on9finduv

January 18, 2023

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import random
import seaborn as sns
```

[3]:
```
df=pd.read_csv("insurance.csv")
df
```

[3]:

	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 x 7 columns]

[4]:
```
df.isnull().sum()
```

[4]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
expenses    0
dtype: int64
```

[6]:
```
df.describe()
```

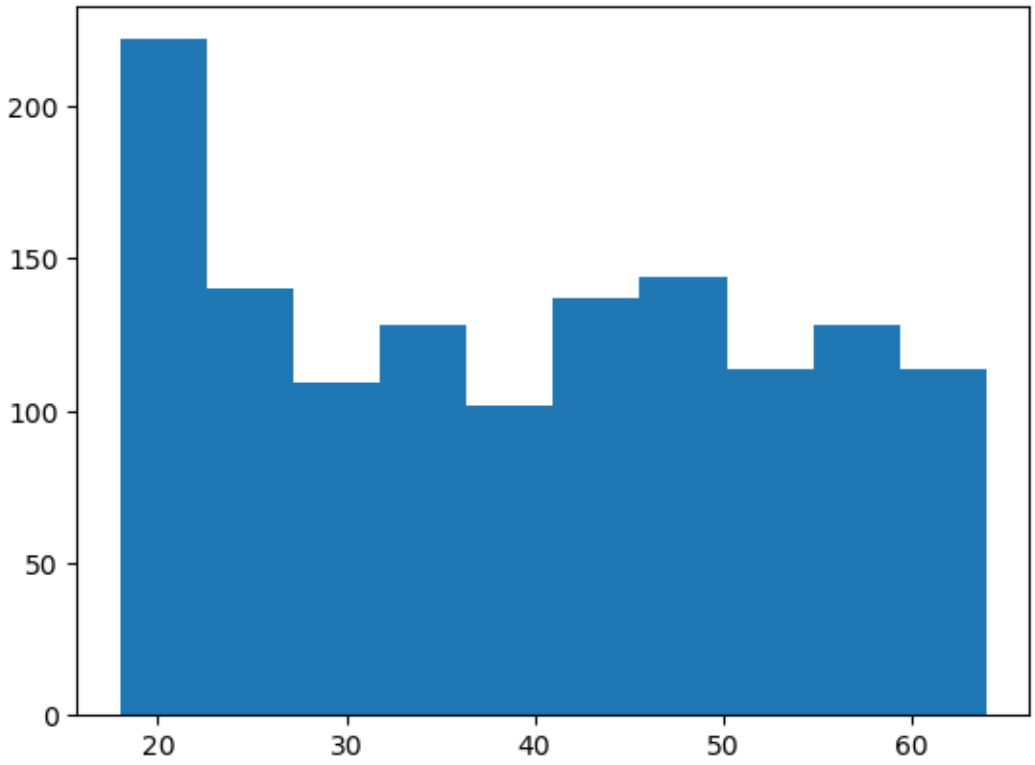
[6]:

	age	bmi	children	expenses
count	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	30.665471	1.094918	13270.422414
std	14.049960	6.098382	1.205493	12110.011240
min	18.000000	16.000000	0.000000	1121.870000
25%	27.000000	26.300000	0.000000	4740.287500
50%	39.000000	30.400000	1.000000	9382.030000
75%	51.000000	34.700000	2.000000	16639.915000
max	64.000000	53.100000	5.000000	63770.430000

