

TON DUC THANG UNIVERSITY

Faculty of Information Technology

Computer Science

Evaluation of Machine Learning Models

LÊ ANH CƯỜNG

Content

- Evaluation for classification
- Evaluation for regression

Content

- Evaluation for classification
- Evaluation for regression

Classifier Evaluation Metrics: Confusion Matrix

Confusion Matrix:

Actual class\Predicted class	C_1	$\neg C_1$
C_1	True Positives (TP)	False Negatives (FN)
$\neg C_1$	False Positives (FP)	True Negatives (TN)

- Given m classes, an entry, $CM_{i,j}$ in a **confusion matrix** indicates # of tuples in class i that were labeled by the classifier as class j
- May have extra rows/columns to provide totals

Classifier Evaluation Metrics: Confusion Matrix

Example of Confusion Matrix:

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	6954	46	7000
buy_computer = no	412	2588	3000
Total	7366	2634	10000

- Given m classes, an entry, $CM_{i,j}$ in a **confusion matrix** indicates # of tuples in class i that were labeled by the classifier as class j
- May have extra rows/columns to provide totals

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

A\P	C	¬C	
C	TP	FN	P
¬C	FP	TN	N
	P'	N'	All

- **Classifier Accuracy**, or recognition rate: percentage of test set tuples that are correctly classified

$$\text{Accuracy} = (\text{TP} + \text{TN}) / \text{All}$$

- **Error rate**: $1 - \text{accuracy}$, or

$$\text{Error rate} = (\text{FP} + \text{FN}) / \text{All}$$

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

A\P	C	¬C	
C	TP	FN	P
¬C	FP	TN	N
	P'	N'	All

- **Class Imbalance Problem:**
 - One class may be *rare*, e.g. fraud, or HIV-positive
 - Significant *majority of the negative class* and minority of the positive class
 - **Sensitivity:** True Positive recognition rate
 - **Sensitivity = TP/P**
 - **Specificity:** True Negative recognition rate
 - **Specificity = TN/N**

Classifier Evaluation Metrics: Precision and Recall, and F-measures

- **Precision:** exactness – what % of tuples that the classifier labeled as positive are actually positive

$$precision = \frac{TP}{TP + FP}$$

- **Recall:** completeness – what % of positive tuples did the classifier label as positive?
- Perfect score is 1.0

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

Classifier Evaluation Metrics: Precision and Recall, and F-measures

- Inverse relationship between precision & recall
- **F measure** (F_1 or **F-score**): harmonic mean of precision and recall,

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

- F_β : weighted measure of precision and recall
 - assigns β times as much weight to recall as to precision

$$F_\beta = \frac{(1 + \beta^2) \times \text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}}$$

Classifier Evaluation Metrics: Example

Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	90	210	300	<i>Sensitivity = ?</i>
cancer = no	140	9560	9700	<i>Specificity = ?</i>
Total	230	9770	10000	<i>Accuracy = ?</i>

- *Precision = ?*
- *Recall = ?*
- *F-score = ?*

Classifier Evaluation Metrics: Example

Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	90	210	300	30.00 (<i>sensitivity</i>)
cancer = no	140	9560	9700	98.56 (<i>specificity</i>)
Total	230	9770	10000	96.40 (<i>accuracy</i>)

- $Precision = 90/230 = 39.13\%$
- $Recall = 90/300 = 30.00\%$

ROC curve



An **ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate
- False Positive Rate

True Positive Rate (TPR) is a synonym for recall and is therefore defined as follows:

