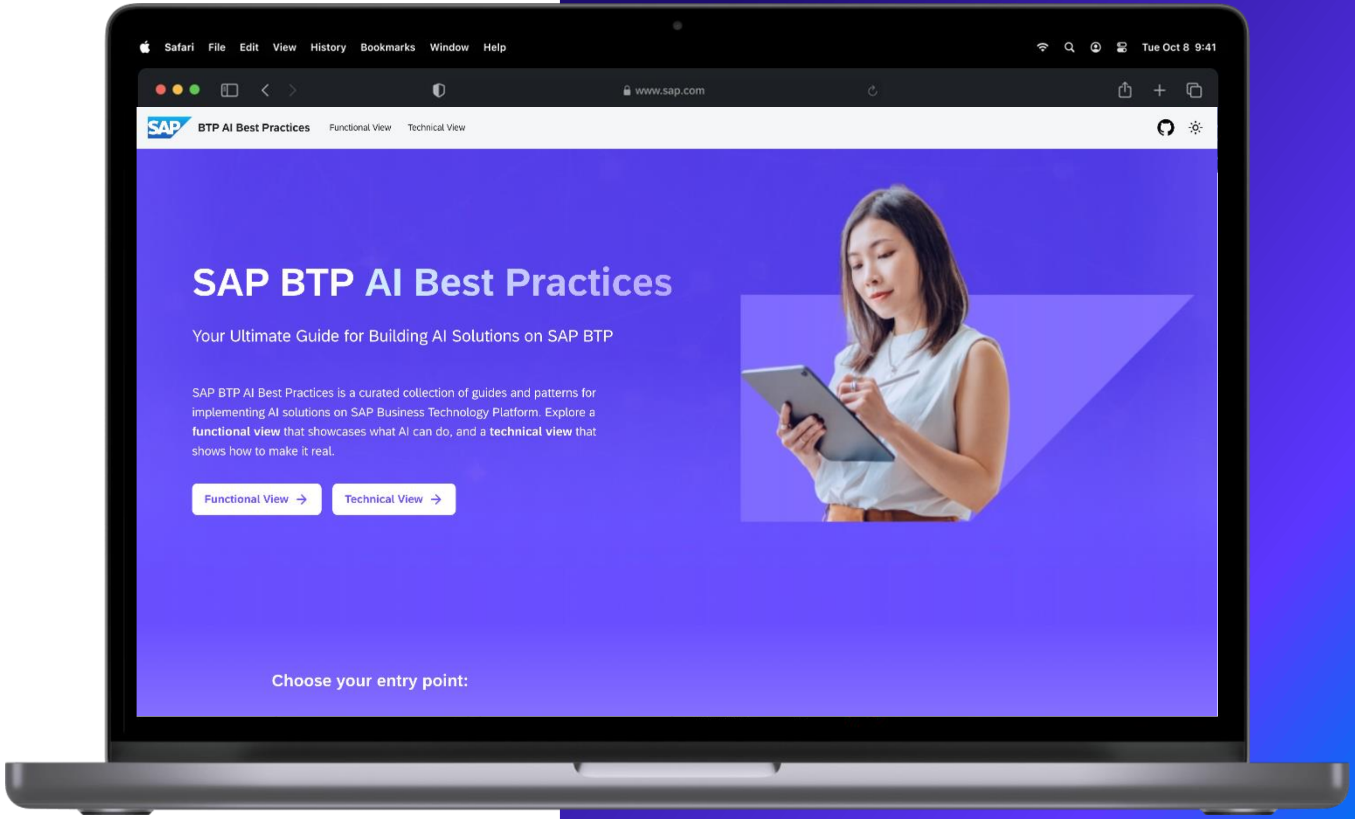
SAP BTP AI Best Practices

Classification

A powerful approach to automate categorization of data into predefined classes, leading to improved decision-making and insights. This is particularly useful for tasks like spam detection, fraud analysis, medical diagnosis., among others.



SAP BTP AI Best Practices

Your Ultimate Guide for Building AI Solutions on SAP BTP

SAP BTP AI Best Practices is a curated collection of guides and patterns for implementing AI solutions on SAP Business Technology Platform. Explore a **functional view** that showcases what AI can do, and a **technical view** that shows how to make it real.

Choose your entry point:

BTP AI Services Center of Excellence

15.09.2025

© 2024 SAP

Steps

1 Overview

2 Pre-requisites

3 Key Choices and Guidelines

4 Implementation

Classification

Powerful approach to categorize data.

Classification is a fundamental machine learning technique aimed at organizing input data into distinct classes through **SAP HANA ML.**

In the SAP ecosystem, this involves leveraging tools within SAP HANA ML (PAL, *hana_ml*) to automate categorization of data.

Expected Outcome

- To recognize patterns in the training data and use these patterns to classify new data into one of the pre-defined classes.
- Optimize business processes by helping to group similar items together while distinguishing them from different ones, leading to improved understanding, prediction, and decision-making.

Key Benefits

Why use SAP HANA ML for Classification?

Algorithm Interchangeability

Easily switch between algorithms (Logistic Regression, Random Decision Trees, Hybrid Gradient Boosting Tree for classification , Support Vector Machine) to best suit your task.

Out-of-the-box Features

Supercharge your development with built-in capabilities for K-Nearest Neighbor (KNN) models , decision tree-based modelling, among many others.

Security & SAP Ecosystem

It's fully integrated into the SAP Ecosystem, leveraging the best of SAP technologies.

Pre-requisites

Supported Environments

- SAP HANA Platform 2.0 SPS 04 or higher
- SAP Hana Cloud (recommended for easier management)
- SAP Datasphere
- SAP Hana express edition (for development and testing)

Required Components

- Application Function Library (AFL) containing PAL and APL
- Script Server enabled for ML algorithms
- Required user authorizations and roles for PAL/APL

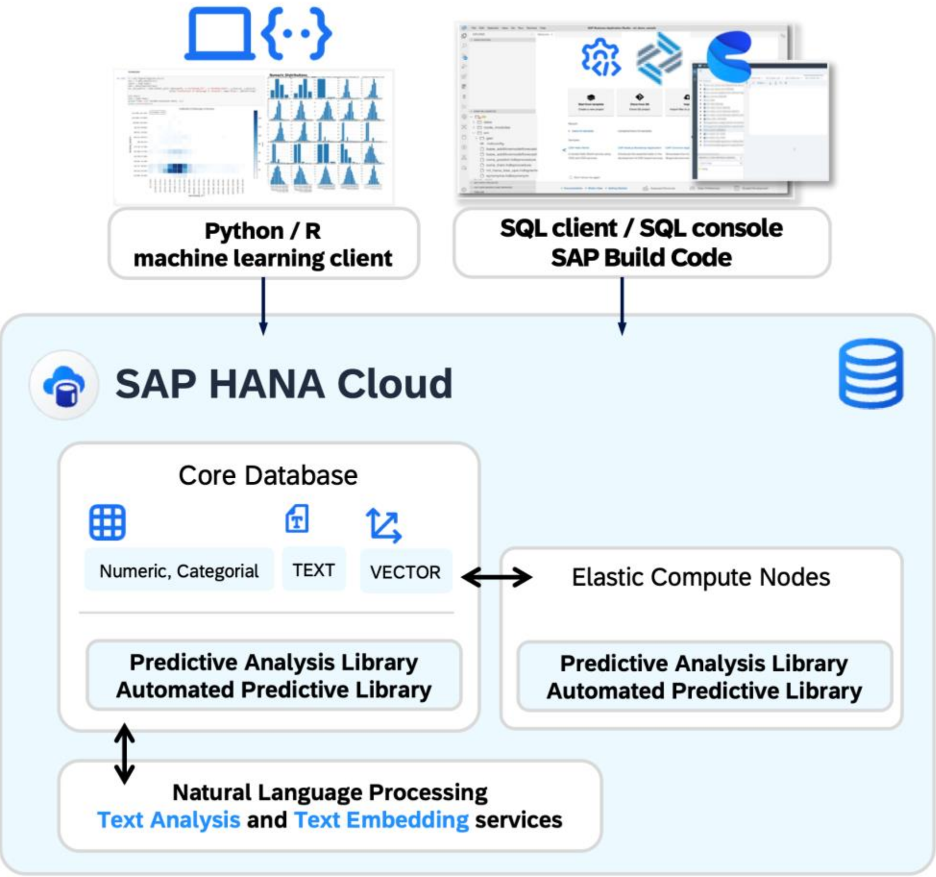
SAP HANA Platform

- SAP HANA is an in-memory database that enables real-time analytics and applications
- The HANA ML libraries (PAL, APL) provide native in-database functions for predictive analysis and Machine Learning.

