Course: DATS 6401 – Visualization of Complex Data

Instructor: Dr. Reza Jafari

Final Term Project

Topic: Airline passenger satisfaction analysis

Name: Pon swarnalaya Ravichandran

RP	05/10/2023
Initials	Date

TABLE OF CONTENTS

ABSTRACT

The proposed project, focuses on developing a web-based application using python dash package. The analysis involves the usage of airline passenger satisfaction dataset which was received in form of feedback. It certainly consists of 103594 observations(datapoints) in it. The application has numerous tabs which helps the user navigate through the different aspects of the analysis. The application circles mainly around the ‘satisfaction’ variable since it is the target variable.

INTRODUCTION

The established dash app is an interactive point-&-click interface to models written in python, vastly expanding its environment to 8 tabs including the summary.

The web-app layout has been designed in such way which is more easily accessible by the user. The user can directly click on the viewable tab which they desire to see. The dataset has been analyzed for missing values. Perhaps, the dataset was found to have 310 missing values overall, hence it was removed and the cleaned dataset is displayed in the 'know your data' tab. Additionally, the user can download the cleaned dataset from the 'download.csv' button.

The cleaned dataset is tested for normality and PCA is done to reduce the dimensions or features of the original space. PCA analysis helps in finding the best number of feature components for the modeling as well as finding the correlation between the variables. The heatmap and scatter matrix is shown in the next tab which helps in finding and understanding the collinearity between the variables.

The plots like line, count, histogram, box, violin, pie, scatter plot with the regression line and kernel density function are developed to understand and work on the dataset in a wider view.

The dashboard is established through the google cloud platform for the worldwide view(https://dashapp-7xpodtpsca-nn.a.run.app/)
A detailed description about these techniques is discussed in the next chapter.

METHOD, THEORY AND PROCEDURES

Dash:

Without the need for Javascript, Dash is an open-source framework for creating analytical apps that is strongly connected with the Plotly graphing toolkit. Dash is a Python framework that is mostly used to create apps for data visualization.

Dash Tabs:

The dcc.Tabs and dcc.Tab components can be used to create tabbed sections in your app. A collection of dcc.Tab components is held by the dcc.Tabs component, which also controls the style and value of each individual tab.

Dash Callbacks:

functions that are automatically called by Dash whenever an input component's property changes, to update some property in another component (the output).

Dash Core Components:

You may import and utilize the Dash Core Components module (dash.dcc) to access a variety of interactive components, such as dropdowns, checklists, and sliders.

Dash HTML Components:

Dash is a web application framework that provides pure Python abstraction around HTML, CSS, and JavaScript. Instead of writing HTML or using an HTML templating engine, you compose your layout using Python with the Dash HTML Components module (dash.html).

Line Plot:

The distribution of a continuous variable is frequently shown using a line plot, sometimes referred to as a dot plot. It is made up of a set of data points linked by straight lines and plotted on a number line. The resulting visual representation of the data may be used to spot trends, patterns, and outliers.

Histogram:

A histogram is a diagram that shows how a continuous variable is distributed. It is made up of a string of contiguous rectangles, each of which has an area that corresponds to the frequency of observations that fall within a certain interval or bin. The form, center, and spread of the data distribution may be determined using the visual representation that is produced.

Pie Chart:

Pie charts can be used to show percentages of a whole and represents percentages at a set point in time. Unlike bar graphs and line graphs, pie charts do not show changes over time. It is more suitable for categorical variable because of how the variables are defined.

Dropdown:

To create a basic dropdown, provide options and a value to dcc.Dropdown in that order.

Graph:

The dcc.Graph component can be used to render any plotly-powered data visualization, passed as the figure argument.

Input:

Number type is now close to native HTML5 input behavior across browsers. We also apply a strict number casting in callbacks: valid number converts into corresponding number types, and invalid number converts into None.

Correlation Matrix:

A correlation matrix is a table showing correlation coefficients between sets of variables. Each cell in the matrix represents the correlation coefficient between two variables, which ranges from -1 (perfect negative correlation) to 1 (perfect positive correlation), with 0 indicating no correlation. Correlation matrices are useful for identifying patterns and relationships between variables, and can help inform decisions about data analysis and modeling.

SVD:

Singular Value Decomposition (SVD) is a matrix factorization technique that decomposes a matrix into three constituent matrices: U, Σ, and V*. In SVD, a rectangular matrix A (m x n) is decomposed into a product of three matrices: U (m x m) which is an orthogonal matrix, Σ (m x n) which is a diagonal matrix with non-negative real values (known as singular values), and V* (n x n) which is the conjugate transpose of another orthogonal matrix.

Principal Component Analysis:

Reducing the number of input variables for a predictive model is referred to as dimensionality reduction. Fewer input variables can result in a simpler predictive model that may have better performance when making predictions on new data. Perhaps the most popular technique for dimensionality reduction in machine learning is Principal Component Analysis, or PCA for short. This is a technique that comes from the field of linear algebra and can be used as a data preparation technique to create a projection of a dataset prior to fitting a model.

Outlier Detection and Removal:

Outliers are data points that are far from other data points. With outlier detection and treatment, anomalous observations are viewed as part of different populations to ensure stable findings for the population of interest.

Outlier Detection- Interquartile Range (IQR) - A commonly used rule says that a data point is an outlier if it is more than 1.5*IQR above the third quartile or below the first quartile. IQR is calculated as : Q3-Q1.

Low outliers are below Q1 − 1.5 * IQR and High outliers are above Q3 + 1.5 * IQR where Q1 is the first quartile and Q3 is the third quartile.

Kolmogorov-Smirnov (K-S) Test:

The Kolmogorov-Smirnov (K-S) Test compares your data with a known distribution and lets you know if they have the same distribution. The K-S test is non-parametric test. It is commonly used as a test for normality to see if your data is normally distributed.

H0: The data are Normally distributed. p-value > alpha

H1: The data are not Normally distributed. p-value < alpha

Shapiro-Wilk Test:

The Shapiro-Wilk test is a way to tell if a random sample comes from normal distribution. In practice, the Shapiro-Wilk test is believed to be a reliable test of normality, although there is some suggestion that the test may be suitable for smaller samples of data.

H0: The data are Normally distributed. p-value > alpha

H1: The data are not Normally distributed. p-value < alpha

D’Agostino’s K2 test:

D’Agostino’s K2 test is a goodness-of-fit measure of departure from normality, that is the test aims to establish whether or not the given sample comes from a normally distributed population.

H0: The data are Normally distributed. p-value > alpha

H1: The data are not Normally distributed. p-value < alpha

Procedure:

1. Load the dataset and necessary libraries
2. Setup the dash app layout with necessary tabs
3. Initialize the layout for tab with necessary dash core components and html components.
4. Setup callback functions to make the app interactive with user's choice of operations
5. Repeat step 3 and 4 for all the tabs
6. Run the app locally to make sure everything is aligned
7. Create an account in google cloud console and add the python file and docker file in the editor
8. Follow the steps provided in the following article to deploy and publish the app in the internet. https://medium.com/kunder/deploying-dash-to-cloud-run-5-minutes-c026eeea46d4

EXPERIMENTAL SETUP

Required libraries:

The following python libraries are required to run the code file seamlessly.

```
import dash
from dash import dash_table
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly.express as px
import seaborn as sns
from dash import dcc
from dash import html
from dash.dependencies import Input, Output, State
from numpy import linalg as la
from scipy.stats import kstest
from scipy.stats import normaltest
from scipy.stats import shapiro
from sklearn.decomposition import PCA
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from motionheading import app
import base64
import datetime
import io
```

DESCRIPTION OF THE DATASET:

The data was collected by surveying passengers who had recently traveled on a major airline, and it includes information about their demographics, flight information, and overall satisfaction with their experience. The dataset contains 129880 rows and 24 columns.

Attribute Information:

Gender: Gender of the passengers (Female, Male)
Customer Type: The customer type (Loyal customer, disloyal customer)
Age: The actual age of the passengers
Type of Travel: Purpose of the flight of the passengers (Personal Travel, Business Travel)
Class: Travel class in the plane of the passengers (Business, Eco, Eco Plus)
Flight distance: The flight distance of this journey
Inflight wifi service: Satisfaction level of the inflight wifi service (0:Not Applicable;1-5)
Departure/Arrival time convenient: Satisfaction level of Departure/Arrival time convenient
Ease of Online booking: Satisfaction level of online booking
Gate location: Satisfaction level of Gate location
Food and drink: Satisfaction level of Food and drink
Online boarding: Satisfaction level of online boarding
Seat comfort: Satisfaction level of Seat comfort
Inflight entertainment: Satisfaction level of inflight entertainment
On-board service: Satisfaction level of On-board service
Leg room service: Satisfaction level of Leg room service
Baggage handling: Satisfaction level of baggage handling
Check-in service: Satisfaction level of Check-in service
Inflight service: Satisfaction level of inflight service
Cleanliness: Satisfaction level of Cleanliness

Departure Delay in Minutes: Minutes delayed when departure
Arrival Delay in Minutes: Minutes delayed when Arrival
Satisfaction: Airline satisfaction level(Satisfaction, neutral or dissatisfaction)

Data source: https://www.kaggle.com/datasets/teejmahal20/airline-passenger-satisfaction

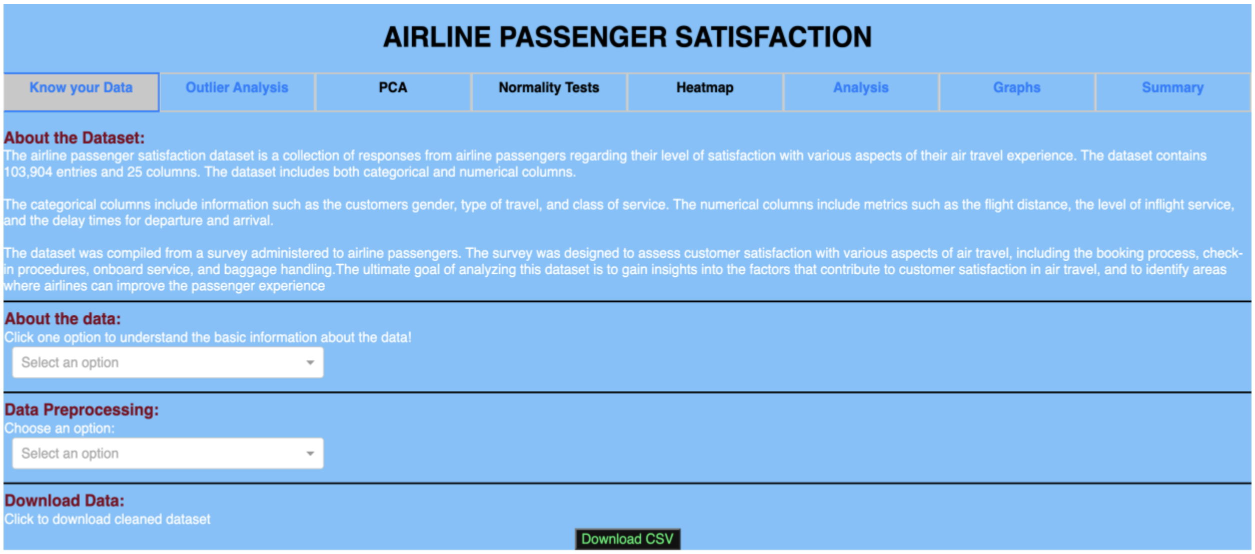
Looking at the dataset, it is clear that the data was collected based on the comments made by an airline's passengers. A person who travels in a different class would thus often have different thoughts about the airline and its comfort. Their input has been recorded as a point between the range of 0-5, in order to accommodate the viewpoints.

There are 1 lakh observations in the dataset, and 24 variables. 19 numerical variables and 5 category variables must be worked with. All other properties are specified as dependent variables, whereas the attribute "satisfaction" is regarded as an independent variable.

The analysis of the dataset assists in drawing conclusions about the degree of satisfaction of passengers that will be released in the future based on a variety of factors and also aids in comprehending the fundamental needs of passengers.

APPLICATION LAYOUT:

The web-app is designed in a way that the user can access from the vaery basic information of the dataset to the high level of analysis under one roof. The app is designed to be easy understanding and user friendly. It gives a clear understanding about what are present under each tab with its name seen on the tabs. Prominently, it has multiple tabs and multiple divisions with a download button.



AIRLINE PASSENGER SATISFACTION

About the Dataset:
The airline passenger satisfaction dataset is a collection of responses from airline passengers regarding their level of satisfaction with various aspects of their air travel experience. The dataset contains 103,904 entries and 25 columns. The dataset includes both categorical and numerical columns.

The categorical columns include information such as the customers gender, type of travel, and class of service. The numerical columns include metrics such as the flight distance, the level of inflight service, and the delay times for departure and arrival.

The dataset was compiled from a survey administered to airline passengers. The survey was designed to assess customer satisfaction with various aspects of air travel, including the booking process, check-in procedures, onboard service, and baggage handling.The ultimate goal of analyzing this dataset is to gain insights into the factors that contribute to customer satisfaction in air travel, and to identify areas where airlines can improve the passenger experience

About the data:
Click one option to understand the basic information about the data!

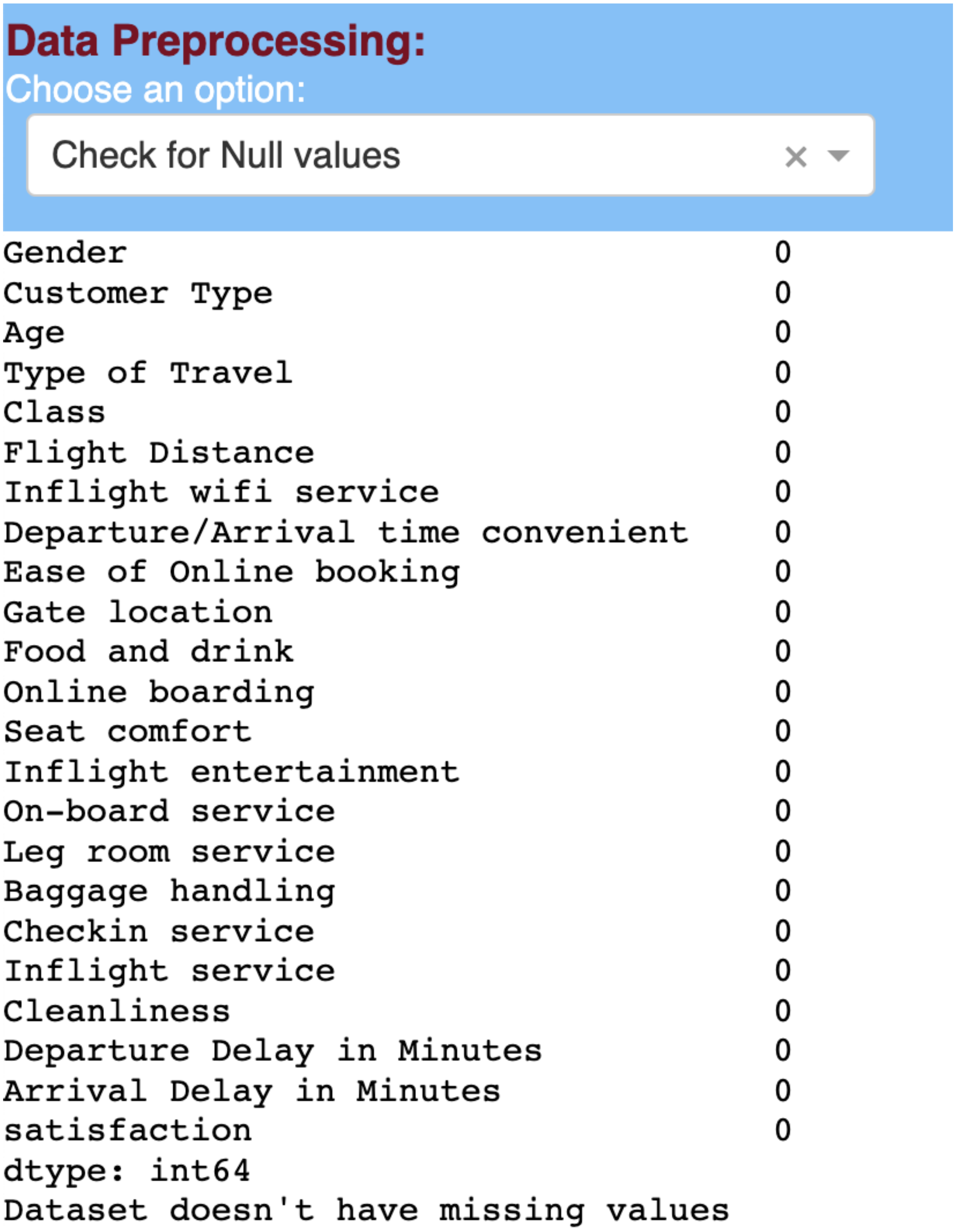
Data Preprocessing:
Choose an option:

Download Data:
Click to download cleaned dataset

Fig 1.1 *Application layout*

PREPROCESSING THE DATASET:

The dataset has been analyzed for missing values. It happened to have 310 missing values in the ‘Arrival Delay in Minutes’ column. It was cleaned and made available for the further analysis.



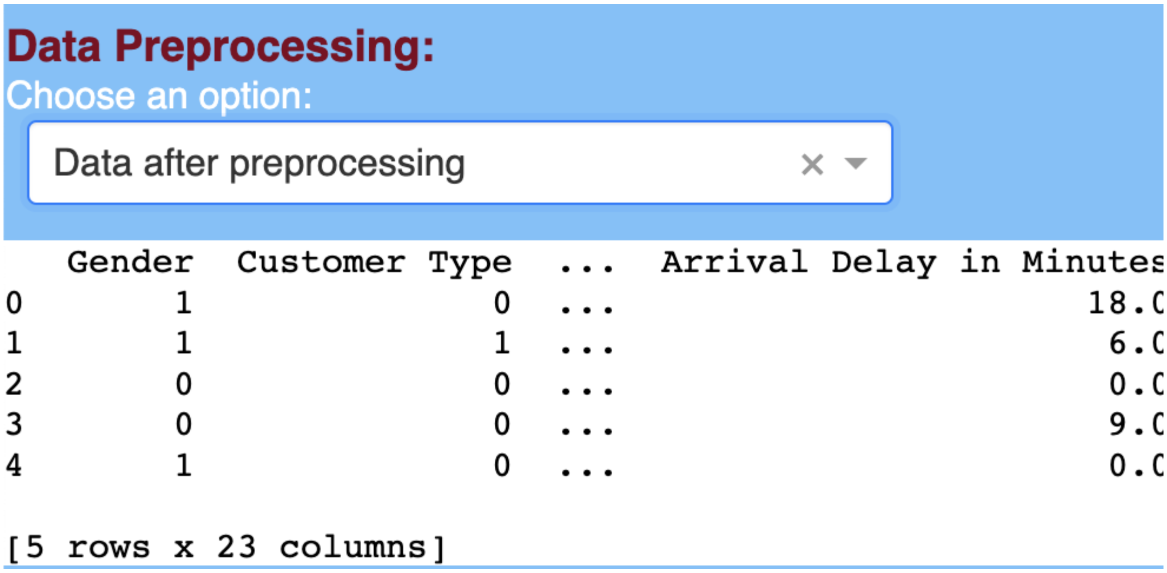
Data Preprocessing:

Choose an option:

Check for Null values

```
Gender                                 0
Customer Type                          0
Age                                    0
Type of Travel                         0
Class                                  0
Flight Distance                        0
Inflight wifi service                  0
Departure/Arrival time convenient      0
Ease of Online booking                 0
Gate location                          0
Food and drink                         0
Online boarding                        0
Seat comfort                           0
Inflight entertainment                 0
On-board service                       0
Leg room service                       0
Baggage handling                       0
Checkin service                        0
Inflight service                       0
Cleanliness                            0
Departure Delay in Minutes             0
Arrival Delay in Minutes               0
satisfaction                           0
dtype: int64
Dataset doesn't have missing values
```

Fig 1.2 ***Data preprocessing***



Data Preprocessing:

Choose an option:

Data after preprocessing

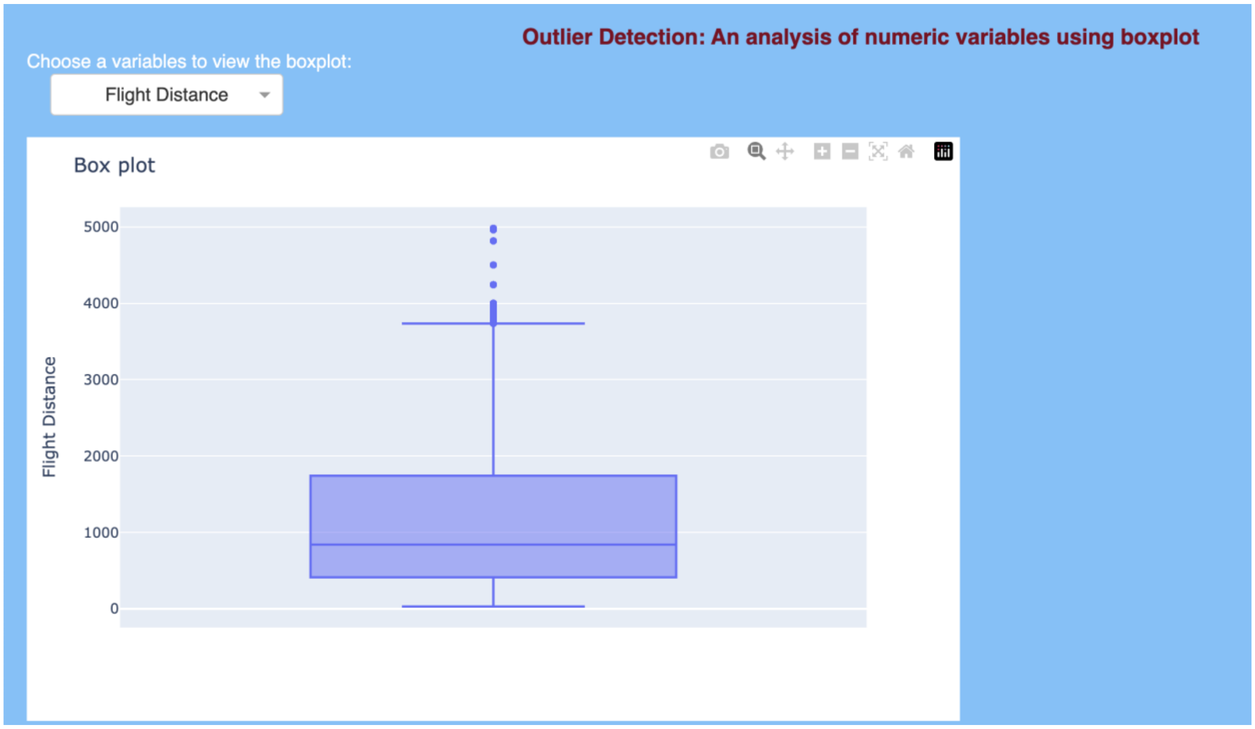
```
   Gender  Customer Type  ...  Arrival Delay in Minutes
0       1              0  ...                      18.0
1       1              1  ...                       6.0
2       0              0  ...                       0.0
3       0              0  ...                       9.0
4       1              0  ...                       0.0

[5 rows x 23 columns]
```

Fig 1.3 ***First five columns of the dataset after preprocessing***

The cleaned dataset can be downloaded from the user end by clicking on the download button available on the end division of 'know your data' tab.

OUTLIER DETECTION & REMOVAL:

The second tab in the application helps in analyzing the outliers of the dataset. Outliers are the vital problem for many datasets and for many statistical analyses. Outliers are data points that significantly deviate from the rest of the data in a dataset. They can be caused by measurement errors, data entry errors, or real-world phenomena. The problems caused by outliers in data analysis and modelling such as skewing of statistical measures, biased estimates, overfitting and misinterpretation of the results. Typically outliers deviate the result in the most terrific way.

