# walmart-Dataset@Dhanureddy

September 7, 2024

## 1 Walmart dataset exploration

**About walmart:**

Walmart is an American multinational retail corporation that operates a chain of supercenters, discount departmental stores, and grocery stores from the United States. Walmart has more than 100 million customers worldwide.

**Business problem:**

The Management team at Walmart Inc. wants to analyze the customer purchase behavior (specifically, purchase amount) against the customer's gender and the various other factors to help the business make better decisions. They want to understand if the spending habits differ between male and female customers: Do women spend more on Black Friday than men? (Assume 50 million customers are male and 50 million are female).

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[ ]:
```
df = pd.read_csv("walmart_data.csv")
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-4c76b50bd59b> in <cell line: 1>()
----> 1 df = pd.read_csv("walmart_data.csv")

NameError: name 'pd' is not defined
```

## 2 Basic data exploration

[ ]:
```
df.head(5)
```

[ ]:
```
   User_ID Product_ID Gender   Age  Occupation City_Category  \
0  1000001  P00069042      F  0-17          10             A
1  1000001  P00248942      F  0-17          10             A
```

```
2  1000001  P00087842      F  0-17          10             A
3  1000001  P00085442      F  0-17          10             A
4  1000002  P00285442      M   55+          16             C

  Stay_In_Current_City_Years  Marital_Status  Product_Category  Purchase
0                          2               0                 3      8370
1                          2               0                 1     15200
2                          2               0                12      1422
3                          2               0                12      1057
4                         4+               0                 8      7969
```

[ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 550068 entries, 0 to 550067
Data columns (total 10 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 0   User_ID                     550068 non-null  int64
 1   Product_ID                  550068 non-null  object
 2   Gender                      550068 non-null  object
 3   Age                         550068 non-null  object
 4   Occupation                  550068 non-null  int64
 5   City_Category               550068 non-null  object
 6   Stay_In_Current_City_Years  550068 non-null  object
 7   Marital_Status              550068 non-null  int64
 8   Product_Category            550068 non-null  int64
 9   Purchase                    550068 non-null  int64
dtypes: int64(5), object(5)
memory usage: 42.0+ MB
```

Inference:

1. there are in total ten columns, with no null values.
2. There are in total five each string and integer datatype columns.

Finding unique values of each column in the dataframe

[ ]:
```
for columns in df.columns:
  unique_count = df[columns].nunique()
  print(columns, "-", unique_count)
```

```
User_ID - 5891
Product_ID - 3631
Gender - 2
Age - 7
Occupation - 21
City_Category - 3
Stay_In_Current_City_Years - 5
```

```
Marital_Status - 2
Product_Category - 20
Purchase - 18105
```

Checking if there are any possible null values in the dataframe

```
[ ]: df.isna().isna().sum()
```

```
[ ]: User_ID                       0
     Product_ID                    0
     Gender                        0
     Age                           0
     Occupation                    0
     City_Category                 0
     Stay_In_Current_City_Years    0
     Marital_Status                0
     Product_Category              0
     Purchase                      0
     dtype: int64
```

Shape of the dataframe

```
[ ]: df.shape
```

```
[ ]: (550068, 10)
```

Summary of the dataframe describing statistical information of categorical variables

```
[ ]: summary = df.describe()
     summary
```

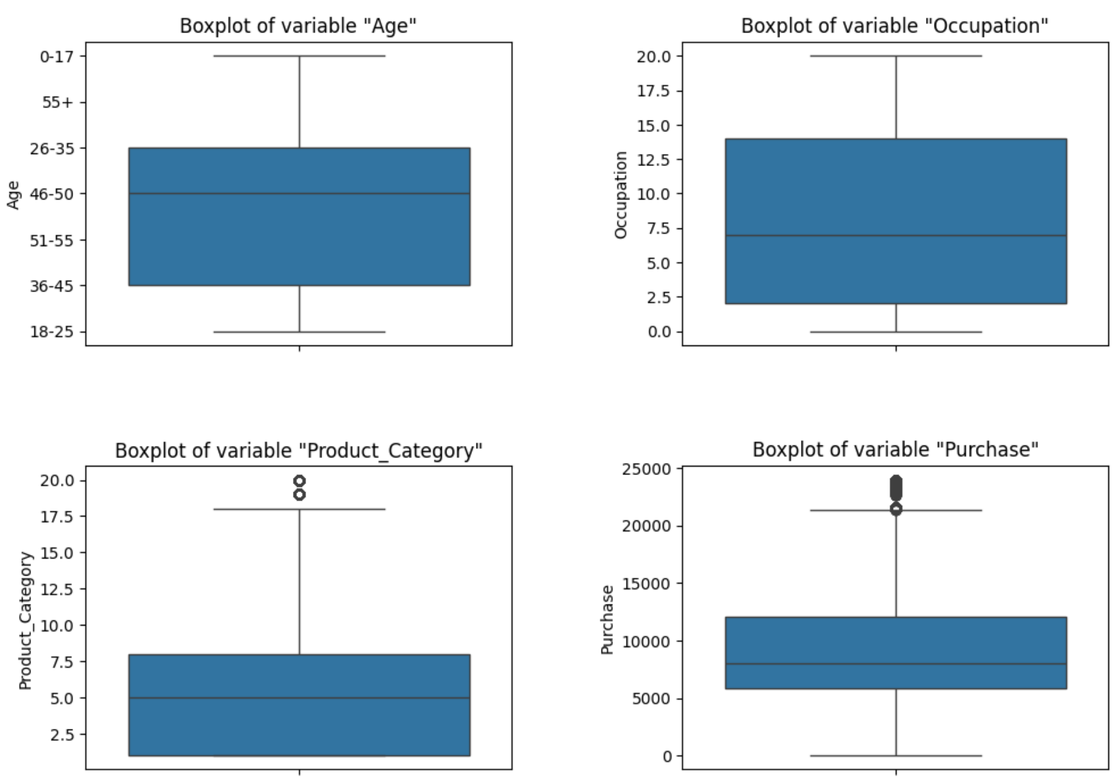
```
[ ]:            User_ID     Occupation  Marital_Status  Product_Category  \
     count  5.500680e+05  550068.000000   550068.000000     550068.000000
     mean   1.003029e+06       8.076707        0.409653          5.404270
     std    1.727592e+03       6.522660        0.491770          3.936211
     min    1.000001e+06       0.000000        0.000000          1.000000
     25%    1.001516e+06       2.000000        0.000000          1.000000
     50%    1.003077e+06       7.000000        0.000000          5.000000
     75%    1.004478e+06      14.000000        1.000000          8.000000
     max    1.006040e+06      20.000000        1.000000         20.000000

                 Purchase
     count  550068.000000
     mean     9263.968713
     std      5023.065394
     min        12.000000
     25%      5823.000000
     50%      8047.000000
     75%     12054.000000
```

```
max      23961.000000
```

Finiding if there are any outliers through use of boxplots

```
[ ]: plt.figure(figsize =(18,8))
     plt.subplots_adjust(left=0.4, right=0.9, top=0.9, bottom=0.1, wspace=0.4,␣
      ↪hspace=0.4)

     plt.subplot(2,2,1)
     sns.boxplot(data = df.Age)
     plt.title('Boxplot of variable "Age"')

     plt.subplot(2,2,2)
     sns.boxplot(data = df.Occupation)
     plt.title('Boxplot of variable "Occupation"')

     plt.subplot(2,2,3)
     sns.boxplot(data = df.Product_Category)
     plt.title('Boxplot of variable "Product_Category"')

     plt.subplot(2,2,4)
     sns.boxplot(data = df.Purchase)
     plt.title('Boxplot of variable "Purchase"')
```

```
[ ]: Text(0.5, 1.0, 'Boxplot of variable "Purchase"')
```



Finding total number of outliers in each categorical varible by setting boundaries

[ ]:
```
Q1 = summary[['Product_Category', 'Purchase']].loc["25%"]
Q3 = summary[['Product_Category', 'Purchase']].loc["75%"]
IQR = Q3 - Q1
print(IQR)
```

```
Product_Category       7.0
Purchase            6231.0
dtype: float64
```

[ ]:
```
Lower_bound = Q1 - 1.5*IQR
Upper_bound = Q3 + 1.5*IQR

bounds_df = pd.DataFrame({"LowerBound" :Lower_bound, "UpperBound" :Upper_bound})
print(bounds_df)
```

```
                  LowerBound  UpperBound
Product_Category        -9.5        18.5
Purchase             -3523.5     21400.5
```

[ ]:
```
outliers_lower = (summary[['Product_Category', 'Purchase']] < Lower_bound).sum()
outliers_upper = (summary[['Product_Category', 'Purchase']] > Upper_bound).sum()
total_outliers = outliers_lower  + outliers_upper

ouliers_count_df = pd.DataFrame({"LowerBound_outliers" :outliers_lower,␣
 ↪"UpperBound_outliers" :outliers_upper, "Total" : total_outliers})
print(ouliers_count_df)
```

```
                  LowerBound_outliers  UpperBound_outliers  Total
Product_Category                    0                    2      2
Purchase                            0                    2      2
```

[ ]:
```
outlier_columns = ['Product_Category', 'Purchase']

for column in outlier_columns:
  threshold = df[column].quantile(0.95)
  df.loc[df[column] > threshold, column] = threshold
```

[ ]:
```
df.describe()
```

[ ]:
```
            User_ID     Occupation  Marital_Status  Product_Category  \
count  5.500680e+05  550068.000000   550068.000000     550068.000000
mean   1.003029e+06       8.076707        0.409653          5.242486
std    1.727592e+03       6.522660        0.491770          3.508509
```

```
min    1.000001e+06       0.000000         0.000000            1.000000
25%    1.001516e+06       2.000000         0.000000            1.000000
50%    1.003077e+06       7.000000         0.000000            5.000000
75%    1.004478e+06      14.000000         1.000000            8.000000
max    1.006040e+06      20.000000         1.000000           13.000000

             Purchase
count   550068.000000
mean      9217.804488
std       4918.694304
min         12.000000
25%       5823.000000
50%       8047.000000
75%      12054.000000
max      19336.000000
```

After imputing 95% data values in Product_category and purchases columns to ensure that the effect of outliers is reduced.

Finding Difference between mean and median

[ ]:
```python
difference_between_Mean_and_Median = (summary[['Product_Category', 'Purchase']].
 ↪loc["mean"] - summary[['Product_Category', 'Purchase']].loc["50%"])
difference_between_Mean_and_Median
```

[ ]:
```
Product_Category       0.404270
Purchase            1216.968713
dtype: float64
```

«————————————————————————————» «————————————————————————————»

«————————————————————————————» «————————————————————————————»

1. Count of Products categories grouped under different age bins

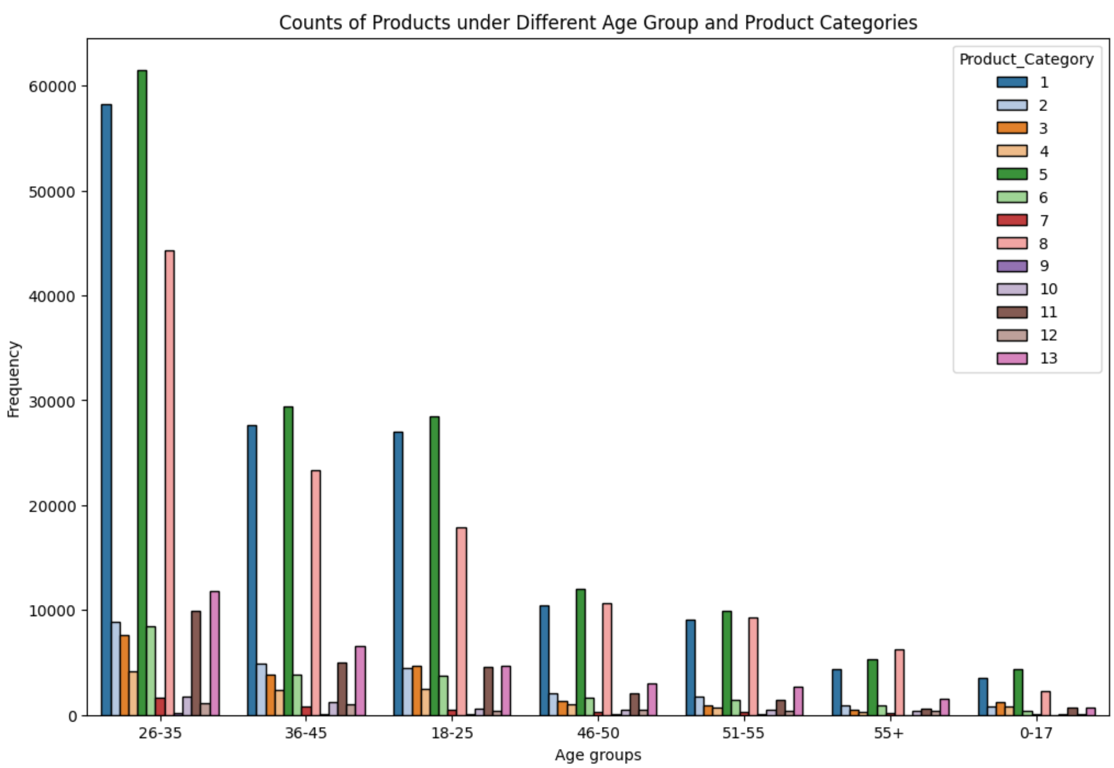
[ ]:
```python
count_of_products_under_agegroup = df.groupby(["Age", "Product_Category"]).
 ↪size()
count_of_products_under_agegroup = count_of_products_under_agegroup.
 ↪sort_values(ascending = False).reset_index(name = "Count")
count_of_products_under_agegroup
```

[ ]:

| | Age | Product_Category | Count |
|---|---|---|---|
| 0 | 26-35 | 5 | 61473 |
| 1 | 26-35 | 1 | 58249 |
| 2 | 26-35 | 8 | 44256 |
| 3 | 36-45 | 5 | 29377 |
| 4 | 18-25 | 5 | 28522 |

```
..     ...                  ...    ...
86    0-17                    7     53
87   46-50                    9     33
88   51-55                    9     29
89    0-17                    9     16
90     55+                    9      8

[91 rows x 3 columns]
```

[ ]:
```python
plt.figure(figsize = (12, 8))
sns.barplot(data = count_of_products_under_agegroup, x = "Age", y = "Count",
             hue = "Product_Category", palette = 'tab20', edgecolor='black')
plt.title("Counts of Products under Different Age Group and Product Categories")
plt.xlabel('Age groups')
plt.ylabel('Frequency')
plt.show()
```




**Inference:** In almost every age bin, category 5 tops the place in terms of purchase count. Along with category 5, we have category 1 and 8 with significant contributions in almost every age bin category.

**Recommendation:** For every age bin, it is highly suggested to target on product categories (1,5,8) combinedly to increase the demand further and improve the business performance.

«——————————————————————————————» «——————————————————————————————»

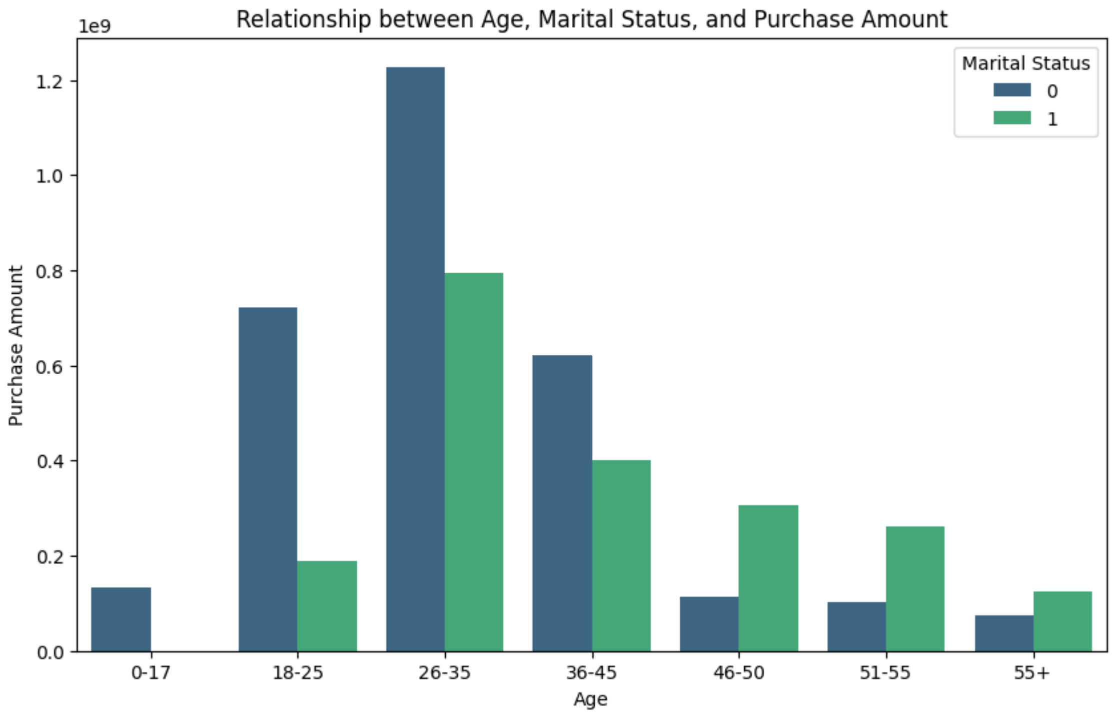
2. Relationship between age, marital status and amount spent

[ ]:
```
Total_Purchase_amount = df.groupby(["Age", "Marital_Status"])["Purchase"].sum()
Total_Purchase_amount = Total_Purchase_amount.sort_index(ascending = True).
↪reset_index()
Total_Purchase_amount
```

[ ]:

| | Age | Marital_Status | Purchase |
|---|---|---|---|
| 0 | 0-17 | 0 | 134278496 |
| 1 | 18-25 | 0 | 720943705 |
| 2 | 18-25 | 1 | 189097355 |
| 3 | 26-35 | 0 | 1227398110 |
| 4 | 26-35 | 1 | 794737279 |
| 5 | 36-45 | 0 | 620936056 |
| 6 | 36-45 | 1 | 400291030 |
| 7 | 46-50 | 0 | 113017885 |
| 8 | 46-50 | 1 | 305637092 |
| 9 | 51-55 | 0 | 103001031 |
| 10 | 51-55 | 1 | 261647408 |
| 11 | 55+ | 0 | 74711660 |
| 12 | 55+ | 1 | 124722172 |

[ ]:
```
plt.figure(figsize=(10, 6))
sns.barplot(data=Total_Purchase_amount, x='Age', y='Purchase',␣
↪hue='Marital_Status', palette='viridis')
plt.title('Relationship between Age, Marital Status, and Purchase Amount')
plt.xlabel('Age')
plt.ylabel('Purchase Amount')
plt.legend(title='Marital Status')
plt.show()
```




**Inference:** From this above graph we can infer that Non married people dominate the purchases below the age bin 45, and married couple does more purchases above the age bin 45.

Also from overall point of view, people between 18-45 age group does more purchases than rest of others. In particular people in the age group of 26-35 are actively purchasing more than any other age groups amounting to 39.87% of total purchases.

**Recommednation:** Focusing on 26-35 age groups yields better results for maintaining purchases demand and also majority of purchases are from people below 45 specifically belonging to non-married; Thus, targetting with some specific type of products or employing certain preferences which cater to them will increase the overall business turnover.

«————————————————» «————————————————»

3. Preferred product categories grouped based on gender

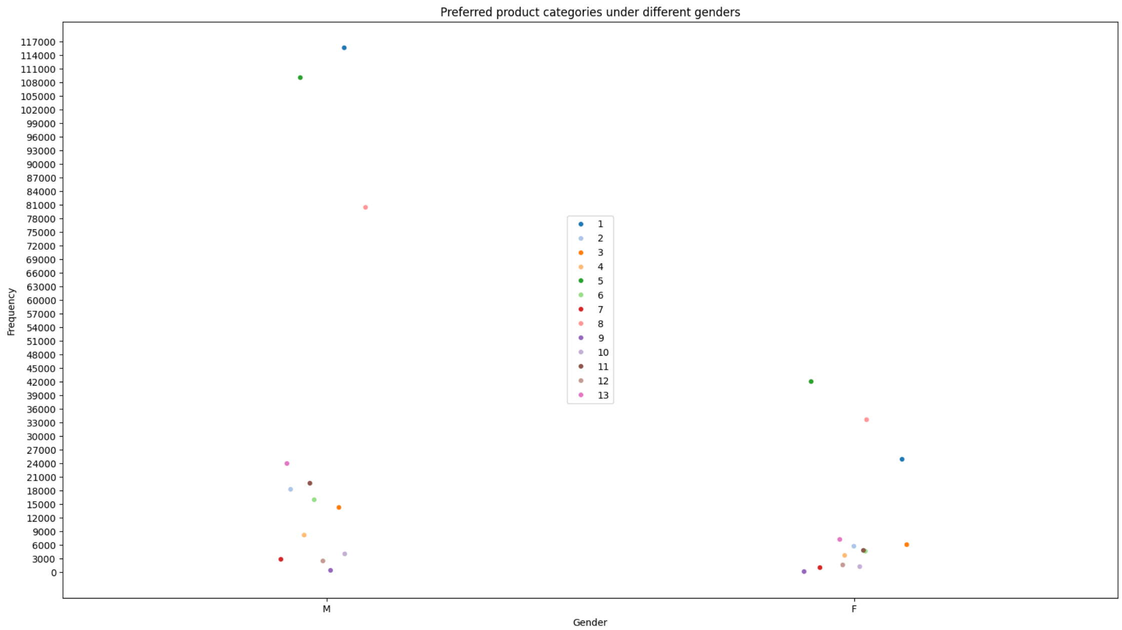
[ ]:
```
count_of_products_under_gender = df.groupby(["Gender", "Product_Category"]).size()
count_of_products_under_gender = count_of_products_under_gender.reset_index(name = 'Count').sort_values(by = ["Gender", "Product_Category"], ascending = [False, True])
count_of_products_under_gender
```

[ ]:

| | Gender | Product_Category | Count |
|---|---|---|---|
| 13 | M | 1 | 115547 |

