**Chronic Kidney Disease**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

df=pd.read\_csv(r"/content/kidney\_disease.csv")

df.head()

df.describe()

import pandas\_profiling as pp

profile = pp.ProfileReport(df, title="Chronic Kidney Disease Profile", html={"style": {"full\_width": True}}, sort=None)

profile

df.rename(columns = {

'age' : 'age',

'bp' : 'blood pressure',

'sg' : 'specific gravity',

'al' : 'albumin',

'su' : 'sugar',

'rbc' : 'red blood cells',

'pc' : 'pus cell',

'pcc' : 'pus cell clumps',

'ba' : 'bacteria',

'bgr' : 'blood glucose random',

'bu' : 'blood urea',

'sc' : 'serum creatinine',

'sod' : 'sodium',

'pot' : 'potassium',

'hemo' : 'hemoglobin',

'pcv' : 'packed cell volume',

'wc' : 'white blood cell count',

'rc' : 'red blood cell count',

'htn' : 'hypertension',

'dm' : 'diabetes mellitus',

'cad' : 'coronary artery disease',

'appet' : 'appetite',

'pe' : 'pedal edema',

'ane' : 'anemia',

'class' : 'class'}, inplace = True)

df.head()

df.info()

df.isnull()

df.isnull().sum()

df.shape[0]

df.isnull().sum()/df.shape[0]

((df.isnull().sum()/df.shape[0])\*100).sort\_values(ascending=False)

df.info()

df['red blood cell count']=pd.to\_numeric(df['red blood cell count'],errors='coerce')

df['white blood cell count']=pd.to\_numeric(df['white blood cell count'],errors='coerce')

df.describe()

df.describe(include="all").T

print(df.apply(lambda col: col.unique()))

df['packed cell volume'] = df['packed cell volume'].replace(to\_replace = {'\t?': np.nan,'\t43':'43'})

df['diabetes mellitus'] = df['diabetes mellitus'].replace(to\_replace = {'\tno':'no','\tyes':'yes',' yes':'yes'})

df['coronary artery disease'] = df['coronary artery disease'].replace(to\_replace = '\tno', value='no')

df['classification'] =df['classification'].replace(to\_replace = 'ckd\t', value = 'ckd')

df['packed cell volume'] = pd.to\_numeric(df['packed cell volume'], errors='coerce')

prda col: col.unique()))int(df.apply(lamb

df\_num=[feature for feature in df.columns if df[feature].dtypes=='float64']

print('total numerical column :',len(df\_num))

print(df\_num)

df\_cat=[feature for feature in df.columns if df[feature].dtypes=='O']

print('total categorical column :',len(df\_cat))

print(df\_cat)

df[df\_num].describe(include="all").T

df[df\_cat].describe(include="all").T

import seaborn as sns

import plotly.express as px

sns.set(rc={"figure.figsize":(15, 8)})

for i in df.columns:

    sns.countplot(x=df[i],data=df)

    plt.xlabel(i)

    plt.ylabel("Count")

    plt.title("target Class")

    plt.show()

    print("\n")

corr\_df = df.corr()

f,ax=plt.subplots(figsize=(15,15))

sns.heatmap(corr\_df,annot=True,fmt=".2f",ax=ax,linewidths=0.5,linecolor="yellow")

print(df.corr())

for i in df.columns:

    if df[i].isna().sum() > 0 :

        if df[i].dtype == 'float64':

            df[i].fillna(df[i].median(), inplace=True)

        else:

            df[i].fillna(df[i].mode()[0], inplace=True)

import warnings

warnings.filterwarnings('ignore')

def violin(col):

    fig = px.violin(df, y=col, x="classification", color="classification", box=True, points="all", hover\_data=df.columns)

    return fig.show()

def kde\_plot(feature):

    grid = sns.FacetGrid(df, hue="classification", aspect = 2)

    grid.map(sns.kdeplot, feature)

    grid.add\_legend()

fig = px.scatter(df, x="red blood cell count", y="hemoglobin", color="classification")

fig.show()

kde\_plot('red blood cell count')

for i in df.columns:

    fig = px.scatter(df, x=i, y="hemoglobin", color="classification")

    fig.show()

kde\_plot('red blood cell count')

for i in df.columns:

    fig = px.scatter(df, x=i, y="hemoglobin", color="classification")

    fig.show()

violin('age')

sns.pairplot(df, hue="classification",corner=False)