Module 2 – Regression Analysis

2 Marks – Theory

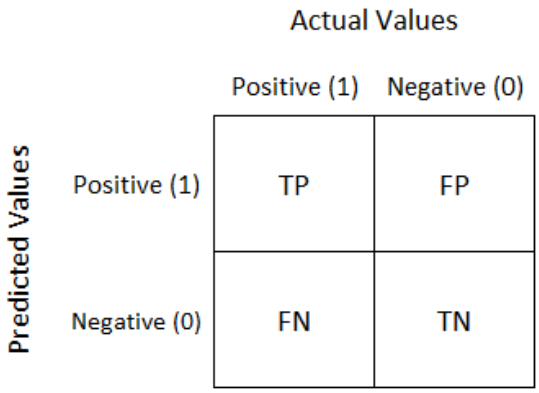
1. Explain any two metrics that measure the overall accuracy of the model. 2

Ans)

Accuracy: Accuracy is a basic metric that measures the overall correctness of a model's predictions. It is calculated as the ratio of the number of correct predictions to the total number of predictions made by the model. Mathematically, it can be expressed as:

$$\text{Accuracy} = \frac{\sum \text{TP} + \text{TN}}{\sum \text{TP} + \text{FP} + \text{FN} + \text{TN}}$$

Confusion Matrix: A matrix that summarizes the number of true positives, false positives, true negatives, and false negatives. Confusion matrix is used to calculate other performance measures such as accuracy, precision, recall, and F1-score.



2. Explain lower t-statistics indicate a predictor should be dropped. 2

Ans)

In a linear regression model, each predictor variable has an associated coefficient estimate and a corresponding t-statistic. The t-statistic is calculated by dividing the estimated coefficient by its standard error. It measures the number of standard errors that the coefficient estimate is away from zero. A higher absolute value of the t-statistic indicates a more significant predictor variable, while a lower absolute value suggests a less significant predictor variable.

A common practice is to use a threshold, such as a significance level (e.g., 0.05 or 0.01), to determine the statistical significance of a predictor. If the absolute value of the t-statistic for a predictor is lower than the threshold, it may indicate that the predictor is not statistically significant in explaining the variation in the dependent variable. In such cases, it may be considered for removal from the model to simplify the model and potentially improve its interpretability and predictive accuracy.

3. What is logistic regression? 1
Ans) Logistic regression is a statistical analysis method to predict a binary outcome, such as yes or no, based on prior observations of a data set.

A logistic regression model predicts a dependent data variable by analyzing the relationship between one or more existing independent variables. For example, a logistic regression could be used to predict whether a political candidate will win or lose an election or whether a high school student will be admitted or not to a particular college. These binary outcomes allow straightforward decisions between two alternatives.

A logistic regression model can take into consideration multiple input criteria. In the case of college acceptance, the logistic function could consider factors such as the student's grade point average, SAT score and number of extracurricular activities. Based on historical data about earlier outcomes involving the same input criteria, it then scores new cases on their probability of falling into one of two outcome categories.

4. Explain how logistic regression differs from linear regression. 2
Ans)

Linear Regression	Logistic Regression
Linear regression is used to predict the continuous dependent variable using a given set of independent variables.	Logistic Regression is used to predict the categorical dependent variable using a given set of independent variables.
Linear Regression is used for solving Regression problem.	Logistic regression is used for solving Classification problems.
In Linear regression, we predict the value of continuous variables.	In logistic Regression, we predict the values of categorical variables.
In linear regression, we find the best fit line, by which we can easily predict the output.	In Logistic Regression, we find the S-curve by which we can classify the samples.
Least square estimation method is used for estimation of accuracy.	Maximum likelihood estimation method is used for estimation of accuracy.
The output for Linear Regression must be a continuous value, such as price, age, etc.	The output of Logistic Regression must be a Categorical value such as 0 or 1, Yes or No, etc.
In Linear regression, it is required that relationship between dependent variable and independent variable must be linear.	In Logistic regression, it is not required to have the linear relationship between the dependent and independent variable.
In linear regression, there may be collinearity between the independent variables.	In logistic regression, there should not be collinearity between the independent variable.

