BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
(Empowered Autonomous Institute Affiliated to University of Mumbai)
[Knowledge is Nectar]

Department of Computer Engineering

Course – Data Analytics Open Elective

UID	2021300126, 2021300125
Name	Pranay Singhvi, Yoshita Singh
Class and Batch	TE Computer Engineering - Batch A
Date	23-01-2024
Lab #	1
Aim	Perform EDA And Plot Graphs
Objective	Perform EDA such as number of data samples, number of features, number of classes, number of data samples per class, removing missing values, conversion to numbers, using the seaborn library to plot different graphs LIKE Histogram, Pie Chart, Bar Plot, Parallel coordinates, etc.
Theory	Exploratory Data Analysis (EDA) is a crucial process that involves examining and summarizing data to reveal patterns and characteristics. This is achieved through various techniques such as data visualization, summary statistics, and the exploration of univariate, bivariate, and multivariate analyses. The importance of EDA in data science lies in providing analysts with a profound understanding of the dataset. It enables the identification of trends, outliers, and anomalies, guiding informed decisions in data preprocessing, model selection, and hypothesis formulation. EDA ensures that data is well-prepared for further analysis, leading to more accurate and meaningful results. Different types of EDA include univariate analysis, focusing on one variable at a time; bivariate analysis, studying relationships between two variables; and multivariate analysis, considering multiple variables simultaneously to explore complex interactions. Techniques like data visualization, employing charts and graphs, and summary statistics, calculating measures like mean and standard deviation, contribute to a comprehensive EDA approach. Specifically, visual tools like parallel sets, depicting relationships between categories through parallelograms, and parallel coordinates, presenting observations across variables, aid in comparisons and distribution analyses. These techniques enhance the exploration of data patterns, contributing to a more thorough understanding of datasets for effective decision-making in data science projects.
Data Set	https://www.kaggle.com/datasets/trolukovich/nutritional-values-for-common-foods-and-products
Colab File Link	https://colab.research.google.com/drive/1n3kQIQqfKdOaQ9YUpYi53MVSM3f7iFVM?usp=sharing
Purpose	Nutritional Research: Analyzing relationships between different nutrients and exploring trends in food composition for scientific and academic research.

BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
(Empowered Autonomous Institute Affiliated to University of Mumbai)
[Knowledge is Nectar]

Department of Computer Engineering

Code (Part 1)

Repair and Normalize Data with Pandas

1) Dataset of Nutritional Value of common food products

```
[18] nutrients=pd.read_csv("nutrition.csv")
     nutrients.head()
```

	Unnamed: 0	name	serving_size	calories	total_fat	saturated_fat	cholesterol	sodium	choline	folate	...	fat	saturated_fatty_acids	monoun
0	0	Cornstarch	100 g	381	0.1g	NaN	0	9.00 mg	0.4 mg	0.00 mcg	...	0.05 g	0.009 g	
1	1	Nuts, pecans	100 g	691	72g	6.2g	0	0.00 mg	40.5 mg	22.00 mcg	...	71.97 g	6.180 g	
2	2	Eggplant, raw	100 g	25	0.2g	NaN	0	2.00 mg	6.9 mg	22.00 mcg	...	0.18 g	0.034 g	
3	3	Teff, uncooked	100 g	367	2.4g	0.4g	0	12.00 mg	13.1 mg	0	...	2.38 g	0.449 g	
4	4	Sherbet, orange	100 g	144	2g	1.2g	1mg	46.00 mg	7.7 mg	4.00 mcg	...	2.00 g	1.160 g	

5 rows × 77 columns

2) Except for 'name', all variables appear to have numeric values. However, as we see in the datatypes table, most of the columns are of type character.
This happens because in these variables, together with the value, they also show the unit in which it is expressed (for example, it shows 13g of protein for x food. That 'g' (of grams) is a character, so the function that we use to load the data assumes it is of type character.
The idea is to pass those variables from 'character' to 'numeric', and the way we're going to do that is by first finding those unit characters (g, mg, mcg, UI) and removing them, and then just converting the type from data to numeric.

```
[62] name_column = df['name']          #save the name column and drop it from df
     df = df.drop('name', axis=1)      #when we remove characters from units, it will also remove characters from food names

[63] df = df.apply(lambda x: pd.to_numeric(x.astype(str).str.replace('[^0-9.]', '', regex=True), errors='coerce')) #convert all units into numeric values
     df.insert(0, 'name', name_column)
     df.head()
```

	name	Unnamed: 0	serving_size	calories	total_fat	saturated_fat	cholesterol	sodium	choline	folate	...	fat	saturated_fatty_acids	monoun
1	Nuts pecans	1	100	691	72.0	6.2	0	0.0	40.5	22.0	...	71.97	6.180	
3	Teff uncooked	3	100	367	2.4	0.4	0	12.0	13.1	0.0	...	2.38	0.449	
4	Sherbet orange	4	100	144	2.0	1.2	1	46.0	7.7	4.0	...	2.00	1.160	

3) Checking for Null Values and Duplicate Values

```
df.isnull().any()    #No null Values
```

```
Unnamed: 0      False
name            False
serving_size    False
calories        False
total_fat       False
                ...
alcohol         False
ash             False
caffeine        False
theobromine     False
water           False
Length: 77, dtype: bool
```

```
df.duplicated().any()  #NO duplicate values
```

```
False
```

4) Assign each column name its unit of measure. As we eliminate all the units of measurement (UM) in which the values are expressed (point B), we are losing valuable information. Since within the same column, the UM does not vary, we can add that information directly in the column name.

```
df = df.rename(columns={
    'calories': 'calories_100g',
    'serving_size': 'serving_size_g',
    'total_free_saccharides': 'total_free_saccharides_g',
    'saturated_fatty_acids': 'saturated_fatty_acids_g',
    'monounsaturated_fatty_acids': 'monounsaturated_fatty_acids_g',
    'polyunsaturated_fatty_acids': 'polyunsaturated_fatty_acids_g',
    'fatty_acids_total_trans': 'fatty_acids_total_trans_g',
    'water': 'water_g',
    'carbohydrate': 'Carbohydrate_g'
})
```

5) Dropping Columns with same values for all rows

```
df = df.drop(['serving_size','lucopene'], axis=1)          #same value for all
```

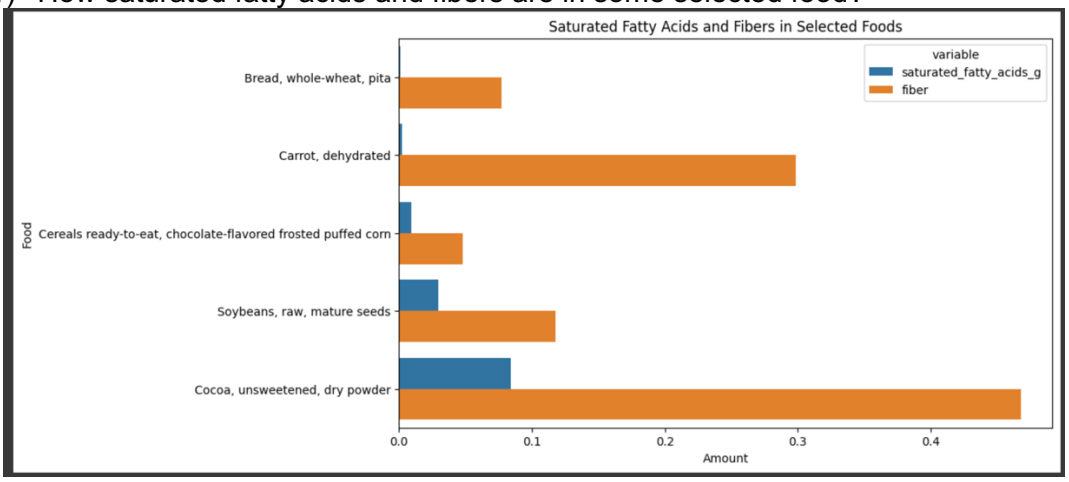
6) Normalizing data with numerical values using the Min Max Scaler

```
# Extracting only the numerical columns for normalization
norma_columns = ['calories_100g', 'total_fat', 'saturated_fat',
       'cholesterol', 'sodium', 'choline', 'folate', 'folic_acid', 'niacin',
       'pantothenic_acid', 'riboflavin', 'thiamin', 'vitamin_a',
       'vitamin_a_rae', 'carotene_alpha', 'carotene_beta',
       'cryptoxanthin_beta', 'lutein_zeaxanthin', 'lucopene', 'vitamin_b12',
       'vitamin_b6', 'vitamin_c', 'vitamin_d', 'vitamin_e', 'tocopherol_alpha',
       'vitamin_k', 'calcium', 'copper', 'irom', 'magnesium', 'manganese',
       'phosphorous', 'potassium', 'selenium', 'zink', 'protein', 'alanine',
       'arginine', 'aspartic_acid', 'cystine', 'glutamic_acid', 'glycine',
       'histidine', 'hydroxyproline', 'isoleucine', 'leucine', 'lysine',
       'methionine', 'phenylalanine', 'proline', 'serine', 'threonine',
       'tryptophan', 'tyrosine', 'valine', 'Carbohydrate_g', 'fiber', 'sugars',
       'fructose', 'galactose', 'glucose', 'lactose', 'maltose', 'sucrose',
       'fat', 'saturated_fatty_acids_g', 'monounsaturated_fatty_acids_g',
       'polyunsaturated_fatty_acids_g', 'fatty_acids_total_trans_g', 'alcohol',
       'ash', 'caffeine', 'theobromine', 'water_g']

scaler = MinMaxScaler()

# Normalizing the numerical columns in the DataFrame
nutrients[norma_columns] = scaler.fit_transform(nutrients[norma_columns])
nutrients
```