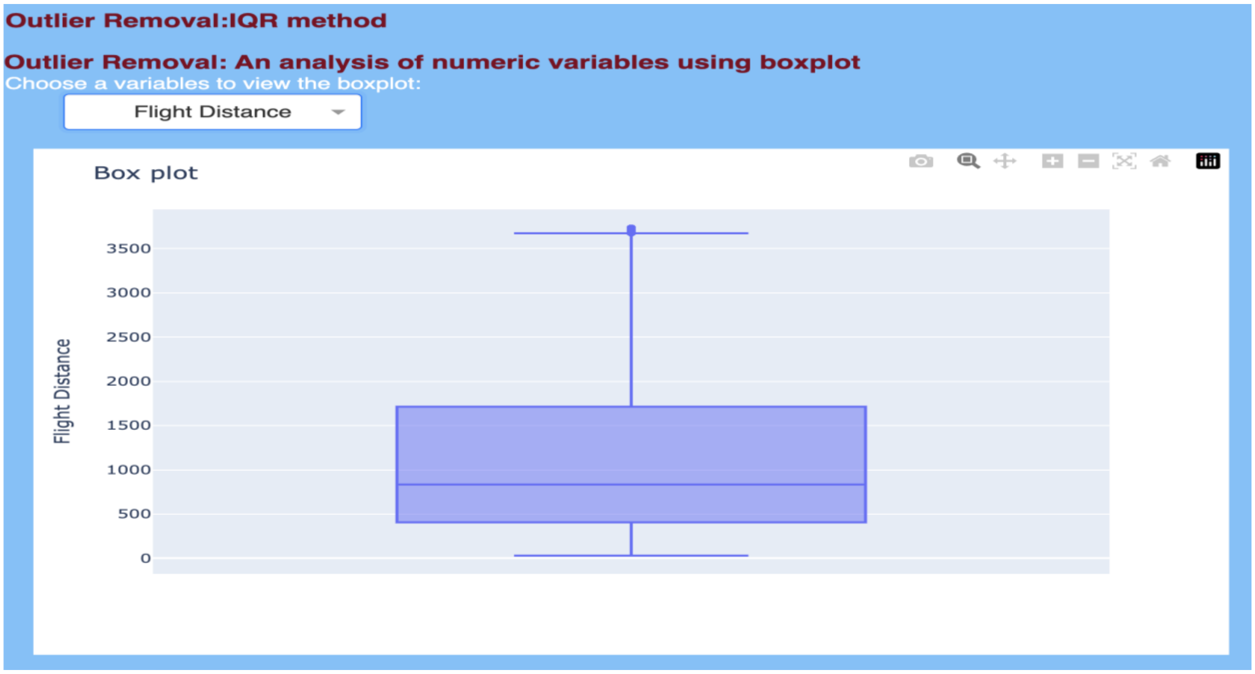


Outlier Detection: An analysis of numeric variables using boxplot

Choose a variables to view the boxplot:

Flight Distance

Fig 1.4 *Variables with outliers*



Outlier Removal:IQR method

Outlier Removal: An analysis of numeric variables using boxplot

Choose a variables to view the boxplot:

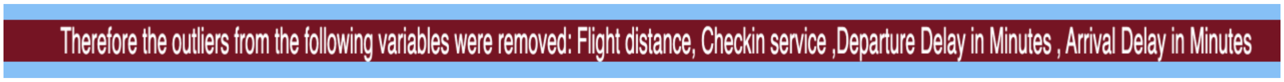
Flight Distance

Fig 1.5 *Variables without outliers*

Outliers detection tab allows the user to visualize boxplots for all the numerical variables which is embedded in the dropdown menu. From the dropdown menu the user can select the variables of which they want to check the presence of the outlier. The Fig 1.4 shows the variable ‘flight distance’ with the outlier whereas the fig 1.5 shows the same variable after the outlier is removed where there is no datapoints after the whiskers.

Boxplots are specifically used for finding the outliers Because they offer a visual representation of a dataset's distribution, including its central tendency, variability, and skewness, box plots are especially used to identify outliers. A box plot shows the median as a line inside the box, which represents the middle 50% of the data. The whiskers reflect the lowest and highest values that fall within a certain range, which is commonly 1.5 times the IQR between the upper and lower quartiles of the data**.**

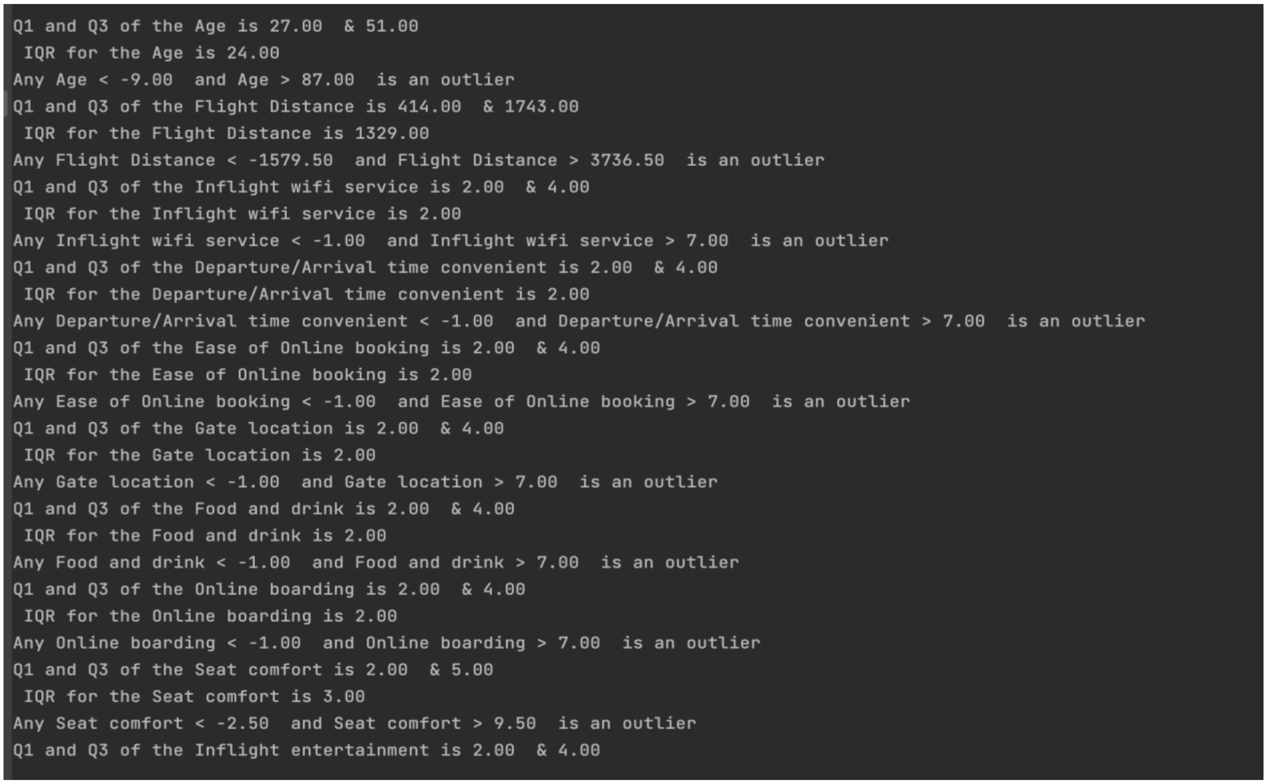
Visualization of all the variables using box plot helped in finding the variables with outliers.



Therefore the outliers from the following variables were removed: Flight distance, Checkin service ,Departure Delay in Minutes , Arrival Delay in Minutes

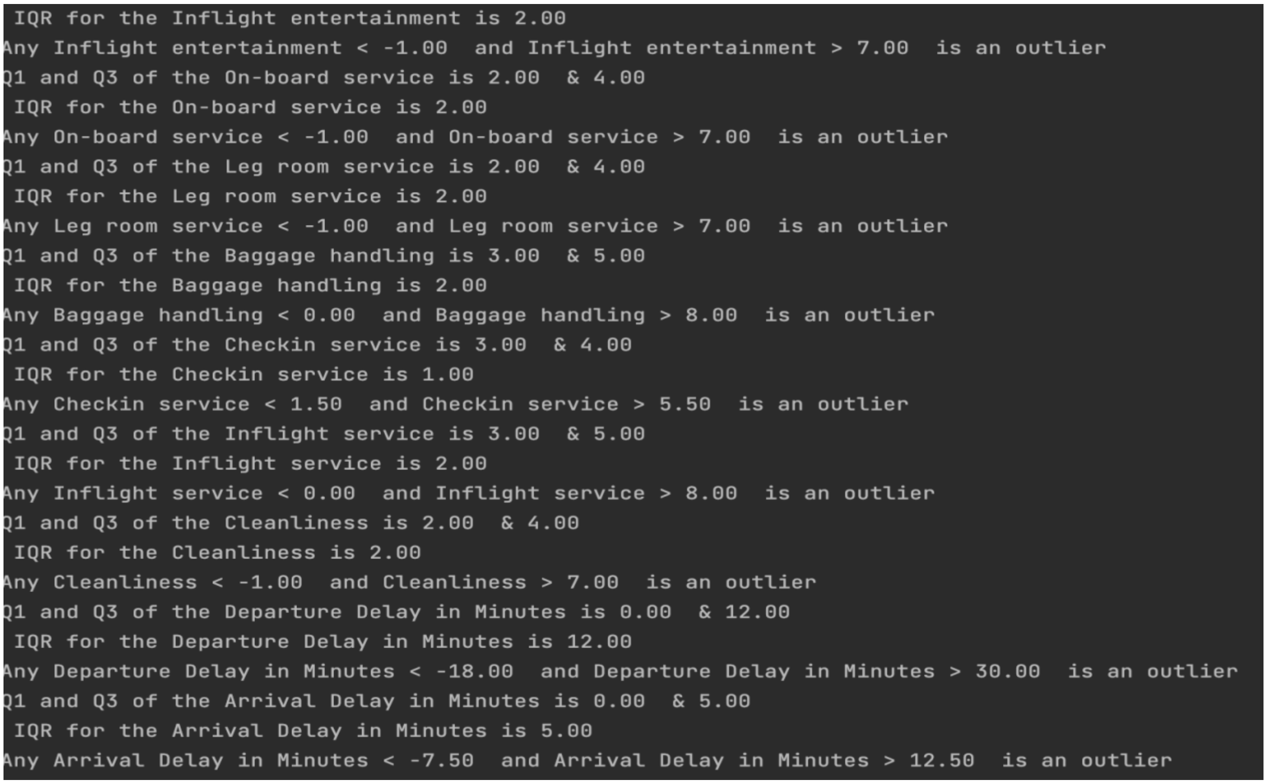
Fig 1.6 ***Variables with the outliers***

Based on the detection of outlier range using the IQR approach presented in the previous chapter, the outliers from these variables will be deleted. Any real data points for the variables that don't fall within the supplied upper and lower bounds are considered outliers and are taken out of the dataset by the IQR analysis.



```
Q1 and Q3 of the Age is 27.00  & 51.00
 IQR for the Age is 24.00
Any Age < -9.00  and Age > 87.00  is an outlier
Q1 and Q3 of the Flight Distance is 414.00  & 1743.00
 IQR for the Flight Distance is 1329.00
Any Flight Distance < -1579.50  and Flight Distance > 3736.50  is an outlier
Q1 and Q3 of the Inflight wifi service is 2.00  & 4.00
 IQR for the Inflight wifi service is 2.00
Any Inflight wifi service < -1.00  and Inflight wifi service > 7.00  is an outlier
Q1 and Q3 of the Departure/Arrival time convenient is 2.00  & 4.00
 IQR for the Departure/Arrival time convenient is 2.00
Any Departure/Arrival time convenient < -1.00  and Departure/Arrival time convenient > 7.00  is an outlier
Q1 and Q3 of the Ease of Online booking is 2.00  & 4.00
 IQR for the Ease of Online booking is 2.00
Any Ease of Online booking < -1.00  and Ease of Online booking > 7.00  is an outlier
Q1 and Q3 of the Gate location is 2.00  & 4.00
 IQR for the Gate location is 2.00
Any Gate location < -1.00  and Gate location > 7.00  is an outlier
Q1 and Q3 of the Food and drink is 2.00  & 4.00
 IQR for the Food and drink is 2.00
Any Food and drink < -1.00  and Food and drink > 7.00  is an outlier
Q1 and Q3 of the Online boarding is 2.00  & 4.00
 IQR for the Online boarding is 2.00
Any Online boarding < -1.00  and Online boarding > 7.00  is an outlier
Q1 and Q3 of the Seat comfort is 2.00  & 5.00
 IQR for the Seat comfort is 3.00
Any Seat comfort < -2.50  and Seat comfort > 9.50  is an outlier
Q1 and Q3 of the Inflight entertainment is 2.00  & 4.00
```

Fig 1.7 IQR ***analysis values for the variables***



```
 IQR for the Inflight entertainment is 2.00
Any Inflight entertainment < -1.00  and Inflight entertainment > 7.00  is an outlier
Q1 and Q3 of the On-board service is 2.00  & 4.00
 IQR for the On-board service is 2.00
Any On-board service < -1.00  and On-board service > 7.00  is an outlier
Q1 and Q3 of the Leg room service is 2.00  & 4.00
 IQR for the Leg room service is 2.00
Any Leg room service < -1.00  and Leg room service > 7.00  is an outlier
Q1 and Q3 of the Baggage handling is 3.00  & 5.00
 IQR for the Baggage handling is 2.00
Any Baggage handling < 0.00  and Baggage handling > 8.00  is an outlier
Q1 and Q3 of the Checkin service is 3.00  & 4.00
 IQR for the Checkin service is 1.00
Any Checkin service < 1.50  and Checkin service > 5.50  is an outlier
Q1 and Q3 of the Inflight service is 3.00  & 5.00
 IQR for the Inflight service is 2.00
Any Inflight service < 0.00  and Inflight service > 8.00  is an outlier
Q1 and Q3 of the Cleanliness is 2.00  & 4.00
 IQR for the Cleanliness is 2.00
Any Cleanliness < -1.00  and Cleanliness > 7.00  is an outlier
Q1 and Q3 of the Departure Delay in Minutes is 0.00  & 12.00
 IQR for the Departure Delay in Minutes is 12.00
Any Departure Delay in Minutes < -18.00  and Departure Delay in Minutes > 30.00  is an outlier
Q1 and Q3 of the Arrival Delay in Minutes is 0.00  & 5.00
 IQR for the Arrival Delay in Minutes is 5.00
Any Arrival Delay in Minutes < -7.50  and Arrival Delay in Minutes > 12.50  is an outlier
```

Fig 1.8 ***IQR analysis values for the variables***

The outliers from the variables are removed based on the IQR range provided. After the removal of outliers, we can see through the below graphs that the outlying points discussed on the previous figures have been significantly reduced.

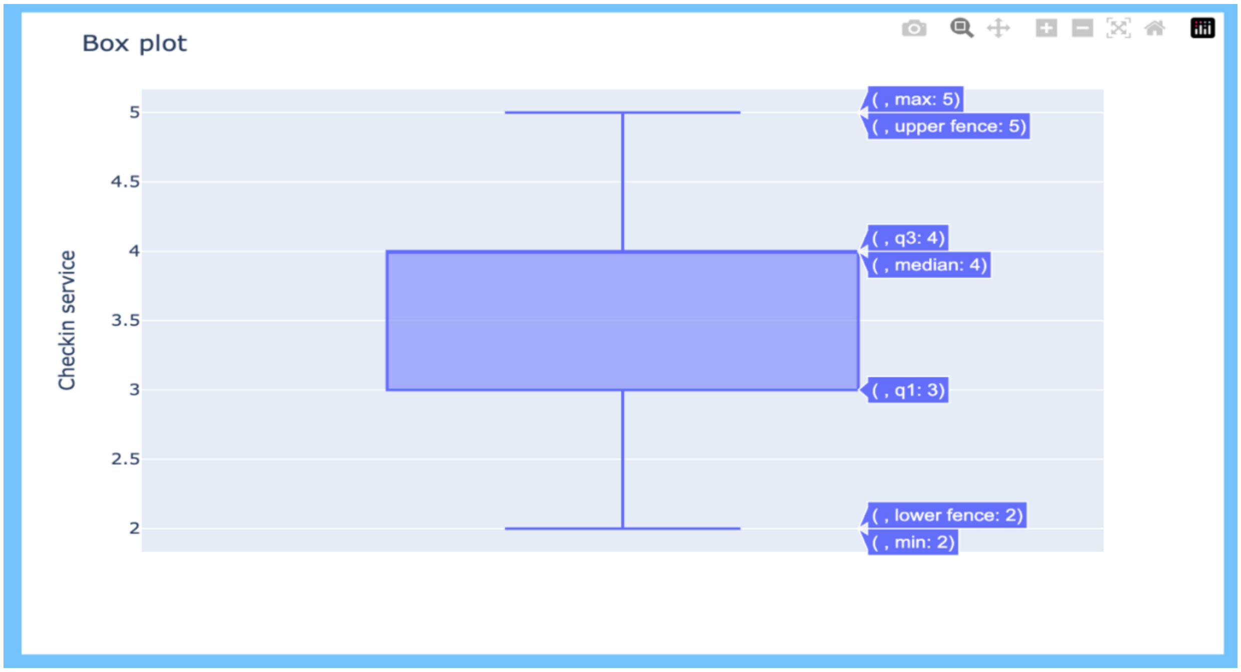
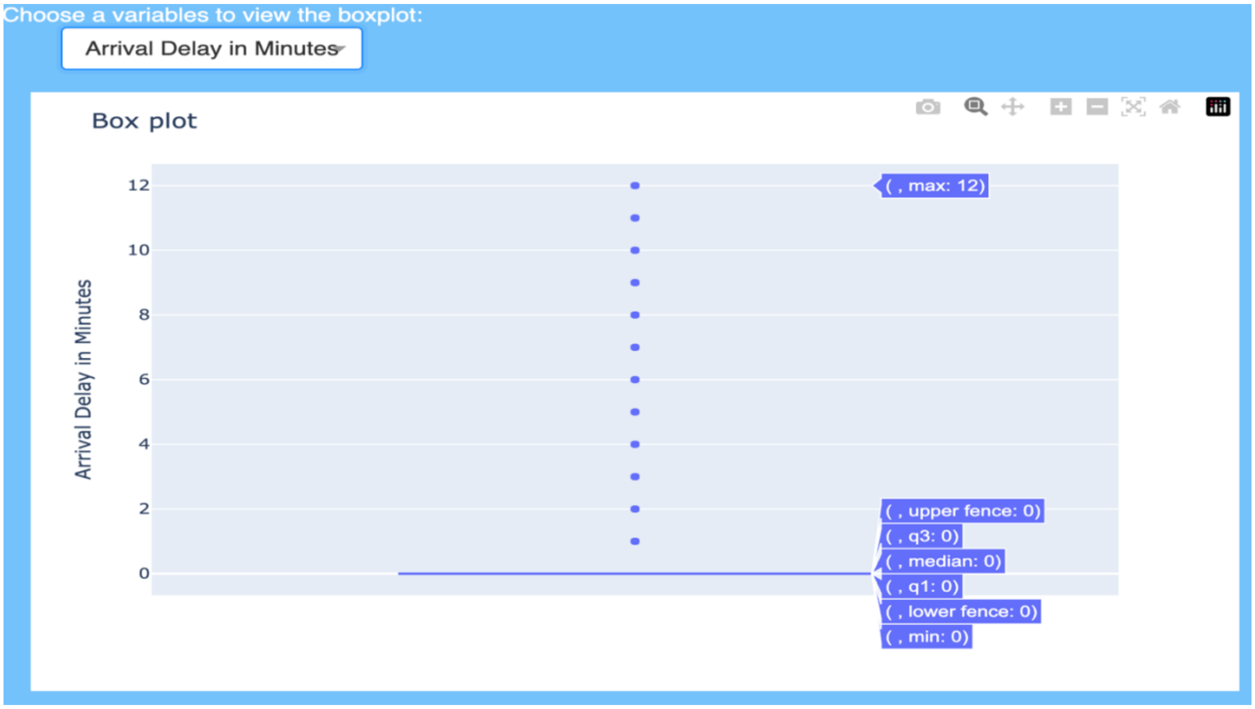


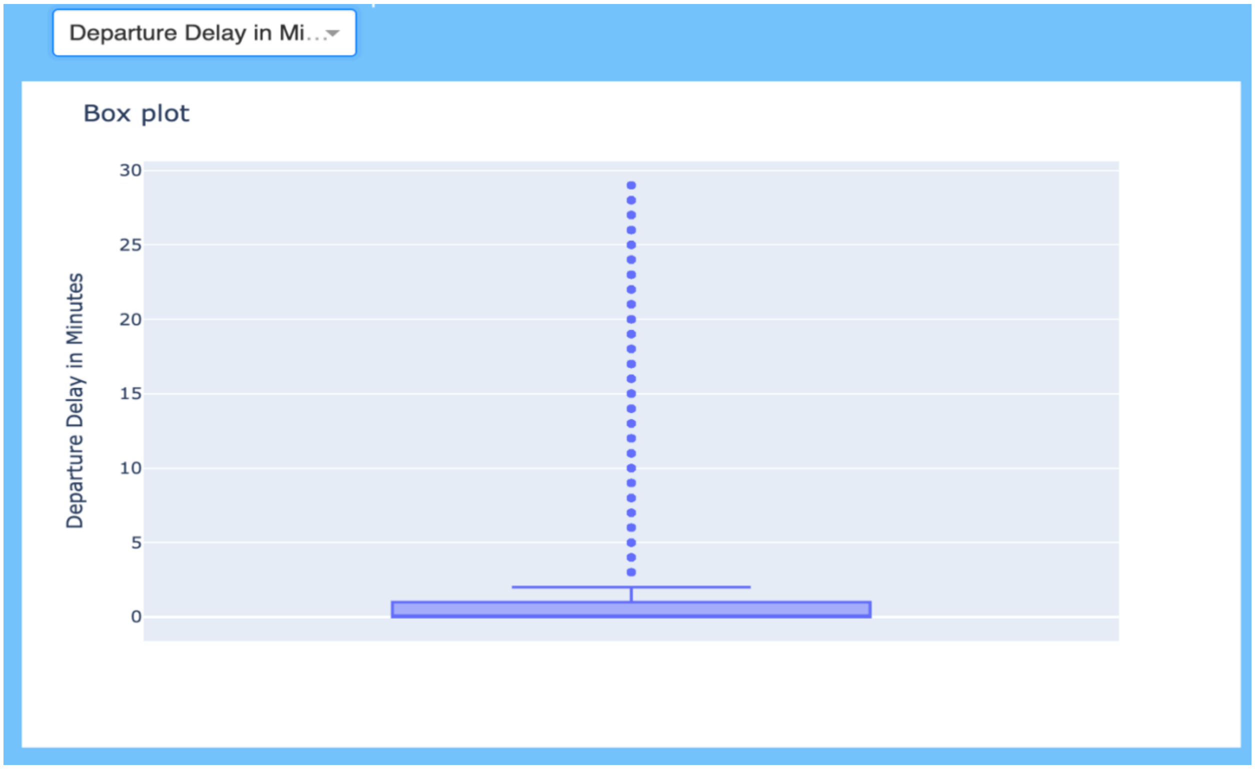
Fig 1.9 ***variable ‘checkin service’ after removing the outlier***



Choose a variables to view the boxplot:

Arrival Delay in Minutes

Fig 1.9.1 ***variable 'Arrival delay in minutes' after removing the outlier***

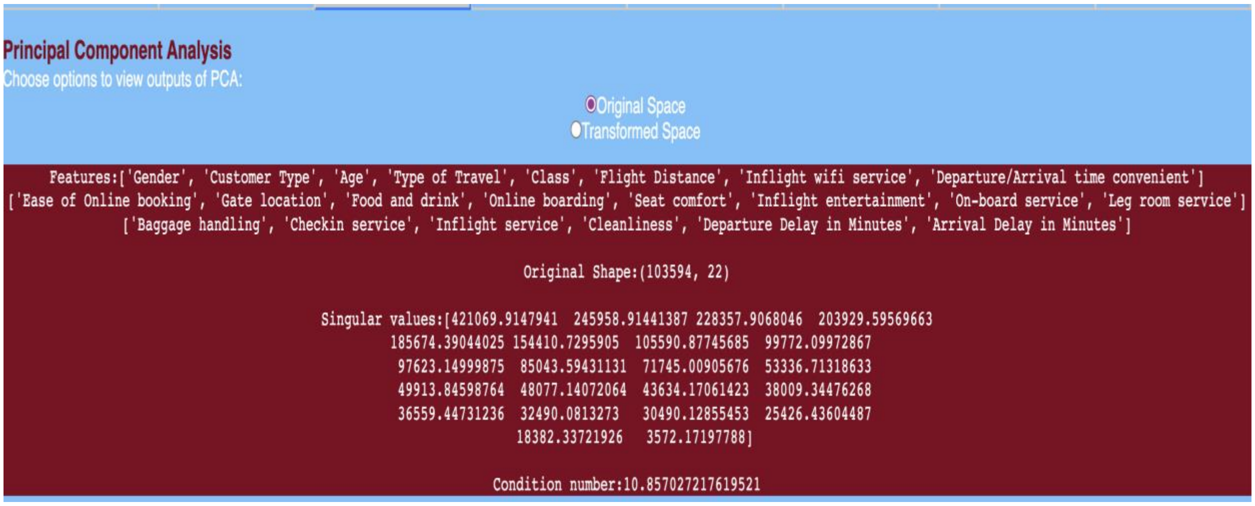


Departure Delay in Mi...

