Date	09 November 2022
Team ID	PNT2022TMID05452
Project Name	Early Detection Of Chronic Kidney Disease Using Machine Learning
Dataset	Sample data/Dataset

Import Packages

```
In [6]:
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
import tensorflow
from tensorflow import keras
from keras.layers import Dense
```

Read dataset

```
In [43]:
data = pd.read_csv("/content/sample_data/Dataset_CKD.csv")
print(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]
```

Understanding Data Type and Features

```
In [44]:
print(data.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
None
```

Handling Missing Values

Remove null values

In [37]:
```
data=data.dropna(how="any")
print(data)
```

```
      id   age    bp     sg   al   su       rbc        pc         pcc  \
3      3  48.0  70.0  1.005  4.0  0.0    normal  abnormal     present
9      9  53.0  90.0  1.020  2.0  0.0  abnormal  abnormal     present
11    11  63.0  70.0  1.010  3.0  0.0  abnormal  abnormal     present
14    14  68.0  80.0  1.010  3.0  2.0    normal  abnormal     present
20    20  61.0  80.0  1.015  2.0  0.0  abnormal  abnormal  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  \
3    notpresent  ...   32   6700  3.9  yes   no   no  poor  yes  yes
9    notpresent  ...   29  12100  3.7  yes  yes   no  poor   no  yes
11   notpresent  ...   32   4500  3.8  yes  yes   no  poor  yes   no
14      present  ...   16  11000  2.6  yes  yes  yes  poor  yes   no
20   notpresent  ...   24   9200  3.2  yes  yes  yes  poor  yes  yes
..          ...  ...  ...    ...  ...  ...  ...  ...   ...  ...  ...
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


9              ckd
11             ckd
14             ckd
20             ckd
..             ...
395         notckd
396         notckd
397         notckd
398         notckd
399         notckd

[158 rows x 26 columns]
```

Label Encoding (String values to Numeric values)

In [38]:
```
data['rbc'] = data['rbc'].map({"abnormal":1,"normal":0})
data['pc'] = data['pc'].map({"abnormal":1,"normal":0})
data['pcc'] = data['pcc'].map({"present":1,"notpresent":0})
data['ba'] = data['ba'].map({"present":1,"notpresent":0})
data['htn'] = data['htn'].map({"yes":1,"no":0})
data['dm'] = data['dm'].map({"yes":1,"no":0})
data['cad'] = data['cad'].map({"yes":1,"no":0})
data['pe'] = data['pe'].map({"yes":1,"no":0})
data['ane'] = data['ane'].map({"yes":1,"no":0})
data['appet'] = data['appet'].map({"poor":1,"good":0})
data['classification'] = data['classification'].map({"ckd":1,"notckd":0})
data['pcv'] = data['pcv'].astype('int')
data['wc'] = data['wc'].astype('int')
data['rc'] = data['rc'].astype('float')
print(data)
```

```
      id   age    bp     sg   al   su  rbc  pc  pcc  ba  ...  pcv     wc   rc  \
3      3  48.0  70.0  1.005  4.0  0.0    0   1    1   0  ...   32   6700  3.9
9      9  53.0  90.0  1.020  2.0  0.0    1   1    1   0  ...   29  12100  3.7
11    11  63.0  70.0  1.010  3.0  0.0    1   1    1   0  ...   32   4500  3.8
14    14  68.0  80.0  1.010  3.0  2.0    0   1    1   1  ...   16  11000  2.6
20    20  61.0  80.0  1.015  2.0  0.0    1   1    0   0  ...   24   9200  3.2
..   ...   ...   ...    ...  ...  ...  ...  ..  ...  ..  ...  ...    ...  ...
395  395  55.0  80.0  1.020  0.0  0.0    0   0    0   0  ...   47   6700  4.9
396  396  42.0  70.0  1.025  0.0  0.0    0   0    0   0  ...   54   7800  6.2
397  397  12.0  80.0  1.020  0.0  0.0    0   0    0   0  ...   49   6600  5.4
398  398  17.0  60.0  1.025  0.0  0.0    0   0    0   0  ...   51   7200  5.9
399  399  58.0  80.0  1.025  0.0  0.0    0   0    0   0  ...   53   6800  6.1
```

```
     htn  dm  cad  appet  pe  ane  classification
3      1   0    0      1   1    1               1
9      1   1    0      1   0    1               1
11     1   1    0      1   1    0               1
14     1   1    1      1   1    0               1
20     1   1    1      1   1    1               1
..   ...  ..  ...    ...  ..  ...             ...
395    0   0    0      0   0    0               0
396    0   0    0      0   0    0               0
397    0   0    0      0   0    0               0
398    0   0    0      0   0    0               0
399    0   0    0      0   0    0               0

[158 rows x 26 columns]
```

Splitting Dependent and Independent Variable

In [39]:

```
X = data.iloc[:,1:25].values
y = data.iloc[:, 25].values
```

Split Train and Test set

In [41]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.30,
random_state = 121)#101
print("X train value")
print(X_train)
print("Y train value")
print(y_train)
```

```
X train value
[[75.    70.     1.02  ...  0.     0.     0.   ]
 [57.    60.     1.02  ...  0.     0.     0.   ]
 [66.    70.     1.025 ...  0.     0.     0.   ]
 ...
 [58.    80.     1.02  ...  0.     0.     0.   ]
 [73.    80.     1.02  ...  0.     0.     0.   ]
 [46.    60.     1.025 ...  0.     0.     0.   ]]
Y train value
[0 0 0 0 0 1 0 0 0 1 0 1 1 0 0 1 1 0 1 0 0 1 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 1 0 0 1 1 0 1 0 0 0 0 1 0 1 0 1 0 1 0 0 0
 0 1 1 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 1 1 0 0 0 1 0 0 1 0 0 0 0 0 1 0]
```