In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
/home/vishal/anaconda3/lib/python3.8/site-packages/pandas/core/computation/expressions.py:20: UserWarning: Pandas
requires version '2.7.3' or newer of 'numexpr' (version '2.7.1' currently installed).
  from pandas.core.computation.check import NUMEXPR_INSTALLED
```

In [2]:
```
with open('adult.data' , 'r') as f2:
    data = f2.readlines()
```

In [3]:
```
lst=[]
cnt=0
for d in data:
    k = d.split(', ')
    k[len(k)-1] = k[len(k)-1].replace("\n","")
    lst.append(k)
```

In [4]:
```
labels = ['age','workclass','fnlwgt','education','education-num',
          'marital-status','occupation','relationship','race',
          'sex','capital-gain','capital-loss','hours-per-week',
          'native-country','Salary(<=50K or >50K)']
```

In [5]:
```
fd = pd.DataFrame(lst , columns = labels)
```

In [6]:
```
fd = fd.replace('?',np.nan)
```

In [7]:
```
def conv_func(value):
    return True if value == '<=50K' else False
```

In [8]:
```
fd['Salary<=50K'] = fd['Salary(<=50K or >50K)'].apply(conv_func)
```

In [9]:
```
Data_M50K = fd[fd['Salary<=50K']== False]
Data_LE50K = fd[fd['Salary<=50K']== True]
```

In [10]:
```
print(len(fd), len(Data_M50K) , len(Data_LE50K))
```

```
32562 7842 24720
```

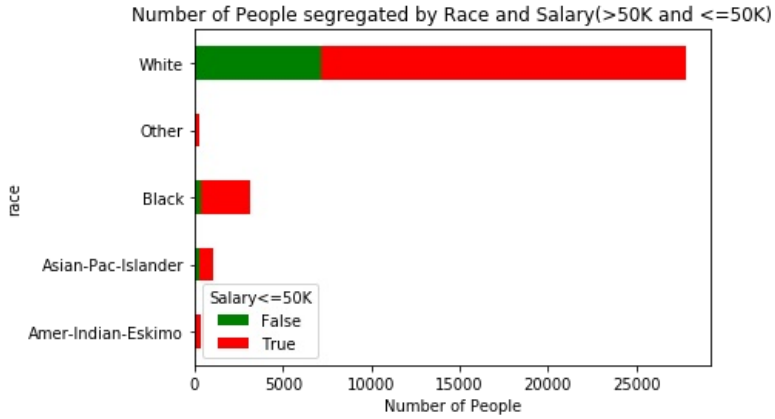
In [11]:
```
from sklearn.utils import resample
und_data = resample(Data_LE50K , replace=False ,
                    n_samples =len(Data_M50K),random_state=42)
fd_r = pd.concat([und_data , Data_M50K])
```

In [12]:
```
import matplotlib.pyplot as plt
import pandas as pd

# Assuming Data_LE50K is your DataFrame
plt.hist(Data_LE50K['age'], bins=20, edgecolor='black')
plt.xlabel('Age')
plt.ylabel('Number of people')
plt.title('Distribution of Age for people with Salary<=50K')
plt.xticks(range(0, 100, 5))
plt.show()
```

In [13]:
```
import matplotlib.pyplot as plt

# Grouping the data and unstacking
```

```
U2_story = fd.groupby(['race', 'Salary<=50K'])['Salary<=50K'].count().unstack()

