INFSCI 2915: Machine Learning

Classification Performance Evaluation– Confusion Matrix, Precision, Recall, ROC

Mai Abdelhakim

School of Computing and Information

610 IS Building

Spring 2018

Skewed Classes

- Skewed classes (dataset is imbalanced): when there is no sufficient training examples for one of the classes
  - For example in the default data set, 3% of the training examples actually defaulted and 97% did not.
    - In this case, a trivial classifier that always predicts that an individual will not default will have error rate of at most 3% (relatively good)
  - Another example: assume that email spam detection system has data set with only 1% of emails are spam.. Predicting all emails are not spam would lead to 99% accuracy (or 1% error rate)

Performance Measures

- Thus, error rate (discussed before) is not sufficient evaluation metric when classes are skewed
- Other metrics are more convenient: confusion matrix, precision, recall, ..
- These measures are also helpful to analyze the performance of classifiers even if classes are not skewed (have balanced dataset)

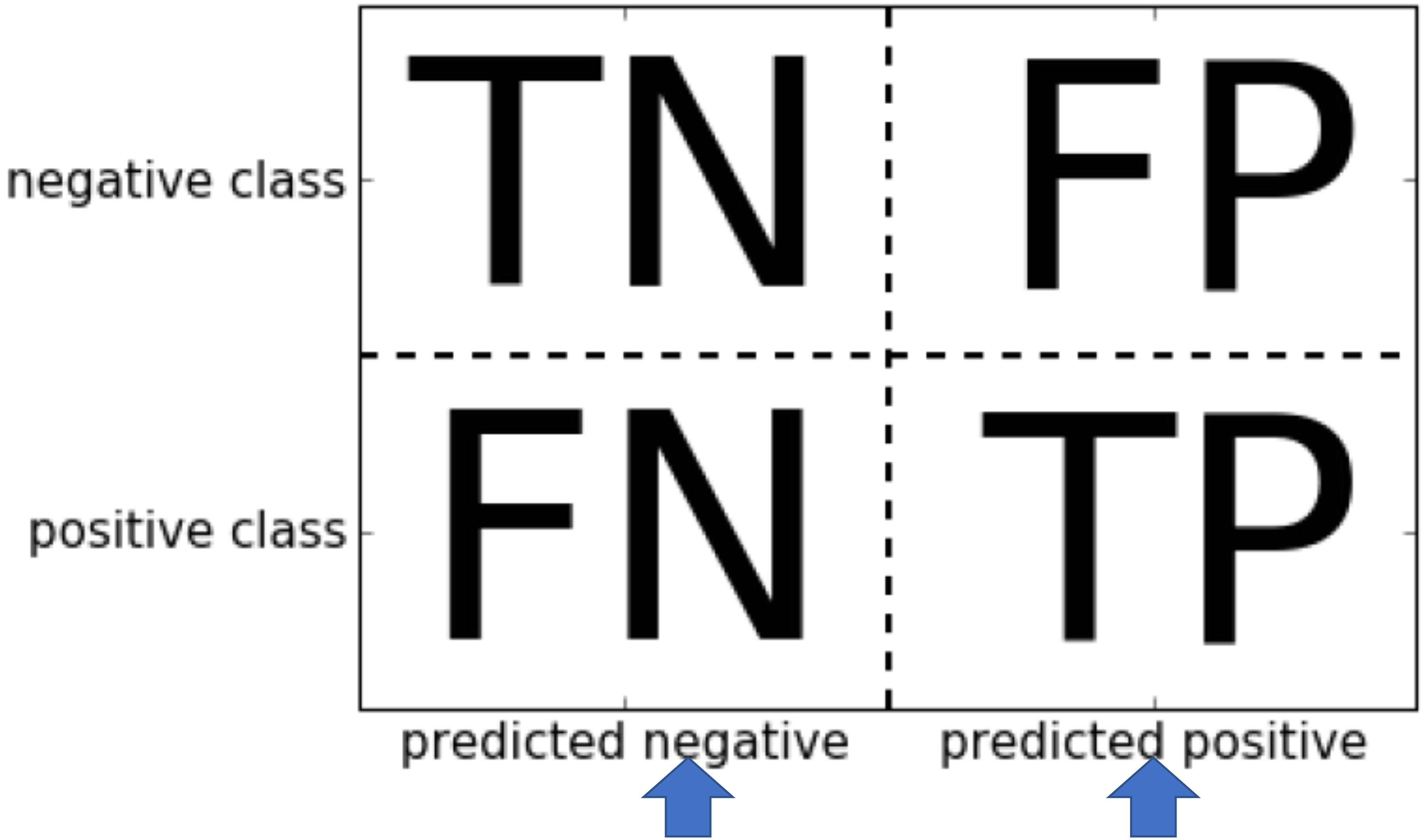
Confusion Matrix

- Assume two classes: negative class (Null) & positive class (Non-null)
  - In case of imbalanced dataset: Positive class is the rare class

		Predicted class	
		− or Null	+ or Non-null
True	− or Null	True Neg. (TN)	False Pos. (FP)
class	+ or Non-null	False Neg. (FN)	True Pos. (TP)

- False positive (FP): # samples in negative class misclassified as positive
- True positive (TP): # correctly classified samples belonging to positive class
- False negative (FN): # samples in positive class misclassified as negative
- True negative (TN): # correctly classified samples belonging to negative class

Confusion matrix



Required: large elements on diagonal & small values on off-diagonal.. Why?

Some of samples predicted negative are true/correct (TN) or false (FN):

Some of samples predicted as positive are true (FP) or false (FP)

Example: Imbalanced DataSet

- The dataset in sklearn (load_digits) contains digits from 0 – 9

