

import piplite

await piplite.install(['nbformat', 'plotly'])

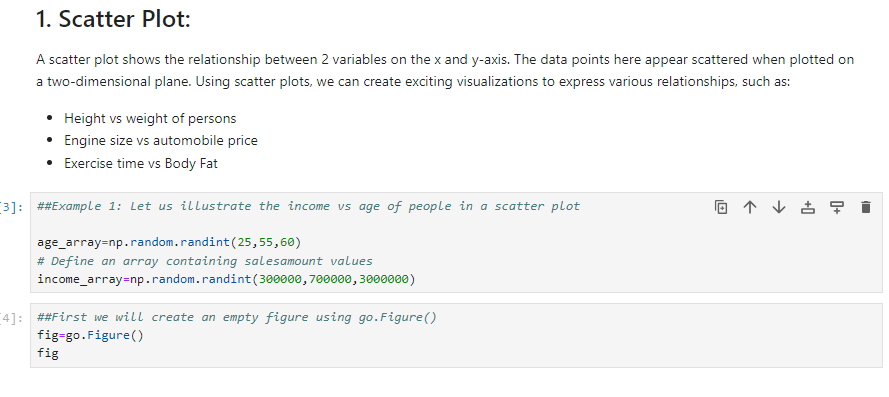
# Import required libraries

import pandas as pd

import numpy as np

import plotly.express as px

import plotly.graph\_objects as go



##Example 1: Let us illustrate the income vs age of people in a scatter plot

age\_array=np.random.randint(25,55,60)

# Define an array containing salesamount values

income\_array=np.random.randint(300000,700000,3000000)

##First we will create an empty figure using go.Figure()

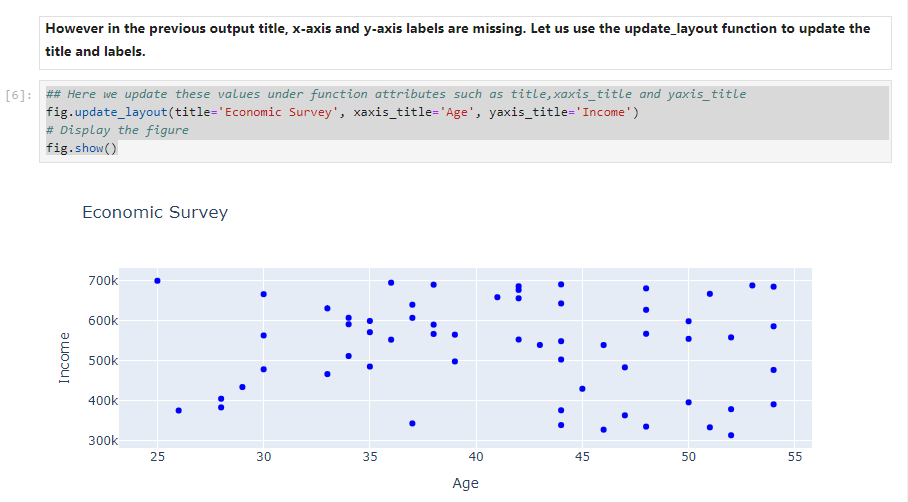
fig=go.Figure()

Fig

#Next we will create a scatter plot by using the add\_trace function and use the go.scatter() function within it

# In go.Scatter we define the x-axis data,y-axis data and define the mode as markers with color of the marker as blue

fig.add\_trace(go.Scatter(x=age\_array, y=income\_array, mode='markers', marker=dict(color='blue')))



****However in the previous output title, x-axis and y-axis labels are missing. Let us use the update\_layout function to update the title and labels.****

**## Here we update these values under function attributes such as title,xaxis\_title and yaxis\_title**

**fig.update\_layout(title='Economic Survey', xaxis\_title='Age', yaxis\_title='Income')**

**# Display the figure**

**fig.show()**



##Example 2: Let us illustrate the sales of bicycles from Jan to August last year using a line chart

# Define an array containing numberofbicyclessold

numberofbicyclessold\_array=[50,100,40,150,160,70,60,45]

# Define an array containing months

months\_array=["Jan","Feb","Mar","April","May","June","July","August"]

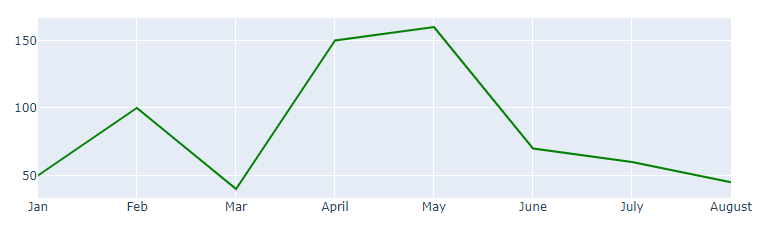
##First we will create an empty figure using go.Figure()

fig=go.Figure()

#Next we will create a line plot by using the add\_trace function and use the go.scatter() function within it

# In go.Scatter we define the x-axis data,y-axis data and define the mode as lines with color of the marker as green

fig.add\_trace(go.Scatter(x=months\_array, y=numberofbicyclessold\_array, mode='lines', marker=dict(color='green')))



## Here we update these values under function attributes such as title,xaxis\_title and yaxis\_title

fig.update\_layout(title='Bicycle Sales', xaxis\_title='Months', yaxis\_title='Number of Bicycles Sold')

# Display the figure

fig.show()





##Example 3: Let us illustrate the average pass percentage of classes from grade 6 to grade 10

# Define an array containing scores of students

score\_array=[80,90,56,88,95]

# Define an array containing Grade names

grade\_array=['Grade 6','Grade 7','Grade 8','Grade 9','Grade 10']

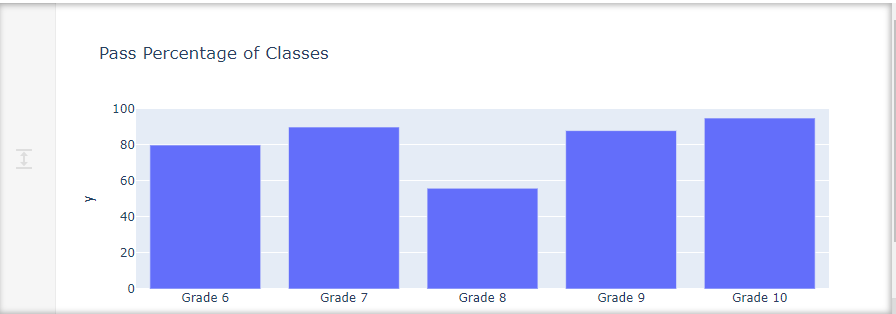
****In plotly express we set the axis values and the title within the same function call**px.<graphtype>(x=<xaxis value source>,y=<y-axis value source>,title=<appropriate title as a string>)**.In the below code we use**px.bar( x=grade\_array, y=score\_array, title='Pass Percentage of Classes')**.****

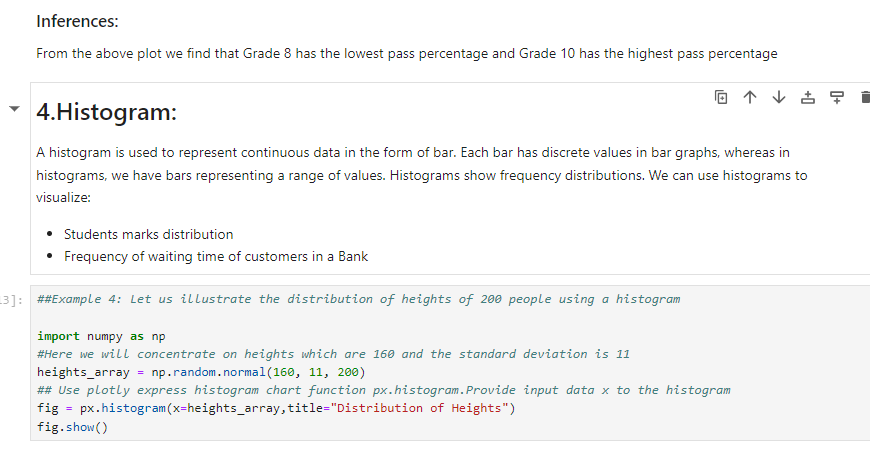
**# Use plotly express bar chart function px.bar.Provide input data, x and y axis variable, and title of the chart.**

**# This will give average pass percentage per class**

**fig = px.bar( x=grade\_array, y=score\_array, title='Pass Percentage of Classes')**

**fig.show()**





##Example 4: Let us illustrate the distribution of heights of 200 people using a histogram