Fig 1.9.3 ***variable 'Departure delay in minutes' after removing the outliers.***

PRINCIPAL COMPONENT ANALYSIS (PCA):

As discussed on the previous chapter the PCA technique is helpful in reducing the feature dimensions. In other words, it helps us to find the best component/feature for further analysis.



Principal Component Analysis

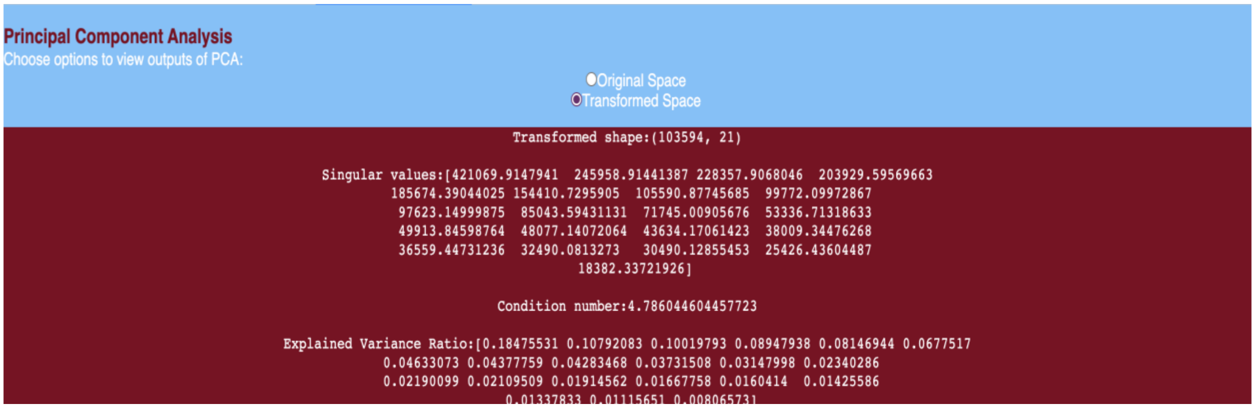
Choose options to view outputs of PCA:

● Original Space
○ Transformed Space

```
    Features:['Gender', 'Customer Type', 'Age', 'Type of Travel', 'Class', 'Flight Distance', 'Inflight wifi service', 'Departure/Arrival time convenient']
['Ease of Online booking', 'Gate location', 'Food and drink', 'Online boarding', 'Seat comfort', 'Inflight entertainment', 'On-board service', 'Leg room service']
      ['Baggage handling', 'Checkin service', 'Inflight service', 'Cleanliness', 'Departure Delay in Minutes', 'Arrival Delay in Minutes']

                                  Original Shape:(103594, 22)

                  Singular values:[421069.9147941  245958.91441387 228357.9068046  203929.59569663
                       185674.39044025 154410.7295905  105590.87745685  99772.09972867
                        97623.14999875  85043.59431131  71745.00905676  53336.71318633
                        49913.84598764  48077.14072064  43634.17061423  38009.34476268
                        36559.44731236  32490.0813273   30490.12855453  25426.43604487
                                        18382.33721926   3572.17197788]

                              Condition number:10.857027217619521
```

Fig 1.10.1 ***original feature space***

The 22 columns are considered as features of the data and the original feature space has a dimension of (103594, 22). The SVD analysis of the original featre sapce shows the condition number to be 10.85 and reducing the components through PCA would result in having much more smaller number. As we can see the singular values are also not converging to zero at the end.



Principal Component Analysis

Choose options to view outputs of PCA:

○ Original Space
● Transformed Space

```
                                 Transformed shape:(103594, 21)

                  Singular values:[421069.9147941  245958.91441387 228357.9068046  203929.59569663
                       185674.39044025 154410.7295905  105590.87745685  99772.09972867
                        97623.14999875  85043.59431131  71745.00905676  53336.71318633
                        49913.84598764  48077.14072064  43634.17061423  38009.34476268
                        36559.44731236  32490.0813273   30490.12855453  25426.43604487
                                            18382.33721926]

                              Condition number:4.786044604457723

     Explained Variance Ratio:[0.18475531 0.10792083 0.10019793 0.08947938 0.08146944 0.0677517
                         0.04633073 0.04377759 0.04283468 0.03731508 0.03147998 0.02340286
                         0.02190099 0.02109509 0.01914562 0.01667758 0.0160414  0.01425586
                                    0.01337833 0.01115651 0.00806573]
```

Fig 1.10.2 ***transformed feature space***

After performing the PCA the total components have been reduced from 22 to 21. It shows most of our features have much importance in the dataset. The explained variance ratio between original and transformed feature space has been displayed in the end of the text area. The condition number is 4.78.

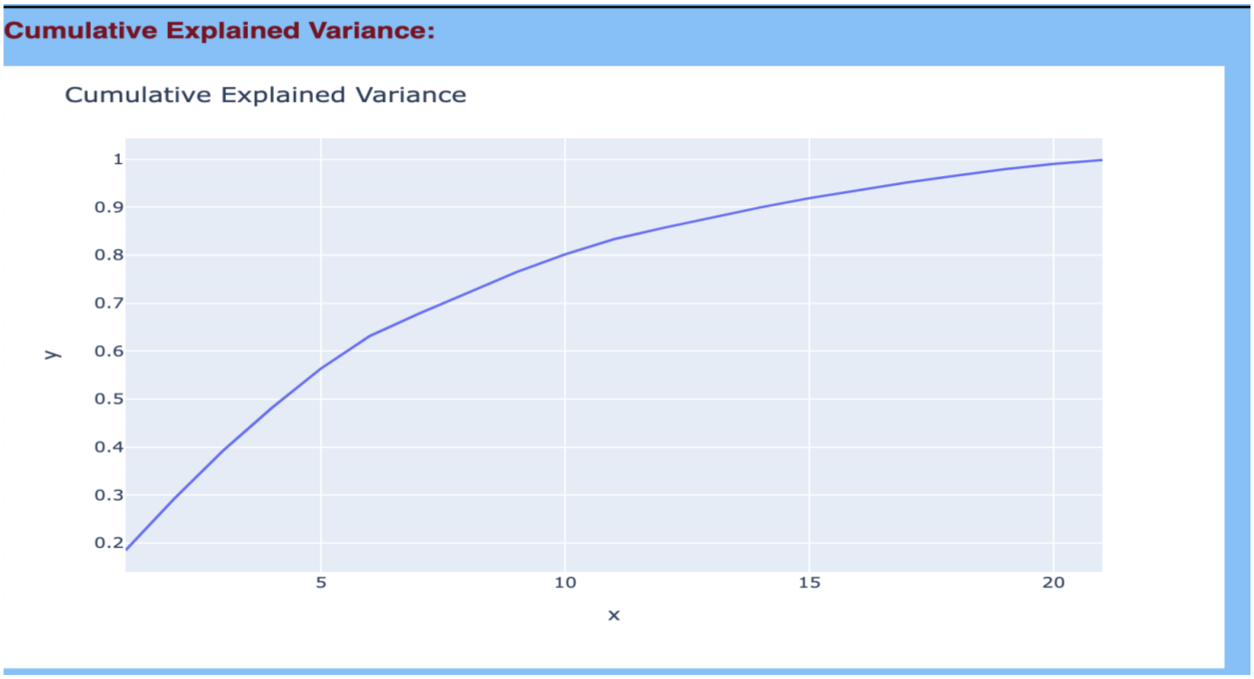
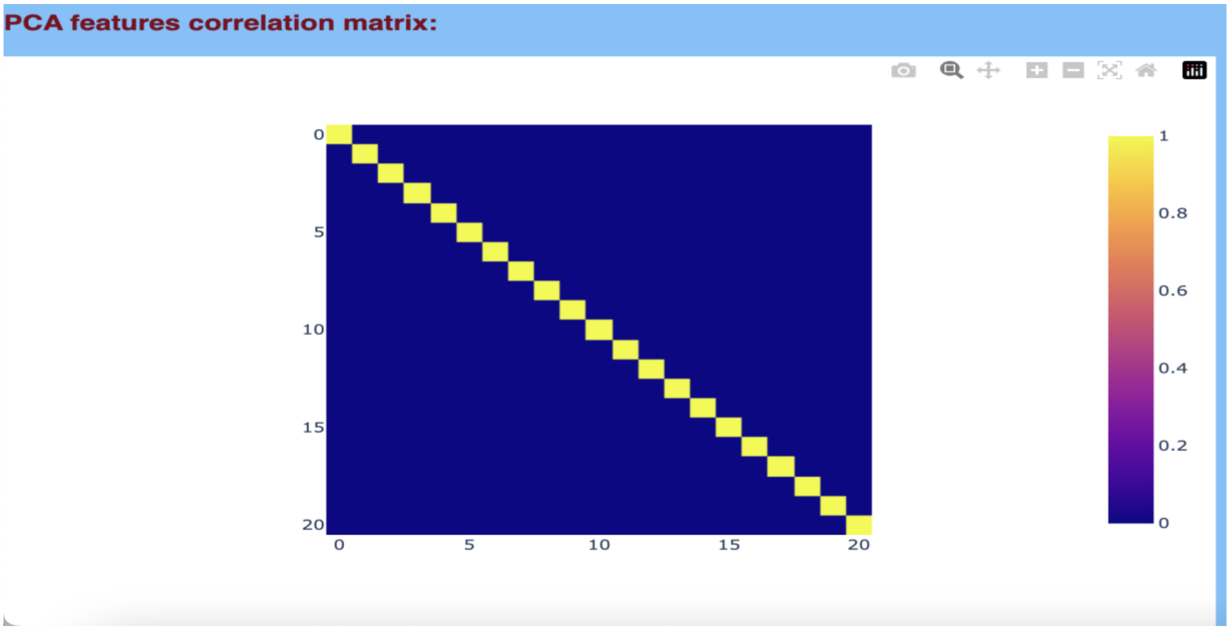


Fig 1.11 ***Cumulative explained variance plot***

The number of principle components is represented on the x-axis of a CEV plot, and the cumulative percentage of variance explained is shown on the y-axis. In the above fig 1.11, the x-axis represents the variables or components that has been reduced after PCA and y-axis represented between the range of 0-1. CEV plots are also useful for identifying the "elbow point", which is the point on the curve where adding more principal.

A heatmap representing the correlation between the components was generated. The graph generated on the dashboard doesn’t have annotations due the deprecation in recent plotly package hence seaborn version of heatmap was produced on console.



PCA features correlation matrix:

Fig 1.12 *PCA correlation matrix*

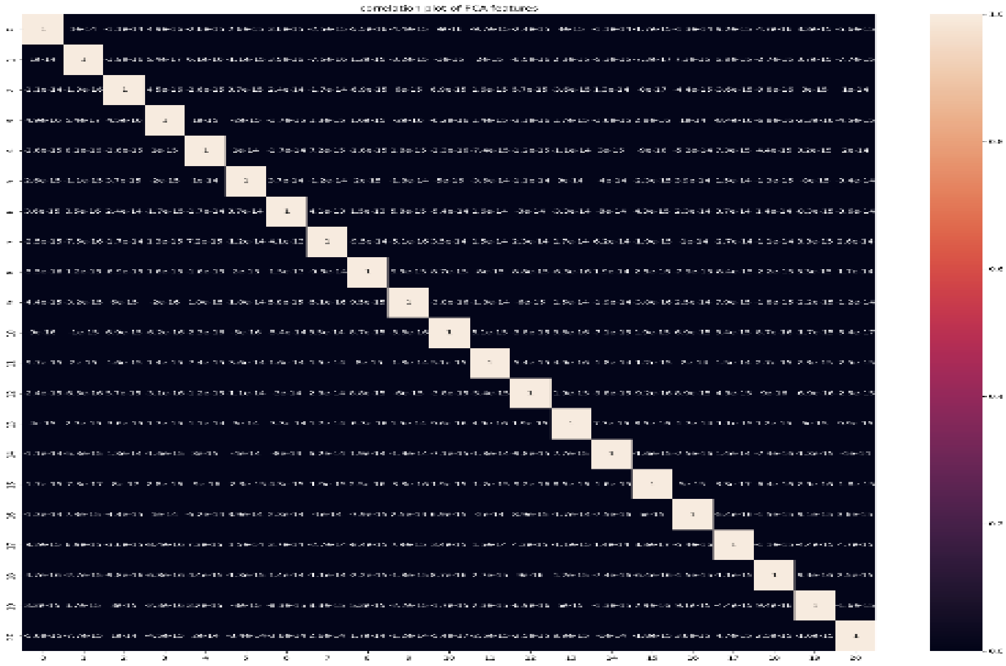
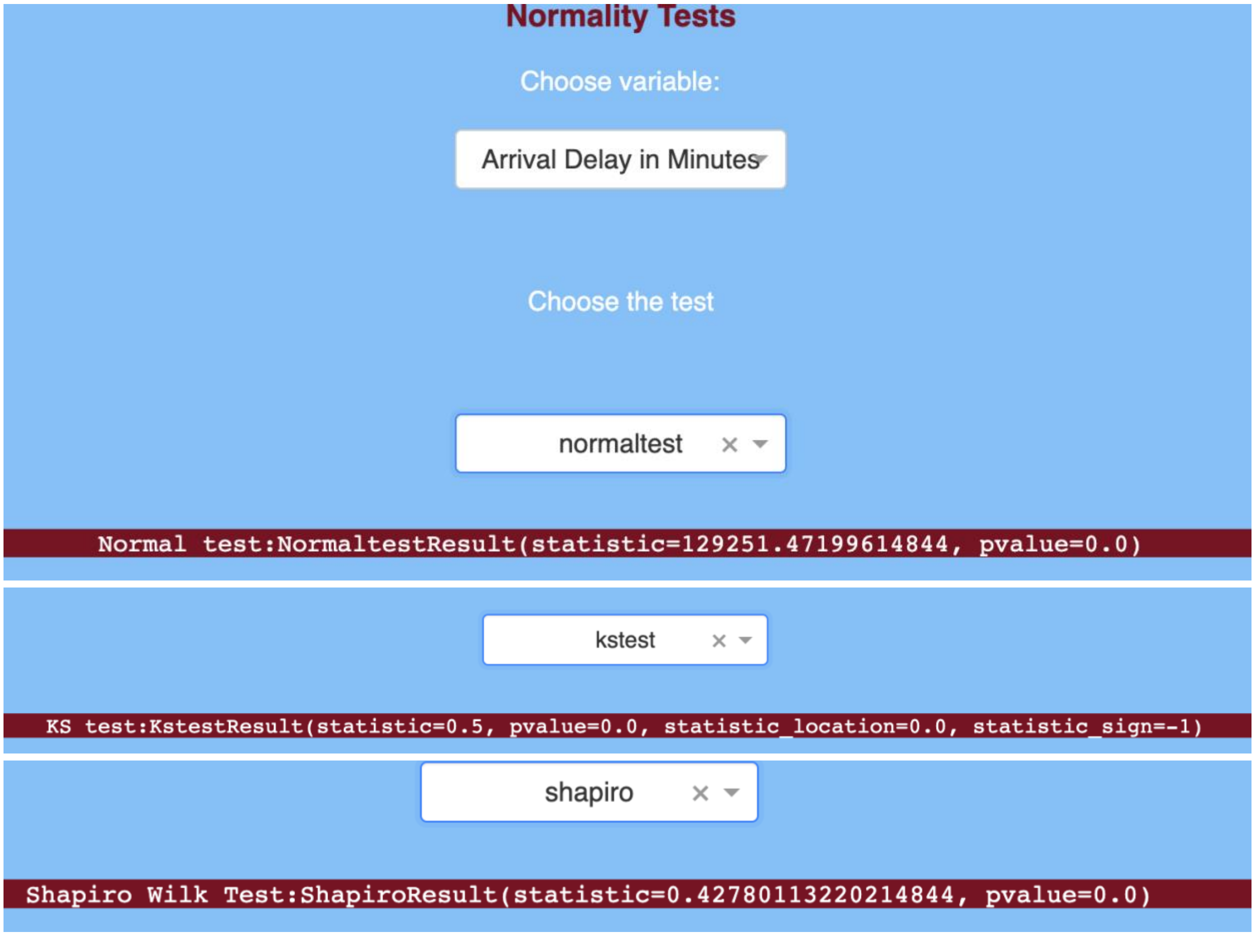


Fig 1.13 *PCA correlation matrix in console*

There might be no visible numbers seen from the above fig 1.13, but it is clear that the diagonal values are of light shade denoting the features with themselves have high correlation. But other parts of the plot has the darkest colour denoting nearly the zero value. It shows the features have very low collinearity amongst others.

NORMALITY TESTS:

The normality test tab environment is more user friendly because the user can select any numerical variable they want and also any test they need to do. Normal test (D-sqaured test), KS test and shapiro test. The user can run the one amongst three normality tests.



Normality Tests

Choose variable:

Arrival Delay in Minutes

Choose the test

normaltest

```
Normal test:NormaltestResult(statistic=129251.47199614844, pvalue=0.0)
```

kstest

```
KS test:KstestResult(statistic=0.5, pvalue=0.0, statistic_location=0.0, statistic_sign=-1)
```

shapiro

```
Shapiro Wilk Test:ShapiroResult(statistic=0.42780113220214844, pvalue=0.0)
```

Fig 1.14 *Normality tests*

Looking at the above figure, it is clear that the variable ‘Arrival delay in minutes’ doesn’t come from a normal distribution. Same way visualizing the other variables the p-value of all the test statistics were zero which shows the data doesn’t come from normal distribution. The screenshots of the other variables were not included considering the length of the report.

HEATMAP (PEARSON CORRELATION COEFFICIENT MATRIX) & SCATTER MATRIX:

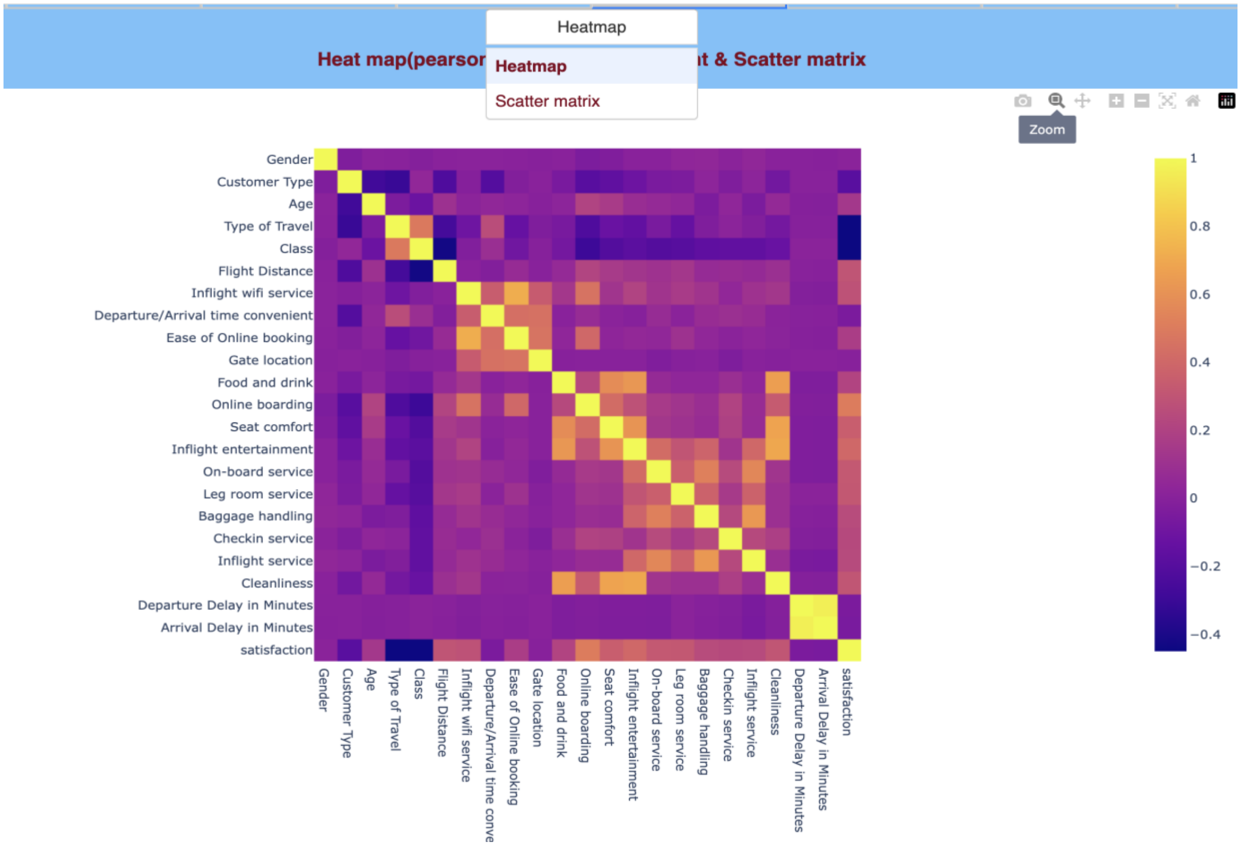


Fig 1.15 *Heatmap*

The generated heatmap provides the correlation amongst the different variables of the dataset. The cleaned dataset is used to produce the correlation of all the data variables and then used for plotting the heatmap. The given figure 1.15 displays a checker board pattern denoting there is some amount of collinearity in the dataset.

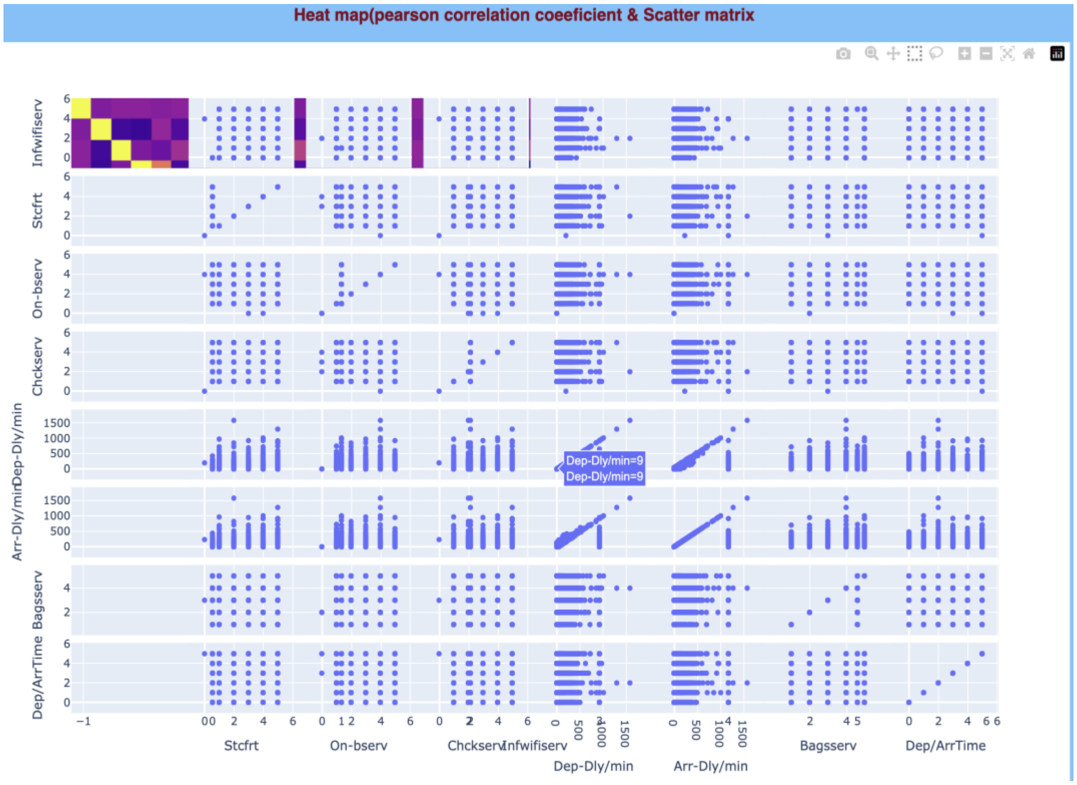
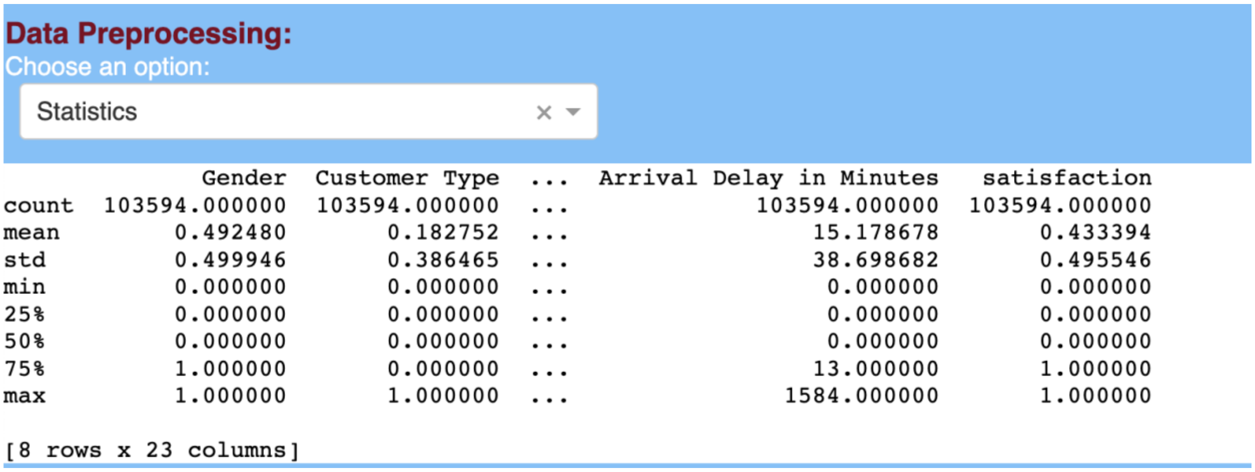


Fig 1.16 ***Scatter matrix***

The scatter matrix is shows the relationship between the different variables. In the above figure, I have taken certain values in order to reduce the congestion of the matrix.

STATISTICS:



Data Preprocessing:

Choose an option:

Statistics

```
              Gender  Customer Type  ...  Arrival Delay in Minutes   satisfaction
count  103594.000000  103594.000000  ...             103594.000000  103594.000000
mean        0.492480       0.182752  ...                 15.178678       0.433394
std         0.499946       0.386465  ...                 38.698682       0.495546
min         0.000000       0.000000  ...                  0.000000       0.000000
25%         0.000000       0.000000  ...                  0.000000       0.000000
50%         0.000000       0.000000  ...                  0.000000       0.000000
75%         1.000000       0.000000  ...                 13.000000       1.000000
max         1.000000       1.000000  ...               1584.000000       1.000000

[8 rows x 23 columns]
```

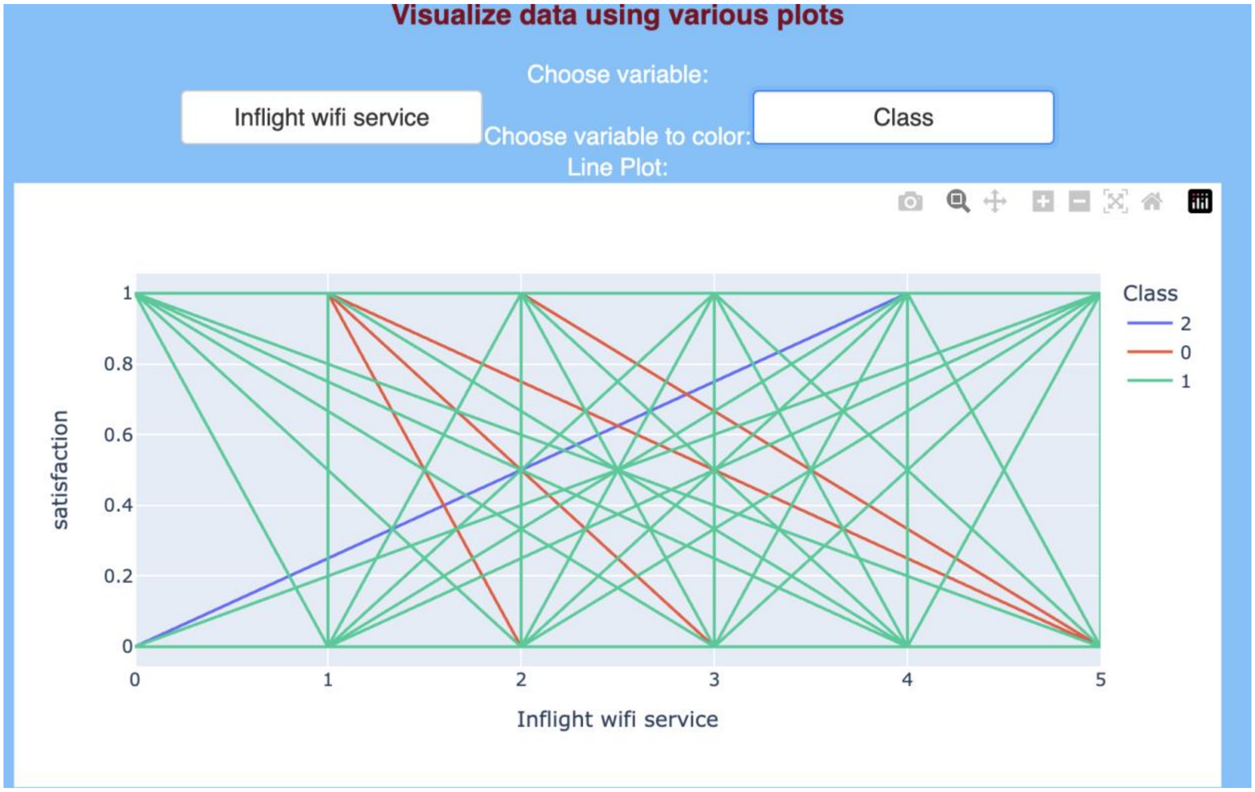
Fig 1.17 ***Descriptive statistics***

The above figure 1.17 shows the total count of values in each column of thw dataset. The mean, median, minimum values of the most columns are values less than or equal to zero. This is because most of the variables in the dataset has value range between 0 to 1 since they represent the aspect of the features in the dataset.

DATA VISUALIZATION:

The analysis tab helps in visualizing the dataset using different plots. Since most of the variables in the dataset are numerical there was limitations in applying more different varieties of plots for visualization.

Line plot:



Visualize data using various plots

Choose variable:

Choose variable to color:

Inflight wifi service

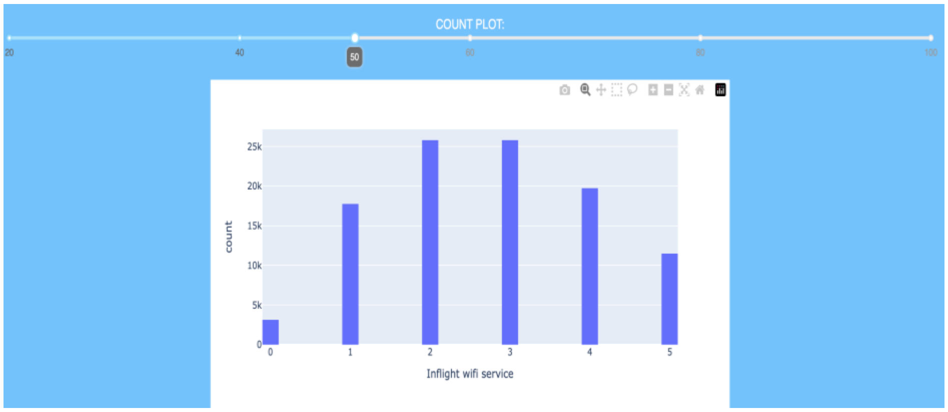
Class

Line Plot:

Fig 1.18 *Line plot*

This layout helps the user to select a variable (numeric) of their choice over the satisfaction level of the satisfaction. This plots uses ‘Inflight wifi service’ in x-axis which can be selected from the first drop-down and y-axis variable is ‘satisfaction’ which is pre decided. The hue for the plot can be selected based on the four categorical columns of the dataset.

Count plot:



COUNT PLOT

Fig 1.19 ***Count plot***

The above figure 1.19 shows the count plot with the range slider between 20 to 100. The count plot was plotted for Inflight wifi service.

The number of bins for the histogram can be selected using the slider option available in the layout. The slider has a tooltip which denotes the bins value highlighted for the user's view. The histogram uses the same variable user selected for the line plot in above division.

Histogram:

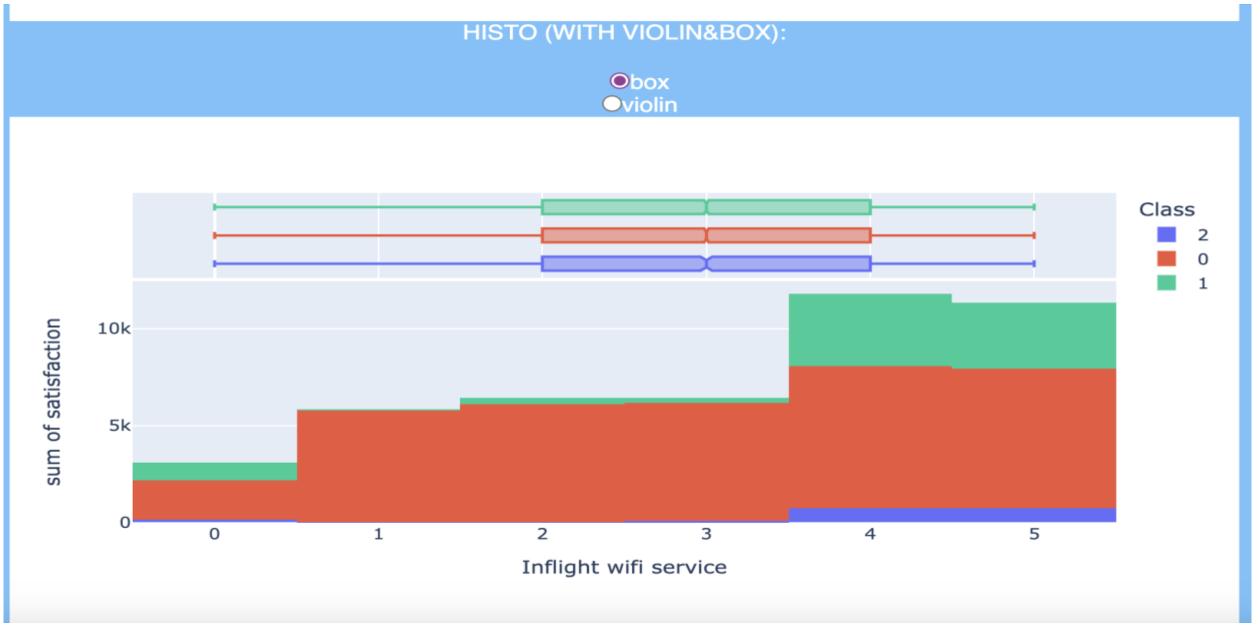


Fig 1.20 *Histogram*

In the above figure 1.20, the histogram is plotted with ‘Inflight wifi service’ in x-axis and ‘sum of satisfaction’ in y-axis. Hue has been taken as ‘class’ which has three features ‘economic’, ‘first class’, ‘business class’. The above plot has also included the box plot and the violin plot of the variable.

Finally, the values for all these plots are selected from the dropdown. Once the user select the desirable variable all the three plots changes accordingly.

GRAPHS:

The graph tab includes the pie chart, scatter plot and kde plot.

Pie plot:

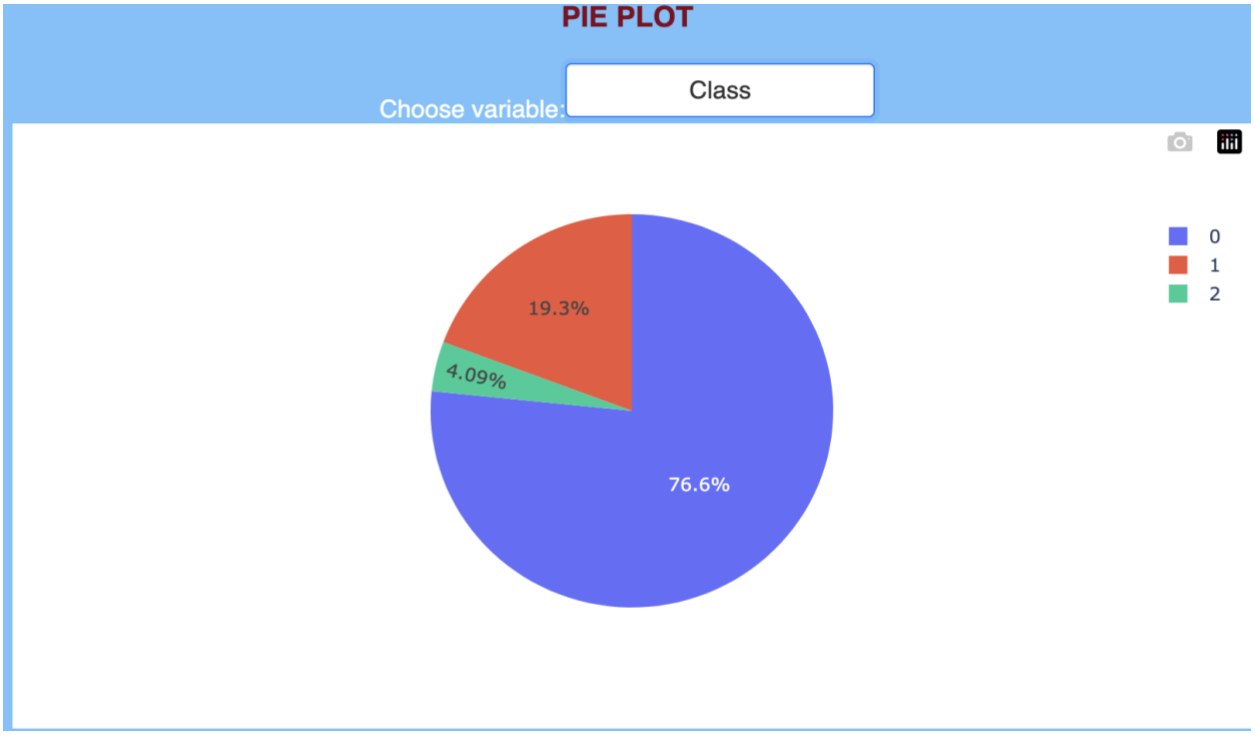
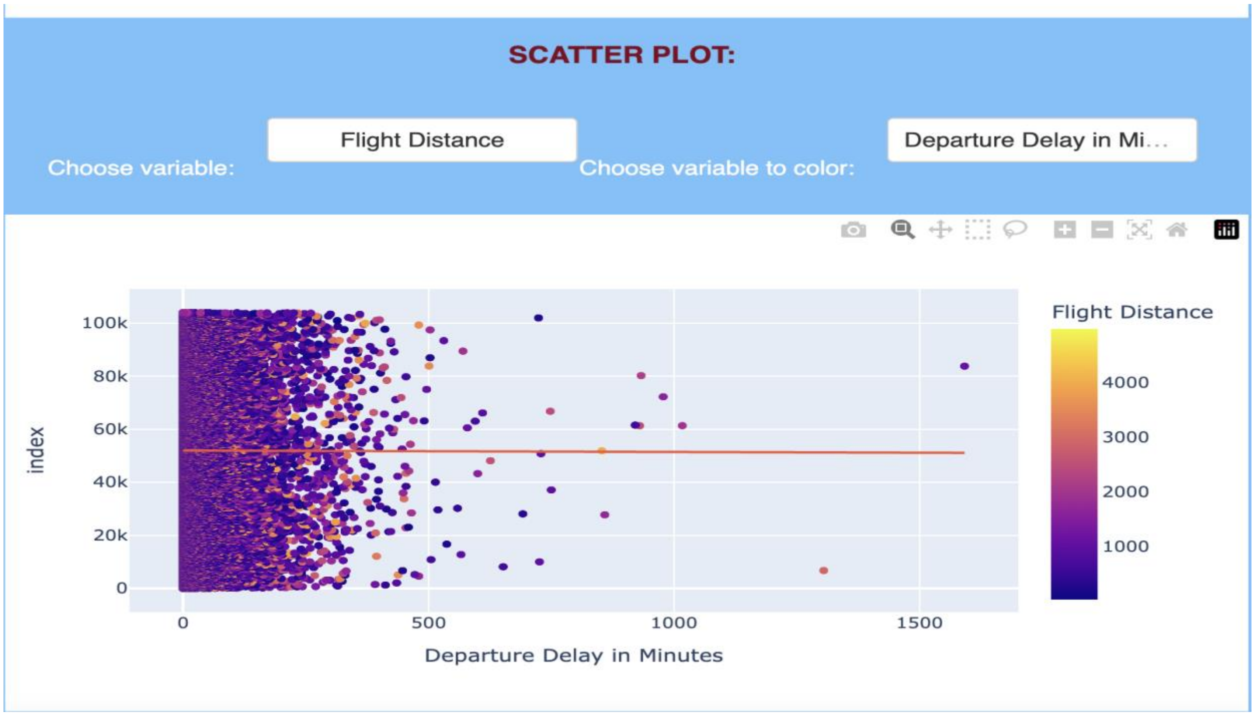


Fig 1.21 ***Pie chart***

The pie chart for representing the categorical columns where created with a dropdown menu which has all the categorical variables embedded in it. The above pie chart represents the percentage of the categorical value ‘class’ . Looking at the pie chart it is evident that one can understand the percentage of the features embedded in the class variable, whereas ‘0’-economic, ‘1’-eco plus, ‘2’-businessclass.

Scatter plot:



SCATTER PLOT:

Choose variable: Flight Distance

Choose variable to color: Departure Delay in Mi...

Fig 1.22 Scatter plot

The scatter plot analysis with a regression line is done in this tab. The x-axis can be selected from the dropdown menu and also the hue can be selected from the dropdown menu as well. So, the scatter matrix gets breifed after the 1000 minutes.

KDE:

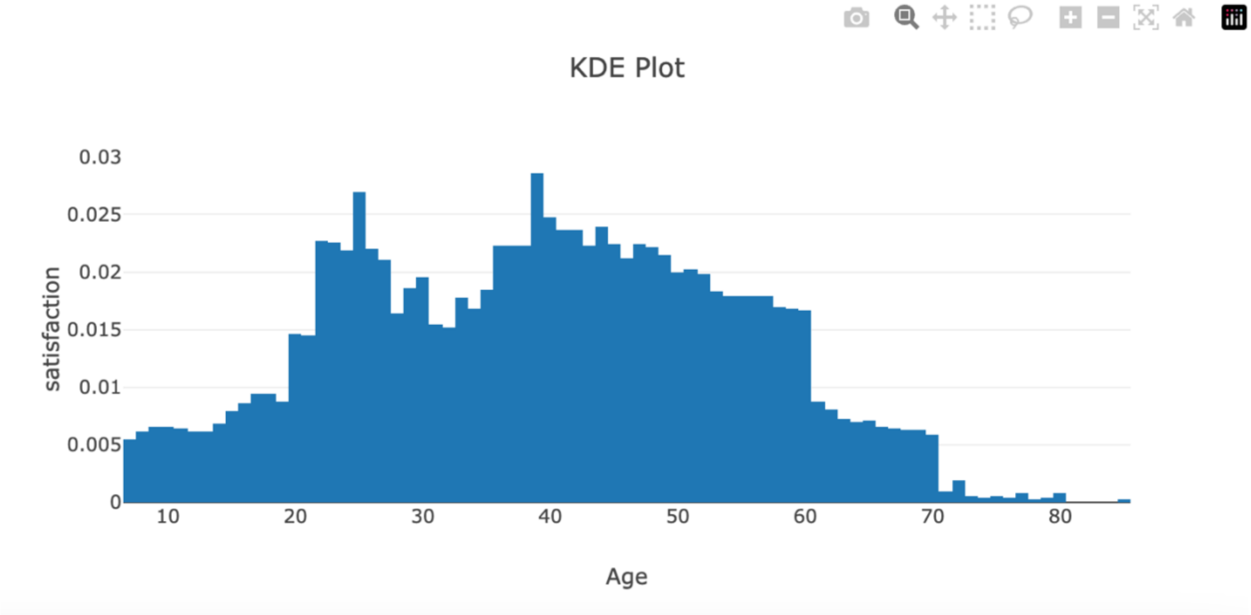


Fig 1.23 ***KDE plot***

KDE charts are helpful for comparing the distribution of different variables and for showing a variable's distribution. They can aid in locating distributional modes, skewness, and gaps that may not be clear from other graphical depictions. In the above figure 1.23 , the age in x-axis is plotted against satisfaction in y-axis.