# Plotting
plt.figure(figsize=(12, 6))
ax = U2_story.plot(kind='barh', stacked=True, color=['green', 'red'])
ax.set_xlabel('Number of People')
ax.set_title('Number of People segregated by Race and Salary(>50K and <=50K)')
#plt.legend(loc = 'lower right')
plt.show()
```

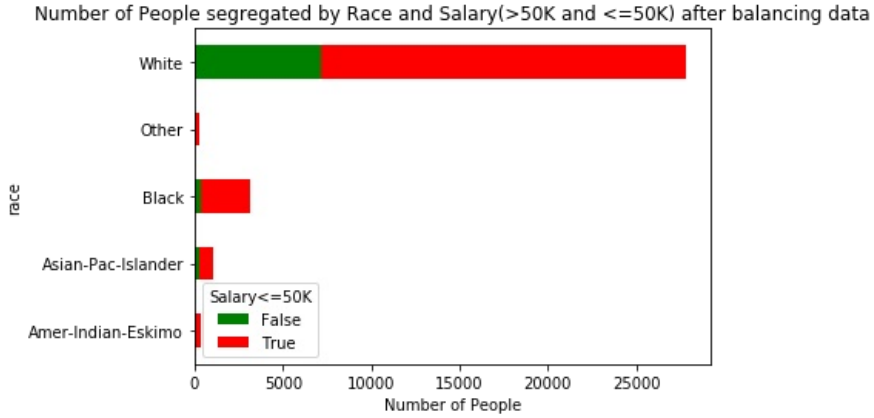
```
<Figure size 864x432 with 0 Axes>
```



In [14]:

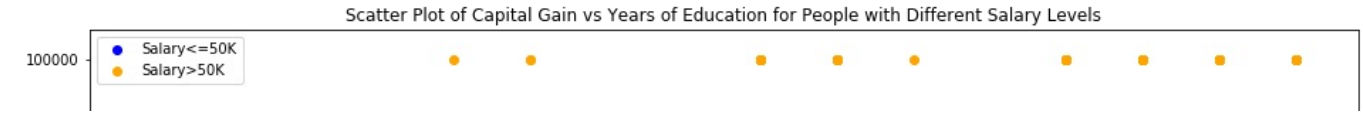
```
import matplotlib.pyplot as plt

# Grouping the data and unstacking
U2_storyb = fd.groupby(['race', 'Salary<=50K'])['Salary<=50K'].count().unstack()

# Plotting
plt.figure(figsize=(12, 6))
ax = U2_storyb.plot(kind='barh', stacked=True, color=['green', 'red'])
ax.set_xlabel('Number of People')
ax.set_title('Number of People segregated by Race and Salary(>50K and <=50K) after balancing data')
#plt.legend(loc = 'lower right')
plt.show()
```

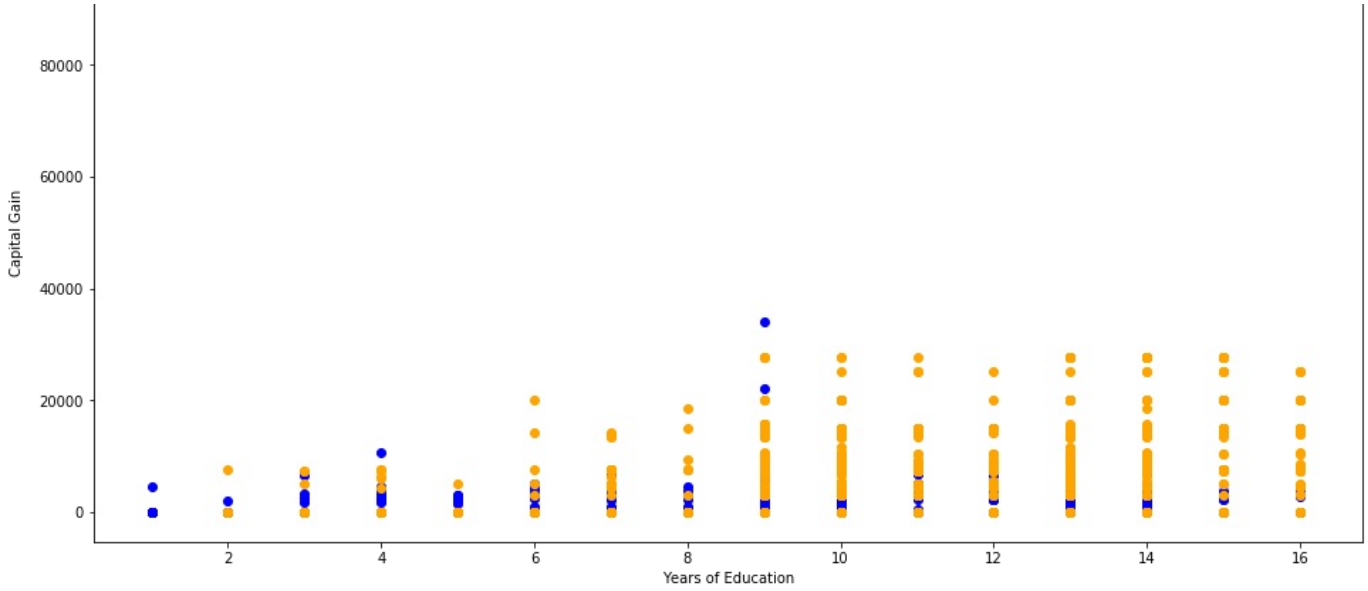
```
<Figure size 864x432 with 0 Axes>
```



In [15]:

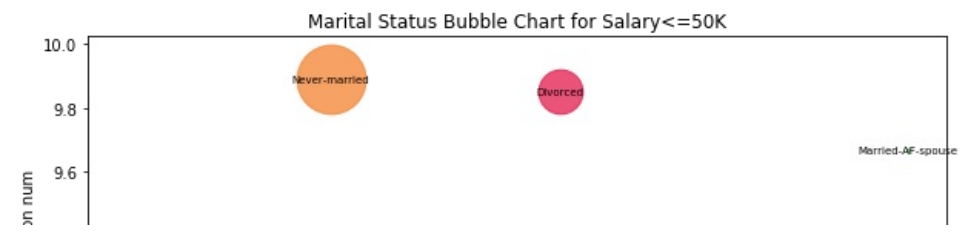
```
import matplotlib.pyplot as plt