```
14      M                   2   18206
15      M                   3   14207
16      M                   4    8114
17      M                   5  108972
18      M                   6   15907
19      M                   7    2778
20      M                   8   80367
21      M                   9     340
22      M                  10    3963
23      M                  11   19548
24      M                  12    2415
25      M                  13   23895
0       F                   1   24831
1       F                   2    5658
2       F                   3    6006
3       F                   4    3639
4       F                   5   41961
5       F                   6    4559
6       F                   7     943
7       F                   8   33558
8       F                   9      70
9       F                  10    1162
10      F                  11    4739
11      F                  12    1532
12      F                  13    7151
```

[ ]:
```
plt.figure(figsize = (20, 11))
sns.stripplot(data = count_of_products_under_gender, x = "Gender", y = "Count",
              hue = "Product_Category", palette = 'tab20', edgecolor='black')
plt.title("Preferred product categories under different genders")
plt.xlabel('Gender')
plt.ylabel('Frequency')
plt.yticks(range(0,120000,3000), fontsize = 10)
plt.legend(loc = 'center')
plt.show()
```




**Inference:** Based on the above graph we can infer that males(75.31% of Total products) domiante the purchases than females. In particular there are three specific categories(1,5,8) stood apart in both males(73.59% of Total Male products) and females(73.89% of Total Female products) purchasing history. Especially for males, both categories 1 and 5 crossed the mark of 100000 in total, whereas in females the highest sales stood below 42000 mark.

Rest of product category purchases in both males and females were below the mark of 21000, and majority of them were below 9000.

**Recommendation:** To increase overall sales, the company should focus more on males specifically from (1,5,8) categories. If there are proper strategies being installed in place to increase the demand of sales from males, then focus should also shift to females for the same categories.

It was best to decrease unwarranted expenditure on cluster of product categories below the sales of 9000.

«————————————————————————» «————————————————————————»

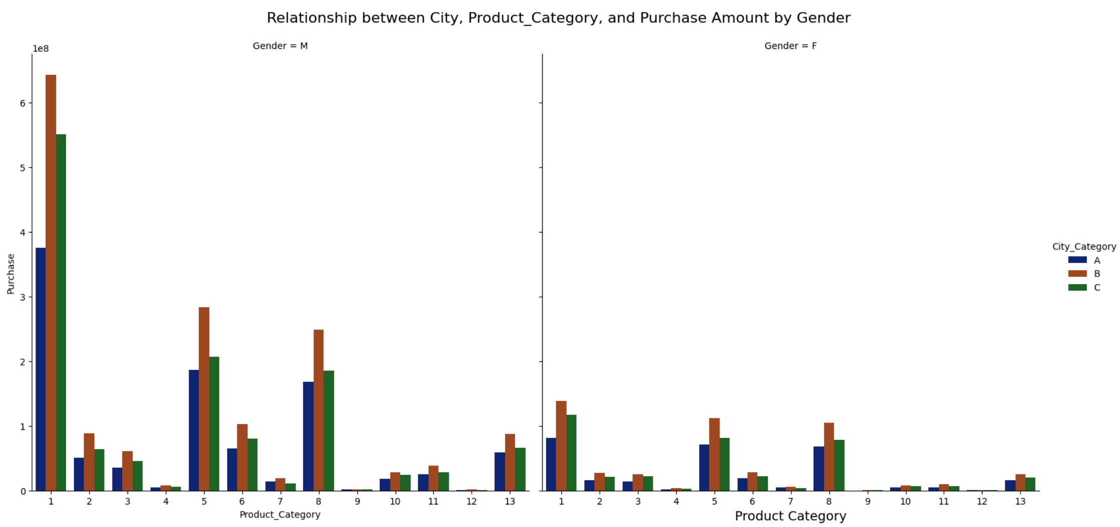
4. Relationship between Purchase amount, Gender, City_Category and Product Category

[ ]:
```
Amount_under_city_and_product = df.groupby(["Gender", "City_Category",␣
↪"Product_Category"])["Purchase"].sum()
Amount_under_city_and_product = Amount_under_city_and_product.reset_index().
↪sort_values(by=["Gender", "City_Category", "Product_Category"],␣
↪ascending=[False, True, True])
Amount_under_city_and_product
```

```
[ ]:    Gender City_Category  Product_Category   Purchase
     39      M             A                 1  375724153
     40      M             A                 2   50930747
     41      M             A                 3   35592296
     42      M             A                 4    4893956
     43      M             A                 5  186780661
     ..    ...           ...               ...        ...
     34      F             C                 9     320187
     35      F             C                10    7226216
     36      F             C                11    6924018
     37      F             C                12     766380
     38      F             C                13   19816800

     [78 rows x 4 columns]
```

```
[ ]: g = sns.catplot(data=Amount_under_city_and_product, kind="bar",␣
      ↪x="Product_Category", y="Purchase",
                     hue="City_Category", col="Gender", palette="dark", height=8,␣
      ↪aspect=1)

     plt.subplots_adjust(top=0.9)
     plt.suptitle("Relationship between City, Product_Category, and Purchase Amount␣
      ↪by Gender", fontsize=16)
     plt.xlabel("Product Category", fontsize=14)
     plt.ylabel("Purchase Amount", fontsize=14)
```

```
[ ]: Text(875.1887119622882, 0.5, 'Purchase Amount')
```




**Inference:** (Male) From the above graph we can infer that category B purchases(41.48% of total male purchase amount) stood top for almost every product category. And if we consider product categories in specific then (1,5,8) combined amounts to 57.67% of total male purchases.

(Female) from the above graph we can infer that again category B purchases(41.61% of Total female purchases) stood top fro almost every product category. And if we consider product categories in specific then (1,5,8) combined amounts to 71.99% of total female purchases.

**Recommendation:** Concentrating on **City_category B** for both male and femlaes and in specific, categories (1,5,8) combinedly will yield high business performance.

«——————————————————————» «——————————————————————»

5. Relationship between Purchase amount, Gender and Occupation

[ ]:
```
Amount_under_gender_occupation_product = df.groupby(["Gender",␣
↪"Occupation"])["Purchase"].sum()
Amount_under_gender_occupation_product = Amount_under_gender_occupation_product.
↪reset_index().sort_values(by=["Gender", "Occupation"], ascending=[False,␣
↪True])
Amount_under_gender_occupation_product
```

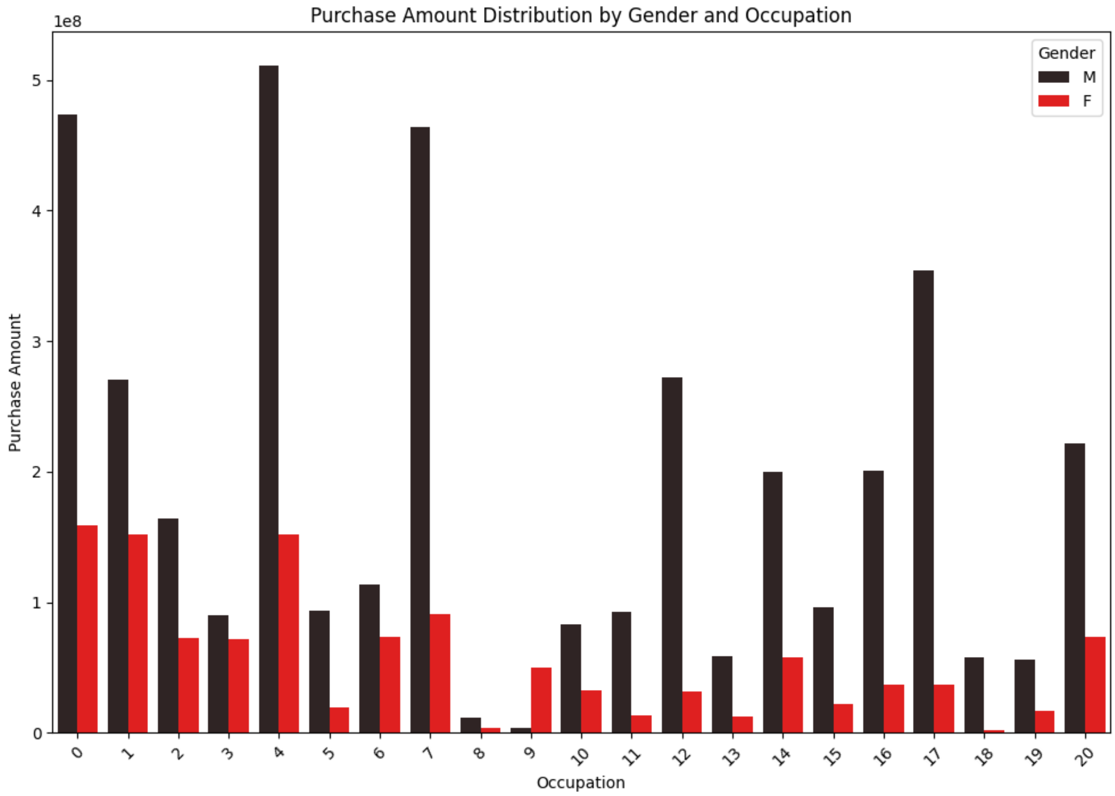
[ ]:

| | Gender | Occupation | Purchase |
|---|---|---|---|
| 21 | M | 0 | 473379475 |
| 22 | M | 1 | 270394178 |
| 23 | M | 2 | 164433706 |
| 24 | M | 3 | 89812488 |
| 25 | M | 4 | 511501375 |
| 26 | M | 5 | 93697409 |
| 27 | M | 6 | 113668118 |
| 28 | M | 7 | 463987090 |
| 29 | M | 8 | 11326625 |
| 30 | M | 9 | 4124239 |
| 31 | M | 10 | 82672282 |
| 32 | M | 11 | 92506063 |
| 33 | M | 12 | 272205077 |
| 34 | M | 13 | 58710622 |
| 35 | M | 14 | 200280243 |
| 36 | M | 15 | 95976961 |
| 37 | M | 16 | 200560053 |
| 38 | M | 17 | 354222595 |
| 39 | M | 18 | 58183215 |
| 40 | M | 19 | 56365774 |
| 41 | M | 20 | 221727691 |
| 0 | F | 0 | 158978001 |
| 1 | F | 1 | 152068617 |
| 2 | F | 2 | 72245910 |
| 3 | F | 3 | 71357286 |
| 4 | F | 4 | 151692403 |
| 5 | F | 5 | 19509624 |

```
6        F        6   73731207
7        F        7   90756272
8        F        8    3367766
9        F        9   50053905
10       F       10   32697825
11       F       11   13549650
12       F       12   31591065
13       F       13   12757780
14       F       14   57824257
15       F       15   22293976
16       F       16   36652520
17       F       17   37262111
18       F       18    2285084
19       F       19   16921100
20       F       20   73087641
```

[ ]:
```python
plt.figure(figsize=(12, 8))
sns.barplot(data=Amount_under_gender_occupation_product, x="Occupation",␣
 ↪y="Purchase", hue="Gender", color = "red")
plt.title("Purchase Amount Distribution by Gender and Occupation")
plt.xlabel("Occupation")
plt.ylabel("Purchase Amount")
plt.legend(title="Gender")
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-27-c93cc4ca4da3>:2: FutureWarning:

Setting a gradient palette using color= is deprecated and will be removed in
v0.14.0. Set `palette='dark:red'` for the same effect.

  sns.barplot(data=Amount_under_gender_occupation_product, x="Occupation",
y="Purchase", hue="Gender", color = "red")
```




**Inference:** (Male) From the above graph we can infer that occupations(0,1,4,7,12,14,16,17,20) combined have a Total Purchases of 76.30%

(Female) From the above graph we can infer that occupations(0,1,4,7,9,14,20) combined have a Total Purchases of 62.19%

**Recommednation:** It is observed that from both genders; Occupations like (0,1,4,7,14,20) have equally high percenatege contribution to purchases. Thus targetting these occupations in both genders yields good business returns.

«————————————————————————» «————————————————————————»

# 3 Calculating confidence Intervals

6. Affect of Gender affecting the purchases made

[ ]:
```
male_data = df[df["Gender"] == 'M']['Purchase']
female_data = df[df["Gender"] == 'F']['Purchase']

def bootstrap_CI(data, bootstrap_samples, alpha):
  boot_means = []
  for _ in range(bootstrap_samples):
```

```
        sample = np.random.choice(data, size = len(data), replace = True)
        boot_means.append(np.mean(sample))

    lower_bound = np.percentile(boot_means, 100 * alpha/2)
    upper_bound = np.percentile(boot_means, 100 * (1 - alpha / 2))
    return lower_bound, upper_bound

bootstrap_samples= 10000
alpha = 0.05
male_CI = bootstrap_CI(male_data, bootstrap_samples, alpha)
female_CI = bootstrap_CI(female_data, bootstrap_samples, alpha)

print("95% Confidence Interval for Males:", male_CI)
print("95% Confidence Interval for Females:", female_CI)
```

```
95% Confidence Interval for Males: (9374.36949263625, 9404.454772074958)
95% Confidence Interval for Females: (8669.137397742417, 8718.33644916758)
```

**Inference:** (Random samples drawn 10000 from entire data set considered as sample)

1. it can be concluded from the above observation of confidence intervals that there was no wider gap between intervals and infact the difference are very low in both females and males regarding their purcahses. Here, we can conclude that the mean calculated from the random 10000 samples from the entire dataset truly represents the population characteristics of the data.
2. As the sample size was entire dataset, the width of the intervals is quite low, but if the smaple size was been lower, we can observe that the width increases gradually to an extent.
3. There has been no evidence of overlapping of male and female samples of mean purchases; Thus, we can conclude that there was significant difference of purchasing behaviour between males and females.
4. As the sample size increases, the sampling distribution of the sample mean approaches a normal distribution, regardless of the shape of the population distribution, according to the Central Limit Theorem. This means that for sufficiently large sample sizes, the distribution of sample means becomes more symmetric and bell-shaped.
5. With larger samples the variability also decreases, this was because larger samples provide more information about the population, thus gap between the intervals gets reduced.

For smaller smaple sizes

[ ]:
```
def bootstrap_CI(data, bootstrap_samples, sample_size, alpha):
    boot_means = []
    for _ in range(bootstrap_samples):
        sample = np.random.choice(data, size=sample_size, replace=True)
        boot_means.append(np.mean(sample))

    lower_bound = np.percentile(boot_means, 100 * alpha / 2)
    upper_bound = np.percentile(boot_means, 100 * (1 - alpha / 2))
```

```python
    return lower_bound, upper_bound


sample_sizes = [300, 3000, 30000]
bootstrap_samples= 10000
alpha = 0.05

cis_data = []

# Calculate confidence intervals for each gender and sample size
for gender, gender_data in {'Male': male_data, 'Female': female_data}.items():
    print(f"Confidence Intervals for Gender: {gender}")
    for sample_size in sample_sizes:
        ci = bootstrap_CI(gender_data, bootstrap_samples, sample_size, alpha)
        print(f"Sample Size: {sample_size}, CI: {ci}")
        cis_data.append({
            'Sample Size': sample_size,
            'Gender': gender,
            'Lower Bound': ci[0],
            'Upper Bound': ci[1]
        })

print("<------------------------------------------------------->")
print("<------------------------------------------------------->")
cis_df = pd.DataFrame(cis_data)
print(cis_df)
```

