Evaluation Metrics & Strategies

CS534 - Machine Learning

Yubin Park, PhD

We choose a loss function to fit a model.

We choose an evaluation metric to evaluate the model.

These two can be sometimes different.

Loss Function $\neq$ Evaluation Metric

Evaluating Regression Tasks

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

Root Mean Sqaured Error (RMSE) = $\sqrt{\text{MSE}}$

R-squared (or the coefficient of determination) = $1 - \frac{\text{MSE}}{\text{Var}(y)}$

Mean Absolute Error (MAE) = $\frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$

Mean Absolute Percentage Error (MAPE) = $\frac{100}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{|y_i|}$

Median Absolute Error (MedAE) = $\text{median}(|y_i - \hat{y}_i|)$

Mean Squared Logarithmic Error (MSLE) = $\frac{1}{n} \sum_{i=1}^{n} (\log(1 + y_i) - \log(1 + \hat{y}_i))$

Regression Evaluation Example

```
>>> from sklearn.metrics import mean_squared_error
>>> y_true = [3, -0.5, 2, 7]
>>> y_pred = [2.5, 0.0, 2, 8]
>>> mean_squared_error(y_true, y_pred)
0.375
```

https://scikit-learn.org/stable/modules/model_evaluation.html#mean-squared-error

```
>>> from sklearn.metrics import mean_squared_log_error
>>> y_true = [3, 5, 2.5, 7]
>>> y_pred = [2.5, 5, 4, 8]
>>> mean_squared_log_error(y_true, y_pred)
0.039...
```

https://scikit-learn.org/stable/modules/model_evaluation.html#mean-squared-logarithmic-error

Evaluating Classification Tasks

Confusion Matrix is the basis for evaluating classification algorithms.

	Predicted 0 (or Negative)	**Predicted 1 (or Positive)**
Actual 0 (or Negative, N)	True Negative (TN)	False Positive (FP), Type I Error
Actual 1 (or Positive, P)	False Negative (FN), Type II Error	True Positive (TP)

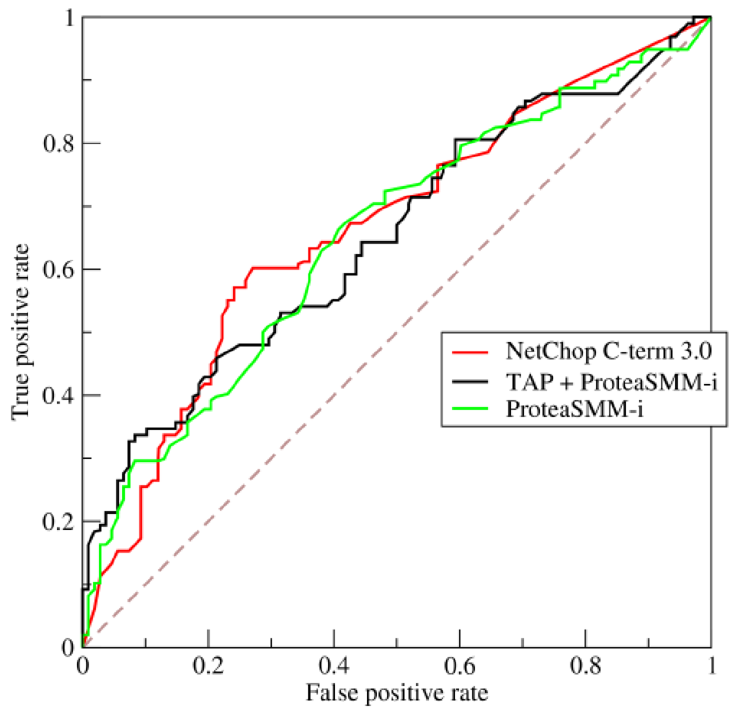
Accuracy = $\frac{TP+TN}{P+N}$

True Positive Rate (TPR) = $\frac{TP}{TP+FN}$

False Positive Rate (FPR) = $\frac{FP}{FP+TN}$

Positive Predictive Value (PPV, or Precision) = $\frac{TP}{TP+FP}$

ROC Curve and Area Under the Curve (AUROC)



If a model is good, **in general**,

AUROC should be above 0.5 and plausibly above 0.7-ish.

Higher AUROCs, in general, imply better models.

However, in practice,

lower AUROC models may be preferred depending on

the **shapes** of the ROC curves.

