ASSIGNMENT 9

PROBLEM STATEMENT:

Perform the data classification algorithm using any Classification algorithm.

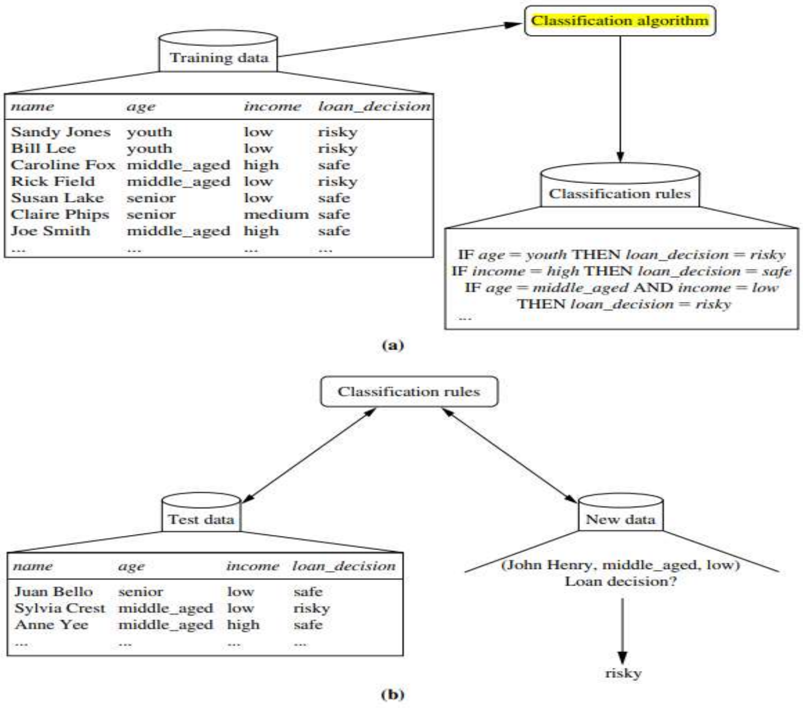
OBJECTIVE:

Students should be able understand the concept of data classification and different data classification algorithm.

THEORY:

Data Classification

- Classification is a form of data analysis that extracts models describing important data classes. Such models, called classifiers, predict categorical (discrete, unordered) class labels.
- For example, we can build a classification model to categorize bank loan applications as either safe or risky.
- Data classification is a two-step process, consisting of a learning step (where a classification model is constructed) and a classification step (where the model is used to predict class labels for given data).
- The process is shown for the loan application data in the following figure:



name	*age*	*income*	*loan_decision*
Sandy Jones	youth	low	risky
Bill Lee	youth	low	risky
Caroline Fox	middle_aged	high	safe
Rick Field	middle_aged	low	risky
Susan Lake	senior	low	safe
Claire Phips	senior	medium	safe
Joe Smith	middle_aged	high	safe
...	...	...	...

IF *age* = *youth* THEN *loan_decision* = *risky*
IF *income* = *high* THEN *loan_decision* = *safe*
IF *age* = *middle_aged* AND *income* = *low*
THEN *loan_decision* = *risky*
...

(a)

name	*age*	*income*	*loan_decision*
Juan Bello	senior	low	safe
Sylvia Crest	middle_aged	low	risky
Anne Yee	middle_aged	high	safe
...	...	...	...

(John Henry, middle_aged, low)
Loan decision?

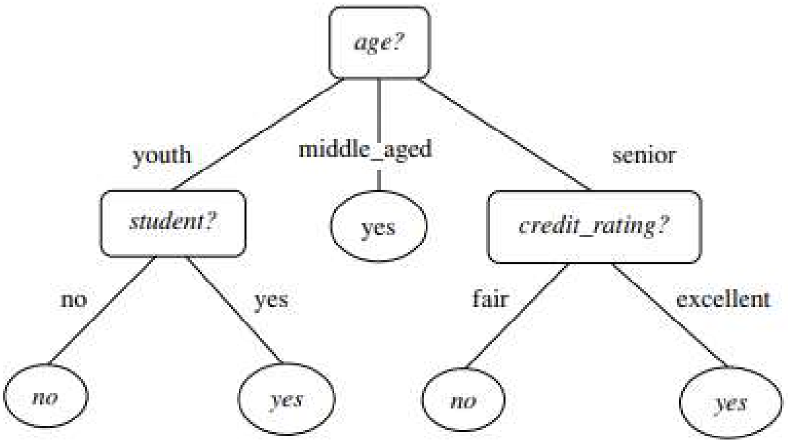
risky

(b)

- The data classification process for above example:
  - Learning: Training data are analyzed by a classification algorithm. Here, the class label attribute is loan decision, and the learned model or classifier is represented in the form of classification rules.
  - Classification: Test data are used to estimate the accuracy of the classification rules. If the accuracy is considered acceptable, the rules can be applied to the classification of new data tuples.

Decision Tree Induction

- Decision tree induction is the learning of decision trees from class-labeled training tuples.
- A decision tree is a flowchart-like tree structure, where each internal node (non leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or terminal node) holds a class label. The topmost node in a tree is the root node.
- A typical decision tree is shown in figure



- A decision tree algorithm known as ID3 (Iterative Dichotomiser).
- ID3, C4.5(a successor of ID3), and CART(Classification and Regression Trees) adopt a greedy (i.e., nonbacktracking) approach in which decision trees are constructed in a top-down recursive divide-and-conquer manner.

Bayes Classification Methods

- Bayesian classifiers are statistical classifiers. They can predict class membership probabilities such as the probability that a given tuple belongs to a particular class.
- Bayesian classification is based on Bayes’ theorem.
- A simple Bayesian classifier known as the na¨ıve Bayesian classifier to be comparable in performance with decision tree and selected neural network classifiers. Bayesian classifiers have also exhibited high accuracy and speed when applied to large databases.

Rule-Based Classification

- Rule-based classifiers, where the learned model is represented as a set of IF-THEN rules.
- An IF-THEN rule is an expression of the form

  IF condition THEN conclusion.

  An example is rule R1

  R1: IF age = youth AND student = yes THEN buys computer = yes.
- The "IF" part (or left side) of a rule is known as the rule antecedent or precondition. The "THEN" part (or right side) is the rule consequent.

Following are the various advanced classification methods:

- **Bayesian belief networks:** which unlike na¨ıve Bayesian classifiers, do not assume class conditional independence.
- **Classification by Backpropagation**: Backpropagation is a neural network learning algorithm. A neural network is a set of connected input/output units in which each connection has a weight associated with it.
- **Support Vector Machines**: a method for the classification of both linear and nonlinear data. It uses a nonlinear mapping to transform the original training data into a higher dimension. Within this new dimension, it searches for the linear optimal separating hyperplane (i.e., a "decision boundary" separating the tuples of one class from another). With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane. The SVM finds this hyperplane using support vectors ("essential" training tuples) and margins (defined by the support vectors).
- **Classification Using Frequent Patterns**: Frequent patterns show interesting relationships between attribute–value pairs that occur frequently in a given data set.

CONCLUSION

In this way we have explored the concept of data classification and implemented the data classification algorithm.

ORAL QUESTION

1. What is the primary objective of a classification algorithm?
2. What are some common types of classification algorithms?
3. How do you evaluate the performance of a classification algorithm?

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report


# Load the iris dataset
iris = load_iris()


# Features
X = iris.data


# Target variable
y = iris.target


# Splitting the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Feature scaling
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)


# Initialize and train the logistic regression model
logreg = LogisticRegression()
logreg.fit(X_train_scaled, y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
# Predictions
y_pred_train = logreg.predict(X_train_scaled)
y_pred_test = logreg.predict(X_test_scaled)


# Model evaluation
train_accuracy = accuracy_score(y_train, y_pred_train)
test_accuracy = accuracy_score(y_test, y_pred_test)


print("Training Accuracy:", train_accuracy)
print("Testing Accuracy:", test_accuracy)
```

```
Training Accuracy: 0.9666666666666667
Testing Accuracy: 1.0
```

```
# Additional evaluation metrics
print("Classification Report on Test Data:")
print(classification_report(y_test, y_pred_test))
```

```
Classification Report on Test Data:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```