PREDICTION OF CHRONIC KIDNEY DISORDER


```
- Vaishnavi D R (1DT16IS103)
  Srujana S K   (1DT16IS098)
  Sazal Malhotra(1DT16IS091)
  Sajid Hasmi   (1DT16IS086)
```


Predicting Chronic Kidney Disease based on health records using Logistic Regression and Decision Tree algorithms

Given 24 health related attributes taken in 2-month period of 400 patients, using the information of the 158 patients with complete records to predict the outcome (i.e. whether one has chronic kidney disease) of the remaining 242 patients (with missing values in their records).

This project is based on Data-mining and Machine Learning technique using Python and Scikit-learn. It is used to predict Chronic Kidney Disease of Patients.

Number of Instances: 400 (250 CKD, 150 notckd) Number of Attributes: 24 + class = 25 (11 numeric ,14 nominal)

1.Age(numerical)age in years 2.Blood Pressure(numerical) bp in mm/Hg 3.Specific Gravity(nominal) sg - (1.005,1.010,1.015,1.020,1.025) 4.Albumin(nominal) al - (0,1,2,3,4,5) 5.Sugar(nominal) su - (0,1,2,3,4,5) 6.Red Blood Cells(nominal) rbc - (normal,abnormal) 7.Pus Cell (nominal) pc - (normal,abnormal) 8.Pus Cell clumps(nominal) pcc - (present,notpresent) 9.Bacteria(nominal) ba - (present,notpresent) 10.Blood Glucose Random(numerical) bgr in mgs/dl 11.Blood Urea(numerical) bu in mgs/dl 12.Serum Creatinine(numerical) sc in mgs/dl 13.Sodium(numerical) sod in mEq/L 14.Potassium(numerical) pot in mEq/L 15.Hemoglobin(numerical) hemo in gms 16.Packed Cell Volume(numerical) 17.White Blood Cell Count(numerical) wc in cells/cumm 18.Red Blood Cell Count(numerical) rc in millions/cmm 19.Hypertension(nominal) htn - (yes,no) 20.Diabetes Mellitus(nominal) dm - (yes,no) 21.Coronary Artery Disease(nominal) cad - (yes,no) 22.Appetite(nominal)appet - (good,poor) 23.Pedal Edema(nominal) pe - (yes,no) 24.Anemia(nominal) ane - (yes,no) 25.Class (nominal) class - (ckd,notckd)

Step 1: Importing libraries

In [65]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

Step 2: Importing Dataset

In [66]:

```
dataset = pd.read_csv(r"C:\Users\admin\Desktop\ckd3.csv",header=0, na_values="?")
dataset.isnull()
```

Out[66]:

	Age	Bp	Sg	Al	Su	Rbc	Pc	Pcc	Ba	Bgr	...	Pcv	Wbcc	Rbcc
0	False	False	False	False	False	True	False	False	False	False	...	False	False	False
1	False	False	False	False	False	True	False	False	False	True	...	False	False	True
2	False	False	False	False	False	False	False	False	False	False	...	False	False	True
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False
5	False	False	False	False	False	True	True	False	False	False	...	False	False	False
6	False	False	False	False	False	True	False	False	False	False	...	False	True	True
7	False	True	False	False	False	False	False	False	False	False	...	False	False	False
8	False	False	False	False	False	False	False	False	False	False	...	False	False	False
9	False	False	False	False	False	False	False	False	False	False	...	False	False	False
10	False	False	False	False	False	True	False	False	False	False	...	False	True	True
11	False	False	False	False	False	False	False	False	False	False	...	False	False	False
12	False	False	False	False	False	True	False	False	False	False	...	False	False	False
13	False	False	True	True	True	True	True	False	False	False	...	True	True	True
14	False	False	False	False	False	False	False	False	False	False	...	False	False	False
15	False	False	False	False	False	True	False	False	False	False	...	False	False	False
16	False	False	False	False	False	True	False	False	False	False	...	True	True	True
17	False	False	True	True	True	True	True	False	False	False	...	True	True	True
18	False	False	False	False	False	True	False	False	False	False	...	False	False	False
19	False	False	False	False	False	True	False	False	False	False	...	False	False	False
20	False	False	False	False	False	False	False	False	False	False	...	False	False	False
21	False	False	True	True	True	True	True	False	False	True	...	False	False	False
22	False	False	False	False	False	False	False	False	False	False	...	False	False	False
23	False	False	False	False	False	True	False	False	False	True	...	True	True	True
24	False	False	False	False	False	False	False	False	False	True	...	False	False	False
25	False	False	False	False	False	True	False	False	False	False	...	False	False	False
26	False	False	False	False	False	True	False	False	False	False	...	False	False	False
27	False	False	False	False	False	False	False	False	False	False	...	False	False	False
28	False	False	True	False	False	True	True	False	False	False	...	True	True	True
29	False	False	False	False	False	False	False	False	False	True	...	False	True	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
370	False	False	False	False	False	False	False	False	False	False	...	False	False	False
371	False	False	False	False	False	False	False	False	False	False	...	False	False	False
372	False	False	False	False	False	False	False	False	False	False	...	False	False	False

	Age	Bp	Sg	Al	Su	Rbc	Pc	Pcc	Ba	Bgr	...	Pcv	Wbcc	Rbcc
373	False	False	False	False	False	False	False	False	False	False	...	False	False	False
374	False	False	False	False	False	False	False	False	False	False	...	False	False	False
375	False	False	False	False	False	False	False	False	False	False	...	False	False	False
376	False	False	False	False	False	False	False	False	False	False	...	False	False	False
377	False	False	False	False	False	False	False	False	False	False	...	False	False	False
378	False	False	False	False	False	False	False	False	False	True	...	False	False	False
379	False	False	False	False	False	False	False	False	False	False	...	False	False	False
380	False	False	False	False	False	False	False	False	False	False	...	False	False	False
381	False	False	False	False	False	True	True	False	False	False	...	False	False	False
382	False	False	False	False	False	False	False	False	False	False	...	False	False	False
383	False	False	False	False	False	False	False	False	False	False	...	False	False	False
384	False	False	False	False	False	False	False	False	False	False	...	False	False	False
385	False	False	False	False	False	False	False	False	False	False	...	False	False	False
386	False	False	False	False	False	False	False	False	False	False	...	False	False	False
387	False	False	False	False	False	False	False	False	False	False	...	False	False	False
388	False	False	False	False	False	False	False	False	False	False	...	False	False	False
389	False	False	False	False	False	False	False	False	False	False	...	False	False	False
390	False	False	False	False	False	False	False	False	False	False	...	False	False	False
391	False	False	False	False	False	False	False	False	False	False	...	False	False	False
392	False	False	False	False	False	False	False	False	False	False	...	False	False	False
393	False	False	False	False	False	False	False	False	False	False	...	False	False	False
394	False	False	False	False	False	False	False	False	False	False	...	False	False	False
395	False	False	False	False	False	False	False	False	False	False	...	False	False	False
396	False	False	False	False	False	False	False	False	False	False	...	False	False	False
397	False	False	False	False	False	False	False	False	False	False	...	False	False	False
398	False	False	False	False	False	False	False	False	False	False	...	False	False	False
399	False	False	False	False	False	False	False	False	False	False	...	False	False	False

400 rows × 25 columns

In [67]:

```
dataset.isnull().sum()
```

Out[67]:

```
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
Wbcc     106
Rbcc     131
Htn        2
Dm         0
Cad        2
Appet      1
pe         1
Ane        1
Class      0
dtype: int64
```

Step 3: Data preprocessing

In [68]:

```
# Convert nominal values to binary values
cleanup = {"Rbc":     {"normal": 1, "abnormal": 0},
           "Pc": {"normal": 1, "abnormal": 0},
           "Pcc": {"present": 1, "notpresent": 0},
           "Ba": {"present": 1, "notpresent": 0},
           "Htn": {"yes": 1, "no": 0},
           "Dm": {"yes": 1, "no": 0},
           "Cad": {"yes": 1, "no": 0},
           "Appet": {"good": 1, "poor": 0},
           "pe": {"yes": 1, "no": 0},
           "Ane": {"yes": 1, "no": 0},
          "Class": {"ckd": 1, "notckd": 0}}
```

In [69]:

```
# Replace binary values into dataset
dataset.replace(cleanup, inplace=True)
```

In [70]:

```
# Fill null values with mean value of the respective column

dataset.fillna(round(dataset.mean(),2), inplace=True)
```

In [71]:

```
print(dataset)
```

```
      Age      Bp     Sg    Al    Su   Rbc    Pc  Pcc   Ba     Bgr  ...   \
0    48.0   80.00  1.020  1.00  0.00  0.81  1.00  0.0  0.0  121.00  ...
1     7.0   50.00  1.020  4.00  0.00  0.81  1.00  0.0  0.0  148.04  ...
2    62.0   80.00  1.010  2.00  3.00  1.00  1.00  0.0  0.0  423.00  ...
3    48.0   70.00  1.005  4.00  0.00  1.00  0.00  1.0  0.0  117.00  ...
4    51.0   80.00  1.010  2.00  0.00  1.00  1.00  0.0  0.0  106.00  ...
5    60.0   90.00  1.015  3.00  0.00  0.81  0.77  0.0  0.0   74.00  ...
6    68.0   70.00  1.010  0.00  0.00  0.81  1.00  0.0  0.0  100.00  ...
7    24.0   76.47  1.015  2.00  4.00  1.00  0.00  0.0  0.0  410.00  ...
8    52.0  100.00  1.015  3.00  0.00  1.00  0.00  1.0  0.0  138.00  ...
9    53.0   90.00  1.020  2.00  0.00  0.00  0.00  1.0  0.0   70.00  ...
10   50.0   60.00  1.010  2.00  4.00  0.81  0.00  1.0  0.0  490.00  ...
11   63.0   70.00  1.010  3.00  0.00  0.00  0.00  1.0  0.0  380.00  ...
12   68.0   70.00  1.015  3.00  1.00  0.81  1.00  1.0  0.0  208.00  ...
13   68.0   70.00  1.020  1.02  0.45  0.81  0.77  0.0  0.0   98.00  ...
14   68.0   80.00  1.010  3.00  2.00  1.00  0.00  1.0  1.0  157.00  ...
15   40.0   80.00  1.015  3.00  0.00  0.81  1.00  0.0  0.0   76.00  ...
16   47.0   70.00  1.015  2.00  0.00  0.81  1.00  0.0  0.0   99.00  ...
17   47.0   80.00  1.020  1.02  0.45  0.81  0.77  0.0  0.0  114.00  ...
```

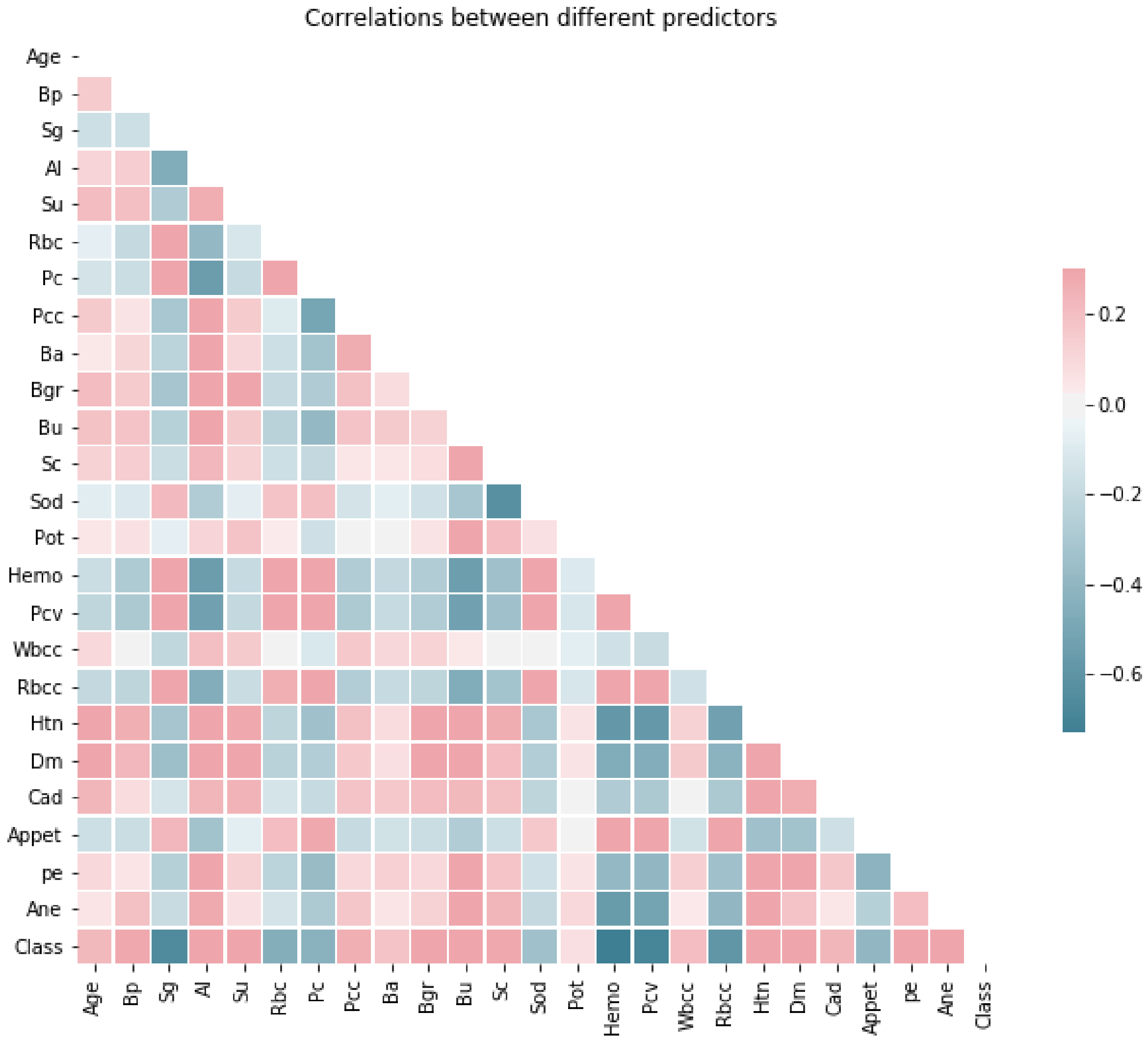
In [72]:

```
corr_df = dataset.corr()
# Generate a mask for the upper triangle
mask = np.zeros_like(corr_df, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(11, 9))

# Generate a custom diverging colormap
cmap = sns.diverging_palette(220, 10, as_cmap=True)

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr_df, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
plt.title('Correlations between different predictors')
plt.show()
```



In [73]:

```
# Save this dataset as final.csv for further prediction
dataset.to_csv("final.csv", sep=',', index=False)
```

Step 5: Defining feature matrix and target vector

In [74]:

```
X = dataset.values[:, 0:24]
Y = dataset.values[:, -1]
```

Step 6: Splitting the dataset

In [75]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( X, Y, test_size = 0.3, random_state =
```

Step 7: Model building (USING LOGISTIC REGRESSION)

In [76]:

```
from sklearn.linear_model import LogisticRegression
logmodel=LogisticRegression()
logmodel.fit(X_train,y_train)
predictions=logmodel.predict(X_test)
```

```
C:\Users\admin\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:
433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Speci
fy a solver to silence this warning.
  FutureWarning)
```

In [77]:

```
predictions
```

Out[77]:

```
array([1., 0., 1., 0., 1., 1., 1., 0., 1., 1., 1., 0., 0., 1., 0., 0., 1.,
       1., 1., 1., 0., 1., 1., 1., 0., 1., 1., 0., 1., 1., 1., 1., 1., 1.,
       0., 0., 1., 0., 1., 1., 1., 1., 1., 0., 0., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 0., 0., 0., 1., 0., 0., 0., 1., 0., 1., 1., 0., 0., 1.,
       1., 0., 1., 1., 1., 1., 1., 0., 1., 1., 0., 1., 1., 1., 1., 0., 1.,
       1., 1., 1., 1., 1., 1., 0., 0., 0., 0., 0., 0., 1., 1., 1., 1., 1.,
       1., 1., 0., 1., 1., 1., 1., 1., 0., 1., 1., 1., 0., 1., 1., 0., 1.,
       1.])
```

Step 8: Evaluating the model

In [78]:

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,predictions)
from sklearn.metrics import accuracy_score
accuracy_score(y_test,predictions)
```

Out[78]:

```
0.9833333333333333
```