```
Confidence Intervals for Gender: Male
Sample Size: 300, CI: (8835.426166666666, 9952.457916666666)
Sample Size: 3000, CI: (9209.9686, 9568.78335)
Sample Size: 30000, CI: (9332.532999166666, 9446.960659999999)
Confidence Intervals for Gender: Female
Sample Size: 300, CI: (8170.728499999999, 9234.30675)
Sample Size: 3000, CI: (8527.628349999999, 8858.382191666666)
Sample Size: 30000, CI: (8640.5189575, 8745.772165)
<------------------------------------------------------->
<------------------------------------------------------->
   Sample Size  Gender  Lower Bound  Upper Bound
0          300    Male  8835.426167  9952.457917
1         3000    Male  9209.968600  9568.783350
2        30000    Male  9332.532999  9446.960660
3          300  Female  8170.728500  9234.306750
4         3000  Female  8527.628350  8858.382192
5        30000  Female  8640.518958  8745.772165
```

Inference: (Randomly 10000 samples drawn for each sample size of 300,3000, 30000 from the dataset)

1. It can be concluded from the above observation of confidence intervals that the gap between intervals gradually got reduced as sample size increased from 300 to 30000. Infact the difference was very low in both females and males regarding their purcahses in highest sample size. With lower difference, we can conclude that the mean calculated from the random 30000 sample size truly represents the population characteristics of the data.

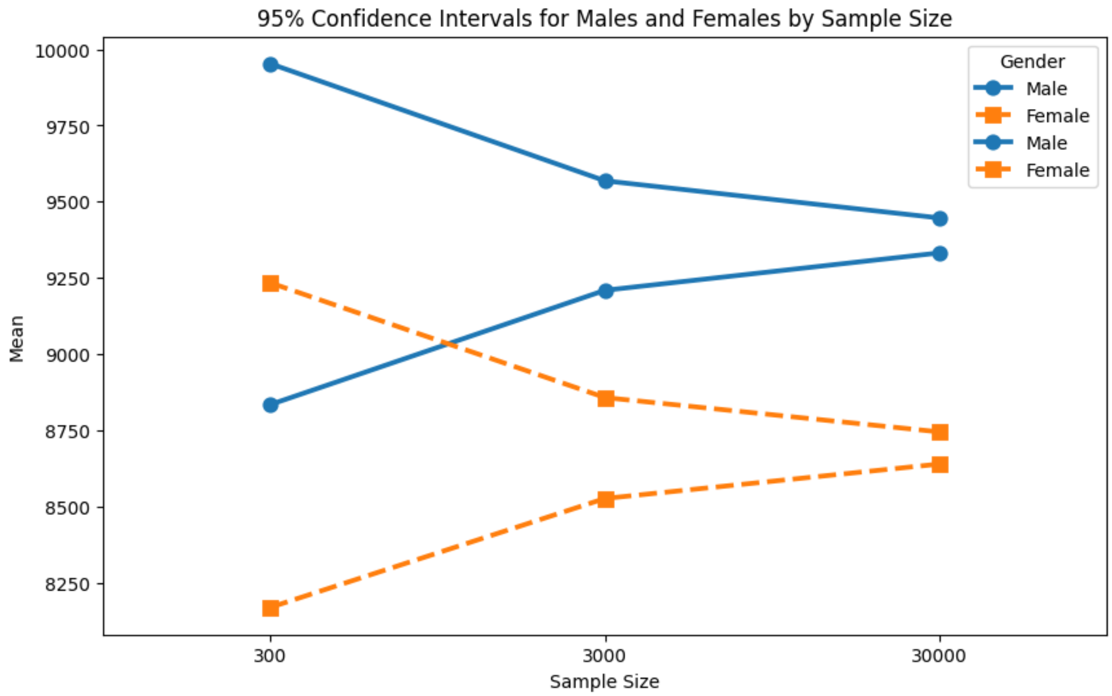
2. There has been evidence of overlapping of male and female samples of mean purchases when the sample size is 300, however when the size increased the overlapping diminished; Thus, we can conclude that there was significant difference of purchasing behaviour between males and females with a reliable sample size.

3. As the sample size increases, the sampling distribution of the sample mean approaches a normal distribution, regardless of the shape of the population distribution, according to the Central Limit Theorem. This means that for sufficiently large sample sizes, the distribution of sample means becomes more symmetric and bell-shaped.

4. With larger samples the variability also decreases, this was because larger samples provide more information about the population, thus gap between the intervals gets reduced.

Visual represenattion for sample sizes

```
[ ]: plt.figure(figsize=(10, 6))
     sns.pointplot(data=cis_df, x='Sample Size', y='Lower Bound', hue='Gender',␣
      ↪markers=['o', 's'], linestyles=['-', '--'])
     sns.pointplot(data=cis_df, x='Sample Size', y='Upper Bound', hue='Gender',␣
      ↪markers=['o', 's'], linestyles=['-', '--'])
     plt.title('95% Confidence Intervals for Males and Females by Sample Size')
     plt.xlabel('Sample Size')
     plt.ylabel('Mean')
     plt.legend(title='Gender')
     plt.show()
```




«——————————————————————» «——————————————————————»

7. Affect of marital status on Purcahses made

[ ]:
```
marital_data = df[df["Marital_Status"] == 1]['Purchase']
Non_marital_data = df[df["Marital_Status"] == 0]['Purchase']

def bootstrap_CI(data, bootstrap_samples, alpha):
  boot_means = []
  for _ in range(bootstrap_samples):
    sample = np.random.choice(data, size = len(data), replace = True)
    boot_means.append(np.mean(sample))

  lower_bound = np.percentile(boot_means, 100 * alpha/2)
  upper_bound = np.percentile(boot_means, 100 * (1 - alpha / 2))
  return lower_bound, upper_bound

bootstrap_samples= 10000
alpha = 0.05
married_CI = bootstrap_CI(marital_data, bootstrap_samples, alpha)
Non_married_CI = bootstrap_CI(Non_marital_data, bootstrap_samples, alpha)

print("95% Confidence Interval for Married:", married_CI)
print("95% Confidence Interval for Non-married:", Non_married_CI)
```

```
95% Confidence Interval for Married: (9193.552044271468, 9234.088529291683)
95% Confidence Interval for Non-married: (9203.916272699556, 9237.902054777647)
```

Inference: (Random samples drawn 10000 from entire data set considered as sample)

1. It can be concluded from the above observation of confidence intervals that there was no wider gap between intervals and infact the difference are very very low in both married and non-married regarding their purcahses. Here, we can conclude that the mean calculated from the random 10000 samples from the entire dataset truly represents the population characteristics of the data.
2. As the sample size was entire dataset, the width of the intervals is quite very low, but if the smaple size was been lower, we can observe that the width increases gradually to an extent.
3. There has been evidence of overlapping of married and non-married samples of mean purchases; Thus, we can conclude that there was no significant difference of purchasing behaviour between the two groups.
4. As the sample size increases, the sampling distribution of the sample mean approaches a normal distribution, regardless of the shape of the population distribution, according to the Central Limit Theorem. This means that for sufficiently large sample sizes, the distribution of sample means becomes more symmetric and bell-shaped.
5. With larger samples the variability also decreases, this was because larger samples provide more information about the population, thus gap between the intervals gets reduced.

For smaller sample sizes

[ ]:
```
def bootstrap_CI(data, bootstrap_samples, sample_size, alpha):
    boot_means = []
    for _ in range(bootstrap_samples):
        sample = np.random.choice(data, size=sample_size, replace=True)
        boot_means.append(np.mean(sample))

    lower_bound = np.percentile(boot_means, 100 * alpha / 2)
    upper_bound = np.percentile(boot_means, 100 * (1 - alpha / 2))
    return lower_bound, upper_bound


sample_sizes = [300, 3000, 30000]
bootstrap_samples= 10000
alpha = 0.05

cis_data = []

# Calculate confidence intervals for each marital status and sample size
for status, status_data in {'Married': marital_data, 'Non-married':
 ↪Non_marital_data}.items():
    print(f"Confidence Intervals for Marital Status: {status}")
    for sample_size in sample_sizes:
        ci = bootstrap_CI(status_data, bootstrap_samples, sample_size, alpha)
```

```python
        print(f"Sample Size: {sample_size}, CI: {ci}")
        cis_data.append({
            'Sample Size': sample_size,
            'Marital Status': status,
            'Lower Bound': ci[0],
            'Upper Bound': ci[1]
        })

print("<------------------------------------------------------->")
print("<------------------------------------------------------->")
cis_df = pd.DataFrame(cis_data)
print(cis_df)
```

```
Confidence Intervals for Marital Status: Married
Sample Size: 300, CI: (8661.559833333333, 9780.522)
Sample Size: 3000, CI: (9039.83935, 9386.649991666667)
Sample Size: 30000, CI: (9156.755615833334, 9268.643386666667)
Confidence Intervals for Marital Status: Non-married
Sample Size: 300, CI: (8660.696833333333, 9785.067833333334)
Sample Size: 3000, CI: (9044.380133333332, 9394.3863)
Sample Size: 30000, CI: (9164.962270833334, 9277.7094125)
<------------------------------------------------------->
<------------------------------------------------------->
   Sample Size Marital Status  Lower Bound  Upper Bound
0          300        Married  8661.559833  9780.522000
1         3000        Married  9039.839350  9386.649992
2        30000        Married  9156.755616  9268.643387
3          300    Non-married  8660.696833  9785.067833
4         3000    Non-married  9044.380133  9394.386300
5        30000    Non-married  9164.962271  9277.709413
```

**Inference:** (Randomly 10000 samples drawn for each sample size of 300,3000, 30000 from the dataset)

1. It can be concluded from the above observation of confidence intervals that there was no significant decrement of gap between intervals as sample size increased from 300 to 30000. Infact the difference were stable between both married and non-married for each sample size. But as sample size got to 30000, the interval was been at 9000 range.

2. There has been evidence of overlapping of married and non-married samples of mean purchases for all sample sizes of 300, 3000 and 30000. Thus, we can conclude that there was no significant difference of purchasing behaviour between the two groups.

3. As the sample size increases, the sampling distribution of the sample mean approaches a normal distribution, regardless of the shape of the population distribution, according to the Central Limit Theorem. This means that for sufficiently large sample sizes, the distribution of sample means becomes more symmetric and bell-shaped.

4. However here even with low sample size we observed very less difference between intervals, does it indicates the strong similarity of purchasing behaviour between married and non

married.

Visual represenattion for sample sizes

```
[ ]: plt.figure(figsize=(10, 6))
     sns.pointplot(data=cis_df, x='Sample Size', y='Lower Bound', hue='Marital␣
      ↪Status', markers=['o', 's'], linestyles=['-', '--'])
     sns.pointplot(data=cis_df, x='Sample Size', y='Upper Bound', hue='Marital␣
      ↪Status', markers=['o', 's'], linestyles=['-', '--'])
     plt.title('95% Confidence Intervals for Married and Non-married by Sample Size')
     plt.xlabel('Sample Size')
     plt.ylabel('Mean')
     plt.legend(title='Marital Status')
     plt.show()
```


«——————————————————————————————» «——————————————————————————————»

8. Affect of Age on Purchases made

```
[ ]: under_18 = df[df["Age"] == '0-17']["Purchase"]
     over_18_to_25 = df[df["Age"] == '18-25']["Purchase"]
     over_25_to_35 = df[df["Age"] == '26-35']["Purchase"]
     over_35_to_45 = df[df["Age"] == '36-45']["Purchase"]
     over_45_to_50 = df[df["Age"] == '46-50']["Purchase"]
     over_50_to_55 = df[df["Age"] == '51-55']["Purchase"]
```

```
over_55 = df[df["Age"] == '55+']["Purchase"]

def bootstrap_CI(data, bootstrap_samples, alpha):
  boot_means = []
  for _ in range(bootstrap_samples):
    sample = np.random.choice(data, size = len(data), replace = True)
    boot_means.append(np.mean(sample))

  lower_bound = np.percentile(boot_means, 100 * alpha/2)
  upper_bound = np.percentile(boot_means, 100 * (1 - alpha / 2))
  return lower_bound, upper_bound

bootstrap_samples= 10000
alpha = 0.05
under_18_CI = bootstrap_CI(under_18, bootstrap_samples, alpha)
over_18_to_25_CI = bootstrap_CI(over_18_to_25, bootstrap_samples, alpha)
over_25_to_35_CI = bootstrap_CI(over_25_to_35, bootstrap_samples, alpha)
over_35_to_45_CI = bootstrap_CI(over_35_to_45, bootstrap_samples, alpha)
over_45_to_50_CI = bootstrap_CI(over_45_to_50, bootstrap_samples, alpha)
over_50_to_55_CI = bootstrap_CI(over_50_to_55, bootstrap_samples, alpha)
over_55_CI =  bootstrap_CI(over_55, bootstrap_samples, alpha)

print("95% Confidence Interval for under_18:", under_18_CI)
print("95% Confidence Interval for over_18_to_25:", over_18_to_25_CI)
print("95% Confidence Interval for over_25_to_35:", over_25_to_35_CI)
print("95% Confidence Interval for over_35_to_45:", over_35_to_45_CI)
print("95% Confidence Interval for over_45_to_50:", over_45_to_50_CI)
print("95% Confidence Interval for over_50_to_55:", over_50_to_55_CI)
print("95% Confidence Interval for over_55:", over_55_CI)
```

```
95% Confidence Interval for under_18: (8813.31731062111, 8971.338127400344)
95% Confidence Interval for over_18_to_25: (9100.691309953843,
9162.240739263496)
95% Confidence Interval for over_25_to_35: (9188.641092596556, 9229.18980028417)
95% Confidence Interval for over_35_to_45: (9253.049399161919, 9311.43042322271)
95% Confidence Interval for over_45_to_50: (9116.58729458874, 9205.634808866327)
95% Confidence Interval for over_50_to_55: (9422.785987376952, 9520.41390808031)
95% Confidence Interval for over_55: (9209.167651367186, 9338.638732328869)
```

**Inference:** (Random samples drawn 10000 from entire data set considered as sample)

1. It can be concluded from the above observation of confidence intervals that there was no wider gap between intervals and infact the difference are very low in every age bin category regarding their purcahses. Here, we can conclude that the mean calculated from the random 10000 samples from the entire dataset truly represents the population characteristics of the data.

2. As the sample size is entire dataset, the width of the intervals are quite low, but if the smaple size was been lower, we can observe that the width increases gradually to an extent.

3. There has been evidence of overlapping of age bins between (over_18_to_25) with (over_45_to_50) and (over_25_to_35) with both(over_45_to_50) and (over_55) samples of mean purchases; Thus, we can conclude that there was no significant difference of purchasing behaviour between these bins.
4. As the sample size increases, the sampling distribution of the sample mean approaches a normal distribution, regardless of the shape of the population distribution, according to the Central Limit Theorem. This means that for sufficiently large sample sizes, the distribution of sample means becomes more symmetric and bell-shaped.
5. With larger samples the variability also decreases, this was because larger samples provide more information about the population, thus gap between the intervals gets reduced.

For smaller sample sizes

[ ]:
```
def bootstrap_CI(data, bootstrap_samples, sample_size, alpha):
    boot_means = []
    for _ in range(bootstrap_samples):
        sample = np.random.choice(data, size=sample_size, replace=True)
        boot_means.append(np.mean(sample))

    lower_bound = np.percentile(boot_means, 100 * alpha / 2)
    upper_bound = np.percentile(boot_means, 100 * (1 - alpha / 2))
    return lower_bound, upper_bound

sample_sizes = [300, 3000, 30000]
bootstrap_samples= 10000
alpha = 0.05


age_groups_data = {
    'under_18': under_18,
    'over_18_to_25': over_18_to_25,
    'over_25_to_35': over_25_to_35,
    'over_35_to_45': over_35_to_45,
    'over_45_to_50': over_45_to_50,
    'over_50_to_55': over_50_to_55,
    'over_55': over_55
}


cis_data = []

for age_group, age_group_data in age_groups_data.items():
    for sample_size in sample_sizes:
        ci = bootstrap_CI(age_group_data, bootstrap_samples, sample_size, alpha)
        cis_data.append({
                'Sample Size': sample_size,
                'Age Group': age_group,
```

```
                'Lower Bound': ci[0],
                'Upper Bound': ci[1]
            })

cis_df = pd.DataFrame(cis_data)
print(cis_df)
```

```
    Sample Size      Age Group  Lower Bound   Upper Bound
0           300       under_18  8331.935500   9459.562500
1          3000       under_18  8714.036658   9068.714517
2         30000       under_18  8834.967429   8947.613172
3           300  over_18_to_25  8581.007917   9696.558333
4          3000  over_18_to_25  8956.992208   9308.366075
5         30000  over_18_to_25  9075.507861   9188.217793
6           300  over_25_to_35  8655.133333   9777.028333
7          3000  over_25_to_35  9034.980017   9382.333333
8         30000  over_25_to_35  9153.729880   9264.736203
9           300  over_35_to_45  8736.419917   9851.515833
10         3000  over_35_to_45  9104.238617   9456.881025
11        30000  over_35_to_45  9228.040717   9338.715458
12          300  over_45_to_50  8623.320167   9718.114167
13         3000  over_45_to_50  8982.710492   9337.373058
14        30000  over_45_to_50  9105.793848   9215.685360
15          300  over_50_to_55  8915.240833  10044.546250
16         3000  over_50_to_55  9297.759358   9649.973008
17        30000  over_50_to_55  9415.847228   9527.896966
18          300        over_55  8737.047917   9827.718417
19         3000        over_55  9102.347033   9447.405333
20        30000        over_55  9218.979421   9328.384222
```

**Inference:** (Randomly 10000 samples drawn for each sample size of 300,3000, 30000 from the dataset)

1. It can be concluded from the above observation of confidence intervals that the gap between intervals gradually got reduced as sample size increased from 300 to 30000. Infact the difference was very low in all age bins regarding their purcahses in highest sample size. With lower difference, we can conclude that the mean calculated from the random 30000 sample size truly represents the population characteristics of the data.

2. There has been evidence of overlapping of certain age bins between over_35_to_45 with over_55 samples of mean purchases at every sample size. Ands Age bins of over_18_to_25, over_25_to_35 and over_45_to_50 interact with under_18, thus the there was less behavioural difference between these age bins.

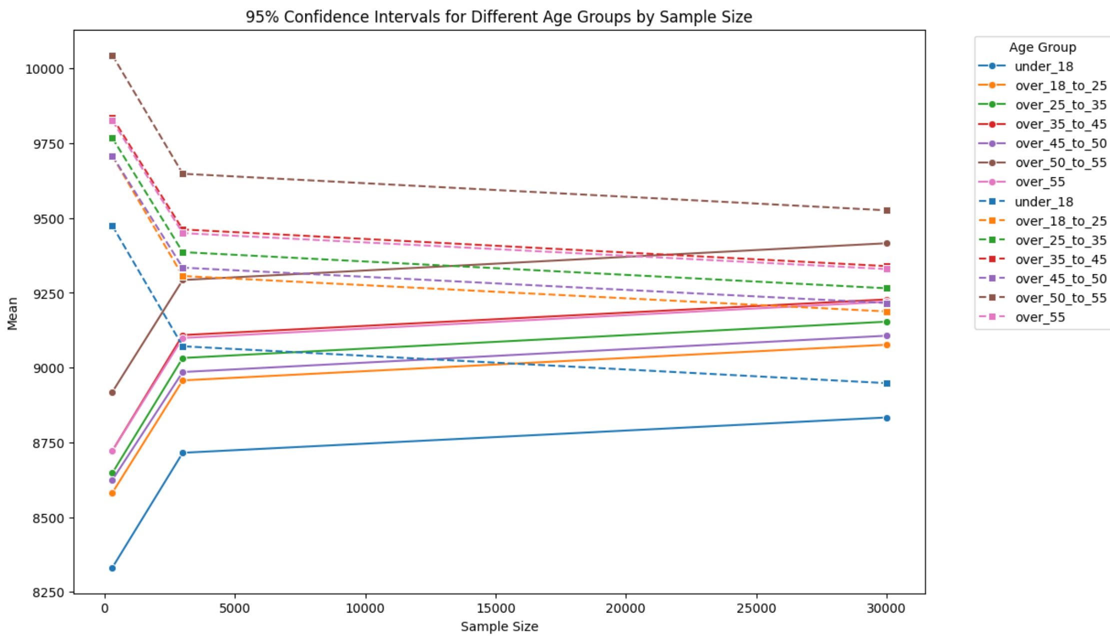
3. As the sample size increases, the sampling distribution of the sample mean approaches a normal distribution, regardless of the shape of the population distribution, according to the Central Limit Theorem. This means that for sufficiently large sample sizes, the distribution of sample means becomes more symmetric and bell-shaped.

4. With larger samples the variability also decreases, this was because larger samples provide

more information about the population, thus gap between the intervals gets reduced.

Visual represenattion for sample sizes

[ ]:
```
plt.figure(figsize=(12, 8))
sns.lineplot(data=cis_df, x='Sample Size', y='Lower Bound', hue='Age Group',␣
  ↪marker='o', linestyle='-')
sns.lineplot(data=cis_df, x='Sample Size', y='Upper Bound', hue='Age Group',␣
  ↪marker='s', linestyle='--')
plt.title('95% Confidence Intervals for Different Age Groups by Sample Size')
plt.xlabel('Sample Size')
plt.ylabel('Mean')
plt.legend(title='Age Group', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```




# 4 Conducting statistical tests to check the relationships between dependent and independent variables

[ ]:
```
from statsmodels.stats import weightstats as stests

male_purchases = df[df['Gender'] == 'M']['Purchase']
female_purchases = df[df['Gender'] == 'F']['Purchase']

alpha = 0.05
```

```
z_stat, p_value = stests.ztest(male_purchases, female_purchases, value = 0,␣
 ↪alternative = 'larger')
print("z_stat:", z_stat, ",", "p_value:", p_value)

print("<----------------------------------------->")

if p_value < alpha:
 print("reject the null hypothesis : There was higher mean purchases of males␣
 ↪than females.")

else:
 print("unable to reject null hypothesis : There was higher mean female␣
 ↪purchases than males.")
```

```
z_stat: 45.33211291666984 , p_value: 0.0
<----------------------------------------->
reject the null hypothesis : There was higher mean purchases of males than
females.
```

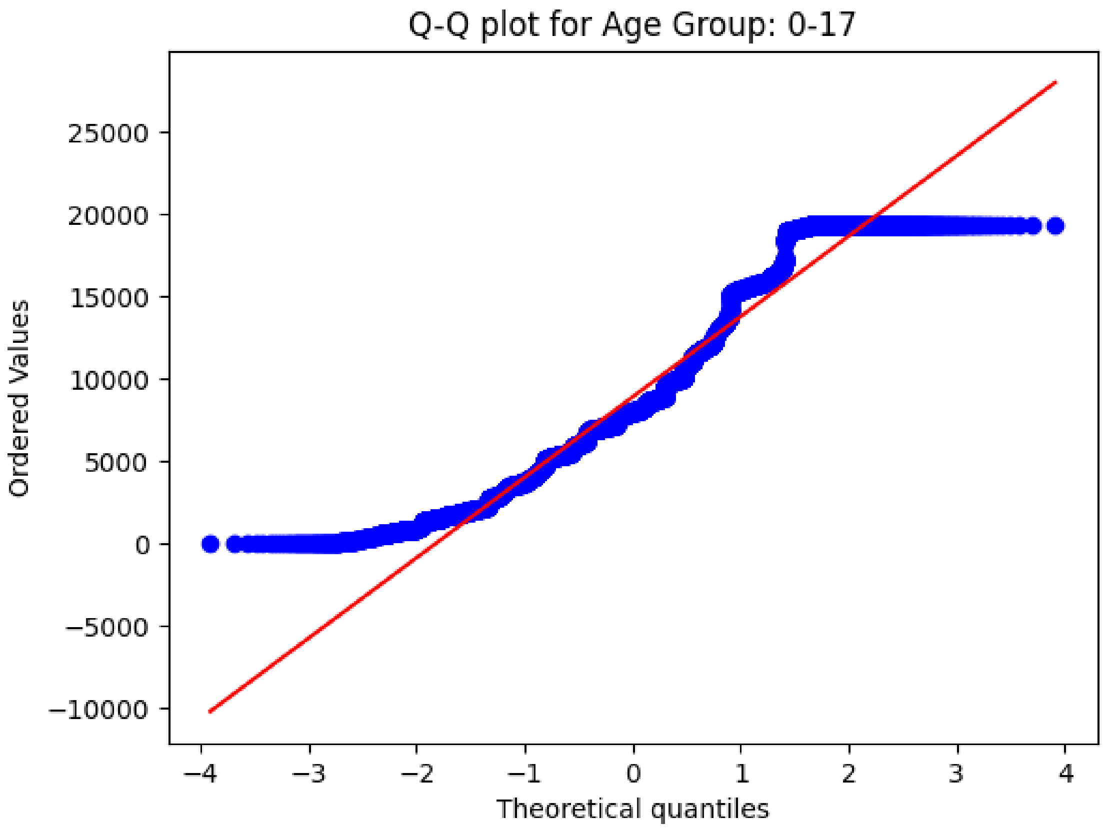
[ ]:
```
Non_married_purchases = df[df['Marital_Status'] == 0]['Purchase']
Married_purchases = df[df['Marital_Status'] == 1]['Purchase']



alpha = 0.05

z_stat, p_value = stests.ztest(Non_married_purchases , Married_purchases, value␣
 ↪= 0, alternative = 'larger')
print("z_stat:", z_stat, ",", "p_value:", p_value)

print("<----------------------------------------->")

if p_value < alpha:
 print("reject the null hypothesis : There was higher mean purchases of␣
 ↪non-married than married.")

else:
 print("unable to reject null hypothesis : There was higher mean married␣
 ↪purchases than non-married.")
```

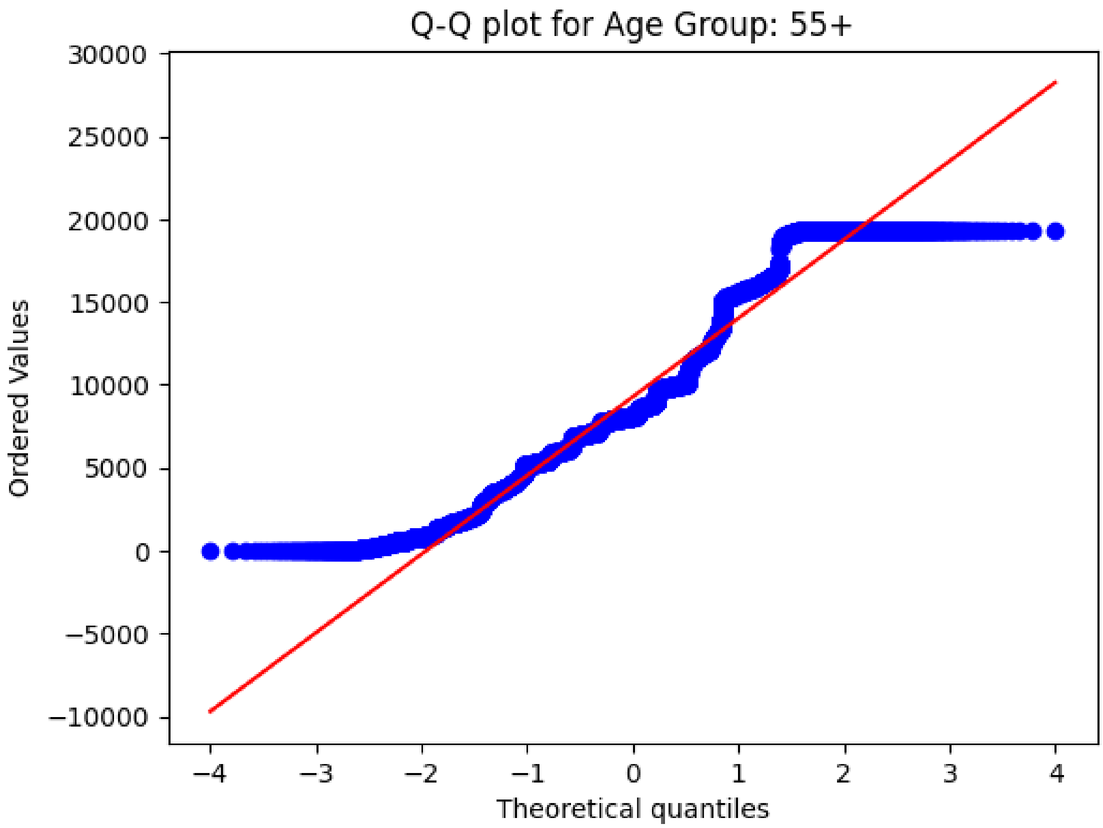
```
z_stat: 0.5463095124235151 , p_value: 0.2924265992847175
<----------------------------------------->
unable to reject null hypothesis : There was higher mean married purchases than
non-married.
```

[ ]:
```
import scipy.stats as stats
from scipy.stats import levene
```

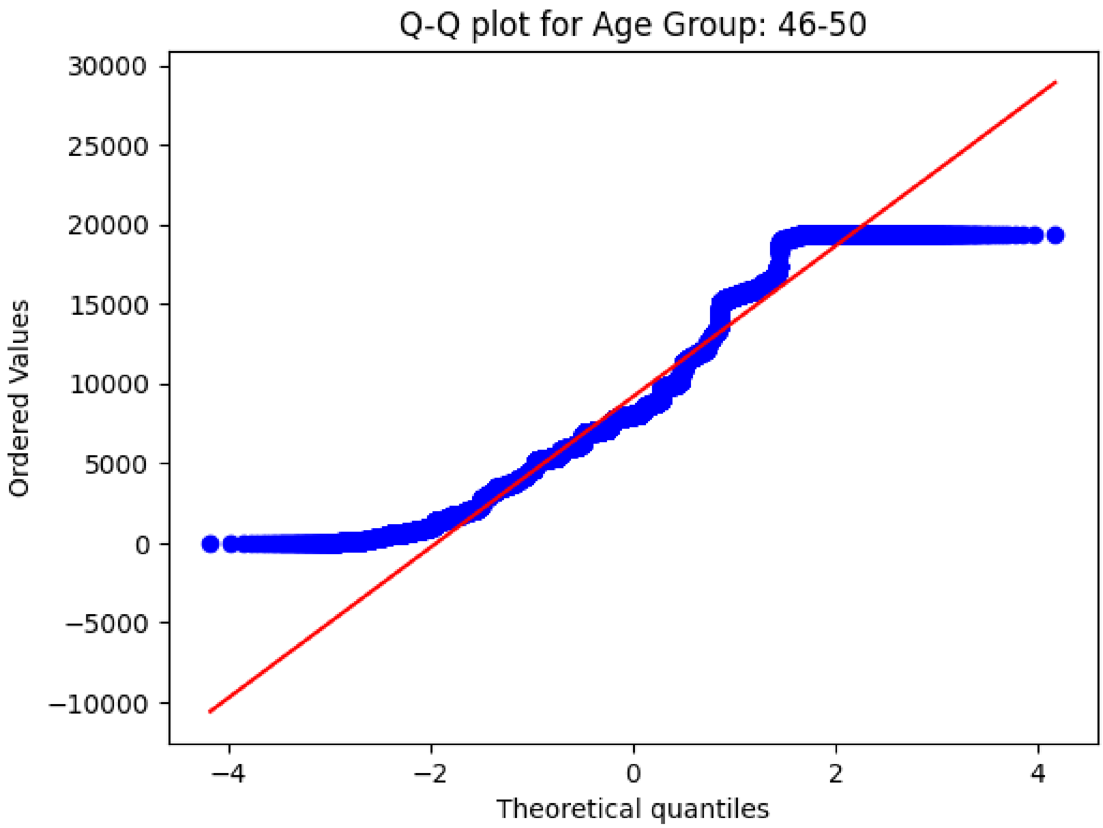
```python
## under qq-plot test,
 # H0: The data is normally distributed
 # H1: The data is not normally distributed
## under Brown-Forsythe Test,
 # H0: The variances are equal across age-groups
 # H1: The variances are different acorss age-groups

alpha = 0.05

age_groups = df["Age"].unique()
age_groups_purchases = [df[df["Age"] == age]["Purchase"] for age in age_groups]


print("<-------Normality assumption result------>")

normality_results = []

for age_group, purchases in zip(age_groups, age_groups_purchases):
    stats.probplot(purchases, dist="norm", plot=plt)
    plt.title(f'Q-Q plot for Age Group: {age_group}')
    plt.show()

# Box plots for variance inspection
plt.figure(figsize=(12, 6))
sns.boxplot(x='Age', y='Purchase', data=df)
plt.title('Box plot of Purchases by Age Group')
plt.show()

print("No normality within age groups, because the data points were not along␣
 ↪the straight line")

print("<------------------------------------------->")

l_stat, p_value = levene(*age_groups_purchases, center = 'median')

print(f"Brown-Forsythe Test: l_stats: {l_stat}, p_value: {p_value}")

print("<------------------------------------------->")

if p_value < alpha:
 print("reject the null hypothesis : There is no equality in variances within␣
 ↪age-groups purchases pattern")
else:
 print("unable to reject null hypothesis : There is equality in variances␣
 ↪within age-groups purchases pattern")
```

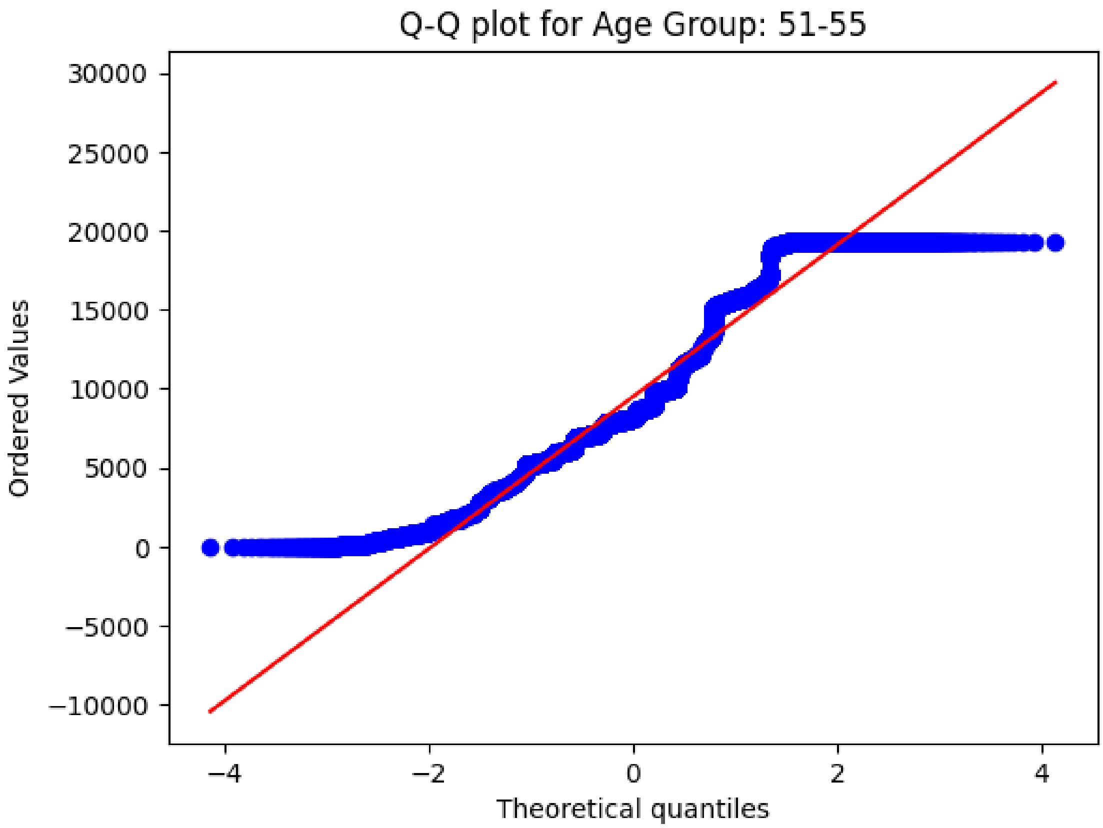
```
<-------Normality assumption result------>
```

