Week 7:
Classification

CMPS 320 : Machine Learning

Regression Versus Classification Problems

- Variables can be characterized as either quantitative or qualitative (also known as categorical).
  - Quantitative variables take on numerical values. Examples include a person's age, height, or income, the value of a house, and the price of a stock.
  - Qualitative variables take on values in one of K different classes, or categories. Examples of qualitative variables include a person's gender (male or female), whether a person defaults on a debt (yes or no), or a cancer diagnosis (Leukemia, or No Leukemia).
- Problems with a quantitative response are known as **regression problems**.
- Qualitative response are often referred to as **classification problems**

Training a Binary Classifier

- Binary classification refers to those classification tasks that have two class labels.
  - Email spam detection (spam or not).
  - Credit Card Application (good credit or bad credit)
- A good classifier is **Stochastic Gradient Descent (SGD) classifier**.
  - The SGD is an efficient approach to fitting linear classifiers and regressors under convex loss functions such as (linear) Support Vector Machines and Logistic Regression.
  - Loss function is used to measure the degree of fit.
- This classifier has the advantage of being capable of handling very large datasets efficiently.
- SGD is discussed in details in Chapter 4 of Geron.

Performance Measures

- We will discuss commonly used methods for testing the quality of a predictive model.

Measuring Accuracy Using Cross-Validation

- A good way to evaluate a model is to use **cross-validation**.
- **Cross-validation** is a technique for evaluating Machine Learning (ML) models by training several ML models on subsets of the available input data and evaluating them on the complementary subset of the data.
- The `cross_val_score()` function is used to evaluate our SGDClassifier model, using K-fold cross-validation.
  - K-fold cross-validation means splitting the training set into K folds, then making predictions and evaluating them on each fold using a model trained on the remaining folds.
- Accuracy (ratio of correct predictions) is generally not the preferred performance measure for classifiers, especially when you are dealing with unbalanced datasets (i.e., when some classes are much more frequent than others).

Confusion Matrix

- A much better way to evaluate the performance of a classifier is to look at the **confusion matrix**.
  - The idea is to count the number of times instances of class A are classified as class B.
- Each row in a confusion matrix represents an actual class, while each column represents a predicted class. Eg.

```
array([[53057,  1522],
       [ 1325,  4096]])
```

Confusion Matrix (Cont.)

```
array([[53057, 1522],
       [1325, 4096]])
```

- The first row considers non-5 images (the negative class):
  - 53,057 correctly classified as non-5s (called **true negatives** (TN))
  - 1,522 were wrongly classified as 5s (called **false positives** (FP)).
- The second row considers the images of 5s (the positive class):
  - 1,325 were wrongly classified as non-5s (**false negatives** (FN))
  - 4,096 were correctly classified as 5s (**true positives** (TP)).
- A perfect classifier would have only true positives and true negatives
  - Its confusion matrix would have nonzero values only on its main diagonal:

```
array([[54579, 0],
       [0, 5421]])
```

Precision

- The confusion matrix gives you a lot of information, but sometimes you may prefer a more concise metric.
- An interesting one to look at is the accuracy of the positive predictions; this is called the **precision** of the classifier.

$$\text{Precision} = \frac{TP}{TP + FP}$$

TP is the number of true positives, and FP is the number of false positives.

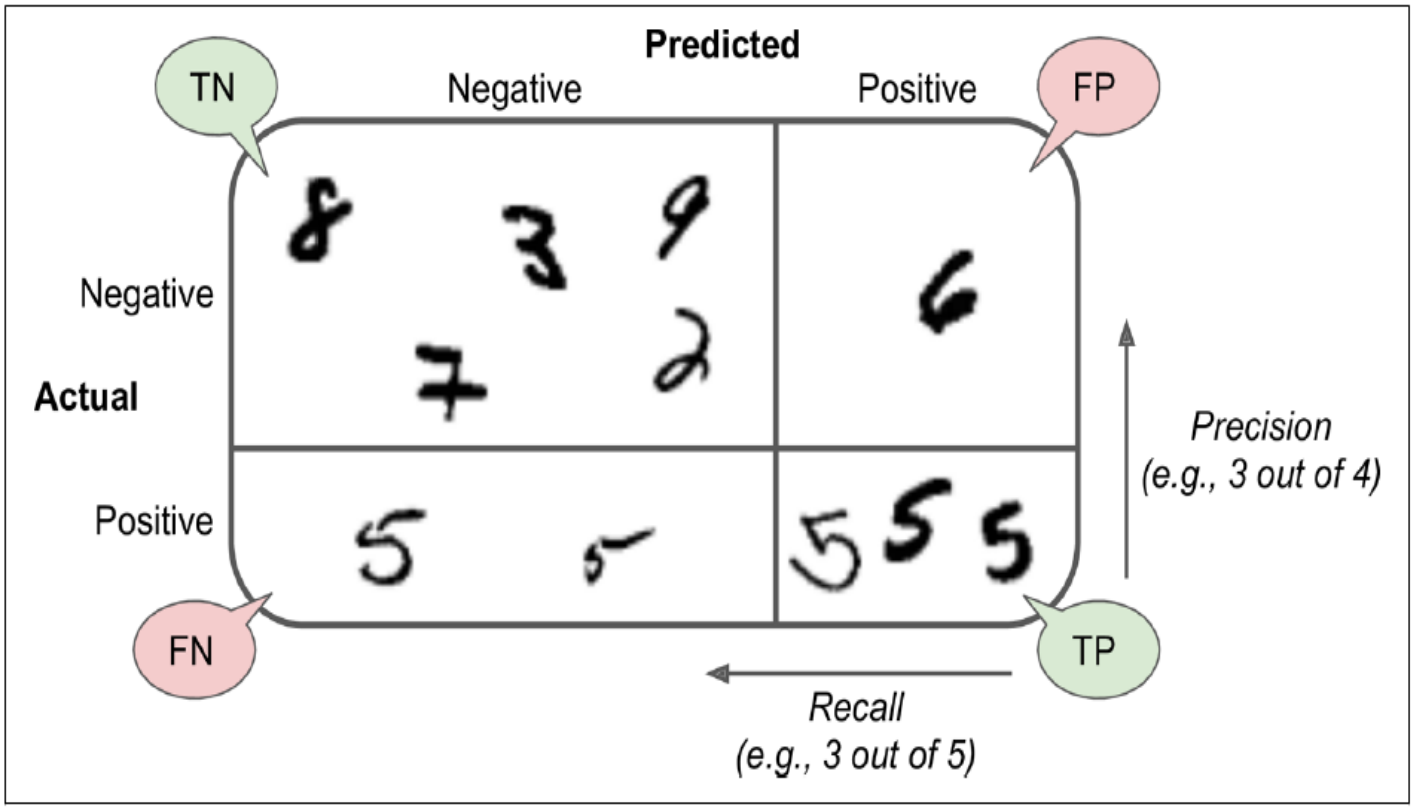
Recall

- Precision is typically used along with another metric named **recall**, also called **sensitivity** or the **true positive rate** (TPR):

$$\text{Recall} = \frac{TP}{TP + FN}$$

  FN is the number of false negatives.
- This is the ratio of positive instances that are correctly detected by the classifier

Summary of Confusion Matrix



- An illustrated confusion matrix shows examples of true negatives (top left), false positives (top right), false negatives (lower left), and true positives (lower right)

Precision and Recall – F_1 score

- It is often convenient to combine precision and recall into a single metric called the F_1 score.
- The F_1 score is the harmonic mean of precision and recall gives much more weight to low values.
- The classifier will only get a high F_1 score if both recall and precision are high.

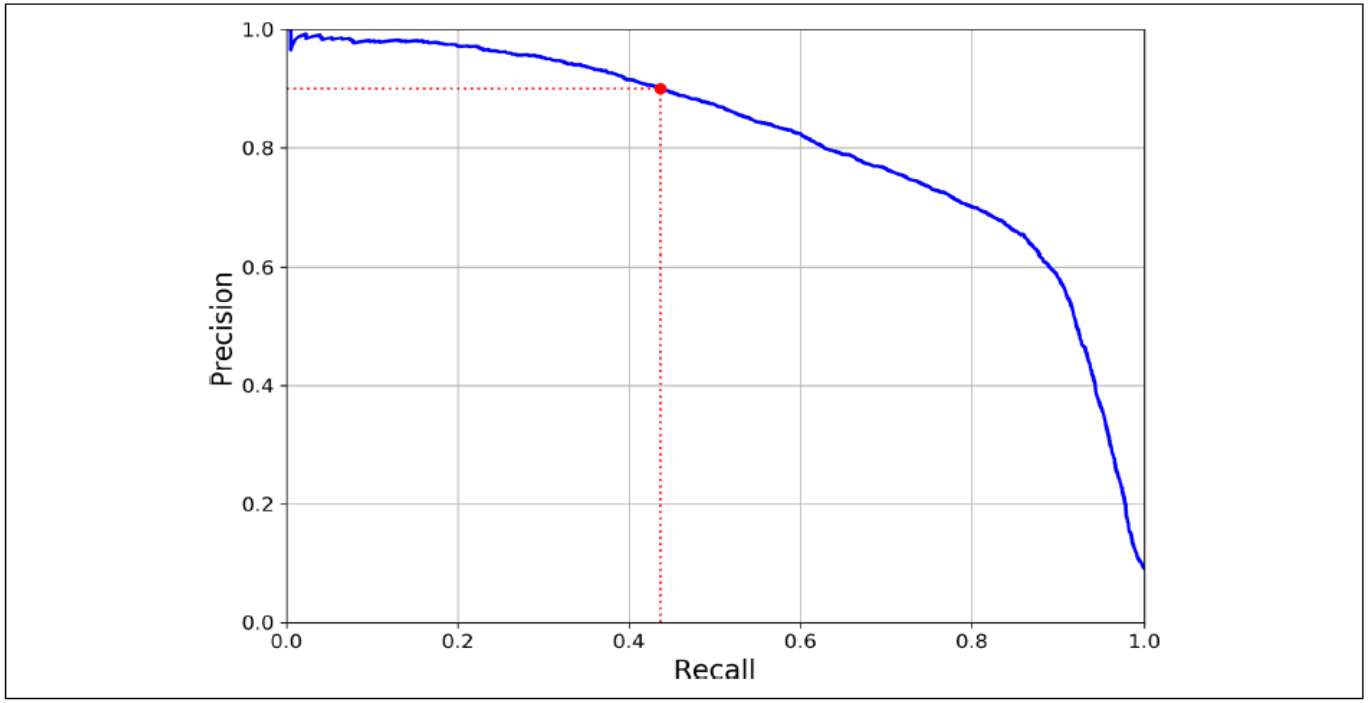
$$F_1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = \frac{TP}{TP + \frac{FN + FP}{2}}$$

F_1 score (cont.)

- In general the F_1 score favors classifiers that have similar precision and recall.
- This is not always what you want: in some contexts you mostly care about precision, and in other contexts you really care about recall.
- For example, if you trained a classifier to detect videos that are safe for kids, you would probably prefer a classifier that rejects many good videos (low recall) but keeps only safe ones (high precision), rather than a classifier that has a much higher recall but lets a few really bad videos show up in your product.
- On the other hand, suppose you train a classifier to detect shoplifters in surveillance images: it is probably fine if your classifier has only 30% precision as long as it has 99% recall.
- Unfortunately, you can't have it both ways: increasing precision reduces recall, and vice versa.
  - This is called the precision/recall trade-off.

Precision and Recall (Cont.)

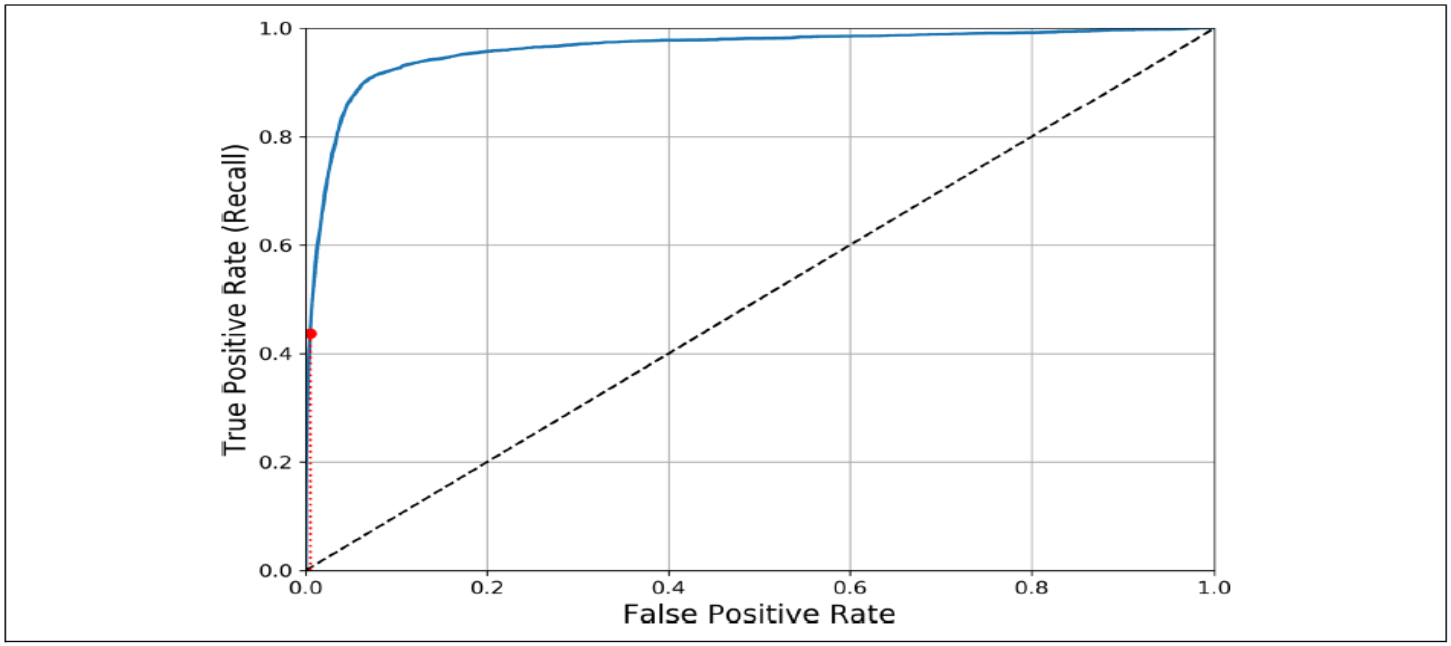
- The precision/recall curve plots the precision versus recall.



The ROC Curve

- The receiver operating characteristic (ROC) curve is another common tool used with binary classifiers.
- The ROC curve plots the true positive rate (another name for recall) against the false positive rate (FPR).
  - The FPR is the ratio of negative instances that are incorrectly classified as positive.
- It is equal to 1 minus the true negative rate (TNR), which is the ratio of negative instances that are correctly classified as negative.
  - The TNR is also called specificity.
- The ROC curve plots sensitivity (recall) versus 1 minus specificity.
- The higher the recall (TPR), the more false positives (FPR) the classifier produces.

The ROC Curve (Cont.)



- This ROC curve plots the false positive rate against the true positive rate for all possible thresholds; the red circle highlights the chosen ratio (at 43.68% recall)
- The dotted line represents the ROC curve of a purely random classifier; a good classifier stays as far away from that line as possible (toward the top-left corner).

Area Under the Curve (AUC)

- Another way to compare classifiers is to measure the area under the curve (AUC).
- A perfect classifier will have a ROC AUC equal to 1,
  - whereas a purely random classifier will have a ROC AUC equal to 0.5.
- **Remarks**: Since the ROC curve is so similar to the precision/recall (PR) curve, which one do we use?
- As a rule of thumb, you should prefer the PR curve whenever the positive class is rare (unbalanced classes) or when you care more about the false positives than the false negatives. Otherwise, use the ROC curve.

Multiclass Classification

- Whereas binary classifiers distinguish between two classes, **multiclass classifiers** (also called **multinomial classifiers**) can distinguish between more than two classes.
- Some algorithms (such as SGD classifiers, Random Forest classifiers, Bagging and Boosting classifiers) are capable of handling multiple classes natively.
- Others (such as Logistic Regression or Support Vector Machine classifiers) are strictly binary classifiers.

Multiclass Classification (cont.)

- One way to create a system that can classify the digit images into 10 classes (from 0 to 9) is to train 10 binary classifiers, one for each digit (a 0-detector, a 1-detector, a 2- detector, and so on).
  - Then when you want to classify an image, you get the decision score from each classifier for that image and you select the class whose classifier outputs the highest score.
  - This is called the one-versus-the-rest (OvR) strategy (also called one-versus-all).
- Another strategy is to train a binary classifier for every pair of digits: one to distinguish 0s and 1s, another to distinguish 0s and 2s, another for 1s and 2s, and so on.
  - This is called the one-versus-one (OvO) strategy. If there are N classes, you need to train $N \times (N-1)/2$ classifiers.
  - For the MNIST problem, this means training 45 binary classifiers.

Multiclass Classification (cont.)

- Some algorithms (such as Support Vector Machine classifiers) scale poorly with the size of the training set.
  - For these algorithms OvO is preferred because it is faster to train many classifiers on small training sets than to train few classifiers on large training sets.
  - For most binary classification algorithms, however, OvR is preferred.

Multilabel Classification

- **Multilabel classification** is a classification system that outputs multiple binary tags.
- Consider a face recognition classifier: what should it do if it recognizes several people in the same picture?
- It should attach one tag per person it recognizes.
  - Say the classifier has been trained to recognize three faces, Alice, Bob, and Charlie.
  - Then when the classifier is shown a picture of Alice and Charlie, it should output [1, 0, 1] (meaning "Alice yes, Bob no, Charlie yes").

multi-output–multiclass (Multi-output) Classification

- **Multi-output classification** is a generalization of multilabel classification where each label can be multiclass (i.e., it can have more than two possible values).