```
import pandas as pd
import io
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model selection import train test split
from sklearn.impute import KNNImputer
from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler
from google.colab import files
uploaded = files.upload()
      Choose Files No file chosen
                                        Upload widget is only available when the cell has been executed in the current browser session. Please
     rerun this cell to enable.
     Saving chronic kidnev disease.csv to chronic kidnev disease (1).csv
df = pd.read csv(io.BytesIO(uploaded['chronic kidney disease.csv']))
X=df.iloc[:,0:24]
Y=df["class"]
df.head()
```

```
pcv wbcc rbcc htn dm cad appet pe ane class
         age bp
                    sg al su rbc pc pcc ba
                                                bgr ...
                                                                                                           حادط
knn_missing_values_imputer = KNNImputer(n_neighbors=5)
df = df.replace('?', np.nan)
X = pd.DataFrame(knn_missing_values_imputer.fit_transform(X),columns = X.columns)
standard feature scaler = StandardScaler()
X = standard feature scaler.fit transform(X)
X = pd.DataFrame(X, columns=['age', 'bp', 'sg', 'al', 'su', 'rbc', 'pc', 'pcc', 'ba', 'bgr', 'bu', 'sc', 'sod', 'pot',
                                                            'hemo', 'pcv', 'wbcc', 'rbcc', 'htn', 'dm',
                                                            'cad', 'appet', 'pe', 'ane'])
X train, X test, Y train, Y test = train test split(X, Y, train size = 0.7, test size = 0.3)
import seaborn as sns
plt.figure(figsize=(12,10))
cor=X train.corr()
sns.heatmap(cor,annot=True,cmap=plt.cm.CMRmap r)
plt.show()
```

1.0

- 0.8

- 0.6

- 0.4

- 0.2

```
0.18·0.150.11 0.210.061/0.094·0.140.0440.21 0.190.130.0530.13·0.190.110.0230.01-0.4-0.380.230.190.096.07
                        0.190.17 0.150.210.170.0470.160.13 0.260.17-0.120.0750.27-0.150.048.049.230.210.07-0.180.0350.2
                          -0.46-0.3-0.22-0.33 0.3 0.19-0.330.270.210.240.010.530.350.0940.430.37 0.39 0.130.23 0.250.27
                0.280.36<mark>0.56</mark>.0.350.340.33_0.4_0.22-0.230.0340.540.370.05-0.380.440.34-0.2-0.290.480.28
           su -0.21 0.15 -0.3 0.28 1 0.13 0.19 -0.20.0920.7 0.066.068.05B.0220.18 0.17 0.0660.16 0.24 0.48 0.12 0.025 0.10 0.05
           rbc +0.0670.21-0.220.360.13 1 0.390.0680.130.15 0.240.140.10.001-0.280.140.090.170.180.160.190.120.160.09
           pc -0.0940.17-0.33<mark>0.56</mark>0.190.39 1 0.510.330.29 0.3 0.11-0.170.0820.430.220.0450.170.31-0.180.170.260.330.28
          pcc --0.10.0470.3 -0.35-0.20.06<mark>:0.51 1 0.4 -0.22-0.160.0260.13 0.010.220.11-0.140.120.14 0.120.17 0.120.0340.15</mark>
           ba -0.0440.160.19<mark>-0.34</mark>0.0920.13-0.33<mark>0.4</mark> 1-0.0520.170.0440.0630.0430.190.0930.0607.0520.0430.150.140.170.053
          bgr -0.21 0.13-0.33 0.33 0.7 0.15 0.29-0.23 0.052 1 0.09 p.0420.140.0340.260.230.05 0.190.37-0.480.190.140.0570.11
           bu -0.19 0.26 0.27 0.40.0660.24 0.3 -0.160.170.091 1 0.51 0.38 0.23 0.590.350.0950.210.420.350.230.260.310.48
           sc -0.130.17-0.210.220.0680.140.110.026.049.0420.51 1 0.720.12-0.340.24-0.130.18-0.3 -0.2 -0.2-0.150.130.22
def correlate(dataset,threshold):
   col corr=set()
   corr matrix=dataset.corr()
   for i in range(len(corr matrix.columns)):
     for j in range(i):
        if abs(corr matrix.iloc[i,j])>threshold:
           colname=corr matrix.columns[i]
           col corr.add(colname)
   return col corr
corr features=correlate(X train, 0.6)
print(len(set(corr features)))
print(corr features)
       {'rbcc', 'pcv', 'sod', 'bgr'}
X train.drop(corr features,axis=1)
X test.drop(corr features,axis=1)
```

	age	bp	sg	al	su	rbc	рс	рсс	ba	bu	sc	
291	-0.257518	0.264037	1.413392	-0.795017	-0.431146	-0.364890	-0.484322	0.360477	0.263664	-0.268332	-0.367591	0.
250	-0.668232	0.264037	1.413392	-0.795017	-0.431146	-0.364890	-0.484322	0.360477	0.263664	-0.955146	-0.332030	0.
107	0.211870	1.745312	-0.409164	-0.023156	3.388529	-0.364890	-0.484322	0.360477	0.263664	-0.086528	-0.047541	-0.
55	-0.961600	0.264037	-2.231720	1.520567	-0.431146	2.740554	-0.484322	0.360477	0.263664	0.123556	-0.043985	-0.
38	1.033299	0.264037	0.502114	1.520567	-0.431146	2.740554	-0.484322	0.360477	0.263664	0.923492	0.183606	0.
54	0.681258	0.264037	-1.320442	0.748706	1.478692	-0.364890	-0.484322	0.360477	0.263664	-0.240052	0.059143	-0.
352	-0.844253	-1.217238	0.502114	-0.795017	-0.431146	-0.364890	-0.484322	0.360477	0.263664	-0.207731	-0.349810	0.
325	0.387890	0.264037	0.502114	-0.795017	-0.431146	-0.364890	-0.484322	0.360477	0.263664	-0.147130	-0.332030	-0.
12	0.974625	-0.476600	-0.409164	1.520567	0.523773	-0.364890	-0.484322	-2.774104	0.263664	0.297279	-0.172005	0.
0	-0.198844	0.264037	0.502114	-0.023156	-0.431146	-0.364890	-0.484322	0.360477	0.263664	-0.429935	-0.332030	-0.

120 rows × 20 columns

```
from sklearn.svm import SVC
from sklearn.metrics import classification_report

linear_svm=SVC(C=1.0,kernel='linear',random_state=0)
linear_svm=linear_svm.fit(X_train,Y_train)

print(linear_svm.score(X_train,Y_train))
print(linear_svm.score(X_test,Y_test))

1.0
0.975

predicted =linear_svm.predict(X_test)
report = classification_report(Y_test, predicted)
```

print(report)

	precision	recall	f1-score	support
ckd notckd	1.00 0.93	0.96 1.00	0.98 0.96	81 39
accuracy macro avg weighted avg	0.96 0.98	0.98 0.97	0.97 0.97 0.98	120 120 120