In [1]: import pandas as pd
import numpy as np
from collections import Counter as c
import matplotlib.pyplot as plt
%matplotlib inline
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 LogisticRegression
import pickle

In [2]: data = pd.read_csv("C:\Users\ELCOT\Desktop\chronickidneydisease.csv")

In [3]: data

Out[3]:

	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	7800	5.2	yes	yes	no	good	no	no	ckd
1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	...	38	6000	NaN	no	no	no	good	no	no	ckd
2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	...	31	7500	NaN	no	yes	no	poor	no	yes	ckd
3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	...	32	6700	3.9	yes	no	no	poor	yes	yes	ckd
4	51.0	80.0	1.010	2.0	0.0	NaN	normal	notpresent	notpresent	...	35	7300	4.6	no	no	no	good	no	no	ckd
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
395	55.0	80.0	1.020	0.0	0.0	normal	normal	notpresent	notpresent	...	47	6700	4.9	no	no	no	good	no	no	notckd
396	42.0	70.0	1.025	0.0	0.0	normal	normal	notpresent	notpresent	...	54	6800	6.2	no	no	no	good	no	no	notckd
397	12.0	80.0	1.020	0.0	0.0	normal	normal	notpresent	notpresent	...	49	6600	5.4	no	no	no	good	no	no	notckd
398	17.0	60.0	1.025	0.0	0.0	normal	normal	notpresent	notpresent	...	51	7200	5.9	no	no	no	good	no	no	notckd
399	58.0	80.0	1.025	0.0	0.0	normal	normal	notpresent	notpresent	...	53	6800	6.1	no	no	no	good	no	no	notckd

400 rows × 26 columns

In [4]: #understanding data

In [5]: data.drop(["id"],axis=1,inplace=True)
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', 'haemoglobin', 'packed_cell_volume', 'white_blood_cell_count', 'red_blood_cell_count', 'hypertension', 'diabetesmellitus', 'coronary_artery_disease', 'appetite', 'pedal_edema', 'anemia', 'class']

In [6]: data

In [7]:

	age	blood pressure	specific gravity	albumin	sugar	red blood cells	pus cell	pus cell clumps	bacteria	blood glucose random	...	packed_cell_volume
0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	121.0	...	...
1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	148.0	...	...
2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	423.0	...	...
3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	117.0	...	...
4	51.0	80.0	1.010	2.0	0.0	NaN	normal	notpresent	notpresent	106.0	...	...
...	...	...	...	...	...	...	...	...	...	...	...	...
395	55.0	80.0	1.020	0.0	0.0	normal	normal	notpresent	notpresent	140.0	...	...
396	42.0	70.0	1.025	0.0	0.0	normal	normal	notpresent	notpresent	75.0	...	...
397	12.0	80.0	1.020	0.0	0.0	normal	normal	notpresent	notpresent	100.0	...	...
398	17.0	60.0	1.025	0.0	0.0	normal	normal	notpresent	notpresent	114.0	...	...
399	58.0	80.0	1.025	0.0	0.0	normal	normal	notpresent	notpresent	131.0	...	...

400 rows × 25 columns

In [8]: categoriesnet=data[data.dtypes=='O'].index.values]
print(categoriesnet)

In [9]: ['hypertension', 'packed_cell_volume', 'diabetesmellitus', 'white_blood_cell_count', 'coronary_artery_disease', 'bacteria', 'pus_cell', 'red_blood_cells', 'pus_cell_clumps', 'red_blood_cell_count', 'anemia', 'appetite', 'class', 'pedal_edema']

In [10]: continuosnet=data[data.dtypes=='O'].index.values]
print(continuosnet)

In [11]: ['hemoglobin', 'blood_pressure', 'sodium', 'specific_gravity', 'age', 'potassium', 'albumin', 'serum_creatinine', 'sugar', 'blood glucose random', 'blood urea']

In [12]: categols.remove('red_blood_cell_count')
categols.remove('packed_cell_volume')
categols.remove('white_blood_cell_count')
categols.remove('specific_gravity')
categols.add('albumin')
categols.add('sugar')
print(categols)

In [13]: ['hypertension', 'diabetesmellitus', 'coronary_artery_disease', 'bacteria', 'pus_cell', 'sugar', 'specific_gravity', 'red_blood_cells', 'pus_cell_clumps', 'anemia', 'appetite', 'albumin', 'class', 'pedal_edema']

In [14]: contcols.remove('specific_gravity')
contcols.remove('albumin')
contcols.remove('sugar')
contcols.remove('packed_cell_volume')
contcols.add('packed_cell_volume')
contcols.add('white_blood_cell_count')
print(contcols)

In [15]: ['hemoglobin', 'blood_pressure', 'packed_cell_volume', 'white_blood_cell_count', 'sodium', 'age', 'red_blood_cell_count', 'potassium', 'serum_creatinine', 'blood glucose random', 'blood_urea']

In [16]: #Get unique values from categorical columns

In [17]: for col in catcols:
print("{}\n{}\n".format(col,data[col].unique())) values \n"]

hypertension has ['yes' 'no' nan] values

diabetesmellitus has ['yes' 'no' ' yes' '\tno' '\tyes' nan] values

coronary_artery_disease has ['no' 'yes' '\tno' nan] values

bacteria has ['notpresent' 'present' nan] values

pus_cell has ['normal' 'abnormal' nan] values

sugar has [0. 3. 4. 1. nan 2. 5.] values

specific_gravity has [1.02 1.01 1.005 1.015 nan 1.025] values

red_blood_cells has [nan 'normal' 'abnormal'] values

pus_cell_clumps has ['notpresent' 'present' nan] values

anemia has ['no' 'yes' nan] values

appetite has ['good' 'poor' nan] values

albumin has [1. 4. 2. 3. 0. nan 5.] values

class has ['ckd' 'ckd\t' 'notckd'] values

pedal_edema has ['no' 'yes' nan] values

In [18]: data['class']=data['class'].replace('ckd\t','ckd')
data['coronary_artery_disease']=data.coronary_artery_disease.replace('\tno','no')
data['diabetesmellitus']=data.diabetesmellitus.replace(to_replace={'\tno':'no','\tyes':'yes',' yes':'yes'})

In [19]: data.packed_cell_volume = pd.to_numeric(data.packed_cell_volume, errors='coerce')
data.white_blood_cell_count = pd.to_numeric(data.white_blood_cell_count, errors='coerce')
data.red_blood_cell_count = pd.to_numeric(data.red_blood_cell_count, errors='coerce')

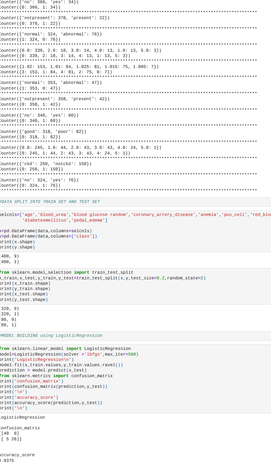
In [20]: # To handle showers in the data

In [21]: plt.figure(figsize = (30,15))
plotnumber =1
for column in contcols:
if plotnumber <= 14 :
ax = plt.subplot(3, 5, plotnumber)
sns.distplot(data[column])
plt.xlabel(column)

plotnumber += 1

plt.tight_layout()
plt.show()

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



In [22]: #some data's still possess skewness because of the null values in it

In [23]: data.isna().sum()

Out[23]:

age	9
blood_pressure	12
specific_gravity	47
albumin	46
sugar	49
red_blood_cells	152
pus_cell	65
pus_cell_clumps	4
bacteria	4
blood glucose random	44
blood_urea	19
serum_creatinine	17
sodium	87
potassium	88
hemoglobin	52
packed_cell_volume	70
white_blood_cell_count	106
red_blood_cell_count	131
hypertension	2
diabetesmellitus	2
coronary_artery_disease	2
appetite	1
pedal_edema	1
anemia	1
class	0
dtype: int64	

In [24]: data['blood_glucose random'].fillna(data['blood glucose random'].mean(),inplace=True)
data['blood_urea'].fillna(data['blood_urea'].mean(),inplace=True)
data['blood_pressure'].fillna(data['blood_pressure'].mean(),inplace=True)
data['hemoglobin'].fillna(data['hemoglobin'].mean(),inplace=True)
data['packed_cell_volume'].fillna(data['packed_cell_volume'].mean(),inplace=True)
data['potassium'].fillna(data['potassium'].mean(),inplace=True)
data['serum_creatinine'].fillna(data['serum_creatinine'].mean(),inplace=True)
data['wbcc_prediction'].fillna(data['wbcc_prediction'].mean(),inplace=True)
data['sodium'].fillna(data['sodium'].mean(),inplace=True)
data['white_blood_cell_count'].fillna(data['white_blood_cell_count'].mean(),inplace=True)

In [25]: data['age'].fillna(data['age'].mode()[0],inplace=True)
data['hypertension'].fillna(data['hypertension'].mode()[0],inplace=True)
data['pus_cell_clumps'].fillna(data['pus_cell_clumps'].mode()[0],inplace=True)
data['appetite'].fillna(data['appetite'].mode()[0],inplace=True)
data['albumin'].fillna(data['albumin'].mode()[0],inplace=True)
data['pus_cell'].fillna(data['pus_cell'].mode()[0],inplace=True)
data['red_blood_cells'].fillna(data['red_blood_cells'].mode()[0],inplace=True)
data['coronary_artery_disease'].fillna(data['coronary_artery_disease'].mode()[0],inplace=True)
data['bacteria'].fillna(data['bacteria'].mode()[0],inplace=True)
data['anemia'].fillna(data['anemia'].mode()[0],inplace=True)
data['sugar'].fillna(data['sugar'].mode()[0],inplace=True)
data['diabetesmellitus'].fillna(data['diabetesmellitus'].mode()[0],inplace=True)
data['pedal_edema'].fillna(data['pedal_edema'].mode()[0],inplace=True)
data['specific_gravity'].fillna(data['specific_gravity'].mode()[0],inplace=True)

In [26]: #label encoder

In [27]: from sklearn.preprocessing import LabelEncoder
for i in catcols:
print("{}\nEncoding OF:\n
{}\n".format(i,
le=LabelEncoder()
print(le.fit_transform(data[i])
print(le.classes_)
print("""100)

Counter({'no': 253, 'yes': 147})
Counter({0: 253, 1: 147})
****
Counter({'no': 355, 'yes': 12})
Counter({0: 355, 1: 12})
****
Counter({'no': 366, 'yes': 34})
Counter({0: 366, 1: 34})
****
Counter({'notpresent': 378, 'present': 22})
Counter({0: 378, 1: 22})
****
Counter({'normal': 324, 'abnormal': 76})
Counter({1: 324, 0: 76})
****
Counter({'ckd': 248, 'ckd\t': 2, 'notckd': 150})
Counter({0: 248, 2: 2, 1: 150})
****
Counter({'no': 355, 'yes': 45})
Counter({0: 355, 1: 45})
****
Counter({'notpresent': 358, 'present': 42})
Counter({0: 358, 1: 42})
****
Counter({'no': 348, 'yes': 69})
Counter({0: 348, 1: 69})
****
Counter({'good': 318, 'poor': 82})
Counter({0: 318, 1: 82})
****
Counter({'no': 141, '\tno': 4, '\tyes': 3, ' yes': 2, 'yes': 250})
Counter({0: 141, 4: 4, 3: 3, 2: 2, 1: 250})
****
Counter({'no': 296, 'notckd': 150})
Counter({0: 296, 1: 150})
****
Counter({'no': 324, 'yes': 76})
Counter({0: 324, 1: 76})
****

In [28]: #DATA SPLIT INTO TRAIN SET AND TEST SET

In [29]: seiosloo['age', 'blood_urea', 'blood glucose random', 'coronary_artery_disease', 'anemia', 'pus_cell', 'red_blood_cells', 'diabetesmellitus', 'pedal_edema']
x=pd.DataFrame(data,columns=seiosloo)
y=pd.DataFrame(data,columns=['class'])
print(x.shape)
print(y.shape)

(400, 9)
(400, 1)

In [30]: from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=2)
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)

(320, 9)
(320, 1)
(80, 9)
(80, 1)

In [31]: #MODEL BUILDING using LogisticRegression

In [32]: from sklearn.linear_model import LogisticRegression
model=LogisticRegression(solver='lbfgs',max_iter=500)
print("LogisticRegressionscore:")
model.fit(x_train.values,y_train.values.ravel())
y_pred=model.predict(x_test)
from sklearn.metrics import confusion_matrix
print("confusion_matrix:")
print(confusion_matrix(prediction,y_test))
print("\n")
print("accuracy_score")
print(accuracy_score(prediction,y_test))
print("\n")

LogisticRegressionscore:

confusion_matrix
[[40 0]
[5 35]]

accuracy_score
0.9375

C:\Users\ELCOT\anaconda3\lib\site-packages\sklearn\base.py:443: UserWarning: X has feature names, but LogisticRegression was fitted without feature names
warnings.warn(

In [33]: #Using RandomForestClassifier

In [34]: from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(x_train,y_train)
prediction = model.predict(x_test)
from sklearn.metrics import confusion_matrix
print("\n")
print("confusion_matrix:")
print(confusion_matrix(prediction,y_test))
print("\n")
print("accuracy_score")
print(accuracy_score(prediction,y_test))
print("\n")

C:\Users\ELCOT\AppData\Local\Temp\ipykernel_9732\2985509392.py:3: 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().
model.fit(x_train,y_train)
RandomForest

confusion_matrix
[[34 1]
[1 44]]

accuracy_score
0.9625

In [35]: #Using DecisionTreeClassifier

In [36]: from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
print('Decision tree')
model.fit(x_train,y_train)
prediction = model.predict(x_test)
from sklearn.metrics import confusion_matrix
print("confusion_matrix:")
print(confusion_matrix(prediction,y_test))
print("\n")
print("accuracy_score")
print(accuracy_score(prediction,y_test))
print("\n")

Decision tree

confusion_matrix
[[34 0]
[3 43]]

accuracy_score
0.925

In [37]: #SAVE THE MODEL

In [38]: import joblib
joblib.dump(model,'model.pkl')

Out[38]: ['model.pkl']

In [39]: