```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import roc_curve, auc, confusion_matrix,
classification_report,accuracy_score
from sklearn.ensemble import RandomForestClassifier
!pip install dython
from dython.nominal import associations
import warnings
warnings.filterwarnings('ignore')
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Requirement already satisfied: dython in /usr/local/lib/python3.7/dist-packages
(0.7.2)
Requirement already satisfied: scipy>=1.7.1 in /usr/local/lib/python3.7/dist-
packages (from dython) (1.7.3)
Requirement already satisfied: seaborn>=0.11.0 in /usr/local/lib/python3.7/dist-
packages (from dython) (0.11.2)
Requirement already satisfied: scikit-learn>=0.24.2 in
/usr/local/lib/python3.7/dist-packages (from dython) (1.0.2)
Requirement already satisfied: numpy>=1.19.5 in /usr/local/lib/python3.7/dist-
packages (from dython) (1.21.6)
Requirement already satisfied: scikit-plot>=0.3.7 in
/usr/local/lib/python3.7/dist-packages (from dython) (0.3.7)
Requirement already satisfied: matplotlib>=3.4.3 in
/usr/local/lib/python3.7/dist-packages (from dython) (3.5.3)
Requirement already satisfied: pandas>=1.3.2 in /usr/local/lib/python3.7/dist-
packages (from dython) (1.3.5)
Requirement already satisfied: psutil>=5.9.1 in /usr/local/lib/python3.7/dist-
packages (from dython) (5.9.4)

Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.7/dist-packages (from matplotlib>=3.4.3->dython) (2.8.2)
Requirement already satisfied: pyparsing>=2.2.1 in /usr/local/lib/python3.7/dist-
packages (from matplotlib>=3.4.3->dython) (3.0.9)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.7/dist-
packages (from matplotlib>=3.4.3->dython) (7.1.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-
packages (from matplotlib>=3.4.3->dython) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.7/dist-packages (from matplotlib>=3.4.3->dython) (4.38.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-
packages (from matplotlib>=3.4.3->dython) (21.3)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.7/dist-packages (from matplotlib>=3.4.3->dython) (1.4.4)
Requirement already satisfied: typing-extensions in
```

```
/usr/local/lib/python3.7/dist-packages (from kiwisolver>=1.0.1-
>matplotlib>=3.4.3->dython) (4.1.1)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-
packages (from pandas>=1.3.2->dython) (2022.5)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages
(from python-dateutil>=2.7->matplotlib>=3.4.3->dython) (1.15.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.24.2->dython)
(3.1.0)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-
packages (from scikit-learn>=0.24.2->dython) (1.2.0)
```

In [ ]:

```
df=pd.read_csv(r"/IBM/chronickidneydisease.csv")
df
```

Out[ ]:

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	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	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	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	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	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	...	35	7300	4.6	no	no	no	good	no	no	ckd
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
395	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	396	42.0	70.0	1.025	0.0	0.0	normal	normal	notpresent	notpresent	...	54	7800	6.2	no	no	no	good	no	no	notckd
397	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	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	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

EXPLORATORY DATA ANALYSIS

```
df.duplicated().sum(
```

Out[ ]:

```
0
```

In [ ]:

```
df.info()
```

```
RangeIndex: 400 entries, 0 to 399
Data columns (total 26 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   id              400 non-null    int64  
 1   age             391 non-null    float64
 2   bp              388 non-null    float64
 3   sg              353 non-null    float64
 4   al              354 non-null    float64
 5   su              351 non-null    float64
 6   rbc             248 non-null    object 
 7   pc              335 non-null    object 
 8   pcc             396 non-null    object 
 9   ba              396 non-null    object 
 10  bgr             356 non-null    float64
 11  bu              381 non-null    float64
 12  sc              383 non-null    float64
 13  sod             313 non-null    float64
```

```
 14  pot             312 non-null    float64
 15  hemo            348 non-null    float64
 16  pcv             330 non-null    object
 17  wc              295 non-null    object
 18  rc              270 non-null    object
 19  htn             398 non-null    object
 20  dm              398 non-null    object
 21  cad             398 non-null    object
 22  appet           399 non-null    object
 23  pe              399 non-null    object
 24  ane             399 non-null    object
 25  classification  400 non-null    object
dtypes: float64(11), int64(1), object(14)
memory usage: 81.4+ KB
```

CONVERTING DATA TYPES

```
def convert_dtype(df,feature):
    df[feature]=pd.to_numeric(df[feature],errors='coerce')
    #whereever we have Nan values , this errors parameter will hanfle that
```

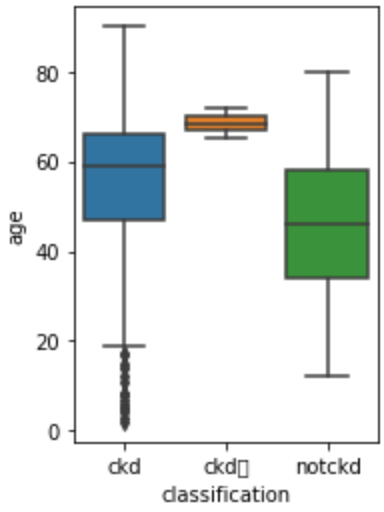
In [ ]:

```
features=['pcv','wc','rc']
for i in features:
    convert_dtype(df,i)
```