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

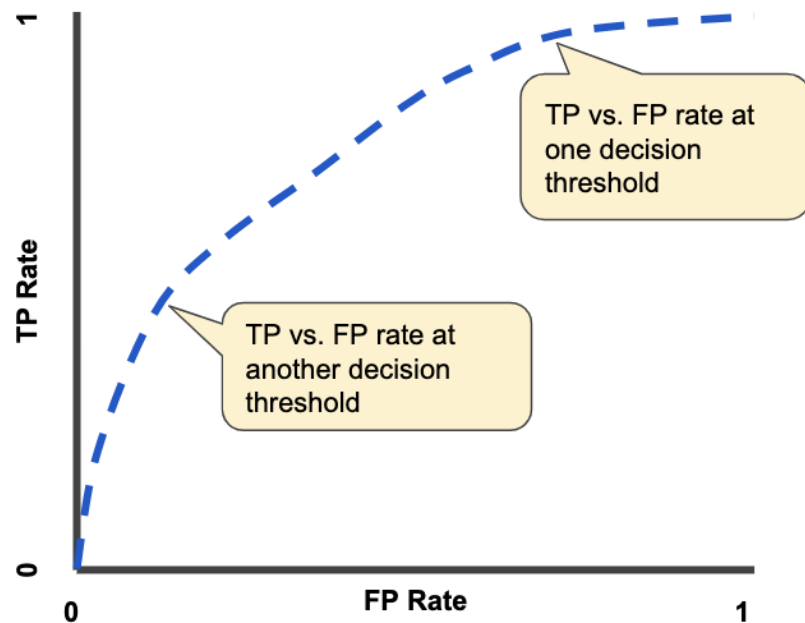
False Positive Rate (FPR) is defined as follows:

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

<https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc>

ROC curve

An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives. The following figure shows a typical ROC curve.



To compute the points in an ROC curve, we could evaluate a logistic regression model many times with different classification thresholds, but this would be inefficient. Fortunately, there's an efficient, sorting-based algorithm that can provide this information for us, called AUC.

Figure 4. TP vs. FP rate at different classification thresholds.

Example:

<https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f>

```
1  #Classification Area under curve
2  import warnings
3  import pandas
4  from sklearn import model_selection
5  from sklearn.linear_model import LogisticRegression
6  from sklearn.metrics import roc_auc_score, roc_curve
7
8  warnings.filterwarnings('ignore')
9
10 url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes
11 dataframe = pandas.read_csv(url)
12 dat = dataframe.values
13 X = dat[:, :-1]
14 y = dat[:, -1]
15 seed = 7
16 #split data
17 X_train, X_test, y_train, y_test = model_selection.train_test_split(X, y, test_size=test
18 model.fit(X_train, y_train)
19
```

Example

```
# predict probabilities
probs = model.predict_proba(X_test)
# keep probabilities for the positive outcome only
probs = probs[:, 1]

auc = roc_auc_score(y_test, probs)
print('AUC - Test Set: %.2f%%' % (auc*100))

# calculate roc curve
fpr, tpr, thresholds = roc_curve(y_test, probs)
# plot no skill
plt.plot([0, 1], [0, 1], linestyle='--')
# plot the roc curve for the model
plt.plot(fpr, tpr, marker='.')
# show the plot
plt.show()
```

Evaluating Classifier Accuracy: Holdout & Cross-Validation Methods

- **Holdout method**
 - Given data is randomly partitioned into two independent sets
 - Training set (e.g., 2/3) for model construction
 - Test set (e.g., 1/3) for accuracy estimation
 - Random sampling: a variation of holdout
 - Repeat holdout k times, accuracy = avg. of the accuracies obtained
- **Cross-validation** (k -fold, where $k = 10$ is most popular)
 - Randomly partition the data into k *mutually exclusive* subsets, each approximately equal size
 - At i -th iteration, use D_i as test set and others as training set
 - Leave-one-out: k folds where $k = \#$ of tuples, for small sized data
 - ***Stratified cross-validation***: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data

