Obesity Analysis

Introduction:
Obesity is a disorder involving excessive body fat that increases the risk of health problems. It occurs when a person's body mass index is 30 or greater. This project aims to better understand the impact of different variables on obesity by using the dataset "obesity.csv" that is provided by the UCI Machine Learning Repository.

Dataset Breakdown:
The dataset contains data of 2111 respondents of a survey that asks for the following information of each respondent: gender, age, height, weight, family history, eating habits and physical conditions. The respondents are from the countries of Colombia, Mexico and Peru ranging from 14 to 61 years old. The dataset specifically contains 17 attributes listed below:

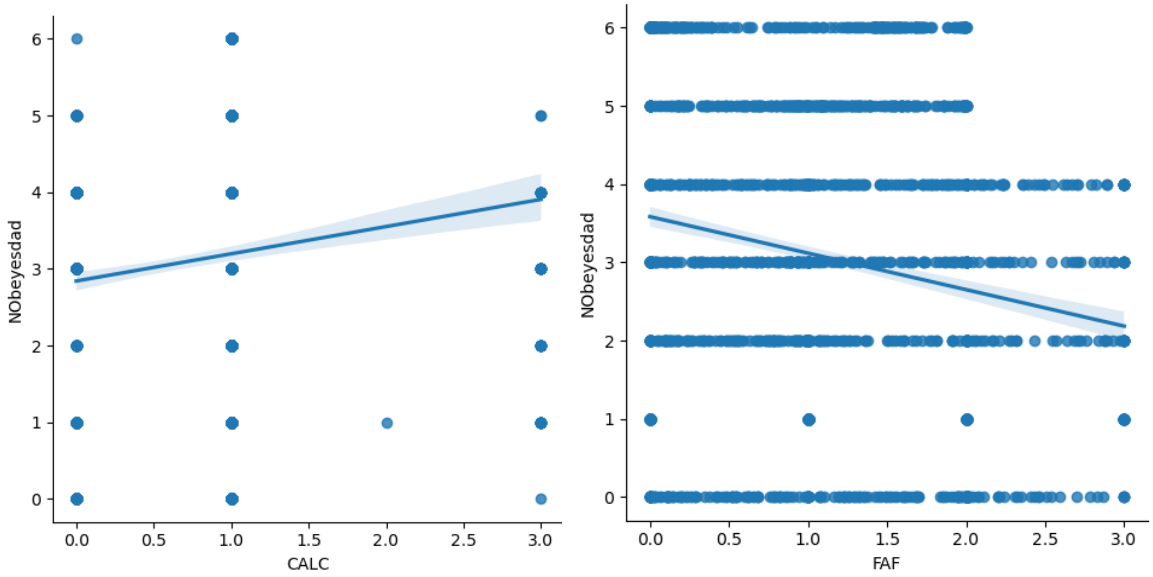
- **NObeyesdad**: Obesity level. This is our variable of interest and values within this attribute are values of 7 categories: Underweight, Normal Weight, Overweight Level I, Overweight Level II, Obesity Type I, Obesity Type II and Obesity Type III. This is based on definition of obesity set by the WHO where each category is determined by the range of Body Mass Index where underweight corresponds to less than 18.5, normal weight corresponds to 18.5 to 24.9, overweight corresponds to 25.0 to 29.9, Obesity Type I corresponds to 30.0 to 34.9, Obesity Type II corresponds to 35.0 to 39.9, Obesity Type III corresponds to 40.0+.
- **Age**: Continuous variable ranging from 14 to 61 years old.
- **Gender**: Categorical variable with values male and female.
- **Height**: Continuous variable measured in meters.
- **Weight**: Continuous variable measured in kilograms.
- **family_history_with_overweight**: Categorical variable with values yes and no.
- **FAVC**: Frequency of consumption of high caloric food; categorical variable with values yes and no.
- **FCVC**: Frequency of consumption of vegetables; categorical variable with numerical values 0 corresponding to never, 1 to sometimes, and 2 to always.
- **NCP**: Number of main meals; numerical variable with values 1-4.
- **CAEC**: Consumption of food in-between meals; categorical variable with values no, sometimes, frequently and always.
- **SMOKE**: Categorical variable that corresponds to smoker and nonsmoker with values yes and no.
- **CH2O**: Consumption of water daily; categorical variable with numerical values 1-3 measured in liters.
- **SCC**: Calories consumption monitoring; categorical variable with values yes and no.
- **FAF**: Physical activity frequency; categorical variable with numerical values 1-4 measured in days over a week.
- **TUE**: Time using technology devices; categorical variable with numerical values 1-3. 1 corresponds to 0-2 hours per day, 2 corresponds to 3-5 hours per day and 3 corresponds to more than 6 hours per day.
- **CALC:** Consumption of alcohol; categorical variable with values no, sometimes, frequently and always.
- **MTRANS**: Mode of transport; categorical variable with values automobile, motorbike, bike, public transport, and walking.

Output Overview:

I wrote a python code file that consists of four functions. One imports, prepares, and cleans the data to be ready for analysis, one generates visuals for observational analysis, one generates descriptive statistics for numerical analysis, and the last one generates a model for regression analysis. Below is the specific breakdown of the output generated for each function:

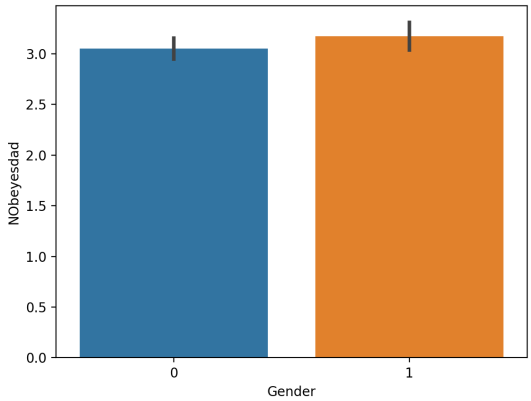
- **import_and_clean**: This function imports and cleans the dataset "obesity.csv", which contains information about obesity rates and its related variables. Firstly, it drops all rows with missing values to ensure the completeness of the dataset. Secondly, it alters certain categorical variables by changing its values to numerical values for more efficient data analysis. For example, “FAVC”, “SMOKE”, and “SCC” are all attributes that consist of values “yes” and “no”, and they are replaced by “1” and “0” to be prepared for regression analysis. “Gender” consists of values “Female” and “Male”, and they are replaced with 0 and 1. For attributes “NObeyesdad”, “CAEC” and “CALC”, they are all categorical variables with values that have numerical significance. For example, “Never”, “Sometimes” and “Always” are categorized based on frequency, while levels of obesity is categorized based on Body Mass Index. Therefore, it’s appropriate to replace the values from these attributes with numerical values that will then be used for analysis.

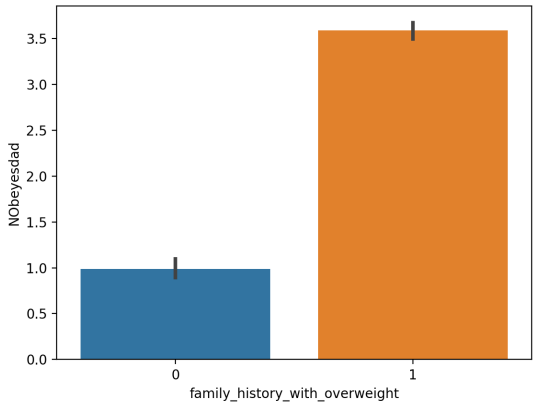
- **data_visualization**: This function generates visuals based on the “obesity.csv” dataset so that we can make some observations. Let’s first go over the lmplots.



  - **Interpretation:** The first graph is a lmplot that shows the relationship between consumption of alcohol and obesity level. There seems to be a positive correlation between alcohol consumption and obesity level, which means this is potentially an example of harmful practice in preventing obesity. The second graph is also a lmplot that shows the relationship between frequency of physical activities and obesity level. There seems to be a negative correlation between obesity frequency of physical activities and obesity level, which means this is potentially an example of beneficial practice in preventing obesity.

