```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

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

# Word Chart
from wordcloud import WordCloud

# Calculate the frequency of each occupation
occupation_counts = df['occupation'].value_counts()

# Print the frequencies sorted in descending order
print("Occupation frequencies (sorted in descending order):")
print(occupation_counts)

# Create a word cloud for the 'occupation' column
text = ' '.join(df['occupation'].dropna())
wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(text)

plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```

```
Occupation frequencies (sorted in descending order):
occupation
Prof-specialty       6172
Craft-repair         6112
Exec-managerial      6086
Adm-clerical         5611
Sales                5504
Other-service        4923
Machine-op-inspct    3022
?                    2809
Transport-moving     2355
Handlers-cleaners    2072
Farming-fishing      1490
Tech-support         1446
Protective-serv       983
Priv-house-serv       242
Armed-Forces           15
Name: count, dtype: int64
```

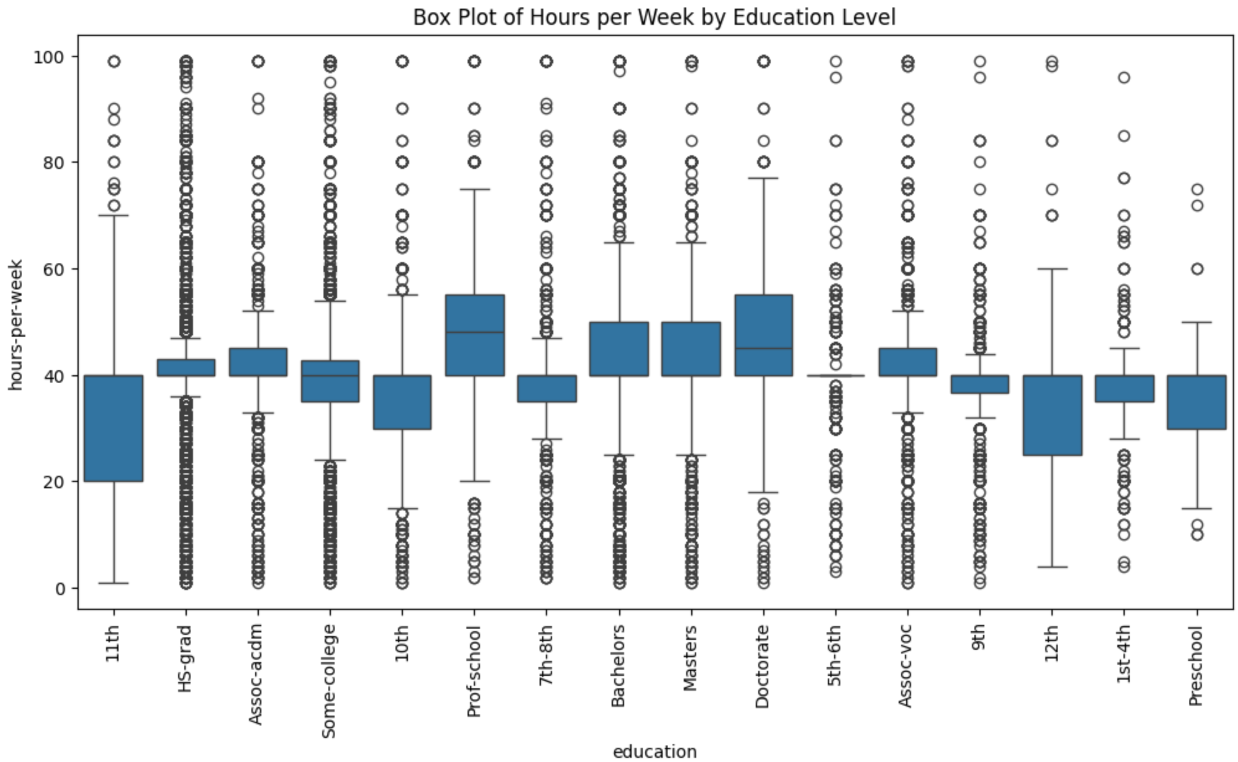
Analysis For Word Chart

Dominant Occupations: Prof-specialty, Craft-repair, and Exec-managerial are the most common occupations, each with over 6,000 occurrences. Sparse Occupations: Roles like Armed-Forces and Priv-house-serv are significantly less frequent, indicating lower representation in the dataset.

```
# Box and Whisker Plot

# Group and aggregate data for the box plot
education_hours = df.groupby('education')['hours-per-week'].describe()

# Print the aggregated data
print("Box Plot Data (Summary Statistics by Education Level):")
print(education_hours)

# Create the box plot
plt.figure(figsize=(12, 6))
sns.boxplot(x='education', y='hours-per-week', data=df)
plt.xticks(rotation=90)
plt.title('Box Plot of Hours per Week by Education Level')
plt.show()
```

```
Box Plot Data (Summary Statistics by Education Level):
                count       mean        std  min    25%   50%    75%
max
education

10th           1389.0  36.986321  13.918007  1.0  30.00  40.0  40.00
```

```
99.0
11th             1812.0  33.952539  13.996803   1.0  20.00  40.0  40.00
99.0
12th              657.0  35.374429  12.620268   4.0  25.00  40.0  40.00
99.0
1st-4th           247.0  38.761134  12.226671   4.0  35.00  40.0  40.00
96.0
5th-6th           509.0  38.923379  11.372194   3.0  40.00  40.0  40.00
99.0
7th-8th           955.0  39.003141  14.562775   2.0  35.00  40.0  40.00
99.0
9th               756.0  38.359788  11.464783   1.0  36.75  40.0  40.00
99.0
Assoc-acdm       1601.0  40.809494  12.199101   1.0  40.00  40.0  45.00
99.0
Assoc-voc        2061.0  41.658418  10.943312   1.0  40.00  40.0  45.00
99.0
Bachelors        8025.0  42.482492  11.423058   1.0  40.00  40.0  50.00
99.0
Doctorate         594.0  46.582492  14.919597   1.0  40.00  45.0  55.00
99.0
HS-grad         15784.0  40.640775  11.423842   1.0  40.00  40.0  43.00
99.0
Masters          2657.0  43.575837  12.140942   1.0  40.00  40.0  50.00
99.0
Preschool          83.0  36.566265  11.434002  10.0  30.00  40.0  40.00
75.0
Prof-school       834.0  47.579137  14.983435   2.0  40.00  48.0  55.00
99.0
Some-college    10878.0  38.865784  12.796180   1.0  35.00  40.0  42.75
99.0
```



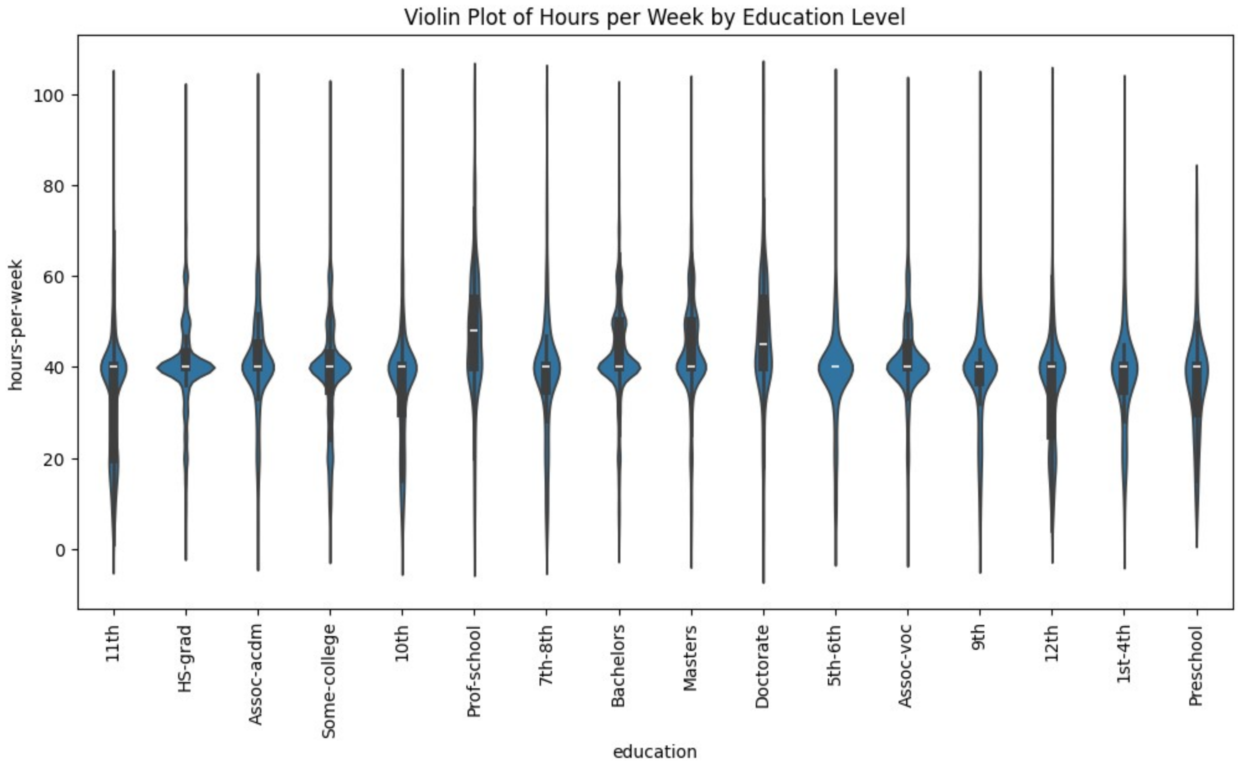
Analysis For Box and Whisker Plot

Higher Education Levels: Individuals with Doctorate and Masters degrees tend to work slightly more hours per week on average, with Doctorate having the highest mean at 46.6 hours. Education and Work Hours: Assoc-acdm and Assoc-voc also show higher average work hours compared to other education levels, while Preschool has the lowest average work hours. Consistency: Most education levels have a median of 40 hours per week, suggesting that a standard work week is common across various education levels.

```
# Violin Plot

# Group and aggregate data for the violin plot
education_hours_violin = df.groupby('education')['hours-per-week'].describe()

# Print the aggregated data
print("Violin Plot Data (Summary Statistics by Education Level):")
print(education_hours_violin)

# Create the violin plot
plt.figure(figsize=(12, 6))
sns.violinplot(x='education', y='hours-per-week', data=df)
plt.xticks(rotation=90)
```

```
plt.title('Violin Plot of Hours per Week by Education Level')
plt.show()
```

```
Violin Plot Data (Summary Statistics by Education Level):
                 count       mean        std   min    25%   50%    75%
max
education

10th            1389.0  36.986321  13.918007   1.0  30.00  40.0  40.00
99.0
11th            1812.0  33.952539  13.996803   1.0  20.00  40.0  40.00
99.0
12th             657.0  35.374429  12.620268   4.0  25.00  40.0  40.00
99.0
1st-4th          247.0  38.761134  12.226671   4.0  35.00  40.0  40.00
96.0
5th-6th          509.0  38.923379  11.372194   3.0  40.00  40.0  40.00
99.0
7th-8th          955.0  39.003141  14.562775   2.0  35.00  40.0  40.00
99.0
9th              756.0  38.359788  11.464783   1.0  36.75  40.0  40.00
99.0
Assoc-acdm      1601.0  40.809494  12.199101   1.0  40.00  40.0  45.00
99.0
Assoc-voc       2061.0  41.658418  10.943312   1.0  40.00  40.0  45.00
99.0
Bachelors       8025.0  42.482492  11.423058   1.0  40.00  40.0  50.00
99.0
Doctorate        594.0  46.582492  14.919597   1.0  40.00  45.0  55.00
99.0
HS-grad        15784.0  40.640775  11.423842   1.0  40.00  40.0  43.00
99.0
Masters         2657.0  43.575837  12.140942   1.0  40.00  40.0  50.00
99.0
Preschool         83.0  36.566265  11.434002  10.0  30.00  40.0  40.00
75.0
Prof-school      834.0  47.579137  14.983435   2.0  40.00  48.0  55.00
99.0
Some-college   10878.0  38.865784  12.796180   1.0  35.00  40.0  42.75
99.0
```



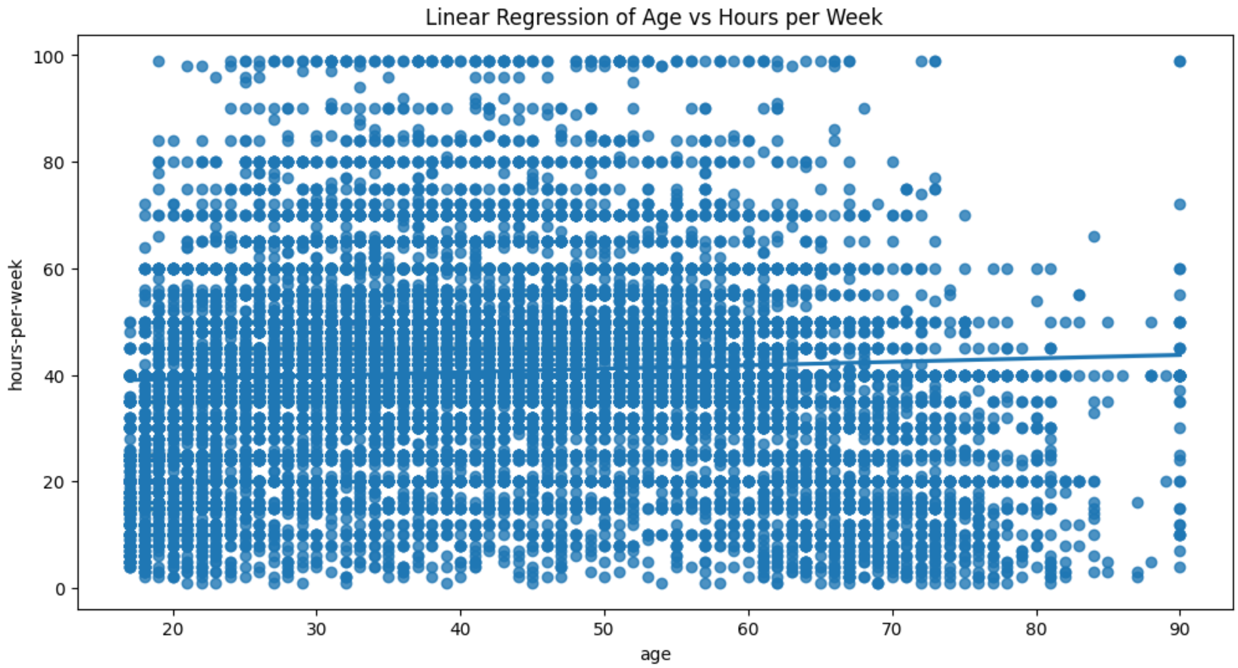
Analysis Of Violin Plot

The violin plot reveals that individuals with higher education levels, such as Doctorate and Masters, tend to work more hours per week on average, with Doctorate showing the highest mean hours. In contrast, those with lower education levels, like Preschool and 10th, generally work fewer hours. The distributions across education levels are relatively similar, with most groups showing a peak around 40 hours per week, indicating a common standard workweek across different educational backgrounds.

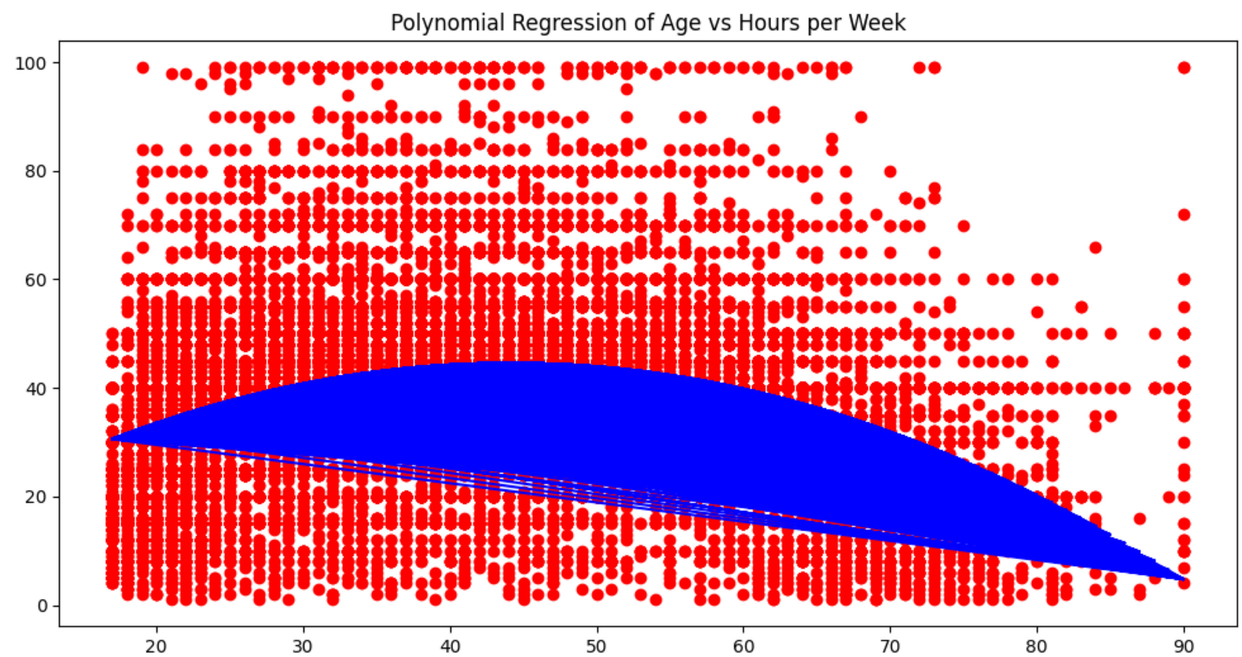
```
# Regression Plot (Linear and Non-Linear)