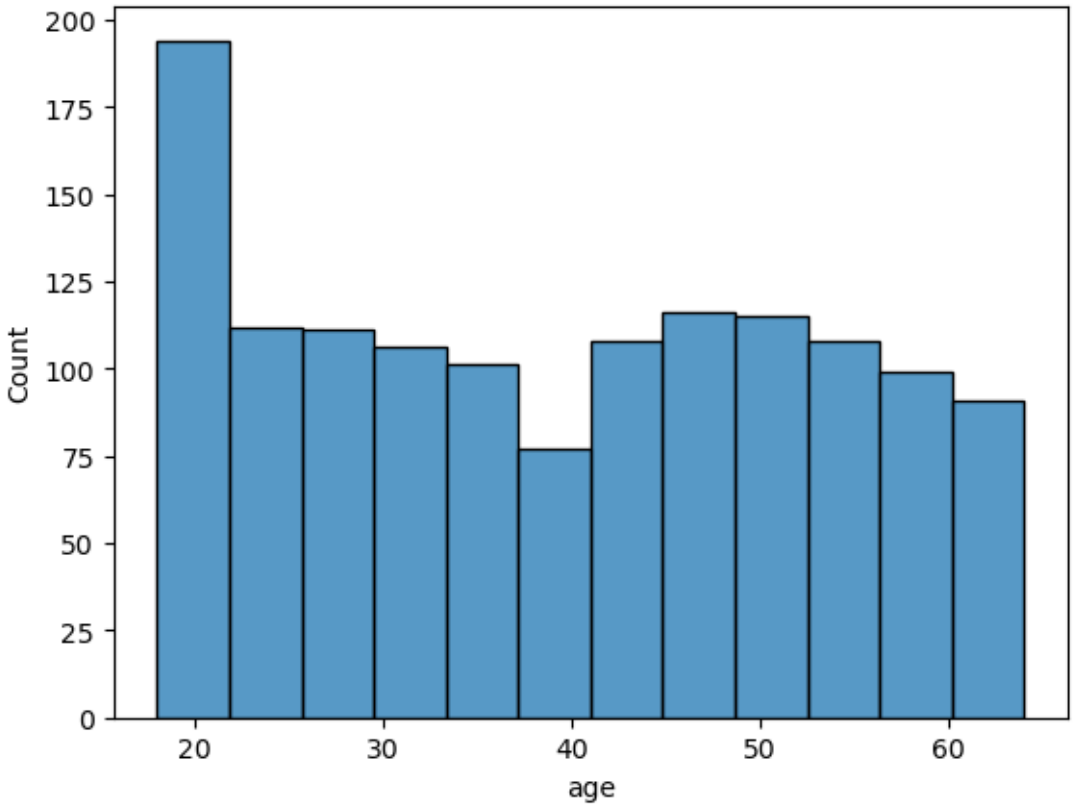
[7]:
```
df.keys()
```

[7]:
```
Index(['age', 'sex', 'bmi', 'children', 'smoker', 'region', 'expenses'],
dtype='object')
```

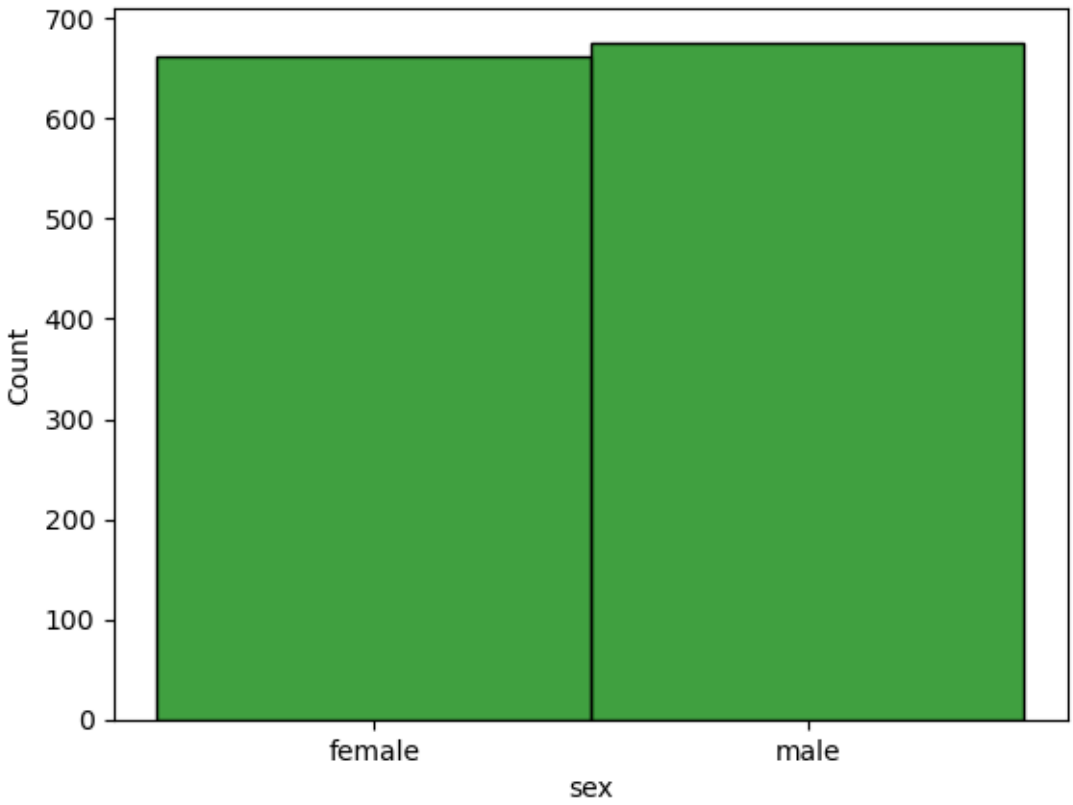
[9]:
```
plt.hist(df['age'])
plt.show()
```