Evaluating Classifier Accuracy: Bootstrap

- **Bootstrap**
 - Works well with small data sets
 - Samples the given training tuples uniformly *with replacement*
 - i.e., each time a tuple is selected, it is equally likely to be selected again and re-added to the training set
- Several bootstrap methods, and a common one is **.632 bootstrap**
 - A data set with d tuples is sampled d times, with replacement, resulting in a training set of d samples. The data tuples that did not make it into the training set end up forming the test set. About 63.2% of the original data end up in the bootstrap, and the remaining 36.8% form the test set (since $(1 - 1/d)^d \approx e^{-1} = 0.368$)
 - Repeat the sampling procedure k times, overall accuracy of the model:

$$Acc(M) = \frac{1}{k} \sum_{i=1}^k (0.632 \times Acc(M_i)_{test_set} + 0.368 \times Acc(M_i)_{train_set})$$

Estimating Confidence Intervals: Classifier Models M_1 vs. M_2

- Suppose we have 2 classifiers, M_1 and M_2 , which one is better?
- Use 10-fold cross-validation to obtain $\overline{err}(M_1)$ and $\overline{err}(M_2)$
- These mean error rates are just *estimates* of error on the true population of *future* data cases
- What if the difference between the 2 error rates is just attributed to *chance*?
 - Use a **test of statistical significance**
 - Obtain **confidence limits** for our error estimates

Estimating Confidence Intervals: Null Hypothesis

- Perform 10-fold cross-validation
- Assume samples follow a **t distribution** with $k-1$ **degrees of freedom** (here, $k=10$)
- Use **t-test** (or **Student's t-test**)
- **Null Hypothesis:** M_1 & M_2 are the same
- If we can **reject** null hypothesis, then
 - we conclude that the difference between M_1 & M_2 is **statistically significant**
 - Chose model with lower error rate

Estimating Confidence Intervals: t-test

- If only 1 test set available: **pairwise comparison**
 - For i^{th} round of 10-fold cross-validation, the same cross partitioning is used to obtain $\overline{err}(M_1)$ and $\overline{err}(M_2)$
 - Average over 10 rounds to get $t = \frac{\overline{err}(M_1) - \overline{err}(M_2)}{\sqrt{\text{var}(M_1 - M_2)/k}}$
 - **t-test** computes **t-statistic** with **$k-1$ degrees of freedom:**

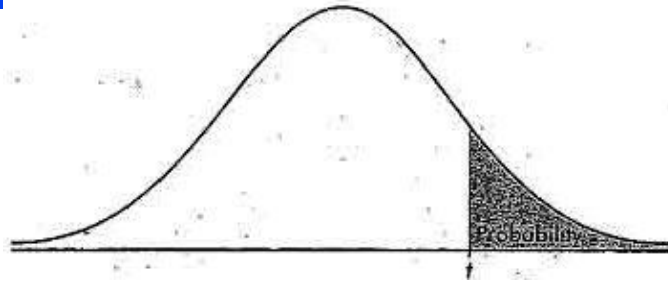
where
$$\text{var}(M_1 - M_2) = \frac{1}{k} \sum_{i=1}^k \left[\text{err}(M_1)_i - \text{err}(M_2)_i - (\overline{err}(M_1) - \overline{err}(M_2)) \right]^2$$

- If two test sets available: use **non-paired t-test**

where
$$\text{var}(M_1 - M_2) = \sqrt{\frac{\text{var}(M_1)}{k_1} + \frac{\text{var}(M_2)}{k_2}},$$

where k_1 & k_2 are # of cross-validation samples used for M_1 & M_2 , resp.

Estimating Confidence Intervals: Table for t-distribution



- Symmetric
- **Significance level**,
e.g., $sig = 0.05$ or
5% means M_1 & M_2
are *significantly*
different for 95% of
population
- **Confidence limit**, z
 $= sig/2$

