Manual Airline Sentiment Analysis

The primary goal of this notebook is to analyze the sentiment of all airlines for which data was present in the utilized dataset.

Dataset Information

The dataset used in this project is available on huggingface at this link. In total it contains 14,640 tweets directed at 6 major airlines.

For this project, the data was split into four datasets:

1. Production Dataset: Contains 200 observations from each airline, for a total of 1,200 observations
2. Training Dataset: Contains 70% of the remaining data (9408 observations), randomly selected from the dataset. This dataset is used to train the model.
3. Validation Dataset: Contains 20% of the remaining data (2688 observations) randomly selected from the 30% of data not used in production or training. This dataset is used to validate the model's performance on production data during the training process.
4. Test Dataset (Optional, but used): Contains the remaining 10% (1344 observations) of the non-production data. Used to estimate the model's performance on production data during the model testing procedure.

Data Processing Information

The dataset originally contained 15 predictors for each datapoint. For the purposes of this project, however, only 4 predictors were used:

- `tweet_id`: a numeric ID that uniquely identifies the tweet on Twitter
- `airline`: a string (could be 'American', 'United', 'Southwest', 'Virgin America', 'US Airways', or 'Delta') that corresponds to the airline being discussed in the tweet
- `airline_sentiment`: the ground truth sentiment used to train the classification model
- `text`: the text of the tweet

Note that this is for the training, validating, and testing procedures. The data used in production has the original ground-truth labels stripped, so as to simulate data that hasn't yet been labeled.

The data must be properly prepared for the Huggingface models. For sequence classification problems such as sentiment analysis, this largely revolves around tokenizing the text using the pretrained tokenizer for the specified model. Additionally, a class label must be created for the string labels, that maps the string values to integer values that the model can recognize. For the purposes of training, the `airline` and `tweet_id` columns were maintained in the datasets, but were not used in the training and prediction processes.

Training Information

The model used for prediction on the production data is a DistilBert Sequence Classification model. It begins from the pretrained Huggingface model hosted here. It was trained on the processed training dataset and validated on the processed validation dataset for 10 epochs, with 82-84% accuracy on the validation dataset during each epoch.

The model was tested on the processed testing dataset once, achieving an accuracy of 84.5%. The accuracy of the model in production should roughly equate to this value, though in the future manual validation of the model will be available for devs to verify that the model is performing with the expected accuracy.

The model deployed in production is also the model used in this notebook to manually analyze the sentiments and create sentiment scores for each airline.

Production Data Importing and Model Prediction

In order to analyze the sentiment of these six airlines, the production data must be imported and the predictions on the dataset must be made.

In [ ]:
```
# first we need to update the system path to include the
# app library
import sys
import os
import shutil

sys.path.append(os.path.abspath("../app"))
```

In [ ]:
```
# importing the production data collection and model prediction
# workflow from the application - this workflow is stored in
# the app_utils file within the application
from app_utils import collect_and_predict

# importing pandas to work with the dataframe resulting from the
# process_and_predict function call
import pandas as pd

# importing plotly.graph_objects to recreate the sentiment plots
# seen in the web application
import plotly.graph_objects as go
```

In [ ]:
```
# collect production data and make sentiment predictions
# using the trained model
preds = collect_and_predict()
preds
```

```
Map:   0%|          | 0/1200 [00:00<?, ? examples/s]
```

Out[ ]:

	idx	airline	text	sentiment
0	569205125278859265	Virgin America	@VirginAmerica thanks so much!	positive
1	568972394297077760	Virgin America	@VirginAmerica, you're doing a great job addin...	positive
2	570282469121007616	Virgin America	@VirginAmerica SFO-PDX schedule is still MIA.	negative
3	569618527948115968	Virgin America	@VirginAmerica You'd think paying an extra $10...	negative
4	569211675108179968	Virgin America	@VirginAmerica Site down? #help	negative
...	...	...	...	...
1195	570241221303468032	American	@AmericanAir my boss is :)	neutral
1196	569604083507556353	American	@AmericanAir is this how you let your employee...	negative
1197	570085859132878848	American	@AmericanAir So break whatever you want, take ...	negative
1198	569997196298293249	American	@AmericanAir #epicfail on connections in #Chic...	negative
1199	570306867878072320	American	@AmericanAir aa employees were rude and unwill...	negative