Code(Part 2)

Data Visualization and EDA with Pandas and Seaborn

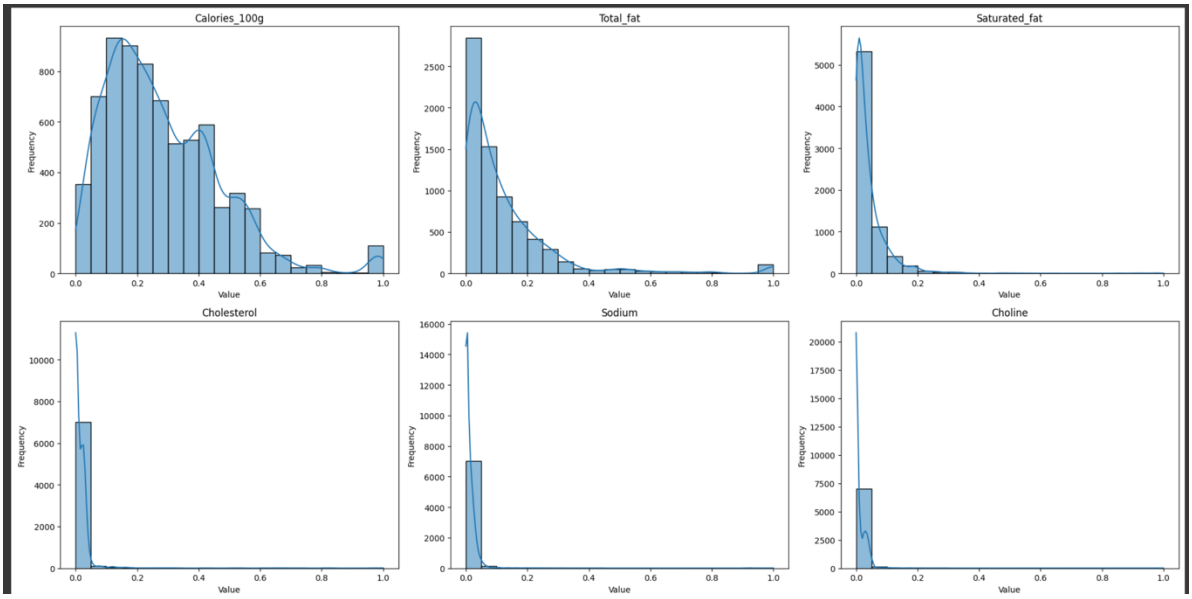
1) How saturated fatty acids and fibers are in some selected food?



From above graph we can conclude that all selected food has more fiber compared to saturated fatty acid.

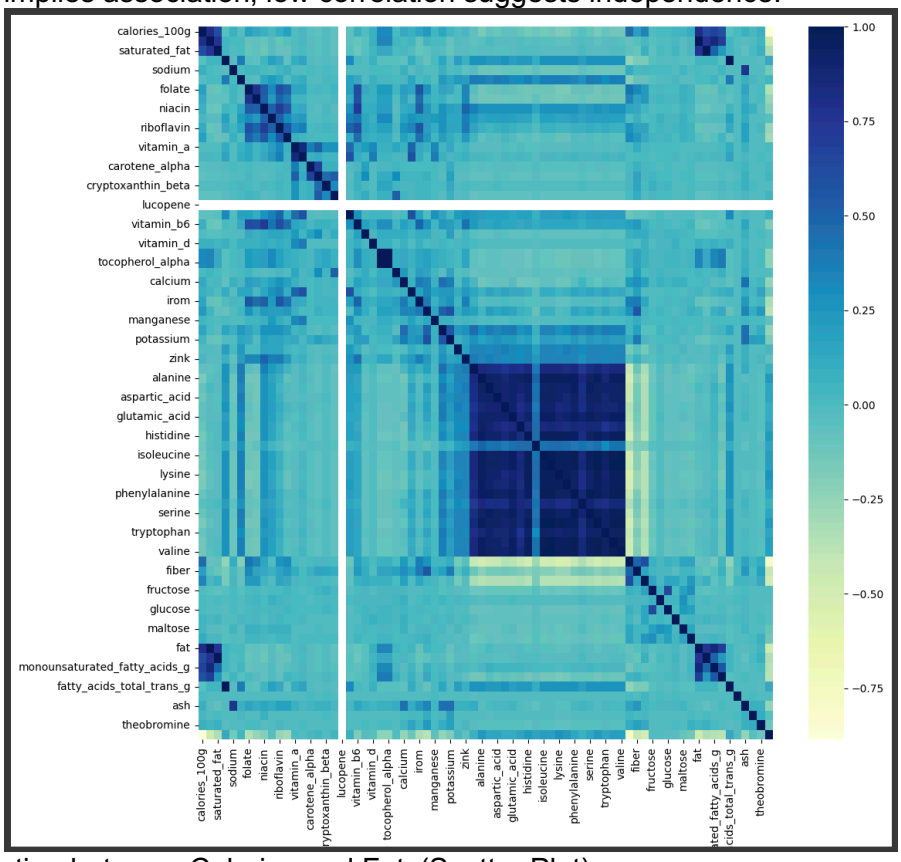
2) Histogram of different attributes

Histograms provide a visual representation of the distribution of each feature. Understanding the distribution of nutritional components helps in identifying patterns, potential outliers, and the overall shape of the data.

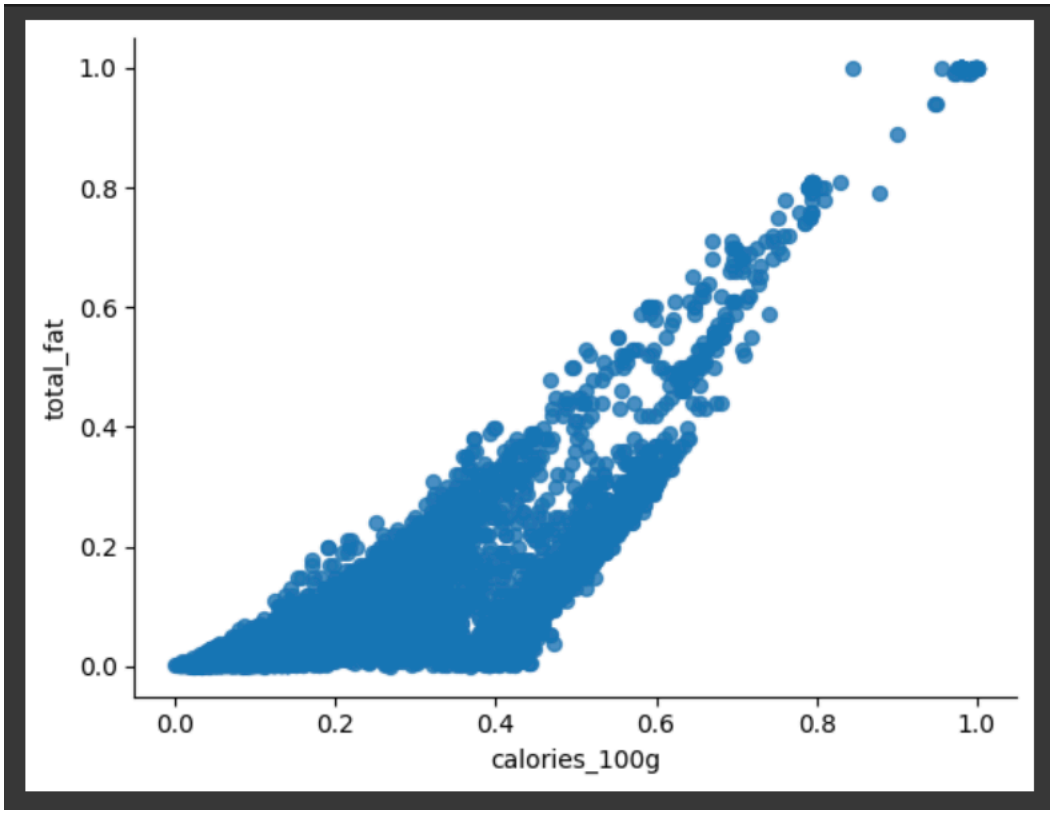


3) Correlation Features

Correlation features reveal relationships between variables, aiding insights. High correlation implies association; low correlation suggests independence.



4) Relation between Calories and Fats(Scatter Plot)

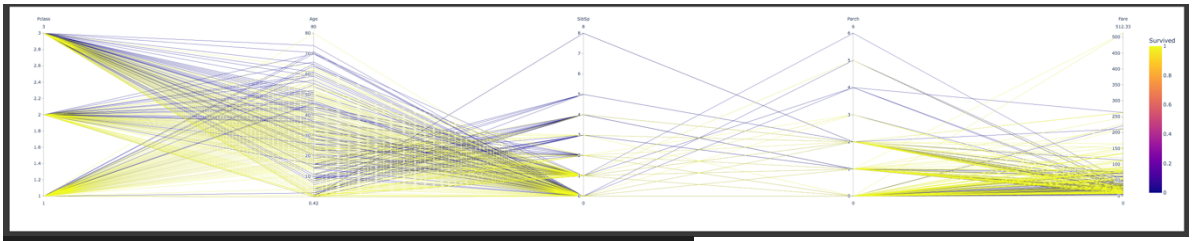


Scatter plots visualize data points' distribution, indicating patterns or trends. X and Y axes display variables, revealing relationships graphically.

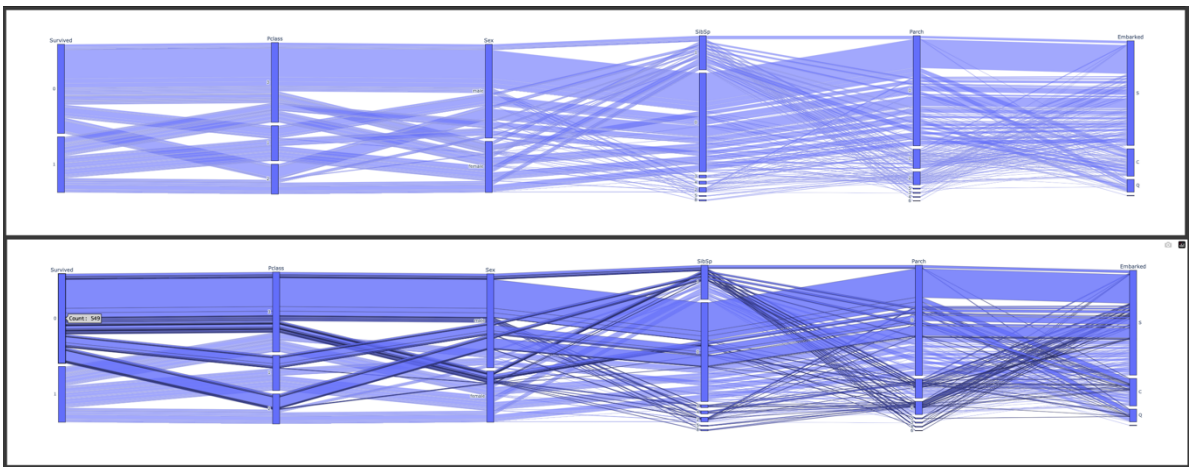
Code(Part 3)

Working on the Titanic dataset for parallel coordinates and parallel sets:

```
xd = px.parallel_coordinates(dataset,dimensions= ["Pclass","Age","SibSp","Parch","Fare"], color= "Survived")
xd.show()
```



```
] flg = px.parallel_categories(dataset)

flg.show()
```

	 From above graph we get to know that there were 549 passengers didn't survive the disaster. This visualization lets you explore and analyze relationships between categorical variables within the dataset, facilitating pattern recognition and insights generation that inform data-driven decision-making.
Conclusion	During the Exploratory Data Analysis (EDA), I investigated essential dataset attributes, like sample count, features, and classes. Ensuring data integrity involved handling missing values and converting categories to numerical formats. Seaborn visualizations, such as histograms and pie charts, offered insights into feature distributions and class compositions. Analyzing class balances and feature interactions aids subsequent modeling efforts. This EDA provides a holistic grasp of the dataset, tackling data quality, exposing patterns, and offering visual insights for informed decision-making in subsequent analyses.