TABLE B: t-DISTRIBUTION CRITICAL VALUES

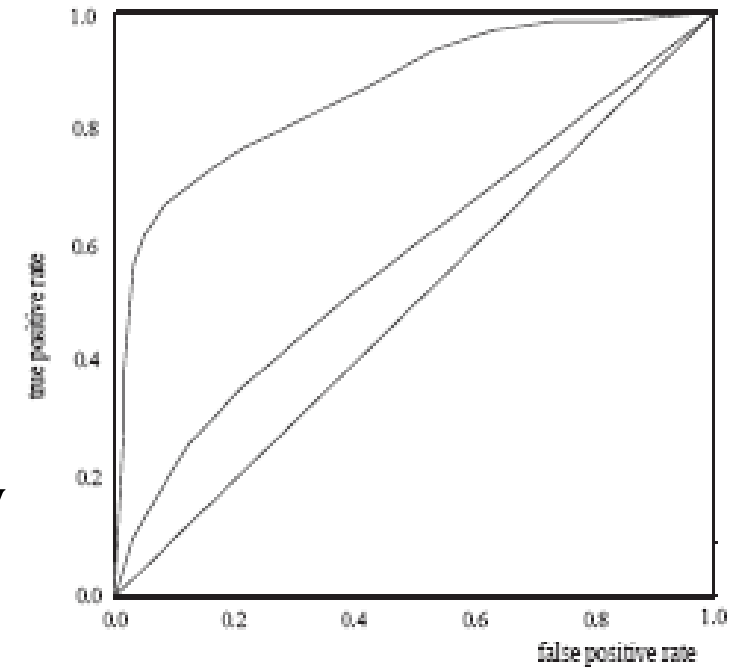
df	Tail probability p											
	.25	.20	.15	.10	.05	.025	.02	.01	.005	.0025	.001	.0005
1	1.000	1.376	1.963	3.078	6.314	12.71	15.89	31.82	63.66	127.3	318.3	636.6
2	.816	1.061	1.386	1.886	2.920	4.303	4.849	6.965	9.925	14.09	22.33	31.60
3	.765	.978	1.250	1.638	2.353	3.182	3.482	4.541	5.841	7.453	10.21	12.92
4	.741	.941	1.190	1.533	2.132	2.776	2.999	3.747	4.604	5.598	7.173	8.610
5	.727	.920	1.156	1.476	2.015	2.571	2.757	3.365	4.032	4.773	5.893	6.869
6	.718	.906	1.134	1.440	1.943	2.447	2.612	3.143	3.707	4.317	5.208	5.959
7	.711	.896	1.119	1.415	1.895	2.365	2.517	2.998	3.499	4.029	4.785	5.408
8	.706	.889	1.108	1.397	1.860	2.306	2.449	2.896	3.355	3.833	4.501	5.041
9	.703	.883	1.100	1.383	1.833	2.262	2.398	2.821	3.250	3.690	4.297	4.781
10	.700	.879	1.093	1.372	1.812	2.228	2.359	2.764	3.169	3.581	4.144	4.587
11	.697	.876	1.088	1.363	1.796	2.201	2.328	2.718	3.106	3.497	4.025	4.437
12	.695	.873	1.083	1.356	1.782	2.179	2.303	2.681	3.055	3.428	3.930	4.318
13	.694	.870	1.079	1.350	1.771	2.160	2.282	2.650	3.012	3.372	3.852	4.221
14	.692	.868	1.076	1.345	1.761	2.145	2.264	2.624	2.977	3.326	3.787	4.140
15	.691	.866	1.074	1.341	1.753	2.131	2.249	2.602	2.947	3.286	3.733	4.073
16	.690	.865	1.071	1.337	1.746	2.120	2.235	2.583	2.921	3.252	3.686	4.015
17	.689	.863	1.069	1.333	1.740	2.110	2.224	2.567	2.898	3.222	3.646	3.965
18	.688	.862	1.067	1.330	1.734	2.101	2.214	2.552	2.878	3.197	3.611	3.922
19	.688	.861	1.066	1.328	1.729	2.093	2.205	2.539	2.861	3.174	3.579	3.883
20	.687	.860	1.064	1.325	1.725	2.086	2.197	2.528	2.845	3.153	3.552	3.850
21	.686	.859	1.063	1.323	1.721	2.080	2.189	2.518	2.831	3.135	3.527	3.819
22	.686	.858	1.061	1.321	1.717	2.074	2.183	2.508	2.819	3.119	3.505	3.792
23	.685	.858	1.060	1.319	1.714	2.069	2.177	2.500	2.807	3.104	3.485	3.768
24	.685	.857	1.059	1.318	1.711	2.064	2.172	2.492	2.797	3.091	3.467	3.745
25	.684	.856	1.058	1.316	1.708	2.060	2.167	2.485	2.787	3.078	3.450	3.725
26	.684	.856	1.058	1.315	1.706	2.056	2.162	2.479	2.779	3.067	3.435	3.707
27	.684	.855	1.057	1.314	1.703	2.052	2.158	2.473	2.771	3.057	3.421	3.690
28	.683	.855	1.056	1.313	1.701	2.048	2.154	2.467	2.763	3.047	3.408	3.674
29	.683	.854	1.055	1.311	1.699	2.045	2.150	2.462	2.756	3.038	3.396	3.659
30	.683	.854	1.055	1.310	1.697	2.042	2.147	2.457	2.750	3.030	3.385	3.646
40	.681	.851	1.050	1.303	1.684	2.021	2.123	2.423	2.704	2.971	3.307	3.551
50	.679	.849	1.047	1.299	1.676	2.009	2.109	2.403	2.678	2.937	3.261	3.496
60	.679	.848	1.045	1.296	1.671	2.000	2.099	2.390	2.660	2.915	3.232	3.460
80	.678	.846	1.043	1.292	1.664	1.990	2.088	2.374	2.639	2.887	3.195	3.416
100	.677	.845	1.042	1.290	1.660	1.984	2.081	2.364	2.626	2.871	3.174	3.390
1000	.675	.842	1.037	1.282	1.646	1.962	2.056	2.330	2.581	2.813	3.098	3.300
∞	.674	.841	1.036	1.282	1.645	1.960	2.054	2.326	2.576	2.807	3.091	3.291
	50%	60%	70%	80%	90%	95%	96%	98%	99%	99.5%	99.8%	99.9%
	Confidence level C											

Estimating Confidence Intervals: Statistical Significance

- Are M_1 & M_2 **significantly different**?
 - Compute t . Select *significance level* (e.g. $sig = 5\%$)
 - Consult table for t-distribution: Find t value corresponding to $k-1$ *degrees of freedom* (here, 9)
 - t-distribution is symmetric: typically upper % points of distribution shown \rightarrow look up value for **confidence limit** $z=sig/2$ (here, 0.025)
 - **If $t > z$ or $t < -z$** , then t value lies in rejection region:
 - **Reject null hypothesis** that mean error rates of M_1 & M_2 are same
 - Conclude: statistically significant difference between M_1 & M_2
 - **Otherwise**, conclude that any difference is **chance**

Model Selection: ROC Curves

- **ROC** (Receiver Operating Characteristics) curves: for visual comparison of classification models
- Originated from signal detection theory
- Shows the trade-off between the true positive rate and the false positive rate
- The area under the ROC curve is a measure of the accuracy of the model
- Rank the test tuples in decreasing order: the one that is most likely to belong to the positive class appears at the top of the list
- The closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model



- Vertical axis represents the true positive rate
- Horizontal axis rep. the false positive rate
- The plot also shows a diagonal line
- A model with perfect accuracy will have an area of 1.0

Issues Affecting Model Selection

- **Accuracy**
 - classifier accuracy: predicting class label
- **Speed**
 - time to construct the model (training time)
 - time to use the model (classification/prediction time)
- **Robustness**: handling noise and missing values
- **Scalability**: efficiency in disk-resident databases
- **Interpretability**
 - understanding and insight provided by the model
- Other measures, e.g., goodness of rules, such as decision tree size or compactness of classification rules

Example

```
1 import warnings
2 import pandas
3 from sklearn import model_selection
4 from sklearn.linear_model import LogisticRegression
5 from sklearn.metrics import log_loss
6 from sklearn.metrics import precision_recall_fscore_support as score, precision_score, r
7
8 warnings.filterwarnings('ignore')
9
10 url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes
11 dataframe = pandas.read_csv(url)
12 dat = dataframe.values
13 X = dat[:, :-1]
14 y = dat[:, -1]
15 test_size = 0.33
16 seed = 7
17
18 model = LogisticRegression()
19 #split data
20 X_train, X_test, y_train, y_test = model_selection.train_test_split(X, y, test_size=test
21 model.fit(X_train, y_train)
22 precision = precision_score(y_test, pred)
23 print('Precision: %f' % precision)
24 # recall: tp / (tp + fn)
25 recall = recall_score(y_test, pred)
26 print('Recall: %f' % recall)
27 # f1: tp / (tp + fp + fn)
28 f1 = f1_score(y_test, pred)
29 print('F1 score: %f' % f1)
```



```
import warnings
import pandas
from sklearn import model_selection
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import log_loss
from sklearn.metrics import precision_recall_fscore_support as score, precision_score, recall_score, f1_score

warnings.filterwarnings('ignore')

url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"
dataframe = pandas.read_csv(url)
dat = dataframe.values
X = dat[:, :-1]
y = dat[:, -1]
test_size = 0.33
seed = 7

model = LogisticRegression()
#split data
X_train, X_test, y_train, y_test = model_selection.train_test_split(X, y, test_size=test_size, random_state=seed)
model.fit(X_train, y_train)
precision = precision_score(y_test, pred)
print('Precision: %f' % precision)
# recall: tp / (tp + fn)
recall = recall_score(y_test, pred)
print('Recall: %f' % recall)
# f1: tp / (tp + fp + fn)
f1 = f1_score(y_test, pred)
print('F1 score: %f' % f1)
```

Content

- Evaluation for classification
- Evaluation for regression

Metrics

Mean squared error

$$MSE = \frac{RSS}{n} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

- on average how far are our predictions from the true values (in squared distance)?
- Interpretation downside: the units are squared units
- Square root of MSE (**RMSE = root mean squared error**) is often used:

$$RMSE = \sqrt{MSE}$$

Metrics

Mean squared error

$$MSE = \frac{RSS}{n} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

RMSE = root mean squared error

$$RMSE = \sqrt{MSE}$$

Mean absolute error

$$MAE = \frac{1}{n} \sum_{i=1}^n \|y_i - \hat{y}_i\|$$

Where $\|y_i - \hat{y}_i\|$ indicates the absolute value of the residual

- Very interpretable: on average how far are our predictions from the true values

Metrics

Mean squared error

$$MSE = \frac{RSS}{n} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

RMSE = root mean squared error

$$RMSE = \sqrt{MSE}$$

Mean absolute error

$$MAE = \frac{1}{n} \sum_{i=1}^n \|y_i - \hat{y}_i\|$$

R-squared

$$TSS = \sum_{i=1}^n (y_i - \bar{y})^2$$

- Define the *total sum of squares* (TSS) as the sum of squared deviations of each response y_i from the mean response \bar{y} :

```
1 import pandas
2 from sklearn import model_selection
3 from sklearn.linear_model import LinearRegression
4 from sklearn.metrics import mean_absolute_error, mean_squared_error
5 from math import sqrt
6 url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/housing.data"
7 dataframe = pandas.read_csv(url, delim_whitespace=True)
8 df = dataframe.values
9 X = df[:, :-1]
10 y = df[:, -1]
11 seed = 7
12 model = LinearRegression()
13 #split data
14 X_train, X_test, y_train, y_test = model_selection.train_test_split(X, y, test_size=test
15 model.fit(X_train, y_train)
16 #predict
17 pred = model.predict(X_test)
18 print("MAE test score:", mean_absolute_error(y_test, pred))
19 print("RMSE test score:", sqrt(mean_squared_error(y_test, pred)))
```