Model Evaluation

Evaluation Metrics for Prediction and Classification

Overview

- Need for model evaluation
- Evaluation Metrics for Prediction(Regression) models
- Evaluation Metrics for Classification models
- Asymmetrical Costs of (mis)classification
- Alternate Evaluation Measures for Classification models

Why do we need to evaluate models?

- Multiple ML algorithms applicable to classification/prediction
- Wide choice of parameter and/or hyperparameter settings possible in these algorithms
- Hence the need to evaluate each model's performance
- In all cases, performance to be evaluated on validation/test data (to avoid wrong interpretations from overfitting on training data)

Evaluating performance in Prediction

- In such scenarios, we need to evaluate how the model predicts **new data**, not how well it fits the data it was trained with (goodness-of-fit)

- Key component of most performance measures is the difference between actual *y* and predicted $\hat{y}$, which is referred to as the 'error' :

  $e_i = y_i - \hat{y}_i$

Error Measure	Formula
Mean Error	$\frac{1}{n}\sum_{i=1}^{n} e_i$
Mean Absolute Error (MAE)	$\frac{1}{n}\sum_{i=1}^{n} \lvert e_i \rvert$
Mean Percentage Error (MPE)	$100 \times \frac{1}{n}\sum_{i=1}^{n} e_i/y_i$
Mean Absolute Percentage Error	$100 \times \frac{1}{n}\sum_{i=1}^{n} \lvert e_i/y_i \rvert$
Sum of Squared Errors (SSE)	$\frac{1}{n}\sum_{i=1}^{n} {e^2}_i$
Root Mean Squared Error (RMSE)	$\sqrt{\frac{1}{n}\sum_{i=1}^{n} {e^2}_i}$

Evaluating performance in Classification

Most Classification algorithms classify via a 2-step process:

For each record,

1. Compute **probability of belonging to class '1'**
2. Compare to cutoff value, and classify accordingly

(Default cutoff value is 0.50, If >= 0.50, classify as "1", If < 0.50, classify as "0")

- Can use different cutoff values and accordingly the classification output varies

- Error = classifying a record as belonging to one class when it actually belongs to another class.

- Error rate = percent of misclassified records out of the total records in the validation/test data

Confusion Matrix

		Actual Class	
		C_1	C_2
Predicted Class	C_1	$n_{1,1}$ = number of C_1 records classified correctly as C_1	$n_{2,1}$ = number of C_2 records classified incorrectly as C_1
	C_2	$n_{1,2}$ = number of C_1 records classified incorrectly as C_2	$n_{2,2}$ = number of C_2 records classified correctly as C_2

$$\text{err} = \frac{n_{1,2} + n_{2,1}}{n},$$

When One Class is More Important & misclassification costs are asymmetrical

- In most cases it is more important to identify members of one class
  - Diagnosing illness (Illness)
  - Detecting SPAM mail (Spam mails)
  - Credit default (Potential Defaulter Class)
  - Tax fraud (Fraudulent Tax Class)
  - Response to promotional offer (Respondent Class)
  - Detecting electronic network intrusion (Malicious Packet class)
  - Predicting delayed flights (Delayed flights)

- In such cases, we are willing to tolerate greater overall error, in return for better identifying the important class for further attention

- The cost of making a misclassification error may be higher for one class than the other(s)

Asymmetrical Costs – Response to Promotional Offer

Suppose we send an offer to 1000 people, with **1% average response rate** ("1" = response, "0" = nonresponse)

- "Naïve rule" (classify everyone as '0') has error rate of 1% (seems good)
- Let's assume that by using some ML model
  - ➢ We can correctly classify eight 1's as 1's
  - ➢ It comes at the cost of misclassifying twenty 0's as 1's and two 1's as 0's.
  - ➢ Error rate = (2+20) = 2.2% (higher than naïve rate)

	Actual 1	Actual 0
Predicted 1	8	20
Predicted 0	2	970

Suppose: Profit from a **'1'** is \$10 & Cost of sending offer is \$1

- Under naïve rule, all are classified as "0", so no offers are sent: no cost, no profit
- Under ML predictions, 28 offers are sent.