Module 3 – Time Series Analysis
2 Marks – Theory
1. What are the components of time series? 2
Ans)
A time series is a collection of observations of well-defined data items obtained through repeated measurements over time. For example, measuring the value of retail sales each month of the year would comprise a time series. Data collected irregularly or only once are not time series. The components of the time series are as follows:

1. The **trend** refers to the long-term movement in a time series. It indicates whether the observation values are increasing or decreasing over time. Examples of trends are a steady increase in sales month over month or an annual decline in fatalities due to car accidents.
2. The **seasonality** component describes the fixed, periodic fluctuation in the observations over time. As the name suggests, the seasonality component is often related to the calendar. For example, monthly retail sales can fluctuate over the year due to the weather and holidays.
3. A **cyclic** component also refers to a periodic fluctuation, but one that is not as fixed as in the case of a seasonality component. For example, retail sales are influenced by the general state of the economy. Thus, a retail sales time series can often follow the lengthy boom-bust cycles of the economy.
4. There is another kind of movement that can be seen in the case of time series. It is pure **Irregular and Random** Movement. As the name suggests, no hypothesis or trend can be used to suggest irregular or random movements in a time series. These outcomes are unforeseen, erratic, unpredictable, and uncontrollable in nature.

2. Write the steps to perform box-jenkins methodology. 2
Ans)
This approach starts with the assumption that the process that generated the time series can be approximated using an ARMA model if it is stationary or an ARIMA model if it is non-stationary. Box-Jenkins methodology is an iterative approach that consists of the following 3 steps:

1. **Identification:** Use the data and all related information to help select a sub-class of model that may best summarize the data.
2. **Estimation:** Use the data to train the parameters of the model (i.e. the coefficients).
3. **Diagnostic Checking:** Evaluate the fitted model in the context of the available data and check for areas where the model may be improved.

It is an iterative process, so that as new information is gained during diagnostics, you can circle back to step 1 and incorporate that into new model classes.

3. What is the result of the absolute value of ACF(h), when it is closer to 1? 2
Ans) The difficulty is that the plot does not provide insight into the covariance of the variables in the time series and its underlying structure. The plot of the autocorrelation function (ACF) provides this insight. For a stationary time series, the ACF is defined as shown,

$$ACF(h) = \frac{cov(y_t, y_{t+h})}{\sqrt{cov(y_t, y_t)\, cov(y_{t+h}, y_{t+h})}} = \frac{cov(h)}{cov(0)}$$

By convention, the quantity h in the ACF is referred to as the lag, the difference between the time points t and t +h. At lag 0, the ACF provides the correlation of every point with itself. So ACF(O) is always equals 1 and it goes on decreasing as the lag increases.

When ACF(h) is closer to 1, it suggests a strong linear relationship between the values of the time series at the current time step and the values at the lag h time step. This indicates that the values of the time series are highly correlated with their past values at the specific lag h.

A value of 1 indicates a perfect positive autocorrelation, -1 indicates a perfect negative autocorrelation, and 0 indicates no autocorrelation.

4. **What are the three conditions for stationary time series?** 2
Ans)
As stated in the first step of the Box-Jenkins methodology, it is necessary to remove any trends or seasonality in the time series. This step is necessary to achieve a time series with certain properties to which autoregressive and moving average models can be applied. Such a time series is known as a stationary time series. A time series, Y_t for t = 1,2,3, ...,, is a stationary time series if the following three conditions are met:

(a) The expected value (mean) of Y_t is constant for all values of t.

(b) The variance of Y_t is finite.

(c)The covariance of Y_t and Y_{t+h}, depends only on the value of h = 0,1,2,... for all t.