Cat-plot:

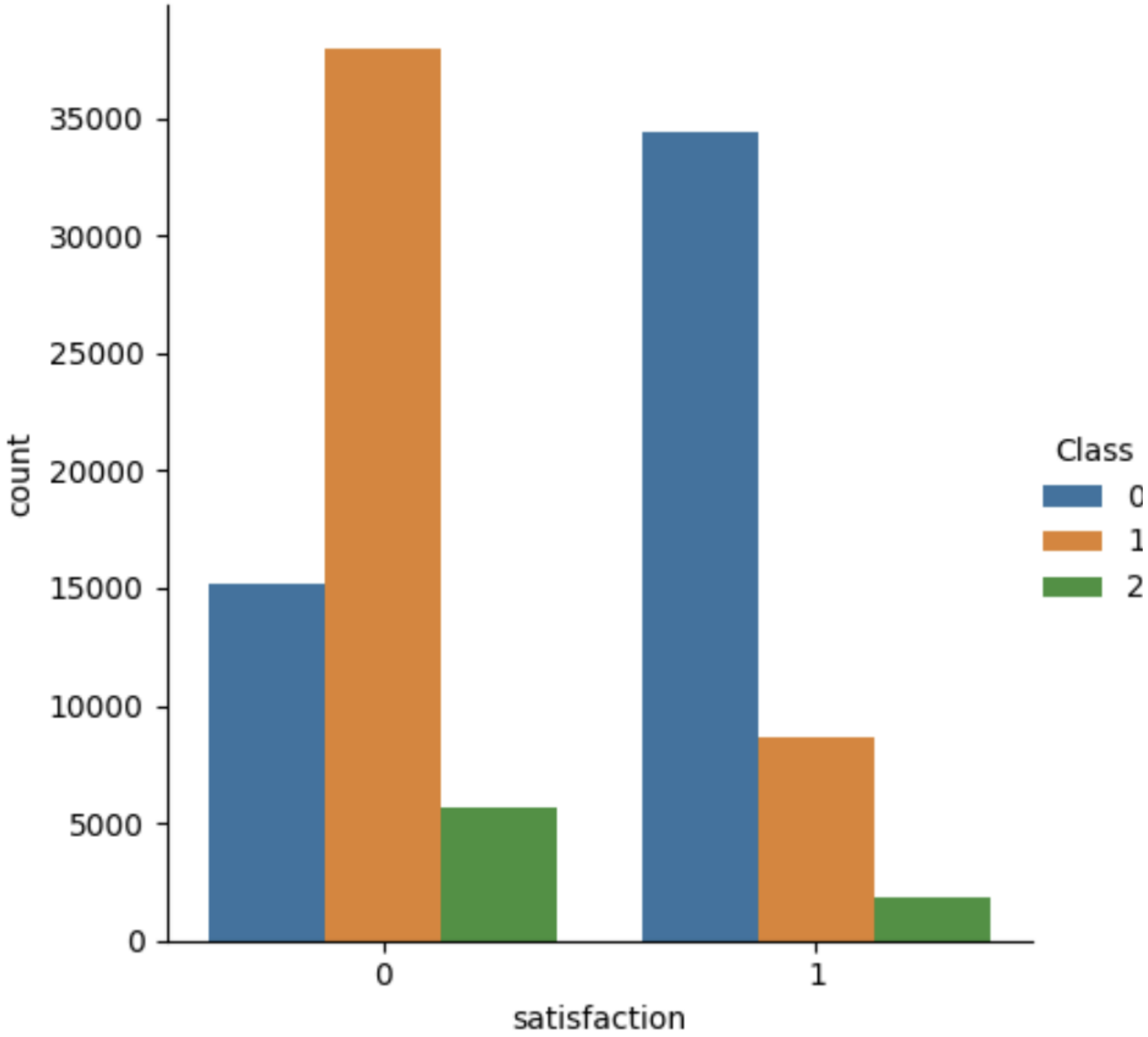


Fig 1.24 ***Cat plot***

Cat plots are useful for identifying patterns and trends in categorical data, such as differences between groups or changes over time. Cat plots may be used in a variety of ways to explore and visualize category data. In the above figure 1.24, the cat plot is plotted with 'satisfaction' in the x-axis and hue is taken as 'class'.

QQ-plot:

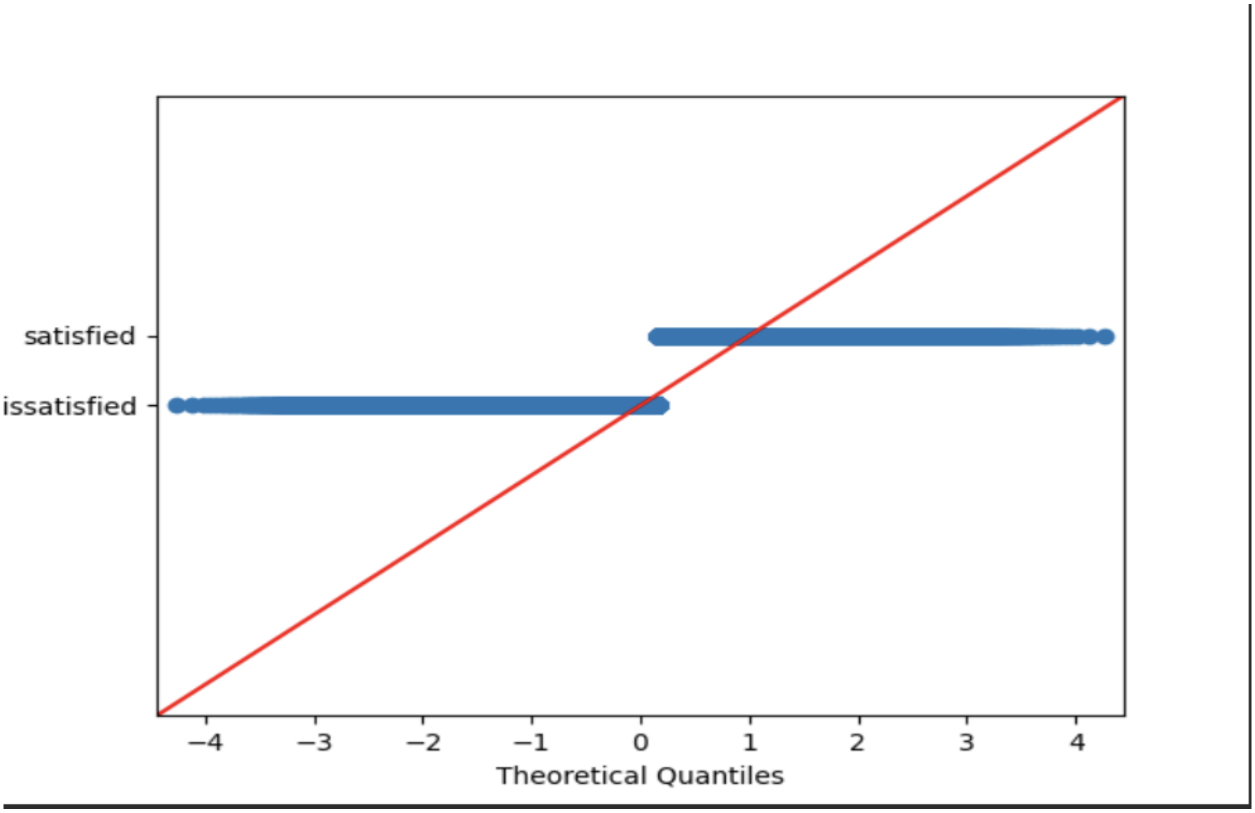


Fig 1.25 ***QQ-plot***

QQ plots (Quantile-Quantile plot) are helpful for determining if a sample of data is regularly distributed and for spotting outliers. This is a visual method for contrasting a sample of data's distribution with a hypothetical probability distribution, such the normal distribution. In the above figure 1.25, the qqplot plot is plotted for the 'satisfaction' variable which is the target variable.

Multivariate box plot:

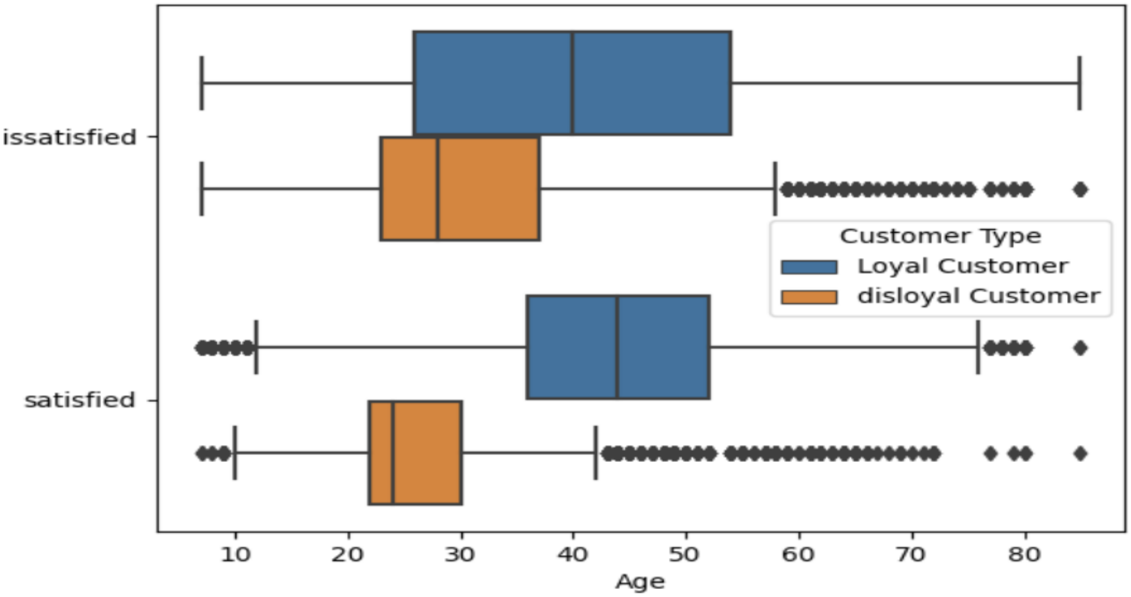


Fig 1.26 ***Multivariate boxplot***

Box plots with multiple variables are helpful for spotting variations in a numerical variable's distribution over various groups or categories. The whiskers extend from the greatest and lowest values that lie within 1.5 times the IQR, respectively, to the top and bottom of each box. Individual points outside the whiskers are plotted to represent outliers. It resembles a conventional box plot but includes more data dimensions.

In the above figure 1.26, the multivariate box plot is plotted with 'age' in x-axis and 'satisfaction' in y-axis with hue as a 'customer type'. It is clearly shown that the people between the age 20 to 30 are disloyal customer but still satisfied, people betweeen the age group of 35 to 55 are loyal customer with maximum level of satisfaction and parallely people betweeen the age group of 20 to 35 are disloyal customer with dissatisfaction, people betweeen the age group of 20 to 55 are loyal customer with maximum level of dissatisfaction

Dash core components:

The following set of dash components were used in the development of the given project.

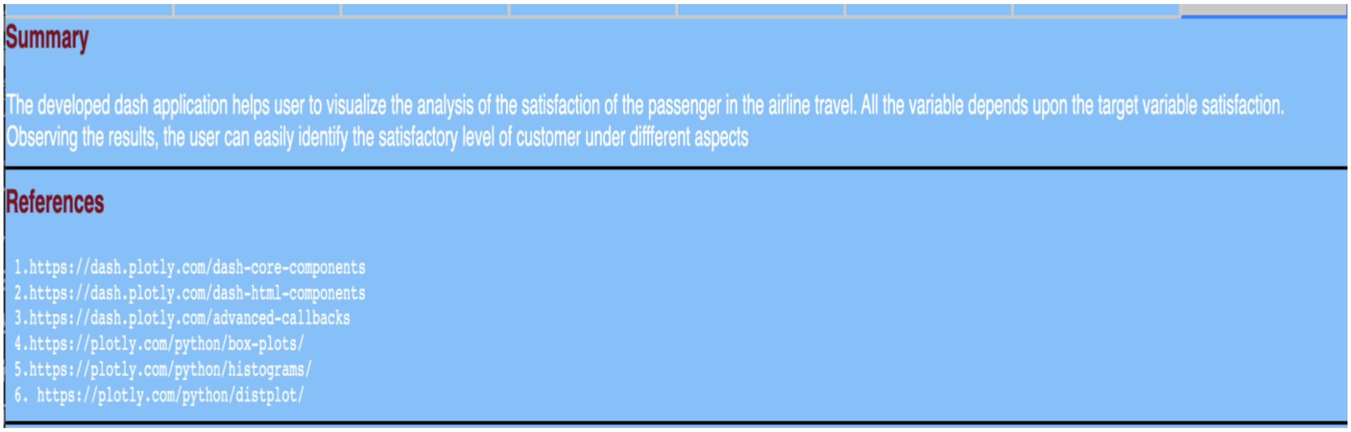
- Graph
- Tabs
- Multiple divisions
- Range slider
- Radio items
- Download component
- Upload component
- Text Area
- Dropdown
- Output
- Button

The following plots were used in the successful deployement of the project.

- Scatter plot with regression line
- Count plot
- Line plot
- Histogram
- Violin plot
- Box plot
- Pie chart
- Scatter matrix
- Kernel density function
- Cat plot (box type)
- QQ-plot

- Multivariate box plot
- Heatmaps

RECOMMENDATIONS:



Summary

The developed dash application helps user to visualize the analysis of the satisfaction of the passenger in the airline travel. All the variable depends upon the target variable satisfaction. Observing the results, the user can easily identify the satisfactory level of customer under diffferent aspects

References

1.https://dash.plotly.com/dash-core-components
2.https://dash.plotly.com/dash-html-components
3.https://dash.plotly.com/advanced-callbacks
4.https://plotly.com/python/box-plots/
5.https://plotly.com/python/histograms/
6. https://plotly.com/python/distplot/

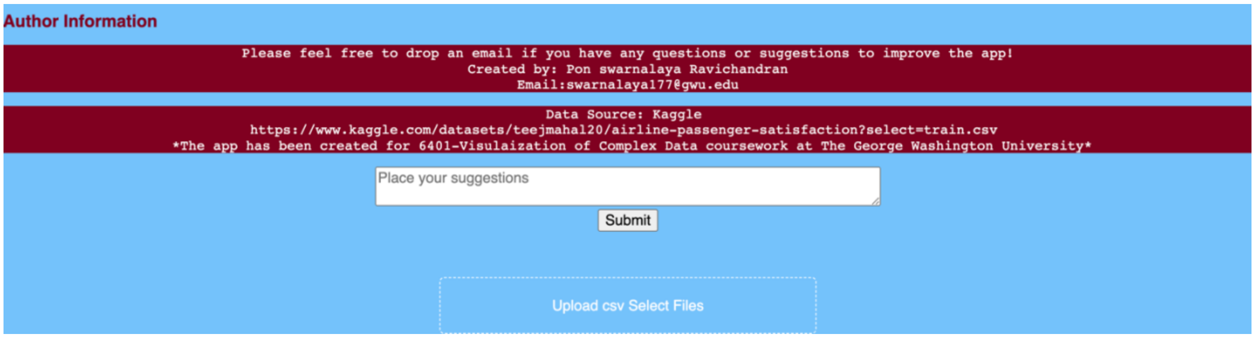
Fig 1.27 ***Summary tab***

The summary tab has the overall analysis of the passenger satisfaction throughout their airline travel with the respected airline. Since the opinions depends upon the various aspects of the airline business feature, the data was dragged across the different categorical and numerical values. The data set was not normal and had more outliers in four variables. After removing the outliers we were able to understand the data in a more well arranged manner. The plots explained that the most of people were neutral or dissatified with the majority percentage.

The established web-application has been designed in a way where the user can find the very basic statistics of the dataset to advanced analysis using plotly. The application is well organized and designed for easy navigation of the desired values. Each tab that has been created is made interactive by the use of the drop down menu which provides an homely environment for the user.

The summary section has the information about the author inclusive of the dataset information as well.

Additionally, there is an upload button and text box along with button to submit the feedback.



Author Information

Please feel free to drop an email if you have any questions or suggestions to improve the app!
Created by: Pon swarnalaya Ravichandran
Email:swarnalaya177@gwu.edu

Data Source: Kaggle
https://www.kaggle.com/datasets/teejmahal20/airline-passenger-satisfaction?select=train.csv
The app has been created for 6401-Visulaization of Complex Data coursework at The George Washington University

Fig 1.28 *Summary tab*

CONCLUSION

The project focused on analyzing the airline passenger data to find whether the people are satisfied or neutral/dissatisfied. The data more outliers only in the four variables and it was removed. The cleaned data was useful for vizualizing the data variables in such a way more information about the feature variables and their influence on the analysis were found. The cat plot, qq-plot and multivariate box plot wasn't able to be displayed in the dash app.

The GCP implementation of the app would be better version of the publishing the application online.

For the look at the established dash app click on the below link:
https://dashapp-7xpodtpsca-nn.a.run.app/

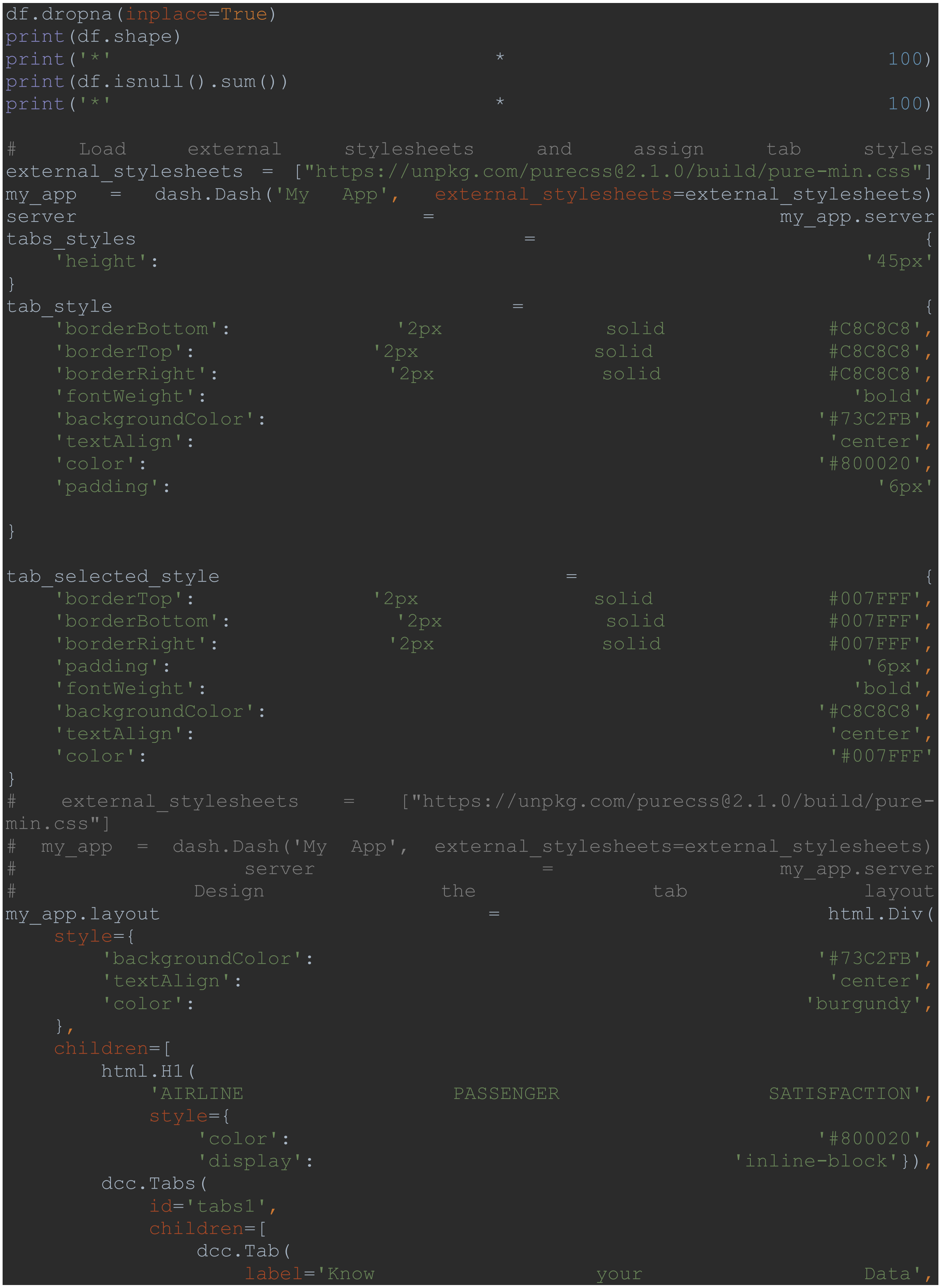
APPENDIX

Steps to run the project file:

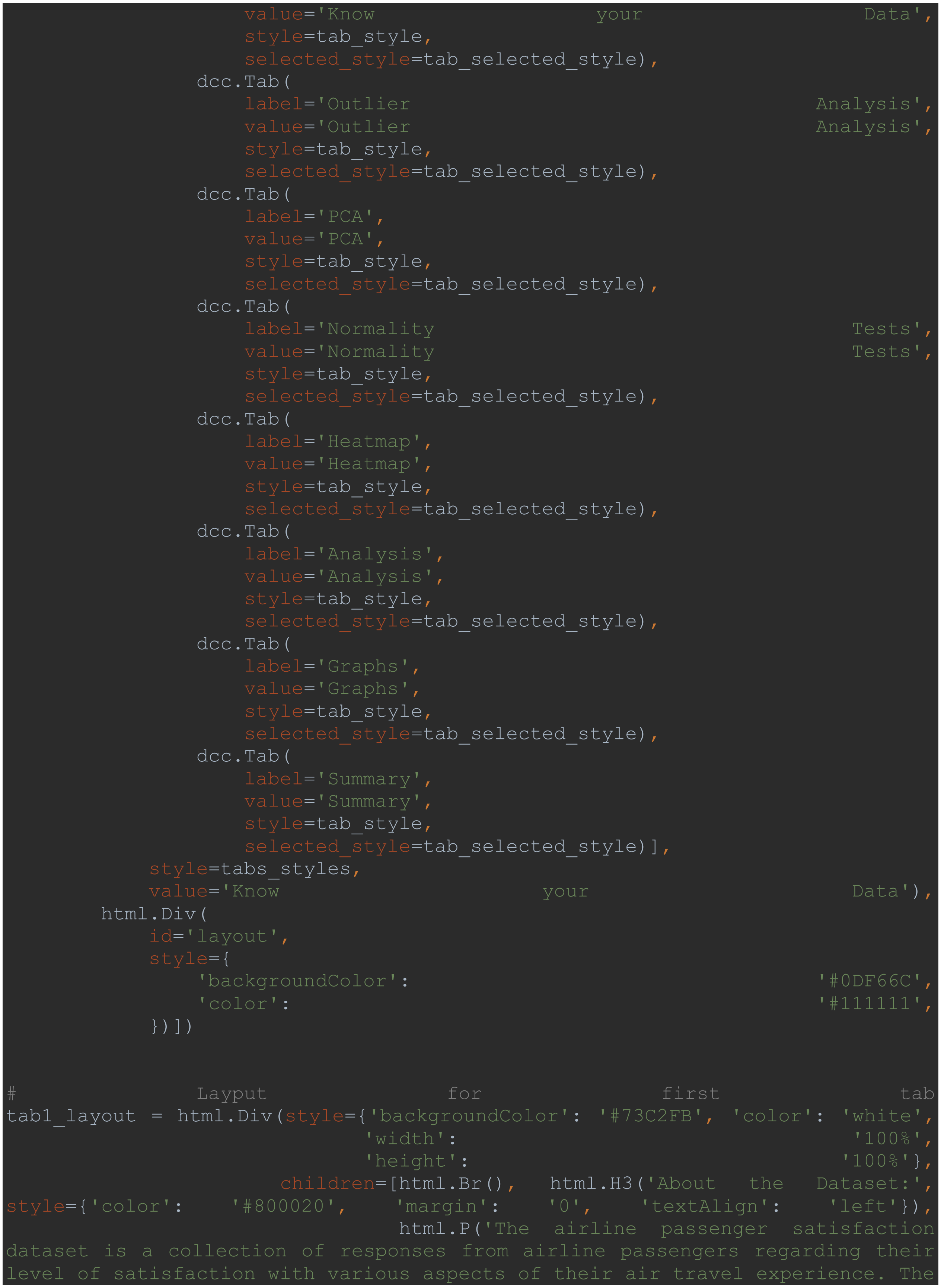
Run the Project_laya.py file to implement the project end-to-end.

