**TRANSFORMATIONS**

Instructions:

Please share your answers filled inline in the word document. Submit code files wherever applicable.

Please ensure you update all the details:

**Name: Vishvash C**

**Batch Id: 23012024**

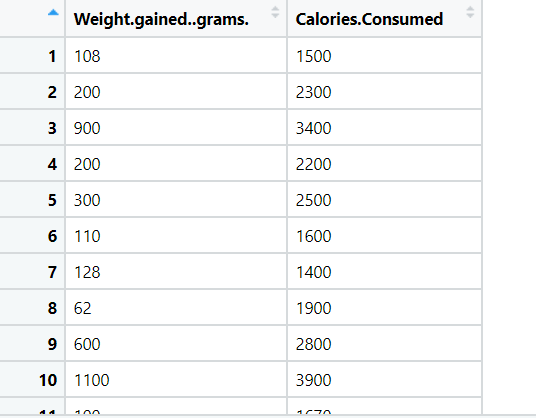
**Topic: Data Pre-Processing**

**Problem Statement:**

Everything will revolve around the data in Analytics world. Proper data will help you to make useful predictions that improve your business. Sometimes the usage of original data as it is does not help to have accurate solutions. It is needed to convert the data from one form to another form to have better predictions. Explore various techniques to transform the data for better model performance. you can go through this link:

<https://360digitmg.com/mindmap-data-science>

1. Prepare the dataset by performing the preprocessing techniques, to have the data which improves model performance.



**Hints:**

For each assignment, the solution should be submitted in the below format

1. Work on each feature to create a data dictionary as displayed in the image displayed below:
2. Hint: Refer to the calories\_consumed.csv dataset
3. Research and perform all possible steps for obtaining the solution.
4. All the codes (executable programs) should execute without errors.
5. Code modularization should be followed.
6. Each line of code should have comments explaining the logic and why you are using that function.

import pandas as pd

# Read data into Python

df = pd.read\_csv(r"C:/Users/Lenovo/Downloads/Study material/EDA/InClass\_DataPreprocessing\_datasets/calories\_consumed.csv")

import scipy.stats as stats

import pylab

import seaborn as sns

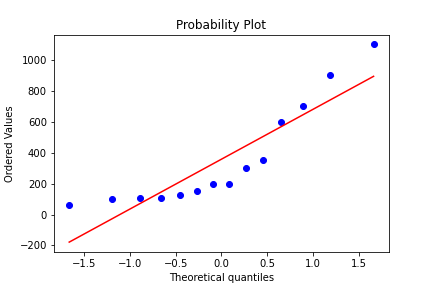
import matplotlib.pyplot as plt

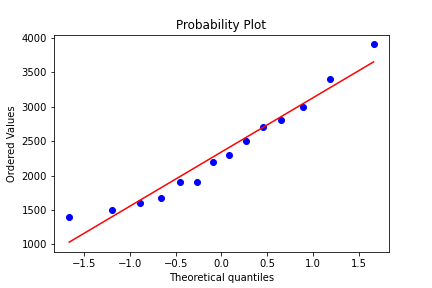
df.info()

# Checking whether data is normally distributed

stats.probplot(df['Weight gained (grams)'], dist = "norm", plot = pylab)

stats.probplot(df['Calories Consumed'], dist = "norm", plot = pylab)

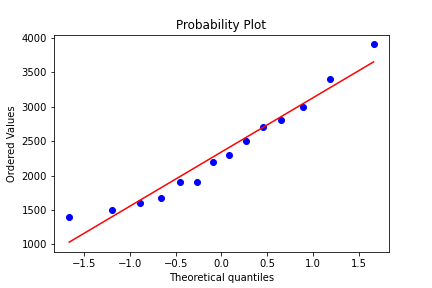




import numpy as np

# Transformation to make workex variable normal

stats.probplot(np.log(df['Weight gained (grams)']), dist = "norm", plot = pylab)



