



DATA SCIENCE BRAIN
@datasciencebrain

MODEL EVALUATION

CHEAT SHEET

**for Machine
Learning**

Save for later reference



01

CLASSIFICATION METRICS

Accuracy:

```
from sklearn.metrics import accuracy_score  
accuracy = accuracy_score(y_true, y_pred)
```

Precision:

```
from sklearn.metrics import precision_score  
precision = precision_score(y_true, y_pred)
```

Recall (Sensitivity):

```
from sklearn.metrics import recall_score  
recall = recall_score(y_true, y_pred)
```

F1 Score:

```
from sklearn.metrics import f1_score  
f1 = f1_score(y_true, y_pred)
```

Confusion Matrix:

```
from sklearn.metrics import confusion_matrix  
cm = confusion_matrix(y_true, y_pred)
```



02

ROC-AUC SCORE

```
from sklearn.metrics import roc_auc_score  
auc_score = roc_auc_score(y_true, y_prob)
```

03

REGRESSION METRICS

Mean Absolute Error (MAE):

```
from sklearn.metrics import mean_absolute_error  
mae = mean_absolute_error(y_true, y_pred)
```

Mean Squared Error (MSE):

```
from sklearn.metrics import mean_squared_error  
mse = mean_squared_error(y_true, y_pred)
```

R-squared (Coefficient of Determination):

```
from sklearn.metrics import r2_score  
r_squared = r2_score(y_true, y_pred)
```



04

CROSS-VALIDATION

Cross-Validation Score:

```
from sklearn.model_selection import  
cross_val_score
```

```
scores = cross_val_score(model, X, y, cv=5,  
scoring='accuracy')
```

Stratified K-Fold Cross-Validation:

```
from sklearn.model_selection import  
StratifiedKFold
```

```
cv = StratifiedKFold(n_splits=5, shuffle=True,  
random_state=42)  
scores = cross_val_score(model, X, y, cv=cv,  
scoring='accuracy')
```



05

HYPERPARAMETER TUNING

Grid Search:

```
from sklearn.model_selection import  
GridSearchCV
```

```
param_grid = {'param_name': [value1, value2, ...]}  
grid_search = GridSearchCV(model, param_grid,  
cv=5)
```

Randomized Search:

```
from sklearn.model_selection import  
RandomizedSearchCV
```

```
param_dist = {'param_name': [value1, value2, ...]}  
random_search = RandomizedSearchCV(model,  
param_distributions=param_dist, n_iter=10, cv=5)
```



05**HANDLING INCONSISTENT DATA TYPES**

Ensure Proper Data Types: Ensure that each column has the correct data type.

```
df = df.astype({'numeric_column': 'float64',  
'categorical_column': 'category'})
```

Use the `astype` method to explicitly set the data type of each column.

Convert Data Types: Convert data types, for example, from string to numeric.

```
df['numeric_column'] =  
pd.to_numeric(df['numeric_column'], errors='coerce')
```

Use `pd.to_numeric` to convert a column to numeric, handling errors as specified.

Convert Text to Lowercase or Uppercase:

Standardize text data by converting to lowercase or uppercase.

```
df['text_column'] = df['text_column'].str.lower()
```

Use `str.lower()` or `str.upper()` to standardize text data.



06

MODEL INTERPRETABILITY

Feature Importance (for tree-based models):

```
importances = model.feature_importances_
```

Permutation Importance:

```
from sklearn.inspection import  
permutation_importance  
result = permutation_importance(model, X_test,  
y_test, n_repeats=10, random_state=42)  
importance = result.importances_mean
```



Never Miss a Post!
Turn on the Notifications



Was it helpful?

Follow Us For More Amazing Data Science & Programming Related Posts



DATA SCIENCE BRAIN
@datasciencebrain



LIKE TO SUPPORT



COMMENT



SHARE



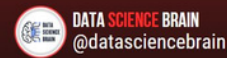
SAVE FOR LATER

Checkout Our Other Posts



07 Killer Data Science Project ideas

With Description



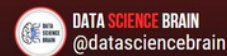
Actual Projects That Data Scientists Work

On In Companies



Data Science Concepts Explained

Overfitting & Underfitting



Data Science Interview

Questions & Answers

Save for later reference