CHECKING THE CKD AND NOT CKD

```
plt.subplot(1,2,1)
sns.boxplot(x=df['classification'],y=df['age'])
```

Out[ ]:



In [ ]:

```
def extract_cat_num(kidney):
    cat_col=[col for col in kidney.columns if kidney[col].dtype=='O']
    num_col=[col for col in kidney.columns if kidney[col].dtype!='O']
    return cat_col,num_col
```

In [ ]:

```
cat_col,num_col=extract_cat_num(df)
```

Analysing distribution of each and every column

```
len(num_col)
```

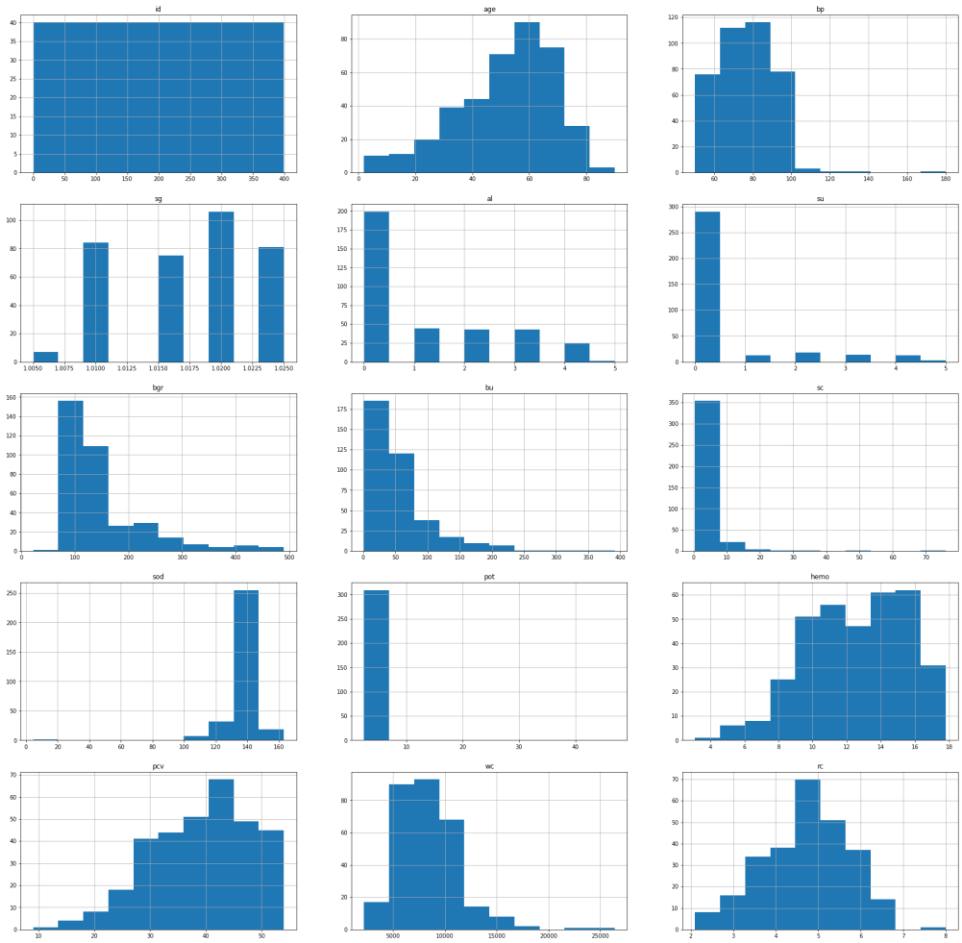
Out[ ]:

```
15
```

In [ ]:

```
plt.figure(figsize=(30,30))
for i,feature in enumerate(num_col):
    plt.subplot(5,3,i+1)
    df[feature].hist()
```

```
    plt.title(feature)
```



