Jan 10th 2021

Ref: hands-on ML Chapter 3 classification.

Loading dataset in Sklearn:

- Sklearn provides helper function for downloading datasets
  eg. `>>> from sklearn.datasets import fetch_openml`
  `>>> mnist = fetch_openml('mnist_784', version=1)`
  contains multiple cleaned datasets.
- The dataset returned from the above operation contains many useful fields:
  Among them:
  - DESCR: Description of the dataset.
  - data: data entries (one row per instance, one col per feature).
  - target: array with labels.

Naive way to split data in to train and test.

`X_train, X_test, y_train, y_test = x[:6000], x[6000:], y[:6000], y[6000:]`

Training a Binary Classifier.

1. modify labels from catagories to true and false.
   e.g. `>>> y_train_5 = (y_train == 5)`.
   new label array: [T, F, ...]
   ↑ is_5, ↖ not_5

2. `sgd_clf = SGDClassifier(random_state = 42)`
   SGDClassifer relies on randomness during training.
   ※ same results will be reproduced if same random_state is set the same.

3. evaluation:
   - Accuracy: percentage of correct predictions.
     This is normally not a good measurement.
     for example: in the case of binary classification where positive cases are a lot less than negatives, even a naive model (such as a model that always predict false) will produce a high accuracy score.

   - Confusion matrix
     Sklearn: `>>> y_train_pred = cross_val_predict(sgd_clf, x_train, y_train_5, cv=3)`.
     return prediction results (labels).
     `>>> confusion_matrix(y_train_5, y_train_pred)`.

     matrix:

	Actual negative	Actual positive
Predicted negative	A (TN)	C (FN)
Predicted positive	B (FP)	D (TP)

   - Precision, Recall, F1.

     $\text{Precision} = \frac{TP}{TP+FP}$ How many correct predictions among predictions that label positive?

     $\text{Recall} = \frac{TP}{TP+FN}$ How many positives are detected.

     $F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = \frac{TP}{TP + \frac{FN+FP}{2}}$

F_1 : harmonic mean of precision and recall. Which gives much more weight to low values.

high F_1 if both recall and precision are high.

F_1 score favors classifiers that have similar precison and recall.

```
>>> from sklearn.metrics import f1_score
    f1_score(y_train_5, y_train_pred)
```