

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [2]: df = pd.read_csv("clean.csv")
df.head()
```

```
Out[2]:
```

	SEQN	RIAGENDR	RIDAGEYR	RIDRETH3	DMDBORN4	DMDEDUC2	INDFMPIR
0	109290.0	2.0	68.0	4.0	1.0	5.0	5.0
1	109290.0	2.0	68.0	4.0	1.0	5.0	5.0
2	109290.0	2.0	68.0	4.0	1.0	5.0	5.0
3	109290.0	2.0	68.0	4.0	1.0	5.0	5.0
4	109300.0	2.0	54.0	6.0	2.0	5.0	5.0

5 rows x 65 columns

```
In [3]: df.columns
```

```
Out[3]: Index(['SEQN', 'RIAGENDR', 'RIDAGEYR', 'RIDRETH3', 'DMDBORN4', 'DMDEDUC2',
              'INDFMPIR', 'DR1TKCAL', 'DR1TPROT', 'DR1TCARB', 'DR1TSUGR', 'DR1TFIB
              E',
              'DR1TTFAT', 'DR1TSFAT', 'DR1TCHOL', 'DR1TALCO', 'DR1TCAFF', 'DR1TSOD
              I',
              'DR1TPOTA', 'DR2TKCAL', 'DR2TPROT', 'DR2TCARB', 'DR2TSUGR', 'DR2TFIB
              E',
              'DR2TTFAT', 'DR2TSFAT', 'DR2TCHOL', 'DR2TALCO', 'DR2TCAFF', 'DR2TSOD
              I',
              'DR2TPOTA', 'BPX0SY1', 'BPX0DI1', 'BPX0SY2', 'BPX0DI2', 'BPX0SY3',
              'BPX0DI3', 'BPX0PLS1', 'BPX0PLS2', 'BPX0PLS3', 'BMXWT', 'BMXHT',
              'BMXBMI', 'BMXWAIST', 'LBXIN', 'LBXGLU', 'LBXTR', 'LBDLDL', 'LBDHD
              D',
              'LBXHSCR', 'PHAFSTHR', 'PAQ605', 'PAQ620', 'PAQ635', 'PAQ650',
              'PAQ665', 'RXDUSE', 'RXDDRUG', 'RXDDRGID', 'RXDRSC1', 'BPQ020',
              'DIQ010', 'DBQ700', 'DBQ197', 'HIQ011'],
              dtype='object')
```

```
In [4]: columns_to_drop = [
    # Day 1 dietary variables
    'DR1TPROT', 'DR1TCARB', 'DR1TCHOL', 'DR1TALCO',
    'DR1TCAFF', 'DR1TSODI', 'DR1TPOTA',

    # Day 2 dietary variables
    'DR2TPROT', 'DR2TCARB', 'DR2TCHOL', 'DR2TALCO',
    'DR2TCAFF', 'DR2TSODI', 'DR2TPOTA',

    # Weight and height
    'BMXWT', 'BMXHT',
```

```

# Blood pressure pulse readings
'BPX0PLS1', 'BPX0PLS2', 'BPX0PLS3',

# Drug metadata
'RXDDRGID', 'RXDRSC1',

'SEQN', 'LBDLDL', 'RIDAGEYR', 'RIDRETH3', 'DMDBORN4', 'DMDDEDUC2', 'PHAFS
'PAQ665', 'BPQ020', 'DIQ010', 'HIQ011'
]
df.drop(columns=columns_to_drop, inplace=True)

```

Feature Engineering

Clinical Risk Indicators (Metabolic Syndrome Flags)

```

In [5]: # Obesity based on waist circumference (gender-specific)
df['flag_obese'] = ((df['RIAGENDR'] == 1) & (df['BMXWAIST'] > 102)) | \
                  ((df['RIAGENDR'] == 2) & (df['BMXWAIST'] > 88))

# Elevated blood pressure
df['avg_sys'] = df[['BPX0SY1', 'BPX0SY2', 'BPX0SY3']].mean(axis=1)
df['avg_dia'] = df[['BPX0DI1', 'BPX0DI2', 'BPX0DI3']].mean(axis=1)
df['flag_high_bp'] = (df['avg_sys'] >= 130) | (df['avg_dia'] >= 85)

# Elevated fasting glucose
df['flag_high_glucose'] = df['LBXGLU'] >= 100

# Elevated triglycerides
df['flag_high_triglycerides'] = df['LBXTR'] >= 150

# Low HDL cholesterol (gender-specific)
df['flag_low_hdl'] = ((df['RIAGENDR'] == 1) & (df['LBDHDD'] < 40)) | \
                    ((df['RIAGENDR'] == 2) & (df['LBDHDD'] < 50))

# HOMA-IR
df['HOMA_IR'] = (df['LBXIN'] * df['LBXGLU']) / 405
df['flag_homa_ir'] = df['HOMA_IR'] > 2.5

```

Derived Metabolic Syndrome Target

```

In [6]: df['met_syndrome_count'] = df[['flag_obese', 'flag_high_bp', 'flag_high_gluc
        'flag_high_triglycerides', 'flag_low_hdl', 'f
df['has_metabolic_syndrome'] = df['met_syndrome_count'] >= 3

```

Lifestyle Feature Engineering

```

In [7]: # Average Daily Intake
df['avg_kcal'] = df[['DR1TKCAL', 'DR2TKCAL']].mean(axis=1)
df['avg_sugar'] = df[['DR1TSUGR', 'DR2TSUGR']].mean(axis=1)
df['avg_fat'] = df[['DR1TTFAT', 'DR2TTFAT']].mean(axis=1)
df['avg_fiber'] = df[['DR1TFIBE', 'DR2TFIBE']].mean(axis=1)

```

```

# Change in Intake
df['delta_kcal'] = df['DR2TKCAL'] - df['DR1TKCAL']
df['delta_sugar'] = df['DR2TSUGR'] - df['DR1TSUGR']
df['delta_fat'] = df['DR2TTFAT'] - df['DR1TTFAT']

# Binary Flags for Intake Reduction
df['reduced_calories'] = df['delta_kcal'] < -100
df['reduced_sugar'] = df['delta_sugar'] < -10

# Likely Dieting (Overweight + trying to lose weight + reduced intake)
df['likely_dieting'] = (
    (df['BMXBMI'] >= 25) &
    df['reduced_calories'] &
    df['DBQ700'].eq(1)
)

# Eating Out Behavior
df['eats_out_often'] = df['DBQ197'].isin([1, 2]) # every day / most days
df['eats_out_rarely'] = df['DBQ197'].isin([4, 5]) # 1-2x/week or never

# Physical Activity
df['physically_active'] = (
    df[['PAQ605', 'PAQ620', 'PAQ635', 'PAQ650']].isin([1]).any(axis=1)
)
df['activity_score'] = df[['PAQ605', 'PAQ620', 'PAQ635', 'PAQ650']].applymap

# Lifestyle Effort Score
df['lifestyle_effort'] = (
    df['likely_dieting'].astype(int)
    + df['eats_out_rarely'].astype(int)
    + df['delta_kcal'].apply(lambda x: 1 if x < 0 else 0) # decreased calor
    + df['delta_sugar'].apply(lambda x: 1 if x < 0 else 0) # decreased suga
    + df['physically_active'].astype(int) # NEW: physical activity
)

```

/var/folders/z1/w_njtp52ss65yn2_5wp9xfm0000gn/T/ipykernel_51747/814351403.p
y:31: FutureWarning: DataFrame.applymap has been deprecated. Use DataFrame.m
ap instead.

```
df['activity_score'] = df[['PAQ605', 'PAQ620', 'PAQ635', 'PAQ650']].applym
ap(lambda x: 1 if x == 1 else 0).sum(axis=1)
```

Med_Class Variable

```

In [8]: met_syn_drugs = [
    # Diabetes-related
    'METFORMIN', 'INSULIN', 'GLIPIZIDE', 'GLYBURIDE', 'PIOGLITAZONE', 'SITAG
    'EMPAGLIFLOZIN', 'DAPAGLIFLOZIN', 'CANAGLIFLOZIN', 'LINAGLIPTIN', 'LIRAG
    'EXENATIDE', 'SEMAGLUTIDE', 'GLIMEPIRIDE',

    # Hypertension
    'LISINOPRIL', 'LOSARTAN', 'AMLODIPINE', 'METOPROLOL', 'RAMIPRIL', 'FUROS
    'HYDROCHLOROTHIAZIDE', 'VALSARTAN', 'BISOPROLOL', 'SPIRONOLACTONE', 'DIL

    # Cholesterol

```

```

    'ATORVASTATIN', 'ROSUVASTATIN', 'SIMVASTATIN', 'PRAVASTATIN', 'PITAVASTATIN',
    'EZETIMIBE', 'FENOFIBRATE', 'GEMFIBROZIL', 'FENOFIBRIC ACID', 'NIACIN',
]

# Clean and classify each record
def classify_meds(drug_str):
    if pd.isna(drug_str) or drug_str.strip().upper() in ['NONE', 'NO MEDICATION']:
        return 'None'

    # Clean and split into list
    drugs = [d.strip().upper() for d in drug_str.split(';')]

    has_met_syn = any(any(med in drug for med in met_syn_drugs) for drug in drugs)
    has_non_met_syn = any(not any(med in drug for med in met_syn_drugs) for drug in drugs)

    if has_met_syn and has_non_met_syn:
        return 'Both MetSyn + Other'
    elif has_met_syn:
        return 'MetSyn-related'
    elif has_non_met_syn:
        return 'Other'
    else:
        return 'None'

# Apply to your column
df['med_class'] = df['RXDDRUG'].astype(str).apply(classify_meds)

```

```

In [9]: # 1 corresponds to using medication
        # 2 corresponds to not using medication

```

Income_Level Variable

```

In [10]: df['income_level'] = pd.cut(df['INDFMPIR'],
                                     bins=[-0.1, 1.3, 3.5, df['INDFMPIR'].max()],
                                     labels=['Low Income', 'Middle Income', 'High Income'])

```