Project_laya.py

```
import dash
import sm
from dash import dash_table
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import plotly.express as px
import seaborn as sns
from dash import dcc
from dash import html
from dash.dependencies import Input, Output, State
from numpy import linalg as la
from scipy.stats import kstest
from scipy.stats import normaltest
from scipy.stats import shapiro
from sklearn.decomposition import PCA
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from motionheading import app
import statsmodels.api as sm
import base64
import datetime
import io


# loading the datset
df = pd.read_csv('train .csv')
print(df.head())
print('*' * 100)

# dropped the missing values
```

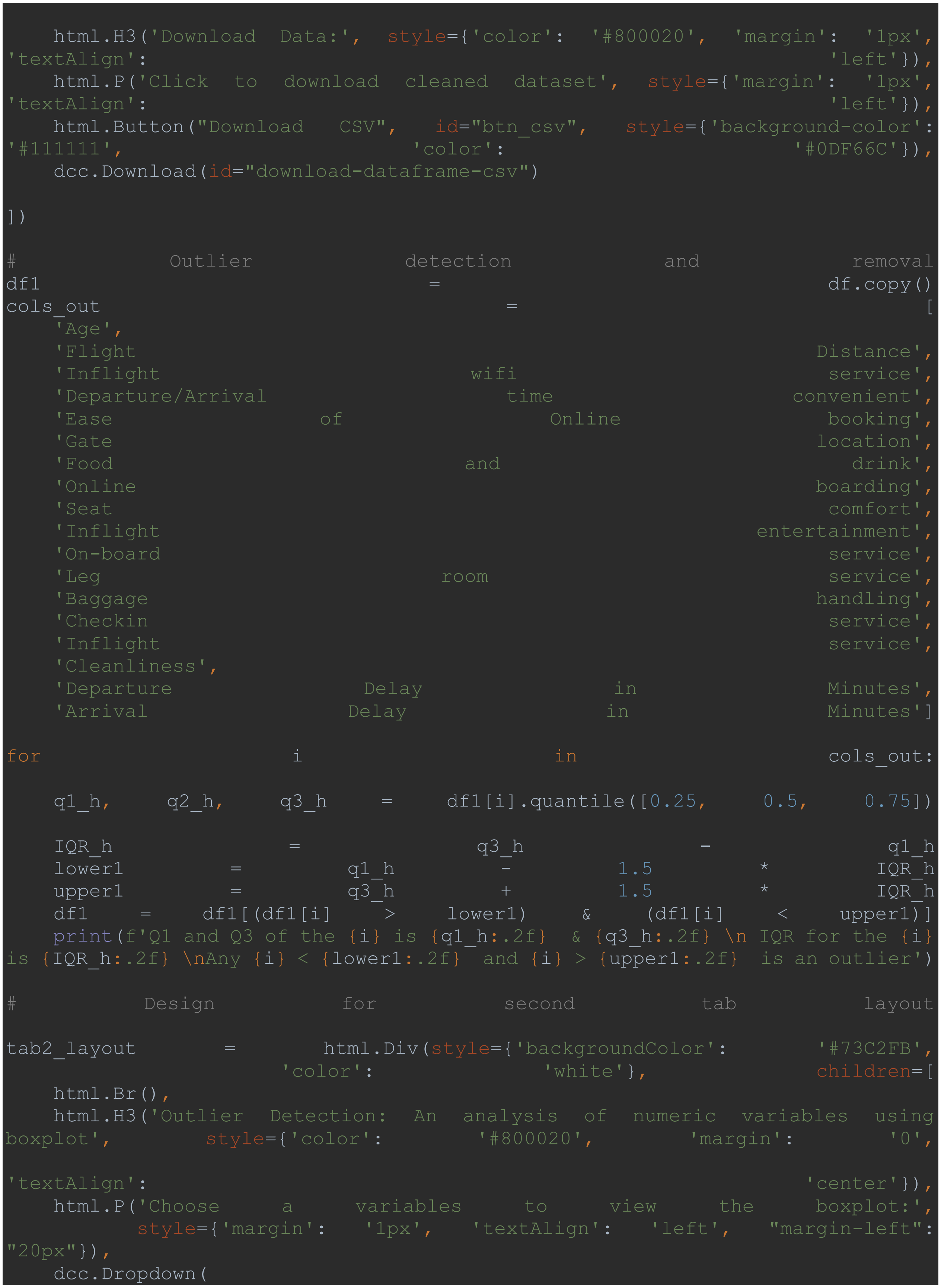


```
df.dropna(inplace=True)
print(df.shape)
print('*' * 100)
print(df.isnull().sum())
print('*' * 100)

# Load external stylesheets and assign tab styles
external_stylesheets = ["https://unpkg.com/purecss@2.1.0/build/pure-min.css"]
my_app = dash.Dash('My App', external_stylesheets=external_stylesheets)
server = my_app.server
tabs_styles = {
    'height': '45px'
}
tab_style = {
    'borderBottom': '2px solid #C8C8C8',
    'borderTop': '2px solid #C8C8C8',
    'borderRight': '2px solid #C8C8C8',
    'fontWeight': 'bold',
    'backgroundColor': '#73C2FB',
    'textAlign': 'center',
    'color': '#800020',
    'padding': '6px'

}

tab_selected_style = {
    'borderTop': '2px solid #007FFF',
    'borderBottom': '2px solid #007FFF',
    'borderRight': '2px solid #007FFF',
    'padding': '6px',
    'fontWeight': 'bold',
    'backgroundColor': '#C8C8C8',
    'textAlign': 'center',
    'color': '#007FFF'
}
# external_stylesheets = ["https://unpkg.com/purecss@2.1.0/build/pure-
min.css"]
# my_app = dash.Dash('My App', external_stylesheets=external_stylesheets)
# server = my_app.server
# Design the tab layout
my_app.layout = html.Div(
    style={
        'backgroundColor': '#73C2FB',
        'textAlign': 'center',
        'color': 'burgundy',
    },
    children=[
        html.H1(
            'AIRLINE PASSENGER SATISFACTION',
            style={
                'color': '#800020',
                'display': 'inline-block'}),
        dcc.Tabs(
            id='tabs1',
            children=[
                dcc.Tab(
                    label='Know your Data',
```



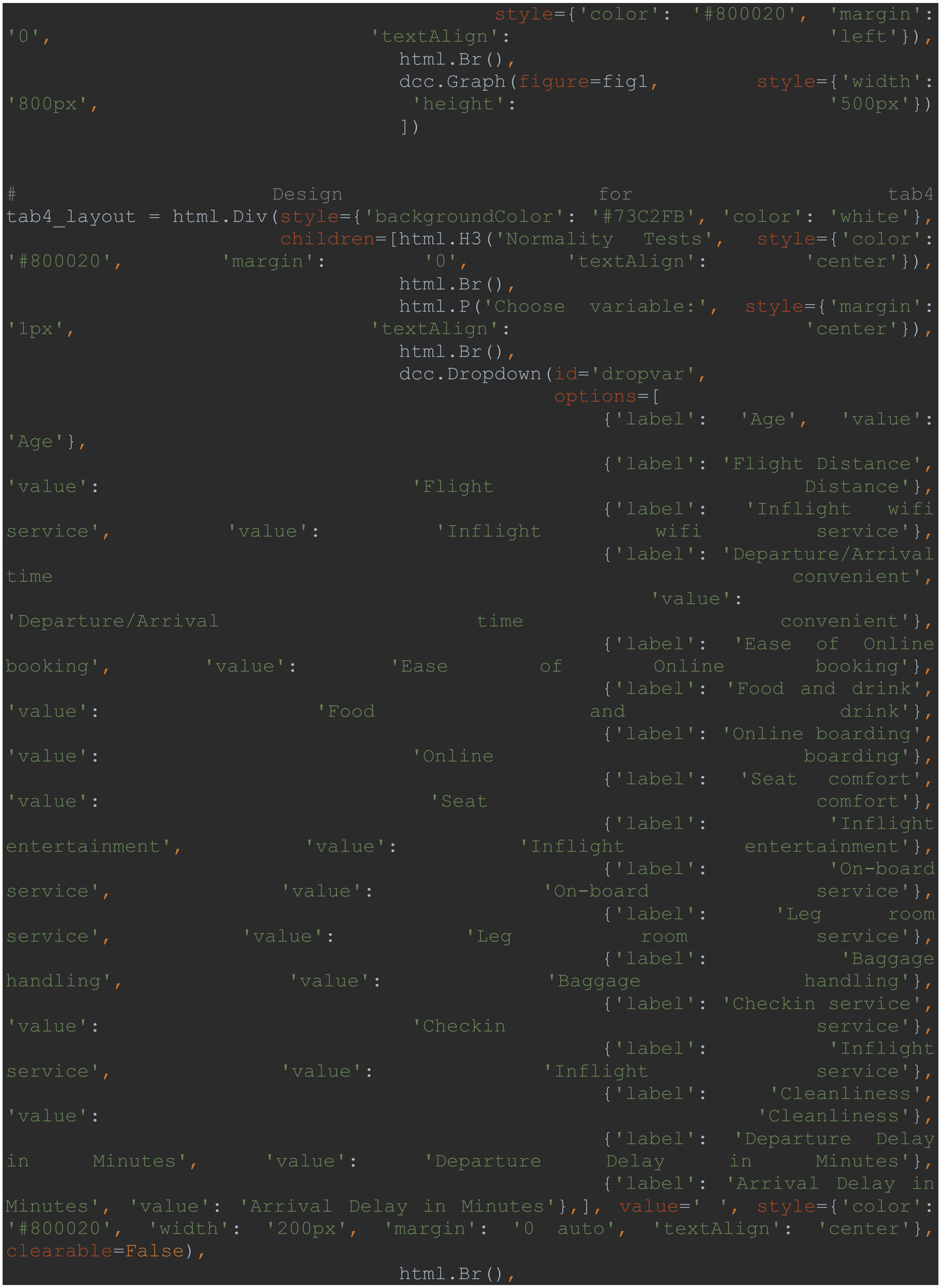
```
                    value='Know your Data',
                    style=tab_style,
                    selected_style=tab_selected_style),
                dcc.Tab(
                    label='Outlier Analysis',
                    value='Outlier Analysis',
                    style=tab_style,
                    selected_style=tab_selected_style),
                dcc.Tab(
                    label='PCA',
                    value='PCA',
                    style=tab_style,
                    selected_style=tab_selected_style),
                dcc.Tab(
                    label='Normality Tests',
                    value='Normality Tests',
                    style=tab_style,
                    selected_style=tab_selected_style),
                dcc.Tab(
                    label='Heatmap',
                    value='Heatmap',
                    style=tab_style,
                    selected_style=tab_selected_style),
                dcc.Tab(
                    label='Analysis',
                    value='Analysis',
                    style=tab_style,
                    selected_style=tab_selected_style),
                dcc.Tab(
                    label='Graphs',
                    value='Graphs',
                    style=tab_style,
                    selected_style=tab_selected_style),
                dcc.Tab(
                    label='Summary',
                    value='Summary',
                    style=tab_style,
                    selected_style=tab_selected_style)],
            style=tabs_styles,
            value='Know your Data'),
        html.Div(
            id='layout',
            style={
                'backgroundColor': '#0DF66C',
                'color': '#111111',
            })])


# Layput for first tab
tab1_layout = html.Div(style={'backgroundColor': '#73C2FB', 'color': 'white',
                              'width': '100%',
                              'height': '100%'},
                       children=[html.Br(), html.H3('About the Dataset:',
style={'color': '#800020', 'margin': '0', 'textAlign': 'left'}),
                                 html.P('The airline passenger satisfaction
dataset is a collection of responses from airline passengers regarding their
level of satisfaction with various aspects of their air travel experience. The
```

```
dataset contains 103,904 entries and 25 columns. The dataset includes both
categorical and numerical columns. ', style={'display': 'inline-block',
'margin': '0', 'textAlign': 'left'}),
                                 html.Br(),
                                 html.Br(),
                                 html.P('The categorical columns include
information such as the customers gender, type of travel, and class of service.
The numerical columns include metrics such as the flight distance, the level of
inflight service, and the delay times for departure and arrival.',
style={'display': 'inline-block', 'margin': '0', 'textAlign': 'left'}),
                                 html.Br(),
                                 html.Br(),
                                 html.P('The dataset was compiled from a survey
administered to airline passengers. The survey was designed to assess customer
satisfaction with various aspects of air travel, including the booking process,
check-in procedures, onboard service, and baggage handling.The ultimate goal of
analyzing this dataset is to gain insights into the factors that contribute to
customer satisfaction in air travel, and to identify areas where airlines can
improve the passenger experience', style={'display': 'inline-block', 'margin':
'0', 'textAlign': 'left'}),
                                 html.Hr(style={'border': '1px solid black'}),

                                 html.H3('About the data:', style={'color':
'#800020', 'margin': '1px', 'textAlign': 'left'}),
                                 html.P('Click one option to understand the
basic information about the data!', style={'margin': '1px', 'textAlign':
'left'}),
                                 dcc.Dropdown(id='infos', options=[
                                     {'label': 'Column Names', 'value':
'Column'},
                                     {'label': 'Count of rows', 'value':
'rows'},
                                     {'label': 'Count of columns', 'value':
'columns'
                                      }],
                           value='                                        ',
                           placeholder='Select an option',
                           style={'color': '#800020', 'margin-left': '5px',
'width': '50%', 'textAlign': 'left'}),
    html.Plaintext(id='datainfo', style={'backgroundColor': 'white', 'color':
'black', 'font-size': '15px', 'textAlign': 'left'}),
    html.Hr(style={'border': '1px solid black'}),

    html.H3('Data Preprocessing:', style={'color': '#800020', 'margin': '1px',
'textAlign': 'left'}),
    html.P('Choose an option:', style={'margin': '1px', 'textAlign': 'left'}),
    dcc.Dropdown(id='cleans', options=[
        {'label': 'Check for Null values', 'value': 'nulls'},
        {'label': 'Statistics', 'value': 'stats'},
        {'label': 'Data after preprocessing', 'value': 'head_d'}],
                           value='Column',
                           placeholder='Select an option',
                           style={'color': '#800020', 'margin-left': '5px',
'width': '50%', 'textAlign': 'left'}),
    html.Plaintext(id='preprocess', style={'backgroundColor': 'white',
'color': 'black', 'font-size': '15px', 'textAlign': 'left'}),
    html.Hr(style={'border': '1px solid black'}),
```

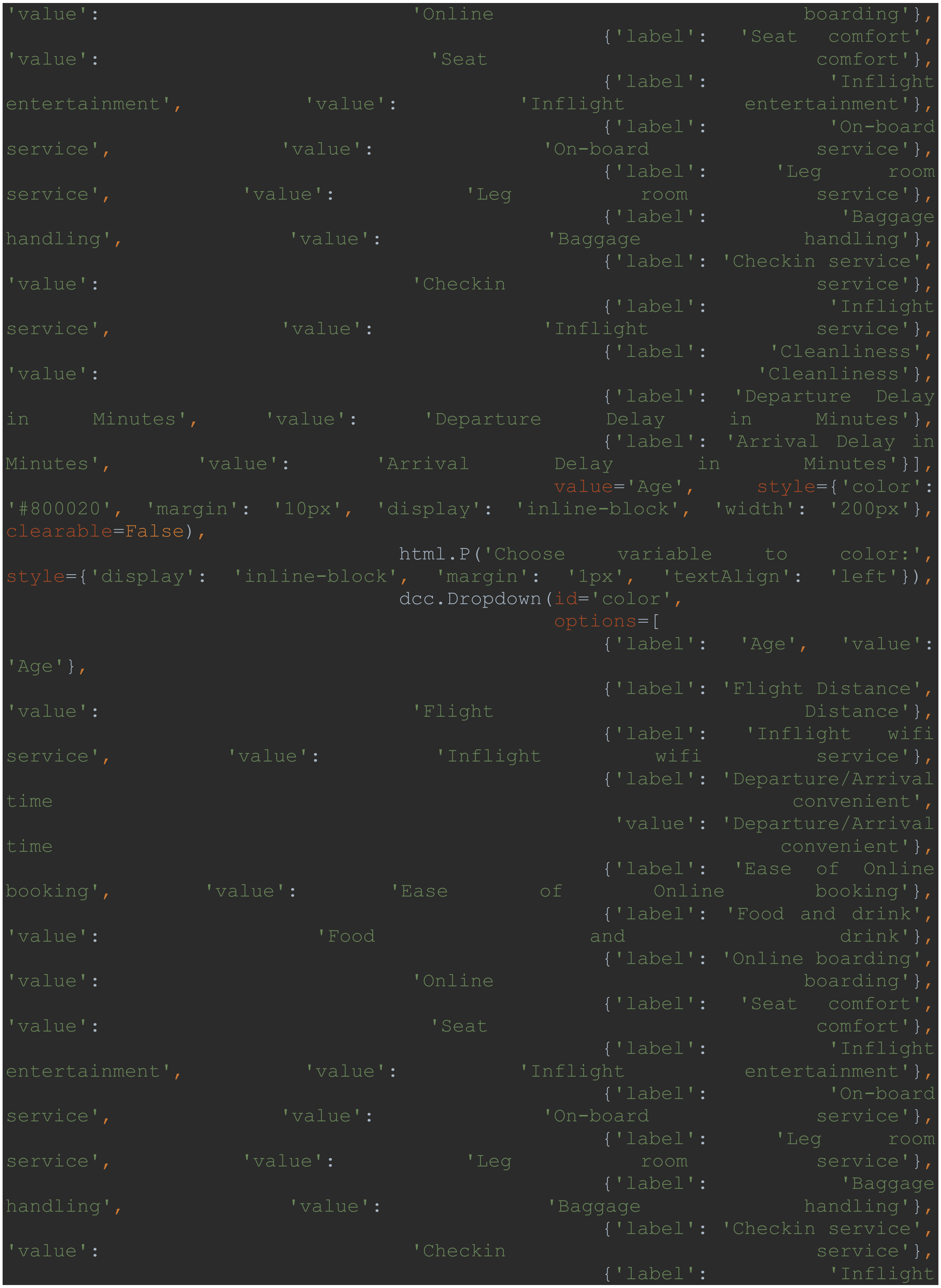


```
    html.H3('Download Data:', style={'color': '#800020', 'margin': '1px',
'textAlign': 'left'}),
    html.P('Click to download cleaned dataset', style={'margin': '1px',
'textAlign': 'left'}),
    html.Button("Download CSV", id="btn_csv", style={'background-color':
'#111111', 'color': '#0DF66C'}),
    dcc.Download(id="download-dataframe-csv")

])

# Outlier detection and removal
df1 = df.copy()
cols_out = [
    'Age',
    'Flight Distance',
    'Inflight wifi service',
    'Departure/Arrival time convenient',
    'Ease of Online booking',
    'Gate location',
    'Food and drink',
    'Online boarding',
    'Seat comfort',
    'Inflight entertainment',
    'On-board service',
    'Leg room service',
    'Baggage handling',
    'Checkin service',
    'Inflight service',
    'Cleanliness',
    'Departure Delay in Minutes',
    'Arrival Delay in Minutes']

for i in cols_out:

    q1_h, q2_h, q3_h = df1[i].quantile([0.25, 0.5, 0.75])

    IQR_h = q3_h - q1_h
    lower1 = q1_h - 1.5 * IQR_h
    upper1 = q3_h + 1.5 * IQR_h
    df1 = df1[(df1[i] > lower1) & (df1[i] < upper1)]
    print(f'Q1 and Q3 of the {i} is {q1_h:.2f} & {q3_h:.2f} \n IQR for the {i}
is {IQR_h:.2f} \nAny {i} < {lower1:.2f} and {i} > {upper1:.2f} is an outlier')

# Design for second tab layout

tab2_layout = html.Div(style={'backgroundColor': '#73C2FB',
'color': 'white'}, children=[
    html.Br(),
    html.H3('Outlier Detection: An analysis of numeric variables using
boxplot', style={'color': '#800020', 'margin': '0',

'textAlign': 'center'}),
    html.P('Choose a variables to view the boxplot:',
style={'margin': '1px', 'textAlign': 'left', "margin-left":
"20px"}),
    dcc.Dropdown(
        id='drop1',
        options=[
            {'label': 'Age', 'value': 'Age'},
            {'label': 'Flight Distance', 'value': 'Flight Distance'},
            {'label': 'Inflight wifi service',
             'value': 'Inflight wifi service'},
            {'label': 'Departure/Arrival time convenient',
             'value': 'Departure/Arrival time convenient'},
            {'label': 'Ease of Online booking',
             'value': 'Ease of Online booking'},
            {'label': 'Food and drink', 'value': 'Food and drink'},
            {'label': 'Online boarding', 'value': 'Online boarding'},
            {'label': 'Seat comfort', 'value': 'Seat comfort'},
            {'label': 'Inflight entertainment',
             'value': 'Inflight entertainment'},
            {'label': 'On-board service', 'value': 'On-board service'},
            {'label': 'Leg room service', 'value': 'Leg room service'},
            {'label': 'Baggage handling', 'value': 'Baggage handling'},
            {'label': 'Checkin service', 'value': 'Checkin service'},
            {'label': 'Inflight service', 'value': 'Inflight service'},
            {'label': 'Cleanliness', 'value': 'Cleanliness'},
            {'label': 'Departure Delay in Minutes',
             'value': 'Departure Delay in Minutes'},
            {'label': 'Arrival Delay in Minutes',
             'value': 'Arrival Delay in Minutes'},
        ],
        value=' ',
        placeholder='select an option',
        clearable=False,
        style={'color': '#800020', 'width': '200px', "margin-left": "20px"},
    ),
    html.Br(),
    dcc.Graph(id='graphbox1', style={'color': '#800020',
              'width': '800px', 'height': '500px', "margin-left": "20px"}),
    html.Hr(style={'border': '1px solid black'}),
    html.H3('Outlier Removal:IQR method', style={'color': '#800020',
            'margin': '1px', 'textAlign': 'left'}),
    html.Br(),
    html.H3('Outlier Removal: An analysis of numeric variables using boxplot',
style={'color': '#800020', 'margin': '0',

'textAlign': 'left'}),
    html.P('Choose a variables to view the boxplot:',
           style={'margin': '1px', 'textAlign': 'left'}),
    dcc.Dropdown(
        id='drop2',
        options=[
            {'label': 'Age', 'value': 'Age'},
            {'label': 'Flight Distance', 'value': 'Flight Distance'},
            {'label': 'Inflight wifi service',
             'value': 'Inflight wifi service'},
            {'label': 'Departure/Arrival time convenient',
             'value': 'Departure/Arrival time convenient'},
            {'label': 'Ease of Online booking',
             'value': 'Ease of Online booking'},
            {'label': 'Food and drink', 'value': 'Food and drink'},
            {'label': 'Online boarding', 'value': 'Online boarding'},
            {'label': 'Seat comfort', 'value': 'Seat comfort'},
            {'label': 'Inflight entertainment', 'value': 'Inflight entertainment'},
            {'label': 'On-board service', 'value': 'On-board service'},
            {'label': 'Leg room service', 'value': 'Leg room service'},
            {'label': 'Baggage handling', 'value': 'Baggage handling'},
            {'label': 'Checkin service', 'value': 'Checkin service'},
            {'label': 'Inflight service', 'value': 'Inflight service'},
            {'label': 'Cleanliness', 'value': 'Cleanliness'},
            {'label': 'Departure Delay in Minutes', 'value': 'Departure Delay in Minutes'},
            {'label': 'Arrival Delay in Minutes', 'value': 'Arrival Delay in Minutes'},
        ],
        value='',
        placeholder='select an option',
        clearable=False,
        style={'color': '#800020', 'width': '200px', "margin-left": "20px"},
    ),
    html.Br(),
    dcc.Graph(id='graphbox2', style={'width': '800px', 'height': '500px', "margin-left": "20px"}),
    html.P('Therefore the outliers from the following variables were removed: Flight distance, Checkin service ,Departure Delay in Minutes , Arrival Delay in Minutes', style={'backgroundColor': '#800020', 'color': 'white', 'font-size': '20px'}),
    html.Hr(style={'border': '1px solid black'}),
])


# PCA
df = df.drop(['Unnamed: 0', 'id'], axis=1)
df.head()
print(df.isnull().sum())

# Create LabelEncoder object
le = LabelEncoder()

# Encode categorical columns
cat_cols = [
    'Gender',
    'Customer Type',
    'Type of Travel',
    'Class',
    'satisfaction']
for col in cat_cols:
    df[col] = le.fit_transform(df[col])


Features = df._get_numeric_data().columns.to_list()[:-1]
x = df[df._get_numeric_data().columns.to_list()[:-1]]

x = x.values
x = StandardScaler().fit_transform(x)

pca = PCA(n_components='mle', svd_solver='full')
```

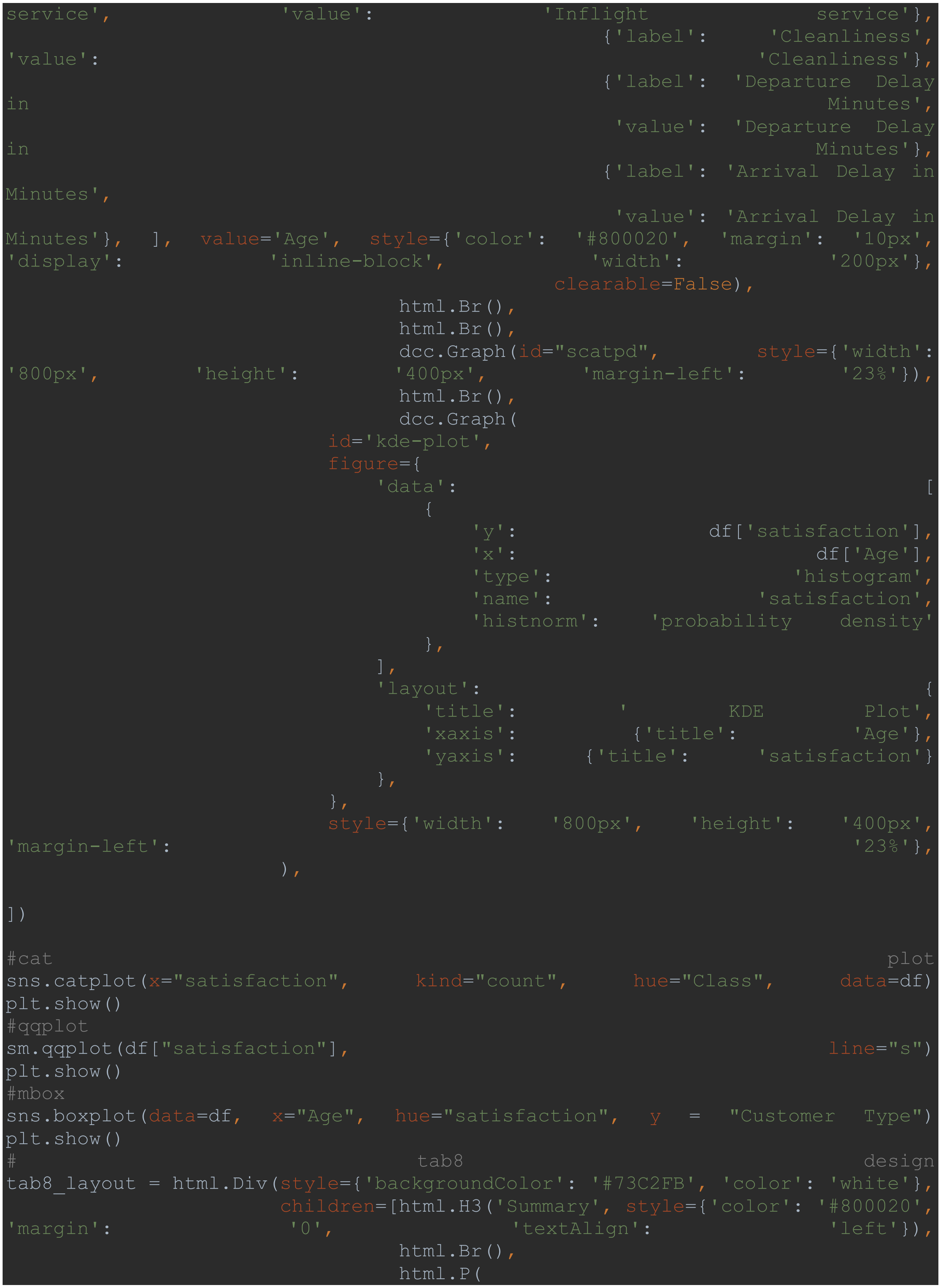
```
pca.fit(x)
x_pca = pca.transform(x)

# plot of cumsum
number_of_components = np.arange(
    1, len(np.cumsum(pca.explained_variance_ratio_)) + 1)
fig = px.line(
    x=number_of_components,
    y=np.cumsum(
        pca.explained_variance_ratio_))
fig.update_layout(title='Cumulative Explained Variance')

# svd and condition number
H = np.matmul(x.T, x)
_, d, _ = np.linalg.svd(H)


# svd and condition number-tranformed
H_pca = np.matmul(x_pca.T, x_pca)
_, d_pca, _ = np.linalg.svd(H_pca)

# PCA correlation matrix
fig1 = px.imshow(pd.DataFrame(x_pca).corr())
# Better visuals
plt.figure(figsize=(20, 20))
sns.heatmap(pd.DataFrame(x_pca).corr(), annot=True)
plt.title('correlation plot of PCA features')
plt.show()


# Design for third tab
tab3_layout = html.Div(style={'backgroundColor': '#73C2FB', 'color': 'white'},
                       children=[html.Br(),
                                 html.H3('Principal Component Analysis',
                                         style={'color': '#800020', 'margin':
'0', 'textAlign': 'left'}),
                                 html.P('Choose options to view outputs of
PCA:',
                                        style={'margin': '1px', 'textAlign':
'left'}),
                                 dcc.RadioItems(id='checkpca', options=[
                                     {'label': 'Original Space', 'value':
'Original'},
                                     {'label': 'Transformed Space', 'value':
'tranformed'}], value='Original', inputStyle={'color': '#800020', "margin-
left": "20px"}),
                                 html.Plaintext(id='pcaout',
style={'backgroundColor': '#800020', 'color': 'white', 'font-size': '15px'}),
                                 html.Hr(style={'border': '1px solid black'}),
                                 html.H3('Cumulative Explained Variance:',
                                         style={'color': '#800020', 'margin':
'0', 'textAlign': 'left'}),
                                 html.Br(),
                                 dcc.Graph(figure=fig, style={'width':
'800px', 'height': '500px'}),
                                 html.Hr(style={'border': '1px solid black'}),
                                 html.H3('PCA features correlation matrix:',
```



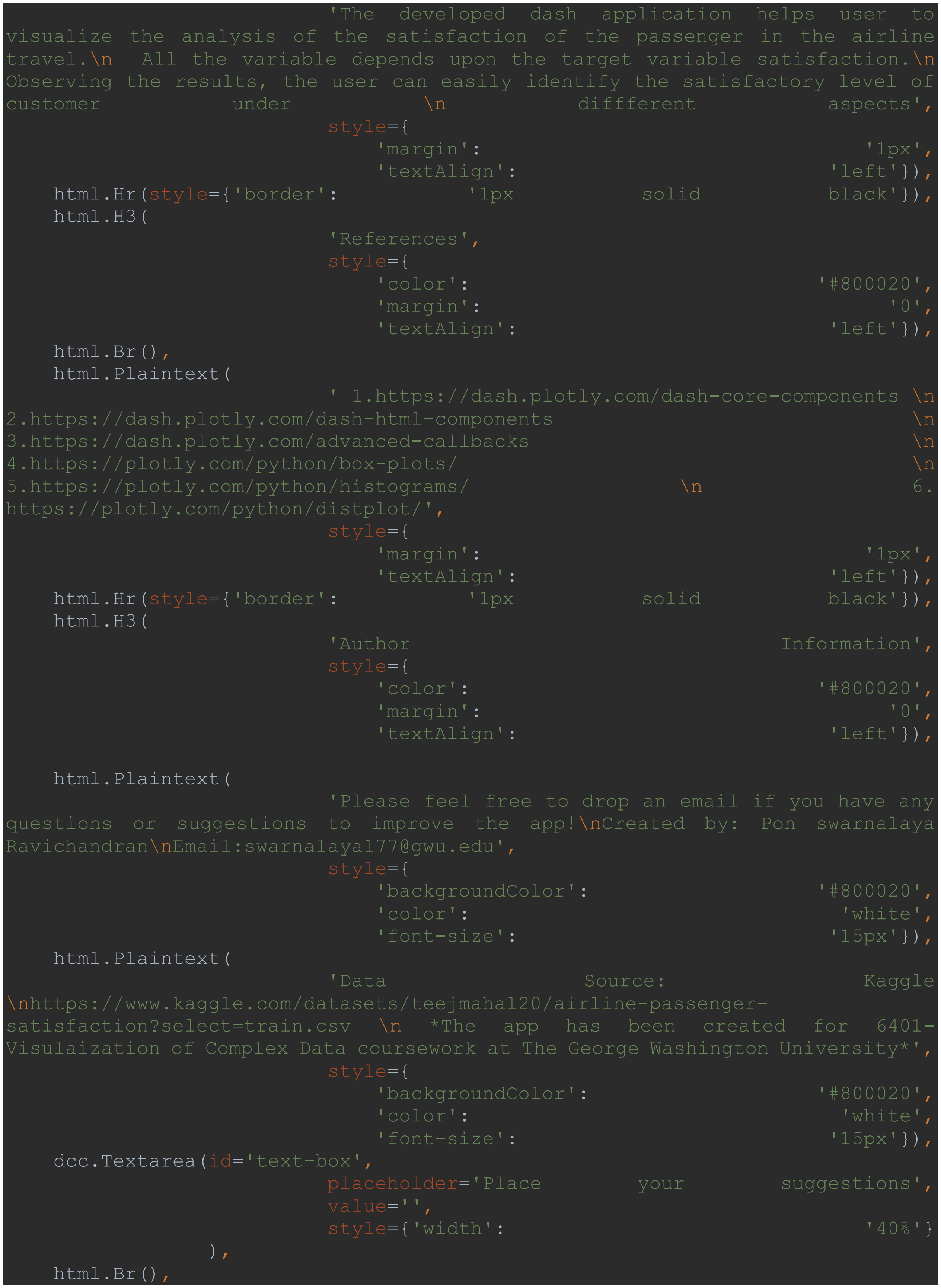
```
                                 style={'color': '#800020', 'margin':
'0', 'textAlign': 'left'}),
                                 html.Br(),
                                 dcc.Graph(figure=fig1, style={'width':
'800px', 'height': '500px'})
                                 ])


# Design for tab4
tab4_layout = html.Div(style={'backgroundColor': '#73C2FB', 'color': 'white'},
                       children=[html.H3('Normality Tests', style={'color':
'#800020', 'margin': '0', 'textAlign': 'center'}),
                                 html.Br(),
                                 html.P('Choose variable:', style={'margin':
'1px', 'textAlign': 'center'}),
                                 html.Br(),
                                 dcc.Dropdown(id='dropvar',
                                              options=[
                                                  {'label': 'Age', 'value':
'Age'},
                                                  {'label': 'Flight Distance',
'value': 'Flight Distance'},
                                                  {'label': 'Inflight wifi
service', 'value': 'Inflight wifi service'},
                                                  {'label': 'Departure/Arrival
time convenient',
                                                      'value':
'Departure/Arrival time convenient'},
                                                  {'label': 'Ease of Online
booking', 'value': 'Ease of Online booking'},
                                                  {'label': 'Food and drink',
'value': 'Food and drink'},
                                                  {'label': 'Online boarding',
'value': 'Online boarding'},
                                                  {'label': 'Seat comfort',
'value': 'Seat comfort'},
                                                  {'label': 'Inflight
entertainment', 'value': 'Inflight entertainment'},
                                                  {'label': 'On-board
service', 'value': 'On-board service'},
                                                  {'label': 'Leg room
service', 'value': 'Leg room service'},
                                                  {'label': 'Baggage
handling', 'value': 'Baggage handling'},
                                                  {'label': 'Checkin service',
'value': 'Checkin service'},
                                                  {'label': 'Inflight
service', 'value': 'Inflight service'},
                                                  {'label': 'Cleanliness',
'value': 'Cleanliness'},
                                                  {'label': 'Departure Delay
in Minutes', 'value': 'Departure Delay in Minutes'},
                                                  {'label': 'Arrival Delay in
Minutes', 'value': 'Arrival Delay in Minutes'},], value=' ', style={'color':
'#800020', 'width': '200px', 'margin': '0 auto', 'textAlign': 'center'},
clearable=False),
                                 html.Br(),
                                 html.P('Choose the test', style={'margin':
'40px', 'textAlign': 'center'}),
                                 html.Br(),
                                 dcc.Dropdown(id='droptest', options=[
                                     {'label': 'normaltest', 'value': 'normal-
test'},
                                     {'label': 'kstest', 'value': 'kstest'},
                                     {'label': 'shapiro', 'value': 'shapiro'}
                                 ], value='normaltest', style={'color':
'#800020', 'width': '200px', 'margin': '0 auto', 'textAlign': 'center'}),
                                 html.Br(),
                                 html.Plaintext(id='ntout',
style={'backgroundColor': '#800020', 'color': 'white', 'font-size': '15px'}),
                                 html.Hr(style={'border': '1px solid black'}),

                                 ])

# Design tab5 layout
tab5_layout = html.Div(
    style={
        'backgroundColor': '#73C2FB', 'color': 'white'}, children=[
            dcc.Dropdown(
                id='drop_down', options=[
                    {
                        'label': 'Heatmap', 'value': 'Heatmap'}, {
                            'label': 'Scatter matrix', 'value': 'Scatter
matrix'}, ], value='Heatmap', style={
                                'color': '#800020', 'width': '200px',
'display': 'inline-block'}), html.H3(
                                    'Heat map(pearson correlation coeeficient
& Scatter matrix', style={
                                        'color': '#800020', 'margin': '0',
'textAlign': 'center'}), html.Br(), dcc.Graph(
                                            id='hs', style={
                                                'width': '1200px', 'height':
'800px', 'margin-left': '10%', }), ])

# Design for tab6
tab6_layout = html.Div(style={'backgroundColor': '#73C2FB', 'color': 'white'},
                       children=[html.H3('Visualize data using various plots',
style={'color': '#800020', 'margin': '0', 'textAlign': 'center'}),
                                 html.Br(),
                                 html.P('Choose variable:', style={'margin':
'1px', 'textAlign': 'center', 'display': 'inline-block'}),
                                 html.Br(),
                                 dcc.Dropdown(id='options_dropdown',
                                              options=[
                                                  {'label': 'Age', 'value':
'Age'},
                                                  {'label': 'Flight Distance',
'value': 'Flight Distance'},
                                                  {'label': 'Inflight wifi
service', 'value': 'Inflight wifi service'},
                                                  {'label': 'Departure/Arrival
time convenient', 'value': 'Departure/Arrival time convenient'},
                                                  {'label': 'Ease of Online
booking', 'value': 'Ease of Online booking'},
                                                  {'label': 'Food and drink', 'value': 'Food and drink'},
                                                  {'label': 'Online boarding', 'value': 'Online boarding'},
                                                  {'label': 'Seat comfort', 'value': 'Seat comfort'},
                                                  {'label': 'Inflight entertainment', 'value': 'Inflight entertainment'},
                                                  {'label': 'On-board service', 'value': 'On-board service'},
                                                  {'label': 'Leg room service', 'value': 'Leg room service'},
                                                  {'label': 'Baggage handling', 'value': 'Baggage handling'},
                                                  {'label': 'Checkin service', 'value': 'Checkin service'},
                                                  {'label': 'Inflight service', 'value': 'Inflight service'},
                                                  {'label': 'Cleanliness', 'value': 'Cleanliness'},
                                                  {'label': 'Departure Delay in Minutes', 'value': 'Departure Delay in Minutes'},
                                                  {'label': 'Arrival Delay in Minutes', 'value': 'Arrival Delay in Minutes'},], style={'color': '#800020', 'width': '200px', 'display': 'inline-block'}, clearable=False),
                                 html.P('Choose variable to color:', style={'display': 'inline-block', 'margin': '1px', 'textAlign': 'left'}),
                                 dcc.Dropdown(id='color',
                                              options=[
                                                  {'label': 'Gender', 'value': 'Gender'},
                                                  {'label': 'Customer Type', 'value': 'Customer Type'},
                                                  {'label': 'Type of Travel', 'value': 'Type of Travel'},
                                                  {'label': 'Class', 'value': 'Class'},], value='Gender', style={'color': '#800020', 'display': 'inline-block', 'width': '200px'}, clearable=False),
                                 html.Br(),

                                 html.P('Line Plot:', style={'margin': '1px', 'textAlign': 'center'}),
                                 dcc.Graph(id='line', style={'width': '800px', 'height': '400px', 'margin': '0 auto'}),
                                 html.Br(),
                                 html.P('COUNT PLOT:', style={'margin': '0 auto', 'textAlign': 'center'}),
                                 dcc.Slider(id='bins', min=20, max=100, value=50, tooltip={"placement": "bottom", "always_visible": True}),
                                 html.Br(),
                                 dcc.Graph(id='bar', style={'width': '800px', 'height': '400px', 'margin': '0 auto'}),
                                 html.P('HISTO (WITH VIOLIN&BOX):', style={'textAlign': 'center', 'margin': '0 auto'}),
                                 html.P("Select Distribution:", style={'margin': '1px', 'textAlign': 'left'}),
                                 dcc.RadioItems(
                             id='distribution',
                             options=[
                                 {'label': 'box', 'value': 'box'},
                                 {'label': 'violin', 'value': 'violin'},
                             ],

                             value='box', inputStyle={'color': '#800020',
"margin-left": "20px"}),
    dcc.Graph(id="graphd", style={'width': '800px', 'height': '400px',
'margin': '0 auto'}),
])

# Design for Tab7
tab7_layout = html.Div(style={'backgroundColor': '#73C2FB', 'color': 'white'},
                         children=[html.H3('PIE PLOT', style={'color':
'#800020', 'fontWeight': 'bold', 'margin': '1px', 'textAlign': 'center'}),
                                   html.Br(),
                                   html.P('Choose variable:', style={'margin':
'0', 'textAlign': 'left', 'display': 'inline-block'}),
dcc.Dropdown(id='pie_drop',

options=[

{'label': 'Gender', 'value': 'Gender'},

{'label': 'Customer Type', 'value': 'Customer Type'},

{'label': 'Type of Travel', 'value': 'Type of Travel'},

{'label': 'Class', 'value': 'Class'}, ], value='Gender', style={'color':
'#800020', 'display': 'inline-block', 'width': '200px'},

clearable=False),
                                   html.Br(),
                                   dcc.Graph(id='pie', style={'width': '800px',
'height': '400px', 'margin-left': '23%'}),
                                   html.Br(),
                                   html.H3('SCATTER PLOT:', style={'color':
'#800020', 'fontWeight': 'bold', 'margin': '1px', 'textAlign': 'center'}),
                                   html.Br(),
                                   html.P('Choose variable:', style={'margin':
'1px', 'textAlign': 'left', 'display': 'inline-block'}),
                                   dcc.Dropdown(id='scat_drop',
                                                options=[
                                                    {'label': 'Age', 'value':
'Age'},
                                                    {'label': 'Flight Distance',
'value': 'Flight Distance'},
                                                    {'label': 'Inflight wifi
service', 'value': 'Inflight wifi service'},
                                                    {'label': 'Departure/Arrival
time convenient', 'value': 'Departure/Arrival time convenient'},
                                                    {'label': 'Ease of Online
booking', 'value': 'Ease of Online booking'},
                                                    {'label': 'Food and drink',
'value': 'Food and drink'},
                                                    {'label': 'Online boarding',
```



