Using appropriate plot on Car manufacturing company dataset auto_mpg.csv given below answer following questions.

In [ ]:
```
import pandas as pd
df = pd.read_csv('dataset/auto_mpg.csv')
df.head()
```

a. Some customers of XYZ Custom Cars are interested in the mileage range of the cars that are restored by the company. They also want to compare the distribution of average mileage and city mileage (25% less than the average mileage).

In [ ]:
```
#importing the required packages
import matplotlib.pyplot as plt
#creating an empty canvas/figure
fig = plt.figure(figsize=[6,6])
#setting axes
ax = fig.add_axes([0, 0, 1, 1])
ax.set_title('Distribution of mileage')
#plotting boxplot
ax.boxplot(df['mpg'])
fig.show()
```

There is no data for city mileage, but city mileage is 25% less than the average mileage i.e. ‘mpg’. Next is to process the data for city mileage.

In [ ]:
```
#processing data for city mileage
df['city_mileage']=df['mpg']*0.75
df.head()
```

A new column ‘city_mileage’ is created . Next, the distribution of the average mileage and city mileage has to be compared.

In [ ]:
```

#Comparing avarage mpg and city mpg using boxplot
mpg_list = [df['mpg'],df['city_mileage']]
#importing the required packages
import matplotlib.pyplot as plt
#creating an empty canvas/figure
fig = plt.figure(figsize=[6,5])
#setting axes
ax = fig.add_axes([0, 0, 1, 1])
ax.set_title('Distribution of Average MPG vs City MPG')
ax.set_ylabel('Mileage per gallon')
#plotting box plot
ax.boxplot(mpg_list,widths = 0.5)
fig.show()
```

b. The engineers at XYZ Custom Cars want to infer the correlation between horsepower and mileage.

In [ ]:
```
#visualizing correlation between mileage and horsepower
#importing the required packages
import matplotlib.pyplot as plt
#creating an empty canvas/figure
fig = plt.figure(figsize=[10,5])
#setting axes
ax = fig.add_axes([0, 0, 1, 1])
#plotting scatter plot
ax.scatter(df['horsepower'],df['mpg'])
ax.set_title("Scatter plot of horsepower and mileage",fontsize=16)
ax.set_ylabel('Mileage per gallon',fontsize=12)
ax.set_xlabel('Horsepower',fontsize=12)
fig.show()
```

c.The engineers at XYZ Custom Cars want to know the number of cars released in each year

In [ ]:
```
#Creating a grouped DataFrame according to model year
grouped_df = df.groupby(['model_year']).count()[['name']]
grouped_df.reset_index(inplace = True)
#plotting the number of cars in each year
#importing the required packages
import matplotlib.pyplot as plt
#creating an empty canvas/figure
fig = plt.figure(figsize=[7,5])
#setting axes
ax = fig.add_axes([0, 0, 1, 1])
ax.set_title("Number of cars in each year",fontsize=16)
ax.set_ylabel('Number of cars',fontsize=12)
ax.set_xlabel('Model year',fontsize=12)
#plotting bar graph
ax.bar(grouped_df['model_year'], grouped_df['name'])
fig.show()
```

d.The engineers at XYZ Custom Cars want to identify the distribution of horsepower.

In [ ]:
```
#creating a histogram for horsepower
#importing the required packages
import matplotlib.pyplot as plt
#creating an empty canvas/figure
fig = plt.figure(figsize=[8,6])
#setting axes
ax = fig.add_axes([0, 0, 1, 1])
#plotting histogram
ax.hist(df['horsepower'], bins=20)
ax.set_title('Distribution of Horsepower',fontsize = 17)
ax.set_ylabel('Frequency',fontsize=12)
ax.set_xlabel('Horsepower',fontsize=12)
fig.show()
```

e. The management of XYZ Custom Cars has decided to open a new branch and is yet to decide the location. They would like to concentrate more on the ‘origin’ of the cars to make a better decision.

In [ ]:
```
pie_df = pd.DataFrame()    #creating a sub dataframe to plot the pie chart
pie_df['Count'] = df['origin'].value_counts()    #getting the count of ‘origin’ and assigning to the df
pie_df = pie_df.reset_index()    #re-arranging the index
pie_df.rename(columns={'index':'Country'},inplace=True)    #re-naming the col
pie_df
```

In [ ]:
```
#importing the required packages
import matplotlib.pyplot as plt
fig = plt.figure(figsize=[8,6])
ax = fig.add_axes([0, 0, 1, 1])
explode = [0.05, 0, 0]
colors = ['yellowgreen', 'mediumaquamarine', 'khaki' ]
ax.set_title('Origin of the Cars')
ax.pie(pie_df['Count'], labels=pie_df['Country'],
        autopct='%0.1f%%',explode = explode,
        shadow = True, startangle = 250,
        colors = colors)
fig.show()
```

