```
import pandas as pd

# Load the dataset
file_path = '/content/DataSampled.csv'
data = pd.read_csv(file_path)
# Let's save the dataframe as a new CSV file with a specific name.
new_file_path = '/content/CleanedDataSample.csv'
data.to_csv(new_file_path, index=False)

new_data = pd.read_csv(new_file_path)


# Remove spaces and make all column names lowercase
 # Conditionally modify column names
new_data.columns = [
    col.strip().replace(' ', '_').capitalize() if col.lower() not in ['bmi', 'mmse']
    else col
    for col in data.columns
]
```

```
new_data.columns
```

```
Index(['MMSE', 'Age', 'Weight', 'Height', 'Waist', 'Hip', 'Smoking',
       'Smoking_(packet/year)', 'Alcohol', 'Dm', 'Dm_duration', 'İnsülin',
       'Dm_drug', 'Hiperlipidemi', 'Dyslipidemia_duration',
       'Dyslipidemia_drugs', 'Kah', 'Kah_duration', 'Hipotiroidi', 'Astim',
       'Koah', 'Op', 'Other(s)', 'Ht', 'Anti-ht_drug_type', 'Ht_duration',
       'Education', 'Occupation', 'Working_status', 'Exercise', 'Lowcst',
       'Cst', 'Gait_speed', 'Low_grip_strength', 'Grip_strength', 'Sarcopenia',
       'Star', 'BMI', 'Gender'],
      dtype='object')
```

```
import pandas as pd

# Assuming 'new_data' is your DataFrame
# Rename the column 'hipertension' to 'hypertension'
new_data = new_data.rename(columns={'hipertension': 'hypertension', 'Hiperlipidemi':'Hyperlipidemia' , 'Hipotiroidi':'Hypothyroidism',

# Verify the column names to ensure the change has been made
print(new_data.columns)
```

```
Index(['MMSE', 'Age', 'Weight', 'Height', 'Waist', 'Hip', 'Smoking',
       'Smoking_(packet/year)', 'Alcohol', 'Dm', 'Dm_duration', 'Insulin',
       'Dm_drug', 'Hyperlipidemia', 'Dyslipidemia_duration',
       'Dyslipidemia_drugs', 'Kah', 'Kah_duration', 'Hypothyroidism', 'Astim',
       'Koah', 'Op', 'Other(s)', 'Ht', 'Anti-ht_drug_type', 'Ht_duration',
       'Education', 'Occupation', 'Working_status', 'Exercise', 'Lowcst',
       'Cst', 'Gait_speed', 'Low_grip_strength', 'Grip_strength', 'Sarcopenia',
       'Star', 'BMI', 'Gender'],
      dtype='object')
```

```
new_data.head()
```

	MMSE	Age	Weight	Height	Waist	Hip	Smoking	Smoking_(packet/year)	Alcohol	D
0	NaN	64	66.0	155.0	89.0	104.0	0.0	0	0	
1	NaN	53	55.0	150.0	77.0	97.0	0.0	0	0	
2	24.0	56	56.0	150.0	112.0	125.0	0.0	0	0	
3	NaN	58	75.0	160.0	113.0	NaN	0.0	0	0	
4	30.0	55	72.0	157.0	98.0	104.0	1.0	4	0	

5 rows × 39 columns

```
def standardize_occupation(occupation):
    occupation = str(occupation).strip().lower()
    if 'house wife' in occupation or 'housewife' in occupation or 'ev hanımı' in occupation:
        return 'Housewife'
    elif 'officer' in occupation or 'police' in occupation:
        return 'Officer'
    elif occupation in ['farmer', 'driver', 'gardener', 'mechanic', 'housekeeper', 'construction worker']:
        return occupation.capitalize()
    else:
        return 'Others'

new_data['Occupation'] = new_data['Occupation'].apply(standardize_occupation)
```

```
import numpy as np
import pandas as pd

def determine_active_level(row):
    occupation = row['Occupation']
    exercise = row['Exercise']

    new_data['Exercise'] = new_data['Exercise'].replace({'1-2/week': 1, '3-4/week': 2})

    # Convert 'exercise' to float if possible, otherwise set to NaN
    try:
        exercise = float(exercise)
    except ValueError:
        exercise = np.nan

    if pd.isna(exercise):  # Check if 'Exercise' is NaN after attempting conversion
        return np.nan

    active_occupations = ['Officer', 'Farmer', 'Driver', 'Gardener', 'Mechanic', 'Housekeeper', 'Construction Worker']

    if occupation in active_occupations:
        if exercise >= 1:
            return 3  # Active and exercises
        elif exercise == 0:
            return 2  # Active but doesn't exercise
    else:
        if exercise >= 1:
            return 3  # Not active but exercises
        elif exercise == 0:
            return 1  # Not active and doesn't exercise

    return np.nan  # For any other cases not covered

# Assuming 'data' is your DataFrame
new_data['Active'] = new_data.apply(determine_active_level, axis=1)
```

```
# Assuming 'data' is your DataFrame and you want to print 'Occupation' and 'Active' columns
print(new_data[['Occupation', 'Exercise', 'Active']])
```

```
      Occupation Exercise  Active
0        Officer        0     2.0
1         Others      NaN     NaN
2      Housewife        0     1.0
3      Housewife        0     1.0
4      Housewife        1     NaN
...          ...      ...     ...
1298      Others        0     1.0
1299   Housewife        2     NaN
1300      Others        0     1.0
1301     Officer        0     2.0
1302      Others        1     NaN

[1303 rows x 3 columns]
```

```
def categorize_bmi(bmi):
    if bmi < 18.5:
        return 'Underweight'
    elif 18.5 <= bmi <= 24.9:
        return 'Healthy Weight'
    elif 25.0 <= bmi <= 29.9:
        return 'Overweight'
    elif bmi >= 30.0:
        return 'Obesity'
    else:
        return 'Unknown'  # In case there are NaN or negative values



new_data['Weight-status'] = new_data['BMI'].apply(categorize_bmi)



print(new_data[['BMI', 'Weight-status']].head())
```

```
      BMI   Weight-status
0  27.40      Overweight
1  24.40  Healthy Weight
2  36.00         Obesity
3  29.30      Overweight
4  29.21      Overweight
```

```
# Cleaning and encoding Gender
new_data['Gender'] = new_data['Gender'].str.upper().str.strip()  # Convert to uppercase and strip spaces
new_data['Gender'] = new_data['Gender'].replace({'FEMALE': 'F', 'MALE': 'M', ' F': 'F', 'M': 'M', 'F ': 'F', 'f': 'F', ' f':'F'})
new_data.Gender.value_counts()
```

```
Gender
F    924
M    379
Name: count, dtype: int64
```

```
new_data['Smoking_(packet/year)'] = new_data['Smoking_(packet/year)'].replace({'20/ex-smoker': 20, 'EX SMOKER 300/YEAR 9 YRS BACK': 300

import pandas as pd
import numpy as np

# Assuming 'new_data' is your DataFrame
# Replace any remaining non-numeric strings with NaN, then convert the column to integer
new_data['Smoking_(packet/year)'] = pd.to_numeric(new_data['Smoking_(packet/year)'], errors='coerce')

# Convert to int, filling NaNs with a placeholder if needed
new_data['Smoking_(packet/year)'] = new_data['Smoking_(packet/year)'].fillna(-1).astype(int)
new_data['Smoking_(packet/year)'].replace(-1, np.nan, inplace=True)  # Replace placeholder back to NaN

# Display unique values to ensure conversion
print(new_data['Smoking_(packet/year)'])
```

```
0        0.0
1        0.0
2        0.0
3        0.0
4        4.0
        ...
1298    50.0
1299    45.0
1300    46.0
1301     0.0
1302    30.0
Name: Smoking_(packet/year), Length: 1303, dtype: float64
```

```
import pandas as pd
import numpy as np

# Assuming 'new_data' is your DataFrame
# Ensure all values are strings and replace as expected
new_data['Alcohol'] = new_data['Alcohol'].astype(str).replace({
    '0': 0, 'social': 1, 'regular': 2, 'nan': np.nan
})

# Convert the entire column to numeric now, setting non-numeric residuals to NaN
new_data['Alcohol'] = pd.to_numeric(new_data['Alcohol'], errors='coerce')

# Check the unique values to ensure changes are applied
print(new_data['Alcohol'].unique())
```

```
[ 0.  1. nan  2.]
```

```
import numpy as np
# Display the cleaned columns
new_data[['Smoking_(packet/year)', 'Alcohol', 'Gender']].head()
```

	Smoking_(packet/year)	Alcohol	Gender
0	0.0	0.0	F
1	0.0	0.0	F
2	0.0	0.0	F
3	0.0	0.0	F
4	4.0	0.0	F

```
new_data.columns
```

```
Index(['MMSE', 'Age', 'Weight', 'Height', 'Waist', 'Hip', 'Smoking',
       'Smoking_(packet/year)', 'Alcohol', 'Dm', 'Dm_duration', 'Insulin',
       'Dm_drug', 'Hyperlipidemia', 'Dyslipidemia_duration',
       'Dyslipidemia_drugs', 'Kah', 'Kah_duration', 'Hypothyroidism', 'Astim',
       'Koah', 'Op', 'Other(s)', 'Ht', 'Anti-ht_drug_type', 'Ht_duration',
       'Education', 'Occupation', 'Working_status', 'Exercise', 'Lowcst',
       'Cst', 'Gait_speed', 'Low_grip_strength', 'Grip_strength', 'Sarcopenia',
       'Star', 'BMI', 'Gender', 'Active', 'Weight-status'],
      dtype='object')
```

```
import pandas as pd

# Assuming 'new_data' is your DataFrame
# List of columns to drop
columns_to_drop = ['Height', 'Weight', 'Waist', 'Other(s)', 'Hip', 'Education', 'Working_status', 'Star', 'Occupation', 'Smoking_(packet/

# Drop columns related to drugs and duration
# These columns need to be listed by their exact names if they exist in your DataFrame
# Example column names for drugs and duration, you might need to adjust these based on your DataFrame
drug_duration_columns = ['Dm_drug', 'Dm_duration', 'Dyslipidemia_duration', 'Dyslipidemia_drugs', 'Kah_duration', 'Anti-ht_drug_type', 'H

# Combine all columns to drop
all_columns_to_drop = columns_to_drop + drug_duration_columns

# Drop the columns from the DataFrame
new_data = new_data.drop(columns=all_columns_to_drop, errors='ignore')  # errors='ignore' will prevent errors if a column is not found

# Display the columns to confirm they are dropped
print(new_data.columns)
```

```
Index(['MMSE', 'Age', 'Smoking', 'Alcohol', 'Dm', 'Insulin', 'Hyperlipidemia',
       'Kah', 'Hypothyroidism', 'Astim', 'Koah', 'Op', 'Ht', 'Cst',
       'Gait_speed', 'Grip_strength', 'Sarcopenia', 'Gender', 'Active',
       'Weight-status'],
      dtype='object')
```

```
# Attempting again to save the modified DataFrame to a new CSV file
new_csv_path = '/content/CleanedNewData.csv'
new_data.to_csv(new_csv_path, index=False)

new_csv_path
new_df = pd.read_csv(new_csv_path)
```

```
new_df.head()
```

```
MMSE  Age  Smoking  Alcohol  Dm  Insulin  Hyperlipidemia  Kah  Hypothyroidism  Astim  Koah   Op  Ht  Cst  Gait_speed  Grip_streng
```