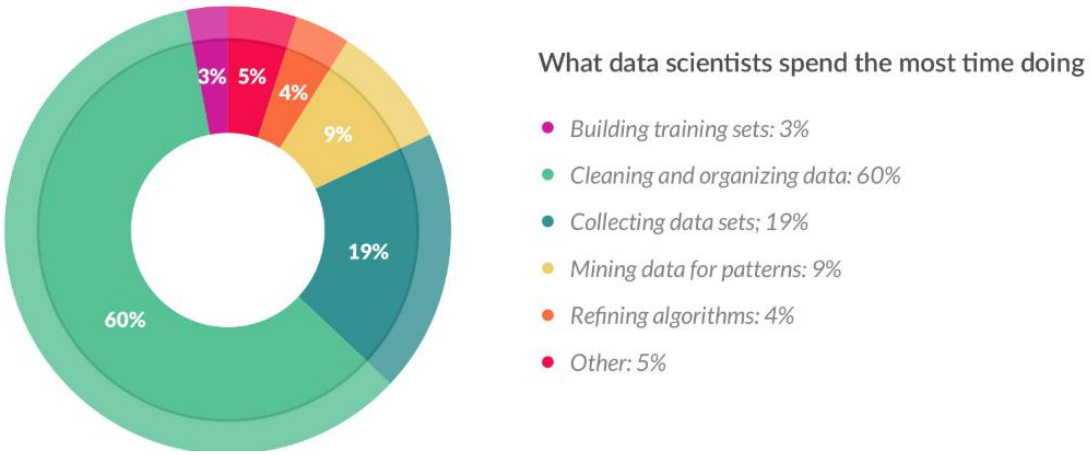
## DATA MANAGEMENT

### Data Cleaning, Wrangling and Standardization

Data collection in any organization is mostly done by non-experts. This leads to data being entered in various formats which leaves us with messy data. The preprocessing of dataset before using it to execute any process or evaluation is very important. Data cleaning plays a very important part although it can seem tedious.




From the pie chart above, cleaning and organization of data is very important and has the highest weightage among all other steps.

First things first, we look into the dataset. The head(10) command is used to look at the first 10 records of our dataset.

```
import pandas as pd

data = pd.read_csv(r"C:\Users\User\Desktop\Capstone Project\health.csv")
print(data.head(10))
```

Output:

```
  Class   Class Language  ...  ZIP code (state)  ZIP code (zip)
0   APH          English  ...               NaN         78741.0
1  PCHW          Spanish  ...               NaN         78758.0
2  ARCF          English  ...               NaN         78753.0
3  PCHW          Spanish  ...               NaN         78617.0
4  ARCF          English  ...               NaN         78720.0
5  PCHW          Spanish  ...               NaN             NaN
6  PCHW          Spanish  ...               NaN             NaN
7   APH  Chinese/English  ...               NaN             NaN
8  PCHW          Spanish  ...               NaN             NaN
9  ARCF          English  ...               NaN             NaN

[10 rows x 25 columns]
```

From the output of the first look into the dataset, we can see that it is a rather messy dataset. Thus, cleaning and preparation of the dataset must be carried out.

Next, we get rid of unwanted columns. The columns that are selected to be removed are redundant and unnecessary for our project.

```
#remove unwanted columns
to_drop = ['Class', 'Class Language', 'Year', 'Insurance Category', 'Medical Home Category',
           'Race/Ethnicity', 'Education Level','Previous Diabetes Education (Yes/No)',
           'Diabetes Knowledge','Food Measurement',
           'Carbohydrate Counting','Problem Area in Diabetes (PAID) Scale Score',
           'ZIP code (address)', 'ZIP code (city)','ZIP code (state)', 'ZIP code (zip)']

df.drop(to_drop, inplace=True, axis=1)
print(df.head(10))
```

Output:

```
[10 rows x 25 columns]
    Age Gender  ... Sugar-Sweetened Beverage Consumption  Exercise
0  47.0      F  ...                                   0     1 day
1  35.0      F  ...                                   2     1 day
2  58.0      F  ...                                 NaN    0 days
3  41.0      F  ...                                   2    4 days
4  56.0      M  ...                                   1    0 days
5  45.0    NaN  ...                                 NaN       NaN
6  44.0      F  ...                                 NaN       NaN
7  73.0    NaN  ...                                   1  >=5 days
8  40.0      F  ...                                 NaN       NaN
9  81.0      F  ...                                   1    0 days

[10 rows x 9 columns]
```

From 25 columns, it became 9 columns after the removal of unwanted columns.

