smoke-duml

May 22, 2024

[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
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

[2]:
```
import warnings
warnings.filterwarnings("ignore")
```

[3]:
```
data = pd.read_csv(r"C:\Users\gagan\Downloads\csv_result-Smoke.csv")
```

1 EDA

1.1 Understanding the data

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

[4]:

	id	age	height(cm)	weight(kg)	waist(cm)	eyesight(left)	\
0	1	35	170	85	97.0	0.9	
1	2	20	175	110	110.0	0.7	
2	3	45	155	65	86.0	0.9	
3	4	45	165	80	94.0	0.8	
4	5	20	165	60	81.0	1.5	

```
   eyesight(right)  hearing(left)  hearing(right)  systolic  …  HDL  LDL  \
0              0.9              1               1       118  …   70  142
1              0.9              1               1       119  …   71  114
2              0.9              1               1       110  …   57  112
3              0.7              1               1       158  …   46   91
4              0.1              1               1       109  …   47   92

   hemoglobin  'Urine  'serum   AST   ALT  Gtp  'dental  smoking{0
0        19.8       1     1.0    61   115  125        1          1
1        15.9       1     1.1    19    25   30        1          0
2        13.7       3     0.6  1090  1400  276        0          0
3        16.9       1     0.9    32    36   36        0          0
4        14.9       1     1.2    26    28   15        0          0

[5 rows x 24 columns]
```

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

[5]:
```
        id  age  height(cm)  weight(kg)  waist(cm)  eyesight(left)  \
1013  1014   20         180          70       80.0             1.2
1014  1015   35         175          80       90.5             1.0
1015  1016   40         175          70       74.0             0.9
1016  1017   65         150          55       85.0             0.4
1017  1018   25         175          70       76.0             1.5

      eyesight(right)  hearing(left)  hearing(right)  systolic  …  HDL  LDL  \
1013              1.0              1               1       119  …   68  103
1014              1.0              1               1       137  …   32  132
1015              1.0              1               1       101  …   66  107
1016              0.6              1               1       140  …   72   98
1017              1.2              1               1       110  …   65   61

      hemoglobin  'Urine  'serum  AST  ALT  Gtp  'dental  smoking{0
1013        15.0       1     1.0   16   12   13        0          0
1014        17.9       1     1.4   33   56   62        0          1
1015        15.8       1     0.9   22   20   16        0          1
1016        12.0       1     0.7   27    9   12        0          0
1017        14.9       1     1.0   19   23   15        0          1

[5 rows x 24 columns]
```

[6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1018 entries, 0 to 1017
Data columns (total 24 columns):
 #   Column           Non-Null Count  Dtype
```

```
---  ------            --------------  -----
 0   id                1018 non-null   int64
 1   age               1018 non-null   int64
 2   height(cm)        1018 non-null   int64
 3   weight(kg)        1018 non-null   int64
 4   waist(cm)         1018 non-null   float64
 5   eyesight(left)    1018 non-null   float64
 6   eyesight(right)   1018 non-null   float64
 7   hearing(left)     1018 non-null   int64
 8   hearing(right)    1018 non-null   int64
 9   systolic          1018 non-null   int64
 10  relaxation        1018 non-null   int64
 11  'fasting          1018 non-null   int64
 12  Cholesterol       1018 non-null   int64
 13  triglyceride      1018 non-null   int64
 14  HDL               1018 non-null   int64
 15  LDL               1018 non-null   int64
 16  hemoglobin        1018 non-null   float64
 17  'Urine            1018 non-null   int64
 18  'serum            1018 non-null   float64
 19  AST               1018 non-null   int64
 20  ALT               1018 non-null   int64
 21  Gtp               1018 non-null   int64
 22  'dental           1018 non-null   int64
 23  smoking{0         1018 non-null   int64
dtypes: float64(5), int64(19)
memory usage: 191.0 KB
```

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

[7]:

	id	age	height(cm)	weight(kg)	waist(cm)
count	1018.000000	1018.000000	1018.000000	1018.000000	1018.000000
mean	509.500000	43.777014	164.833006	66.110020	82.327701
std	294.015589	12.233487	9.105608	12.665645	9.504902
min	1.000000	20.000000	140.000000	35.000000	57.000000
25%	255.250000	35.000000	160.000000	55.000000	76.000000
50%	509.500000	40.000000	165.000000	65.000000	82.000000
75%	763.750000	50.000000	170.000000	75.000000	88.000000
max	1018.000000	80.000000	190.000000	120.000000	118.000000

	eyesight(left)	eyesight(right)	hearing(left)	hearing(right)
count	1018.000000	1018.000000	1018.000000	1018.000000
mean	1.006385	0.998723	1.021611	1.027505
std	0.433695	0.431695	0.145481	0.163630
min	0.100000	0.100000	1.000000	1.000000
25%	0.800000	0.800000	1.000000	1.000000
50%	1.000000	1.000000	1.000000	1.000000

```
75%          1.200000         1.200000       1.000000        1.000000
max          9.900000         9.900000       2.000000        2.000000

          systolic  …          HDL          LDL   hemoglobin       'Urine  \
count  1018.000000  …  1018.000000  1018.000000  1018.000000  1018.000000
mean    121.524558  …    57.036346   112.792731    14.603143     1.066798
std      13.872289  …    14.187800    36.514747     1.600513     0.336939
min      72.000000  …    24.000000    16.000000     7.100000     1.000000
25%     112.000000  …    46.000000    90.000000    13.600000     1.000000
50%     120.000000  …    55.000000   110.500000    14.800000     1.000000
75%     130.000000  …    65.000000   134.000000    15.700000     1.000000
max     203.000000  …   112.000000   590.000000    21.100000     5.000000

            'serum          AST          ALT          Gtp      'dental  \
count  1018.000000  1018.000000  1018.000000  1018.000000  1018.000000
mean      0.891159    27.167976    28.196464    40.750491     0.221022
std       0.205864    35.905046    47.480604    49.806120     0.415139
min       0.100000     9.000000     4.000000     7.000000     0.000000
25%       0.800000    19.000000    15.000000    17.000000     0.000000
50%       0.900000    23.000000    21.000000    25.000000     0.000000
75%       1.000000    29.000000    30.000000    44.750000     0.000000
max       2.200000  1090.000000  1400.000000   586.000000     1.000000

         smoking{0
count  1018.000000
mean      0.354617
std       0.478632
min       0.000000
25%       0.000000
50%       0.000000
75%       1.000000
max       1.000000

[8 rows x 24 columns]
```

[8]:
```
data.duplicated().sum()
```

[8]:
```
0
```

[9]:
```
data.nunique()
```

[9]:
```
id                  1018
age                   13
height(cm)            11
weight(kg)            18
waist(cm)            236
eyesight(left)        15
```

```
eyesight(right)      14
hearing(left)         2
hearing(right)        2
systolic             75
relaxation           58
'fasting            102
Cholesterol         177
triglyceride        268
HDL                  78
LDL                 163
hemoglobin           91
'Urine                5
'serum               17
AST                  68
ALT                  94
Gtp                 146
'dental               2
smoking{0             2
dtype: int64
```

[10]:
```
data.isnull().sum()
```

[10]:
```
id                  0
age                 0
height(cm)          0
weight(kg)          0
waist(cm)           0
eyesight(left)      0
eyesight(right)     0
hearing(left)       0
hearing(right)      0
systolic            0
relaxation          0
'fasting            0
Cholesterol         0
triglyceride        0
HDL                 0
LDL                 0
hemoglobin          0
'Urine              0
'serum              0
AST                 0
ALT                 0
Gtp                 0
'dental             0
smoking{0           0
dtype: int64
```

[11]:
```
data.dtypes
```

[11]:
```
id                    int64
age                   int64
height(cm)            int64
weight(kg)            int64
waist(cm)           float64
eyesight(left)      float64
eyesight(right)     float64
hearing(left)         int64
hearing(right)        int64
systolic              int64
relaxation            int64
'fasting              int64
Cholesterol           int64
triglyceride          int64
HDL                   int64
LDL                   int64
hemoglobin          float64
'Urine                int64
'serum              float64
AST                   int64
ALT                   int64
Gtp                   int64
'dental               int64
smoking{0             int64
dtype: object
```

[12]:
```
data.shape
```

[12]:
```
(1018, 24)
```

[13]:
```
rename = {'smoking{0':'smoking'}
data.rename(columns=rename, inplace=True)
```

[14]:
```
columns =  data.columns.tolist()
columns
```

[14]:
```
['id',
 'age',
 'height(cm)',
 'weight(kg)',
 'waist(cm)',
 'eyesight(left)',
 'eyesight(right)',
 'hearing(left)',
 'hearing(right)',
```

```
 'systolic',
 'relaxation',
 "'fasting",
 'Cholesterol',
 'triglyceride',
 'HDL',
 'LDL',
 'hemoglobin',
 "'Urine",
 "'serum",
 'AST',
 'ALT',
 'Gtp',
 "'dental",
 'smoking']
```

[15]:
```
#before encoding
for column in columns:
    print(column,"=>")
    print(data[column].unique())
    print("\n")
```

```
id =>
[   1    2    3 … 1016 1017 1018]


age =>
[35 20 45 60 40 50 75 55 25 30 70 65 80]


height(cm) =>
[170 175 155 165 160 180 150 140 145 185 190]


weight(kg) =>
[ 85 110  65  80  60  50  90  75  55  40  70  95  45 100  35 105 115 120]


waist(cm) =>
[ 97.  110.   86.   94.   81.   78.   95.   85.   74.   77.6  72.   89.
  71.   62.   92.   84.   78.5  80.   83.   76.4  75.   79.  106.   83.8
  64.2  90.8  83.5  75.5  91.5 101.   77.   59.   70.   69.   91.   76.7
  87.2  67.   88.   82.   71.2  87.   68.   61.   65.   96.  100.   76.2
  85.1  90.  118.   94.1  98.   62.2  73.   89.1  75.2  93.3 104.9  71.6
  82.7  93.   73.5  73.8  88.5  78.4  90.1  86.5  81.5  68.5  79.5  81.2
  60.   85.5  82.2  75.8  89.4  99.   66.   98.1  75.7  92.5  81.3  71.8
  80.6  76.   84.2  78.7  63.   84.1  95.3  80.4  85.8  82.5  72.6  95.5
```

```
  83.4  85.2  96.2 100.6  84.5  87.5  70.5  75.3  74.2  92.1 102.   84.4
  95.4 112.   92.4 104.   83.1  72.4  70.1  90.5 107.  114.8  83.9  67.2
 105.3  77.5  67.3  64.3  95.2  82.4  71.3  81.6  66.8  94.5  90.2  91.1
  94.4  82.3  94.7  88.9 101.2  81.1  79.9  97.8  71.1  86.4 105.   80.1
  58.  111.8  77.2  66.5  86.3  75.6  81.8  76.8  73.1  71.5  74.3  76.5
  92.3  89.5  78.6  87.4 103.   62.5  83.6  88.4  74.5  83.2 102.8  73.2
  99.2  94.8  89.2 104.5 100.4  69.5  79.8  77.3  94.2  89.6  73.4  67.8
  98.6  85.4 103.7  82.9  82.1  68.1  79.4  72.5  79.3  79.2  80.5  88.7
  99.5  69.3  88.8  93.8  82.8  80.2 113.   74.8  72.2  64.5  96.5  93.2
  57.   64.  108.4  83.3  68.6  83.7  76.1 109.   98.7  77.8 103.5  67.1
  87.1  87.7  70.4 115.   92.8  84.9  97.3  73.3  86.8  99.6 102.5  97.2
  93.6  69.7  86.1  74.9  86.7  91.3  78.2  74.1]


eyesight(left) =>
[0.9 0.7 0.8 1.5 1.  1.2 0.5 0.3 0.4 0.6 0.1 0.2 9.9 2.  1.8]


eyesight(right) =>
[0.9 0.7 0.1 1.  1.5 0.5 0.8 1.2 0.4 0.2 0.3 0.6 2.  9.9]


hearing(left) =>
[1 2]


hearing(right) =>
[1 2]


systolic =>
[118 119 110 158 109 126 130  89 114 112 125 101 127 120 151 100 128 115
 121 116 124 139 102 122 117 165 166 135 148 113 134 106 143 105 129 108
 150 123  98 132 160 155 133 104 180 138  90 107  93  96 169 136 103 137
  95  99 111 141 140 142 154 203 131 144 159 152  92 145 153 164 156  94
  88  72  86]


relaxation =>
[ 78  79  80  88  64  75  60  57  81  76  68  72  84  73 103  70  62  71
  77  56  67  99  69  86  82  83  90  61  89  85 100 104  74  66 110  65
  59 102  87  92  94 146  63  93  98  96 106  51 107  53  55  97  42  58
  54  95  91 101]


'fasting =>
[ 97  88  80 249 100 114  90  83  96  94  86  78 129 268  55 134  93  81
  92 102  85  95  84 135  79 127  87  99 106 109  91  89  76 105 107  74
```

```
 108 104  72 123 125 122 170 167 103 101  82 139 133 110 128 115  98 143
 132 199 121 151 197  62 157 111 124 130 141 142 112 113 165 116  70 178
 118 223 117 183 149 140 152 154 147  75 302 160 145  77 158 205  73 192
 126 119  69  64 161 136 144 146 227 120 207 137]


Cholesterol =>
[239 211 193 210 179 177 207 170 178 184 154 135 230 259 163 226 164 174
 122 215 162 125 190 172 275 169 188 225 197 153 208 139 157 181 227 203
 271 220 198 175 228 200 204 241 176 245 167 219 191 165 234 173 171 257
 252 256 192 145 237 158 168 212 182 141 238 166 270 235 253 189 144 151
 223 213 195 160 243 187 205 229 280 217 146 313 216 159 202 199 233 209
 206 266 142 156 236 224 201 137 248 214 222 183 276 263 112 318  98 129
 132 115 186 185 258 242 265 264 180 194 196 149 221 231 286 299 148 116
 287 251 268 150 249 161 130 124 152 108 147 126 274 244 143 304 232 136
 279 123  93 109 277 291 262 218 289  86 246 282 155 131 117 240 120 255
 285 133 284 302 254 111 140 261 127 272 118 267 310 335 250]


triglyceride =>
[153 128 120 366 200  74 331  62  69 177  91  35  39  71 105 130 118  37
  89  66  92  98  85  87 218  54 189 164 167 129 198  61 114  50 121 320
  34  51 165 174 264 203  97  52 187  64 343  82 208  95 341 101  56 216
 353 110 251 139  70  49 102  81 225 398  78 349 103 157  76  94 142  86
 159 380 143  72 149 217 179 223 226 136 109 180 210  68 112 155  88 301
 378 125  59  63 166 211 220 237  32 122 324 104  75  30 124 377 213 219
 232 205 345  79  93 161  47 233 314 363 111 133 191 173  43 339  99 256
 228 108 172 127 144 243 145 141 202  53  90 126  42 214 132 212  45 245
 182 271 147 134 123 146 261 119 252 106  73 303 257 162  55 224 156 275
 241 151 293 169 238 170 288 277 168 248 370 185 115 117 190 254 204  77
  57 113 160  22 192  83  48 196 292 276 337  84  60  58  41 188 236 137
 259 338 135 100  46  96 308 181 235 107  65  44 171 178 194 297 138 321
 294 333  67 280  40 240 154 186 300  31 222 229 258 385  29 199  26 184
 268  33 269  80 231 394 152 163 183 325 386 255 206 193  28 116 289 274
 244 201 140 234 175 131 253 207  36 279 399 270 150 347 365 299]


HDL =>
[ 70  71  57  46  47  98  39  58  60  41  63  59  87  72  49  53  79  61
  36  48  44  51  37  74  34  43  54  77  40  35  64  38  66  76  45  73
  67  56  55  52  86  62  30  32 103  50  80  89  65  83  93  42  69 101
  85  90  68  88  29  82  94  75  33  78  91  99  92 112  25  81  96  31
 100  24  84 108  97  28]


LDL =>
[142 114 112  91  92  64 102  99 104 107  73  69 144 167  88 149  78  95
  96 121 163 111 145 106 101  87 124  50 141 127 131 186 126 115  77 123
```

```
 150  72  79 125 105  83 143 175  67  85  58 132 169 103  61 165 120  93
  66 110 140 109  82 174 148 194 152  76  53  71 139 119 117 170 162  68
 122 133 191 182  98 134 154 201  86  97 129 113 100 151 176 137 168 161
  62  51 590  59 204  38 213  25  57  90 118 116  47 135 147 159 166 202
 156 128  89  74  80  94 155 184  75  63 160 157 172 130 190  65 177 183
  37 146  81  54 136 164  56 138  34  32  55 108 173 214 158  20 181 195
 153  30 178  42 197 171  70  84 187  45  44 209 220  16  43  48 189 247
 199]


hemoglobin =>
[19.8 15.9 13.7 16.9 14.9 13.9 16.5 14.  12.9 13.1 14.3 12.5 12.8 12.6
 13.4 16.4 15.4 16.1 15.1 14.8 16.3 11.7 13.  15.  11.6 16.  16.7 13.2
  7.5 15.5 15.6 15.8 13.5 12.7 11.4 15.2 17.1 16.2 14.7 12.1 14.5 17.
 17.8 21.1 15.7 14.4 16.6 15.3 14.1 12.2 13.3 18.4 14.6  7.1 12.3 14.2
 11.8 11.1 18.  13.6 17.5 16.8 12.4 17.9  8.8 17.3 11.5  8.3 17.6 13.8
 10.  11.2 10.5 11.9 12.   7.3  9.4 11.3  7.9  7.7 17.4  9.9 17.7 10.2
  9.6 10.8 10.4 17.2 11.  18.3  9.2]


'Urine =>
[1 3 2 5 4]


'serum =>
[1.  1.1 0.6 0.9 1.2 1.4 0.7 0.5 0.8 1.3 1.7 0.4 1.9 1.6 1.5 0.1 2.2]


AST =>
[  61   19 1090   32   26   47   29   17   22   23   24   18   20   27
   21   58   28   14   37   38   33   25   15   35   41   39   36   40
   16   89   30  114   43   44   49   59   31   34   73   55   45   51
   79  153   12  105   13  160   46   56   53   54  157   60   67   50
   42   48  126   11   68    9   52   71   93   75   72   65]


ALT =>
[ 115   25 1400   36   28   23   22   20   17   15   13   12   16   11
   64   35   78   30  165   33   31   14   45   21   61  104   10   39
   40   41  124   26   24   18   27   19   65   52   37   34   69   75
   67   32   43   63    5   76   59    7   42   62   29   44   86   95
   50  111   57   93   48   49    8   38   84   70  211   53   91   77
   66   58   82   55  118   47   72    9   46  148   51    6   79  120
   74   68   60    4   71   94   56  203   80   89]


Gtp =>
[125  30 276  36  15  70  19  32  14  56   9  11  10  12  18  65  33  79
```

```
 21  17  23  16  31  44  34  29 107  66  40  28  68  46  25  13  87  48
 38  51  37 145 231  74  59   8  61  35  24  55 150  97  39  49  22 117
 98  27  91  54  47  20 141  60  99 124 149 174  42  64  43  71 420  53
191  86 270 103 586 119 105  88 118  81  93  89 158  85  52 237  41 179
104 108  77  96  26  45  84  73 167 490 140  58  92  57  82  63 169 114
139  50 136 164  75 325 151 512 163 310 211 101 110  72 111 187 102 242
234 127  67 120 335 258 132  62  69 216 161   7  78 135  76 116 106  94
280 212]


'dental =>
[1 0]


smoking =>
[1 0]
```

[16]:
```
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)
```

[16]:

	id	age	height(cm)	weight(kg)	waist(cm)	eyesight(left)	\
808	808	2	7	7	85	10	
647	647	4	4	6	161	7	
749	749	7	6	7	109	9	
901	901	6	8	8	125	9	
554	554	6	2	6	99	10	
522	522	4	8	5	106	7	

	eyesight(right)	hearing(left)	hearing(right)	systolic	…	HDL	LDL	\
808	10	0	0	38	…	52	18	
647	3	0	0	29	…	24	72	
749	9	1	1	30	…	24	51	
901	10	0	0	41	…	28	72	
554	11	0	0	42	…	25	92	
522	6	0	0	23	…	31	52	

	hemoglobin	'Urine	'serum	AST	ALT	Gtp	'dental	smoking
808	42	0	6	13	24	44	1	1
647	35	0	6	6	1	8	0	0
749	65	0	7	33	25	116	0	1

```
901          50       0       7   12    5    8        1        0
554          45       0       4   18   20   10        0        0
522          53       0       5   20   19   19        0        1

[6 rows x 24 columns]
```

[17]:
```
#after encoding
for column in columns:
    print(column,"=>")
    print(data[column].unique())
    print("\n")
```

```
id =>
[   0    1    2 … 1015 1016 1017]


age =>
[ 3  0  5  8  4  6 11  7  1  2 10  9 12]


height(cm) =>
[ 6  7  3  5  4  8  2  0  1  9 10]


weight(kg) =>
[10 15  6  9  5  3 11  8  4  1  7 12  2 13  0 14 16 17]


waist(cm) =>
[197 229 137 182  99  80 189 131  53  78  41 156  34   5 171 125  83  93
 115  70  60  86 225 123  10 166 120  63 170 212  74   2  30  26 167  72
 146  17 150 106  36 144  22   4  13 194 209  69 132 162 235 183 201   6
  46 157  61 179 222  39 112 177  51  52 152  82 163 141 103  24  90 101
   3 135 108  66 159 205  14 202  65 175 102  40  98  67 127  85   8 126
 191  96 136 111  45 193 119 133 195 211 129 148  33  62  55 172 214 128
 192 231 174 220 116  43  31 165 226 233 124  19 224  77  20  11 190 110
  37 104  16 186 164 168 185 109 187 155 213 100  92 200  35 140 223  94
   1 230  75  15 139  64 105  73  47  38  56  71 173 160  84 147 217   7
 121 151  57 117 216  48 206 188 158 221 210  28  91  76 184 161  50  21
 203 134 219 114 107  23  89  44  88  87  97 153 207  27 154 181 113  95
 232  58  42  12 196 178   0   9 227 118  25 122  68 228 204  79 218  18
 145 149  32 234 176 130 199  49 143 208 215 198 180  29 138  59 142 169
  81  54]


eyesight(left) =>
[ 8  6  7 11  9 10  4  2  3  5  0  1 14 13 12]
```

```
eyesight(right) =>
[ 8  6  0  9 11  4  7 10  3  1  2  5 12 13]


hearing(left) =>
[0 1]


hearing(right) =>
[0 1]


systolic =>
[30 31 22 66 21 38 42  3 26 24 37 13 39 32 60 12 40 27 33 28 36 51 14 34
 29 70 71 47 58 25 46 18 55 17 41 20 59 35 10 44 68 64 45 16 73 50  4 19
  6  9 72 48 15 49  8 11 23 53 52 54 63 74 43 56 67 61  5 57 62 69 65  7
  2  0  1]


relaxation =>
[27 28 29 37 13 24  9  6 30 25 17 21 33 22 52 19 11 20 26  5 16 48 18 35
 31 32 39 10 38 34 49 53 23 15 56 14  8 51 36 41 43 57 12 42 47 45 54  1
 55  2  4 46  0  7  3 44 40 50]


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


Cholesterol =>
[126  98  80  97  66  64  94  57  65  71  41  23 117 145  50 113  51  61
  12 102  49  15  77  59 158  56  75 112  84  40  95  26  44  68 114  90
 155 107  85  62 115  87  91 128  63 132  54 106  78  52 121  60  58 143
 138 142  79  32 124  45  55  99  69  28 125  53 154 122 139  76  31  38
 110 100  82  47 130  74  92 116 162 104  33 174 103  46  89  86 120  96
  93 151  29  43 123 111  88  25 134 101 109  70 159 148   6 175   2  18
  21   7  73  72 144 129 150 149  67  81  83  36 108 118 166 170  35   8
 167 137 153  37 135  48  19  14  39   3  34  16 157 131  30 172 119  24
 161  13   1   4 160 169 147 105 168   0 133 163  42  20   9 127  11 141
 165  22 164 171 140   5  27 146  17 156  10 152 173 176 136]
```

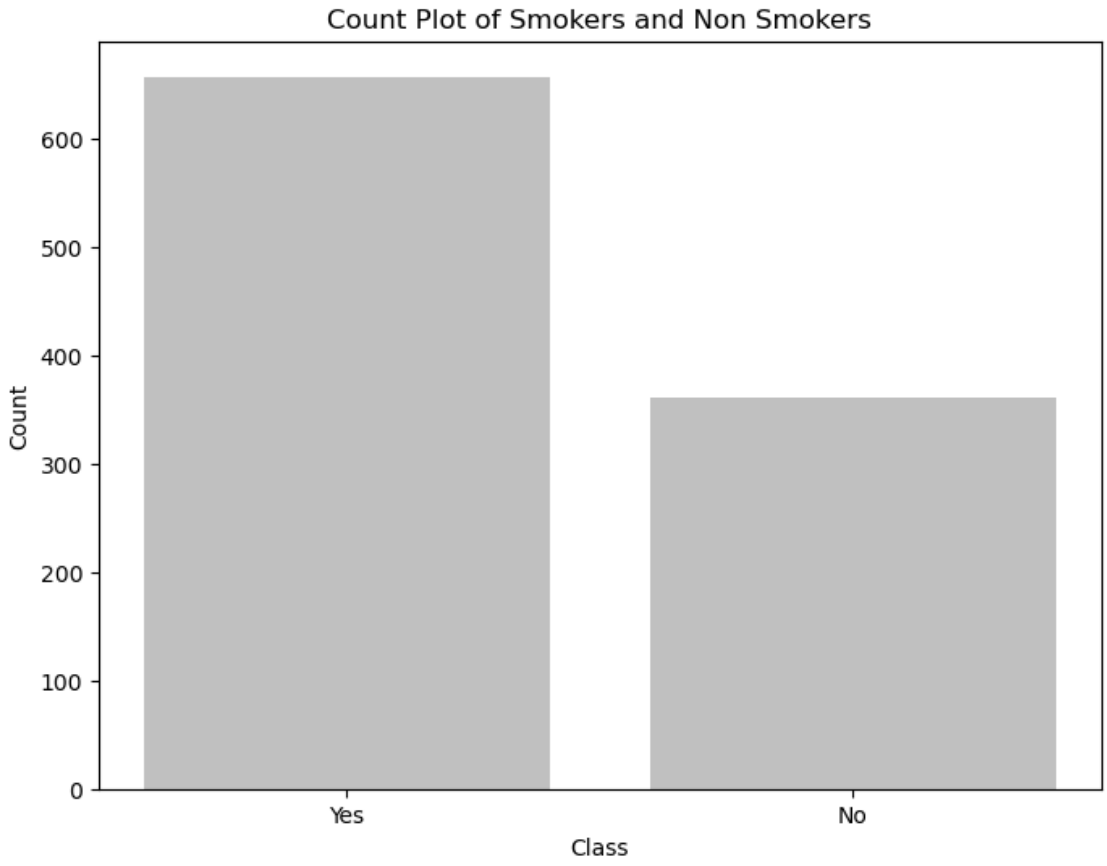
```
triglyceride =>
[125 101  93 258 168  47 245  35  42 147  64   9  12  44  78 103  91  11
  62  39  65  71  58  60 184  27 159 135 138 102 166  34  87  23  94 241
   8  24 136 145 218 171  70  25 157  37 251  55 176  68 250  74  29 182
 255  83 208 112  43  22  75  54 190 266  51 254  76 129  49  67 115  59
 130 262 116  45 121 183 149 188 191 109  82 150 177  41  85 127  61 237
 261  98  32  36 137 178 186 200   6  95 243  77  48   4  97 260 180 185
 195 173 252  52  66 132  20 196 240 256  84 106 161 144  16 249  72 213
 192  81 143 100 117 204 118 114 170  26  63  99  15 181 105 179  18 206
 152 222 120 107  96 119 217  92 209  79  46 238 214 133  28 189 128 224
 203 123 232 140 201 141 229 226 139 207 259 155  88  90 160 211 172  50
  30  86 131   0 162  56  21 165 231 225 247  57  33  31  14 158 199 110
 216 248 108  73  19  69 239 151 198  80  38  17 142 148 164 234 111 242
 233 246  40 228  13 202 126 156 236   5 187 193 215 263   3 167   1 154
 219   7 220  53 194 265 124 134 153 244 264 212 174 163   2  89 230 223
 205 169 113 197 146 104 210 175  10 227 267 221 122 253 257 235]


HDL =>
[44 45 31 20 21 71 13 32 34 15 37 33 61 46 23 27 53 35 10 22 18 25 11 48
  8 17 28 51 14  9 38 12 40 50 19 47 41 30 29 26 60 36  4  6 75 24 54 63
 39 57 67 16 43 74 59 64 42 62  3 56 68 49  7 52 65 72 66 77  1 55 69  5
 73  0 58 76 70  2]


LDL =>
[104  76  74  53  54  26  64  61  66  69  35  31 106 129  50 111  40  57
  58  83 125  73 107  68  63  49  86  14 103  89  93 145  88  77  39  85
 112  34  41  87  67  45 105 137  29  47  21  94 131  65  23 127  82  55
  28  72 102  71  44 136 110 150 114  38  16  33 101  81  79 132 124  30
  84  95 149 142  60  96 116 154  48  59  91  75  62 113 138  99 130 123
  24  15 162  22 156   7 158   2  20  52  80  78  12  97 109 121 128 155
 118  90  51  36  42  56 117 144  37  25 122 119 134  92 148  27 139 143
   6 108  43  17  98 126  19 100   5   4  18  70 135 159 120   1 141 151
 115   3 140   8 152 133  32  46 146  11  10 157 160   0   9  13 147 161
 153]


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


'Urine =>
[0 2 1 4 3]
```

```
'serum =>
[ 7  8  3  6  9 11  4  2  5 10 14  1 15 13 12  0 16]


AST =>
[50  9 67 22 16 37 19  7 12 13 14  8 10 17 11 47 18  4 27 28 23 15  5 25
 31 29 26 30  6 59 20 62 33 34 39 48 21 24 56 45 35 41 58 64  2 61  3 66
 36 46 43 44 65 49 52 40 32 38 63  1 53  0 42 54 60 57 55 51]


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


Gtp =>
[106  23 137  29   8  63  12  25   7  49   2   4   3   5  11  58  26  72
  14  10  16   9  24  37  27  22  95  59  33  21  61  39  18   6  78  41
  31  44  30 114 131  67  52   1  54  28  17  48 116  86  32  42  15 101
  87  20  81  47  40  13 113  53  88 105 115 124  35  57  36  64 142  46
 127  77 136  91 145 103  93  79 102  73  83  80 118  76  45 133  34 125
  92  96  70  85  19  38  75  66 122 143 112  51  82  50  74  56 123  99
 111  43 110 121  68 140 117 144 120 139 128  89  97  65  98 126  90 134
 132 107  60 104 141 135 108  55  62 130 119   0  71 109  69 100  94  84
 138 129]


'dental =>
[1 0]


smoking =>
[1 0]
```

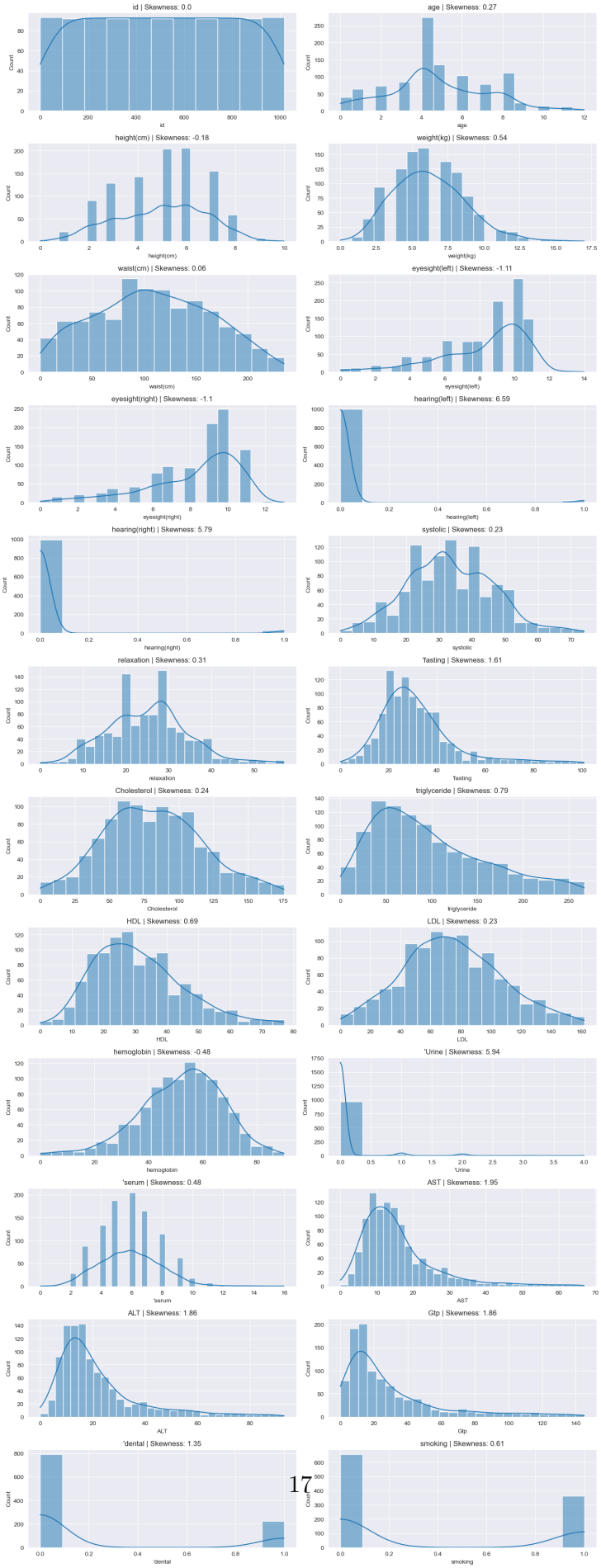
[18]:
```
rename = {'smoking{0':'smoking'}
data.rename(columns=rename, inplace=True)
```

[19]:
```
quality_counts = data['smoking'].value_counts()
labels = ['Yes','No']
plt.figure(figsize=(8, 6))
plt.bar(quality_counts.index, quality_counts, color='silver')
```

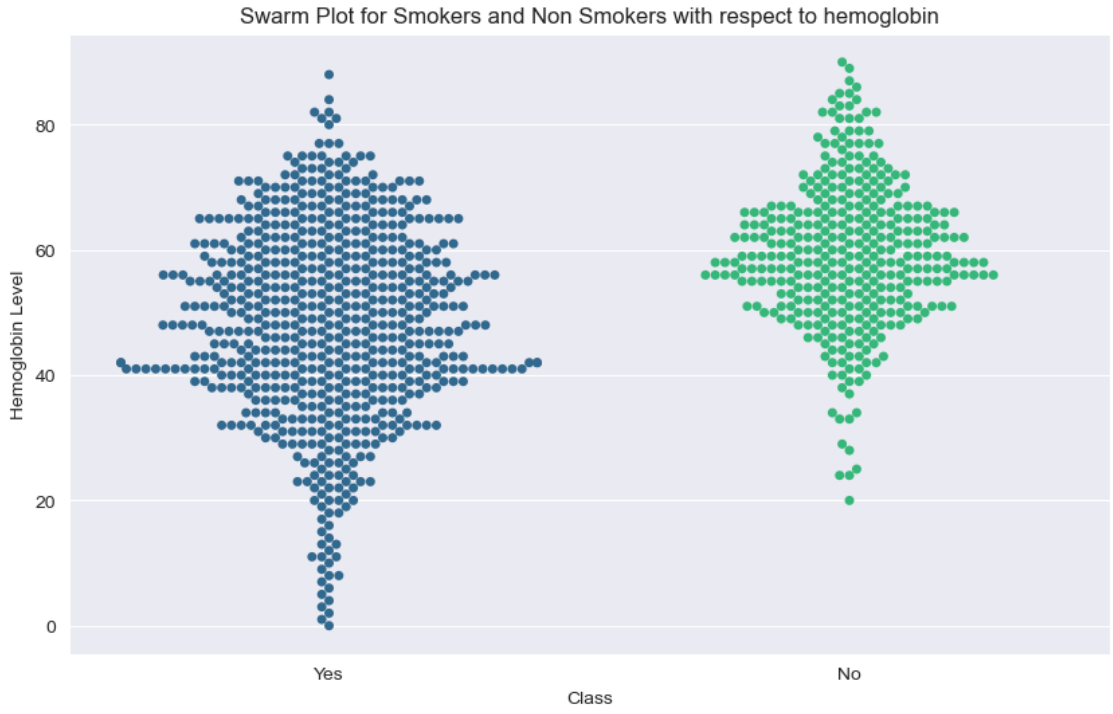
```
plt.title('Count Plot of Smokers and Non Smokers')
plt.xticks(range(2),labels)
plt.xlabel('Class')
plt.ylabel('Count')
plt.show()
```



[20]:
```
sns.set_style("darkgrid")
numerical_columns = data.select_dtypes(include=["int64", "float64"]).columns
plt.figure(figsize=(14, len(numerical_columns) * 3))
for idx, feature in enumerate(numerical_columns, 1):
        plt.subplot(len(numerical_columns), 2, idx)
        sns.histplot(data[feature], kde=True)
        plt.title(f"{feature} | Skewness: {round(data[feature].skew(), 2)}")
plt.tight_layout()
plt.show()
```


id | Skewness: 0.0
age | Skewness: 0.27
height(cm) | Skewness: -0.18
weight(kg) | Skewness: 0.54
waist(cm) | Skewness: 0.06
eyesight(left) | Skewness: -1.11
eyesight(right) | Skewness: -1.1
hearing(left) | Skewness: 6.59
hearing(right) | Skewness: 5.79
systolic | Skewness: 0.23
relaxation | Skewness: 0.31
'fasting | Skewness: 1.61
Cholesterol | Skewness: 0.24
triglyceride | Skewness: 0.79
HDL | Skewness: 0.69
LDL | Skewness: 0.23
hemoglobin | Skewness: -0.48
'Urine | Skewness: 5.94
'serum | Skewness: 0.48
AST | Skewness: 1.95
ALT | Skewness: 1.86
Gtp | Skewness: 1.86
'dental | Skewness: 1.35
smoking | Skewness: 0.61

[21]:
```
plt.figure(figsize=(10, 6))
sns.swarmplot(x="smoking", y="hemoglobin", data=data, palette='viridis')
plt.title('Swarm Plot for Smokers and Non Smokers with respect to hemoglobin')
plt.xlabel('Class')
plt.ylabel('Hemoglobin Level')
labels = ['Yes','No']
plt.xticks(range(2),labels)
plt.show()
```



[22]:
```
columns = [
    "age",
    "height(cm)",
    "weight(kg)",
    "waist(cm)",
    "systolic",
    "relaxation",
    "Cholesterol",
    "triglyceride",
    "HDL",
    "LDL",
    "hemoglobin",
```

```
    "AST",
    "ALT",
    "Gtp"
]
```

2 Undersampling

[23]:
```
data['smoking'].value_counts()
```

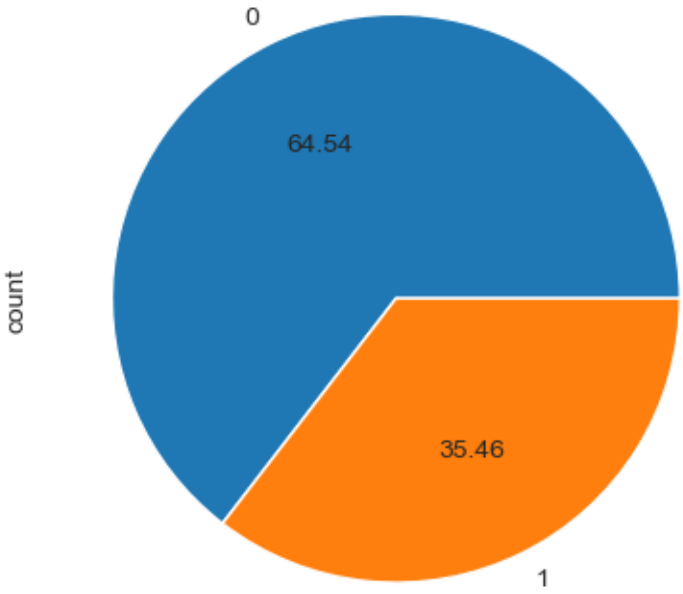
[23]:
```
smoking
0    657
1    361
Name: count, dtype: int64
```

[24]:
```
data = data.dropna(subset=columns + ['smoking'])
```

[25]:
```
x = data.drop(columns='smoking', axis=1)
y = data['smoking']
```

[26]:
```
#before undersampling
y.value_counts().plot.pie(autopct = '%.2f')
```

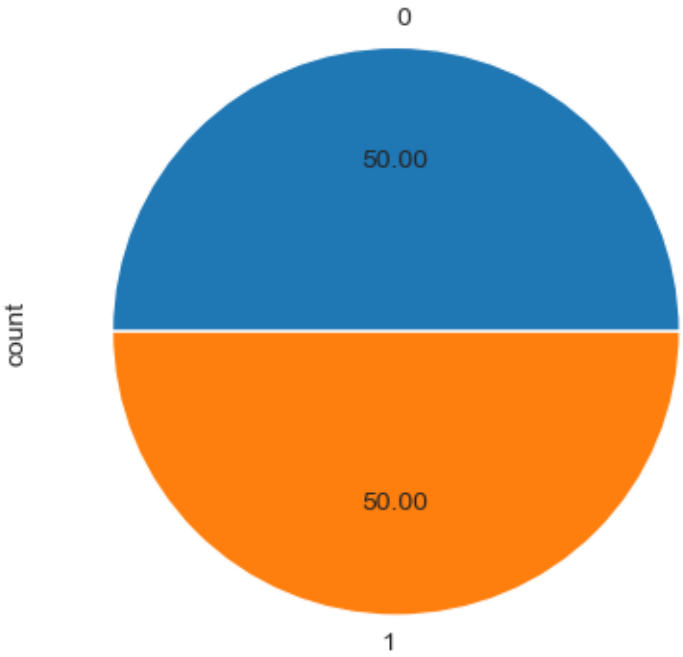
[26]:
```
<Axes: ylabel='count'>
```



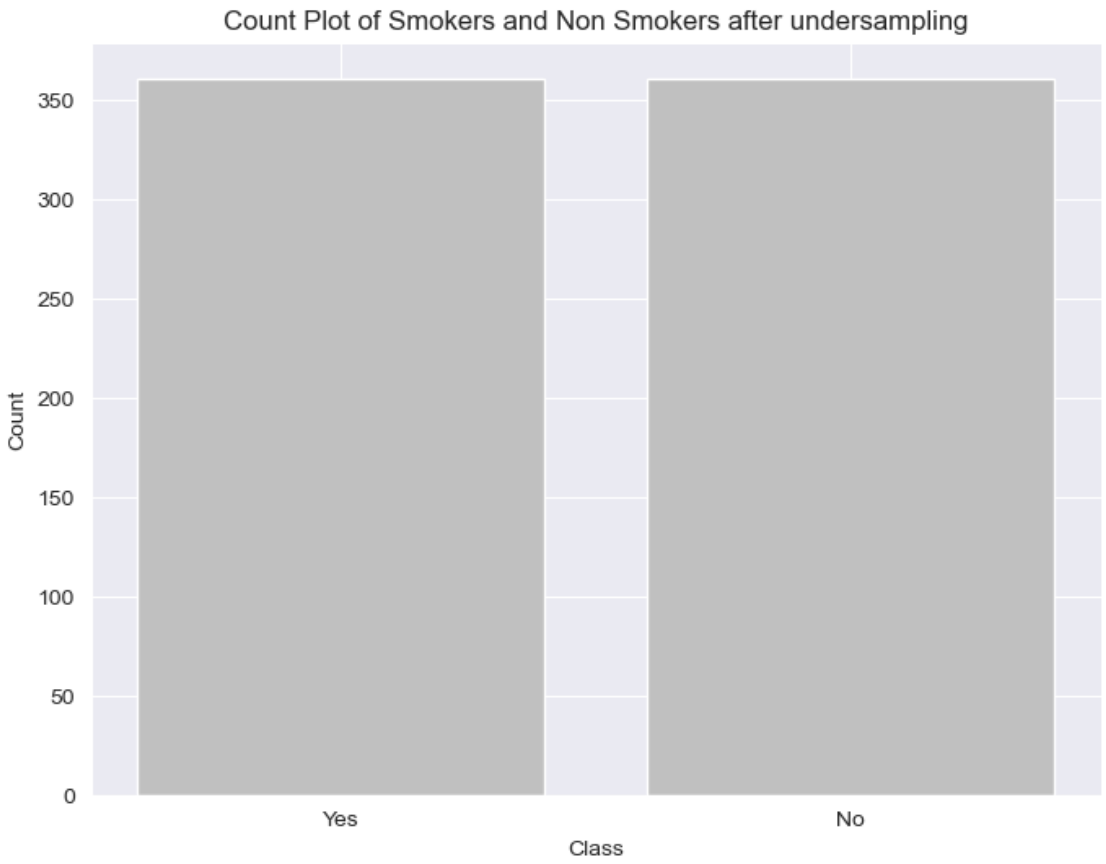
[27]:
```
from imblearn.under_sampling import RandomUnderSampler
rus = RandomUnderSampler(sampling_strategy=1)
x, y = rus.fit_resample(x, y)
```

[28]:
```
#after undersampling
y.value_counts().plot.pie(autopct = '%.2f')
```

[28]:
```
<Axes: ylabel='count'>
```



[29]:
```
quality_counts = y.value_counts()
labels = ['Yes','No']
plt.figure(figsize=(8, 6))
plt.bar(quality_counts.index, quality_counts, color='silver')
plt.title('Count Plot of Smokers and Non Smokers after undersampling')
plt.xticks(range(2),labels)
plt.xlabel('Class')
plt.ylabel('Count')
plt.show()
```



3 Model Building

[30]:
```
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.30,train_size
↪= 0.70,random_state = 42)
```

[31]:
```
print(x.shape, xtrain.shape, xtest.shape)
```

```
(722, 23) (505, 23) (217, 23)
```

[32]:
```
print(y.shape, ytrain.shape, ytest.shape)
```

```
(722,) (505,) (217,)
```

3.1 Logistic Regression

[33]:
```
model = LogisticRegression()
```

[34]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_data = scaler.fit_transform(xtrain)
```

[35]:
```
model.fit(scaled_data, ytrain)
```

[35]:
```
LogisticRegression()
```

[36]:
```
X_train_prediction = model.predict(scaled_data)
training_data_accuracy = accuracy_score(X_train_prediction, ytrain)
```

[37]:
```
print('Accuracy score on Test Data : ', training_data_accuracy)
```

```
Accuracy score on Test Data :  0.7524752475247525
```

[38]:
```
X_test_prediction_dt = model.predict(scaler.transform(xtest))
```

[39]:
```
test_data_accuracy = accuracy_score(X_test_prediction_dt, ytest)
```

[40]:
```
print('Accuracy score on Test Data : ', test_data_accuracy)
```

```
Accuracy score on Test Data :  0.7004608294930875
```

4 Random Forest

[41]:
```
model = RandomForestClassifier(n_estimators=100, random_state=42)
```

[42]:
```
model.fit(xtrain, ytrain)
```

[42]:
```
RandomForestClassifier(random_state=42)
```

[43]:
```
y_pred = model.predict(xtest)
```

[44]:
```
accuracy = accuracy_score(ytest, y_pred)
print(f'Model Accuracy: {accuracy * 100:.2f}%')
```

```
Model Accuracy: 70.51%
```

[45]:
```
feature_importance_df = pd.DataFrame({
    'feature': x.columns,
    'importance': model.feature_importances_
}).sort_values(by='importance', ascending=False)
print(feature_importance_df)
```

```
           feature  importance
16       hemoglobin    0.107073
2        height(cm)    0.099099
```

```
21              Gtp    0.082325
4         waist(cm)    0.057109
13     triglyceride    0.054287
12      Cholesterol    0.050983
18           'serum    0.048794
20              ALT    0.048250
0                id    0.047966
14              HDL    0.046947
15              LDL    0.045400
10       relaxation    0.043147
11         'fasting    0.041474
9          systolic    0.039360
19              AST    0.037126
3        weight(kg)    0.035392
5    eyesight(left)    0.035343
1               age    0.034308
6   eyesight(right)    0.031761
22          'dental    0.006919
17           'Urine    0.003671
8    hearing(right)    0.002530
7     hearing(left)    0.000737
```

[ ]:
```
def predict_credit_card_scam(input_data):
    input_data = input_data[numerical_columns]
    input_data_scaled = sklearn.preprocessing.StandardScaler().
 ↪fit_transform(input_data)
    predictions = model.predict(input_data_scaled)
    return predictions
def get_file():
    file_path = filedialog.askopenfilename()
    if file_path:
        selected_file_label.config(text="Selected file: " + file_path)
        global ted
        ted = pd.read_csv(file_path)
        print("File loaded into 'ted' variable.")
        ted2 = ted
        numerical_columns = ['id',
 'age',
 'height(cm)',
 'weight(kg)',
 'waist(cm)',
 'eyesight(left)',
 'eyesight(right)',
 'hearing(left)',
 'hearing(right)',
 'systolic',
 'relaxation',
```

```
 "'fasting",
 'Cholesterol',
 'triglyceride',
 'HDL',
 'LDL',
 'hemoglobin',
 "'Urine",
 "'serum",
 'AST',
 'ALT',
 'Gtp',
 "'dental",
 'smoking']
        ted = ted[numerical_columns]
        ted.columns = numerical_columns
        predictions = predict_credit_card_scam(ted)
        for i in predictions:
            if i==1:
                print(f"Smoking {i}")
            else:
                print(f"No Smoking {i}")
root = tk.Tk()
button = tk.Button(root, text="Select your test file", command=get_file)
button.pack(pady=100)
selected_file_label = tk.Label(root, text="Selected file: None")
selected_file_label.pack()
root.geometry("300x300")
root.mainloop()
```