- Suppose you want to build a classifier that classifies digit 9 (against the remaining digits 0 – 8)
  - Your prediction is either the digit is 9 or not
  - You created an imbalanced dataset:
    - Since number of times where 9 appears is much less than the number of times the other digits appear

  - A dummy classifier that selects majority (not 9) will have accuracy around 80%

Example .. cont

Confusion Matrix of Logistic Regression (C=0.1)

	predicted 'not nine'	predicted 'nine'
true 'not nine'	401	2
true 'nine'	8	39

Positive class is the rare class

Example: Apply LDA to Credit Card Default Data Set

- Objective: predict whether or not an individual will default (i.e., 2 classes: default, not default)
  - Two features (p=2): **income** and **balance** on the credit card
  - Dataset contains information of n=10,000 individuals
- Confusion matrix (here applied on training data for illustration, in real-world we do not use training for evaluation)

		Predicted default status		
		No	Yes	
True default status	No	**9644**	**23**	Total not defaulted=9667
	Yes	**252**	**81**	Total actual defaulted =333

LDA is applied here

Two-Class Classification:

- By understanding the performance better through the confusion matrix, we can modify the classifier to go better job based on the requirement.

- For example: with LDA we use Bayes rule and, we predict an individual will default if:

$$P(default = Yes|X = x) \ > \ P(default = No|X = x) \, .$$

This is equivalent to deciding that an individual will default if:

$$\Pr(\texttt{default} = \texttt{Yes}|X = x) > 0.5$$

Note that: $P(default = Yes|X = x) + \ P(default = No|X = x) = 1$

- We can modify the classifier by changing the 0.5 threshold!

Example

- If credit card company wants to avoid incorrectly classifying an individual who will default (& misclassification is less problematic)
  - Then can lower the threshold:

$$\Pr(\texttt{default} = \texttt{Yes}|X = x) > 0.2$$

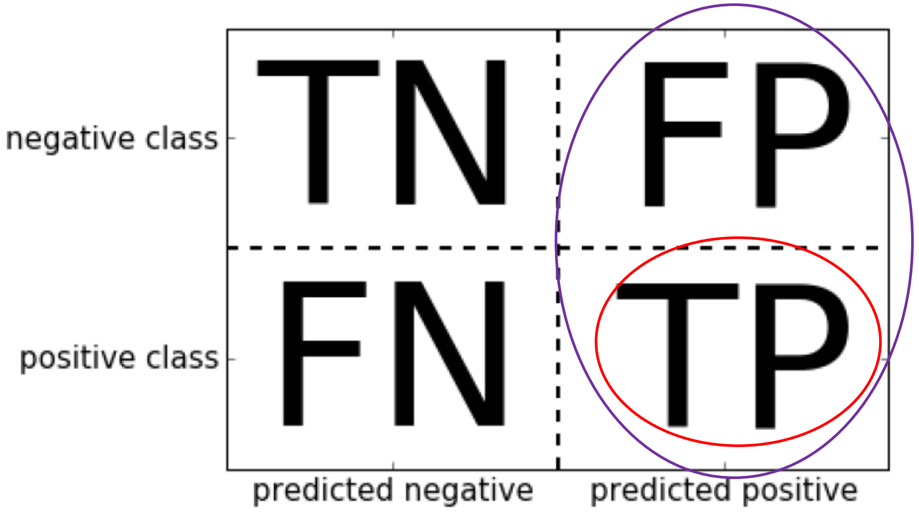
The resulted confusion matrix in this case is:

		Predicted default status	
		No	Yes
True default status	No	**9432**	**235**
	Yes	**138**	**195**

Precision and Recall

- **Precision**: Out of the all classes that we **predicted positive**, what fraction is actually positive

$$Precision = \frac{True\ positive}{All\ predicted\ positives}$$
$$= \frac{True\ positive\ (TP)}{True\ Positive(TP) + False\ Positive(FP)}$$

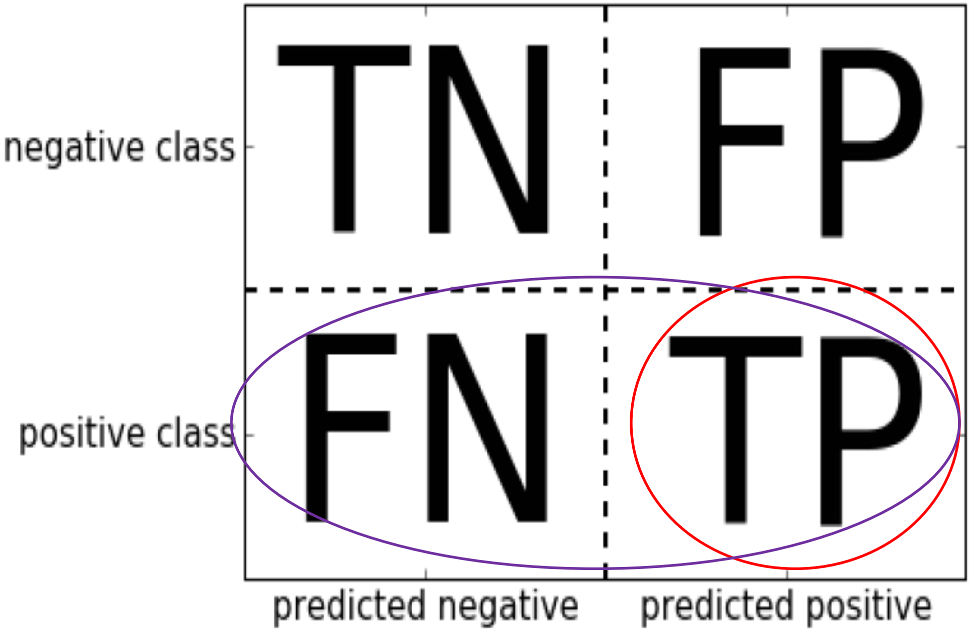


Positive class is the rare class

Precision and Recall

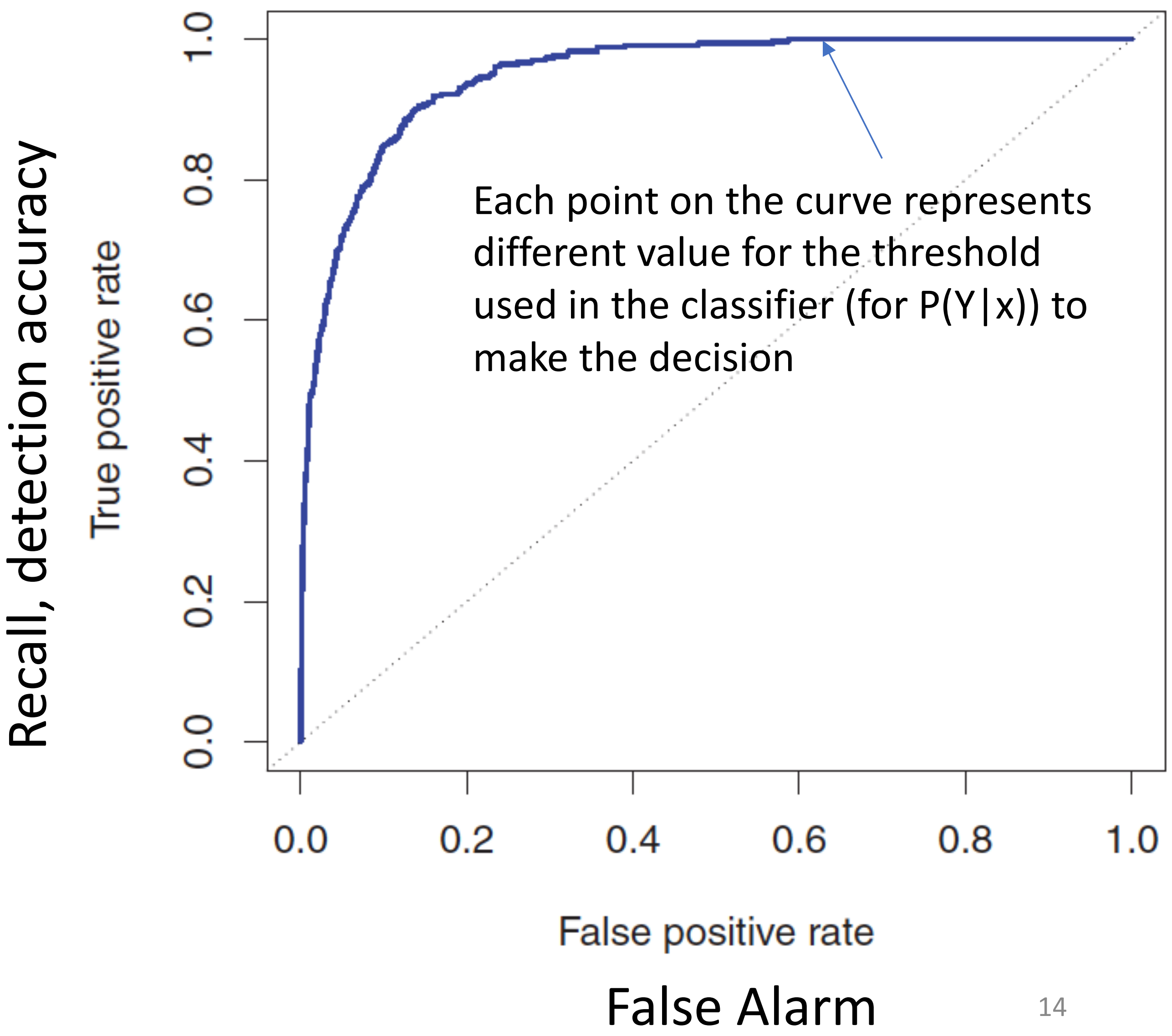
- **Recall (detection accuracy, true positive rate)**: Out of all the **actual positive** examples, what fraction we correctly detect as positive

$$Recall = \frac{True\ positive}{Actual\ positives} = \frac{True\ positive\ (TP)}{True\ Positive(TP) + False\ Negative\ (FN)}$$



Receiver Operating Characteristics (ROC) Curve

- Curve shows: **false positive rate** (FPR) versus **true positive rate** (recall)
  - **FPR = FP/(FP+TN)**
    - Number of times you **misclassified as negative** divided **by all the negative examples**
- ROC is also used to analyze the behavior of the classifier
  - Each point represent **different threshold** used for classification
- Required: high recall (true positive) and low false positive rate
  - But there is a trade-off between them



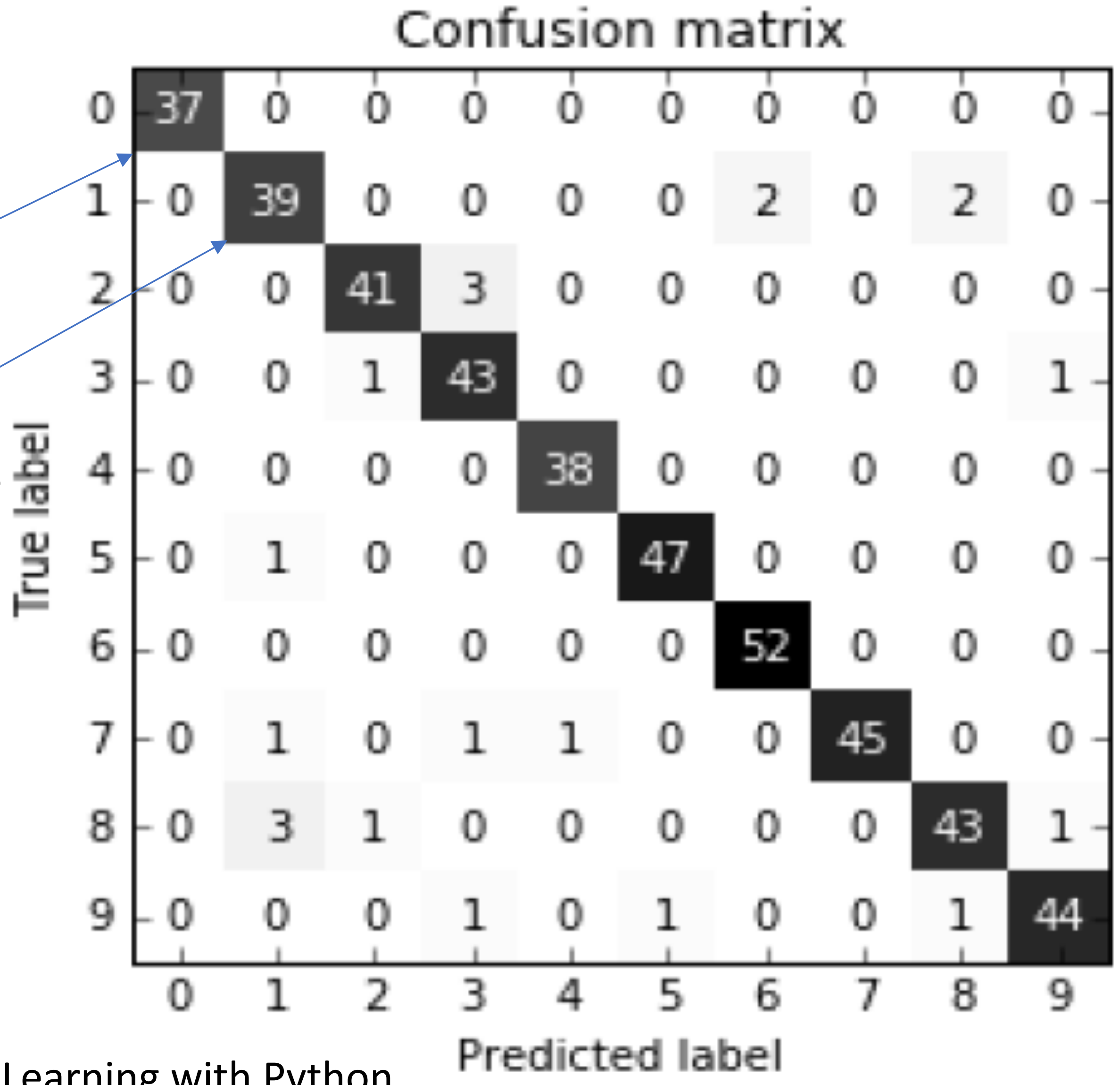
Multiclass Classification

- Similar to two-class classification, average error or accuracy will not be adequate if the data set is imbalanced datasets
  - Example: Assume 3 classes with a dataset that has: 90% of examples in class 1, 5% in class 2, and 5% in class 3
- We also use confusion matrix:
  - Each row represents a true label, and column elements represents the predicted label

Example: 10 handwritten digit classification

- Confusion matrix when applying logistic regression with default setting to the load_digit data set

All digit Zero are classified correctly,
For digit 1, 39 examples are classified correctly, 2 samples are misclassified as 6, and 2 were misclassified as 8



- Precession, Recall can be evaluated for each class in a similar manner as two-class classification
  - Classification report in python

Reference: Muller, Introduction to Machine Learning with Python

Evaluation Metrics in Python

```
from sklearn.metrics import confusion_matrix, precision_score, recall_score
PredictedOutput=Model.predict(X_test) # predicted output of classification
confusion=confusion_matrix(Y_test,PredictedOutput)
print(precision_score(Y_test,PredictedOutput))
print(recall_score(Y_test,PredictedOutput))
```

- Classification report for multiclass classification: http://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html