Check distribution of categorical Data

```
len(cat_col)
```

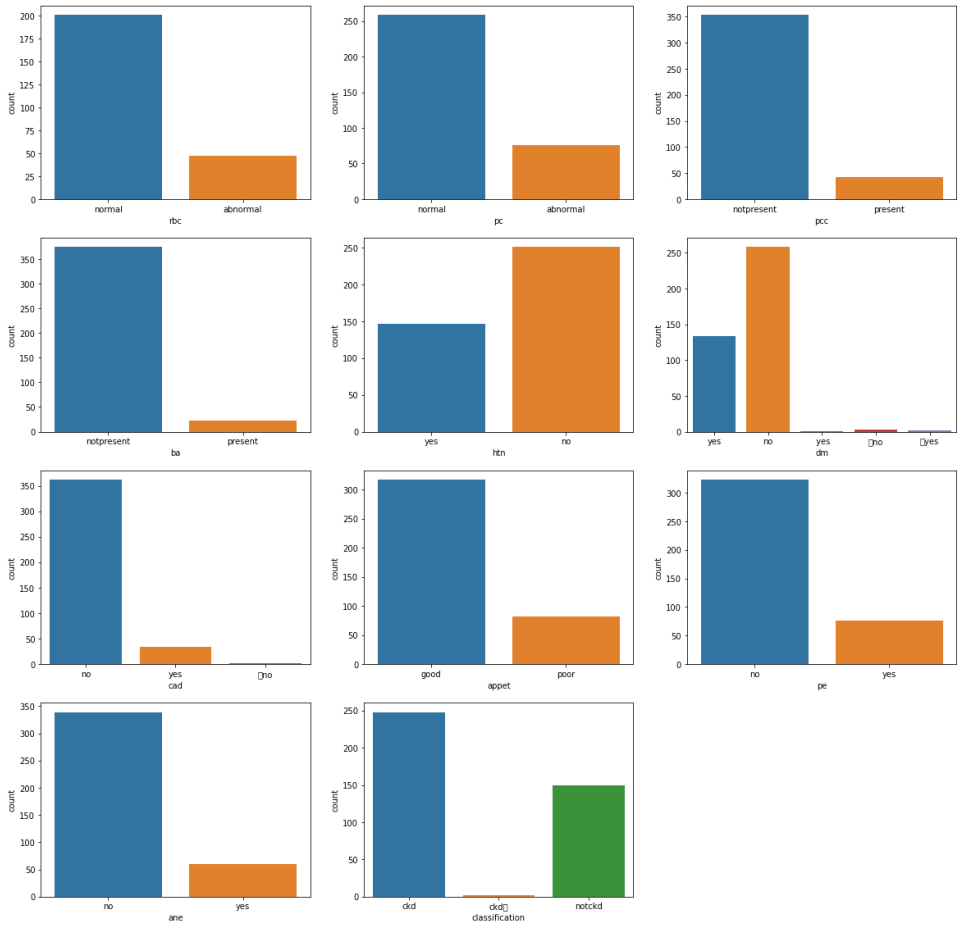
Out[ ]:

```
11
```

In [ ]:

```
plt.figure(figsize=(20,20))

for i,feature in enumerate(cat_col):
    plt.subplot(4,3,i+1)
    sns.countplot(df[feature])
```




```
df.groupby(['rbc','classification'])['rc'].agg(['count','mean','median','min','ma
x'])
```

Out[ ]:

		count	mean	median	min	max
rbc	**classification**					
abnormal	**ckd**	25	3.832000	3.7	2.5	5.6
	ckd	40	3.782500	3.8	2.1	8.0
normal	**ckd\t**	0	NaN	NaN	NaN	NaN
	notckd	134	5.368657	5.3	4.4	6.

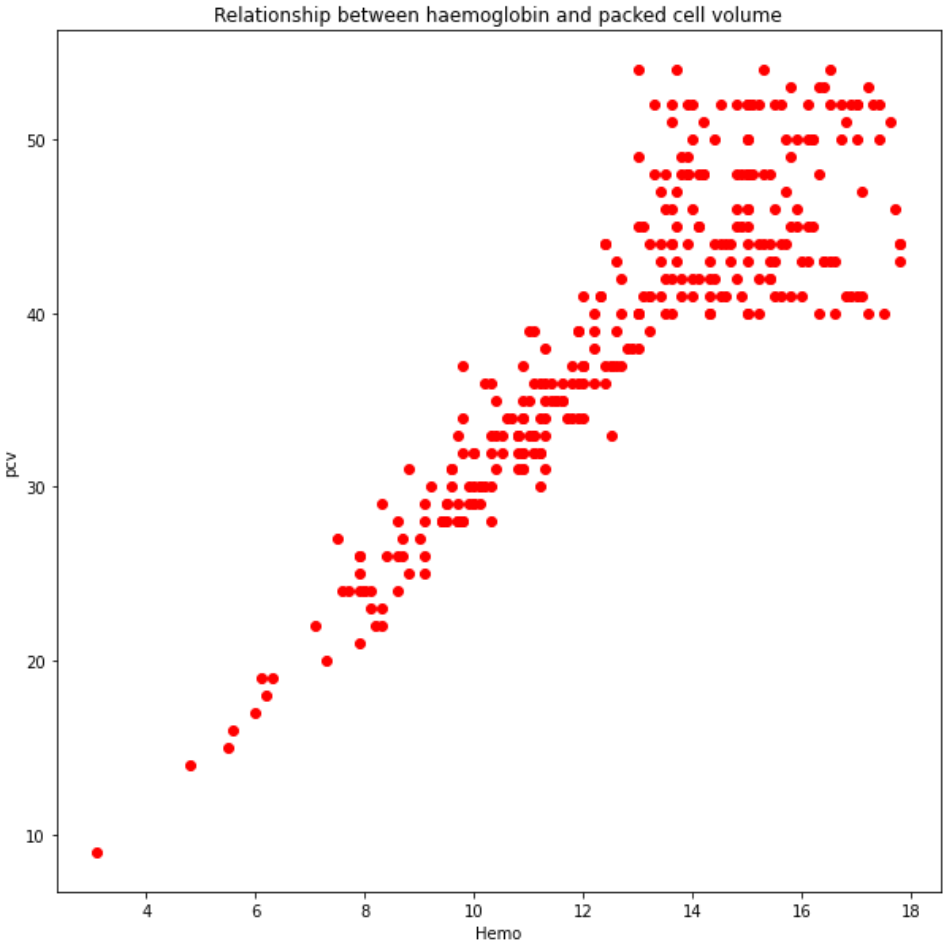
Relationship between haemoglobin and packed cell volume

```
plt.figure(figsize=(10,10))
plt.scatter(x=df.hemo,y=df['pcv'],color="red")
```

```
plt.xlabel('Hemo')
plt.ylabel('pcv')
plt.title('Relationship between haemoglobin and packed cell volume')
```

Out[ ]:

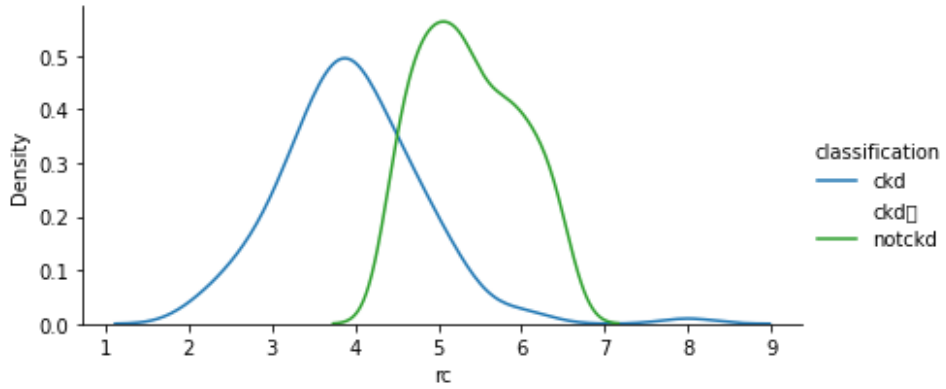
```
Text(0.5, 1.0, 'Relationship between haemoglobin and packed cell volume')
```



Analyse distribution of red blood cell count chronic as well as non chronic

```
grid=sns.FacetGrid(df,hue='classification',aspect=2)
grid.map(sns.kdeplot,'rc')
grid.add_legend()
```

Out[ ]:

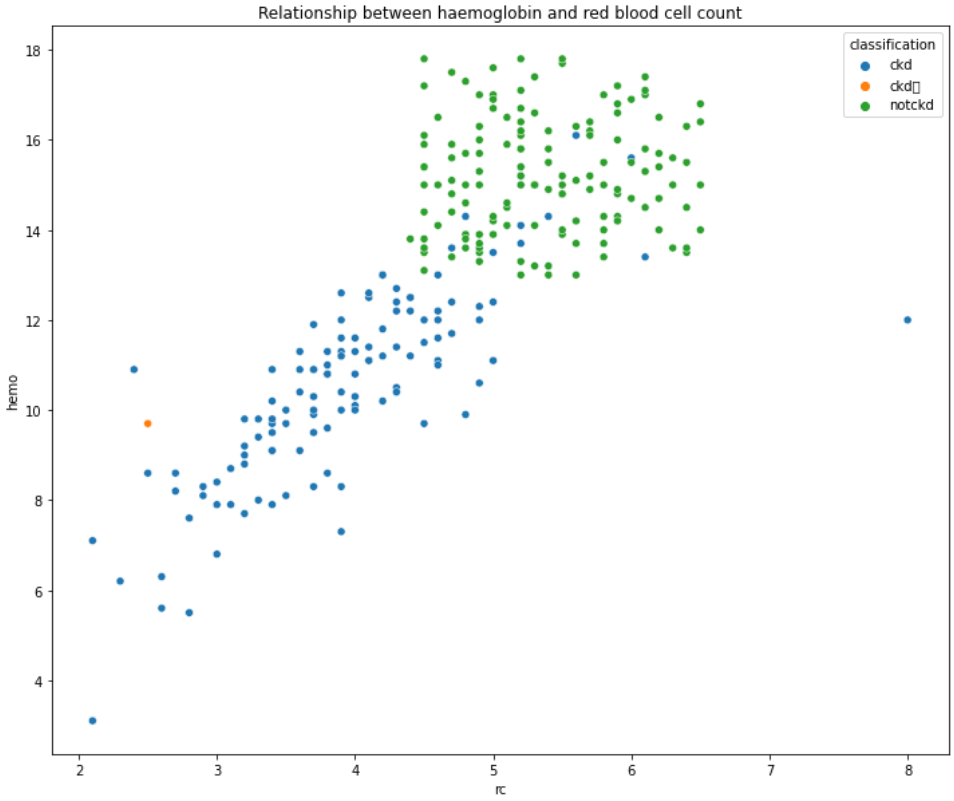


In [ ]:

```
plt.figure(figsize=(12,10))
sns.scatterplot(x=df['rc'],y=df['hemo'],hue=df['classification'])
plt.xlabel('rc')
plt.ylabel('hemo')
plt.title('Relationship between haemoglobin and red blood cell count')
```

Out[ ]:

```
Text(0.5, 1.0, 'Relationship between haemoglobin and red blood cell count')
```




```
from dython.nominal import identify_nominal_columns
categorical_features=identify_nominal_columns(df)
```

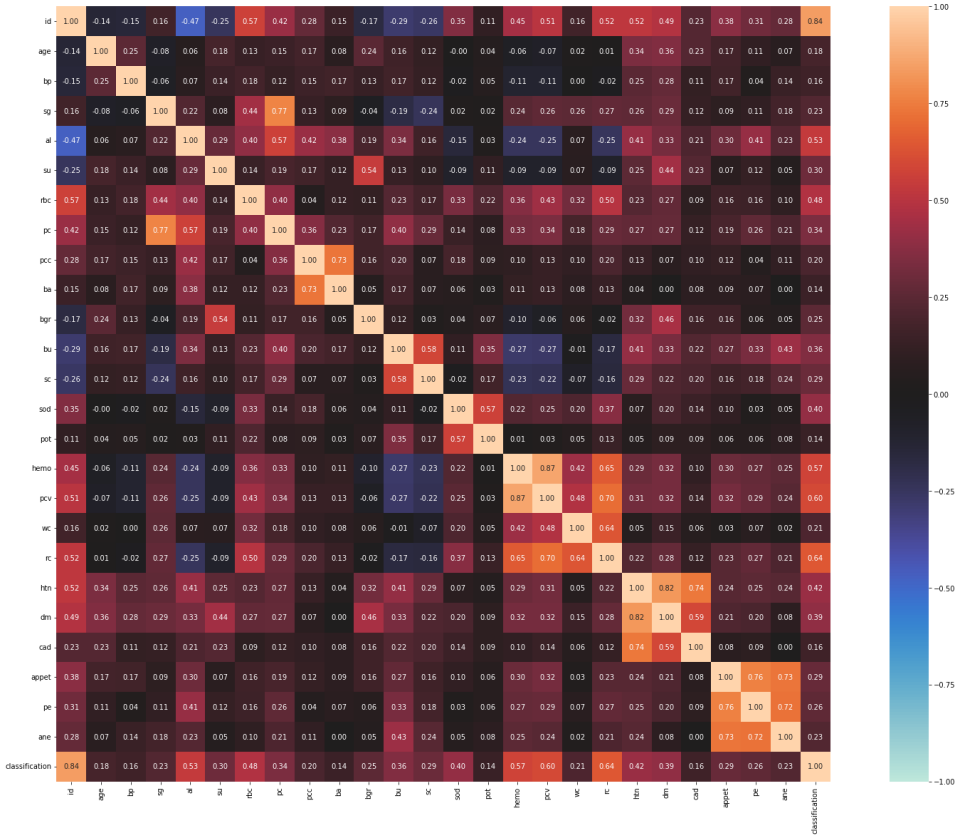
```
categorical_features
```

Out[ ]:

```
['rbc',
 'pc',
 'pcc',
 'ba',
 'htn',
 'dm',
 'cad',
 'appet',
 'pe',
 'ane',
 'classification']
```

In [ ]:

```
complete_correlation= associations(df, filename= 'complete_correlation.png',
figsize=(30,20))
```



COMPLETE CORRELATION DATASET

```
df_complete_corr=complete_correlation['corr']
df_complete_corr.dropna(axis=1, how='all').dropna(axis=0,
how='all').style.background_gradient(cmap='nipy_spectral_r',
axis=None).set_precision(2)
```

Out[ ]:

	id	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
id	1.00	-0.14	-0.15	0.16	-0.47	-0.25	0.57	0.42	0.28	0.15	-0.17	-0.29	-0.26	0.35	0.11	0.45	0.51	0.16	0.52	0.52	0.49	0.23	0.38	0.31	0.28	0.84
age	-0.14	1.00	0.25	-0.08	0.06	0.18	0.13	0.15	0.17	0.08	0.24	0.16	0.12	-0.00	0.04	-0.06	-0.07	0.02	0.01	0.34	0.36	0.23	0.17	0.11	0.07	0.18
bp	-0.15	0.25	1.00	-0.06	0.07	0.14	0.18	0.12	0.15	0.17	0.13	0.17	0.12	-0.02	0.05	0.11	0.11	0.00	-0.02	0.25	0.28	0.11	0.17	0.04	0.14	0.16
sg	0.16	-0.08	-0.06	1.00	0.22	0.08	0.44	0.77	0.13	0.09	0.04	0.19	0.24	0.02	0.02	0.24	0.26	0.26	0.27	0.26	0.29	0.12	0.09	0.11	0.18	0.23
al	-0.47	0.06	0.07	0.22	1.00	0.29	0.40	0.57	0.42	0.38	0.19	0.34	0.16	-0.15	0.03	-0.24	-0.25	0.07	-0.25	0.41	0.33	0.21	0.30	0.41	0.23	0.53
su	-0.25	0.18	0.14	0.08	0.29	1.00	0.14	0.19	0.17	0.12	0.54	0.13	0.10	-0.09	0.11	-0.09	-0.09	0.07	-0.09	0.25	0.44	0.23	0.07	0.12	0.05	0.30
rbc	0.57	0.13	0.18	0.44	0.40	0.14	1.00	0.40	0.40	0.12	0.11	0.23	0.17	0.33	0.22	0.36	0.43	0.32	0.50	0.23	0.27	0.09	0.16	0.16	0.10	0.48
pc	0.42	0.15	0.12	0.77	0.57	0.19	0.40	0.00	0.36	0.23	0.17	0.40	0.29	0.14	0.08	0.33	0.34	0.18	0.29	0.27	0.27	0.12	0.19	0.26	0.21	0.34

pcc	0.28	0.17	0.15	0.13	0.42	0.17	0.04	0.36	1.00	0.73	0.16	0.20	0.07	0.18	0.09	0.10	0.13	0.10	0.20	0.13	0.07	0.10	0.12	0.04	0.11	0.20
ba	0.15	0.08	0.17	0.09	0.38	0.12	0.12	0.23	0.73	1.00	0.05	0.17	0.07	0.06	0.03	0.11	0.13	0.08	0.13	0.04	0.00	0.08	0.09	0.07	0.00	0.14
bgr	-0.17	0.24	0.13	-0.04	0.19	0.54	0.11	0.17	0.16	0.05	1.00	0.12	0.03	0.04	0.07	-0.10	-0.06	0.06	-0.02	0.32	0.46	0.16	0.16	0.06	0.05	0.25
bu	-0.29	0.16	0.17	-0.19	0.34	0.13	0.23	0.40	0.20	0.17	0.12	1.00	0.58	0.11	0.35	-0.27	-0.27	-0.01	-0.17	0.41	0.33	0.22	0.27	0.33	0.43	0.36
sc	-0.26	0.12	0.12	-0.24	0.16	0.10	0.17	0.29	0.07	0.07	0.03	0.58	1.00	-0.02	0.17	-0.23	-0.22	-0.07	-0.16	0.29	0.22	0.20	0.16	0.18	0.24	0.29
sod	0.35	-0.00	-0.02	0.02	-0.15	-0.09	0.33	0.14	0.18	0.06	0.04	0.11	-0.02	1.00	0.57	0.22	0.25	0.20	0.37	0.07	0.20	0.14	0.10	0.03	0.05	0.40
pot	0.11	0.04	0.05	0.02	0.03	0.11	0.22	0.08	0.09	0.03	0.07	0.35	0.17	0.57	1.00	0.01	0.03	0.05	0.13	0.05	0.09	0.09	0.06	0.06	0.08	0.14
hemo	0.45	-0.06	-0.11	0.24	-0.24	-0.09	0.36	0.33	0.10	0.11	-0.10	-0.27	-0.23	0.22	0.01	1.00	0.87	0.42	0.65	0.29	0.32	0.10	0.30	0.27	0.25	0.57
pcv	0.51	-0.07	-0.11	0.26	-0.25	-0.09	0.43	0.34	0.13	0.13	-0.06	-0.27	-0.22	0.25	0.03	0.87	1.00	0.48	0.70	0.31	0.32	0.14	0.32	0.29	0.24	0.60
wc	0.16	0.02	0.00	0.26	0.07	0.07	0.32	0.18	0.10	0.08	0.06	-0.01	-0.07	0.20	0.05	0.42	0.48	1.00	0.64	0.05	0.15	0.06	0.03	0.07	0.02	0.21
rc	0.52	0.01	-0.02	0.27	-0.25	-0.09	0.50	0.29	0.20	0.13	-0.02	-0.17	-0.16	0.37	0.13	0.65	0.70	0.64	1.00	0.22	0.28	0.12	0.23	0.27	0.21	0.64
htn	0.52	0.34	0.25	0.26	0.41	0.25	0.23	0.27	0.13	0.04	0.32	0.41	0.29	0.07	0.05	0.29	0.31	0.05	0.22	1.00	0.82	0.74	0.24	0.25	0.24	0.42
dm	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.	0.	0.

cad	49	36	28	29	33	44	27	27	07	00	46	33	22	20	09	32	32	15	28	82	00	59	21	20	08	39	
	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	10	00	00	00	00	
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
	23	23	11	12	21	23	09	12	10	08	16	22	20	14	09	10	14	06	12	74	59	00	08	09	00	16	
appet	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
	38	17	17	09	30	07	16	19	12	09	16	27	16	06	06	30	23	23	24	41	18	00	76	73	29		
pe	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0		
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.		
	31	11	04	11	41	12	16	26	04	07	06	33	18	03	06	07	29	07	27	25	09	76	00	72	60		
	10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	10	00		
ane	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.		
	28	07	14	18	23	05	10	21	11	00	05	43	24	05	08	25	24	02	21	24	08	00	73	72	00	23	
classification	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
	84	18	16	23	53	30	48	34	20	14	25	36	29	40	14	57	60	21	64	42	39	16	29	26	23	00	

```
df.corr().style.background_gradient(cmap="nipy_spectral_r")
```

Out[ ]:

	id	age	bp	sg	al	su	bgr	bu	sc	sod	pot	hemo	pcv	wc	rc
id	1.000000	-0.185308	-0.245744	0.642156	-0.541993	-0.283416	-0.338673	-0.307175	-0.268683	0.364251	-0.092347	0.640298	0.630019	-0.198641	0.605072
age	-0.185308	1.000000	0.159480	-0.191096	0.122091	0.220866	0.244992	0.196985	0.132531	-0.100046	0.058377	-0.192928	-0.242119	0.118339	-0.268896
bp	-0.245744	0.159480	1.000000	-0.218836	0.160689	0.222576	0.160193	0.188517	0.146222	-0.116422	0.075151	-0.306540	-0.326319	0.029753	-0.261936
sg	0.642156	-0.191096	-0.218836	1.000000	-0.469760	-0.296234	-0.374710	-0.314295	-0.361473	0.412190	-0.072787	0.602582	0.603560	-0.236215	0.579476
al	-0.541993	0.122091	0.160689	-0.469760	1.000000	0.269305	0.379464	0.453528	0.399198	-0.459896	0.129038	-0.634632	-0.611891	0.231989	-0.566437
su	-0.283416	0.220866	0.222576	-0.296234	0.269305	1.000000	0.717827	0.168583	0.223244	-0.131776	0.219450	-0.224775	-0.239189	0.184893	-0.237448

bgr	-0.338673	0.244992	0.160193	-0.374710	0.379464	0.717827	1.000000	0.143322	0.114875	-0.267848	0.066966	-0.306189	-0.301385	0.150015	-0.281541
bu	-0.307175	0.196985	0.188517	-0.314295	0.453528	0.168583	0.143322	1.000000	0.586368	-0.323054	0.357049	-0.610360	-0.607621	0.050462	-0.579087
sc	-0.268683	0.132531	0.146222	-0.361473	0.399198	0.223244	0.114875	0.586368	1.000000	-0.690158	0.326107	-0.401670	-0.404193	0.006390	-0.400852
sod	0.364251	-0.100046	-0.116422	0.412190	-0.459896	-0.131776	-0.267848	-0.323054	-0.690158	1.000000	0.097887	0.365183	0.376914	0.007277	0.344873
pot	-0.092347	0.058377	0.075151	-0.072787	0.129038	0.219450	0.066966	0.357049	0.326107	0.097887	1.000000	-0.133746	-0.163182	-0.105576	-0.158309
hemo	0.640298	-0.192928	-0.306540	0.602582	-0.634632	-0.224775	-0.306189	-0.610360	-0.401670	0.365183	-0.133746	1.000000	0.895382	-0.169413	0.798880
pcv	0.630019	-0.242119	-0.326319	0.603560	-0.611891	-0.239189	-0.301385	-0.607621	-0.404193	0.376914	-0.163182	0.895382	1.000000	-0.197022	0.791625
wc	-0.198641	0.118339	0.029753	-0.236215	0.231989	0.184893	0.150015	0.050462	-0.006390	0.007277	-0.105576	-0.169413	-0.197022	1.000000	-0.158163
rc	0.605072	-0.268896	-0.261936	0.579476	-0.566437	-0.237448	-0.281541	-0.579087	-0.400852	0.344873	-0.158309	0.798880	0.791625	-0.158163	1.000000

```
df.corr().style.background_gradient(cmap="nipy_spectral_r")
```

Out[ ]:

	id	**age**	**bp**	**sg**	**al**	**su**	**bgr**	**bu**	**sc**	**sod**	**pot**	**hemo**	**pcv**	**wc**	**rc**
id	1.000000	-0.185308	-0.245744	0.642156	-0.541993	-0.283416	-0.338673	-0.307175	-0.268683	0.364251	-0.092347	0.640298	0.630019	-0.198641	0.605072
age	-0.185308	1.000000	0.159480	-0.191096	0.122091	0.220866	0.244992	0.196985	0.132531	-0.100046	0.058377	-0.192928	-0.242119	0.118339	-0.268896
bp	-0.245744	0.159480	1.000000	-0.218836	0.160689	0.222576	0.160193	0.188517	0.146222	-0.116422	0.075151	-0.306540	-0.326319	0.029753	-0.261936
sg	0.642156	-0.191096	-0.218836	1.000000	-0.469760	-0.296234	-0.374710	-0.314295	-0.361473	0.412190	-0.072787	0.602582	0.603560	-0.236215	0.579476
al	-0.541993	0.122091	0.160689	-0.469760	1.000000	0.269305	0.379464	0.453528	0.399198	-0.459896	0.129038	-0.634632	-0.611891	0.231989	-0.566437
su	-0.283416	0.220866	0.222576	-0.296234	0.269305	1.000000	0.717827	0.168583	0.223244	-0.131776	0.219450	-0.224775	-0.239189	0.184893	-0.237448
bgr	-0.338673	0.244992	0.160193	-0.374710	0.379464	0.717827	1.000000	0.143322	0.114875	-0.267848	0.066966	-0.306189	-0.301385	0.150015	-0.281541

bu	-0.307175	0.196985	0.188517	-0.314295	0.453528	0.168583	0.143322	1.000000	0.586368	-0.323054	0.357049	-0.610360	-0.607621	0.050462	-0.579087
sc	-0.268683	0.132531	0.146222	-0.361473	0.399198	0.223244	0.114875	0.586368	1.000000	-0.690158	0.326107	-0.401670	-0.404193	-0.006390	-0.400852
sod	0.364251	-0.100046	-0.116422	0.412190	-0.459896	-0.131776	-0.267848	-0.323054	-0.690158	1.000000	0.097887	0.365183	0.376914	0.007277	0.344873
pot	-0.092347	0.058377	0.075151	-0.072787	0.129038	0.219450	0.066966	0.357049	0.326107	0.097887	1.000000	-0.133746	-0.163182	-0.105576	-0.158309
hemo	0.640298	-0.192928	-0.306540	0.602582	-0.634632	-0.224775	-0.306189	-0.610360	-0.401670	0.365183	-0.133746	1.000000	0.895382	-0.169413	0.798880
pcv	0.630019	-0.242119	-0.326319	0.603560	-0.611891	-0.239189	-0.301385	-0.607621	-0.404193	0.376914	-0.163182	0.895382	1.000000	-0.197022	0.791625
wc	-0.198641	0.118339	0.029753	-0.236215	0.231989	0.184893	0.150015	0.050462	-0.006390	-0.007277	-0.105576	-0.169413	-0.197022	1.000000	-0.158163
rc	0.605072	-0.268896	-0.261936	0.579476	-0.566437	-0.237448	-0.281541	-0.579087	-0.400852	0.344873	-0.158309	0.798880	0.791625	-0.158163	1.000000

DESCRIPTIVE STATISTICS

```
df.describe()
```

Out[ ]:

	id	**age**	**bp**	**sg**	**al**	**su**	**bgr**	**bu**	**sc**	**sod**	**pot**	**hemo**	**pcv**	**wc**	**rc**
count	400.000000	391.000000	388.000000	353.000000	354.000000	351.000000	356.000000	381.000000	383.000000	313.000000	312.000000	348.000000	329.000000	294.000000	269.000000
mean	199.500000	51.483376	76.469072	1.017408	1.016949	0.450142	148.036517	57.425722	3.072454	137.528754	4.627244	12.526437	38.884498	8406.122449	4.707435
std	115.614301	17.169714	13.683637	0.005717	1.352679	1.099191	79.281714	50.503006	5.741126	10.408752	3.193904	2.912587	8.990105	2944.474190	1.025323
min	0.000000	2.000000	50.000000	1.005000	0.000000	0.000000	22.000000	1.500000	0.400000	4.500000	2.500000	3.100000	9.000000	2200.000000	2.100000
25%	99.750000	42.000000	70.000000	1.010000	0.000000	0.000000	99.000000	27.000000	0.900000	135.000000	3.800000	10.300000	32.000000	6500.000000	3.900000
50%	199.500000	55.000000	80.000000	1.020000	0.000000	0.000000	121.000000	42.000000	1.300000	138.000000	4.400000	12.650000	40.000000	8000.000000	4.800000

75%	299.250000	64.500000	80.000000	1.020000	2.000000	0.000000	163.000000	66.000000	2.800000	142.000000	4.900000	15.000000	45.000000	9800.000000	5.400000
max	399.000000	90.000000	180.000000	1.025000	5.000000	5.000000	490.000000	391.000000	76.000000	163.000000	47.000000	17.800000	54.000000	26400.000000	

NULL VALUES

```
missing_values=df.columns[df.isnull().any()]
df[missing_values].isnull().sum()
```

Out[ ]:

```
age        9
bp        12
sg        47
al        46
su        49
rbc      152
pc        65
pcc        4
ba         4
bgr       44
bu        19
sc        17
sod       87
pot       88
hemo      52
pcv       71
wc       106
rc       131
htn        2
dm         2
cad        2
appet      1
pe         1
ane        1
dtype: int64
```

GRAPHICAL REPRESENTATION OF NULL VALUES

```
labels = []
valuecount = []
percentcount = []
```

```
for col in missing_values:
    labels.append(col)
    valuecount.append(df[col].isnull().sum())
    percentcount.append(df[col].isnull().sum()/df.shape[0])
ind = np.arange(len(labels))
fig, (ax1, ax2) = plt.subplots(1,2,figsize=(10,5))
rects = ax1.barh(ind, np.array(valuecount), color='yellow')
ax1.set_yticks(ind)
ax1.set_yticklabels(labels, rotation='horizontal')
ax1.set_xlabel("Count of missing values")
ax1.set_title("Variables with missing values");
rects = ax2.barh(ind, np.array(percentcount), color='green')
ax2.set_yticks(ind)
ax2.set_yticklabels(labels, rotation='horizontal')
ax2.set_xlabel("Percentage of missing values")
ax2.set_title("Variables with missing values");
```