1.INTRODUCTION

Early Prediction for Chronic Kidney Disease Detection: A Progressive Approach to Health Management Chronic Kidney Disease (CKD) is a major medical problem and can be cured if treated in the early stages. Usually, people are not aware that medical tests we take for different purposes could contain valuable information concerning kidney diseases. Consequently, attributes of various medical tests are investigated to distinguish which attributes may contain helpful information about the disease. The information says that it helps us to measure the severity of the problem, the predicted survival of the patient after the illness, the pattern of the disease and work for curing the disease. In todays world as we know most of the people are facing so many disease and as this can be cured if we treat people in early stages this project can use a pretrained model to predict the Chronic Kidney Disease which can help in treatments of peoples who are suffer from this disease.

1.1 Overview

Your kidneys, each just the size of a computer mouse, filter all the blood in your body every 30 minutes. They work hard to remove wastes, toxins, and excess fluid. They also help control blood pressure, stimulate production of red blood cells, keep your bones healthy, and regulate blood chemicals that are essential to life.Kidneys that function properly are critical for maintaining good health, however, more than one in seven American adults are estimated to have chronic kidney disease (CKD).

1.2 PURPOSE

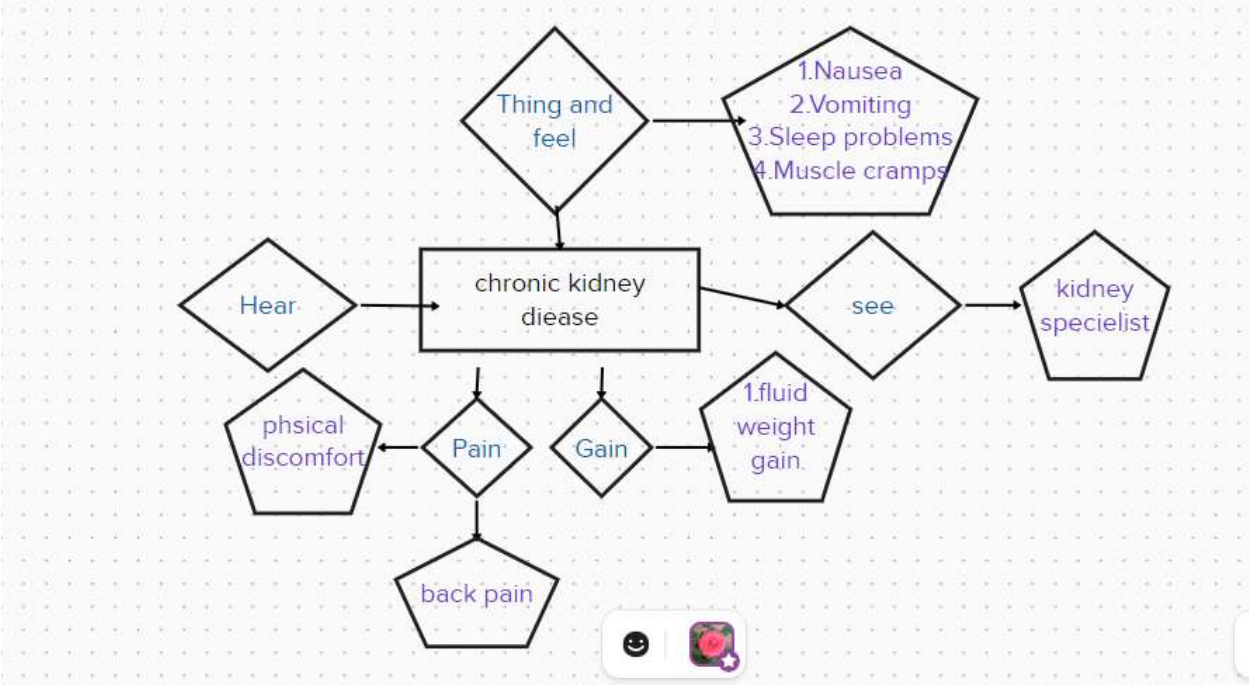
Preventing chronic kidney disease (CKD) and its complications is possible by managing risk factors and treating the disease to slow its progression and reduce the risk of complications. To keep healthy kidneys, it is important to control those risk factors for CKD that can be modified.

