1) Demonstrate Simple Linear Regression model using R/Python

a) Define Problem Statement

Simple linear regression is a statistical method used to establish a relationship between two variables, where one variable is the independent variable and the other is the dependent variable. The goal of simple linear regression is to create a linear equation that can be used to predict the value of the dependent variable based on the value of the independent variable.

The problem statement for simple linear regression typically involves the following:

- A dataset that contains two variables: the independent variable (X) and the dependent variable (Y).
- The goal is to find a linear equation that can accurately predict the value of Y based on the value of X.
- The linear equation is of the form Y = a + bX, where a is the intercept and b is the slope of the line.
- The objective is to find the values of a and b that minimize the difference between the predicted values of Y and the actual values of Y in the dataset.

b) Define Null Hypothesis

In simple linear regression, the null hypothesis typically states that there is no significant linear relationship between the independent variable (X) and the dependent variable (Y). In other words, the null hypothesis assumes that the slope of the regression line is zero, which means that the value of Y does not change as the value of X changes.

Mathematically, the null hypothesis can be expressed as:

H0: β1 = 0

where β1 is the slope coefficient in the regression equation Y = β0 + β1X.

If the null hypothesis is true, then any variation in the value of Y can be attributed to random error or other factors that are not related to the value of X. On the other hand, if the null hypothesis is rejected, then it can be concluded that there is a significant linear relationship between X and Y, and the slope of the regression line is non-zero.

c) Perform Pre-processing operations on dataset

Pre-processing of the dataset is an important step in data analysis that involves preparing the dataset for modeling. Pre-processing operations on a dataset may include the following:

1. Data Cleaning: This involves identifying and handling missing values, duplicates, and outliers. Missing values can be imputed using techniques such as mean imputation, median imputation, or interpolation. Duplicates can be identified and removed. Outliers can be detected and handled using techniques such as winsorization or removal.
2. Data Integration: This involves combining multiple datasets into one dataset. This is often required when the data is collected from different sources or different time periods.
3. Data Transformation: This involves converting the data into a suitable format for modeling. This may include scaling or normalization of the data to ensure that all variables are on the same scale, encoding categorical variables into numerical values, or creating new features through feature engineering.
4. Data Reduction: This involves reducing the dimensionality of the dataset by removing redundant features or performing feature selection. This can help to improve the model's performance and reduce overfitting.
5. Data Sampling: This involves selecting a subset of the data to use for modeling. This can be done to balance the dataset or to reduce the computational cost of modeling.

To evaluate a regression model, there are several metrics and techniques that can be used:

1. Mean Squared Error (MSE): MSE measures the average squared difference between the predicted values and the actual values. A lower MSE indicates a better fit.
2. Root Mean Squared Error (RMSE): RMSE is the square root of MSE and is often used to put the error metric into the same units as the original data..
3. Mean Absolute Error (MAE): MAE measures the average absolute difference between the predicted values and the actual values. It is less sensitive to outliers than MSE.
4. Residual Analysis: Residual analysis involves examining the difference between the predicted values and the actual values. A scatterplot of the residuals should show no clear pattern, indicating that the model is a good fit.
5. Cross-validation: Cross-validation is a technique used to test the performance of the model on new data. The data is split into training and testing sets, and the model is trained on the training set and tested on the testing set. This process is repeated multiple times to obtain an average performance metric.

When evaluating a regression model, it is important to consider the specific requirements of the problem being solved and choose the appropriate metrics and techniques accordingly.

2) Demonstrate Multiple Linear Regression using R/Python

a) Define Problem Statement

Multiple linear regression is a statistical technique used to model the relationship between two or more independent variables and a dependent variable. The problem statement for multiple linear regression involves identifying the factors that are most strongly associated with the dependent variable and using this information to make predictions or draw conclusions about the population.

For example, a problem statement for multiple linear regression might involve predicting the sales of a product based on factors such as advertising expenditure, product price, and consumer demographics. The goal of the analysis would be to determine which of these factors have the strongest impact on sales and how much of that impact can be attributed to each individual factor.

Another example could be predicting the housing prices based on factors such as location, size, number of bedrooms, and amenities.

b) Define Null Hypothesis

In multiple regression analysis, the null hypothesis states that there is no significant relationship between the dependent variable and the independent variables included in the model. This can be expressed mathematically as:

H0: β1 = β2 = β3 = ... = βk = 0

where β1, β2, β3, ..., βk are the regression coefficients for the independent variables in the model.

This null hypothesis assumes that there is no linear relationship between the independent variables and the dependent variable, or that any relationship that does exist is purely due to chance. In other words, if the null hypothesis is true, the independent variables do not have a significant impact on the dependent variable.

c) Perform Pre-processing operations on dataset

Pre-processing of the dataset is an important step in data analysis that involves preparing the dataset for modeling. Pre-processing operations on a dataset may include the following:

1. Data Cleaning: This involves identifying and handling missing values, duplicates, and outliers. Missing values can be imputed using techniques such as mean imputation, median imputation, or interpolation. Duplicates can be identified and removed. Outliers can be detected and handled using techniques such as winsorization or removal.
2. Data Integration: This involves combining multiple datasets into one dataset. This is often required when the data is collected from different sources or different time periods.
3. Data Transformation: This involves converting the data into a suitable format for modeling. This may include scaling or normalization of the data to ensure that all variables are on the same scale, encoding categorical variables into numerical values, or creating new features through feature engineering.
4. Data Reduction: This involves reducing the dimensionality of the dataset by removing redundant features or performing feature selection. This can help to improve the model's performance and reduce overfitting.
5. Data Sampling: This involves selecting a subset of the data to use for modeling. This can be done to balance the dataset or to reduce the computational cost of modeling.

Evaluate the Multiple Regression Model:

To evaluate a regression model, there are several metrics and techniques that can be used:

6. Mean Squared Error (MSE): MSE measures the average squared difference between the predicted values and the actual values. A lower MSE indicates a better fit.
7. Root Mean Squared Error (RMSE): RMSE is the square root of MSE and is often used to put the error metric into the same units as the original data..
8. Mean Absolute Error (MAE): MAE measures the average absolute difference between the predicted values and the actual values. It is less sensitive to outliers than MSE.
9. Residual Analysis: Residual analysis involves examining the difference between the predicted values and the actual values. A scatterplot of the residuals should show no clear pattern, indicating that the model is a good fit.
10. Cross-validation: Cross-validation is a technique used to test the performance of the model on new data. The data is split into training and testing sets, and the model is trained on the training set and tested on the testing set. This process is repeated multiple times to obtain an average performance metric.

When evaluating a regression model, it is important to consider the specific requirements of the problem being solved and choose the appropriate metrics and techniques accordingly.

3) Demonstrate Logistic Regression using R/Python

a) Define Problem Statement

Logistic regression is a statistical technique used to model the relationship between a binary dependent variable (i.e., a variable that can take on only two values, such as 0 or 1) and one or more independent variables. The problem statement for logistic regression involves identifying the factors that are most strongly associated with the dependent variable and using this information to predict the probability of the dependent variable taking on a particular value.

For example, a problem statement for logistic regression might involve predicting whether a customer will purchase a product based on factors such as age, gender, income, and past purchase history. The goal of the analysis would be to determine which of these factors have the strongest impact on the likelihood of a purchase and how much of that impact can be attributed to each individual factor.

Another example could be predicting the likelihood of a patient developing a particular disease based on factors such as age, family history, and lifestyle choices

b) Define Null Hypothesis

In logistic regression, the null hypothesis states that there is no significant relationship between the independent variables and the probability of the binary outcome. Specifically, the null hypothesis assumes that the logistic regression coefficients for all independent variables are equal to zero, which means that the probability of the binary outcome does not vary with the values of the independent variables.

Mathematically, the null hypothesis can be expressed as:

$H0: \beta 1 = \beta 2 = \beta 3 = \ldots = \beta k = 0$

where $\beta 1, \beta 2, \beta 3, \ldots, \beta k$ are the logistic regression coefficients for the independent variables.

To test the null hypothesis, the logistic regression model is compared to a baseline model that assumes that all coefficients are equal to zero. The significance of the difference between the two models is then evaluated using a statistical test, such as the likelihood ratio test, Wald test, or score test. If the p-value for the test is less than the significance level (usually 0.05), the null hypothesis is rejected, and it is concluded that there is a significant relationship between the independent variables and the probability of the binary outcome.

c) Is it classification or prediction problem. Explain.

Logistic regression can be used for both classification and prediction problems, but it is most commonly used for classification problems. The primary goal of logistic regression is to predict the probability of an event occurring based on the values of one or more predictor variables.

In a classification problem, the logistic regression model is used to predict the probability of a binary outcome (e.g., 0 or 1, true or false, yes or no, etc.) based on the values of the independent variables. The output of the logistic regression model is a probability value, which can be used to classify the observation into one of the two categories based on a decision threshold. For example, if the probability is greater than or equal to 0.5, the observation is classified as belonging to the positive class (1); otherwise, it is classified as belonging to the negative class (0).

In a prediction problem, the logistic regression model can be used to predict the probability of an event occurring based on the values of the independent variables.

d) Perform Pre-processing operations on dataset

Pre-processing of the dataset is an important step in data analysis that involves preparing the dataset for modeling. Pre-processing operations on a dataset may include the following:

1. Data Cleaning: This involves identifying and handling missing values, duplicates, and outliers. Missing values can be imputed using techniques such as mean imputation, median imputation, or interpolation. Duplicates can be identified and removed. Outliers can be detected and handled using techniques such as winsorization or removal.
2. Data Integration: This involves combining multiple datasets into one dataset. This is often required when the data is collected from different sources or different time periods.
3. Data Transformation: This involves converting the data into a suitable format for modeling. This may include scaling or normalization of the data to ensure that all variables are on the same scale, encoding categorical variables into numerical values, or creating new features through feature engineering.
4. Data Reduction: This involves reducing the dimensionality of the dataset by removing redundant features or performing feature selection. This can help to improve the model's performance and reduce overfitting.

5. Data Sampling: This involves selecting a subset of the data to use for modeling. This can be done to balance the dataset or to reduce the computational cost of modeling.

Evaluate Model:

To evaluate a logistic regression model, several metrics can be used. Here are some commonly used evaluation metrics:

1. Accuracy: Accuracy is the proportion of correct predictions to the total number of predictions. It is a common evaluation metric for classification models and can be misleading if the data is imbalanced.
2. Precision: Precision is the proportion of true positives to the total number of positive predictions. It measures the model's ability to avoid false positives.
3. Recall: Recall is the proportion of true positives to the total number of actual positives. It measures the model's ability to identify all positive cases.
4. F1 Score: The F1 score is the harmonic mean of precision and recall. It is a balanced metric that takes both precision and recall into account.
5. Receiver Operating Characteristic (ROC) Curve: The ROC curve is a plot of the true positive rate (sensitivity) versus the false positive rate (1-specificity) at different probability thresholds. It is a useful visualization for comparing different models.

4) Perform following Hypothesis testing methods using R/Python

a) One sample t-test

The one-sample t-test is a hypothesis test used to determine whether the mean of a sample of continuous data is significantly different from a known or hypothesized population mean. It is typically used when the sample size is small (n < 30) and the population standard deviation is unknown.

The null hypothesis for a one-sample t-test is that there is no significant difference between the sample mean and the hypothesized population mean. The alternative hypothesis is that there is a significant difference between the sample mean and the hypothesized population mean.

