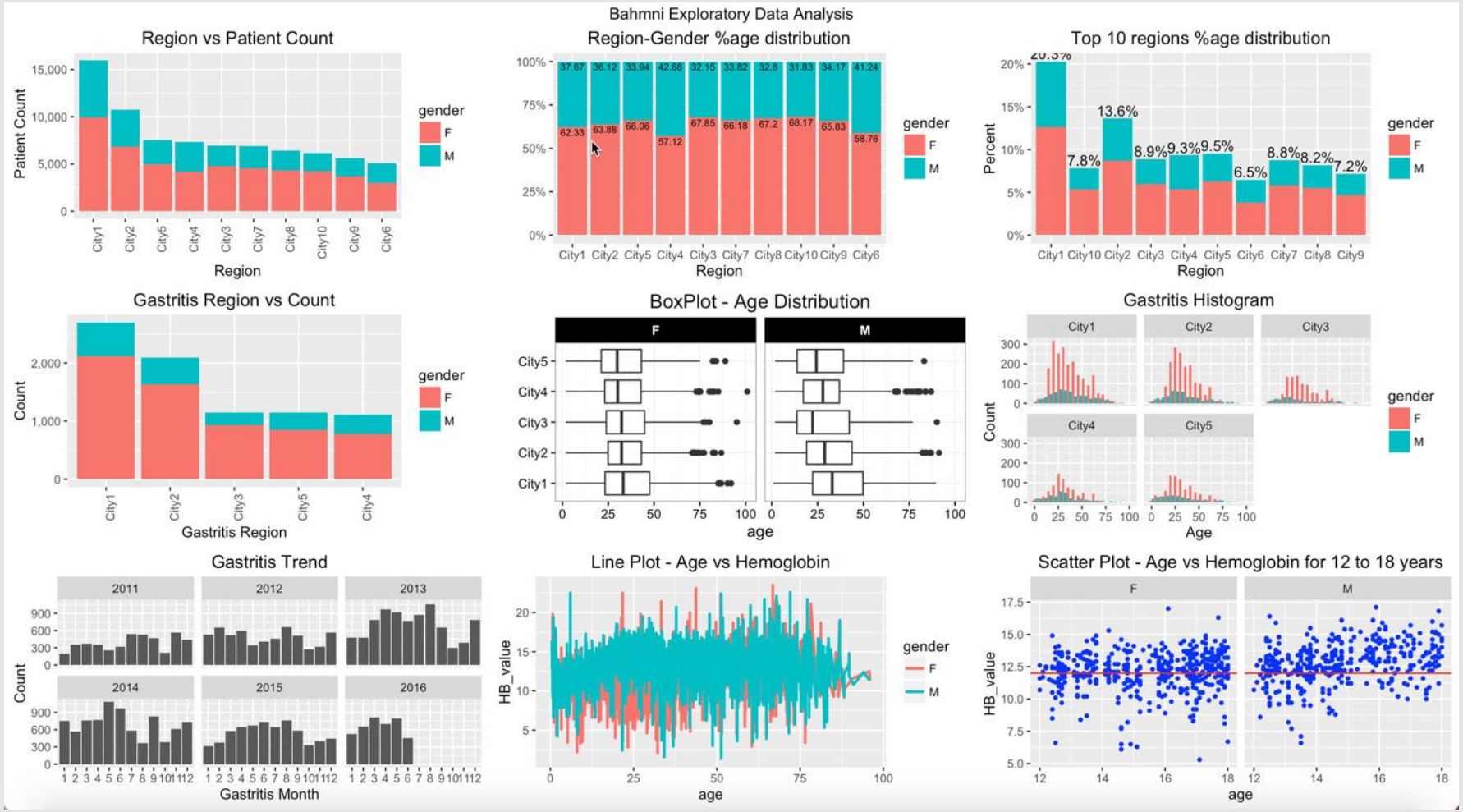
ME 6543
MACHINE LEARNING & DATA ANALYTICS

Syed Hasib Akhter Faruqi
syed-hasib-akhter.faruqui@utsa.edu

Step 1: Explore the Data



Step 2: Handling Missing Values

Column 0	age	years_seniority	income	parking_space	attending_party	entree	pets	emergency_contact
Tony	48	27		1	5	shrimp		Pepper
Donald	67	25	86	10	2	beef		Jane
Henry	69	21	95	6	1	chicken	62	Janet
Janet	62	21	110	3	1	beef		Henry
Nick		17		4				
Bruce	37	14	63		1	veggie		NA
Steve	83		77	7	1	chicken		n/a
Clint	27	9	118	9		shrimp	3	None
Wanda	19	7	52	2	2	shrimp		empty
Natasha	26	4	162	5	3			_
Carol		3	127	11	1	veggie	1	""""
Mandy	44	2	68	8	1	chicken		null

Source: Azure AI Library

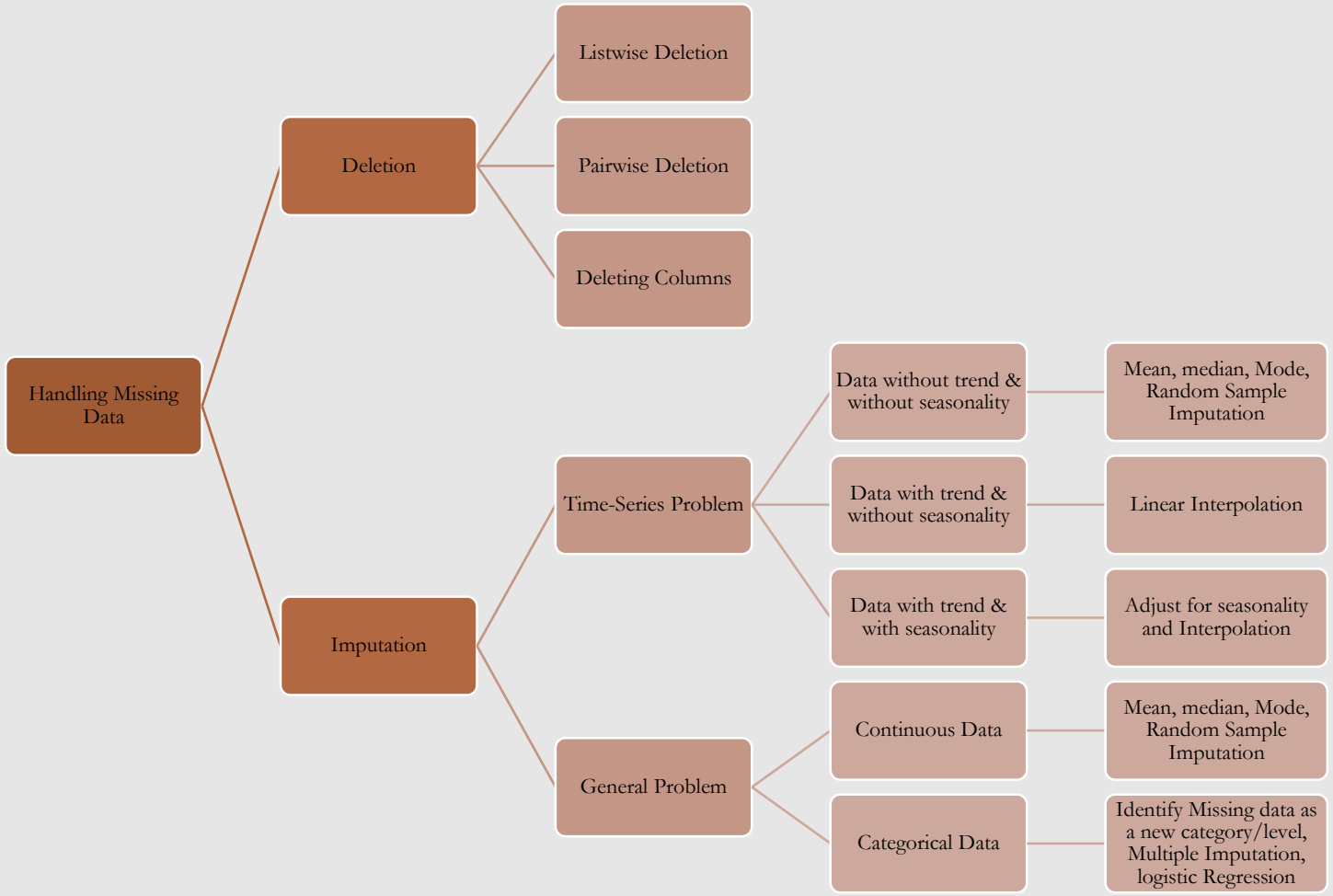
Step 2: Handling Missing Values (Cont'd)

- **Assumptions**:
  - Missing Completely at Random (**MCAR**): The probability of missingness is unrelated to the values of any variable
  - Missing at Random (**MAR**): Missingness is unrelated to the value that is missing but is related to the value of other variables.
  - Missing Not at Random (**MNAR**): Missingness is related to the value that is missing and often to the values of other variables.

