4- Heart.csv- Data cleaning, processing,…

1. # import pandas library

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

import matplotlib.pyplot as plt ……………..no o/p

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

from sklearn.metrics import accuracy\_score,confusion\_matrix

from sklearn.linear\_model import LogisticRegression

import seaborn as sns

import matplotlib.pyplot as plt

1. # Reading csv file

df = pd.read\_csv("Heart.csv")

df.head()

Data Cleaning

1. df = df.drop\_duplicates() ………..no o/p
2. # Count ,min,max ,etc of each column

df.describe()

1. # Information about each column data

df.info()

1. #Finding null values in each column

df.isna().sum()

Data Integration

1. df.head()
2. df.fbs.unique()
3. import pandas as pd

# Create subsets

subSet1 = df[['age', 'sex']].copy()

subSet2 = df[['cp', 'thalachh']].copy()

# Add dummy key column for cross join

subSet1['key'] = 1

subSet2['key'] = 1

# Merge on dummy key to simulate cross join

merged\_df = pd.merge(subSet1, subSet2, on='key').drop('key', axis=1)

# Show result

print(merged\_df.head())

1. import pandas as pd

print(pd.\_\_version\_\_)

ERROR CORRECTING

1. df.columns
2. def remove\_outliers(column): ………no o/p

Q1 = column.quantile(0.25)

Q3 = column.quantile(0.75)

IQR = Q3 - Q1

threshold = 1.5 \* IQR

outlier\_mask = (column < Q1 - threshold) | (column > Q3 + threshold)

return column[~outlier\_mask]

1. # Remove outliers for each column using a loop ……no o/p

col\_name = ['cp','thalachh','exng','oldpeak','slp','caa']

for col in col\_name:

df[col] = remove\_outliers(df[col])

1. plt.figure(figsize=(10, 6)) # Adjust the figure size if needed

for col in col\_name:

sns.boxplot(data=df[col])

plt.title(col)

plt.show()

1. df = df.dropna()
2. df.isna().sum()
3. df = df.drop('fbs',axis=1)
4. # Compute correlations between features and target

correlations = df.corr()['output'].drop('output')

# Print correlations

print("Correlation with the Target:")

print(correlations)

print()

# Plot correlation heatmap

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

sns.heatmap(df.corr(), annot=True, cmap='coolwarm')

plt.title('Correlation Heatmap')

plt.show()

DATA TRANSFORMATION

1. from sklearn.preprocessing import StandardScaler
2. scaler = StandardScaler()
3. x\_train\_scaled = scaler.fit\_transform(x\_train)

x\_test\_scaled = scaler.transform(x\_test)

DATA MODEL BULIDING

1. y\_train= np.array(y\_train).reshape(-1, 1)

y\_test= np.array(y\_test).reshape(-1, 1)

1. y\_train.shape
2. model = LogisticRegression()

model.fit(x\_train\_scaled, y\_train)

# Make predictions on the test set

y\_pred = model.predict(x\_test\_scaled)

# Evaluate the model's accuracy

accuracy = accuracy\_score(y\_test, y\_pred)

print("Accuracy:", accuracy)

1. #Classification model using Decision Tree

from sklearn.tree import DecisionTreeClassifier

tc=DecisionTreeClassifier(criterion='entropy')

tc.fit(x\_train\_scaled,y\_train)

y\_pred=tc.predict(x\_test\_scaled)

print("Training Accuracy Score :",accuracy\_score(y\_pred,y\_test))

print("Training Confusion Matrix:",confusion\_matrix(y\_pred,y\_test))