TRƯỜNG ĐẠI HỌC VĂN LANG
KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP CÁ NHÂN BUỔI 3
Môn: NHẬP MÔN PHÂN TÍCH DỮ LIỆU VÀ HỌC SÂU

SVTH: Quang Mỹ Tâm

MSSV: 2274802010784

GVHD: GV. Nguyễn Thái Anh

Github:
https://github.com/CutieeYummyy/DATAANALYSISDEEPLEARNING

TP. Hồ Chí Minh – năm 2024

```python
import pandas as pd
import numpy as np
#Problem 1
# Thêm header vào DataFrame để diễn giải dữ liệu
column_names = ['ID', 'Name', 'Age', 'Weight', 'm0006', 'm0612', 'm1218',
'f0006', 'f0612', 'f1218']

# Read the CSV file
df = pd.read_csv(r'/home/patient_heart_rate.csv', names=column_names,
on_bad_lines='skip', skiprows=1)

# Display the first few rows of the dataframe
print(df.head())

#Problem 2
df[['FirstName', 'LastName']] = df['Name'].str.split(expand=True)
df= df.drop('Name', axis=1)
print(df)

#Problem 3
#Get the Weight column
weight = df["Weight"]

def convert_weight(weight):
    if isinstance(weight, str):
        if 'lbs' in weight:
            weight = weight[:-3]
            return str(int(float(weight) / 2.2)) + 'kgs'
    return weight

# Áp dụng hàm chuyển đổi cho cột 'Weight' trên bản sao DataFrame
df["Weight"] = weight.apply(convert_weight)
#print(df)

#Problem 4
df.dropna(how='all', inplace=True)
#print(df)

#Problem 5
df=df.drop_duplicates(subset=["FirstName","LastName","Age","Weight"])
#print(df)

#Problem 6
df.FirstName.replace({r'[\x00-\x7F]+':''}, regex=True, inplace=True)
```

```
df.LastName.replace({r'[\x00-\x7F]+':''}, regex=True, inplace=True)
#print(df)

#Problem 7
#Deletion: Remove records with missing values
df = df.dropna(subset=['Age', 'Weight'])
#print("\nDataFrame after Deletion:")
#print(df.head())
#Dummy substitution: Replace missing values with a dummy but valid
value:e.g.: 0 for numerical values
df_dummy = df.fillna({'Age': 0, 'Weight': '0kgs'})
#print("\nDataFrame after Dummy substitution:")
#print(df_dummy.head())
#Mean substitution: Replace the missing values with the mean
df_mean = df.copy()
df_mean['Age'] = df_mean['Age'].fillna(df_mean['Age'].mean())
df_mean['Weight'] =
df_mean['Weight'].fillna(df_mean['Weight'].apply(lambda x: int(x[:-3]) if
isinstance(x, str) else x).mean())
df_mean['Weight'] = df_mean['Weight'].astype(str) + 'kgs'
#print("\nDataFrame after Mean substitution:")
#print(df_mean.head())
#Frequent substitution: Replace the missing values with the most frequent
item.
