



TIC

Evaluación del modelo





TIC

Contenido

Esta actividad llamada Evaluación del Modelo se centra en proporcionar una comprensión profunda y completa de cómo evaluar el rendimiento de los modelos de inteligencia artificial. Exploraremos las métricas de evaluación clave, como precisión, recall, F1- score y curva ROC, para entender cómo el modelo se comporta en diferentes escenarios y cómo interpretar estos resultados para tomar decisiones informadas.



Además, aprenderemos sobre la matriz de confusión, una herramienta invaluable para comprender la calidad de las predicciones del modelo y cómo se comparan con las verdaderas etiquetas de clase. Estos subtemas nos equiparán con las habilidades necesarias para evaluar y mejorar continuamente nuestros modelos en una variedad de aplicaciones de aprendizaje automático.



Evaluación del Modelo: Métricas de Evaluación y Matriz de Confusión

La evaluación del modelo es un paso crítico en el desarrollo de sistemas de inteligencia artificial, que nos permite comprender cómo se desempeña nuestro modelo en datos no vistos y tomar decisiones informadas sobre su rendimiento. A continuación se detalla la evaluación del modelo a través de métricas clave y el uso de la matriz de confusión.



1. Métricas de Evaluación:

Todas las métricas ofrecidas por scikit-learn las puede encontrar en:
https://scikit-learn.org/stable/modules/model_evaluation.html

Scoring	Function	Comment
Classification		
'accuracy'	<code>metrics.accuracy_score</code>	
'balanced_accuracy'	<code>metrics.balanced_accuracy_score</code>	
'top_k_accuracy'	<code>metrics.top_k_accuracy_score</code>	
'average_precision'	<code>metrics.average_precision_score</code>	
'neg_brier_score'	<code>metrics.brier_score_loss</code>	
'f1'	<code>metrics.f1_score</code>	for binary targets
'f1_micro'	<code>metrics.f1_score</code>	micro-averaged
'f1_macro'	<code>metrics.f1_score</code>	macro-averaged
'f1_weighted'	<code>metrics.f1_score</code>	weighted average
'f1_samples'	<code>metrics.f1_score</code>	by multilabel sample
'neg_log_loss'	<code>metrics.log_loss</code>	requires <code>predict_proba</code> support
'precision' etc.	<code>metrics.precision_score</code>	suffixes apply as with 'f1'
'recall' etc.	<code>metrics.recall_score</code>	suffixes apply as with 'f1'
'jaccard' etc.	<code>metrics.jaccard_score</code>	suffixes apply as with 'f1'
'roc_auc'	<code>metrics.roc_auc_score</code>	
'roc_auc_ovr'	<code>metrics.roc_auc_score</code>	
'roc_auc_ovo'	<code>metrics.roc_auc_score</code>	
'roc_auc_ovr_weighted'	<code>metrics.roc_auc_score</code>	
'roc_auc_ovo_weighted'	<code>metrics.roc_auc_score</code>	

**Métrica de
clasificación**

Métrica de clustering

Scoring Clustering	Function
'adjusted_mutual_info_score'	<code>metrics.adjusted_mutual_info_score</code>
'adjusted_rand_score'	<code>metrics.adjusted_rand_score</code>
'completeness_score'	<code>metrics.completeness_score</code>
'fowlkes_mallows_score'	<code>metrics.fowlkes_mallows_score</code>
'homogeneity_score'	<code>metrics.homogeneity_score</code>
'mutual_info_score'	<code>metrics.mutual_info_score</code>
'normalized_mutual_info_score'	<code>metrics.normalized_mutual_info_score</code>
'rand_score'	<code>metrics.rand_score</code>
'v_measure_score'	<code>metrics.v_measure_score</code>

Métrica de regresión

Scoring	Function
Regression	
'explained_variance'	<code>metrics.explained_variance_score</code>
'max_error'	<code>metrics.max_error</code>
'neg_mean_absolute_error'	<code>metrics.mean_absolute_error</code>
'neg_mean_squared_error'	<code>metrics.mean_squared_error</code>
'neg_root_mean_squared_error'	<code>metrics.root_mean_squared_error</code>
'neg_mean_squared_log_error'	<code>metrics.mean_squared_log_error</code>
'neg_root_mean_squared_log_error'	<code>metrics.root_mean_squared_log_error</code>
'neg_median_absolute_error'	<code>metrics.median_absolute_error</code>
'r2'	<code>metrics.r2_score</code>
'neg_mean_poisson_deviance'	<code>metrics.mean_poisson_deviance</code>
'neg_mean_gamma_deviance'	<code>metrics.mean_gamma_deviance</code>
'neg_mean_absolute_percentage_error'	<code>metrics.mean_absolute_percentage_error</code>
'd2_absolute_error_score'	<code>metrics.d2_absolute_error_score</code>
'd2_pinball_score'	<code>metrics.d2_pinball_score</code>
'd2_tweedie_score'	<code>metrics.d2_tweedie_score</code>

A continuación vamos a presentar las más comunes pero las demas las pueden consultar en la documentación oficial.

Precisión:

Es la proporción de ejemplos positivos clasificados correctamente sobre el total de ejemplos clasificados como positivos. Se calcula como $TP / (TP + FP)$, donde TP son los verdaderos positivos y FP son los falsos positivos. Una alta precisión indica que el modelo tiene pocos falsos positivos.

Recall (Sensibilidad o Tasa de Verdaderos Positivos):

Es la proporción de ejemplos positivos clasificados correctamente sobre el total de ejemplos positivos en los datos reales. Se calcula como $TP / (TP + FN)$, donde FN son los falsos negativos. Un alto recall indica que el modelo tiene pocos falsos negativos.

F1-score:

Es la media armónica de precisión y recall. Proporciona un balance entre ambas métricas y es útil cuando hay un desequilibrio entre las clases. Se calcula como $2 * (\text{precisión} * \text{recall}) / (\text{precisión} + \text{recall})$.

Curva ROC (Receiver Operating Characteristic):

Es una representación gráfica de la relación entre la tasa de verdaderos positivos (TPR) y la tasa de falsos positivos (FPR) para varios umbrales de clasificación. Es útil para evaluar el rendimiento del modelo en diferentes puntos de corte y comparar modelos.



TIC

2. Matriz de Confusión:

La matriz de confusión es una tabla que describe el rendimiento del modelo en términos de verdaderos positivos (TP), verdaderos negativos (TN), falsos positivos (FP) y falsos negativos (FN). Es una herramienta poderosa para entender cómo el modelo clasifica las muestras en diferentes clases y para identificar cualquier sesgo o errores que pueda tener.

Verdaderos Positivos (TP):

Predicciones correctas de ejemplos positivos.

Verdaderos Negativos (TN):

Predicciones correctas de ejemplos negativos.

Falsos Positivos (FP):

Predicciones incorrectas de ejemplos negativos como positivos.

Falsos Negativos (FN): Predicciones incorrectas de ejemplos positivos como negativos.

Interpretar la matriz de confusión nos permite entender mejor cómo nuestro modelo está clasificando las muestras y nos proporciona información detallada sobre su rendimiento en diferentes clases.

`sklearn.metrics.confusion_matrix`

```
sklearn.metrics.confusion_matrix(y_true, y_pred, *, labels=None, sample_weight=None, normalize=None)
```

[\[source\]](#)

Compute confusion matrix to evaluate the accuracy of a classification.

By definition a confusion matrix C is such that $C_{i,j}$ is equal to the number of observations known to be in group i and predicted to be in group j .

Thus in binary classification, the count of true negatives is $C_{0,0}$, false negatives is $C_{1,0}$, true positives is $C_{1,1}$ and false positives is $C_{0,1}$.

Read more in the [User Guide](#).

La documentación oficial de scikit-learn respecto a la matriz de confusión la puede encontrar en el siguiente enlace:

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

Ejercicio 1: Cálculo de Métricas de Evaluación

```
from sklearn.datasets import load_digits

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score

# Cargar el conjunto de datos

digits = load_digits()

X, y = digits.data, digits.target
```

```
# Dividir el conjunto de datos en entrenamiento y prueba

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

# Entrenar el modelo

model = LogisticRegression(max_iter=1000)

model.fit(X_train, y_train)

# Realizar predicciones

y_pred = model.predict(X_test)
```

```
# Calcular métricas de evaluación
```

```
accuracy = accuracy_score(y_test, y_pred)
```

```
precision = precision_score(y_test, y_pred, average='weighted')
```

```
recall = recall_score(y_test, y_pred, average='weighted')
```

```
f1 = f1_score(y_test, y_pred, average='weighted')
```

```
# Mostrar las métricas
```

```
print("Precisión:", accuracy)
```

```
print("Recall:", recall)
```

```
print("F1-score:", f1)
```