import numpy as np

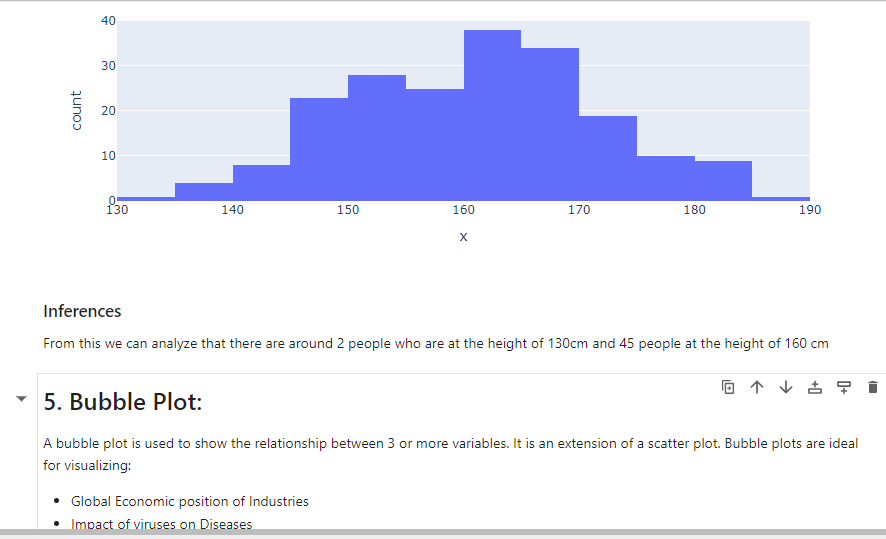
#Here we will concentrate on heights which are 160 and the standard deviation is 11

heights\_array = np.random.normal(160, 11, 200)

## Use plotly express histogram chart function px.histogram.Provide input data x to the histogram

fig = px.histogram(x=heights\_array,title="Distribution of Heights")

fig.show()



##Example 4: Let us illustrate crime statistics of US cities with a bubble chart

#Create a dictionary having city,numberofcrimes and year as 3 keys

crime\_details = {

'City' : ['Chicago', 'Chicago', 'Austin', 'Austin','Seattle','Seattle'],

'Numberofcrimes' : [1000, 1200, 400, 700,350,1500],

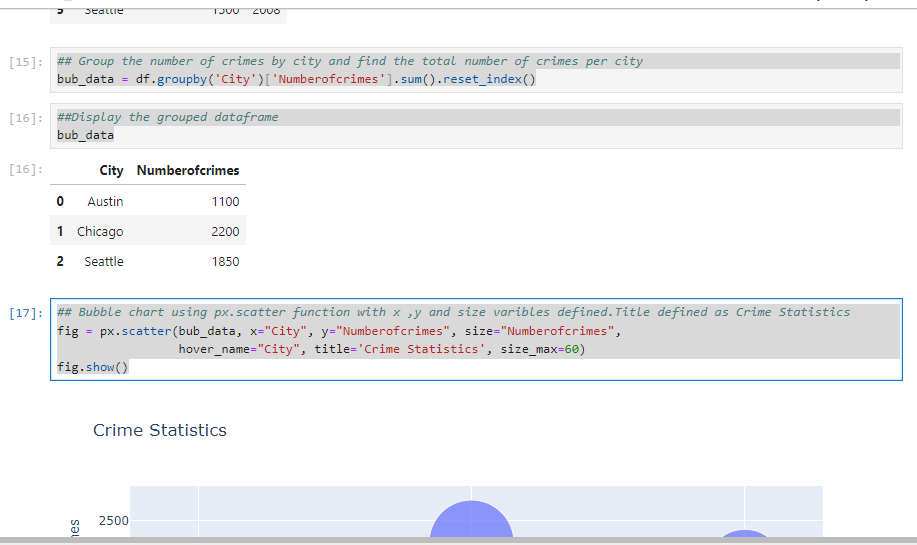
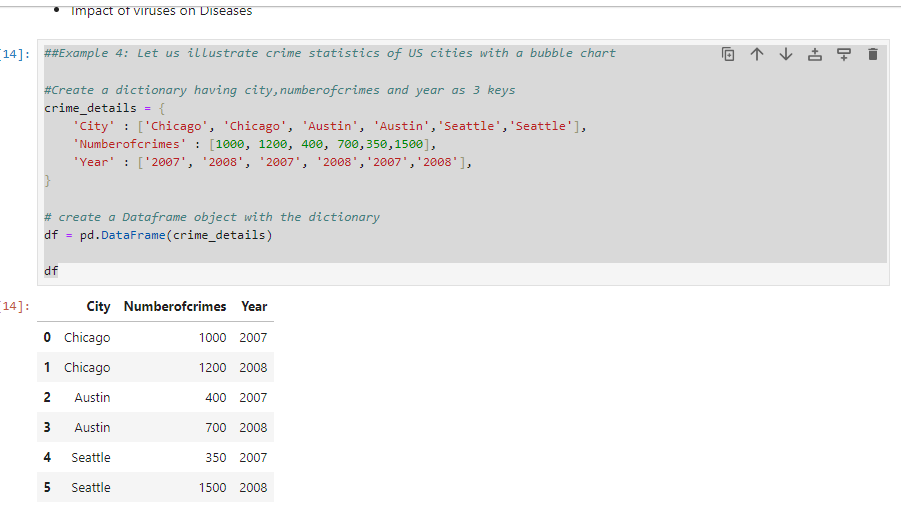
'Year' : ['2007', '2008', '2007', '2008','2007','2008'],