  8 respond with profit of \$10 each

  20 fail to respond, cost \$1 each

  972 receive nothing (no cost, no profit)

	Actual 1	Actual 0
Predicted 1	80\$	-20\$
Predicted 0	0	0

Net profit = \$60

Thus, we need to look beyond the traditional error/accuracy metrics in classification scenarios

Ref: Shmueli et al,Data Mining for Business Analytics: Concepts, Techniques and Applications in Python, Wiley, 2019

Alternate Accuracy Measures

Predicted Class \ Actual Class	C_1	C_2
C_1	$n_{1,1}$ = number of C_1 records classified correctly as C_1 **True Positive (TP)**	$n_{2,1}$ = number of C_2 records classified incorrectly as C_1 **False Positive (FP)**
C_2	$n_{1,2}$ = number of C_1 records classified incorrectly as C_2 **False Negative (FN)**	$n_{2,2}$ = number of C_2 records classified correctly as C_2 **True Negative (TN)**

If "C_1" is the important class,

- **Sensitivity (also called "recall)** = % of actual C_1 class correctly classified

  $$n_{1,1} / (n_{1,1} + n_{1,2})$$

- Ability of the classifier to detect the important class members correctly.

- Also referred to as **True Positive Rate, TPR** = TP/ (TP + FN)

Alternate Accuracy Measures

Predicted Class	Actual Class: C_1	Actual Class: C_2
C_1	$n_{1,1}$ = number of C_1 records classified correctly as C_1 **True Positive (TP)**	$n_{2,1}$ = number of C_2 records classified incorrectly as C_1 **False Positive (FP)**
C_2	$n_{1,2}$ = number of C_1 records classified incorrectly as C_2 **False Negative (FN)**	$n_{2,2}$ = number of C_2 records classified correctly as C_2 **True Negative (TN)**

- If "C_1" is the important class,
- **Specificity** = % of actual C_2 class correctly classified

$$n_{2,2} / (n_{2,1} + n_{2,2})$$

- Ability of the classifier to rule out the other class members (C_2) correctly.
- Also referred to as **True Negative Rate, TNR** = TN / (FP + TN)
- **False Positive Rate (FPR)** = 1- Specificity

$$FPR = FP / (FP + TN)$$

Alternate Accuracy Measures

		Actual Class C_1	Actual Class C_2
Predicted Class	C_1	**$n_{1,1}$ = number of C_1 records classified correctly as C_1** **True Positive (TP)**	**$n_{2,1}$ = number of C_2 records classified incorrectly as C_1** **False Positive (FP)**
	C_2	**$n_{1,2}$ = number of C_1 records classified incorrectly as C_2** **False Negative (FN)**	**$n_{2,2}$ = number of C_2 records classified correctly as C_2** **True Negative (TN)**

If "C_1" is the important class,

- **Precision**= % of predicted C_1 that are actually C_1

  $n_{1,1}$ / ($n_{1,1}$ + $n_{2,1}$)

  TP/ (TP + FP)

- **Recall (also called "sensitivity")** = % of actual C_1 class correctly classified

  $n_{1,1}$ / ($n_{1,1}$ + $n_{1,2}$)

  TP/ (TP + FN)

- F-Measure provides a way to combine both precision and recall into a single measure that captures both properties. Also know as F-Score or F1-Score

  F1-Score= (2*Precision*Recall) /(Precision + Recall)

  - Common metric used on classification problems on imbalanced datasets.

Summary

- Evaluation metrics are important for comparing across different ML models, for choosing the right configuration of a specific ML model
- Metrics are computed from validation/test data
- Preferred metrics for evaluating regression(prediction) : RMSE
- Confusion Matrix forms the basis for evaluation in classification scenarios
- Asymmetric Costs of Mis-classification and need to go beyond error rate
- Metrics for evaluation in Classification generated from Confusion Matrix: Sensitivity, Specificity, Precision, Recall, F1 Score, etc.

References

- https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/
- https://medium.com/analytics-vidhya/complete-guide-to-machine-learning-evaluation-metrics-615c2864d916
- https://python-data-science.readthedocs.io/en/latest/evaluation.html