The formula for the one-sample t-test statistic is:

$t = (\bar{x} - \mu) / (s / \text{sqrt}(n))$

where $\bar{x}$ is the sample mean, μ is the hypothesized population mean, s is the sample standard deviation, and n is the sample size.

To conduct a one-sample t-test, we first set a significance level (α), typically 0.05. We then calculate the t-test statistic and its associated p-value using the formula above

b) Two sampled t-test

The two-sample t-test is a hypothesis test used to determine whether there is a significant difference between the means of two independent samples of continuous data. It is typically used when comparing the means of two groups, such as a control group and a treatment group, or two different populations.

The null hypothesis for a two-sample t-test is that there is no significant difference between the means of the two groups. The alternative hypothesis is that there is a significant difference between the means of the two groups.

There are two types of two-sample t-tests: the independent samples t-test and the paired samples t-test.

The independent samples t-test is used when the two samples are independent of each other, meaning that the observations in one sample are not related to the observations in the other sample. The formula for the independent samples t-test statistic is:

t = (x̄1 - x̄2) / sqrt((s1^2/n1) + (s2^2/n2))

where x̄1 and x̄2 are the sample means, s1 and s2 are the sample standard deviations, n1 and n2 are the sample sizes, and the denominator is the standard error of the difference between the two sample means.

The paired samples t-test is used when the two samples are related or paired, such as when the same individuals are measured before and after a treatment. The formula for the paired samples t-test statistic is:

t = (x̄d - μd) / (sd / sqrt(n))

where x̄d is the mean difference between the paired observations, μd is the hypothesized population mean difference, sd is the standard deviation of the differences, and n is the number of pairs.

c) Paired sampled t-test

The paired samples t-test is a hypothesis test used to determine whether there is a significant difference between the means of two related or paired samples of continuous data. It is typically used when comparing the means of two groups where the observations in one group are related or paired with the observations in the other group, such as in a before-and-after study or a matched pairs design.

The null hypothesis for a paired samples t-test is that there is no significant difference between the means of the two related samples. The alternative hypothesis is that there is a significant difference between the means of the two related samples.

The formula for the paired samples t-test statistic is:

t = (x̄d - μd) / (sd / sqrt(n))

where x̄d is the mean difference between the paired observations, μd is the hypothesized population mean difference, sd is the standard deviation of the differences, and n is the number of pairs.

d) ANOVA (F-TEST)

Paired ANOVA F test is a statistical hypothesis test used to compare the means of three or more paired or dependent groups. It is an extension of the paired samples t-test to more than two groups. Paired ANOVA F test is also known as repeated measures ANOVA, within-subjects ANOVA or ANOVA for matched pairs.

The null hypothesis for paired ANOVA F test is that there is no significant difference in the means of the paired groups, while the alternative hypothesis is that at least one mean is significantly different from the others.

The test statistic used in paired ANOVA F test is the F-ratio, which is calculated as the ratio of the between-group variability to the within-group variability. The within-group variability is the average of the variances of each paired group, while the between-group variability is the variance of the means of the paired groups.

To perform paired ANOVA F test, first the differences between each pair of observations are calculated, then the mean and variance of these differences are calculated. The F-ratio is then computed using the following formula:

F = (SS_between / (k - 1)) / (SS_within / (n - k))

where k is the number of paired groups, n is the total number of observations, SS_between is the sum of squared differences between the means of the paired groups and the overall mean, and SS_within is the sum of squared differences within each paired group.

5) Implement Decision Tree Algorithm (Classifier) using R/Python

a) Define Problem

Given a dataset with multiple variables, including a target variable, we want to create a decision tree model that can predict the target variable based on the values of the other variables. The goal is to create a model that can accurately classify new observations into the appropriate target variable category.

For example, we may have a dataset of customer information for a bank, including variables such as age, income, education level, and whether or not they have a credit card. The target variable could be whether or not the customer is likely to default on their credit card payments.

b) Implement Decision Tree Algorithm on suitable dataset.

1. Collect and preprocess the dataset: The first step is to collect the dataset that is suitable for a decision tree algorithm. Preprocessing of data is an important step, which includes handling missing values, removing irrelevant variables, and handling categorical variables.
2. Split the dataset: Split the dataset into two parts, one for training and one for testing the model. The split should be such that the training set has a significant number of observations, while the testing set is large enough to give reliable estimates of model performance.
3. Build the decision tree: Choose an appropriate algorithm to build the decision tree. There are various algorithms available such as ID3, C4.5, CART, and CHAID. Based on the algorithm, train the decision tree model on the training dataset.
4. Evaluate the model: Evaluate the performance of the model on the testing dataset. This can be done by calculating metrics such as accuracy, precision, recall, F1 score, and confusion matrix.
5. Tune the model: If the performance of the model is not satisfactory, tune the model by changing the hyperparameters such as the maximum depth of the tree, minimum samples per leaf, and splitting criteria.
6. Visualize the decision tree: Visualize the decision tree to understand the rules that the model has learned. This will help in identifying the important variables and rules that are used to classify the data.
7. Predict the outcome: Use the trained model to predict the outcome of new observations. This can be done by passing the new observation through the decision tree and following the rules to reach the final outcome.

c) How to evaluate the above algorithm?

Decision tree algorithms can be evaluated using various metrics. Here are some commonly used methods for evaluating a decision tree algorithm:

1. Accuracy: Accuracy measures the proportion of correct predictions made by the model. It is calculated as the number of correct predictions divided by the total number of predictions.
2. Confusion Matrix: A confusion matrix shows the number of true positives, true negatives, false positives, and false negatives. It helps to evaluate the performance of the model on each class and calculate metrics such as precision, recall, and F1 score.
3. Precision: Precision measures the proportion of correct positive predictions made by the model out of all the positive predictions. It is calculated as true positives divided by the sum of true positives and false positives.
4. Recall: Recall measures the proportion of true positives correctly identified by the model out of all the actual positive observations. It is calculated as true positives divided by the sum of true positives and false negatives.

6) Implement PCA using R/Python

a) What is Dimension reduction?

Dimension reduction in PCA (Principal Component Analysis) is the process of reducing the number of variables or features in a dataset while retaining most of the important information contained in the data. PCA is a popular technique used for dimensionality reduction that aims to transform a high-dimensional dataset into a lower-dimensional space.

PCA works by finding the principal components of the dataset, which are the linear combinations of the original variables that explain the maximum variance in the data. The first principal component explains the most variance, followed by the second, and so on. The number of principal components to keep is determined by the amount of variance that needs to be explained.

After finding the principal components, PCA creates a new set of variables that are linear combinations of the original variables. These new variables are called the principal components, and they are orthogonal to each other.

b) What are different methods for dimension reduction?

There are several methods for dimension reduction, including:

1. Principal Component Analysis (PCA): PCA is a popular unsupervised method that identifies the most important linear combinations of the original variables that explain the most variance in the data.
2. Linear Discriminant Analysis (LDA): LDA is a supervised method that aims to find linear combinations of the original variables that maximize the separation between different classes or categories.
3. t-Distributed Stochastic Neighbor Embedding (t-SNE): t-SNE is a non-linear dimension reduction technique that aims to preserve the local structure of the data in a low-dimensional space.
4. Independent Component Analysis (ICA): ICA is a method that aims to separate a multivariate signal into independent, non-Gaussian components.
5. Autoencoders: Autoencoders are neural network architectures that aim to learn a compressed representation of the input data.
6. Factor Analysis: Factor Analysis is a method that aims to identify the underlying factors that explain the correlations between observed variables.

7. Non-negative Matrix Factorization (NMF): NMF is a method that factorizes a non-negative matrix into two non-negative matrices that represent a compressed and interpretable version of the original data.

c) Why Dimension reduction is important?

1. Reducing computational complexity: High-dimensional datasets can be computationally intensive and time-consuming to analyze, especially when using machine learning algorithms. By reducing the dimensionality of the dataset, we can speed up the analysis process and make it more feasible.
2. Handling sparse data: In some cases, datasets may have many variables, but most of them are empty or have very few values. By reducing the dimensionality, we can focus on the variables that contain the most information and discard the ones that are not useful.
3. Improving model performance: Some machine learning algorithms may suffer from the curse of dimensionality, where increasing the number of variables leads to overfitting and poor generalization performance. By reducing the dimensionality, we can improve the model's ability to generalize to new data.
4. Improving visualization: High-dimensional data is difficult to visualize in 2D or 3D. By reducing the dimensionality, we can project the data into a lower-dimensional space that can be easily visualized, making it easier to identify patterns and relationships.
5. Interpreting the data: High-dimensional datasets can be difficult to interpret, and it may not be clear which variables are the most important. By reducing the dimensionality, we can identify the most important variables or features that are driving the patterns in the data, which can help us to understand the underlying mechanisms or processes.

d) Implement PCA Algorithm on suitable dataset.

Import necessary libraries:

```
import numpy as np

import pandas as pd

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

import matplotlib.pyplot as plt
```

Load the dataset:

```
data = pd.read_csv("dataset.csv")
```

Preprocess the data:

```
# Separate the target variable from the feature variables
X = data.drop('target', axis=1)

# Scale the data to have zero mean and unit variance
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Perform PCA:

```
# Create a PCA object with the desired number of components
pca = PCA(n_components=2)

# Fit the PCA model to the data and transform it into the reduced space
X_pca = pca.fit_transform(X_scaled)

# Print the explained variance ratio for each principal component
print("Explained variance ratio:", pca.explained_variance_ratio_)
```

Visualize the results:

```
# Plot the reduced data in the reduced space
```

```
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=data['target'], cmap='viridis')

plt.xlabel('Principal Component 1')

plt.ylabel('Principal Component 2')