# Separate data based on salary
DM50K_r = fd_r[fd_r['Salary<=50K'] == False].dropna().astype({'capital-gain': int, 'education-num': int})
DLE50K_r = fd_r[fd_r['Salary<=50K'] == True].dropna().astype({'capital-gain': int, 'education-num': int})

# Plot scatter plot
plt.figure(figsize=(16,8))
plt.scatter(DLE50K_r['education-num'], DLE50K_r['capital-gain'], color='blue', label='Salary<=50K')
plt.scatter(DM50K_r['education-num'], DM50K_r['capital-gain'], color='orange', label='Salary>50K')
plt.xlabel('Years of Education')
plt.ylabel('Capital Gain')
plt.title('Scatter Plot of Capital Gain vs Years of Education for People with Different Salary Levels')
plt.legend()
plt.show()
```

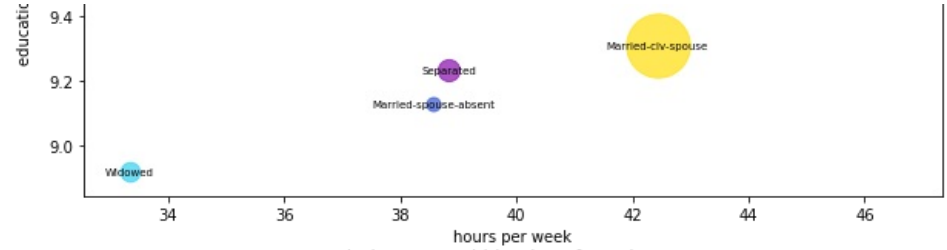


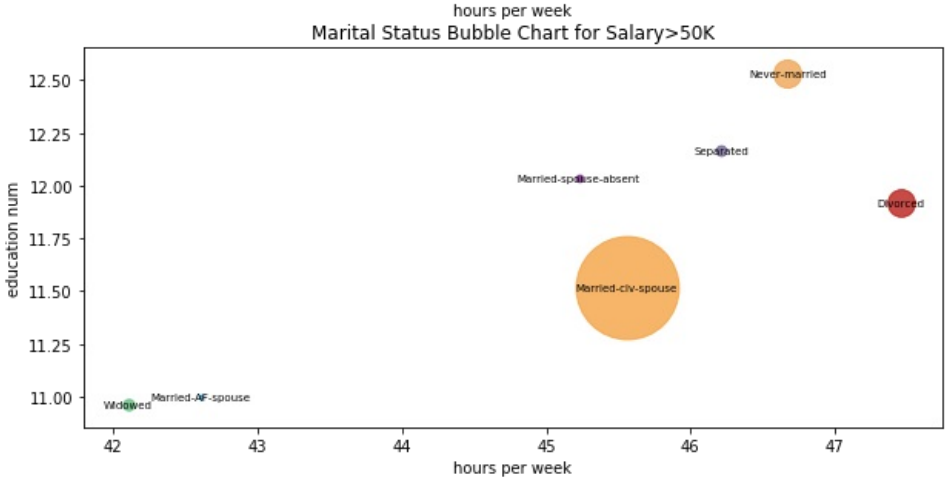


In [16]:
```
import plotly.express as px
U4_data = fd_r[['marital-status','hours-per-week','education-num','Salary(<=50K or >50K)']].dropna(subset=['marit
U4_data.head()
U4_data['hours-per-week'] = U4_data['hours-per-week'].astype(int)
U4_data['education-num'] = U4_data['education-num'].astype(int)
g_d = U4_data.groupby(['marital-status', 'Salary(<=50K or >50K)']).agg({'education-num': 'mean', 'hours-per-week'
g_d.rename(columns={'Salary(<=50K or >50K)': 'count'}, inplace=True)
```

In [17]:
```
import seaborn as sns
import matplotlib.pyplot as plt

