```
# Importing necessary libraries
import numpy as np
from sklearn.datasets import fetch_openml
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score

# Load the MNIST dataset
mnist = fetch_openml('mnist_784')

# Extracting features and labels
X, y = mnist['data'], mnist['target']

# Convert X to a NumPy array
X = np.array(X)

# Preprocess the data
# Normalization
X = X / 255.0
# Flattening
X_flat = X.reshape(X.shape[0], -1)

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

# Implementing a supervised learning algorithm (Random Forest)
rf_clf = RandomForestClassifier()
rf_clf.fit(X_train, y_train)

# Making predictions
y_pred_rf = rf_clf.predict(X_test)

# Evaluating the classification model's performance
accuracy_rf = accuracy_score(y_test, y_pred_rf)
precision_rf = precision_score(y_test, y_pred_rf, average='weighted')
recall_rf = recall_score(y_test, y_pred_rf, average='weighted')
print("Supervised Learning (Random Forest) Performance Metrics:")
print("Accuracy:", accuracy_rf)
print("Precision:", precision_rf)
print("Recall:", recall_rf)

# Provide comments and explanations
# MNIST dataset is loaded and preprocessed by normalizing pixel values and flattening the images.
# The dataset is split into training and testing sets.
# A supervised learning algorithm, Random Forest, is implemented and trained on the training data.
# The performance of the model is evaluated using accuracy, precision, and recall metrics.
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/datasets/_openml.py:968: FutureWarning: The default value of `parser` will change fr
  warn(
Supervised Learning (Random Forest) Performance Metrics:
Accuracy: 0.9675
Precision: 0.9675110852996096
Recall: 0.9675
```