Machine Learning Training Module

A Comprehensive Guide for Beginners

Prepared by xAI Training Team

May 07, 2025

Downloadable PDF version available for training purposes.

Contents

1 Introduction to Machine Learning

Machine Learning (ML) is a field of artificial intelligence that enables systems to learn from data and improve over time. This module introduces ML concepts—supervised and unsupervised learning, model evaluation, and more—with Python examples using scikit-learn.

1.1 Why Study Machine Learning?

- **Automation**: Enables predictive and decision-making systems.
- **Applications**: Used in healthcare, finance, and autonomous systems.
- **Innovation**: Drives advancements in AI and data science.

1.2 Setting Up the Environment

Install Python and libraries like scikit-learn, numpy, and pandas. Use a Jupyter Notebook or IDE like PyCharm. Install dependencies with:

```
pip install scikit-learn numpy pandas matplotlib
```

2 Supervised Learning

Supervised learning uses labeled data to predict outcomes.

2.1 Linear Regression Example

Predict a continuous variable using linear regression.

```python
import numpy as np
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

# Sample data
X = np.array([[1], [2], [3], [4], [5]])
y = np.array([2, 4, 5, 4, 5])

# Train model
model = LinearRegression()
model.fit(X, y)

# Predict
y_pred = model.predict(X)

# Plot
plt.scatter(X, y, color=’blue’, label=’Data’)
plt.plot(X, y_pred, color=’red’, label=’Fit’)
plt.xlabel(’X’)
plt.ylabel('y')
plt.legend()
plt.savefig('linear_regression.png')
```

Explanation:

- `LinearRegression`: Fits a linear model.
- `model.fit`: Trains the model on input data.
- `plt.savefig`: Saves the plot as an image.

3 Classification

Classification predicts discrete labels.

3.1 Logistic Regression Example

Classify data using logistic regression.

```
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

# Load data
iris = load_iris()
X, y = iris.data, iris.target

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
    =0.2, random_state=42)

# Train model
model = LogisticRegression(max_iter=200)
model.fit(X_train, y_train)

# Evaluate
accuracy = model.score(X_test, y_test)
print(f"Accuracy: {accuracy:.2f}")
```

4 Unsupervised Learning

Unsupervised learning finds patterns in unlabeled data.

4.1 K-Means Clustering Example

Group data into clusters.

```
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

# Sample data
X = np.array([[1, 2], [1.5, 1.8], [5, 8], [8, 8], [1, 0.6], [9, 11]])

# Train model
kmeans = KMeans(n_clusters=2, random_state=42)
kmeans.fit(X)

# Plot clusters
plt.scatter(X[:, 0], X[:, 1], c=kmeans.labels_, cmap=’viridis’)
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=200, c=’red’, marker=’X’)
plt.xlabel(’Feature 1’)
plt.ylabel(’Feature 2’)
plt.savefig(’kmeans_clustering.png’)
```

5 Model Evaluation

Evaluate models to ensure reliability.

5.1 Cross-Validation Example

Use cross-validation to assess model performance.

```
from sklearn.datasets import load_iris
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import RandomForestClassifier

# Load data
iris = load_iris()
X, y = iris.data, iris.target

# Train model
model = RandomForestClassifier(random_state=42)
scores = cross_val_score(model, X, y, cv=5)

# Print results
print(f”Cross-validation scores: {scores}”)
print(f”Average accuracy: {scores.mean():.2f}”)
```

6 Feature Engineering

Feature engineering improves model performance by transforming data.

6.1 Standard Scaling Example

Normalize features using StandardScaler.

```python
from sklearn.preprocessing import StandardScaler
import numpy as np

# Sample data
X = np.array([[1, 2], [3, 4], [5, 6], [7, 8]])

# Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Print results
print(”Scaled data:”)
print(X_scaled)
```

7 Decision Trees

Decision trees make decisions based on feature splits.

7.1 Decision Tree Classifier Example

Classify data using a decision tree.

```python
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split

# Load data
iris = load_iris()
X, y = iris.data, iris.target

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train model
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)

# Evaluate
accuracy = model.score(X_test, y_test)
print(f”Accuracy: {accuracy:.2f}”)
```

8 Neural Networks (Introduction)

Neural networks model complex patterns.

8.1 Simple Neural Network Example

Use scikit-learn’s MLPClassifier for a basic neural network.

```python
from sklearn.neural_network import MLPClassifier
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split

# Load data
iris = load_iris()
X, y = iris.data, iris.target

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train model
model = MLPClassifier(hidden_layer_sizes=(10,), max_iter=1000, random_state=42)
model.fit(X_train, y_train)

# Evaluate
accuracy = model.score(X_test, y_test)
print(f”Accuracy: {accuracy:.2f}”)
```

9 Hyperparameter Tuning

Optimize model performance by tuning hyperparameters.

9.1 Grid Search Example

Use GridSearchCV to find the best parameters.

```python
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC
from sklearn.datasets import load_iris

# Load data
iris = load_iris()
X, y = iris.data, iris.target

# Define parameter grid
param_grid = {’C’: [0.1, 1, 10], ’kernel’: [’linear’, ’rbf’]}

# Perform grid search
model = GridSearchCV(SVC(), param_grid, cv=5)
model.fit(X, y)

# Print results
print(f”Best parameters: {model.best_params_}”)
print(f”Best score: {model.best_score_:.2f}”)
```

10 Conclusion

This module covers machine learning fundamentals—supervised and unsupervised learning, model evaluation, feature engineering, and neural networks. Practice these examples and explore advanced topics like deep learning with TensorFlow or PyTorch.

11 References

- Scikit-learn Documentation: `https://scikit-learn.org`
- Machine Learning, Tom Mitchell