def preprocess_data(data):
    data = data[['marital-status', 'hours-per-week', 'education-num']].dropna(subset=['marital-status'])
    data['hours-per-week'] = data['hours-per-week'].astype(int)
    data['education-num'] = data['education-num'].astype(int)
    return data

def group_and_aggregate(data):
    grouped_data = data.groupby(['marital-status']).agg({'education-num': 'mean', 'hours-per-week': 'mean', 'mari
    grouped_data.rename(columns={'marital-status': 'count'}, inplace=True)
    return grouped_data

def plot_bubble_chart(ax, data, colors):
    for index, row in data.iterrows():
        ax.scatter(row['hours-per-week'], row['education-num'], s=row['count']*0.7, color=colors[index], alpha=0.
        ax.text(row['hours-per-week'], row['education-num'], index, ha='center', va='center', fontsize=7)
    ax.set_xlabel('hours per week')
    ax.set_ylabel('education num')

# Data preparation
U4_dt = preprocess_data(DLE50K_r)
U4_dtm = preprocess_data(DM50K_r)

# Grouping and aggregation
g_d = group_and_aggregate(U4_dt)
g_d1 = group_and_aggregate(U4_dtm)

# Define colors
colors = {'Divorced': '#e6194B', 'Married-AF-spouse': '#3cb44b', 'Married-civ-spouse': '#ffe119',
          'Married-spouse-absent': '#4363d8', 'Never-married': '#f58231', 'Separated': '#911eb4', 'Widowed': '#42
colors1 = {'Divorced': '#b80f0a', 'Married-AF-spouse': '#007cbe', 'Married-civ-spouse': '#f49d37',
           'Married-spouse-absent': '#6b1b7f', 'Never-married': '#f0a14b', 'Separated': '#6e5f98', 'Widowed': '#5

# Plotting
fig, (ax, ax1) = plt.subplots(2, 1, figsize=(10, 10))
plot_bubble_chart(ax, g_d, colors)
ax.set_title('Marital Status Bubble Chart for Salary<=50K')

plot_bubble_chart(ax1, g_d1, colors1)
ax1.set_title('Marital Status Bubble Chart for Salary>50K')

