1). Investigate the missing data in this dataset.

Specifically, for each of the following variables that have missing data, decide if any imputation is possible. Give your reasoning and code if you decide to impute missing values. Columns for investigation: CARRIER, CARRIER_NAME, MANUFACTURE_YEAR, NUMBER_OF_SEATS, CAPACITY_IN_POUNDS, and AIRLINE_ID. For example, watch out for "North American Airlines" aircrafts. Are the CARRIER/UNIQUE_CARRIER column really missing?

First, I wanted to understand the dataset's size and columns and where the missing data is to get a better sense of the problem. Next, I looked at CARRIER and UNIQUE_CARRIER for North American Airlines, noticing that the “NA” values were being interpreted by Python as NaN. This made it seem as though that data was missing even though it was not, so I simply inserted the string “NA” into all values for CARRIER and UNIQUE_CARRIER under North American Airlines.

Second, I focused on missing values within the columns CARRIER_NAME, AIRLINE ID, UNIQUE_CARRIER_NAMES and UNIQUE_CARRIER. I noticed that all of the listed columns had 105 missing values, which indicated to me that they may be the same 105 rows with missing values. Indeed, this was the case. Among the 105 rows with these missing values, I found there to be only two CARRIER types: L4 and OH. My initial reaction was that the values from previous rows with full data for these carriers could be used to impute the missing values. Upon further investigation, this assumption proved to be correct for L4 carrier because for all other data in this subset of there dataset, there was only one value inputted for CARRIER_NAME, AIRLINE ID, UNIQUE_CARRIER_NAMES and UNIQUE_CARRIER, respectively. However, the OH carrier had two possible values for each of the columns under inspection. Therefore, simply filling the missing data with information from previous rows with the OH carrier would not be completely accurate. To determine a way to differentiate between the two values used for each of the columns under inspection, I looked at the model names. After checking that all models used by the OH carrier used a unique combination of CARRIER_NAME, AIRLINE ID, UNIQUE_CARRIER_NAMES and UNIQUE_CARRIER values, I determined that by grouping the missing data by model type, I would be able to accurately impute the data by copying data used in previous rows of the OH carrier.

Third, I investigated NUM_SEATS. I found that all of the missing number of seat values come from the M6 carrier. Because all M6 flights have 0 seats, I determined that simple imputation could be done to fill in the missing values.

Fourth, I looked at MANUFACTURE_YEAR. Because there are only 3 manufacturing years missing, and there is little correlation between the manufacturing year and the other columns' values, simply removing these three rows rather than "guessing" with imputation should not impact the analysis in a negative way. This is especially true because 3 rows out of the original 132313 is only 0.002% of the data. Therefore, I removed these 3 rows instead of pursuing imputation.

Last, I honed in on CAPACITY_IN_POUNDS. First, I plotted a histogram of the column's data across the whole dataset to understand the distribution. The histogram showed that the capacity in pounds is heavily right skewed, so using metrics such as mean to impute the missing values would likely not produce accurate results. One method that could work is nearest neighbors because this method will compare rows that are similar (similar aircraft models, number of seats, etc.) to estimate the missing capacity. Before following through with the KNN algorithm, I wanted to confirm that there was differentiation in capacities among the other characteristics in the dataset. I decided to look into the model types. After plotting a histogram of the distribution of capacities for each model with missing data, I found that model-type was a good indicator of capacity. Each model had a unique range of values of its capacity. Additionally, models B767-300 and A-320-PSGRneo had constant capacities, so I filled them with simple imputation before KNN was performed. After performing KNN on the remaining missing capacities, I plotted the imputed data against the original values for the whole dataset, but it was difficult to see how accurate the imputation was. I then broke it down by model type to better visualize how well the imputation did. I plotted the imputed against the original values for each model and found that KNN performed relatively well. The capacities that it imputed were all, for the most part, within a reasonable range of possible capacities for the given models.

2). Inspect the columns MANUFACTURER, MODEL, AIRCRAFT_STATUS, and OPERATING_STATUS.

Decide, for each column, if transformation or standardization of data are required. Give your reasoning and code if you decide to transform the data.
Hints: For very messy data like manufacturer/model names, give your best attempt. It is okay to not catch them all. Use value_counts() to identify "big wins". Break down into multiple steps, instead of having one line of code to do them all.

First, I looked into OPERATING_STATUS and found that it had found possible values: 'Y', 'N', 'y', and ' '. On the BTS dataset attribute summary website, operating status under the "Analysis" tab is shown to mean Y= yes, N=no, and y=unknown. However, I chose to merge y and Y because since it is indicated with a y, it may be more likely for the status to be Y. I understand that this is an assumption, and in real-world situations, I would want to confirm that this course of action makes sense. This standardization, under my assumption, should ensure that all aircrafts that are operating have the same value, which could be useful in further analysis.