- **data_visualization:** We can then go over the barplots that examine categorical variables that can be potentially significant in relation to obesity levels.





  - **Interpretation:** The first graph is a bar plot that compares the average obesity level of all the male respondents against all the female respondents. It shows that there's a slightly higher level of average obesity level among female respondents compared to male respondent, but the different isn't significant enough. On the other hand, the second graph is also a bar plot that shows the difference in obesity level between people who have family history with overweight problems compared to people who don't have such history. It shows that on average, people with overweight history have a significantly higher obesity level than people who don't have that history, suggesting that this is an important variable in correlation to obesity level.

- **descriptive_statistics:** This function generates descriptive statistics (mean, median, standard deviation etc) of each column of the "obesity" data frame and we can make certain observations on them.

	Age	Height	Weight	NCP	FAF	CALC	NObeyesdad
count	2111.000000	2111.000000	2111.000000	2111.000000	2111.000000	2111.000000	2111.000000
mean	24.312600	1.701677	86.586058	2.685628	1.010298	0.764093	3.112269
std	6.345968	0.093305	26.191172	0.778039	0.850592	0.616717	1.985062
min	14.000000	1.450000	39.000000	1.000000	0.000000	0.000000	0.000000
25%	19.947192	1.630000	65.473343	2.658738	0.124505	0.000000	1.000000
50%	22.777890	1.700499	83.000000	3.000000	1.000000	1.000000	3.000000
75%	26.000000	1.768464	107.430682	3.000000	1.666678	1.000000	5.000000
max	61.000000	1.980000	173.000000	4.000000	3.000000	3.000000	6.000000

- **Observations:** From this table above, the average respondent is around 24.3 years old, 1.70 meters tall, weighing 86.6 kilograms, has 2.70 meals a day, exercises 1-2 days a week and drinks alcohol at a frequency between “no drinking at all” and “sometimes”. The average respondent is also on a scale of 3.1 out of 6 in terms of obesity levels, which places this person around the “Overweight Level II” mark. The respondents range from 14-61 years old, 1.45 to 1.98 meters tall, weighing from 39 to 173 kilograms and ranging from “Underweight” to “Obesity Level III. The descriptive stats of eating habits attributes like NCP and CALC are interpretated in context as I have replaced the categorical values with numerical ones. Overall, this group of respondents seems representative to the overall population as this group has a diverse range of age, height, weight, eating and exercise habits.

