

STROKE PREDICTION USING HEALTH-CARE DATASET

IMPORTING LIBRARIES

```
In [1]: 1 import pandas as pd
        2 import numpy as np
        3 import matplotlib.pyplot as plt
        4 import seaborn as sns
```

LOADING THE DATASET

```
In [2]: 1 df =pd.read_csv('C:/Users/hp/OneDrive/Desktop/healthcare-dataset-stroke-d
        2
```

DATA PRE-PROCESSING

```
In [3]: 1 df.head()
```

```
Out[3]:
```


	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type
0	9046	Male	67.0	0	1	Yes	Private	Urban
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural
2	31112	Male	80.0	0	1	Yes	Private	Rural
3	60182	Female	49.0	0	0	Yes	Private	Urban
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural



```
In [7]: 1 df.tail()
```

```
Out[7]:
```

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type
5105	18234	Female	80.0	1	0	Yes	Private	Urban
5106	44873	Female	81.0	0	0	Yes	Self-employed	Urban
5107	19723	Female	35.0	0	0	Yes	Self-employed	Rural
5108	37544	Male	51.0	0	0	Yes	Private	Rural
5109	44679	Female	44.0	0	0	Yes	Govt_job	Urban



```
In [7]: 1 df.shape
```

```
Out[7]: (5110, 12)
```

```
In [8]: 1 df.columns
```

```
Out[8]: Index(['id', 'gender', 'age', 'hypertension', 'heart_disease', 'ever_married',  
              'work_type', 'Residence_type', 'avg_glucose_level', 'bmi',  
              'smoking_status', 'stroke'],  
             dtype='object')
```

```
In [12]: 1 df.duplicated().sum()
```

```
Out[12]: 0
```

NULL VALUES TREATMENT

```
In [13]: 1 df.isnull().sum()
```

```
Out[13]: id                0  
gender                0  
age                  0  
hypertension         0  
heart_disease        0  
ever_married         0  
work_type            0  
Residence_type       0  
avg_glucose_level    0  
bmi                 201  
smoking_status       0  
stroke               0  
dtype: int64
```

In [14]: 1 df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5110 entries, 0 to 5109
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype  
---  -
0   id                    5110 non-null  int64  
1   gender                5110 non-null  object  
2   age                  5110 non-null  float64 
3   hypertension          5110 non-null  int64  
4   heart_disease         5110 non-null  int64  
5   ever_married          5110 non-null  object  
6   work_type             5110 non-null  object  
7   Residence_type        5110 non-null  object  
8   avg_glucose_level     5110 non-null  float64 
9   bmi                  4909 non-null  float64 
10  smoking_status        5110 non-null  object  
11  stroke                5110 non-null  int64  
dtypes: float64(3), int64(4), object(5)
memory usage: 479.2+ KB
```

In [15]: 1 df.describe()

Out[15]:

	id	age	hypertension	heart_disease	avg_glucose_level	bmi
count	5110.000000	5110.000000	5110.000000	5110.000000	5110.000000	4909.000000
mean	36517.829354	43.226614	0.097456	0.054012	106.147677	28.893237
std	21161.721625	22.612647	0.296607	0.226063	45.283560	7.854067
min	67.000000	0.080000	0.000000	0.000000	55.120000	10.300000
25%	17741.250000	25.000000	0.000000	0.000000	77.245000	23.500000
50%	36932.000000	45.000000	0.000000	0.000000	91.885000	28.100000
75%	54682.000000	61.000000	0.000000	0.000000	114.090000	33.100000
max	72940.000000	82.000000	1.000000	1.000000	271.740000	97.600000

In [17]: 1 df.nunique()

Out[17]:

id	5110
gender	3
age	104
hypertension	2
heart_disease	2
ever_married	2
work_type	5
Residence_type	2
avg_glucose_level	3979
bmi	418
smoking_status	4
stroke	2

dtype: int64

```
In [4]: 1 df['bmi'].fillna(df['bmi'].mean(),inplace=True)
        2 df.isnull().sum()
```

```
Out[4]: id                0
gender                0
age                  0
hypertension          0
heart_disease          0
ever_married          0
work_type              0
Residence_type        0
avg_glucose_level     0
bmi                   0
smoking_status         0
stroke                0
dtype: int64
```

EXPLORATORY DATA ANALYSIS

```
In [5]: 1 df.drop('id',axis=1,inplace=True)
        2 df
```

```
Out[5]:
```

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_
0	Male	67.0	0	1	Yes	Private	Urban	
1	Female	61.0	0	0	Yes	Self-employed	Rural	
2	Male	80.0	0	1	Yes	Private	Rural	
3	Female	49.0	0	0	Yes	Private	Urban	
4	Female	79.0	1	0	Yes	Self-employed	Rural	
...
5105	Female	80.0	1	0	Yes	Private	Urban	
5106	Female	81.0	0	0	Yes	Self-employed	Urban	
5107	Female	35.0	0	0	Yes	Self-employed	Rural	
5108	Male	51.0	0	0	Yes	Private	Rural	
5109	Female	44.0	0	0	Yes	Govt_job	Urban	

5110 rows × 11 columns



```
In [6]: 1 df.drop('ever_married',axis=1,inplace=True)
        2 df
```

```
Out[6]:
```

	gender	age	hypertension	heart_disease	work_type	Residence_type	avg_glucose_level
0	Male	67.0	0	1	Private	Urban	228.69
1	Female	61.0	0	0	Self-employed	Rural	202.21
2	Male	80.0	0	1	Private	Rural	105.92
3	Female	49.0	0	0	Private	Urban	171.23
4	Female	79.0	1	0	Self-employed	Rural	174.12
...
5105	Female	80.0	1	0	Private	Urban	83.75
5106	Female	81.0	0	0	Self-employed	Urban	125.20
5107	Female	35.0	0	0	Self-employed	Rural	82.99
5108	Male	51.0	0	0	Private	Rural	166.29
5109	Female	44.0	0	0	Govt_job	Urban	85.28

5110 rows × 10 columns



Label Encoding

```
In [7]: 1 from sklearn.preprocessing import LabelEncoder
        2 enc=LabelEncoder()
```

```
In [8]: 1 Residence_type=enc.fit_transform(df['Residence_type'])
        2 Residence_type
```

```
Out[8]: array([1, 0, 0, ..., 0, 0, 1])
```

```
In [9]: 1 gender=enc.fit_transform(df['gender'])
        2 gender
```

```
Out[9]: array([1, 0, 1, ..., 0, 1, 0])
```

```
In [10]: 1 smoking_status=enc.fit_transform(df['smoking_status'])
```

```
In [18]: 1 work_type=enc.fit_transform(df['work_type'])
        2
```

```
In [12]: 1 Residence_type=enc.fit_transform(df['Residence_type'])
```

```
In [19]: 1 df['work_type']=work_type
2 df['Residence_type']=Residence_type
3 df['smoking_status']=smoking_status
4 df['gender']=gender
```

```
In [38]: 1 df["Residence_type"] = df["Residence_type"].replace("Urban", 1)
2 df["Residence_type"] = df["Residence_type"].replace("Rural", 0)
3
```

```
In [39]: 1 df.head()
```

```
Out[39]:
```

	gender	age	hypertension	heart_disease	work_type	Residence_type	avg_glucose_level	
0	1	67.0	0	1	2	1	228.69	36
1	0	61.0	0	0	3	0	202.21	28
2	1	80.0	0	1	2	0	105.92	32
3	0	49.0	0	0	2	1	171.23	34
4	0	79.0	1	0	3	0	174.12	24

```
In [40]: 1 df["smoking_status"] = df["smoking_status"].replace("never smoked", 1)
2 df["smoking_status"] = df["smoking_status"].replace("Unknown", 0)
3 df["smoking_status"] = df["smoking_status"].replace("formerly smoked", 2)
4 df["smoking_status"] = df["smoking_status"].replace("smokes", 3)
5
```

In [21]:

```
1 df
```

Out[21]:

	gender	age	hypertension	heart_disease	work_type	Residence_type	avg_glucose_level
0	1	67.0	0	1	2	Urban	228.69
1	0	61.0	0	0	3	Rural	202.21
2	1	80.0	0	1	2	Rural	105.92
3	0	49.0	0	0	2	Urban	171.23
4	0	79.0	1	0	3	Rural	174.12
...
5105	0	80.0	1	0	2	Urban	83.75
5106	0	81.0	0	0	3	Urban	125.20
5107	0	35.0	0	0	3	Rural	82.99
5108	1	51.0	0	0	2	Rural	166.29
5109	0	44.0	0	0	0	Urban	85.28

5110 rows × 12 columns



In [22]:

```
1 df['smoking_status'].value_counts()
```

Out[22]:

```
1    1892
0    1544
2     885
3     789
Name: smoking_status, dtype: int64
```

In [41]:

```
1 df['Residence_type'].value_counts()
2
```

Out[41]:

```
1    2596
0    2514
Name: Residence_type, dtype: int64
```

HANDLING NULL VALUES

In []:

```
1
```

In [86]:

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5110 entries, 0 to 5109
Data columns (total 13 columns):
#   Column                Non-Null Count  Dtype  
---  -
0   gender                 5110 non-null  int64  
1   age                   5110 non-null  float64
2   hypertension           5110 non-null  int64  
3   heart_disease         5110 non-null  int64  
4   work_type             5110 non-null  int64  
5   Residence_type        5110 non-null  int64  
6   avg_glucose_level     5110 non-null  float64
7   bmi                   5110 non-null  float64
8   smoking_status        5110 non-null  int64  
9   stroke                 5110 non-null  int64  
10  ever_married           5110 non-null  int32  
11  Residence type         5110 non-null  int32  
12  smoking status        5110 non-null  int32  
dtypes: float64(3), int32(3), int64(7)
memory usage: 459.2 KB
```

In [42]:

```
1 X=df.drop('stroke',axis=1)
```

In [43]:

```
1 X.head()
```

Out[43]:

	gender	age	hypertension	heart_disease	work_type	Residence_type	avg_glucose_level	
0	1	67.0	0	1	2	1	228.69	36
1	0	61.0	0	0	3	0	202.21	28
2	1	80.0	0	1	2	0	105.92	32
3	0	49.0	0	0	2	1	171.23	34
4	0	79.0	1	0	3	0	174.12	24

In [28]:

```
1 Y=df['stroke']
2 Y.head()
```

Out[28]:

```
0    1
1    1
2    1
3    1
4    1
Name: stroke, dtype: int64
```


SPLITTING DATA FOR TRAIN AND TEST


```
In [44]: 1 from sklearn.model_selection import train_test_split
        2 X_train, X_test, Y_train, Y_test=train_test_split(X,Y,test_size=0.2,random
```

```
In [45]: 1 X_train
```

Out[45]:

	gender	age	hypertension	heart_disease	work_type	Residence_type	avg_glucose_level
2285	1	49.0	0	0	2	0	79.64
4733	1	67.0	0	0	2	0	83.16
3905	1	78.0	0	0	2	1	208.85
4700	1	47.0	0	0	2	0	110.14
4939	0	59.0	0	0	2	1	71.08
...
1180	0	62.0	0	0	2	0	82.57
3441	0	59.0	0	0	3	1	90.06
1344	1	47.0	0	0	2	0	86.37
4623	1	25.0	0	0	0	1	166.38
1289	0	80.0	0	0	3	0	72.61

4088 rows × 11 columns



```
In [46]: 1 Y_train
```

Out[46]: 2285 0
4733 0
3905 0
4700 0
4939 0
..
1180 0
3441 0
1344 0
4623 0
1289 0
Name: stroke, Length: 4088, dtype: int64

In [47]:

1 X_test

Out[47]:

	gender	age	hypertension	heart_disease	work_type	Residence_type	avg_glucose_level
2413	0	58.00	0	0	2	0	100.42
1141	1	57.00	0	0	2	0	90.06
146	1	65.00	0	0	3	1	68.43
3883	0	1.64	0	0	4	1	69.89
1044	0	79.00	0	0	0	1	93.89
...
2261	1	59.00	0	0	2	1	60.35
4712	1	57.00	0	0	2	1	93.04
4971	0	63.00	0	0	2	1	57.06
2224	1	57.00	0	0	2	0	76.28
4825	0	14.00	0	0	4	1	71.80

1022 rows × 11 columns



In [48]:

1 Y_test

Out[48]:

```
2413    0
1141    0
146     1
3883    0
1044    0
..
2261    0
4712    0
4971    0
2224    0
4825    0
```


Name: stroke, Length: 1022, dtype: int64

NORMALIZE

In [34]: 1 df.describe()

Out[34]:

	gender	age	hypertension	heart_disease	work_type	avg_glucose_level	
count	5110.000000	5110.000000	5110.000000	5110.000000	5110.000000	5110.000000	5
mean	0.414286	43.226614	0.097456	0.054012	2.167710	106.147677	
std	0.493044	22.612647	0.296607	0.226063	1.090293	45.283560	
min	0.000000	0.080000	0.000000	0.000000	0.000000	55.120000	
25%	0.000000	25.000000	0.000000	0.000000	2.000000	77.245000	
50%	0.000000	45.000000	0.000000	0.000000	2.000000	91.885000	
75%	1.000000	61.000000	0.000000	0.000000	3.000000	114.090000	
max	2.000000	82.000000	1.000000	1.000000	4.000000	271.740000	



In [49]: 1 from sklearn.preprocessing import StandardScaler
2 std = StandardScaler()

In [50]: 1 X_train_std = std.fit_transform(X_train)
2 X_test_std = std.transform(X_test)

In [51]: 1 X_train_std

Out[51]: array([[1.19359699, 0.2521852 , -0.33069968, ..., 1.75888007,
-1.01777039, 1.51158251],
[1.19359699, 1.04686385, -0.33069968, ..., 0.79061967,
-1.01777039, -0.35191245],
[1.19359699, 1.5325008 , -0.33069968, ..., 0.79061967,
 0.98253988, -0.35191245],
...,
[1.19359699, 0.16388757, -0.33069968, ..., 1.75888007,
-1.01777039, 1.51158251],
[1.19359699, -0.80738634, -0.33069968, ..., -0.17764073,
 0.98253988, 0.57983503],
[-0.83780372, 1.62079843, -0.33069968, ..., -0.17764073,
-1.01777039, 0.57983503]])

```
In [52]: 1 X_test_std
```

```
Out[52]: array([[ -0.83780372,  0.64952452, -0.33069968, ...,  1.75888007,
                -1.01777039,  1.51158251],
                [ 1.19359699,  0.60537571, -0.33069968, ..., -1.14590113,
                -1.01777039, -1.28365994],
                [ 1.19359699,  0.95856622, -0.33069968, ...,  0.79061967,
                0.98253988, -0.35191245],
                ...,
                [-0.83780372,  0.87026859, -0.33069968, ..., -0.17764073,
                0.98253988,  0.57983503],
                [ 1.19359699,  0.60537571, -0.33069968, ...,  0.79061967,
                -1.01777039, -0.35191245],
                [-0.83780372, -1.29302329, -0.33069968, ..., -1.14590113,
                0.98253988, -1.28365994]])
```

LOGISTIC REGRESSION

```
In [53]: 1 from sklearn.linear_model import LogisticRegression
        2 lr=LogisticRegression()
```

```
In [54]: 1 lr.fit(X_train_std,Y_train)
```

```
Out[54]: LogisticRegression()
```

```
In [55]: 1 Y_pred_lr=lr.predict(X_test_std)
```

```
In [56]: 1 Y_pred_lr[:5]
```

```
Out[56]: array([0, 0, 0, 0, 0], dtype=int64)
```

EVALUATION FOR LOGISTIC REGRESSION

```
In [57]: 1 from sklearn.metrics import accuracy_score
```

```
In [58]: 1 ac_lr=accuracy_score(Y_test,Y_pred_lr)
```

```
In [59]: 1 ac_lr*100
```

```
Out[59]: 93.9334637964775
```

KNN: KNEIGHBORS CLASSIFIER

```
In [60]: 1 from sklearn.neighbors import KNeighborsClassifier
2 knn=KNeighborsClassifier(n_neighbors=5,p=2)
3 knn.fit(X_train, Y_train)
```

Out[60]: KNeighborsClassifier()

```
In [61]: 1 # predictions
2 Y_pred=knn.predict(X_test)
3 Y_pred
```

C:\Users\hp\anaconda3\lib\site-packages\sklearn\neighbors_classification.py: 228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will change: the default value of `keepdims` will become False, the `axis` over which the statistic is taken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to avoid this warning.

```
mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

Out[61]: array([0, 0, 0, ..., 0, 0, 0], dtype=int64)

```
In [62]: 1 Y_test
```

Out[62]: 2413 0
1141 0
146 1
3883 0
1044 0
..
2261 0
4712 0
4971 0
2224 0
4825 0
Name: stroke, Length: 1022, dtype: int64

EVALUATION FOR KNN

```
In [63]: 1 from sklearn.metrics import confusion_matrix
2 cm=confusion_matrix(Y_test, Y_pred)
3 cm
```

Out[63]: array([[957, 3],
[61, 1]], dtype=int64)

```
In [64]: 1 from sklearn.metrics import classification_report
2 print(classification_report(Y_test, Y_pred))
```

	precision	recall	f1-score	support
0	0.94	1.00	0.97	960
1	0.25	0.02	0.03	62
accuracy			0.94	1022
macro avg	0.60	0.51	0.50	1022
weighted avg	0.90	0.94	0.91	1022

```
In [ ]: 1 ac_lr=accuracy_score(Y_test,Y_pred_lr)
```

VISUALIZATIONS

```
In [4]: 1 df1=pd.read_csv('C:/Users/hp/OneDrive/Desktop/healthcare-dataset-stroke-d
2 df1.head()
```

Out[4]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type
0	9046	Male	67.0	0	1	Yes	Private	Urban
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural
2	31112	Male	80.0	0	1	Yes	Private	Rural
3	60182	Female	49.0	0	0	Yes	Private	Urban
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural

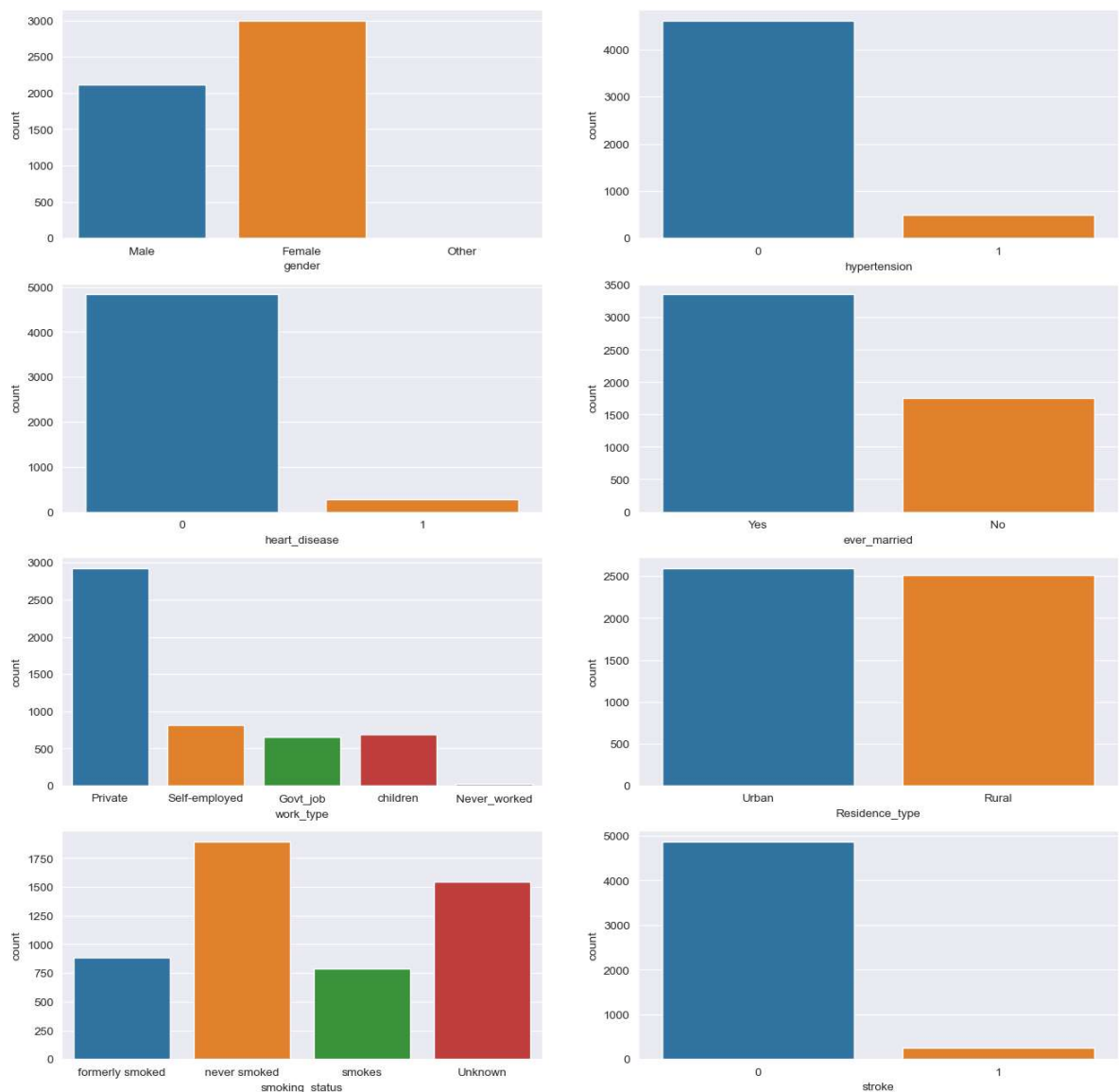


BAR-GRAPHS

In [135]:

```
1 fig,axes = plt.subplots(4,2,figsize = (16,16))
2 sns.set_style('darkgrid')
3 fig.suptitle("Count plot for various categorical features")
4
5 sns.countplot(ax=axes[0,0],data=df1,x='gender')
6 sns.countplot(ax=axes[0,1],data=df1,x='hypertension')
7 sns.countplot(ax=axes[1,0],data=df1,x='heart_disease')
8 sns.countplot(ax=axes[1,1],data=df1,x='ever_married')
9 sns.countplot(ax=axes[2,0],data=df1,x='work_type')
10 sns.countplot(ax=axes[2,1],data=df1,x='Residence_type')
11 sns.countplot(ax=axes[3,0],data=df1,x='smoking_status')
12 sns.countplot(ax=axes[3,1],data=df1,x='stroke')
13 plt.show()
```

Count plot for various categorical features



AGE

```
In [136]: 1 df1.groupby('gender').mean()[['age', 'stroke']]
```

```
Out[136]:
```

	age	stroke
gender		
Female	43.757395	0.047094
Male	42.483385	0.051064
Other	26.000000	0.000000

Chances of stroke is more in men than women

WORK TYPE

```
In [139]: 1 df1['work_type'].unique()
```

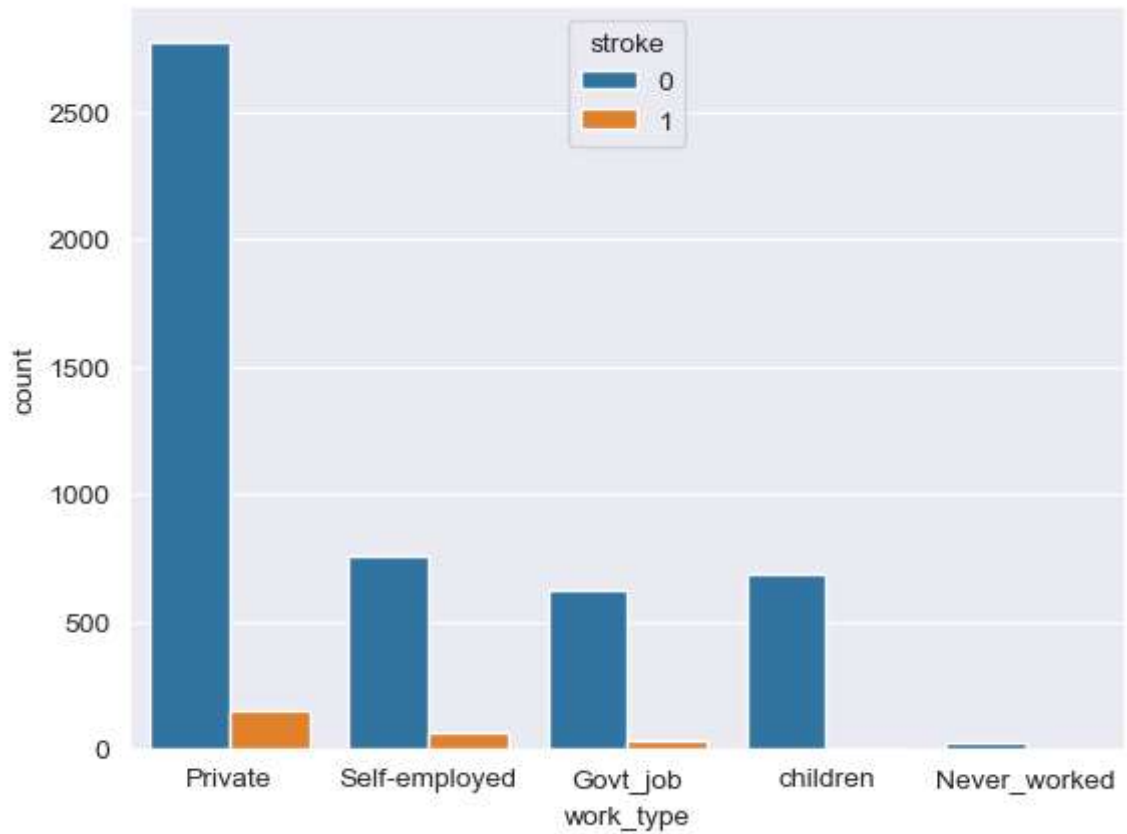
```
Out[139]: array(['Private', 'Self-employed', 'Govt_job', 'children', 'Never_worked'],  
              dtype=object)
```

```
In [140]: 1 df1['work_type'].value_counts()
```

```
Out[140]: Private          2925  
Self-employed          819  
children              687  
Govt_job              657  
Never_worked           22  
Name: work_type, dtype: int64
```

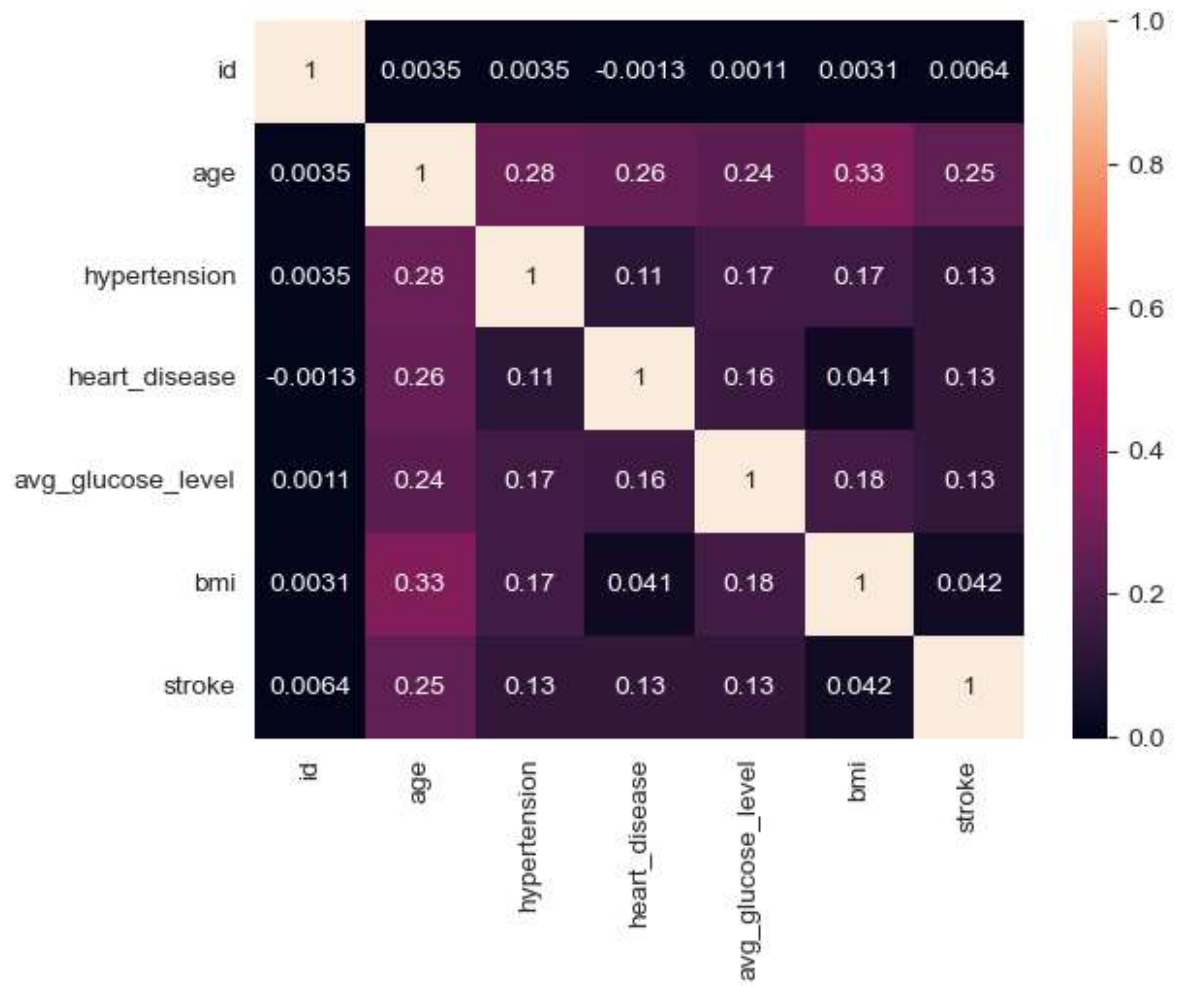


```
In [142]: 1 sns.countplot(data=df1,x='work_type',hue='stroke')
          2 plt.show()
```



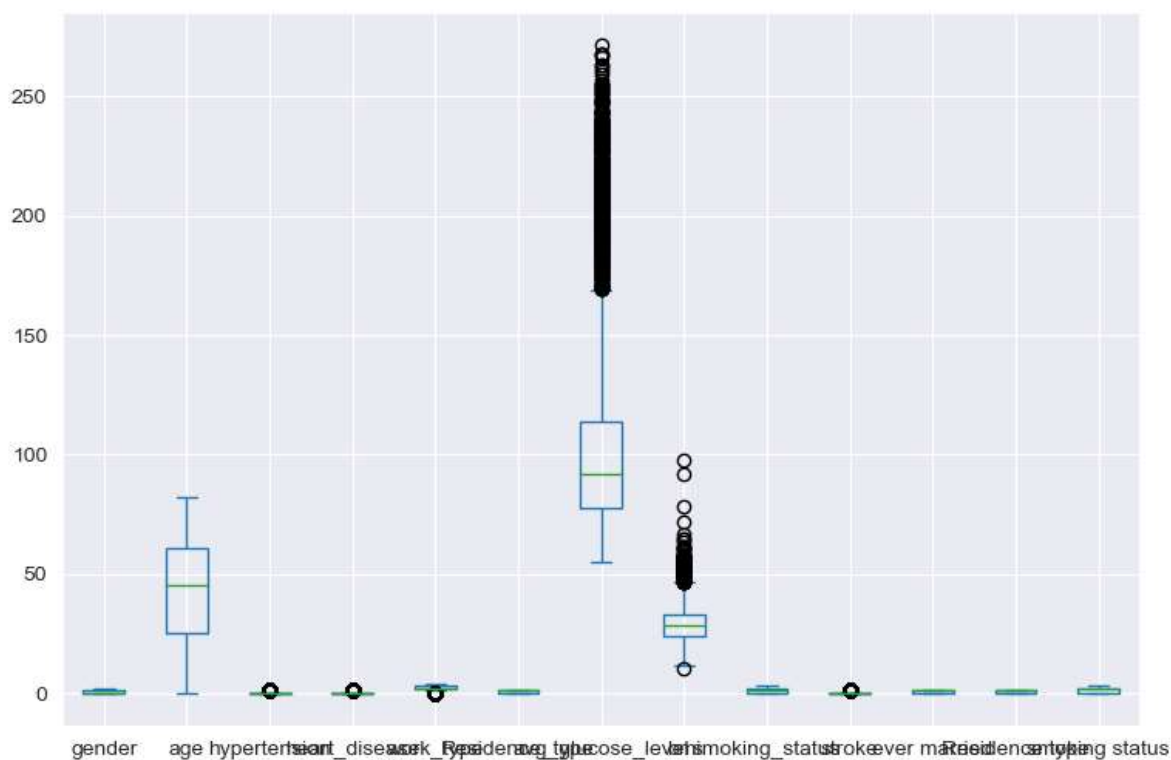
HEATMAP

```
In [143]: 1 sns.heatmap(df1.corr(),annot=True)
          2 plt.show()
```



BOX-PLOT

```
In [147]: 1 df.plot(kind='box',figsize=(9,6))
          2 plt.show()
```



```
In [154]: 1 df1.head()
```

Out[154]:

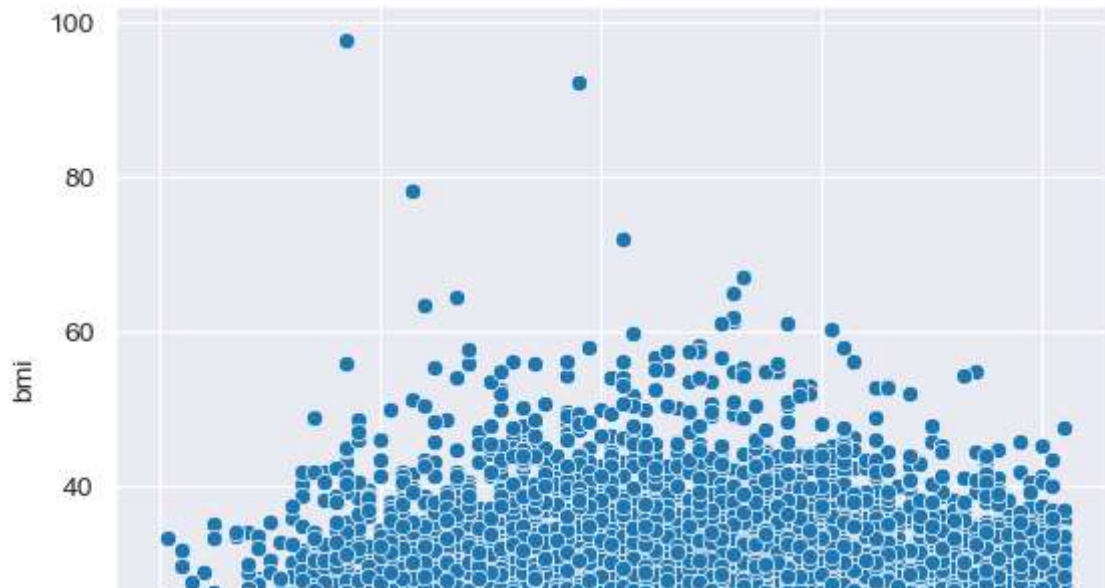
	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type
0	9046	Male	67.0	0	1	Yes	Private	Urban
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural
2	31112	Male	80.0	0	1	Yes	Private	Rural
3	60182	Female	49.0	0	0	Yes	Private	Urban
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural



SCATTER-PLOT

```
In [156]: 1 sns.scatterplot(x='age',y='bmi',data=df1)
```

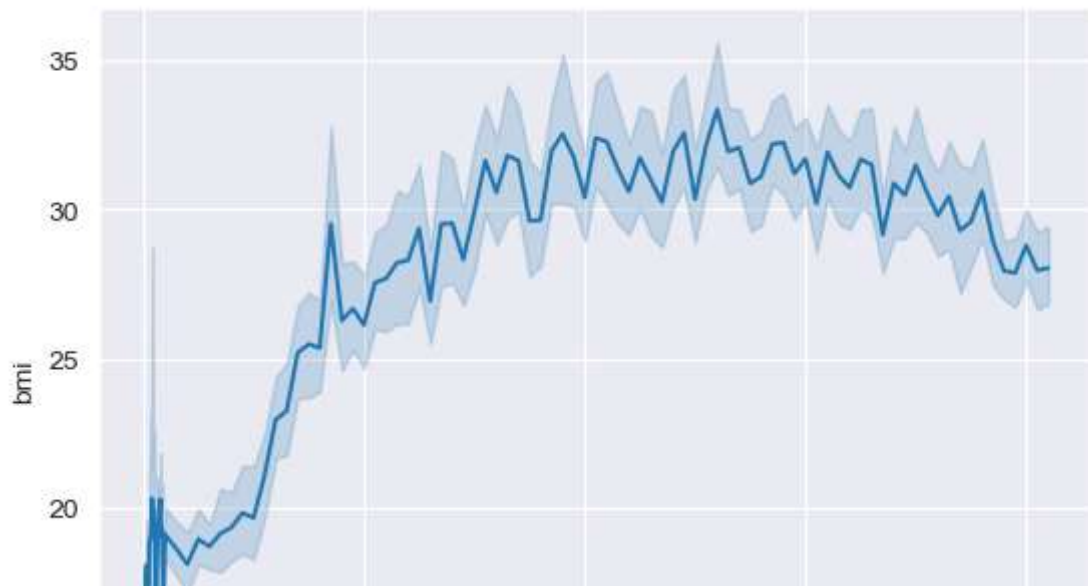
```
Out[156]: <AxesSubplot:xlabel='age', ylabel='bmi'>
```



LINE-PLOT

```
In [157]: 1 sns.lineplot(x='age',y='bmi',data=df1)
```

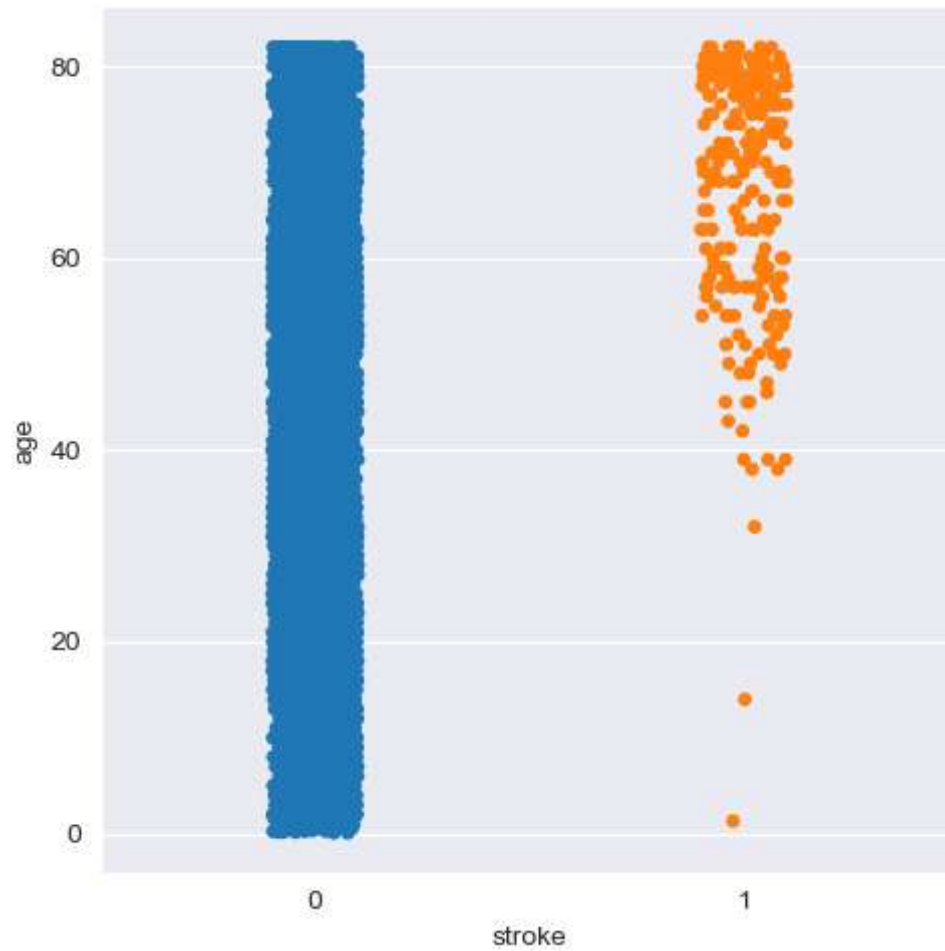
```
Out[157]: <AxesSubplot:xlabel='age', ylabel='bmi'>
```



CAT-PLOT

```
In [161]: 1 sns.catplot(y='age',x='stroke',data=df1)
```

```
Out[161]: <seaborn.axisgrid.FacetGrid at 0x196fc6c3880>
```

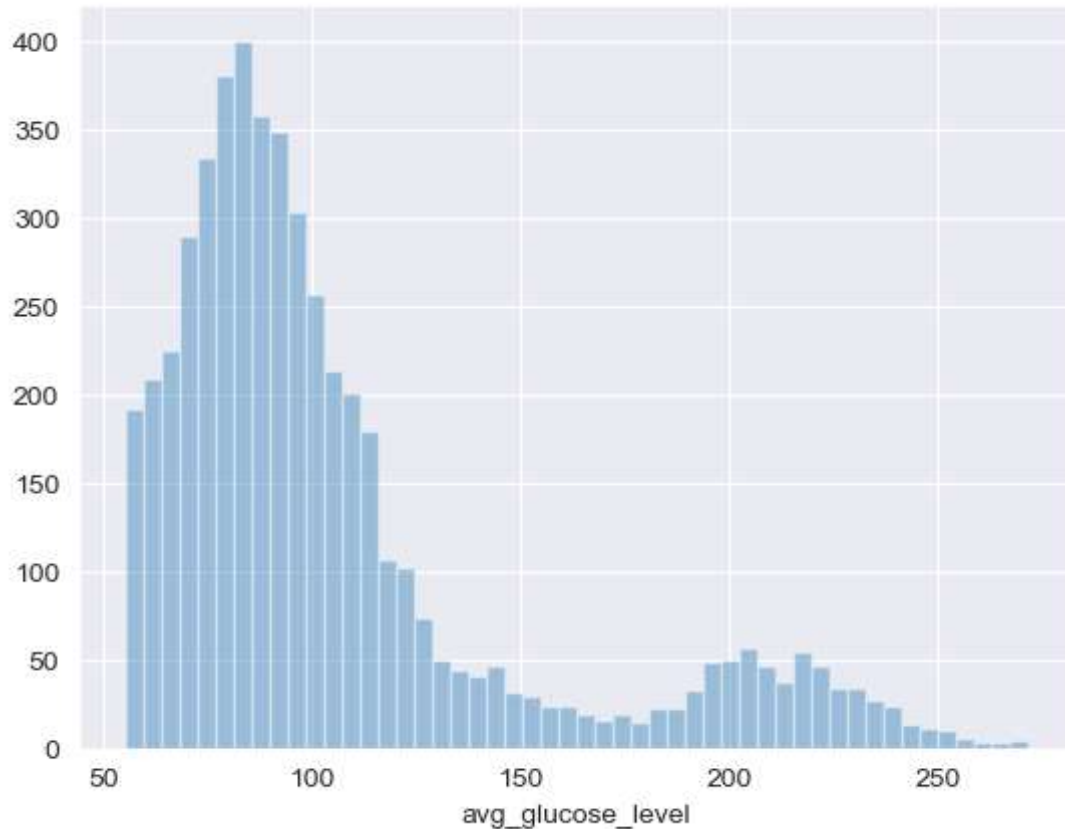


```
In [164]: 1 sns.distplot(df1['avg_glucose_level'])
```

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

```
warnings.warn(msg, FutureWarning)
```

```
Out[164]: <AxesSubplot:xlabel='avg_glucose_level'>
```



SWARNPLOT

```
In [5]: 1 sns.swarmplot(x='age',y='smoking_status',data=df1)
```

C:\Users\hp\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 50.7% of the points cannot be placed; you may want to decrease the size of the markers or use stripplot.

warnings.warn(msg, UserWarning)

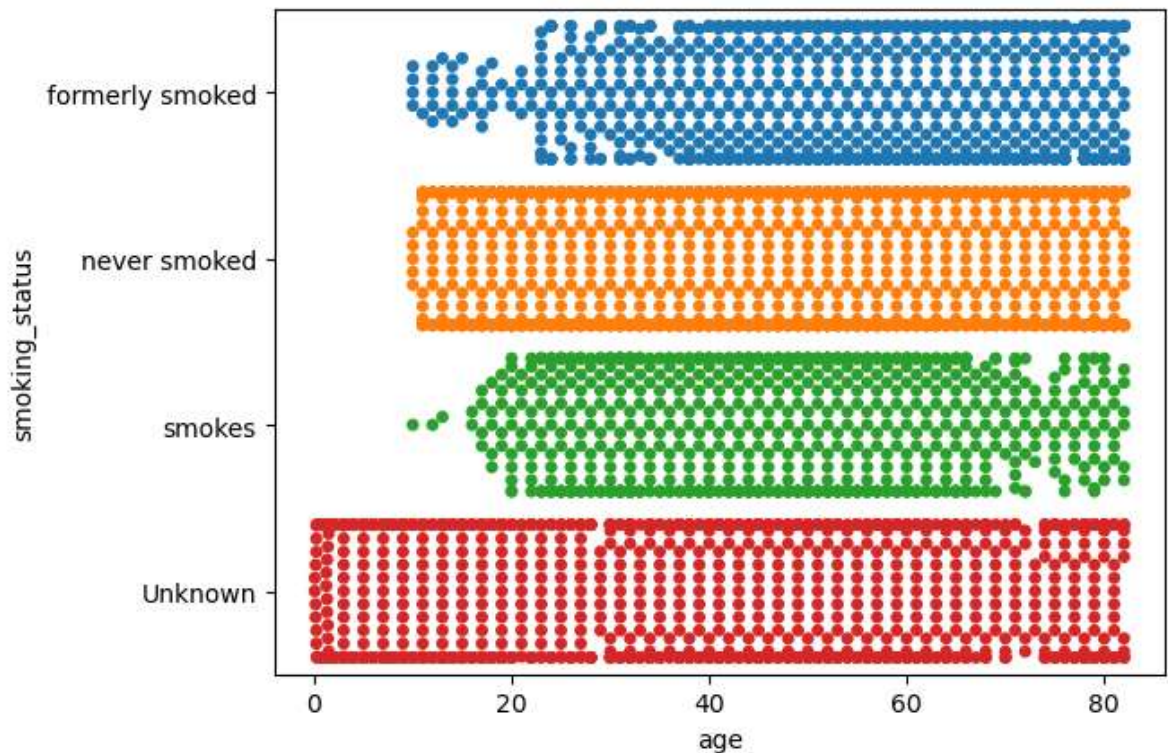
C:\Users\hp\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 78.8% of the points cannot be placed; you may want to decrease the size of the markers or use stripplot.

warnings.warn(msg, UserWarning)

C:\Users\hp\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 70.1% of the points cannot be placed; you may want to decrease the size of the markers or use stripplot.

warnings.warn(msg, UserWarning)

```
Out[5]: <AxesSubplot:xlabel='age', ylabel='smoking_status'>
```




```
In [171]: 1 sns.pairplot(df1,hue="gender")
```

```
Out[171]: <seaborn.axisgrid.PairGrid at 0x196805a7dc0>
```



```
In [ ]: 1
```

```
In [ ]: 1
```

```
In [ ]: 1
```