Module: ITNPBD2 Autumn 2025

Assessment: Main assignment

Due Date/Time: 14/11/2025

AIAS Levels Allowed: 2

	Please tick the boxes/include appropriate information below
Student ID Number	3459046
Word Count (penalties apply for exceeding the stated limit)	21 Pages
I have read and understand the severity of academic misconduct – see link below	☐
I give consent for my work to be used as an exemplar to future students.	☐
I have checked my submitted document to ensure it complies with module requirements.	☐
Link to version-controlled file (i.e on OneDrive, Google Docs, Github, or other) which contain evidence of the process I undertook to complete this assignment. Information on how to create a Microsoft 365 OneDrive folder is available HERE. *Please see notes below	https://colab.research.google.com/drive/1x3I-3u_oPSuXvPT-wIYpLmXIKhsv94ac?usp=sharing https://github.com/jeetparashar/3459046_BD2_Assignment
I understand that if there is a concern about potential academic misconduct, including inappropriate use of AI tools, then I could be asked to provide evidence of my drafting process during an academic integrity meeting if I have not done so using the link above. Not providing evidence of my drafting process could prejudice the outcome of academic misconduct cases.	☐
Tailored feedback. If you would like tailored feedback on a specific aspect (or aspects) of your work (e.g., referencing, writing style, grammar), then please give details here.	[student can provide extra information on specific feedback they would like]
If you used AI at (or below) the level allowed, please explain briefly which AI, how you used it, and for what purpose.	[student to provide information about their AI use]

Introduction

In this project, I took the initiative to assist JC Penney better understand the relationships between their consumers, their online product reviews, and their online product reviews. JC Penney is a major retailer in the United States, and they are large quantity of customer product reviews. My objective was to analyze this data to determine the most frequent reviewers, and if different age demographic and geographical divisions, and product pricing, affect the level of interest in a product.

For my analysis, I utilized five primary data files. The products.csv contains the core product data, including the product's unique identification codes, names, prices, and ratings. The reviews.csv contains the user reviews, the ratings of the reviews, and we can determine which products receive the most reviews. The users.csv provides the most demographic information of the purchasers, including username, region, and age. I also utilized two json files, jcpenney_products.json which contains additional data about the products, and jcpenney_reviewers.json which provides background information about reviewers, their reviews and histories to assist in my analysis.

In Python, I cleaned and merged all these files to ensure I had dependable information that was easy to work with.

Objectives

The main objectives of my project are:

1. To improve data quality and usability through cleaning and merging multiple datasets from JC Penney.
2. To analyze how factors like product price and category affect consumer satisfaction and activity level in reviews.
3. To analyze product reviews by age and region in order to identify differences in shopping behaviors of specific customer segments.
4. To provide valuable and actionable information for JC Penney's marketing, pricing, and assortment planning decisions.

Achieving these objectives will help JC Penney gain valuable insights into their customer base, leading to more effective business decisions.

Data Preparation and Cleaning

2.1 Data Sources

The coding files interacted directly with users.csv, jcpenney_reviewers.json, reviews.csv, products.csv, and jcpenney_products.json. The product information is real, while the user information is fictitious. This ensured consistency among the business insights and reproducibility among the Python workflows.

2.2 Loading & Validation

Leveraging pandas and the json library in Python to access CSV as well as JSON files, I began the process of loading the files and doing an initial inspection by outputting the first and the last rows of the data. I performed username checks to validate the merging keys, and I pruned the duplicated usernames in the data by utilizing the drop_duplicates() function. Initially, I checked for missing values and applied appropriate treatment on them, and I transformed the date columns to datetime objects to facilitate further profiling.

2.3 Variable Engineering

The primary engagement variable, Activity, was created with the use of pandas .apply() and a lambda in code, categorizing every user with one or more reviews as Active and as Inactive otherwise. In parallel with the review count, state, and user name, this variable formed the foundation of all subsequent code and analysis involving groupby and aggregation.

2.4 Quality Check

Recorded mitigation actions and verification rationale were incorporated sequentially into the code for the purposes of facilitating audits and providing clarity for the academic community and business professionals alike. User datasets underwent testing for duplicates and completeness as well as validation for key matching criteria prior to analytics generation.

3. Key Analyses & Visualizations

a) Overall User Engagement: pie Chart

Code: After an Activity variable was generated utilizing Pandas, value_counts, along with Matplotlib and Seaborn, were used to visualize the proportions of active and inactive individuals.

Insight: Identified from the code output and chart labels, the corresponding data indicate that 19% of the users, which translates to approximately one-fifth of the users for this app, are inactive users. This is one of the major marketing opportunities to capture this sector of the potential users.

Business Meaning: This segment is most likely to return high engagement when investing resources such as emails, discounts, or prompts. The code for the pie chart and the chart output are the most significant for reporting to the executives. The comments noted the alternative styles for the bar or comparison chart.

b) Distribution of Users Review Activity: Histogram

Code: To understand the engagement of users, I evaluated the number of products reviewed by each user based on the Reviewed column and .apply(len). then presented this data as a num_reviews column of the original dataframe. then added the summary statistics, such as mean, and maximum, as well as the number of people not providing any reviews.tthen plotted data as a histogram, utilizing Matplotlib to distribute the reviews, with rounded sky blue bars, and easily readable x and y axis.