This function calculates the number of missing values in each column.

```
#check number of missing values in each column
print(df.isnull().sum())
```

Output:

```
Age                                          32
Gender                                       37
Diabetes Status (Yes/No)                     30
Heart Disease (Yes/No)                      100
High Blood Pressure (Yes/No)                 93
Tobacco Use (Yes/No)                        124
Fruits & Vegetable Consumption               52
Sugar-Sweetened Beverage Consumption         53
Exercise                                     78
dtype: int64
```

In the dataset, many rows had NaN values. Since our dataset is big enough, we can remove rows with NaN values without risking the main outcome of our project.

```
21  #drop any row that has NaN values
22  print(df.dropna())
```

Initially we had 1688 rows in the dataset. After removing rows, we now have 1369 rows.

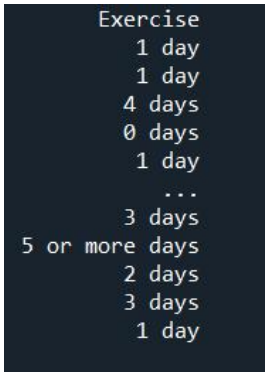
Output:

```
[1369 rows x 9 columns]
```

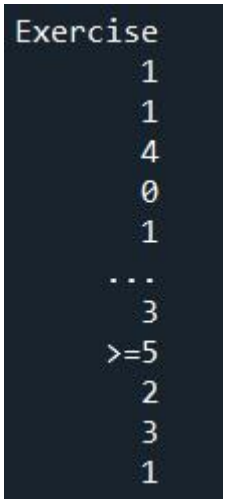
```
24  #standardize values
25  df['Exercise'] = df['Exercise'].replace( # Changing the format of the string
26                                      to_replace=['0 days','1 day', '2 days',
27                                                  '3 days', '4 days', '5 or more days',],
28                                      value=['0','1', '2', '3', '4', '>=5'])
29
30  df['Sugar-Sweetened Beverage Consumption'] = df['Sugar-Sweetened Beverage Consumption'].replace(to_replace=['
31                                              value=['>=3'])
32
33  print(df)
34
```

The code above is used to standardize the dataset. The dataset was found to have many unstandardized inputs.

Before standardizing the ‘Exercise’ column:




After standardizing the ‘Exercise’ column:




**Finding correlations and determining important variables**

Correlation

The term "correlation" refers to a relationship between quantities, or interaction between them. Correlation is often the first step towards understanding these relationships, and then building better business and statistical models. Correlation or correlation coefficient numerically captures the association between two variables.

Variable selection

In the case of a dataset a variable simply means a column. When we get some dataset, not every column (variable) will necessarily have an effect on the output variable. If we add these irrelevant variables in the model, it will just make the model worst. High correlation variables are more linearly dependent, and thus have nearly the same effect on the dependent variable. And if there is a high correlation between two variables, we can remove one of the two features. Apart from that we could also remove variables that have no correlations with any of the variables or less correlation than that are intended.

Cramer's V

Cramér's V is a number between 0 and 1 that indicates how strongly two categorical variables are associated. A measure that does indicate the strength of the association is Cramér's V, defined as

$$\phi_c = \sqrt{\frac{\chi^2}{N(k-1)}}$$

Where

- ϕc denotes Cramér's V;
- χ2 is the Pearson chi-square statistic from the aforementioned test;
- N is the sample size involved in the test and
- k is the lesser number of categories of either variable.

Chi-Square Test

The purpose of the Chi-square test is to check how likely an observed distribution is due to chance. It is often called a "goodness of fit" statistic, as it calculates how well the observed data distribution matches with the predicted distribution if the variables are independent. The chi-square independence test is a procedure for testing if two categorical variables are related in some population.

The Chi-Squared test is a statistical hypothesis test that assumes (the null hypothesis) that the observed frequencies for a categorical variable match the expected frequencies for the categorical variable. The test calculates a statistic that has a chi-squared distribution. The Chi-Squared test does this for a contingency table, first calculating the expected frequencies for the groups, then determining whether the division of the groups, called the observed frequencies, matches the expected frequencies. The result of the test is a test statistic that has a chi-squared distribution and can be interpreted to reject or fail to reject the assumption or null hypothesis that the observed and expected frequencies are the same.

The variables are considered independent if the observed and expected frequencies are similar, that the levels of the variables do not interact, are not dependent. We can interpret the test statistic in the context of the chi-squared distribution with the requisite number of degrees of freedom and p- value.

The p-value and a chosen significance level (alpha), the test can be interpreted as follows:

- If p-value <= alpha: significant result, reject null hypothesis (H0), dependent.
- If p-value > alpha: not significant result, fail to reject null hypothesis (H0), independent.

Applying Cramer's V

The Cramer's V is applied to check whether any two variables are highly correlated so that they might have the chance of elimination.

