Task 50: Machine Learning Concepts and Database Handling

Series vs. Dataframes

- **Series**: A one-dimensional array with axis labels (similar to a column in a table).
    - Example: A list of numbers, names, or dates.
- **DataFrame**: A two-dimensional, size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns).
    - Example: A table with multiple columns, where each column is a Series.

Create Database and Table in MySQL and Read with Pandas

Create Database and Table with Dummy Data:

```
CREATE DATABASE Travel_Planner;
USE Travel_Planner;

CREATE TABLE bookings (
    user_id INT,
    flight_id INT,
    hotel_id INT,
    activity_id INT,
    booking_date DATE
);

INSERT INTO bookings (user_id, flight_id, hotel_id, activity_id,
booking_date) VALUES
(1, 101, 201, 301, '2024-01-01'),
(2, 102, 202, 302, '2024-01-02'),
(3, 103, 203, 303, '2024-01-03');
```

1.

Read Table Content Using Pandas:

```
import pandas as pd
import mysql.connector

# Establish connection to MySQL
connection = mysql.connector.connect(
    host='localhost',
    user='your_username',
    password='your_password',
    database='Travel_Planner'
)
```

```
# Query to fetch data
query = "SELECT * FROM bookings"

# Read the data into a DataFrame
df = pd.read_sql(query, connection)
connection.close()

print(df)
```

2.

Difference Between `loc` and `iloc`

- **`loc`**: Label-based data selection. Use when you know the label name of the row/column you want to select.
    - Example: `df.loc[1, 'column_name']` selects the value in the row with label 1 and column named 'column_name'.
- **`iloc`**: Integer position-based data selection. Use when you know the position (index) of the row/column you want to select.
    - Example: `df.iloc[0, 1]` selects the value at the first row and second column (0-based index).

Supervised vs. Unsupervised Learning

- **Supervised Learning**: Uses labeled data to train the model. The goal is to predict outcomes for new data based on learned relationships.
    - Examples: Classification (e.g., spam detection), Regression (e.g., house price prediction).
- **Unsupervised Learning**: Uses unlabeled data. The goal is to find hidden patterns or intrinsic structures in the input data.
    - Examples: Clustering (e.g., customer segmentation), Dimensionality Reduction (e.g., PCA).

Bias-Variance Tradeoff

- **Bias**: Error due to overly simplistic models that fail to capture the underlying trend of the data.
- **Variance**: Error due to overly complex models that capture noise in the data as if it were a real signal.
- **Tradeoff**: Balancing bias and variance is crucial for model performance. Too much bias leads to underfitting, and too much variance leads to overfitting.

Precision and Recall

- **Precision**: The ratio of correctly predicted positive observations to the total predicted positives.

    - Formula: Precision=TPTP+FP\text{Precision} = \frac{TP}{TP + FP}Precision=TP+FPTP
- **Recall**: The ratio of correctly predicted positive observations to all observations in the actual class.
    - Formula: Recall=TPTP+FN\text{Recall} = \frac{TP}{TP + FN}Recall=TP+FNTP
- **Accuracy**: The ratio of correctly predicted observations to the total observations.
    - Formula: Accuracy=TP+TNTP+TN+FP+FN\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}Accuracy=TP+TN+FP+FNTP+TN

Overfitting and Prevention

- **Overfitting**: A model performs well on training data but poorly on new, unseen data.
- **Prevention Methods**:
    - Use cross-validation.
    - Prune the model (for decision trees).
    - Use regularization techniques (L1, L2).
    - Simplify the model (reduce the number of features).

Cross-Validation

- **Concept**: A technique for assessing how a model will generalize to an independent dataset. It involves partitioning the data into subsets, training the model on some subsets (training set) and validating it on the remaining subset (validation set).
- **Types**: k-fold cross-validation, stratified k-fold cross-validation, leave-one-out cross-validation.

