



Practical Machine Learning and Data Exploration

Project 1: Gym Exercise

Date November 2025

Course Title: Practical Machine Learning and Data Exploration

Weighting: 50%

Lecturer: Dr. Hager Saleh

Hand-out Date: 1st Nov 2025

Hand-in Date: 15th Nov 2025

Hand-in Method: Online Submission via E-Learning.

Student ID: 253000598

Student Name: Ahmed Abdullah

Assignment Report: Project (1): Gym Exercise

Table of Contents

Introduction	3
Gym Exercise Data Set	3
Data Cleansing	4
Dataset Description	4
Data Visualization	7
Correlation heatmap	7
Bar Chart (1)	8
Bar Chart (2)	9
Pie Chart	12
Line Chart	13
Count Plot	14
ScatterPlot	15
BoxPlot	16
Appendix (1): More Visualization	17

Introduction

In This assignment, we will explores the Gym Exercise Dataset that we downloaded from Keggel, focusing on analyzing and visualizing exercise information such as type, level, equipment, and ratings.

Using Python libraries like Pandas, Seaborn, and Matplotlib, we perform the following exercises:

- Data cleaning.
- Statistical summaries
- Visualizations.

Gym Exercise Data Set

Our Gym Exercise dataset has the following fields

#	Title	Desc	Type	BodyPart	Equipment	Level	Rating	RatingDesc
0	Partner plank band row	The partner plank band row is an abdominal exercise where two partners perform single-arm planks while pulling on the opposite ends of an exercise band. This technique can be done for time or reps in any ab-focused workout.	Strength	Abdominals	Bands	Intermediate	0	
1	Banded crunch isometric hold	The banded crunch isometric hold is an exercise targeting the abdominal muscles, particularly the rectus abdominis..etc.	Strength	Abdominals	Bands	Intermediate		

Data Cleansing

Dataset Description

Step 1: Loads the gym dataset into a Pandas data frame for analysis.

```
df = pd.read_csv('megaGymDataset.csv')
```

Step 2: Shows the first five rows of the dataset for quick preview.

```
df.head()
```

Result

```
df.head()
```

	Unnamed: 0	Title	Desc	Type	BodyPart	Equipment	Level	Rating	RatingDesc
0	0	Partner plank band row	The partner plank band row is an abdominal exe...	Strength	Abdominals	Bands	Intermediate	0.0	NaN
1	1	Banded crunch isometric hold	The banded crunch isometric hold is an exercis...	Strength	Abdominals	Bands	Intermediate	NaN	NaN
2	2	FYR Banded Plank Jack	The banded plank jack is a variation on the pl...	Strength	Abdominals	Bands	Intermediate	NaN	NaN
3	3	Banded crunch	The banded crunch is an exercise targeting the...	Strength	Abdominals	Bands	Intermediate	NaN	NaN
4	4	Crunch	The crunch is a popular core exercise targetin...	Strength	Abdominals	Bands	Intermediate	NaN	NaN

Step 3: Getting features type

```
df.dtypes
```

Result

```
[29]: df.dtypes
```

```

[29]: Unnamed: 0      int64
      Title         object
      Desc         object
      Type         object
      BodyPart      object
      Equipment     object
      Level         object
      Rating      float64
      RatingDesc   object
      dtype: object

```

Step 4: Print columns

```
df.columns
```

Result

```
df.columns
```

```

Index(['car_ID', 'symboling', 'CarName', 'fueltype', 'aspiration',
      'doornumber', 'carbody', 'drivewheel', 'enginelocation', 'wheelbase',
      'carlength', 'carwidth', 'carheight', 'curbweight', 'enginetype',
      'cylindernumber', 'enginesize', 'fuelsystem', 'boreratio', 'stroke',
      'compressionratio', 'horsepower', 'peakrpm', 'citympg', 'highwaympg',
      'price'],
      dtype='object')

```

Step 5: Getting the count of missing values

```
df.isnull().sum()
```

Result

```
df.isnull().sum()
```

```
Unnamed: 0      0
Title           0
Desc          1550
Type           0
BodyPart       0
Equipment      32
Level          0
Rating        1887
RatingDesc     2056
dtype: int64
```

Step 6: Deleting columns that have 30% missing values

Python Code

```
missing_percentage = df.isnull().mean() * 100
# Set threshold (e.g., 30%)
threshold = 30
# Identify columns to drop
columns_to_drop = missing_percentage[missing_percentage > threshold].index
# Drop those columns
df = df.drop(columns=columns_to_drop)
# Print results
print(f"Columns dropped: {list(columns_to_drop)}")
print(f"Remaining columns: {list(df.columns)}")
# Optionally, show the shape after dropping
print(f"\nDataset shape after dropping: {df.shape}")
```

Result

```
Columns dropped: ['Desc', 'Rating', 'RatingDesc']
Remaining columns: ['Unnamed: 0', 'Title', 'Type', 'BodyPart', 'Equipment', 'Level']
Dataset shape after dropping: (2918, 6)
```

Step 7: Pre-processing on dataset

Python Code

```
from sklearn.preprocessing import LabelEncoder
import pandas as pd
df = pd.read_csv("megaGymDataset.csv")
label_encoder = LabelEncoder()
for column in df.select_dtypes(include=['object']).columns:
    df[column] = label_encoder.fit_transform(df[column].astype(str))
df.head()
```