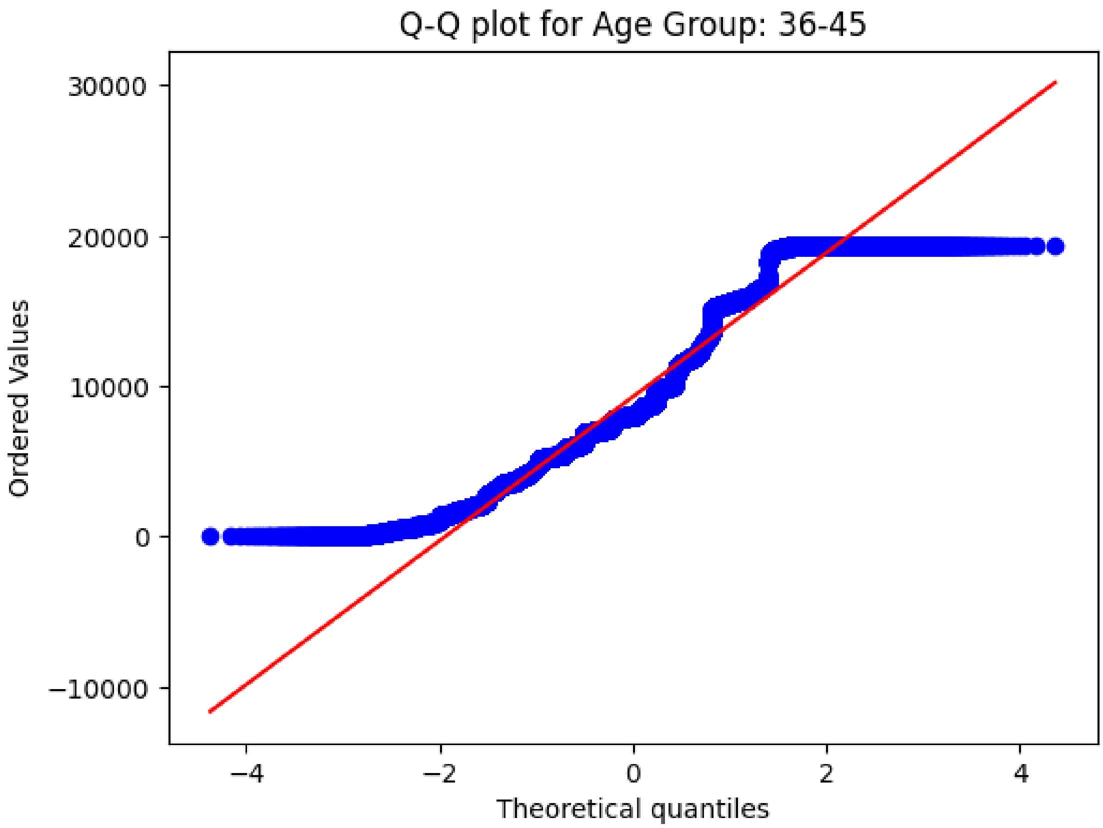

Q-Q plot for Age Group: 0-17
Ordered Values
Theoretical quantiles
25000
20000
15000
10000
5000
0
−5000
−10000
−4
−3
−2
−1
0
1
2
3
4

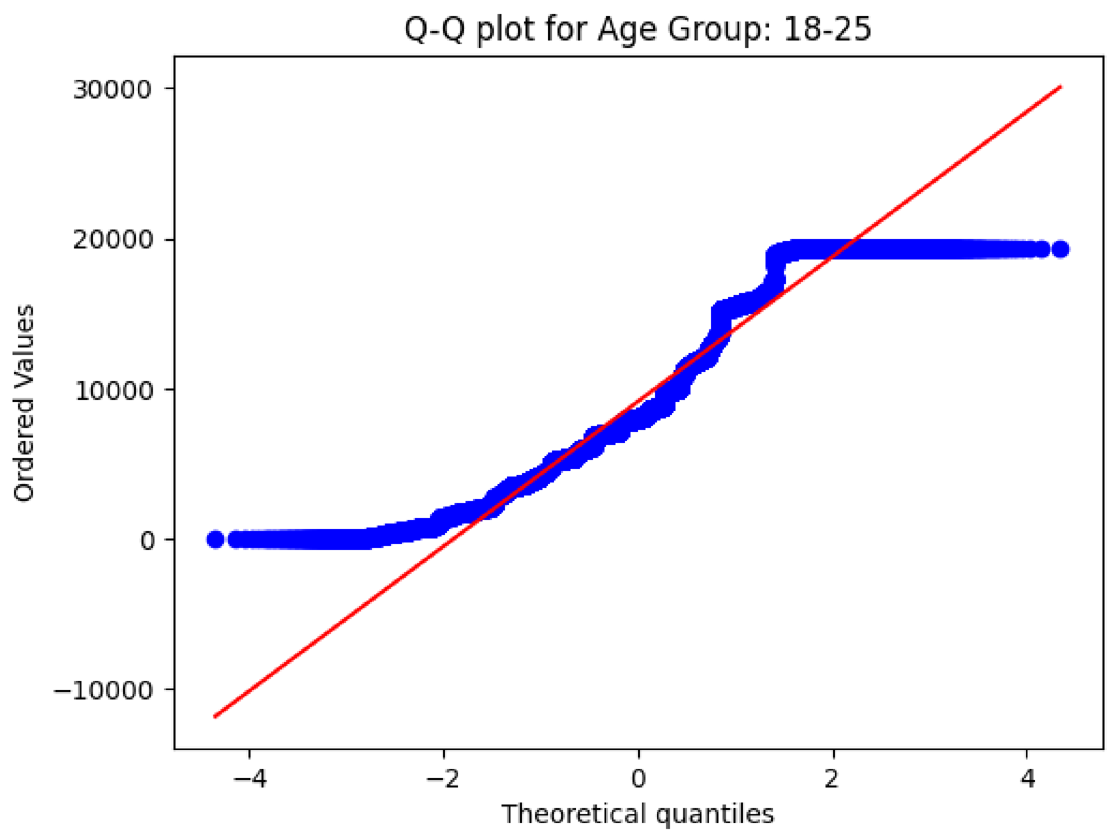

Q-Q plot for Age Group: 55+
30000
25000
20000
15000
10000
5000
0
−5000
−10000
Ordered Values
−4
−3
−2
−1
0
1
2
3
4
Theoretical quantiles

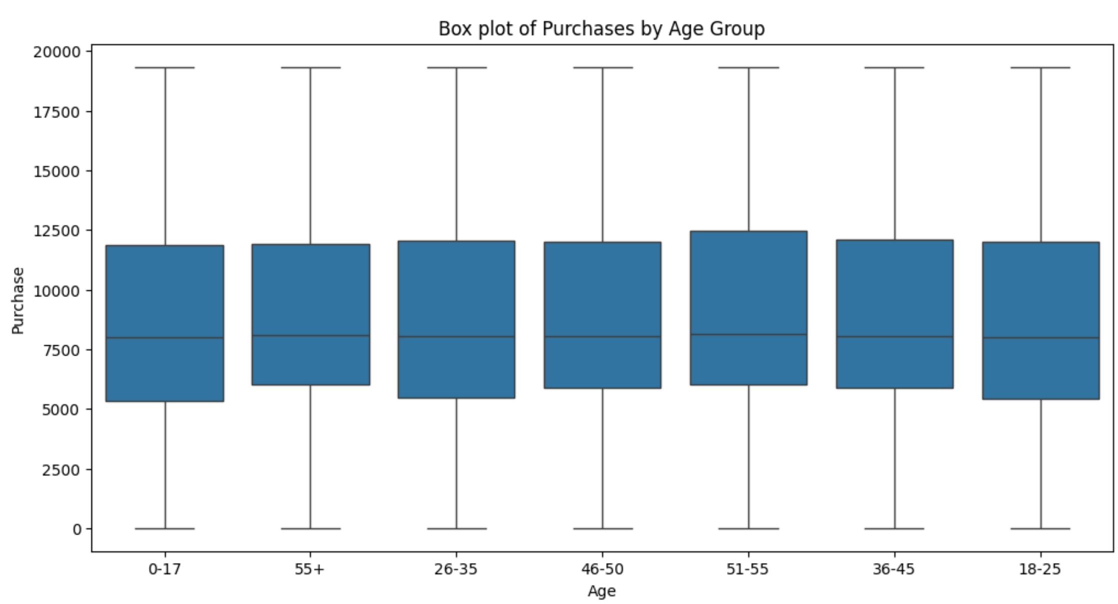

Q-Q plot for Age Group: 26-35
30000
20000
10000
0
−10000
Ordered Values
−4
−2
0
2
4
Theoretical quantiles


Q-Q plot for Age Group: 46-50
30000
25000
20000
15000
10000
5000
0
−5000
−10000
Ordered Values
−4
−2
0
2
4
Theoretical quantiles


Q-Q plot for Age Group: 51-55
Ordered Values
Theoretical quantiles
30000
25000
20000
15000
10000
5000
0
−5000
−10000
−4
−2
0
2
4


Q-Q plot for Age Group: 36-45
30000
20000
10000
0
−10000
Ordered Values
−4
−2
0
2
4
Theoretical quantiles




No normality within age groups, because the data points were not along the

```
straight line
<----------------------------------------->
Brown-Forsythe Test: l_stats: 14.018057268077934, p_value: 5.065333108187857e-16
<----------------------------------------->
reject the null hypothesis : There is no equality in variances within age-groups
purchases pattern
```

As normality and equality of variance assumptions were violated, i used kruskal to calculate the p_value.

[ ]:
```
from scipy.stats import kruskal

 # H0: The means of rental bikes usage of different seasons are equal
 # H1: The means of rental bikes usage of differnet seasons are not equal

alpha = 0.05


k_stat, p_value = kruskal(*age_groups_purchases)

print("k_stat :", k_stat, ",", "p_value :", p_value)

print("<----------------------------------------->")

if p_value < alpha:
 print("reject the null hypothesis : There is significant mean difference of␣
 ↪purchasing patterns between different age groups")
else:
 print("unable to reject null hypothesis : This is no mean difference of␣
 ↪purchasing patterns between different age groups")
```

```
k_stat : 311.5253430385636 , p_value : 2.7707941449758802e-64
<----------------------------------------->
reject the null hypothesis : There is significant mean difference of purchasing
patterns between different age groups
```

[ ]:
```
from scipy.stats import chi2_contingency

 # H0: There is independency between age and marital_status
 # H1: There is dependency between age and marital_status

 # Construct a contingency table

contingency_table = pd.crosstab(df.Age, df.Marital_Status)
print(contingency_table)

print("<---------------------------------->")
print("<---------------------------------->")
```

```
 # Perform chi-square test of independence

chi2, p_value, dof, expected_values = chi2_contingency(contingency_table)


# Print results
print("Chi-square statistic:", chi2)
print("p-value:", p_value)
print("Degrees of freedom:", dof)
print("Expected frequencies table:")
print(expected_values)
print("<---------------------------------->")


# Interpret the results
alpha = 0.05

if p_value < alpha:
 print("Reject the null hypothesis: There is a significant association between␣
 ↪Age and Marital status (dependency exists).")
else:
 print("Fail to reject the null hypothesis: There is no significant association␣
 ↪between Age and marital Status (independency exists).")
```

```
Marital_Status       0      1
Age
0-17             15102      0
18-25            78544  21116
26-35           133296  86291
36-45            66377  43636
46-50            12690  33011
51-55            10839  27662
55+               7883  13621
<---------------------------------->
<---------------------------------->
Chi-square statistic: 65038.31034963805
p-value: 0.0
Degrees of freedom: 6
Expected frequencies table:
[[  8915.42056982   6186.57943018]
 [ 58833.98318026  40826.01681974]
 [129632.52924548  89954.47075452]
 [ 64945.84579179  45067.15420821]
 [ 26979.44877906  18721.55122094]
 [ 22728.95029524  15772.04970476]
 [ 12694.82213835   8809.17786165]]
```

```
<-------------------------------->
Reject the null hypothesis: There is a significant association between Age and
Marital status (dependency exists).
```

[ ]:
```python
# H0: There is independency between age and Occupation
# H1: There is dependency between age and Occupation

 # Construct a contingency table

contingency_table = pd.crosstab(df.Occupation, df.Age )
print(contingency_table)

print("<-------------------------------->")
print("<-------------------------------->")

 # Perform chi-square test of independence

chi2, p_value, dof, expected_values = chi2_contingency(contingency_table)


# Print results
print("Chi-square statistic:", chi2)
print("p-value:", p_value)
print("Degrees of freedom:", dof)
print("Expected frequencies table:")
print(expected_values)
print("<-------------------------------->")


# Interpret the results
alpha = 0.05

if p_value < alpha:
 print("Reject the null hypothesis: There is a significant association between␣
 ↪Age and Occupation (dependency exists).")
else:
 print("Fail to reject the null hypothesis: There is no significant association␣
 ↪between Age and Occupation (independency exists).")
```

```
Age          0-17  18-25  26-35  36-45  46-50  51-55   55+
Occupation
0            2134   9095  34204  13393   4488   4602  1722
1             387   3820  19080   9501   7089   4410  3139
2             144   4364  12617   5183   2124   1344   812
3               0   1860   8159   4126   1599   1094   812
4             113  48241  21829   1747    129    249     0
5               0   1450   6082   3066   1187    377    15
6               0   1144   7216   4822   2561   3952   660
```

```
7             139   2078  24060  18762   6664   5355  2075
8              29     14    378     98    549    317   161
9               0    559   1489   3096    528    398   221
10          10951   1649     26    170      0      0   134
11             18    717   5009   2732   1584   1383   143
12            237   4585  15279   6848   2491   1417   322
13             15      0      0    427    631   1785  4870
14             93   4388  13446   5590   1445   1012  1335
15              0    906   6874   2585    854    514   432
16              0   1816   7070   7572   3032   3918  1963
17             35   3944  17064  10252   4662   2528  1558
18              0   1085   2243   1527   1124    531   112
19            807   2500   3468   1008    261    200   217
20              0   5445  13994   7508   2699   3115   801
<--------------------------------->
<--------------------------------->
Chi-square statistic: 593445.4632990315
p-value: 0.0
Degrees of freedom: 120
Expected frequencies table:
[[ 1911.89648553 12616.84569908 27799.4711672  13927.52404066
   5785.6960194   4874.18398816  2722.38259997]
 [ 1302.07074762  8592.52885098 18932.4466466   9485.14826894
   3940.26852316  3319.4958187   1854.041144  ]
 [  729.96788761  4817.15002509 10613.9225623   5317.57099849
   2208.99632045  1860.97825723  1039.41394882]
 [  484.57699775  3197.78463754  7045.87532814  3529.98074783
   1466.40533534  1235.3793531    689.9976003 ]
 [ 1985.20076791 13100.5898907  28865.33446047 14461.5211283
   6007.52617495  5061.06573733  2826.76184035]
 [  334.31694627  2206.1996335   4861.05517681  2435.38671764
   1011.6950577    852.30676389   476.03970418]
 [  558.84219769  3687.87004516  8125.71061214  4070.97779729
   1691.14337682  1424.71086302   795.74510788]
 [ 1623.4839438  10713.57501254 23605.87794782 11826.53549925
   4912.91482689  4138.90579528  2311.70697441]
 [   42.44510133   280.10056938   617.16279078   309.19831366
    128.44547583   108.20943229    60.43831672]
 [  172.71806758  1139.78828072  2511.36553481  1258.19313794
    522.67172604   440.32699775   245.93625515]
 [  354.99040119  2342.62636619  5161.6525775   2585.98589629
   1074.25614651   905.01161675   505.47699557]
 [  318.09116691  2099.12367198  4625.12813325  2317.18736229
    962.59332664   810.94080368   452.93553524]
 [  856.0128166   5648.93638605 12446.64854709  6235.76599075
   2590.42787255  2182.31687537  1218.89151159]
 [  212.17059709  1400.14049172  3085.01555444  1545.5915705
    642.06121425   540.90717511   302.11339689]
```

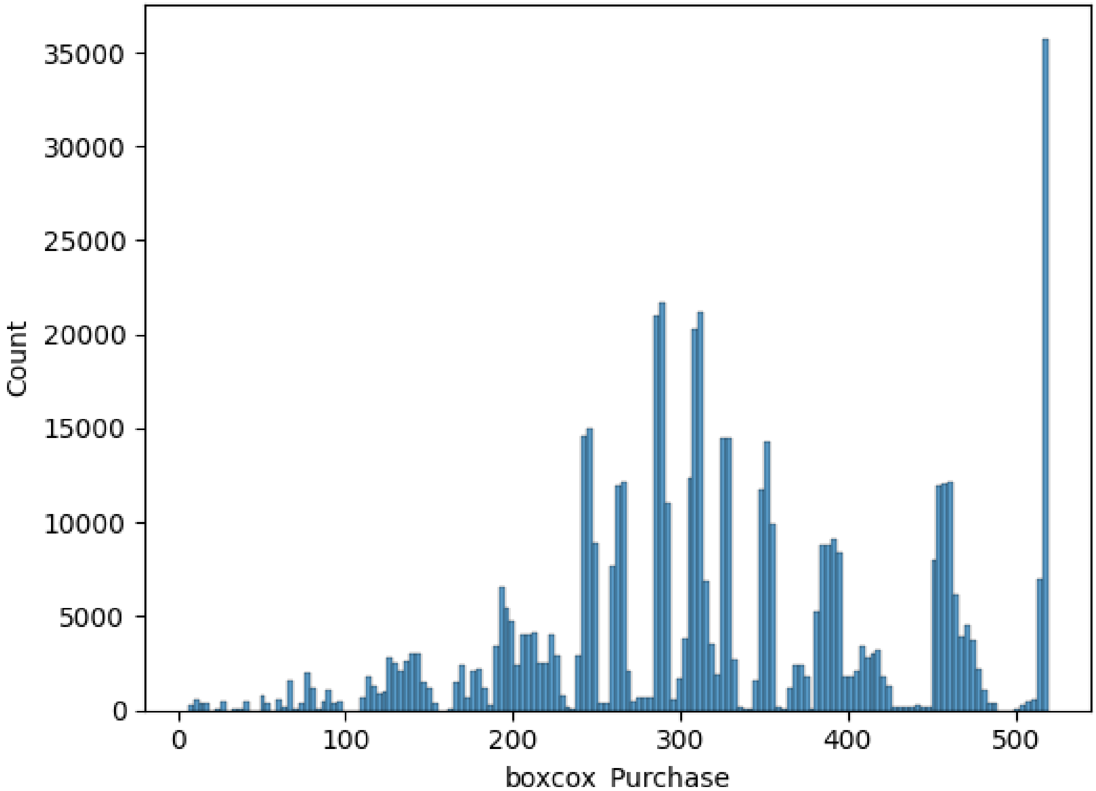
```
 [  749.76278933  4947.77907459 10901.74557146  5461.77021205
   2268.8987707   1911.44332882  1067.60025306]
 [  333.98748882  2204.0255023   4856.26477999  2432.98673073
   1010.69806824   851.46684592   475.570584  ]
 [  696.55541133  4596.65688606 10128.09648443  5074.17232597
   2107.8849724   1775.79657606   991.83734375]
 [ 1099.37205218  7254.89463121 15985.15500084  8008.55632213
   3326.87075598  2802.73628533  1565.41495233]
 [  181.80560222  1199.75806628  2643.50064719  1324.39277689
    550.17201873   463.4947352    258.87615349]
 [  232.29495626  1532.94367242  3377.62896042  1692.19077096
    702.96065396   592.21216468   330.76882131]
 [  921.43757499  6080.68260651 13397.94151632  6712.36339144
   2788.41336344  2349.11058633  1312.05096097]]
<-------------------------------->
Reject the null hypothesis: There is a significant association between Age and
Occupation (dependency exists).
```