Second, I investigated AIRCRAFT_STATUS. I found that the possible values were 'A', 'a', 'B', 'b', 'O', 'o', and 'L'. On the BTS dataset attribute summary website, the values in aircraft status are O, a, and b. Because 'L' is not included in the dataset attribute summary, I am unsure of what it stands for and how it could be properly standardized, so I left it as is. Therefore, I standardized the data such that o was set to O, A was set to a, and B was set to b. In this way, I am assuming that O & o, A & a, and B & b have the same meaning. This action will ensure that the data matches the description on the website and analysis correctly considers this case sensitive data.

Third, I focused on MANUFACTURER. It was immediately obvious that the same companies were being listed multiple times under variations of the same company name (ex: BOEING and Boeing). These must be standardized so that companies are accurately represented and considered as one entity rather than separately under the variations of names. Next, I transformed the data to be all lowercase to reduce some of the variations and then looked company by company searching for variations that occurred in the data and standardized it to only one name. After this standardization, many of the replicated company names correctly appear under only 1 version of the name. This reduced the number of unique manufacturers in the dataset by 62 (183-121).

Lastly, I looked at MODEL. It was clear that there are models that are the same but have been entered into the data under different names. This could confuse analysis by considering these models to be separate from each other when in reality they have the same meaning. Therefore, the model names must be standardized if they are representing the same model. First, I capitalized all letters to eliminate case differentiations between model names. Next, I manually looked through model names containing similar characters and mapped names appearing to represent the same model to a uniform name. I mapped, for the most part, passenger planes under slightly different names to be the same and performed the same standardization for non-passenger planes, respectively (ex: 'B737-300PAX' to 'B737-300-PSGR' and '737-300' to 'B737-300'). The number of models included in the data was reduced by 75 (1340-1265) after standardizing model names.

3). Remove data rows that still have missing values. Report the amount of remaining data you obtained.

The cleaned data has 101,400 rows.

4). Transformation and derivative variables

For the columns NUMBER_OF_SEATS and CAPACITY_IN_POUNDS, check the skewness in the variable and plot a histogram for each variable.
The Box-Cox transformation (scipy.stats.boxcox) is one possible way to transform variables into a "more-normal-like" variable. Apply the Box-Cox transformation for these two columns and save them as new columns, i.e. XXXXXXXXX_BOXCOX.
Plot a histogram for each transformed variable.
Describe what you observe before and after transformation.

I calculated the numerical value representing the skewness and plotted the histograms of the initial data. Both datasets are initially right skewed. The quantified value of the skewness of the number of seats is lower than that of capacity in pounds. The same trend is seen in the histograms where capacity in pounds visually has a much stronger right skew than number of seats.

Post Box-Cox transformation, the quantified skewness is more left skewed for the number of seats and capacity in pounds is still right skewed but much less severe than prior to the transformation. The histograms of the transformed data show the same trends. However, the transformed data has a significantly smaller range of x-values for both columns of data.

5). Feature engineering

Create a new column SIZE by the quartiles of NUMBER_OF_SEATS

below 25% percentile: SMALL
25% - 50% percentile: MEDIUM
50% - 75% percentile: LARGE
above 75% percentile: XLARGE
For each size group, provide and plot the proportions of aircrafts that are operating versus not (OPERATING_STATUS). For each size group, provide and plot the proportions of aircrafts belonging to each aircraft status group (AIRCRAFT_STATUS). Provide a written summary of your findings.

I first defined the quantiles of the number of seats and then defined a function to categorize the data into sizes.

Across all sizes, the proportion of planes operating remains relatively static (around 94 to 97%). This indicates that the majority of planes of all size types are operating. XLARGE planes are the most likely to be operating (97.3%) and MEDIUM planes are the least likely to be operating (94.6%).

The aircrafts that are small, large, and xlarge have majority "O" status, which means that they are owned, according to the online summary of the data. Small airplanes have the highest proportion of "O" status at 72.7%. A very low proportion, less than 0.2%, of aircrafts across all plane sizes have "L" status. This is interesting because “L” status is not described in the dataset summary and has very low representation, so it should be investigated further to ensure that it is providing some sort of information and otherwise it should be standardized/removed. Medium planes have majority "b" status (51.4%). After "L", "a" is the second least common status among all plane sizes (at most 12.3% for large aircrafts).