DWDM ASSIGNMENT-7

21BCE7371
RADHA KRISHNA GARG

CODE

```
import numpy as np
import matplotlib.pyplot as plt

# Generate 100 random numbers for the salary attribute (100K-1000K)
randomlist = np.random.randint(100000, 1000000, size=100)

# Plot equal-width histogram (10 bins)
n, bins, patches = plt.hist(randomlist, bins=10, edgecolor='black')
plt.title('Equal-Width Histogram')
plt.show()

# Function to compute equal-frequency bins
def equalObs(x, nbin):
nlen = len(x)
return np.interp(np.linspace(0, nlen, nbin + 1), np.arange(nlen),
np.sort(x))

# Plot equal-frequency histogram (20 values)
n, bins, patches = plt.hist(randomlist, equalObs(randomlist, 20),
edgecolor='black')
plt.title('Equal-Frequency Histogram')
plt.show()

# Sampling techniques
from random import choices, sample

strata = ["low", "medium", "high"]
randomlist2 = np.random.randint(100, 1000, size=100) # Assuming 100 values
for demonstration

# Simple random sampling without replacement
srs_wr = choices(randomlist2, k=5)
print("Simple Random Sampling with Replacement:", srs_wr)

# Simple random sampling with replacement
srs_wo_wr = sample(randomlist2, 5)
print("Simple Random Sampling without Replacement:", srs_wo_wr)

# Stratified sampling
stratum = np.random.choice(strata, size=100, p=[0.3, 0.4, 0.3]) # Assuming
stratification for demonstration
stratified_sample = [sample(randomlist2[stratum == s], 1)[0] for s in
strata]
print("Stratified Sampling:", stratified_sample)
```

```
import pandas as pd
from sklearn.impute import SimpleImputer
import numpy as np
import matplotlib.pyplot as plt
# a.
df = pd.read_csv('crx.data', header=None)
df.columns = ['A'+str(i) for i in range(1, 17)] # Assigning column names A1
to A16
df = df.replace('?', np.nan)
df['A2'] = df['A2'].astype(float)
df['A14'] = df['A14'].astype(float)
df['A16'] = df['A16'].map({'+': 1, '-': 0})
df.to_csv('Transformed_crx.csv', index=False)

# b.
df1 = pd.read_csv('Transformed_crx.csv')
print("Percentage of missing values for each variable:\n",
(df1.isnull().sum() / len(df1)).sort_values())

df2 = df1.dropna()
print("Original dataset size:", df1.shape)
print("Complete case dataset size:", df2.shape)

# c.
num_vars = ['A2', 'A3', 'A8', 'A11', 'A15']
imputer_mean = SimpleImputer(strategy='mean')
imputer_median = SimpleImputer(strategy='median')

df_mean = pd.DataFrame(imputer_mean.fit_transform(df1[num_vars]),
columns=num_vars)
df_median = pd.DataFrame(imputer_median.fit_transform(df1[num_vars]),
columns=num_vars)

# d.
cat_vars = ['A4', 'A5', 'A6', 'A7']
imputer_mode = SimpleImputer(strategy='most_frequent')

df_mode = pd.DataFrame(imputer_mode.fit_transform(df1[cat_vars]),
columns=cat_vars)

# e.
from sklearn.linear_model import LinearRegression

for var in num_vars:
df_missing = df1[df1[var].isnull()]
df_complete = df1.dropna(subset=[var])

lr = LinearRegression()
```

```
lr.fit(df_complete[['A3', 'A8', 'A11', 'A15']], df_complete[var])
predicted_values = lr.predict(df_missing[['A3', 'A8', 'A11', 'A15']])
df1.loc[df1[var].isnull(), var] = predicted_values

print("Attributes 'A3', 'A8', 'A11', 'A15' have been imputed using linear
regression.")
print(df1.isnull().sum())

import pandas as pd
from sklearn.impute import SimpleImputer
import numpy as np
import matplotlib.pyplot as plt
# a.
df = pd.read_csv('crx.data', header=None)
df.columns = ['A'+str(i) for i in range(1, 17)] # Assigning column names A1
to A16
df = df.replace('?', np.nan)
df['A2'] = df['A2'].astype(float)
df['A14'] = df['A14'].astype(float)
df['A16'] = df['A16'].map({'+': 1, '-': 0})
df.to_csv('Transformed_crx.csv', index=False)

# b.
df1 = pd.read_csv('Transformed_crx.csv')
print("Percentage of missing values for each variable:\n",
(df1.isnull().sum() / len(df1)).sort_values())

df2 = df1.dropna()
print("Original dataset size:", df1.shape)
print("Complete case dataset size:", df2.shape)

# c.
num_vars = ['A2', 'A3', 'A8', 'A11', 'A15']
from sklearn.linear_model import LinearRegression

# Check if there are missing values for the variables
for var in num_vars:
df_missing = df1[df1[var].isnull()]
if not df_missing.empty: # Proceed only if there are missing values for the
variable
df_complete = df1.dropna(subset=[var])

if not df_complete.empty: # Proceed only if there are complete samples for
the variable
lr = LinearRegression()
lr.fit(df_complete[['A3', 'A8', 'A11', 'A15']], df_complete[var])
predicted_values = lr.predict(df_missing[['A3', 'A8', 'A11', 'A15']])
df1.loc[df1[var].isnull(), var] = predicted_values
else:
print(f"No complete samples available for {var}. Skipping imputation.")
else:
```

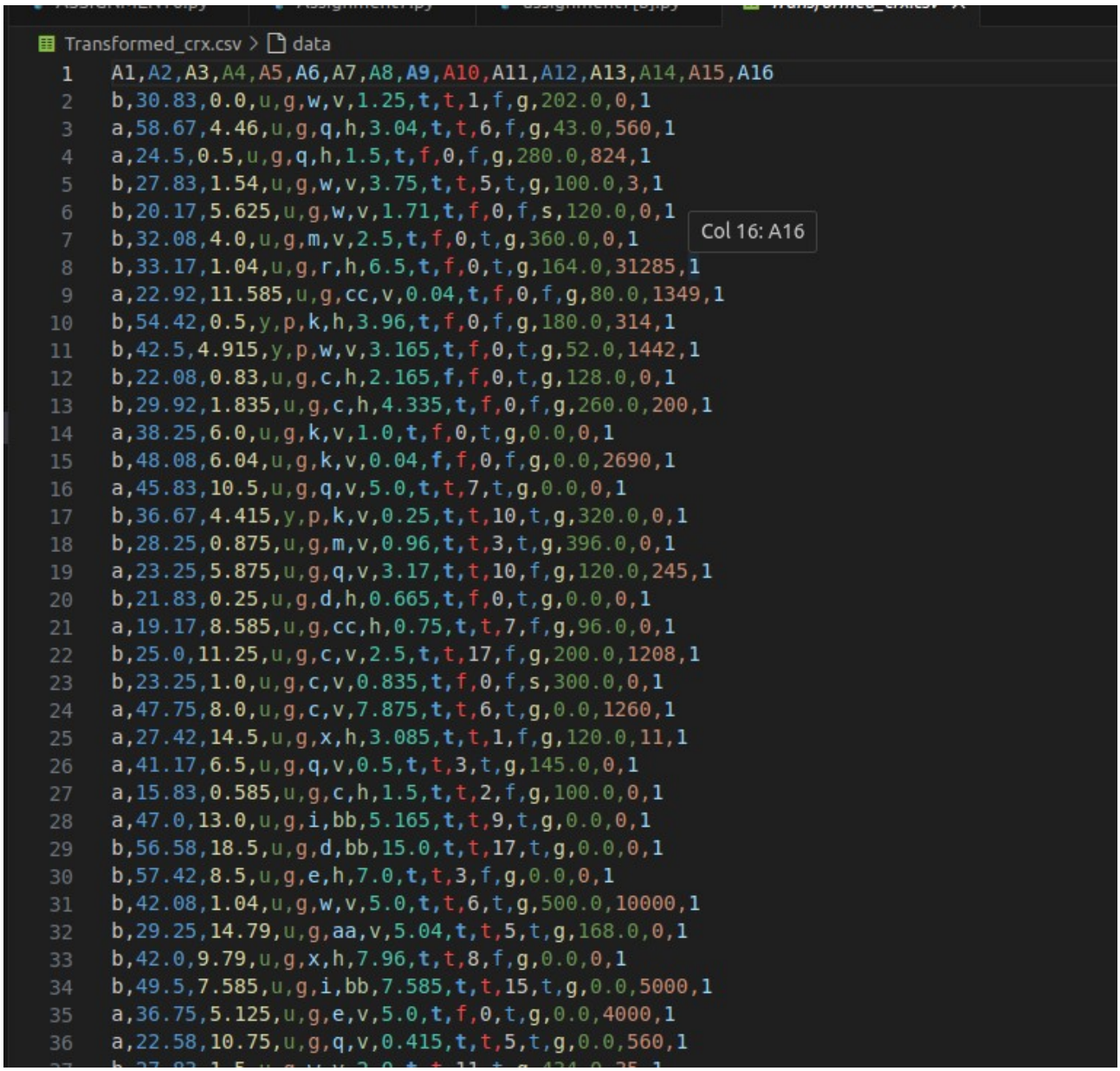
```
print(f"No missing values found for {var}. Skipping imputation.")

print("Attributes 'A3', 'A8', 'A11', 'A15' have been imputed using linear regression.")
print(df1.isnull().sum())
```

OUTPUT

```
botk@botk:/media/botk/OS/Users/krish/Documents/RK/PROJECTS_RK/DWDM LAB$  cd /media/botk/OS/User
/.vscode/extensions/ms-python.debugpy-2024.2.0-linux-x64/bundled/libs/debugpy/adapter/../../deb
LAB/assignment7\[b\].py
Percentage of missing values for each variable:
 A3     0.000000
A8     0.000000
A9     0.000000
A10    0.000000
A11    0.000000
A12    0.000000
A13    0.000000
A15    0.000000
A16    0.000000
A4     0.008696
A5     0.008696
A6     0.013043
A7     0.013043
A1     0.017391
A2     0.017391
A14    0.018841
dtype: float64
Original dataset size: (690, 16)
Complete case dataset size: (653, 16)
No missing values found for A3. Skipping imputation.
No missing values found for A8. Skipping imputation.
No missing values found for A11. Skipping imputation.
No missing values found for A15. Skipping imputation.
Attributes 'A3', 'A8', 'A11', 'A15' have been imputed using linear regression.
A1     12
A2      0
A3      0
A4      6
A5      6
A6      9
A7      9
A8      0
A9      0
A10     0
A11     0
A12     0
A13     0
A14    13
A15     0
A16     0
dtype: int64
botk@botk:/media/botk/OS/Users/krish/Documents/RK/PROJECTS_RK/DWDM LAB$
```

TRANSFORMED CSV


Transformed_crx.csv > data
A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16
b,30.83,0.0,u,g,w,v,1.25,t,t,1,f,g,202.0,0,1
a,58.67,4.46,u,g,q,h,3.04,t,t,6,f,g,43.0,560,1
a,24.5,0.5,u,g,q,h,1.5,t,f,0,f,g,280.0,824,1
b,27.83,1.54,u,g,w,v,3.75,t,t,5,t,g,100.0,3,1
b,20.17,5.625,u,g,w,v,1.71,t,f,0,f,s,120.0,0,1
b,32.08,4.0,u,g,m,v,2.5,t,f,0,t,g,360.0,0,1
Col 16: A16
b,33.17,1.04,u,g,r,h,6.5,t,f,0,t,g,164.0,31285,1
a,22.92,11.585,u,g,cc,v,0.04,t,f,0,f,g,80.0,1349,1
b,54.42,0.5,y,p,k,h,3.96,t,f,0,f,g,180.0,314,1
b,42.5,4.915,y,p,w,v,3.165,t,f,0,t,g,52.0,1442,1
b,22.08,0.83,u,g,c,h,2.165,f,f,0,t,g,128.0,0,1
b,29.92,1.835,u,g,c,h,4.335,t,f,0,f,g,260.0,200,1
a,38.25,6.0,u,g,k,v,1.0,t,f,0,t,g,0.0,0,1
b,48.08,6.04,u,g,k,v,0.04,f,f,0,f,g,0.0,2690,1
a,45.83,10.5,u,g,q,v,5.0,t,t,7,t,g,0.0,0,1
b,36.67,4.415,y,p,k,v,0.25,t,t,10,t,g,320.0,0,1
b,28.25,0.875,u,g,m,v,0.96,t,t,3,t,g,396.0,0,1
a,23.25,5.875,u,g,q,v,3.17,t,t,10,f,g,120.0,245,1
b,21.83,0.25,u,g,d,h,0.665,t,f,0,t,g,0.0,0,1
a,19.17,8.585,u,g,cc,h,0.75,t,t,7,f,g,96.0,0,1
b,25.0,11.25,u,g,c,v,2.5,t,t,17,f,g,200.0,1208,1
b,23.25,1.0,u,g,c,v,0.835,t,f,0,f,s,300.0,0,1
a,47.75,8.0,u,g,c,v,7.875,t,t,6,t,g,0.0,1260,1
a,27.42,14.5,u,g,x,h,3.085,t,t,1,f,g,120.0,11,1
a,41.17,6.5,u,g,q,v,0.5,t,t,3,t,g,145.0,0,1
a,15.83,0.585,u,g,c,h,1.5,t,t,2,f,g,100.0,0,1
a,47.0,13.0,u,g,i,bb,5.165,t,t,9,t,g,0.0,0,1
b,56.58,18.5,u,g,d,bb,15.0,t,t,17,t,g,0.0,0,1
b,57.42,8.5,u,g,e,h,7.0,t,t,3,f,g,0.0,0,1
b,42.08,1.04,u,g,w,v,5.0,t,t,6,t,g,500.0,10000,1
b,29.25,14.79,u,g,aa,v,5.04,t,t,5,t,g,168.0,0,1
b,42.0,9.79,u,g,x,h,7.96,t,t,8,f,g,0.0,0,1
b,49.5,7.585,u,g,i,bb,7.585,t,t,15,t,g,0.0,5000,1
a,36.75,5.125,u,g,e,v,5.0,t,f,0,t,g,0.0,4000,1
a,22.58,10.75,u,g,q,v,0.415,t,t,5,t,g,0.0,560,1