# 5 Building Linear regression model to establish relationships

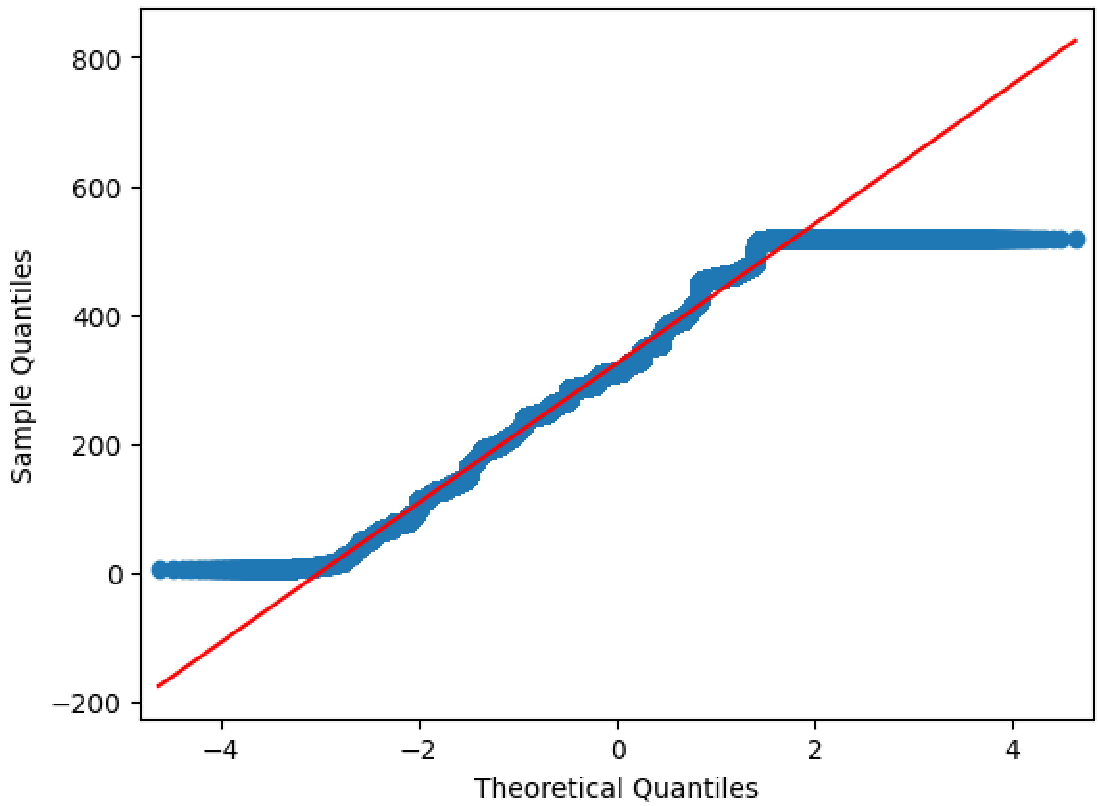
[ ]:
```python
import statsmodels.api as sm
from statsmodels.formula.api import ols, glm
```

[ ]:
```python
from scipy.stats import boxcox
df['boxcox_Purchase'], _ = boxcox(df['Purchase'] + 1)

sns.histplot(x = 'boxcox_Purchase', data = df)
```

[ ]: `<Axes: xlabel='boxcox_Purchase', ylabel='Count'>`




[ ]:
```
fig = sm.qqplot(df['boxcox_Purchase'], line = 's')
plt.show()
```




[ ]:
```
df_encoded = pd.get_dummies(df, columns=['Marital_Status', 'Age', 'Gender'],␣
  ↪drop_first=True)
X = df_encoded.drop(columns=['Purchase', 'User_ID', 'Product_ID', 'Occupation',␣
  ↪'City_Category', 'Stay_In_Current_City_Years', 'Product_Category',␣
  ↪'boxcox_Purchase'])
y = df['boxcox_Purchase']
```

[ ]:
```
X
```

[ ]:
```
        Marital_Status_1  Age_18-25  Age_26-35  Age_36-45  Age_46-50  \
0                  False      False      False      False      False
1                  False      False      False      False      False
2                  False      False      False      False      False
3                  False      False      False      False      False
4                  False      False      False      False      False
…                      …          …          …          …          …
550063              True      False      False      False      False
550064             False      False       True      False      False
550065              True      False       True      False      False
550066             False      False      False      False      False
550067              True      False      False      False       True
```

```
        Age_51-55  Age_55+  Gender_M
0           False    False     False
1           False    False     False
2           False    False     False
3           False    False     False
4           False     True      True
…             …      …       …
550063       True    False      True
550064      False    False     False
550065      False    False     False
550066      False     True     False
550067      False    False     False

[550068 rows x 8 columns]
```

[ ]:
```
X.corr(method = 'spearman')
```

[ ]:
```
                  Marital_Status_1  Age_18-25  Age_26-35  Age_36-45  \
Marital_Status_1          1.000000  -0.189174  -0.027654  -0.013227
Age_18-25                -0.189174   1.000000  -0.383431  -0.235194
Age_26-35                -0.027654  -0.383431   1.000000  -0.407567
Age_36-45                -0.013227  -0.235194  -0.407567   1.000000
Age_46-50                 0.191389  -0.141595  -0.245369  -0.150507
Age_51-55                 0.172278  -0.129045  -0.223622  -0.137168
Age_55+                   0.091778  -0.094879  -0.164415  -0.100851
Gender_M                 -0.011603  -0.000246   0.029811  -0.000088

                  Age_46-50  Age_51-55   Age_55+  Gender_M
Marital_Status_1   0.191389   0.172278  0.091778 -0.011603
Age_18-25         -0.141595  -0.129045 -0.094879 -0.000246
Age_26-35         -0.245369  -0.223622 -0.164415  0.029811
Age_36-45         -0.150507  -0.137168 -0.100851 -0.000088
Age_46-50          1.000000  -0.082580 -0.060716 -0.029262
Age_51-55         -0.082580   1.000000 -0.055334 -0.006416
Age_55+           -0.060716  -0.055334  1.000000  0.004921
Gender_M          -0.029262  -0.006416  0.004921  1.000000
```

[ ]:
```
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Calculate the variance inflation factor for each variable.
vif = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]

# Create a DataFrame with the VIF results for the column names in X.
df_vif = pd.DataFrame(vif, index=X.columns, columns = ['VIF'])

# Display the VIF results.
```

```
df_vif
```

[ ]:

```
                        VIF
Marital_Status_1   0.000009
Age_18-25          0.000016
Age_26-35          0.000011
Age_36-45          0.000016
Age_46-50          0.000031
Age_51-55          0.000036
Age_55+            0.000055
Gender_M           0.000009
```

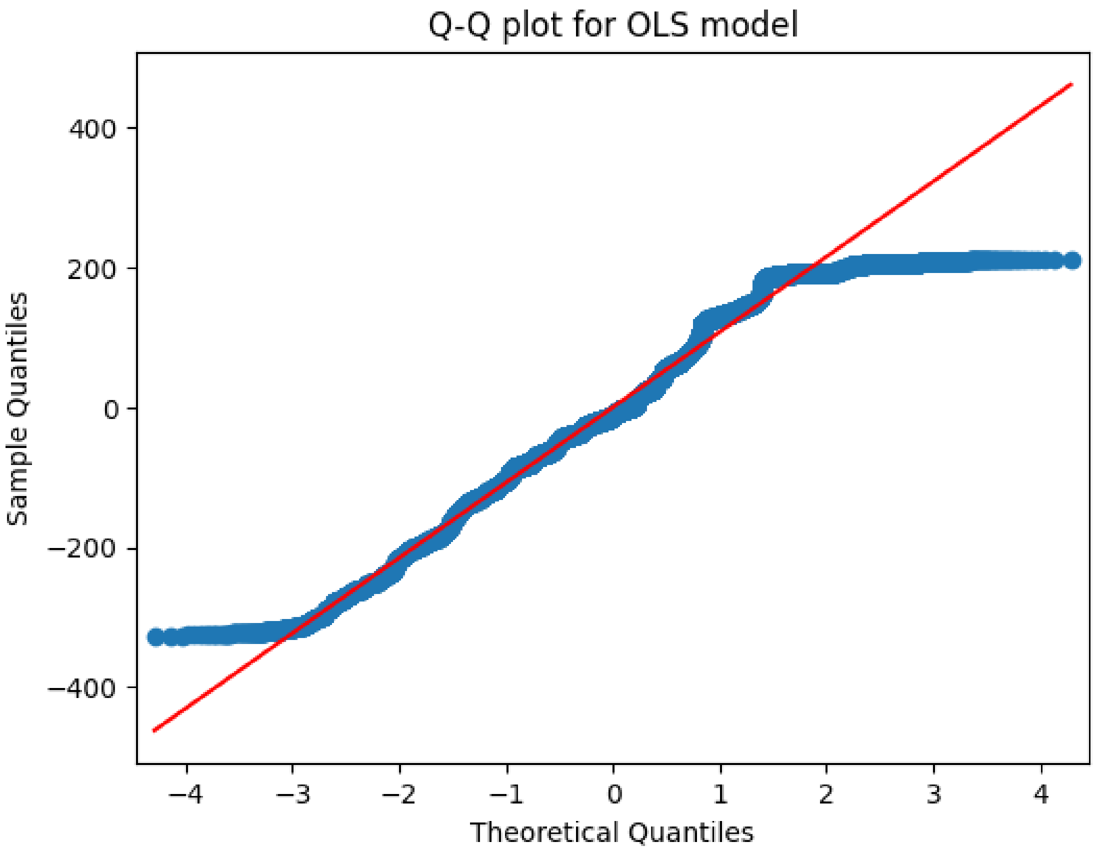
[ ]:

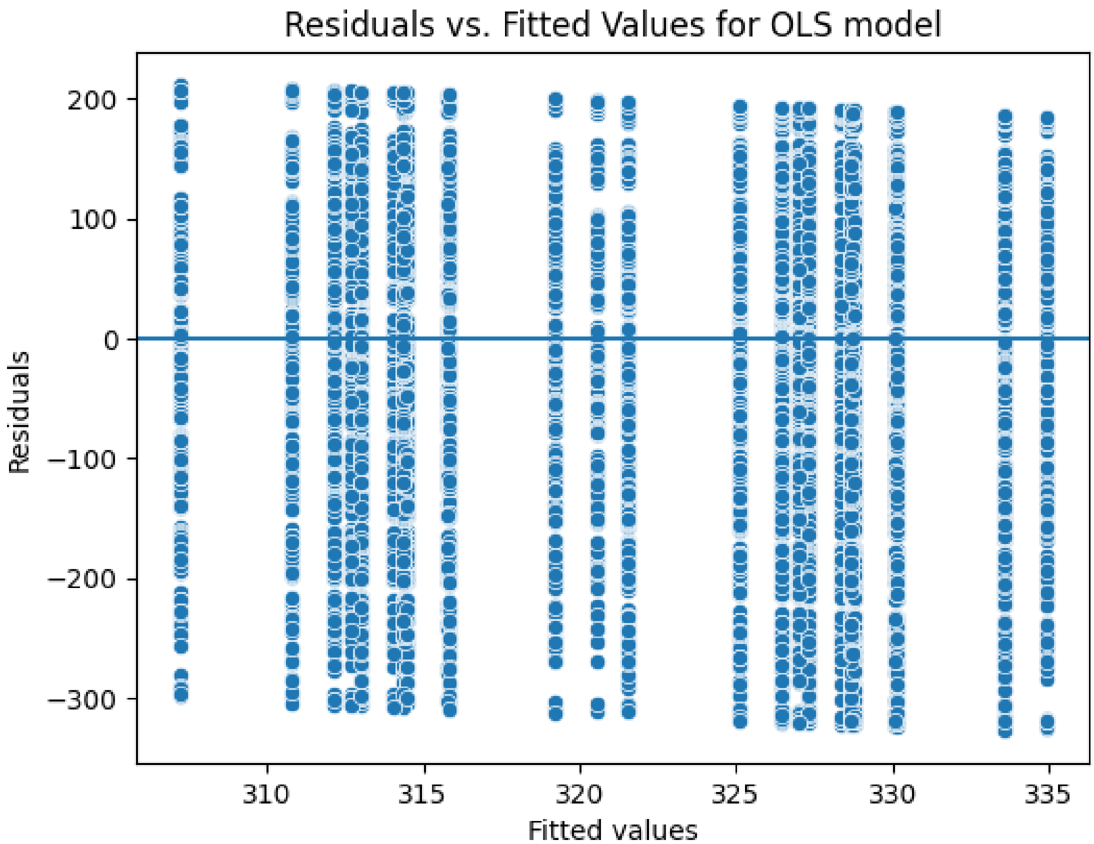
```
from sklearn.linear_model import Ridge, Lasso, LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

ols = LinearRegression()
ols.fit(X_train, y_train)
y_pred_ols = ols.predict(X_test)
print("(OLS Regression)")
print("Coefficients:", ols.coef_)
print("Intercept:", ols.intercept_)
print("Mean Squared Error:", mean_squared_error(y_test, y_pred_ols))
print("R2 Score:", r2_score(y_test, y_pred_ols))
print()

ridge = Ridge(alpha=1.0)
ridge.fit(X_train, y_train)
y_pred_ridge = ridge.predict(X_test)
print("(Ridge Regression)")
print("Coefficients:", ridge.coef_)
print("Intercept:", ridge.intercept_)
print("Mean Squared Error:", mean_squared_error(y_test, y_pred_ridge))
print("R2 Score:", r2_score(y_test, y_pred_ridge))
print()

lasso = Lasso(alpha=1.0)
lasso.fit(X_train, y_train)
y_pred_lasso = lasso.predict(X_test)
```

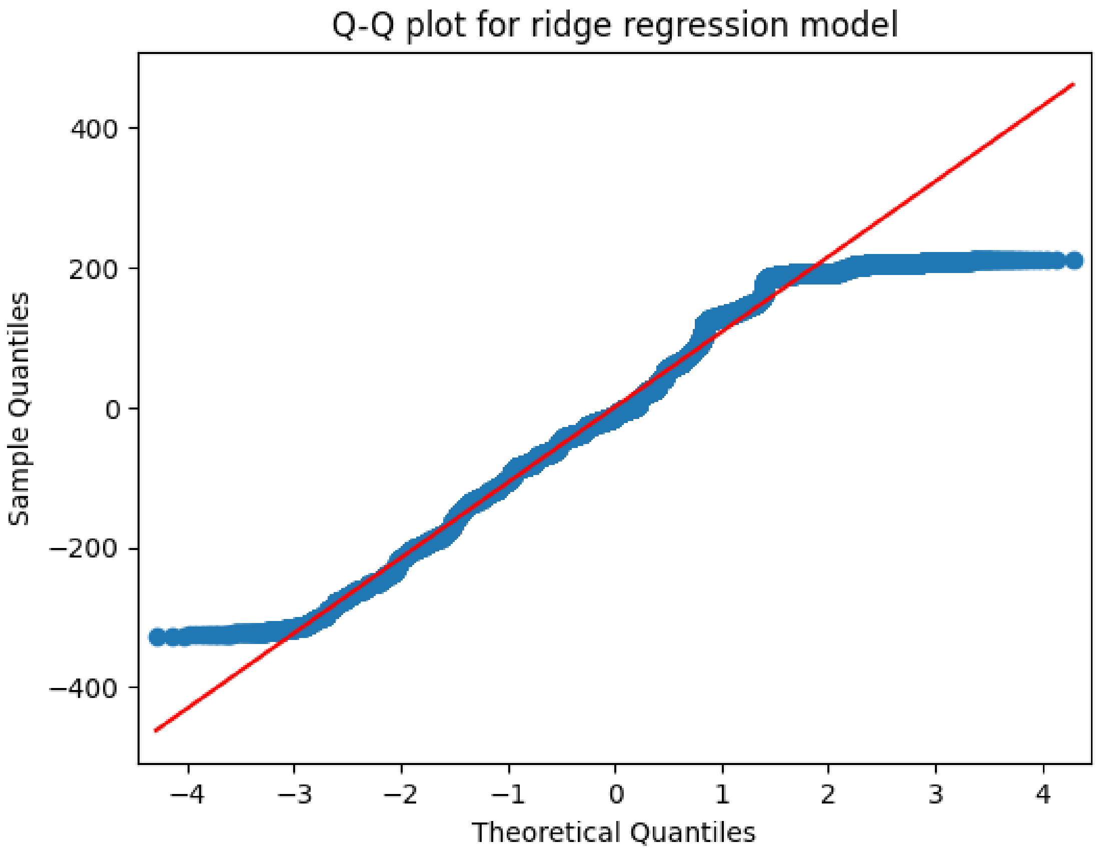
```python
print("(Lasso Regression)")
print("Coefficients:", lasso.coef_)
print("Intercept:", lasso.intercept_)
print("Mean Squared Error:", mean_squared_error(y_test, y_pred_lasso))
print("R2 Score:", r2_score(y_test, y_pred_lasso))
```