from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression

# Print data for Linear Regression
print("Data for Linear Regression:")
print(df[['age', 'hours-per-week']].describe())

# Linear Regression Plot
plt.figure(figsize=(12, 6))
sns.regplot(x='age', y='hours-per-week', data=df)
plt.title('Linear Regression of Age vs Hours per Week')
plt.show()
```

```python
# Polynomial Regression Data Preparation
poly = PolynomialFeatures(degree=2)
X_poly = poly.fit_transform(df[['age']])
poly_reg = LinearRegression()
poly_reg.fit(X_poly, df['hours-per-week'])

# Print data used for Polynomial Regression
print("\nData for Polynomial Regression:")
print("Polynomial features (first few rows):")
print(X_poly[:5])
print("\nPredictions (first few rows):")
print(poly_reg.predict(X_poly)[:5])

# Polynomial Regression Plot
plt.figure(figsize=(12, 6))
plt.scatter(df['age'], df['hours-per-week'], color='red')
plt.plot(df['age'], poly_reg.predict(X_poly), color='blue')
plt.title('Polynomial Regression of Age vs Hours per Week')
plt.show()
```

```
Data for Linear Regression:
                age  hours-per-week
count  48842.000000    48842.000000
mean      38.643585       40.422382
std       13.710510       12.391444
min       17.000000        1.000000
25%       28.000000       40.000000
50%       37.000000       40.000000
75%       48.000000       45.000000
max       90.000000       99.000000
```




```
Data for Polynomial Regression:
Polynomial features (first few rows):
[[1.000e+00 2.500e+01 6.250e+02]
 [1.000e+00 3.800e+01 1.444e+03]
 [1.000e+00 2.800e+01 7.840e+02]
 [1.000e+00 4.400e+01 1.936e+03]
 [1.000e+00 1.800e+01 3.240e+02]]

Predictions (first few rows):
[37.59888179 43.84055745 39.60756583 44.56180091 31.58592522]
```



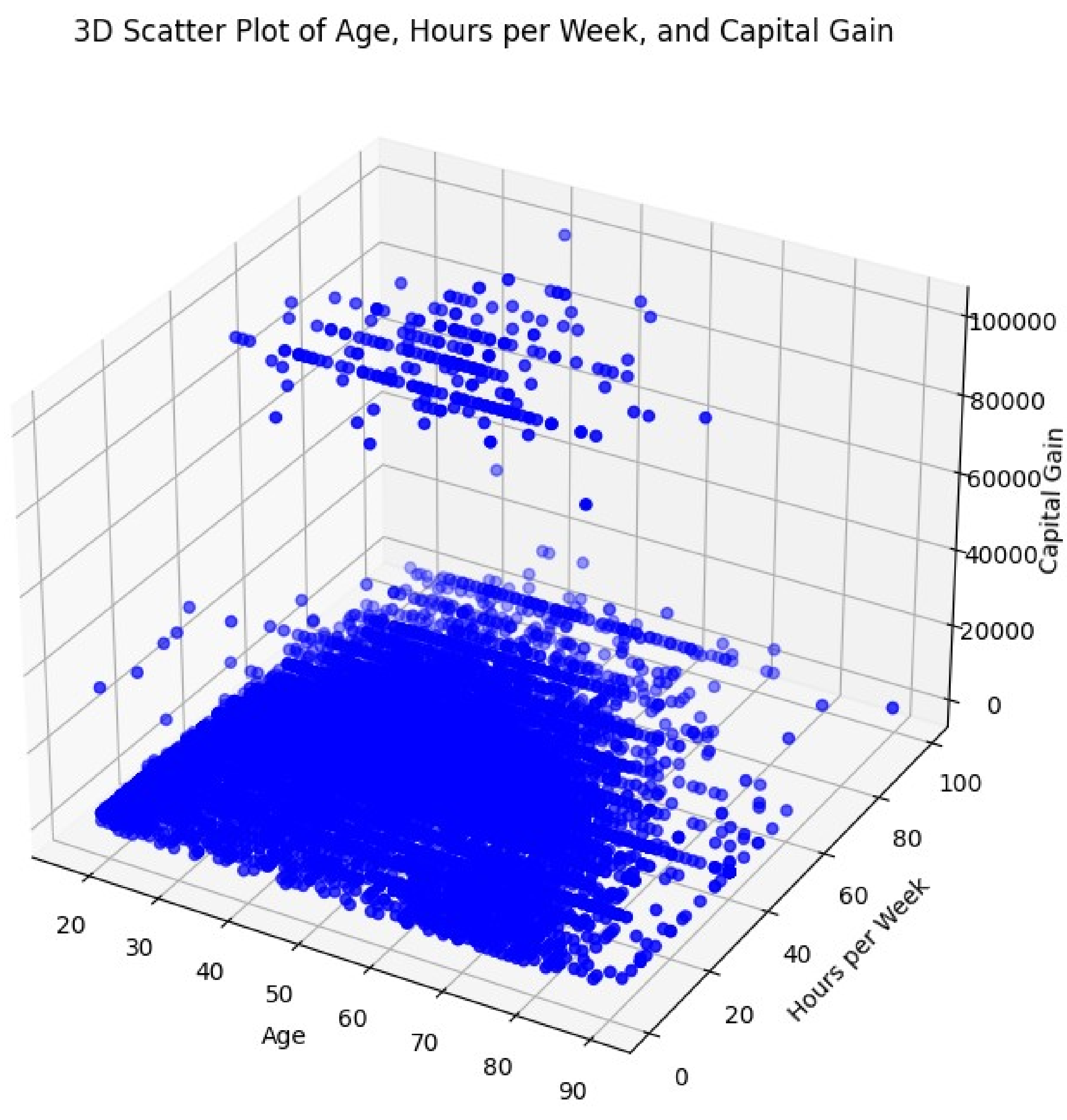
Analysis For Regression Plot (Linear and Non-Linear)

The linear regression plot indicates a moderate positive relationship between age and hours worked per week, with older individuals tending to work slightly more hours. However, the polynomial regression plot reveals a more nuanced pattern, with hours worked showing variability at different ages, suggesting that the relationship between age and work hours is more complex than a simple linear trend. The polynomial regression captures this non-linearity, highlighting that the impact of age on work hours varies rather than increasing uniformly.

```
# 3D Scatter Plot

from mpl_toolkits.mplot3d import Axes3D

# Print data for 3D Scatter Plot
print("Data for 3D Scatter Plot:")
print(df[['age', 'hours-per-week', 'capital-gain']].describe())

# Create 3D Scatter Plot
fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, projection='3d')

ax.scatter(df['age'], df['hours-per-week'], df['capital-gain'], c='b',
marker='o')
ax.set_xlabel('Age')
ax.set_ylabel('Hours per Week')
```

```
ax.set_zlabel('Capital Gain')
plt.title('3D Scatter Plot of Age, Hours per Week, and Capital Gain')
plt.show()
```

```
Data for 3D Scatter Plot:
                age  hours-per-week  capital-gain
count  48842.000000    48842.000000  48842.000000
mean      38.643585       40.422382   1079.067626
std       13.710510       12.391444   7452.019058
min       17.000000        1.000000      0.000000
25%       28.000000       40.000000      0.000000
50%       37.000000       40.000000      0.000000
75%       48.000000       45.000000      0.000000
max       90.000000       99.000000  99999.000000
```



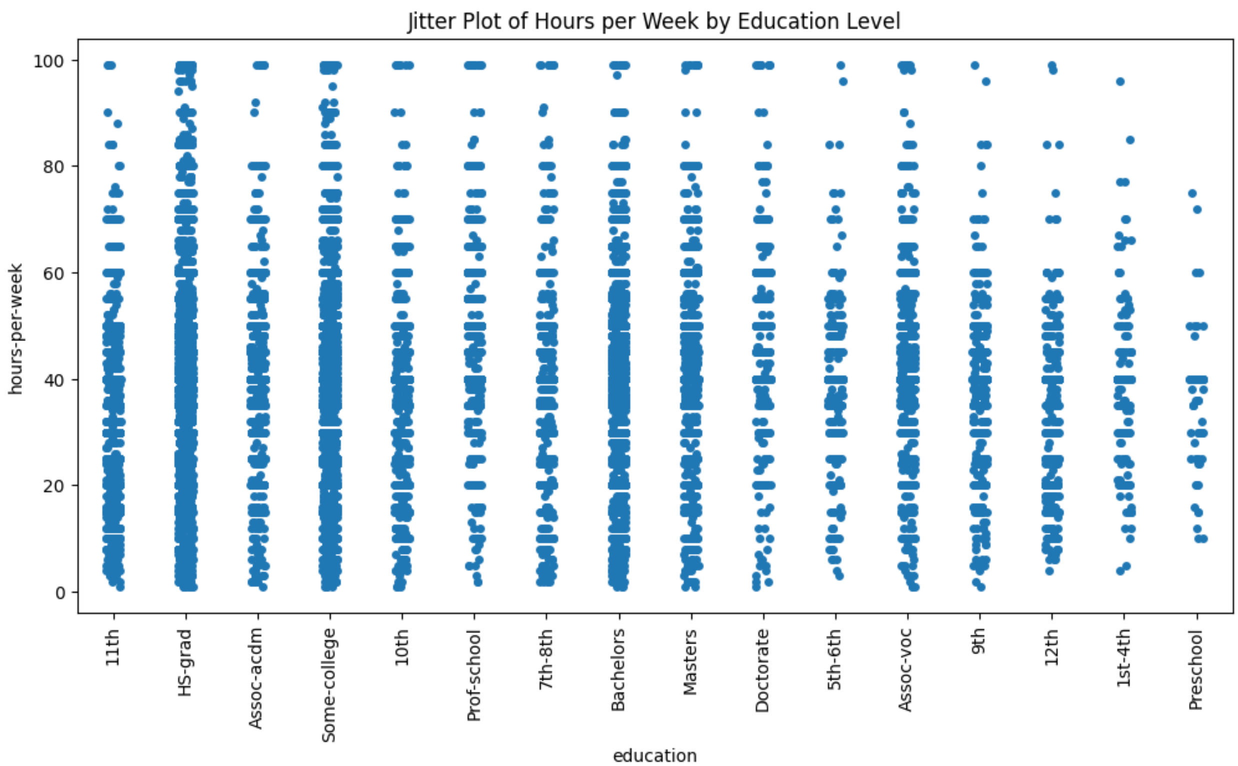
Analysis Of 3D Chart

The 3D scatter plot reveals that most individuals are concentrated in the lower ranges of capital gain, with a mean of around 1079. However, there is a high degree of variability, as evidenced by the maximum capital gain reaching nearly 100,000 despite a narrow range of hours worked and ages. This indicates that while age and hours per week have some impact, capital gain is influenced by other factors, potentially including job role, industry, or investment decisions.

```
# Jitter Plot
```

```python
# Print data for Jitter Plot
print("Data for Jitter Plot:")
print(df[['education', 'hours-per-week']].describe())

# Jitter Plot
plt.figure(figsize=(12, 6))
sns.stripplot(x='education', y='hours-per-week', data=df, jitter=True)
plt.xticks(rotation=90)
plt.title('Jitter Plot of Hours per Week by Education Level')
plt.show()
```

```
Data for Jitter Plot:
       hours-per-week
count    48842.000000
mean        40.422382
std         12.391444
min          1.000000
25%         40.000000
50%         40.000000
75%         45.000000
max         99.000000
```

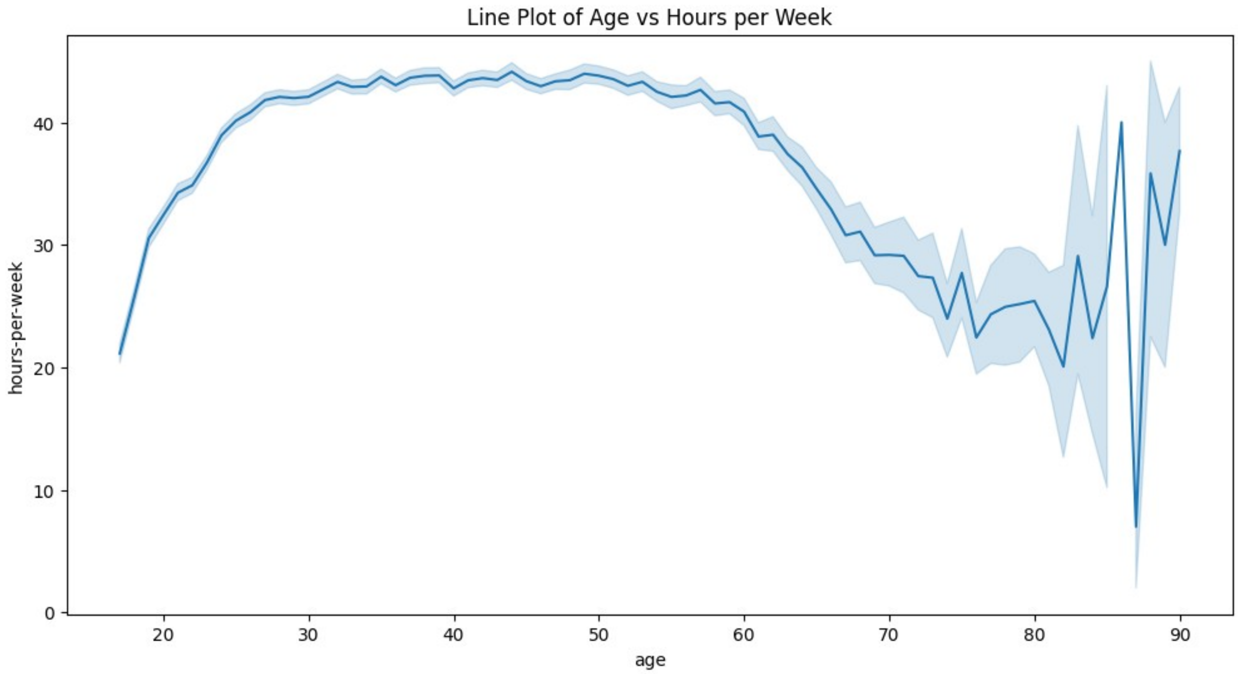


Analysis Of Jitter Plot

The Jitter Plot data reveals that most individuals work around 40 hours per week, with a standard deviation indicating some variability. The majority work between 40 and 45 hours per week, while a small number work significantly fewer or more hours. This distribution shows a concentration of work hours around the standard 40-hour workweek, with occasional extremes.

```python
# Line Plot

# Print data for Line Plot
print("Data for Line Plot:")
print(df[['age', 'hours-per-week']].describe())

# Line Plot
plt.figure(figsize=(12, 6))
sns.lineplot(x='age', y='hours-per-week', data=df)
plt.title('Line Plot of Age vs Hours per Week')
plt.show()
```

```
Data for Line Plot:
                age  hours-per-week
count  48842.000000    48842.000000
mean      38.643585       40.422382
std       13.710510       12.391444
min       17.000000        1.000000
25%       28.000000       40.000000
50%       37.000000       40.000000
75%       48.000000       45.000000
max       90.000000       99.000000
```

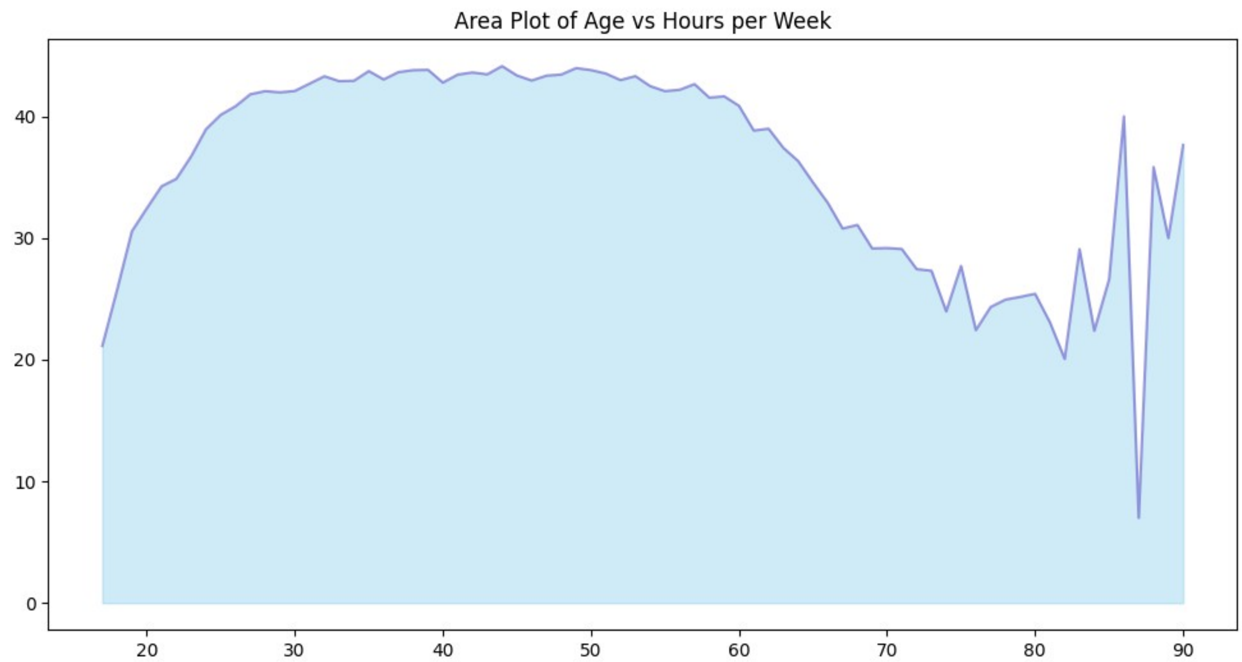


Analysis For Line Plot

The line plot depicts the relationship between age and hours worked per week. The data shows that the average hours worked per week remains relatively stable across different ages, with a mean of around 40.42 hours. Although the standard deviation is significant, suggesting variability, there is a general trend where the number of hours worked peaks slightly around middle age, reflecting consistent work hours across the age range.