- **data_model:** This function generates a model for the given obesity data in “obesity.csv” and uses OLS regression to estimate such model. It then estimates the accuracy of the results generated by the OLS regression.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:             NObeyesdad   R-squared:                       0.951
Model:                            OLS   Adj. R-squared:                  0.951
Method:                 Least Squares   F-statistic:                     2899.
Date:                Fri, 17 Dec 2021   Prob (F-statistic):               0.00
Time:                        20:44:09   Log-Likelihood:                -1261.2
No. Observations:                2111   AIC:                             2552.
Df Residuals:                    2096   BIC:                             2637.
Df Model:                          14
Covariance Type:            nonrobust
=================================================================================================
                                    coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------------------
const                             9.0185      0.269     33.585      0.000       8.492       9.545
Gender                           -0.0653      0.027     -2.460      0.014      -0.117      -0.013
Age                               0.0137      0.002      8.166      0.000       0.010       0.017
Weight                            0.0771      0.001    143.382      0.000       0.076       0.078
Height                           -7.5931      0.166    -45.816      0.000      -7.918      -7.268
family_history_with_overweight    0.3233      0.030     10.889      0.000       0.265       0.382
FAVC                              0.0386      0.032      1.204      0.229      -0.024       0.102
FCVC                             -0.0008      0.020     -0.039      0.969      -0.040       0.038
NCP                               0.0266      0.013      2.054      0.040       0.001       0.052
CAEC                             -0.1395      0.016     -8.726      0.000      -0.171      -0.108
SMOKE                            -0.0638      0.068     -0.934      0.351      -0.198       0.070
CH2O                             -0.0182      0.017     -1.096      0.273      -0.051       0.014
FAF                              -0.0803      0.013     -6.422      0.000      -0.105      -0.056
TUE                              -0.0098      0.017     -0.583      0.560      -0.043       0.023
CALC                             -0.0463      0.016     -2.856      0.004      -0.078      -0.015
==============================================================================
Omnibus:                      124.101   Durbin-Watson:                   1.244
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              294.634
Skew:                          -0.349   Prob(JB):                     1.05e-64
Kurtosis:                       4.692   Cond. No.                     3.05e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.05e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
Test accuracy =  78.49360492657507 %
```

- **significant values:**
    - Gender, Age, Weight, Height: These attributes are the biological information provided by each respondent. They all have a p-value that's near 0 (lower than 0.05), suggesting that they are all statistically significant in relation to obesity levels. The coefficient for height is negative, suggesting that there's a negative correlation between height and obesity levels.
    - family_history_with_overweight: This variable has a p-value that's near 0 (lower than 0.05), suggesting that family history with obesity is statistically significant in relation to obesity levels. This is no surprise as genetics can play a significant role in obesity. It also has a negative coefficient, suggesting there's a negative correlation between family history with obesity and obesity levels.
    - CAEC: This variable is acronym for consumption of food in between meals, which basically suggests snacks consumption. This variable has a p-value that's near 0 (less than 0.05), suggesting that it's statistically significant in correlation to obesity levels. It also has a negative, but small coefficient, suggesting that there's a negative correlation between frequency of snacks consumed and obesity levels.
    - FAF: This variable is acronym for frequency of physical activities. This variable has a p-value near 0 (lower than 0.05), suggesting that it's statistically significant in correlation to obesity levels. It also has a negative, but small coefficient, suggesting that there's a weak, negative correlation between frequency of physical activities and obesity levels.
    - NCP: This variable is acronym for number of main meals. This variable has a p-value of 0.04 (less than 0.05), suggesting that it's statistically significant in correlation to obesity levels. It also has a positive coefficient, suggesting that there's a positive correlation between number of main meals consumed during a day and obesity levels.
- **insignificant values:**
    - FAVC: This variable is acronym for frequency of consumption of vegetables. It has a p-value of 0.969, higher than 0.05, suggesting that the variable is statistically insignificant and there's no correlation between consumption of vegetables and obesity level in this model.
    - FCVC: This variable is acronym for frequency of consumption of high calorie foods. It has a p-value of 0.229, higher than 0.05, suggesting that the variable is statistically insignificant and there's no correlation between consumption of vegetables and obesity level in this model.
    - SMOKE: This variable categorizes smokers and nonsmokers. It has a p-value of 0.351, higher than 0.05, suggesting that it's not statistically significant and there's no correlation between being a smoker (or not) and obesity level in this model.
    - CH2O: This variable is acronym for frequency of consumption of water. It has a p-value of 0.273, higher than 0.05, suggesting that it's not statistically significant and there's no correlation between water consumption and obesity level in this model.
    - TUE: This variable is acronym for time/duration of using technology devices, like phones, laptops, and tablets etc. It has a p-value of 0.560, higher than 0.05, suggesting that it's not statistically significand and there's no correlation between duration of using technology devices and obesity level in this model.

- **model overview**
    - The OLS Regression model has an R-Squared value of 0.95, suggesting that 95% of the variances within the obesity dataset can be explained by the model. It also has an accuracy score of around 78.5%, suggesting that this model is fairly accurate in predicting obesity levels based on given attributes.

Overall Findings:

The results generated from this project aligns with what we've know about obesity. From graphs and the regression model, they've shown that family history, levels of physical activities and eating habits all play an important factor in obesity. The model yields some surprising results, however, as duration of using technology and smoking isn't a factor in determining overall obesity levels. Some ideas for further investigation include getting access to more data with continuous attributes so that statistical analysis can yield more accurate results.

Work Cited:

https://archive.ics.uci.edu/ml/index.php
https://www.sciencedirect.com/science/article/pii/S2352340919306985?via%3Dihub#fig1

Code (functions.py):

```
"""
This is the functions portion of the module for the
final project where it aims to better understand the impact
of different variables on obesity.
"""
import seaborn as sns
import pandas as pd
import statsmodels.api as sm
from sklearn.metrics import accuracy_score


def import_and_clean(dataset: str):
    """
    This function imports and cleans the dataset
    "obesity.csv", which contains information about
    obesity rates and its related variables. It replaces
    the categorical variable "NObeyesdad" with numerical
    values 0-6, from insufficient weight to obesity type III.
    Parameters
    ==========
    dataset: string
        name of the csv dataset file
    Returns
    =======
    "cleaned" dataset with missing values removed and "NObeyesdad"
    column original values replaced with numerical values for easier
    analysis.
    """
    obesity = pd.read_csv(dataset).dropna()
    # replace values in the "gender" column where male=0 and female=1
    obesity["Gender"].replace(["Male","Female"],[0,1],inplace=True)
    # replace values in the "family_history_with_overweight" column
    # where no=0 and yes=1.
    obesity["family_history_with_overweight"].replace(["no","yes"],[0,1],inplace=True)
    # replace values in the "SCC" column where no corresponds to 0
    # and yes corresponds to 1.
    obesity["FAVC"].replace(["no","yes"],[0,1],inplace=True)
    # replace values in the "SCC" column where no corresponds to 0
    # and yes corresponds to 1.
    obesity["SCC"].replace(["no","yes"],[0,1],inplace=True)
    # replace values in the "SMOKE" column where no corresponds to 0
    # and yes corresponds to 1.
    obesity["SMOKE"].replace(["no","yes"],[0,1],inplace=True)
    # replace values in the "NObeyesdad" column where categorical variables are
    # replaced with numericals from 0-6 for easier analysis.
```

```
    obesity["NObeyesdad"].replace(["Insufficient_Weight", "Normal_Weight",
                "Overweight_Level_I", "Overweight_Level_II",
                "Obesity_Type_I", "Obesity_Type_II", "Obesity_Type_III"],
                [0, 1, 2, 3, 4, 5, 6], inplace=True)
    # replace values in the "CAEC" column where categorical variables are
    # replaced with numericals from 0-3 for easier analysis.
    obesity["CAEC"].replace(["no", "Sometimes", "Always", "Frequently"],
                        [0,1,2,3], inplace=True)
    # replace values in the "NObeyesdad" column where categorical variables are
    # replaced with numericals from 0-3 for easier analysis.
    obesity["CALC"].replace(["no", "Sometimes", "Always", "Frequently"],
                        [0,1,2,3], inplace=True)
    return obesity


def data_visualization(dataset: str):
    """
    This function generates graphs and other visuals to show
    the relationships between each variable and obesity
    rates. This will give us a better undertstanding of
    what factors can make someone more susceptible to obesity.
    Parameters
    ==========
    dataset: string
        name of the csv dataset file
    """
    obesity = import_and_clean(dataset)
    # lmplot that shows relationship between alcohol consumption
    # and obesity level.
    sns.lmplot(x = "CALC", y = "NObeyesdad", data = obesity)
    # lmplot that shows relationship between frequency of physical
    # activities and obesity level.
    sns.lmplot(x = "FAF", y = "NObeyesdad", data = obesity)
    # barplot that shows the difference in obesity level between
    # males and females.
    sns.barplot(x = "Gender", y = "NObeyesdad", data = obesity)
    # barplot that shows the difference in obesity level between
    # people who have overweight family history and people who don't
    sns.barplot(x = "family_history_with_overweight", y = "NObeyesdad",
            data = obesity)


def descriptive_statistics(dataset: str):
    """
    This function provides descriptive statistics like median,
    mean and standard deviations etc for us to better understand
```

```
    the dataset. Specifically, comparison of obesity level between
    male vs female, family history vs no family history, smoker vs
    nonsmoker.
    Parameters
    ==========
    dataset: string
        name of the csv dataset file
    """
    obesity = import_and_clean(dataset)
    # Overview of dataset
    relevant_vars = obesity[["Age","Height","Weight","NCP","FAF","CALC","NObeyesdad"]]
    print(relevant_vars.describe())


def data_model(dataset: str):
    """
    This function generates a model for obesity data and estimates
    such model by producing an OLS regression table.
    Parameters
    ==========
    dataset: string
        name of the csv dataset file
    """
    obesity = import_and_clean(dataset)
    lhs = obesity["NObeyesdad"]
    ind_vars = ["Gender","Age","Weight","Height","family_history_with_overweight",
                "FAVC","FCVC","NCP","CAEC","SMOKE","CH2O","FAF","TUE","CALC"]
    rhs = obesity.loc[:, ind_vars]
    rhs = sm.add_constant(rhs)
    mod = sm.OLS(lhs, rhs)
    res = mod.fit()
    print(res.summary())
    yhat = res.predict(rhs)
    prediction = list(map(round, yhat))
    accuracy = accuracy_score(lhs, prediction)
    print('Test accuracy = ', accuracy*100, '%')
```

Code: testing.py

```
"""
This is the testing portion of the module draft for the final project where it tests the 4 functions in
the functions.py file.
"""
import os.path
import pandas as pd
import FinalProject.functions as f

def test_import_and_clean():
    """
    This function tests the returned output generated
    by the import_and_clean function in functions.py
    """
    dataframe = pd.read_csv("obesity.csv").dropna()
    dataframe_generated = f.import_and_clean("obesity.csv")
    # check number of columns for the dataframe generated here
    assert len(dataframe_generated.columns) ==  17
    # check null values
    assert dataframe_generated.isNull().sum() == 0
    # check columns of the cleaned datadframe in functions.py
    # compared to data frame generated here
    assert len(dataframe_generated.columns) == len(dataframe.columns)
    print("test_import_and_clean passed")

def test_data_visualization():
    """
    This function is a unit test that tests the returned
    output by the data_visualization function in functions.py
    """
    # test for figure 1
    path_1 = 'Documents/ECON406/FinalProject/FinalProject/Figure1.png'
    if os.path.isfile(path_1) and os.access(path_1, os.R_OK):
        print("File exists and is readable. test_data_visualization passed.")
    else:
        print("Either the file is missing or not readable")
    # test for figure 2
    path_2 = 'Documents/ECON406/FinalProject/FinalProject/Figure2.png'
    if os.path.isfile(path_2) and os.access(path_2, os.R_OK):
        print("File exists and is readable. test_data_visualization passed.")
    else:
        print("Either the file is missing or not readable")
    # test for figure 3
    path_3 = 'Documents/ECON406/FinalProject/FinalProject/Figure3.png'
    if os.path.isfile(path_3) and os.access(path_3, os.R_OK):
        print("File exists and is readable. test_data_visualization passed.")
```

```
    else:
        print("Either the file is missing or not readable")
    # test for figure 4
    path_4 = 'Documents/ECON406/FinalProject/FinalProject/Figure4.png'
    if os.path.isfile(path_4) and os.access(path_4, os.R_OK):
        print("File exists and is readable. test_data_visualization passed.")
    else:
        print("Either the file is missing or not readable")

def test_descriptive_statistics():
    """
    This function is a unit test that tests the returned
    output by the descriptive_statistics function in functions.py
    """
    des_stat = f.descriptive_statistics("obesity.csv")
    cols = ["Age","Height","Weight","NCP","FAF","CALC","NObeyesdad"]
    assert len(des_stat.columns) == len(cols)
    assert isinstance(des_stat, list) is True
    print("test_ddescriptive_statistics passed.")

def test_data_model():
    """
    This function is a unit test that tests the returned
    output by the data_model function in functions.py
    """
    cleaned_data = f.import_and_clean("obesity.csv")
    ols_model = f.data_model(cleaned_data)
    ols_fit = ols_model.fit()
    assert ols_fit.params["const"] != 0
    assert len(ols_fit.params) == 15
    assert ols_fit.cov_type == "nonrobust"
    assert len(ols_fit.predict()) == len(cleaned_data["NObeyesdad"])
    print("test_data_model passed.")

def main_test():
    """
    This function runs all the tests together at once to
    determine if all the tests are passed.
    """
    test_import_and_clean()
    test_data_visualization()
    test_descriptive_statistics()
    test_data_model()
    print("All unit tests are passed!")
```