1200 rows × 4 columns

In [ ]:
```
# generate the sentiment score as follows: negative tag is -1,
# neutral tag is 0, positive tag is +1
preds["sentiment_score"] = pd.Series([-1 if sent=="negative" else 0 if sent=="neutral" else 1 for sent in preds["sentiment"]])
preds
```

Out[ ]:

	idx	airline	text	sentiment	sentiment_score
0	569205125278859265	Virgin America	@VirginAmerica thanks so much!	positive	1
1	568972394297077760	Virgin America	@VirginAmerica, you're doing a great job addin...	positive	1
2	570282469121007616	Virgin America	@VirginAmerica SFO-PDX schedule is still MIA.	negative	-1
3	569618527948115968	Virgin America	@VirginAmerica You'd think paying an extra $10...	negative	-1
4	569211675108179968	Virgin America	@VirginAmerica Site down? #help	negative	-1
...	...	...	...	...	...
1195	570241221303468032	American	@AmericanAir my boss is :)	neutral	0
1196	569604083507556353	American	@AmericanAir is this how you let your employee...	negative	-1
1197	570085859132878848	American	@AmericanAir So break whatever you want, take ...	negative	-1
1198	569997196298293249	American	@AmericanAir #epicfail on connections in #Chic...	negative	-1
1199	570306867878072320	American	@AmericanAir aa employees were rude and unwill...	negative	-1

1200 rows × 5 columns

We can also collect the list of airlines from the data processing script as well...

In [ ]:
```
with open("../data/airlines.txt", "r") as f:
    airlines = [line.strip() for line in f.readlines()]

airlines
```

Out[ ]:
```
['Virgin America', 'United', 'Southwest', 'Delta', 'US Airways', 'American']
```

Sentiment Analysis on Production Data

We now have our predicted sentiments from the model, and can do a bit of analysis...

We can count the number of times each type of sentiment appears for each airline, and with this information can do the following:

- create pie charts that allow for visualization of each airline's sentiment distribution
- quantify the sentiment of each airline by assigning values to different types of sentiments and computing the average value
- discover which airline has the most positive sentiment on a tweet-by-tweet basis, which has the most negative sentiment on a tweet-by-tweet basis, and which has the most neutral sentiment on a tweet-by-tweet basis

All of this is done in the next two cells.

In [ ]:
```
# create an output directory for the sentiment
# distribution plots

if not os.path.exists("./plots"):
    os.mkdir("./plots")
else:
    shutil.rmtree("./plots")
    os.mkdir("./plots")
```