8. Consider the credit card dataset which contains the following columns

- **CLIENTNUM: Primary key of the dataset**
- **Attrition_Flag: Indicates if a customer is retained or attrited**
- **Customer_Age: Age of the customer**
- **Gender: Gender of the customer**
- **Dependent_count: Number of people dependent on the customer**
- **Education_Level: Highest level of education of the customer**
- **Income_Category: Range of income of the customer**
- **Credit_Limit: Credit card limit**
- **Total_Revolving_Bal: Pending balance of the credit**
- **Avg_Purchase: Amount of purchase made by the customer on credit card**
- **Total_Trans_Amt: Total transaction amount**

Click here to download the dataset.

Based on the above information, import relevant libraries, and perform the following steps:

In [ ]:
```
import pandas as pd
df = pd.read_csv('dataset/CreditCard_DV.csv')
df
```

1. Create a bivariate plot to find if there is a correlation between credit card limit and average purchase made on the card.

In [ ]:
```
#importing the required packages
import matplotlib.pyplot as plt
#creating an empty canvas/figure
fig = plt.figure(figsize=[10,5])
#setting axes
ax = fig.add_axes([0, 0, 1, 1])
#plotting scatter plot
ax.scatter(df['Credit_Limit'],df['Avg_Purchase'])
ax.set_title("Credit_Limit vs Avg_Purchase",fontsize=16)
ax.set_ylabel('Avg_Purchase',fontsize=12)
ax.set_xlabel('Credit_Limit',fontsize=12)
fig.show()
```

2. Visualise the distribution of values for credit card limit and average purchase made on the card. Also, identify the outliers in the data, if any.

In [ ]:
```
df_list = [df['Credit_Limit'],df['Avg_Purchase']]
#importing the required packages
import matplotlib.pyplot as plt
#creating an empty canvas/figure
fig = plt.figure(figsize=[6,5])
#setting axes
ax = fig.add_axes([0, 0, 1, 1])
ax.set_title('Credit_Limit and Avg_Purchase ')
#plotting box plot
ax.boxplot(df_list,widths = 0.5)
fig.show()
```

3. Provide a visual representation of the number of customers in each income group using a bar chart.

In [ ]:
```
grouped_df = df.groupby(['Income_Category']).count()[['Customer_Age']]
grouped_df.reset_index(inplace = True)
#plotting the number of cars in each year
#importing the required packages
import matplotlib.pyplot as plt
#creating an empty canvas/figure
fig = plt.figure(figsize=[7,5])
#setting axes
ax = fig.add_axes([0, 0, 1, 1])
ax.set_title("Income_Category",fontsize=16)
ax.set_ylabel('Count',fontsize=12)
ax.set_xlabel('Income_Category',fontsize=12)
#plotting bar graph
ax.bar(grouped_df['Income_Category'], grouped_df['Customer_Age'])
fig.show()
```

4. Plot the frequency distribution of the total transaction amount

In [ ]:
```
#importing the required packages
import matplotlib.pyplot as plt
#creating an empty canvas/figure
fig = plt.figure(figsize=[8,6])
#setting axes
ax = fig.add_axes([0, 0, 1, 1])
#plotting histogram
ax.hist(df['Total_Trans_Amt'], bins=20)
ax.set_title('Distribution of  total transaction amount',fontsize = 17)
ax.set_ylabel('Frequency',fontsize=12)
ax.set_xlabel(' total transaction amount',fontsize=12)
fig.show()
```

5. Graphically represent the percentage of customers retained and those attrited. Highlight the latter by slicing it apart from the main pie.

In [ ]:
```
pie_df = pd.DataFrame()    #creating a sub dataframe to plot the pie chart
pie_df['Count'] = df['Attrition_Flag'].value_counts()    #getting the count of 'Attrition_Flag' and assigning to the d
pie_df = pie_df.reset_index()    #re-arranging the index
pie_df.rename(columns={'index':'customers'},inplace=True)    #re-naming the col
pie_df
```

In [ ]:
```
#importing the required packages
import matplotlib.pyplot as plt
fig = plt.figure(figsize=[8,6])
ax = fig.add_axes([0, 0, 1, 1])
explode = [0.05, 0]
colors = ['yellowgreen', 'mediumaquamarine']
ax.set_title('Attrition Flag')
ax.pie(pie_df['Count'], labels=pie_df['customers'],
        autopct='%0.1f%%',explode = explode,
        shadow = True, startangle = 250,
        colors = colors)
fig.show()
```

In [ ]:

In [ ]:

In [ ]: