In [90]:
```
import pandas as pd
import requests
from zipfile import ZipFile
from io import BytesIO
import numpy as np
pd.set_option('display.max_columns', None)
```

In [105…
```
response = requests.get("https://archive.ics.uci.edu/ml/machine-learning-databases/0
files = ZipFile(BytesIO(response.content))
df = pd.read_csv(files.open("dataset_diabetes/diabetic_data.csv"))
```

In [106…
```
#variavel para pegar as colunas do df em fomra de lista
cols = df.columns

#variavel para selecionar colunas numéricas do df
num_cols = df._get_numeric_data().columns

#variavel que diminui as colunas numéricas das colunas, obtendo-se as colunas categó
cat_col = list(set(cols) - set(num_cols))
```

In [107…
```
#laço for para valores únicos categóricos
for coluna in cat_col:
    print(coluna + str(df[coluna].unique()) + '\n')

#laço for para valores únicos numéricos
for col in num_cols:
    print( str(type(col)) + col + str(df[col].unique()) + '\n')
```

```
readmitted['NO' '>30' '<30']

glipizide-metformin['No' 'Steady']

troglitazone['No' 'Steady']

glyburide['No' 'Steady' 'Up' 'Down']

glimepiride['No' 'Steady' 'Down' 'Up']

chlorpropamide['No' 'Steady' 'Down' 'Up']

tolazamide['No' 'Steady' 'Up']

max_glu_serum['None' '>300' 'Norm' '>200']

payer_code['?' 'MC' 'MD' 'HM' 'UN' 'BC' 'SP' 'CP' 'SI' 'DM' 'CM' 'CH' 'PO' 'WC' 'OT'
 'OG' 'MP' 'FR']

rosiglitazone['No' 'Steady' 'Up' 'Down']

citoglipton['No']

change['No' 'Ch']

metformin-pioglitazone['No' 'Steady']

diag_3['?' '255' 'V27' '403' '250' 'V45' '38' '486' '996' '197' '250.6' '427'
 '627' '414' '416' '714' '428' '582' 'V43' '250.01' '263' '250.42' '276'
 '482' '401' '250.41' '585' '781' '278' '998' '568' '682' '618' '250.02'
 '305' '707' '496' '599' '715' '424' '518' '553' '794' '411' 'V42' '531'
 '511' '490' '562' '250.8' '250.7' '250.52' '784' '491' '581' '420' '8'
```

```
 '724' '730' '789' '131' '250.82' '999' '41' '493' '250.03' '753' '786'
 '529' 'E888' '425' '595' '303' '560' '711' '492' '332' '296' '438' '362'
 '250.4' '654' '244' 'V70' '737' '625' '681' '250.51' '404' 'V10' '810'
 '280' '440' '785' '588' '569' '272' '997' '250.43' '918' '584' '54' '788'
 '426' '722' '250.92' '196' '461' '535' '787' '891' '284' '458' '648'
 '780' '182' '285' '593' '413' '664' '564' '201' '356' 'V15' '292' '782'
 '473' '455' 'E932' '357' '348' '294' '250.23' '459' 'E878' '437' '733'
 '507' '525' '250.53' '397' '572' '805' '453' '331' '736' '402' '591'
 '576' '465' '533' '703' '349' '315' '658' '608' '578' '716' '382' '300'
 '282' '571' '536' '596' '287' '644' 'V11' '558' 'E885' '162' '198' '218'
 '412' '396' 'V14' '570' '433' 'E934' '882' '288' '577' '443' '729' '836'
 '295' '799' '281' '304' '153' '410' '616' '250.83' '601' '291' '75' '512'
 '660' '250.5' '598' '337' '574' '653' 'V58' '311' '415' '386' '602' '790'
 '112' '873' '620' '436' '70' '155' '138' '663' '530' '710' '42' '342'
 '250.91' 'E884' '515' '307' '704' '728' '731' '583' '238' '441' '293'
 '573' '532' '290' '594' '319' '250.13' '250.12' '519' '346' '380' '135'
 '642' '698' '924' '905' 'E933' '555' '309' 'E879' '286' '565' '752' '580'
 '446' '444' '344' '252' '35' '813' '394' '301' '575' '258' 'V17' '802'
 '435' '746' 'V12' '709' '881' 'E935' '139' '250.81' '718' '365' '202'
 '334' '185' '398' 'V44' '517' 'E849' '614' '466' '626' '250.9' '368'
 '605' '883' '289' '478' '617' '429' '442' 'V25' '866' '610' '557' '959'
 'E942' '94' '920' '345' '313' '379' '79' '516' '586' '821' '600' '242'
 '373' '592' 'V64' '487' '253' '706' 'E947' '117' '340' 'E950' '656'
 'E949' '590' 'V09' '250.22' '934' '694' '203' '250.93' '995' '726' '923'
 '958' '275' 'E929' '211' 'V18' 'V66' '199' '665' '53' '279' '522' '791'
 '890' '456' 'E938' 'E816' '122' '721' 'V65' '136' '480' '423' 'E920'
 '793' '647' '537' '351' '845' '336' '274' '719' '945' '434' '494' '227'
 '157' '208' '174' 'V57' '812' '734' '150' 'V23' '447' '692' '228' 'V16'
 '756' '405' 'E928' '823' '552' '528' '389' '240' '454' '792' '366' 'E939'
 '907' '270' '310' '266' '387' 'E931' '783' '245' '607' '355' 'E930' '705'
 '372' '369' '611' '283' 'V46' '110' '867' 'E956' '251' '250.2' '820'
 '712' '695' '567' '343' '723' 'V08' '273' '623' '807' '451' '495' '701'
 '34' 'V53' '314' '472' 'E945' '11' '189' '534' '354' '333' 'V54' '277'
 '659' '708' '452' '655' '816' '670' '621' '246' '953' '865' 'E817' '646'
 '151' '378' '78' '298' '840' '641' '521' '745' '619' '912' '506' 'E904'
 '259' 'E870' 'E980' '383' '204' '696' '566' '727' '47' 'E943' '358' '191'
 '965' '921' '432' '27' 'E861' '758' '477' '524' '751' '652' '556' '188'
 '825' '919' '732' '908' '951' '962' '685' 'E850' 'E944' '527' '341' '693'
 '250.1' 'V49' '860' '323' 'V55' '579' '508' '969' '205' '462' 'E880'
 '680' '697' '826' '200' '457' '717' '738' '742' '735' '235' '308' '725'
 '241' '824' '464' '260' '917' '239' '661' '892' '261' 'E883' '943' '744'
 'E936' '796' '318' '967' '350' '854' 'E905' '9' '741' 'E941' '170' '643'
 '317' '759' '909' 'V22' '831' '713' '180' '801' '360' '359' '501' '335'
 '250.11' '306' '811' '690' 'V02' '271' '214' '847' '543' 'V63' '906'
 '842' '686' '445' '808' '861' 'E852' '220' 'E887' 'E858' '915' '970'
 '256' '747' '395' '243' '815' '481' '5' 'E927' '297' '299' '851' '864'
 '922' '384' 'E876' '225' '158' 'E937' '871' '88' '966' 'E917' 'E812'
 'V62' 'E924' '604' '233' 'E916' '377' '797' 'V72' '172' '7' '421' '852'
 'E819' '972' '916' '956' '3' 'E965' '173' '193' '154' '347' '862' '250.3'
 '987' '470' '262' 'E855' '161' '115' '179' '910' '312' '17' '460' '265'
 '66' '163' 'V60' '870' 'E906' '514' '944' '844' '417' '152' '183' '991'
 '216' '385' '164' '935' '510' '814' '485' '850' '250.21' 'E919' '872'
 '195' '431' '597' '933' '171' '884' '156' '868' '483' 'E815' '542' 'V61'
 '853' '374' 'E881' 'E882' 'E822' '192' '754' '327' '523' '500' 'V85'
 '992' '657' '684' '603' 'E826' '550' '913' '376' '755' '361' '186' '720'
 '250.31' '674' '911' 'E813' '226' '365.44' 'E818' '146' '955' 'E894'
 '475' 'V13' '880' '930' 'E915' '381' '132' '353' '795' '893' 'V01' 'E853'
 '863' '540' 'E828' '430' '800' 'E865' '148' 'E946' '822' '879' '848'
 'V86' 'V03' '338' '989' '388' 'E966' '111' 'E922' '123' '757' 'E901'
 '141' '268' 'E892' '649' '702' '948' '223' '484' 'E886' '838' '928' '236'
 '624' '837' 'E987' 'V07' '841' '622' 'E912' 'E955' '463' 'V06' 'E864'
 '217' '877' '391' 'E825' '952' '669' '875' 'E900' '215' '538' '980' '834'
 '448' '175' '49' '876' '230' '57' 'E854' '942' '14' '750' '370' '671'
 '971']

diag_2['?' '250.01' '250' '250.43' '157' '411' '492' '427' '198' '403' '288'
 '998' '507' '174' '425' '456' '401' '715' '496' '428' '585' '250.02'
 '410' '999' '996' '135' '244' '41' '571' '276' '997' '599' '424' '491'
 '553' '707' '286' '440' '493' '242' '70' 'V45' '250.03' '357' '511' '196'
```

```
 '396' '197' '414' '250.52' '577' '535' '413' '285' '53' '780' '518' '150'
 '566' '250.6' '867' '486' 'V15' '8' '788' '340' '574' '581' '228' '530'
 '250.82' '786' '294' '567' '785' '512' '305' '729' '250.51' '280' '648'
 '560' '618' '444' '38' 'V10' '578' '277' '781' '250.42' '278' '426' '584'
 '462' '402' '153' '272' '733' '34' '881' '203' '250.41' '250.13' '293'
 '245' '250.12' '558' '787' '342' '573' '626' '303' '250.53' '458' '710'
 '415' 'V42' '284' '569' '759' '682' '112' '292' '435' '290' '250.93'
 '642' '536' '398' '319' '711' 'E878' '446' '255' 'V44' '250.7' '784'
 '300' '562' '162' '287' '447' '789' '790' '591' '200' '154' '304' '117'
 '847' '852' '250.83' '250.11' '816' '575' '416' '412' '441' '515' '372'
 '482' '382' 'V65' '572' '283' '78' '250.81' '576' '432' '595' '295' 'V12'
 '204' '466' '721' '434' '590' '271' '813' '368' '227' '783' '250.5' '258'
 '253' '309' '250.91' '519' '333' '459' '250.92' '250.4' '179' '420' '345'
 '433' '661' '537' '205' '722' '405' '437' '714' '211' 'E812' '263' '202'
 '397' '250.23' 'E932' '201' '301' '723' '614' '568' '861' 'V57' '724'
 '189' '297' '453' 'E888' '730' '354' '451' '738' 'E939' '805' 'V43' '155'
 '910' '218' '358' '220' 'E937' '583' '958' '794' '564' '436' '250.22'
 '620' '621' '331' '617' '596' '314' '378' '250.8' '625' '478' '731' '172'
 '404' '681' '470' '279' '281' '531' '443' '799' '337' '311' '719' 'E944'
 '423' 'E870' '465' 'E849' '782' '481' '480' 'V23' '199' '79' '438' '348'
 '42' 'E950' '473' '627' '726' '54' '490' '317' '332' '508' '369' '600'
 '349' '485' '208' '922' '431' '296' 'E934' '753' 'E935' '386' '728' '607'
 'E915' '344' '716' '289' '191' '873' '850' '611' '377' '352' '616' 'V17'
 '136' '455' '933' 'E885' '860' '513' '603' '484' '223' 'V72' '291' '151'
 'V58' '550' '510' '891' '185' '592' '791' '138' '598' '336' '362' '217'
 '825' '298' '821' 'E880' '343' '429' 'E879' '579' '225' '250.9' 'V49'
 '696' '233' '658' '969' '275' '250.1' '601' '704' '808' 'E890' 'V18'
 '920' '380' '570' 'E817' '359' '812' '274' 'V14' '324' '758' 'V66' '911'
 'E931' 'E924' '593' '792' '727' 'V46' '394' '532' 'V64' '557' '864' '718'
 'E942' '807' '604' '924' '820' '580' '273' '241' '282' '824' 'V61' '646'
 '701' '736' '565' '383' '250.2' 'E947' '452' '872' '905' 'E930' '921'
 '131' '448' '389' '421' '214' '705' '494' '752' '623' '9' '299' '959'
 '365' '967' 'E858' '40' '691' '909' '5' '814' '746' '250.31' '556' '680'
 '745' '351' '306' '110' '695' '552' '346' '918' '882' '947' '520' '188'
 '31' '356' '737' 'V08' '322' '182' '517' '974' 'E929' 'V53' '912' '252'
 '608' '516' 'E933' '94' '702' '923' '594' '647' '111' '934' '430' '487'
 '709' '796' '156' '977' '915' '756' '840' '341' '259' '693' '725' 'V62'
 '528' '683' '953' '457' '501' 'E900' 'V09' '522' '919' '461' '506' '193'
 '483' 'E936' '717' '802' '335' 'V54' '320' '945' '906' '239' '454' '826'
 '823' 'E941' '226' '795' '684' '844' '250.33' '308' '615' '588' '712'
 '663' '706' '833' '741' '713' '533' 'E884' '586' '555' '755' 'E928' '742'
 '869' '962' 'V11' '543' '373' '870' '913' '152' '810' '965' '907' '908'
 '995' '845' '474' '442' '751' '323' '472' '464' '686' '250.32' '540'
 '251' '811' '652' '659' '851' '422' '815' '307' '325' '463' '992' '692'
 '521' '917' 'E965' '524' '916' 'E813' '173' '238' '137' '514' '312' '837'
 '355' '980' '622' '475' '500' '754' '261' '801' '868' '968' '381' '11'
 '250.21' '694' '610' '734' 'E814' '310' '130' '246' '892' '846' '634'
 '75' 'E927' 'E905' '183' '379' 'E917' '163' 'E868' '495' '747' '989'
 'E854' '240' '832' '605' '602' '644' 'V16' '35' 'V70' '376' '266' 'E918'
 '619' '477' '656' '46' '883' '171' 'V13' '698' '842' 'E850' '800' '269'
 '664' 'E887' '952' '164' 'E881' '527' '685' '366' '836' '27' 'V63' '865'
 '793' '232' '990' '52' '831' '327' '542' '806' '972' '862' 'E829' 'E919'
 '944' 'E916' '963' '316' '645' '347' 'V85' '374' 'V02' '748' '256' '186'
 '866' '975' '96' '395' '262' 'E819' '654' '994' '318' 'E826' '879' '674'
 '641' '822' '145' '797' '353' 'E938' 'E816' '948' '987' '99' '192'
 '250.3' 'E906' '534' '115' 'E818' 'E980' '360' '338' '529' '871' '750'
 '212' '302' '955' '141' '88' 'V25' '215' '350' 'V50' 'V03' 'E853' 'E968'
 'E882' '140' '703' '991' '893' 'E821' '235' 'V69' '670' '195' 'V55' '388'
 '268' '894' '114' '260' '853' '7' '880' 'V86' '180' 'E945' '523' '863'
 '649' '270' '665' '460' '942' '364' '66' 'E883' '123' '884' 'V60' '843'
 '927']

diabetesMed['No' 'Yes']

diag_1['250.83' '276' '648' '8' '197' '414' '428' '398' '434' '250.7' '157'
 '518' '999' '410' '682' '402' '737' '572' 'V57' '189' '786' '427' '996'
 '277' '584' '462' '473' '411' '174' '486' '998' '511' '432' '626' '295'
 '196' '250.6' '618' '182' '845' '423' '808' '250.4' '722' '403' '250.11'
 '784' '707' '440' '151' '715' '997' '198' '564' '812' '38' '590' '556'
```

```
 '578' '250.32' '433' 'V58' '569' '185' '536' '255' '250.13' '599' '558'
 '574' '491' '560' '244' '250.03' '577' '730' '188' '824' '250.8' '332'
 '562' '291' '296' '510' '401' '263' '438' '70' '250.02' '493' '642' '625'
 '571' '738' '593' '250.42' '807' '456' '446' '575' '250.41' '820' '515'
 '780' '250.22' '995' '235' '250.82' '721' '787' '162' '724' '282' '514'
 'V55' '281' '250.33' '530' '466' '435' '250.12' 'V53' '789' '566' '822'
 '191' '557' '733' '455' '711' '482' '202' '280' '553' '225' '154' '441'
 '250.81' '349' '?' '962' '592' '507' '386' '156' '200' '728' '348' '459'
 '426' '388' '607' '337' '82' '531' '596' '288' '656' '573' '492' '220'
 '516' '210' '922' '286' '885' '958' '661' '969' '250.93' '227' '112'
 '404' '823' '532' '416' '346' '535' '453' '250' '595' '211' '303'
 '250.01' '852' '218' '782' '540' '457' '285' '431' '340' '550' '54' '351'
 '601' '723' '555' '153' '443' '380' '204' '424' '241' '358' '694' '331'
 '345' '681' '447' '290' '158' '579' '436' '335' '309' '654' '805' '799'
 '292' '183' '78' '851' '458' '586' '311' '892' '305' '293' '415' '591'
 '794' '803' '79' '655' '429' '278' '658' '598' '729' '585' '444' '604'
 '727' '214' '552' '284' '680' '708' '41' '644' '481' '821' '413' '437'
 '968' '756' '632' '359' '275' '512' '781' '420' '368' '522' '294' '825'
 '135' '304' '320' '250.31' '669' '868' '496' '250.43' '826' '567' '3'
 '203' '53' '251' '565' '161' '495' '49' '250.1' '297' '663' '576' '355'
 '850' '287' '250.2' '611' '840' '350' '726' '537' '620' '180' '366' '783'
 '11' '751' '716' '250.3' '199' '464' '580' '836' '664' '283' '813' '966'
 '289' '965' '184' '480' '608' '333' '972' '212' '117' '788' '924' '959'
 '621' '238' '785' '714' '942' '250.23' '710' '47' '933' '508' '478' '844'
 '7' '736' '233' '42' '250.5' '397' '395' '201' '421' '253' '250.92' '600'
 '494' '977' '39' '659' '312' '614' '647' '652' '646' '274' '861' '425'
 '527' '451' '485' '217' '250.53' '442' '970' '193' '160' '322' '581'
 '475' '623' '374' '582' '568' '465' '801' '237' '376' '150' '461' '913'
 '226' '617' '987' '641' '298' '790' '336' '362' '228' '513' '383' '746'
 '353' '911' '506' '873' '155' '860' '534' '802' '141' 'V45' '396' '310'
 '341' '242' '719' '239' '533' '616' '519' '301' 'V66' '5' '989' '230'
 '385' '300' '853' '871' '570' '848' '463' '9' '934' '250.21' '236' '361'
 '594' '501' '810' '643' '430' '528' '205' '791' '983' '992' '490' '172'
 '171' '622' '306' '863' '864' '474' '660' '759' '356' '634' '967' '551'
 '695' '187' '732' '747' '323' '308' '370' '252' '152' '846' '164' '365'
 '718' '48' '266' '720' '94' '344' '797' '170' '878' '904' 'V56' '882'
 '843' '709' '973' '454' '686' '939' '487' '229' '991' '483' '357' '692'
 '796' '693' '935' '936' '800' '920' 'V26' '261' '307' '262' '250.9' '831'
 '145' '223' 'V71' '839' '685' 'V54' '35' '34' '179' '964' '136' '324'
 '389' '815' '334' '143' '526' '588' '192' 'V67' '394' '917' '88' '219'
 '325' '792' '717' '994' '990' '793' '207' '637' '195' '373' '847' '827'
 '31' '891' '814' 'V60' '703' '865' '352' '627' '378' '342' '886' '369'
 '745' '705' '816' '541' '986' '610' '633' '640' '753' '173' '835' '379'
 '445' '272' '382' '945' '619' '881' '250.52' '866' '405' '916' '215'
 '893' '75' '671' '928' '906' '897' '725' '867' '115' '890' '734' '521'
 '674' '470' '834' '146' '696' '524' '980' '691' '384' '142' '879'
 '250.51' '246' '208' '448' '955' '653' '149' '245' '735' '883' '854'
 '952' '838' '194' 'V43' '163' '216' '147' '354' '27' '477' '318' '880'
 '921' '377' '471' '683' '175' '602' '250.91' '982' '706' '375' '417'
 '131' '347' '870' '148' '862' '61' '817' '914' '360' '684' '314' 'V63'
 '36' '57' '240' '915' '971' '795' '988' '452' '963' '327' '731' '842'
 'V25' '645' '665' '110' '944' '603' '923' '412' '363' '957' '976' '698'
 '299' '700' '273' '974' '97' '529' '66' '98' '605' '941' '52' '806' '84'
 '271' '837' '657' '895' '338' '523' '542' '114' '543' '372' 'V70' 'E909'
 '583' 'V07' '422' '615' '279' '500' '903' '919' '875' '381' '804' '704'
 '23' '58' '649' '832' '133' '975' '833' '391' '690' '10' 'V51']

metformin-rosiglitazone['No' 'Steady']

tolbutamide['No' 'Steady']

medical_specialty['Pediatrics-Endocrinology' '?' 'InternalMedicine'
 'Family/GeneralPractice' 'Cardiology' 'Surgery-General' 'Orthopedics'
 'Gastroenterology' 'Surgery-Cardiovascular/Thoracic' 'Nephrology'
 'Orthopedics-Reconstructive' 'Psychiatry' 'Emergency/Trauma'
 'Pulmonology' 'Surgery-Neuro' 'Obsterics&Gynecology-GynecologicOnco'
 'ObstetricsandGynecology' 'Pediatrics' 'Hematology/Oncology'
 'Otolaryngology' 'Surgery-Colon&Rectal' 'Pediatrics-CriticalCare'
 'Endocrinology' 'Urology' 'Psychiatry-Child/Adolescent'
```

```
 'Pediatrics-Pulmonology' 'Neurology' 'Anesthesiology-Pediatric'
 'Radiology' 'Pediatrics-Hematology-Oncology' 'Psychology' 'Podiatry'
 'Gynecology' 'Oncology' 'Pediatrics-Neurology' 'Surgery-Plastic'
 'Surgery-Thoracic' 'Surgery-PlasticwithinHeadandNeck' 'Ophthalmology'
 'Surgery-Pediatric' 'Pediatrics-EmergencyMedicine'
 'PhysicalMedicineandRehabilitation' 'InfectiousDiseases' 'Anesthesiology'
 'Rheumatology' 'AllergyandImmunology' 'Surgery-Maxillofacial'
 'Pediatrics-InfectiousDiseases' 'Pediatrics-AllergyandImmunology'
 'Dentistry' 'Surgeon' 'Surgery-Vascular' 'Osteopath'
 'Psychiatry-Addictive' 'Surgery-Cardiovascular' 'PhysicianNotFound'
 'Hematology' 'Proctology' 'Obstetrics' 'SurgicalSpecialty' 'Radiologist'
 'Pathology' 'Dermatology' 'SportsMedicine' 'Speech' 'Hospitalist'
 'OutreachServices' 'Cardiology-Pediatric' 'Perinatology'
 'Neurophysiology' 'Endocrinology-Metabolism' 'DCPTEAM' 'Resident']

repaglinide['No' 'Up' 'Steady' 'Down']

acetohexamide['No' 'Steady']

weight['?' '[75-100)' '[50-75)' '[0-25)' '[100-125)' '[25-50)' '[125-150)'
 '[175-200)' '[150-175)' '>200']

glyburide-metformin['No' 'Steady' 'Down' 'Up']

race['Caucasian' 'AfricanAmerican' '?' 'Other' 'Asian' 'Hispanic']

insulin['No' 'Up' 'Steady' 'Down']

A1Cresult['None' '>7' '>8' 'Norm']

pioglitazone['No' 'Steady' 'Up' 'Down']

glimepiride-pioglitazone['No' 'Steady']

examide['No']

glipizide['No' 'Steady' 'Up' 'Down']

miglitol['No' 'Steady' 'Down' 'Up']

metformin['No' 'Steady' 'Up' 'Down']

acarbose['No' 'Steady' 'Up' 'Down']

age['[0-10)' '[10-20)' '[20-30)' '[30-40)' '[40-50)' '[50-60)' '[60-70)'
 '[70-80)' '[80-90)' '[90-100)']

gender['Female' 'Male' 'Unknown/Invalid']

nateglinide['No' 'Steady' 'Down' 'Up']

<class 'str'>encounter_id[  2278392    149190     64410 ... 443854148 443857166 4438
67222]

<class 'str'>patient_nbr[  8222157  55629189  86047875 ... 140199494 120975314 17542
9310]

<class 'str'>admission_type_id[6 1 2 3 4 5 8 7]

<class 'str'>discharge_disposition_id[25  1  3  6  2  5 11  7 10  4 14 18  8 13 12 1
6 17 22 23  9 20 15 24 28
 19 27]

<class 'str'>admission_source_id[ 1  7  2  4  5  6 20  3 17  8  9 14 10 22 11 25 13]

<class 'str'>time_in_hospital[ 1  3  2  4  5 13 12  9  7 10  6 11  8 14]

<class 'str'>num_lab_procedures[ 41  59  11  44  51  31  70  73  68  33  47  62  60
55  49  75  45  29
```

```
 35  42  66  36  19  64  25  53  52  87  27  37  46  28  48  72  10   2
 65  67  40  54  58  57  43  32  83  34  39  69  38  56  22  96  78  61
 88  50   1  18  82   9  63  24  71  77  81  76  90  93   3 103  13  80
 85  16  15  12  30  23  17  21  79  26   5  95  97  84  14  74 105  86
 98  20   6  94   8 102 100   7  89  91  92   4 101  99 114 113 111 129
107 108 106 104 109 120 132 121 126 118]

<class 'str'>num_procedures[0 5 1 6 2 3 4]

<class 'str'>num_medications[ 1 18 13 16  8 21 12 28 17 11 15 31  2 23 19  7 20 14 1
0 22  9 27 25  4
 32  6 30 26 24 33  5 39  3 29 61 40 46 41 36 34 35 50 43 42 37 51 38 45
 54 52 49 62 55 47 44 53 48 57 59 56 60 63 58 70 67 64 69 65 68 66 81 79
 75 72 74]

<class 'str'>number_outpatient[ 0  2  1  5  7  9  3  8  4 12 11  6 20 15 10 13 14 16
21 35 17 29 36 18
 19 27 22 24 42 39 34 26 33 25 23 28 37 38 40]

<class 'str'>number_emergency[ 0  1  2  4  3  9  5  7  6  8 22 25 10 13 42 16 11 28
15 14 18 12 21 20
 19 46 76 37 64 63 54 24 29]

<class 'str'>number_inpatient[ 0  1  2  3  6  5  4  7  8  9 15 10 11 14 12 13 17 16
21 18 19]

<class 'str'>number_diagnoses[ 1  9  6  7  5  8  3  4  2 16 12 13 15 10 11 14]
```

In [108…

```python
#valores a serem excluídos
drop_values = ["?","Unknown/Invalid"]
```

In [109…

```python
df
```

Out[109…

	encounter_id	patient_nbr	race	gender	age	weight	admission_type_id	dischar
0	2278392	8222157	Caucasian	Female	[0-10)	?	6	
1	149190	55629189	Caucasian	Female	[10-20)	?	1	
2	64410	86047875	AfricanAmerican	Female	[20-30)	?	1	
3	500364	82442376	Caucasian	Male	[30-40)	?	1	
4	16680	42519267	Caucasian	Male	[40-50)	?	1	
...	...	...	...	...	...	...	...	
101761	443847548	100162476	AfricanAmerican	Male	[70-80)	?	1	
101762	443847782	74694222	AfricanAmerican	Female	[80-90)	?	1	
101763	443854148	41088789	Caucasian	Male	[70-80)	?	1	
101764	443857166	31693671	Caucasian	Female	[80-90)	?	2	

	encounter_id	patient_nbr	race	gender	age	weight	admission_type_id	dischar
101765	443867222	175429310	Caucasian	Male	[70-80)	?	1	

101766 rows × 50 columns

In [110…

```
#lógica para excluir linhas que possuem valores na lista drop_values
df = df[~df.isin(drop_values).any(axis=1)]

cols = df.columns
```

In [111…

```
#variavel para selecionar colunas numéricas do df
num_cols = df._get_numeric_data().columns

#variavel que diminui as colunas numéricas das colunas, obtendo-se as colunas categó
cat_col = list(set(cols) - set(num_cols))

#laço for para valores únicos categóricos
for coluna in cat_col:
    print(coluna + str(df[coluna].unique()) + '\n')

#laço for para valores únicos numéricos
for col in num_cols:
    print( str(type(col)) + col + str(df[col].unique()) + '\n')
```

```
readmitted['<30' 'NO' '>30']

glipizide-metformin['No']

troglitazone['No']

glyburide['No' 'Steady' 'Up' 'Down']

glimepiride['No' 'Steady' 'Down' 'Up']

chlorpropamide['No']

tolazamide['No']

max_glu_serum['None']

payer_code['UN' 'CP' 'DM' 'BC' 'MC' 'HM' 'OT' 'MD' 'CM' 'SP' 'WC']

rosiglitazone['No' 'Steady']

citoglipton['No']

change['Ch' 'No']

metformin-pioglitazone['No']

diag_3['428' '403' '401' '70' '250' '276' '250.01' '780' '715' '396' '424' '41'
 '996' '584' '578' '593' '305' '348' '333' '427' '414' '285' '599' '394'
 '682' '250.6' 'E878' '511' '158' '492' 'E812' '402' '591' '455' '530'
 '197' '411' '274' '404' '496' 'E849' '458' '425' '416' 'E942' '790' '491'
 '272' '280' '486' '218' 'E879' '789' '998' '287' '357' '617' '202' '719'
 '515' '397' '571' '787' '574' '721' '250.02' '443' '135' '728' '250.8'
 '707' 'E885' '196' '295' '410' '997' '250.13' 'E935' '518' '517' '412'
 '466' '562' '792' 'E888' '298' '493' '733' '681' '785' '440' '244' '573'
 '426' '867' 'V42' '453' '278' '553' '537' 'E947' '478' '152' '596' '459'
```

```
 '531' '731' 'V12' '250.41' '362' '788' 'E919' '522' '377' '716' '198'
 '725' '642' '250.92' '710' '331' '620' '711' '378' '850' '8' '250.4'
 'E939' '482' 'V43' 'E917' '585' '250.42' 'V45' '674' 'V27' '568' '311'
 'E931' '211' 'V58' '252' '162' '540' 'V09' 'E944' '738' '289' '714' '256'
 '112' 'E934' '138' '201' '730' '729' '250.43' '153' 'V15' '560' '327'
 '726' '588' '236' 'V85' '611' '296' '456' '799' '533' '457' '535' '433'
 '873' '294' '784' '366' '312' '413' '958' '49' '204' '564' '429' '600'
 '438']

diag_2['402' '780' '411' '303' '199' '403' '425' '478' '250' 'E878' '428' '427'
 '414' '730' '424' '584' '8' '996' '162' '682' '518' '401' '285' '512'
 '288' '733' '41' '578' 'V42' '404' '250.01' '496' '276' '788' '38' '197'
 '790' '581' '592' '413' '200' '815' '599' '569' '782' '491' '198' '577'
 'E879' '729' '250.41' '727' 'E885' '305' 'E935' '618' 'E942' '596' '280'
 '571' '493' '574' '250.02' '511' '614' '112' '591' '218' '707' '250.4'
 '250.81' '536' '490' '342' '553' '568' '724' '475' '324' '284' '203'
 '881' '410' '272' '356' '515' '255' '426' '560' '135' '552' '535' '287'
 '345' '617' 'E944' '396' '300' '357' '789' 'E884' '995' '528' '394' '486'
 '530' '919' '292' 'E947' '466' '648' '250.6' '464' '202' '808' '710'
 '233' '585' '277' '453' '136' '590' '154' '250.11' '250.51' '784' '211'
 '459' 'V10' '883' '250.92' '188' '787' '304' '79' '289' '623' '473' '245'
 '70' '196' 'V15' '156' '378' '864' '963' 'E932' 'V58' '153' '442' 'V43'
 '327' '924' 'E917' '625' '785' '456' '620' '718' '723' '412' 'E816' '205'
 'E928' '214' '434' '746' 'E934' '715' 'E881' 'E941' 'E888' '998' '151'
 'E936' '9' '626' '920' '997' '532' '731' '250.13' '719' '850' '816' '455'
 '594' '295' '278' '781' '204' '753' '550' 'V45' '745' '458' '807' '331'
 '756' '296' '812' '348' '435' '193' '250.2' 'V85' '722' '440' '332' '555']

diabetesMed['Yes' 'No']

diag_1['715' '250.2' '414' '296' '196' '250.4' '786' '574' '727' '998' '996'
 '410' '250.8' '188' '287' '402' '789' '440' '532' '198' '250' '431' '386'
 '572' '427' '428' '8' '531' '569' '493' '707' '577' '250.3' '780' '404'
 'V55' 'V58' '599' '225' '591' '491' '578' '276' '891' '562' '437' '733'
 '584' 'V54' '413' '38' '415' '824' '204' '250.11' '537' '618' '486' '590'
 '433' '726' '518' '182' '458' '682' '434' '592' '250.6' '233' '295' '530'
 '453' '331' '560' '465' '274' '435' '153' '246' '154' '250.7' '722' '873'
 '490' '211' '285' 'V53' '482' '189' '230' '280' '250.13' '617' '297'
 '155' '812' '550' '997' '425' '432' '579' '353' '426' '250.02' '575'
 '724' '218' '459' '438' '284' '784' '642' '729' '193' '403' '564' '202'
 '820' 'V45' '197' '174' '716' '443' '604' '721' '852' '401' '557' '816'
 '250.81' '340' '511' '346' '664' '185' '201' '536' '241' '250.22' '576'
 '782' '252' '411' '405' '507' '865' '965' '250.01' '593' '641' '924'
 '756' '728' '250.41' '162' '648' '658' '626' '552' 'V71' '53' '558' '787'
 '429' '250.82' '455' '430' '881' '555' '534' '351' '736' '424' '805'
 '288' '294' '711' '785' '790' '220' '250.92' '710' '35' '456' '799' '466'
 '250.12' '250.83' '515' '151' '840' '659' '566' '535' '239' '236' '310'
 '157' '292' '644' '850' '553' '398' '600' '447' '808']

metformin-rosiglitazone['No']

tolbutamide['No']

medical_specialty['Surgery-General' 'Family/GeneralPractice' 'Cardiology' 'Psychiatr
y'
 'InternalMedicine' 'ObstetricsandGynecology' 'Dentistry' 'Pediatrics']

repaglinide['No' 'Steady' 'Up']

acetohexamide['No']

weight['[75-100)' '[50-75)' '[100-125)' '[125-150)' '[25-50)' '[150-175)'
 '[175-200)' '[0-25)']

glyburide-metformin['No']

race['Caucasian' 'AfricanAmerican' 'Other' 'Asian']

insulin['Steady' 'No' 'Down' 'Up']
```

```
A1Cresult['None' 'Norm' '>8' '>7']

pioglitazone['No' 'Steady' 'Up' 'Down']

glimepiride-pioglitazone['No']

examide['No']

glipizide['Steady' 'No' 'Up' 'Down']

miglitol['No']

metformin['No' 'Steady' 'Up' 'Down']

acarbose['No' 'Steady']

age['[70-80)' '[80-90)' '[60-70)' '[40-50)' '[50-60)' '[90-100)' '[20-30)'
 '[30-40)' '[0-10)' '[10-20)']

gender['Female' 'Male']

nateglinide['No' 'Steady']

<class 'str'>encounter_id[ 88792836  88986678  89032962 ... 437959022 439577312 4396
06454]

<class 'str'>patient_nbr[100654011  58682736  69250302  62022042  30950811  58763808
63813420
  84387969 110949741  49167621  56356434 109527102  78098634  21850101
  99090900  79327116  83177253 106419474  83232054 108730161  49469211
  77730093 114086232  84746214  20875761  38644884   3749778  63002484
 107551395 101722149 110721573  99154062  63872937 110306691  85456377
  75016944   5347143  70932573  87758757  16352748 108564255  58151448
 113098887  80369244  97780491  50173128 113262084  67316697 109737909
  25685226 100627389 104373990  61658370  91441701  39852603  87254991
  74547189  70100586   2554614  87701625   2728863 105968844  46422936
  72874890  80348409  77525172  88485201  22855590 100179639  86806809
 103359609  16853211  60125787 100592829  75753909  16622676 107613405
  85160916  41828310 113203098  75511719  47738574  63453312  52100514
  53485947  38785977 114508206  65129058  94630329  64493568 102548961
   3771018  68278050  66399966  62785548  98244774  88116966 113452965
   5303844  78806592  87737328  28219392  62105535   1497051 108595224
 105565014  72537417  60014520 101705697  37413621  37247364  74804031
  91503747  59484456  75963663  59722938  69795441  79927353 100912059
  21936186  98288091 110854314   7040790  91153809  73368684  59874444
  56929257 114852780  30068118  42241986  22994640  56410452  79591419
  77919759  97115823  97522011 111853026  87222429  89135919  78893280
  84937059  69176511  65624859  28177659  72835722  11617695  93379203
  89169642  55827234  85784589 100893636  13393998  55662687  98626689
  88183908  85233987  84653451  54143262  11246814  83081124  79415739
  61583247   1648098  72468270     89469   5660262  77191740  45372672
  22889808 100095786  65530926  74629494  56127150 105211854  86407056
  59131863 103384989  81695034  91480338  66197817  53341551  21265290
  66475818  43481835  84606228  61200441  93432033  49159611  72980181
   6033906  80265726  92701881   9777123  48758103  52657614  68925204
  60225480  82264860  89608131  85294989  65081178  80202366  78637509
 104270778  66392199 111292938  45551070  34597422  66907593 112806657
  41952078    384696  25776972  90818226 104301927  92529045  56532105
  98886357  75752478  74364813  53829720  84194640  69754275  55193310
  49215114 114379875  92807865 112653738   7467957   6571683  97702812
  64778958  77741667  55799235 103373586  68656032  94905882  78114438
  56460321 104326587  92496681 109240173  38623563  21261294  55992978
 110395755 112622724  21709845  88182144  18556578 112874778  96965082
  77151915  40153572  41899392  23098716  92992446   7512273  52453962
  82335024  77841360  19821492  64473588  66437532 105401079 102285594
 109364517 100378215  97221033 105239097  89068014 103590630 113396184
  54115434  99928710  63119061 110673657   4312467 114798618  60095250
 115040115  91446426  65618370  89130672 110319021 101447343  89405451
```

```
  57434598   7247070  82938132  98993592  21834108  96484509  83842281
  94244436  98221716 101469798  90962127  90969408 113072247  58261860
  96914124  67732857  77714739   6189669  16556049  50444955  86305293
 114119964 114184737  71439624  19830726   7698573  63716085  88084233
  98437572  86765679 112008231 102960612 107270208  69095970 108771921
  68339691 113251653 109176507  80690850 114411429  91597482  88350345
  62358795  65998008  98616393  90490095  55269468 110745081  62794638
   3609108  15244776   4580973  50591394  10865385  69823296  78424272
  69990066  80007165  56325231  39866616  84664215  58805109  21749949
   3021966  74202138  27820854  79519941 108457488 114113106 113164263
  44199522  62724330  64593639  10706778  50200974  45170577  27106011
  81332874  91322901  73004931  18314037  70291575  72755055  83983401
  88972605  49267224  65756124  90420588  66976416  97728264  12982563
 105994638  98802900  52206939  80567523  53318430   2342115  76209363
  21951936  34158420 105380712  85453425  88183611  88096374  52348014
  60242445  71580672 101263491  20940120  48202425  96032745  85181346
  65276874 101423286  56638323  49233285  65914875  96549183  99938439
  75618207  94831866  45867465  94053213  53423370 105882327  22743558
  69633603  98880399  96946884 111866535 104364801  57630609 101217393
  56175597  98870022 101904210  76494231  83260566 101131281  60864309
  64809036  64269702  21684645 107987202  86903613 104098230  75755376
  95106312  71709237  87109101  97420221  73392147 104374458  44076375
  64758897  66111804   1743804  65795742  96869835  81622962   6758775
  91660203 110351556  93773835 103687677  53953623 112505013 106555239
  65390841  46064637  92921319  62950113 100403415  64339902  29363643
  26084007  19047780 112115556  80333055  12132864    206118   5053059
  71833644   5396931  88282737  98531514  38330631 113940450  88530201
 105878223  73285398  95102190 104779629  16386921  22674519  13845591
  64778733  92633913  78793623  65334870  11327310  58614777 109842435
    779760  40126266  69598224  65846214  71071542   2859183  50289336
  25450866  66022812  68344614  61998570  54763974  94472451  86672664
 100810809  46690020  68172498 108897210  80803998  82218582    761670
  99811710  54672858  69769476 101555280 107678277  42611859 110118132
  74720043  77686245  87120279  89643150   2301264 103564602 101233647
  89495955  73054188   6028875  97648119  93088233  20488338  77140440
  56409066  50168322  21629853 105642540 102265866  61854309  16666668
  67578831 114893640  86267655 104704110  63453591  86470605  55728909
  77536260  45004320  40379652  22881249  87711696  35133336  72843030
 110587005  55286262  97402077  42514713  61122384  66380058  66316248
 110051091  80625402  68670450  99584334  91223523  76346199  42280380
 113027103 110907864  79706430  85357377  83906802  60711624  63243981
  33465285 102067785  65011347 107235018  60554493 102491856 102680127
  49850208 103817322 111481452 111576771  37327131  52211700 109214118
  99370737  65029644  95580027  97888410  53291412  80795781   2944836
   1595322  80130699  11282175   6428826  77827617  67461381  55380096
  76976226  27120762 104752071  58868541 106595586  77687415  37750986
  90537534  84923541  93640977  88531686  44337429 106912116  79843788
  52344585 113936868 113575707  49492692  69256188  92295351  88245207
  11620728  20969919 113427171  51883884  76749948  67883436  71585730
  67010049 111488328  82967229  84630636 114914214  90536139  81229968
 113649093  98777394  40376178  25580034  19609686  53609220  99801477
  78702129  82184103  90000675 102451068  82704168  98724150  65349441
 108619785  41915691  45648837  46670517  86704641  75740436 103329594
  21542841   5412573  69933024  64672299  62632638  53171748  76273614
  13987701  80041266  39256587  73878543   6057711  84750579  81841095
  15569379  65211831  65290077  22937733  72389961  65270808  92400894
 113243436   6824169   8105490  62058582  98817516  85200516  89034768
  99578259  93998961  16921152  80034237  93559455  52340247  69172002
 102833064  23058477 101007450  95405175  72320544 102523032  94770171
  84178620   3135807 101132973  75189573  55669932  73726452  68244219
  55808829  73122228  95990742  92434806  65418912  33056145  89578980
  33073200 113896179  55477791  70076169  13579380  81929466  42294267
    920619  57623949  92960091  21834369  88544232  82325961  56220255
  22999122  97243407 109829538  34776900  14359797  72877500 102299175
 110949183   7315353  77912109  61204185  84774825  81509751  96133068
  31004532  51516873  32021208  80895078  76092336  51609195  73997415
  60401475 113993721 102420747  23170572 113272641  55028079  72075951
  39067056 104832666  21645018  54387162  94593078  21938337  69187482
 111136032  21653595  61094367  20434689  78156405  51748416  78590844
```

```
  47343420  80882010  92330604  65228292  63937845 108826011  85268259
  69870915  84140199 115179624 106584858   4945113  54955197  21739896
  21148866  51672447  81362061  85663323  49094244 108000405  79480854
 101754036  83724669  81529380  34511220 101447406  23441985  80096418
  98935092 109806192  98229618  66612906  32890041  80133525  21620106
 109926063 108520641 106598295  60092856  94270959 108470223  89231274
 107610858   5732190  84448305  81850653  55143027 111055230 114691221
  85440906  10397718  83292120  61983675  52109901 110116746 108612801
 106000974  88476876  66544470  62916849  52809273   1840977  62416152
 101634570  91231434  49884534  92525355  91551195 108462762  78516324
  29666772  88422057 108963711  18505530   3157407  51079491   6469308
  89952723  60914322 100616103  68391864  62795961  88981110  36627768
  75949227 105455421  36102690  94901697  85736007  60001884  11528595
 109931175  83082510  88996941 102998124 100983141  21011481 104920191
 103333095  75976587 104516208  73237077  91563588  55275210  85354254
  80147826  86280867  25770735  16528968  82766511  80913897  63822510
  97787358  66578481 109575621  27465723 103397130  80966655  70095411
  20256345  41290605  87170643  93890061  58963203  42517854  72267075
  82970667  65705580  88125012 111711870  30840552  48648636 107450397
  29121129  21642084  78660819  83628900  22325814  83669193 104844231
   2836278  97074369  22724991  65952252  28022877  45296928 126247874
  96392187  91541142 102961089  68660595  88603146  73590318 100178811
 103321521  69640065  80873244 104956029  61696620  60750342  14612832
  53482221  62917911  81850140  66569706  86580594  96867576  98183286
  89244846  73798299  98225649  77498496   7959330  42846687 108074322
  48911328  85662477  94843530  75455487 113431365 106407477  36184986
  51543576  45149013  88646436  64296630  94783464  96584418  67659543
  92317950 117405302 182408828  65099880  67372344  18843498  88298091
 184274798  65858589  54222390  94037994  91383840  75760884 166875089
  40593096  82133919 100694196 110778327  57186270  51354945 112982949
 114986772  47543940  26106084 104922396  51168771  80256915 111222522
  44945478  72144513  94573413  90853668 123812618 117516146   8420319
  97708986 113392368]

<class 'str'>admission_type_id[3 1 2 5]

<class 'str'>discharge_disposition_id[ 3 11  1  6  5  7 13  4  2 18]

<class 'str'>admission_source_id[ 1  5  7  4  6 17]

<class 'str'>time_in_hospital[10  6  2  3  4  1  5 11  8  7 14  9 12 13]

<class 'str'>num_lab_procedures[ 65  73  58  33   5  63  75  46   1  50  45  41  47
61  52  70  54  59
  71  69   4  66  76  53  23  44  72  32  79  84  62  67  60  49  21  42
  86  36  27  80  10  51  64  68  43  55  12  56  40  77  35  25  22  78
  57  30  82  48  38  24  81   8  29  39  31  34  74  28  87  18  37  16
  85   7  14  20  83   2  93  15  92  19  11 105  13  88  26  17  89]

<class 'str'>num_procedures[1 0 3 4 2 6 5]

<class 'str'>num_medications[28 16 12  7 11 20 15 10 21 23 17  9 18  8 26 25 31 13 1
4  5 22 44  4 27
 24  3  6 19 32 40 33 39 41 29 30 42 43 36 58 34 55  1  2 52 57 46]

<class 'str'>number_outpatient[ 1  0  4  2  3  5  6 14 10  9  8  7 11 13 15]

<class 'str'>number_emergency[ 1  0  3  2  6  4  5 11]

<class 'str'>number_inpatient[ 3  0  6  1  5  2  4  9 10  7]

<class 'str'>number_diagnoses[9 7 6 8 5 3 4]
```

In [112…

```
df.shape
```

Out[112…

```
(1043, 50)
```

In [113…

```
#lista para colunas a serem excluídas
colunas_excluidas = ['payer_code'
,'medical_specialty'
,'admission_type_id'
,'admission_source_id'
,'discharge_disposition_id'
,'patient_nbr']

df = df.drop(colunas_excluidas, axis = 1)
```

In [114…

```
#colunas diagnoticos para o tipo string
df['diag_1'] = df['diag_1'].astype(pd.StringDtype())
df['diag_2'] = df['diag_2'].astype(pd.StringDtype())
df['diag_3'] = df['diag_3'].astype(pd.StringDtype())
```

In [115…

```
#atribuição do diconarioa de valores para equalizar numericamente as variaveis
df['glipizide-metformin'] = df['glipizide-metformin'].map({'No': 0}) #['No']
df['change'] = df['change'].map({'No': 0, 'Ch': 1}) #['Ch' 'No']
df['tolbutamide'] = df['tolbutamide'].map({'No': 0}) #['No']
df['examide'] = df['examide'].map({'No': 0}) #['No']
df['pioglitazone'] = df['pioglitazone'].map({'No': 0, 'Steady': 1,'Up': 2, 'Down': 3
df['diabetesMed'] = df['diabetesMed'].map({'No': 0, 'Yes': 1}) #['Yes' 'No']
df['citoglipton'] = df['citoglipton'].map({'No': 0}) #['No']
df['insulin'] = df['insulin'].map({'Steady': 0, 'No': 1,'Down': 2, 'Up': 3}) #['Stea
df['troglitazone'] = df['troglitazone'].map({'No': 0}) #['No']
df['glimepiride-pioglitazone'] = df['glimepiride-pioglitazone'].map({'No': 0}) #['No
df['glyburide-metformin'] = df['glyburide-metformin'].map({'No': 0}) #['No']
df['metformin-pioglitazone'] = df['metformin-pioglitazone'].map({'No': 0}) #['No']
df['miglitol'] = df['miglitol'].map({'No': 0}) #['No']
df['glyburide'] = df['glyburide'].map({'No': 0, 'Steady': 1,'Up': 2, 'Down':3}) #['N
df['tolazamide'] = df['tolazamide'].map({'No': 0}) #['No']
df['max_glu_serum'] = df['max_glu_serum'].map({'None': 0, 'Norm': 1,'>200': 2, '>300
df['nateglinide'] = df['nateglinide'].map({'No': 0, 'Steady': 1}) #['No' 'Steady']
df['metformin-rosiglitazone'] = df['metformin-rosiglitazone'].map({'No': 0}) #['No']
df['glimepiride'] = df['glimepiride'].map({'No': 0, 'Steady': 1,'Down': 2, 'Up': 3})
df['metformin'] = df['metformin'].map({'No': 0, 'Steady': 1,'Up': 2, 'Down': 3}) #['
df['A1Cresult'] = df['A1Cresult'].map({'None': 0, 'Norm': 1,'>8': 2, '>7': 3}) #['No
df['chlorpropamide'] = df['chlorpropamide'].map({'No': 0}) #['No']
df['acetohexamide'] = df['acetohexamide'].map({'No': 0}) #['No']
df['rosiglitazone'] = df['rosiglitazone'].map({'No': 0, 'Steady': 1})
df['glipizide'] = df['glipizide'].map({'Steady': 0, 'No': 1,'Up': 2, 'Down': 3})
df['acarbose'] = df['acarbose'].map({'No': 0, 'Steady': 1})
df['repaglinide'] = df['repaglinide'].map({'No': 0, 'Steady': 1,'Up': 2})
df['readmitted'] = df['readmitted'].map({'NO': 0, '<30': 1,'>30': 1})
df["gender"] = df['gender'].map({'Male': 0, 'Female': 1})
df["age"] = df['age'].map({'[0-10)': 0, '[10-20)': 1, '[20-30)': 2, '[30-40)': 3, '[
df['race'] = df['race'].map({'Caucasian': 0, 'AfricanAmerican': 1,'Asian': 2, 'Hispa
df["weight"] = df['weight'].map({'[0-25)': 0, '[25-50)': 1, '[50-75)': 2, '[75-100)'
```

In [117…

```
df = df.dropna()
```

In [121…

```
#lista de atribuição das patologias
Circulatory = np.arange(390,460).tolist() + [785]
Circulatory_str = [str(elemento) for elemento in Circulatory]

Respiratory = np.arange(460,520).tolist() + [786]
Respiratory_str = [str(elemento) for elemento in Respiratory]
```

```
Digestive = np.arange(520,580).tolist() + [787]
Digestive_str = [str(elemento) for elemento in Digestive]

Injury = np.arange(800,1000).tolist() + [786]
Injury_str = [str(elemento) for elemento in Injury]

Musculoskeletal = np.arange(710,740).tolist() + [786]
Musculoskeletal_str = [str(elemento) for elemento in Musculoskeletal]

Genitourinary = np.arange(580,630).tolist() +  [788]
Genitourinary_str = [str(elemento) for elemento in Genitourinary]

Neoplasms = np.arange(140,240).tolist()
Neoplasms_str = [str(elemento) for elemento in Neoplasms]

Other = np.arange(240,250).tolist() + np.arange(251,280).tolist() + [780,781,784,790
Other_str = [str(elemento) for elemento in Other]
```

In [122…

```
#atribuição das patologias dentre as colunas de diagnostico
df.loc[df['diag_1'].str.startswith('250'), 'diag_1'] = 'Diabetes'
df.loc[df['diag_2'].str.startswith('250'), 'diag_2'] = 'Diabetes'
df.loc[df['diag_3'].str.startswith('250'), 'diag_3'] = 'Diabetes'

df.loc[df['diag_1'].isin(Circulatory_str), 'diag_1'] = 'Circulatory'
df.loc[df['diag_1'].isin(Respiratory_str), 'diag_1'] = 'Respiratory'
df.loc[df['diag_1'].isin(Digestive_str), 'diag_1'] = 'Digestive'
df.loc[df['diag_1'].isin(Injury_str), 'diag_1'] = 'Injury'
df.loc[df['diag_1'].isin(Musculoskeletal_str), 'diag_1'] = 'Musculoskeletal'
df.loc[df['diag_1'].isin(Genitourinary_str), 'diag_1'] = 'Genitourinary'
df.loc[df['diag_1'].isin(Neoplasms_str), 'diag_1'] = 'Neoplasms'

df.loc[df['diag_2'].isin(Circulatory_str), 'diag_2'] = 'Circulatory'
df.loc[df['diag_2'].isin(Respiratory_str), 'diag_2'] = 'Respiratory'
df.loc[df['diag_2'].isin(Digestive_str), 'diag_2'] = 'Digestive'
df.loc[df['diag_2'].isin(Injury_str), 'diag_2'] = 'Injury'
df.loc[df['diag_2'].isin(Musculoskeletal_str), 'diag_2'] = 'Musculoskeletal'
df.loc[df['diag_2'].isin(Genitourinary_str), 'diag_2'] = 'Genitourinary'
df.loc[df['diag_2'].isin(Neoplasms_str), 'diag_2'] = 'Neoplasms'

df.loc[df['diag_3'].isin(Circulatory_str), 'diag_3'] = 'Circulatory'
df.loc[df['diag_3'].isin(Respiratory_str), 'diag_3'] = 'Respiratory'
df.loc[df['diag_3'].isin(Digestive_str), 'diag_3'] = 'Digestive'
df.loc[df['diag_3'].isin(Injury_str), 'diag_3'] = 'Injury'
df.loc[df['diag_3'].isin(Musculoskeletal_str), 'diag_3'] = 'Musculoskeletal'
df.loc[df['diag_3'].isin(Genitourinary_str), 'diag_3'] = 'Genitourinary'
df.loc[df['diag_3'].isin(Neoplasms_str), 'diag_3'] = 'Neoplasms'

patology = ["Circulatory","Respiratory","Digestive","Injury","Musculoskeletal","Geni

df.loc[~df['diag_1'].isin(patology), 'diag_1'] = 'Other'
df.loc[~df['diag_2'].isin(patology), 'diag_2'] = 'Other'
df.loc[~df['diag_3'].isin(patology), 'diag_3'] = 'Other'
```

In [123…

```
#verificação da atribuiçãode valores
colunas_excluidas_2 = ['encounter_id']
df = df.drop(colunas_excluidas_2, axis = 1)

len(df["diag_1"].unique()) == len(df["diag_2"].unique()) == len(df["diag_3"].unique(

#laço for para atribuição
diags = ['diag_1','diag_2','diag_3']
```

```
for diag in diags:
    df[diag] = df[diag].map({'Circulatory': 0, 'Respiratory': 1,'Digestive': 2, 'Inj
                , 'Musculoskeletal': 4, 'Genitourinary': 5, 'Neoplasms': 6, 'Other':

df['age'] = df['age'].astype(int)
```

In [126…

```
#importação da biblioteca para plotagem e exploração
from collections import Counter
import matplotlib.pyplot as plt

#slice das variaveis
gender = Counter(df["gender"])
gender
```

Out[126…

```
Counter({1: 508, 0: 513})
```

In [127…

```
age = Counter(df["age"])
age
```

Out[127…

```
Counter({7: 305, 8: 171, 6: 255, 4: 68, 5: 177, 2: 18, 3: 25, 0: 1, 1: 1})
```

In [128…

```
weight = Counter(df["weight"])
weight
```

Out[128…

```
Counter({3: 431, 2: 270, 4: 220, 5: 46, 1: 26, 6: 16, 7: 6, 0: 6})
```

In [129…

```
race = Counter(df["race"])
race
```

Out[129…

```
Counter({0: 977, 1: 30, 4: 13, 2: 1})
```

In [130…

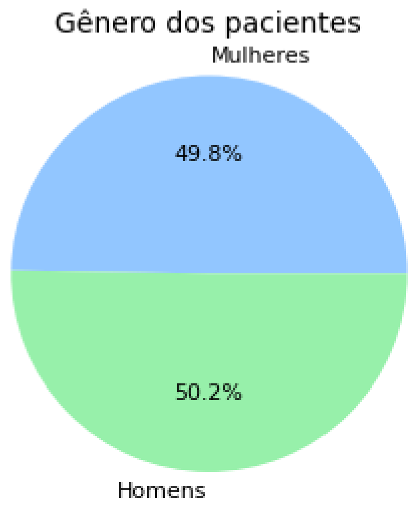
```
race = Counter(df["race"])
race
```

Out[130…

```
Counter({0: 977, 1: 30, 4: 13, 2: 1})
```

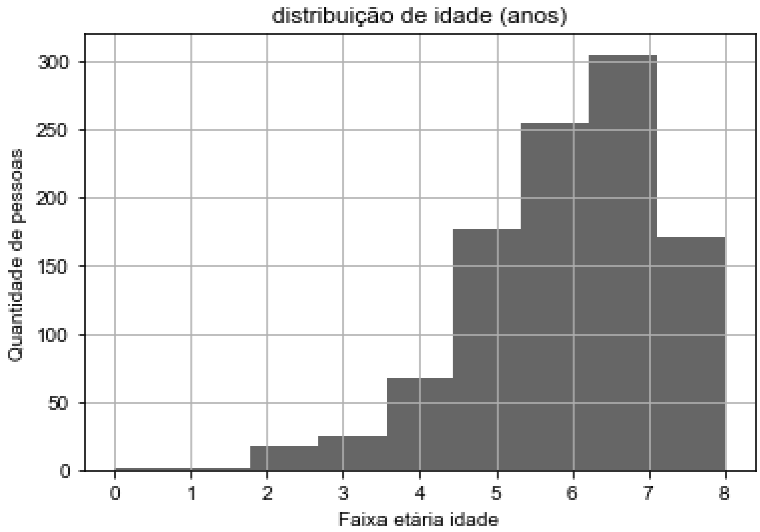
In [131…

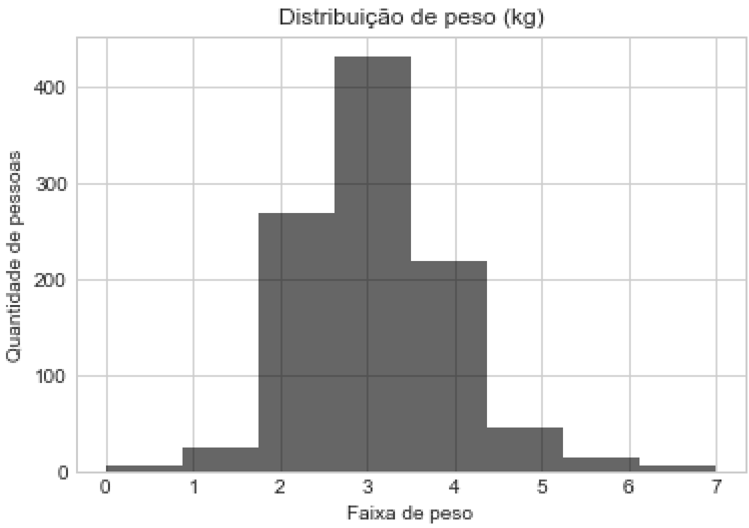
```
plt.style.use('seaborn-pastel')
plt.pie(gender.values(), labels = ["Mulheres","Homens"],
autopct = '%1.1f%%', textprops={'fontsize': 11})
plt.axis("image")
plt.title("Gênero dos pacientes", fontsize=14,pad =12)
plt.show()
```



In [132…

```
df.age.hist(bins=9, color = 'black', alpha = 0.6)
plt.style.use('seaborn-whitegrid')
plt.xlabel("Faixa etária idade")
plt.ylabel("Quantidade de pessoas")
plt.title("distribuição de idade (anos)")
plt.show()

df.weight.hist(bins=8, color = 'black', alpha = 0.6)
plt.style.use('seaborn-whitegrid')
plt.xlabel("Faixa de peso")
plt.ylabel("Quantidade de pessoas")
plt.title("Distribuição de peso (kg)")
plt.show()
```

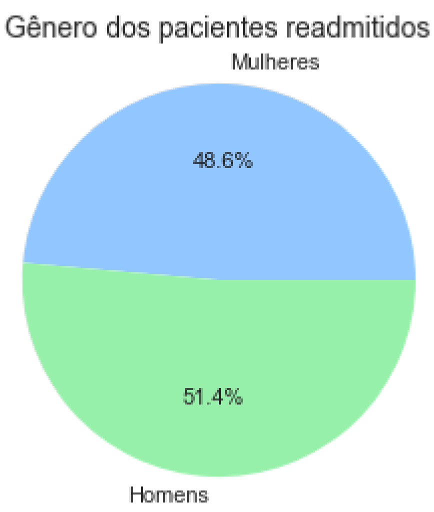


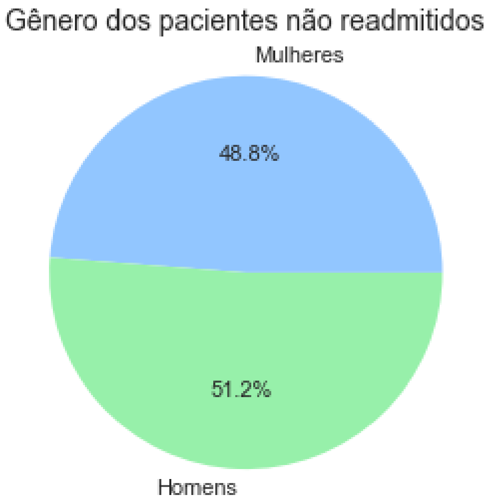


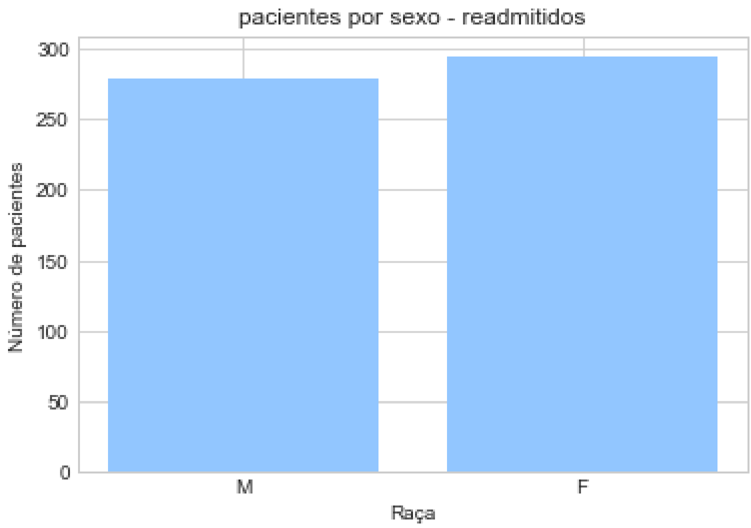
In [133…

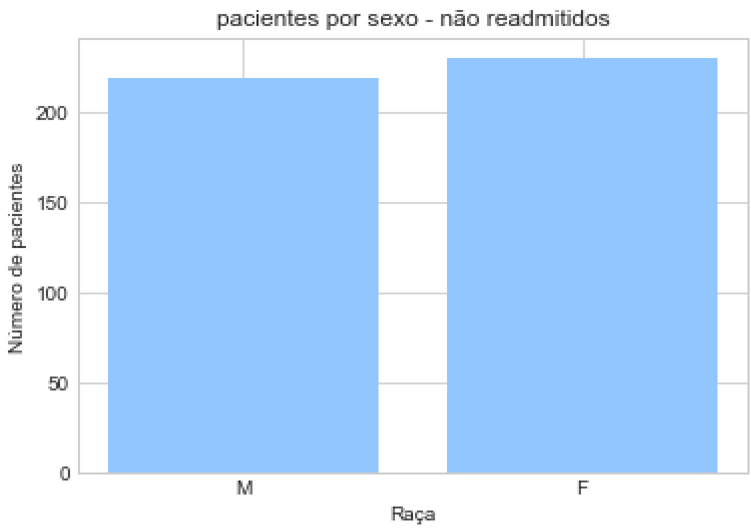
```
#slice dataframe pela variavel target
df_readmitted = df.loc[df['readmitted'] == 1 ]
df_not_readmitted = df.loc[df['readmitted'] == 0]

#slice do genero nos dataframes
gender_not_readmitted = Counter(df_not_readmitted["gender"])
gender_not_readmitted
gender_readmitted = Counter(df_readmitted["gender"])
gender_readmitted

plt.style.use('seaborn-pastel')
plt.pie(gender_readmitted.values(), labels = ["Mulheres","Homens"],
autopct = '%1.1f%%', textprops={'fontsize': 11})
plt.axis("image")
plt.title("Gênero dos pacientes readmitidos", fontsize=14,pad =12)
plt.show()

plt.style.use('seaborn-pastel')
plt.pie(gender_not_readmitted.values(), labels = ["Mulheres","Homens"],
autopct = '%1.1f%%', textprops={'fontsize': 11})
plt.axis("image")
plt.title("Gênero dos pacientes não readmitidos", fontsize=14,pad =12)
plt.show()

race_readmitted = Counter(df_readmitted["race"])
race_readmitted
race_not_readmitted = Counter(df_not_readmitted["race"])
race_not_readmitted
weight_readmitted = Counter(df_not_readmitted["weight"])
weight_readmitted
weight_not_readmitted = Counter(df_not_readmitted["weight"])
weight_not_readmitted

plt.style.use('seaborn-whitegrid')
plt.bar(['M','F'], gender_readmitted.values())
plt.ylabel('Número de pacientes')
plt.xlabel('Raça')
plt.title('pacientes por sexo - readmitidos')
plt.show()

plt.style.use('seaborn-whitegrid')
plt.bar(['M','F'], gender_not_readmitted.values())
plt.ylabel('Número de pacientes')
plt.xlabel('Raça')
plt.title('pacientes por sexo - não readmitidos')
```

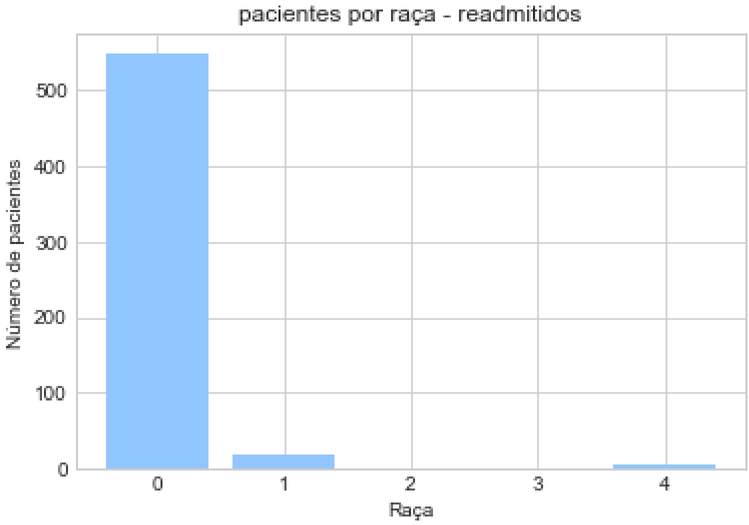
```
plt.show()

plt.style.use('seaborn-whitegrid')
plt.bar(list(race_readmitted.keys()), race_readmitted.values())
plt.ylabel('Número de pacientes')
plt.xlabel('Raça')
plt.title('pacientes por raça - readmitidos')
plt.show()

plt.style.use('seaborn-whitegrid')
plt.bar(list(race_not_readmitted.keys()), race_not_readmitted.values())
plt.ylabel('Número de pacientes')
plt.xlabel('Raça')
plt.title('pacientes por raça - não readmitidos')
plt.show()

plt.style.use('seaborn-whitegrid')
plt.bar(['M','F'], gender_readmitted.values())
plt.ylabel('Número de pacientes')
plt.xlabel('Raça')
plt.title('pacientes por sexo - readmitidos')
plt.show()

plt.style.use('seaborn-whitegrid')
plt.bar(['M','F'], gender_not_readmitted.values())
plt.ylabel('Número de pacientes')
plt.xlabel('Raça')
plt.title('pacientes por sexo - não readmitidos')
plt.show()

plt.style.use('seaborn-whitegrid')
plt.bar(list(weight_readmitted.keys()), weight_readmitted.values())
plt.ylabel('Número de pacientes')
plt.xlabel('Peso (kg)')
plt.title('Pacientes por peso (kg) - readmitidos')
plt.show()

plt.style.use('seaborn-whitegrid')
plt.bar(list(weight_not_readmitted.keys()), weight_not_readmitted.values())
plt.ylabel('Número de pacientes')
plt.xlabel('Peso (kg)')
plt.title('Pacientes por peso (kg) - não readmitidos')
plt.show()
```

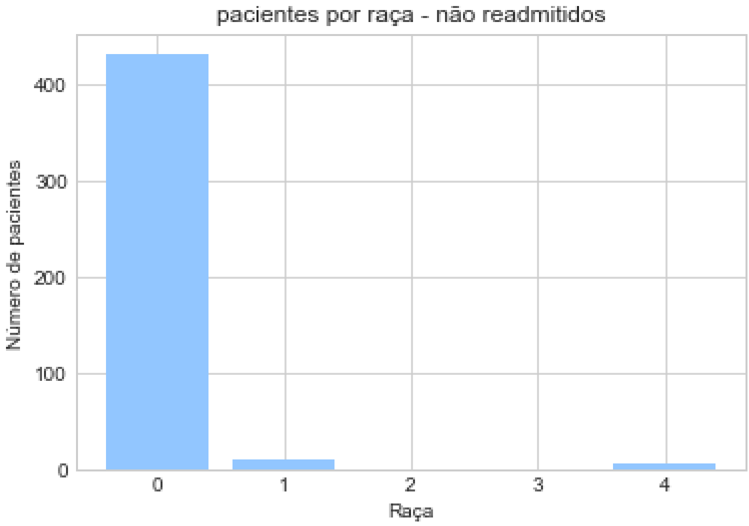


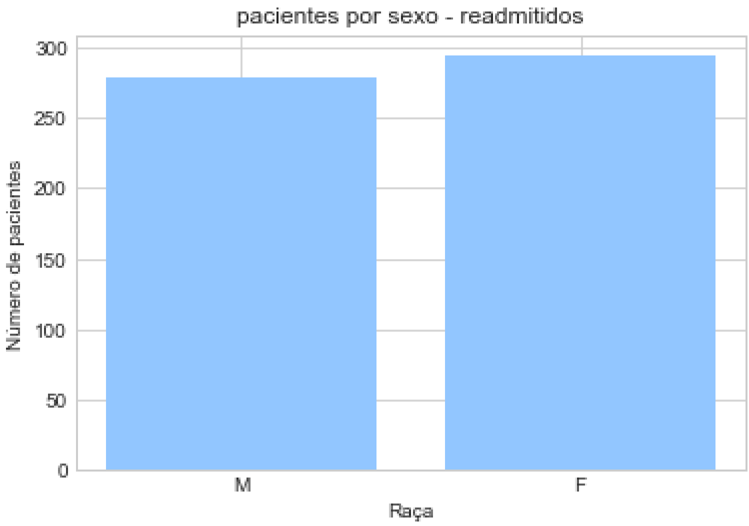

Gênero dos pacientes não readmitidos
Mulheres
48.8%
51.2%
Homens

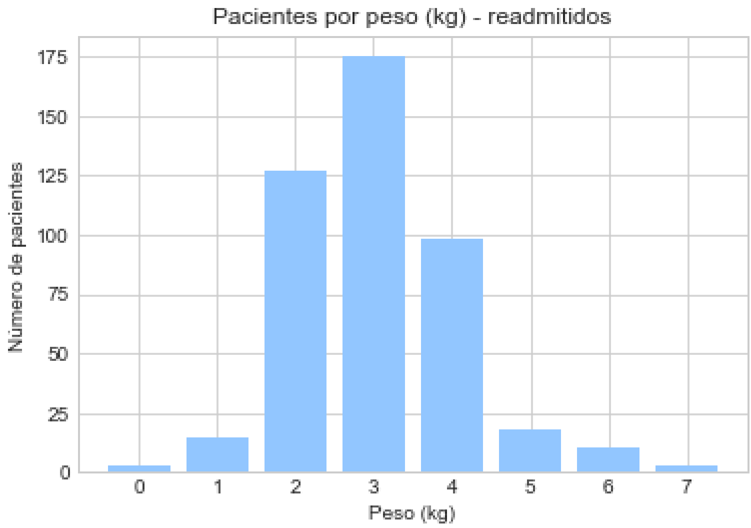

pacientes por sexo - readmitidos
Número de pacientes
300
250
200
150
100
50
0
M
F
Raça

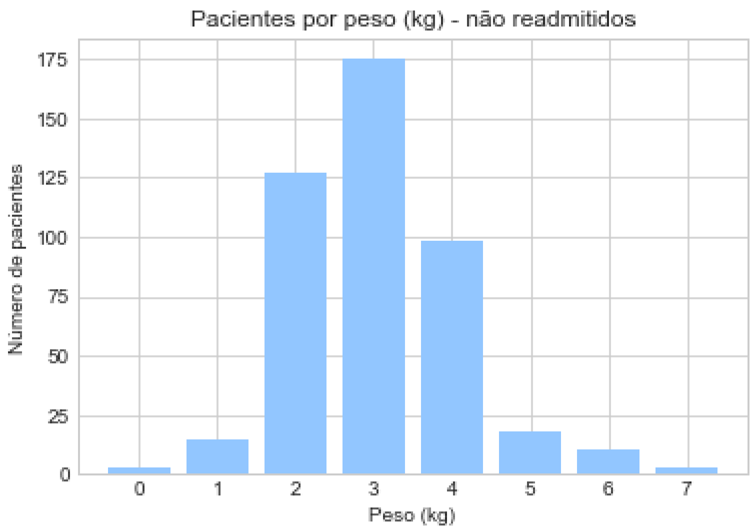

pacientes por sexo - não readmitidos
Número de pacientes
200
150
100
50
0
M
F
Raça


pacientes por raça - readmitidos
Número de pacientes
0
100
200
300
400
500
0
1
2
3
4
Raça


pacientes por raça - não readmitidos
Número de pacientes
0
100
200
300
400
0
1
2
3
4
Raça


pacientes por sexo - readmitidos
Número de pacientes
0
50
100
150
200
250
300
M
F
Raça





In [134…

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score

#separação dados de treinno e teste
X = df.drop('readmitted',axis=1).values
y = df['readmitted'].values
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=0)
```

In [135…

```
#aplicação algoritmo KNN
from sklearn.neighbors import KNeighborsClassifier
```

```
knn = KNeighborsClassifier()
knn = knn.fit(X_train, y_train)
print("Acurácia: ", knn.score(X_train, y_train))
tp_knn = knn.predict(X_test)
print("Acurácia de previsão: ", accuracy_score(y_test, tp_knn))
print(classification_report(y_test, tp_knn))
from sklearn.metrics import precision_score, accuracy_score, recall_score, f1_score
print(f"Accuracy: {round(accuracy_score(y_test, tp_knn), 2)}")
print(f"Precision: {round(precision_score(y_test, tp_knn), 2)}")
print(f"Recall: {round(recall_score(y_test, tp_knn), 2)}")
print(f"F1_score: {round(f1_score(y_test, tp_knn), 2)}")
```

```
Acurácia:  0.7058823529411765
Acurácia de previsão:  0.46579804560260585
              precision    recall  f1-score   support

           0       0.41      0.44      0.42       136
           1       0.52      0.49      0.50       171

    accuracy                           0.47       307
   macro avg       0.46      0.46      0.46       307
weighted avg       0.47      0.47      0.47       307

Accuracy: 0.47
Precision: 0.52
Recall: 0.49
F1_score: 0.5
```

In [136…

```
#aplicação algoritmo Naive Bayes
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, classification_report
nb = GaussianNB()
nb = nb.fit(X_train, y_train)
print("Acurácia: ", nb.score(X_train, y_train))
tp_nb = nb.predict(X_test)
print(classification_report(y_test, tp_nb))
from sklearn.metrics import precision_score, accuracy_score, recall_score, f1_score
print(f"Accuracy: {round(accuracy_score(y_test, tp_nb), 2)}")
print(f"Precision: {round(precision_score(y_test, tp_nb), 2)}")
print(f"Recall: {round(recall_score(y_test, tp_nb), 2)}")
print(f"F1_score: {round(f1_score(y_test, tp_nb), 2)}")
```

```
Acurácia:  0.6386554621848739
              precision    recall  f1-score   support

           0       0.48      0.63      0.55       136
           1       0.61      0.46      0.53       171

    accuracy                           0.54       307
   macro avg       0.55      0.55      0.54       307
weighted avg       0.56      0.54      0.54       307

Accuracy: 0.54
Precision: 0.61
Recall: 0.46
F1_score: 0.53
```

In [137…

```
#aplicação algoritmo gradiente descendente
from sklearn.linear_model import SGDClassifier
sgd = SGDClassifier()
sgd = sgd.fit(X_train, y_train)
print("Acurácia: ", sgd.score(X_train, y_train))
tp_sgd = sgd.predict(X_test)
print(classification_report(y_test, tp_sgd))
from sklearn.metrics import precision_score, accuracy_score, recall_score, f1_score
```

```
print(f"Accuracy: {round(accuracy_score(y_test, tp_sgd), 2)}")
print(f"Precision: {round(precision_score(y_test, tp_sgd), 2)}")
print(f"Recall: {round(recall_score(y_test, tp_sgd), 2)}")
print(f"F1_score: {round(f1_score(y_test, tp_sgd), 2)}")
```

```
Acurácia:  0.5700280112044818
              precision    recall  f1-score   support

           0       0.57      0.03      0.06       136
           1       0.56      0.98      0.71       171

    accuracy                           0.56       307
   macro avg       0.57      0.51      0.38       307
weighted avg       0.57      0.56      0.42       307

Accuracy: 0.56
Precision: 0.56
Recall: 0.98
F1_score: 0.71
```

In [ ]: