W4995 Applied Machine Learning

Model evaluation

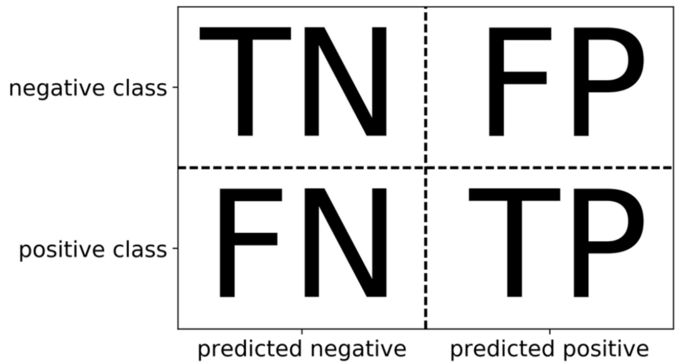
02/25/19

Andreas C. Müller

(Adapted and modified for CC 6021236 @ PCC/Ciencias/UCV by Eugenio Scalise, October 2019)

Metrics for Binary Classification

Review : confusion matrix



$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

```python
from sklearn.datasets import load_breast_cancer
from sklearn.linear_model import LogisticRegression
data = load_breast_cancer()

X_train, X_test, y_train, y_test = train_test_split(
    data.data, data.target, stratify=data.target, random_state=0)

lr = LogisticRegression().fit(X_train, y_train)
y_pred = lr.predict(X_test)

from sklearn.metrics import confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(lr.score(X_test, y_test))
```

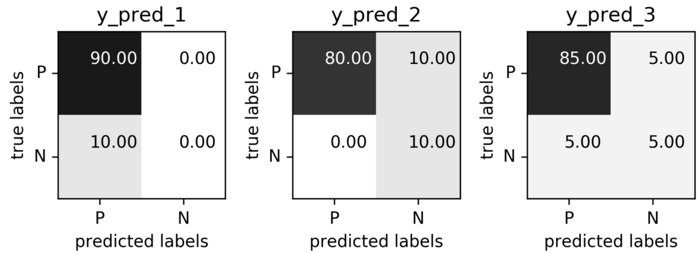
```
[[49  4]
 [ 5 85]]
0.937062937063
```

Problems with Accuracy

Data with 90% positives:

```python
from sklearn.metrics import accuracy_score
for y_pred in [y_pred_1, y_pred_2, y_pred_3]:
    print(accuracy_score(y_true, y_pred))
```

```
0.9
0.9
0.9
```



Precision, Recall, f-score

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

Positive Predicted Value (PPV), [Precisión]

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

Sensitivity, coverage, true positive rate.
[Exhaustividad]

$$\text{F} = 2\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Harmonic mean of precision and recall (F1-score)

Precision, Recall, FPR

- Precision: what fraction of positive predictions are correct?

- Recall (True Positive Rate, sensitivity, probability of detection): what fraction of all positive instances does the classifier correctly identify as positive?

- False Positive Rate (Specificity): what fraction of all negative instances does the classifier incorrectly identify as positive.

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{TN} + \mathrm{FP}}$$

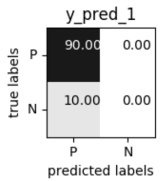
Tradeoff between precision and recall

- Recall-oriented machine learning tasks:
  - Search and information extraction in legal discovery
  - Tumor detection
  - Often paired with a human expert to filter out false positives
- Precision-oriented machine learning tasks:
  - Sarch engine ranking, query suggestion
  - Document classification
  - Many customer-facing tasks (users remember failures!)

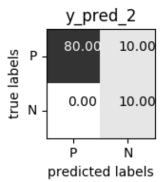
The Zoo

		True condition				
	Total population	Condition positive	Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**, Power	**False positive**, Type I error	Positive predictive value (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$	F_1 score $= \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	True negative rate (TNR), Specificity (SPC) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$		

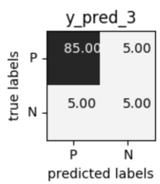
https://en.wikipedia.org/wiki/Precision_and_recall



	precision	recall	f1-score	support
0	0.90	1.00	0.95	90
1	0.00	0.00	0.00	10
avg / total	0.81	0.90	0.85	100



	precision	recall	f1-score	support
0	1.00	0.89	0.94	90
1	0.50	1.00	0.67	10
avg / total	0.95	0.90	0.91	100



	precision	recall	f1-score	support
0	0.94	0.94	0.94	90
1	0.50	0.50	0.50	10
avg / total	0.90	0.90	0.90	100

Goal setting!

- What do I want? What do I care about?
- Can I assign costs to the confusion matrix? (i.e. a false positive costs me 10 dollars; a false negative, 100 dollars)
- What guarantees do we want to give?

Changing Thresholds

```
data = load_breast_cancer()

X_train, X_test, y_train, y_test = train_test_split(
    data.data, data.target, stratify=data.target, random_state=0)

lr = LogisticRegression().fit(X_train, y_train)
y_pred = lr.predict(X_test)

print(classification_report(y_test, y_pred))
```

```
          precision   recall  f1-score  support
0              0.91     0.92      0.92       53
1              0.96     0.94      0.95       90
avg/total      0.94     0.94      0.94      143
```

```
y_pred = lr.predict_proba(X_test)[:, 1] > .85

print(classification_report(y_test, y_pred))
```

```
          precision   recall  f1-score  support
0              0.84     1.00      0.91       53
1              1.00     0.89      0.94       90
avg/total      0.94     0.93      0.93      143
```

Precision-Recall curve

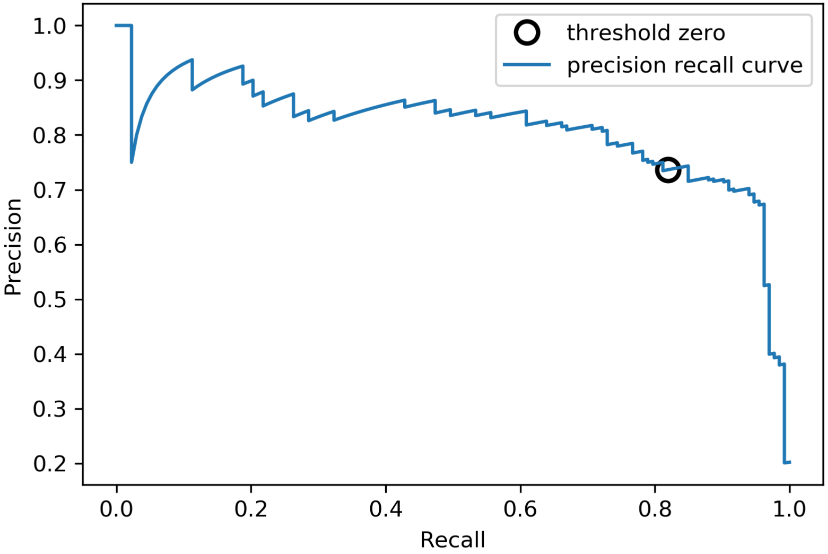
```
X, y = make_blobs(n_samples=(2500, 500), cluster_std=[7.0, 2],
                  random_state=22)

X_train, X_test, y_train, y_test = train_test_split(X, y)

svc = SVC(gamma=.05).fit(X_train, y_train)

precision, recall, thresholds = precision_recall_curve(
    y_test, svc.decision_function(X_test))
```

Precision-Recall curve



F1 vs average Precision

```python
from sklearn.metrics import f1_score
print("f1_score of random forest: {:.3f}".format(
      f1_score(y_test, rf.predict(X_test))))
print("f1_score of svc: {:.3f}".format(f1_score(y_test, svc.predict(X_test))))
```

```
f1_score of random forest: 0.709
f1_score of svc: 0.715
```

```python
from sklearn.metrics import average_precision_score
ap_rf = average_precision_score(y_test, rf.predict_proba(X_test)[:, 1])
ap_svc = average_precision_score(y_test, svc.decision_function(X_test))
print("Average precision of random forest: {:.3f}".format(ap_rf))
print("Average precision of svc: {:.3f}".format(ap_svc))
```

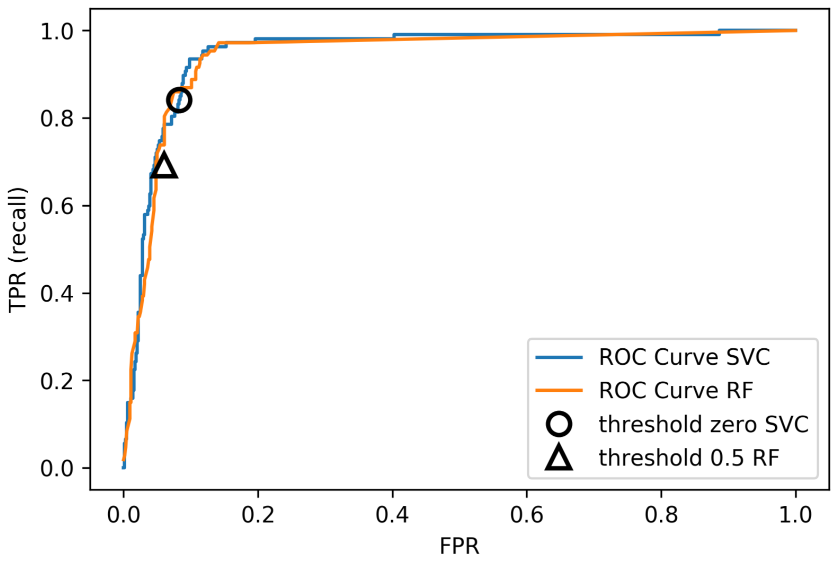
```
Average precision of random forest: 0.682
Average precision of svc: 0.693
```

ROC Curve

		True condition				
	Total population	Condition positive	Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**, Power	**False positive**, Type I error	Positive predictive value (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$	F_1 score $= \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	True negative rate (TNR), Specificity (SPC) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$		

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \text{recall}$$


ROC Curve SVC
ROC Curve RF
threshold zero SVC
threshold 0.5 RF
TPR (recall)
FPR
0.0
0.2
0.4
0.6
0.8
1.0

ROC AUC

- Area under ROC Curve
- Always .5 for random/constant prediction

```
from sklearn.metrics import roc_auc_score
rf_auc = roc_auc_score(y_test, rf.predict_proba(X_test)[:,1])
svc_auc = roc_auc_score(y_test, svc.decision_function(X_test))
print("AUC for random forest: {:, .3f}".format(rf_auc))
print("AUC for SVC: {:, .3f}".format(svc_auc))
```

```
AUC for random forest: 0.937
AUC for SVC: 0.916
```

The Relationship Between Precision-Recall and ROC Curves
https://www.biostat.wisc.edu/~page/rocpr.pdf

Summary of metrics for binary classification

Threshold-based:

- accuracy
- precision, recall, f1

Ranking:

- average precision
- ROC AUC

Multi-class classification

Confusion Matrix

```
from sklearn.datasets import load_digits
from sklearn.metrics import accuracy_score

digits = load_digits()
 # data is between 0 and 16
X_train, X_test, y_train, y_test = train_test_split(
    digits.data / 16., digits.target, random_state=0)
lr = LogisticRegression().fit(X_train, y_train)
pred = lr.predict(X_test)
print("Accuracy: {:.3f}".format(accuracy_score(y_test, pred)))
print("Confusion matrix:")
print(confusion_matrix(y_test, pred))
```

```
Accuracy: 0.964
Confusion matrix:
[[37  0  0  0  0  0  0  0  0  0]
 [ 0 41  0  0  0  0  1  0  1  0]
 [ 0  0 44  0  0  0  0  0  0  0]
 [ 0  0  0 43  0  0  0  0  1  1]
 [ 0  0  0  0 37  0  0  1  0  0]
 [ 0  0  0  0  0 47  0  0  0  1]
 [ 0  1  0  0  0  0 51  0  0  0]
 [ 0  0  0  0  1  0  0 47  0  0]
 [ 0  3  1  0  0  1  0  0 43  0]
 [ 0  0  0  0  0  2  0  0  1 44]]
```

```
print(classification_report(y_test, pred))
```

```
             precision    recall  f1-score   support

          0       1.00      1.00      1.00        37
          1       0.91      0.95      0.93        43
          2       0.98      1.00      0.99        44
          3       1.00      0.96      0.98        45
          4       0.97      0.97      0.97        38
          5       0.94      0.98      0.96        48
          6       0.98      0.98      0.98        52
          7       0.98      0.98      0.98        48
          8       0.93      0.90      0.91        48
          9       0.96      0.94      0.95        47

avg / total       0.96      0.96      0.96       450
```

Note: read about "micro" and "macro" average.

Using metrics

Using metrics in cross-validation

```python
X, y = make_blobs(n_samples=(2500, 500), cluster_std=[7.0, 2],
                  random_state=22)

# default scoring for classification is accuracy
scores_default = cross_val_score(SVC(), X, y)

# providing scoring="accuracy" doesn't change the results
explicit_accuracy =  cross_val_score(SVC(), X, y, scoring="accuracy")

# using ROC AUC
roc_auc =  cross_val_score(SVC(), X, y, scoring="roc_auc")

print("Default scoring:", scores_default)
print("Explicit accuracy scoring:", explicit_accuracy)
print("AUC scoring:", roc_auc)
```

```
Default scoring: [ 0.92   0.904  0.913]
Explicit accuracy scoring: [ 0.92   0.904  0.913]
AUC scoring: [ 0.93   0.885  0.923]
```

Built-in scoring

```python
from sklearn.metrics.scorer import SCORERS
print("\n".join(sorted(SCORERS.keys())))
```

```
print("\n".join(sorted(SCORERS.keys())))
accuracy                       adjusted_mutual_info_score     adjusted_rand_score
average_precision              balanced_accuracy              brier_score_loss
completeness_score             explained_variance             f1
f1_macro                       f1_micro                       f1_samples
f1_weighted                    fowlkes_mallows_score          homogeneity_score
max_error                      mutual_info_score              neg_log_loss
neg_mean_absolute_error        neg_mean_squared_error         neg_mean_squared_log_error
neg_median_absolute_error      normalized_mutual_info_score   precision
precision_macro                precision_micro                precision_samples
precision_weighted             r2                             recall
recall_macro                   recall_micro                   recall_samples
recall_weighted                roc_auc                        v_measure_score
```

Questions ?