Script Server

- Auxiliary SAP HANA process responsible for executing application function libraries
- Critical component that must be enabled for PAL and APL functionality
- Serves as execution environment for Machine Learning procedures

High-level reference architecture



Python / R
machine learning client

SQL client / SQL console
SAP Build Code

SAP HANA Cloud

Core Database

Numeric, Categorial

TEXT

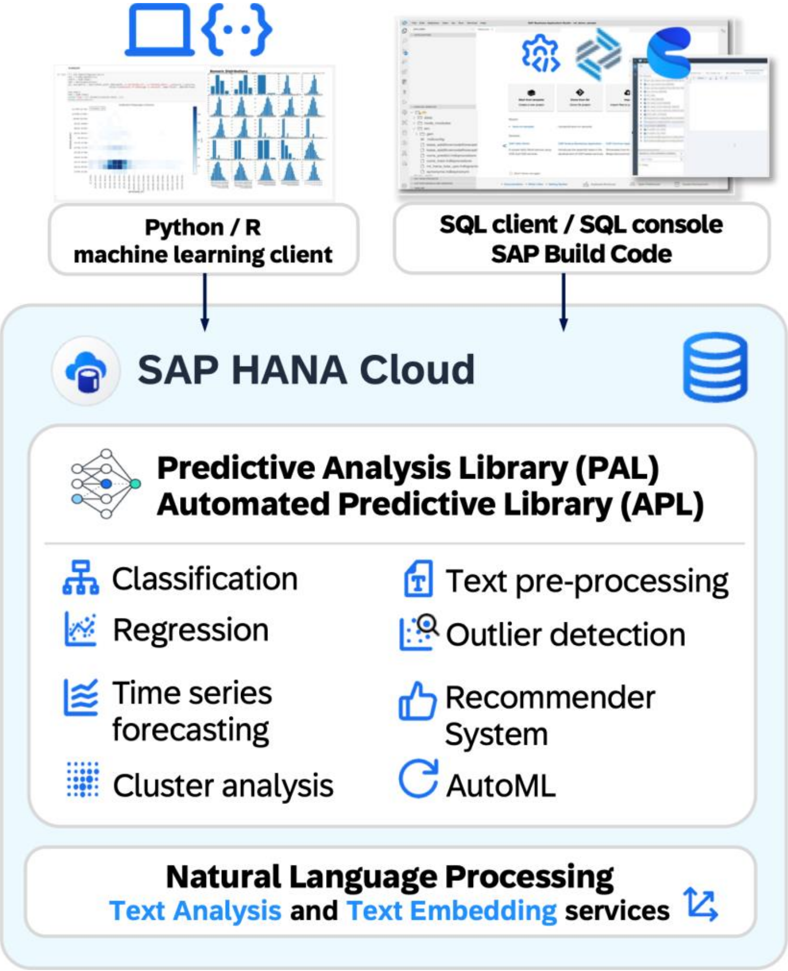
VECTOR

Elastic Compute Nodes

Predictive Analysis Library
Automated Predictive Library

Predictive Analysis Library
Automated Predictive Library

Natural Language Processing
Text Analysis and Text Embedding services



Python / R
machine learning client

SQL client / SQL console
SAP Build Code

SAP HANA Cloud

Predictive Analysis Library (PAL)
Automated Predictive Library (APL)

- Classification
- Regression
- Time series forecasting
- Cluster analysis
- Text pre-processing
- Outlier detection
- Recommender System
- AutoML

Natural Language Processing
Text Analysis and Text Embedding services

Key Choices and Guidelines

Decisions that impact the performance and utility of the application

Algorithm Selection

- With the great variety of algorithms, the classification scenario type as well as data patterns like number of rows (observations), number of features / type of features and cardinality of features influence the selection of classification models and their use cases.

Dataset Size	Algorithm	Description
Small to Medium size data sets	Logistic Regression	Used in simple binary classification problems, such as predicting "yes" or "no" labels.
	k - Nearest Neighbors (k - NN)	Used in cases for easy model interpretability and model explainability.
	Decision Trees	
Medium to Large data sets	Random Forest	Used for complex classification problems where accuracy is highly relevant.
	Support Vector Machines (SVM)	Used for classification scenarios with clear margin of separation, e.g., image classification.
	XGBoost	Used in high performance classification problems, e.g., fraud detection.
	Neural Networks	Also known as Deep Learning, it is used for complex and high dimensional data scenarios, e.g., speech recognition.
	LightGBM	Used for large scale and high-performance applications, e.g., web search ranking.
	Multinomial Naive Bayes	Used for discrete data, specially for text classification tasks where word counts are used as features for sentiment analysis.

