## **Classification Assignment**

#### **Problem Statement or Requirement:**

A requirement from the Hospital, Management asked us to create a predictive model which will predict the Chronic Kidney Disease (CKD) based on the several parameters. The Client has provided the dataset of the same.

1.) Identify your problem statement

Need to predict CKD - Yes / No

- i.) Data table given so we can choose ML
- ii.) Both Input and Output are clearly given Supervised Learning
- iii.) Output is categorical Classification
- 2.) Tell basic info about the dataset (Total number of rows, columns)

399 rows × 25 columns

Input Contains both numerical and categorical dataset
Output contains categorical dataset

```
Index(['age', 'bp', 'sg', 'al', 'su', 'rbc', 'pc', 'pcc', 'ba', 'bgr', 'bu',
   'sc', 'sod', 'pot', 'hrmo', 'pcv', 'wc', 'rc', 'htn', 'dm', 'cad',
   'appet', 'pe', 'ane', 'classification']
```

3.) Mention the pre-processing method if you're doing any (like converting string to number – nominal data)

Using get dummies function convert the str to flt

4.) <u>Develop a good model with good evaluation metric. You can use any machine learning algorithm; you can create many models. Finally, you have to come up with final model.</u>

Created models in Logistics regression, KNN, Navie bayes and SVM

- roc auc score in KNN 0.9878048780487805
- roc auc score in Log 0.5
- roc\_auc\_score in NBs
  o Gaussian NB's 1.0
- o Complement NB's (while doing Standard scaler the values goes negative and Complement NB's getting value error (Complement NB's will not work with negative input). After removing SC got ROC -0.9356767097082734
- roc auc score in SVM 1.0

# 5.) All the research values of each algorithm should be documented.

### Gaussian NB's

```
In [29]: print("The f1_macro value for best parameter {}:".format(grid.best_params_),f1_macro)
print((cm))
        print(clf_report)
         The f1_macro value for best parameter {'var_smoothing': 1e-08}: 0.9775556904684072
          [ 3 79]]
                      precision recall f1-score support
                           0.94 1.00
                   1
                          1.00
                                  0.96
                                            0.98
                                                         82
             accuracy
                                             0.98
                                                        133
                          0.97
                                 0.98
0.98
                                           0.98
0.98
           macro avg
                                                     133
                                                        133
         weighted avg
                           0.98
In [30]: from sklearn.metrics import roc auc score
        roc_auc_score(y_test,grid.predict_proba(x_test)[:,1])
Out[30]: 1.0
```

### SVM:

```
In [25]: from sklearn.metrics import roc_auc_score
         roc_auc_score(y_test,grid.predict_proba(x_test)[:,1])
Out[25]: 1.0
In [28]: print("The f1_macro value for best parameter {}:".format(grid.best_params_),f1_macro)
         The f1_macro value for best parameter {'C': 1.0, 'class_weight': 'balanced', 'gamma': 'scale', 'kernel': 'linear'}: 0.977555690
         4684072
[[51 0]
         [ 3 79]]
                      precision recall f1-score support
                                 0.96
                          1.00
                                            0.98
                                                        82
                                                       133
                                            0.98
            accuracy
                                 0.98
0.98
                         0.97
                                             0.98
         weighted avg
                         0.98
                                            0.98
                                                        133
```

6.) Mention your final model, justify why u have chosen the same.

We can choose either SVM or Gaussian Navie Bayes. Since both are having

giving same results.