T-test one-way ANOVA_ p-value concepts

In [1]:

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

In [2]:

```
from scipy import stats
import statistics as stat
```

In [3]:

```
edu_p_school=[1200, 1400, 1600, 1800, 2000, 2200, 2400]
edu_h_school=[400, 500, 600, 650,700,750, 800, 900, 1000]
edu_p_college=[1100, 1300, 1500, 1700, 2900]
edu_Bechelore=[100, 200, 300, 400, 600, 1000, 1800]
edu_Graduate=[2500, 2600, 2700, 280, 2900, 3000, 3100]

print('p_school:'  +str(int(stat.mean(edu_p_school))))
print('h_school:'  +str(int(stat.mean(edu_h_school))))
print('p_college:'  +str(int(stat.mean(edu_p_college))))
print('Bechelore:'  +str(int(stat.mean(edu_Bechelore))))
print('p_Graduate:'  +str(int(stat.mean(edu_Graduate))))
```

```
p_school:1800
h_school:700
p_college:1700
Bechelore:628
p_Graduate:2440
```

In [4]:

```
t, p= stats.ttest_ind(edu_p_school, edu_p_college)

print(f't_value: {t}')
print(f'p_value: {p}')
```

```
t_value: 0.3057497440438928
p_value: 0.7660652423223867
```

In [5]:

```
t1, p1= stats.ttest_ind(edu_h_school, edu_Bechelore)
```

```
print(f't_value: {t1}')
print(f'p_value: {p1}')
```

```
t_value: 0.3409594133082003
p_value: 0.7381969448005667
```

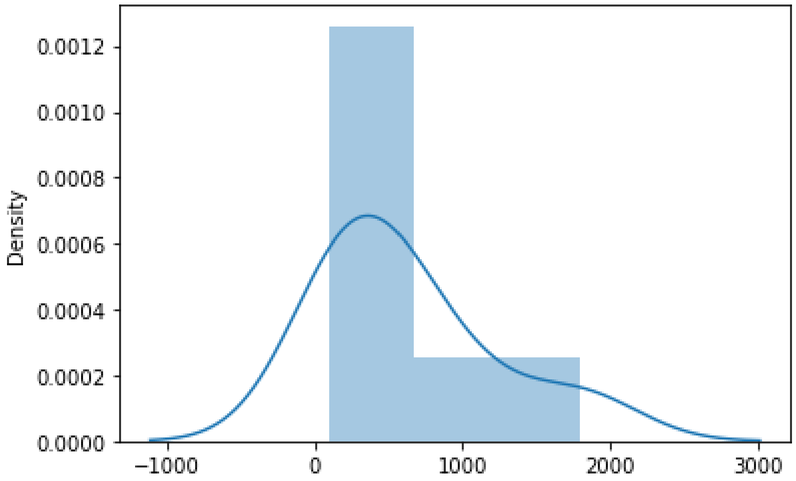
In [6]:

```
import seaborn as sns
sns.distplot(edu_Bechelore)
```

```
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[6]:

```
<AxesSubplot:ylabel='Density'>
```



In [7]:

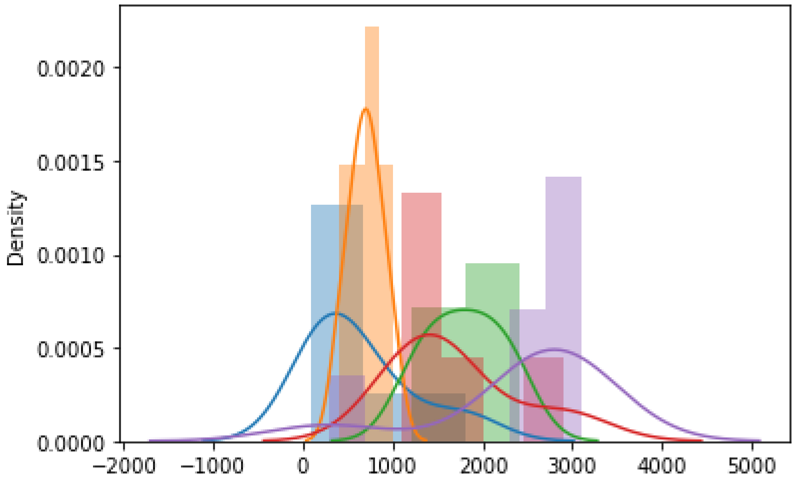
```
import seaborn as sns
sns.distplot(edu_Bechelore, label="Bechelore")
sns.distplot(edu_h_school, label="High school")
sns.distplot(edu_p_school, label="partial school")
sns.distplot(edu_p_college, label="Partial college")
sns.distplot(edu_Graduate, label="Graduate")
```

```
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
```

```
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[7]:

```
<AxesSubplot:ylabel='Density'>
```



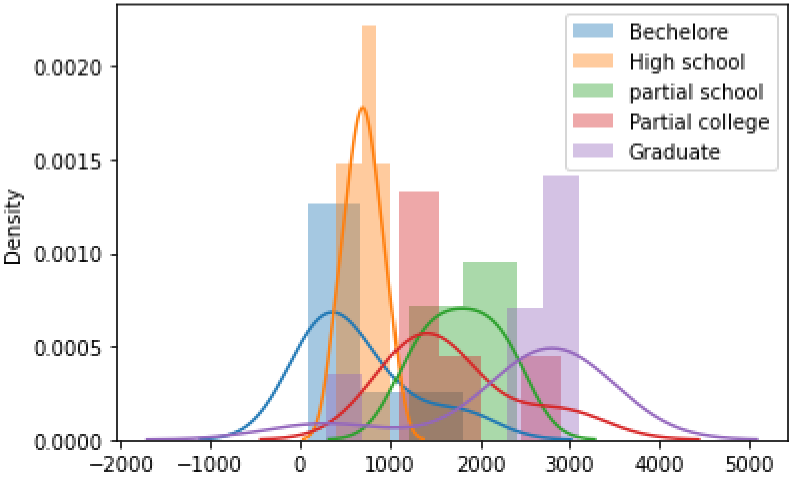
In [8]:

```python
import seaborn as sns
import matplotlib.pyplot as plt
sns.distplot(edu_Bechelore, label="Bechelore")
sns.distplot(edu_h_school, label="High school")
sns.distplot(edu_p_school, label="partial school")
sns.distplot(edu_p_college, label="Partial college")
sns.distplot(edu_Graduate, label="Graduate")
plt.legend()
```

```
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprec
ated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level fun
ction with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[8]:

```
<matplotlib.legend.Legend at 0x1cac722ef10>
```



In [9]:

```
f, p= stats.f_oneway(edu_Bechelore, edu_h_school, edu_p_school, edu_p_college, edu_Graduate)

print(f'f.value: {f}')
print(f'p.value: {p}')
```

```
f.value: 11.797908099455338
p.value: 7.0892689408058515e-06
```

In [10]:

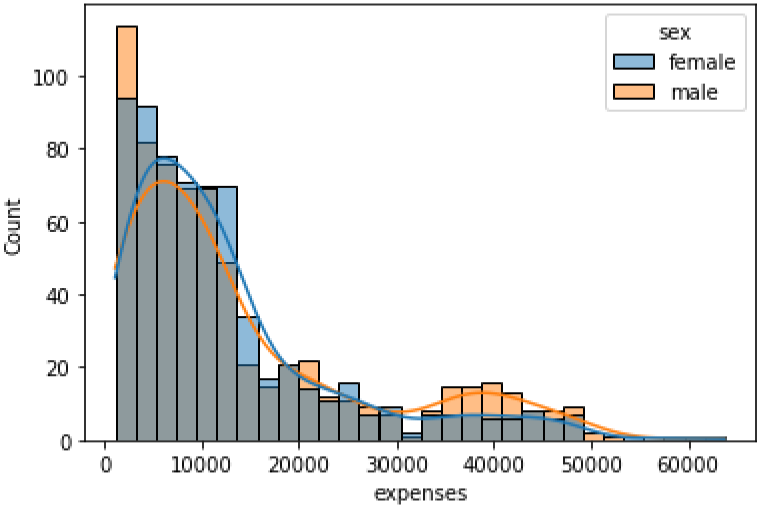
```
df=pd.read_csv("D:\\datasets\\insurance.csv")
df
```

Out[10]:

```
      age     sex   bmi  children  smoker     region  expenses
   0   19  female  27.9         0     yes  southwest  16884.92
   1   18    male  33.8         1      no  southeast   1725.55
   2   28    male  33.0         3      no  southeast   4449.46
   3   33    male  22.7         0      no  northwest  21984.47
   4   32    male  28.9         0      no  northwest   3866.86
 ...  ...     ...   ...       ...     ...        ...       ...
1333   50    male  31.0         3      no  northwest  10600.55
1334   18  female  31.9         0      no  northeast   2205.98
1335   18  female  36.9         0      no  southeast   1629.83
1336   21  female  25.8         0      no  southwest   2007.95
1337   61  female  29.1         0     yes  northwest  29141.36

1338 rows × 7 columns
```

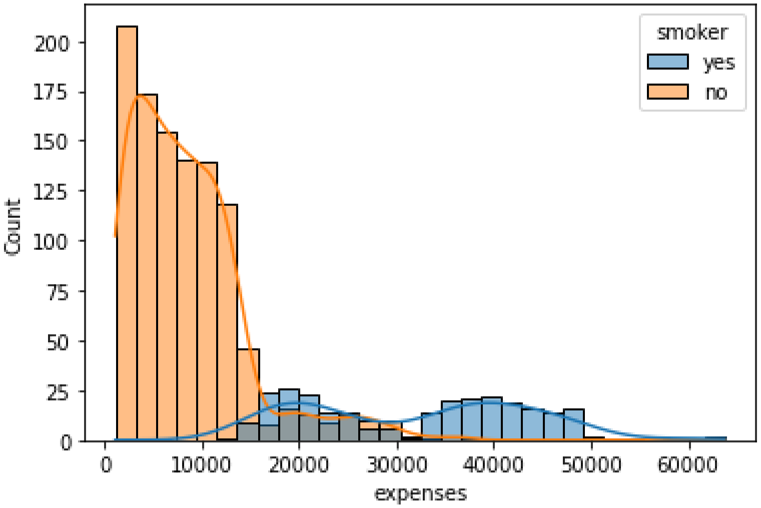
In [11]:

```
import seaborn as sns
sns.histplot(data=df, x="expenses", hue="sex", kde=True);
```



In [12]:

```
sns.histplot(data=df, x="expenses", hue="smoker", kde=True);
```



In [13]:

```
S_complet_y=df[df['smoker']=='yes']
S_complet_n=df[df['smoker']=='no']

stats.ttest_ind(S_complet_y['expenses'], S_complet_n['expenses'])
```

Out[13]:

```
Ttest_indResult(statistic=46.6649210792002, pvalue=8.271449574495316e-283)
```

In [14]:

```
feature='smoker'
label='expenses'

groups=df[feature].unique()
for group in groups:
    print(group)
```

```
yes
no
```

In [15]:

```
feature='smoker'
label='expenses'
groups=df[feature].unique()
grouped_values=[]
for group in groups:
    grouped_values.append(df[df[feature]==group][label])


grouped_values
```

Out[15]:

```
[0       16884.92
 11      27808.73
 14      39611.76
 19      36837.47
 23      37701.88
           ...
 1313    36397.58
 1314    18765.88
 1321    28101.33
 1323    43896.38
 1337    29141.36
 Name: expenses, Length: 274, dtype: float64,
 1        1725.55
 2        4449.46
 3       21984.47
 4        3866.86
 5        3756.62
           ...
 1332    11411.69
 1333    10600.55
 1334     2205.98
 1335     1629.83
 1336     2007.95
 Name: expenses, Length: 1064, dtype: float64]
```

In [16]:

```
feature='smoker'
label='expenses'

groups=df[feature].unique()
grouped_values=[]
for group in groups:
    grouped_values.append(df[df[feature]==group][label])


grouped_values
stats.f_oneway(*grouped_values)
```

Out[16]:

```
F_onewayResult(statistic=2177.6148593279827, pvalue=8.27144957450302e-283)
```

one-way ANOVA t-test *bonferroni* tukeyhsd_ barplot

In [17]:

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

df=pd.read_csv("D:\\datasets\\bike_buyers.csv")
df.head()
```

Out[17]:

	ID	Marital_Status	Gender	Income	Children	Education	Occupation	Home_Owner	Cars	Commute_Distance	Region	Age	Purchased_Bi
0	12496	Married	Female	40000	1	Bachelors	Skilled Manual	Yes	0	0-1 Miles	Europe	42	
1	24107	Married	Male	30000	3	Partial College	Clerical	Yes	1	0-1 Miles	Europe	43	
2	14177	Married	Male	80000	5	Partial College	Professional	No	2	2-5 Miles	Europe	60	
3	24381	Single	Male	70000	0	Bachelors	Professional	Yes	1	5-10 Miles	Pacific	41	

	ID	Marital_Status	Gender	Income	Children	Education	Occupation	Home_Owner	Cars	Commute_Distance	Region	Age	Purchased_Bi
4	25597	Single	Male	30000	0	Bachelors	Clerical	No	0	0-1 Miles	Europe	36	

In [18]:

```
df.Education.unique()
```

Out[18]:

```
array(['Bachelors', 'Partial College', 'High School',
       'Partial High School', 'Graduate Degree'], dtype=object)
```

In [19]:

```
df['Education_rank']=df['Education']
df.Education_rank.replace(['Bachelors', 'Graduate Degree', 'High School', 'Partial College',
            'Partial High School'],[1, 2, 3, 4, 5], inplace=True)
df.Education_rank.astype('int64')
```

Out[19]:

```
0      1
1      4
2      4
3      1
4      1
      ..
995    3
996    2
997    1
998    1
999    3
Name: Education_rank, Length: 1000, dtype: int64
```

In [20]:

```
df['Commute_rank']=df['Commute_Distance']
df.Commute_rank.replace(['0-1 Miles', '1-2 Miles', '2-5 Miles', '5-10 Miles', '10+ Miles'],[0, 1, 2, 5, 10], inplace=True
df.Commute_rank.astype('int64')
```

Out[20]:

```
0       0
1       0
2       2
3       5
4       0
       ..
995     2
996     2
997     0
```

```
998     1
999    10
Name: Commute_rank, Length: 1000, dtype: int64
```

In [21]:

```
def clean_bike_buyers():
    import pandas as pd
    df=pd.read_csv("D:\\datasets\\bike_buyers.csv")
    df['Education_rank']=df['Education']
    df['Commute_rank']=df['Commute_Distance']
    df['Purchased_Bike']=df['Purchased_Bike']

    df.Education_rank.replace(['Bachelors', 'Graduate Degree', 'High School', 'Partial College',
           'Partial High School'],[1, 2, 3, 4, 5], inplace=True)
    df.Commute_rank.replace(['0-1 Miles', '1-2 Miles', '2-5 Miles', '5-10 Miles', '10+ Miles'],[0, 1, 2, 5, 10], inplace=
    df['Purchased_Bike'].replace(['Yes', 'No'], [0,1], inplace=True)

    df.astype({'Education_rank': 'int64'})
    df.astype({'Commute_rank': 'int64'})
    df['Purchased_Bike'].astype('int64')
    return df
```

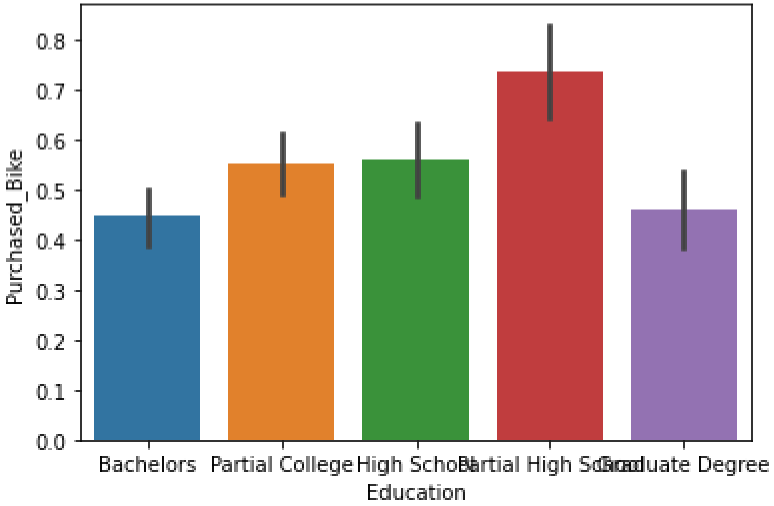
In [22]:

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

df= clean_bike_buyers()
```

In [23]:

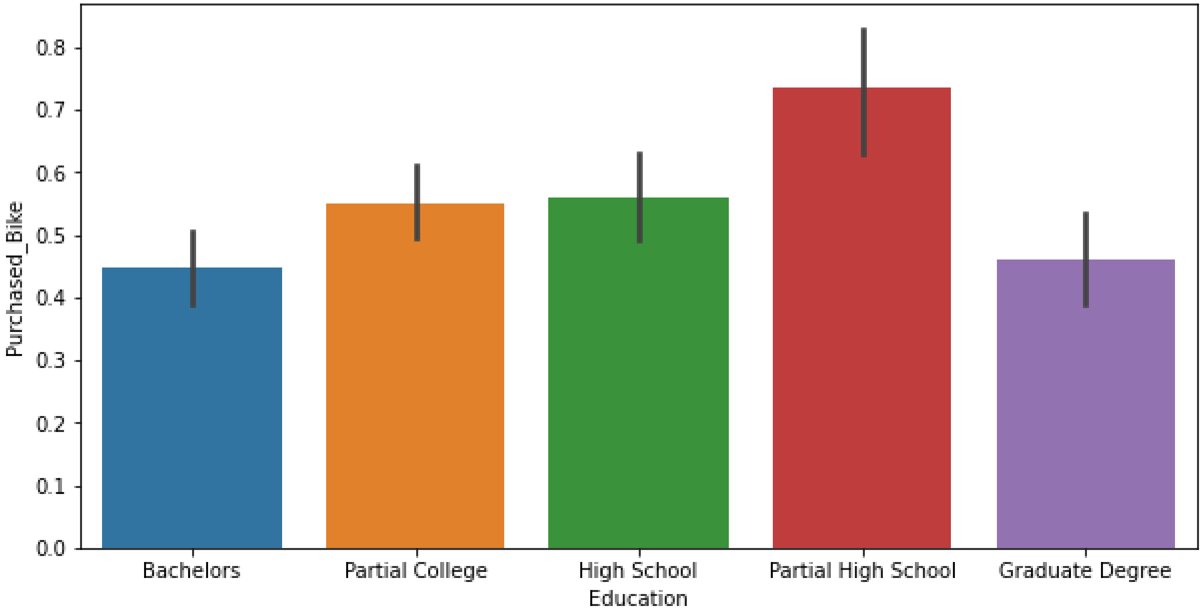
```
import seaborn as sns
sns.barplot(df['Education'], df['Purchased_Bike']);
```

```
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variabl
es as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other argumen
ts without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [24]:

```
plt.figure(figsize=(10, 5))
sns.barplot(x=df['Education'], y=df['Purchased_Bike']);
```

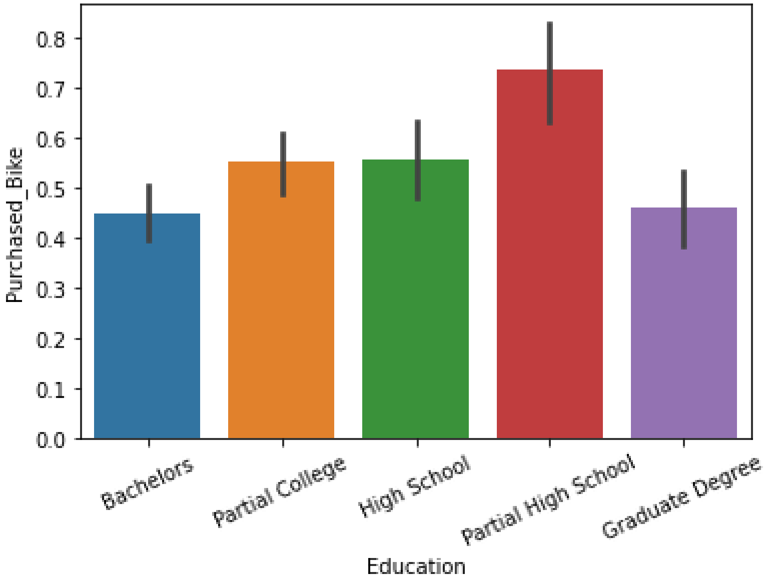


In [25]:

```
viz =sns.barplot(x=df['Education'], y=df['Purchased_Bike'])
viz.set_xticklabels(viz.get_xticklabels(), rotation=25)
```

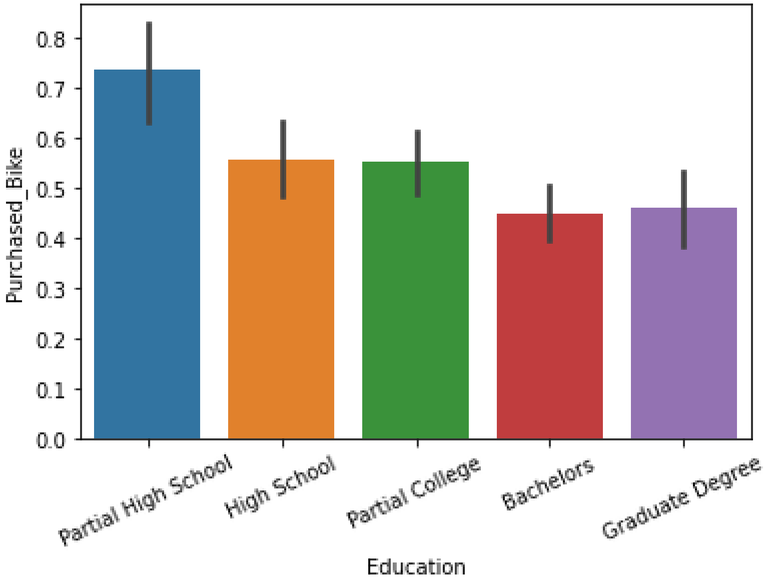
Out[25]:

```
[Text(0, 0, 'Bachelors'),
 Text(1, 0, 'Partial College'),
 Text(2, 0, 'High School'),
 Text(3, 0, 'Partial High School'),
 Text(4, 0, 'Graduate Degree')]
```



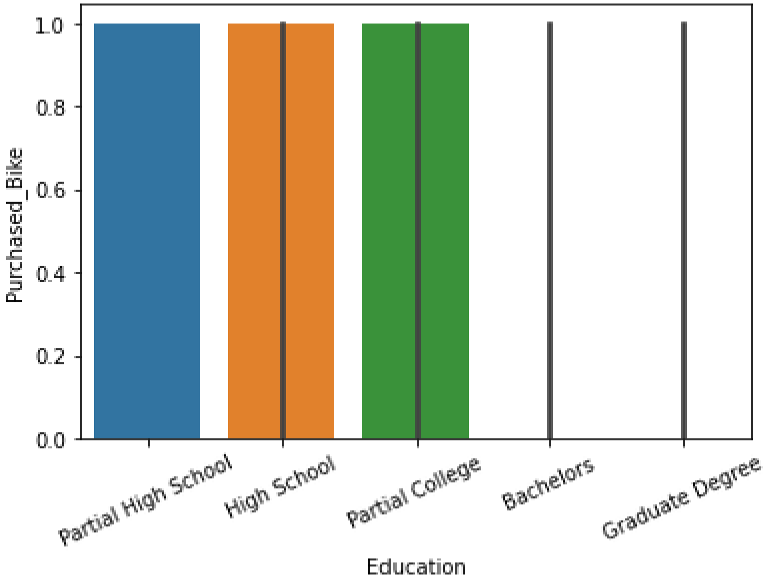
In [26]:

```
viz =sns.barplot(x=df['Education'], y=df['Purchased_Bike'],
                order=['Partial High School', 'High School', 'Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(viz.get_xticklabels(), rotation=25);
```



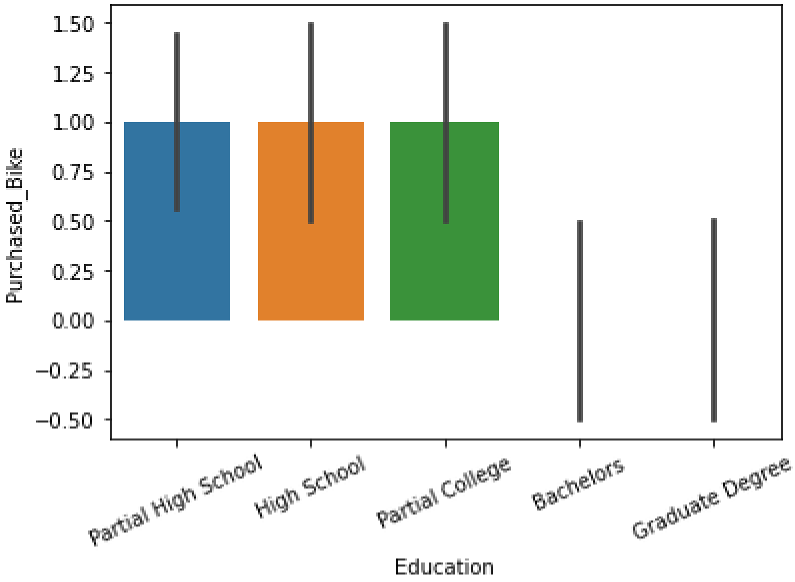
In [27]:

```
viz =sns.barplot(x=df['Education'], y=df['Purchased_Bike'], estimator=np.median,
                order=['Partial High School', 'High School', 'Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(viz.get_xticklabels(), rotation=25);
```



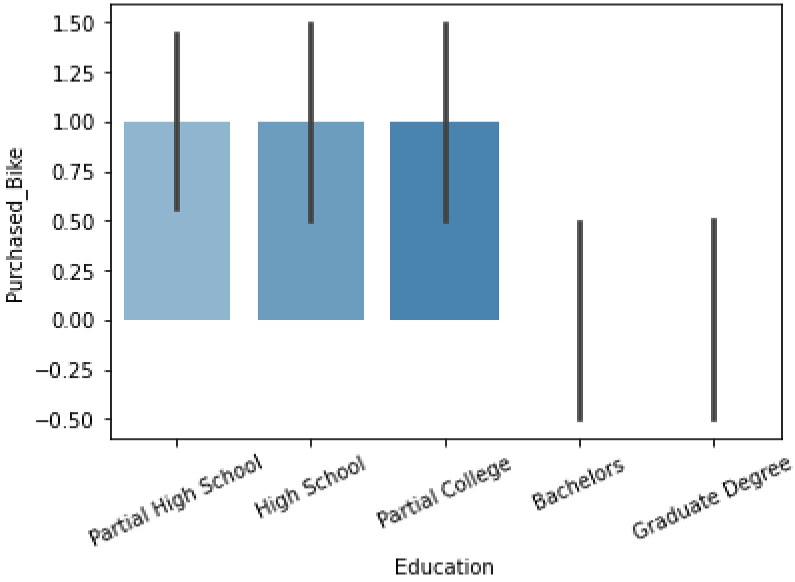
In [28]:

```
viz =sns.barplot(x=df['Education'], y=df['Purchased_Bike'], estimator=np.median, ci='sd',
                order=['Partial High School', 'High School', 'Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(viz.get_xticklabels(), rotation=25);
```



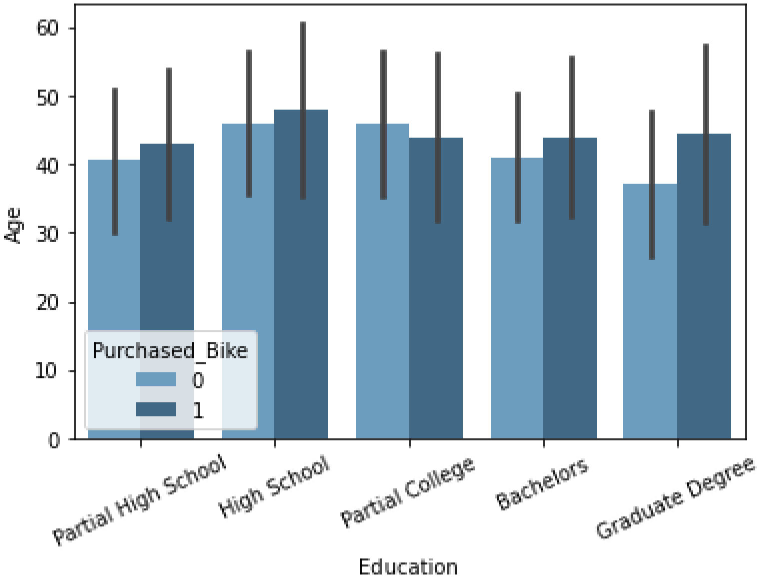
In [29]:

```
viz =sns.barplot(x=df['Education'], y=df['Purchased_Bike'], estimator=np.median, ci='sd', palette='Blues_d',
                order=['Partial High School', 'High School', 'Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(viz.get_xticklabels(), rotation=25);
```



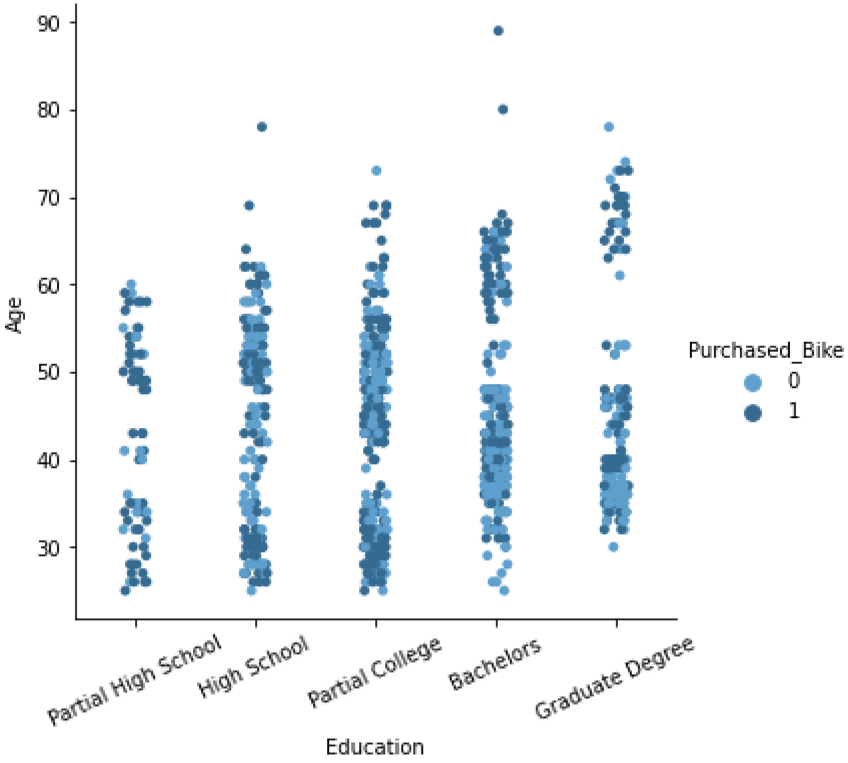
In [30]:

```
viz =sns.barplot(x=df['Education'], y=df['Age'], hue=df['Purchased_Bike'], estimator=np.median, ci='sd', palette='Blues_c
                order=['Partial High School', 'High School', 'Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(viz.get_xticklabels(), rotation=25);
```



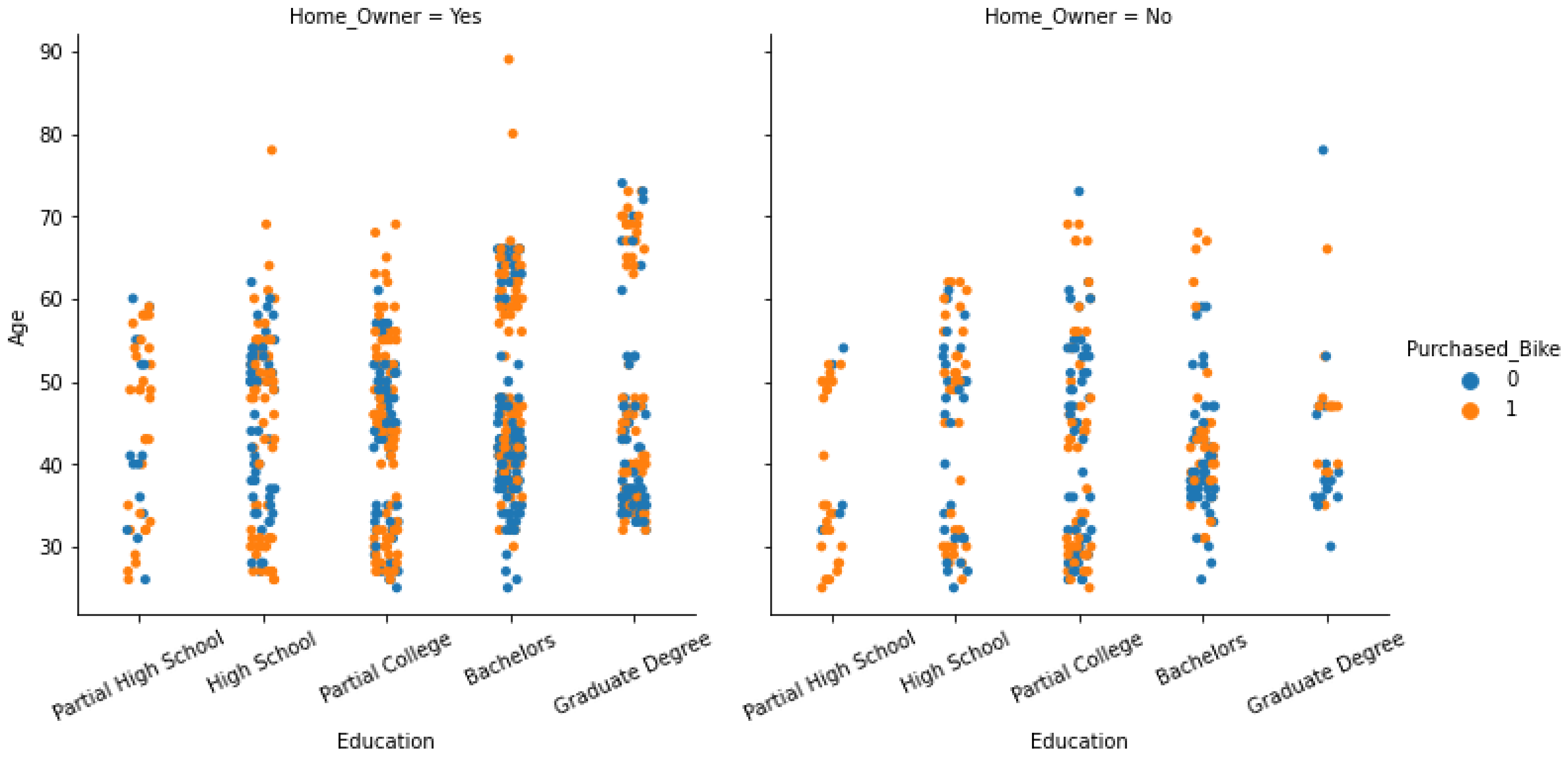
In [31]:

```
viz =sns.catplot(data=df, x='Education', y='Age', hue='Purchased_Bike', color='Home_Owner',
                 estimator=np.median, ci='sd', palette='Blues_d', order=['Partial High School', 'High School',
                                                                         'Partial College', 'Bachelors', 'Graduate Degree
viz.set_xticklabels(rotation=25);
```



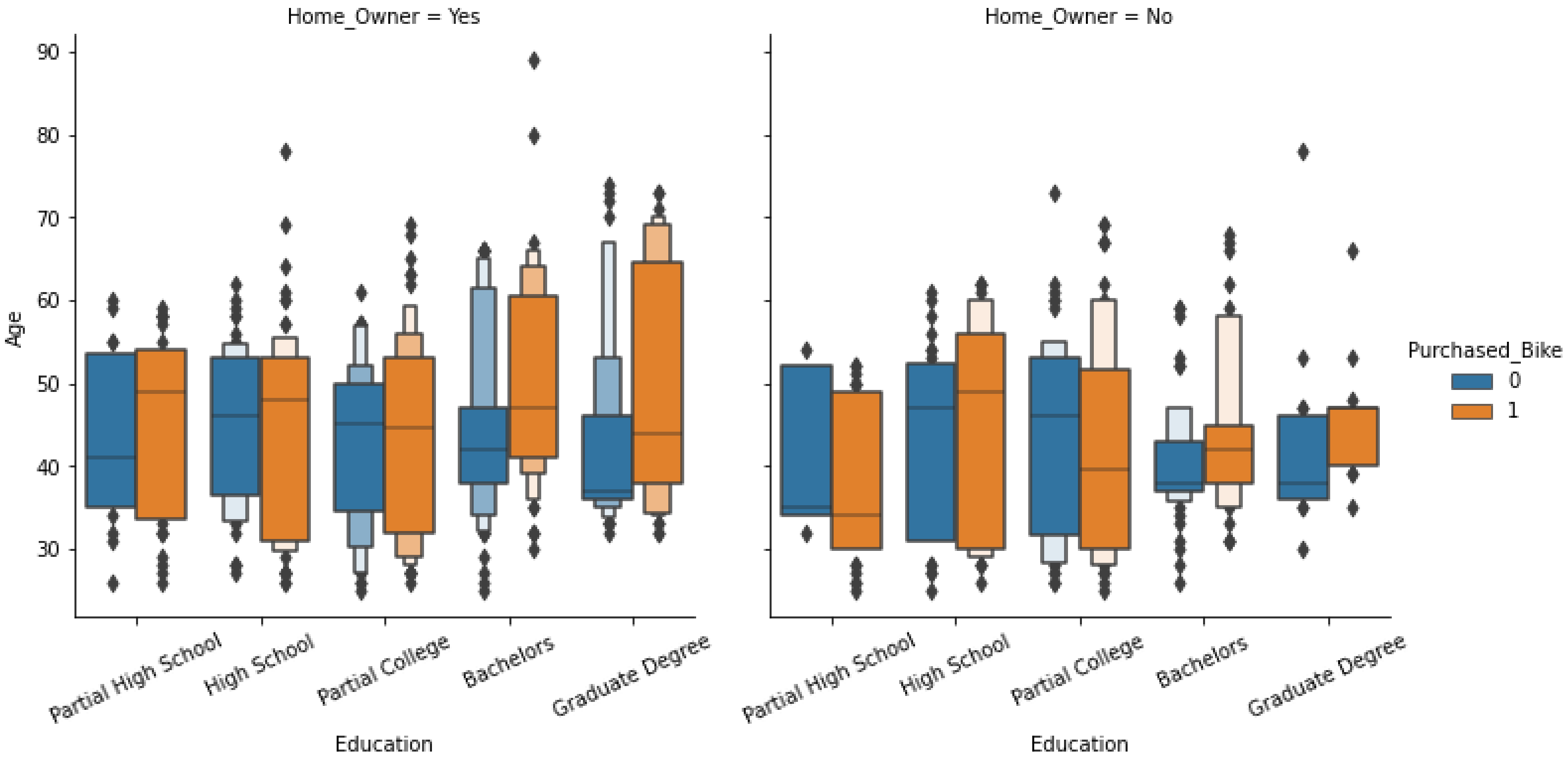
In [32]:

```
viz =sns.catplot(data=df, x='Education', y='Age', hue='Purchased_Bike', col='Home_Owner',
                 estimator=np.median, ci='sd', order=['Partial High School', 'High School',
                                                    'Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(rotation=25);
```



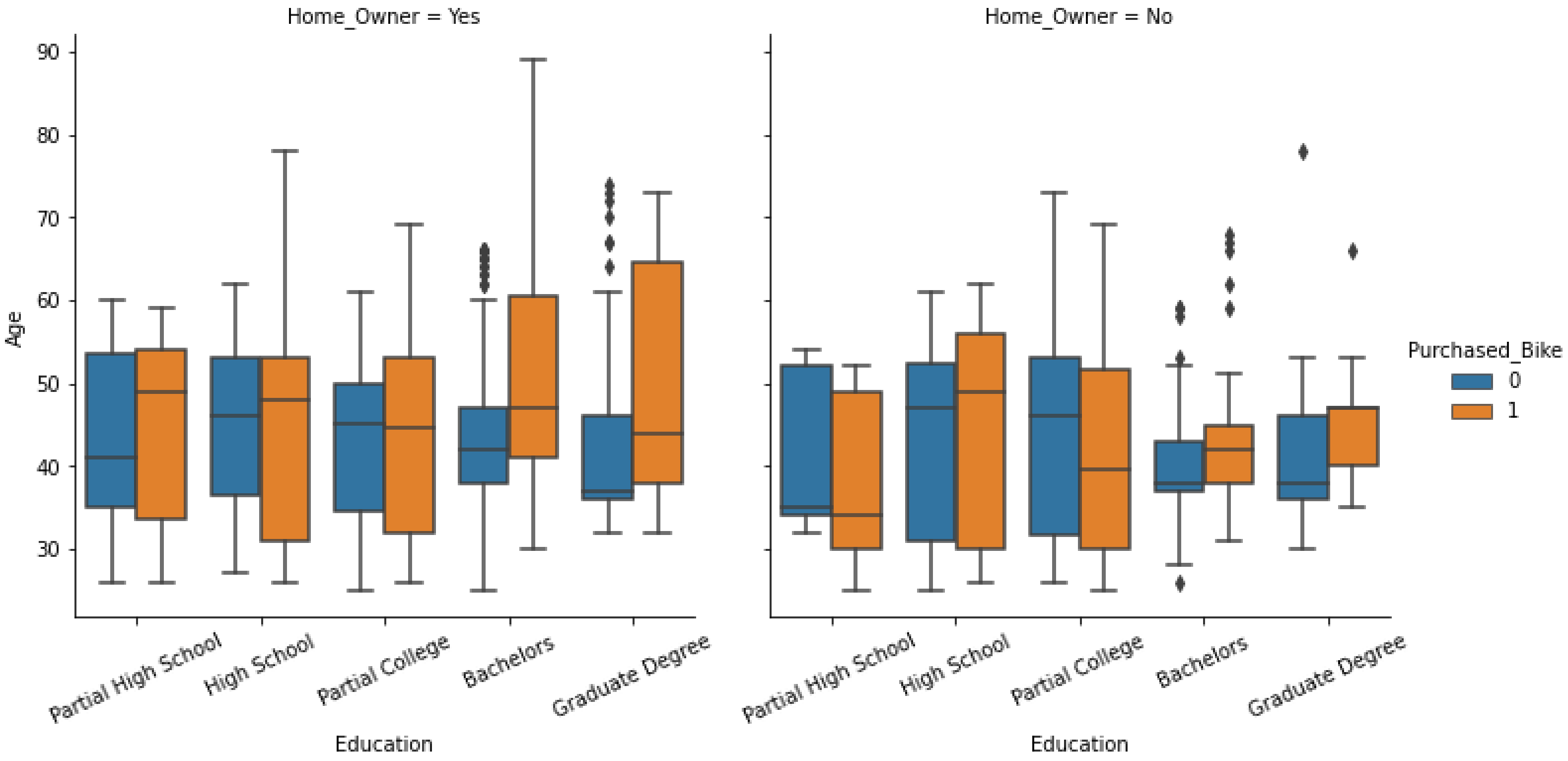
In [33]:

```
viz =sns.catplot(data=df, x='Education', y='Age', hue='Purchased_Bike', col='Home_Owner',
                 estimator=np.median, ci='sd', kind='boxen', order=['Partial High School', 'High School',
                                                                    'Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(rotation=25);
```



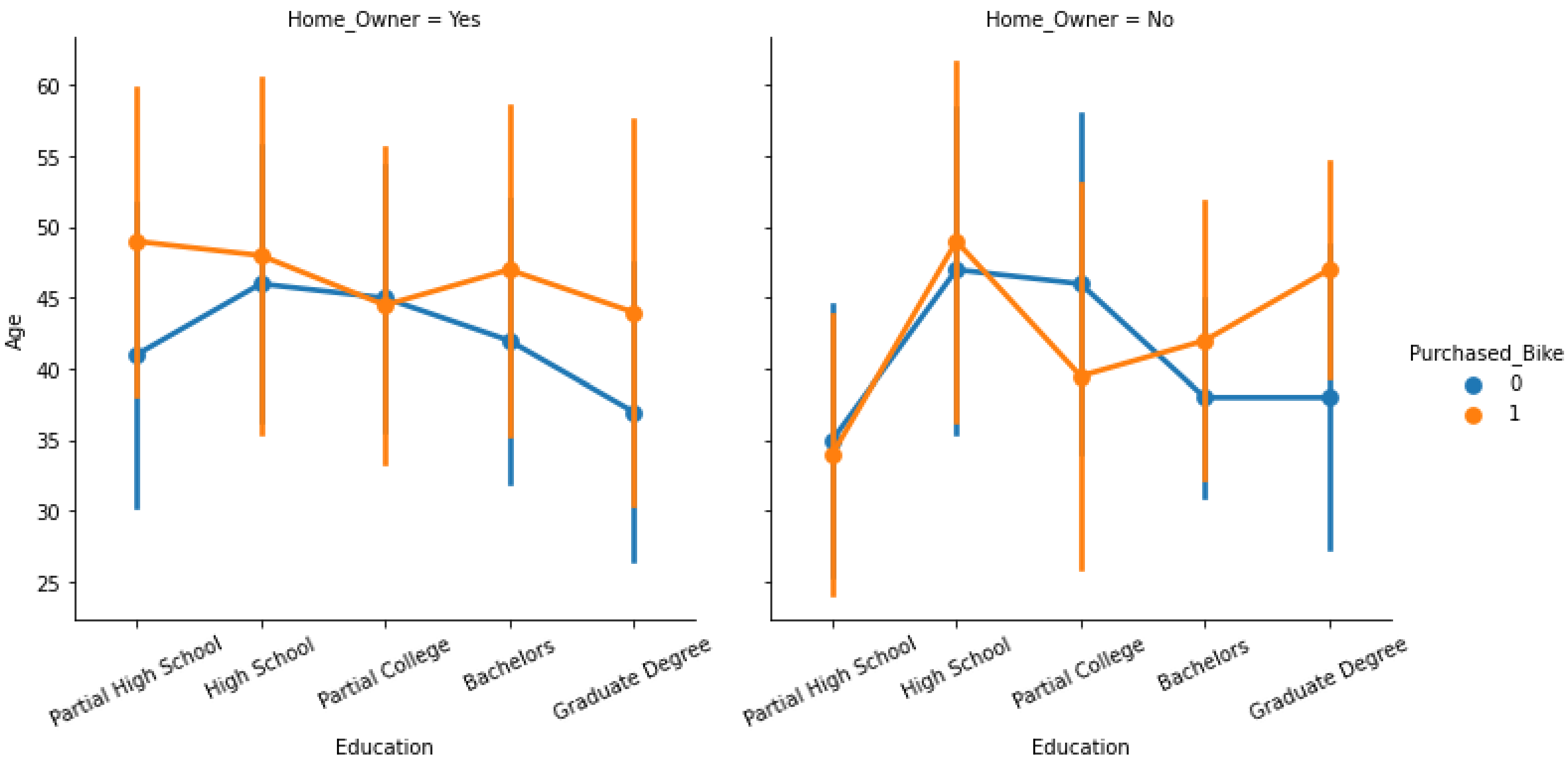
In [34]:

```
viz =sns.catplot(data=df, x='Education', y='Age', hue='Purchased_Bike', col='Home_Owner',
                   estimator=np.median, ci='sd', kind='box', order=['Partial High School', 'High School',
                                                           'Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(rotation=25);
```



In [35]:

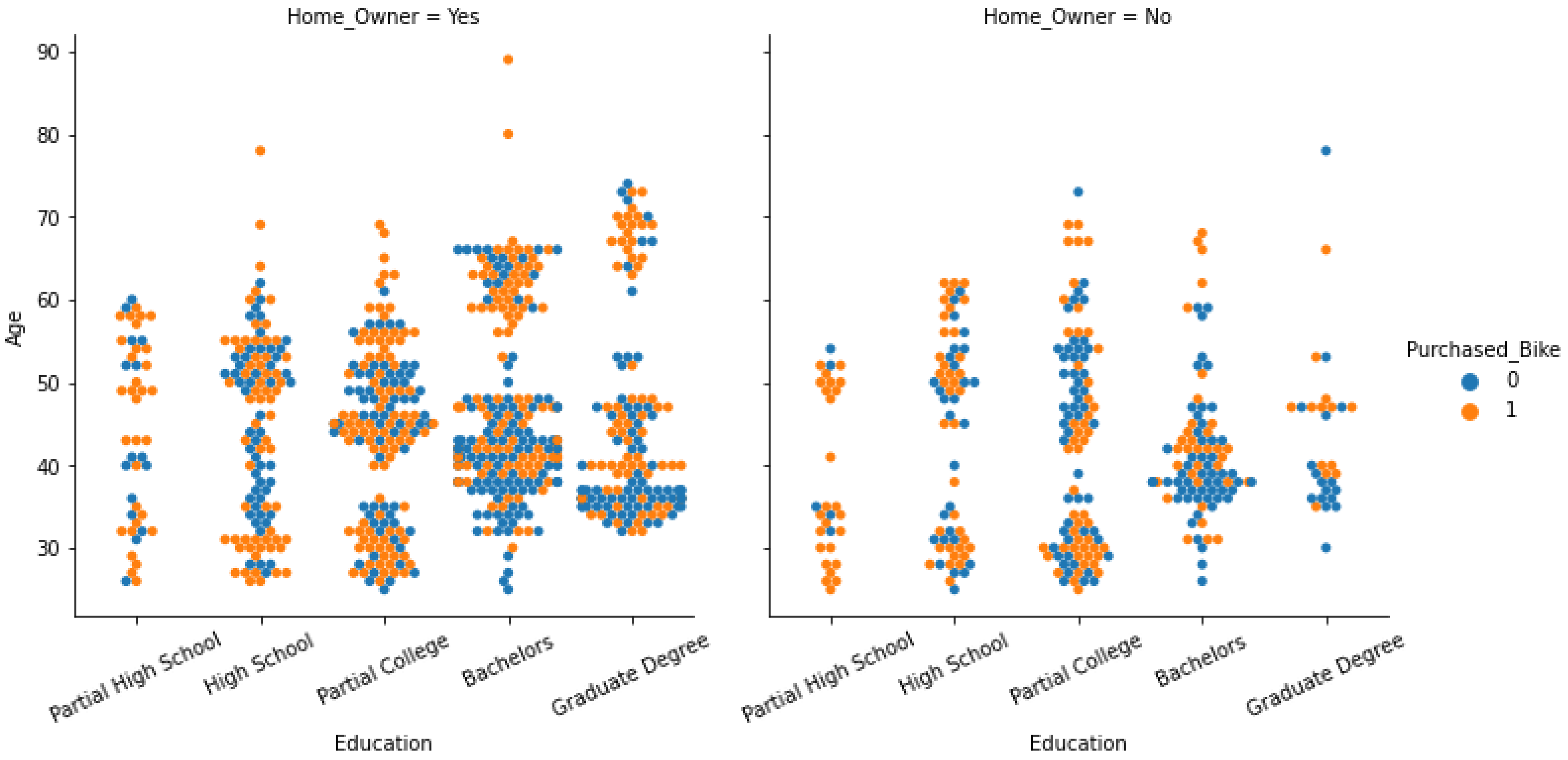
```
viz =sns.catplot(data=df, x='Education', y='Age', hue='Purchased_Bike', col='Home_Owner',
                 estimator=np.median, ci='sd', kind='point', order=['Partial High School', 'High School',
                                                                    'Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(rotation=25);
```



In [36]:

```
viz =sns.catplot(data=df, x='Education', y='Age', hue='Purchased_Bike', col='Home_Owner', estimator=np.median,
ci='sd', kind='swarm', order=['Partial High School', 'High School','Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(rotation=25);
```

```
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 16.1% of the points cannot
be placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\TAWAB COMPUTERS\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 12.5% of the points cannot
be placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```

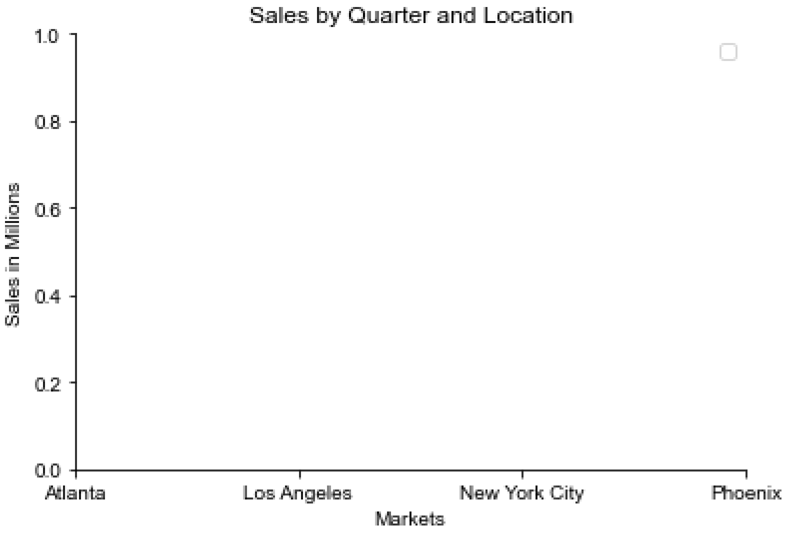


In [37]:

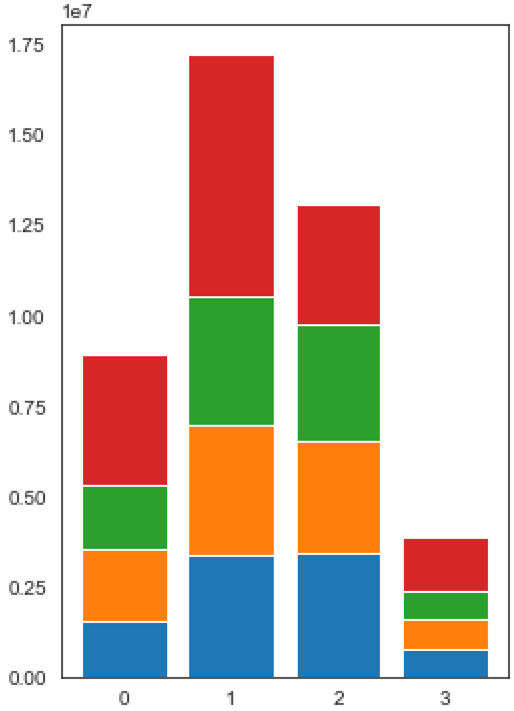
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

df= pd.DataFrame({'Locations':['Atlanta', 'Los Angeles', 'New York City', 'Phoenix'],
                  'Q1 Sales':[1567811, 3391023, 3409871, 789123],
                  'Q2 Sales':[1981237, 3609877, 3100098, 810988],
                  'Q3 Sales':[1761231, 3509889, 3209876, 751233],
                  'Q4 Sales':[3578500, 6712333, 3378900, 1500092]})

#creat the position of the bar
x= np.arange(len(df.Locations))
# store the three columns from the DataFrame and "flatten" them
#to appear as regular python list structure
list_1 = df['Q1 Sales'].values.flatten()
list_2 = df['Q2 Sales'].values.flatten()
list_3 = df['Q3 Sales'].values.flatten()
list_4 = df['Q4 Sales'].values.flatten()
```

```
#plot the pokem names as the x ticks
plt.xticks(x, df.Locations)

# creat a legend
plt.legend(loc='upper right')

#Add label and title
plt.xlabel('Markets')
plt.ylabel('Sales in Millions')
plt.title('Sales by Quarter and Location')

#add an sns style and increase figure size
sns.set_style('white')
sns.set_context({"figure.figsize": (4, 6)})
sns.despine(top=True, right=True)

#show the plot
plt.show()
#Plot thr Barchat
plt.bar(x, list_1, label='Q1')
plt.bar(x, list_2, bottom=list_1, label='Q2')
plt.bar(x, list_3, bottom=list_1+list_2, label='Q3')
plt.bar(x, list_4, bottom=list_1+list_2+list_3, label='Q4')
```

```
No handles with labels found to put in legend.
```



```
Out[37]: <BarContainer object of 4 artists>
```



In [38]:

```
from scipy import stats
import pandas as pd

df= clean_bike_buyers()

groups= df['Education'].unique()
group_labels=[]
for g in groups:
    group_labels.append(df[df["Education"]==g]["Purchased_Bike"])

#now calculet the ANOVA results
F, p =stats.f_oneway(*group_labels)

print('F: ' +str(round(F, 4)))
print('p: ' +str(round(p, 4)))
```

```
F: 6.4653
p: 0.0
```

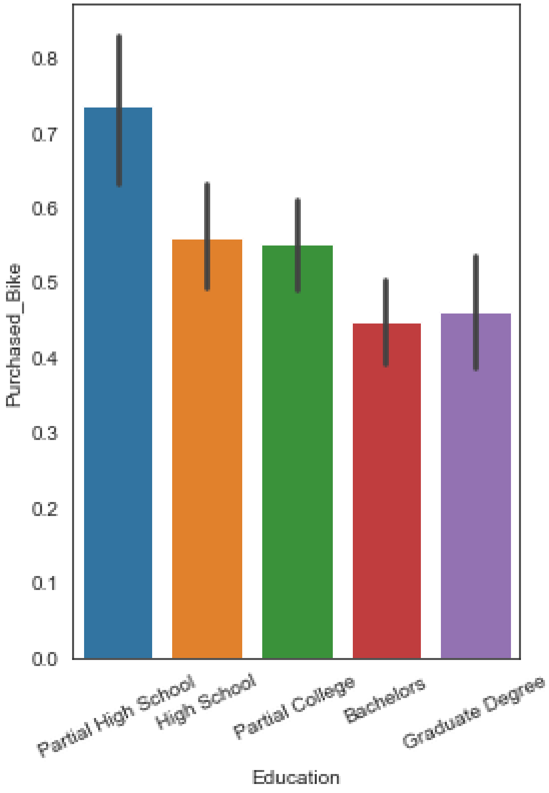
In [39]:

```
r, p = stats.pearsonr(df["Education_rank"], df["Purchased_Bike"])

print('r: ' + str(round(r, 4)))
print('p: ' + str(round(p, 4)))
```

```
r: 0.1413
p: 0.0
```

In [40]:

```
viz =sns.barplot(x=df['Education'], y=df['Purchased_Bike'],
                order=['Partial High School', 'High School', 'Partial College', 'Bachelors', 'Graduate Degree']);
viz.set_xticklabels(viz.get_xticklabels(), rotation=25);
```



In [41]:

```
Partial_high_school=df[df.Education== 'Partial High School']
High_school=df[df.Education== 'High School']
t, p= stats.ttest_ind(Partial_high_school['Purchased_Bike'], High_school['Purchased_Bike'])
```

```
print('t: ' + str(round(t, 2)))
print('p: ' + str(round(p, 2)))
```

```
t: 2.7
p: 0.01
```

In [42]:

```
from statsmodels.stats.multicomp import MultiComparison

mc= MultiComparison(df['Purchased_Bike'], df['Education'])
print(mc.tukeyhsd())
```

```
           Multiple Comparison of Means - Tukey HSD, FWER=0.05           
=========================================================================
    group1          group2      meandiff p-adj   lower  upper  reject
-------------------------------------------------------------------------
      Bachelors     Graduate Degree   0.0121    0.9 -0.1163 0.1404  False
      Bachelors         High School   0.1109 0.1206 -0.0162 0.2381  False
      Bachelors     Partial College   0.1032 0.0941 -0.0102 0.2166  False
      Bachelors Partial High School   0.2891  0.001  0.1159 0.4623   True
Graduate Degree         High School   0.0989 0.3296  -0.045 0.2428  False
Graduate Degree     Partial College   0.0912 0.3235 -0.0407  0.223  False
Graduate Degree Partial High School   0.2771  0.001  0.0913 0.4629   True
    High School     Partial College  -0.0077    0.9 -0.1385  0.123  False
    High School Partial High School   0.1782 0.0656 -0.0068 0.3632  False
Partial College Partial High School   0.1859 0.0322  0.0101 0.3617   True
-------------------------------------------------------------------------
```

In [43]:

```
e_types= df.Education.unique()
ttests= []

for i, e in enumerate(e_types):
    print(i, '', '', e)
```

```
0   Bachelors
1   Partial College
2   High School
3   Partial High School
4   Graduate Degree
```

In [44]:

```
e_types= df.Education.unique()
ttests= []

for i, e in enumerate(e_types):
    for i2, e2 in enumerate(e_types):
        if i2>i:
            g1 = df[df.Education==e]['Purchased_Bike']
            g2 = df[df.Education==e2]['Purchased_Bike']
            t, p = stats.ttest_ind(g1, g2)
            ttests.append(f'{e} - {e2}: {t.round(4)}, {p.round(4)}')
            print(f'{e} - {e2}: {t.round(4)}, {p.round(4)}')
    #print(i, '', '', e)
```

```
Bachelors - Partial College: -2.4693, 0.0138
Bachelors - High School: -2.3675, 0.0183
Bachelors - Partial High School: -4.6253, 0.0
Bachelors - Graduate Degree: -0.2546, 0.7991
Partial College - High School: -0.1601, 0.8729
Partial College - Partial High School: -2.9354, 0.0036
Partial College - Graduate Degree: 1.8728, 0.0618
High School - Partial High School: -2.698, 0.0074
High School - Graduate Degree: 1.862, 0.0634
Partial High School - Graduate Degree: 4.1685, 0.0
```

In [45]:

```
e_types= df.Education.unique()
ttests= []

for i, e in enumerate(e_types):
    for i2, e2 in enumerate(e_types):
        if i2>i:
            g1 = df[df.Education==e]['Purchased_Bike']
            g2 = df[df.Education==e2]['Purchased_Bike']
            t, p = stats.ttest_ind(g1, g2)
            ttests.append(f'{e} - {e2}: {t.round(4)}, {p.round(4)}')

threshold = 0.05/ len(ttests)
print(f'Significant t-test below {threshold}:')
for t in ttests:
    if t[2] <= threshold:
        print(t)
```

```
Significant t-test below 0.005:

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:\Users\TAWABC~1\AppData\Local\Temp/ipykernel_2528/2544714263.py in <module>
     13 print(f'Significant t-test below {threshold}:')
     14 for t in ttests:
---> 15     if t[2] <= threshold:
     16         print(t)
```

```
TypeError: '<=' not supported between instances of 'str' and 'float'
```

In [ ]:

Intro to MLR OLS in statmodels.api

In [46]:

```
import pandas as pd, numpy as np, statsmodels.api as sm
```

In [47]:

```
df=pd.read_csv("D:\\datasets\\insurance.csv")
df.head()
```

Out[47]:

	age	sex	bmi	children	smoker	region	expenses
0	19	female	27.9	0	yes	southwest	16884.92
1	18	male	33.8	1	no	southeast	1725.55
2	28	male	33.0	3	no	southeast	4449.46
3	33	male	22.7	0	no	northwest	21984.47
4	32	male	28.9	0	no	northwest	3866.86

In [48]:

```
label = "expenses"

y=df.expenses
x=df[['age', 'bmi', 'children']].assign(const=1)

model= sm.OLS(y, x)
results= model.fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               expenses   R-squared:                       0.120
Model:                            OLS   Adj. R-squared:                  0.118
Method:                 Least Squares   F-statistic:                     60.74
Date:                Thu, 10 Aug 2023   Prob (F-statistic):           8.32e-37
```

```
Time:                        16:21:39   Log-Likelihood:                -14392.
No. Observations:                1338   AIC:                         2.879e+04
Df Residuals:                    1334   BIC:                         2.881e+04
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
age          239.9626     22.288     10.766      0.000     196.239     283.686
bmi          332.5216     51.307      6.481      0.000     231.870     433.173
children     543.0436    258.230      2.103      0.036      36.462    1049.625
const      -6929.3145   1757.434     -3.943      0.000   -1.04e+04   -3481.678
==============================================================================
Omnibus:                      325.223   Durbin-Watson:                   2.012
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              602.850
Skew:                           1.520   Prob(JB):                    1.24e-131
Kurtosis:                       4.254   Cond. No.                         290.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [49]:

```python
df['predictions']=results.fittedvalues
df
```

Out[49]:

	age	sex	bmi	children	smoker	region	expenses	predictions
0	19	female	27.9	0	yes	southwest	16884.92	6907.326136
1	18	male	33.8	1	no	southeast	1725.55	9172.284502
2	28	male	33.0	3	no	southeast	4449.46	12391.979997
3	33	male	22.7	0	no	northwest	21984.47	8537.689692
4	32	male	28.9	0	no	northwest	3866.86	10359.360926
...	...	...	...	...	...	...	...	...
1333	50	male	31.0	3	no	northwest	10600.55	17006.113044
1334	18	female	31.9	0	no	northeast	2205.98	7997.449896
1335	18	female	36.9	0	no	southeast	1629.83	9660.057790
1336	21	female	25.8	0	no	southwest	2007.95	6688.955930

	age	sex	bmi	children	smoker	region	expenses	predictions
1337	61	female	29.1	0	yes	northwest	29141.36	17384.779329

1338 rows × 8 columns

In [50]:

```
print(results.predict([19, 27.9, 0, 1]))
```

```
[6907.32613573]
```

MLR with categorical values dummy codes

In [51]:

```
for col in df:
    if not pd.api.types.is_numeric_dtype(df[col]):
        df= pd.get_dummies(df, columns=[col], drop_first=True)
df.head()
```

Out[51]:

	age	bmi	children	expenses	predictions	sex_male	smoker_yes	region_northwest	region_southeast	region_southwest
0	19	27.9	0	16884.92	6907.326136	0	1	0	0	1
1	18	33.8	1	1725.55	9172.284502	1	0	0	1	0
2	28	33.0	3	4449.46	12391.979997	1	0	0	1	0
3	33	22.7	0	21984.47	8537.689692	1	0	1	0	0
4	32	28.9	0	3866.86	10359.360926	1	0	1	0	0

In [52]:

```
import pandas as pd, numpy as np, statsmodels.api as sm
x= df.drop(columns=[label]).assign(const=1)
results = sm.OLS(y, x).fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               expenses   R-squared:                       0.751
Model:                            OLS   Adj. R-squared:                  0.749
Method:                 Least Squares   F-statistic:                     500.9
Date:                Thu, 10 Aug 2023   Prob (F-statistic):               0.00
```

```
Time:                        16:21:40   Log-Likelihood:                -13548.
No. Observations:                1338   AIC:                         2.711e+04
Df Residuals:                    1329   BIC:                         2.716e+04
Df Model:                           8
Covariance Type:            nonrobust
=====================================================================================
                        coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------
age                -154.8900     31.521     -4.914      0.000    -216.727     -93.053
bmi                -231.2527     30.546     -7.571      0.000    -291.177    -171.328
children           -456.0686    149.330     -3.054      0.002    -749.018    -163.120
predictions           1.7158      0.142     12.122      0.000       1.438       1.993
sex_male           -131.3520    332.935     -0.395      0.693    -784.488     521.784
smoker_yes         2.385e+04    413.139     57.723      0.000     2.3e+04    2.47e+04
region_northwest   -352.7901    476.261     -0.741      0.459   -1287.095     581.515
region_southeast -1035.5957    478.681     -2.163      0.031   -1974.648     -96.544
region_southwest   -959.3058    477.912     -2.007      0.045   -1896.850     -21.762
const               -52.2030     12.612     -4.139      0.000     -76.944     -27.462
==============================================================================
Omnibus:                      300.499   Durbin-Watson:                   2.088
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              719.382
Skew:                           1.212   Prob(JB):                    6.14e-157
Kurtosis:                       5.652   Cond. No.                     2.34e+19
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 4.75e-28. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

MLR OLS *standardization* normalization

In [53]:

```
from sklearn import preprocessing

df_zscore= pd.DataFrame(preprocessing.StandardScaler().fit_transform(df), columns=df.columns)
df_zscore.head()
```

Out[53]:

	age	bmi	children	expenses	predictions	sex_male	smoker_yes	region_northwest	region_southeast	region_southwest
0	-1.438764	-0.453646	-0.908614	0.298583	-1.516295	-1.010519	1.970587	-0.566418	-0.611324	1.765481
1	-1.509965	0.514186	-0.078767	-0.953689	-0.976566	0.989591	-0.507463	-0.566418	1.635795	-0.566418

	age	bmi	children	expenses	predictions	sex_male	smoker_yes	region_northwest	region_southeast	region_southwest
2	-0.797954	0.382954	1.580926	-0.728675	-0.209329	0.989591	-0.507463	-0.566418	1.635795	-0.566418
3	-0.441948	-1.306650	-0.908614	0.719843	-1.127787	0.989591	-0.507463	1.765481	-0.611324	-0.566418
4	-0.513149	-0.289606	-0.908614	-0.776802	-0.693692	0.989591	-0.507463	1.765481	-0.611324	-0.566418

In [54]:

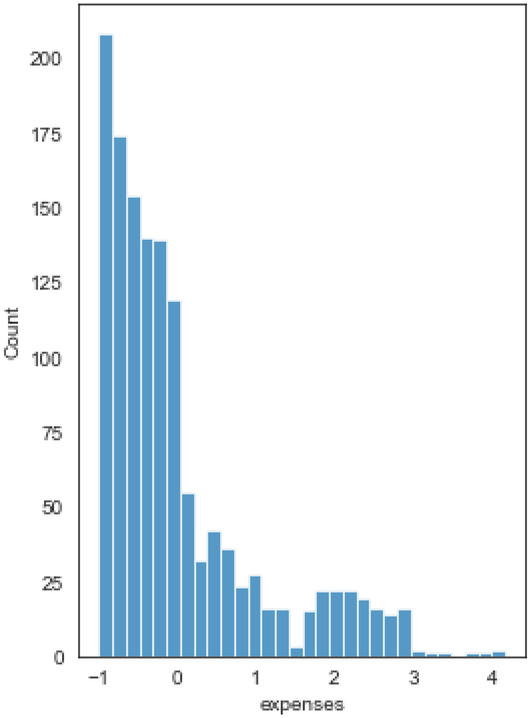
```python
y= df_zscore.expenses

x=df_zscore.drop(columns=['predictions', 'expenses']).assign(const=1)

model= sm.OLS(y, x)
results= model.fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               expenses   R-squared:                       0.751
Model:                            OLS   Adj. R-squared:                  0.749
Method:                 Least Squares   F-statistic:                     500.9
Date:                Thu, 10 Aug 2023   Prob (F-statistic):               0.00
Time:                        16:21:41   Log-Likelihood:                -968.62
No. Observations:                1338   AIC:                             1955.
Df Residuals:                    1329   BIC:                             2002.
Df Model:                           8
Covariance Type:            nonrobust
====================================================================================
                       coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------
age                  0.2980      0.014     21.586      0.000       0.271       0.325
bmi                  0.1709      0.014     11.864      0.000       0.143       0.199
children             0.0474      0.014      3.452      0.001       0.020       0.074
sex_male            -0.0054      0.014     -0.395      0.693      -0.032       0.022
smoker_yes           0.7950      0.014     57.723      0.000       0.768       0.822
region_northwest    -0.0125      0.017     -0.741      0.459      -0.046       0.021
region_southeast    -0.0381      0.018     -2.163      0.031      -0.073      -0.004
region_southwest    -0.0340      0.017     -2.007      0.045      -0.067      -0.001
const             3.296e-17      0.014   2.41e-15      1.000      -0.027       0.027
==============================================================================
Omnibus:                      300.499   Durbin-Watson:                   2.088
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              719.382
Skew:                           1.212   Prob(JB):                    6.14e-157
Kurtosis:                       5.652   Cond. No.                         2.21
```

```
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

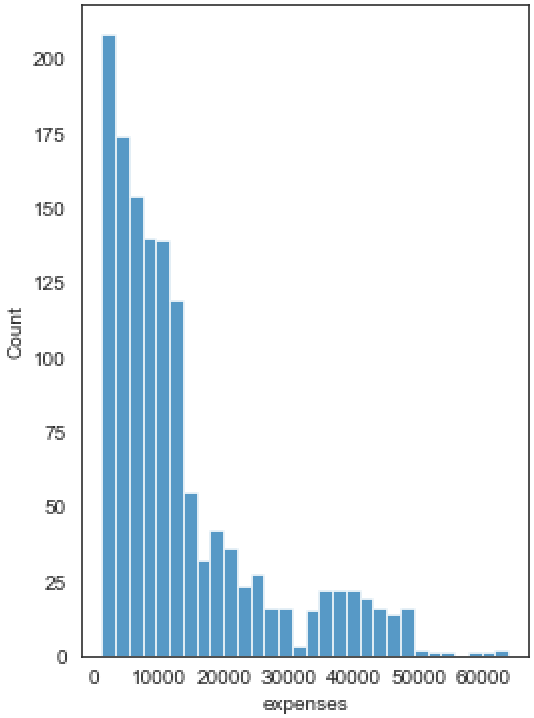
In [55]:

```
import seaborn as sns

sns.histplot(y);
```



In [56]:

```
sns.histplot(df.expenses);
```



In [57]:

```
df_minmax= pd.DataFrame(preprocessing.MinMaxScaler().fit_transform(df), columns=df.columns)
df_minmax.head()
```

Out[57]:

	age	bmi	children	expenses	predictions	sex_male	smoker_yes	region_northwest	region_southeast	region_southwest
0	0.021739	0.320755	0.0	0.251611	0.198761	0.0	1.0	0.0	0.0	1.0
1	0.000000	0.479784	0.2	0.009636	0.306026	1.0	0.0	0.0	1.0	0.0
2	0.217391	0.458221	0.6	0.053115	0.458506	1.0	0.0	0.0	1.0	0.0
3	0.326087	0.180593	0.0	0.333010	0.275973	1.0	0.0	1.0	0.0	0.0
4	0.304348	0.347709	0.0	0.043816	0.362244	1.0	0.0	1.0	0.0	0.0

In [58]:

```
y= df_minmax.expenses

x=df_minmax.drop(columns=['predictions', 'expenses']).assign(const=1)

model= sm.OLS(y, x)
```

```
results= model.fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               expenses   R-squared:                       0.751
Model:                            OLS   Adj. R-squared:                  0.749
Method:                 Least Squares   F-statistic:                     500.9
Date:                Thu, 10 Aug 2023   Prob (F-statistic):               0.00
Time:                        16:21:41   Log-Likelihood:                 1230.9
No. Observations:                1338   AIC:                            -2444.
Df Residuals:                    1329   BIC:                            -2397.
Df Model:                           8
Covariance Type:            nonrobust
====================================================================================
                       coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------
age                  0.1886      0.009     21.586      0.000       0.171       0.206
bmi                  0.2009      0.017     11.864      0.000       0.168       0.234
children             0.0380      0.011      3.452      0.001       0.016       0.060
sex_male            -0.0021      0.005     -0.395      0.693      -0.013       0.008
smoker_yes           0.3807      0.007     57.723      0.000       0.368       0.394
region_northwest    -0.0056      0.008     -0.741      0.459      -0.021       0.009
region_southeast    -0.0165      0.008     -2.163      0.031      -0.032      -0.002
region_southwest    -0.0153      0.008     -2.007      0.045      -0.030      -0.000
const               -0.0481      0.009     -5.137      0.000      -0.066      -0.030
==============================================================================
Omnibus:                      300.499   Durbin-Watson:                   2.088
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              719.382
Skew:                           1.212   Prob(JB):                    6.14e-157
Kurtosis:                       5.652   Cond. No.                         9.58
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

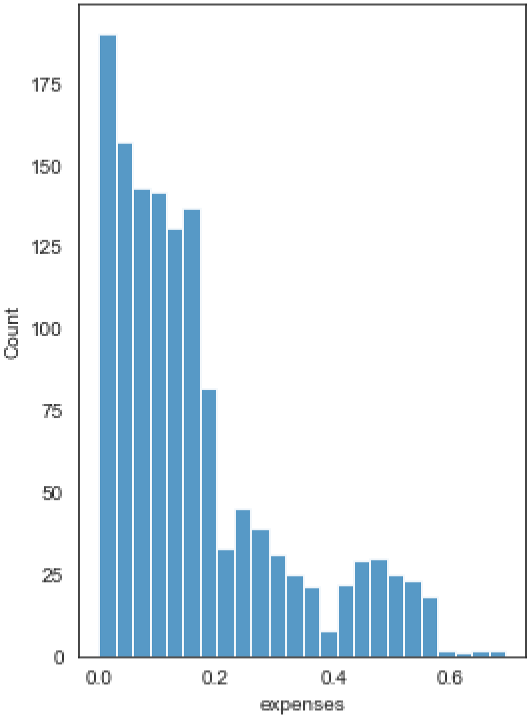
_MLR_OLS assumptions normality multicollinearity VIF

In [59]:

```
y=np.log1p(y)
sns.histplot(y)
```

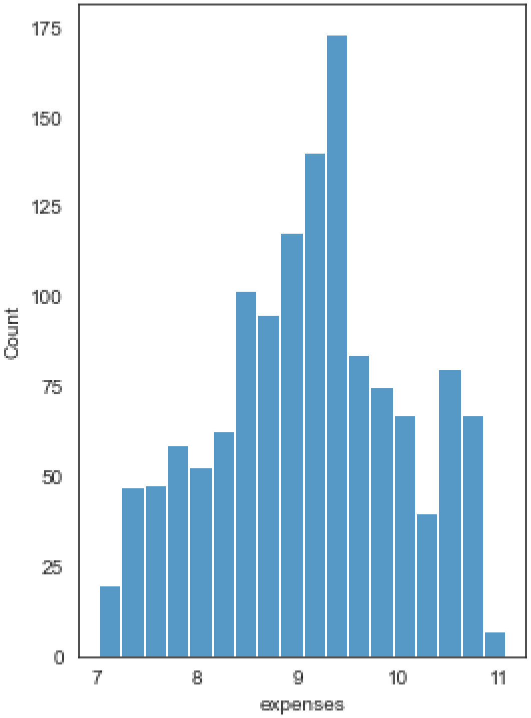
Out[59]:

```
<AxesSubplot:xlabel='expenses', ylabel='Count'>
```



In [60]:

```
sns.histplot(np.log(df.expenses));
```



In [61]:

```
y= np.log(df.expenses)
x= df.drop(columns=['predictions', 'expenses']).assign(const=1)


print(sm.OLS(y, x).fit().summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               expenses   R-squared:                       0.768
Model:                            OLS   Adj. R-squared:                  0.767
Method:                 Least Squares   F-statistic:                     549.7
Date:                Thu, 10 Aug 2023   Prob (F-statistic):               0.00
Time:                        16:21:42   Log-Likelihood:                -808.54
No. Observations:                1338   AIC:                             1635.
Df Residuals:                    1329   BIC:                             1682.
Df Model:                           8
Covariance Type:            nonrobust
====================================================================================
                       coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------
```

```
age                   0.0346      0.001     39.654      0.000       0.033       0.036
bmi                   0.0134      0.002      6.377      0.000       0.009       0.017
children              0.1019      0.010     10.086      0.000       0.082       0.122
sex_male             -0.0754      0.024     -3.090      0.002      -0.123      -0.028
smoker_yes            1.5543      0.030     51.330      0.000       1.495       1.614
region_northwest     -0.0638      0.035     -1.827      0.068      -0.132       0.005
region_southeast     -0.1572      0.035     -4.480      0.000      -0.226      -0.088
region_southwest     -0.1289      0.035     -3.680      0.000      -0.198      -0.060
const                 7.0308      0.072     97.111      0.000       6.889       7.173
==============================================================================
Omnibus:                      463.941   Durbin-Watson:                   2.046
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1674.108
Skew:                           1.679   Prob(JB):                         0.00
Kurtosis:                       7.331   Cond. No.                         311.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [66]:

```python
# VIF = variance inflation factor= 1/ (1-R^2)
def VIF(df):
    import pandas as pd
    from sklearn.linear_model import LinearRegression

    #initialize dictionaries
    vif_dict, tolerance_dict = {}, {}

    #from input data for each exogenous veriable

    for col in df.drop(columns=['const']):
        y= df[col]
        x= df.drop(columns=[col])

        #extract r_squared from the fit

        r_squared = LinearRegression().fit(x, y).score(x, y)

        #calculate VIF
        if r_squared < 1: # Prevent division by zero runtime error
            vif = 1/(1- r_squared)
        else:
            vif = 100
        vif_dict[col] = vif

            #calculate tolerance
        tolerance = 1- r_squared
        tolerance_dict[col] = tolerance

            # generate the DataFrame to return
        df.output = pd.DataFrame({'VIF': vif_dict, 'Tolerance': tolerance_dict})

    return df_output.sort_values(by=['VIF'] , ascending=False)
VIF()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
C:\Users\TAWABC~1\AppData\Local\Temp/ipykernel_2528/3845429643.py in <module>
     32 
     33     return df_output.sort_values(by=['VIF'] , ascending=False)
---> 34 VIF(X)

NameError: name 'X' is not defined
```

In [ ]: