bytewise.

Bytewise Fellowship Program

DATA SCIENCE

Task 13

BWT- Data Science (Group1)

Submitted to: Mahrukh Khan
Submitted by: Usama Malik

Task: Handling Missing Data, Filling and Replacing Values, Removing Duplicates, Detecting and Removing Outliers. Decision Trees

1. Handling Missing Data

Definition: Missing data can occur when no value is stored for a variable in an observation.

Example:

```
import pandas as pd
import numpy as np
# Sample DataFrame with missing values
data = {
   'Name': ['John', 'Anna', 'Peter', 'Linda', 'James'],
   'Age': [28, 24, np.nan, 32, 29],
   'Salary': [50000, 54000, 58000, np.nan, 62000]
}
df = pd.DataFrame(data)
print(df)
# Filling missing values
df_filled = df.fillna(df.mean())
print(df_filled)
```

2. Removing Duplicates

Definition: Duplicates are repeated rows in the dataset that need to be removed to ensure data quality.

Example:

```
# Sample DataFrame with duplicates
data = {
   'Name': ['John', 'Anna', 'Peter', 'Anna', 'James'],
   'Age': [28, 24, 30, 24, 29],
```

```
    'Salary': [50000, 54000, 58000, 54000, 62000]
}
df = pd.DataFrame(data)
print(df)

# Removing duplicates
df_no_duplicates = df.drop_duplicates()
print(df_no_duplicates)
```

3. Detecting and Removing Outliers

Definition: Outliers are data points that are significantly different from other observations.

Example:

```
# Sample DataFrame with outliers
data = {
    'Name': ['John', 'Anna', 'Peter', 'Linda', 'James'],
    'Age': [28, 24, 30, 32, 150],
    'Salary': [50000, 54000, 58000, 60000, 62000]
}
df = pd.DataFrame(data)
print(df)

# Removing outliers using Z-score
from scipy import stats

z_scores = np.abs(stats.zscore(df[['Age', 'Salary']]))
df_no_outliers = df[(z_scores < 3).all(axis=1)]
print(df_no_outliers)
```

Decision Trees (Chapter 6)

1. Training and Visualizing a Decision Tree

Definition: Training a decision tree involves splitting the data into subsets based on feature values, and visualizing helps understand the tree structure.

Example:

```
from sklearn.datasets import load_iris

from sklearn.tree import DecisionTreeClassifier

from sklearn import tree

import matplotlib.pyplot as plt


iris = load_iris()

X, y = iris.data, iris.target


# Train a decision tree

clf = DecisionTreeClassifier()

clf.fit(X, y)


# Visualize the decision tree

plt.figure(figsize=(20,10))

tree.plot_tree(clf, filled=True, feature_names=iris.feature_names,
class_names=iris.target_names)

plt.show()
```

2. Making Predictions

Definition: Using the trained decision tree to predict the class labels of new data points.

Example:

```
# Predicting new data points

new_data = [[5.0, 3.6, 1.4, 0.2]]

prediction = clf.predict(new_data)
```

print(f'Prediction: {prediction}')

3. Estimating Class Probabilities

Definition: Decision trees can provide the probability of each class for a given input.

Example:

 # Estimating class probabilities

probabilities = clf.predict_proba(new_data)

print(f'Class Probabilities: {probabilities}')

4. The CART Training Algorithm

Definition: Classification and Regression Trees (CART) is an algorithm that splits the data into subsets to build a decision tree.

Example: The DecisionTreeClassifier in scikit-learn uses the CART algorithm.

5. Computational Complexity

Definition: The computational complexity of a decision tree is related to the depth of the tree and the number of features.

6. Gini Impurity or Entropy

Definition: Metrics used to measure the quality of splits in the decision tree. Gini impurity is often used by default in CART.

Example:

 # Using entropy instead of Gini impurity

clf_entropy = DecisionTreeClassifier(criterion='entropy')

clf_entropy.fit(X, y)

7. Regularization Hyperparameters

Definition: Parameters like max_depth, min_samples_split, and min_samples_leaf used to control the complexity of the tree and avoid overfitting.

Example:

 # Setting regularization hyperparameters

clf_reg = DecisionTreeClassifier(max_depth=3, min_samples_split=5)

clf_reg.fit(X, y)

8. Regression

Definition: Decision trees can also be used for regression tasks to predict continuous values.

Example:

from sklearn.tree import DecisionTreeRegressor

 # Sample regression data

X = np.array([[1], [2], [3], [4], [5]])

y = np.array([1.2, 1.8, 3.6, 3.8, 5.1])

 # Train a decision tree regressor

reg = DecisionTreeRegressor()

reg.fit(X, y)

 # Predicting new data points

reg_prediction = reg.predict([[6]])

print(f'Regression Prediction: {reg_prediction}')

9. Instability

Definition: Decision trees can be unstable because small variations in the data can result in different splits and, consequently, different trees.