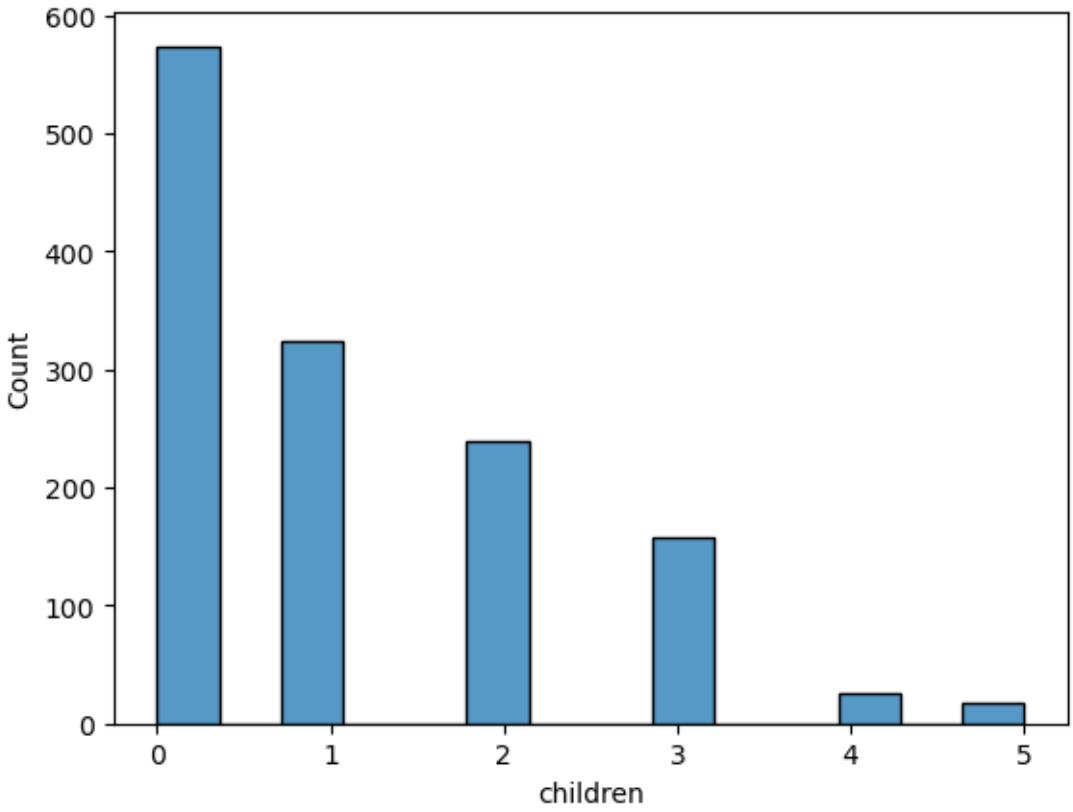
[10]:
```
sns.histplot(df['age'])
plt.show()
```



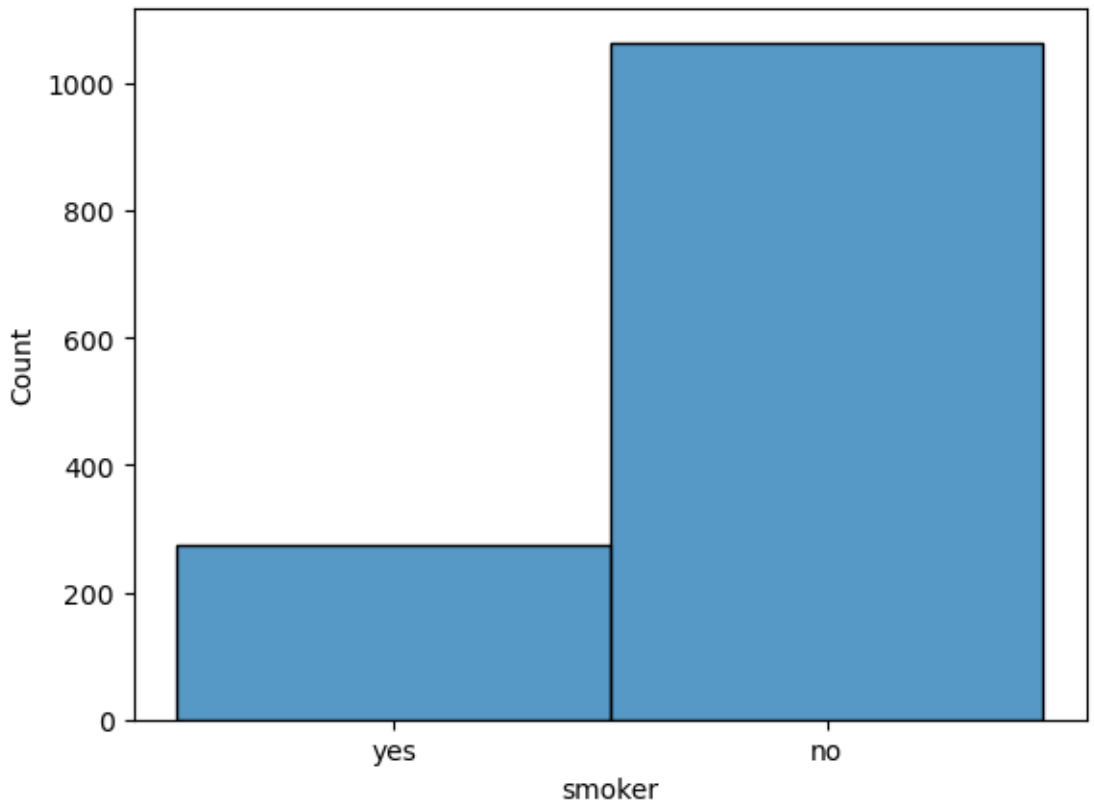
[12]:
```
sns.histplot(df['sex'],color='green')
plt.show()
```



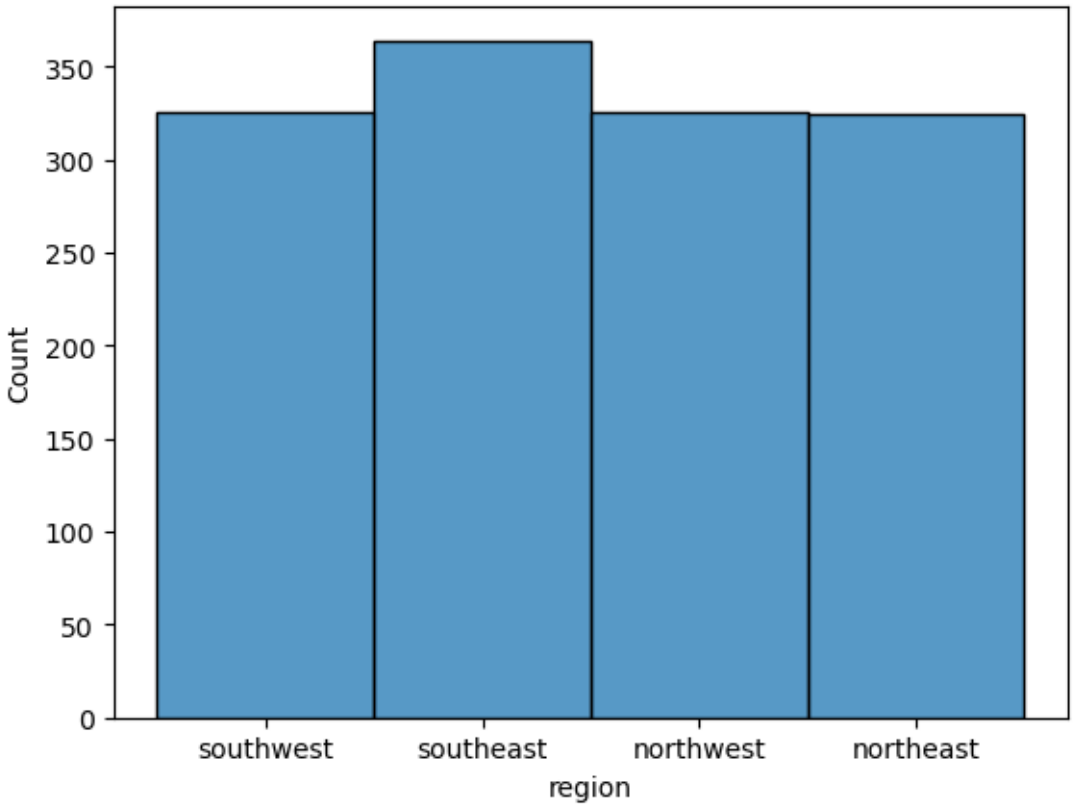
[13]:
```
sns.histplot(df['children'])
plt.show()
```