5 Marks – Sums
1. Explain the application of time series in the following sectors. Finance, Economic, Engineering, Retail and Manufacturing. 2
Ans)
Finance: Time series analysis has become an intrinsic part of financial analysis and can be used in predicting interest rates, foreign currency risk, volatility in stock markets and many more. Policymakers and business experts use financial forecasting to make decisions about production, purchases, market sustainability, allocation of resources, etc. In investment, this analysis is employed to track the price fluctuations and price of a security over time. For instance, the price of a security can be recorded.

Economics: Time series analysis is used in economics to study macroeconomic trends, forecast economic indicators, and measure the effectiveness of economic policies. For example,

economists may use time series analysis to analyze trends in GDP, inflation, and unemployment. They can then use this information to identify economic cycles and assess the impact of economic policies.

Engineering: In engineering, time series analysis can be used to analyze data collected over time to identify patterns and trends, and to make predictions about future behavior. For example, time series analysis can be used to analyze sensor data from equipment to detect signs of wear and tear, and to schedule maintenance accordingly. It can also be used to monitor energy consumption and optimize energy usage in buildings or industrial processes.

Retail sales: For various product lines, a clothing retailer is looking to forecast future monthly sales. These forecasts need to account for the seasonal aspects of the customer's purchasing decisions. For example, in the northern hemisphere, sweater sales are typically brisk in the fall season, and swimsuit sales are the highest during the late spring and early summer. Thus, an appropriate time series model needs to account for fluctuating demand over the calendar year.

Manufacturing: Time series analysis is used in manufacturing to monitor production processes, predict maintenance needs, and optimize product quality. For example, manufacturers may use time series analysis to monitor machine performance, predict maintenance needs, and identify trends in product quality. They can then use this information to optimize production processes, reduce downtime, and improve product quality.

2. Which are the models used for forecasting? 2
Ans)

Autoregressive Moving Average (ARMA):

The autoregressive moving average (ARMA) model is a combination of the autoregressive and moving average models. The ARMA model is defined as a regression model in which the dependent/response variable is a linear function of past values of both the dependent/response variable and the error term. The order of an ARMA model is represented by ‘p’ for the autoregressive part and ‘q’ for the moving average part.

Autoregressive Integrated Moving Average (ARIMA):

The autoregressive integrated moving average (ARIMA) model is a generalization of the ARMA model. The ARIMA model is defined as a regression model in which the dependent/response variable is a linear function of past values of both the dependent/response variable and the error term, where the error term has been differentiated ‘d’ times. The order of an ARIMA model is represented by ‘p’ for the autoregressive part, ‘q’ for the moving average part, and ‘d’ for the differencing part.

Seasonal Autoregressive Integrated Moving-Average (SARIMA):

SARIMA is a type of time-series forecasting model that takes into account both seasonality and autocorrelation. SARIMA models are based on a combination of differencing, autoregression, and moving average processes. These models can be used to forecast short-term or long-term trends in data. SARIMA models are generally considered to be more accurate than other types of time-series forecasting models, such as ARIMA models. SARIMA models are also relatively easy to interpret and use. The SARIMA model can be used to forecast demand for a product or service over the course of a year.

Vector Autoregression (VAR):

VAR is a multivariate time series model that can forecast multiple variables simultaneously. It models the dependencies and interactions between multiple time series variables, making it useful for forecasting in situations where multiple variables influence each other.

3. Explain Auto-correlation function and partial auto-correlation function. 2
Ans)
Auto-correlation Function:
The difficulty is that the simple plot does not provide insight into the covariance of the variables in the time series and its underlying structure. The plot of the autocorrelation function (ACF) provides this insight. For a stationary time series, the ACF is defined as shown,

$$ACF(h) = \frac{cov(y_t, y_{t+h})}{\sqrt{cov(y_t, y_t)\, cov(y_{t+h}, y_{t+h})}} = \frac{cov(h)}{cov(0)}$$

By convention, the quantity h in the ACF is referred to as the lag, the difference between the time points t and t+h. At lag 0, the ACF provides the correlation of every point with itself. So ACF(O) always equals 1 and it goes on decreasing as the lag increases.

A value of 1 indicates a perfect positive autocorrelation, -1 indicates a perfect negative autocorrelation, and 0 indicates no autocorrelation.

Partial Auto-correlation Function:

A measure of the autocorrelation between Y_t and Y_{t+h} for h = 1, 2, 3 ... with the effect of the

Y_{t+1} and Y_{t+h-1} values excluded from the measure. The partial autocorrelation function (PACF) provides such a measure.
The PACF measures the correlation between a time series and its lagged values, while accounting for the correlations that may exist at shorter lags. By removing the influence of intermediate lags, the PACF allows us to identify the "pure" or "direct" relationship between the time series and its lagged values.
The PACF can be interpreted in a similar way to the autocorrelation function (ACF), but it measures the correlation between a time series and its lagged values after controlling for the influence of shorter lags.

Because the ACF and PACF are based on correlations, negative and positive values are possible. Thus, the magnitudes of the functions at the various lags should be considered in terms of absolute values.

4. What are the other methods of Time Series Analysis? 2
Ans)
Additional time series methods include the following:

• **Autoregressive Moving Average with Exogenous inputs** (ARM AX) is used to analyze a time series that is dependent on another time series. For example, retail demand for products can be modeled based on the previous demand combined with a weather-related time series such as temperature or rainfall.
• **Spectral analysis** is commonly used for signal processing and other engineering applications. Speech recognition software uses such techniques to separate the signal for the spoken words from the overall signal that may include some noise.
• **Generalized Autoregressive Conditionally Heteroscedastic** (GARCH) is a useful model for addressing time series with nonconstant variance or volatility. GARCH is used for modeling stock market activity and price fluctuations.
• **Kalman filtering** is useful for analyzing real-time inputs about a system that can exist in certain states. Typically, there is an underlying model of how the various components of the system interact and affect each other. A Kalman filter processes the various inputs, attempts to identify the errors in the input, and predicts the current state. For example, a Kalman filter in a vehicle navigation system can process various inputs, such as speed and direction, and update the estimate of the current location.
• **Vector ARIMA (VARIMA):** Multivariate time series analysis examines multiple time series and their effect on each other. Vector ARIMA (VARIMA**)** extends ARIMA by considering a vector of several time series at a particular time, t. VARIMA can be used in marketing analyses that examine the time series related to a company's price and sales volume as well as related time series for the competitors.

Module 5 – Data analytics and visualization with R
2 Marks – Theory
5. Explain Kernel density plot in r with proper example. 2
Ans)
A kernel density plot is a type of plot that displays the distribution of values in a dataset using one continuous curve.
A kernel density plot is similar to a histogram, but it’s even better at displaying the shape of a distribution since it isn’t affected by the number of bins used in the histogram

- Create One Kernel Density Plot

The following code shows how to create a kernel density plot for one dataset in R:

```
#create data
```

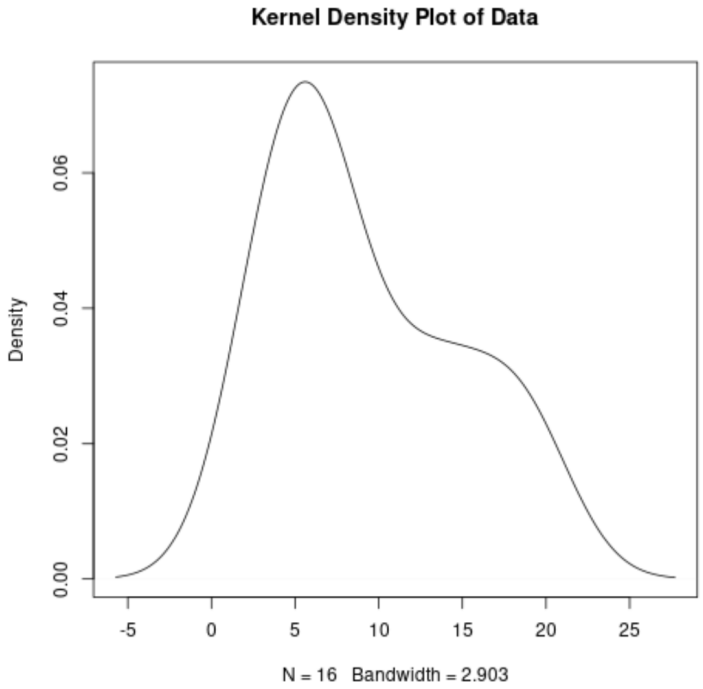
```
data <- c(3, 3, 4, 4, 5, 6, 7, 7, 7, 8, 12, 13, 14, 17, 19, 19)

#define kernel density

kd <- density(data)

#create kernel density plot

plot(kd, main='Kernel Density Plot of Data')
```