If you are at risk, get tested for CKD regularly. Ask your doctor to test your blood or urine. Find it early. Treat it early .If you have diabetes, get tested yearly.

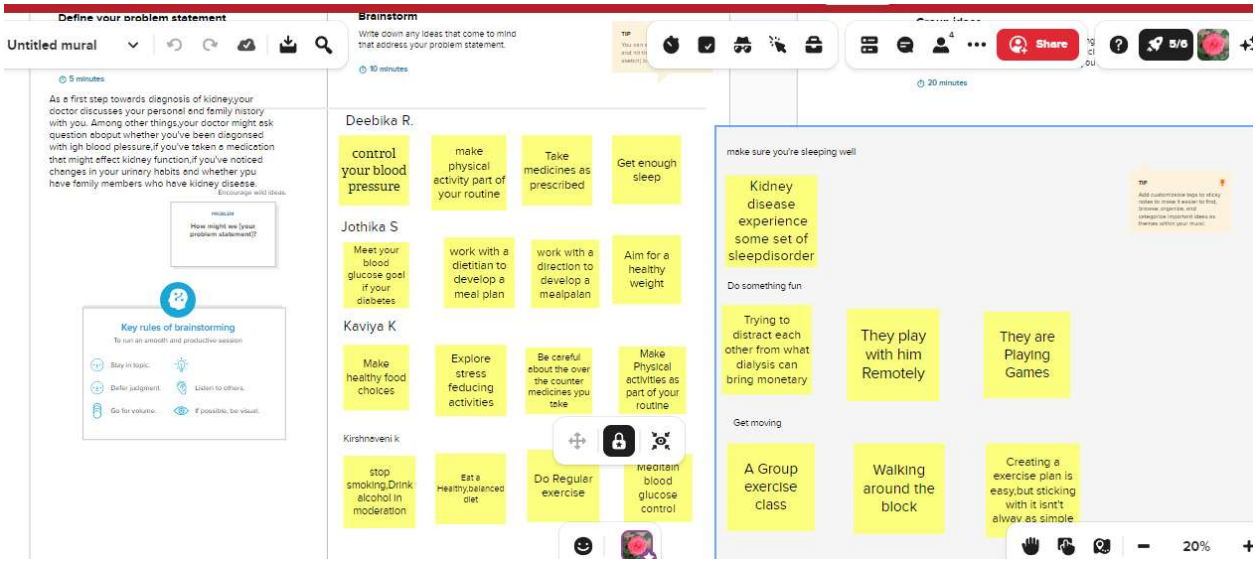
Lose weight if you are overweight Get active Physical activity helps control blood sugar levels, Quit smoking. Getting a checkup? Make sure to get your kidneys checked too. Take medications as directed. Keep your blood pressure below 140/90, or ask your doctor what the best blood pressure target is for you.

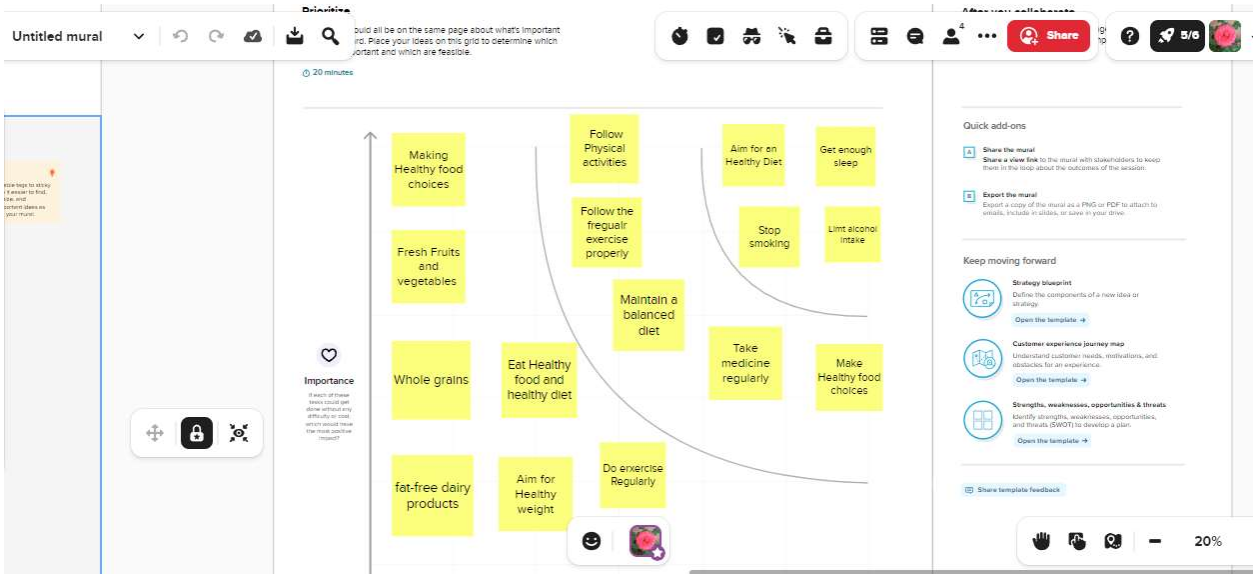
2. PROBLEM DEFINITION &DESIGN THINKING

2.1 EMPATHY MAP

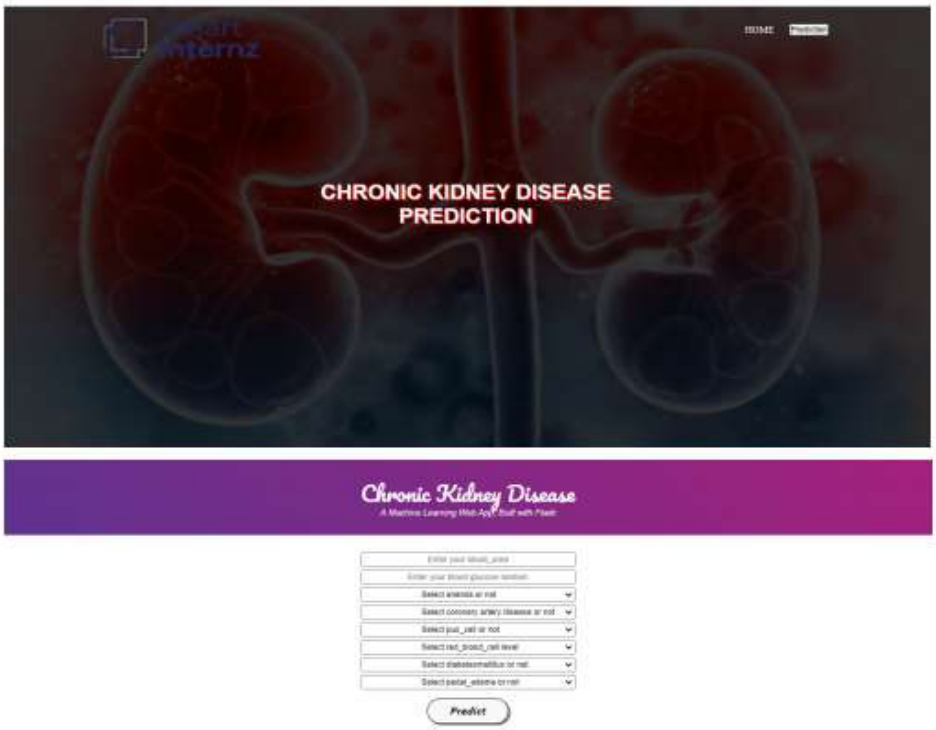


2.2 IDEAS & BRAINSTORMING MAP





3.RESULT



Input - Now, the user will give inputs to get the predicted result after clicking onto the submit button.


Chronic Kidney Disease
Predict


Chronic Kidney Disease
Prediction: Great! You DON'T have Chronic Kidney Disease
IT'S TIME TO PARTYYYYYY !

4.ADVANTAGES & DISADVANTAGE

The early detection of CKD allows patients to receive timely treatment, slowing the disease's progression. Due to its rapid recognition performance and accuracy, machine learning models can effectively assist physicians in achieving this goal.

The early detection of CKD allows patients to receive timely treatment, slowing the disease's progression. Due to its rapid recognition performance and accuracy, machine learning models can effectively assist physicians in achieving this goal.

Advantages of preventing chronic kidney disease

- ❖ Make healthy food choices.
- ❖ Make physical activity part of your routine.
- ❖ Aim for a healthy weight.
- ❖ Get enough sleep.
- ❖ Stop smoking.
- ❖ Limit alcohol intake link.
- ❖ Explore stress-reducing activities.
- ❖ Manage diabetes, high blood pressure, and heart disease.

Disadvantages

Factors that can increase your risk of chronic kidney disease include:

- Diabetes
- High blood pressure
- Heart (cardiovascular) disease
- Smoking
- Obesity
- Being Black, Native American or Asian American
- Family history of kidney disease

- Abnormal kidney structure
- Older age
- Frequent use of medications that can damage the kidneys

Chronic kidney disease can affect almost every part of your body.

- Fluid retention, which could lead to swelling in your arms and legs, high blood pressure, or fluid in your lungs (pulmonary edema)
- A sudden rise in potassium levels in your blood (hyperkalemia), which could impair your heart's function and can be life-threatening
- Anemia
- Heart disease
- Weak bones and an increased risk of bone fractures
- Decreased sex drive, erectile dysfunction or reduced fertility
- Damage to your central nervous system, which can cause difficulty concentrating, personality changes or seizures
- Decreased immune response, which makes you more vulnerable to infection
- Pericarditis, an inflammation of the saclike membrane that envelops your heart (pericardium)
- Pregnancy complications that carry risks for the mother and the developing fetus
- Irreversible damage to your kidneys (end-stage kidney disease), eventually requiring either dialysis or a kidney transplant for survival

5. APPLICATIONS

As a first step toward diagnosis of kidney disease, your doctor discusses your personal and family history with you. Among other things, your doctor might ask questions about whether you've been diagnosed with high blood pressure, if you've taken a medication that might affect kidney function, if you've noticed changes in your urinary habits and whether you have family members who have kidney disease.

AI and machine learning hold promise for more accurately predicting, preventing, and treating a variety of health problems, including chronic kidney disease (CKD), which **affects** 37 million Americans. CKD involves the gradual loss of function in the kidneys over time. The disease can lead to high blood pressure, low blood count, weak bones, and it can increase the risk for heart disease.

Patients with early stages of CKD often have no symptoms, but the disease can progress to end-stage kidney failure, which is deadly without routine dialysis or a kidney transplant. Millions of Americans are at higher risk for CKD, including people who have diabetes, high blood pressure, and family history of kidney failure.

6.CONCLUSION

A reliable tool to assess public knowledge and awareness about CKD was developed and validated. The overall knowledge was good, however, important gaps in CKD awareness were detected in some areas and subpopulations. Therefore, public health stakeholders need to implement targeted CKD educational activities to minimize the disease burden.

The current study constructed and validated the CKD knowledge scale as a reliable tool to assess public knowledge about CKD, and promote public health awareness and research. It also assessed the level of knowledge on CKD and determined predictors of better knowledge among the Lebanese population.

We found a good overall knowledge about CKD, with a significant better knowledge associated with age, area of residence, level of education, occupation, and a recent assessment of kidney function.

7.FUTURE SCOPE

You can protect your kidneys by preventing or managing health conditions that cause kidney damage, such as diabetes and high blood pressure. The steps described below may help keep your whole body healthy, including your kidneys.

During your next medical visit, you may want to ask your health care provider about your kidney health. Early kidney disease may not have any symptoms, so getting tested may be the only way to know your kidneys are healthy. Your health care provider will help decide how often you should be tested.

See a provider right away if you develop a urinary tract infection (UTI), which can cause kidney damage if left untreated.

8.APPENDIX

```
import pandas as pd

import numpy as np

from collections import Counter as c

import matplotlib.pyplot as plt

import seaborn as sns

import missingno as msno

from sklearn.metrics import accuracy_score, confusion_matrix

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder

from sklearn.linear_model import LogisticRegression

import pickle


data=pd.read_csv("/content/kidney_disease.csv")#loading the csv data

data.head()#return you the first 5 rows values
```

```
data.columns


data.columns=['age','blood_pressure','specific_gravity','albumin',

'sugar','red_blood_cells','pus_cell','pus_cell_clumps','bacteria',

        'blood glucose random','blood_urea','serum_creatinine','sodium',

        'pottassium','hemoglobin','packed_cell_volume',

'white_blood_cell_count','red_blood_cell_count','hypertension',

        'diabetesmellitus','coronary_artery_disease','appetite',

        'pedal_edema','anemia','claas']

data.columns


data.info()

data.isnull().any()
```

```
data['blood glucose random'].fillna(data['blood glucose random'].mean(),inplace=True)

data['blood_pressure'].fillna(data['blood_pressure'].mean(),inplace=True)

data['blood_urea'].fillna(data['blood_urea'].mean(),inplace=True)

data['hemoglobin'].fillna(data['hemoglobin'].mean(),inplace=True)

data['packed_cell_volume'].fillna(data['packed_cell_volume'].mean(),inplace=True)

data['potassium'].fillna(data['potassium'].mean(),implace=True)

data['red_blood_cell_count'].fillna(data['red_blood_cell_count'].mean(),inplace=True)

data['serum_creatinine'].fillna(data['serum_creatinine'].mean(),inplace=True)

data['sodium'].fillna(data['sodium'].mean(),inplace=True)

data['white_blood_cell_count'].fillna(data['white_blood_cell_count'].mean(),inplace=True)
```

data['age'].fillna(data['age'].mode()[0],inplace=True)

data['hypertension'].fillna(data['hypertension'].mode()[0],inplace=True)

data['pus_cell_clumps'].fillna(data['pus_cell_clumps'].mode()[0],inplace=True)

data['apptite'].fillna(data['appetite'].mode()[0],inplace=True)

data['albumin'].fillna(data['albumin'].mode()[0],inplace=True)

data['pus_cell'].fillna(data['pus_cell'].mode()[0],inplace=True)

data['red_blood_cells'].fillna(data['red_blood_cells'].mode()[0],inplace=True)

data['coronary_artery_disease'].fillna(data['coronary_artery_disease'].mode()[0],inplace=True)

data['bacteria'].fillna(data['bacteria'].mode()[0],inplace=True)

data['anemia'].fillna(data['anemia'].mode()[0],inplace=True)

data['sugar'].fillna(data['sugar'].mode()[0],inplace=True)

```
data['diabetesmellitus'].fillna(data['diabetesmellitus'].mode()[0],inplace=True)

data['pedal_edema'].fillna(data['pedal_edema'].mode()[0],inplace=True)

data['specific_gravity'].fillna(data['specific_gravity'].mode()[0],inplace=True)


catcols=set(data.dtypes[data.dtypes=='O'].index.values)

print(catcols)


for i in catcols:

    print("Columns :",i)

    print(c(data[i]))

    print('*'*120+'\n')


catcols.remove('red_blood_cell_count')#remove is used for removing a particular column
```

```
catcols.remove('packed_cell_volume')

catcols.remove('white_blood_cell_count')

print(catcols)


catcols=['anemia','pedel_edema','appetite','bacteria','class','coronary_artery_diease','diabetesmellit','hypertension','pus_cell_clupms','red_blood_cells']


from sklearn.preprocessing import LabelEncoder

for i in catcols:

    print("LABLE ENCODING OF:",i)

    LEi = LabelEncoder()

    print(c(data[i]))

    data[i] = LEi.fit_transform(data[i])

    print(c(data[i]))

    print("*"*100)
```

```
contcols=set(data.dtypes[data.dtypes!='0'].index.values)

print(contcols)


for i in contcols:

    print("Continous Columns :",i)

    print(c(data[i]))

    print('*'*120+'\n')


contcols.remove('specific_gravity')

contcols.remove('albumin')

contcols.remove('sugar')

print(contcols)


contcols.add('red_blood_cell_count')

contcols.add('packed_cell_volume')
```

```
contcols.add('white_blood_cell_count')

print(contcols)


contcols.add('specific_gravity')

contcols.add('albumin')

contcols.add('sugar')

print(catcols)


data['coronary_artery_disease']=data.cornary_artery_diease.replac
e('\tno',no)

c(data['cornary_artery_disease'])


data['diabetesmellitus']=data.diabetesmellitus.replace{'\tno':'no','\ty
pes':'yes':}

c(data['diabetesmellitus'])
```

```
data.describe()


sns.displot(data.age)


import matplotlib.pyplot as plt

fig=plt.figure(figsize=(5,5))

plt.scatter(data['age'],data['blood_pressure'],color='blue')

plt.xlabel('age')

plt.ylabel('blood pressure')

plt.title("age vs blood scatter plot")


plt.figure(figsize=(20,15),facecolor='white')

plotnumber = 1


for column in cotncols:

    if plotnumber<=11 :
```

```
    ax=plt.subplot(3,4,plotnumber)
    plt.scatter(data['age'],data[column])
    plt.xlabel(column,fontsize=20)

    plotnumber+=1
plt.show()

f,ax=plt.subplot(figsize=(18,10))
sns.heatmap(data.corr(),annot=True,fmt="2f",ax=ax,linewidths=0.5
,linecolor="orange")
plt.xticks(rotation=45)
plt.yticks(rotation=45)
plt.show()

sns.countplot(data['class'])
```

```
from sklearn.preprocessing import StandardScaler

sc=StandarScaler()

x_bal=sc.fit_transform(x)



selcols=['red_blood_cells'.'pus_cell','blood glucose random','blood_urea','pedal_edema','anemia','diabetesmellitus','cornary_artery_diease']

x=pd.DataFrame(data,columns=selcols)

y=pd.DataFrame(data,columns=['class'])

print(x.shape)

print(y.shape)


from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=2)
```

```
import tensorflow

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense


classification=Sequential()

classification.add(Dense(30,activation='relu'))

classification.add(Dense(128,activation='relu'))

classification.add(Dense(64,activation='relu'))

classification.add(Dense(32,activation='relu'))

classification.add(Dense(1,activation='sigmoid'))


classification.compile(optimizer='adam',loss='binary_crossentropy',
metrics=['accuracy'])


classification.fit(x_train,y_train,batch_size=10,validation_split=0.2,e
pochs=100)
```

```
from sklearn.ensemble import RandomForestClassifier

rfc=RandomForestClassifier(n_estimators=10,criterion='entropy')


rfc.fit(x_train,y_train)


y_predict=rfc.predict(x_test)


y_predict_train=rfc.predict(x_train)


from sklearn.tree import DecisionTreeClassifier

dtc=DecisionTreeClassifier(max_depth=4,splitter='best',criterion='e
ntropy')


dtc.fit(x_train,y_train)
```

```
y_predict= dtc.predict(x_test)

y_predict


y_predict_train=dtc.predict(x_train)


from sklearn.linear_model import LogisticRegression

lgr = LogisticRegression()

lgr.fit(x_train,y_train)


from sklearn.metrics import accuracy_score,classification_report

y_predict = lgr.predict(x_test)


y_pred = lgr.predict([[1,1,121.000000,36.0,0,0,1,0]])

print(y_pred)
```

```
(y_pred)


y_pred = dtc.predict([[1,1,121.000000,36.0,0,0,1,0]])

print(y_pred)

(y_pred)


y_pred = rfc.predict([[1,1,121.000000,36.0,0,0,1,0]])

print(y_pred)

(y_pred)


classification.save("ckd.h5")


y_pred = classification.predict(x_test)


y_pred
```

```
y_pred=(y_pred>0.5)

y_pred


def predict_exit(sample_value):

  sample_value = np.array(sample_value)

  sample_value = sample_value.reshape(1,-1)

  sample_value=sc.transform(sample_value)

  return classifier.predict(sample_value)


test=classification.predict([[1,1,121,000000,36.0,0,0,1,0]])

if test==1;

print('prediction:high chance of CKDI')

else:

  print('prediction: low chance of CKD.')


from sklearn import model_selection
```

```
  dfs = []
 models =[
        ('LogReg, LogisticRegistaion()'),
        ('RF',RandomForestClassifier()),
        ('DecisionTree',DesitionTreeCladssifier()),
       ]
 result= []
       names=[]

scoring=['accuracy','precision_weighted','recall_weighted',f1_weighted','roc_auc']
       target_names=['NO CKD','CKD']
       for name,mode1 in models:
        kflod=mode1_selection.KFold(n_splits=5,
suffle=True,random_atate=90210)
```

```
cv_results=model_selection.cross_validate(model,x_train,y_train,cv
=kflod,scoring=scoring)

        clf = model.fit(x_train,y_train)

        y_pred==clf.predict(x_test)

        print(name)


print(classification_report(y_test,y_pred,target_names=target_nam
es))

        results.append(name)

        this_df=pd.DateFrame(cv_results)

        this _df=pd.DataFrame(cv_results)

        this _df['model']=name

        dfs.append(this_df)

    final=pd.concat(dfs,ignore_index=True)

    return final
```

```
from sklearn.metrics import confusion_matrix

cm=confusion_matrix(y_test,y_predict)

cm



plt.figure(figsize=(8,6))

sns.hestmap(cm,cmap'Blues',annot=True, xticklabels=['no ckd','ckd'],yticklabels=['no ckd','ckd'])

plt.xlabel('predicted values')

plt.ylabel('Actual values')

plt.titel('Confusion matrix for logistic Regression model')

plt.show()


from sklearn.metrics import confusion_matrix

cm=confusion_matrix(y_test,y_predict)

cm
```

```
plt.figure(figsize=(8,6))

sns.hestmap(cm,cmap'Blues',annot=True,          xticklabels=['no
ckd','ckd'],yticklabels=['no ckd','ckd'])

plt.xlabel('predicted values')

plt.ylabel('Actual values')

plt.titel('Confusion matrix for RandomForestClassifier')

plt.show()


from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test,y_pred)

cm


plt.figure(figsize=(8,6))

sns.heatmap(cm,cmap'Blues',annot=True,          xticklabels=['no
ckd','ckd'],yticklabels=['no ckd','ckd'])
```

```
plt.xlabel('predicted values')

plt.ylabel('Actual values')

plt.titel('Confusion matrix for DecisionTreeClassifier')

plt.show()


bootstraps=[]

for model in list(set(final.model.values)):

  model_df=final.loc[final.model1==model]

  bootstrap=model_df.sample(n=30,replace=true)

  bootstraps.append(bootstrap)

  bootstrap_df=pd.concat(bootstrap,ignore_index=true)


results_long=pd.melt(bootstrap_df,id_vars=['metrics'].isin(time_me
trics)])))

   time_metrics =['fit_time','score_time']

results_long_nofit=results_long.loc[~results_long['metrics'].isin(tim
e_metrics)]
```

```
results_long_nofit=results_long_nofit.sort_values(by='values')

results_long_nofit=results_long.loc[results_long['metrics'].isin(time_
metrics)]

results_long_nofit=results_long_fit.sort_values(by='values')


import matplotlib.pyplot as plt

import seaborn as sns

plt.figure(figsize=(20,12))

sns.set(font_scale=2.5)

g=sns.boxplot(x="model",y="values",hue="metrics",data=results_l
ong_nofit,palette="srt3")
```