```

In [11]: df.columns

```

```
Out[11]: Index(['RIAGENDR', 'INDFMPPIR', 'DR1TKCAL', 'DR1TSUGR', 'DR1TFIBE', 'DR1TTFA
T',
            'DR1TSFAT', 'DR2TKCAL', 'DR2TSUGR', 'DR2TFIBE', 'DR2TTFAT', 'DR2TSFA
T',
            'BPX0SY1', 'BPX0DI1', 'BPX0SY2', 'BPX0DI2', 'BPX0SY3', 'BPX0DI3',
            'BMXBMI', 'BMXWAIST', 'LBXIN', 'LBXGLU', 'LBXTR', 'LBDHDD', 'LBXHSCR
P',
            'PAQ605', 'PAQ620', 'PAQ635', 'PAQ650', 'RXDUSE', 'RXDDRUG', 'DBQ70
0',
            'DBQ197', 'flag_obese', 'avg_sys', 'avg_dia', 'flag_high_bp',
            'flag_high_glucose', 'flag_high_triglycerides', 'flag_low_hdl',
            'HOMA_IR', 'flag_homa_ir', 'met_syndrome_count',
            'has_metabolic_syndrome', 'avg_kcal', 'avg_sugar', 'avg_fat',
            'avg_fiber', 'delta_kcal', 'delta_sugar', 'delta_fat',
            'reduced_calories', 'reduced_sugar', 'likely_dieting', 'eats_out_of_t
en',
            'eats_out_rarely', 'physically_active', 'activity_score',
            'lifestyle_effort', 'med_class', 'income_level'],
            dtype='object')
```

EDA

Do people on meds show differences in effort or health outcomes compared to those not on meds?

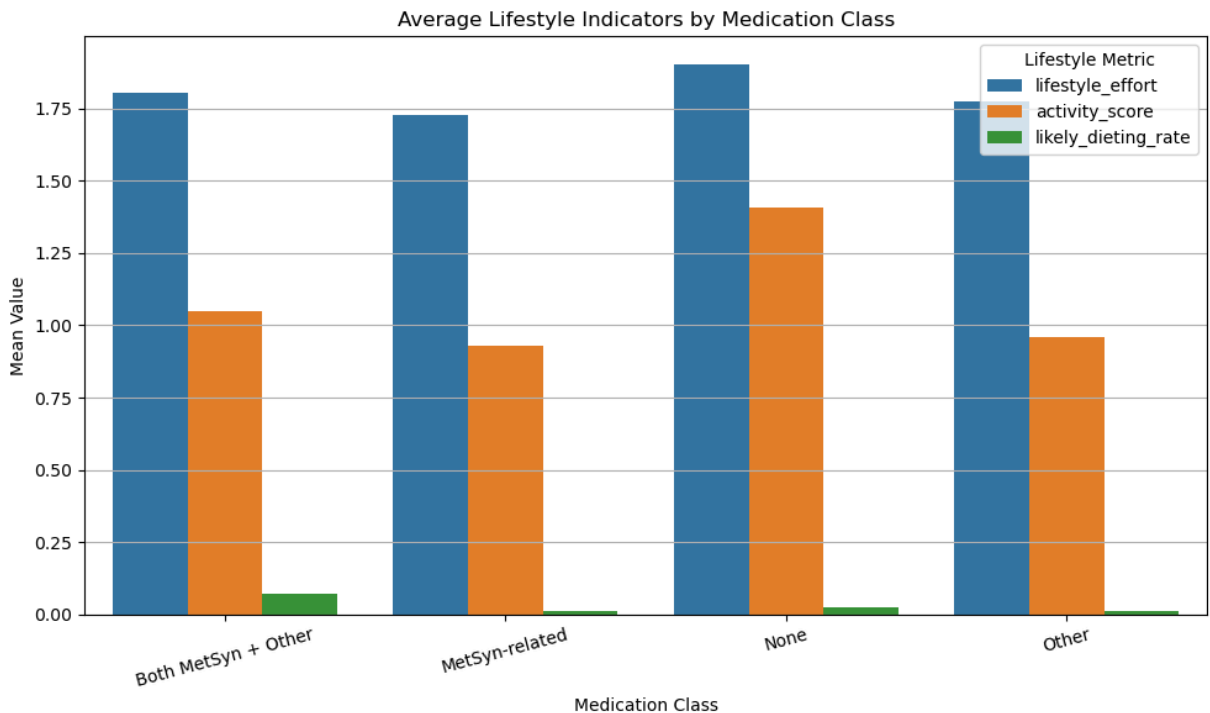
```
In [12]: df.groupby('med_class')[['lifestyle_effort', 'activity_score', 'likely_dieting']].mean()
```

```
Out[12]:
```

	lifestyle_effort	activity_score	likely_dieting
med_class			
Both MetSyn + Other	1.804878	1.048780	0.073171
MetSyn-related	1.728976	0.928880	0.011533
None	1.904219	1.405929	0.023945
Other	1.775388	0.958241	0.013603

```
In [13]: df['likely_dieting_numeric'] = df['likely_dieting'].astype(int)
# Calculate means grouped by medication class
grouped_means = df.groupby('med_class')[['lifestyle_effort', 'activity_score', 'likely_dieting_numeric']].mean()
grouped_means = grouped_means.rename(columns={'likely_dieting_numeric': 'likely_dieting_numeric'})
# Melt for plotting
melted = pd.melt(grouped_means, id_vars='med_class', var_name='Metric', value_name='Mean Value')
# Plotting
plt.figure(figsize=(10, 6))
sns.barplot(data=melted, x='med_class', y='Mean Value', hue='Metric')
plt.title('Average Lifestyle Indicators by Medication Class')
plt.ylabel('Mean Value')
plt.xlabel('Medication Class')
plt.xticks(rotation=15)
plt.tight_layout()
```

```
plt.legend(title='Lifestyle Metric')
plt.grid(axis='y')
plt.show()
```



Those who don't take medication have slightly higher lifestyle_effort and activity_score, but other than that the values for lifestyle_effort, activity_score, and likely_dieting_rate are generally around the same. Overall however, the averages for all of these are quite low, the highest average of lifestyle_effort is slightly over 1.75 and for those who aren't taking medicine, same for activity_score is between 1.25 and 1.50, which is also for those who aren't taking medicine. Then the highest average for likely_dieting_rate is well under 0.25 and it is for those who are taking medicine for metabolic syndrome and other.

Who is healthy without medication?

```
In [14]: # Number of people not taking medication and are healthy
df[(df['med_class'] == 'None') & (df['has_metabolic_syndrome'] == False)].sh
```

```
Out[14]: (528, 62)
```

```
In [15]: # Number of people not taking medication and are healthy
df[(df['med_class'] == 'None') & (df['has_metabolic_syndrome'] == True)].sha
```

```
Out[15]: (349, 62)
```

```
In [16]: # Number of people taking medication and are not healthy
df[(df['med_class'] == 'MetSyn-related') & (df['has_metabolic_syndrome'] ==
```

```
Out[16]: (1561, 62)
```

```
In [17]: df[(df['med_class'] == 'MetSyn-related') & (df['has_metabolic_syndrome'] ==
```

```
Out[17]: (520, 62)
```

Metabolic syndrome vs med_class

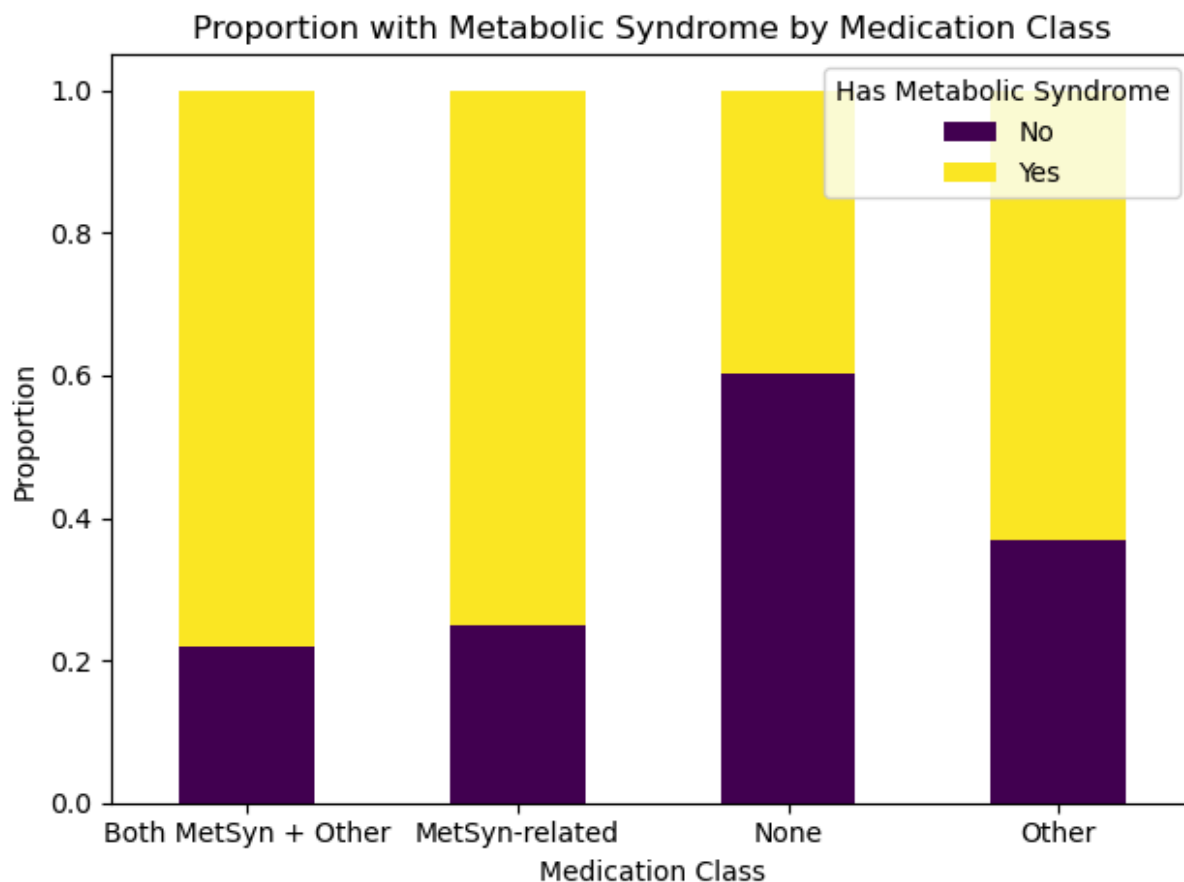
```
In [18]: df.groupby('med_class')['has_metabolic_syndrome'].value_counts(normalize=True)
```

```
Out[18]: med_class      has_metabolic_syndrome
Both MetSyn + Other  True      0.780488
                   False      0.219512
MetSyn-related      True      0.750120
                   False      0.249880
None                False      0.602052
                   True       0.397948
Other               True       0.632395
                   False      0.367605
Name: proportion, dtype: float64
```

```
In [19]: # Get normalized proportions
met_syn_dist = df.groupby('med_class')['has_metabolic_syndrome'].value_counts()

# Plot as stacked bar
met_syn_dist.plot(kind='bar', stacked=True, colormap='viridis')

plt.title('Proportion with Metabolic Syndrome by Medication Class')
plt.ylabel('Proportion')
plt.xlabel('Medication Class')
plt.legend(title='Has Metabolic Syndrome', labels=['No', 'Yes'])
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```



the None medication class has the lowest proportion for those who have metabolic syndrome and the highest proportion for those who don't have metabolic syndrome. For the rest who do take any kind of medication, it shows that they are all similar in that only 1/3 at the most don't have metabolic syndrome, and that the rest of the proportion do.

Does med_class improve prediction?

This project explores how well lifestyle and behavior predict metabolic syndrome in comparison to models that also include medication use. While medication is usually prescribed after diagnosis, and thus expected to correlate with the presence of metabolic syndrome, our goal is to understand its added predictive value. Through exploratory analysis and predictive modeling, we examine whether individuals on medications differ in their health outcomes and efforts compared to those who are not. Ultimately, we assess whether lifestyle-based indicators alone are strong enough for predictive modeling, reinforcing prevention-focused strategies.

```
In [20]: # Bin lifestyle_effort
df['lifestyle_bin'] = pd.cut(df['lifestyle_effort'],
                             bins=[-0.1, 1, 2, df['lifestyle_effort'].max()],
                             labels=['Poor', 'Moderate', 'Good'])

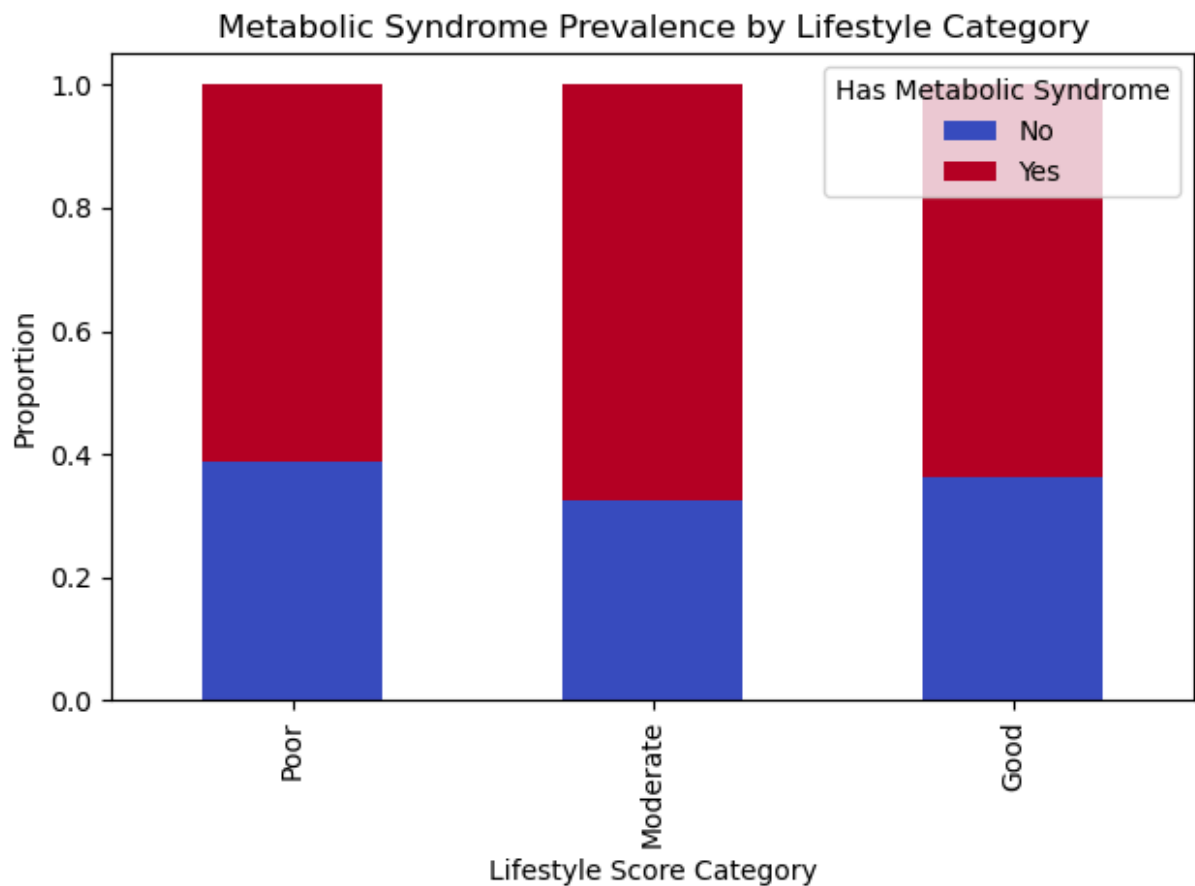
# Group and normalize
lifestyle_met_syn = df.groupby('lifestyle_bin')['has_metabolic_syndrome'].va
```



```
# Plot
lifestyle_met_syn.plot(kind='bar', stacked=True, colormap='coolwarm')
plt.title('Metabolic Syndrome Prevalence by Lifestyle Category')
plt.ylabel('Proportion')
plt.xlabel('Lifestyle Score Category')
plt.legend(title='Has Metabolic Syndrome', labels=['No', 'Yes'])
plt.tight_layout()
plt.show()
```

/var/folders/z1/w_njtp52ss65yn2_5wp9xfm0000gn/T/ipykernel_51747/2994981896.py:7: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence the warning.

```
lifestyle_met_syn = df.groupby('lifestyle_bin')['has_metabolic_syndrome'].value_counts(normalize=True).unstack()
```



Surprisingly, the proportions generally seem the same, about 40% or slightly less of those with poor, moderate, or good lifestyles don't have metabolic syndrome, and the rest do.

```
In [21]: # Create income bins
df['income_level'] = pd.cut(df['INDFMPIR'],
                             bins=[-0.1, 1.3, 3.5, df['INDFMPIR'].max()],
                             labels=['Low Income', 'Middle Income', 'High Income'])

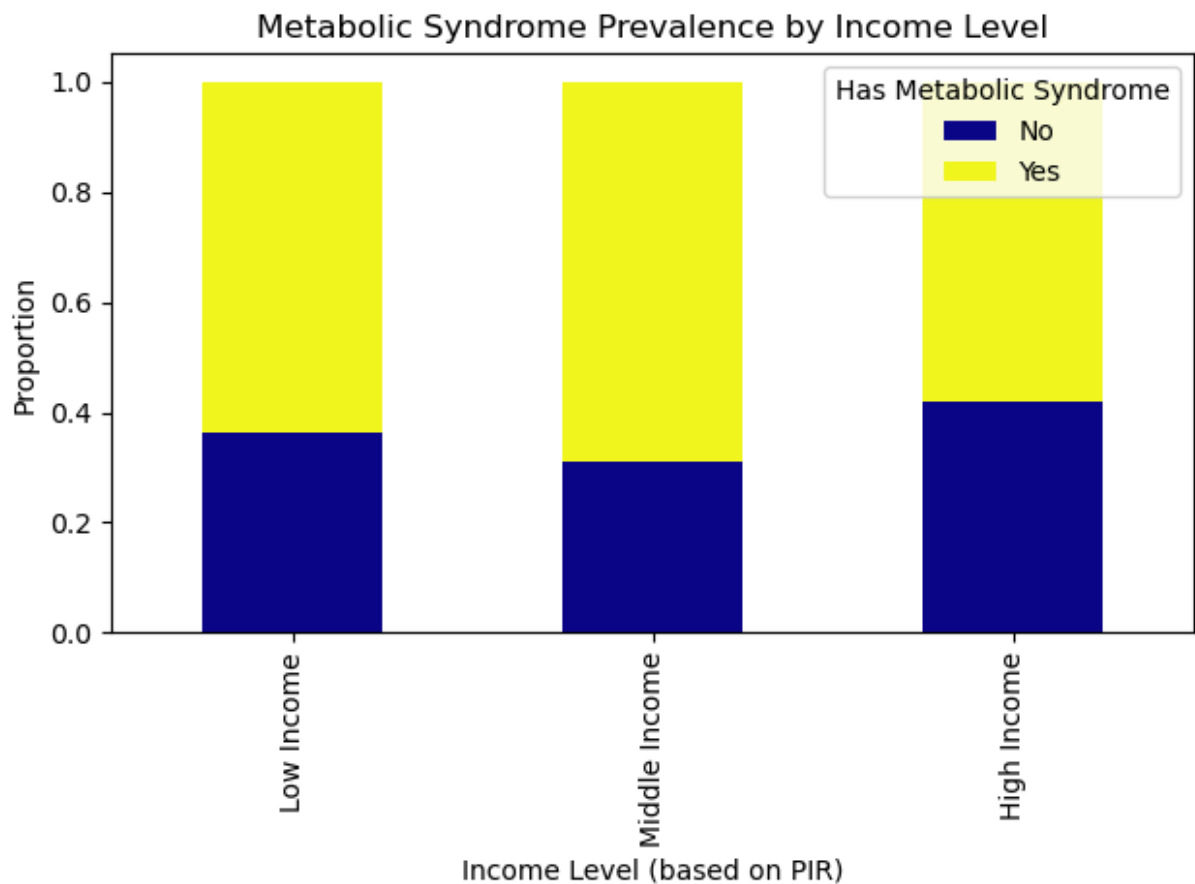
# Group and normalize
```

```
income_met_syn = df.groupby('income_level')['has_metabolic_syndrome'].value_

# Plot
income_met_syn.plot(kind='bar', stacked=True, colormap='plasma')
plt.title('Metabolic Syndrome Prevalence by Income Level')
plt.ylabel('Proportion')
plt.xlabel('Income Level (based on PIR)')
plt.legend(title='Has Metabolic Syndrome', labels=['No', 'Yes'])
plt.tight_layout()
plt.show()
```

/var/folders/z1/w_njtp52ss65yn2_5wp9xfm0000gn/T/ipykernel_51747/3630438039.py:7: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.

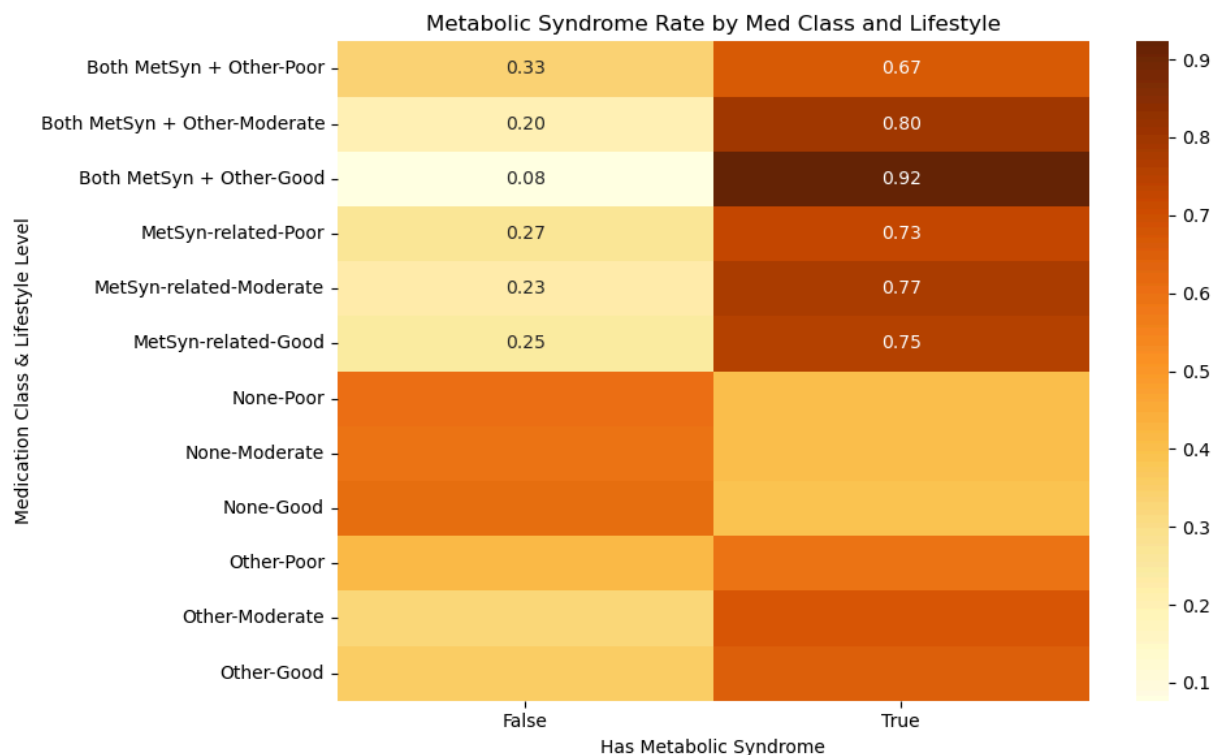
```
income_met_syn = df.groupby('income_level')['has_metabolic_syndrome'].value_counts(normalize=True).unstack()
```



The same pattern is noticed across income levels, 40% or less of the proportion doesn't have metabolic syndrome while the rest do, and this is for low, middle, and high income individuals.

```
In [22]: # Cross-tab counts
cross_tab = pd.crosstab([df['med_class'], df['lifestyle_bin']],
                        df['has_metabolic_syndrome'],
                        normalize='index')
```

```
# Plot as heatmap
plt.figure(figsize=(10,6))
sns.heatmap(cross_tab, annot=True, fmt=".2f", cmap='YlOrBr')
plt.title('Metabolic Syndrome Rate by Med Class and Lifestyle')
plt.xlabel('Has Metabolic Syndrome')
plt.ylabel('Medication Class & Lifestyle Level')
plt.tight_layout()
plt.show()
```



the cross tab is pretty much a mix of the previous plots and has similar results

Does med_class improve prediction?

```
In [23]: df_plot = df.copy()
df_plot['MetS'] = df_plot['has_metabolic_syndrome'].map({False: 'No MetS', 1
```

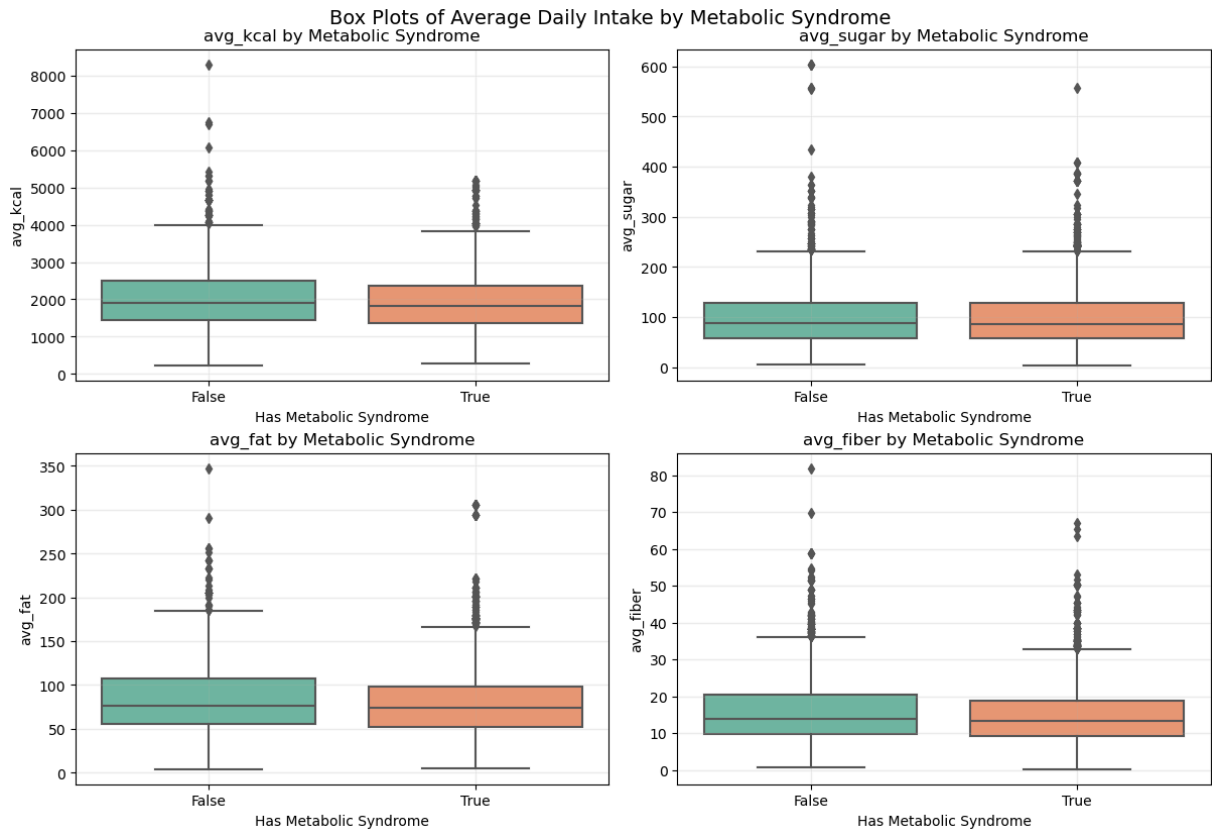
```
In [26]: avg_vars = ['avg_kcal', 'avg_sugar', 'avg_fat', 'avg_fiber']

fig, axes = plt.subplots(2, 2, figsize=(12, 8), constrained_layout=True)
axes = axes.ravel()

for ax, col in zip(axes, avg_vars):
    sns.boxplot(
        data=df,
        x='has_metabolic_syndrome',
        y=col,
        palette='Set2',
        ax=ax
    )
    ax.set_title(f"{col} by Metabolic Syndrome", fontsize=12)
    ax.set_xlabel("Has Metabolic Syndrome")
    ax.set_ylabel(col)
```

```
ax.grid(alpha=0.2)

plt.suptitle("Box Plots of Average Daily Intake by Metabolic Syndrome", y=1.1)
plt.savefig("boxplots_avg_intake.png", dpi=300, bbox_inches='tight')
plt.show()
```



```
In [64]: df[df['has_metabolic_syndrome'] == True][avg_vars].describe()
```

```
Out[64]:
```

	avg_kcal	avg_sugar	avg_fat	avg_fiber
count	3941.000000	3941.000000	3941.000000	3941.000000
mean	1906.121670	97.676370	79.839429	14.805379
std	776.213294	57.399837	38.797628	7.921685
min	283.500000	4.055000	5.105000	0.200000
25%	1358.000000	57.595000	52.665000	9.250000
50%	1833.500000	86.920000	74.560000	13.200000
75%	2365.500000	127.990000	98.225000	18.750000
max	5175.500000	556.920000	305.295000	67.100000

```
In [65]: df[df['has_metabolic_syndrome'] == False][avg_vars].describe()
```

Out [65]:

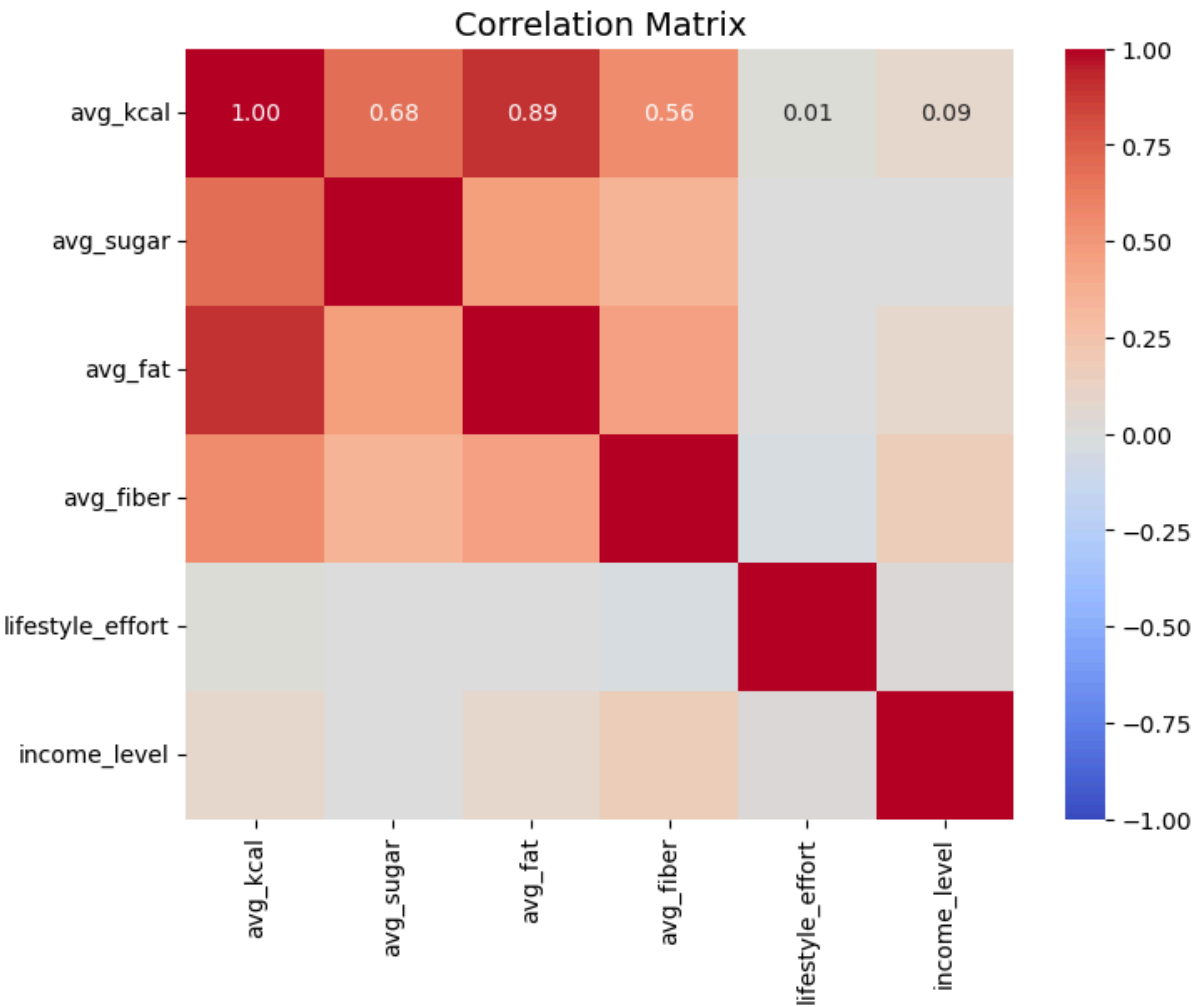
	avg_kcal	avg_sugar	avg_fat	avg_fiber
count	2219.000000	2219.000000	2219.000000	2219.000000
mean	2045.445020	100.574351	83.949349	15.957346
std	847.757015	63.069199	40.286931	9.313710
min	227.000000	6.120000	3.950000	0.650000
25%	1451.250000	57.825000	56.050000	9.650000
50%	1913.000000	88.095000	76.710000	13.900000
75%	2491.000000	128.655000	107.865000	20.300000
max	8313.000000	604.345000	347.585000	81.900000

```
In [50]: numeric_df = df[['avg_kcal', 'avg_sugar', 'avg_fat', 'avg_fiber',
                          'lifestyle_effort', 'income_level']].copy()

# Mapping income levels
income_mapping = {'Low Income': 0, 'Middle Income': 1, 'High Income': 2}
numeric_df['income_level'] = numeric_df['income_level'].map(income_mapping).

# Pearson correlation
corr_matrix = numeric_df.corr(method='pearson')

# Simple heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(
    corr_matrix,
    annot=True,          # show correlation values
    fmt=".2f",           # two decimal places
    cmap="coolwarm",     # color palette
    vmin=-1, vmax=1      # correlation range
)
plt.title("Correlation Matrix", fontsize=14)
plt.savefig("correlation_matrix.png", dpi=300, bbox_inches='tight')
plt.show()
```



```
In [34]: numeric_df.corr(method='pearson')
```

Out [34]:

	avg_kcal	avg_sugar	avg_fat	avg_fiber	lifestyle_effort	income_level
avg_kcal	1.000000	0.680347	0.892725	0.556251	0.008749	0.085
avg_sugar	0.680347	1.000000	0.464911	0.348755	0.000237	0.001
avg_fat	0.892725	0.464911	1.000000	0.448292	-0.002858	0.078
avg_fiber	0.556251	0.348755	0.448292	1.000000	-0.045648	0.169
lifestyle_effort	0.008749	0.000237	-0.002858	-0.045648	1.000000	0.023
income_level	0.085876	0.001470	0.078461	0.169010	0.023422	1.000

```
In [ ]:
```