The x-axis shows the values of the dataset and the y-axis shows the relative frequency of each value. The highest point in the plot shows where the values occur most often.

6. What is the main idea for exploratory data analysis and why do you need visualization before analysis?

Ans)

Exploratory data analysis is a data analysis approach to reveal the important characteristics of a dataset, mainly through visualization. The main purpose of EDA is to help look at data before making any assumptions. It can help identify obvious errors, as well as better understand patterns within the data, detect outliers or anomalous events, and find interesting relations among the variables. Data scientists can use exploratory analysis to ensure the results they produce are valid and applicable to any desired business outcomes and goals.

Visualization is an important part of EDA because it allows us to quickly and easily explore the data and identify patterns or trends that may not be immediately apparent from numerical summaries or statistics. By visualizing the data, we can gain insight into its distribution, identify outliers or unusual observations, and see how different variables are related to each other.

For example, suppose we have a dataset of sales data for a retail store, including variables such as date, product, price, and quantity sold. Before analyzing the data, we may want to visualize it to gain a better understanding of its structure and identify any patterns or trends. We might create a line chart of sales over time to see if there are any seasonal patterns, a scatterplot of price versus quantity sold to see if there is a relationship between these variables, or a histogram of sales by product to see which products are selling the most. By visualizing the data in this way, we can identify interesting features or patterns and generate hypotheses about the data, which we can then test using statistical techniques.

5 Marks – Theory

1. How would you use facet wrap and facet grid methods of visualization with R and give proper examples. 3

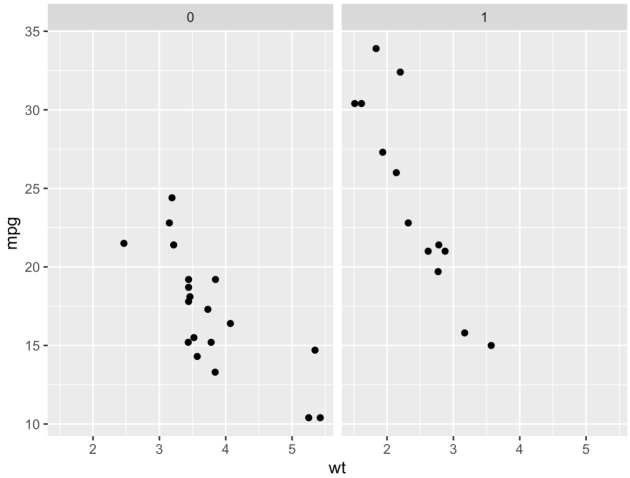
Ans)

In R, the ggplot2 package provides the facet_wrap() and facet_grid() functions for creating multi-panel plots that allow you to visualize subsets of your data. Both functions are useful for exploring relationships between variables or comparing groups of data.

Example using the mtcars dataset that comes with R:

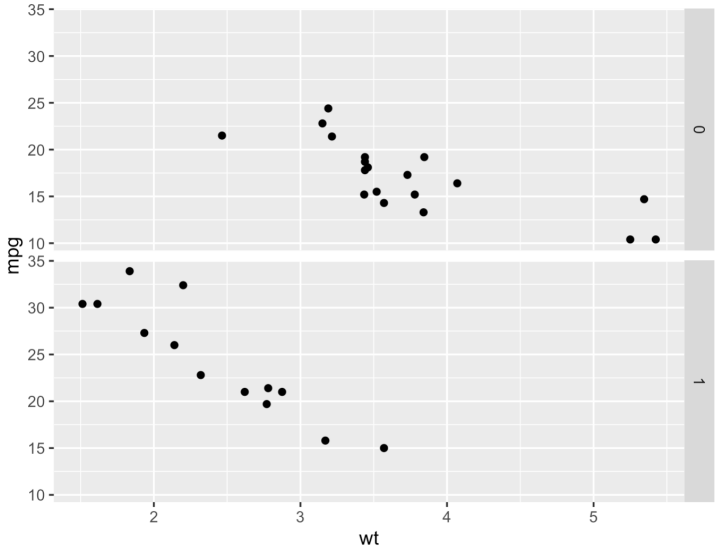
Suppose we want to compare the relationship between miles per gallon (mpg) and weight (wt) for different types of transmission (am):

```
library(ggplot2)
ggplot(mtcars, aes(x = wt, y = mpg)) +
geom_point() +
facet_wrap(~ am)
```



This code creates a scatterplot of mpg versus wt using the ggplot2 package. The facet_wrap() function is used to create separate panels for each type of transmission (am). The ~ symbol in the facet_wrap() function indicates that am is the variable used to create the subsets.
Alternatively, we can use facet_grid() to create a 2x1 grid with separate panels for automatic (am = 0) and manual (am = 1) transmissions:

```
ggplot(mtcars, aes(x = wt, y = mpg)) +
geom_point() +
facet_grid(am ~ .)
```



This code creates the same scatterplot, but with facet_grid() instead of facet_wrap(). The am ~ . argument tells R to use am for the rows and to put all remaining variables (in this case, none) in the columns.

Both facet_wrap() and facet_grid() are useful for comparing subsets of your data and identifying patterns or trends within those subsets.

2. How will you enhance the following R code to display horizontal bar charts and avoid axis labels to overlap? 6
Ans) (TAKING A CODE FOR EXAMPLE ANY CODE MIGHT COME IN EXAM)

To display horizontal bar charts in R, you can use the coord_flip() function in ggplot2 package. This function flips the X and Y axes of a plot, effectively turning a vertical bar chart into a horizontal bar chart.
To avoid axis labels overlapping, you can adjust the spacing between the labels using the theme() function in ggplot2. Specifically, you can adjust the axis.text.y argument to increase the spacing between the Y-axis labels.
Here's an example of how you could modify the R code to display horizontal bar charts and avoid axis label overlap:

```
library(ggplot2)
# Load data
data <- mtcars
# Create bar chart with horizontal bars
p <- ggplot(data, aes(x = mpg, y = factor(cyl))) +
geom_bar(stat = "identity") +
coord_flip()
# Increase spacing between Y-axis labels to avoid overlap
p + theme(axis.text.y = element_text(margin = margin(r = 10)))
```

In this code, we first load the ggplot2 package and the mtcars dataset. We then create a vertical bar chart of miles per gallon (mpg) by number of cylinders (cyl) using geom_bar() and stat = "identity".

To create a horizontal bar chart, we add the coord_flip() function. This flips the X and Y axes of the plot, effectively turning the vertical bars into horizontal bars.
Finally, to adjust the spacing between the Y-axis labels and avoid overlap, we use the theme() function with the axis.text.y argument. The margin argument specifies the amount of space to add to the right margin of each label. In this case, we add 10 units of space to the right margin.
With these modifications, the plot should display as a horizontal bar chart with enough space between the Y-axis labels to avoid overlap.

Module 6 – Data analytics and visualization with Python
2 Marks – Theory
1. How would you apply str.cat() to following python code to concatenate the address column with the name column? 3
Ans)
Assuming you have a DataFrame with columns "name" and "address" and you want to concatenate these two columns together into a new column named "full_name_address", you can use the str.cat() method in the following way:

```
import pandas as pd
# create example DataFrame
df = pd.DataFrame({ 'name': ['John', 'Jane', 'Bob'], 'address': ['123 Main St', '456 Oak Ave', '789 Elm Blvd'] })
# concatenate name and address columns into a new column named full_name_address
df['full_name_address'] = df['name'].str.cat(df['address'], sep=', ')
print(df)
```

```
   name         address          full_name_address
0  John     123 Main St         John, 123 Main St
1  Jane     456 Oak Ave         Jane, 456 Oak Ave
2  Bob      789 Elm Blvd        Bob, 789 Elm Blvd
```

2. How would you find the determinant and rank for the following matrix using python code? 2

Ans)

To find the determinant and rank of a matrix in Python, you can use the NumPy library. Here's an example code for a matrix A:

```
import numpy as np

# define the matrix A

A = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

# calculate the determinant of A

det_A = np.linalg.det(A)

print("Determinant of A:", det_A)

# calculate the rank of A

rank_A = np.linalg.matrix_rank(A)

print("Rank of A:", rank_A)
```

Output:

```
Determinant of A: 0.0

Rank of A: 2
```

In this example, we first define the matrix A using NumPy's array() function. Then, we use the linalg.det() function to calculate the determinant of A and store it in the variable det_A. We use the linalg.matrix_rank() function to calculate the rank of A and store it in the variable rank_A. Finally, we print out the values of det_A and rank_A.

5 Marks – Theory

1. How would you use CSR and CSC in scipy to handle spare data, explain the methods with examples? 3

Ans)

CSR and CSC are two storage formats used in the Scipy library to efficiently handle sparse matrices. CSR stands for "Compressed Sparse Row" and it represents a sparse matrix using

three arrays: one for the non-zero values, one for the column indices of the non-zero values, and one for the row pointers that indicate the start and end of each row in the first two arrays. Here's an example of how to create a CSR matrix using Scipy:

```
import numpy as np
from scipy.sparse import csr_matrix
# define a dense matrix
A_dense = np.array([[1, 0, 2], [0, 3, 0], [4, 0, 5]])
# create a CSR matrix from the dense
matrix A_csr = csr_matrix(A_dense)
# print the CSR matrix
print(A_csr)
```

Output:

```
(0, 0)1

(0, 2)2

(1, 1)3

(2, 0)4

(2, 2)5
```

In this example, we first define a dense matrix A_dense. Then, we use the csr_matrix() function to create a CSR matrix A_csr from the dense matrix. Finally, we print out the CSR matrix using the print() function. As you can see, the CSR matrix only stores the non-zero values and their indices, and it uses the row pointers to reconstruct the original matrix.

CSC stands for "Compressed Sparse Column" and it represents a sparse matrix using the same three arrays as CSR, but with the column indices and row pointers swapped. Here's an example of how to create a CSC matrix using Scipy:

```
import numpy as np
from scipy.sparse import csc_matrix
# define a dense matrix
A_dense = np.array([[1, 0, 2], [0, 3, 0], [4, 0, 5]])
# create a CSC matrix from the dense
matrix A_csc = csc_matrix(A_dense)
# print the CSC matrix
print(A_csc)
```

Output:

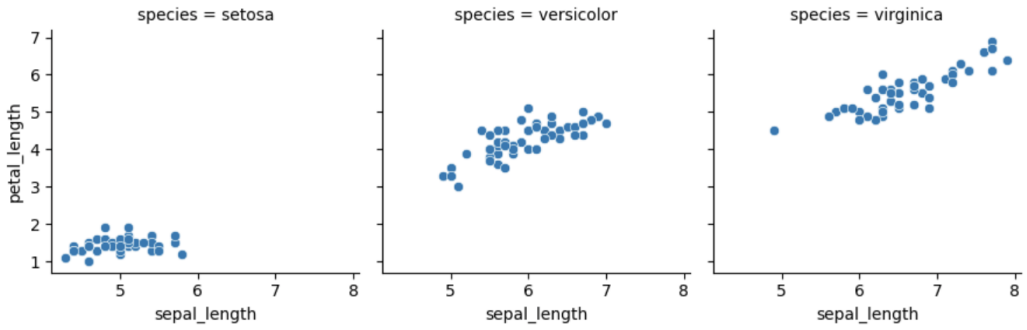
```
(0, 0)   1
(2, 0)   4
(1, 1)   3
(0, 2)   2
```

(2, 2) 5

In this example, we first define a dense matrix A_dense. Then, we use the csc_matrix() function to create a CSC matrix A_csc from the dense matrix. Finally, we print out the CSC matrix using the print() function. As you can see, the CSC matrix stores the same non-zero values and their indices as the CSR matrix, but it uses the column indices and row pointers to reconstruct the original matrix.
Both CSR and CSC formats are useful for handling sparse matrices because they can save a lot of memory and computation time compared to dense matrices. The choice between CSR and CSC depends on the specific problem and the available algorithms for each format.

2. How would you use facet grid and pair grid methods of visualization with python seaborn and give proper examples. 3
Ans)
Facet Grid
Suppose we want to compare the relationship between sepal length (sepal_length) and petal length (petal_length) for each species (species) in the iris dataset. We can use FacetGrid() to create separate panels for each species:

```
import seaborn as sns
# Load iris dataset
iris = sns.load_dataset("iris")
# Create FacetGrid
g = sns.FacetGrid(iris, col="species")
g.map(sns.scatterplot, "sepal_length", "petal_length")
```

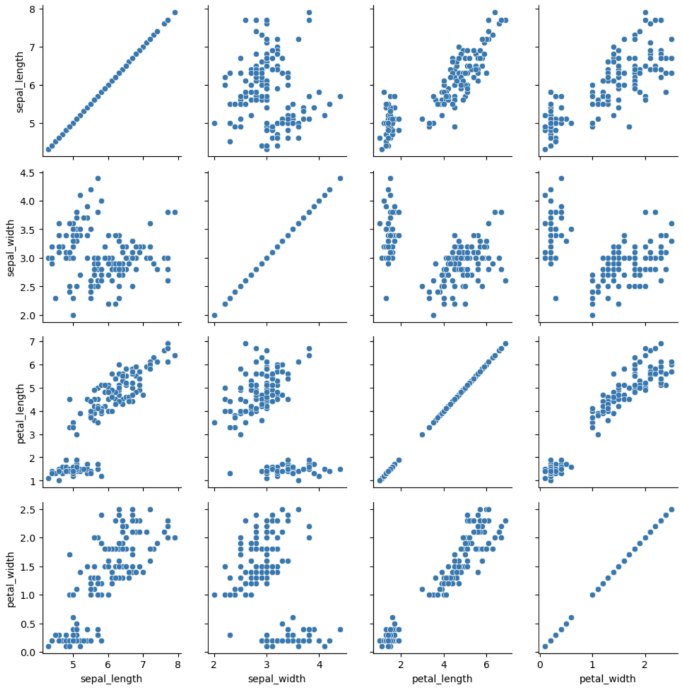


This code creates a FacetGrid with col="species", which tells Seaborn to create separate panels for each species in the iris dataset. We then use map() to apply a scatterplot to each panel, plotting sepal_length on the X-axis and petal_length on the Y-axis.

Pair Grid
Suppose we want to compare the relationship between all pairs of variables in the iris dataset. We can use PairGrid() to create a grid of scatterplots, with each variable plotted against all other variables:

```
import seaborn as sns
```

```
# Load iris dataset
iris = sns.load_dataset("iris")
# Create PairGrid
g = sns.PairGrid(iris)
g.map(sns.scatterplot)
```



This code creates a PairGrid with the iris dataset. We then use map() to apply a scatterplot to each cell in the grid, plotting one variable on the X-axis and another variable on the Y-axis.

Both FacetGrid and PairGrid are useful for comparing subsets of your data and identifying patterns or trends within those subsets.

Note: Please refer to the answers and learn and also modify the content as per your understanding and marks.