Machine Learning Classification Algorithms

By Md Anique Zzama

Introduction to Classification in ML

- Classification is a supervised learning technique used to predict categorical labels.

Types of Classification Algorithms

- Common types: Logistic Regression, Decision Trees, Random Forest, SVM, KNN, Naïve Bayes, Gradient Boosting, etc.

Step 1: Data Preprocessing

- Handling missing values, encoding categorical variables, feature scaling.

- Code:
- from sklearn.preprocessing import StandardScaler
- scaler = StandardScaler()
- X_scaled = scaler.fit_transform(X)

Step 2: Train-Test Split

- Splitting data into training and testing sets.

- Code:
- from sklearn.model_selection import train_test_split
- X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

Logistic Regression

- A simple linear classifier for binary classification problems.

- Code:
- from sklearn.linear_model import LogisticRegression
- model = LogisticRegression().fit(X_train, y_train)

Decision Tree Classifier

- A tree-based algorithm that splits features to make predictions.

- Code:
- from sklearn.tree import DecisionTreeClassifier
- model = DecisionTreeClassifier().fit(X_train, y_train)

Random Forest Classifier

- An ensemble of decision trees to improve accuracy.

- Code:
- from sklearn.ensemble import RandomForestClassifier
- model = RandomForestClassifier().fit(X_train, y_train)

Support Vector Machine (SVM)

- A model that finds the optimal hyperplane for classification.

- Code:
- from sklearn.svm import SVC
- model = SVC().fit(X_train, y_train)

K-Nearest Neighbors (KNN)

- Classifies data points based on the nearest neighbors.

- Code:
- from sklearn.neighbors import KNeighborsClassifier
- model = KNeighborsClassifier().fit(X_train, y_train)

Naïve Bayes Classifier

- A probabilistic classifier based on Bayes' theorem.

- Code:
- from sklearn.naive_bayes import GaussianNB
- model = GaussianNB().fit(X_train, y_train)

Gradient Boosting Classifier

- Boosting algorithm that combines weak learners.

- Code:
- from sklearn.ensemble import GradientBoostingClassifier
- model = GradientBoostingClassifier().fit(X_train, y_train)

XGBoost Classifier

- Optimized gradient boosting model.

- Code:
- from xgboost import XGBClassifier
- model = XGBClassifier().fit(X_train, y_train)

LightGBM Classifier

- Lightweight gradient boosting model.

- Code:
- from lightgbm import LGBMClassifier
- model = LGBMClassifier().fit(X_train, y_train)

CatBoost Classifier

- Boosting model designed for categorical data.

- Code:
- from catboost import CatBoostClassifier
- model = CatBoostClassifier().fit(X_train, y_train)

Neural Networks for Classification

- Deep learning-based classifiers like MLP.

- Code:
- from tensorflow.keras.models import Sequential
- model = Sequential([...])

Performance Metrics

- Metrics: Accuracy, Precision, Recall, F1-score.

- Code:
- from sklearn.metrics import accuracy_score
- accuracy_score(y_test, y_pred)

Confusion Matrix

- Visualizing classification performance.

- Code:
- from sklearn.metrics import confusion_matrix
- print(confusion_matrix(y_test, y_pred))

ROC Curve & AUC Score

- Evaluating model performance.

- Code:
- from sklearn.metrics import roc_auc_score
- roc_auc_score(y_test, y_pred_prob)

Hyperparameter Tuning

- Improving model performance.

- Code:
- from sklearn.model_selection import GridSearchCV
- GridSearchCV(model, params).fit(X_train, y_train)

Feature Selection

- Selecting the most important features.

- Code:
- from sklearn.feature_selection import SelectKBest
- SelectKBest().fit(X, y)

Overfitting & Underfitting

- Balancing model complexity.

Ensemble Methods

- Using multiple models for better performance.

Comparison of Classification Models

- Pros & cons of different classifiers.

Case Study: Real-World Example

- Applying classification in a real-world scenario.

Deployment of Models

- Deploying models using Flask or Streamlit.

Challenges & Best Practices

- Common pitfalls & how to avoid them.

Future Trends

- The future of classification algorithms.

Final Thoughts & Recommendations

- Choosing the right model for the right task.

Thank You

- By Md Anique Zzama