Result

	Unnamed: 0	Title	Desc	Type	BodyPart	Equipment	Level	Rating	RatingDesc
0	0	2078	696	4	0	0	2	0.0	1
1	1	323	139	4	0	0	2	NaN	1
2	2	803	143	4	0	0	2	NaN	1
3	3	322	138	4	0	0	2	NaN	1
4	4	590	298	4	0	0	2	NaN	1

Step 8: Filling missing values based on feature type

Python Code

```
df.fillna(df.mean(numeric_only=True), inplace=True)
for col in df.select_dtypes(include='object').columns:
    df[col].fillna(df[col].mode()[0], inplace=True)
df.isnull().sum()
```

Result

```
Unnamed: 0    0
Title         0
Desc          0
Type          0
BodyPart      0
Equipment     0
Level         0
Rating        0
RatingDesc    0
dtype: int64
```

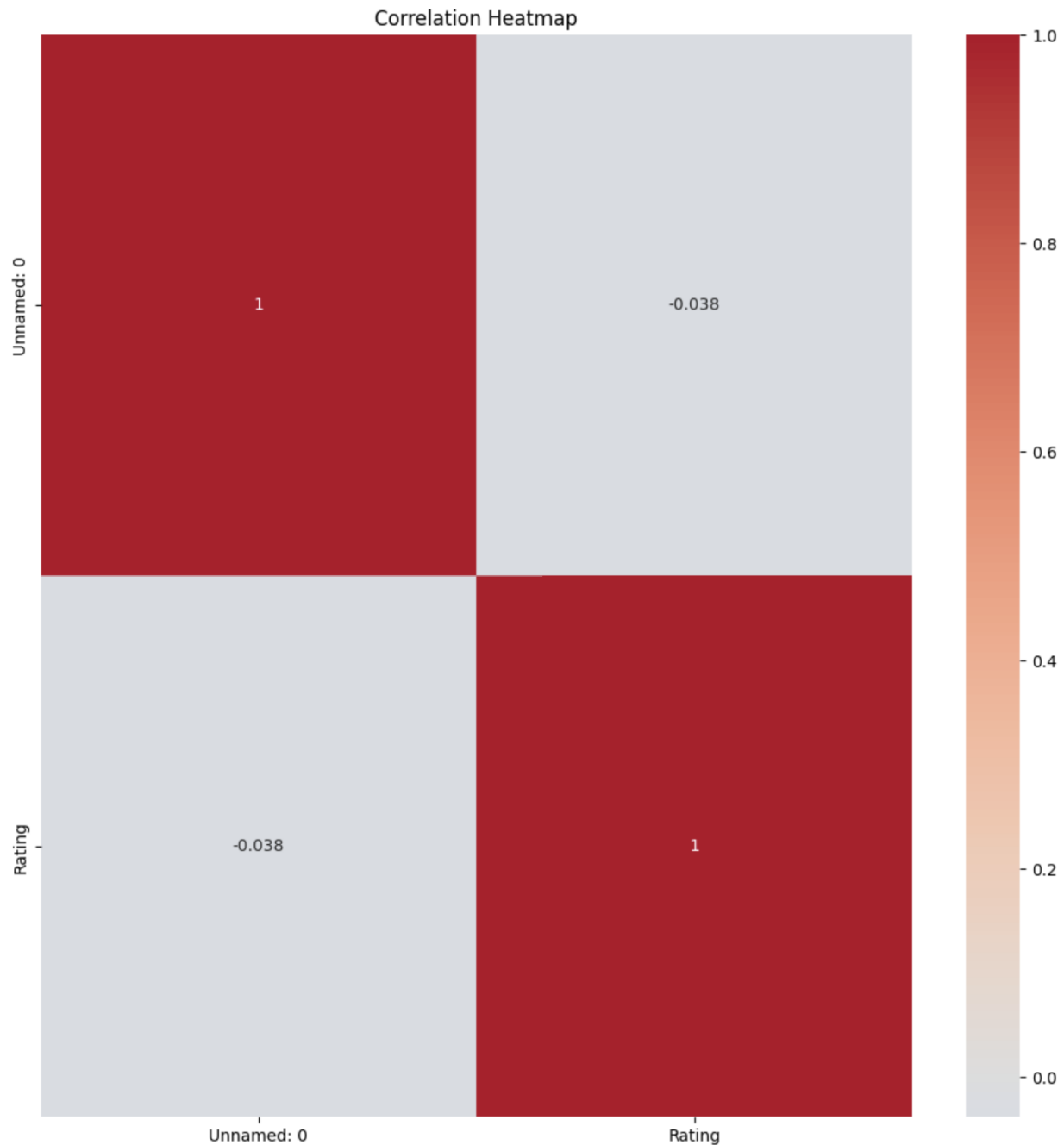
Data Visualization

Correlation heatmap

Python Code

```
plt.figure(figsize=(12, 12))
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm', center=0)
plt.title('Correlation Heatmap')
plt.show()
```