df_frequent = df.copy()
df_frequent['Age'] =
df_frequent['Age'].fillna(df_frequent['Age'].mode()[0])
df_frequent['Weight'] =
df_frequent['Weight'].fillna(df_frequent['Weight'].mode()[0])
#print("\nDataFrame after Frequent substitution:")
#print(df_frequent.head())
#Yêu cầu
# Thống kê thông tin dữ liệu thiếu
missing_info = df[['Age', 'Weight']].isnull().sum()
missing_info_percentage = df[['Age', 'Weight']].isnull().mean() * 100

#print("Missing values count:\n", missing_info)
#print("Missing values percentage:\n", missing_info_percentage)

#Problem 8
#Mekt the Sex+ time range column in single column
df = pd.melt(df, id_vars=['ID','Age','Weight', 'FirstName', 'LastName'],
var_name='sex_and_time').sort_values(['ID','Age','Weight','FirstName','Las
tName'])
#Extract Sex, Hour Lower bound and Hour upper bound group
```

```
tmp_df = df["sex_and_time"].str.extract(r"(\D)(\d+)(\d{2})", expand=True)
# Added opening parenthesis before \D to balance the expression
#Name columns
tmp_df.columns = ['Sex', 'Hour_Lower', 'Hour_Upper']
#Create Time column based in 'Hour_Lower' and "hour_Upper" columns
tmp_df["Time"]=tmp_df["Hour_Lower"]+ "-" +tmp_df["Hour_Upper"]
#merge
df = pd.concat([df, tmp_df], axis=1)
#Drop unneccessary columns and rows
df = df.drop(['sex_and_time','Hour_Lower','Hour_Upper'], axis=1)
df = df.dropna()
df.to_csv('outputcleanup.csv', index=False)
#print(df)

#Problem 9

# Calculate the missing data percentage on blood pressure columns
missing_bp = df[['m0006', 'm0612', 'm1218', 'f0006', 'f0612',
'f1218']].isnull().mean() * 100
print("Missing data percentage on blood pressure columns:")
print(missing_bp)

# Function to replace missing blood pressure values

def replace_missing_bp(df):
    for col in ['m0006', 'm0612', 'm1218', 'f0006', 'f0612', 'f1218']:
        for i in range(len(df)):
            if pd.isna(df.loc[i, col]):
                replaced = False

                # 1) Replace with average of immediate neighbors
                if i > 0 and i < len(df) - 1 and pd.notnull(df.loc[i-1,
col]) and pd.notnull(df.loc[i+1, col]):
                    neighbor1 = float(df.loc[i-1, col]) if
pd.notnull(df.loc[i-1, col]) and isinstance(df.loc[i-1, col], str) else
df.loc[i-1, col]
                    neighbor2 = float(df.loc[i+1, col]) if
pd.notnull(df.loc[i+1, col]) and isinstance(df.loc[i+1, col], str) else
df.loc[i+1, col]
                    df.loc[i, col] = (neighbor1 + neighbor2) / 2
                    replaced = True

                # 2) Replace with average of two previous values
                if not replaced and i > 1 and pd.notnull(df.loc[i-1, col])
and pd.notnull(df.loc[i-2, col]):
```

```
                    prev1 = float(df.loc[i-1, col]) if
pd.notnull(df.loc[i-1, col]) and isinstance(df.loc[i-1, col], str) else
df.loc[i-1, col]
                    prev2 = float(df.loc[i-2, col]) if
pd.notnull(df.loc[i-2, col]) and isinstance(df.loc[i-2, col], str) else
df.loc[i-2, col]
                    df.loc[i, col] = (prev1 + prev2) / 2
                    replaced = True

                # 3) Replace with average of two subsequent values
                if not replaced and i < len(df) - 2 and
pd.notnull(df.loc[i+1, col]) and pd.notnull(df.loc[i+2, col]):
                    next1 = float(df.loc[i+1, col]) if
pd.notnull(df.loc[i+1, col]) and isinstance(df.loc[i+1, col], str) else
df.loc[i+1, col]
                    next2 = float(df.loc[i+2, col]) if
pd.notnull(df.loc[i+2, col]) and isinstance(df.loc[i+2, col], str) else
df.loc[i+2, col]
                    df.loc[i, col] = (next1 + next2) / 2
                    replaced = True
                # 4) Replace with average of all values for that person
                if not replaced:
                    person_bp_values = df.loc[i, ['m0006', 'm0612',
'm1218', 'f0006', 'f0612', 'f1218']].dropna().astype(float)
                    if len(person_bp_values) > 0:
                        df.loc[i, col] = person_bp_values.mean()
                    else:
                        # 5) Replace with average of all values for that
gender group
                        gender_group = col[0]
                        gender_bp_values = df[df['gender'] ==
gender_group][['m0006', 'm0612', 'm1218', 'f0006', 'f0612',
'f1218']].stack().astype(float)
                        if len(gender_bp_values) > 0:
                            df.loc[i, col] = gender_bp_values.mean()
                        else:
                            # 6) Replace with overall average
                            overall_bp_values = df[['m0006', 'm0612',
'm1218', 'f0006', 'f0612', 'f1218']].stack().astype(float)
                            df.loc[i, col] = overall_bp_values.mean()


    return df

# Apply the function to replace missing blood pressure values
```

```
df = replace_missing_bp(df)

# Display the dataframe after replacing missing blood pressure values
print("\nDataFrame after replacing missing blood pressure values:")
print(df.head())

# Problem 10

# Drop any rows with missing values in the dataset
df_clean = df.dropna()

# Reindex the cleaned dataframe
df_clean.reset_index(drop=True, inplace=True)

# Save the cleaned data to a CSV file
df_clean.to_csv(r'home/patient_data_clean_fix.csv', index=False)

print("Cleaned data saved to patient_data_clean.csv successfully.")
