Vietnamese - German University

BILLIONAIRES STATISTICS 2023

BUSINESS IT 2

INSTRUCTOR
DR. DO DUC TAN

TABLE OF CONTENT

A INTRODUCTION

B DATASET

C GLOSSARY

D PLOTS & DESCRIPTION

E CONCLUSION

F EXERCISES

A Introduction

Billionaires are more than just the wealthiest individuals on the planet—they are the architects of industries, the pioneers of innovation, and the powerhouses shaping the global economy. Their fortunes, businesses, and industries serve as a mirror reflecting economic trends, market dynamics, and the forces driving wealth creation.

As the project for the Business IT 2 course, this report leverages Python programming based on the R programming code to analyze a comprehensive dataset of the world's billionaires. We will explore wealth distribution, dominant industries, and the key demographic attributes of the ultra-rich. By applying data science techniques, this report will uncover fascinating insights into how billionaires accumulate and sustain their wealth.

Through data visualization and statistical modelling, we will decode the secrets of billionaires, examining which business sectors fuel their fortunes, the geographic trends in wealth accumulation, and the factors that influence billionaire success. By undertaking this exploration, our findings will not only appeal to financial analysts and economists but also provide valuable insights for business strategists and data scientists looking to understand global wealth patterns.

B Dataset

AUTHOR

Nidula Elgiriyewithana

LINK

Original dataset : Billionaires Statistics Dataset (2023)

Used dataset: Billionaires_Statistics_Dataset

WHAT IS IN IT ?

34 columns and 2640 rows

This dataset provides a comprehensive overview of the world's billionaires, offering insights into their wealth, industries, and demographics. It includes key financial and personal details such as net worth, age, country, industries, business sources, and citizenship status. Additionally, it incorporates economic indicators like GDP, life expectancy, tax revenue, and population data for each billionaire's country.

By analyzing this dataset, we can explore wealth distribution, the most dominant industries, the geographic concentration of billionaires, and the impact of economic factors on wealth accumulation. This dataset serves as a valuable resource for researchers, economists, and business strategists aiming to understand the global billionaire landscape.

B Dataset

WHY DID WE CHOOSE THIS DATASET?

Billionaires are not just individuals with immense wealth; they represent economic power, industry trends, and investment patterns worldwide. Understanding the distribution, industries, and sources of billionaire wealth allows us to uncover key insights into economic success and financial concentration.

For our Business IT 2 course - MATLAB & Python, we selected this dataset because it provides a comprehensive breakdown of billionaire demographics, including net worth, industries, self-made status, and geographic distribution. The dataset offers diverse elements that help us analyse billionaire wealth across different sectors and regions.

By using Python programming, we can visualize and interpret these financial patterns, making this dataset an excellent resource for exploring wealth distribution, economic influences, and business trends. Python helps us simplify the data processing and present our findings clearly and compellingly.

B Dataset

WHAT DO WE WANT TO EXPLORE FROM IT ?

Through statistical and graphical analysis, we aim to answer the following key questions:

- Which countries have the highest number of billionaires?
- What are the dominant industries producing the most billionaires?
- How does age correlate with billionaire net worth?
- What percentage of billionaires are self-made versus those who inherited wealth?
- Are there specific regions where billionaires accumulate more wealth?

B Dataset

VARIABLE EXPLAINATION

- **rank**: The ranking of the billionaire in terms of wealth.
- **finalWorth**: The final net worth of the billionaire in U.S. dollars.
- **category**: The category or industry in which the billionaire's business operates.
- **personName**: The full name of the billionaire.
- **age**: The age of the billionaire.
- **country**: The country in which the billionaire resides.
- **city**: The city in which the billionaire resides.
- **source**: The source of the billionaire's wealth.
- **industries**: The industries associated with the billionaire's business interests.
- **countryOfCitizenship**: The country of citizenship of the billionaire.
- **organization**: The name of the organization or company associated with the billionaire.
- **selfMade**: Indicates whether the billionaire is self-made (True/False).

B Dataset

VARIABLE EXPLAINATION

- **status:** "D" represents self-made billionaires (Founders/Entrepreneurs) and "U" indicates inherited or unearned wealth.
- **gender:** The gender of the billionaire.
- **birthDate:** The birthdate of the billionaire.
- **lastName:** The last name of the billionaire.
- **firstName:** The first name of the billionaire.
- **title:** The title or honorific of the billionaire.
- **date:** The date of data collection.
- **state:** The state in which the billionaire resides.
- **residenceStateRegion:** The region or state of residence of the billionaire.
- **birthYear:** The birth year of the billionaire.
- **birthMonth:** The birth month of the billionaire.
- **birthDay:** The birthday of the billionaire.

B Dataset

VARIABLE EXPLAINATION

- **cpi_country:** Consumer Price Index (CPI) for the billionaire's country.
- **cpi_change_country:** CPI change for the billionaire's country.
- **gdp_country:** Gross Domestic Product (GDP) for the billionaire's country.
- **gross_tertiary_education_enrollment:** Enrollment in tertiary education in the billionaire's country.
- **gross_primary_education_enrollment_country:** Enrollment in primary education in the billionaire's country.
- **life_expectancy_country:** Life expectancy in the billionaire's country.
- **tax_revenue_country_country:** Tax revenue in the billionaire's country.
- **total_tax_rate_country:** Total tax rate in the billionaire's country.
- **population_country:** Population of the billionaire's country.
- **latitude_country:** Latitude coordinate of the billionaire's country.
- **longitude_country:** Longitude coordinate of the billionaire's country.

B Dataset

DATA PROCESSING ADJUSTMENTS

While working with the dataset, we encountered a minor issue with the “gdp_country” column, which originally contained the "$" symbol. Since we want to obtain faster and easier coding responses in Python programming, keeping the symbol would lead to a delay when performing calculations or visualizing data.

To resolve this, we followed the guidance of our lecturer, Dr. Đỗ Đức Tân, and removed the dollar sign ($) from the values in the “gdp_country” column. This transformation ensures that the column is treated as pure numerical data, making it compatible with Python programming.

After making this modification, we carefully validated the integrity of the data to confirm that no unintended changes were introduced. This adjustment allows for smoother processing and more accurate economic analysis in our report.

B Dataset

HOW TO DOWNLOAD THE PACKAGE AND LIBRARY

To begin working with data manipulation and visualization in Python, we first need to install the necessary libraries. These libraries provide powerful tools, similar to R's “**tidyverse,**” for effective data analysis and graphical representation.

To further enhance our data analysis and visualization experience, we also utilize several additional libraries for specialized functionality. These include:

- **Geospatial Data:** geopandas (gpd)
- **Data Structures:** numpy (np), pathlib.Path
- **Plot Customization:** matplotlib.patches (Patch), matplotlib.ticker.FuncFormatter
- **Nonparametric Smoothing:** statsmodels.nonparametric.smoothers_lowess.lowess
- **Graphing Objects:** plotly.graph_objects (go), plotly.subplots.make_subplots
- **Country Data:** pycountry, country_converter (coco)
- **Other Utilities:** squarify, math.pi

B Dataset

HOW TO DOWNLOAD THE PACKAGE AND LIBRARY

To install all of the necessary libraries, use the following command:

- ***!pip install geopandas pandas matplotlib seaborn statsmodels plotly pycountry squarify country_converter***

This command will ensure that you have all the essential tools needed for both data manipulation and visualization in our projects.

B Dataset

HOW TO DOWNLOAD THE ADDITIONAL DATA

To visualize a world map in Python using Geopandas and Matplotlib, we need geographic boundary data for all countries. This data is not built into Geopandas by default, so we must manually download it from a trusted open-source source.

In this project, we used the **"Admin 0 - Countries"** dataset from **Natural Earth Data**. This dataset provides global country boundaries in shapefile format, which is compatible with Geopandas.
To download the file: **Click here**

This will download a zipped file named “**ne_110m_admin_0_countries.zip**”.
Extract this file into a folder.
For convenience and cleaner code, rename the extracted folder to shapefiles.

⚠️ Important: Make sure that the shapefiles folder is located in the same directory as your Python script or notebook. Otherwise, the code will not be able to find and load the shapefile properly.

C GLOSSARY
Key Economic Variables

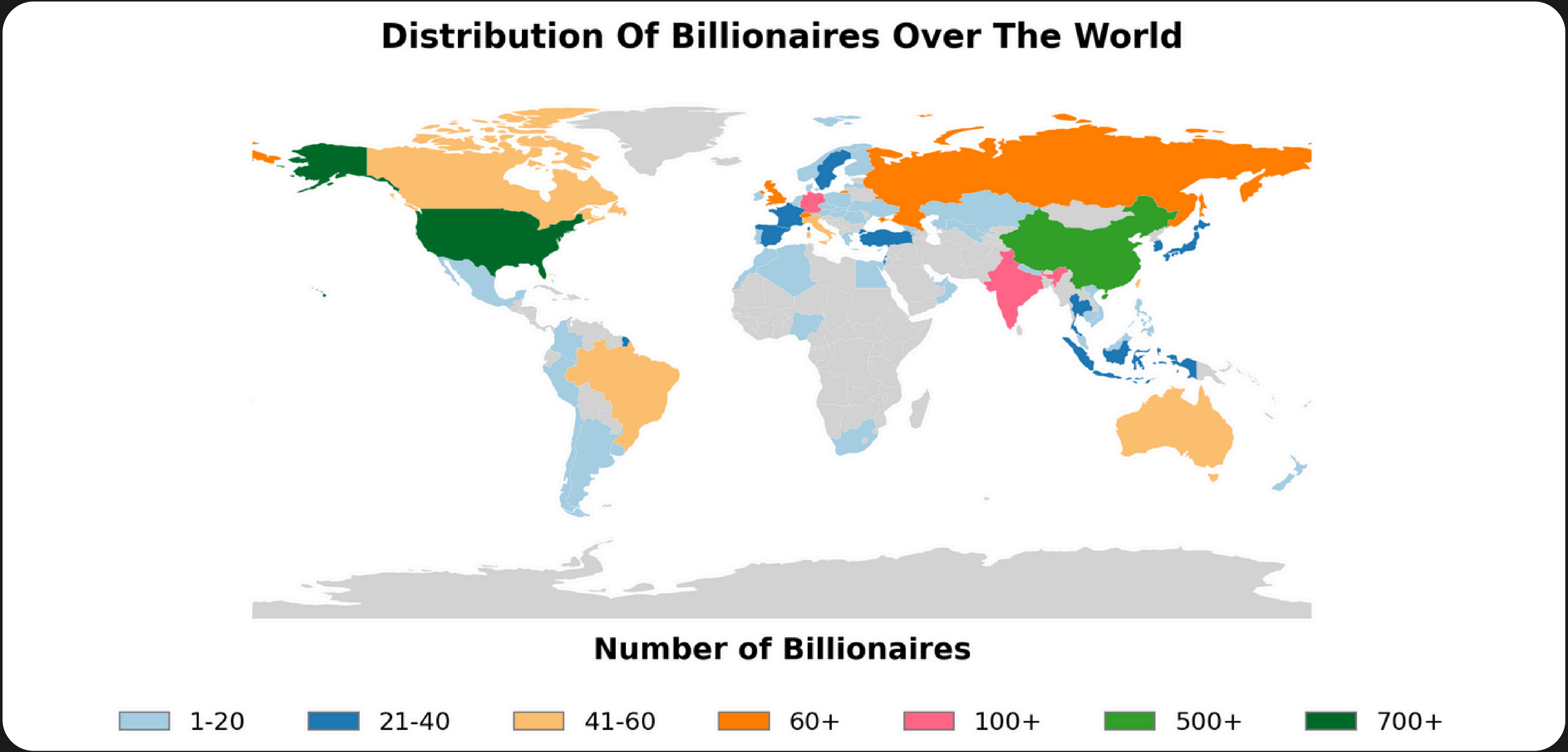
- **CPI (cpi_country):** Measures the average change in prices of goods and services over time in the billionaire's country. Higher CPI indicates inflation.
- **CPI Change (cpi_change_country):** The percentage change in CPI, showing inflation or deflation trends.
- **GDP (gdp_country):** The total value of goods and services produced in a billionaire’s country, reflecting economic strength.
- **Tertiary Education Enrollment (gross_tertiary_education_enrollment):** Percentage of eligible individuals enrolled in universities or vocational institutions.
- **Primary Education Enrollment (gross_primary_education_enrollment_country):** Percentage of children enrolled in elementary schools.
- **Tax Revenue (tax_revenue_country_country):** The percentage of a country’s GDP collected through taxes.
- **Total Tax Rate (total_tax_rate_country):** The overall tax burden, including income and corporate taxes, in a billionaire’s country.
- **Citizenship:** The official nationality of a billionaire, which affects taxation and business regulations.

D PLOTS & DESCRIPTIONS

1. WORLD MAP
2. SCATTER PLOT
3. STACKED BAR CHART
4. PERCENTAGE STACKED BAR CHART
5. FACETED LINE CHART
6. DONUT CHART
7. TREE MAP
8. PIE-DONUT CHART
13. RADAR CHART
9. RIDGELINE PLOT
10. STACKED BAR CHART WITH LINE
11. BOX PLOT
12. HEAT MAP

PLOTS & DESCRIPTIONS

Plot 1: World Map



PLOTS & DESCRIPTIONS

Plot 1: World Map Analysis

The map illustrates the global distribution of billionaires, emphasizing important economic inequalities. Billionaires tend to be concentrated in developed countries with strong financial systems, while nations with weaker economies feature considerably fewer due to limited access to capital, slower levels of industrial growth, and economic instability.

The classification system reflects various phases of economic development. Countries in the 1-20 range usually have developing economies, where financial limitations obstruct large-scale wealth creation. The 21-40 tier comprises nations with moderate advancements, witnessing gradual but restricted billionaire expansion. The 41-60 and 60+ groups symbolize transitioning economies that have achieved important financial progress yet still lack the scale of economic giants. The 100+ category, illustrated by India, indicates swift economic growth, propelled by industrialization, technological progress, and policy reforms. The 500+ and 700+ tiers, spearheaded by the U.S. and China, signify financial powerhouses with highly sophisticated capital markets, plenty investment options, and thriving entrepreneurial environments that promote billionaire development.

PLOTS & DESCRIPTIONS

Plot 1: World Map Analysis

This categorization adeptly sorts billionaire distribution while emphasizing the strong relationship between economic maturity and wealth accumulation. By clearly defining economic levels, it emphasizes how financial access, industrial advancement, and government policies shape the worldwide environment of billionaires.