The coding for Cramer's V

```
def cramers_v(x, y):
    confusion_matrix = pd.crosstab(x,y)
    chi2 = stats.chi2_contingency(confusion_matrix)[0]
    n = confusion_matrix.sum().sum()
    phi2 = chi2/n
    r,k = confusion_matrix.shape
    phi2corr = max(0, phi2-((k-1)*(r-1))/(n-1))
    rcorr = r-((r-1)**2)/(n-1)
    kcorr = k-((k-1)**2)/(n-1)
    return np.sqrt(phi2corr/min((kcorr-1),(rcorr-1)))
```

```
print('Finding association')
print('=============================================================================')
testColumns2 = [data['Gender'],data['High Blood Pressure (Yes/No)'],data['Tobacco Use (Yes/No)'],
                data['Fruits & Vegetable Consumption'],data['Sugar-Sweetened Beverage Consumption'],data['Exercise']]
for var2 in testColumns2:
    print(cramers_v(var2,data["Age"]))
```

The correlation coefficient between other variables with Age variable

Gender
High blood pressure
Tobacco use
Fruit and vegetable consumption
Sugar-Sweetened Beverage Consumption
Exercise

```
0.09791141965974116
0.4484777042516069
0.0
0.0700323336275799
0.04594943964389152
0.0476184791936856
```

There are no high correlations between the potential predictive variables.

The correlation coefficient between other variables with Gender variable

Age
High blood pressure
Tobacco use
Fruit and vegetable consumption
Sugar-Sweetened Beverage Consumption
Exercise

```
0.09791141965974116
0.1002921870216487
0.1180546960764078
0.029362528208722177
0.043748827765085O7
0.042565134714856 29
```

There are no high correlations between the potential predictive variables.

The correlation coefficient between other variables with High blood pressure variable

Age
Gender
Tobacco use
Fruit and vegetable consumption
Sugar-Sweetened Beverage Consumption
Exercise

```
0.4484777042516069
0.1002921870216487
0.059325979658957 89
0.0
0.08935419089561886
0.0
```

There are no high correlations between the potential predictive variables.

The correlation coefficient between other variables with Tobacco use variable

Age
Gender
High blood pressure
Fruit and vegetable consumption
Sugar-Sweetened Beverage Consumption
Exercise

```
0.0
0.1180546960764078
0.059325979658957 89
0.08121848434898575
0.1283311028416905
0.0
```

There are no high correlations between the potential predictive variables.

The correlation coefficient between other variables with Fruits & Vegetable Consumption variable

Age
Gender
High blood pressure
Tobacco use
Sugar-Sweetened Beverage Consumption
Exercise

```
0.0700323336275799
0.029362528208722195
0.0
0.08121848434898575
0.17645571149429132
0.15004364552506538
```

There are no high correlations between the potential predictive variables.

The correlation coefficient between other variables with Sugar-Sweetened Beverage Consumption variable

```
0.045949439643891 52
0.043748827765085055
0.08935419089561886
0.1283311028416905
0.1764557114942913
0.0954307187677095
```

Age
Gender
High blood pressure
Tobacco use
Fruits & Vegetable Consumption
Exercise

There are no high correlations between the potential predictive variables.

The correlation coefficient between other variables with Sugar-Sweetened Beverage Consumption variable

Age
Gender
High blood pressure
Tobacco use
Fruits & Vegetable Consumption
Exercise

```
0.0476184791936857
0.04256513471485628
0.0
0.0
0.15004364552506538
0.0954307187677095
>>>
```

There are no high correlations between the potential predictive variables.

There is no need for eliminating any variables as there are no higher correlations between the potential predictive variables.

Applying Chi-Square Test

Next, we applied chi square test to see which variables are important and not important in predictive task.

The code for chi-square test

```
class ChiSquare:
    def __init__(self, dataframe):
        self.data = dataframe
        self.p = None #P-Value
        self.chi2 = None #Chi Test Statistic
        self.dof = None

        self.dfObserved = None
        self.dfExpected = None

    def _print_chisquare_result(self, colX, alpha):
        result = ""
        print("The p value is ")
        print(self.p)
        if self.p<alpha:
            result="{0} is IMPORTANT for Prediction".format(colX)
        else:
            result="{0} is NOT an important predictor.".format(colX)
        print(result)
```

```
    def TestIndependence(self,colX,colY, alpha=0.05):
        X = self.data[colX].astype(str)
        Y = self.data[colY].astype(str)

        self.dfObserved = pd.crosstab(Y,X)
        chi2, p, dof, expected = stats.chi2_contingency(self.dfObserved.values)
        self.p = p
        self.chi2 = chi2
        self.dof = dof

        self.dfExpected = pd.DataFrame(expected, columns=self.dfObserved.columns,

        self._print_chisquare_result(colX,alpha)

data['dummyCat'] = np.random.choice([0, 1], size=(len(data),), p=[0.5, 0.5])
cT = ChiSquare(data)
testColumns = ['Age','Gender','High Blood Pressure (Yes/No)','Tobacco Use (Yes/No)
               'Sugar-Sweetened Beverage Consumption','Exercise']
for var in testColumns:
    cT.TestIndependence(colX=var,colY="Heart Disease (Yes/No)" )
```

Result

```
Finding predictive variables
==============================================================================
The p value is
1.1594351022782816-06
Age is IMPORTANT for Prediction
The p value is
0.016296565214755476
Gender is IMPORTANT for Prediction
The p value is
4.343515712482093e-24
High Blood Pressure (Yes/No) is IMPORTANT for Prediction
The p value is
0.7559426274291035
Tobacco Use (Yes/No) is NOT an important predictor.
The p value is
0.0011772233466563985
Fruits & Vegetable Consumption is IMPORTANT for Prediction
The p value is
0.1438915802013509
Sugar-Sweetened Beverage Consumption is NOT an important predictor.
The p value is
0.7314697851982384
Exercise is NOT an important predictor.
```

Analyzing the result.

From the chi-square test we found that ‘Tobacco use’, ‘Sugar-Sweetened Beverage Consumption’ and ‘Exercise’ have higher p-value which makes them less important.

Based on our domain research we decided to keep ‘Tobacco use’ and ‘Exercise’. According to (https://www.cdc.gov/heartdisease/index.htm) Centers for Disease Control and Prevention, they have created a quality improvement tool that related to tobacco usage named ‘The Tobacco Cessation Change Package (TCCP)’ which shows the important of that variable. Apart from that, the same organization promote physical activity as to reduce heart disease. Considering such knowledge, we do not want to eliminate ‘Exercise’ variable as well.

**Data visualization, Summary and Statistics**

Data Visualization

**Data visualization** is the graphical representation of information and **data**. By using visual elements like charts, graphs, and maps, **data visualization** tools provide an accessible way to see and understand trends, outliers, and patterns in **data**.

Matplotlib.Pyplot
**matplotlib** .**pyplot** is a collection of **command** style **functions** that make **matplotlib** work like MATLAB. Each **pyplot function** makes some change to a figure: e.g., creates a figure, creates a **plotting** area in a figure, **plots** some lines in a **plotting** area, decorates the **plot** with labels, etc.

Import Pandas As Pd
**pandas** is a library that you **install**, so it's local to your Python installation. **import pandas as pd**. Simply **imports** the library the current namespace, but rather than using the name **pandas** , it's instructed to use the name **pd** instead.

df = pd.read_csv('C:/Users/Zaiti/Desktop/modified_health.csv')
The Python **Pandas read_csv** function is used to read or load data from CSV files into Pandas DataFrames, and how to write DataFrames back to CSV files post **analysis**.

df['Fruits & Vegetable Consumption'].value_counts().plot(kind='bar')
Write a Pandas **program** to create a bar **plot** of the '**value_counts**' for the ... diamonds DataFrame:") diamonds.cut.**value_counts().plot(kind='bar')**.

plt.show()
required to show the plot

```
In [6]: import matplotlib.pyplot as plt

In [16]: import pandas as pd

In [19]: df = pd.read_csv('C:/Users/Zaiti/Desktop/modified_health.csv')

In [21]: df['Gender'].value_counts().plot(kind='bar')
         plt.show()
```

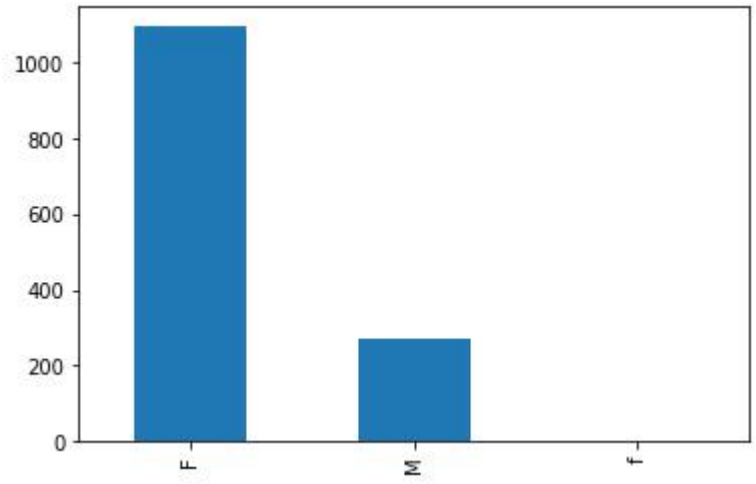



Figure1 shows majority female take the health test rather than male

```
In [26]: import matplotlib.pyplot as plt

In [27]: import pandas as pd

In [28]: df = pd.read_csv('C:/Users/Zaiti/Desktop/modified_health.csv')

In [31]: df['Diabetes Status (Yes/No)'].value_counts().plot(kind='area')
         plt.show()
```

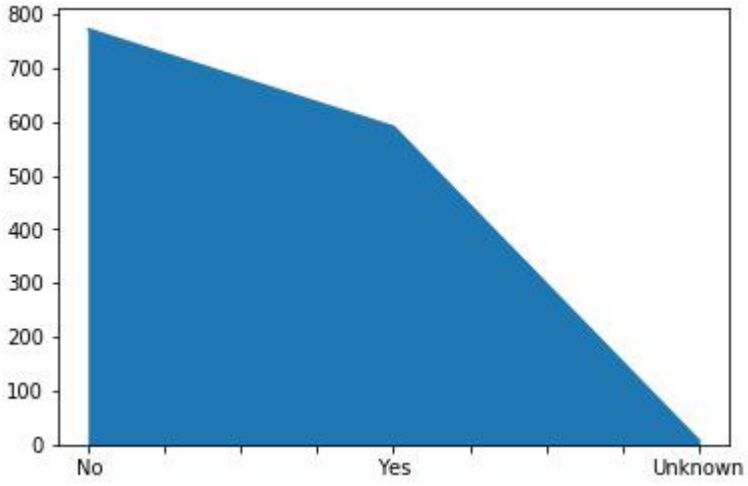



Figure 2 shows majority of the people had no diabetes status rather than the people that has diabetes status and less than 50 people doesn't take diabetes checkup.

```
In [44]: df['Heart Disease (Yes/No)'].value_counts().plot(kind='pie')
         plt.show()
```

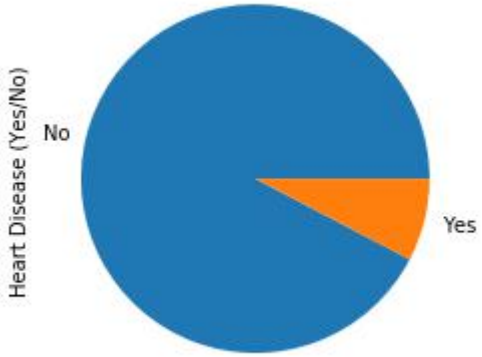



Figure 3 shows the least amount of people that have heart disease rather than people that doesn't have heart disease.

```
In [51]: df['High Blood Pressure (Yes/No)'].value_counts().plot(kind='bar')
         plt.show()
```

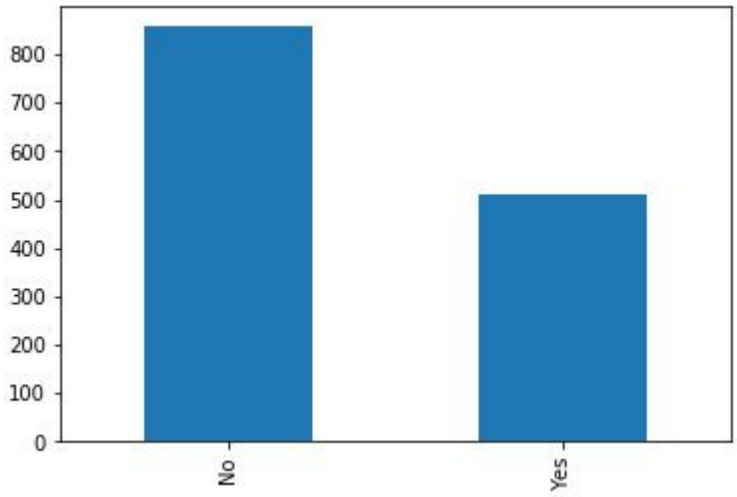



Figure 4 shows more than 500 people have a high blood pressure problem and more than 800 people doesn't have high blood pressure.

```
In [55]: df['Tobacco Use (Yes/No)'].value_counts().plot(kind='bar')
         plt.show()
```

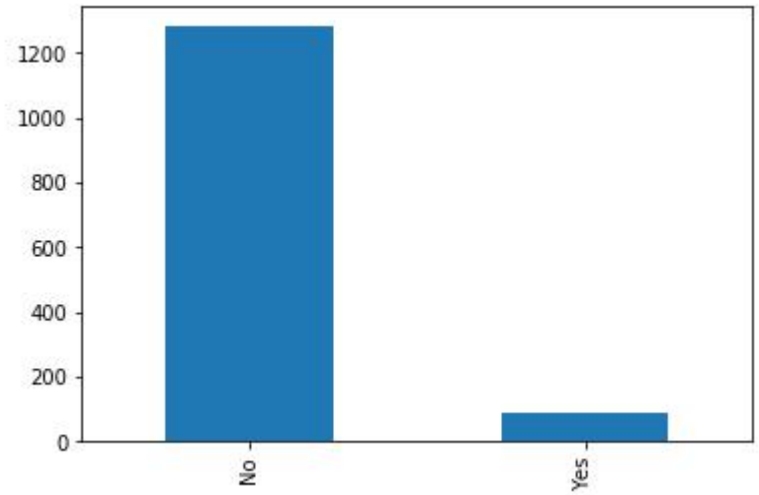



Figure 5 shows that the number of people that use tobacco is 10 times lesser than the people that didn't use tobacco.

```
In [61]: df['Fruits & Vegetable Consumption'].value_counts().plot(kind='bar')
         plt.show()
```

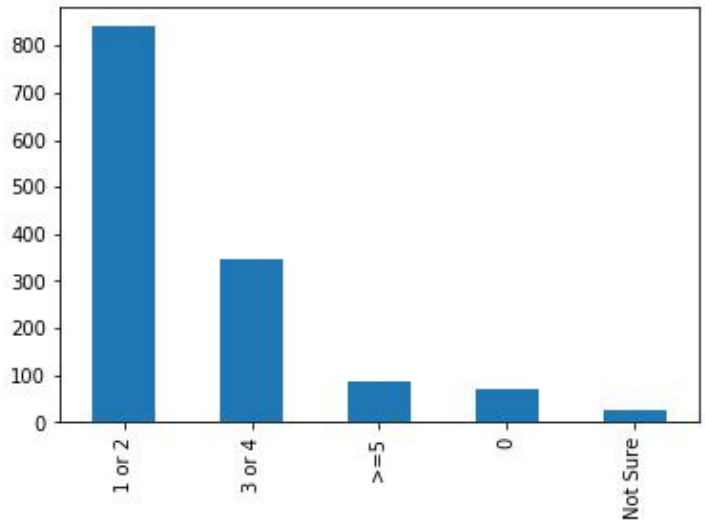



Figure 6 shows that there are still have least of  people that doesn't sure about their fruit and vegetables intake in daily life.

```
In [68]: df['Exercise'].value_counts().plot(kind='bar')
         plt.show()
```

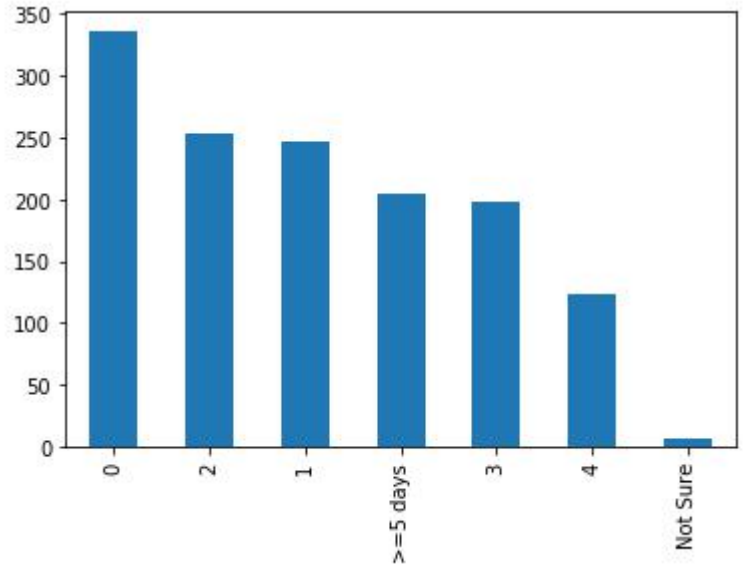



Figure 7 shows that majority of people out there don't exercise in their daily life.

Data Summary and statistics

Statistics is the science of collecting data and analyzing them to infer proportions (sample) that are representative of the population. In other words, statistics is interpreting data in order to make predictions for the population.

**Commonly Used Measures**

1. Measures of Central Tendency
2. Measures of Dispersion (or Variability)

**Measures of Central Tendency**

A Measure of Central Tendency is a one number summary of the data that typically describes the center of the data. These one number summary is of three types.

1. **Mean :** Mean is defined as the ratio of the sum of all the observations in the data to the total number of observations. This is also known as Average. Thus mean is a number around which the entire data set is spread.

2. **Median :** Median is the point which divides the entire data into two equal halves. One-half of the data is less than the median, and the other half is greater than the same. Median is calculated by first arranging the data in either ascending or descending order.

- If the number of observations are odd, median is given by the middle observation in the sorted form.
- If the number of observations are even, median is given by the mean of the two middle observation in the sorted form.

An important point to note that the order of the data (ascending or descending) does not effect the median.

**3. Mode :** Mode is the number which has the maximum frequency in the entire data set, or in other words,mode is the number that appears the maximum number of times. A data can have one or more than one mode.

- If there is only one number that appears maximum number of times, the data has one mode, and is called **Uni-modal**.
- If there are two numbers that appear maximum number of times, the data has two modes, and is called **Bi-modal**.
- If there are more than two numbers that appear maximum number of times, the data has more than two modes, and is called **Multi-modal**.

*Example to compute the Measures of Central Tendency*

Consider the following data points.

**17, 16, 21, 18, 15, 17, 21, 19, 11, 23**

- Mean — Mean is calculated as

- Median — To calculate Median, lets arrange the data in ascending order.

11, 15, 16, 17, 17, 18, 19, 21, 21, 23

Since the number of observations is even (10), median is given by the average of the two middle observations (5th and 6th here).

- Mode — Mode is given by the number that occurs maximum number of times. Here, 17 and 21 both occur twice. Hence, this is a Bimodal data and the modes are 17 and 21.

*Note-*

1. Since Median and Mode does not take all the data points for calculations, these are robust to outliers, i.e. these are not effected by outliers.

2. At the same time, Mean shifts towards the outlier as it considers all the data points. This means if the outlier is big, mean overestimates the data and if it is small, the data is underestimated.

3. If the distribution is symmetrical, Mean = Median = Mode. Normal distribution is an example.

**Measures of Dispersion (or Variability)**

Measures of Dispersion describes the spread of the data around the central value (or the Measures of Central Tendency)

**1. Variance** — Variance measures how far are data points spread out from the mean. A high variance indicates that data points are spread widely and a small variance indicates that the data points are closer to the mean of the data set. It is calculated as

$$Variance = \frac{1}{N}\sum_{i=1}^{N}\left(X_i - \overline{X}\right)^2$$

**2. Standard Deviation** — The square root of Variance is called the Standard Deviation. It is calculated as

$$Std\ Deviation = \sqrt{Variance} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(X_i - \overline{X}\right)^2}$$

```
In [12]: import pandas as pd
```

```
In [13]: data=pd.read_csv(r"C:\Users\Zaiti\Desktop\modified_health.csv")
```

```
In [15]: print ('Mean Age: ' + str(mean1))
         print ('Sum of Age: ' + str(sum1))
         print ('Max Age: ' + str(max1))
         print ('Min Age: ' + str(min1))
         print ('Count of Age: ' + str(count1))
         print ('Median Age: ' + str(median1))
         print ('Std of Age: ' + str(std1))
         print ('Var of Age: ' + str(var1))
```

```
Mean Age: 49.18699780861943
Sum of Age: 67337
Max Age: 93
Min Age: 16
Count of Age: 1369
Median Age: 49.0
Std of Age: 15.16358481704277
Var of Age: 229.93430450365003
```

```
In [14]: mean1 = data['Age'].mean()
         sum1 = data['Age'].sum()
         max1 = data['Age'].max()
         min1 = data['Age'].min()
         count1 = data['Age'].count()
         median1 = data['Age'].median()
         std1 = data['Age'].std()
         var1 = data['Age'].var()
```

**Finding outliers and handling them**

Outliers are data points that are often described as the unusual values in a dataset. The outliers can be a result of a mistake during data collection or it can be just an indication of variance in our data. Unfortunately, they can cause problems in statistical procedures if they are not handled properly.

The most common causes of outliers on a dataset may be :

- data entry errors (human errors)
- Measurement errors (instrument errors)
- Experimental errors (data extraction or experiment planning/executing errors)
- Intentional (dummy outliers made to test detection methods)
- Data processing errors (data manipulation or data set unintended mutations)
- Sampling errors (extracting or mixing data from wrong or various sources)
- Natural (not an error, novelties in data)

In the process of producing, collecting, processing and analyzing data, outliers can come from many sources and hide in many dimensions.

Finding outliers

We will first load the dataset and separate out the features and targets.

```
import pandas as pd

df = pd.read_csv(r"D:/notes/sem 4/DS/project/modified_health.csv")

print(df)
```

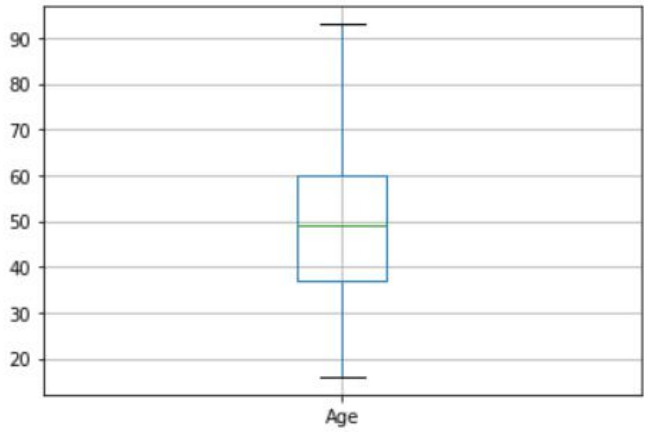
```
      Age Gender Diabetes Status (Yes/No) Heart Disease (Yes/No) High Blood Pressure (Yes/No) Tobacco Use (Yes/No) Fruits & Vegetable Consumption  Exercise
0      47      F                      Yes                     No                         No                    No                          3 or 4         1
1      35      F                       No                     No                         No                    No                          1 or 2         1
2      41      F                       No                     No                        Yes                    No                          1 or 2         4
3      56      M                       No                     No                         No                    No                          1 or 2         0
4      34      F                       No                     No                         No                    No                          1 or 2         1
...   ...    ...                      ...                    ...                        ...                   ...                             ...       ...
1364   38      F                      Yes                     No                        Yes                    No                          1 or 2         3
1365   43      F                       No                     No                         No                    No                          1 or 2  >=5 days
1366   41      M                      Yes                     No                         No                    No                          3 or 4         2
1367   27      F                      Yes                     No                         No                    No                          3 or 4         3
1368   58      F                       No                     No                         No                    No                          1 or 2         1

[1369 rows x 8 columns]
```

The output shows the columns in the dataset.

To visualize the outliers, we use box plot using pandas. Box Plot is the visual representation of the depicting groups of numerical data through their quartiles. Boxplot is also a way to detect the outlier in data set. It captures the summary of the data efficiently with a simple box and whiskers and allows us to compare easily across groups. Boxplot summarizes a sample data using 25th, 50th and 75th percentiles. These percentiles are also known as the lower quartile, median and upper quartile.

One way to plot boxplot using pandas dataframe is to use boxplot() function that is part of pandas library.

```
In [1]: import pandas as pd

In [2]: import numpy as np

In [3]: import matplotlib.pyplot as plt

In [4]: from IPython import get_ipython

In [5]: get_ipython().run_line_magic('matplotlib', 'inline')

In [14]: df = pd.read_csv(r"D:/notes/sem 4/DS/project/modified_health.csv")

In [15]: df.boxplot(return_type='dict')
         plt.plot()
```




The figure shows the boxplot of attribute “Age”

From the figure above, we can know these statistical things:

- The bottom horizontal line of box plot is minimum value.
- First black horizontal line of rectangle shape of the box plot is First quartile of 25%
- Second black horizontal line of rectangle shape of the box plot is Second quartile or 50% or median.
- Third black horizontal line of rectangle shape of the box plot is third quartile or 75%
- Top black horizontal line of rectangle shape of the box plot is maximum value.
- There is no outlier detected, thus there is no outlier to be handled.

**Data management plan**

Goal

The goal of this data management is to describes the data type, to visualise the data and to improvise the structure to make it more usable for the research project.

Data collection

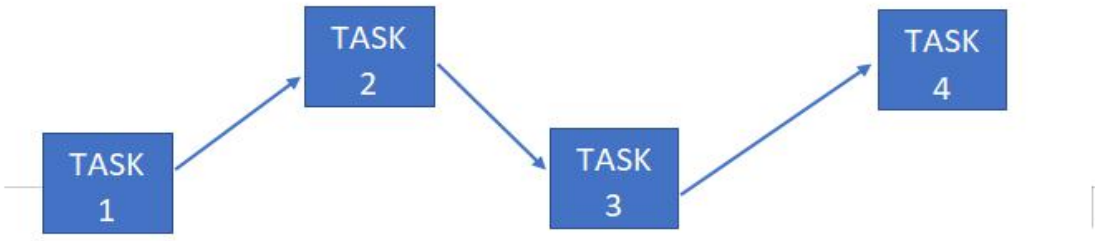
The dataset used of this project is "austin-public-health-diabetes-self-management-education-participant-demographics-2015-2017-1" from data.gov. This dataset contains 25 variables.

Data presentation

Many visualisation techniques were used in the data management plan in order to present the data in a way other can understand. We also performed some statistical analysis to understand the data better.

Data management tasks

| Task code | Task | Dependency |
|---|---|---|
| T1 | Data cleaning | - |
| T2 | Data wrangling | - |
| T3 | Data Standardization | - |
| T4 | Data correlation | T1, T2, T3 |
| T5 | Choosing important prediction attributes | T1, T2, T3 |
| T6 | Data visualisation | T1, T2, T3, T4, T5 |
| T7 | Summary and statistic | T1, T2, T3, T4, T5 |
| T8 | Check for and handle outliers | T1, T2, T3, T4, T5, T6, T7 |
| T9 | Handle imbalanced data | T1, T2, T3, T4, T5, T6, T7 |
| T10 | Documenting Data Management | T1, T2, T3, T4, T5, T6, T7 |



**References**


1. https://towardsdatascience.com/3-steps-to-a-clean-dataset-with-pandas-2b80ef0c81ae
2. https://github.com/mramshaw/Data-Cleaning
3. https://www.kdnuggets.com/2019/11/data-cleaning-preprocessing-beginners.html
4. https://medium.com/@rrfd/cleaning-and-prepping-data-with-python-for-data-science-best-practices-and-helpful-packages-af1edfbe2a3
5. https://blogs.oracle.com/datascience/introduction-to-correlation
6. https://statisticsbyjim.com/basics/correlations/
7. https://cmdlinetips.com/2019/08/how-to-compute-pearson-and-spearman-correlation-in-python/
8. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6172294/
9. https://www.cdc.gov/heartdisease/index.htm
10. https://www.geeksforgeeks.org/box-plot-visualization-with-pandas-and-seaborn/