Insight: The data illustrated in the histogram indicates that the majority of users only submit a limited number of reviews, whereas a small number of users submit a large quantity of reviews. This illustrates that the user engagement is concentrated to a small portion of the entire user population.

Business Meaning: With this information JC Penney can start implementing strategies on how to communicate effectively with this large group of less-active users while rewarding the active users with "super user" status to keep them engaged. The histogram can be utilized to know what part of your business needs to be changed to increase user activity and increase the number of reviews given.

c) Top 10 States by active user %

Code: I did the segmentation of users based on the user 'State' and I found the percentage of users within each state who were tagged as "Active" as per their review counts. I created a function in pandas utilizing a lambda function to determine the proportion of active users per state, and I sorted the data to obtain the first ten states. I represented this data using a horizontal bar chart in Matplotlib, as it was the best way to visualize and compare the level of user engagement per state. I emphasized the bars in specific colors, as I titled the chart as "Top 10 States by % of Active Users" to provide a clearer understanding of the data.

Insight: The chart displayed a strong engagement discrepancy with a majority of states in the US falling within a low percentage tier, and a select few with a high percentage of active users. This indicates that some states have a considerable number of customers available who submit reviews, thus categorizing those states as a feedback and engagement review hotspot.

Business Meaning: These states on the peak of the list are the marketing sweet spots for JC Penney in terms of promotional and campaign efforts. They also provide a more efficient way of utilizing business resources to improve engagement and foster closer relationships with customers.The chart is good in helping executives figure out which services to invest in to increase reviews and customer interaction.

d)Engagement with age group : Bar Chart

Code: Using the year of birth to determine age and placing users into age bands (e.g. 20-29, 30-39, etc.). I calculated the engaged participants of each age band using pandas and plotted the data in gold colored bar charts using the Matplotlib package. The data from the participants of each age range were represented in a visual manner allowing for easy engagement analysis of the age segments.

Insights: Bar diagrams made it easy to determine which age group(s) had higher engagement in reviews, especially the 30-49 age range. The engagement in this age range was much higher in JC Penney reviews than that of younger users (<= 20) and the older age group (>= 60) which resulted in a peak engagement in review participation.

Business Maning: Based on the age segments and engagement level in review participation, business marketing and review resourcing efforts would be best spent on the 30~49 age group, which would generate a higher possibility of engaging participants and feedback. Strategies to motivate participation of the younger and older group on reviews would also widen the low engagement user base.

e) Price Band vs Average Review Score : Heat Map

Code: Products data was cleaned by removing negative prices, and reviews, for each price range, were averaged and visualized on a Seaborn heatmap, which presented a tidy finish along with discrete average score values.

Insights: The heatmap made evident that a clear trend was absent. Review scores were consistent, across each range, which indicates that pricing was likely not a determining factor for the ratings.

Business Meaning: JC Penney ought to enhance product quality and customer experience across every price point. Marketing and quality improvements initiatives can be directed towards any price tier, concentrating on the lower-rated products irrespective of the price point.

Conclusion

This report features JC Penney customer engagement and product reviews analysis by importing and cleaning different datasets including reviews, products and users. By analyzing users activity, it was possible to identify that the vast majority of users submited a single review, whereas a small number of users were super reviers. By utilizing barcharts, it was possible to observe that the most highly engaged users were middle age users (30-49), and the user spatial distriburotion by state provided the user or states cluster contained the most engaged users. Price ranges were defined and the average review scores were reviewed in a heatmap in relation to multiple products average price. The analysis conducted indicates that prices were not signifficant in scoring reviewed products. The included scores indicate that price did not influence sentiment of the users. The вehavior of JC Penney indicates the efforts would be better challenged on inactive users, and the active high engagement age groups, and region, also on improving products quality, and active sighning reward to super rvieers to raise the customer engagement.

In [177]:

```
#Importing required libraries for data processing and visualisation
import pandas as pd
import numpy as np
import json
import matplotlib.pyplot as plt
import seaborn as sns
```

Data Loading & Initial Exploration

In this section, I have loaded the user and review data of JC Penney for better understanding of structure which ensure accurate analysis

In [178]:

```
#Load CSV data
productsdf = pd.read_csv('/content/products.csv')
#Loading users.csv file which contains the user information
users_df = pd.read_csv('users.csv')
print("\nUsers data (first 5 rows): ")
```

```
print(users_df.head()) #Previewing the first few rows to check the
structure

#Loading jcpenny_reviewers.json as JSON Lines
reviews = []
with open('/content/jcpenney_reviewers.json', 'r') as file:
  for line in file:
    line = line.strip()
    if line:
      #Converting the JSON line to a python dictionar and adding it to list
      reviews.append(json.loads(line))

#Converting the list of review dicts to a dataframe
reviews_df = pd.DataFrame(reviews)

print("\nReviews data(first 5 rows): ")
print(reviews_df.head()) #Previewing the reviews data to confirm its loaded

#Showing the shape and columns for both datasets
print('Users data shape:', users_df.shape)
print("Users columns:", users_df.columns.tolist())

print("Reviews data shape: ", reviews_df.shape)
print("Reviews columns:", reviews_df.columns.tolist())

Users data (first 5 rows):
   Username         DOB          State
0  bkpn1412  31.07.1983         Oregon
1  gqjs4414  27.07.1998  Massachusetts
2  eehe1434  08.08.1950          Idaho
3  hkxj1334  03.08.1969        Florida
4  jjbd1412  26.07.2001        Georgia

Reviews data(first 5 rows):
   Username         DOB          State  \
0  bkpn1412  31.07.1983         Oregon
1  gqjs4414  27.07.1998  Massachusetts
2  eehe1434  08.08.1950          Idaho
3  hkxj1334  03.08.1969        Florida
4  jjbd1412  26.07.2001        Georgia

                                          Reviewed
0                [cea76118f6a9110a893de2b7654319c0]
1                [fa04fe6c0dd5189f54fe600838da43d3]
2                                                []
3  [f129b1803f447c2b1ce43508fb822810, 3b0c9bc0be6...
4                                                []
Users data shape: (5000, 3)
Users columns: ['Username', 'DOB', 'State']
```

```
Reviews data shape:  (5000, 4)
Reviews columns: ['Username', 'DOB', 'State', 'Reviewed']
```