```

```
Problem 1:
    ID            Name   Age      Weight m0006 m0612 m1218 f0006 f0612 f1218
0  2.0     Donald Duck  34.0   154.89lbs     -     -     -    85    84    76
1  3.0      Mini Mouse  16.0         NaN     -     -     -    65    69    72
2  4.0  Scrooge McDuck   NaN        78kgs    78    79    72     -     -     -
3  5.0    Pink Panther  54.0  198.658lbs     -     -     -    69   NaN    75
4  6.0     Huey McDuck  52.0      189lbs     -     -     -    68    75    72
5  7.0    Dewey McDuck  19.0       56kgs     -     -     -    71    78    75
6  8.0      Scööpy Doo  32.0        78kgs    78    76    75     -     -     -
7  NaN             NaN   NaN         NaN   NaN   NaN   NaN   NaN   NaN   NaN
8  NaN             NaN   NaN         NaN   NaN   NaN   NaN   NaN   NaN   NaN
9  9.0     Huey McDuck  52.0      189lbs     -     -     -    68    75    72
```

```
Problem 2:
    ID   Age      Weight m0006 m0612 m1218 f0006 f0612 f1218 FirstName  \
0  2.0  34.0   154.89lbs     -     -     -    85    84    76    Donald
1  3.0  16.0         NaN     -     -     -    65    69    72      Mini
2  4.0   NaN       78kgs    78    79    72     -     -     -   Scrooge
3  5.0  54.0  198.658lbs     -     -     -    69   NaN    75      Pink
4  6.0  52.0      189lbs     -     -     -    68    75    72      Huey
5  7.0  19.0       56kgs     -     -     -    71    78    75     Dewey
6  8.0  32.0       78kgs    78    76    75     -     -     -    Scööpy
7  NaN   NaN         NaN   NaN   NaN   NaN   NaN   NaN   NaN       NaN
8  NaN   NaN         NaN   NaN   NaN   NaN   NaN   NaN   NaN       NaN
9  9.0  52.0      189lbs     -     -     -    68    75    72      Huey

  LastName
0     Duck
1    Mouse
2   McDuck
3  Panther
4   McDuck
5   McDuck
6      Doo
7      NaN
8      NaN
9   McDuck
```

```
Problem 3:
    ID   Age Weight m0006 m0612 m1218 f0006 f0612 f1218 FirstName LastName
0  2.0  34.0  70kgs     -     -     -    85    84    76    Donald     Duck
1  3.0  16.0    NaN     -     -     -    65    69    72      Mini    Mouse
2  4.0   NaN  78kgs    78    79    72     -     -     -   Scrooge   McDuck
3  5.0  54.0  90kgs     -     -     -    69   NaN    75      Pink  Panther
4  6.0  52.0  85kgs     -     -     -    68    75    72      Huey   McDuck
5  7.0  19.0  56kgs     -     -     -    71    78    75     Dewey   McDuck
6  8.0  32.0  78kgs    78    76    75     -     -     -    Scööpy      Doo
7  NaN   NaN    NaN   NaN   NaN   NaN   NaN   NaN   NaN       NaN      NaN
8  NaN   NaN    NaN   NaN   NaN   NaN   NaN   NaN   NaN       NaN      NaN
9  9.0  52.0  85kgs     -     -     -    68    75    72      Huey   McDuck
Problem 4:
      ID   Age Weight m0006 m0612 m1218 f0006 f0612 f1218 FirstName LastName
0    2.0  34.0  70kgs     -     -     -    85    84    76    Donald     Duck
1    3.0  16.0    NaN     -     -     -    65    69    72      Mini    Mouse
2    4.0   NaN  78kgs    78    79    72     -     -     -   Scrooge   McDuck
3    5.0  54.0  90kgs     -     -     -    69   NaN    75      Pink  Panther
4    6.0  52.0  85kgs     -     -     -    68    75    72      Huey   McDuck
5    7.0  19.0  56kgs     -     -     -    71    78    75     Dewey   McDuck
6    8.0  32.0  78kgs    78    76    75     -     -     -    Scööpy      Doo
9    9.0  52.0  85kgs     -     -     -    68    75    72      Huey   McDuck
10  10.0  12.0  45kgs     -     -     -    92    95    87     Louie   McDuck
11  11.0   NaN  60kgs    78    75    72     -     -     -     Henry      Nam
```

```
Problem 5:
      ID   Age Weight m0006 m0612 m1218 f0006 f0612 f1218 FirstName LastName
0    2.0  34.0  70kgs     -     -     -    85    84    76    Donald     Duck
1    3.0  16.0    NaN     -     -     -    65    69    72      Mini    Mouse
2    4.0   NaN  78kgs    78    79    72     -     -     -   Scrooge   McDuck
3    5.0  54.0  90kgs     -     -     -    69   NaN    75      Pink  Panther
4    6.0  52.0  85kgs     -     -     -    68    75    72      Huey   McDuck
5    7.0  19.0  56kgs     -     -     -    71    78    75     Dewey   McDuck
6    8.0  32.0  78kgs    78    76    75     -     -     -    Scööpy      Doo
10  10.0  12.0  45kgs     -     -     -    92    95    87     Louie   McDuck
11  11.0   NaN  60kgs    78    75    72     -     -     -     Henry      Nam
12  12.0  34.0    NaN    65    67    55     -     -     -    Michel     Long
Problem 6:
      ID   Age Weight m0006 m0612 m1218 f0006 f0612 f1218 FirstName LastName
0    2.0  34.0  70kgs     -     -     -    85    84    76
1    3.0  16.0    NaN     -     -     -    65    69    72
2    4.0   NaN  78kgs    78    79    72     -     -     -
3    5.0  54.0  90kgs     -     -     -    69   NaN    75
4    6.0  52.0  85kgs     -     -     -    68    75    72
5    7.0  19.0  56kgs     -     -     -    71    78    75
6    8.0  32.0  78kgs    78    76    75     -     -     -        öö
10  10.0  12.0  45kgs     -     -     -    92    95    87
11  11.0   NaN  60kgs    78    75    72     -     -     -
12  12.0  34.0    NaN    65    67    55     -     -     -
```

```
Problem 7:
Missing values count:
 Age       0
Weight    0
dtype: int64
Missing values percentage:
 Age       0.0
Weight    0.0
dtype: float64
```

```
Problem 8:
     ID   Age Weight FirstName LastName value Sex   Time
0   2.0  34.0  70kgs                          -   m  00-06
7   2.0  34.0  70kgs                          -   m  06-12
14  2.0  34.0  70kgs                          -   m  12-18
21  2.0  34.0  70kgs                         85   f  00-06
28  2.0  34.0  70kgs                         84   f  06-12
35  2.0  34.0  70kgs                         76   f  12-18
1   5.0  54.0  90kgs                          -   m  00-06
8   5.0  54.0  90kgs                          -   m  06-12
15  5.0  54.0  90kgs                          -   m  12-18
22  5.0  54.0  90kgs                         69   f  00-06

Missing values before processing:
ID           0
Age          0
Weight       0
FirstName    0
LastName     0
value        0
Sex          0
Time         0
dtype: int64
```

```
Problem 9:
    ID   Age      Weight  m0006    m0612    m1218   f0006   f0612   f1218  \
0  2.0  34.0   154.89lbs   78.0  79.0000  72.0000  85.000  84.000  76.000
1  3.0  16.0         NaN   78.0  79.0000  72.0000  65.000  69.000  72.000
2  4.0   NaN       78kgs   78.0  79.0000  72.0000  67.000  76.500  73.500
3  5.0  54.0  198.658lbs   78.0  79.0000  72.0000  69.000  75.750  75.000
4  6.0  52.0      189lbs   78.0  79.0000  72.0000  68.000  75.000  72.000
5  7.0  19.0       56kgs   78.0  77.5000  73.5000  71.000  78.000  75.000
6  8.0  32.0       78kgs   78.0  76.0000  75.0000  69.500  76.500  73.500
7  NaN   NaN         NaN   78.0  76.7500  74.2500  70.250  77.250  74.250
8  NaN   NaN         NaN   78.0  76.3750  74.6250  69.125  76.125  73.125
9  9.0  52.0      189lbs   78.0  76.5625  74.4375  68.000  75.000  72.000

  FirstName LastName
0    Donald     Duck
1      Mini    Mouse
2   Scrooge   McDuck
3      Pink  Panther
4      Huey   McDuck
5     Dewey   McDuck
6    Scöopy      Doo
7       NaN      NaN
8       NaN      NaN
9      Huey   McDuck
```

```
DataFrame after replacing missing blood pressure values:
    ID   Age      Weight  m0006  m0612  m1218  f0006  f0612  f1218 FirstName  \
0  2.0  34.0   154.89lbs   78.0   79.0   72.0   85.0  84.00   76.0    Donald
1  3.0  16.0         NaN   78.0   79.0   72.0   65.0  69.00   72.0      Mini
2  4.0   NaN       78kgs   78.0   79.0   72.0   67.0  76.50   73.5   Scrooge
3  5.0  54.0  198.658lbs   78.0   79.0   72.0   69.0  75.75   75.0      Pink
4  6.0  52.0      189lbs   78.0   79.0   72.0   68.0  75.00   72.0      Huey

  LastName
0     Duck
1    Mouse
2   McDuck
3  Panther
4   McDuck
```

```
Problem 10:
     ID   Age      Weight  m0006     m0612     m1218  f0006  f0612  f1218  \
0   2.0  34.0   154.89lbs   78.0  79.00000  72.00000   85.0  84.00   76.0
1   5.0  54.0  198.658lbs   78.0  79.00000  72.00000   69.0  75.75   75.0
2   6.0  52.0      189lbs   78.0  79.00000  72.00000   68.0  75.00   72.0
3   7.0  19.0       56kgs   78.0  77.50000  73.50000   71.0  78.00   75.0
4   8.0  32.0       78kgs   78.0  76.00000  75.00000   69.5  76.50   73.5
5   9.0  52.0      189lbs   78.0  76.56250  74.43750   68.0  75.00   72.0
6  10.0  12.0       45kgs   78.0  75.78125  73.21875   92.0  95.00   87.0

  FirstName LastName
0    Donald     Duck
1      Pink  Panther
2      Huey   McDuck
3     Dewey   McDuck
4    Scöopy      Doo
5      Huey   McDuck
6     Louie   McDuck
Cleaned data saved to patient_data_clean.csv successfully.
```