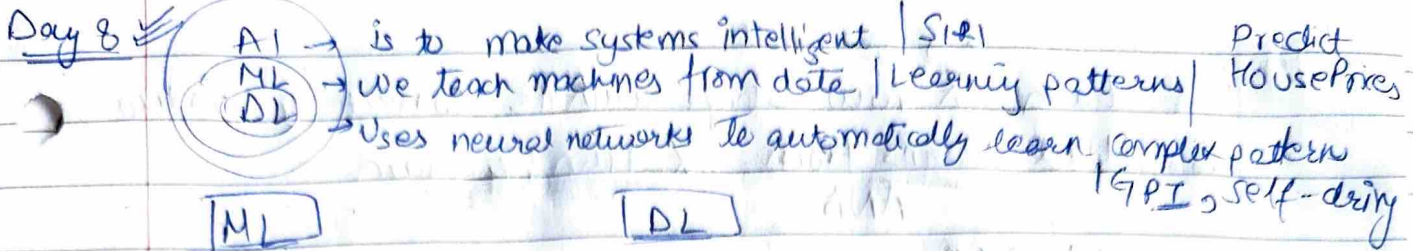


13 Oct

SCIKIT LEARN

Goal:



ML
↓
Scikit learn

DL
↓
PyTorch

- ↳ A python library for classical machine learning. It provides tools for data preprocessing, training, evaluation, ^{model} selection using algorithms like regression, classification etc.
- used for smaller datasets, structured data
 - Sklearn does not have neural networks like CNN, RNN, Transformed
 - It provides ML Algos (mathematically defined)
 - Regression → Linear
 - Classification → Logistic Regression, Random Forest, SVM, Decision Tree
 - Clustering → KMeans
 - Dimensionality Reduction → PCA
 - Preprocessing → Scaling

Convention "X" → Input / Features (multiple cols)
[[1200, 3, 8-2], [4, 3, 2], ...]
→ Matrix

"y" → Output / Target (vector = a single col)

Ex: $y = wX + b$ In Linear Regression

⇒ $X_{train}, X_{test}, y_{train}, y_{test} = \text{train_test_split}(X, y, \text{test_split} = 0.2)$

(CANNOT CHANGE ORDER)

• Randomly ^{selected} 80% rows

(put features & labels in X_{train} & y_{train})

• Remaining 20% rows, ~~Each~~
put their features → X_{test}
labels → y_{test}

- ∴ Each 'X' row stays with its correct 'y' value
- ∴ Model learns from X_train, y_train

AND

model's ~~model~~ accuracy is checked using X-test, y-test

② Review

↳ Ex3 Classification

- ① from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier.
 - ② x, y
~~data~~ = load_breast_cancer(~~data~~ return_X_y=True)
x → multidimensional numpy array
y → 0, 1
 - ③ $X_{train}, y_{train} = \text{train_test_split}(x, y, \text{test_size}=0.2)$
 - ④ $s = \text{StandardScaler}()$
 $X_{train_scaled} = s.\text{fit_transform}(X_{train})$
 $X_{test_scaled} = s.\text{transform}(X_{test})$

(fit_transform)
↳ calculates mean, std | learns parameters to scale data
↳ Applies scaling
- X-test ∴ we don't want to do (fit_transform)
as that (20% data) is for testing purposes, not training
we don't want to calculate its mean, std again
as it will be misleading good.
Test data is scaled on unseen data / as it is new
no sense in (training & test data)?

(5) ~~knn~~ knn = KNeighborsClassifier()
knn.fit(X_train_scaled, y_train)

(6) print(knn.score(X_test_scaled, y_test))
O/p $\rightarrow 0.95 = 95\%$ accuracy
correct classification

WORKFLOW

- (1) We have a dataset \rightarrow split, scale
- (2) We either want to do Regression, Classification, Clustering.
- (3) We then train on that data
- (4) We want to make predictions ^{with} our trained model.
(How well our model will perform on unseen data.)

\rightarrow load \rightarrow fetch \rightarrow make. $\} \text{from sklearn.datasets import}$