```
# Area Plot

# Grouping data for area plot
df_grouped = df.groupby('age')['hours-per-week'].mean().reset_index()

# Print the grouped data
print("Data for Area Plot:")
print(df_grouped)

# Create area plot
plt.figure(figsize=(12, 6))
plt.fill_between(df_grouped['age'], df_grouped['hours-per-week'],
color="skyblue", alpha=0.4)
plt.plot(df_grouped['age'], df_grouped['hours-per-week'],
color="Slateblue", alpha=0.6)
plt.title('Area Plot of Age vs Hours per Week')
plt.show()
```

```
Data for Area Plot:
    age  hours-per-week
0    17       21.137815
1    18       25.745940
2    19       30.560304
3    20       32.432165
4    21       34.252737
..  ...             ...
69   86       40.000000
70   87        7.000000
71   88       35.833333
72   89       30.000000
73   90       37.654545

[74 rows x 2 columns]
```



Analysis For Area Plot

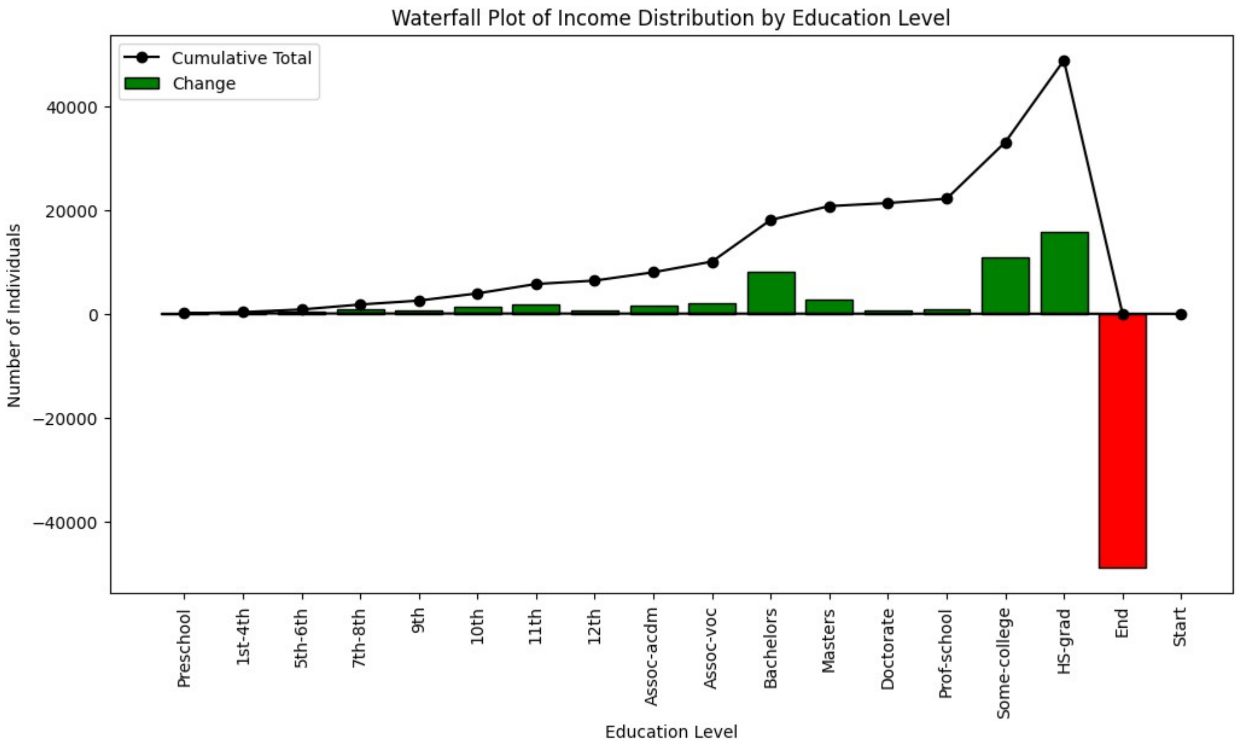
The area plot provides a visualization of the average hours worked per week across different ages. The data shows that average hours per week generally increase with age, reaching a peak around the mid-30s and then fluctuating with some decrease in older age groups. This indicates that work patterns may vary significantly with age, possibly due to changes in career stages or retirement.

```
# Waterfall Chart

# Define categories and values for the waterfall plot
```

```python
education_counts = df['education'].value_counts().sort_index()
education_levels = ['Preschool', '1st-4th', '5th-6th', '7th-8th',
'9th', '10th', '11th', '12th', 'Assoc-acdm', 'Assoc-voc', 'Bachelors',
'Masters', 'Doctorate', 'Prof-school', 'Some-college', 'HS-grad']
values = [education_counts.get(level, 0) for level in
education_levels]

# Create a waterfall plot data
categories = ['Start'] + education_levels + ['End']
cumulative_values = np.zeros(len(categories))
cumulative_values[0] = 0  # Starting point

# Compute cumulative values for the waterfall plot
for i, value in enumerate(values, 1):
    cumulative_values[i] = cumulative_values[i-1] + value

# Define colors for each segment
colors = ['lightgray' if x == 0 else 'green' if x > 0 else 'red' for x
in np.diff(cumulative_values)]

# Print data for Waterfall Plot
print("Data for Waterfall Plot:")
for category, value in zip(categories, cumulative_values):
    print(f"{category}: {value}")

# Create the waterfall plot
plt.figure(figsize=(12, 6))
plt.bar(categories[1:], np.diff(cumulative_values), color=colors,
edgecolor='black', label='Change')
plt.plot(categories, cumulative_values, marker='o', color='black',
label='Cumulative Total')
plt.title('Waterfall Plot of Income Distribution by Education Level')
plt.xlabel('Education Level')
plt.ylabel('Number of Individuals')
plt.xticks(rotation=90)
plt.legend()
plt.show()
```

```
Data for Waterfall Plot:
Start: 0.0
Preschool: 83.0
1st-4th: 330.0
5th-6th: 839.0
7th-8th: 1794.0
9th: 2550.0
10th: 3939.0
11th: 5751.0
12th: 6408.0
Assoc-acdm: 8009.0
Assoc-voc: 10070.0
```

```
Bachelors: 18095.0
Masters: 20752.0
Doctorate: 21346.0
Prof-school: 22180.0
Some-college: 33058.0
HS-grad: 48842.0
End: 0.0
```



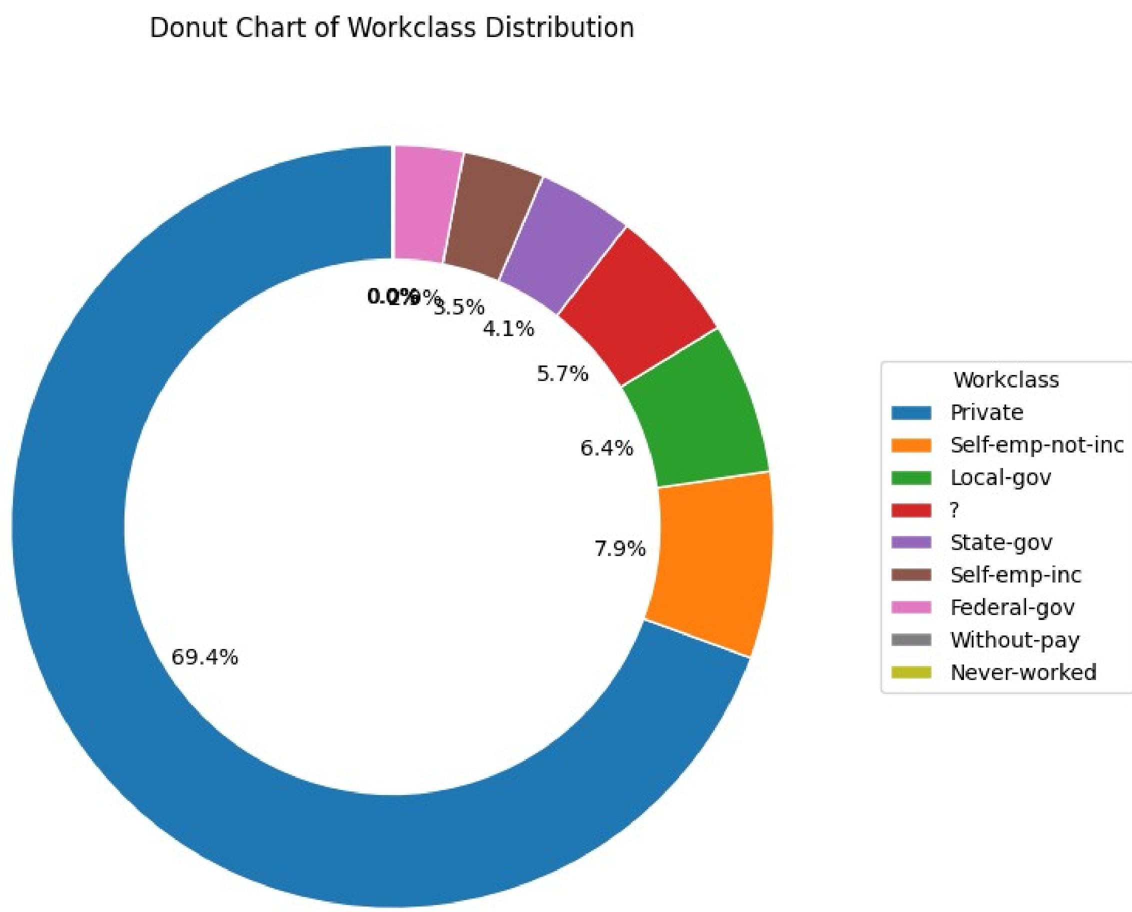
Analysis For Waterfall Chart

The waterfall plot illustrates a progressive accumulation of individuals across various education levels. Starting from zero, the chart shows a steady increase in counts as each education level is added, culminating in the highest count at the 'HS-grad' level with 48,842 individuals. This visualization effectively highlights the significant growth in the number of individuals as they advance through higher education levels, with the largest increments observed between the 'Some-college' and 'HS-grad' categories.

```
# Donut Chart

sizes = df['workclass'].value_counts()
labels = sizes.index

print("Data for Donut Chart:")
print(sizes)
```

```python
# Donut Chart
plt.figure(figsize=(10, 8))
wedges, texts, autotexts = plt.pie(sizes, labels=None, autopct='%1.1f%
%', startangle=90, wedgeprops=dict(width=0.3, edgecolor='w'))

# Adding legend to avoid overlapping labels
plt.legend(wedges, labels, title="Workclass", loc="center left",
bbox_to_anchor=(1, 0, 0.5, 1))

plt.title('Donut Chart of Workclass Distribution')
plt.show()
```

```
Data for Donut Chart:
workclass
Private             33906
Self-emp-not-inc     3862
Local-gov            3136
?                    2799
State-gov            1981
Self-emp-inc         1695
Federal-gov          1432
Without-pay            21
Never-worked           10
Name: count, dtype: int64
```



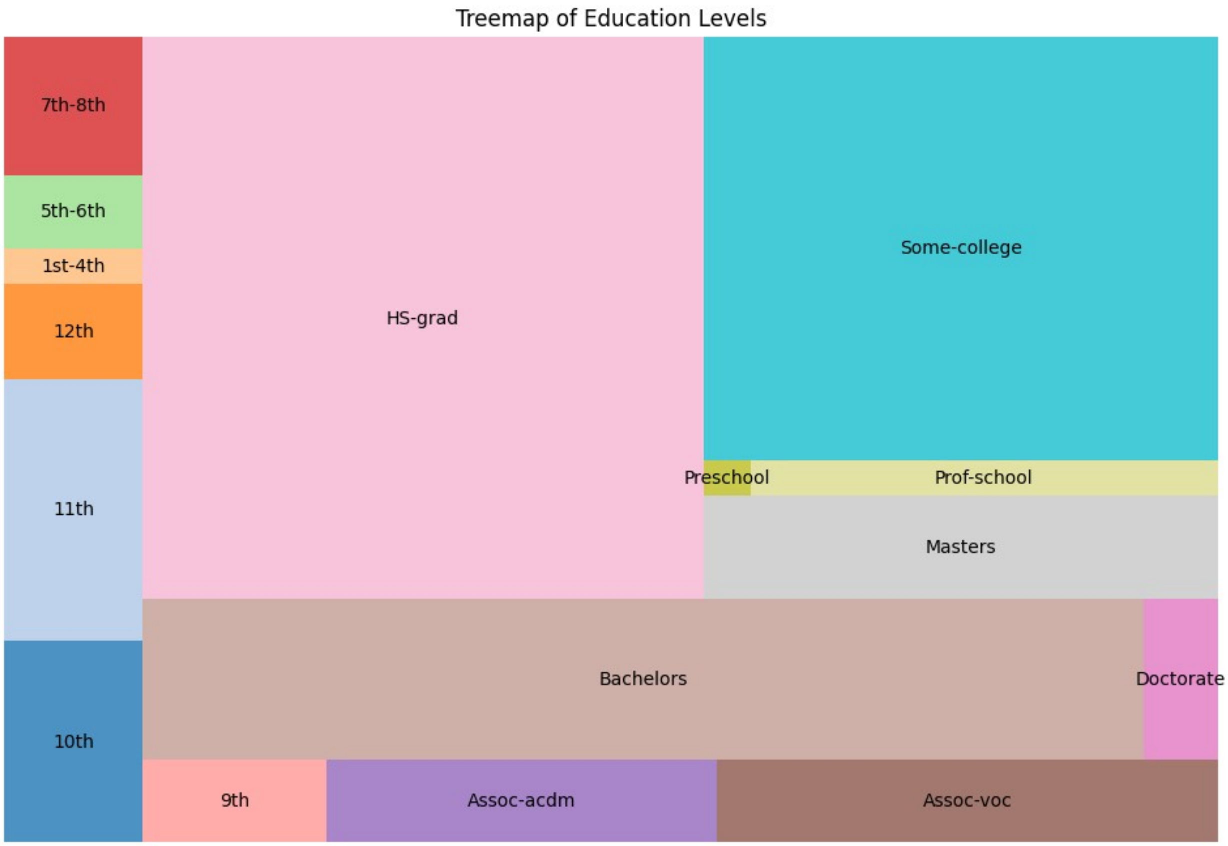
Analysis Of Donut Chart

The Donut Chart for workclass distribution highlights that "Private" is the dominant category, making up a significant majority at 33906 counts. The smaller segments, such as "Self-emp-not-inc" and "Local-gov," have notably fewer counts, with the least frequent categories being "Without-pay" and "Never-worked," each having fewer than 25 counts. The chart effectively illustrates the disparity in representation among different workclasses, with a clear concentration in the "Private" sector.

