Classification using Deep neural network: Multiclass classification using Deep Neural Networks: Example: Use the OCR letter recognition dataset.

```
import numpy as np
from sklearn.datasets import fetch_openml
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.utils import to_categorical

# Load the OCR Letter Recognition dataset
data = fetch_openml(name='letter', version=1, as_frame=True)

# Split the dataset into features and target
X = data.data
y = data.target

# Preprocess the data
label_encoder = LabelEncoder()
y = label_encoder.fit_transform(y)
y = to_categorical(y)

# Split the data 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)

# Scale the data for better convergence
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Define the model
model = Sequential()
model.add(Dense(128, activation='relu', input_shape=(X_train.shape[1],)))
model.add(Dense(64, activation='relu'))
model.add(Dense(len(label_encoder.classes_), activation='softmax'))

# Compile the model
model.compile(optimizer=Adam(learning_rate=0.001), loss='categorical_crossentropy', metrics=['accuracy'])

# Train the model
model.fit(X_train, y_train, batch_size=32, epochs=10, verbose=1)

# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test, verbose=0)
print('Test Loss:', loss)
print('Test Accuracy:', accuracy)

# Make predictions
predictions = model.predict(X_test)
predicted_labels = np.argmax(predictions, axis=1)
predicted_letters = label_encoder.inverse_transform(predicted_labels)

# Print some predicted and actual letters
```

```
for i in range(10):
    predicted_label = predicted_letters[i]
    actual_label = label_encoder.inverse_transform([np.argmax(y_test[i])])
    print('Predicted:', predicted_label, 'Actual:', actual_label)
```