Checking information from Users and Reviews DataFrame

In this section i have inspected both datasets to check for missing and duplicate values, and to prepare the data for analysis.

Users DataFrame

In [179]:

```
#Show Data types and non null counts for each column in Users
print("\nUsers DataFrame Info\n")
print(users_df.info())

#Checking for the missing values
print("\nMissing values in Users DataFrame\n")
print(users_df.isnull().sum())

#Display descriptive stats for numeric value
print("\nUsers DataFrame : Descriptive Statistics\n")
print(users_df.describe(include='all')) # This gives an overview of value
ranges, counts, common entires
```

```
Users DataFrame Info

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 3 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Username  5000 non-null   object
 1   DOB       5000 non-null   object
 2   State     5000 non-null   object
dtypes: object(3)
memory usage: 117.3+ KB
None

Missing values in Users DataFrame

Username    0
DOB         0
State       0
dtype: int64

Users DataFrame : Descriptive Statistics

        Username         DOB          State
count       5000        5000           5000
```

```
unique      4999          52             57
top     dqft3311  07.08.1953  Massachusetts
freq           2         112            107
```

Reviews DataFrame

In [180]:

```
#Printing out the data tpes and non null counts for each column in Reviews
print("\nReviews DataFrame Info:\n")
print(reviews_df.info()) #This helps in checking missing data and
understanding the values in each column

#Checking missing values
print("\nMissing values in column (reviews):\n")
print(reviews_df.isnull().sum()) #finding the missing values

#Display Descriptive stats
print("\nReviews DataFrame: Descriptive Stats\n")
print(reviews_df.describe(include='all')) #Quick overview of values ranges,
counts, common entries
```

```
Reviews DataFrame Info:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Username  5000 non-null   object
 1   DOB       5000 non-null   object
 2   State     5000 non-null   object
 3   Reviewed  5000 non-null   object
dtypes: object(4)
memory usage: 156.4+ KB
None

Missing values in column (reviews):

Username    0
DOB         0
State       0
Reviewed    0
dtype: int64

Reviews DataFrame: Descriptive Stats

        Username         DOB          State Reviewed
count       5000        5000           5000     5000
unique      4999          52             57     4030
```

```
top     dqft3311  07.08.1953  Massachusetts       []
freq           2         112            107      971
```

All usernames are unique. No missing values found, so te data is ready for further process

Random Samples from both DataFrames

Here is the random samples form both datasets to check for variety and unusual records.

In [181]:

```
#Printing random sample of 5 rows from the users data
print("\nRandom samples from the users data:\n")
print(users_df.sample(5))

#Printing the random sample of 5 rows from the reviews data
print("\nRandom samples from the reviews data:\n")
print(reviews_df.sample(5))
```

```
Random samples from the users data:

      Username         DOB                State
3328  vssd4424  07.08.1954              Alabama
3813  sfnm1233  03.08.1968             Colorado
4508  zlrf4111  28.07.1995         South Dakota
2749  epns2341  29.07.1988  U.S. Virgin Islands
1921  shse1442  08.08.1950       American Samoa

Random samples from the reviews data:

      Username         DOB                      State  \
3442  anfp3234  28.07.1992  Northern Mariana Islands
2648  jlsl2112  26.07.2000                New Mexico
4191  onsp1244  31.07.1983             West Virginia
2420  rghz2223  01.08.1977            North Carolina
448   qqev4243  30.07.1984                    Alaska

                                              Reviewed
3442  [5795fd6b81d00c177c8bf256ee0c6aaf, c63d1af97d8...
2648                                                 []
4191                 [829b145c65cab22a5fb92c56e0461511]
2420  [deb89d63b044aa8695c3713b8740999c, fc5049d17fa...
448                  [e7dfc34b04500be1ab450f7a94a64aab]
```

Checking for Missing Values and Duplicate Values

In [182]:

```
#Checking for missing values in users data
print("\nMisising values in user data:\n")
```

```
print(users_df.isnull().sum()) #Cheking the Users file if there is any null
value

#Cheking for duplicate in the users data
print("\nDuplicate usernames in users data:",
      users_df['Username'].duplicated().sum())

#Checking for missing values in reviews data
print("\nMissing values in reviews data:\n")
print(reviews_df.isnull().sum()) #Cheking the Reviews file it htere is an
null value
print("\nDuplicate usernames in reviews data:",
      reviews_df['Username'].duplicated().sum()) # Checking for duplicates
in reviews file
```

```
Misising values in user data:

Username    0
DOB         0
State       0
dtype: int64

Duplicate usernames in users data: 1

Missing values in reviews data:

Username    0
DOB         0
State       0
Reviewed    0
dtype: int64

Duplicate usernames in reviews data: 1
```

Converting Date of Birth(DOB) to Data Type for easier analysis

In [183]:

```
#Converting DOB column from text to date time for both dataframes
users_df['DOB'] = pd.to_datetime(users_df['DOB'], format='%d.%m.%Y',
errors='coerce' )
reviews_df['DOB'] = pd.to_datetime(reviews_df['DOB'], format='%d.%m.%Y',
errors='coerce' )
```

Quick check for Data Matching

In [184]:

```
#Checking if the usernames present in reviews are also present in users as
well, so we could merge it later, after matching it.
review_usernames = set(reviews_df['Username'])
```

```
user_usernames = set(users_df['Username'])

#Finding any usernames in reviews that is missing in users
missing_users = review_usernames - user_usernames

#printing out the number of missing usernames
print("Usernames in reviews but not users file:", len(missing_users))
Usernames in reviews but not users file: 0
```

Merging Users and Reviews

In [185]:

```
#Merging the uers reviews and users dataframe on username
df = pd.merge(reviews_df, users_df, on='Username', how='inner')

#Counting total no. of reviews for per user
df['num_reviews'] = df.groupby('Username')['Reviewed'].transform('count')

print(df.head()) # Printing the rows of the merged dataframe
print(df.columns) # Printing all the columns names to easily analyse the
available data
   Username      DOB_x        State_x  \
0  bkpn1412 1983-07-31         Oregon
1  gqjs4414 1998-07-27  Massachusetts
2  eehe1434 1950-08-08          Idaho
3  hkxj1334 1969-08-03        Florida
4  jjbd1412 2001-07-26        Georgia

                                          Reviewed      DOB_y  \
0                [cea76118f6a9110a893de2b7654319c0] 1983-07-31
1                [fa04fe6c0dd5189f54fe600838da43d3] 1998-07-27
2                                                [] 1950-08-08
3  [f129b1803f447c2b1ce43508fb822810, 3b0c9bc0be6... 1969-08-03
4                                                [] 2001-07-26

         State_y  num_reviews
0         Oregon            1
1  Massachusetts            1
2          Idaho            1
3        Florida            1
4        Georgia            1
Index(['Username', 'DOB_x', 'State_x', 'Reviewed', 'DOB_y', 'State_y',
       'num_reviews'],
      dtype='object')
```

Removing Duplicates from Users and Reviews

In [186]:

```
#Dropping the duplicate users and reviews data to see if the duplicates and
errors are gone
users_df = users_df.drop_duplicates(subset='Username')
reviews_df = reviews_df.drop_duplicates(subset='Username')

#Showing number of rows left after dropping the duplicates
print("Users data shape: ", users_df.shape)
print("Reviews data shape: ", reviews_df.shape)
Users data shape:  (4999, 3)
Reviews data shape:  (4999, 4)
```

Checking and Summerising Cleaned Data

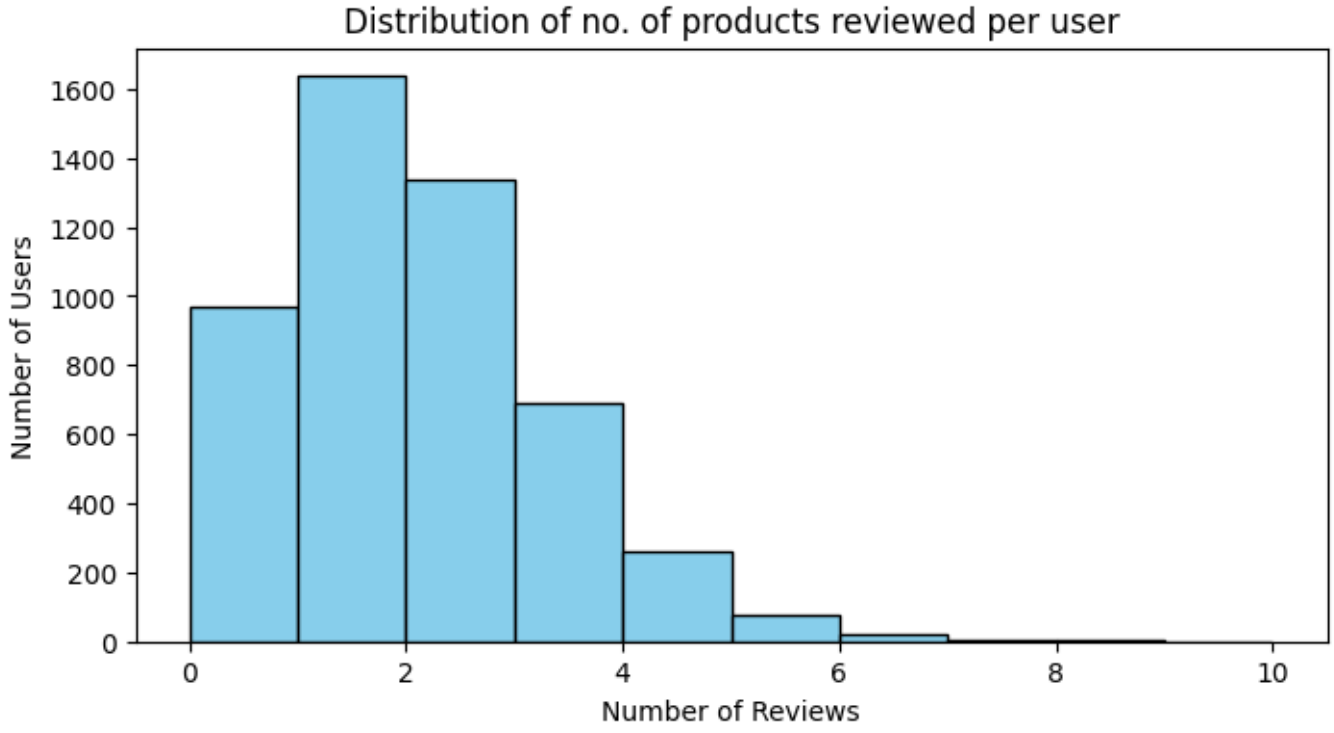
In [187]:

```
#Checking the cleaned users data and reviews data
print("Cleaned users data (first 5 rows):\n")
print(users_df.head())

# Print the first 5 rows of the cleaned reviews data
print("\nCleaned reviews data (first 5 rows):\n")
print(reviews_df.head())
Cleaned users data (first 5 rows):

   Username        DOB          State
0  bkpn1412 1983-07-31         Oregon
1  gqjs4414 1998-07-27  Massachusetts
2  eehe1434 1950-08-08          Idaho
3  hkxj1334 1969-08-03        Florida
4  jjbd1412 2001-07-26        Georgia

Cleaned reviews data (first 5 rows):

   Username        DOB          State  \
0  bkpn1412 1983-07-31         Oregon
1  gqjs4414 1998-07-27  Massachusetts
2  eehe1434 1950-08-08          Idaho
3  hkxj1334 1969-08-03        Florida
4  jjbd1412 2001-07-26        Georgia

                                            Reviewed
0                 [cea76118f6a9110a893de2b7654319c0]
1                 [fa04fe6c0dd5189f54fe600838da43d3]
2                                                 []
3  [f129b1803f447c2b1ce43508fb822810, 3b0c9bc0be6...
4                                                 []
```

Data Visualization

In [188]:

```
#Counting the number of products reviewed by each user.
reviews_df['num_reviews'] = reviews_df['Reviewed'].apply(len)

#Showing basic stats about the review activity
print("Average no. of products reviewed per user:",
      reviews_df['num_reviews'].mean())
print("Max products reviewed by per user:",
      reviews_df['num_reviews'].max())
print("Users with zero reviews:", (reviews_df['num_reviews'] == 0).sum())

#Plotting Histogram to gain insights about distribution or review count per
user
plt.figure(figsize=(8,4))
plt.hist(reviews_df['num_reviews'],
bins=range(0,reviews_df['num_reviews'].max()+2), color='skyblue',
edgecolor='black')
plt.title('Distribution of no. of products reviewed per user')
plt.xlabel('Number of Reviews')
plt.ylabel('Number of Users')
plt.show()
Average no. of products reviewed per user: 1.5965193038607721
Max products reviewed by per user: 9
Users with zero reviews: 971
```



States With Most Active Reviewers

In [189]:

```
#Finding top 5 states by total number of reviews
```

```
state_review_counts =
reviews_df.groupby('State')['num_reviews'].sum().sort_values(ascending=Fals
e)

#Print the top 5 states with most reviews
print("\nTop 5 States by total number of product reviews:\n")
print(state_review_counts.head(5))
```

```
Top 5 States by total number of product reviews:

State
Kentucky         177
Vermont          175
New Jersey       174
Massachusetts    173
Oklahoma         170
Name: num_reviews, dtype: int64
```

Finding which users are most active

In [190]:

```
#Finding top 5 reviewers by number of products the have reviewed
top_reviewers = reviews_df.sort_values('num_reviews',
ascending=False)[['Username','num_reviews']].head(5)

#Print top 5 most active users
print("Top 5 users by number of reviews:\n")
print(top_reviewers)
```

```
Top 5 users by number of reviews:

      Username  num_reviews
2583  igqu1322            9
922   jyqm4141            8
2074  gaeq2421            8
1197  trbs1413            7
3098  fuss4231            7
```

Finding % of inactive users

In [191]:

```
#Finding % of users who never left any review
inactive_percentage = (reviews_df['num_reviews'] == 0).mean()*100

#Printing out the inactive users [rounded to two decimal places]
print(f"% of users never reviewed product: {inactive_percentage:.2f}%")
```

```
% of users never reviewed product: 19.42%
```

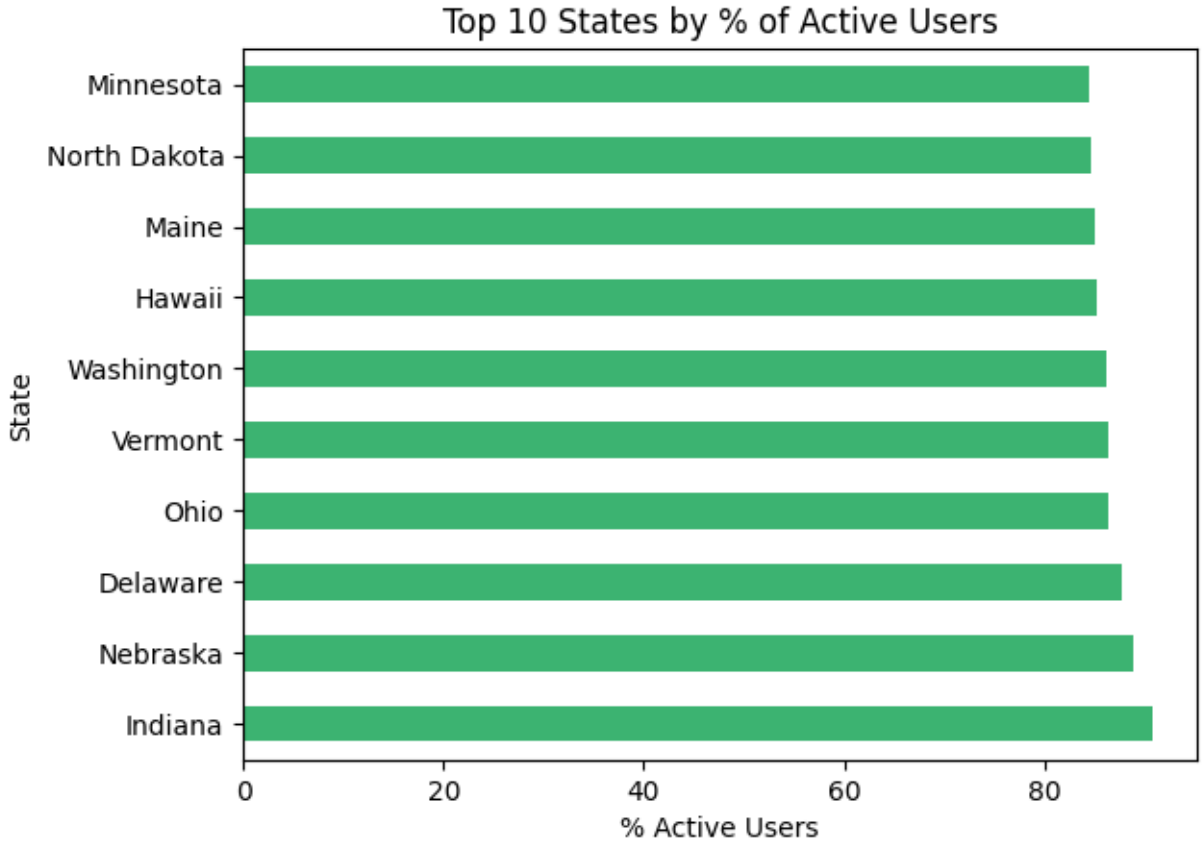
Creating a horizontal bar chart to find Top 10 states by % of active users

In [192]:

```
#Creating a new column called "Activity " that labels users as "Inactive"
or "Active".
reviews_df['Activity'] = reviews_df['num_reviews'].apply(lambda x:
'Inactive' if x==0 else 'Active')
```

In [193]:

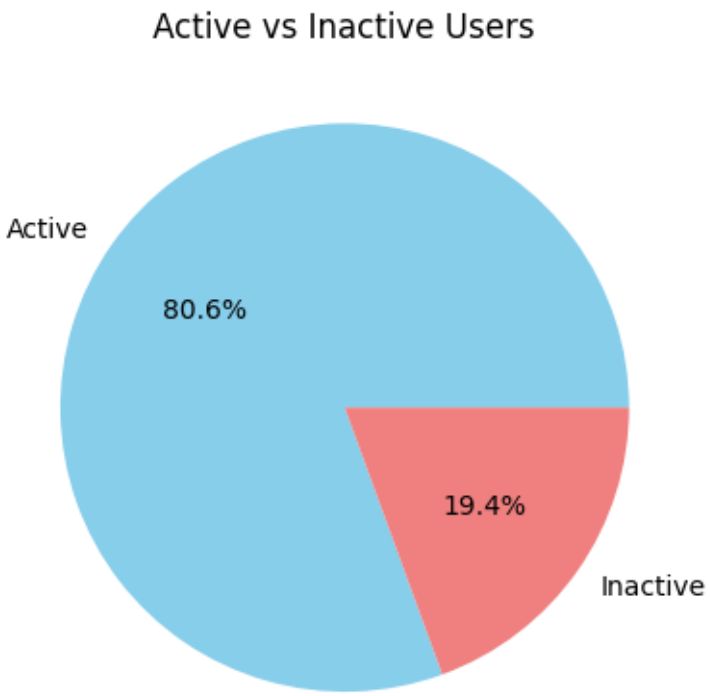
```
#First i'll calculate the % of active users in each satate
percent_active_by_state =
(reviews_df.groupby('State')['Activity'].apply(lambda x: (x==
'Active').mean()*100)).sort_values(ascending=False).head(10) #to find top
10 states)

#plot horizontal ar chart
percent_active_by_state.plot(kind='barh', color = 'mediumseagreen')
plt.xlabel('% Active Users')
plt.title('Top 10 States by % of Active Users')
plt.show()
```



Creating Pie Chart: Active Users and Inactive users to find overall engagement

In [194]:

```
#Count active and inactive users overall
activity_counts = reviews_df['Activity'].value_counts()

#Pie chart to see the engagement
activity_counts.plot.pie(
    autopct = '%.1f%%',
    colors = ['skyblue', 'lightcoral']
)

plt.title('Active vs Inactive Users')
plt.ylabel('')
plt.show()
```

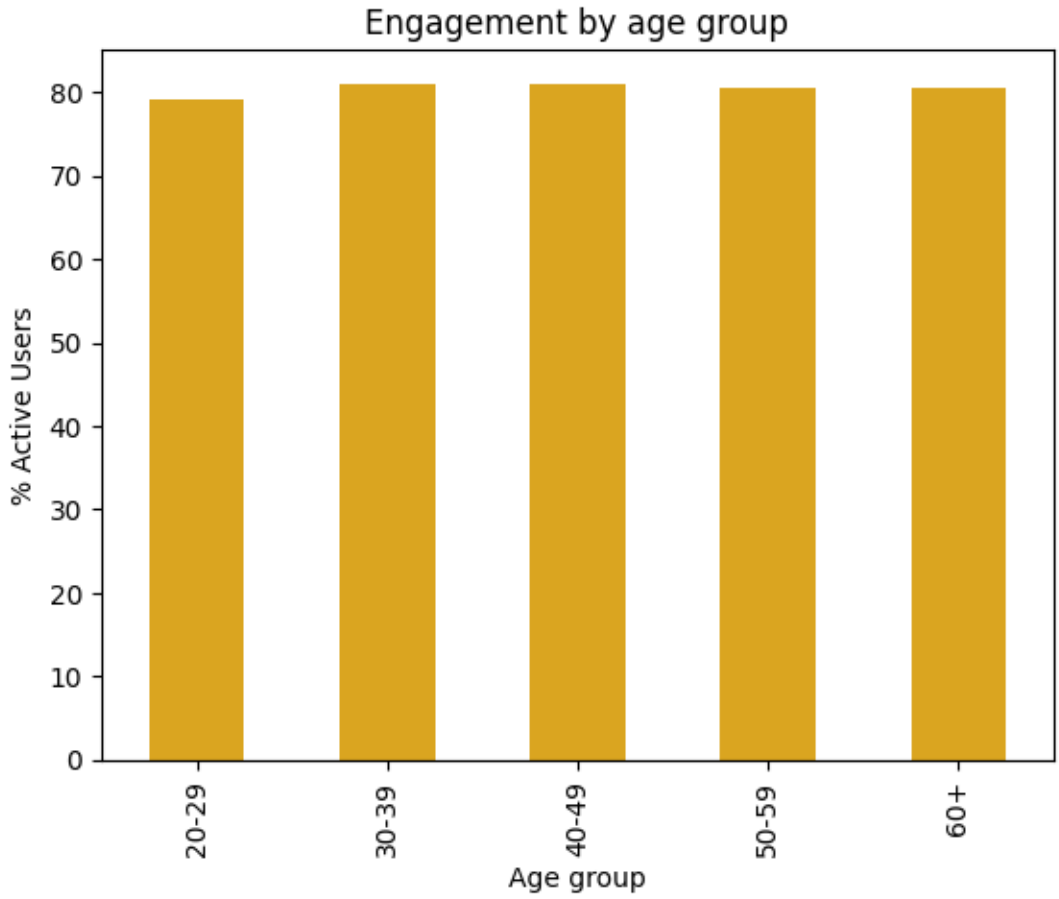


Age Group Engagement

Finding out which age group are most engaged which can be helpful for the targeting campaigns for like marketing purpose and other purpose as well.

In [195]:

```
#Cheking if 'DOB_x' and 'Reviewed' columns exist in dataframe before
processing

if 'DOB_x' in df.columns and 'Reviewed' in df.columns:
  today = pd.to_datetime('today') #Calculating the age in years from DOB to
today
  df['age'] = (today-pd.to_datetime(df['DOB_x'])).dt.days // 365

  #Removing the rows where age could not be calculated
  df = df.dropna(subset=['age'])

  # Creating age group bins as categorical ranges
  bins = [0,20,30,40,50,60,100]
  labels = ['20','20-29','30-39','40-49','50-59','60+']
  df['age_group'] = pd.cut(df['age'], bins=bins, labels=labels,
right=False)

  #Creating the flag to show if the user has reviewed atleat one product.
  df['Reviewed_flag'] = df['Reviewed'].apply(lambda x: isinstance(x, list)
and len(x) > 0)

  #Finding engagement according to % of active users as per the age groups
  engagement_by_age = df.groupby('age_group',
observed=True)['Reviewed_flag'].mean() * 100

  #Plotting the engagement by age group as bar chart
  engagement_by_age.plot(kind='bar', color='goldenrod')
  plt.title('Engagement by age group')
  plt.xlabel('Age group')
  plt.ylabel('% Active Users')
  plt.show()
else:
    print("Columns 'DOB_x' and 'Activity' not found in the DataFrame.")
```



Price Band Vs Average Review Score Heatmap

Visualising how the review counts distribute across price bands ranges using heatmap for density.

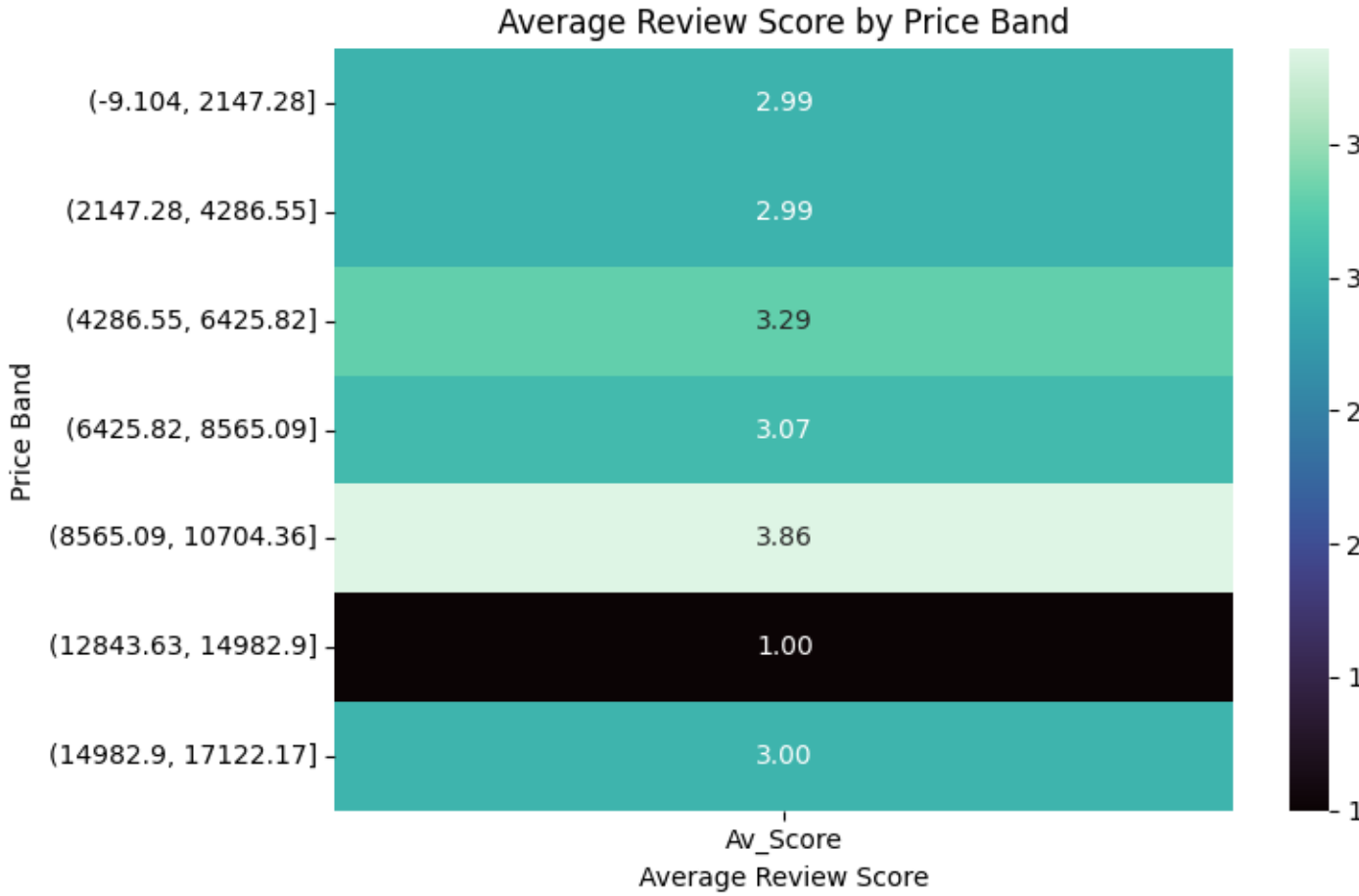
In [196]:

```
productsdf = productsdf[productsdf['Price'] >= 0]
```

In [197]:

```
#Reloding the original products data
productsdf = pd.read_csv('/content/products.csv')

#Removing products with negative prices
productsdf = productsdf[productsdf['Price'] >= 0]

#Creating a manual bin to start from 0 and show upto max price
max_price = productsdf['Price'].max()
bins = [0, max_price/8, max_price*2/8, max_price*3/8, max_price*4/8,
max_price*5/8, max_price*6/8, max_price*7/8, max_price]
#Analyzing how product qualit varies across different price range
#Bin the product price into 8 bands for analysis
productsdf['price_band'] = pd.cut(productsdf['Price'], bins=8)

#Calculating the average review score for price band
```

```
heatmap_data = productsdf.groupby('price_band',
observed=False)['Av_Score'].mean().reset_index()

#Converting summer into matrix format for the heatmap purpose
heatmap_matrix = heatmap_data.pivot_table(values='Av_Score',
index='price_band', observed=False)

#Creating heatmap plot
plt.figure(figsize=(8,5))
sns.heatmap(heatmap_matrix, annot=True, fmt='.2f', cmap='mako') #annot
displays the value, cmap sets color stle
plt.title('Average Review Score by Price Band')
#x-axis label
plt.xlabel('Average Review Score')
#y-axis label
plt.ylabel('Price Band')
plt.tight_layout() #prevents from overlapping of labels and titles
plt.show() #display the plot
```



References :

1. McKinney, W., 2024. pandas documentation: pandas.read_csv. Available at: https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html

2. McKinney, W., 2024. pandas documentation: pandas.cut. Available at: https://pandas.pydata.org/docs/reference/api/pandas.cut.html

3. Waskom, M., 2024. seaborn documentation: seaborn.heatmap. Available at: https://seaborn.pydata.org/generated/seaborn.heatmap.html