plt.show()
```

e) How to evaluate the above algorithm?

Evaluating the performance of a PCA algorithm can be done using several metrics, including:

1. Explained variance ratio: This measures the proportion of the total variance in the data that is explained by each principal component. The cumulative sum of these ratios can be plotted to determine how many principal components are needed to explain a certain percentage of the total variance.
2. Scree plot: This plots the eigenvalues of each principal component in descending order. A sharp drop in the eigenvalues indicates the point at which additional principal components do not explain a significant amount of additional variance.
3. Reconstruction error: This measures how well the reduced-dimensional data can be reconstructed back to the original dataset. This can be done by calculating the difference between the original data and the reconstructed data in terms of mean squared error (MSE) or root mean squared error (RMSE).
4. Visualization: Plotting the reduced-dimensional data can also be helpful in evaluating the algorithm's performance. If the reduced data is well separated and shows distinct groupings or patterns, it is a good indication that the algorithm is performing well.
5. Cross-validation: Cross-validation can be used to evaluate the performance of the PCA algorithm. This involves splitting the data into training and test sets and using the training set to perform PCA. The reduced-dimensional training set can then be used to train a model, and the performance of the model can be evaluated on the test set. This can help to determine whether the reduced-dimensional data is able to accurately represent the original data and whether it improves model performance.

Overall, evaluating the performance of a PCA algorithm requires careful consideration of the specific problem being solved and the characteristics of the data. By using a combination of these techniques, it is possible to gain a better understanding of how well the algorithm is performing and identify areas for improvement.

7) Implement K-means clustering using R/Python

a) What is clustering?

Clustering is a technique in machine learning and data analysis that involves grouping together similar objects or data points based on their characteristics or features. The goal of clustering is to identify patterns and structures in data without prior knowledge of what these patterns and structures may be.

In clustering, each data point is assigned to a cluster or group based on its similarity to other data points in the same cluster, while data points in different clusters are dissimilar. Clustering can be used for a variety of applications, such as market segmentation, image segmentation, anomaly detection, and recommendation systems.

b) Write steps of K-means clustering algorithm.

1. Initialize the algorithm:
- Select the number of clusters, K, that you want to partition your data into.
- Randomly initialize K cluster centroids.
2. Assign each data point to the nearest centroid:
- Compute the distance between each data point and each cluster centroid.
- Assign each data point to the cluster corresponding to the nearest centroid.
3. Recompute the centroid of each cluster:
- Compute the mean of all data points assigned to each cluster.
- Set the centroid of each cluster to the mean of its assigned data points.
4. Repeat steps 2-3 until convergence:
- Check for convergence by calculating the difference between the new and old centroid positions.
- If the difference is below a certain threshold, then stop the algorithm.
- Otherwise, repeat steps 2-3 until convergence.
5. Output the final cluster assignments:
- Assign each data point to the cluster corresponding to its nearest centroid after convergence.

c) How to determine best value of k?

1. Elbow Method:
- Plot the sum of squared distances (SSD) of data points to their nearest centroid for each value of K.
- The optimal value of K is where the SSD begins to level off or form an elbow shape, indicating diminishing returns for increasing K.
2. Silhouette Score:

- Compute the average silhouette score for each value of K.
- The silhouette score measures the cohesion and separation of clusters, with values ranging from -1 to 1. A higher score indicates better clustering.
- The optimal value of K is where the silhouette score is highest.

3. Gap Statistic:

- Compare the SSD of the data with the expected SSD of a null reference distribution for each value of K.
- The expected SSD is calculated from a reference dataset with the same dimensions and number of data points as the original dataset, but with random values.
- The optimal value of K is where the gap statistic is largest, indicating that the observed clustering structure is more significant than the random structure.

d) Implement K-means clustering on suitable dataset.

e) How to evaluate the above algorithm?

Evaluating the performance of a K-means clustering algorithm can be done using several metrics, including:

1. Within-cluster sum of squares (WSS) or sum of squared errors (SSE): This measures the sum of the squared distances between each data point and its assigned centroid. A lower WSS or SSE indicates better cluster cohesion.
2. Silhouette coefficient: This measures the quality of a clustering solution by computing the distance between data points within a cluster and the distance between data points in different clusters. A higher silhouette coefficient indicates better cluster separation.
3. Calinski-Harabasz index: This measures the ratio of between-cluster variance to within-cluster variance. A higher Calinski-Harabasz index indicates better cluster separation.
4. Davies-Bouldin index: This measures the average similarity between each cluster and its most similar cluster. A lower Davies-Bouldin index indicates better cluster separation.
5. Visual inspection: Plotting the data points and their assigned clusters can also be helpful in evaluating the algorithm's performance. If the clusters are well-separated and the data points within each cluster are tightly grouped, it is a good indication that the algorithm is performing well.

8) Implement Time-series forecasting using R/Python

a) What is time-series data? Give example.

Time-series data is a type of data that is collected over time, where each observation is recorded at a specific point in time. In time-series data, the time variable is usually the independent variable, and the dependent variable is the quantity being measured or observed.

Examples of time-series data include stock prices, weather data, sales data, and website traffic data. In these examples, the values are recorded at different time intervals such as daily, hourly, or monthly.

Time-series data can be analyzed using various statistical methods such as trend analysis, seasonality analysis, and forecasting. These methods can help to identify patterns and trends in the data, detect anomalies, and make predictions about future values.

One of the key characteristics of time-series data is that it is often correlated with its own past values, which is known as autocorrelation.

b) Define the problem.

The problem statement for time series forecasting involves predicting the future values of a time series based on its past values. The objective is to develop a model that can accurately capture the underlying patterns and trends in the data and make reliable predictions about its future behavior.

The problem statement typically includes the following:

1. A time series dataset that includes historical observations recorded at regular intervals over time.
2. A target variable that represents the quantity being measured or observed in the dataset.
3. A set of predictor variables that can be used to capture the underlying patterns and trends in the data.
4. The goal is to develop a model that can accurately predict the future values of the target variable based on the values of the predictor variables.
5. The quality of the time series forecast is typically evaluated using metrics such as mean squared error (MSE), mean absolute error (MAE), or root mean squared error (RMSE).

c) Implement Time-series forecasting on suitable dataset.

1. Data Preparation: This involves cleaning and pre-processing the time series data to ensure that it is suitable for modeling. This may include identifying and handling missing values, outliers, and anomalies. The data may also be transformed and scaled to ensure that it is on the same scale and has a similar distribution.
2. Data Visualization: This involves visualizing the time series data to gain insights into its patterns and trends. This may involve plotting the time series data, identifying trends, seasonality, and other patterns.
3. Model Selection: This involves selecting an appropriate time series forecasting model that can capture the underlying patterns and trends in the data. Common models include ARIMA, exponential smoothing, and neural networks.
4. Model Training: This involves fitting the selected model to the training data and estimating the model parameters.
5. Model Evaluation: This involves evaluating the performance of the trained model on a validation dataset using appropriate evaluation metrics such as mean squared error (MSE), mean absolute error (MAE), or root mean squared error (RMSE).
6. Model Tuning and Forecasting: This involves fine-tuning the model parameters based on the performance on the validation dataset and using the model to make predictions on new, unseen data.

d) How to evaluate the above algorithm?

Splitting the data: One common approach is to split the time series data into two parts: a training set and a test set. The training set is used to train the algorithm, and the test set is used to evaluate its performance. The test set should contain data that the algorithm has not seen during training.
Cross-validation: Cross-validation is a technique that involves repeatedly splitting the data into training and test sets and evaluating the algorithm's performance on each split. This can help to reduce the impact of outliers and other sources of noise in the data.
Visualization: Visualizing the results of the algorithm can also be helpful in evaluating its performance. This may involve plotting the predicted values against the actual values or examining the residuals (i.e., the difference between the predicted and actual values) to identify patterns or trends.

Overall, evaluating the performance of a time series algorithm requires careful consideration of the specific problem being solved and the characteristics of the data. By using a combination of these techniques, it is possible to gain a better understanding of how well the algorithm is performing and identify areas for improvement.

9) Titanic Theory: -

The titanic and titanic2 data frames describe the survival status of individual passengers on the Titanic. The titanic data frame does not contain information from the crew, but it does contain actual ages of half of the passengers. The principal source for data about Titanic passengers is the Encyclopedia Titanica. The datasets used here were begun by a variety of researchers. One of the original sources is Eaton & Haas (1994) Titanic: Triumph and Tragedy, Patrick Stephens Ltd, which includes a passenger list created by many researchers and edited by Michael A. Findlay. The variables on our extracted dataset are pclass, survived, name, age, embarked, home.dest, room, ticket, boat, and sex. pclass refers to passenger class (1st, 2nd, 3rd), and is a proxy for socio-economic class. Age is in years, and some infants had fractional values. The titanic2 data frame has no missing data and includes records for the crew, but age is dichotomized at adult vs. child. These data were obtained from Robert Dawson, Saint Mary's University, E-mail. The variables are pclass, age, sex, survived. These data frames are useful for demonstrating many of the functions in Hmisc as well as demonstrating binary logistic regression analysis using the Design library. For more details and references see Simonoff, Jeffrey S (1997): The "unusual episode" and a second statistics course. J Statistics Education, Vol. 5 No. 1. Thomas Cason of UVa has greatly updated and improved the titanic data frame using the Encyclopedia Titanica and created a new dataset called titanic3. These datasets reflects the state of data available as of 2 August 1999. Some duplicate passengers have been dropped, many errors corrected, many missing ages filled in, and new variables created. Click here for information about how this datatset was constructed