```
!pip install squarify

Collecting squarify
  Downloading squarify-0.4.4-py3-none-any.whl.metadata (600 bytes)
Downloading squarify-0.4.4-py3-none-any.whl (4.1 kB)
Installing collected packages: squarify
Successfully installed squarify-0.4.4
```

```python
# Treemap Chart

import squarify

# Group by education level and aggregate counts
df_grouped = df.groupby('education').size().reset_index(name='count')

# Print data for Treemap
print("Data for Treemap:")
print(df_grouped)

# Treemap Plotting
sizes = df_grouped['count']
labels = df_grouped['education']

# Create a color map
cmap = plt.get_cmap('tab20')
colors = [cmap(i / len(labels)) for i in range(len(labels))]

plt.figure(figsize=(12, 8))
squarify.plot(sizes=sizes, label=labels, color=colors, alpha=.8)
plt.title('Treemap of Education Levels')
plt.axis('off')
plt.show()
```

```
Data for Treemap:
       education  count
0           10th   1389
1           11th   1812
2           12th    657
3        1st-4th    247
4        5th-6th    509
5        7th-8th    955
6            9th    756
7     Assoc-acdm   1601
8      Assoc-voc   2061
9      Bachelors   8025
10     Doctorate    594
11       HS-grad  15784
12       Masters   2657
13     Preschool     83
14   Prof-school    834
15  Some-college  10878
```



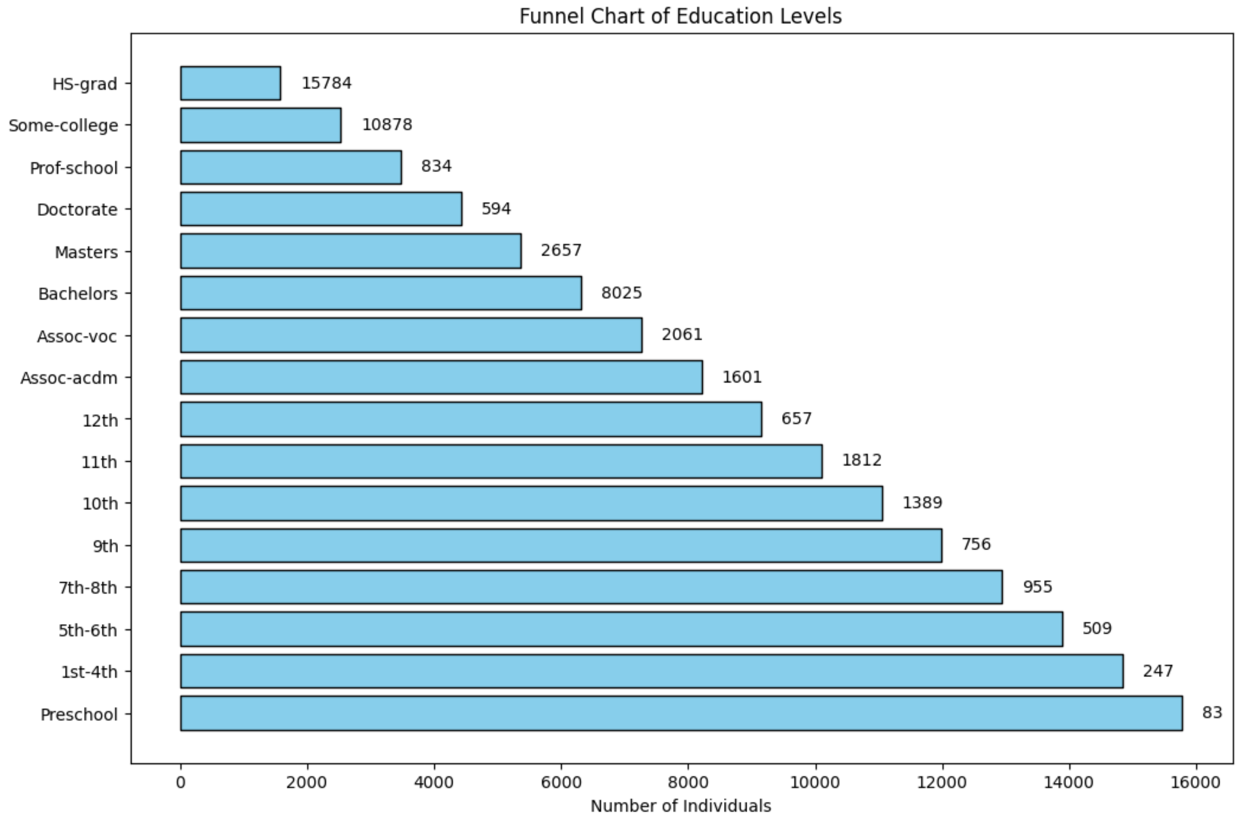
Data Analysis For Treemap

The data shows the distribution of education levels among the dataset. The `HS-grad` category has the highest count of 15,784, indicating it is the most common education level in the dataset. In contrast, the `Preschool` category has the lowest count of 83, reflecting its rarity. The `Bachelors` and `Some-college` categories also have significant counts, suggesting that higher education levels are relatively common. The treemap will visually represent these distributions, highlighting the proportion of each education level effectively.

```
# Funnel Chart

# Data for the funnel chart
education_counts = df['education'].value_counts().sort_index()
education_levels = ['Preschool', '1st-4th', '5th-6th', '7th-8th',
'9th', '10th', '11th', '12th', 'Assoc-acdm', 'Assoc-voc', 'Bachelors',
'Masters', 'Doctorate', 'Prof-school', 'Some-college', 'HS-grad']
values = [education_counts.get(level, 0) for level in
education_levels]

# Print the data
print("Data for Funnel Chart:")
for level, value in zip(education_levels, values):
```

```
    print(f"{level}: {value}")

# Adjust the values to create a funnel effect
max_value = max(values)
funnel_widths = np.linspace(1, 0.1, len(values)) * max_value

# Plot the funnel chart
plt.figure(figsize=(12, 8))
for i in range(len(values)):
    plt.fill_betweenx([i-0.4, i+0.4], 0, funnel_widths[i],
color='skyblue', edgecolor='black')
    plt.text(funnel_widths[i] + max_value*0.02, i, f'{values[i]}',
va='center')

plt.yticks(range(len(values)), education_levels)
plt.title('Funnel Chart of Education Levels')
plt.xlabel('Number of Individuals')
plt.show()
```

```
Data for Funnel Chart:
Preschool: 83
1st-4th: 247
5th-6th: 509
7th-8th: 955
9th: 756
10th: 1389
11th: 1812
12th: 657
Assoc-acdm: 1601
Assoc-voc: 2061
Bachelors: 8025
Masters: 2657
Doctorate: 594
Prof-school: 834
Some-college: 10878
HS-grad: 15784
```



Analysis Of Funnel Chart

The funnel chart reveals a substantial drop in the number of individuals as educational attainment increases. Starting with a large base at the "HS-grad" level (15,784 individuals), the numbers decrease progressively through each education level. "Some-college" has the next highest count (10,878), followed by "Bachelors" (8,025) and "Masters" (2,657). This trend reflects the increasing rarity of higher educational qualifications, with the lowest numbers at the "Preschool" level (83) and "Doctorate" (594), illustrating the narrowing funnel effect.