Result

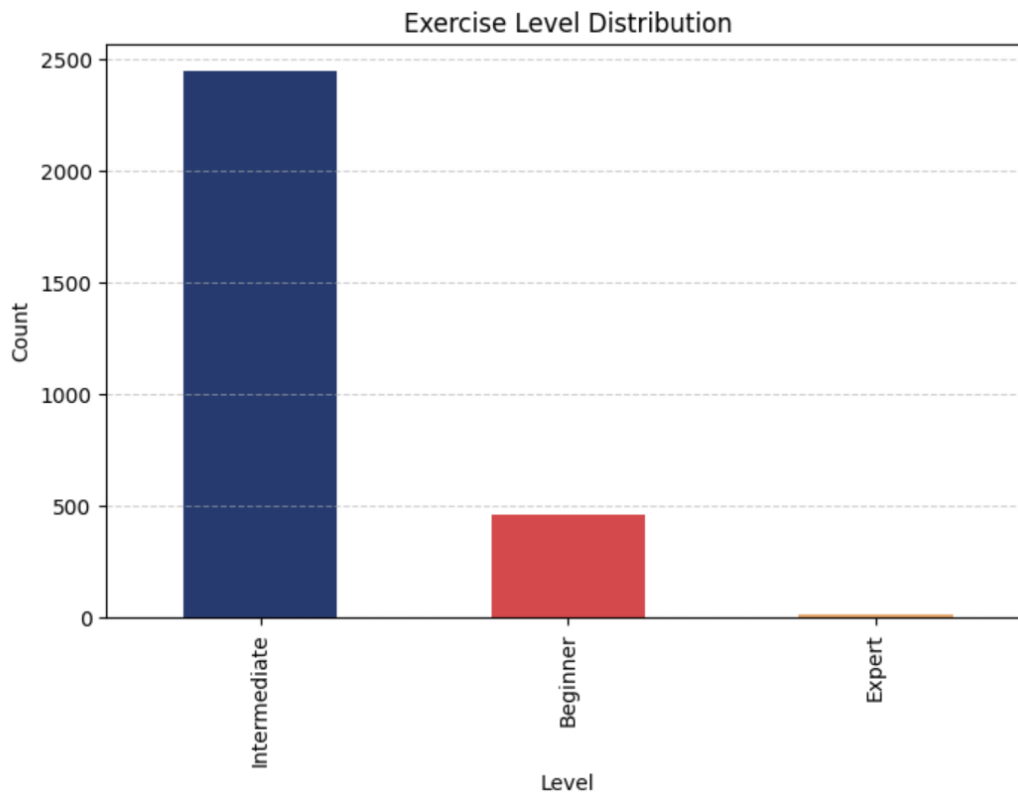


Bar Chart (1)

Python Code

```
# Create a bar chart for Level distribution
plt.figure(figsize=(8, 5))
df['Level'].value_counts().plot(kind='bar', color=['#1f3b73', '#e63946', '#f4a261'])
plt.title('Exercise Level Distribution')
plt.xlabel('Level')
plt.ylabel('Count')
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.show()
```

Result

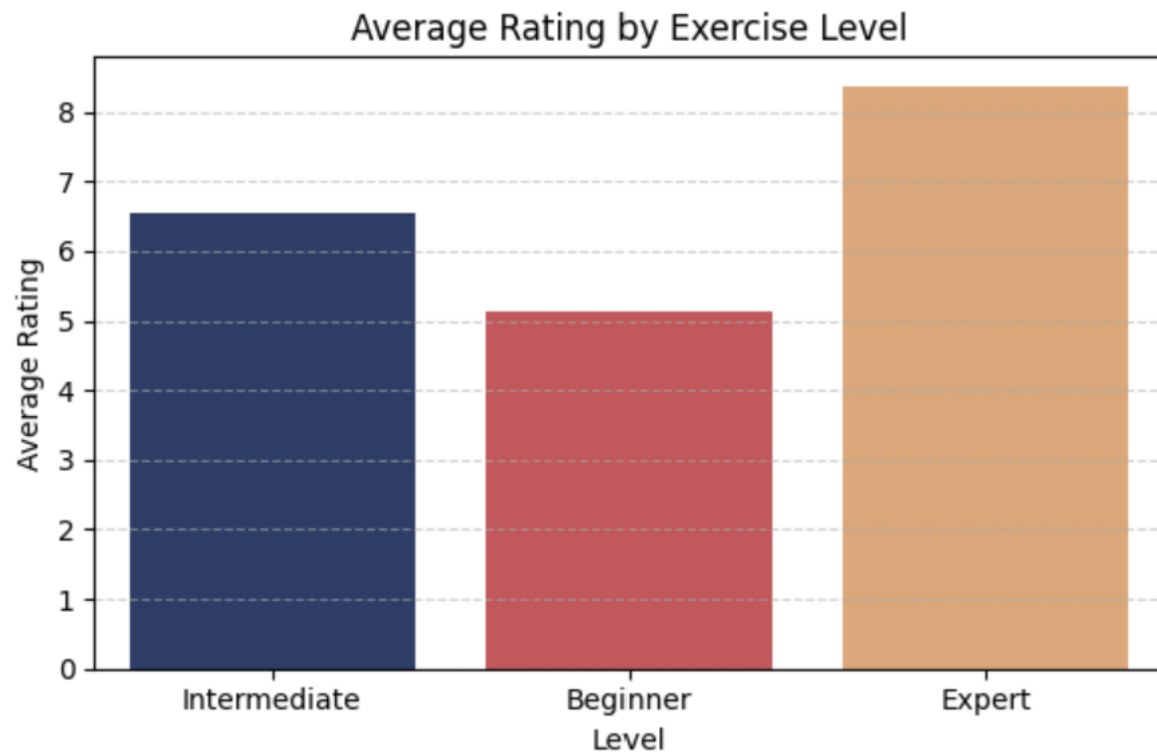


Bar Chart (2)

Python Code

```
plt.figure(figsize=(6, 4))
sns.barplot(x='Level', y='Rating', data=df, estimator='mean', ci=None, palette=['#1f3b73', '#e63946', '#f4a261'])
plt.title("Average Rating by Exercise Level")
plt.xlabel("Level")
plt.ylabel("Average Rating")
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.tight_layout()
plt.show()
```

Result



Python Code

```

cat_cols = [col for col in df.columns if df[col].dtype == 'object']
num_cols = [col for col in df.columns if df[col].dtype != 'object']
for col in cat_cols:
    print(f'{col} has {df[col].nunique()} unique values')
plt.figure(figsize=(8, 6))
sns.histplot(df['Rating'].dropna(), bins=15, kde=True, color='blue')
plt.title('Rating Distribution')
plt.xlabel('Rating')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()

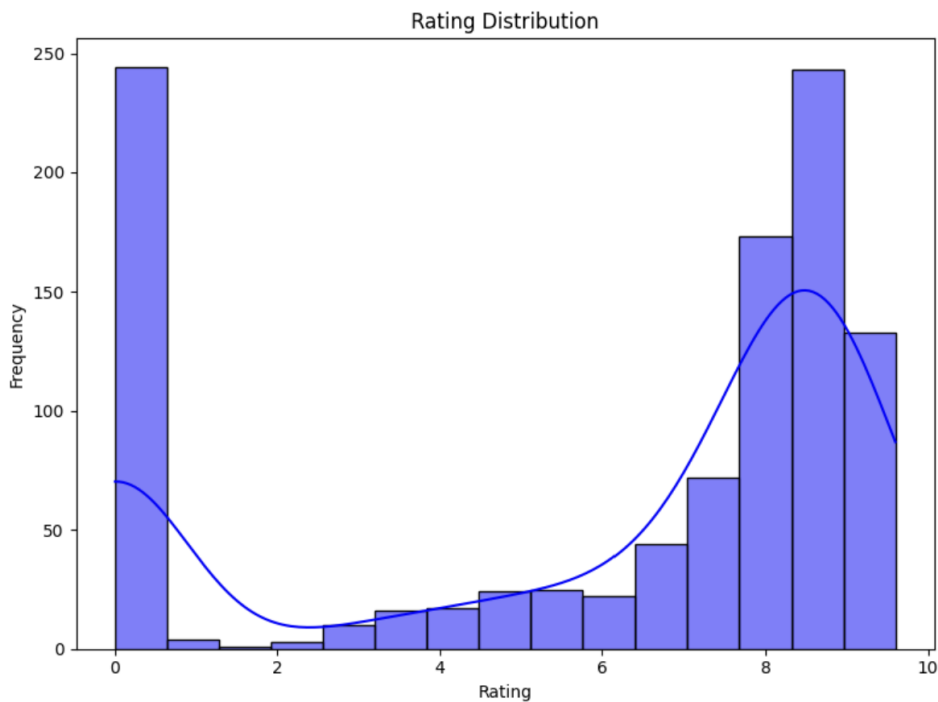
```

Result

```

Title has 2909 unique values
Desc has 1050 unique values
Type has 7 unique values
BodyPart has 17 unique values
Equipment has 12 unique values
Level has 3 unique values
RatingDesc has 1 unique values

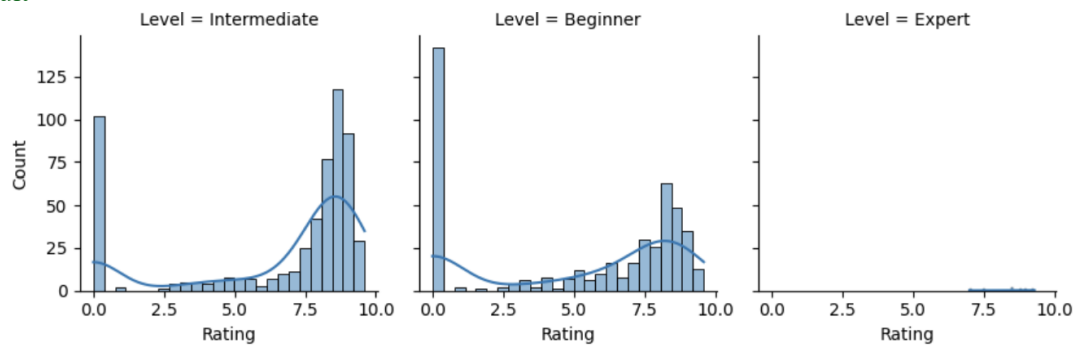
```



Python Code

```
g = sns.FacetGrid(df, col="Level")
g.map(sns.histplot, "Rating", bins=25, kde=True)
plt.show()
```

Result

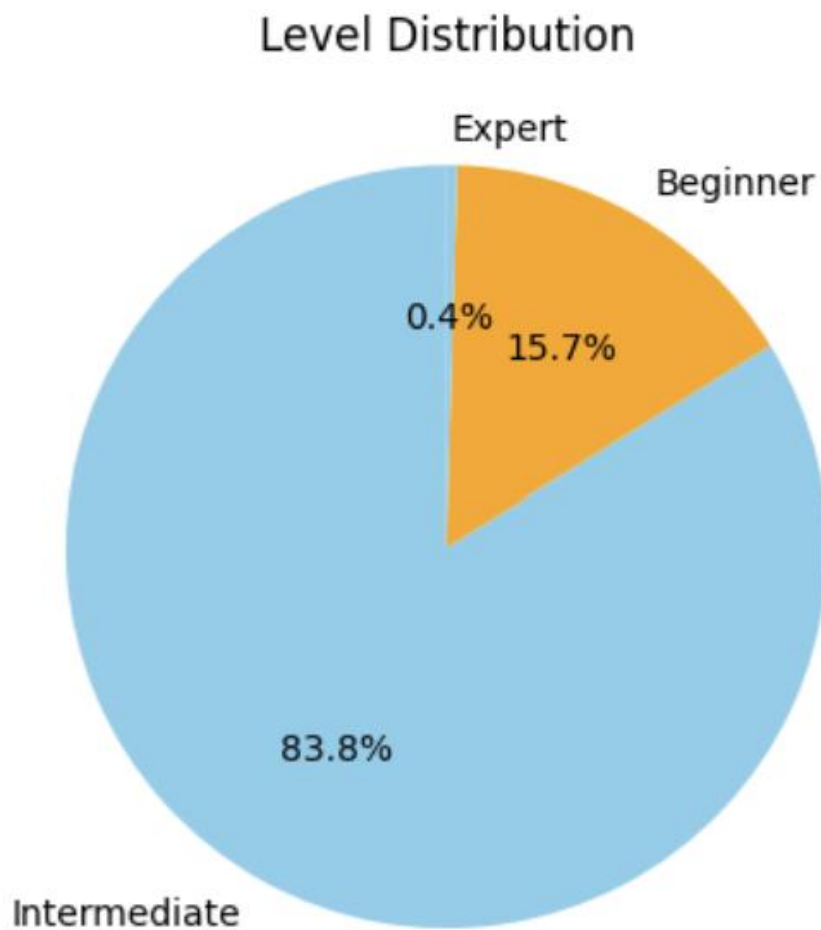


Pie Chart

Python Code

```
plt.pie(df['Level'].value_counts(),
        labels=df['Level'].value_counts().index,
        autopct='%1.1f%%',
        startangle=90,
        colors=['skyblue', 'orange'])
plt.title('Level Distribution')
plt.show()
```

Result

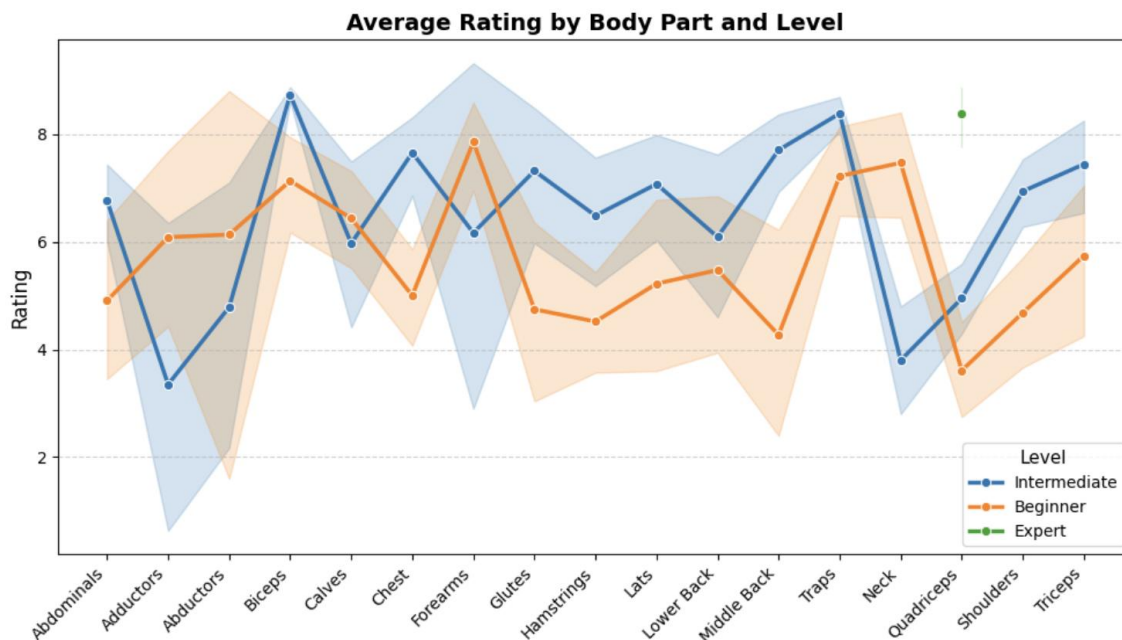


Line Chart

Python Code

```
plt.figure(figsize=(10, 6))
sns.lineplot(x='BodyPart', y='Rating', hue='Level', data=df, marker='o', linewidth=2.5)
plt.title("Average Rating by Body Part and Level", fontsize=14, fontweight='bold')
plt.xlabel("Body Part", fontsize=12)
plt.ylabel("Rating", fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.legend(title="Level", fontsize=10, title_fontsize=11)
plt.tight_layout()
plt.show()
```

Result



Observation

The chart shows that intermediate exercises have the highest ratings across most body parts, followed by beginner and expert levels.

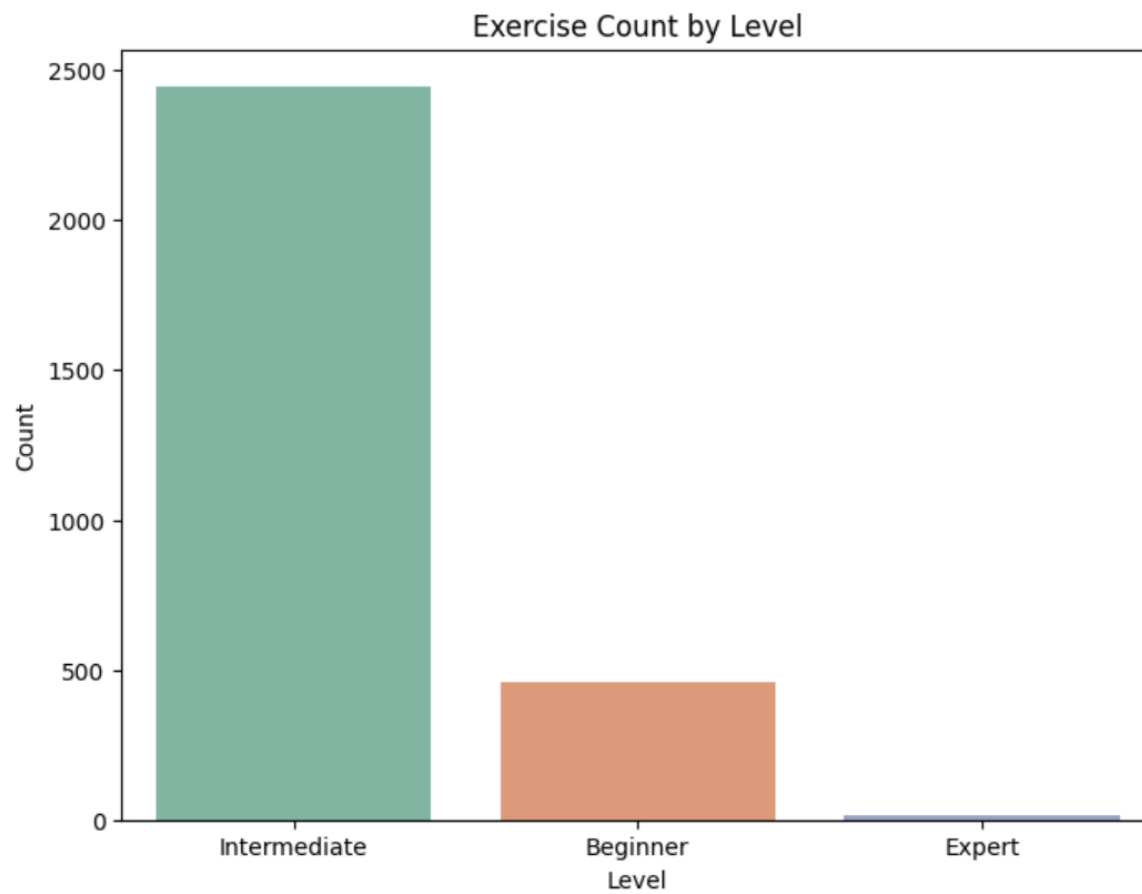
Ratings peak around biceps, chest, and shoulders, suggesting these areas are the most popular and well-rated among users.

Count Plot

Python Code

```
plt.figure(figsize=(8, 6))
sns.countplot(x='Level', data=df, palette='Set2')
plt.title('Exercise Count by Level')
plt.xlabel('Level')
plt.ylabel('Count')
plt.show()
```

Result

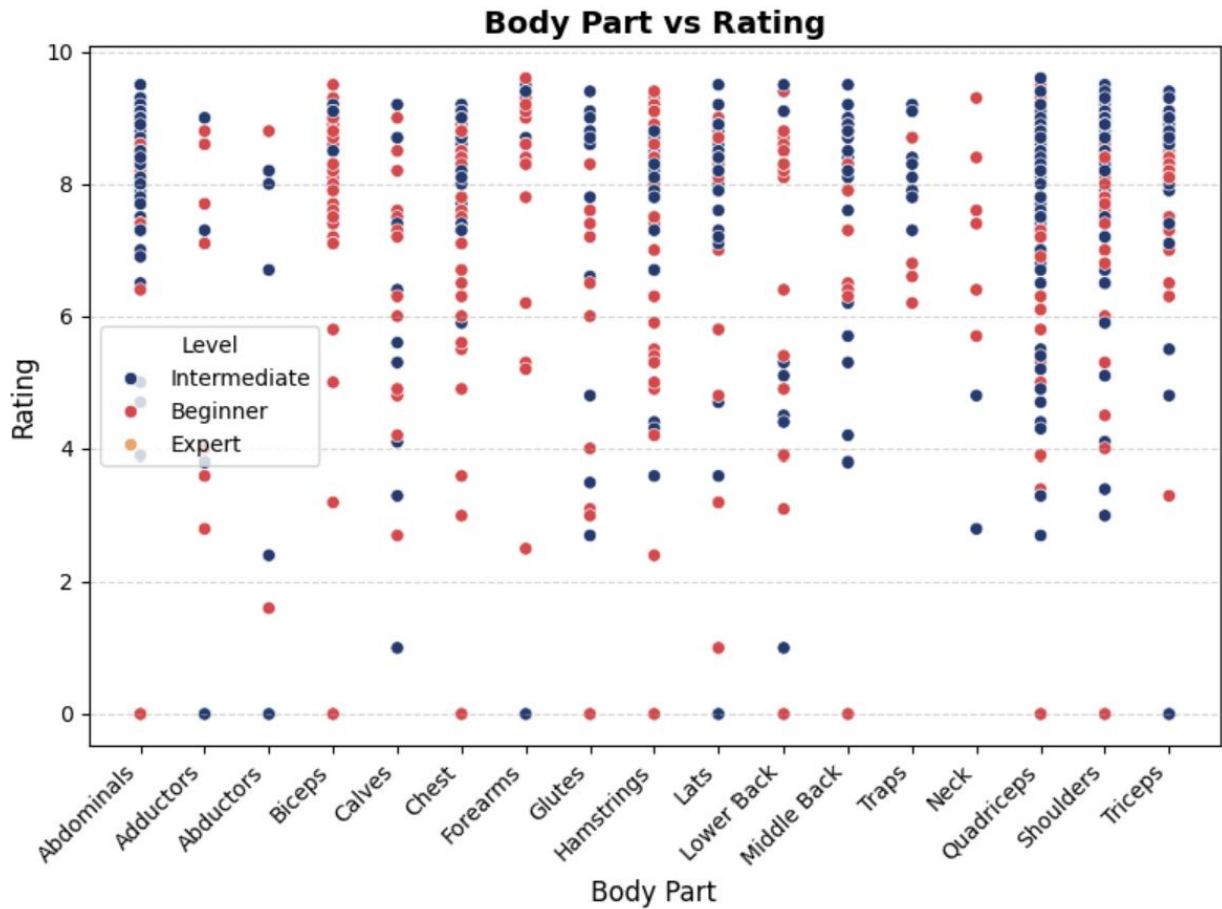


ScatterPlot

Python Code

```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='BodyPart', y='Rating', data=df, hue='Level', palette=['#1f3b73', '#e63946', '#f4a261'])
plt.title('Body Part vs Rating', fontsize=14, fontweight='bold')
plt.xlabel('Body Part', fontsize=12)
plt.ylabel('Rating', fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

Result

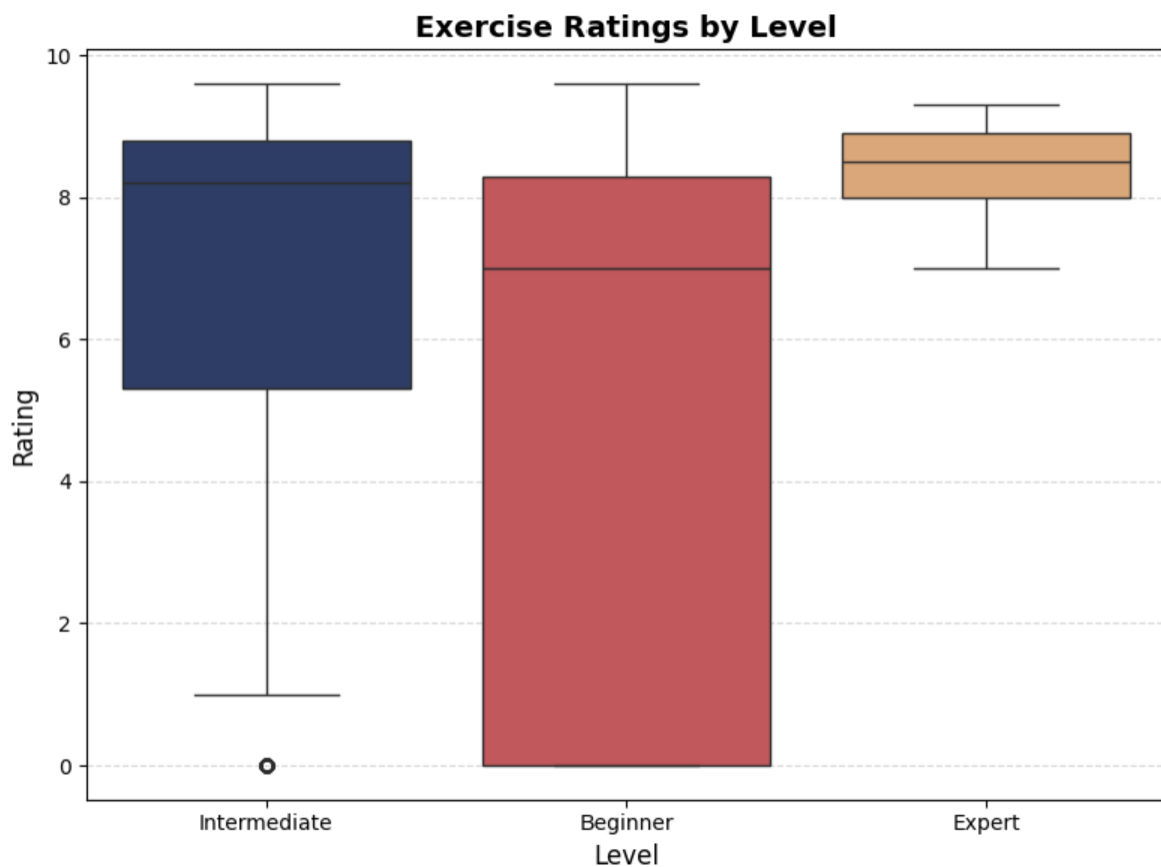


BoxPlot

Python Code