[14]:
```
sns.histplot(df['smoker'])
plt.show()
```

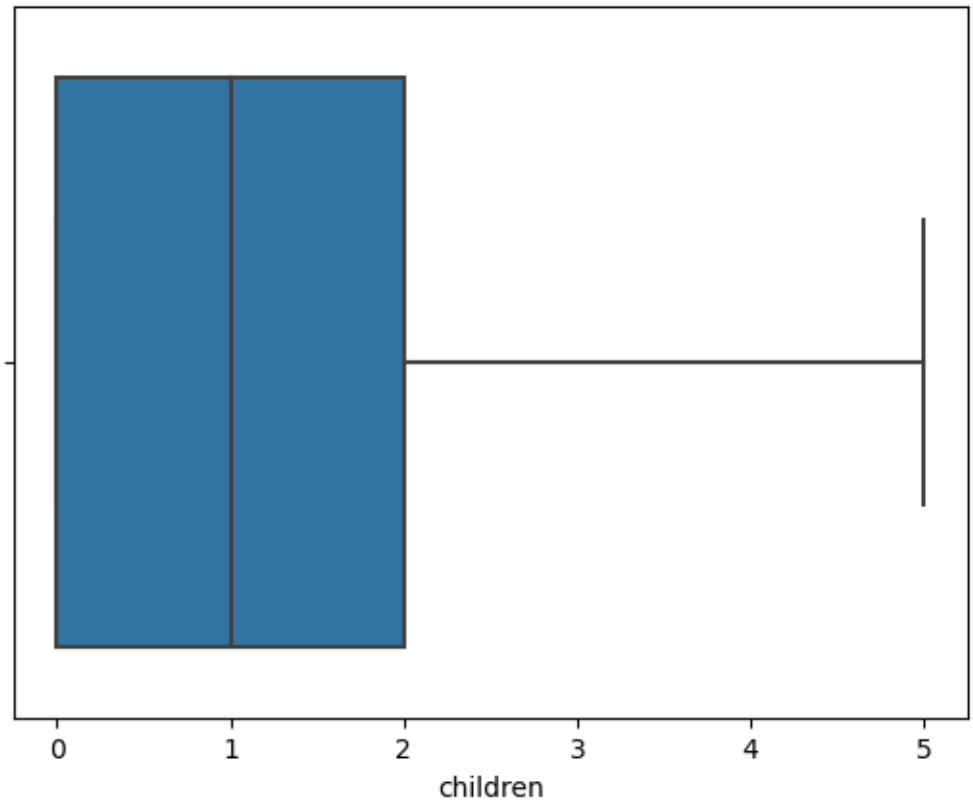


[15]:
```
sns.histplot(df['region'])
plt.show()
```



[17]:
```
sns.boxplot(df['children'])
```

[17]:
```
<AxesSubplot:xlabel='children'>
```



[22]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1338 non-null   int64
 1   sex       1338 non-null   object
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64
 4   smoker    1338 non-null   object
 5   region    1338 non-null   object
 6   expenses  1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

[23]:
```
df
```

[23]:

	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 x 7 columns]
```

[58]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

[59]:
```
df['new_region']=le.fit_transform(df['region'])
```

[60]:
```
#northeast=0,northwest=1,southeast=2,southwest=3
df
```

[60]:
```
      age     sex   bmi  children smoker     region  expenses  new_region
0      19  female  27.9         0    yes  southwest  16884.92           3
1      18    male  33.8         1     no  southeast   1725.55           2
2      28    male  33.0         3     no  southeast   4449.46           2
3      33    male  22.7         0     no  northwest  21984.47           1
4      32    male  28.9         0     no  northwest   3866.86           1
…   …      …     …       …     …         …       …          …
1333   50    male  31.0         3     no  northwest  10600.55           1
1334   18  female  31.9         0     no  northeast   2205.98           0
1335   18  female  36.9         0     no  southeast   1629.83           2
1336   21  female  25.8         0     no  southwest   2007.95           3
1337   61  female  29.1         0    yes  northwest  29141.36           1

[1338 rows x 8 columns]
```

[61]:
```
df1=df.drop('region',axis=1)
df1
```

[61]:
```
      age     sex   bmi  children smoker  expenses  new_region
0      19  female  27.9         0    yes  16884.92           3
1      18    male  33.8         1     no   1725.55           2
2      28    male  33.0         3     no   4449.46           2
3      33    male  22.7         0     no  21984.47           1
4      32    male  28.9         0     no   3866.86           1
…   …      …     …       …     …       …          …
1333   50    male  31.0         3     no  10600.55           1
```

```
1334   18  female  31.9         0     no   2205.98           0
1335   18  female  36.9         0     no   1629.83           2
1336   21  female  25.8         0     no   2007.95           3
1337   61  female  29.1         0    yes  29141.36           1

[1338 rows x 7 columns]
```

[62]:
```
df2=pd.get_dummies(df1,drop_first=True)
df2
```

[62]:

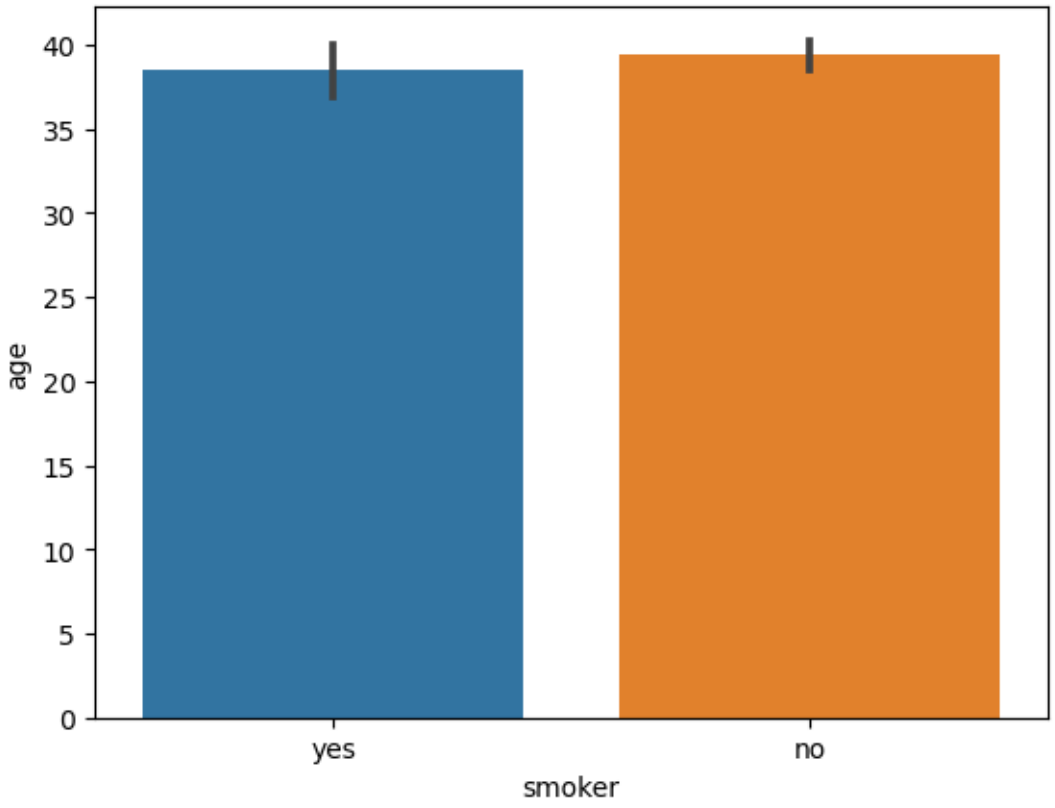
	age	bmi	children	expenses	new_region	sex_male	smoker_yes
0	19	27.9	0	16884.92	3	0	1
1	18	33.8	1	1725.55	2	1	0
2	28	33.0	3	4449.46	2	1	0
3	33	22.7	0	21984.47	1	1	0
4	32	28.9	0	3866.86	1	1	0
…	…	…	…	…	…	…	…
1333	50	31.0	3	10600.55	1	1	0
1334	18	31.9	0	2205.98	0	0	0
1335	18	36.9	0	1629.83	2	0	0
1336	21	25.8	0	2007.95	3	0	0
1337	61	29.1	0	29141.36	1	0	1

[1338 rows x 7 columns]

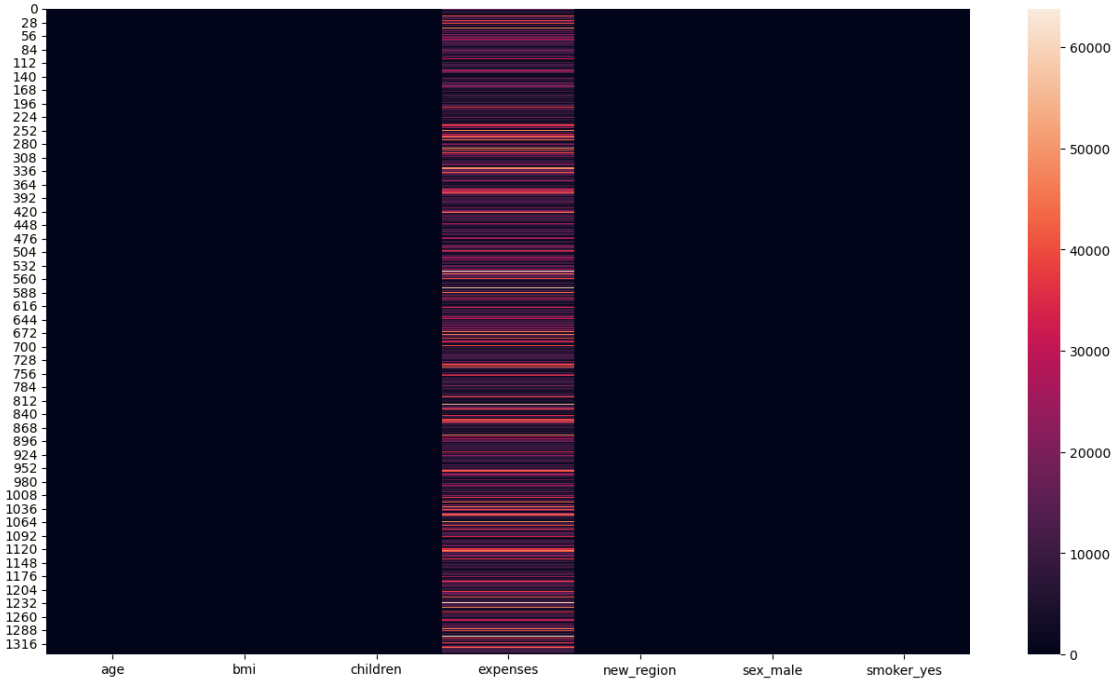
[67]:
```
sns.barplot(df['smoker'],df['age'])
```

```
E:\anaconda new\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass
the following variables as keyword args: x, y. From version 0.12, the only valid
positional argument will be `data`, and passing other arguments without an
explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

[67]:
```
<AxesSubplot:xlabel='smoker', ylabel='age'>
```



[69]:
```
plt.figure(figsize=(16,9))
sns.heatmap(df2)
plt.show()
```



[70]:
```
X=df2.drop('expenses',axis=1)
y=df2['expenses']
```

[73]:
```
print(X.shape)
print(y.shape)
```

```
(1338, 6)
(1338,)
```

[74]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.
  ↪2,random_state=51)
```

[75]:
```
X_train.shape
```

[75]: (1070, 6)

[76]:
```
X_test.shape
```

[76]: (268, 6)

[77]:
```
y_train.shape
```

[77]: (1070,)

[78]:
```
y_test.shape
```

[78]:
```
(268,)
```

[79]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(X_train,y_train)
lr.score(X_test,y_test)
```

[79]:
```
0.7492919486970459
```

[81]:
```
from sklearn.tree import DecisionTreeRegressor
dt=DecisionTreeRegressor()
dt.fit(X_train,y_train)
dt.score(X_test,y_test)
```

[81]:
```
0.7347942886707529
```

[82]:
```
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor()
rf.fit(X_train,y_train)
rf.score(X_test,y_test)
```

[82]:
```
0.842147293276142
```

[88]:
```
rf1=RandomForestRegressor(n_estimators=330)
rf1.fit(X_train,y_train)
rf1.score(X_test,y_test)
```

[88]:
```
0.8433661866146849
```

[92]:
```
from sklearn.neighbors import KNeighborsRegressor
knn=KNeighborsRegressor()
knn.fit(X_train,y_train)
knn.score(X_test,y_test)
```

[92]:
```
0.1637670441758985
```

[84]:
```
rf2=RandomForestRegressor(n_estimators=300)
rf2.fit(X_train,y_train)
rf2.score(X_test,y_test)
```

[84]:
```
0.8454383668671903
```

[96]:
```
y_pred=rf2.predict(X_test)
y_pred
```

```
[96]: array([14653.98856667, 45771.04853333,  9852.454     , 13814.42126667,
             39888.92673333, 10734.6162    , 10144.81863333, 12957.12866667,
              5967.25343333, 10025.10653333,  8735.5097    , 11654.1127    ,
              7361.66793333,  2207.89846667,  5183.54453333, 12845.6       ,
              5009.7557    ,  7093.51679492, 17571.1301    , 18918.5843    ,
              4106.9001    ,  7186.72753333, 44447.8181    , 13511.0781    ,
              7696.84353333, 16278.49946667, 14394.253     ,  2303.6591    ,
             24673.46773333, 14649.26183333,  2784.31996667, 20308.0392    ,
              2181.23316667,  3169.97206667,  7205.0291    , 14727.36636667,
              7084.44646667,  1855.31343333, 11895.43803333, 10574.03913333,
             13185.61883333,  1757.2239    ,  5367.08956667,  2332.13243333,
              8793.65363333, 18646.95433333, 11258.04606667, 43176.97516667,
              8633.38733333, 13642.23706667,  5842.13953333, 37924.0411    ,
              9822.53406667, 47799.26503333, 11702.369     , 35214.37653333,
              6186.8277    , 12897.71436667, 13215.70816667,  6742.19483333,
              9256.81093333,  6665.3763    , 26559.88833333,  3406.28813333,
             13204.73626667,  8238.74446667, 11201.42483333,  5814.86603333,
             14664.3356    ,  6673.54006667, 26458.62926667, 15832.9251    ,
              6614.64536667,  6899.8641    , 14375.91443333, 12579.44206667,
             14328.02906667, 44368.0729    ,  6079.84956667, 10332.77673333,
             10288.6431    , 14770.55176667,  8148.2221    ,  3351.82996667,
             11065.01136667, 48508.66796667,  4493.7067    , 11294.44753333,
             16436.4506    ,  6100.63646667,  4424.12126667, 15960.02003333,
              5870.61666667, 12810.16346667,  6532.59706667, 19094.83146667,
             41569.0122    , 12341.27636667, 18062.00316667,  7930.47766667,
              3572.48953333,  1848.54583333, 43961.8789    ,  6486.99153333,
              6221.28503333,  7663.56393333,  6749.1922    ,  7124.77803333,
             10003.07613333, 38989.307     ,  7765.7334    ,  8614.87863333,
              7280.18513333,  6290.73753333, 13105.53223333,  9789.5398    ,
              3052.61263333, 14097.575     , 14513.15896667, 29009.1572    ,
             46377.38446667,  5487.1972    , 47003.63423333,  6576.3241    ,
              1686.54766667, 19985.854     ,  4973.8016    , 35933.2477    ,
             10186.5971    ,  3257.69316667,  1981.19693333,  5895.17563333,
              4278.03736667,  6755.71603333,  3696.98336667, 23737.21883333,
              5718.11647778,  5518.7394    ,  5657.12276667,  6012.03213333,
             40863.05466667,  1831.5616    , 42817.24993333,  2887.58370667,
              9764.24783333, 11836.096     , 11557.66833333, 22134.43726667,
             13101.97633333, 16492.3318    , 46052.2593    , 46604.0714    ,
              9692.41413333, 13438.20123333,  1423.4901    , 12406.21676667,
             26704.02146667, 13226.38853333,  4120.8696    ,  2512.33133333,
             43620.39096667, 20311.92553333, 20741.56796667,  6371.79196667,
              7198.33283333, 12102.2409    , 13822.77033333, 14625.7675    ,
             35995.23786667,  4654.5486    ,  2287.04853333,  4431.7548    ,
             12621.97953333,  3668.5705    , 28042.59353333, 26888.18336667,
              6643.58926667, 40277.97456667, 13254.27976667,  5570.53293333,
              4867.58406667,  1375.26523333, 11299.66653333,  9859.5841    ,
             12118.05663333,  6106.35423333, 12255.93283333,  6192.1275    ,
```

```
 5256.08093333, 11949.02873333,  8941.0316    , 24239.50836667,
 6824.45843333, 19809.17506667, 13042.65746667,  6587.7404    ,
 2995.86      , 21312.01336667,  9171.69646667, 10802.65306667,
 6584.6542    , 39145.69293333, 40545.63813333, 12686.54836667,
15670.64683333,  2110.58206667, 14540.71323333, 13785.08693333,
 6513.19933333,  1771.8699    ,  4454.7928    , 44240.5207    ,
13915.24006667, 13571.82113333,  4906.7945    , 19557.50113333,
26149.5249    , 45251.98726667, 37488.13983333,  5165.36806667,
11751.9097    ,  5070.15113333, 10712.84473333,  4587.01      ,
 6669.9172    ,  5367.67566667,  7782.2201    ,  5239.81336667,
 7200.02116667,  5344.6368    ,  4154.0208    ,  2366.56066667,
 7386.78843333,  2258.6074    ,  3471.0843    , 15474.57513333,
23544.44913333,  9504.77666667,  7331.45653333, 14402.43676667,
 6395.86806667, 10996.17413333,  6071.7437    ,  1667.75226667,
11085.39953333, 25146.1288    ,  9978.87096667,  4011.00596667,
17490.02343333, 14920.472     ,  1576.75616667,  4429.44073333,
 1585.44923333, 11165.01746667, 44480.36453333, 40160.26693333,
 9216.6265    , 20258.37776667,  2825.04783333,  8649.1141    ,
 3712.62333333,  7702.1621    , 46814.45246667, 12053.57533333,
27634.15466667, 33915.1556    , 46156.32843333, 16532.58563333])
```

[100]:
```
y_test
```

[100]:
```
151      7789.64
1146    52590.83
1305     2464.62
392      8964.06
123     39556.49
           …
209      6610.11
1250    18648.42
503     32548.34
668     45710.21
366     13430.27
Name: expenses, Length: 268, dtype: float64
```

Random forest regressor gives 84% accuracy

[ ]:
```

```