# creating axes to draw plots

fig, ax = plt.subplots(1, 2)

# Plotting the original data (non-normal) and fitted data (normal)

sns.distplot(df['Weight gained (grams)'], hist = False, kde = True,

kde\_kws = {'shade': True, 'linewidth': 2},

label = "Non-Normal", color = "green", ax = ax[0])

sns.distplot(fitted\_data, hist = True, kde = True,

kde\_kws = {'shade': True, 'linewidth': 2},

label = "Normal", color = "green", ax = ax[1])

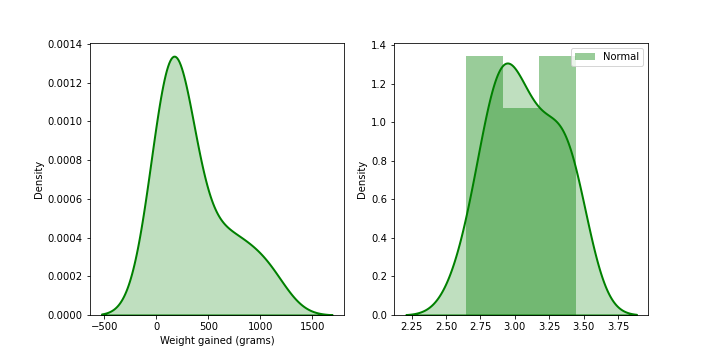
# adding legends to the subplots

plt.legend(loc = "upper right")

# rescaling the subplots

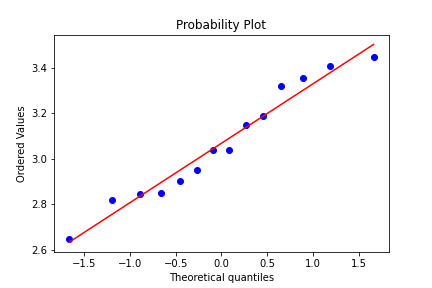
fig.set\_figheight(5)

fig.set\_figwidth(10)



# Transformed data

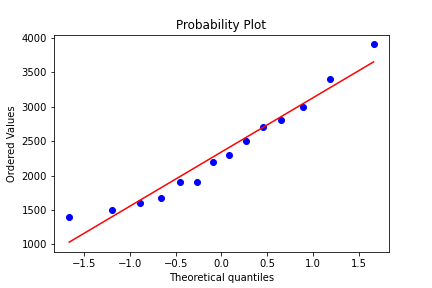
prob = stats.probplot(fitted\_data, dist = stats.norm, plot = pylab)



# Yeo-Johnson Transform

# Original data

prob = stats.probplot(df['Calories Consumed'], dist = stats.norm, plot = pylab)



from feature\_engine import transformation

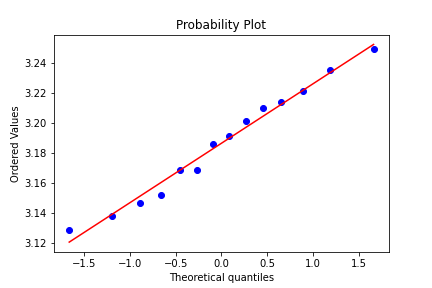
# Set up the variable transformer

tf = transformation.YeoJohnsonTransformer(variables = 'Calories Consumed')

df = tf.fit\_transform(df)

# Transformed data

prob = stats.probplot(df['Calories Consumed'], dist = stats.norm, plot = pylab)



|  |  |  |  |
| --- | --- | --- | --- |
| **Name of Feature** | **Description** | **Type** | **Relevance** |
| **ID** |  | **Quantitative/ Nominal** | **Irrelevant (ID does not provide useful information)** |
| Weight gained (grams) | The amount of weight gained by the subject in grams during the study period. | Quantitative | Relevant |
| Calories Consumed | The total number of calories consumed by the subject during the study period. | Quantitative | Relevant |