Step 2: Handling Missing Values (Cont'd)

- **Deletion Methods**: Delete cases or variables
  - Listwise Method
  - Pairwise deletion: Ignore the cases with missing data on the variables
  - Variable deletion
- **Imputation Methods**: Substitute the missing values with estimates
  - Single Imputation
    - Mean Imputation
    - Last Observation Carried forward
    - Regression Imputation
    - KNN imputation
    - Worst Case Imputation
    - Best Case Imputation
    - EM imputation
  - Multiple Imputation

Step 2: Handling Missing Values (Cont'd)

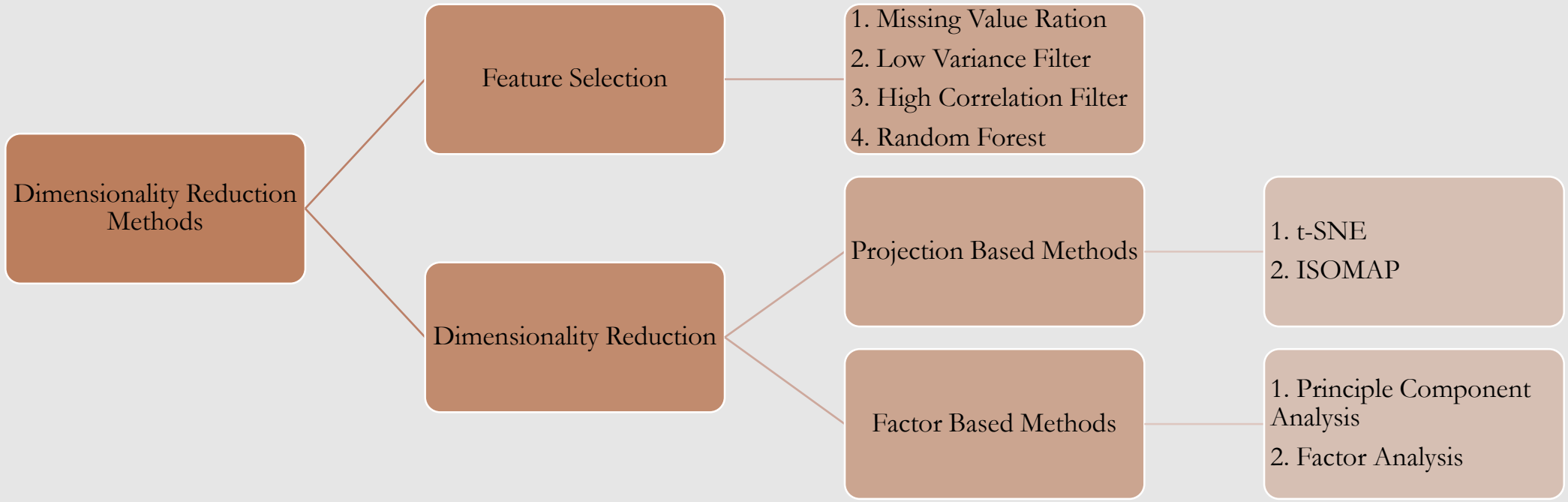


Source: Azure AI Library

Step 3: Normalization / Standardization

- **Why do we need normalization**?
  - We need normalization to make the training data less sensitive to the feature scales. Say, you have two features/variables where one has a range of (2~15) where the other has a range of (10~100). The features are not on the same feature scale/unit. As a result separating the behaviors and significance of the features become difficult. On top of that, the prediction model might become more sensitive towards the second variables as it has large variance in comparison to the first one.
  - Normalization also ensures faster convergence, as it removes higher variance from the dataset (Remember lecture from last class).
- **What is the correct way of performing normalizatio**n?
  - Naïve way: It's what you mentioned in your email. Take the largest of the whole dataset and perform the normalization.
  - Better than Naïve: As you suggested in your email, for multivariate case, Divide a feature/variable column by it's own max number.
  - Normalizing with Standard Score: One performs the normalization by the population's known parameters like the standard score. You need to apply the formula, $\frac{X-\mu}{\sigma}$. It is to be noted this method works better for populations that are normally distributed.
  - Min-Max feature Scaling (Normalizing within a scale): Bring all the features within a desired range in space, say [-1,1] (as discussed in the class). This is a unity-based normalization method and is calculated using $X' = \frac{X - X_{min}}{X_{max} - X_{min}}$. <u>This is one of the most used methods.</u>

Step 4: Dimensionality Reduction



Step 5: Selecting Test Data

- Select test data in a way so that all combination of cases are present there.
  - **Why?**

Step *X*: *You select*!

- You might need to do some additional steps too! That depends on your data and analysis you are doing.
  - Data Discretization (Creating Dummy variables)
  - Hierarchy Generation
  - Imbalanced Data pre-processing
  - Encoding etc.

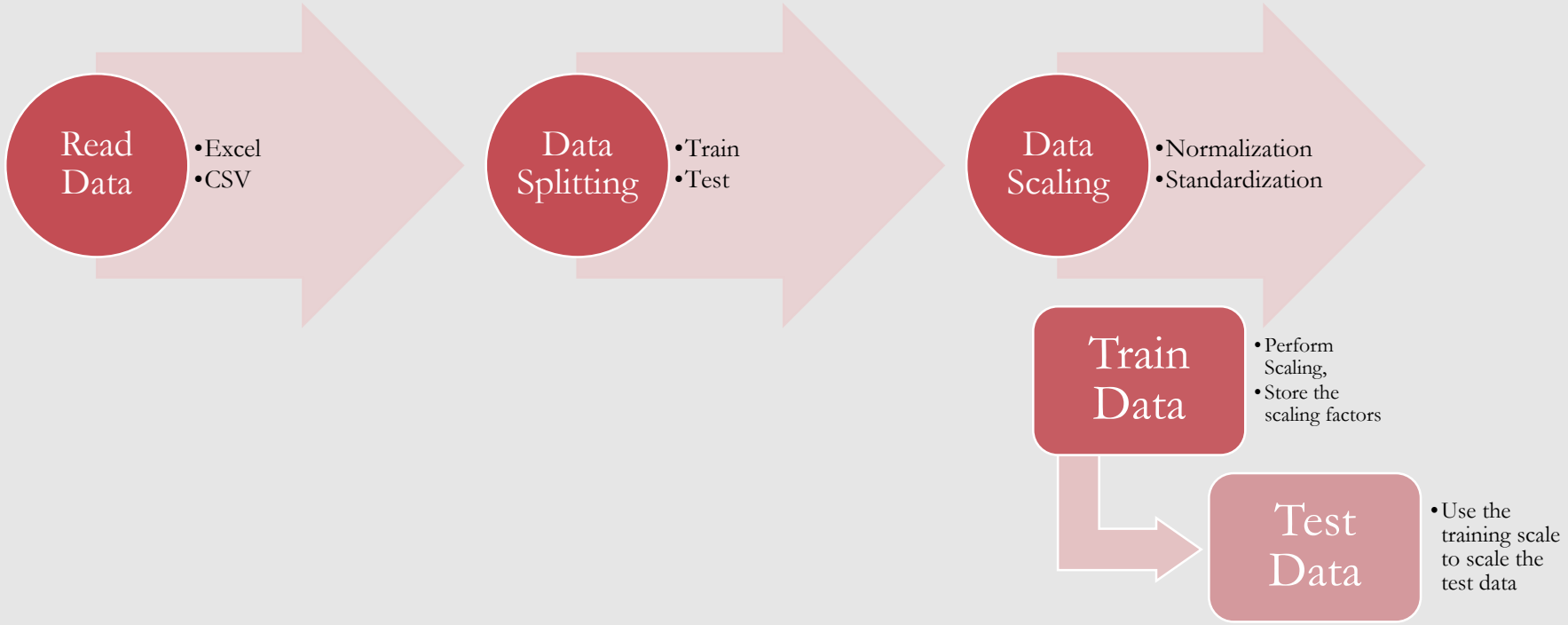
Some Common Steps

- Data Exploration
- Reading Excel File
- Reading CSV File
- Normalizing Data
- Standardizing Data
- Invert Scaling Data
- Train – Test Splitting
- Plotting: Prediction vs Actual etc.

Build your own function/class to do it at one go!!

Defining a pipeline!

- Define a function that can do the initial few steps given any data.



Defining a pipeline!

```
def Process_Time_Series_Data(self, Access_Sheet_Number, scale = (-1,1)):
    # Reads File
    series = self.ReadFile(Access_Sheet_Number)
    # Splits Data into Train-Test
    Train, Test = self.Train_Test_Split(series)
    # Scales the Data
    Train, Test, Scale = self.Normalizer(Train, Test, scale = scale)
    return Train, Test, Scale
```

Data File Number

Select Scale for Data

Reads the Data File

Splits the Data

Scales the Data

Invert Scaling Data