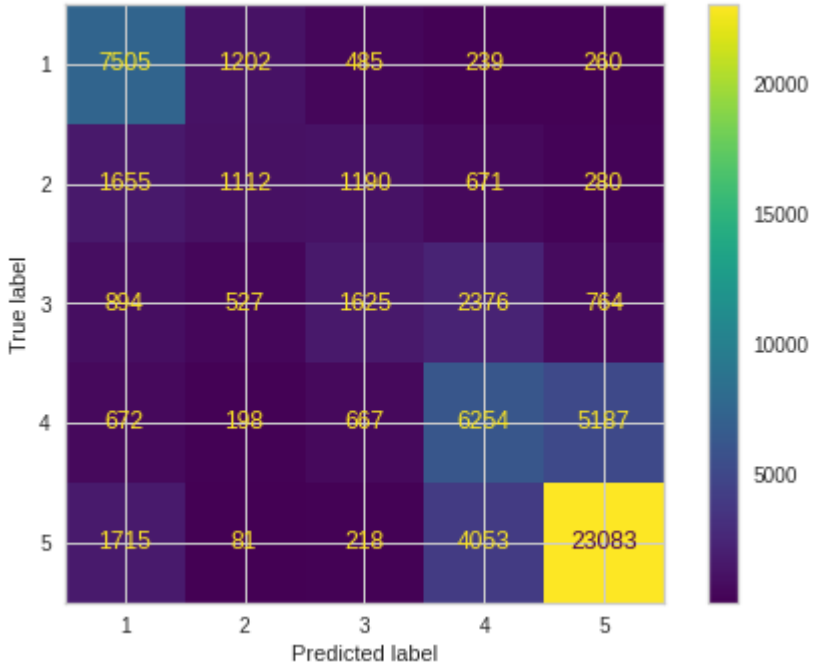
De las gráficas anteriores se puede ver la importancia de algunas features que nos den información importante a la hora de determinar una calificación, el token *Venezuela* tiene importancia en el modelo, esto se debe a que dentro del dataset de reseñas hay reseñas de comida y dicho estilo de comida es popular dentro de los datos. También sobresalen adjetivos como outstanding o wisdom, en el caso de wisdom se refiere a la expresión de *wisdom tooth* que se refiere las cordales, dado que hay una cantidad considerable de reseñas odontológicas, en estos casos las

personas creen que vale la pena reseñar su proceso de extracción de muelas del juicio.





Dado que los algoritmos de RandomForest y DecisionTree presentan mejor desempeño en clasificar las categorías 1 y 5, es consecuente con la necesidad de calificar un muy buen servicio odontológico o uno muy malo.
El de Naive Bayes estuvo logro la meta, pero por poco por lo que se podría intentar mejorar haciendo cambios en el proceso de entrenamiento. Para los otros dos algoritmos, no están lo suficientemente cerca de la meta por lo que no deben ser tomado en cuenta.



El modelo creado con Naive Beyes resulta bueno en predecir textos que tienen 1, 4 y 5 estrellas. Esto, aunque no es lo ideal, ya puede ser de ayuda para una empresa que quiera clasificar comentarios como buenos o malos.
Se le sugiere a cualquier empresa que quiera usar este modelo que lo use de esa forma, para definir comentarios buenos y malos teniendo en cuenta los comentarios que tienen 1,4 y 5 estrellas.
Estos resultados se dieron por la cantidad desproporcionada de comentarios con 5 y 1 estrellas.

Trabajo en equipo

Se definieron los siguientes roles:

- Líder de proyecto: Kevin Andres Babativa Santos
- Líder de negocio: Nicolas Angarita Moreno
- Líder de datos: Kevin Andres Babativa Santos
- Líder de analítica: Nicolas Arturo Alvarado Vargas

Las tareas que realizó cada rol fueron las siguientes:

Líder de proyecto	**Líder de negocio**	**Líder de datos**	**Líder de analítica**
Coordinar reuniones para el seguimiento de las tareas	Identificar la oportunidad de negocio	Encargado del entendimiento de los datos	Garantizar la calidad de los modelos
Evaluación del progreso de las tareas	Garantizar la calidad de la entrega final	Comunicar de manera eficaz la organización de los datos al equipo	Elección de los mejores modelos para el proyecto

Coordinar la entrega final del proyecto	Sacar insights de los resultados	Realizar el preprocesamiento de los datos	Tomar las mejores decisiones de cara a los modelos, tales como hiperparametros.

Video:

https://youtu.be/P7tYnbtCGQs