PLOTS & DESCRIPTIONS

Plot 1: Code

```
# Load libraries
import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.patches import Patch
import numpy as np
from pathlib import Path
csv_filename = "Billionaires_Statistics_Dataset.csv"
try:
 try:
  script_dir = Path(__file__).parent
 except NameError:
  script_dir = Path.cwd()
 csv_path = script_dir / csv_filename
 data_billionaires_raw = pd.read_csv(csv_path)
 data_billionaires = data_billionaires_raw.copy()
 data_billionaires.columns = data_billionaires.columns.str.replace(' ', '_',
regex=False).str.lower()
 if 'country' in data_billionaires_raw.columns and 'country' not in
data_billionaires.columns:
  if 'country_' in data_billionaires.columns: data_billionaires.rename(columns=
{'country_':'country'}, inplace=True)
except FileNotFoundError:
 print(f"ERROR: CSV file '{csv_filename}' not found in the script's directory.")
 print(">>> Please ensure the CSV file is in the same folder as the Python script.")
 exit()
except Exception as e:
 print(f"ERROR reading or processing CSV: {e}")
 exit()

# Data Processing
try:
 country_col_name = 'country'
 if country_col_name not in data_billionaires.columns: raise KeyError(f"Column
'{country_col_name}' not found after standardization.")
 data_billionaires[country_col_name] =
data_billionaires[country_col_name].astype(str).str.replace(r'\s+', ' ', regex=True).str.strip()
 country_name_mapping = {
  "United States": "United States of America", "South Korea": "Korea, Republic of",
  "Czech Republic": "Czechia", "Russia": "Russian Federation",
  "Hong Kong SAR": "Hong Kong",
 }
 data_billionaires[country_col_name] =
data_billionaires[country_col_name].replace(country_name_mapping)
 data_summary = data_billionaires.groupby(country_col_name, as_index=False).size()
 data_summary.rename(columns={'size': 'billionaires', country_col_name: 'country'},
inplace=True)
```

PLOTS & DESCRIPTIONS

Plot 1: Code

```
 # Create billionaire count categories
 bins = [-np.inf, 20, 40, 60, 100, 500, 700, np.inf]
 labels = ["1-20", "21-40", "41-60", "60+", "100+", "500+", "700+"]
 data_summary['category'] = pd.cut(data_summary['billionaires'], bins=bins,
labels=labels, right=True).astype(str)
 data_summary['category'].fillna("No Data", inplace=True)
except KeyError as e:
 print(f"ERROR: Incorrect column name during data processing: {e}")
 exit()
except Exception as e:
 print(f"ERROR during data processing: {e}")
 exit()

# Map Data Processing
shapefile_folder = "shapefiles"
shapefile_name = "ne_110m_admin_0_countries.shp"
shapefile_path = script_dir / shapefile_folder / shapefile_name
try:
 world_map = gpd.read_file(shapefile_path)
except Exception as map_error:
 print(f"ERROR reading shapefile at '{shapefile_path}': {map_error}")
 print(">>> Please check the 'shapefiles' subfolder and its contents (downloaded from
Natural Earth).")
 exit()
```

```
# Select necessary columns and standardize country name column from map data
country_name_col_map = 'ADMIN'
if country_name_col_map not in world_map.columns:
  if 'NAME' in world_map.columns: country_name_col_map = 'NAME'
  elif 'SOVEREIGNT' in world_map.columns: country_name_col_map = 'SOVEREIGNT'
  else:
    print(f"ERROR: Could not find a suitable country name column (e.g., 'ADMIN', 'NAME')
in {shapefile_path}")
    print("Available columns:", world_map.columns)
    exit()
world_map = world_map[[country_name_col_map, 'geometry']].copy()
world_map.rename(columns={country_name_col_map: 'map_country_name'},
inplace=True)

# Standardize country names in map data for merging
world_map['map_country_name'] =
world_map['map_country_name'].astype(str).str.replace(r'\s+', ' ', regex=True).str.strip()
world_map['map_country_name'] = world_map['map_country_name'].replace({
 # Ensure consistency with data_summary country names
 "United States of America": "United States of America",
 "South Korea": "Korea, Republic of",
 "Czech Rep.": "Czechia",
 "Russia": "Russian Federation"
})
```

PLOTS & DESCRIPTIONS

Plot 1: Code

```
# Merge Map and Billionaire Data
merged_data = world_map.merge(data_summary, how='left',
left_on='map_country_name', right_on='country')
category_order = ["1-20", "21-40", "41-60", "60+", "100+", "500+", "700+", "No
Data"]
if 'category' not in merged_data.columns:
  merged_data['category'] = "No Data"
else:
  merged_data['category'] = merged_data['category'].astype(str)
  merged_data['category'] = merged_data['category'].fillna("No Data")

# Convert to ordered Categorical for correct plotting and legend order
merged_data['category'] = pd.Categorical(merged_data['category'],
categories=category_order, ordered=True)

# Plotting using Geopandas/Matplotlib
color_mapping = {
 "1-20": "#A6CEE3", "21-40": "#1F78B4", "41-60": "#FDBF6F",
 "60+": "#FF7F00", "100+": "#ff6688", "500+": "#33A02C",
 "700+": "#006D2C", "No Data": "#D3D3D3"
}
```

```
fig, ax = plt.subplots(1, 1, figsize=(20, 18))
merged_data.plot(
 column='category',
 ax=ax,
 edgecolor="#f6f6f6",
 linewidth=0.1,
 cmap=plt.cm.colors.ListedColormap([color_mapping[cat] for cat in
category_order]),
 missing_kwds={"color": color_mapping["No Data"], "label": "No Data"},
 legend=False
)

# Create Legend Elements
unique_categories_in_data = merged_data['category'].unique()
legend_elements = [Patch(facecolor=color_mapping[label], edgecolor='grey',
label=label)
      for label in category_order if label in unique_categories_in_data]
```

PLOTS & DESCRIPTIONS

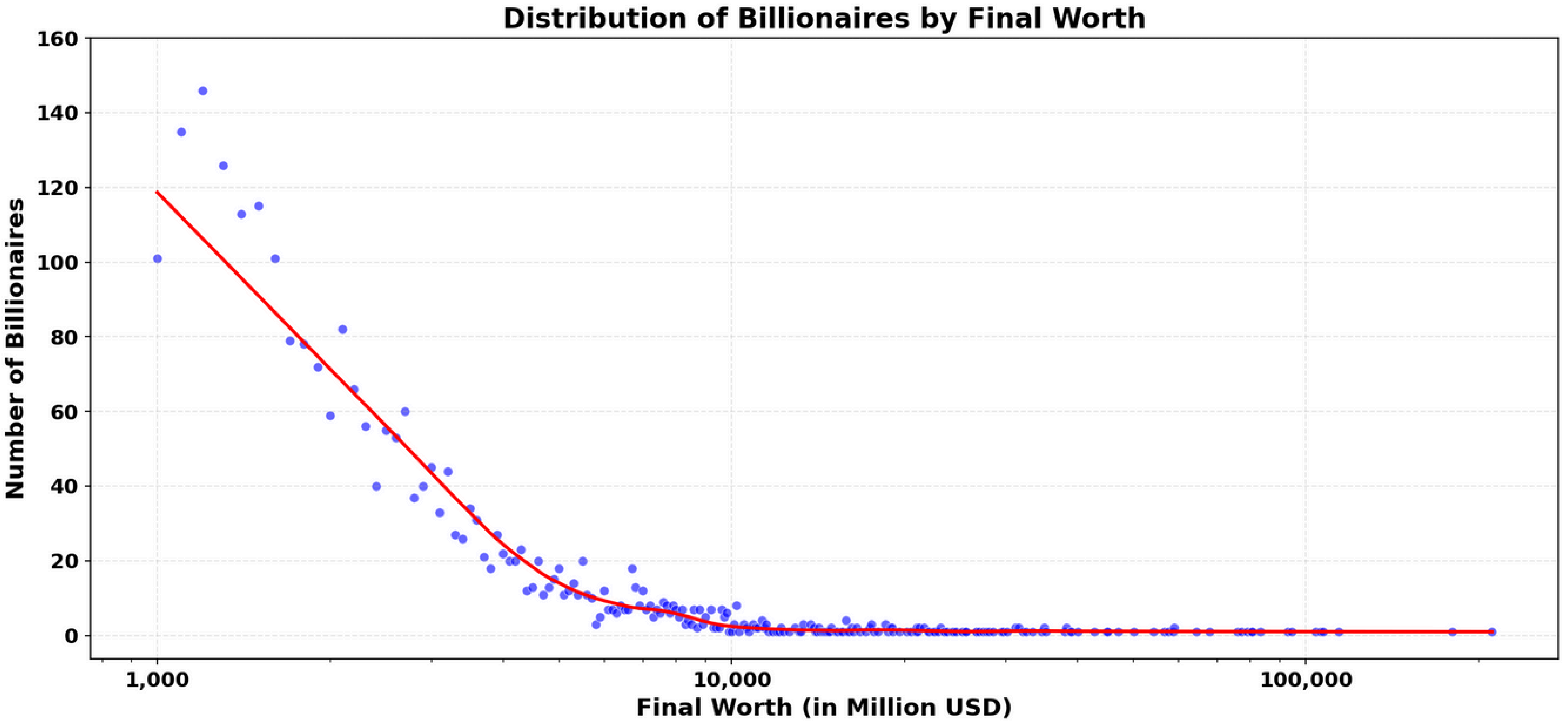
Plot 1: Code

```
# Legend Configuration
ax.legend(handles=legend_elements,
 title="Number of Billionaires\n",
 loc='lower center',
 bbox_to_anchor=(0.5, -0.2),
 ncol=len(legend_elements),
 fontsize=15,
 title_fontproperties={'weight':'bold', 'size':18},
 facecolor=ax.get_facecolor(),
 frameon=False,
 columnspacing=2.5
 )

 # Title Configuration
ax.set_title(
 "Distribution Of Billionaires Over The World",
 fontsize=20,
 fontweight='bold',
 pad=20
)
```

```
# Axes and Background Configuration
ax.set_axis_off()
ax.set_facecolor('white')
fig.set_facecolor('white')

# Adjust Layout
plt.subplots_adjust(
    left=0.01,
    right=0.99,
    top=0.94,
    bottom=0.20
)

# Display Plot
plt.show()
```

PLOTS & DESCRIPTIONS

Plot 2: Scatter Plot



PLOTS & DESCRIPTIONS

Plot 2: Scatter Plot Analysis

This scatter plot with a fitted regression line shows the distribution of billionaires based on their final net worth in million USD. The x-axis represents the wealth of billionaires on a log scale, and the y-axis represents the number of individuals in each wealth bracket.

The data reveal a very strong negative relationship between net worth and the number of billionaires. The highest concentration is in the lowest wealth category (1,000–10,000 million USD), with over 100 billionaires in the lowest category. The number of billionaires decreases exponentially as wealth increases, forming a long tail where very few individuals possess over 100,000 million USD.

The red regression line precisely portrays this exponential drop, with a bit of uncertainty band (gray shading) at the extremely high-net-worth end. The overall trend captures the rarity of extreme wealth concentration, confirming the perception that only a very few amass enormous financial assets.
This graph provides a dramatic picture of wealth inequality among billionaires, pointing out that while many people become billionaires, few reach the highest ranks of wealth.

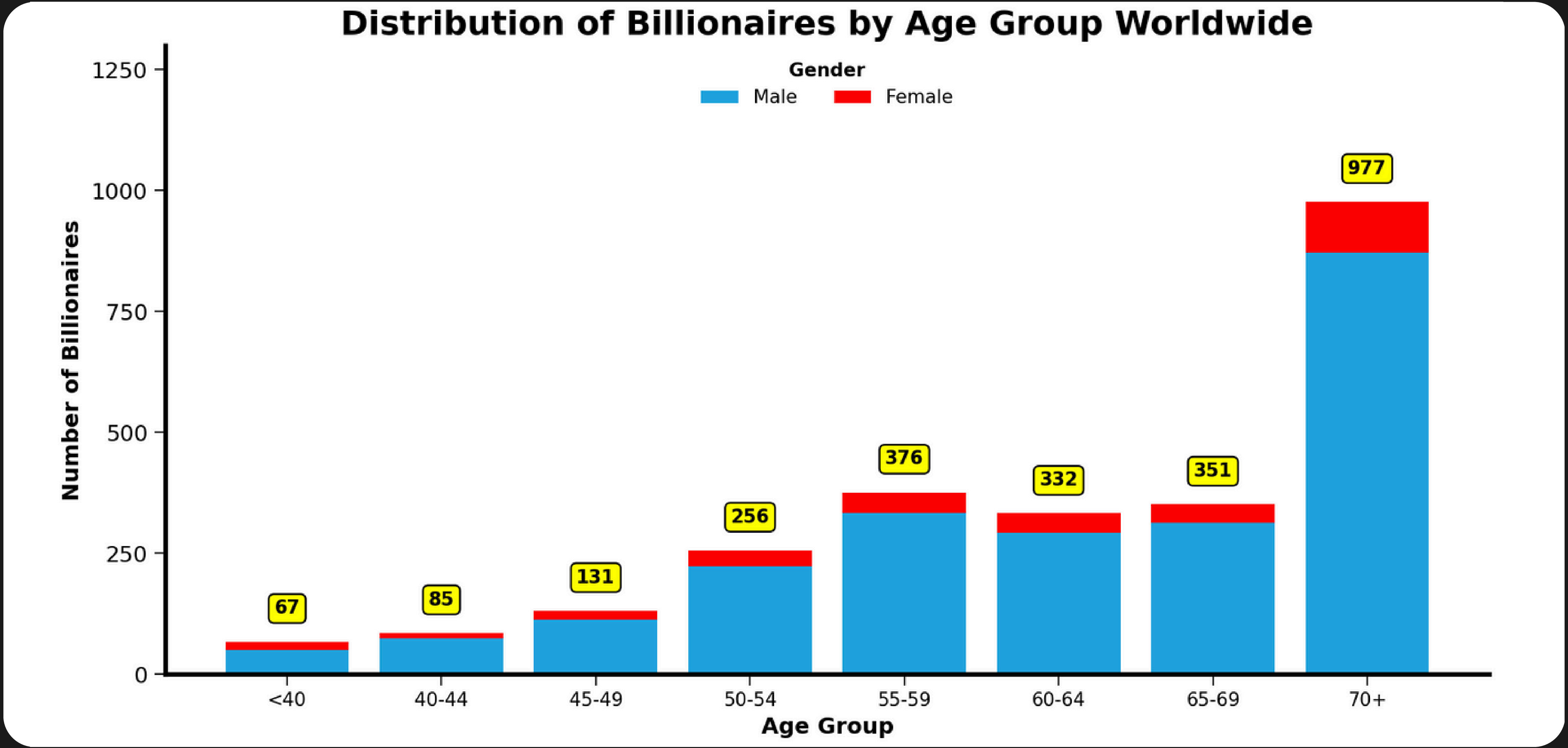
PLOTS & DESCRIPTIONS

Plot 2: Code

```
# Load library
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from matplotlib.ticker import FuncFormatter
from statsmodels.nonparametric.smoothers_lowess import lowess

# Load and prepare the dataset
df = pd.read_csv("Billionaires_Statistics_Dataset.csv")
billionaire_counts = df['finalWorth'].value_counts().reset_index()
billionaire_counts.columns = ['finalWorth', 'n']
billionaire_counts = billionaire_counts.sort_values('finalWorth')

# Apply LOWESS smoothing
smoothed = lowess(
  endog=billionaire_counts['n'],
  exog=np.log10(billionaire_counts['finalWorth']),
  frac=0.15
)

# Plot the distribution
plt.figure(figsize=(14, 6))
sns.scatterplot(data=billionaire_counts, x='finalWorth', y='n', color='blue',
s=30, alpha=0.6)
plt.plot(10**smoothed[:, 0], smoothed[:, 1], color='red', linewidth=2)
plt.xscale('log')

# Format x-axis ticks to show full numbers
def full_number_formatter(x, pos):
  return f'{int(x):,}'
plt.gca().xaxis.set_major_formatter(FuncFormatter(full_number_formatter))

# Chart styling
plt.title("Distribution of Billionaires by Final Worth", fontsize=16,
fontweight='bold')
plt.xlabel("Final Worth (in Million USD)", fontsize=14, fontweight='bold')
plt.ylabel("Number of Billionaires", fontsize=14, fontweight='bold')
plt.xticks(fontsize=12, fontweight='bold')
plt.yticks([0, 20, 40, 60, 80, 100, 120, 140, 160], fontsize=12,
fontweight='bold')
plt.grid(True, which='major', axis='both', linestyle='--', alpha=0.3)
plt.tight_layout()
plt.show()
```

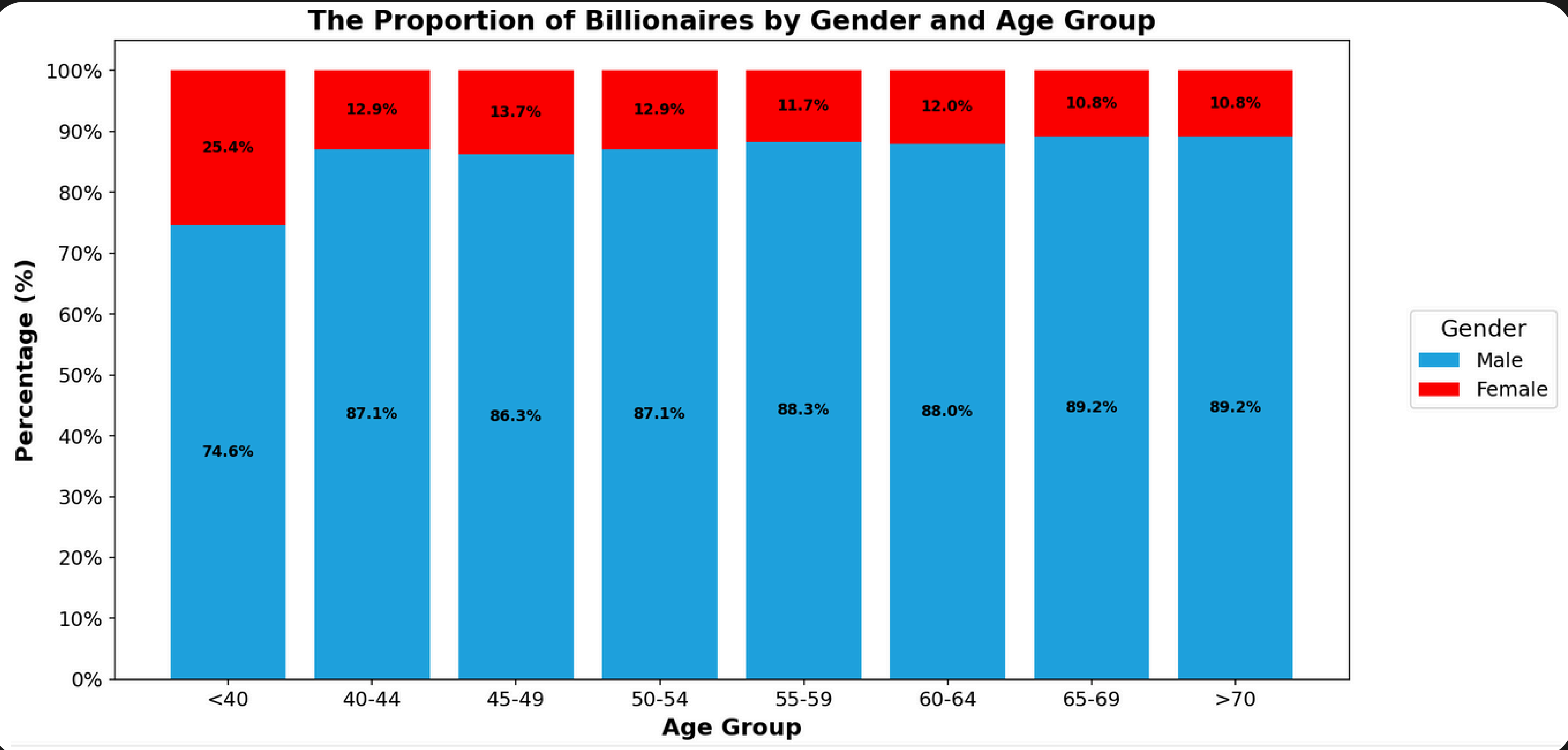
PLOTS & DESCRIPTIONS

Plot 3: Stacked Bar Chart



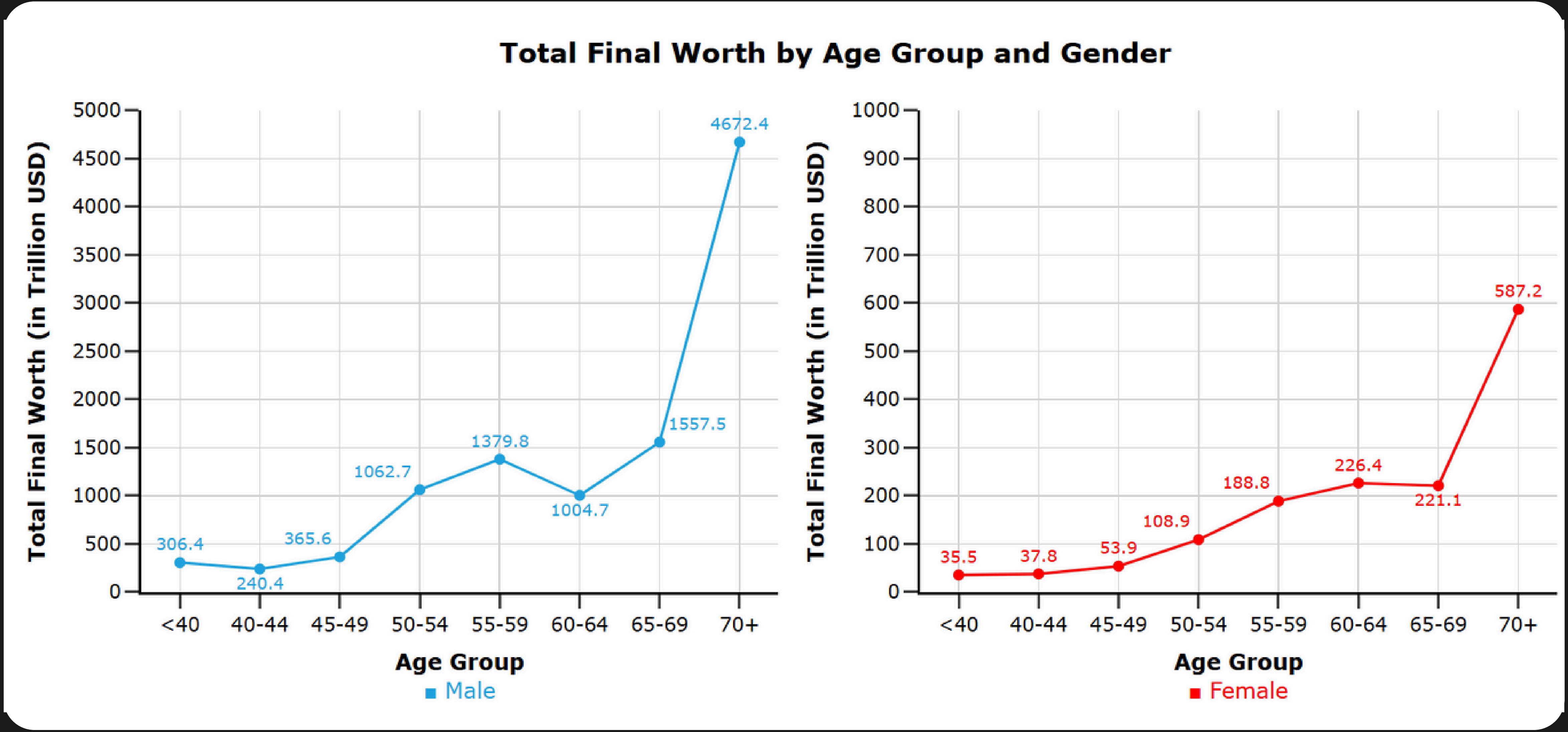
PLOTS & DESCRIPTIONS

Plot 4: Percentage Stacked Bar Chart



PLOTS & DESCRIPTIONS

Plot 5: Faceted Line Chart



PLOTS & DESCRIPTIONS

Plots 3+4+5: Combined Analysis

The three charts analyze billionaire demographics through age distribution, gender proportions, and total wealth accumulation. The stacked bar chart shows billionaire numbers by age and gender. The proportional bar chart highlights the widening gender gap across age groups. The faceted line chart reveals stark disparities in wealth accumulation, with male billionaires amassing exponentially greater fortunes than females. Collectively, these trends underscore the profound impact of age and gender on billionaire status, illustrating long-term accumulation patterns and structural wealth disparities.

The age groups are structured around key financial milestones. The under-40 group includes young billionaires, often tech entrepreneurs or recent heirs. The 40-54 range represents peak earning and business expansion years. The 55-69 bracket marks wealth consolidation, leveraging decades of accumulation. The 70+ category reflects long-established billionaires benefiting from sustained investments and generational wealth.This categorization provides a clearer understanding of how wealth evolves over time, offering insights into long-term financial trajectories.

PLOTS & DESCRIPTIONS

Plots 3+4+5: Combined Analysis

Billionaire numbers increase significantly with age. The under-40 group comprises only 67 individuals, whereas the 55-59 cohort reaches 376 billionaires, marking a fivefold growth. A slight decline occurs at 60-64 (332 billionaires) before rebounding at 65-69 (351 billionaires). The highest concentration is observed in the 70+ category, with 977 billionaires—almost three times the 55-59 group and fifteen times the youngest cohort. This pattern suggests that billionaire status is predominantly achieved later in life, largely due to career longevity, investment growth, or inherited wealth, reinforcing the notion that wealth accumulation is a long-term process.

Male billionaires dominate every age bracket. Among those under 40, men constitute 74.6%, while women represent 25.4%—the highest female proportion observed. However, as age increases, the gender gap widens. By 40-44, the male share rises to 87.1% and remains above 86% in all older groups. The most extreme disparity is in the 70+ category, where men account for 89.2% of billionaires, leaving only 10.8% female. This trend reflects structural barriers and historical wealth concentration among men, highlighting the persistent challenges women face in achieving billionaire status, whether due to industry biases, leadership disparities, or inheritance patterns.

PLOTS & DESCRIPTIONS

Plots 3+4+5: Combined Analysis

Male billionaires consistently accumulate significantly greater wealth than females. In the under-40 cohort, men control 306.4 trillion USD, nearly nine times the 35.5 trillion USD held by women. By 65-69, male net worth reaches 1557.5 trillion USD, while female wealth stands at 221.1 trillion USD—a sevenfold difference. The greatest disparity appears in the 70+ category, where male wealth surges threefold to 4672.4 trillion USD, whereas female wealth rises 2.6 times to 578.2 trillion USD. This pattern underscores the accelerating pace of male wealth accumulation, particularly in later life, further widening the financial divide.

The data highlights three dominant patterns. Billionaire numbers increase with age, peaking at 70+, reinforcing the long-term nature of wealth accumulation. Gender disparity intensifies, with the male-female gap widening as age progresses. Most notably, male wealth accelerates in later life, tripling from 65-69 to 70+, while female wealth grows 2.6 times. These findings emphasize how age and gender shape billionaire demographics, reflecting deep-rooted economic structures. The trends suggest that while time is a crucial factor in amassing wealth, systemic gender imbalances persist, influencing both billionaire representation and the scale of financial success.

PLOTS & DESCRIPTIONS

Plot 3: Code

```
# Load library
import pandas as pd
import matplotlib.pyplot as plt

# Load dataset from CSV
file_path = "Billionaires_Statistics_Dataset.csv"
df = pd.read_csv(file_path)

# Clean and standardize gender and age columns
df['gender'] = df['gender'].str.strip()
df['gender'] = df['gender'].replace({'M': 'Male', 'F': 'Female'})
df = df[(df['age'].notna()) & (df['age'] > 0) & (df['gender'].notna())]

# Categorize age into groups
def age_group(age):
    if age < 40:
        return "<40"
    elif age < 45:
        return "40-44"
    elif age < 50:
        return "45-49"
    elif age < 55:
        return "50-54"
    elif age < 60:
        return "55-59"
    elif age < 65:
        return "60-64"
    elif age < 70:
        return "65-69"
    else:
        return "70+"
df['age_group'] = df['age'].apply(age_group)

# Count by gender and age_group
age_distribution = df.groupby(['age_group',
'gender']).size().reset_index(name='count')

# Total per age group
total_counts = age_distribution.groupby('age_group')
['count'].sum().reset_index(name='total_count')
age_distribution = pd.merge(age_distribution, total_counts, on='age_group')
```

PLOTS & DESCRIPTIONS

Plot 3: Code

```
# Sort age groups in desired order
age_order = ["<40", "40-44", "45-49", "50-54", "55-59", "60-64", "65-69", "70+"]
age_distribution['age_group'] = pd.Categorical(age_distribution['age_group'],
categories=age_order, ordered=True)
age_distribution.sort_values('age_group', inplace=True)

# Bar plot
plt.figure(figsize=(14, 4))
colors = {'Male': '#22a4de', 'Female': '#fe0000'}
bottom_vals = {}
for gender in ['Male', 'Female']:
 gender_data = age_distribution[age_distribution['gender'] == gender]
 plt.bar(gender_data['age_group'], gender_data['count'],
 bottom=[bottom_vals.get(x, 0) for x in gender_data['age_group']],
 color=colors[gender], label=gender)
 for i, row in gender_data.iterrows():
 bottom_vals[row['age_group']] = bottom_vals.get(row['age_group'], 0) +
row['count']

# Add total count labels
for i, row in total_counts.iterrows():
 plt.text(row['age_group'], row['total_count'] + 50, str(row['total_count']),
 ha='center', va='bottom', fontsize=10, fontweight='bold',
 bbox=dict(boxstyle="round,pad=0.3", facecolor='yellow'))
```

```
# Customize appearance
plt.title("Distribution of Billionaires by Age Group Worldwide", fontsize=18,
fontweight='bold')
plt.xlabel("Age Group", fontsize=12, fontweight='bold')
plt.ylabel("Number of Billionaires", fontsize=12, fontweight='bold')
plt.xticks(fontsize=10)
plt.yticks(range(0, 1300, 250), fontsize=12)
plt.ylim(0, 1300)
plt.legend(title="Gender", fontsize=10, title_fontproperties={'weight': 'bold'},
loc='upper center', ncol=2, frameon=False)

# Border lines
plt.gca().spines['top'].set_visible(False)
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['left'].set_linewidth(2.5)
plt.gca().spines['left'].set_color('black')
plt.gca().spines['bottom'].set_color('black')
plt.gca().spines['bottom'].set_linewidth(2.5)
plt.grid(False)

# Adding tick marks
plt.tick_params(axis='both', which='both', length=6)
plt.tick_params(axis='x', which='major', length=6)
plt.tick_params(axis='y', which='minor', length=6)
plt.show()
```

PLOTS & DESCRIPTIONS

Plot 4: Code

```
# Load library
import pandas as pd
import matplotlib.pyplot as plt

# Load and preprocess data
data = pd.read_csv('Billionaires_Statistics_Dataset.csv')
data = data.dropna(subset=['age', 'gender'])

# Create age groups
bins = [0, 40, 45, 50, 55, 60, 65, 70, float('inf')]
labels = ["<40", "40-44", "45-49", "50-54", "55-59", "60-64", "65-69",
">70"]
data['age_group'] = pd.cut(data['age'], bins=bins, labels=labels,
right=False)

# Standardize gender labels
data['gender'] = data['gender'].map({'M': 'Male', 'F': 'Female'})

# Calculate percentage distribution
summary = data.groupby(['age_group', 'gender'],
observed=True).size().unstack().fillna(0)
summary_percent = summary.div(summary.sum(axis=1), axis=0)

# Plot gender distribution by age group
fig, ax = plt.subplots(figsize=(10, 6), constrained_layout=True)
colors = {'Male': '#22a4de', 'Female': '#fe0000'}
bottom = pd.Series([0] * len(summary_percent),
index=summary_percent.index)
for gender in ['Male', 'Female']:
  ax.bar(summary_percent.index, summary_percent[gender],
bottom=bottom, label=gender, color=colors[gender])
  for i, value in enumerate(summary_percent[gender]):
    if value > 0.01:
      ax.text(i, bottom.iloc[i] + value / 2, f'{value:.1%}', ha='center',
va='center', fontsize=9, fontweight='bold')
  bottom += summary_percent[gender]

# Customize chart appearance
ax.set_title("The Proportion of Billionaires by Gender and Age Group",
fontsize=16, fontweight='bold')
ax.set_xlabel("Age Group", fontsize=14, fontweight='bold')
ax.set_ylabel("Percentage (%)", fontsize=14, fontweight='bold')
ax.set_yticks([i / 10 for i in range(0, 11)])
ax.set_yticklabels([f"{int(i * 10)}%" for i in range(0, 11)])
ax.tick_params(axis='x', labelsize=12)
ax.tick_params(axis='y', labelsize=12)
```

PLOTS & DESCRIPTIONS

Plot 4: Code

```
# Add legend
ax.legend(
 title="Gender",
 fontsize=12,
title_fontsize=14,
  loc='center left',
  bbox_to_anchor=(1.05, 0.5),
  borderaxespad=0
)
plt.show()
```

PLOTS & DESCRIPTIONS

Plot 5: Code

```
# Load libraries
import pandas as pd
import numpy as np
from pathlib import Path
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import pycountry

# Read CSV file
csv_filename = "Billionaires_Statistics_Dataset.csv"
data_billionaires = None
try:
  try:
    script_dir = Path(__file__).parent
  except NameError:
    script_dir = Path.cwd()
  csv_path = script_dir / csv_filename
  data_billionaires_raw = pd.read_csv(csv_path)
  data_billionaires = data_billionaires_raw.copy()
```

```
  # Standardize column names
  original_columns = data_billionaires.columns.tolist()
  data_billionaires.columns = data_billionaires.columns.str.replace(' ', '_',
regex=False).str.lower()
  if 'country' in original_columns and 'country_' in data_billionaires.columns and 'country'
not in data_billionaires.columns:
     data_billionaires.rename(columns={'country_':'country'}, inplace=True)
  required_cols = ['gender', 'age', 'finalworth']
  for col in required_cols:
    if col not in data_billionaires.columns:
      potential_original_names = [c for c in original_columns if col in c.lower().replace(' ','_')]
      if len(potential_original_names) == 1:
        original_name_std = potential_original_names[0].lower().replace(' ','_')
        if original_name_std in data_billionaires.columns:
          data_billionaires.rename(columns={original_name_std: col}, inplace=True)
      else:
        raise KeyError(f"Required column '{col}' not found after standardization. Available:
{data_billionaires.columns.tolist()}")
except FileNotFoundError:
  print(f"ERROR: CSV file '{csv_filename}' not found in the script's directory.")
  print(">>> Please ensure the CSV file is in the same folder as the Python script.")
  exit()
except Exception as e:
  print(f"ERROR reading or processing CSV: {e}")
  exit()
```

PLOTS & DESCRIPTIONS

Plot 5: Code

```
# Data Processing
try:
  # Standardize gender values
  data_billionaires['gender'] = data_billionaires['gender'].astype(str).str.strip()
  gender_mapping = {"M": "Male", "F": "Female"}
  data_billionaires['gender'] = data_billionaires['gender'].map(gender_mapping)
  # Filter valid data
  data_billionaires['age'] = pd.to_numeric(data_billionaires['age'], errors='coerce')
  data_billionaires['finalworth'] = pd.to_numeric(data_billionaires['finalworth'],
errors='coerce')
  data_billionaires.dropna(subset=['age', 'gender', 'finalworth'], inplace=True)
  data_billionaires['age'] = data_billionaires['age'].astype(int)
  # Categorize age groups
  bins = [-np.inf, 39, 44, 49, 54, 59, 64, 69, np.inf]
  labels = ["<40", "40-44", "45-49", "50-54", "55-59", "60-64", "65-69", "70+"]
  data_billionaires['age_group'] = pd.cut(data_billionaires['age'], bins=bins,
labels=labels, right=True)
  data_billionaires['age_group'] = pd.Categorical(data_billionaires['age_group'],
categories=labels, ordered=True)
  # Calculate total final worth by age group and gender
  finalworth_data = data_billionaires.groupby(['age_group', 'gender'],
observed=False)['finalworth'].sum().reset_index()
  finalworth_data['total_finalworth'] = finalworth_data['finalworth'] / 1000

  # Get ISO ALPHA-3 codes for country mapping
  def get_iso_alpha3(country_name):
    try:
      country = pycountry.countries.search_fuzzy(country_name)
      if country:
        if isinstance(country, list): return country[0].alpha_3
        else: return country.alpha_3
      else:
        if country_name == "United States of America": return "USA"
        return None
    except Exception:
      return None
except KeyError as e:
  print(f"ERROR: Column key error during data processing: {e}")
  exit()
except Exception as e:
  print(f"ERROR during data processing: {e}")
  exit()

# Separate male and female data
male_data = finalworth_data[finalworth_data['gender'] ==
'Male'].sort_values('age_group')
female_data = finalworth_data[finalworth_data['gender'] ==
'Female'].sort_values('age_group')
```

PLOTS & DESCRIPTIONS

Plot 5: Code

```
# Plotting with Plotly Subplots
fig = make_subplots(
    rows=1, cols=2,
    shared_xaxes=True,
    shared_yaxes=False
)

# Define text positions
male_text_positions = {
    "<40": "top center", "40-44": "bottom center", "45-49": "top left",
    "50-54": "top left", "55-59": "top center", "60-64": "bottom center",
    "65-69": "top right", "70+": "top center"
}
female_text_positions = {
    "<40": "top center", "40-44": "top center", "45-49": "top center",
    "50-54": "top left", "55-59": "top left", "60-64": "top center",
    "65-69": "bottom center", "70+": "top center"
}

# Add Male trace
fig.add_trace(go.Scatter(
    x=male_data['age_group'],
    y=male_data['total_finalworth'],
    mode='lines+markers+text',
    name='Male',
    line=dict(color="#22a4de", width=2, shape='linear'),
    marker=dict(color="#22a4de", size=8),
    text=male_data['total_finalworth'].round(1).astype(str),
    textposition=male_data['age_group'].map(male_text_positions).fillna("top center"),
    textfont=dict(color="#22a4de", size=12),
    legendgroup="Male"
    ),
    row=1, col=1
)
# Add Female trace
fig.add_trace(go.Scatter(
    x=female_data['age_group'],
    y=female_data['total_finalworth'],
    mode='lines+markers+text',
    name='Female',
    line=dict(color="#fe0000", width=2, shape='linear'),
    marker=dict(color="#fe0000", size=8),
    text=female_data['total_finalworth'].round(1).astype(str),
    textposition=female_data['age_group'].map(female_text_positions).fillna("top center"),
    textfont=dict(color="#fe0000", size=12),
    legendgroup="Female"
    ),
    row=1, col=2
)
```

PLOTS & DESCRIPTIONS

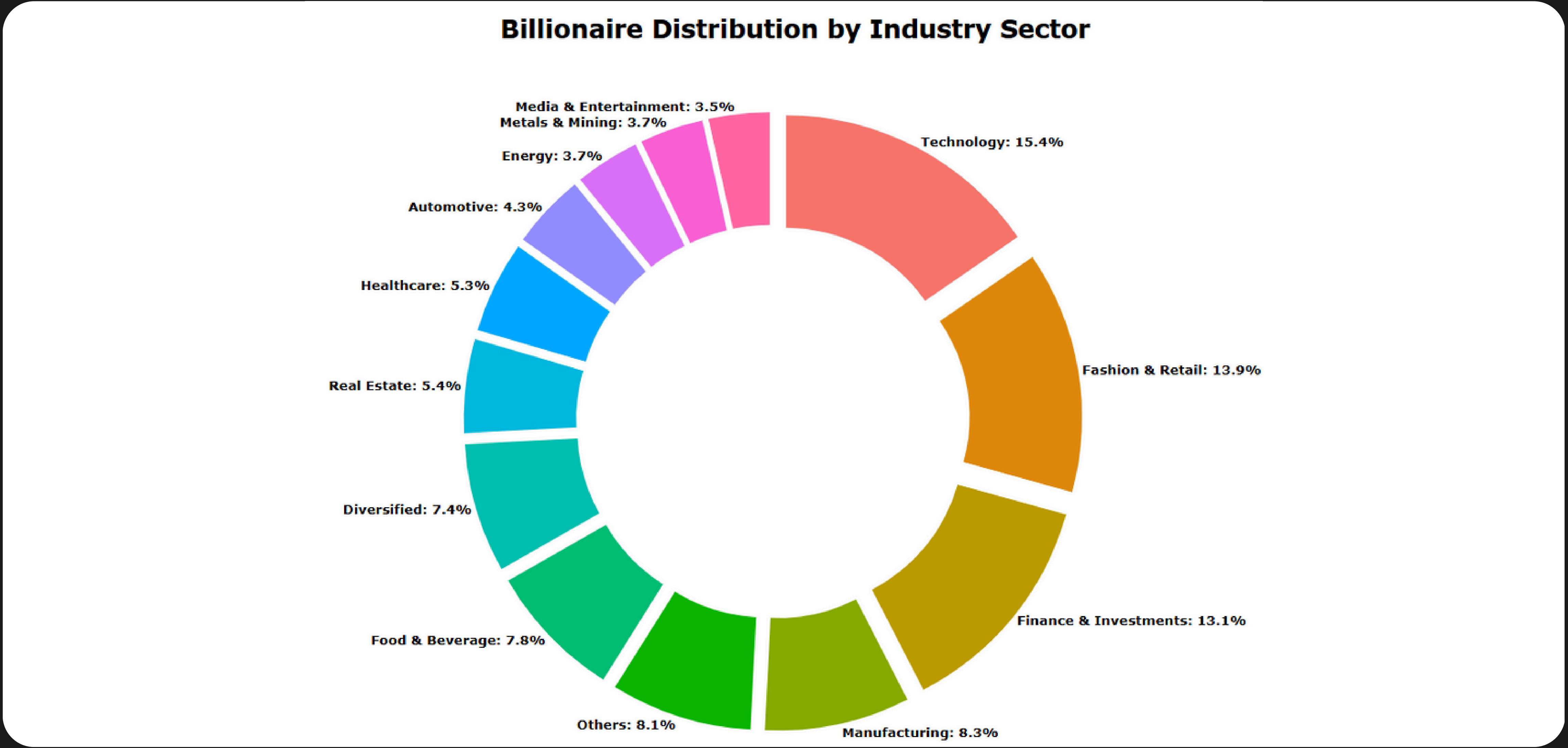
Plot 5: Code

```
# Update Layout and Axes
fig.update_layout(
    title=dict(
        text="<b>Total Final Worth by Age Group and Gender</b>",
        font=dict(size=20, color="black"),
        x=0.5
    ),
    plot_bgcolor='white',
    paper_bgcolor='white',
    showlegend=False,
    margin=dict(l=100, r=80, t=70, b=70),
    height=510,
    width=1210,
    font=dict(size=14, color="black")
)
fig.update_xaxes(
    type='category',
    tickvals=labels,
    ticks="outside", ticklen=10, tickwidth=1.5,
    showline=True, linecolor="black", linewidth=1.5,
    tickfont=dict(size=14, color="black"),
    showgrid=True, gridwidth=1, gridcolor='LightGrey'
)
```

```
# Set specific titles for each X axis
fig.update_xaxes(title_text="<b>Age Group</b><br><span style='color:#22a4de;'>
Male</span>", title_font=dict(size=16, color="black"), row=1, col=1)
fig.update_xaxes(title_text="<b>Age Group</b><br><span style='color:#fe0000;'>
Female</span>", title_font=dict(size=16, color="black"), row=1, col=2)
# Male Y Axis (yaxis)
fig.update_yaxes(
    title_text="<b>Total Final Worth (in Trillion USD)</b>", range=[0, 5000],
tickvals=np.arange(0, 5001, 500),
    ticks="outside", ticklen=10, tickwidth=1.5, showline=True, zeroline=False,
    linecolor="black", linewidth=1.5, title_font=dict(size=16, color="black"),
tickfont=dict(size=14, color="black"),
 showgrid=True, gridwidth=1, gridcolor='LightGrey',  row=1, col=1
)
# Female Y Axis (yaxis2)
fig.update_yaxes(
    title_text="<b>Total Final Worth (in Trillion USD)</b>", range=[0, 1000],
tickvals=np.arange(0, 1001, 100),
    ticks="outside", ticklen=10, tickwidth=1.5, showline=True, zeroline=False,
    linecolor="black", linewidth=1.5, title_font=dict(size=16, color="black"),
tickfont=dict(size=14, color="black"),
    showgrid=True, gridwidth=1, gridcolor='LightGrey', row=1, col=2
)
# Display the plot
fig.show()
```

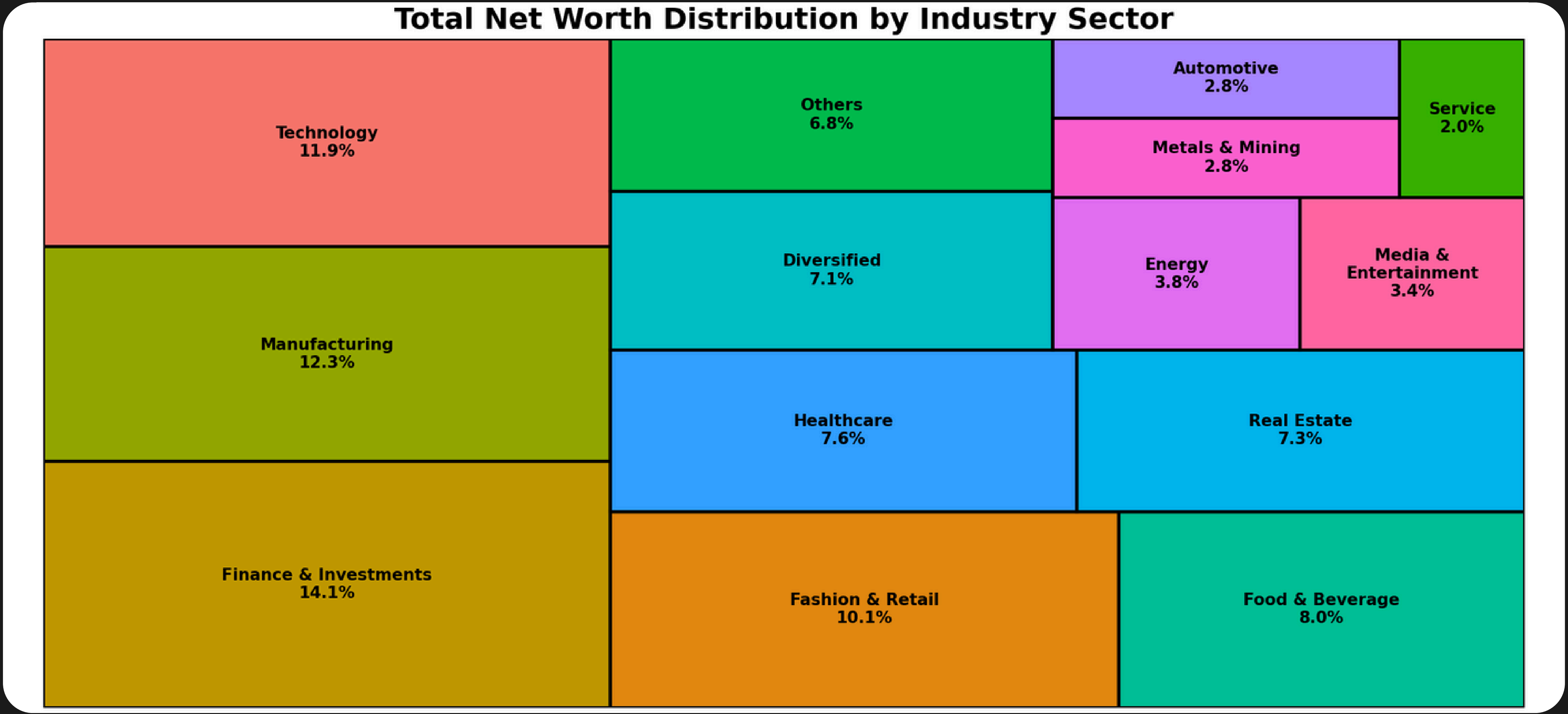
PLOTS & DESCRIPTIONS

Plot 6: Donut Chart



PLOTS & DESCRIPTIONS

Plot 7: Tree Map



PLOTS & DESCRIPTIONS

Plots 6+7: Combined Analysis

The two charts show the distribution of billionaires by industry and their total net worth. The first chart (a treemap) shows each sector's share of the global net worth, while the second chart (a donut chart) displays the number of billionaires in each sector. From these, questions like "Which industries are the wealthiest?" and "Are the richest industries also the ones with the most billionaires?" may arise. You might also wonder how concentrated billionaire wealth is across sectors.

The biggest contributor to global billionaire wealth is finance & investments, which makes up 14.1% of the total net worth. It is followed then by industries manufacturing at 12.3% and technology at 11.9%. Despite the popularity of technology, finance remains the most lucrative industry. Food & beverage, fashion & retail, and healthcare together account for between 7% and 10% of billionaire wealth, highlighting the diversity of wealth sources. On the other hand, sectors like metals & mining, automotive, and services each contribute less than 3%, which raises questions about their scale or profitability.

PLOTS & DESCRIPTIONS

Plots 6+7: Combined Analysis

While technology isn't the largest source of wealth, it leads in the number of billionaires, making up 15.4% of the total, as shown in the second chart. This means that although there are more tech billionaires, their personal wealth may be slightly less. Finance & investments comes next with 13.1%, followed closely by fashion & retail at 13.9%. Interestingly, manufacturing has a higher average net worth per billionaire, even though it ranks second in total wealth and sixth in the number of billionaires (8.3%). One surprising fact is that, despite its high profile, only 3.5% of billionaires are in the industry of media and entertainment, which represents a smaller share of overall wealth.

In conclusion, these graphs represent the total wealth of a sector; it doesn’t always match the number of billionaires it has. While technology has the most billionaires, manufacturing and finance are where the majority of wealth lies. This highlights that quantities don’t always equal high valuation and that the size and structure of companies differ greatly across industries.

PLOTS & DESCRIPTIONS

Plot 6: Code

```
# Load library
import pandas as pd
import plotly.graph_objects as go

# Load the dataset
Billionaires_Statistics_Dataset = pd.read_csv("Billionaires_Statistics_Dataset.csv")

# Compute total worth per category and calculate percentages
final_data = (Billionaires_Statistics_Dataset.groupby("category")["finalWorth"]
        .sum()
        .reset_index()
        .rename(columns={"finalWorth": "totalWorth"}))
final_data["percentage"] = (final_data["totalWorth"] /
final_data["totalWorth"].sum()) * 100
final_data["category"] = final_data.apply(lambda row: "Others" if row["percentage"]
< 2 else row["category"], axis=1)
final_data = (final_data.groupby("category")[["totalWorth", "percentage"]]
        .sum()
        .reset_index()
        .sort_values(by="percentage", ascending=False))
# Manually define colors
my_colors = ["#F8766D", "#DF8A0E", "#BC9C00", "#87AB00", "#0CB702",
"#00BE73", "#00C0AF", "#00B9DE", "#00A9FF", "#948EFF", "#D972F6",
"#FA62D4", "#FF68A1"][:len(final_data)]
```

```
# Create a donut chart
fig = go.Figure(go.Pie(
 labels=[f"<b>{cat}: {perc:.1f}%</b>" for cat, perc in zip(final_data["category"],
final_data["percentage"])],
 values=final_data["percentage"],
 marker=dict(colors=my_colors),
 hole=0.6,
 pull=[0.1] * len(final_data),
 sort=False,
 textinfo="label",
 textposition="outside"
))
fig.update_traces(
 sort=False,
 direction='clockwise',
 textfont=dict(size=8.9, color='black'),)
fig.update_layout(
 height=600,
 width=800,
 title_text="<b>Billionaire Distribution by Industry Sector</b>",
 title_x=0.5,
 showlegend=False,
 font=dict(color='black'))
fig.show()
```

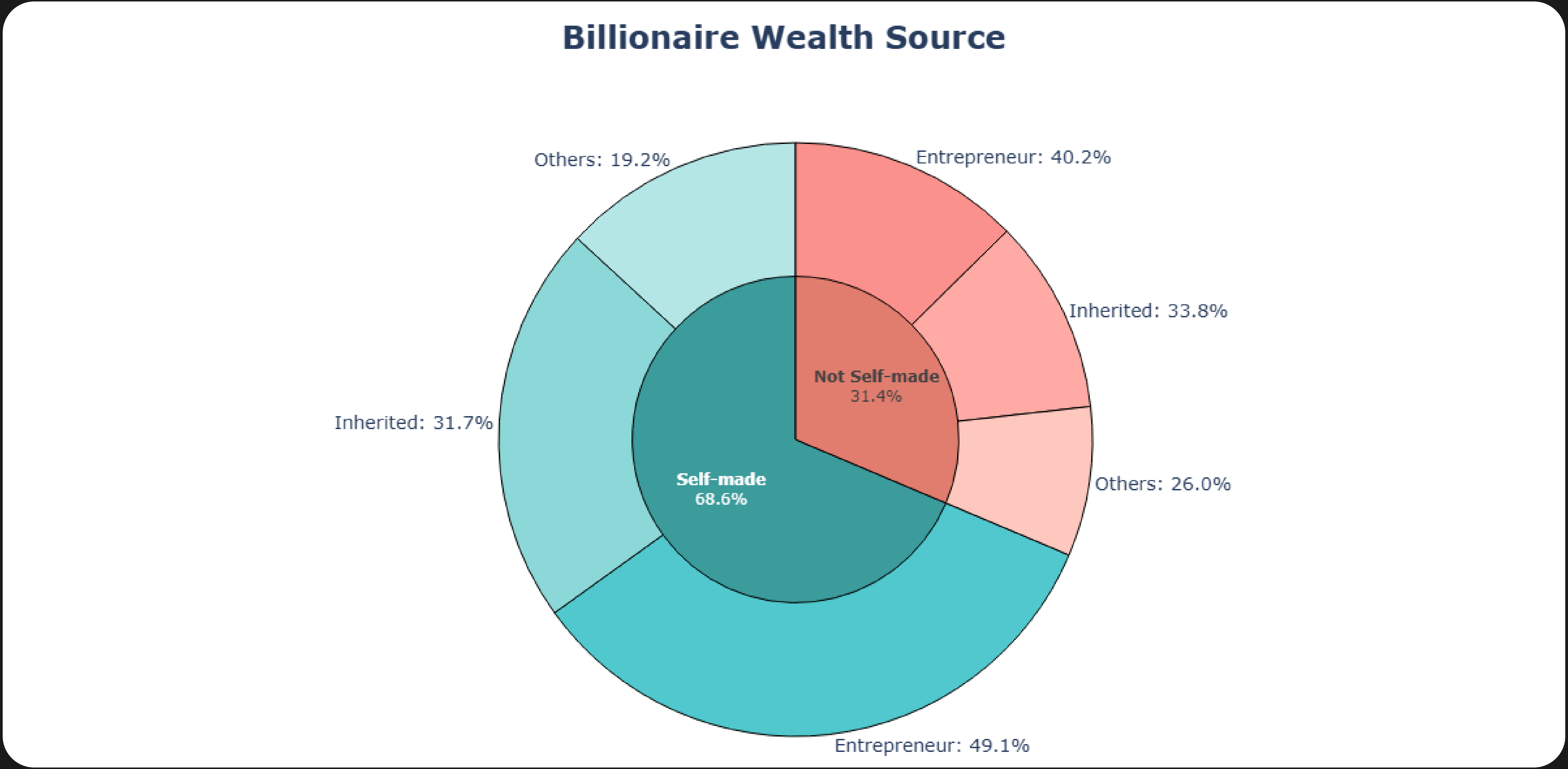
PLOTS & DESCRIPTIONS

Plot 7: Code

```
# Load library
import pandas as pd
import matplotlib.pyplot as plt
import squarify
df = pd.read_csv("Billionaires_Statistics_Dataset.csv")

# Process data
category_counts = df.groupby("category").size().reset_index(name="count")
category_counts["percentage"] = (category_counts["count"] /
category_counts["count"].sum()) * 100
category_counts["category"] = category_counts.apply(lambda row: "Others" if
row["percentage"] < 2 else row["category"], axis=1)
category_counts = category_counts.groupby("category")[["count",
"percentage"]].sum().reset_index()

# Rename category for formatting
category_counts["category"] = category_counts["category"].replace({"Media &
Entertainment": "Media &\nEntertainment"})
category_counts["label"] = category_counts["category"] + "\n" +
category_counts["percentage"].round(1).astype(str) + "%"

# Define the category order
category_order = ["Finance & Investments", "Manufacturing", "Technology", "Fashion
& Retail", "Food & Beverage", "Healthcare", "Real Estate", "Diversified", "Others",
"Energy", "Media &\nEntertainment", "Metals & Mining", "Automotive", "Service"]

# Convert category column to categorical type with specified order
category_counts["category"] = pd.Categorical(category_counts["category"],
categories=category_order, ordered=True)
category_counts = category_counts.sort_values("category")

# Define colors
my_colors = ["#C29A00", "#95A800", "#F8766D", "#E28810", "#00C096",
        "#32A0FF", "#00B4EC", "#00BEC6", "#02BC4F", "#E16EF2",
        "#FF68A1", "#FC62D1", "#A887FF", "#37B301"]

# Create treemap with bold labels
plt.figure(figsize=(10, 6))
squarify.plot(
 sizes=category_counts["count"],
 label=category_counts["label"],
 color=my_colors,
 edgecolor="black",
 linewidth=2,
 text_kwargs={'fontsize': 10, 'fontweight': 'bold'}
)
plt.title("Total Net Worth Distribution by Industry Sector", fontsize=18,
fontweight="bold")
plt.axis("off")
plt.show()
```

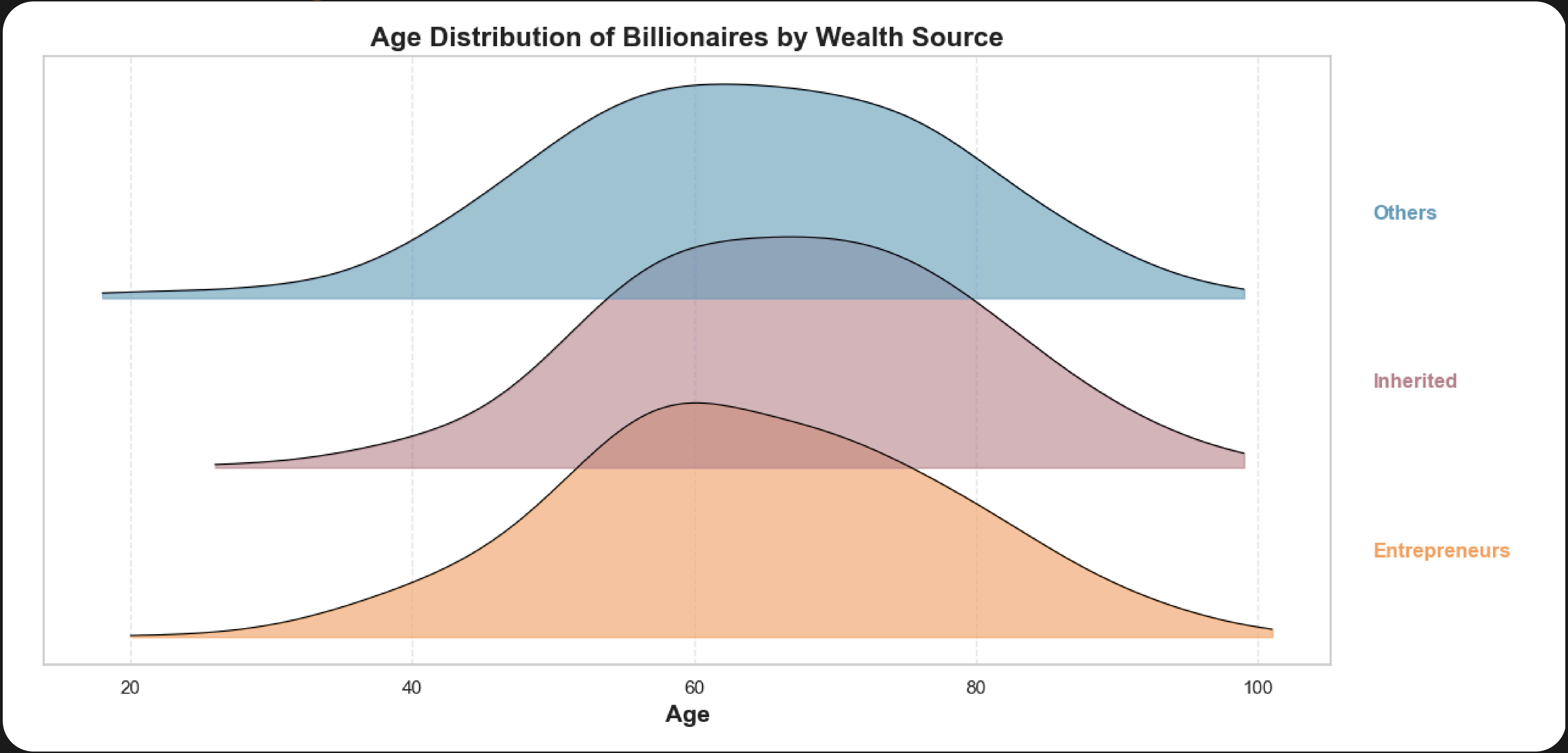
PLOTS & DESCRIPTIONS

Plot 8: Pie-Donut Chart



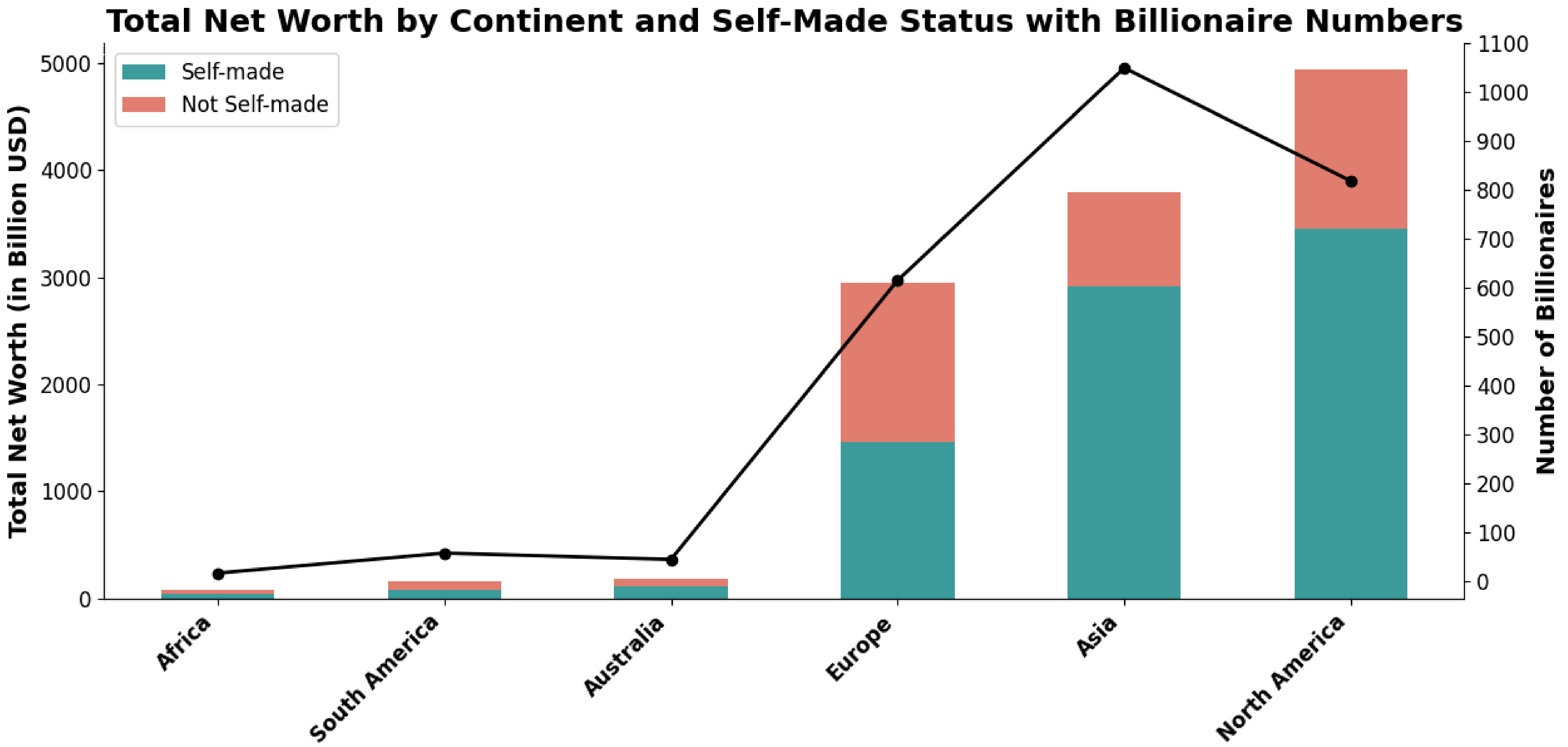
PLOTS & DESCRIPTIONS

Plot 9: Ridgeline Plot



PLOTS & DESCRIPTIONS

Plot 10: Stacked Bar Chart with Line



PLOTS & DESCRIPTIONS

Plots 8+9+10: Combined Analysis

The three charts break down how billionaires built their wealth, how age varies by wealth source, and how net worth and self-made status differ by continent in 2023.

Overall, most billionaires are self-made, with entrepreneurship being the top way to achieve that status. Entrepreneurs are younger in age compared to those who inherit wealth. North America and Asia are the most dominant, leading in both the amount of wealth and the number of billionaires.
The first graph which is a pie-donut chart shows that 68.64% of billionaires are self-made and 31.36% are not. Among the self-made, nearly half (49.1%) made their wealth through entrepreneurship. Another 31.7% had a mix of inheritance and other sources, and 19.2% through other means. Interestingly, even among the non–self-made, entrepreneurship remains a major factor (40.2%), though inheritance takes the lead at 33.8%.

The second graph shows how age differs by wealth origin. Entrepreneurs tend to peak in their late 50s to early 60s, making them younger than those who inherited their wealth or gained it through other channels. The inherited wealth group spans a broader age range, suggesting their wealth often sticks around for longer.

PLOTS & DESCRIPTIONS

Plots 8+9+10: Combined Analysis

The third chart compares billionaire net worth and population across continents by whether they're self-made. North America leads the pack in both total wealth and billionaire count, followed by Asia and Europe. Across all three regions, self-made billionaires account for the larger share of total net worth. While Africa, South America, and Australia have fewer billionaires and lower overall wealth, self-made individuals still make up the majority in these areas.

In summary, most billionaires around the world are self-made, with entrepreneurship being the main driving force. They also tend to be younger than those who inherited wealth. North America and Asia are home to the wealthiest and most numerous billionaires on the planet.

PLOTS & DESCRIPTIONS

Plot 8: Code

```
# Load library
import pandas as pd
import numpy as np
import plotly.graph_objects as go

# Load Dataset
billionaires = pd.read_csv("Billionaires_Statistics_Dataset.csv")
df = billionaires.copy()

# Data Transformation
df['status'] = df['status'].replace({'D': 'Entrepreneur', 'U': 'Inherited'})
df['status'] = np.where(df['status'].isin(['E', 'R', 'N', 'Split Family Fortune']),
'Others', df['status'])

df['selfMade'] = df['selfMade'].map({
  True: 'Self-made',
  False: 'Not Self-made'
}).fillna(df['selfMade'].astype(str))

# Inner Circle
table3 = (
  df.groupby('selfMade')
  .size()
  .reset_index(name='Frequency')
  .rename(columns={'selfMade': 'Selfmade'})
)
table3['Percentage'] = round((table3['Frequency'] / table3['Frequency'].sum())
* 100, 2)

inner_labels = table3['Selfmade'].tolist()
inner_values = table3['Percentage'].tolist()
inner_colors = ["#e27f72", "#3d9b9d" ]

# Outer Circle
group_totals = df['selfMade'].value_counts(normalize=True).round(4)

breakdown = (
  df.groupby(['selfMade', 'status'])
  .size()
  .reset_index(name='count')
)

breakdown['within_group'] = breakdown.groupby('selfMade')
['count'].transform(lambda x: x / x.sum())
breakdown['scaled_value'] = breakdown.apply(
  lambda row: row['within_group'] * group_totals[row['selfMade']], axis=1)
```

PLOTS & DESCRIPTIONS

Plot 8: Code

```
breakdown['scaled_value'] *= 100
breakdown['display_pct'] = (breakdown['within_group'] * 100).round(2)
outer_labels = breakdown.apply(
  lambda row: f"{row['status']}: {row['display_pct']}%", axis=1
).tolist()

outer_values = breakdown['scaled_value'].round(2).tolist()
outer_colors = ["#fa938d", "#ffaea5", "#ffc9c2","#51ccd0", "#8fd9db", "#b7e6e7"]

# Visualization
rotation_angle = 0
fig = go.Figure()

# Inner Pie
fig.add_trace(go.Pie(
  labels=inner_labels,
  values=inner_values,
  hole=0,
  sort=False,
  direction='clockwise',
  rotation=rotation_angle,
  marker=dict(colors=inner_colors, line=dict(color='black', width=1)),
  pull=[0] * len(inner_labels),
  texttemplate="<b>%{label}</b><br>%{percent}",
  textposition='inside',
  insidetextorientation='horizontal',
  textfont=dict(size=12),
  showlegend=False,
  domain={'x': [0.20, 0.80], 'y': [0.20, 0.80]}
))

# Outer Ring
fig.add_trace(go.Pie(
  labels=outer_labels,
  values=outer_values,
  hole=0.55,
  sort=False,
  direction='clockwise',
  rotation=rotation_angle,
  marker=dict(colors=outer_colors, line=dict(color='black', width=1)),
  pull=[0] * len(outer_labels),
  textinfo='label',
  textposition='outside',
  textfont=dict(size=14),
  showlegend=False,
  domain={'x': [0, 1], 'y': [0, 1]}))
```

PLOTS & DESCRIPTIONS

Plot 8: Code

```
fig.update_layout(
  title_text="<b>Billionaire Wealth Source</b>",
  title_x=0.5,
  title_font=dict(size=24, color='black'),
  margin=dict(t=100, b=100, l=50, r=50),
  height=700,
  width=700
)

fig.show()
```

PLOTS & DESCRIPTIONS

Plot 9: Code

```
# Load library
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import gaussian_kde

# Load data
df = pd.read_csv('Billionaires_Statistics_Dataset.csv')
df = df.dropna(subset=['age']).copy()

# Classify wealth source
mapping = {'D': 'Entrepreneurs', 'U': 'Inherited'}
df['wealth_source'] = pd.Categorical(
  df['status'].map(mapping).fillna('Others'),
  categories=['Entrepreneurs', 'Inherited', 'Others'],
  ordered=True
)

# Plot settings
sns.set(style="whitegrid")
colors = ['#F4A261', '#B5838D', '#669BBC']
offsets = np.linspace(0, 0.04, 3)
```

```
fig, ax = plt.subplots(figsize=(10, 6))

# Plot each wealth source
for i, (group, color, offset) in
enumerate(zip(df['wealth_source'].cat.categories, colors, offsets)):
  subset = df[df['wealth_source'] == group]['age']
  xs = np.linspace(subset.min(), subset.max(), 200)
  ys = gaussian_kde(subset, bw_method=5/np.std(subset))(xs) + offset
  ax.fill_between(xs, offset, ys, alpha=0.6, color=color)
  ax.plot(xs, ys, color='black', linewidth=0.8)

# Add text labels
xmax = ax.get_xlim()[1]
for group, color, offset in zip(df['wealth_source'].cat.categories, colors,
offsets):
  ax.text(xmax + 3, offset + 0.01, group, ha='left', va='center',
       fontsize=12, fontweight='bold', color=color, clip_on=False)

# Styling
ax.set(title='Age Distribution of Billionaires by Wealth Source',
    xlabel='Age')
ax.set_xlabel('Age', fontsize=14, fontweight='bold')
ax.title.set_fontsize(16)
```

PLOTS & DESCRIPTIONS

Plot 9: Code

```
ax.title.set_fontweight('bold')
ax.set_yticks([])
ax.grid(axis='x', linestyle='--', alpha=0.4)

plt.tight_layout()
plt.show()
```

PLOTS & DESCRIPTIONS

Plot 10: Code

```
# Import necessary packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import country_converter as coco

# Load the dataset
df = pd.read_csv("Billionaires_Statistics_Dataset.csv")
continent_order = ["Africa", "South America", "Australia", "Europe", "Asia",
"North America"]

# Mapping countries
def custom_country_to_continent(country):
    overrides = {
        'United States': 'North America', 'Canada': 'North America', 'Mexico':
'North America',
        'Bahamas': 'North America', 'Cayman Islands': 'North America', 'Bermuda':
'North America',
        'British Virgin Islands': 'North America', 'Turks and Caicos Islands': 'North
America',
        'Brazil': 'South America', 'Argentina': 'South America', 'Chile': 'South
America',
'Colombia': 'South America', 'Peru': 'South America', 'Venezuela': 'South America',
        'Ecuador': 'South America', 'Uruguay': 'South America', 'Paraguay': 'South
America',
        'Bolivia': 'South America',
        'Russia': 'Europe', 'France': 'Europe', 'Germany': 'Europe', 'United Kingdom':
'Europe',
        'Switzerland': 'Europe', 'Belgium': 'Europe', 'Austria': 'Europe', 'Monaco':
'Europe',
        'Czech Republic': 'Europe', 'Sweden': 'Europe', 'Norway': 'Europe', 'Denmark':
'Europe',
        'Netherlands': 'Europe', 'Poland': 'Europe', 'Ukraine': 'Europe', 'Greece': 'Europe',
        'Turkey': 'Europe', 'Portugal': 'Europe', 'Latvia': 'Europe', 'Finland': 'Europe',
        'Luxembourg': 'Europe', 'Ireland': 'Europe', 'Cyprus': 'Europe', 'Guernsey':
'Europe',
        'Liechtenstein': 'Europe', 'Romania': 'Europe', 'Slovakia': 'Europe', 'Hungary':
'Europe',
        'Andorra': 'Europe',
        'Egypt': 'Africa', 'South Africa': 'Africa', 'Nigeria': 'Africa', 'Kenya': 'Africa',
        'Eswatini (Swaziland)': 'Africa', 'Algeria': 'Africa', 'Morocco': 'Africa', 'Tanzania':
'Africa',
        'China': 'Asia', 'India': 'Asia', 'Japan': 'Asia', 'South Korea': 'Asia', 'Indonesia':
'Asia',
```

PLOTS & DESCRIPTIONS

Plot 10: Code

```
 'Saudi Arabia': 'Asia', 'United Arab Emirates': 'Asia', 'Thailand': 'Asia',
'Uzbekistan': 'Asia',
    'Singapore': 'Asia', 'Israel': 'Asia', 'Malaysia': 'Asia', 'Philippines': 'Asia',
'Taiwan': 'Asia',
    'Kazakhstan': 'Asia', 'Vietnam': 'Asia', 'Cambodia': 'Asia', 'Lebanon': 'Asia',
'Oman': 'Asia',
    'Qatar': 'Asia', 'Nepal': 'Asia', 'Armenia': 'Asia', 'Bahrain': 'Asia', 'Georgia':
'Asia',
    'Australia': 'Australia', 'New Zealand': 'Australia'
  }
  if pd.isnull(country):
    return np.nan
  if country in overrides:
    return overrides[country]
  try:
    result = coco.convert(names=[country], to='continent')
    return result
  except Exception:
    return np.nan

df['continent'] = df['country'].apply(custom_country_to_continent)
df['selfMade'] = np.where(df['selfMade'], 'Self-made', 'Not Self-made')
df = df[df['continent'].isin(continent_order)]
```

```
billionaires_grouped = df.groupby(['continent', 'selfMade'],
as_index=False).agg({
  'finalWorth': 'sum'
})
pivot_data = billionaires_grouped.pivot(index='continent', columns='selfMade',
values='finalWorth').fillna(0)

billionaires_count = df.groupby('continent',
as_index=False).size().rename(columns={'size': 'count_of_billionaires'})

pivot_data = pivot_data.reindex(continent_order)
colors = {'Self-made': '#3d9b9d', 'Not Self-made': '#e27f72'}
scaling_factor = 4543
fig, ax1 = plt.subplots(figsize=(12, 8))

# Stacked Bar Plot
pivot_data[['Self-made', 'Not Self-made']].plot(
  kind='bar',
  stacked=True,
  color=[colors['Self-made'], colors['Not Self-made']],
  ax=ax1
)
```

PLOTS & DESCRIPTIONS

Plot 10: Code

```
ax1.ticklabel_format(style='plain', axis='y')
y_ticks = ax1.get_yticks()
ax1.set_yticklabels((y_ticks/1000).astype(int))

ax2 = ax1.twinx()
counts = billionaires_count.set_index('continent').loc[continent_order]
['count_of_billionaires']
ax2.plot(
    pivot_data.index,
    counts * scaling_factor,
    color='black',
    marker='o',
    linewidth=2,
    label='Number of Billionaires'
)

for spine in ['left', 'bottom', 'top', 'right']:
    ax1.spines[spine].set_color('black')
    ax2.spines[spine].set_color('black')

# Axis labels and title
ax1.set_ylabel('Total Net Worth (in Billion USD)', fontsize=14,
fontweight='bold', color='black')
ax2.set_ylabel('Number of Billionaires', fontsize=14, fontweight='bold',
color='black')
ax1.set_xlabel('')
ax1.set_title(
    'Total Net Worth by Continent and Self-Made Status with Billionaire
Numbers',
    fontsize=18,
    fontweight='bold',
    color='black'
)

# Customize ticks and labels
ax1.set_xticklabels(ax1.get_xticklabels(), rotation=45, ha='right', fontsize=12,
fontweight='bold', color='black')
ax1.tick_params(axis='y', labelsize=12, colors='black')
ax2.tick_params(axis='y', labelsize=12, colors='black')
ax2.set_yticks(np.arange(0, (counts.max() + 100) * scaling_factor, 100 *
scaling_factor))
ax2.set_yticklabels(np.arange(0, counts.max() + 100, 100))
```

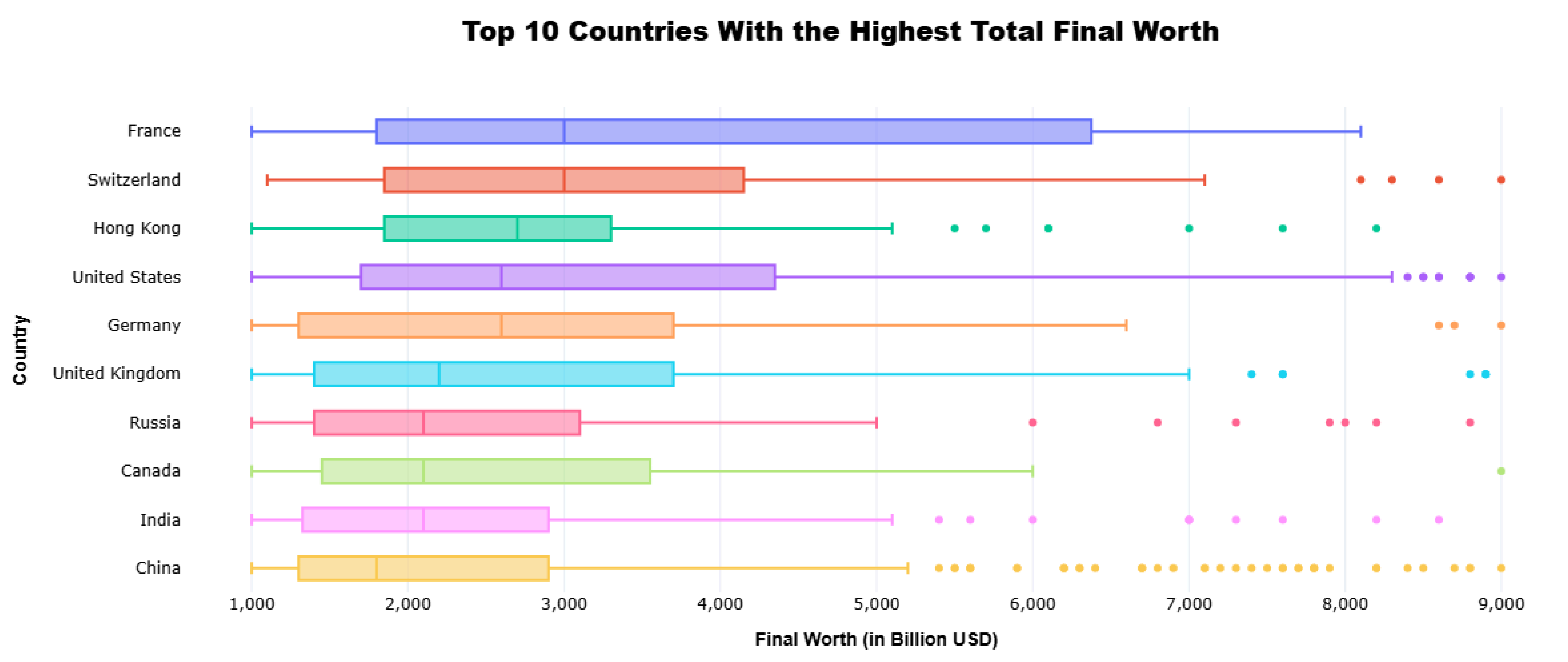
PLOTS & DESCRIPTIONS

Plot 10: Code

```
# Legends
ax1.legend(fontsize=12)
sns.despine()
ax2.spines["right"].set_visible(True)
plt.tight_layout()
plt.show()
```

PLOTS & DESCRIPTIONS

Plot 11: Box Plot



PLOTS & DESCRIPTIONS

Plot 11: Box Plot Analysis

The box plot illustrates the distribution of billionaire net worth across ten different countries. It presents key statistical indicators such as the median, interquartile range (IQR), and outliers. Upon first glance, a viewer may wonder: Which country's billionaires are the wealthiest? Which nations have more evenly distributed wealth? And where are the extremes most visible?

One immediate observation is that the United States and France stand out due to their extremely high outliers, reaching above 7,500 million. Particularly, the United States shows a wide range with numerous individual outliers, suggesting the presence of exceptionally wealthy individuals. Similarly, France, despite a narrower IQR, has a remarkably high upper whisker and many outliers, indicating that several of its billionaires possess far above-average wealth. These patterns may lead one to question whether such wealth concentration reflects dominant global companies or economic policies favoring the ultra-rich.

PLOTS & DESCRIPTIONS

Plot 11: Box Plot Analysis

France has the highest median billionaire net worth among all countries, suggesting that most of its billionaires are wealthier than those in other nations. In contrast, China, India, and Russia display lower median values, even though they contain a significant number of outliers. This raises the question: Does having many billionaires with extreme net worth necessarily mean a country's average billionaire is richer? Apparently not. Countries such as Germany, the UK, and Switzerland have balanced IQRs and fewer extreme values, implying a more evenly distributed billionaire wealth range.

Interestingly, Russia and Hong Kong show narrow IQRs but include multiple extreme outliers, indicating that only a few individuals control most of the wealth. Conversely, countries like Canada and Switzerland have more symmetric and compressed distributions, suggesting more consistency among their billionaire populations. This variation might reflect national differences in economic equality or dominant industry sectors.

PLOTS & DESCRIPTIONS

Plot 11: Box Plot Analysis

In summary, while some countries such as France and the United States are home to billionaires with high net worth, others like Germany and Switzerland exhibit more uniform distributions. These findings suggest that billionaire wealth is not only influenced by economic size but also by how wealth is concentrated or distributed in every nation.

PLOTS & DESCRIPTIONS

Plot 11: Code

```
# Import library
import pandas as pd
import plotly.express as px

# Load dataset
billionaires = pd.read_csv('Billionaires_Statistics_Dataset.csv')

# Filter top 10 countries by total net worth
top_10_countries = (
   billionaires[billionaires['finalWorth'].notna()]
   .groupby('country')['finalWorth']
   .sum()
   .sort_values(ascending=False)
   .head(10)
   .index
)

# Filter dataset to top 10 countries and remove outliers using IQR
top_countries_data =
billionaires[billionaires['country'].isin(top_10_countries)]
Q1 = top_countries_data['finalWorth'].quantile(0.25)
Q3 = top_countries_data['finalWorth'].quantile(0.75)
IQR = Q3 - Q1
```

```
top_countries_data_cleaned = top_countries_data[
   (top_countries_data['finalWorth'] >= (Q1 - 1.5 * IQR)) &
   (top_countries_data['finalWorth'] <= (Q3 + 1.5 * IQR))
]

# Compute median final worth for each country
median_df = top_countries_data_cleaned.groupby('country',
as_index=False)['finalWorth'].median()
median_df.rename(columns={'finalWorth': 'medianWorth'}, inplace=True)

# Merge median data and sort by median worth
top_countries_data_cleaned =
top_countries_data_cleaned.merge(median_df, on='country', how='left')
top_countries_data_cleaned =
top_countries_data_cleaned.sort_values('medianWorth', ascending=False)

# Create Plotly boxplot
fig = px.box(
   top_countries_data_cleaned,
   x='finalWorth',
   y='country',
   color='country',
```

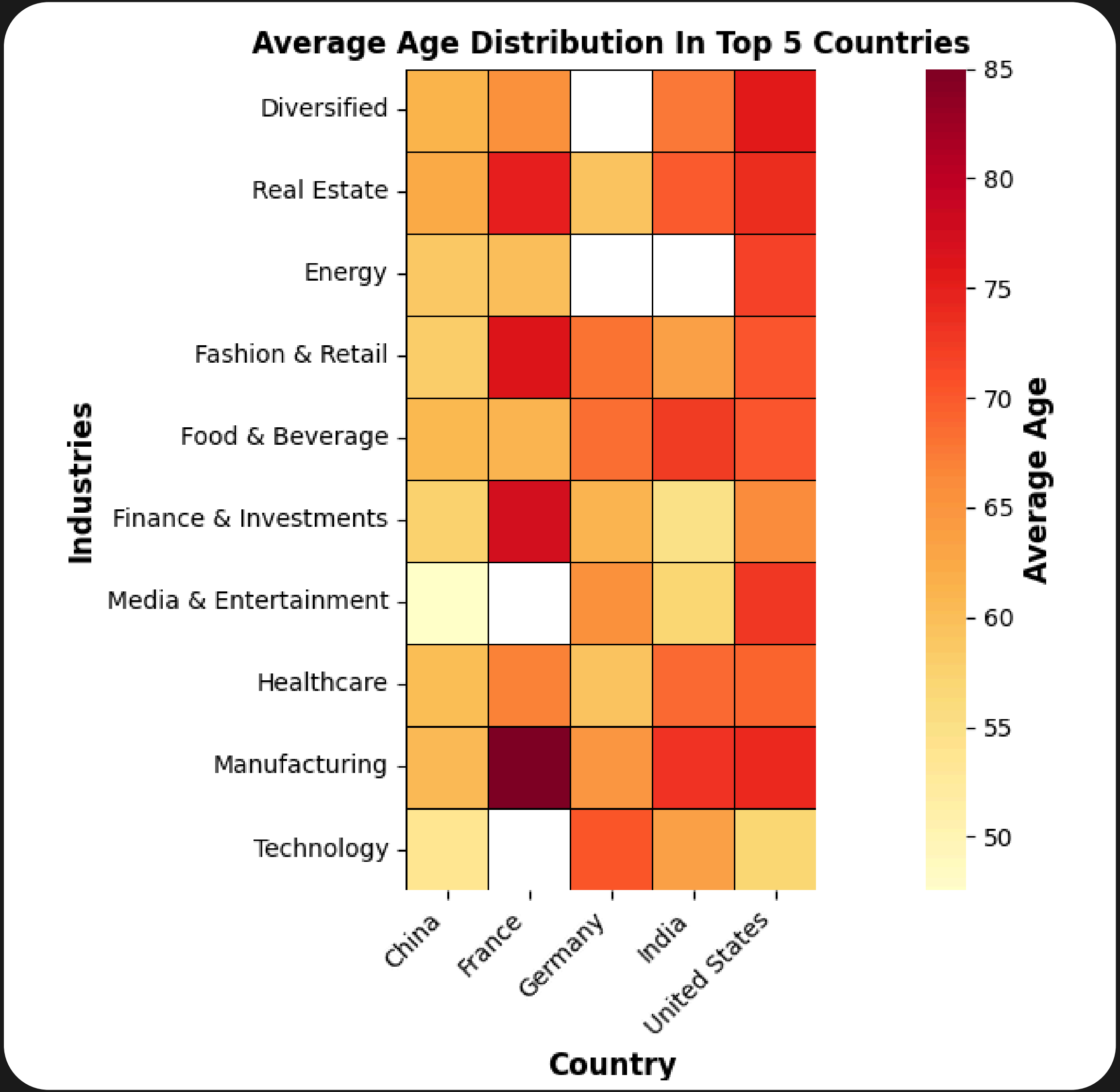
PLOTS & DESCRIPTIONS

Plot 11: Code

```
 title='Top 10 Countries With the Highest Total Final Worth',
   labels={'finalWorth': '**Final Worth (in Billion USD)**', 'country':
'**Country**'},
   points='outliers'
)

# Update layout to center title and set text colors
fig.update_layout(
   title_font=dict(size=20, family="Arial Black", color="black"),
   xaxis_title_font=dict(size=14, family="Arial", color="black"),
   yaxis_title_font=dict(size=14, family="Arial", color="black"),
   xaxis=dict(title=dict(text='<b>Final Worth (in Billion USD)</b>',
font=dict(size=14, color="black"))),
   yaxis=dict(title=dict(text='<b>Country</b>', font=dict(size=14,
color="black"))),
   showlegend=False,
   template='plotly_white',
   font=dict(color="black"), # Set font color to black for all text elements
   title_x=0.5, # Center the title
)

# Set tick font color to black
fig.update_xaxes(tickformat=',', tickfont=dict(color="black"))
fig.update_yaxes(tickfont=dict(color="black"))

fig.show()
```

PLOTS & DESCRIPTIONS

Plot 12: Heat Map



PLOTS & DESCRIPTIONS

Plot 12: Heat Map Analysis

The heatmap illustrates the average age of billionaires across various industries in five countries: China, France, Germany, India, and the United States. Color intensity increases with higher average age, ranging from light yellow (younger) to dark brown (older). Several questions arise: Which sector has the youngest or oldest billionaires? Do some countries produce younger billionaires overall? And how does industry type influence age?

At first glance, France and Germany appear to have relatively older billionaires, with darker shades dominating across most sectors, especially in Manufacturing, Fashion & Retail, and Finance & Investments, where average ages exceed 70. This suggests that billionaires in these traditional industries tend to accumulate wealth later in life. The Manufacturing sector in Germany shows the darkest cell in the chart, indicating the oldest average age among all countries and sectors, possibly due to long-established family businesses and conservative economic growth patterns.

PLOTS & DESCRIPTIONS

Plot 12: Heat Map Analysis

In contrast, China and India demonstrate relatively lighter shades, indicating younger average billionaire ages, particularly in Technology, Media & Entertainment, and Energy. For instance, China's average billionaire age in the Technology sector appears significantly lower than in other industries, suggesting a younger generation capitalizing on the tech boom. Similarly, India's energy and tech billionaires also seem younger, potentially reflecting rapid growth in start-ups and tech-driven industries.

Regardless of country, certain sectors tend to skew older or younger. Fashion & Retail, Real Estate, and Manufacturing often show higher ages across most countries, while Technology and Media & Entertainment generally attract younger billionaires. This raises an important insight: age may not just depend on geography but also on industry life cycle—tech sectors are fast-moving, while real estate or fashion may require longer-term capital accumulation.

In summary, the heatmap reveals that industry type has a stronger influence on billionaire age than country alone. While traditional sectors foster older billionaires, emerging sectors like technology are led by younger individuals, especially in developing countries like China and India. This highlights how innovation and market maturity affect wealth creation timelines across the globe.

PLOTS & DESCRIPTIONS

Plot 12: Code

```
# Import library
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load dataset
billionaires = pd.read_csv("Billionaires_Statistics_Dataset.csv")
billionaires.columns = billionaires.columns.str.lower()

# Rename columns for consistency
billionaires.rename(columns={'country': 'Country', 'industries': 'Industries'},
inplace=True)

# Select top 5 countries by total net worth
top_countries = (
  billionaires.dropna(subset=['Country', 'finalworth'])
  .groupby('Country')['finalworth']
  .sum()
  .nlargest(5)
  .index
)
# Select top 10 industries by number of billionaires
top_industries = (
billionaires.dropna(subset=['Industries'])
  .groupby('Industries')
  .size()
  .nlargest(10)
  .index
)

# Compute average age per industry-country pair
heatmap_data = (
  billionaires[
    billionaires['Country'].isin(top_countries) &
    billionaires['Industries'].isin(top_industries)
  ]
  .groupby(['Industries', 'Country'])['age']
  .mean()
  .unstack(fill_value=float('nan'))
)

# Compute average age per industry overall
industry_avg_age = (
  billionaires[billionaires['Industries'].isin(top_industries)]
  .groupby('Industries')['age']
  .mean()
```

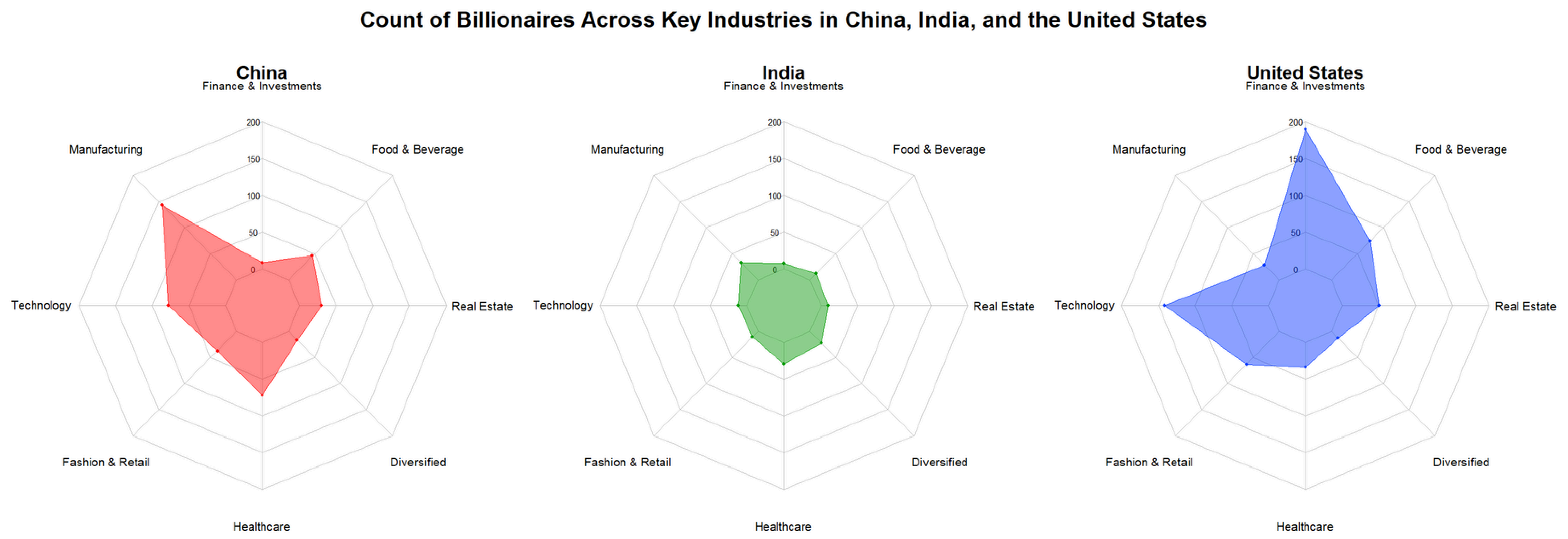
PLOTS & DESCRIPTIONS

Plot 12: Code

```
.sort_values(ascending=False)
)

# Reorder industries based on overall average age
heatmap_data = heatmap_data.loc[industry_avg_age.index]

# Set color map
custom_cmap = sns.color_palette("YlOrRd", as_cmap=True)

# Plot heatmap
plt.figure(figsize=(10, 7))
sns.heatmap(
    heatmap_data,
    cmap=custom_cmap,
    annot=False,
    linewidths=0.5,
    linecolor='black',
    cbar_kws={"label": "Average Age"},
    square=True
)

# Customize title and labels
plt.title("Average Age Distribution In Top 5 Countries", fontsize=12,
fontweight='bold')
plt.xticks(rotation=45, ha="right", fontsize=10)
plt.xlabel("Country", fontsize=12, fontweight='bold')
plt.ylabel("Industries", fontsize=12, fontweight='bold')

# Customize colorbar label
cbar = plt.gca().collections[0].colorbar
cbar.set_label("Average Age", fontsize=12, fontweight='bold')

plt.tight_layout()
plt.show()
```

PLOTS & DESCRIPTIONS

Plot 13: Radar Chart



PLOTS & DESCRIPTIONS

Plot 13: Radar Chart Analysis

The radar chart indicates the distribution of the billionaires in eight major Chinese, Indian, and United States industries in 2023. The eight major industries include Manufacturing, Finance & Investments, Food & Beverage, Real Estate, Fashion & Retail, Healthcare, Technology, and Diversified. These industries have been selected because they are the top eight in terms of billionaire distribution.

The scale on the graph goes up to 200 because 190 is the highest number of billionaires in any sector across the three countries. The ranges 50, 100, 150, and 200 are sufficient to allow easy comparison between sectors in the three countries.

Manufacturing and Finance & Investments take the lead when it comes to Chinese billionaires, with the highest numbers belonging to Manufacturing, followed by Technology with the second-highest numbers. Other sectors like Healthcare and Finance & Investments have much lower numbers.

PLOTS & DESCRIPTIONS

Plot 13: Radar Chart Analysis

The Indian portfolio is better diversified, with strong representations in Finance & Investments and Food & Beverage. Billionaires in areas like Technology and Real Estate are fewer in comparison to China and the US. Fashion & Retail is another strong sector in India.

In the US, the distribution is diversified. Finance & Investments and Food & Beverage both have high numbers of billionaires, as does Real Estate. Technology is also a very strong category, and Manufacturing has proportionately fewer billionaires than in China.

In general, the graph shows that China's billionaires are primarily in the Manufacturing and Healthcare segments, while Indian billionaires are evenly split across different segments. The United States shows an even distribution with a strong presence in several segments. The fact that 200 is the ceiling, along with clear scale intervals, makes it easy to observe the fluctuations in the distribution of billionaires across these segments and countries.

PLOTS & DESCRIPTIONS

Plot 13: Code

```
# Import library
import pandas as pd
import plotly.graph_objects as go

# Load dataset
billionaires = pd.read_csv("Billionaires_Statistics_Dataset.csv")

# Ordered industries
selected_industries = [
    "Finance & Investments", "Food & Beverage", "Real Estate", "Diversified",
    "Healthcare", "Fashion & Retail", "Technology", "Manufacturing"
]

# Top 3 countries
top_countries = billionaires['country'].value_counts().nlargest(3).index.tolist()

# Filter data
df_filtered = billionaires[
    (billionaires['industries'].isin(selected_industries)) &
    (billionaires['country'].isin(top_countries))
]
# Count and reindex
industry_counts = (
df_filtered.groupby(['industries', 'country'])
    .size()
    .unstack(fill_value=0)
    .reindex(selected_industries)
)

# Helper: close radar loop
def get_country_data(country):
    values = industry_counts.get(country, pd.Series([0] *
len(selected_industries))).tolist()
    return values + [values[0]]

industries_closed = selected_industries + [selected_industries[0]]
china = get_country_data("China")
india = get_country_data("India")
usa = get_country_data("United States")

# Create figure
fig = go.Figure()

# China
fig.add_trace(go.Scatterpolar(
    r=china,
```

PLOTS & DESCRIPTIONS

Plot 13: Code

```
theta=industries_closed,
  fill='toself',
  name='China',
  line=dict(color='red'),
  subplot='polar'
))
# India
fig.add_trace(go.Scatterpolar(
  r=india,
  theta=industries_closed,
  fill='toself',
  name='India',
  line=dict(color='green'),
  subplot='polar2'
))
# United States
fig.add_trace(go.Scatterpolar(
  r=usa,
  theta=industries_closed,
  fill='toself',
  name='United States',
  line=dict(color='blue'),
  subplot='polar3'
))
```

```
# Layout with smaller radar charts and centered titles
fig.update_layout(
  title=dict(
    text="<b>Number of Billionaires Across Key Industries (by Country)</b>",
    x=0.5,
    xanchor='center',
    font=dict(size=20)
  ),
  polar=dict(
    domain=dict(x=[0.02, 0.22]),
    radialaxis=dict(range=[0, 200], visible=True),
    angularaxis=dict(direction="clockwise", rotation=90)
  ),
  polar2=dict(
    domain=dict(x=[0.40, 0.60]),
    radialaxis=dict(range=[0, 200], visible=True),
    angularaxis=dict(direction="clockwise", rotation=90)
  ),
  polar3=dict(
    domain=dict(x=[0.78, 0.98]),
    radialaxis=dict(range=[0, 200], visible=True),
    angularaxis=dict(direction="clockwise", rotation=90)
  ),
  annotations=[
    dict(text="<b>China</b>", x=0.12, y=1.12, xref="paper", yref="paper",
showarrow=False,
```

PLOTS & DESCRIPTIONS

Plot 13: Code

```
 font=dict(size=16), align='center', xanchor='center'),

      dict(text="<b>India</b>", x=0.50, y=1.12, xref="paper", yref="paper",
showarrow=False,
          font=dict(size=16), align='center', xanchor='center'),

      dict(text="<b>United States</b>", x=0.88, y=1.12, xref="paper", yref="paper",
showarrow=False,
          font=dict(size=16), align='center', xanchor='center'),
   ],
   showlegend=False,
   height=500,
   width=1400,
   margin=dict(t=100)
)

fig.show()
```

E Conclusion

Billionaires are mainly concentrated in financially developed nations like the U.S. and China, with significant economic disparities between countries. Wealth is more concentrated in these nations due to their developed financial systems and strong industrial growth, while developing countries have fewer billionaires due to economic limitations.

The distribution of billionaires' net worth is highly skewed, with most concentrated in lower wealth brackets, while a few control substantial wealth, especially in sectors like Finance, Manufacturing, and Technology. Wealth is concentrated in a small elite, highlighting global wealth inequality.

Billionaire numbers rise with age, peaking in the 70+ group, with men overwhelmingly dominating across all age groups. The gender gap widens with age, and male billionaires tend to accumulate much more wealth, especially in later years.

Industries like Finance, Manufacturing, and Technology dominate in wealth creation, but while Technology has the most billionaires, its wealth is lower compared to Finance and Manufacturing. Some industries like Metals & Mining and Automotive contribute less to total wealth.

Conclusion

Self-made billionaires, primarily through entrepreneurship, make up the majority (68.64%), with Asia seeing rapid wealth generation. While inherited wealth still plays a role, self-made wealth is on the rise, indicating a trend toward meritocracy in billionaire creation.
North America leads in total billionaire wealth, while Asia has the most billionaires, especially self-made. Europe shows more evenly distributed wealth, while South America and Africa have fewer billionaires due to economic constraints.

The U.S. and France have a broad range of billionaire wealth, including extreme outliers, while countries like Germany and Switzerland have more evenly distributed wealth. Finally, China leads in technology and manufacturing-driven wealth, while the U.S. focuses more on Finance and Services, reflecting differences in national economic structures.