plt.show()
```

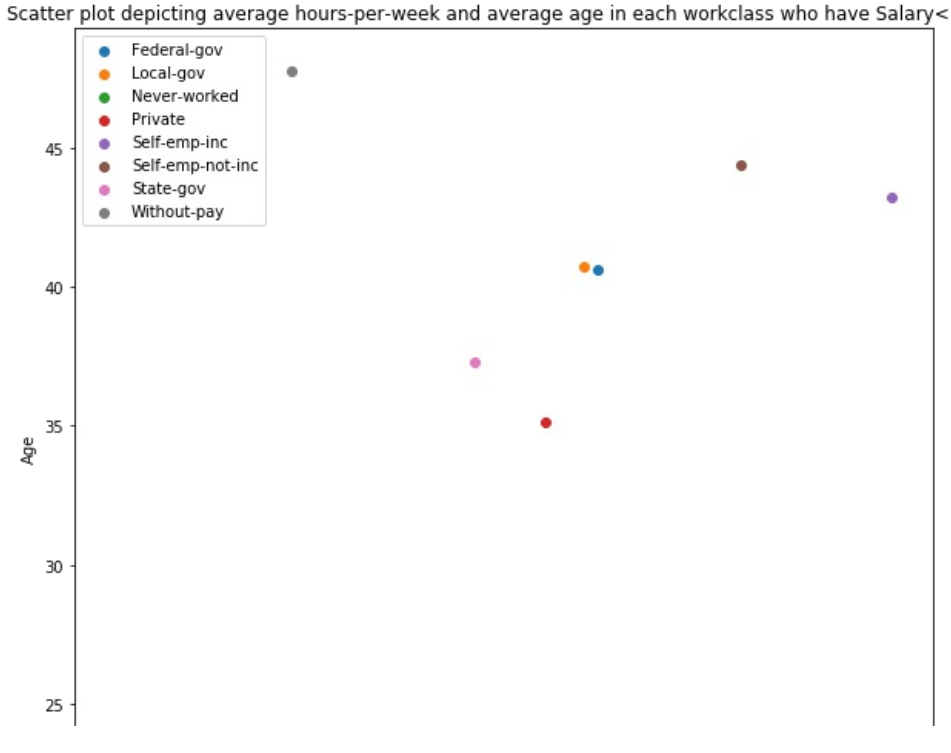