Key Choices and Guidelines

Decisions that impact the performance and utility of the application

Algorithm Selection

- With the great variety of algorithms, the classification scenario type as well as data patterns like number of rows (observations), number of features / type of features and cardinality of features influence the selection of classification models and their use cases.

Very Large data sets	Deep Learning, i.e., Convolutional Neural Networks (CNN) architecture for images, Recurrent Neural Networks (RNN) for sequence data.	Used for very large and complex datasets, e.g., natural language processing.
	Distributed Machine Learning Frameworks such as Spark MLlib, TensorFlow.	Used for scalable machine learning scenarios across distributed computing environments, e.g., large scale recommendation systems.

Key Choices and Guidelines

Decisions that impact the performance and utility of the application

Model Evaluation

- Model performance metrics are crucial for evaluating and comparing the effectiveness of machine learning models. Common metrics specific to classification tasks that are illustrated below are: precision, recall, F1 score, and area under the ROC curve and AUC.

Metric	Description	How to Interpret	SAP HANA ML Implementation (hana-ml / PAL)
Confusion Matrix	A table displaying: - **True Positives (TP)** - **True Negatives (TN)** - **False Positives (FP)** (Type I error) - **False Negatives (FN)** (Type II error)	Details correct and incorrect classifications, revealing error types. Minimizing False Negatives is often critical when missing an event (like an anomaly) has a high cost.	hana_ml.algorithms.pal.metrics.confusion_matrix
Precision, Recall, F1-Score	- **Precision:** TP / (TP + FP). Measures the accuracy of positive predictions. - **Recall:** TP / (TP + FN). Measures how many actual positives were identified. - **F1-Score:** 2 *(Precision* Recall) / (Precision + Recall). The harmonic mean of Precision and Recall.	High precision minimizes false alarms; high recall minimizes missed positives. The F1-score provides a balance, which is especially useful for imbalanced datasets.	Typically derived from the Confusion Matrix.
ROC AUC	**Area Under the Receiver Operating Characteristic Curve.** Plots True Positive Rate vs. False Positive Rate.	Measures a model's ability to distinguish between classes across all thresholds. A value of 1 is perfect; 0.5 is random. It can be misleading with highly imbalanced datasets.	hana_ml.algorithms.pal.metrics.auc

Key Choices and Guidelines

Decisions that impact the performance and utility of the application

Data Quality

- Ensure independent variables are accurate, complete, and free from errors or outliers that could distort the analysis. Data cleaning, outlier removal, and normalization are crucial steps in preparing your data.
- Ensure sufficient data points, specially considering the number of independent variables and the noise level of your data.
- Validate imputation strategies and model fit using techniques like cross-validation.

Key Choices and Guidelines

Decisions that impact the performance and utility of the application

Feature Selection

- Consider techniques like feature importance analysis or model-based selection to identify the most relevant predictors.
- The former provides insights into which features are most influential in a model's output, allowing for better understanding of the data and model performance; several methods exist, including permutation importance, SHapley Additive exPlanations (SHAP) values, and built-in importance scores from specific models.
- For instance, **SHAP values** offer a way to understand feature contributions by considering all possible combinations of features and calculating their marginal contributions.

Implementation

Programming Model Selection Guidelines

Data Science Workflows

Utilize the **Python** hana-ml library for a streamlined, intuitive experience aligned with standard data science practices, including convenient data manipulation and integration with machine learning workflows

Alternative Approaches

Use **SQLScript** to directly call PAL procedures when tight integration with SAP HANA artifacts is needed, or the R interface via the external **SAP HANA R client**.

Python

SDK

- Hana_ml

Reference Code

- SAP BTP AI Best Practices - Sample Code

Learning Journeys

- Build your Machine Learning Scenario for your SAP HANA Cloud application from Python
- Exploring ML Explainability in SAP HANA PAL – Classification and Regression

SQLScript

SDK

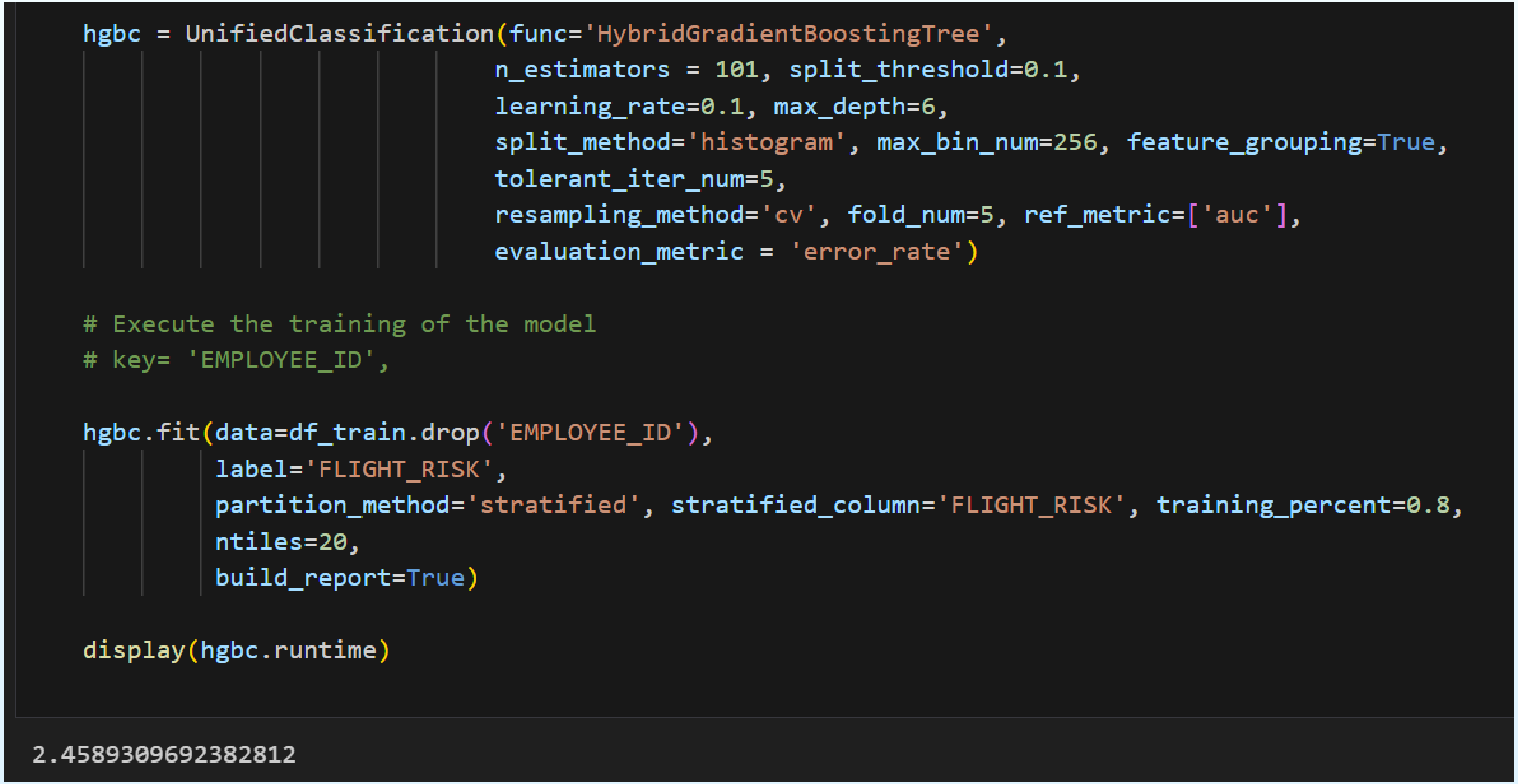
- SAP HANA Predictive Analysis Library (PAL)

Learning Journeys

- SAP HANA PAL quick start
- SAP HANA PAL Hybrid Gradient Boosting Tree Classifier

Code Sample

Python – Classification



```python
hgbc = UnifiedClassification(func='HybridGradientBoostingTree',
                            n_estimators = 101, split_threshold=0.1,
                            learning_rate=0.1, max_depth=6,
                            split_method='histogram', max_bin_num=256, feature_grouping=True,
                            tolerant_iter_num=5,
                            resampling_method='cv', fold_num=5, ref_metric=['auc'],
                            evaluation_metric = 'error_rate')

# Execute the training of the model
# key= 'EMPLOYEE_ID',

hgbc.fit(data=df_train.drop('EMPLOYEE_ID'),
         label='FLIGHT_RISK',
         partition_method='stratified', stratified_column='FLIGHT_RISK', training_percent=0.8,
         ntiles=20,
         build_report=True)

display(hgbc.runtime)
```

```
2.4589309692382812
```

Contributors

Pacheco-Sanchez, Sergio

Rzhaksynskyi, Andrii

Robledo, Francisco

Thank you