Python Project Correction

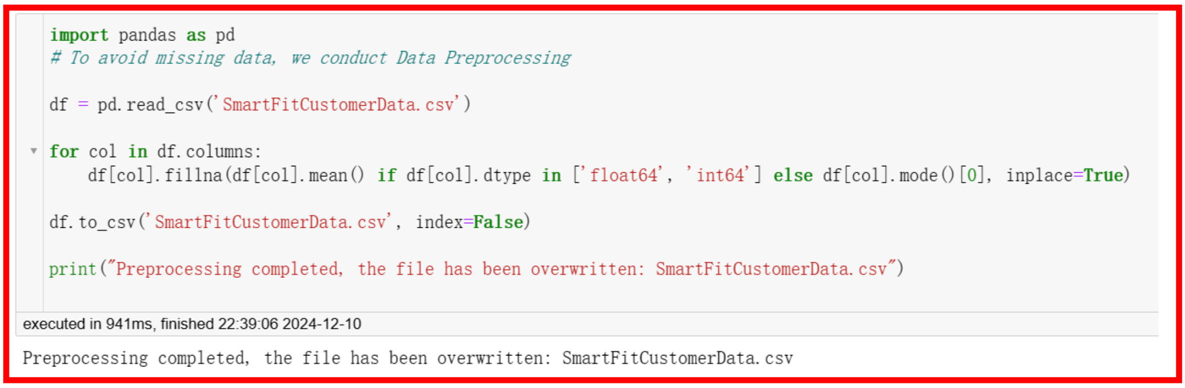
2230005063	**Yuanheng Zhang (Rocky)**
2330036065	**Rongyuan LI (Kayla)**
2330036069	**Yaning LI (Katherine)**
2330024064	**Jiantao HE (Laurence)**

Code Correction

(The modified part is in the red box)

1. Question Solving

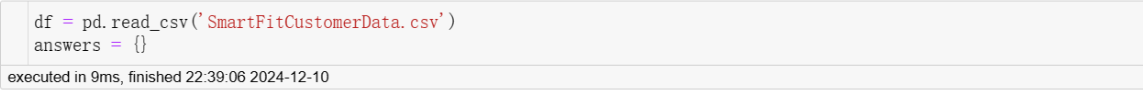
*Data pre-processing



```
import pandas as pd
# To avoid missing data, we conduct Data Preprocessing

df = pd. read_csv(’SmartFitCustomerData. csv’)

for col in df. columns:
    df[col]. fillna(df[col]. mean() if df[col]. dtype in [’float64’, ’int64’] else df[col]. mode()[0], inplace=True)

df. to_csv(’SmartFitCustomerData. csv’, index=False)

print(”Preprocessing completed, the file has been overwritten: SmartFitCustomerData. csv”)
```

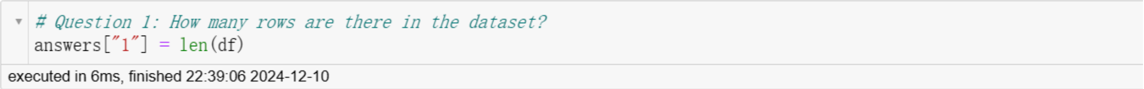
```
executed in 941ms, finished 22:39:06 2024-12-10
```

```
Preprocessing completed, the file has been overwritten: SmartFitCustomerData. csv
```



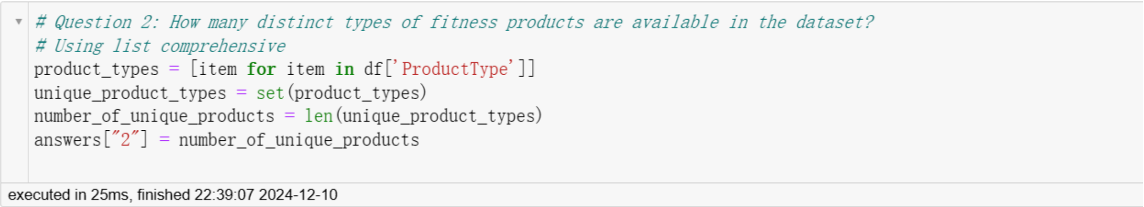
```
df = pd. read_csv(’SmartFitCustomerData. csv’)
answers = {}
```

```
executed in 9ms, finished 22:39:06 2024-12-10
```



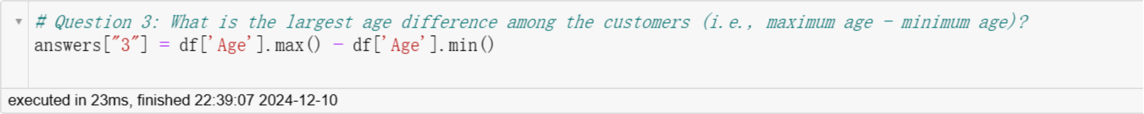
```
# Question 1: How many rows are there in the dataset?
answers[”1”] = len(df)
```

```
executed in 6ms, finished 22:39:06 2024-12-10
```



```
# Question 2: How many distinct types of fitness products are available in the dataset?
# Using list comprehensive
product_types = [item for item in df[’ProductType’]]
unique_product_types = set(product_types)
number_of_unique_products = len(unique_product_types)
answers[”2”] = number_of_unique_products
```

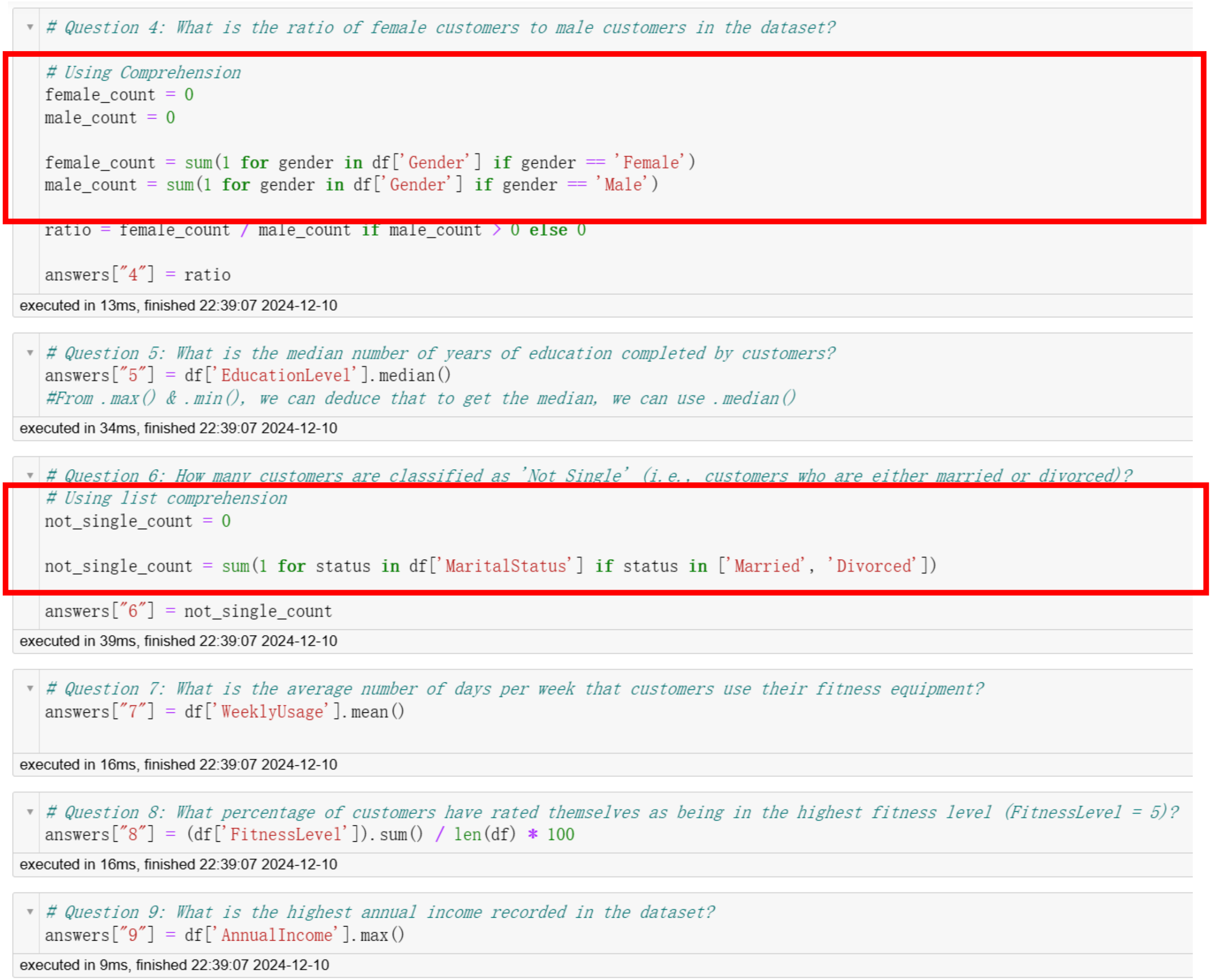
```
executed in 25ms, finished 22:39:07 2024-12-10
```



```
# Question 3: What is the largest age difference among the customers (i. e., maximum age - minimum age)?
answers[”3”] = df[’Age’]. max() - df[’Age’]. min()
```

```
executed in 23ms, finished 22:39:07 2024-12-10
```

*Optimization algorithm



```
# Question 4: What is the ratio of female customers to male customers in the dataset?

# Using Comprehension
female_count = 0
male_count = 0

female_count = sum(1 for gender in df['Gender'] if gender == 'Female')
male_count = sum(1 for gender in df['Gender'] if gender == 'Male')

ratio = female_count / male_count if male_count > 0 else 0

answers["4"] = ratio
```

executed in 13ms, finished 22:39:07 2024-12-10

```
# Question 5: What is the median number of years of education completed by customers?
answers["5"] = df['EducationLevel'].median()
#From .max() & .min(), we can deduce that to get the median, we can use .median()
```

executed in 34ms, finished 22:39:07 2024-12-10

```
# Question 6: How many customers are classified as 'Not Single' (i.e., customers who are either married or divorced)?
# Using list comprehension
not_single_count = 0

not_single_count = sum(1 for status in df['MaritalStatus'] if status in ['Married', 'Divorced'])

answers["6"] = not_single_count
```

executed in 39ms, finished 22:39:07 2024-12-10

```
# Question 7: What is the average number of days per week that customers use their fitness equipment?
answers["7"] = df['WeeklyUsage'].mean()
```

executed in 16ms, finished 22:39:07 2024-12-10

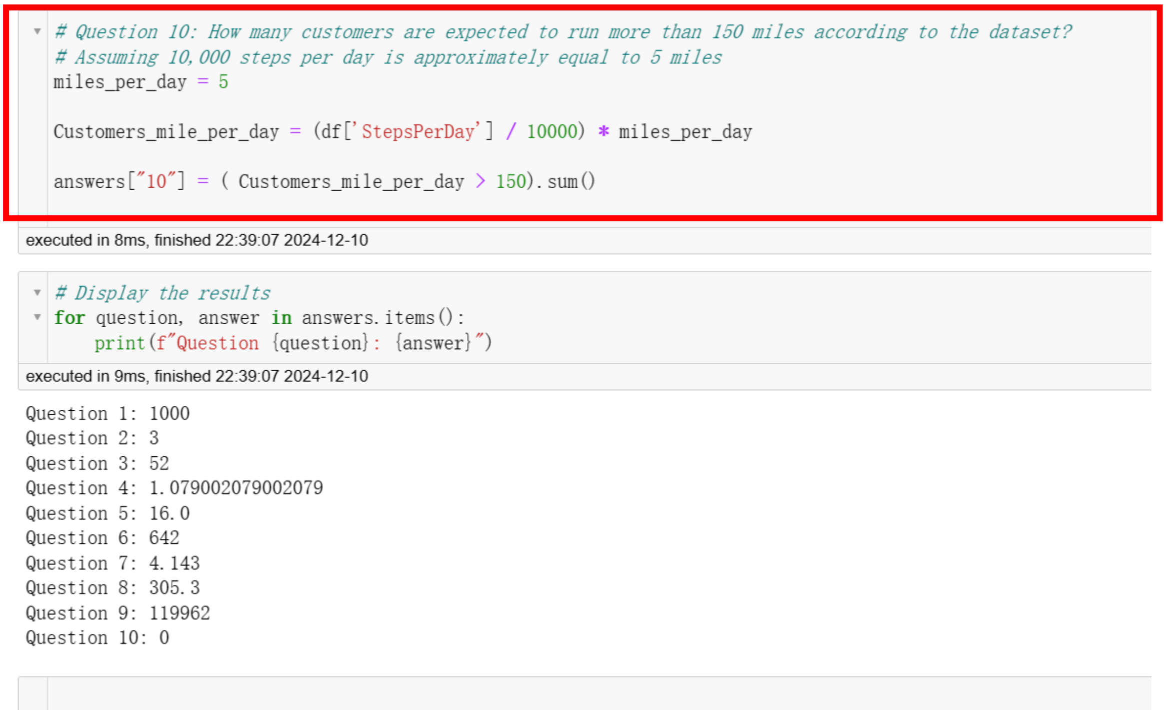
```
# Question 8: What percentage of customers have rated themselves as being in the highest fitness level (FitnessLevel = 5)?
answers["8"] = (df['FitnessLevel']).sum() / len(df) * 100
```

executed in 16ms, finished 22:39:07 2024-12-10

```
# Question 9: What is the highest annual income recorded in the dataset?
answers["9"] = df['AnnualIncome'].max()
```

executed in 9ms, finished 22:39:07 2024-12-10

***By setting a new variable, called “miles_per_day” can walk, the calculation results are clearer, and the calculation error is avoided.**



```
# Question 10: How many customers are expected to run more than 150 miles according to the dataset?
# Assuming 10,000 steps per day is approximately equal to 5 miles
miles_per_day = 5

Customers_mile_per_day = (df['StepsPerDay'] / 10000) * miles_per_day

answers["10"] = ( Customers_mile_per_day > 150).sum()
```

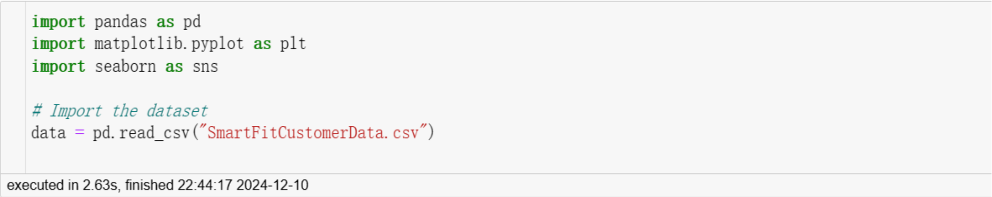
executed in 8ms, finished 22:39:07 2024-12-10

```
# Display the results
for question, answer in answers.items():
    print(f"Question {question}: {answer}")
```

executed in 9ms, finished 22:39:07 2024-12-10

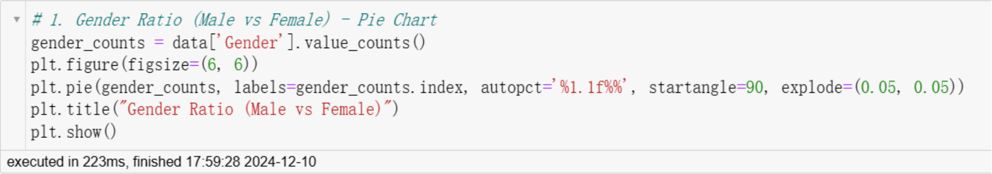
```
Question 1: 1000
Question 2: 3
Question 3: 52
Question 4: 1.079002079002079
Question 5: 16.0
Question 6: 642
Question 7: 4.143
Question 8: 305.3
Question 9: 119962
Question 10: 0
```

2. Visual part



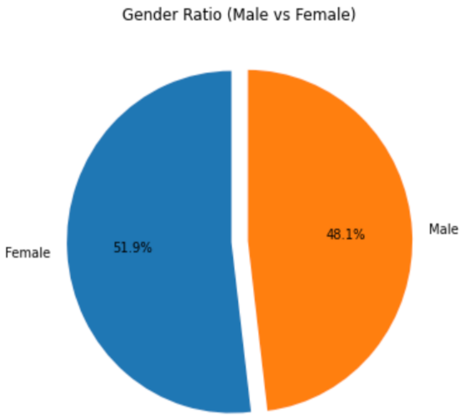
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Import the dataset
data = pd.read_csv("SmartFitCustomerData.csv")
```

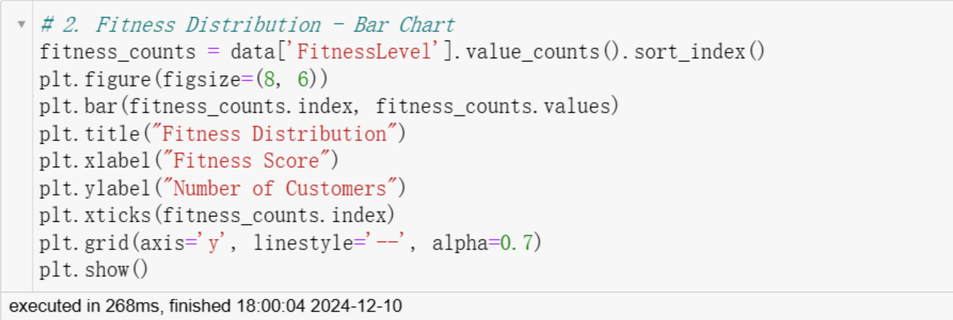
executed in 2.63s, finished 22:44:17 2024-12-10



```
# 1. Gender Ratio (Male vs Female) - Pie Chart
gender_counts = data['Gender'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(gender_counts, labels=gender_counts.index, autopct='%1.1f%%', startangle=90, explode=(0.05, 0.05))
plt.title("Gender Ratio (Male vs Female)")
plt.show()
```

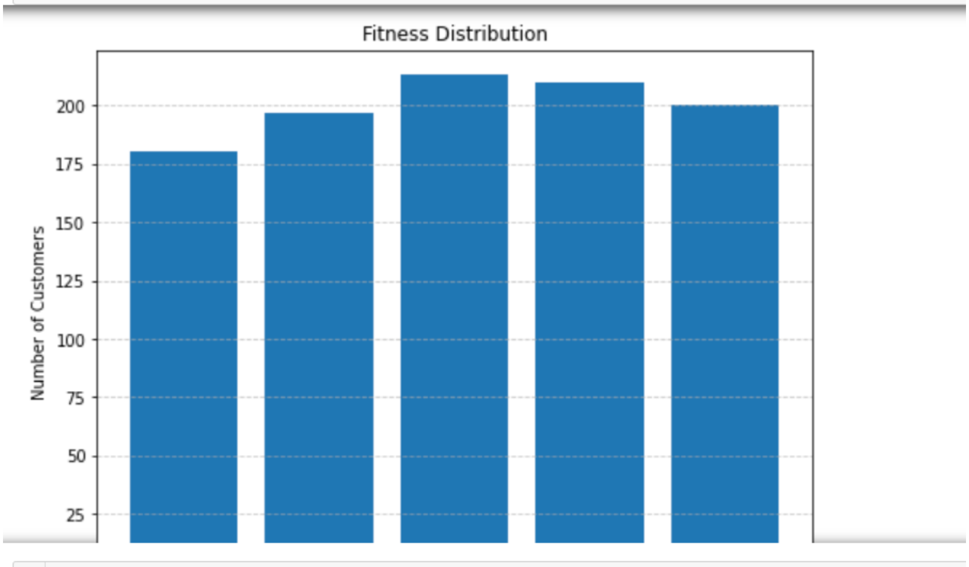
executed in 223ms, finished 17:59:28 2024-12-10



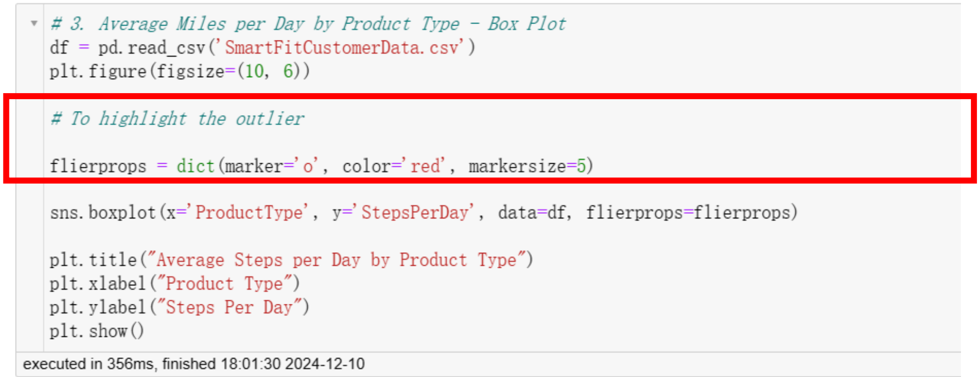


```
# 2. Fitness Distribution - Bar Chart
fitness_counts = data['FitnessLevel'].value_counts().sort_index()
plt.figure(figsize=(8, 6))
plt.bar(fitness_counts.index, fitness_counts.values)
plt.title("Fitness Distribution")
plt.xlabel("Fitness Score")
plt.ylabel("Number of Customers")
plt.xticks(fitness_counts.index)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

executed in 268ms, finished 18:00:04 2024-12-10

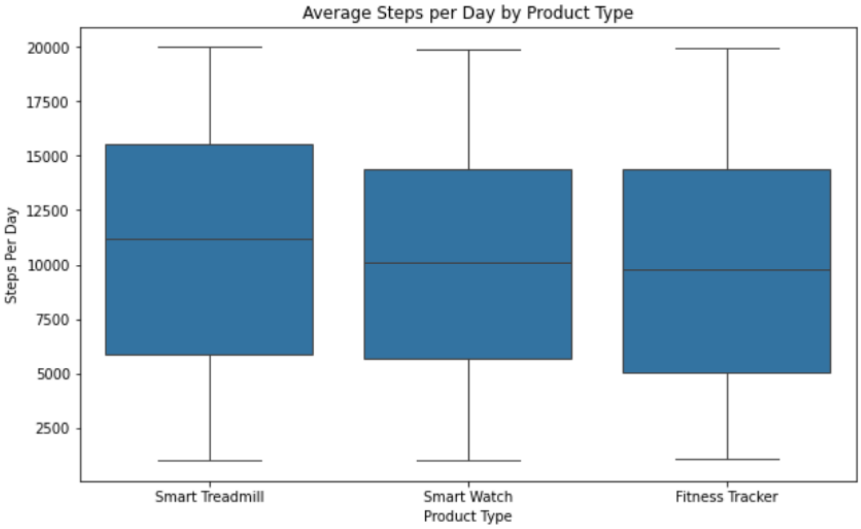


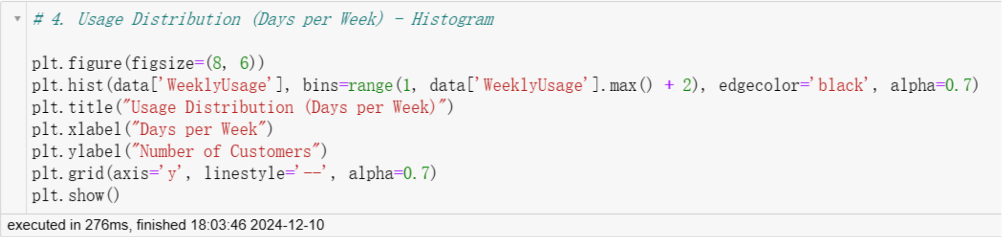
*Adding outlier determination



```
# 3. Average Miles per Day by Product Type – Box Plot
df = pd.read_csv(’SmartFitCustomerData.csv’)
plt.figure(figsize=(10, 6))

# To highlight the outlier

flierprops = dict(marker=’o’, color=’red’, markersize=5)

sns.boxplot(x=’ProductType’, y=’StepsPerDay’, data=df, flierprops=flierprops)

plt.title(”Average Steps per Day by Product Type”)
plt.xlabel(”Product Type”)
plt.ylabel(”Steps Per Day”)
plt.show()
```

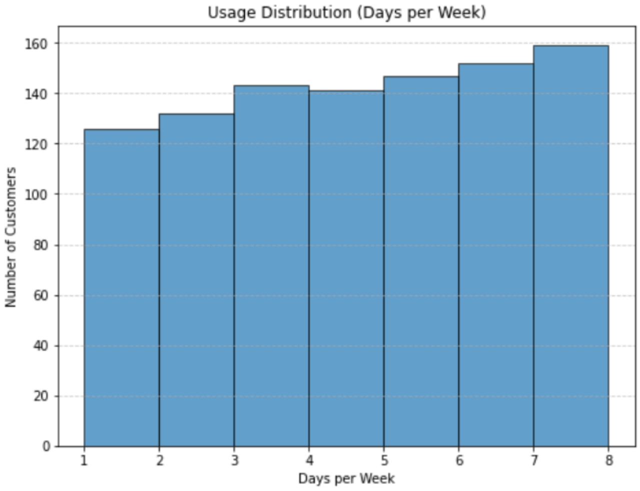
executed in 356ms, finished 18:01:30 2024-12-10





```
# 4. Usage Distribution (Days per Week) – Histogram

plt.figure(figsize=(8, 6))
plt.hist(data[’WeeklyUsage’], bins=range(1, data[’WeeklyUsage’].max() + 2), edgecolor=’black’, alpha=0.7)
plt.title(”Usage Distribution (Days per Week)”)
plt.xlabel(”Days per Week”)
plt.ylabel(”Number of Customers”)
plt.grid(axis=’y’, linestyle=’--’, alpha=0.7)
plt.show()
```

executed in 276ms, finished 18:03:46 2024-12-10



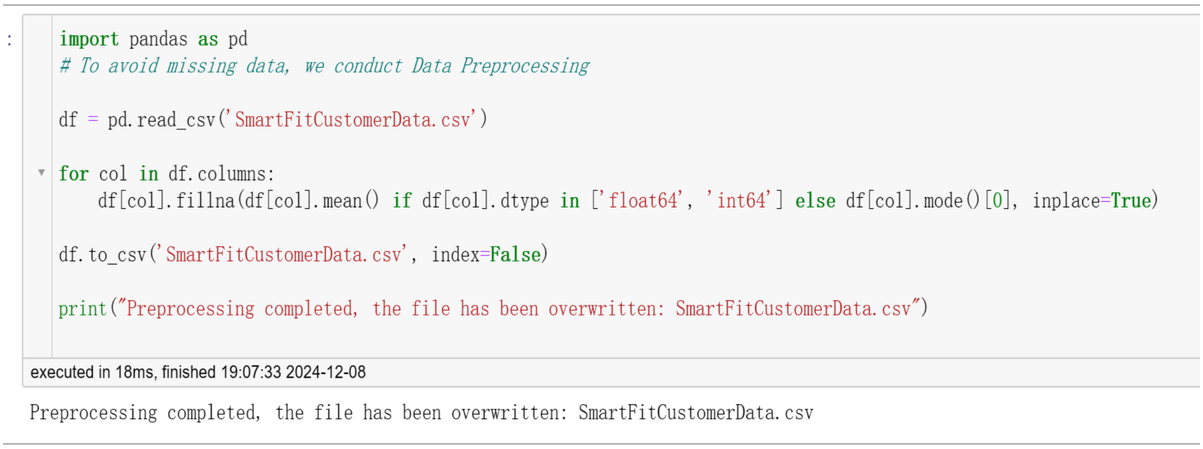
Q&A session

Question 1:

What to do when data is missing?

Group answer:

First, before data processing and visualization, we do data pre-processing. With a code called ‘.fillna’ to retrieve the data that has no gaps, and replaces the data with the average value. Thus, complete data reprocessing.



```
:
import pandas as pd
# To avoid missing data, we conduct Data Preprocessing

df = pd.read_csv('SmartFitCustomerData.csv')

for col in df.columns:
    df[col].fillna(df[col].mean() if df[col].dtype in ['float64', 'int64'] else df[col].mode()[0], inplace=True)

df.to_csv('SmartFitCustomerData.csv', index=False)

print("Preprocessing completed, the file has been overwritten: SmartFitCustomerData.csv")
```

executed in 18ms, finished 19:07:33 2024-12-08

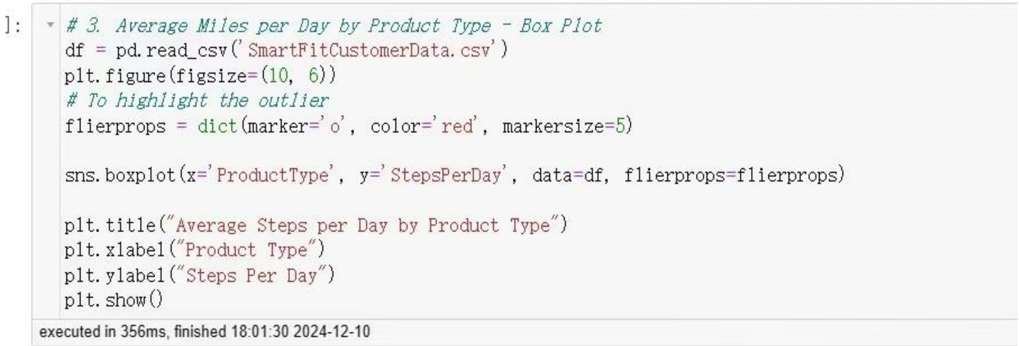
```
Preprocessing completed, the file has been overwritten: SmartFitCustomerData.csv
```

Question 2:

How to solve boxplot outlier?

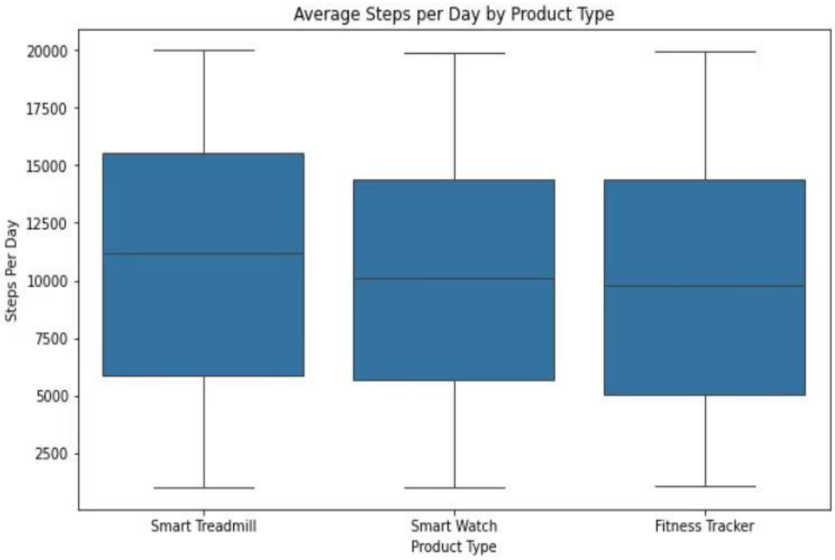
Group answer:

Using code called ‘flierprops’, you set the outlier property and represent it on an icon. (Since the database generated by our code does not have outliers, it is not shown in the end.)



```
# 3. Average Miles per Day by Product Type - Box Plot
df = pd.read_csv('SmartFitCustomerData.csv')
plt.figure(figsize=(10, 6))
# To highlight the outlier
flierprops = dict(marker='o', color='red', markersize=5)

sns.boxplot(x='ProductType', y='StepsPerDay', data=df, flierprops=flierprops)

plt.title("Average Steps per Day by Product Type")
plt.xlabel("Product Type")
plt.ylabel("Steps Per Day")
plt.show()
```

```
executed in 356ms, finished 18:01:30 2024-12-10
```

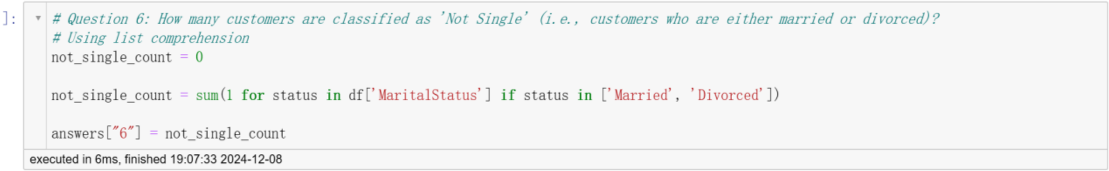


Question 3:

Can you improve the for loop?

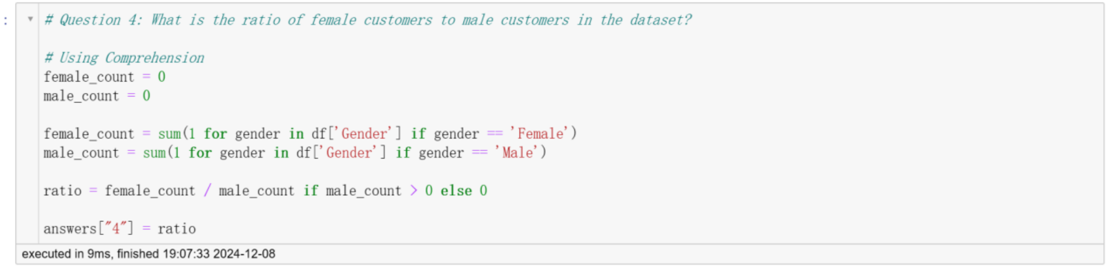
Group answer:

We can solve this by using list comprehension.



```
# Question 6: How many customers are classified as 'Not Single' (i.e., customers who are either married or divorced)?
# Using list comprehension
not_single_count = 0

not_single_count = sum(1 for status in df['MaritalStatus'] if status in ['Married', 'Divorced'])

answers["6"] = not_single_count
```

executed in 6ms, finished 19:07:33 2024-12-08



```
# Question 4: What is the ratio of female customers to male customers in the dataset?

# Using Comprehension
female_count = 0
male_count = 0

female_count = sum(1 for gender in df['Gender'] if gender == 'Female')
male_count = sum(1 for gender in df['Gender'] if gender == 'Male')

ratio = female_count / male_count if male_count > 0 else 0

answers["4"] = ratio
```

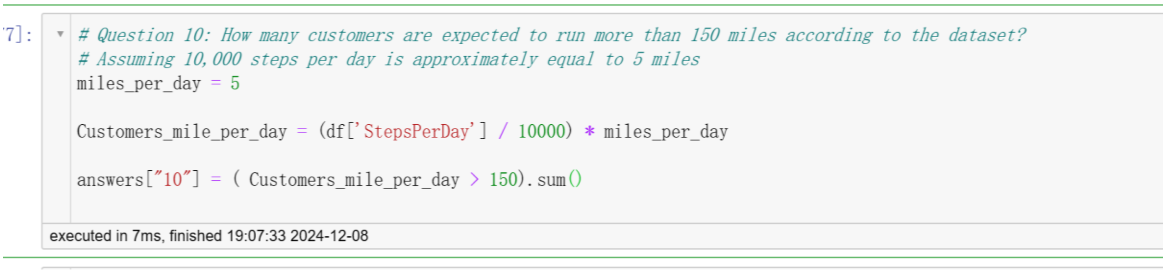
executed in 9ms, finished 19:07:33 2024-12-08

Question 4:

Can the solution of step and mile be clearer?

Group answer:

By setting a new variable called the distance each person can walk. Thus, the calculation results are more clear and clear to avoid calculation errors.



```
# Question 10: How many customers are expected to run more than 150 miles according to the dataset?
# Assuming 10,000 steps per day is approximately equal to 5 miles
miles_per_day = 5

Customers_mile_per_day = (df['StepsPerDay'] / 10000) * miles_per_day

answers["10"] = ( Customers_mile_per_day > 150).sum()
```

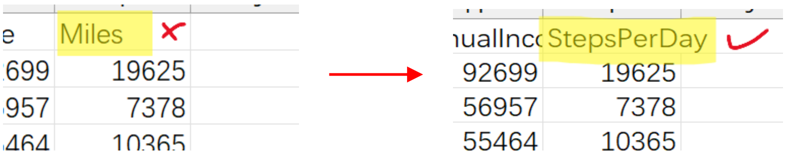
```
executed in 7ms, finished 19:07:33 2024-12-08
```

Question 5:

Is one of the column of data on the csv mile or steps per day?

Group answer:

Firstly, we checked the file generated by AI, and what is generated is steps per day, so there is no problem with our code; secondly, we found out that even though the question asks for miles, we can solve the problem by the transformation method of 10,000 steps equal to 5miles. So, the column header on the csv file is steps per day.



e	Miles ✗
699	19625
957	7378
464	10365

ıualInc	StepsPerDay ✓
92699	19625
56957	7378
55464	10365