In [ ]:
```
# airline sentiment score dictionary initialization
airline_sentiment_scores = {}

# airline sentiment value counts dictionary initialization
airline_sentiment_counts_dict = {}

for airline in airlines:
    # query the dataframe using the airline name
    airline_df = preds.query(
        "airline==@airline"
    )

    # compute the overall airline sentiment score
    # by averaging the sentiment scores of each tweet
    # (should yield a value between -1 and 1, with -1 being
    # completely negative, 0 being net neutral, and 1 being
    # completely positive)
    airline_sentiment_score = airline_df["sentiment_score"].aggregate("mean")
    airline_sentiment_scores.update({
        airline: airline_sentiment_score
    })

    # create a dictionary that contains each airline name as the keys
    # and the counts of each sentiment value as the values in the
    # dictionary
    airline_sentiment_counts_dict.update({
        airline: airline_df["sentiment"].value_counts().to_dict()
    })

    # create a new dataframe that includes the counts
    # of each sentiment
    airline_sentiment_counts = pd.DataFrame(
        airline_df["sentiment"].value_counts()
    ).reset_index().sort_values(by="sentiment")

    # plot the counts of the sentiment and save them
    # to plots corresponding to the airline name
    fig = go.Figure(
        data=[
            go.Pie(
                labels=airline_sentiment_counts["sentiment"],
                values=airline_sentiment_counts["count"],
                sort=False,
                marker=dict(colors=["red", "blue", "green"]),
                title=f"Sentiment Plot For Airline {airline}"
            )
        ]
    )

    fig.write_image(f"./plots/{airline.lower().replace(' ', '_')}_sentiment_plot.png", format="png")

# create a dataframe from the average sentiment scores and sort
# the results in descending order
airline_sentiment_scores_frame = pd.DataFrame.from_dict({
    'airline': list(airline_sentiment_scores.keys()),
    'score': list(airline_sentiment_scores.values()),
}).sort_values(by="score", ascending=False).reset_index().drop(columns="index")

# use the airline sentiment counts dictionary to find the most negative, most neutral,
# and most positively viewed airlines, in terms of sentiment count

# trackers
most_negative = ''
most_negative_count = 0
most_neutral = ''
most_neutral_count = 0
most_positive = ''
most_positive_count=0

# iterate through the dictionary
for airline, counts_dict in airline_sentiment_counts_dict.items():

    airline_negative_count = counts_dict.get('negative')
    airline_neutral_count = counts_dict.get('neutral')
    airline_positive_count = counts_dict.get('positive')

    if airline_negative_count > most_negative_count:
        most_negative = airline
        most_negative_count = airline_negative_count
    if airline_neutral_count > most_neutral_count:
        most_neutral = airline
        most_neutral_count = airline_neutral_count
    if airline_positive_count > most_positive_count:
        most_positive = airline
        most_positive_count = airline_positive_count

airline_superlatives_summary = {
    'negative': {
        'airline': most_negative,
        'count': most_negative_count
    },
    'neutral': {
        'airline': most_neutral,
        'count': most_neutral_count
    },
    'positive': {
        'airline': most_positive,
        'count': most_positive_count
    }
}

print(airline_superlatives_summary)
airline_sentiment_scores_frame
```

```
{'negative': {'airline': 'US Airways', 'count': 156), 'neutral': {'airline': 'Virgin America', 'count': 69), 'positive': {'airline': 'Virgin America', 'count': 58)}
```

Out[ ]:

	airline	score
0	Virgin America	-0.075
1	Delta	-0.155
2	Southwest	-0.210
3	American	-0.600
4	United	-0.625
5	US Airways	-0.710

All of the plots for the sentiment distributions are available in the `plots` folder. The rest of the information is above.

Conclusions

Analysis of the production data shows that Virgin America has the most positive sentiment on Twitter. This is clearly visible through all three methods of analysis:

1. When using a pie chart to visualize the airline sentiment, Virgin America has the largest percentage of positive sentiment tweets, as well as the lowest percentage of negative sentiment tweets.
2. When using the overall sentiment scores, Virgin America's score was twice higher the second most positively viewed airline's score
3. When counting sentiment reviews, Virgin America had both the most neutral tweets and the most positive tweets.

Interestingly enough, the sentiment scores seem to roughly correlate with the performance of the stocks of the airlines in question. Below, we calculate the difference between the current stock price (as of the day of writing this, 09/13/2023) and the stock price in February 2014, when the tweets were written (tweets were scraped from various days and the only freely avilable stock data is monthly data from 2015).

1. Though the The US Airways twitter account was still active, they had been acquired by American Airlines in 2013. The American Airlines stock change between 02/24/2015 and present is used for US Airways.
2. Virgin America was still a publicly traded stock at the time of the tweets, but were acquired by Alaska Air in 2016. To simplify, the difference was calculated using the Virgin America stock price at the time of the tweets, and the current stock price of Alaska Air.

In [ ]:
```
airline_sentiment_scores_frame["stock_price_diff_since022015"] = pd.Series([10.30, -4.58, -12.35, -34.6, -13.75, -34.6])
airline_sentiment_scores_frame
```

Out[ ]:

	airline	score	stock_price_diff_since022015
0	Virgin America	-0.075	10.30
1	Delta	-0.155	-4.58
2	Southwest	-0.210	-12.35
3	American	-0.600	-34.60
4	United	-0.625	-13.75
5	US Airways	-0.710	-34.60

In [ ]:
```
airline_sentiment_scores_frame.corr(method='pearson', numeric_only=True)
```

Out[ ]:

	score	stock_price_diff_since022015
score	1.000000	0.862415
stock_price_diff_since022015	0.862415	1.000000

The correlation score between the calculated sentiment score and the stock performance over the last ~8 years is a staggering 0.862! This is a clear indication that the sentiment of airlines on twitter actually reflects how the stock performs in the future.