}

# create a Dataframe object with the dictionary

df = pd.DataFrame(crime\_details)

Df



## Group the number of crimes by city and find the total number of crimes per city

bub\_data = df.groupby('City')['Numberofcrimes'].sum().reset\_index()

##Display the grouped dataframe

bub\_data

## Bubble chart using px.scatter function with x ,y and size varibles defined.Title defined as Crime Statistics

fig = px.scatter(bub\_data, x="City", y="Numberofcrimes", size="Numberofcrimes",

hover\_name="City", title='Crime Statistics', size\_max=60)

fig.show()

## Monthly expenditure of a family

# Random Data

exp\_percent= [20, 50, 10,8,12]

house\_holdcategories = ['Grocery', 'Rent', 'School Fees','Transport','Savings']

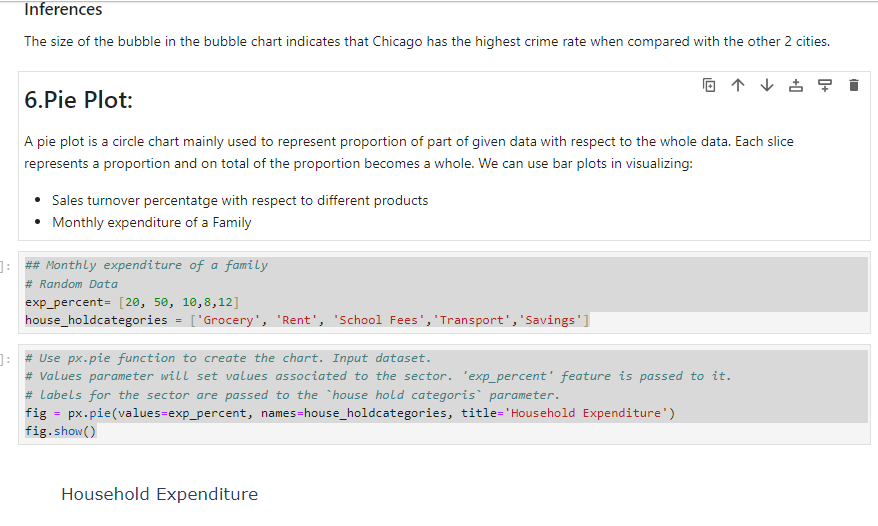
# Use px.pie function to create the chart. Input dataset.

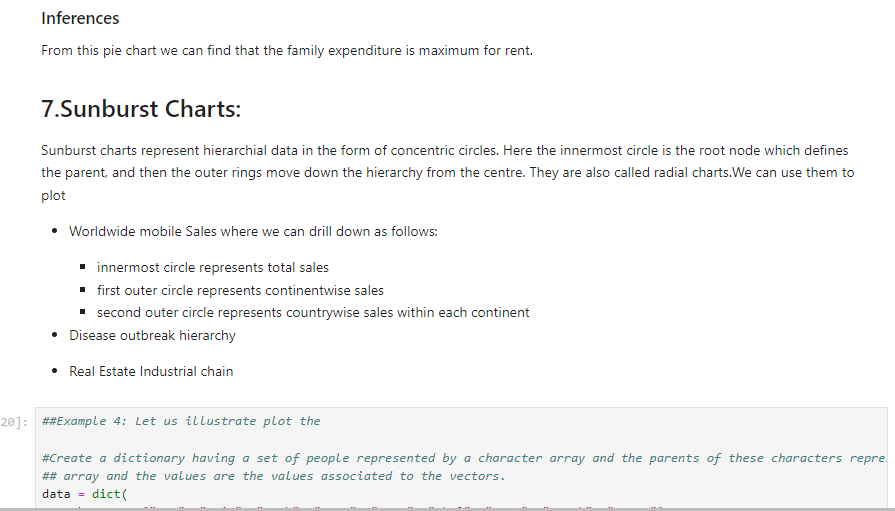
# Values parameter will set values associated to the sector. 'exp\_percent' feature is passed to it.

# labels for the sector are passed to the `house hold categoris` parameter.

fig = px.pie(values=exp\_percent, names=house\_holdcategories, title='Household Expenditure')

fig.show()





##Example 4: Let us illustrate plot the

#Create a dictionary having a set of people represented by a character array and the parents of these characters represented in another