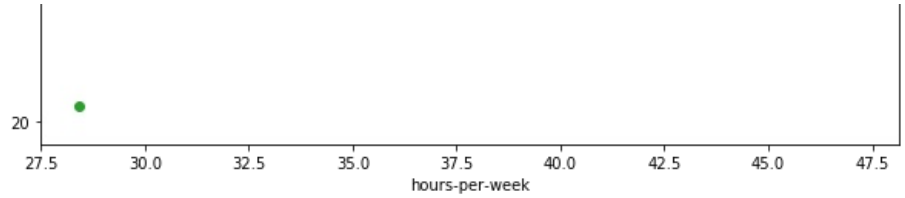




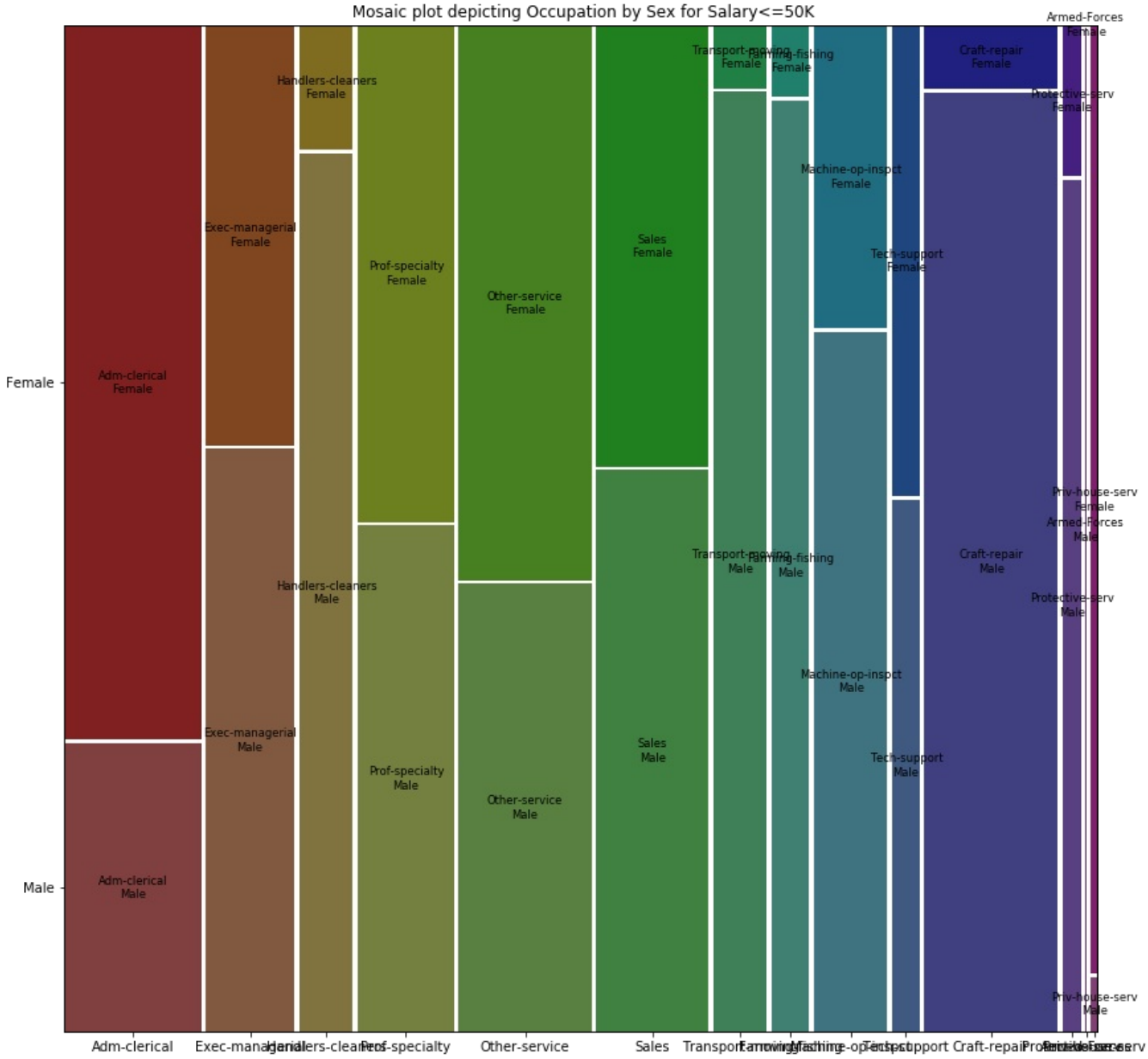
In [18]:

```
import seaborn as sns

U5_dt = Data_LE50K[['Salary(<=50K or >50K)', 'workclass', 'hours-per-week', 'education-num', 'age']].copy()

int_columns = ['hours-per-week', 'education-num', 'age']
U5_dt[int_columns] = U5_dt[int_columns].astype(int)

U5_dt = U5_dt.dropna(subset=['workclass'])

g_5d = U5_dt.groupby(['workclass']).agg({
    'Salary(<=50K or >50K)': 'count',
    'hours-per-week': 'mean',
    'age': 'mean'
})

g_5d.rename(columns={'Salary(<=50K or >50K)': 'count'}, inplace=True)

fig, bx = plt.subplots(figsize=(10, 10))

for indx, rw in g_5d.iterrows():
    bx.scatter(rw['hours-per-week'], rw['age'], label=indx)

plt.legend()
plt.xlabel('hours-per-week')
plt.ylabel('Age')
plt.title('Scatter plot depicting average hours-per-week and average age in each workclass who have Salary<=50K')
plt.show()
```

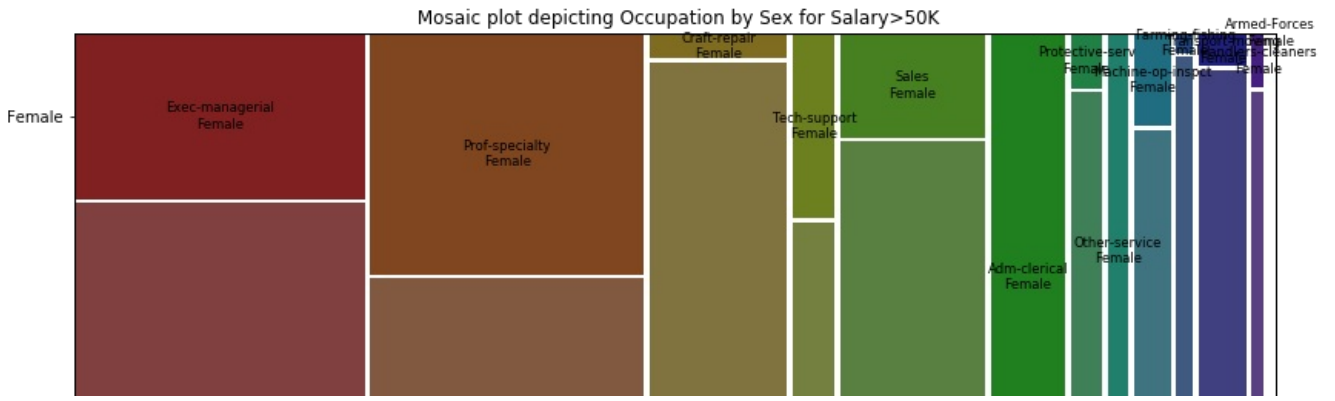


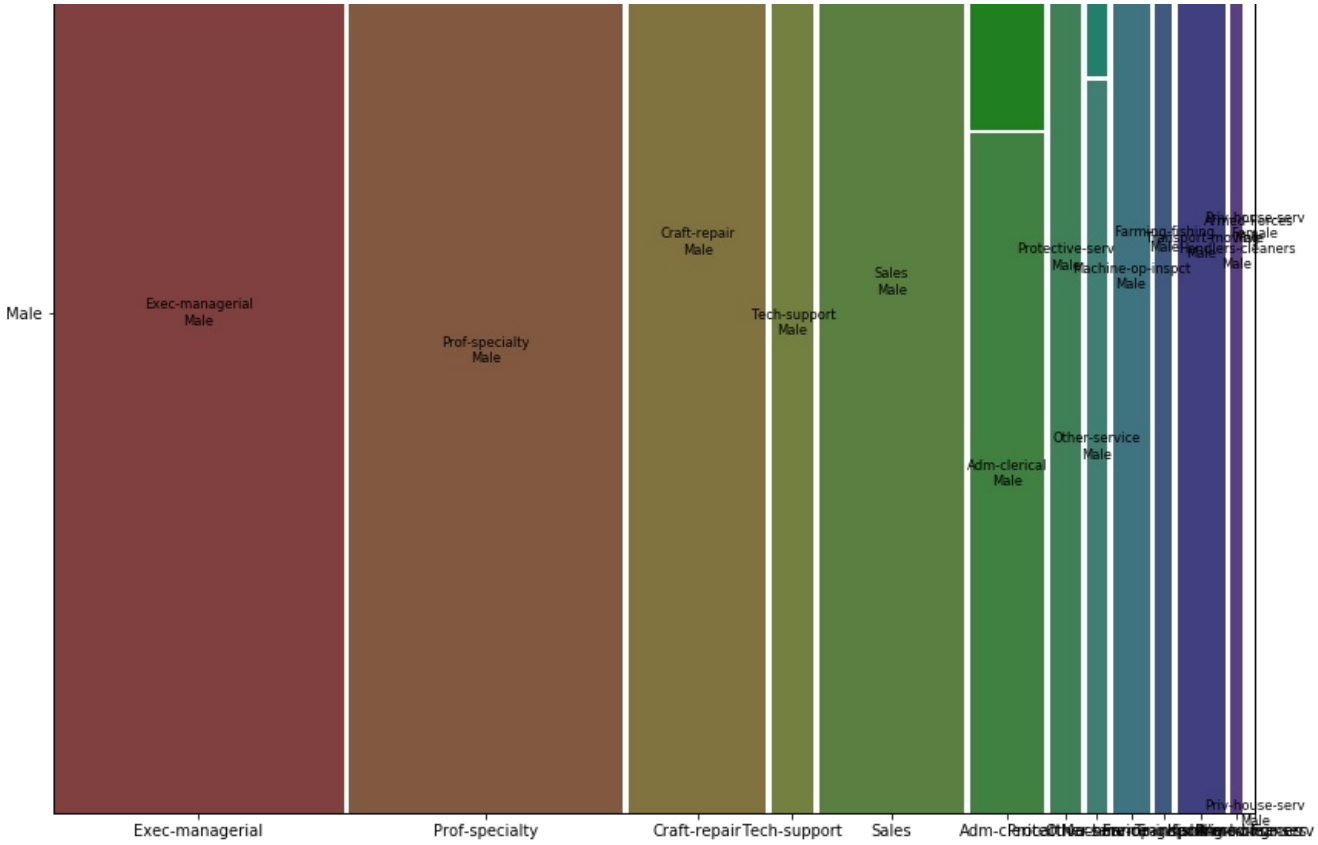


In [19]:
```