```
plt.figure(figsize=(8, 6))
sns.boxplot(x='Level', y='Rating', data=df, palette=['#1f3b73', '#e63946', '#f4a261'])
plt.title('Exercise Ratings by Level', fontsize=14, fontweight='bold')
plt.xlabel('Level', fontsize=12)
plt.ylabel('Rating', fontsize=12)
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

Result



Appendix (1): More Visualization

Distribution of Exercises by Level

Python Code

```
# Choose a categorical column to visualize
col = 'Level' # You can also try 'Type' or 'BodyPart'

# Colors (athletic theme)
colors = ['#1f3b73', '#e63946', '#f4a261'] # navy, red, orange

# Order based on data
order = df[col].value_counts().index

# --- Set up the figure ---
plt.figure(figsize=(14, 6))
plt.suptitle(f'{col} Distribution', fontsize=16, fontweight='bold')

# --- Pie Chart ---
plt.subplot(1, 2, 1)
plt.title('Pie Chart', fontsize=14)
plt.pie(df[col].value_counts(),
        labels=df[col].value_counts().index,
        colors=colors[:len(df[col].unique())],
        autopct='%1f%%',
        startangle=90)

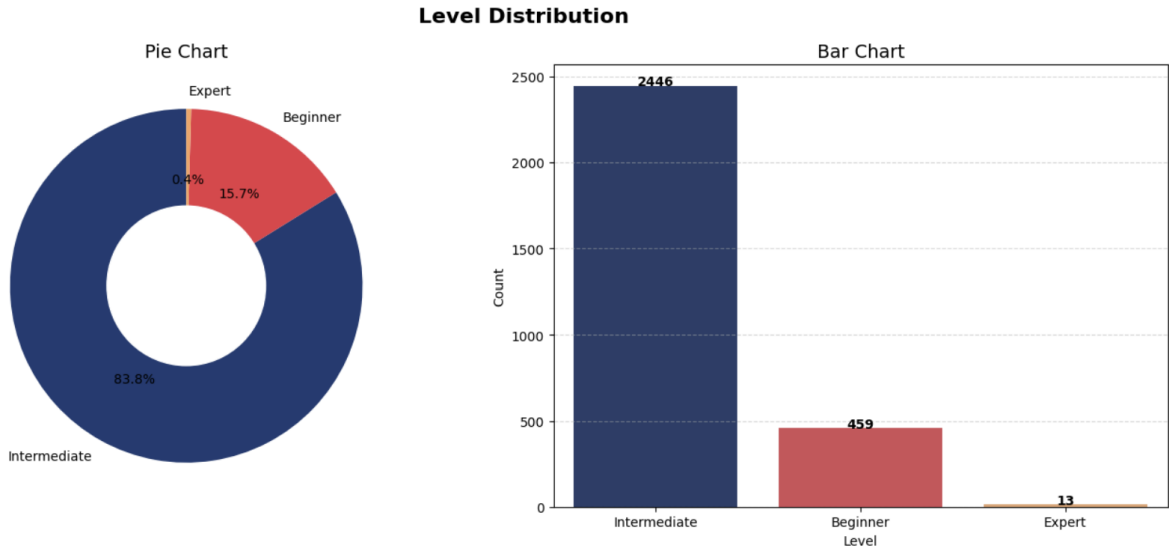
# Donut hole
centre = plt.Circle((0, 0), 0.45, color='white')
plt.gca().add_artist(centre)

# --- Bar Chart ---
plt.subplot(1, 2, 2)
plt.title('Bar Chart', fontsize=14)
ax = sns.countplot(x=col, data=df, palette=colors, order=order)

# Add count labels on bars
for bar in ax.patches:
    height = bar.get_height()
    ax.text(bar.get_x() + bar.get_width()/2, height + 1, int(height),
            ha='center', fontsize=10, fontweight='bold')

plt.xlabel(col)
plt.ylabel('Count')
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

Result



Exercise Type Distribution (Pie + Bar Chart)

Python Code

```

colors = ['#1f3b73', '#e63946', '#f4a261', '#2a9d8f', '#264653'] # athletic color theme
labels = df['Type'].unique()
order = df['Type'].value_counts().index # bar order by frequency

plt.figure(figsize=(16, 8))
plt.suptitle('Exercise Type Distribution', fontsize=16, fontweight='bold')

plt.subplot(1, 2, 1)
plt.title('Pie Chart', fontsize=14)
plt.pie(df['Type'].value_counts(),
        labels=df['Type'].value_counts().index,
        colors=colors[:len(df['Type'].unique())],
        autopct='%0.1f%%',
        startangle=90)
centre = plt.Circle((0, 0), 0.45, color='white') # donut-style center
plt.gca().add_artist(centre)

plt.subplot(1, 2, 2)
plt.title('Bar Chart', fontsize=14)
ax = sns.countplot(x='Type', data=df, palette=colors, order=order)

# Add count labels on bars
for bar in ax.patches:
    height = bar.get_height()
    ax.text(bar.get_x() + bar.get_width() / 2, height + 1, int(height),
            ha='center', fontsize=10, fontweight='bold')

plt.xlabel('Exercise Type')
plt.ylabel('Total')
plt.xticks(rotation=30, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()

print('*' * 30)
print('Exercise Type Total')
print('*' * 30)
print(df['Type'].value_counts(dropna=False))

```

Result