Classification vs. Regression

- **Classification**: Predicts categorical outcomes (e.g., spam or not spam).
- **Regression**: Predicts continuous outcomes (e.g., price of a house).

Ensemble Learning

- **Concept**: Combines multiple models to improve overall performance.
- **Methods**: Bagging (e.g., Random Forests), Boosting (e.g., Gradient Boosting), Stacking.

Gradient Descent

- **Concept**: An optimization algorithm used to minimize the cost function in machine learning models.
- **Process**: Iteratively adjusts the model parameters in the opposite direction of the gradient of the cost function.

Batch vs. Stochastic Gradient Descent

- **Batch Gradient Descent**: Uses the entire dataset to compute the gradient.
    - Pros: More stable convergence.
    - Cons: Computationally expensive for large datasets.

- **Stochastic Gradient Descent (SGD)**: Uses one data point at a time to compute the gradient.
    - Pros: Faster, can handle large datasets.
    - Cons: More noise in the convergence process.

Curse of Dimensionality

- **Concept**: As the number of features increases, the volume of the space increases exponentially, leading to sparse data and making models prone to overfitting and difficulty in finding patterns.

L1 vs. L2 Regularization

- **L1 Regularization (Lasso)**: Adds the absolute value of coefficients as a penalty term to the loss function. Can shrink coefficients to zero, leading to sparse models.
- **L2 Regularization (Ridge)**: Adds the squared value of coefficients as a penalty term to the loss function. Tends to shrink coefficients but not to zero.

Confusion Matrix

- **Concept**: A table used to describe the performance of a classification model.
- **Components**:
    - True Positive (TP)
    - False Positive (FP)
    - True Negative (TN)
    - False Negative (FN)

AUC-ROC Curve

- **Concept**: A graphical plot illustrating the diagnostic ability of a binary classifier system.
- **AUC (Area Under the Curve)**: Measures the entire two-dimensional area underneath the entire ROC curve. Higher AUC indicates better model performance.

k-Nearest Neighbors Algorithm

- **Concept**: A non-parametric method used for classification and regression.
- **Process**: Classifies a data point based on how its neighbors are classified.

Support Vector Machine (SVM)

- **Concept**: A supervised learning algorithm used for classification and regression tasks.
- **Kernel Trick**: Transforms data into a higher dimension to make it easier to classify using linear separation.
- **Types of Kernels**:
    - Linear
    - Polynomial
    - Radial Basis Function (RBF)
    - Sigmoid

Decision Tree

- **Construction**: Splits the data into subsets based on the most significant attribute, recursively.
- **Information Gain**: Measures the reduction in entropy. Used to determine which attribute to split.
- **Gini Impurity**: Measures the likelihood of an incorrect classification of a new instance.

Random Forest

- **Concept**: An ensemble of decision trees.
- **Bootstrapping**: Sampling with replacement to create different subsets of data for training each tree.
- **Feature Importance**: Determines the importance of each feature in predicting the target variable.

Logistic Regression

- **Concept**: A regression model for binary classification.
- **Sigmoid Function**: Maps predicted values to probabilities.
- **Cost Function**: Measures the error between predicted and actual values.

XGBoost

- **Concept**: An optimized distributed gradient boosting library.
- **Advantages**: Speed and performance, handles missing values, regularization.
- **Gradient Boosting Process**: Sequentially adds models to correct errors made by existing models.

Example Code for Reading MySQL Table Using Pandas

Here is the code again for reference:

```
import pandas as pd
import mysql.connector

# Establish connection to MySQL
connection = mysql.connector.connect(
    host='localhost',
    user='your_username',
    password='your_password',
    database='Travel_Planner')

# Query to fetch data
query = "SELECT * FROM bookings"
```

```
# Read the data into a DataFrame
df = pd.read_sql(query, connection)
connection.close()

print(df)
```

Conclusion

This covers the requested topics, from basic database operations to advanced machine learning concepts. Let me know if you need any further explanations or details!