from statsmodels.graphics.mosaicplot import mosaic
fig, ax = plt.subplots(figsize=(14, 14))
mosaic(Data_LE50K, ['occupation', 'sex'], title=' Mosaic plot depicting Occupation by Sex for Salary<=50K', ax=ax)
plt.show()
```



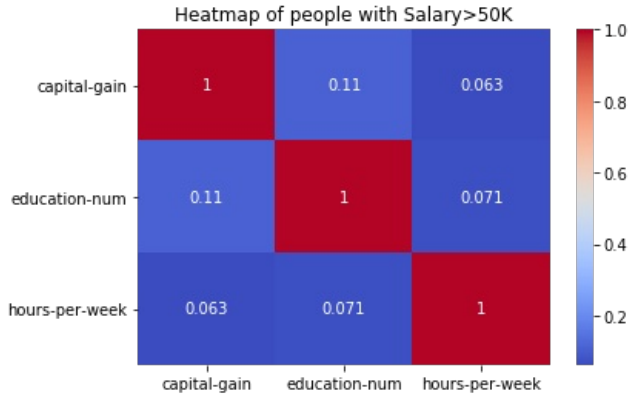
In [20]:
```
fig, ax = plt.subplots(figsize=(14, 14))
mosaic(Data_M50K, ['occupation', 'sex'], title=' Mosaic plot depicting Occupation by Sex for Salary>50K', ax=ax)
plt.show()
```

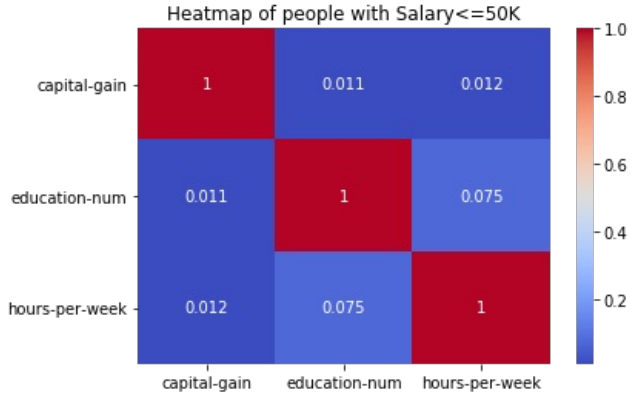




In [21]:

```
import seaborn as sns
import matplotlib.pyplot as plt

def generate_heatmap(data, title):
    data['hours-per-week'] = data['hours-per-week'].astype(int)
    sns.heatmap(data[['capital-gain','education-num','hours-per-week']].corr(), annot=True , cmap='coolwarm')
    plt.title(title)
    plt.show()

generate_heatmap(DM50K_r, 'Heatmap of people with Salary>50K')
generate_heatmap(DLE50K_r, 'Heatmap of people with Salary<=50K')
```





In [ ]: