stroke-duml

May 11, 2024

1 STROKE DETECTION

[1]:
```
import numpy as np
import pandas as pd
import sklearn
import matplotlib.pyplot as  plt
import seaborn as sns
import tkinter as tk
from tkinter import filedialog
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report,accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
import sklearn
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
```

[2]:
```
data = pd.read_csv(r"C:
↪\Users\gagan\OneDrive\Desktop\healthcare-dataset-stroke-data.csv")
```

2 EDA

[3]:
```
data.head(5)
```

[3]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked
1	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked
2	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes
3	1665	Female	79.0	1	0	Yes					
4	56669	Male	81.0	0	0	Yes					

```
3  Self-employed          Rural             174.12  24.0     never smoked
4        Private          Urban             186.21  29.0  formerly smoked

  stroke
0    yes
1    yes
2    yes
3    yes
4    yes
```

[4]:
```
data.tail(5)
```

[4]:
```
         id  gender   age  hypertension  heart_disease ever_married  \
4904  14180  Female  13.0             0              0           No
4905  44873  Female  81.0             0              0          Yes
4906  19723  Female  35.0             0              0          Yes
4907  37544    Male  51.0             0              0          Yes
4908  44679  Female  44.0             0              0          Yes

          work_type Residence_type  avg_glucose_level   bmi   smoking_status  \
4904       children          Rural             103.08  18.6          Unknown
4905  Self-employed          Urban             125.20  40.0     never smoked
4906  Self-employed          Rural              82.99  30.6     never smoked
4907        Private          Rural             166.29  25.6  formerly smoked
4908       Govt_job          Urban              85.28  26.2          Unknown

     stroke
4904     no
4905     no
4906     no
4907     no
4908     no
```

[5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4909 entries, 0 to 4908
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   id                 4909 non-null   int64
 1   gender             4909 non-null   object
 2   age                4909 non-null   float64
 3   hypertension       4909 non-null   int64
 4   heart_disease      4909 non-null   int64
 5   ever_married       4909 non-null   object
 6   work_type          4909 non-null   object
```

```
 7   Residence_type     4909 non-null   object
 8   avg_glucose_level  4909 non-null   float64
 9   bmi                4909 non-null   float64
 10  smoking_status     4909 non-null   object
 11  stroke             4909 non-null   object
dtypes: float64(3), int64(3), object(6)
memory usage: 460.3+ KB
```

[6]:
```
data.isnull().sum()
```

[6]:
```
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
```

[7]:
```
data.describe(include = 'all')
```

[7]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
count	4909.000000	4909	4909.000000	4909.000000	4909.000000	4909	4909	4909	4909.000000	4909.000000	4909	4909
unique	NaN	3	NaN	NaN	NaN	2	5	2	NaN	NaN	4	2
top	NaN	Female	NaN	NaN	NaN	Yes	Private	Urban	NaN	NaN	never smoked	0
freq	NaN	2897	NaN	NaN	NaN	3204	2811	2490	NaN	NaN	1749	4700
mean	37064.313506	NaN	42.865374	0.091872	0.049501	NaN	NaN	NaN	105.305150	28.893237	NaN	NaN
std	20995.098457	NaN	22.555115	0.288875	0.216934	NaN	NaN	NaN	44.424341	7.854067	NaN	NaN
min	77.000000	NaN	0.080000	0.000000	0.000000	NaN	NaN	NaN	55.120000	10.300000	NaN	NaN
25%	18605.000000	NaN	25.000000	0.000000	0.000000	NaN	NaN	NaN	77.240000	23.800000	NaN	NaN
50%	37608.000000	NaN	44.000000	0.000000	0.000000	NaN	NaN	NaN	91.680000	28.400000	NaN	NaN
75%	55220.000000	NaN	60.000000	0.000000	0.000000	NaN	NaN	NaN	113.570000	32.800000	NaN	NaN
max	72940.000000	NaN	82.000000	1.000000	1.000000	NaN	NaN	NaN	271.740000	48.900000	NaN	NaN

```
25%             NaN       NaN            NaN          77.070000    23.500000
50%             NaN       NaN            NaN          91.680000    28.100000
75%             NaN       NaN            NaN         113.570000    33.100000
max             NaN       NaN            NaN         271.740000    97.600000

       smoking_status stroke
count            4909   4909
unique              4      2
top      never smoked     no
freq             1852   4700
mean              NaN    NaN
std               NaN    NaN
min               NaN    NaN
25%               NaN    NaN
50%               NaN    NaN
75%               NaN    NaN
max               NaN    NaN
```

[8]:
```
columns = []
for column in data:
    columns.append(column)
print(type(columns),"\n",[i for i in columns])
```

```
<class 'list'>
 ['id', 'gender', 'age', 'hypertension', 'heart_disease', 'ever_married',
'work_type', 'Residence_type', 'avg_glucose_level', 'bmi', 'smoking_status',
'stroke']
```

[9]:
```
#before encoding
for column in columns:
    print(column,"=>")
    print(data[column].unique())
    print("\n")
```

```
id =>
[ 9046 31112 60182 … 19723 37544 44679]


gender =>
['Male' 'Female' 'Other']


age =>
[6.70e+01 8.00e+01 4.90e+01 7.90e+01 8.10e+01 7.40e+01 6.90e+01 7.80e+01
 6.10e+01 5.40e+01 5.00e+01 6.40e+01 7.50e+01 6.00e+01 7.10e+01 5.20e+01
 8.20e+01 6.50e+01 5.70e+01 4.20e+01 4.80e+01 7.20e+01 5.80e+01 7.60e+01
 3.90e+01 7.70e+01 6.30e+01 7.30e+01 5.60e+01 4.50e+01 7.00e+01 5.90e+01
 6.60e+01 4.30e+01 6.80e+01 4.70e+01 5.30e+01 3.80e+01 5.50e+01 4.60e+01
```

```
 3.20e+01 5.10e+01 1.40e+01 3.00e+00 8.00e+00 3.70e+01 4.00e+01 3.50e+01
 2.00e+01 4.40e+01 2.50e+01 2.70e+01 2.30e+01 1.70e+01 1.30e+01 4.00e+00
 1.60e+01 2.20e+01 3.00e+01 2.90e+01 1.10e+01 2.10e+01 1.80e+01 3.30e+01
 2.40e+01 3.60e+01 6.40e-01 3.40e+01 4.10e+01 8.80e-01 5.00e+00 2.60e+01
 3.10e+01 7.00e+00 1.20e+01 6.20e+01 2.00e+00 9.00e+00 1.50e+01 2.80e+01
 1.00e+01 1.80e+00 3.20e-01 1.08e+00 1.90e+01 6.00e+00 1.16e+00 1.00e+00
 1.40e+00 1.72e+00 2.40e-01 1.64e+00 1.56e+00 7.20e-01 1.88e+00 1.24e+00
 8.00e-01 4.00e-01 8.00e-02 1.48e+00 5.60e-01 1.32e+00 1.60e-01 4.80e-01]


hypertension =>
[0 1]


heart_disease =>
[1 0]


ever_married =>
['Yes' 'No']


work_type =>
['Private' 'Self-employed' 'Govt_job' 'children' 'Never_worked']


Residence_type =>
['Urban' 'Rural']


avg_glucose_level =>
[228.69 105.92 171.23 …  82.99 166.29  85.28]


bmi =>
[36.6 32.5 34.4 24.  29.  27.4 22.8 24.2 29.7 36.8 27.3 28.2 30.9 37.5
 25.8 37.8 22.4 48.9 26.6 27.2 23.5 28.3 44.2 25.4 22.2 30.5 26.5 33.7
 23.1 32.  29.9 23.9 28.5 26.4 20.2 33.6 38.6 39.2 27.7 31.4 36.5 33.2
 32.8 40.4 25.3 30.2 47.5 20.3 30.  28.9 28.1 31.1 21.7 27.  24.1 45.9
 44.1 22.9 29.1 32.3 41.1 25.6 29.8 26.3 26.2 29.4 24.4 28.  28.8 34.6
 19.4 30.3 41.5 22.6 56.6 27.1 31.3 31.  31.7 35.8 28.4 20.1 26.7 38.7
 34.9 25.  23.8 21.8 27.5 24.6 32.9 26.1 31.9 34.1 36.9 37.3 45.7 34.2
 23.6 22.3 37.1 45.  25.5 30.8 37.4 34.5 27.9 29.5 46.  42.5 35.5 26.9
 45.5 31.5 33.  23.4 30.7 20.5 21.5 40.  28.6 42.2 29.6 35.4 16.9 26.8
 39.3 32.6 35.9 21.2 42.4 40.5 36.7 29.3 19.6 18.  17.6 19.1 50.1 17.7
 54.6 35.  22.  39.4 19.7 22.5 25.2 41.8 60.9 23.7 24.5 31.2 16.  31.6
 25.1 24.8 18.3 20.  19.5 36.  35.3 40.1 43.1 21.4 34.3 27.6 16.5 24.3
 25.7 21.9 38.4 25.9 54.7 18.6 24.9 48.2 20.7 39.5 23.3 64.8 35.1 43.6
```

```
 21.  47.3 16.6 21.6 15.5 35.6 16.7 41.9 16.4 17.1 29.2 37.9 44.6 39.6
 40.3 41.6 39.  23.2 18.9 36.1 36.3 46.5 16.8 46.6 35.2 20.9 13.8 31.8
 15.3 38.2 45.2 17.  49.8 27.8 60.2 23.  22.1 26.  44.3 51.  39.7 34.7
 21.3 41.2 34.8 19.2 35.7 40.8 24.7 19.  32.4 34.  28.7 32.1 51.5 20.4
 30.6 71.9 19.3 40.9 17.2 16.1 16.2 40.6 18.4 21.1 42.3 32.2 50.2 17.5
 18.7 42.1 47.8 20.8 30.1 17.3 36.4 12.  36.2 55.7 14.4 43.  41.7 33.8
 43.9 22.7 57.5 37.  38.5 16.3 44.  32.7 54.2 40.2 33.3 17.4 41.3 52.3
 14.6 17.8 46.1 33.1 18.1 43.8 50.3 38.9 43.7 39.9 15.9 19.8 12.3 78.
 38.3 41.  42.6 43.4 15.1 20.6 33.5 43.2 30.4 38.  33.4 44.9 44.7 37.6
 39.8 53.4 55.2 42.  37.2 42.8 18.8 42.9 14.3 37.7 48.4 50.6 46.2 49.5
 43.3 33.9 18.5 44.5 45.4 55.  54.8 19.9 17.9 15.6 52.8 15.2 66.8 55.1
 18.2 48.5 55.9 57.3 10.3 14.1 15.7 56.  44.8 13.4 51.8 38.1 57.7 44.4
 38.8 49.3 39.1 54.  56.1 97.6 53.9 13.7 11.5 41.4 14.2 49.4 15.4 45.1
 49.2 48.7 53.8 42.7 48.8 52.7 53.5 50.5 15.8 45.3 14.8 51.9 63.3 40.7
 61.2 48.  46.8 48.3 58.1 50.4 11.3 12.8 13.5 14.5 15.  59.7 47.4 52.5
 13.2 52.9 61.6 49.9 54.3 47.9 13.  13.9 50.9 57.2 64.4 92.  50.8 57.9
 45.8 47.6 14.  46.4 46.9 47.1 13.3 48.1 51.7 46.3 54.1 14.9]


smoking_status =>
['formerly smoked' 'never smoked' 'smokes' 'Unknown']


stroke =>
['yes' 'no']
```

[10]:
```python
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
data = data.copy()
for column in columns:
    data[column] = encoder.fit_transform(data[column]) #suffix
    #data[column + '_encoded'] = encoder.fit_transform(data[column]) #suffix
data.sample(6)
```

[10]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type
4164	2018	0	51	0	0	1	2
898	2571	1	52	0	0	1	2
3498	2060	1	32	0	0	0	4
1563	2645	0	53	0	0	0	2
4124	3547	1	68	0	0	1	2
3044	1571	0	78	1	0	1	2

	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
4164	1	1606	325	2	0
898	1	1420	174	3	0

```
3498              0              2034   77              0       0
1563              1               484   84              2       0
4124              1               319  141              3       0
3044              1              1727  311              1       0
```

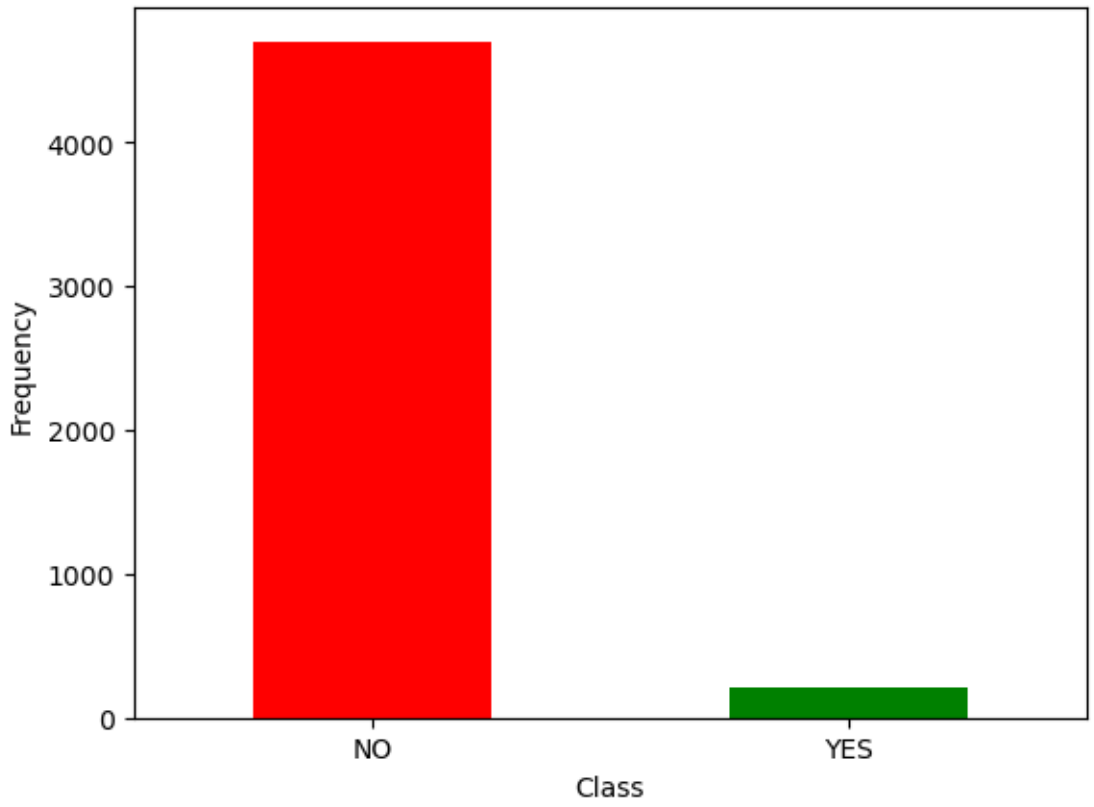
[11]:
```
#after encoding
for column in columns:
    print(column,"=>")
    print(data[column].unique())
    print("\n")
```

```
id =>
[ 594 2019 4042 … 1289 2451 2923]


gender =>
[1 0 2]


age =>
[ 88 101  70 100 102  95  90  99  82  75  71  85  96  81  92  73 103  86
  78  63  69  93  79  97  60  98  84  94  77  66  91  80  87  64  89  68
  74  59  76  67  53  72  35  24  29  58  61  56  41  65  46  48  44  38
  34  25  37  43  51  50  32  42  39  54  45  57   7  55  62  10  26  47
  52  28  33  83  23  30  36  49  31  21   3  12  40  27  13  11  16  20
   2  19  18   8  22  14   9   4   0  17   6  15   1   5]


hypertension =>
[0 1]


heart_disease =>
[1 0]


ever_married =>
[1 0]


work_type =>
[2 3 0 4 1]


Residence_type =>
[1 0]
```

```
avg_glucose_level =>
[3734 2429 3309 … 1290 3289 1426]


bmi =>
[239 198 217 113 163 147 101 115 170 241 146 155 182 248 131 251  97 355
 139 145 108 156 314 127  95 178 138 210 104 193 172 112 158 137  75 209
 259 265 150 187 238 205 201 277 126 175 343  76 173 162 154 184  90 143
 114 330 313 102 164 196 284 129 171 136 135 167 117 153 161 219  67 176
 288  99 398 144 186 183 190 231 157  74 140 260 222 123 111  91 148 119
 202 134 192 214 242 246 328 215 109  96 244 322 128 181 247 218 152 168
 331 298 228 142 327 188 203 107 180  78  88 273 159 295 169 227  42 141
 266 199 232  85 297 278 240 166  69  53  49  64 362  50 388 223  93 267
  70  98 125 291 407 110 118 185  33 189 124 121  56  73  68 233 226 274
 304  87 216 149  38 116 130  92 257 132 389  59 122 349  80 268 106 412
 224 308  83 341  39  89  28 229  40 292  37  44 165 252 318 269 276 289
 263 105  62 234 236 336  41 337 225  82  12 191  26 255 324  43 360 151
 406 103  94 133 315 370 270 220  86 285 221  65 230 281 120  63 197 213
 160 194 371  77 179 414  66 282  45  34  35 279  57  84 296 195 363  48
  60 294 345  81 174  46 237   3 235 394  18 303 290 211 311 100 401 243
 258  36 312 200 386 275 206  47 286 375  20  51 332 204  54 310 364 262
 309 272  32  71   4 415 256 283 299 307  24  79 208 305 177 253 207 321
 319 249 271 380 393 293 245 301  61 302  17 250 351 367 333 359 306 212
  58 317 326 391 390  72  52  29 378  25 413 392  55 352 395 400   0  15
  30 396 320   9 373 254 402 316 261 357 264 384 397 417 383  11   2 287
  16 358  27 323 356 353 382 300 354 377 381 366  31 325  21 374 410 280
 408 347 338 350 404 365   1   5  10  19  23 405 342 376   7 379 409 361
 387 346   6  13 369 399 411 416 368 403 329 344  14 335 339 340   8 348
 372 334 385  22]


smoking_status =>
[1 2 3 0]


stroke =>
[1 0]
```

[12]:
```
labels = ['NO','YES']
sh = pd.Series(data['stroke']).value_counts(sort = True)
sh.plot(kind = 'bar',rot = 0,color = ['red','green'])
plt.xticks(range(2), labels)
plt.xlabel("Class")
plt.ylabel("Frequency")
```

[12]: Text(0, 0.5, 'Frequency')



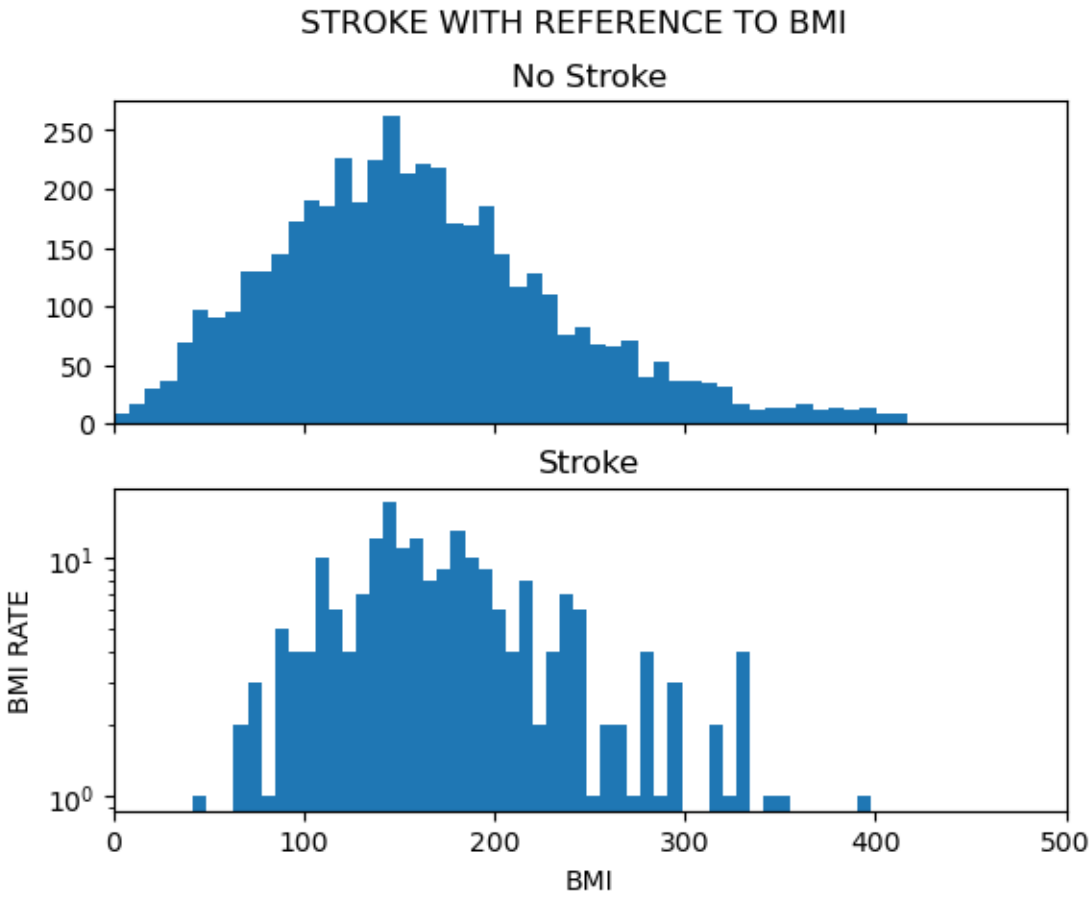
[13]:
```
data['stroke'].value_counts()
```

[13]:
```
stroke
0    4700
1     209
Name: count, dtype: int64
```

[14]:
```
ys = data[data['stroke'] == 1]
ns = data[data['stroke'] == 0]
```

[15]:
```
f, (ax1, ax2) = plt.subplots(2, 1, sharex=True)
f.suptitle('STROKE WITH REFERENCE TO BMI')
bins = 50
ax1.hist(ns.bmi, bins = bins)
ax1.set_title('No Stroke')
ax2.hist(ys.bmi, bins = bins)
ax2.set_title('Stroke')
plt.xlabel('BMI')
plt.ylabel('BMI RATE')
```

```
plt.xlim((0, 500))
plt.yscale('log')
plt.show()
```



[16]:
```
ys.value_counts()
```

[16]:
```
id    gender  age  hypertension  heart_disease  ever_married  work_type
Residence_type  avg_glucose_level  bmi  smoking_status  stroke
9     1       102  0             0              1             3          0
1760                187  2                1         1
2419  0       102  0             0              1             2          0
1122                107  2                1         1
3189  1       101  0             1              1             2          1
498                 115  3                1         1
3236  1       95   0             0              1             2          1
2119                129  0                1         1
3279  0       35   0             0              0             4          0
99                  182  0                1         1
```

```
                                                                    ..
1845  1       82   1             1              1             2          1
2655                247  3              1         1
1856  0       99   0             0              1             3          0
2554                181  2              1         1
1857  1       78   0             0              1             2          1
1478                190  0              1         1
1909  1       97   1             0              1             3          0
3416                143  1              1         1
4907  0       74   1             0              1             2          1
263                 176  0              1         1
Name: count, Length: 209, dtype: int64
```

[17]:
```
ns.value_counts()
```

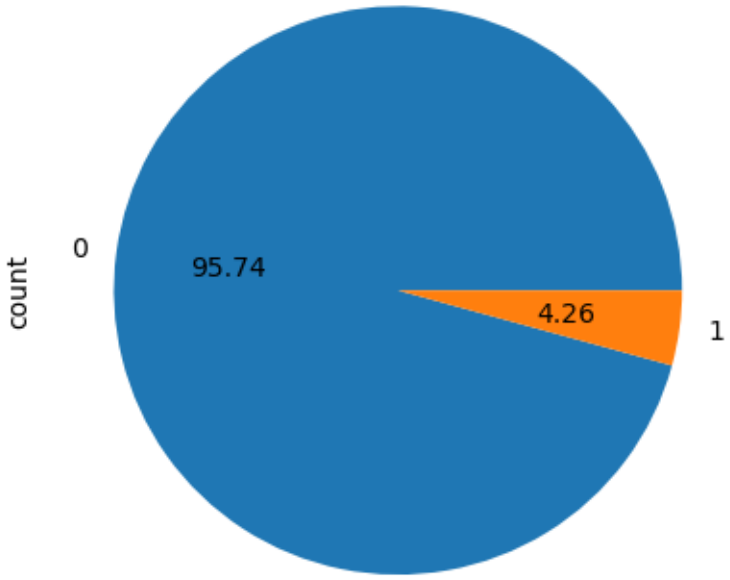
[17]:
```
id    gender  age  hypertension  heart_disease  ever_married  work_type
Residence_type  avg_glucose_level  bmi  smoking_status  stroke
0     0       34   0             0              0             4          0
1448                59   0              0         1
3257  0       52   0             0              0             2          1
2439                97   2              0         1
3273  1       90   0             0              1             2          1
2137                112  1              0         1
3272  1       96   0             0              1             2          0
856                 118  1              0         1
3271  0       33   0             0              0             4          1
107                 86   2              0         1
                                                                    ..
1631  1       65   0             0              1             0          1
1930                133  1              0         1
1630  0       61   0             0              0             2          0
3635                167  1              0         1
1629  1       83   0             0              1             0          0
2321                192  0              0         1
1628  0       35   0             0              0             2          0
1752                116  2              0         1
4908  0       23   0             0              0             4          1
2318                49   0              0         1
Name: count, Length: 4700, dtype: int64
```

[18]:
```
x = data.drop(columns='stroke', axis=1)
y = data['stroke']
```

2.1 UNDERSAMPLING

[19]:
```
#before undersampling
y.value_counts().plot.pie(autopct = '%.2f')
```

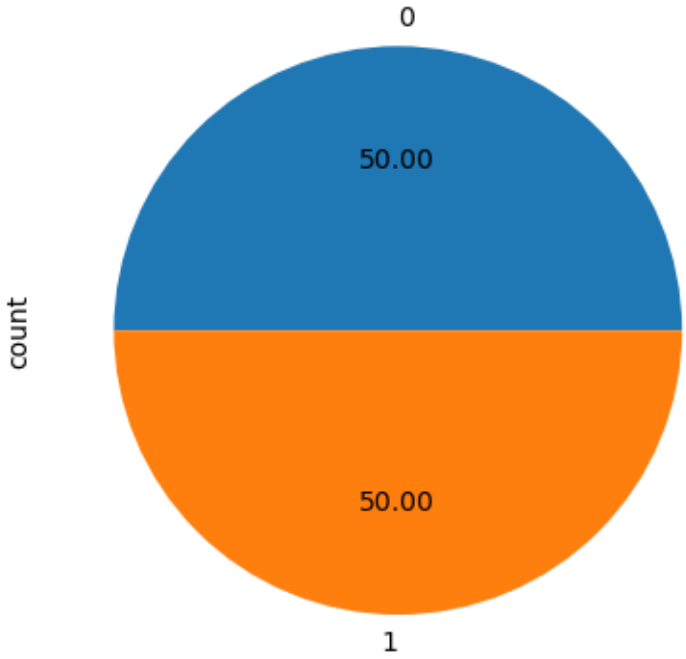
[19]: <Axes: ylabel='count'>



[20]:
```
from imblearn.under_sampling import RandomUnderSampler
rus = RandomUnderSampler(sampling_strategy=1)
x, y = rus.fit_resample(x, y)
```

[21]:
```
#after undersampling
y.value_counts().plot.pie(autopct = '%.2f')
```

[21]: <Axes: ylabel='count'>



[22]:
```
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.
↪20,random_state = 42)
```

[23]:
```
print(x.shape, xtrain.shape, xtest.shape)
```

```
(418, 11) (334, 11) (84, 11)
```

[24]:
```
print(y.shape, ytrain.shape, ytest.shape)
```

```
(418,) (334,) (84,)
```

2.2 LOGISTIC REGRESSION

[25]:
```
model = LogisticRegression()
```

[26]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_data = scaler.fit_transform(xtrain)
```

[27]:
```
model.fit(scaled_data, ytrain)
```

[27]:
```
LogisticRegression()
```

[28]:
```
X_train_prediction = model.predict(scaled_data)
training_data_accuracy = accuracy_score(X_train_prediction, ytrain)
```

[29]:
```
print('Accuracy score on Test Data : ', training_data_accuracy)
```

```
Accuracy score on Test Data :  0.7544910179640718
```

[30]:
```
X_test_prediction_dt = model.predict(scaler.transform(xtest))
```

[31]:
```
test_data_accuracy = accuracy_score(X_test_prediction_dt, ytest)
```

[32]:
```
print('Accuracy score on Test Data : ', test_data_accuracy)
```

```
Accuracy score on Test Data :  0.8214285714285714
```