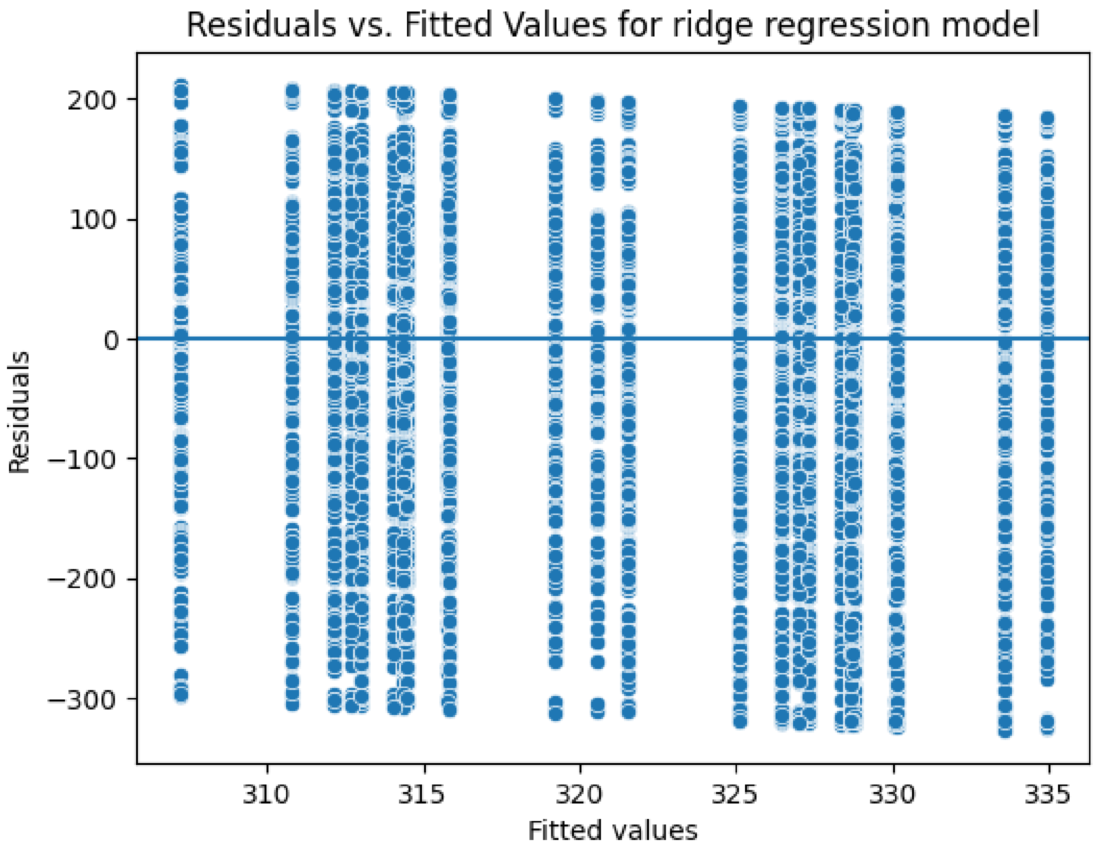
```
(OLS Regression)
Coefficients: [-0.66151696  1.88490466  3.47854389  3.43751237  1.87956537
3.39884185
  1.65554822  6.17369049]
Intercept: 324.74590811718497
Mean Squared Error: 11590.76733722138
R2 Score: 0.0036023864558111124

(Ridge Regression)
Coefficients: [-0.66150191  1.88472475  3.47831229  3.43732058  1.87943041
3.39871397
  1.65545329  6.1736792 ]
Intercept: 324.74590811718497
Mean Squared Error: 11590.767382107771
R2 Score: 0.0036023825971628343

(Lasso Regression)
Coefficients: [-0.         -0.          0.          0.         -0.
0.54219623
  0.          5.21889183]
Intercept: 324.74590811718497
Mean Squared Error: 11595.64956499867
R2 Score: 0.0031826868824749166
```

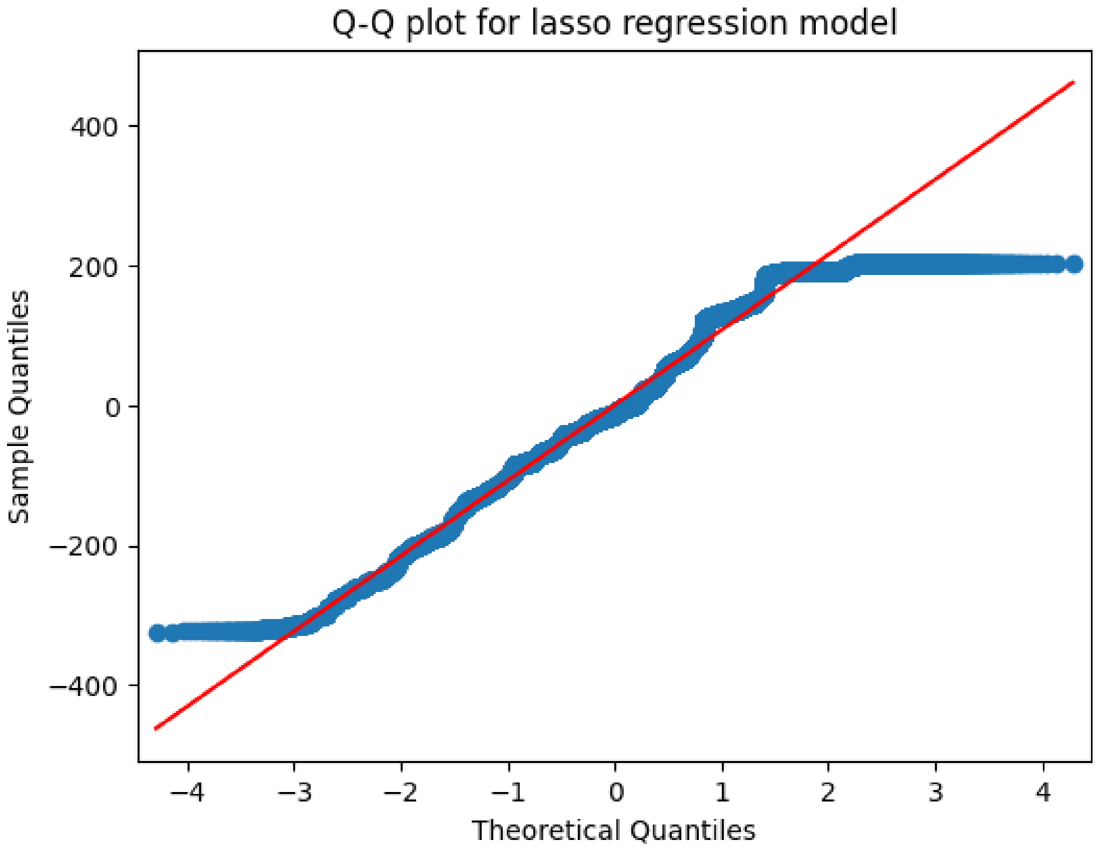
[ ]:
```python
def evaluate_model(y_test, y_pred, model_name):
  residuals = y_test - y_pred

  sm.qqplot(residuals, line = 's')
  plt.title(f"Q-Q plot for {model_name}")
  plt.show()

  sns.scatterplot(x = y_pred, y = residuals)
  plt.axhline(0)
  plt.xlabel('Fitted values')
  plt.ylabel('Residuals')
  plt.title(f"Residuals vs. Fitted Values for {model_name}")
  plt.show()

evaluate_model(y_test, y_pred_ols, "OLS model")
evaluate_model(y_test, y_pred_ridge, "ridge regression model")
evaluate_model(y_test, y_pred_lasso, "lasso regression model")
```

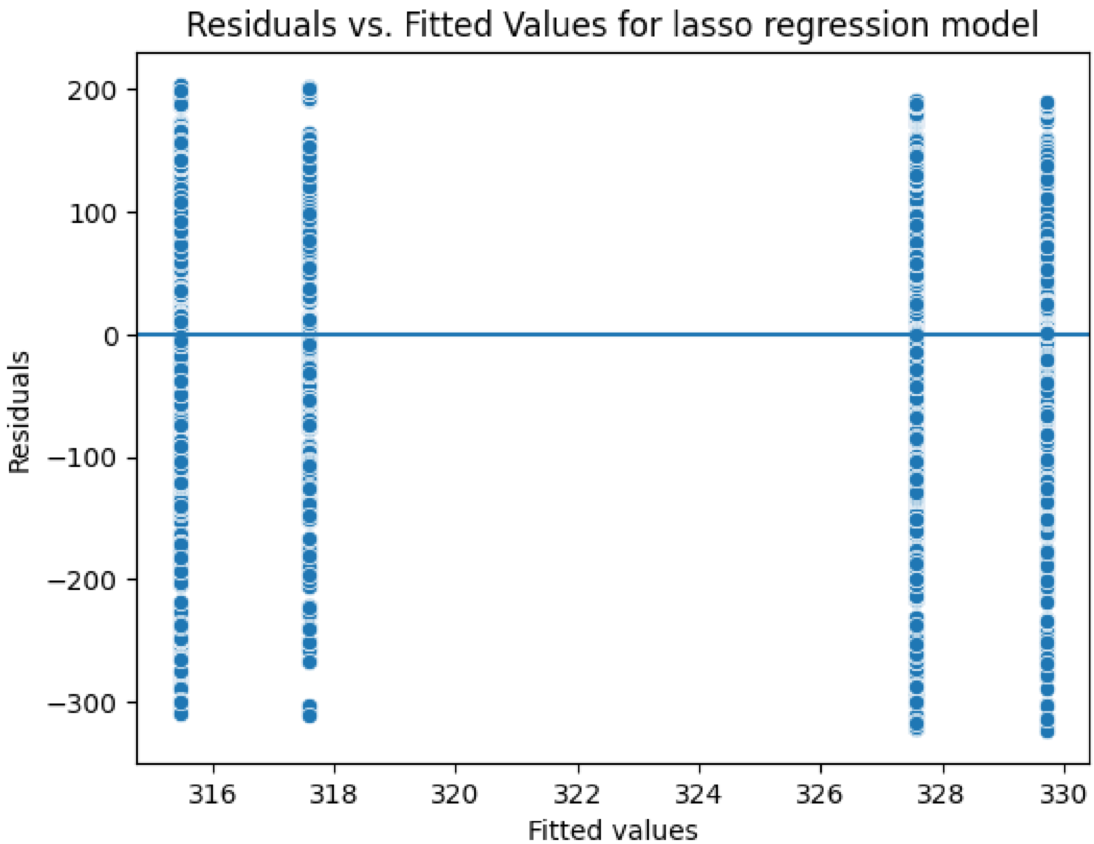

Q-Q plot for OLS model
Sample Quantiles
Theoretical Quantiles
400
200
0
−200
−400
−4
−3
−2
−1
0
1
2
3
4


Residuals vs. Fitted Values for OLS model
200
100
0
−100
−200
−300
Residuals
310
315
320
325
330
335
Fitted values


Q-Q plot for ridge regression model
400
200
0
−200
−400
Sample Quantiles
−4
−3
−2
−1
0
1
2
3
4
Theoretical Quantiles


Residuals vs. Fitted Values for ridge regression model
200
100
0
−100
−200
−300
Residuals
310
315
320
325
330
335
Fitted values


Q-Q plot for lasso regression model
Sample Quantiles
Theoretical Quantiles
400
200
0
−200
−400
−4
−3
−2
−1
0
1
2
3
4




