Avisek Gupta (avisek003@gmail.com)

Machine Learning

16 – Classification Validation

November 14, 2022

Validating Classification Models: Accuracy

Let $X = \{\mathbf{x}_1, ..., \mathbf{x}_n\}, \mathbf{x}_i \in \mathbb{R}^d$ be a set of data with associated classification labels y_i, $i = 1, ..., n$. Let the class labels predicted by a classifier be $\hat{y}_i$, $i = 1, ..., n$.

Accuracy: The accuracy (ACC) of a classifier is the fraction of its correct predictions:

$$ACC = \frac{1}{n} \sum_{i=1}^{n} I(y_i = \hat{y}_i)$$

Validating Classification Models: Accuracy

Let $X = \{\mathbf{x}_1, ..., \mathbf{x}_n\}, \mathbf{x}_i \in \mathbb{R}^d$ be a set of data with associated classification labels y_i, $i = 1, ..., n$. Let the class labels predicted by a classifier be $\hat{y}_i$, $i = 1, ..., n$.

Accuracy: The accuracy (ACC) of a classifier is the fraction of its correct predictions:

$$ACC = \frac{1}{n}\sum_{i=1}^{n} I(y_i = \hat{y}_i)$$

The *training*, *validation*, or *test* accuracy are similarly defined over the respective data sets:

$$ACC^{train} = \frac{1}{n}\sum_{i=1}^{n} I(y_i^{train} = \hat{y}_i^{train})$$

$$ACC^{val} = \frac{1}{n}\sum_{i=1}^{n} I(y_i^{val} = \hat{y}_i^{val}), \quad ACC^{test} = \frac{1}{n}\sum_{i=1}^{n} I(y_i^{test} = \hat{y}_i^{test})$$

Validating Classification Models: Error Rates

Let $X = \{\mathbf{x}_1, ..., \mathbf{x}_n\}, \mathbf{x}_i \in \mathbb{R}^d$ be a set of data with associated classification labels y_i, $i = 1, ..., n$. Let the class labels predicted by a classifier be $\hat{y}_i$, $i = 1, ..., n$.

Error Rate: The error rate of a classifier is the fraction of incorrect predictions:

$$\varepsilon = \frac{1}{n} \sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

The *training*, *validation*, or *test* error rates are similarly defined over the respective data sets:

$$\varepsilon^{train} = \frac{1}{n} \sum_{i=1}^{n} I(y_i^{train} \neq \hat{y}_i^{train})$$

$$\varepsilon^{val} = \frac{1}{n} \sum_{i=1}^{n} I(y_i^{val} \neq \hat{y}_i^{val}), \quad \varepsilon^{test} = \frac{1}{n} \sum_{i=1}^{n} I(y_i^{test} \neq \hat{y}_i^{test})$$

Validating Classification Models: Confusion Matrix

For a k-class classification problem with class labels $\{c_1, ..., c_k\}$, we can partition a data set based on their true labels as $D = \{D_1, ..., D_k\}$, where,

$$D_j = \{x_i \in D | y_i = c_j\}$$

We can also paritition the data set based on the labels predicted by a classifier $R = \{R_1, ..., R_k\}$, where,

$$R_j = \{x_i \in D | \hat{y}_i = c_j\}$$

The partitions D and R induce a $k \times k$ *contingency matrix*, called a **confusion matrix** N, defined as,

$$N(i, j) = |D_i \cap R_j| = |\{x_a \in D | y_a = c_i \text{ and } \hat{y}_a = c_j\}|$$

Validating Classification Models: Confusion Matrix

Confusion Matrix:

$$N(i, j) = |D_i \cap R_j| = |\{x_a \in D | y_a = c_i \text{ and } \hat{y}_a = c_j\}|$$

where, $D_j = \{x_i \in D | y_i = c_j\}$, $R_j = \{x_i \in D | \hat{y}_i = c_j\}$.

Example: On the Iris data set (no. of classes = 3), training a kNN classifier (with no. of neighbors = 5) yields the following confusion matrix.

	R_1	R_2	R_3
D_1	50	0	0
D_2	0	47	3
D_3	0	2	48

Validating Classification Models: Precision

Confusion Matrix:

$$N(i,j) = |D_i \cap R_j| = |\{x_a \in D | y_a = c_i \text{ and } \hat{y}_a = c_j\}|$$

where, $D_j = \{x_i \in D | y_i = c_j\}$, $R_j = \{x_i \in D | \hat{y}_i = c_j\}$.

Precision: The class-specific accuracy, or precision, is defined as the fraction of correct predictions overall all points predicted to be in class c_j,

$$\text{Precision}_j = \frac{n_{jj}}{\sum_{i=1}^{k} n_{ij}} = \frac{n_{jj}}{|R_j|}$$

Example: Confusion Matrix of a k-NN classifier on the Iris data set:

	R_1	R_2	R_3
D_1	50	0	0
D_2	0	47	3
D_3	0	2	48

Validating Classification Models: Precision

Confusion Matrix:

$$N(i, j) = |D_i \cap R_j| = |\{x_a \in D | y_a = c_i \text{ and } \hat{y}_a = c_j\}|$$

where, $D_j = \{x_i \in D | y_i = c_j\}$, $R_j = \{x_i \in D | \hat{y}_i = c_j\}$.

Precision: The class-specific accuracy, or precision:

$$\text{Precision}_j = \frac{n_{jj}}{\sum_{i=1}^{k} n_{ij}} = \frac{n_{jj}}{|R_j|}$$

The overall precision is the weighted average of the class-specific precision,

$$\sum_{j=1}^{k} \frac{|R_j|}{n}.\text{Precision}_i = \frac{1}{n} \sum_{j=1}^{k} n_{jj}$$

Validating Classification Models: Recall

Confusion Matrix:

$$N(i,j) = |D_i \cap R_j| = |\{x_a \in D | y_a = c_i \text{ and } \hat{y}_a = c_j\}|$$

where, $D_j = \{x_i \in D | y_i = c_j\}$, $R_j = \{x_i \in D | \hat{y}_i = c_j\}$.

Recall: The class-specific recall is defined as the fraction of correct predictions over all points that are actually in class c_i,

$$\text{Recall}_i = \frac{n_{ii}}{\sum_{j=1}^{k} n_{ij}} = \frac{n_{ii}}{|D_i|}$$

Example: Confusion Matrix of a k-NN classifier on the Iris data set:

	R_1	R_2	R_3
D_1	50	0	0
D_2	0	47	3
D_3	0	2	48

Validating Classification Models: F-measure

There is often a trade-off between the Precision and the Recall of a classifier.

- We can easily make $\text{Recall}_i = 1$ by assigning all data to class i. Then Precision_i will be low.
- Precision_i can be made very high by making very few predictions to class i (e.g., only the data instances for which the classifier has very high confidence will be assigned to class i). But then Recall_i will be low.

F-Measure: To obtain a measure that balances the precision and recall of a class, the class-specific F-Measure is defined as the harmonic mean between them.

$$\text{F}_i = \frac{2}{\frac{1}{\text{Precision}_i} + \frac{1}{\text{Recall}_i}} = \frac{2\,\text{Precision}_i\,\text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} = \frac{2n_{ii}}{|D_i| + |R_i|}$$

Validating Classification Models: F-measure

F-Measure: To obtain a measure that balances the precision and recall of a class, the class-specific F-Measure is defined as the harmonic mean between them.

$$\mathrm{F}_i = \frac{2}{\frac{1}{\mathrm{Precision}_i} + \frac{1}{\mathrm{Recall}_i}} = \frac{2\,\mathrm{Precision}_i\,\mathrm{Recall}_i}{\mathrm{Precision}_i + \mathrm{Recall}_i} = \frac{2n_{ii}}{|D_i| + |R_i|}$$

The overall F-measure is the mean of the class-specific values,

$$F = \frac{1}{k}\sum_{i=1}^{k} F_i$$

The F-measure of a perfect classifier is 1.

Example

F-Measure:

$$\mathrm{F}_i = \frac{2\,\mathrm{Precision}_i\,\mathrm{Recall}_i}{\mathrm{Precision}_i + \mathrm{Recall}_i} = \frac{2n_{ii}}{|D_i| + |R_i|}$$

Example 1: Confusion Matrix of a k-NN classifier on the Iris data set:

	R_1	R_2	R_3
D_1	50	0	0
D_2	0	47	3
D_3	0	2	48

$$\mathrm{Precision}_1 = 1, \quad \mathrm{Precision}_2 = 47/49, \quad \mathrm{Precision}_3 = 48/51$$
$$\mathrm{Recall}_1 = 1, \quad \mathrm{Recall}_2 = 47/50, \quad \mathrm{Precision}_3 = 48/50$$
$$F_1 = 1, \quad F_2 = 0.949, \quad F_3 = 0.95$$
$$F = 0.966$$

F-Measure is better for class-imbalanced problems

Example 2:

	R_1	R_2
D_1	9990	10
D_2	90	10

$$\underline{ACC = 0.998}$$

F-Measure is better for class-imbalanced problems

Example 2:

	R_1	R_2
D_1	9990	10
D_2	90	10

$$\underline{ACC = 0.998}$$

$$\text{Precision}_1 = 0.991$$

$$\text{Precision}_2 = 0.5$$

$$\text{Recall}_1 = 0.999$$

$$\text{Recall}_2 = 0.1$$

$$F_1 = 0.995$$

$$F_2 = 0.167$$

$$\underline{F = 0.581}$$

Binary Classification: TP, TN, FP, FN

For binary classification $k = 2$, we call a class c_1 the **positive** class, and the other class c_2 as the **negative** class. We obtain a 2×2 confusion matrix, whose entries have the following names.

	R_1 (Predicted Positive)	R_2 (Predicted Negative)
D_1 (GT Positive)	True Positive (TP)	False Negative (FN)
D_2 (GT Negative)	False Positive (FP)	True Negative (TN)

True Positives (TP): The number of positive-class instances that have been classified correctly.

$$\text{TP} = n_{11} = |\{x_i | \hat{y}_i = y_i = c_1\}|$$

True Negatives (TN): The number of negative-class instances that have been classified correctly.

$$\text{TN} = n_{22} = |\{x_i | \hat{y}_i = y_i = c_2\}|$$

Binary Classification: TP, TN, FP, FN

For binary classification $k = 2$, we call a class c_1 the **positive** class, and the other class c_2 as the **negative** class. We obtain a 2×2 confusion matrix, whose entries have the following names.

	R_1 (Predicted Positive)	R_2 (Predicted Negative)
D_1 (GT Positive)	True Positive (TP)	False Negative (FN)
D_2 (GT Negative)	False Positive (FP)	True Negative (TN)

False Positives (FP): The number of instances that have been incorrectly classified as positive.

$$\text{FP} = n_{21} = |\{x_i | \hat{y}_i = c_1 \text{ and } y_i = c_2\}|$$

False Negatives (FN): The number of instances that have been incorrectly classified as negative.

$$\text{FN} = n_{12} = |\{x_i | \hat{y}_i = c_2 \text{ and } y_i = c_1\}|$$

Binary Classification: Accuracy, Precision

	R_1 (Predicted Positive)	R_2 (Predicted Negative)
D_1 (GT Positive)	True Positive (TP)	False Negative (FN)
D_2 (GT Negative)	False Positive (FP)	True Negative (TN)

Accuracy:

$$ACC = \frac{TP + TN}{n}$$

Error Rates:

$$ER = \frac{FP + FN}{n}$$

Binary Classification: Accuracy, Precision

	R_1 (Predicted Positive)	R_2 (Predicted Negative)
D_1 (GT Positive)	True Positive (TP)	False Negative (FN)
D_2 (GT Negative)	False Positive (FP)	True Negative (TN)

Accuracy:

$$ACC = \frac{TP + TN}{n}$$

Error Rates:

$$ER = \frac{FP + FN}{n}$$

Positive-class Precision:

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

Negative-class Precision:

$$Precision_N = \frac{TN}{TN + FN}$$

Binary Classification: TPR, FPR

	R_1 (Predicted Positive)	R_2 (Predicted Negative)
D_1 (GT Positive)	True Positive (TP)	False Negative (FN)
D_2 (GT Negative)	False Positive (FP)	True Negative (TN)

True Positive Rate (Sensitivity):

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

True Negative Rate (Specificity):

$$TNR = Recall_N = \frac{TN}{TN + FP}$$

False Positive Rate:

$$FPR = \frac{FP}{FP + TN} = 1 - Recall_N$$

False Negative Rate:

$$FNR = \frac{FN}{FN + TP} = 1 - Recall_P$$