## array and the values are the values associated to the vectors.

data = dict(

character=["Eve", "Cain", "Seth", "Enos", "Noam", "Abel", "Awan", "Enoch", "Azura"],

parent=["", "Eve", "Eve", "Seth", "Seth", "Eve", "Eve", "Awan", "Eve" ],

value=[10, 14, 12, 10, 2, 6, 6, 4, 4])

fig = px.sunburst(

data,

names='character',

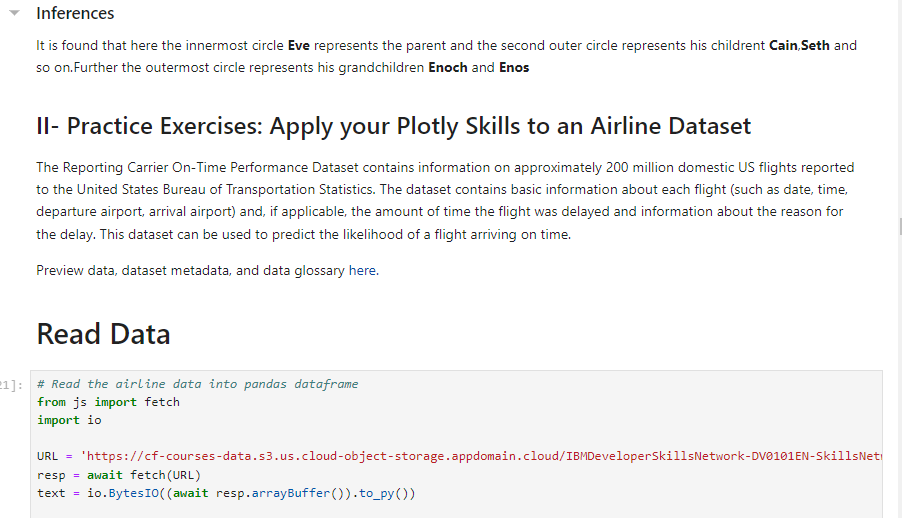
parents='parent',

values='value',

title="Family chart"

)

fig.show()



# Read the airline data into pandas dataframe

from js import fetch

import io

URL = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DV0101EN-SkillsNetwork/Data%20Files/airline\_data.csv'

resp = await fetch(URL)

text = io.BytesIO((await resp.arrayBuffer()).to\_py())

airline\_data = pd.read\_csv(text,

encoding = "ISO-8859-1",

dtype={'Div1Airport': str, 'Div1TailNum': str,

'Div2Airport': str, 'Div2TailNum': str})

print('Data downloaded and read into a dataframe!')

# Preview the first 5 lines of the loaded data

airline\_data.head()

# Shape of the data

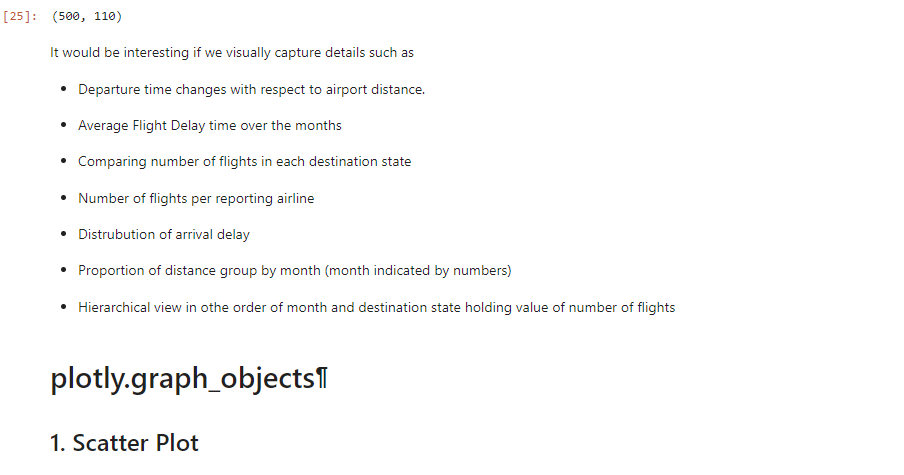
airline\_data.shape

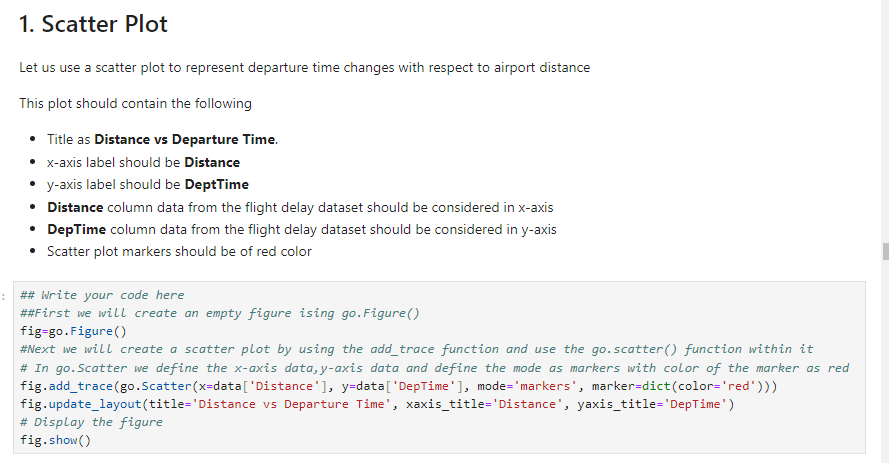
# Randomly sample 500 data points. Setting the random state to be 42 so that we get same result.

data = airline\_data.sample(n=500, random\_state=42)

# Get the shape of the trimmed data

data.shape





## Write your code here

##First we will create an empty figure ising go.Figure()

fig=go.Figure()

#Next we will create a scatter plot by using the add\_trace function and use the go.scatter() function within it

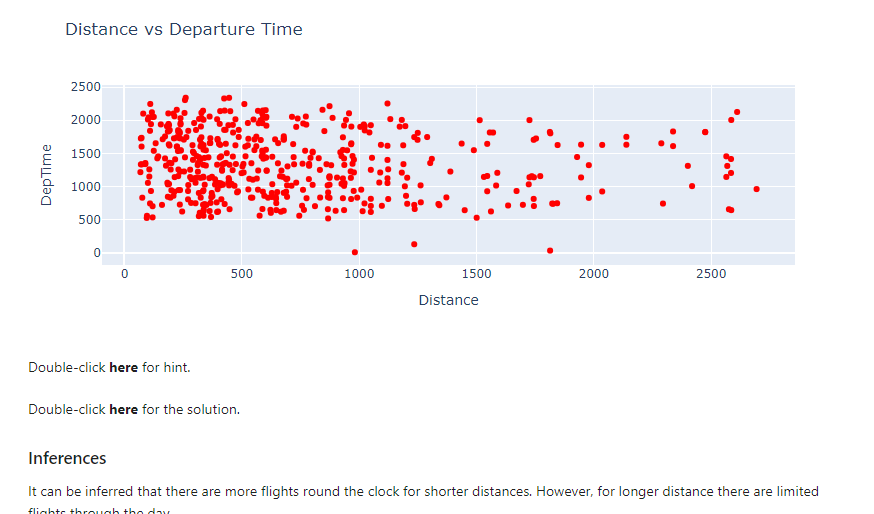
# In go.Scatter we define the x-axis data,y-axis data and define the mode as markers with color of the marker as red

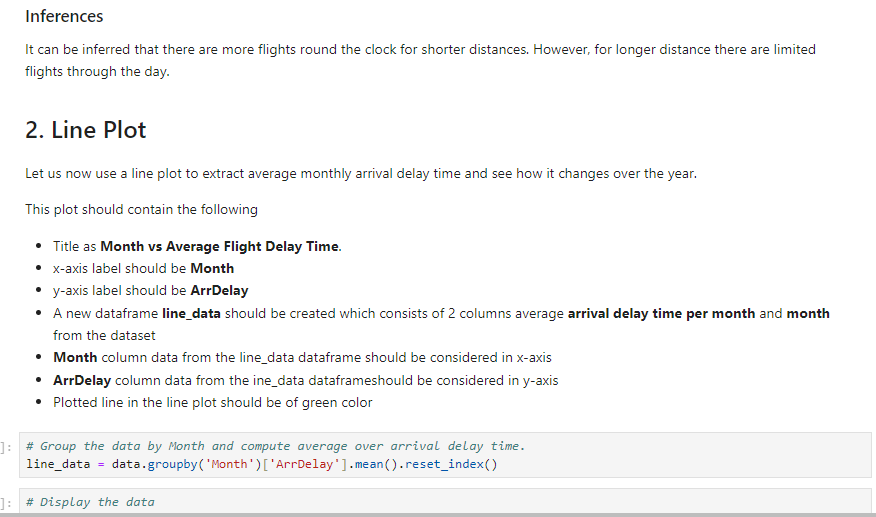
fig.add\_trace(go.Scatter(x=data['Distance'], y=data['DepTime'], mode='markers', marker=dict(color='red')))

fig.update\_layout(title='Distance vs Departure Time', xaxis\_title='Distance', yaxis\_title='DepTime')

# Display the figure

fig.show()





# Group the data by Month and compute average over arrival delay time.

line\_data = data.groupby('Month')['ArrDelay'].mean().reset\_index()

# Display the data

line\_data

## Write your code here

##First we will create an empty figure ising go.Figure()

fig=go.Figure()

##Next we will create a line plot by using the add\_trace function and use the go.scatter() function within it

# In go.Scatter we define the x-axis data,y-axis data and define the mode as lines with color of the marker as green

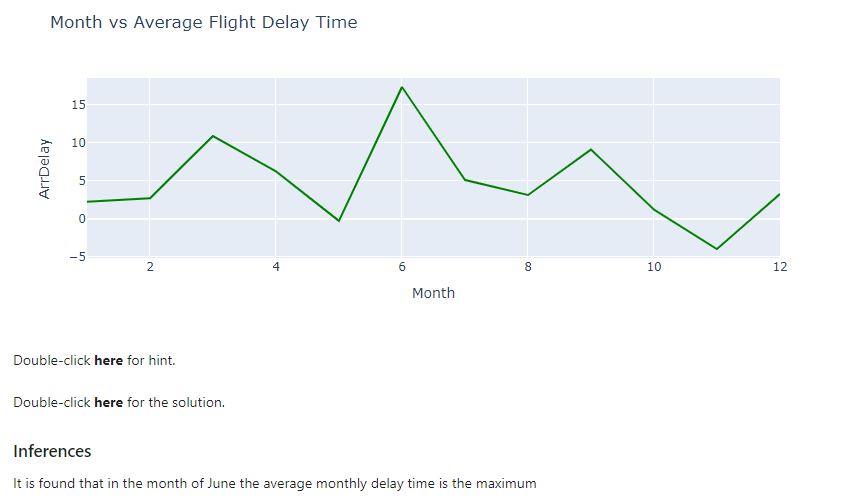
fig.add\_trace(go.Scatter(x=line\_data['Month'], y=line\_data['ArrDelay'], mode='lines', marker=dict(color='green')))

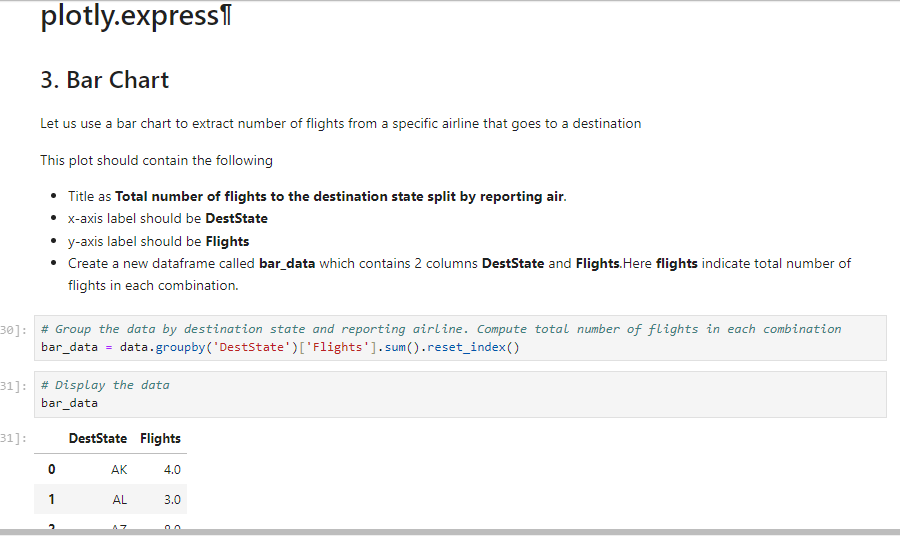
# Create line plot here

## Here we update these values under function attributes such as title,xaxis\_title and yaxis\_title

fig.update\_layout(title='Month vs Average Flight Delay Time', xaxis\_title='Month', yaxis\_title='ArrDelay')

fig.show()





## Write your code here

# Use plotly express bar chart function px.bar. Provide input data, x and y axis variable, and title of the chart.

# This will give total number of flights to the destination state.

fig = px.bar(bar\_data, x="DestState", y="Flights", title='Total number of flights to the destination state split by reporting airline')

fig.show()





# Set missing values to 0

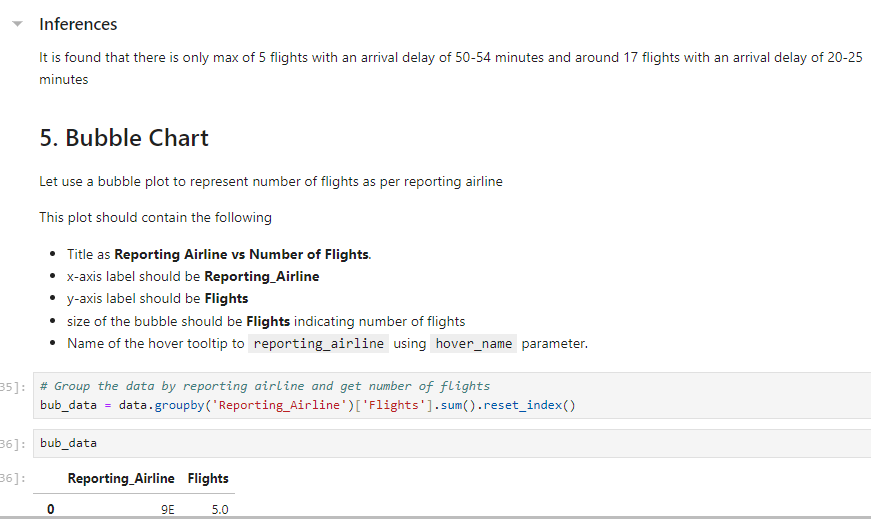
data['ArrDelay'] = data['ArrDelay'].fillna(0)

## Write your code here

## Use plotly express histogram chart function px.histogram.Provide input data x to the histogram

fig = px.histogram(data, x="ArrDelay",title="Total number of flights to the destination state split by reporting air.")

fig.show()



# Group the data by reporting airline and get number of flights

bub\_data = data.groupby('Reporting\_Airline')['Flights'].sum().reset\_index()

bub\_data

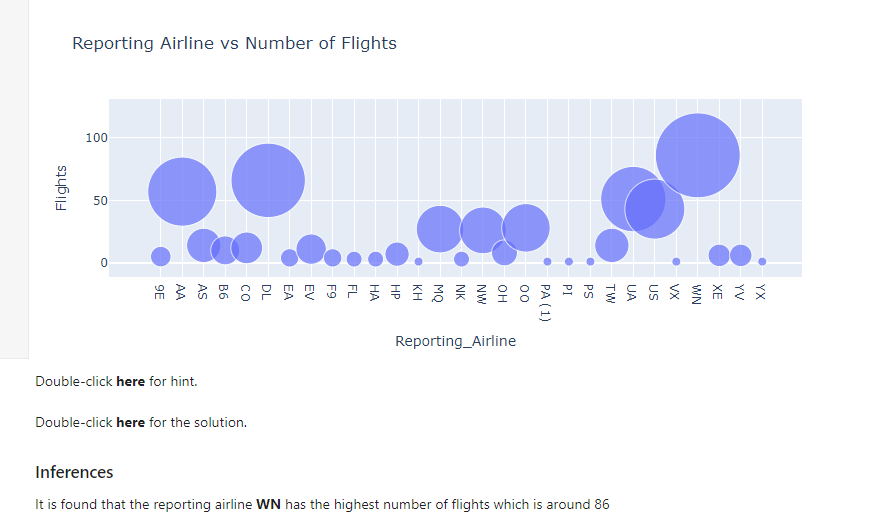
## Write your code here

## Bubble chart using px.scatter function with x ,y and size variables defined.Title defined as Reporting Airline vs Number of Flights

fig = px.scatter(bub\_data, x="Reporting\_Airline", y="Flights", size="Flights",

hover\_name="Reporting\_Airline", title='Reporting Airline vs Number of Flights', size\_max=60)

fig.show()





## Write your code here

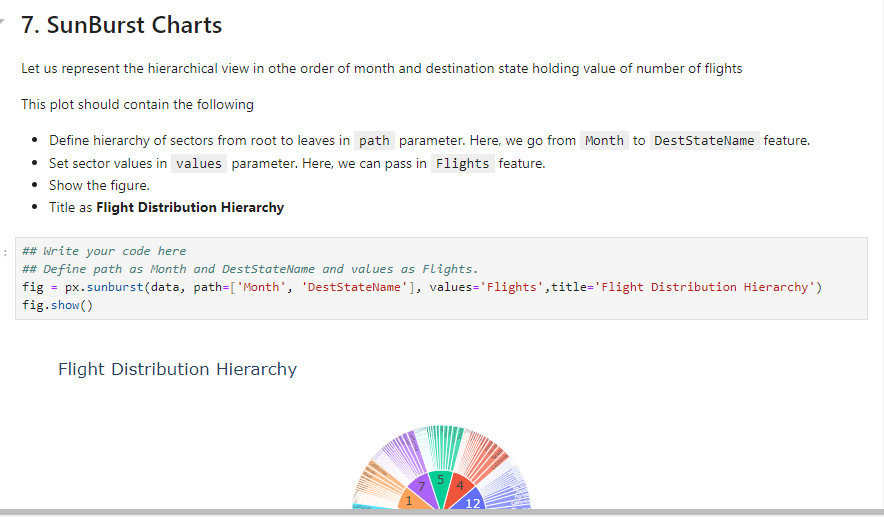
# Use px.pie function to create the chart. Input dataset.

# Values parameter will set values associated to the sector. 'Flights' feature is passed to it.

# labels for the sector are passed to the `names` parameter.

fig = px.pie(data, values='Flights', names='DistanceGroup', title='Flight propotion by Distance Group')

fig.show()



## Write your code here

## Define path as Month and DestStateName and values as Flights.

fig = px.sunburst(data, path=['Month', 'DestStateName'], values='Flights',title='Flight Distribution Hierarchy')

fig.show()