Which American low cost carrier is better for you: Jet Blue or Southwest?

Sentiment Analysis on Twitter about airlines from USA

Final report:
Refactored-waddle

Which American low cost carrier is better for you: Jet Blue or Southwest?

Sentiment Analysis on Twitter about airlines from USA

Contenido

1. Problem definition 2
2. System design 2
3. Data source, ingestion, and feature engineering 3
4. Machine learning component 4
5. System evaluation 5
6. Model Serving 6
7. Reflection 7
8. Broader Impacts 7

1. Problem definition

We believe that users of Low Cost Carriers (LCC) have incomplete information about the service they purchase with a plane ticket. They know objective information, such as the price and the characteristics of the service that the company provides, however they do not know the experience of users who have already acquired the services of the LCC previously. Therefore, we consider LCC users could improve their decisions if they had knowledge of the experience of other users. That is, by including subjective information about the service provided by the airlines and details of the flights extracted from Twitter.

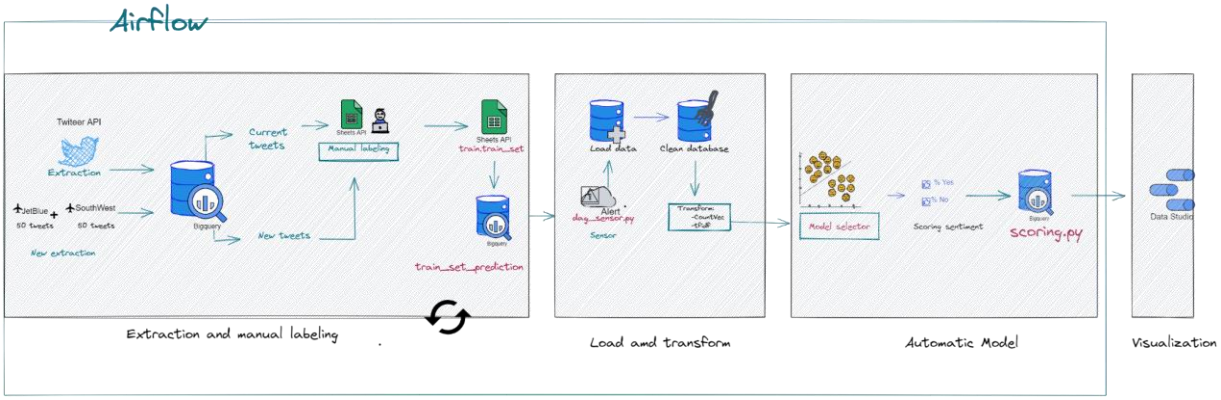
Sentiment analysis is one of the most popular ways to analyze text, such as customer support issues, online reviews, live chats, customer feedback, and customer support management, it can help companies stay on top of customer satisfaction. This technique is part of Natural Language Processing or NLP (Amigo, 2021; MonkeyLearn, s.f. Retamar, s.f.; Hegde, et al, 2018). In the case of airlines, Amigo (2021), Anitsal et al (2017) y Kumar (2019), assess the satisfaction of airline users.

We perform sentiment analysis on tweets about two low-cost airlines, JetBlue and SouthWest, to provide users with additional qualitative information besides the data they could find on the airlines' websites or the one provided by travel agencies. We offer a product that gives customer experience as an element to decide between these two similar low-cost airlines. Users' perception of airline service is a relevant variable that would help compare the service provided by airlines and facilitate users' decisions. Our application allows knowing the sentiment of users regarding the service of these airlines, which allows them to make an informed decision.

2. System design

The main objective of this system is to provide an intuitive, easy, and low-use resources tool to those low-cost airline users interested in making decisions related to JetBlue or SouthWest lines. The architecture designed to process information in an automated way is based on Google Cloud resources such as Cloud Storage, Big Query, Airflow, and Google Data Sheets.
The goal of the architecture is to standardize our process to assign a score to each tweet and show it to users. Our infrastructure manages each of the steps to reach that goal. Thus, we linked extraction, storage, processing, transformation, analysis, results, and visualization.
To achieve this, we use tools that we learned throughout the Data Product Architecture course and others that we explored by ourselves.