[ ]:
```
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc
```

```
Reading package lists… Done
Building dependency tree… Done
Reading state information… Done
The following additional packages will be installed:
  dvisvgm fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-
texgyre
  fonts-urw-base35 libapache-pom-java libcmark-gfm-extensions0.29.0.gfm.3
libcmark-gfm0.29.0.gfm.3
  libcommons-logging-java libcommons-parent-java libfontbox-java libfontenc1
libgs9 libgs9-common
  libidn12 libijs-0.35 libjbig2dec0 libkpathsea6 libpdfbox-java libptexenc1
libruby3.0 libsynctex2
  libteckit0 libtexlua53 libtexluajit2 libwoff1 libzzip-0-13 lmodern pandoc-data
poppler-data
  preview-latex-style rake ruby ruby-net-telnet ruby-rubygems ruby-webrick ruby-
xmlrpc ruby3.0
  rubygems-integration t1utils teckit tex-common tex-gyre texlive-base texlive-
binaries
```

```
  texlive-fonts-recommended texlive-latex-base texlive-latex-recommended
texlive-pictures
  texlive-plain-generic tipa xfonts-encodings xfonts-utils
Suggested packages:
  fonts-noto fonts-freefont-otf | fonts-freefont-ttf libavalon-framework-java
  libcommons-logging-java-doc libexcalibur-logkit-java liblog4j1.2-java texlive-
luatex
  pandoc-citeproc context wkhtmltopdf librsvg2-bin groff ghc nodejs php python
libjs-mathjax
  libjs-katex citation-style-language-styles poppler-utils ghostscript fonts-
japanese-mincho
  | fonts-ipafont-mincho fonts-japanese-gothic | fonts-ipafont-gothic fonts-
arphic-ukai
  fonts-arphic-uming fonts-nanum ri ruby-dev bundler debhelper gv | postscript-
viewer perl-tk xpdf
  | pdf-viewer xzdec texlive-fonts-recommended-doc texlive-latex-base-doc
python3-pygments
  icc-profiles libfile-which-perl libspreadsheet-parseexcel-perl texlive-latex-
extra-doc
  texlive-latex-recommended-doc texlive-pstricks dot2tex prerex texlive-
pictures-doc vprerex
  default-jre-headless tipa-doc
The following NEW packages will be installed:
  dvisvgm fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-
texgyre
  fonts-urw-base35 libapache-pom-java libcmark-gfm-extensions0.29.0.gfm.3
libcmark-gfm0.29.0.gfm.3
  libcommons-logging-java libcommons-parent-java libfontbox-java libfontenc1
libgs9 libgs9-common
  libidn12 libijs-0.35 libjbig2dec0 libkpathsea6 libpdfbox-java libptexenc1
libruby3.0 libsynctex2
  libteckit0 libtexlua53 libtexluajit2 libwoff1 libzzip-0-13 lmodern pandoc
pandoc-data
  poppler-data preview-latex-style rake ruby ruby-net-telnet ruby-rubygems ruby-
webrick ruby-xmlrpc
  ruby3.0 rubygems-integration t1utils teckit tex-common tex-gyre texlive
texlive-base
  texlive-binaries texlive-fonts-recommended texlive-latex-base texlive-latex-
extra
  texlive-latex-recommended texlive-pictures texlive-plain-generic texlive-xetex
tipa
  xfonts-encodings xfonts-utils
0 upgraded, 59 newly installed, 0 to remove and 49 not upgraded.
Need to get 202 MB of archives.
After this operation, 728 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-droid-fallback all
1:6.0.1r16-1.1build1 [1,805 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-lato all 2.0-2.1
```

```
[2,696 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/main amd64 poppler-data all
0.4.11-1 [2,171 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tex-common all 6.17
[33.7 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-urw-base35 all
20200910-1 [6,367 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgs9-common
all 9.55.0~dfsg1-0ubuntu5.9 [752 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libidn12 amd64
1.38-4ubuntu1 [60.0 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy/main amd64 libijs-0.35 amd64
0.35-15build2 [16.5 kB]
Get:9 http://archive.ubuntu.com/ubuntu jammy/main amd64 libjbig2dec0 amd64
0.19-3build2 [64.7 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgs9 amd64
9.55.0~dfsg1-0ubuntu5.9 [5,033 kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libkpathsea6
amd64 2021.20210626.59705-1ubuntu0.2 [60.4 kB]
Get:12 http://archive.ubuntu.com/ubuntu jammy/main amd64 libwoff1 amd64
1.0.2-1build4 [45.2 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy/universe amd64 dvisvgm amd64
2.13.1-1 [1,221 kB]
Get:14 http://archive.ubuntu.com/ubuntu jammy/universe amd64 fonts-lmodern all
2.004.5-6.1 [4,532 kB]
Get:15 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-noto-mono all
20201225-1build1 [397 kB]
Get:16 http://archive.ubuntu.com/ubuntu jammy/universe amd64 fonts-texgyre all
20180621-3.1 [10.2 MB]
Get:17 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libapache-pom-java
all 18-1 [4,720 B]
Get:18 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcmark-
gfm0.29.0.gfm.3 amd64 0.29.0.gfm.3-3 [115 kB]
Get:19 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcmark-gfm-
extensions0.29.0.gfm.3 amd64 0.29.0.gfm.3-3 [25.1 kB]
Get:20 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcommons-parent-
java all 43-1 [10.8 kB]
Get:21 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcommons-logging-
java all 1.2-2 [60.3 kB]
Get:22 http://archive.ubuntu.com/ubuntu jammy/main amd64 libfontenc1 amd64
1:1.1.4-1build3 [14.7 kB]
Get:23 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libptexenc1
amd64 2021.20210626.59705-1ubuntu0.2 [39.1 kB]
Get:24 http://archive.ubuntu.com/ubuntu jammy/main amd64 rubygems-integration
all 1.18 [5,336 B]
Get:25 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby3.0 amd64
3.0.2-7ubuntu2.7 [50.1 kB]
Get:26 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby-rubygems all
```

```
3.3.5-2 [228 kB]
Get:27 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby amd64 1:3.0~exp1
[5,100 B]
Get:28 http://archive.ubuntu.com/ubuntu jammy/main amd64 rake all 13.0.6-2 [61.7
kB]
Get:29 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby-net-telnet all
0.1.1-2 [12.6 kB]
Get:30 http://archive.ubuntu.com/ubuntu jammy/universe amd64 ruby-webrick all
1.7.0-3 [51.8 kB]
Get:31 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby-xmlrpc all
0.3.2-1ubuntu0.1 [24.9 kB]
Get:32 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libruby3.0
amd64 3.0.2-7ubuntu2.7 [5,113 kB]
Get:33 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libsynctex2
amd64 2021.20210626.59705-1ubuntu0.2 [55.6 kB]
Get:34 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libteckit0 amd64
2.5.11+ds1-1 [421 kB]
Get:35 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtexlua53
amd64 2021.20210626.59705-1ubuntu0.2 [120 kB]
Get:36 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtexluajit2
amd64 2021.20210626.59705-1ubuntu0.2 [267 kB]
Get:37 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libzzip-0-13 amd64
0.13.72+dfsg.1-1.1 [27.0 kB]
Get:38 http://archive.ubuntu.com/ubuntu jammy/main amd64 xfonts-encodings all
1:1.0.5-0ubuntu2 [578 kB]
Get:39 http://archive.ubuntu.com/ubuntu jammy/main amd64 xfonts-utils amd64
1:7.7+6build2 [94.6 kB]
Get:40 http://archive.ubuntu.com/ubuntu jammy/universe amd64 lmodern all
2.004.5-6.1 [9,471 kB]
Get:41 http://archive.ubuntu.com/ubuntu jammy/universe amd64 pandoc-data all
2.9.2.1-3ubuntu2 [81.8 kB]
Get:42 http://archive.ubuntu.com/ubuntu jammy/universe amd64 pandoc amd64
2.9.2.1-3ubuntu2 [20.3 MB]
Get:43 http://archive.ubuntu.com/ubuntu jammy/universe amd64 preview-latex-style
all 12.2-1ubuntu1 [185 kB]
Get:44 http://archive.ubuntu.com/ubuntu jammy/main amd64 t1utils amd64
1.41-4build2 [61.3 kB]
Get:45 http://archive.ubuntu.com/ubuntu jammy/universe amd64 teckit amd64
2.5.11+ds1-1 [699 kB]
Get:46 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tex-gyre all
20180621-3.1 [6,209 kB]
Get:47 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 texlive-
binaries amd64 2021.20210626.59705-1ubuntu0.2 [9,860 kB]
Get:48 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-base all
2021.20220204-1 [21.0 MB]
Get:49 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-fonts-
recommended all 2021.20220204-1 [4,972 kB]
Get:50 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-base
```

```
all 2021.20220204-1 [1,128 kB]
Get:51 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-
recommended all 2021.20220204-1 [14.4 MB]
Get:52 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive all
2021.20220204-1 [14.3 kB]
Get:53 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libfontbox-java all
1:1.8.16-2 [207 kB]
Get:54 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libpdfbox-java all
1:1.8.16-2 [5,199 kB]
Get:55 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-pictures
all 2021.20220204-1 [8,720 kB]
Get:56 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-extra
all 2021.20220204-1 [13.9 MB]
Get:57 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-plain-
generic all 2021.20220204-1 [27.5 MB]
Get:58 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tipa all 2:1.3-21
[2,967 kB]
Get:59 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-xetex all
2021.20220204-1 [12.4 MB]
Fetched 202 MB in 15s (13.4 MB/s)
Extracting templates from packages: 100%
Preconfiguring packages …
Selecting previously unselected package fonts-droid-fallback.
(Reading database … 123597 files and directories currently installed.)
Preparing to unpack …/00-fonts-droid-fallback_1%3a6.0.1r16-1.1build1_all.deb
…
Unpacking fonts-droid-fallback (1:6.0.1r16-1.1build1) …
Selecting previously unselected package fonts-lato.
Preparing to unpack …/01-fonts-lato_2.0-2.1_all.deb …
Unpacking fonts-lato (2.0-2.1) …
Selecting previously unselected package poppler-data.
Preparing to unpack …/02-poppler-data_0.4.11-1_all.deb …
Unpacking poppler-data (0.4.11-1) …
Selecting previously unselected package tex-common.
Preparing to unpack …/03-tex-common_6.17_all.deb …
Unpacking tex-common (6.17) …
Selecting previously unselected package fonts-urw-base35.
Preparing to unpack …/04-fonts-urw-base35_20200910-1_all.deb …
Unpacking fonts-urw-base35 (20200910-1) …
Selecting previously unselected package libgs9-common.
Preparing to unpack …/05-libgs9-common_9.55.0~dfsg1-0ubuntu5.9_all.deb …
Unpacking libgs9-common (9.55.0~dfsg1-0ubuntu5.9) …
Selecting previously unselected package libidn12:amd64.
Preparing to unpack …/06-libidn12_1.38-4ubuntu1_amd64.deb …
Unpacking libidn12:amd64 (1.38-4ubuntu1) …
Selecting previously unselected package libijs-0.35:amd64.
Preparing to unpack …/07-libijs-0.35_0.35-15build2_amd64.deb …
Unpacking libijs-0.35:amd64 (0.35-15build2) …
```

```
Selecting previously unselected package libjbig2dec0:amd64.
Preparing to unpack …/08-libjbig2dec0_0.19-3build2_amd64.deb …
Unpacking libjbig2dec0:amd64 (0.19-3build2) …
Selecting previously unselected package libgs9:amd64.
Preparing to unpack …/09-libgs9_9.55.0~dfsg1-0ubuntu5.9_amd64.deb …
Unpacking libgs9:amd64 (9.55.0~dfsg1-0ubuntu5.9) …
Selecting previously unselected package libkpathsea6:amd64.
Preparing to unpack …/10-libkpathsea6_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libkpathsea6:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libwoff1:amd64.
Preparing to unpack …/11-libwoff1_1.0.2-1build4_amd64.deb …
Unpacking libwoff1:amd64 (1.0.2-1build4) …
Selecting previously unselected package dvisvgm.
Preparing to unpack …/12-dvisvgm_2.13.1-1_amd64.deb …
Unpacking dvisvgm (2.13.1-1) …
Selecting previously unselected package fonts-lmodern.
Preparing to unpack …/13-fonts-lmodern_2.004.5-6.1_all.deb …
Unpacking fonts-lmodern (2.004.5-6.1) …
Selecting previously unselected package fonts-noto-mono.
Preparing to unpack …/14-fonts-noto-mono_20201225-1build1_all.deb …
Unpacking fonts-noto-mono (20201225-1build1) …
Selecting previously unselected package fonts-texgyre.
Preparing to unpack …/15-fonts-texgyre_20180621-3.1_all.deb …
Unpacking fonts-texgyre (20180621-3.1) …
Selecting previously unselected package libapache-pom-java.
Preparing to unpack …/16-libapache-pom-java_18-1_all.deb …
Unpacking libapache-pom-java (18-1) …
Selecting previously unselected package libcmark-gfm0.29.0.gfm.3:amd64.
Preparing to unpack …/17-libcmark-gfm0.29.0.gfm.3_0.29.0.gfm.3-3_amd64.deb …
Unpacking libcmark-gfm0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) …
Selecting previously unselected package libcmark-gfm-
extensions0.29.0.gfm.3:amd64.
Preparing to unpack …/18-libcmark-gfm-
extensions0.29.0.gfm.3_0.29.0.gfm.3-3_amd64.deb …
Unpacking libcmark-gfm-extensions0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) …
Selecting previously unselected package libcommons-parent-java.
Preparing to unpack …/19-libcommons-parent-java_43-1_all.deb …
Unpacking libcommons-parent-java (43-1) …
Selecting previously unselected package libcommons-logging-java.
Preparing to unpack …/20-libcommons-logging-java_1.2-2_all.deb …
Unpacking libcommons-logging-java (1.2-2) …
Selecting previously unselected package libfontenc1:amd64.
Preparing to unpack …/21-libfontenc1_1%3a1.1.4-1build3_amd64.deb …
Unpacking libfontenc1:amd64 (1:1.1.4-1build3) …
Selecting previously unselected package libptexenc1:amd64.
Preparing to unpack …/22-libptexenc1_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
```

```
Unpacking libptexenc1:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package rubygems-integration.
Preparing to unpack …/23-rubygems-integration_1.18_all.deb …
Unpacking rubygems-integration (1.18) …
Selecting previously unselected package ruby3.0.
Preparing to unpack …/24-ruby3.0_3.0.2-7ubuntu2.7_amd64.deb …
Unpacking ruby3.0 (3.0.2-7ubuntu2.7) …
Selecting previously unselected package ruby-rubygems.
Preparing to unpack …/25-ruby-rubygems_3.3.5-2_all.deb …
Unpacking ruby-rubygems (3.3.5-2) …
Selecting previously unselected package ruby.
Preparing to unpack …/26-ruby_1%3a3.0~exp1_amd64.deb …
Unpacking ruby (1:3.0~exp1) …
Selecting previously unselected package rake.
Preparing to unpack …/27-rake_13.0.6-2_all.deb …
Unpacking rake (13.0.6-2) …
Selecting previously unselected package ruby-net-telnet.
Preparing to unpack …/28-ruby-net-telnet_0.1.1-2_all.deb …
Unpacking ruby-net-telnet (0.1.1-2) …
Selecting previously unselected package ruby-webrick.
Preparing to unpack …/29-ruby-webrick_1.7.0-3_all.deb …
Unpacking ruby-webrick (1.7.0-3) …
Selecting previously unselected package ruby-xmlrpc.
Preparing to unpack …/30-ruby-xmlrpc_0.3.2-1ubuntu0.1_all.deb …
Unpacking ruby-xmlrpc (0.3.2-1ubuntu0.1) …
Selecting previously unselected package libruby3.0:amd64.
Preparing to unpack …/31-libruby3.0_3.0.2-7ubuntu2.7_amd64.deb …
Unpacking libruby3.0:amd64 (3.0.2-7ubuntu2.7) …
Selecting previously unselected package libsynctex2:amd64.
Preparing to unpack …/32-libsynctex2_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libsynctex2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libteckit0:amd64.
Preparing to unpack …/33-libteckit0_2.5.11+ds1-1_amd64.deb …
Unpacking libteckit0:amd64 (2.5.11+ds1-1) …
Selecting previously unselected package libtexlua53:amd64.
Preparing to unpack …/34-libtexlua53_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libtexlua53:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libtexluajit2:amd64.
Preparing to unpack
…/35-libtexluajit2_2021.20210626.59705-1ubuntu0.2_amd64.deb …
Unpacking libtexluajit2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libzzip-0-13:amd64.
Preparing to unpack …/36-libzzip-0-13_0.13.72+dfsg.1-1.1_amd64.deb …
Unpacking libzzip-0-13:amd64 (0.13.72+dfsg.1-1.1) …
Selecting previously unselected package xfonts-encodings.
Preparing to unpack …/37-xfonts-encodings_1%3a1.0.5-0ubuntu2_all.deb …
```

```
Unpacking xfonts-encodings (1:1.0.5-0ubuntu2) …
Selecting previously unselected package xfonts-utils.
Preparing to unpack …/38-xfonts-utils_1%3a7.7+6build2_amd64.deb …
Unpacking xfonts-utils (1:7.7+6build2) …
Selecting previously unselected package lmodern.
Preparing to unpack …/39-lmodern_2.004.5-6.1_all.deb …
Unpacking lmodern (2.004.5-6.1) …
Selecting previously unselected package pandoc-data.
Preparing to unpack …/40-pandoc-data_2.9.2.1-3ubuntu2_all.deb …
Unpacking pandoc-data (2.9.2.1-3ubuntu2) …
Selecting previously unselected package pandoc.
Preparing to unpack …/41-pandoc_2.9.2.1-3ubuntu2_amd64.deb …
Unpacking pandoc (2.9.2.1-3ubuntu2) …
Selecting previously unselected package preview-latex-style.
Preparing to unpack …/42-preview-latex-style_12.2-1ubuntu1_all.deb …
Unpacking preview-latex-style (12.2-1ubuntu1) …
Selecting previously unselected package t1utils.
Preparing to unpack …/43-t1utils_1.41-4build2_amd64.deb …
Unpacking t1utils (1.41-4build2) …
Selecting previously unselected package teckit.
Preparing to unpack …/44-teckit_2.5.11+ds1-1_amd64.deb …
Unpacking teckit (2.5.11+ds1-1) …
Selecting previously unselected package tex-gyre.
Preparing to unpack …/45-tex-gyre_20180621-3.1_all.deb …
Unpacking tex-gyre (20180621-3.1) …
Selecting previously unselected package texlive-binaries.
Preparing to unpack …/46-texlive-
binaries_2021.20210626.59705-1ubuntu0.2_amd64.deb …
Unpacking texlive-binaries (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package texlive-base.
Preparing to unpack …/47-texlive-base_2021.20220204-1_all.deb …
Unpacking texlive-base (2021.20220204-1) …
Selecting previously unselected package texlive-fonts-recommended.
Preparing to unpack …/48-texlive-fonts-recommended_2021.20220204-1_all.deb …
Unpacking texlive-fonts-recommended (2021.20220204-1) …
Selecting previously unselected package texlive-latex-base.
Preparing to unpack …/49-texlive-latex-base_2021.20220204-1_all.deb …
Unpacking texlive-latex-base (2021.20220204-1) …
Selecting previously unselected package texlive-latex-recommended.
Preparing to unpack …/50-texlive-latex-recommended_2021.20220204-1_all.deb …
Unpacking texlive-latex-recommended (2021.20220204-1) …
Selecting previously unselected package texlive.
Preparing to unpack …/51-texlive_2021.20220204-1_all.deb …
Unpacking texlive (2021.20220204-1) …
Selecting previously unselected package libfontbox-java.
Preparing to unpack …/52-libfontbox-java_1%3a1.8.16-2_all.deb …
Unpacking libfontbox-java (1:1.8.16-2) …
Selecting previously unselected package libpdfbox-java.
```

```
Preparing to unpack …/53-libpdfbox-java_1%3a1.8.16-2_all.deb …
Unpacking libpdfbox-java (1:1.8.16-2) …
Selecting previously unselected package texlive-pictures.
Preparing to unpack …/54-texlive-pictures_2021.20220204-1_all.deb …
Unpacking texlive-pictures (2021.20220204-1) …
Selecting previously unselected package texlive-latex-extra.
Preparing to unpack …/55-texlive-latex-extra_2021.20220204-1_all.deb …
Unpacking texlive-latex-extra (2021.20220204-1) …
Selecting previously unselected package texlive-plain-generic.
Preparing to unpack …/56-texlive-plain-generic_2021.20220204-1_all.deb …
Unpacking texlive-plain-generic (2021.20220204-1) …
Selecting previously unselected package tipa.
Preparing to unpack …/57-tipa_2%3a1.3-21_all.deb …
Unpacking tipa (2:1.3-21) …
Selecting previously unselected package texlive-xetex.
Preparing to unpack …/58-texlive-xetex_2021.20220204-1_all.deb …
Unpacking texlive-xetex (2021.20220204-1) …
Setting up fonts-lato (2.0-2.1) …
Setting up fonts-noto-mono (20201225-1build1) …
Setting up libwoff1:amd64 (1.0.2-1build4) …
Setting up libtexlua53:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libijs-0.35:amd64 (0.35-15build2) …
Setting up libtexluajit2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libfontbox-java (1:1.8.16-2) …
Setting up rubygems-integration (1.18) …
Setting up libzzip-0-13:amd64 (0.13.72+dfsg.1-1.1) …
Setting up fonts-urw-base35 (20200910-1) …
Setting up poppler-data (0.4.11-1) …
Setting up tex-common (6.17) …
update-language: texlive-base not installed and configured, doing nothing!
Setting up libfontenc1:amd64 (1:1.1.4-1build3) …
Setting up libjbig2dec0:amd64 (0.19-3build2) …
Setting up libteckit0:amd64 (2.5.11+ds1-1) …
Setting up libapache-pom-java (18-1) …
Setting up ruby-net-telnet (0.1.1-2) …
Setting up xfonts-encodings (1:1.0.5-0ubuntu2) …
Setting up t1utils (1.41-4build2) …
Setting up libidn12:amd64 (1.38-4ubuntu1) …
Setting up fonts-texgyre (20180621-3.1) …
Setting up libkpathsea6:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up ruby-webrick (1.7.0-3) …
Setting up libcmark-gfm0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) …
Setting up fonts-lmodern (2.004.5-6.1) …
Setting up libcmark-gfm-extensions0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) …
Setting up fonts-droid-fallback (1:6.0.1r16-1.1build1) …
Setting up pandoc-data (2.9.2.1-3ubuntu2) …
Setting up ruby-xmlrpc (0.3.2-1ubuntu0.1) …
Setting up libsynctex2:amd64 (2021.20210626.59705-1ubuntu0.2) …
```

```
Setting up libgs9-common (9.55.0~dfsg1-0ubuntu5.9) …
Setting up teckit (2.5.11+ds1-1) …
Setting up libpdfbox-java (1:1.8.16-2) …
Setting up libgs9:amd64 (9.55.0~dfsg1-0ubuntu5.9) …
Setting up preview-latex-style (12.2-1ubuntu1) …
Setting up libcommons-parent-java (43-1) …
Setting up dvisvgm (2.13.1-1) …
Setting up libcommons-logging-java (1.2-2) …
Setting up xfonts-utils (1:7.7+6build2) …
Setting up libptexenc1:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up pandoc (2.9.2.1-3ubuntu2) …
Setting up texlive-binaries (2021.20210626.59705-1ubuntu0.2) …
update-alternatives: using /usr/bin/xdvi-xaw to provide /usr/bin/xdvi.bin
(xdvi.bin) in auto mode
update-alternatives: using /usr/bin/bibtex.original to provide /usr/bin/bibtex
(bibtex) in auto mode
Setting up lmodern (2.004.5-6.1) …
Setting up texlive-base (2021.20220204-1) …
/usr/bin/ucfr
/usr/bin/ucfr
/usr/bin/ucfr
/usr/bin/ucfr
mktexlsr: Updating /var/lib/texmf/ls-R-TEXLIVEDIST…
mktexlsr: Updating /var/lib/texmf/ls-R-TEXMFMAIN…
mktexlsr: Updating /var/lib/texmf/ls-R…
mktexlsr: Done.
tl-paper: setting paper size for dvips to a4:
/var/lib/texmf/dvips/config/config-paper.ps
tl-paper: setting paper size for dvipdfmx to a4:
/var/lib/texmf/dvipdfmx/dvipdfmx-paper.cfg
tl-paper: setting paper size for xdvi to a4: /var/lib/texmf/xdvi/XDvi-paper
tl-paper: setting paper size for pdftex to a4: /var/lib/texmf/tex/generic/tex-
ini-files/pdftexconfig.tex
Setting up tex-gyre (20180621-3.1) …
Setting up texlive-plain-generic (2021.20220204-1) …
Setting up texlive-latex-base (2021.20220204-1) …
Setting up texlive-latex-recommended (2021.20220204-1) …
Setting up texlive-pictures (2021.20220204-1) …
Setting up texlive-fonts-recommended (2021.20220204-1) …
Setting up tipa (2:1.3-21) …
Setting up texlive (2021.20220204-1) …
Setting up texlive-latex-extra (2021.20220204-1) …
Setting up texlive-xetex (2021.20220204-1) …
Setting up rake (13.0.6-2) …
Setting up libruby3.0:amd64 (3.0.2-7ubuntu2.7) …
Setting up ruby3.0 (3.0.2-7ubuntu2.7) …
Setting up ruby (1:3.0~exp1) …
Setting up ruby-rubygems (3.3.5-2) …
```

```
Processing triggers for man-db (2.10.2-1) …
Processing triggers for fontconfig (2.13.1-4.2ubuntu5) …
Processing triggers for libc-bin (2.35-0ubuntu3.4) …
/sbin/ldconfig.real: /usr/local/lib/libur_adapter_level_zero.so.0 is not a
symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_0.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc_proxy.so.2 is not a symbolic
link

/sbin/ldconfig.real: /usr/local/lib/libur_loader.so.0 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbb.so.12 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libur_adapter_opencl.so.0 is not a symbolic
link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_5.so.3 is not a symbolic link

Processing triggers for tex-common (6.17) …
Running updmap-sys. This may take some time… done.
Running mktexlsr /var/lib/texmf … done.
Building format(s) --all.
        This may take some time…
```

[ ]:
```
!jupyter nbconvert --to pdf "/content/drive/MyDrive/Colab Notebooks/
↪walmart-Dataset@Dhanureddy.ipynb"
```

```
[NbConvertApp] Converting notebook /content/drive/MyDrive/Colab
Notebooks/walmart-Dataset@Dhanureddy.ipynb to pdf
[NbConvertApp] Support files will be in walmart-Dataset@Dhanureddy_files/
[NbConvertApp] Making directory ./walmart-Dataset@Dhanureddy_files
[NbConvertApp] Making directory ./walmart-Dataset@Dhanureddy_files
[NbConvertApp] Making directory ./walmart-Dataset@Dhanureddy_files
[NbConvertApp] Making directory ./walmart-Dataset@Dhanureddy_files
[NbConvertApp] Making directory ./walmart-Dataset@Dhanureddy_files
[NbConvertApp] Making directory ./walmart-Dataset@Dhanureddy_files
[NbConvertApp] Making directory ./walmart-Dataset@Dhanureddy_files
[NbConvertApp] Making directory ./walmart-Dataset@Dhanureddy_files
[NbConvertApp] Making directory ./walmart-Dataset@Dhanureddy_files
[NbConvertApp] Writing 142243 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
```

```
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | bibtex had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 559949 bytes to /content/drive/MyDrive/Colab
Notebooks/walmart-Dataset@Dhanureddy.pdf
```