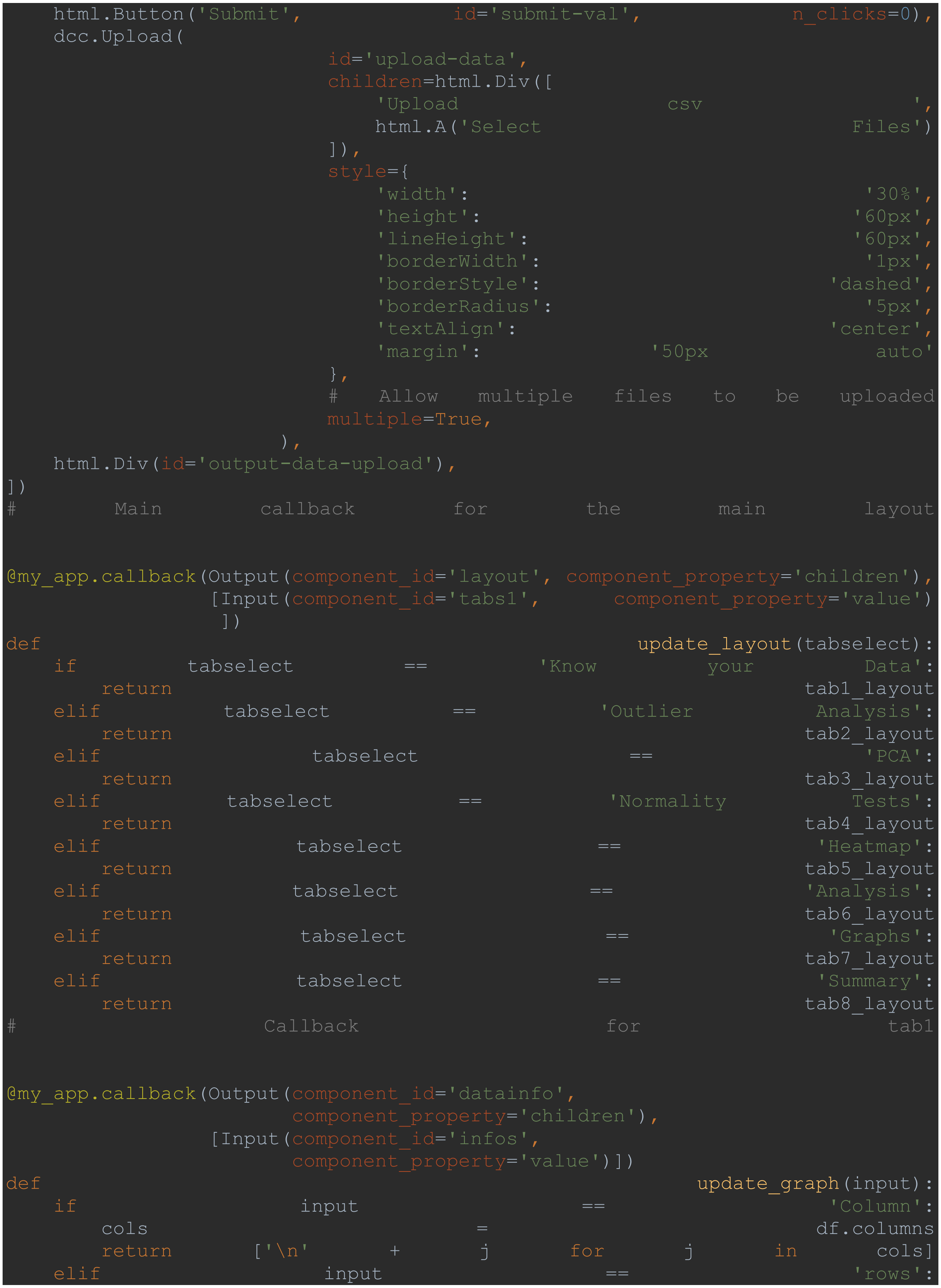
```
'value': 'Online boarding'},
                                                              {'label': 'Seat comfort',
'value': 'Seat comfort'},
                                                              {'label': 'Inflight
entertainment', 'value': 'Inflight entertainment'},
                                                              {'label': 'On-board
service', 'value': 'On-board service'},
                                                              {'label': 'Leg room
service', 'value': 'Leg room service'},
                                                              {'label': 'Baggage
handling', 'value': 'Baggage handling'},
                                                              {'label': 'Checkin service',
'value': 'Checkin service'},
                                                              {'label': 'Inflight
service', 'value': 'Inflight service'},
                                                              {'label': 'Cleanliness',
'value': 'Cleanliness'},
                                                              {'label': 'Departure Delay
in Minutes', 'value': 'Departure Delay in Minutes'},
                                                              {'label': 'Arrival Delay in
Minutes', 'value': 'Arrival Delay in Minutes'}],
                                                         value='Age', style={'color':
'#800020', 'margin': '10px', 'display': 'inline-block', 'width': '200px'},
clearable=False),
                                         html.P('Choose variable to color:',
style={'display': 'inline-block', 'margin': '1px', 'textAlign': 'left'}),
                                         dcc.Dropdown(id='color',
                                                         options=[
                                                              {'label': 'Age', 'value':
'Age'},
                                                              {'label': 'Flight Distance',
'value': 'Flight Distance'},
                                                              {'label': 'Inflight wifi
service', 'value': 'Inflight wifi service'},
                                                              {'label': 'Departure/Arrival
time convenient',
                                                               'value': 'Departure/Arrival
time convenient'},
                                                              {'label': 'Ease of Online
booking', 'value': 'Ease of Online booking'},
                                                              {'label': 'Food and drink',
'value': 'Food and drink'},
                                                              {'label': 'Online boarding',
'value': 'Online boarding'},
                                                              {'label': 'Seat comfort',
'value': 'Seat comfort'},
                                                              {'label': 'Inflight
entertainment', 'value': 'Inflight entertainment'},
                                                              {'label': 'On-board
service', 'value': 'On-board service'},
                                                              {'label': 'Leg room
service', 'value': 'Leg room service'},
                                                              {'label': 'Baggage
handling', 'value': 'Baggage handling'},
                                                              {'label': 'Checkin service',
'value': 'Checkin service'},
                                                              {'label': 'Inflight
```



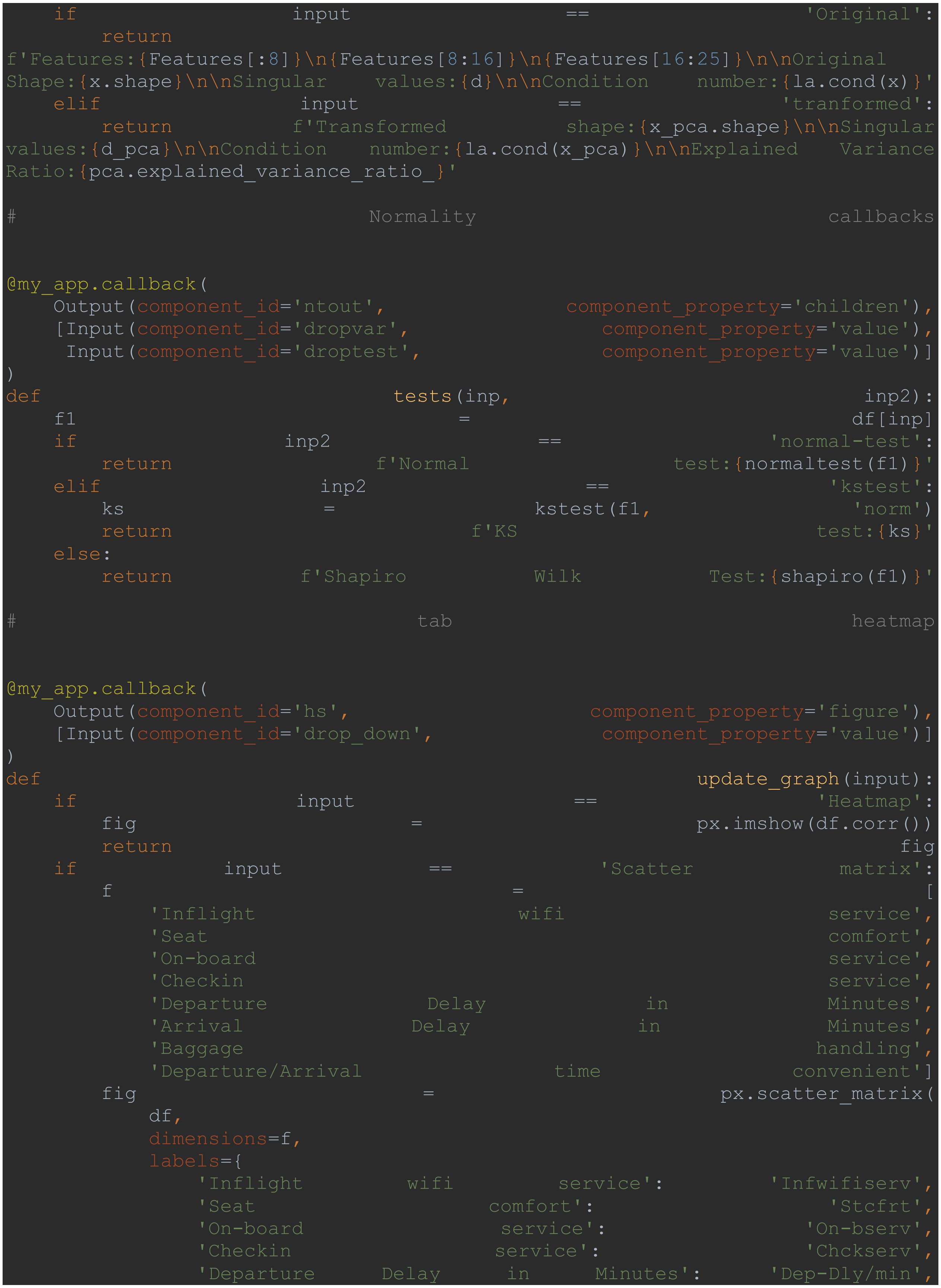
```
service',                'value':                'Inflight                service'},
                                                  {'label':    'Cleanliness',
'value':                                                       'Cleanliness'},
                                                  {'label':  'Departure  Delay
in                                                                    Minutes',
                                                   'value':  'Departure  Delay
in                                                                   Minutes'},
                                                  {'label': 'Arrival Delay in
Minutes',
                                                   'value': 'Arrival Delay in
Minutes'},   ],  value='Age',  style={'color':  '#800020',  'margin':  '10px',
'display':          'inline-block',          'width':                '200px'},
                                              clearable=False),
                                 html.Br(),
                                 html.Br(),
                                 dcc.Graph(id="scatpd",          style={'width':
'800px',        'height':        '400px',        'margin-left':        '23%'}),
                                 html.Br(),
                                 dcc.Graph(
                           id='kde-plot',
                           figure={
                               'data':                                        [
                                   {
                                       'y':                 df['satisfaction'],
                                       'x':                          df['Age'],
                                       'type':                     'histogram',
                                       'name':                  'satisfaction',
                                       'histnorm':    'probability    density'
                                   },
                               ],
                               'layout':                                      {
                                   'title':        '         KDE         Plot',
                                   'xaxis':          {'title':         'Age'},
                                   'yaxis':      {'title':      'satisfaction'}
                               },
                           },
                           style={'width':    '800px',    'height':    '400px',
'margin-left':                                                         '23%'},
                       ),

])

#cat                                                                      plot
sns.catplot(x="satisfaction",     kind="count",     hue="Class",     data=df)
plt.show()
#qqplot
sm.qqplot(df["satisfaction"],                                          line="s")
plt.show()
#mbox
sns.boxplot(data=df,  x="Age",  hue="satisfaction",  y  =  "Customer  Type")
plt.show()
#                               tab8                                    design
tab8_layout = html.Div(style={'backgroundColor': '#73C2FB', 'color': 'white'},
                       children=[html.H3('Summary', style={'color': '#800020',
'margin':            '0',            'textAlign':                    'left'}),
                                 html.Br(),
                                 html.P(
```



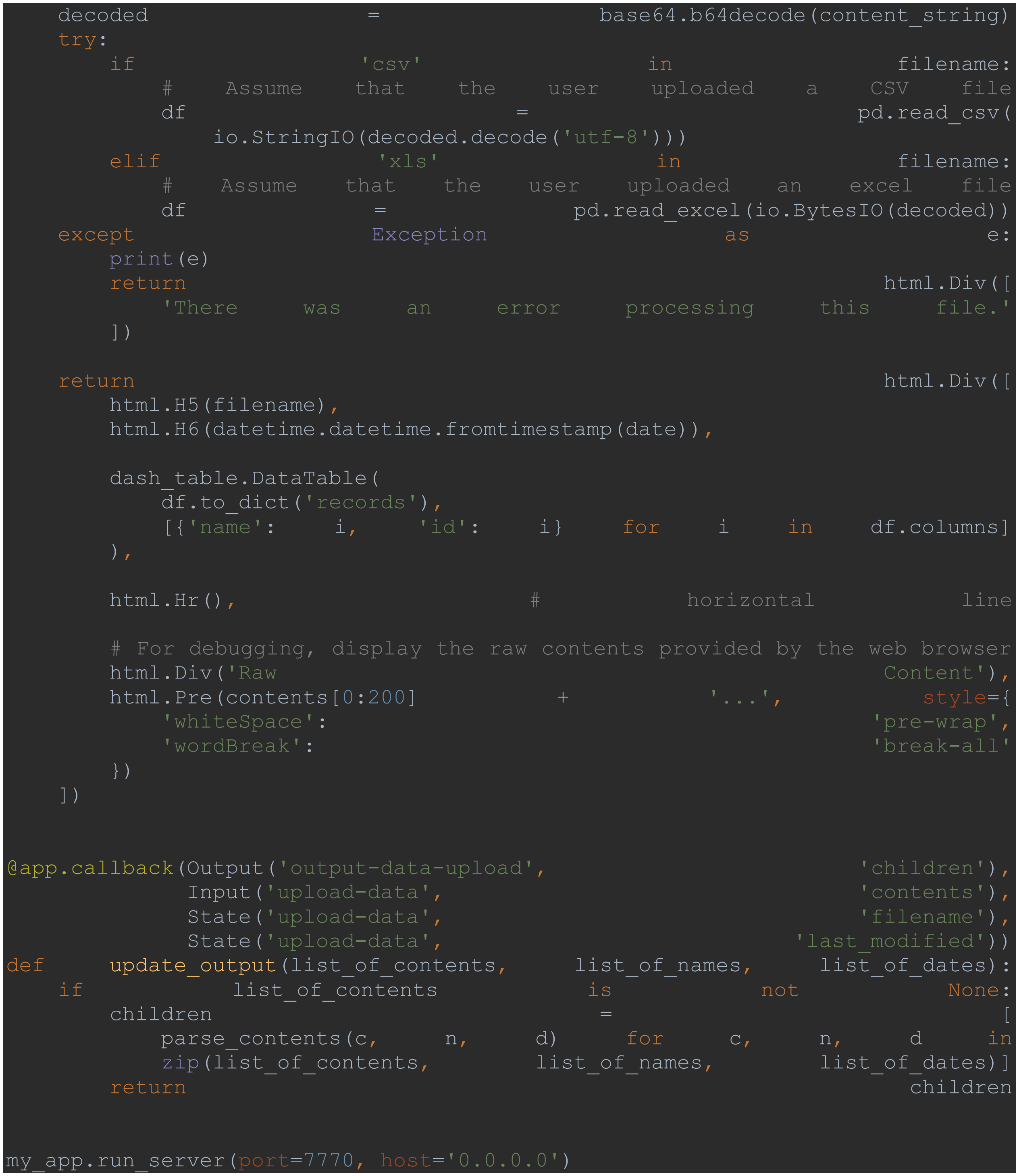
```
                           'The developed dash application helps user to
visualize the analysis of the satisfaction of the passenger in the airline
travel.\n  All the variable depends upon the target variable satisfaction.\n
Observing the results, the user can easily identify the satisfactory level of
customer under \n diffferent aspects',
                           style={
                               'margin': '1px',
                               'textAlign': 'left'}),
    html.Hr(style={'border': '1px solid black'}),
    html.H3(
                           'References',
                           style={
                               'color': '#800020',
                               'margin': '0',
                               'textAlign': 'left'}),
    html.Br(),
    html.Plaintext(
                           ' 1.https://dash.plotly.com/dash-core-components \n
2.https://dash.plotly.com/dash-html-components \n
3.https://dash.plotly.com/advanced-callbacks \n
4.https://plotly.com/python/box-plots/ \n
5.https://plotly.com/python/histograms/ \n 6.
https://plotly.com/python/distplot/',
                           style={
                               'margin': '1px',
                               'textAlign': 'left'}),
    html.Hr(style={'border': '1px solid black'}),
    html.H3(
                           'Author Information',
                           style={
                               'color': '#800020',
                               'margin': '0',
                               'textAlign': 'left'}),

    html.Plaintext(
                           'Please feel free to drop an email if you have any
questions or suggestions to improve the app!\nCreated by: Pon swarnalaya
Ravichandran\nEmail:swarnalaya177@gwu.edu',
                           style={
                               'backgroundColor': '#800020',
                               'color': 'white',
                               'font-size': '15px'}),
    html.Plaintext(
                           'Data Source: Kaggle
\nhttps://www.kaggle.com/datasets/teejmahal20/airline-passenger-
satisfaction?select=train.csv \n *The app has been created for 6401-
Visulaization of Complex Data coursework at The George Washington University*',
                           style={
                               'backgroundColor': '#800020',
                               'color': 'white',
                               'font-size': '15px'}),
    dcc.Textarea(id='text-box',
                           placeholder='Place your suggestions',
                           value='',
                           style={'width': '40%'}
                 ),
    html.Br(),
```



```
    html.Button('Submit', id='submit-val', n_clicks=0),
    dcc.Upload(
                           id='upload-data',
                           children=html.Div([
                               'Upload csv ',
                               html.A('Select Files')
                           ]),
                           style={
                               'width': '30%',
                               'height': '60px',
                               'lineHeight': '60px',
                               'borderWidth': '1px',
                               'borderStyle': 'dashed',
                               'borderRadius': '5px',
                               'textAlign': 'center',
                               'margin': '50px auto'
                           },
                           # Allow multiple files to be uploaded
                           multiple=True,
                       ),
    html.Div(id='output-data-upload'),
])
# Main callback for the main layout


@my_app.callback(Output(component_id='layout', component_property='children'),
                 [Input(component_id='tabs1', component_property='value')
                  ])
def update_layout(tabselect):
    if tabselect == 'Know your Data':
        return tab1_layout
    elif tabselect == 'Outlier Analysis':
        return tab2_layout
    elif tabselect == 'PCA':
        return tab3_layout
    elif tabselect == 'Normality Tests':
        return tab4_layout
    elif tabselect == 'Heatmap':
        return tab5_layout
    elif tabselect == 'Analysis':
        return tab6_layout
    elif tabselect == 'Graphs':
        return tab7_layout
    elif tabselect == 'Summary':
        return tab8_layout
# Callback for tab1


@my_app.callback(Output(component_id='datainfo',
                        component_property='children'),
                 [Input(component_id='infos',
                        component_property='value')])
def update_graph(input):
    if input == 'Column':
        cols = df.columns
        return ['\n' + j for j in cols]
    elif input == 'rows':
        i = len(df)
        return f'Number of rows:{i}'
    elif input == 'columns':
        cols = df.columns
        return f'Number of columns:{len(cols)}'


@my_app.callback(Output(component_id='preprocess',
                        component_property='children'),
                 [Input(component_id='cleans',
                        component_property='value')])
def update_graph(input):
    if input == 'nulls':
        df.isnull().sum()
        df.dropna(inplace=True)
        d = df.isnull().sum()
        return f"{d}\nDataset doesn't have missing values"
    if input == 'stats':
        return f'{df.describe()}'
    elif input == 'head_d':
        return f'{df.head()}'


@my_app.callback(
    Output("download-dataframe-csv", "data"),
    Input("btn_csv", "n_clicks"),
    prevent_initial_call=True,
)
def func(n_clicks):
    return dcc.send_data_frame(df.to_csv, "train .csv")

# Callbacks fro tab2 components


@my_app.callback(Output(component_id='graphbox1',
component_property='figure'),
                 [Input(component_id='drop1', component_property='value')])
def update_graph(input):
    fig = px.box(df, y=input)
    fig.update_layout(title='Box plot')
    return fig


@my_app.callback(Output(component_id='graphbox2',
component_property='figure'),
                 [Input(component_id='drop2', component_property='value')])
def update_graph(input):
    fig = px.box(df1, y=input)
    fig.update_layout(title='Box plot')
    return fig

# PCA callbacks


@my_app.callback(Output(component_id='pcaout', component_property='children'),
                 [Input(component_id='checkpca', component_property='value')])
def update_graph(input):
```



```
    if input == 'Original':
        return
f'Features:{Features[:8]}\n{Features[8:16]}\n{Features[16:25]}\n\nOriginal
Shape:{x.shape}\n\nSingular values:{d}\n\nCondition number:{la.cond(x)}'
    elif input == 'tranformed':
        return f'Transformed shape:{x_pca.shape}\n\nSingular
values:{d_pca}\n\nCondition number:{la.cond(x_pca)}\n\nExplained Variance
Ratio:{pca.explained_variance_ratio_}'

# Normality callbacks


@my_app.callback(
    Output(component_id='ntout', component_property='children'),
    [Input(component_id='dropvar', component_property='value'),
     Input(component_id='droptest', component_property='value')]
)
def tests(inp, inp2):
    f1 = df[inp]
    if inp2 == 'normal-test':
        return f'Normal test:{normaltest(f1)}'
    elif inp2 == 'kstest':
        ks = kstest(f1, 'norm')
        return f'KS test:{ks}'
    else:
        return f'Shapiro Wilk Test:{shapiro(f1)}'

# tab heatmap


@my_app.callback(
    Output(component_id='hs', component_property='figure'),
    [Input(component_id='drop_down', component_property='value')]
)
def update_graph(input):
    if input == 'Heatmap':
        fig = px.imshow(df.corr())
        return fig
    if input == 'Scatter matrix':
        f = [
            'Inflight wifi service',
            'Seat comfort',
            'On-board service',
            'Checkin service',
            'Departure Delay in Minutes',
            'Arrival Delay in Minutes',
            'Baggage handling',
            'Departure/Arrival time convenient']
        fig = px.scatter_matrix(
            df,
            dimensions=f,
            labels={
                'Inflight wifi service': 'Infwifiserv',
                'Seat comfort': 'Stcfrt',
                'On-board service': 'On-bserv',
                'Checkin service': 'Chckserv',
                'Departure Delay in Minutes': 'Dep-Dly/min',
                'Arrival Delay in Minutes': 'Arr-Dly/min',
                'Baggage handling': 'Bagsserv',
                'Departure/Arrival time convenient': 'Dep/ArrTime'})
        return fig


# Analysis callbacks
@my_app.callback(Output(component_id='line', component_property='figure'),
                 Output(component_id='bar', component_property='figure'),
                 Output(component_id='graphd', component_property='figure'),
                 [Input(component_id='options_dropdown',
component_property='value'),
                  Input(component_id='color', component_property='value'),
                  Input(component_id='bins', component_property='value'),
                  Input(component_id='distribution',
component_property='value')])
def update_graph(input, inp2, inp3, inp4):
    fig = px.line(df, x=input, y='satisfaction', color=inp2) # line
    fig1 = px.histogram(df, x=input, nbins=inp3) # count
    fig3 = px.histogram(
        df,
        x=input,
        y='satisfaction',
        color=inp2,
        marginal=inp4) # histo
    return fig, fig1, fig3

# graphs callbacks


@my_app.callback(Output(component_id='pie', component_property='figure'),
                 Output(component_id='scatpd', component_property='figure'),
                 [Input(component_id='pie_drop', component_property='value'),
                  Input(component_id='color', component_property='value'),
                  Input(component_id='scat_drop',
component_property='value')])
def update_graph(inp, inp2, inp3):
    pie1 = px.pie(df, names=inp, values='satisfaction')
    scat1 = px.scatter(df, x=inp2, color=inp3, trendline='ols')
    return pie1, scat1
# summary


@app.callback(
    Output('container-button-basic', 'children'),
    Input('submit-val', 'n_clicks'),
    prevent_initial_call=True,
)
def update_output(n_clicks, value):
    return 'The input value was "{}" and the button has been clicked {}
times'.format(
        value, n_clicks)


def parse_contents(contents, filename, date):
    content_type, content_string = contents.split(',')
```



```
    decoded = base64.b64decode(content_string)
    try:
        if 'csv' in filename:
            # Assume that the user uploaded a CSV file
            df = pd.read_csv(
                io.StringIO(decoded.decode('utf-8')))
        elif 'xls' in filename:
            # Assume that the user uploaded an excel file
            df = pd.read_excel(io.BytesIO(decoded))
    except Exception as e:
        print(e)
        return html.Div([
            'There was an error processing this file.'
        ])

    return html.Div([
        html.H5(filename),
        html.H6(datetime.datetime.fromtimestamp(date)),

        dash_table.DataTable(
            df.to_dict('records'),
            [{'name': i, 'id': i} for i in df.columns]
        ),

        html.Hr(), # horizontal line

        # For debugging, display the raw contents provided by the web browser
        html.Div('Raw Content'),
        html.Pre(contents[0:200] + '...', style={
            'whiteSpace': 'pre-wrap',
            'wordBreak': 'break-all'
        })
    ])


@app.callback(Output('output-data-upload', 'children'),
              Input('upload-data', 'contents'),
              State('upload-data', 'filename'),
              State('upload-data', 'last_modified'))
def update_output(list_of_contents, list_of_names, list_of_dates):
    if list_of_contents is not None:
        children = [
            parse_contents(c, n, d) for c, n, d in
            zip(list_of_contents, list_of_names, list_of_dates)]
        return children


my_app.run_server(port=7770, host='0.0.0.0')
```

REFERENCES:

- https://dash.plotly.com/dash-core-components
- https://dash.plotly.com/dash-html-components
- https://dash.plotly.com/advanced-callbacks
- https://plotly.com/python/box-plots/
- https://plotly.com/python/histograms/
- https://plotly.com/python/distplot/
- All the lecture notes, inclasscodes, homeworks, labs.
- https://medium.com/@chris.bacani7/explaining-airline-passenger-satisfaction-using-interpretable-machine-learning-88d29aa55677
- https://www.kaggle.com/datasets/teejmahal20/airline-passenger-satisfaction/code?select=train.csv