```
import pandas as pd
import numpy as np
from collections import Counter as c
import matplotlib.pyplot as plt
import seaborn as sns
import missingno as msno
from sklearn.metrics import accuracy_score,confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression
import pickle

data=pd.read_csv("chronickidneydisease.csv")
data
```

```
      id   age    bp     sg   al   su     rbc        pc         pcc  \
0      0  48.0  80.0  1.020  1.0  0.0     NaN    normal  notpresent
1      1   7.0  50.0  1.020  4.0  0.0     NaN    normal  notpresent
2      2  62.0  80.0  1.010  2.0  3.0  normal    normal  notpresent
3      3  48.0  70.0  1.005  4.0  0.0  normal  abnormal     present
4      4  51.0  80.0  1.010  2.0  0.0  normal    normal  notpresent
..   ...   ...   ...    ...  ...  ...     ...       ...         ...
395  395  55.0  80.0  1.020  0.0  0.0  normal    normal  notpresent
396  396  42.0  70.0  1.025  0.0  0.0  normal    normal  notpresent
397  397  12.0  80.0  1.020  0.0  0.0  normal    normal  notpresent
398  398  17.0  60.0  1.025  0.0  0.0  normal    normal  notpresent
399  399  58.0  80.0  1.025  0.0  0.0  normal    normal  notpresent

             ba  ...  pcv    wc   rc  htn   dm  cad appet   pe  ane  \
0    notpresent  ...   44  7800  5.2  yes  yes   no  good   no   no
1    notpresent  ...   38  6000  NaN   no   no   no  good   no   no
2    notpresent  ...   31  7500  NaN   no  yes   no  poor   no  yes
3    notpresent  ...   32  6700  3.9  yes   no   no  poor  yes  yes
4    notpresent  ...   35  7300  4.6   no   no   no  good   no   no
..          ...  ...  ...   ...  ...  ...  ...  ...   ...  ...  ...
395  notpresent  ...   47  6700  4.9   no   no   no  good   no   no
396  notpresent  ...   54  7800  6.2   no   no   no  good   no   no
397  notpresent  ...   49  6600  5.4   no   no   no  good   no   no
398  notpresent  ...   51  7200  5.9   no   no   no  good   no   no
399  notpresent  ...   53  6800  6.1   no   no   no  good   no   no

    classification
0              ckd
1              ckd
2              ckd
3              ckd
4              ckd
..             ...
395         notckd
396         notckd
```

```
397         notckd
398         notckd
399         notckd

[400 rows x 26 columns]
```

```
data.drop(["id"],axis=1,inplace=True)
```

```
data
```

```
      age    bp     sg   al   su     rbc        pc         pcc
ba  \
0    48.0  80.0  1.020  1.0  0.0     NaN    normal  notpresent
notpresent
1     7.0  50.0  1.020  4.0  0.0     NaN    normal  notpresent
notpresent
2    62.0  80.0  1.010  2.0  3.0  normal    normal  notpresent
notpresent
3    48.0  70.0  1.005  4.0  0.0  normal  abnormal     present
notpresent
4    51.0  80.0  1.010  2.0  0.0  normal    normal  notpresent
notpresent
..    ...   ...    ...  ...  ...     ...       ...         ...
...
395  55.0  80.0  1.020  0.0  0.0  normal    normal  notpresent
notpresent
396  42.0  70.0  1.025  0.0  0.0  normal    normal  notpresent
notpresent
397  12.0  80.0  1.020  0.0  0.0  normal    normal  notpresent
notpresent
398  17.0  60.0  1.025  0.0  0.0  normal    normal  notpresent
notpresent
399  58.0  80.0  1.025  0.0  0.0  normal    normal  notpresent
notpresent

       bgr  ...  pcv    wc   rc  htn   dm cad appet   pe  ane
classification
0    121.0  ...   44  7800  5.2  yes  yes  no  good   no   no
ckd
1      NaN  ...   38  6000  NaN   no   no  no  good   no   no
ckd
2    423.0  ...   31  7500  NaN   no  yes  no  poor   no  yes
ckd
3    117.0  ...   32  6700  3.9  yes   no  no  poor  yes  yes
ckd
4    106.0  ...   35  7300  4.6   no   no  no  good   no   no
ckd
..     ...  ...  ...   ...  ...  ...  ...  ..   ...  ...  ...
...
395  140.0  ...   47  6700  4.9   no   no  no  good   no   no
notckd
```

```
396   75.0  ...   54  7800  6.2   no   no  no  good   no   no
notckd
397  100.0  ...   49  6600  5.4   no   no  no  good   no   no
notckd
398  114.0  ...   51  7200  5.9   no   no  no  good   no   no
notckd
399  131.0  ...   53  6800  6.1   no   no  no  good   no   no
notckd

[400 rows x 25 columns]
```

```
data['rc'].unique()
```

```
array(['5.2', nan, '3.9', '4.6', '4.4', '5', '4.0', '3.7', '3.8',
'3.4',
       '2.6', '2.8', '4.3', '3.2', '3.6', '4', '4.1', '4.9', '2.5',
'4.2',
       '4.5', '3.1', '4.7', '3.5', '6.0', '5.0', '2.1', '5.6', '2.3',
       '2.9', '2.7', '8.0', '3.3', '3.0', '3', '2.4', '4.8', '\t?',
'5.4',
       '6.1', '6.2', '6.3', '5.1', '5.8', '5.5', '5.3', '6.4', '5.7',
       '5.9', '6.5'], dtype=object)
```

```
catcols=set(data.dtypes[data.dtypes=='O'].index.values)
print(catcols)
```

```
{'pcv', 'wc', 'appet', 'classification', 'pe', 'pc', 'ane', 'pcc',
'cad', 'ba', 'dm', 'htn', 'rbc', 'rc'}
```

```
catcols.remove('rc')
catcols.remove('pcv')
catcols.remove('wc')
print(catcols)
```

```
{'appet', 'classification', 'pe', 'pc', 'ane', 'pcc', 'cad', 'ba',
'dm', 'htn', 'rbc'}
```

```
contcols=set(data.dtypes[data.dtypes!='O'].index.values)
print(contcols)
```

```
{'sg', 'al', 'sod', 'pot', 'bgr', 'sc', 'age', 'su', 'bu', 'hemo',
'bp'}
```

```
contcols.remove('sg')
contcols.remove('al')
contcols.remove('su')
print(contcols)
```

```
{'sod', 'pot', 'bgr', 'sc', 'age', 'bu', 'hemo', 'bp'}
```

```
contcols.add('rc')
contcols.add('pcv')
```

```
contcols.add('wc')
print(contcols)
```

```
{'pcv', 'wc', 'sod', 'pot', 'bgr', 'sc', 'age', 'bu', 'hemo', 'bp',
'rc'}
```

```
catcols.add('sg')
catcols.add('al')
catcols.add('su')
print(catcols)
```

```
{'appet', 'classification', 'sg', 'pe', 'al', 'pc', 'ane', 'pcc',
'cad', 'ba', 'su', 'dm', 'htn', 'rbc'}
```

```
data['cad']=data.cad.replace('\tno','no')
c(data['cad'])
```

```
Counter({'no': 364, 'yes': 34, nan: 2})
```

```
data['dm']=data.dm.replace(to_replace={'\tno':'no','\tyes':'yes','
yes':'yes'})
c(data['dm'])
```

```
Counter({'yes': 137, 'no': 261, nan: 2})
```

```
data.isnull().any()
```

```
age                True
bp                False
sg                 True
al                 True
su                 True
rbc                True
pc                 True
pcc                True
ba                 True
bgr               False
bu                False
sc                False
sod               False
pot               False
hemo              False
pcv               False
wc                False
rc                False
htn                True
dm                 True
cad                True
appet              True
pe                 True
ane                True
```

```
classification    False
dtype: bool
```

```
data.isnull().sum()
```

```
age               0
bp                0
sg                0
al                0
su                0
rbc               0
pc                0
pcc               0
ba                0
bgr               0
bu                0
sc                0
sod               0
pot               0
hemo              0
pcv               0
wc                0
rc                0
htn               0
dm                0
cad               0
appet             0
pe                0
ane               0
classification    0
dtype: int64
```

```
data.pcv=pd.to_numeric(data.pcv, errors='coerce')
data.wc=pd.to_numeric(data.wc, errors='coerce')
data.rc=pd.to_numeric(data.rc, errors='coerce')
```

```
data['ane'].unique()
```

```
array(['no', 'yes', nan], dtype=object)
```

```
data['bgr'].fillna(data['bgr'].mean(),inplace=True)
data['bp'].fillna(data['bp'].mean(),inplace=True)
data['bu'].fillna(data['bu'].mean(),inplace=True)
data['hemo'].fillna(data['hemo'].mean(),inplace=True)
data['pcv'].fillna(data['pcv'].mean(),inplace=True)
data['pot'].fillna(data['pot'].mean(),inplace=True)
data['rc'].fillna(data['rc'].mean(),inplace=True)
data['sc'].fillna(data['sc'].mean(),inplace=True)
data['sod'].fillna(data['sod'].mean(),inplace=True)
data['wc'].fillna(data['wc'].mean(),inplace=True)
```

```
data['age'].fillna(data['age'].mode()[0],inplace=True)
data['htn'].fillna(data['htn'].mode()[0],inplace=True)
data['pcc'].fillna(data['pcc'].mode()[0],inplace=True)
data['appet'].fillna(data['appet'].mode()[0],inplace=True)
data['al'].fillna(data['al'].mode()[0],inplace=True)
data['pc'].fillna(data['pc'].mode()[0],inplace=True)
data['rbc'].fillna(data['rbc'].mode()[0],inplace=True)
data['cad'].fillna(data['cad'].mode()[0],inplace=True)
data['ba'].fillna(data['ba'].mode()[0],inplace=True)
data['ane'].fillna(data['ane'].mode()[0],inplace=True)
data['su'].fillna(data['su'].mode()[0],inplace=True)
data['dm'].fillna(data['dm'].mode()[0],inplace=True)
data['pe'].fillna(data['pe'].mode()[0],inplace=True)
data['sg'].fillna(data['sg'].mode()[0],inplace=True)

data
```

```
      age    bp     sg   al   su     rbc        pc         pcc
ba  \
0    48.0  80.0  1.020  1.0  0.0  normal    normal  notpresent
notpresent
1     7.0  50.0  1.020  4.0  0.0  normal    normal  notpresent
notpresent
2    62.0  80.0  1.010  2.0  3.0  normal    normal  notpresent
notpresent
3    48.0  70.0  1.005  4.0  0.0  normal  abnormal     present
notpresent
4    51.0  80.0  1.010  2.0  0.0  normal    normal  notpresent
notpresent
..    ...   ...    ...  ...  ...     ...       ...         ...
...
395  55.0  80.0  1.020  0.0  0.0  normal    normal  notpresent
notpresent
396  42.0  70.0  1.025  0.0  0.0  normal    normal  notpresent
notpresent
397  12.0  80.0  1.020  0.0  0.0  normal    normal  notpresent
notpresent
398  17.0  60.0  1.025  0.0  0.0  normal    normal  notpresent
notpresent
399  58.0  80.0  1.025  0.0  0.0  normal    normal  notpresent
notpresent

            bgr  ...   pcv      wc        rc  htn   dm  cad  appet
pe  ane  \
0    121.000000  ...  44.0  7800.0  5.200000  yes  yes   no   good
no   no
1    148.036517  ...  38.0  6000.0  4.707435   no   no   no   good
no   no
2    423.000000  ...  31.0  7500.0  4.707435   no  yes   no   poor
no  yes
```

```
3    117.000000  ...  32.0  6700.0  3.900000  yes   no   no   poor
yes  yes
4    106.000000  ...  35.0  7300.0  4.600000   no   no   no   good
no   no
..          ...  ...   ...     ...       ...  ...  ...  ...    ...  ..
.  ...
395  140.000000  ...  47.0  6700.0  4.900000   no   no   no   good
no   no
396   75.000000  ...  54.0  7800.0  6.200000   no   no   no   good
no   no
397  100.000000  ...  49.0  6600.0  5.400000   no   no   no   good
no   no
398  114.000000  ...  51.0  7200.0  5.900000   no   no   no   good
no   no
399  131.000000  ...  53.0  6800.0  6.100000   no   no   no   good
no   no

    classification
0              ckd
1              ckd
2              ckd
3              ckd
4              ckd
..             ...
395         notckd
396         notckd
397         notckd
398         notckd
399         notckd

[400 rows x 25 columns]
```

```
#'specific_gravity','albumin', 'sugar'(as these columns are  numerical
it is removed)
catcols=['anemia','pedal_edema','appetite','bacteria','class','coronar
y_artery_disease','diabetesmellitus',
 'hypertension','pus_cell','pus_cell_clumps','red_blood_cells']
for i in catcols: #looping through all the categorical columns
    print("LABEL ENCODING OF:",i)
    LEi = LabelEncoder() # creating an object of LabelEncoder
    print(c(data[i])) #getting the classes values before
transformation
    data[i] = LEi.fit_transform(data[i])# trannsforming our text
classes to numerical values
    print(c(data[i])) #getting the classes values after transformation
    print("*"*100)
```

```
LABEL ENCODING OF: anemia
Counter({0: 340, 1: 60})
Counter({0: 340, 1: 60})
```

```
**********************************************************************
*****************************
LABEL ENCODING OF: pedal_edema
Counter({0: 324, 1: 76})
Counter({0: 324, 1: 76})
**********************************************************************
*****************************
LABEL ENCODING OF: appetite
Counter({0: 318, 1: 82})
Counter({0: 318, 1: 82})
**********************************************************************
*****************************
LABEL ENCODING OF: bacteria
Counter({0: 378, 1: 22})
Counter({0: 378, 1: 22})
**********************************************************************
*****************************
LABEL ENCODING OF: class
Counter({0: 248, 2: 150, 1: 2})
Counter({0: 248, 2: 150, 1: 2})
**********************************************************************
*****************************
LABEL ENCODING OF: coronary_artery_disease
Counter({0: 366, 1: 34})
Counter({0: 366, 1: 34})
**********************************************************************
*****************************
LABEL ENCODING OF: diabetesmellitus
Counter({0: 263, 1: 137})
Counter({0: 263, 1: 137})
**********************************************************************
*****************************
LABEL ENCODING OF: hypertension
Counter({0: 253, 1: 147})
Counter({0: 253, 1: 147})
**********************************************************************
*****************************
LABEL ENCODING OF: pus_cell
Counter({1: 324, 0: 76})
Counter({1: 324, 0: 76})
**********************************************************************
*****************************
LABEL ENCODING OF: pus_cell_clumps
Counter({0: 358, 1: 42})
Counter({0: 358, 1: 42})
**********************************************************************
*****************************
LABEL ENCODING OF: red_blood_cells
Counter({1: 353, 0: 47})
Counter({1: 353, 0: 47})
```

```
**********************************************************************
*****************************
```

```
selcols=['red_blood_cells','pus_cell', 'blood glucose
random','blood_urea',
            'pedal_edema',
'anemia','diabetesmellitus','coronary_artery_disease']
x=pd.DataFrame(data,columns=selcols)
y=pd.DataFrame(data,columns=['class'])
print(x.shape)
print(y.shape)
```

```
(400, 8)
(400, 1)
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,rando
m_state=2)#train test split
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(320, 8)
(320, 1)
(80, 8)
(80, 1)
```

```
from sklearn.linear_model import LogisticRegression
lgr = LogisticRegression()
lgr.fit(x_train.values,y_train.values)
```

```
C:\Users\Dell\anaconda3\lib\site-packages\sklearn\utils\
validation.py:993: DataConversionWarning: A column-vector y was passed
when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\Dell\anaconda3\lib\site-packages\sklearn\linear_model\
_logistic.py:814: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(

LogisticRegression()
```

```
y_pred = lgr.predict([[129,99,1,0,0,1,0,1]])

print(y_pred)
c(y_pred)
```

```
[2]
```

```
Counter({2: 1})
```

```
accuracy_score(y_test,y_pred)
```

```
---------------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
last)
~\AppData\Local\Temp\ipykernel_6964\772673080.py in <module>
----> 1 accuracy_score(y_test,y_pred)

~\anaconda3\lib\site-packages\sklearn\metrics\_classification.py in
accuracy_score(y_true, y_pred, normalize, sample_weight)
    209
    210     # Compute accuracy for each possible representation
--> 211     y_type, y_true, y_pred = _check_targets(y_true, y_pred)
    212     check_consistent_length(y_true, y_pred, sample_weight)
    213     if y_type.startswith("multilabel"):

~\anaconda3\lib\site-packages\sklearn\metrics\_classification.py in
_check_targets(y_true, y_pred)
     82     y_pred : array or indicator matrix
     83     """
---> 84     check_consistent_length(y_true, y_pred)
     85     type_true = type_of_target(y_true)
     86     type_pred = type_of_target(y_pred)

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in
check_consistent_length(*arrays)
    330     uniques = np.unique(lengths)
    331     if len(uniques) > 1:
--> 332         raise ValueError(
    333             "Found input variables with inconsistent numbers
of samples: %r"
    334             % [int(l) for l in lengths]

ValueError: Found input variables with inconsistent numbers of
samples: [80, 1]
```

```
data.columns #return all the column names
```

```
Index(['age', 'bp', 'sg', 'al', 'su', 'rbc', 'pc', 'pcc', 'ba', 'bgr',
'bu',
       'sc', 'sod', 'pot', 'hemo', 'pcv', 'wc', 'rc', 'htn', 'dm',
```

```
'cad',
       'appet', 'pe', 'ane', 'classification'],
      dtype='object')
```

```
data.columns=['age','blood_pressure','specific_gravity','albumin',

'sugar','red_blood_cells','pus_cell','pus_cell_clumps','bacteria',
               'blood glucose
random','blood_urea','serum_creatinine','sodium','potassium',

'hemoglobin','packed_cell_volume','white_blood_cell_count','red_blood_
cell_count',

'hypertension','diabetesmellitus','coronary_artery_disease','appetite'
,
               'pedal_edema','anemia','class'] # manually giving the
name  of the columns
data.columns
```

```
Index(['age', 'blood_pressure', 'specific_gravity', 'albumin',
'sugar',
       'red_blood_cells', 'pus_cell', 'pus_cell_clumps', 'bacteria',
       'blood glucose random', 'blood_urea', 'serum_creatinine',
'sodium',
       'potassium', 'hemoglobin', 'packed_cell_volume',
       'white_blood_cell_count', 'red_blood_cell_count',
'hypertension',
       'diabetesmellitus', 'coronary_artery_disease', 'appetite',
       'pedal_edema', 'anemia', 'class'],
      dtype='object')
```