Figure 1. System components



For the **extraction and labeling** of tweets, we use Twitter API, Big Query, and Google Sheets. Twitter API helps with the network service communication to the network service. Also, it helps us to extract a limited number of tweets. BigQuery is a data warehouse that is part of Google Cloud. In this section, we use Google Sheets to allocate our extracted information in a new spreadsheet. We obtained the tweets mentioned with the names of the airlines we are analyzing, BlueJet and SouthWest, and stored them in BigQuery. And then, once we have gotten the tweets, we manually tag them in Google Sheets.

BigQuery performs **data loading and transformation**. Once the tweets have been tagged, we send the signal through a sensor to unleash the following process, which involves cleaning the data and transforming it into a sparse nuance and a matrix of inverted weights.

The **modeling** execution is through pipelines developed with Python, and the score obtained from the modeling is stored in Big Query. We execute four pipelines and select the one with the best mean performance (Brier score) of five folds built with stratified cross-validation. In the next section, we detail all the information about these pipelines. The score range obtained is between 0 and 1, the closer to 1, the more positive the sentiment, and the closer to zero, the more negative. We sketch the calculated score model with our tool Data Studio. In the visualization, we show some examples of the best and worst scores of each airline.

Airflow is our **orchestrator** used to schedule our tasks and monitor our workflows. This platform allowed us to repeat the extraction, load, and transform our data regularly. This tool is programmed to start with the extraction and runs the model as we request.

If further information is required, we placed the support and documentation of the developed process in a GitHub repository called Refactored-Waddle: https://github.com/vserranoc/refactored-waddle

3. Data source, ingestion, and feature engineering

We analyze the sentiments obtained from the conversations that occur on Twitter, the social network that is the most popular in the world. We use the data of Twitter because it had more than 330 million monthly active users, at the beginning of 2019, and it is used by the media to know the opinion of users about certain services. We extract the tweets that users leave towards airlines. As users of the social network, we request the developer account in the link: https://developer.twitter.com/en/apps. We access the “Keys and Access Tokens” tab and click on the “Generate My Access Token and Token Secret” button, these are credentials to be able to access the Twitter API and extract the tweets that are needed.

We extract 4,982 tweets mentioning @JetBlue and @SouthwestAir from the Twitter API for the last week of March 2022. After removing duplicates, we cleaned the database: convert text to lowercase, remove all URLs, remove Twitter accounts (@), remove specific names, trim blank spaces, remove punctuations, symbols, and numbers, remove non-English tweets, remove hashtags (#) and convert emojis to text. We filtered out retweets, we create a dataset with 2,695 tweets, of which 76% are negative and 24% are positive. During extraction, we define the following variables:

- user (Twitter user account originating the tweet)

- text (Full-text content in each tweet)
- followers (Number of user accounts that follow the given user)
- location (Geographic location from the given user)
- verified (Indicator equal 1 if the account has this condition).

4. Machine learning component

We use traditional models, Logistic regression, and Naive Bayes, these algorithms were selected because of their scalability and the common use in classical machine learning for sentiment analysis. Logistic regression is a type of regression analysis, which is used to predict the outcome of a categorical variable, in our case sentiment (positive or negative), based on the independent or predictor variables, for us these are the words that we obtained from Twitter. On the other hand, Naive Bayes applies the Bayes theorem with a Naive assumption of no relationship between different features and this probabilistic classifier helps us to learn about the pattern of an examined set of data previously categorized.

Before pouring the core into the model, we generate two methods for vectorizing: Sparse Matrix y tf-idf. Sparse Matrix is built by vectorizing the frequency of the unique words that we have in the entire core. In other words, a dictionary of unique words is generated, from which a matrix is built that only contains numbers, which in turn correspond to the frequency of each word in each of the tweets. To build this matrix automatically, we use CountVectorizer, which is a tool provided by the scikit-learn library in Python. On the other hand, Tf-idf allows us to calculate the weight of the words according to their frequency and apply the inverse of this. Thus, the weirdest words will receive the highest weight.

We combine the two ways of vectorizing the core and the two machine learning models (logistic regression and naive Bayes), with which we obtain 4 pipelines to optimize by stratified cross-validation:

- Logistic regression - CountVectorizer,
- Logistic regression - Tf-idf,
- Naive Bayes CountVectorizer,
- Naive Bayes Tf-idf.

The performance of each of these models is measured through 5 metrics: recall, accuracy, F1_ score, AUC, and Brier.

Table1. Model comparison

model	preprocessing	recall	accuracy	f1_score	auc	brier
LogisticRegression	CountVec	0.718571	0.853803	0.718571	0.890704	0.104238
Naive Bayes	CountVec	0.705714	0.847124	0.705714	0.881749	0.127636
LogisticRegression	tfidf	0.712857	0.850835	0.712857	0.893292	0.139246
Naive Bayes	tfidf	0.630000	0.807792	0.630000	0.850711	0.189950

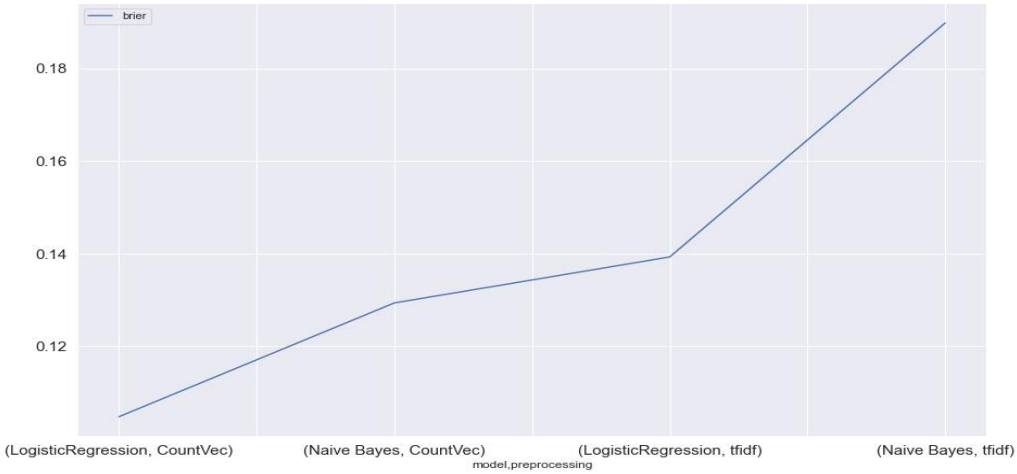
The iterative development of our models could be described as follows: we take a text sample, clean, vectorize and perform the modeling, to predict the sentiment as negative or positive with future tweets. Specifically, we connect to labeled information in Google Sheets and upload it to BQ, we extract JetBlue and Southwest tweets to BQ monthly, subsequently, sample 50 tweets for every airline (unlabeled), and join them with existing label data in BQ for model training and predictions, and upload train set back to Google Sheets for labeling. Immediately, we generate an alert when the previous step has been completed for later calculation scoring, which follows the next steps: load data, load artifacts, clean data, transform the data (generate sparse matrix), score sentiment, and save scoring in big query.

5. System evaluation

For data extraction, as mentioned before, we use the Tweepy API. We have detected that this extraction runs correctly, and we got the desired results. We obtained around 5,000 Tweets from users' accounts mentioned by JetBlue and Southwest. There is an important restriction, as Twitter levies a rate limit on the number of requests made through Tweepy. Only 900 requests can be executed over any 15 minutes allowed. Also, it limits us to get only the last 3200 tweets in a timeline. But our biggest limitation is under the labeling process. In this part we are not able to tag all of our tweets in an automated way, we need to perform this activity manually. This could impact us creating a new bias and affecting our model results.

For our model, we selected two classical Machine Learning algorithms: logistic regression and the Naive Bayes algorithm. To make the analysis numerical, we used TfidfVectorizer and CountVectorizer to convert the text into vectors and process them numerically. This design intended to use as low resources as possible, even run it locally. We understand this can be a limitation as we are not able to perform deep learning, transformers, or apply the most advanced NLP techniques. As we understand who our potential clients can be, we agree that sometimes we need to sacrifice power or resources for simplicity.

To evaluate the performance of each of the pipelines, we used the metrics: recall, accuracy, F1_ score, AUC, and Brier through cross-validation. With the inclusion of these metrics, we evaluated our model, and we could observe that the one that gives us the best results is the logistic regression in combination with Convect, we have the best score for Brier, which gave us 0.104238. In the graphic, we can see that we have a good model for granular sentiment analysis.



During the exploratory data analysis, we found information about Twitter and the airlines, which despite not being used in the model, could serve to contextualize the analysis carried out. The analyzed tweets have an extension of fewer than 40 words predominate, the words written in the tweets are usually short, even the words used in the positive tweets are shorter than those used in the tweet's negatives, and the analyzed tweets use few punctuation marks. Moreover, three out of four tweets are negative, in general users are dissatisfied with these LCCs. However, the opinion of JetBlue users is worse compared to SouthwestAir users.

After this evaluation, we found that our product had more limitations and some biases that we did not identify until we ran the models. We were missing two key elements, the first one is sarcasm this is important in our metrics, our app can be cheated. The second one is the neutral opinions, as we are not including this category it is possible that we are missing important information from our users.

The Airflow orchestrator is our administrator who oversees managing all the activities so that our product flows automatically. It consists of two DAGs, the first one consists of five tasks that start when the orchestrator sends the signal in a programmed way so that the extraction of the information from our Twitter API is executed monthly. Once the first DAG finishes, the second one receives the signal to start, this DAG performs the execution of our model. The limitations we found in this system is that Airflow does not allow us to monitor the quality of the data, it only takes care of the execution of the tasks. So, at this point, we would have no way to detect errors, and the monitoring would have to be performed manually. Another important aspect to highlight is that Airflow does not have an intuitive interface, during our testing we realize that it is not so easy to do it in a consequent way we had to stop our instance and start again. This is important to us because if we do extractions constantly throughout the day, the orchestrator can get annoyed and may not respond. In addition to this, the access to the main repository cannot be local, and all this can be time-consuming to perform debugging activities.

6. Model Serving

In production, the model is executed after activating the alert that warns when the data set to be used for training/prediction is complete. After the database with new information is hosted in Big Query, the model is executed on the database called train.train_set _prediction. During execution, the four previously described pipelines are estimated and the five mentioned metrics are obtained from the model with the best performance and the score of each tweet is kept. Then we could make a batch inference because our API is for consultation and not for real-time decision making.

To further automate model development and execution, the features that would need to be added to the project are Vertex AI, Auto ML Text, and AutoML Translation. Vertex AI would allow for faster testing and model development by making it easier to prepare and store the dataset and to access Google's machine learning tools. The Auto ML Text tool would provide custom text and sentiment extraction, as well as support for large data volumes. In case of expanding the exercise on airlines with users of languages other than English, AutoML Translation would be used to perform text translation.

The manual component of the model is the labeling of the first data set used to train the model, restriction to the four previously mentioned pipelines, although the selection of the best model is not manual, our exercise is limited by these four proposed pipelines. These steps could be automated through the Google Cloud Natural Language API. In case the DAG fails, we will carry out the manual process, through the scripts that we have in the Github repository.

7. Reflection

The main idea of this project was to perform Sentiment Analysis to help marketing teams to know the user opinions about traveling and help with the analysis of consumer trends. After a team discussion, we agreed to refine the idea using Sentiment Analysis in two of the most popular airlines in the US. This analysis gives us a closer look at a real-world application, and the approach is not only for the marketing agencies but also usable for all kinds of users interested in making decisions related to low-cost airlines like JetBlue or SouthWest lines.

In terms of the design, we were able to understand the problem and provide good results. It is important to mention that Google offered us a good variety of technological tools that help us in all the stages of our product, from data extraction to results delivery. However, we think that our product has three weak spots that we can improve. The first one is that we could improve our tagging system. The tags were applied manually, and we could use some external tools to implement them in an automated way. The second one, as all our team members have different backgrounds, some tools were applied to a basic level and our model is a combination of classic Machine Learning algorithms, it was designed in this way to provide support in a cloud or the local system. We have the same opinion that our architecture could be more robust, performing retraining with an auto mix of pipelines, and could be implemented in real-time. The last one, we believe with more practice and research we will be able to build a better workflow with a better task administrator.

As a team, one of our weakest points was initial communication. Despite we had a strong beginning, after some sessions we had some ups and downs, and that affected the dynamic team. By the end of the course, we were able to improve our communication skills, align our strategy, and deliver a good product. The problem is that we did not have enough time to implement the Google predictor considered since the initial design of our product.

We have demonstrated that if the task is clearly defined, we can solve it with a simple approach with good results. But for future work, we think we could develop the context and perform an analysis of the entire phrase. We are contemplating that “Semantic Analysis” of the text is of great consideration instead of a granularity approach. We also considered the possibility of expanding the lexicon used with different concepts that refer not only to a positive or negative situation but also include a neutral approach and emotion detection (happy, sad, disappointed, grateful).

8. Broader Impacts

Sentiment analysis is a field under development. Although there are already numerous industries that use these types of techniques to build their Marketing campaigns, we found out that we can implement different tools and techniques to develop powerful systems. The results of our model show that this simple approach delivers pretty good results for all those users that need guidance to get information about a low-cost airline. Our model is adaptable to similar brands and products across the traveling industry.

However, we find some issues related to language problems. For example, if we use a different language like Spanish or Chinese, we will find a lack of resources. Also, cultural opinions or sarcasm could affect the results of our model.

Another harm is that once we have the notion of the sentiment. This is a granular approach, and the context depends on the researcher. If the user wants to scope phrases, we see a limitation, and all the results could be affected by this granularity. Additionally, sentiments are reflecting some emotions or feelings while emotions reflect attitude.

Twitter is a good source of information for sentiment analysis. This social network does a really good job capturing the people's feelings, or opinions about the selected airline users. But, in a more complex setting, Tweepy API provides some limitations to extracting data like data volume and data frame. We can use only public Tweets. We recommend that using Twitter or social networks should not be the only source of information that could be a start, but the data still needs a deeper analysis.

Teamwork

Component	**Responsable**
Documentation GitHub	Valeria
ELT Dag	Paulina
Sensor	Luis
Model	Enrique
Model Dag	Enrique
Dashboard	Paulina
Final report	Valeria, Luis
Presentation	Team

References

Amigo P. Adrián. (2020). Análisis de sentimiento en Twitter. Reputación de las principales aerolíneas europeas tras la crisis sanitaria del covid-19, Tesis Maestría, Economía, Finanzas y Computación. https://dspace.unia.es/handle/10334/5915

Anitsal, M.M., Anitsal, I., & Anitsal, S. (2017). A Sentiment Analysis of Air Passengers of Top Ten U.S. Based Airlines . (pp. 37-50). https://digitalcommons.kennesaw.edu/cgi/viewcontent.cgi?referer=&httpsredir=1&article=1163&context=ama_proceedings

Gaba, V., Verma, V. (2022). Sentiment Analysis of Twitter Data Using Machine Learning Approaches. In: Patel, K.K., Doctor, G., Patel, A., Lingras, P. (eds) Soft Computing and its Engineering Applications. icSoftComp 2021. Communications in Computer and Information Science, vol 1572. Springer, Cham. https://doi.org/10.1007/978-3-031-05767-0_12

Hegde B., Nagashree H S, Madhura Prakash(2018). Sentiment analysis of Twitter data: A machine learning approach to analyse demonetization tweets. Consulted in https://www.irjet.net/archives/V5/i6/IRJET-V5I6355.pdf

Kumar, S., Zymbler, M. (2019). A machine learning approach to analyze customer satisfaction from airline tweets. *J Big Data* **6,** 62. https://doi.org/10.1186/s40537-019-0224-1

Monkey Learn. 8 Applications of Sentiment Analysis. (s.f.) https://monkeylearn.com/blog/sentiment-analysis-applications/

Retamar, et al. (s.f.). Análisis de Sentimientos en Twitter: Una Implementación sobre Cloudera. https://ria.utn.edu.ar/bitstream/handle/20.500.12272/4704/An%C3%A1lisis_de_Sentimientos_en_Twitter_%28167%29.pdf?sequence=1&isAllowed=y

Scikit-learn: Machine Learning in Python, https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

https://scikit-learn.org/stable/modules/naive_bayes.html

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html