Precision and Recall

Precision = $\frac{TP}{TP+FP}$ and Recall = $\frac{TP}{TP+FN}$

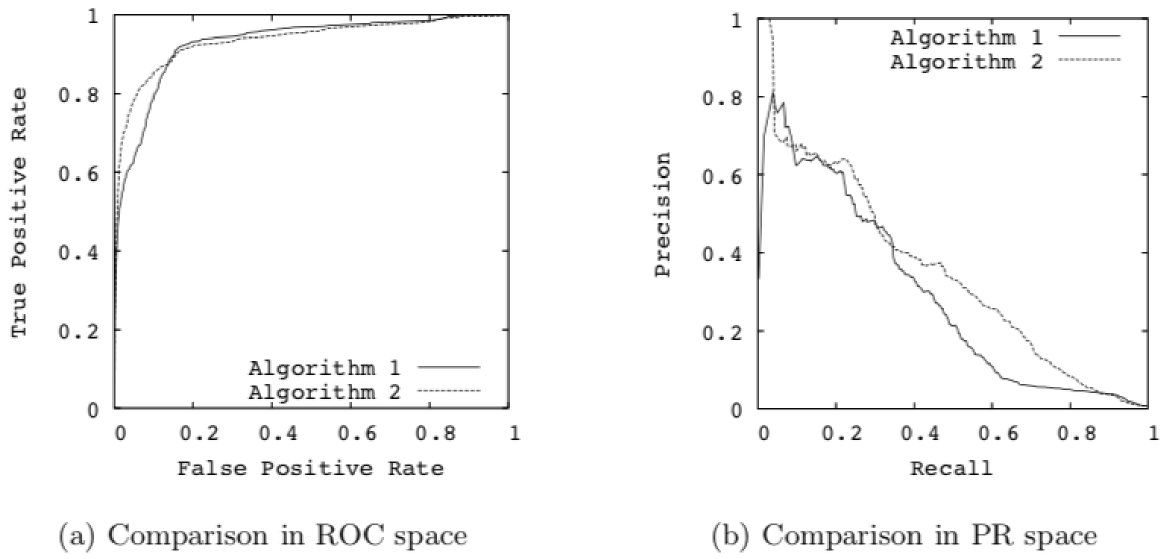


Figure 1. The difference between comparing algorithms in ROC vs PR space

J. Davis and M. Goadrich, The Relationship Between Precision-Recall and ROC Curves. ICML 2006. https://www.biostat.wisc.edu/~page/rocpr.pdf

Brier Score

ROC and PR curves only care about the rankings of the predicted values.

For example, if the target is `[0, 1]`, then predicted values of `[0.1, 0.9]` and `[0.4, 0.7]` would be the same in ROC and PR.

On the other hand, Brier Score measures the accuracy of **probabilistic** predictions.

For a binary classification problem where $y = \{0, 1\}$ and $\hat{y} \in [0, 1]$,

$$\text{Brier Score (BS)} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

i.e. MSE applied to a binary classification task

Now we know **what metrics** to measure.

But, do we know **where/how** to apply?

Why are we training a model?

To Predict Outcomes

(for the inputs you haven't seen yet)

Train and Test

By definition, you cannot measure the performance on the **unseen** data.

However, you can simulate such performance assuming that the **unseen** data is not much different from the **data we already have**.

Randomly split the data into two groups:

- **Train set** for mimicking the data that we already have
- **Test set** for mimicking the unseen data that we will have in the future

Common split ratio: 70% for Train and 30% for Test

You use the Test set only for measuring the evaluation metric.

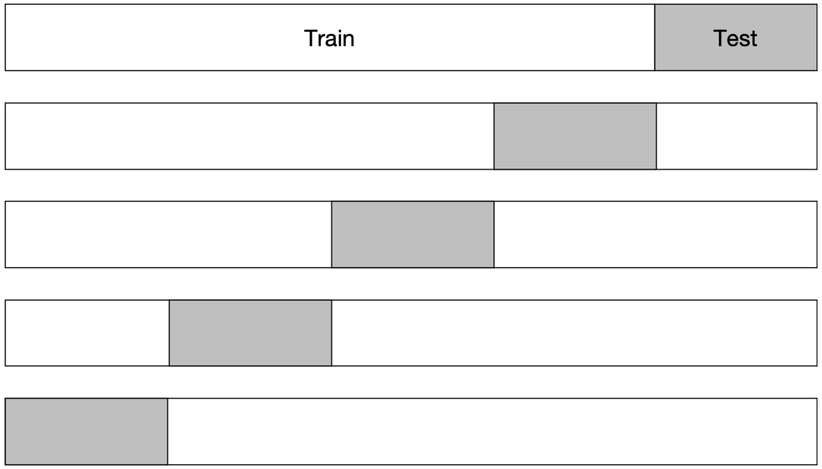
You **DO NOT** touch the Test set while training.

K-Fold Cross-Validation

One train/test split gives one estimate for the evaluation metric of interest.
Can we trust the value?

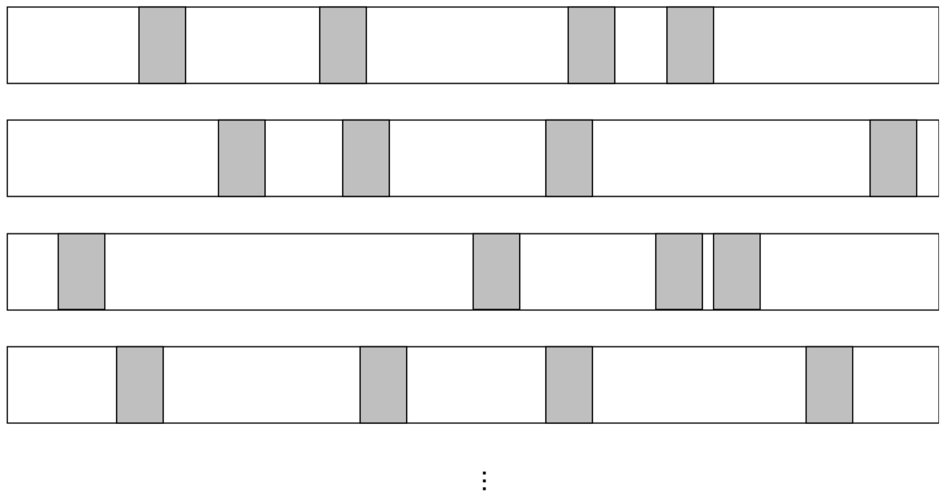
Can we get more estimates so that we can calculate variances of such estimates?

Randomly split data into k partitions, and use one of the partition as a Test set, and then iterate over the other $k-1$ Test sets.



Monte Carlo Cross-Validation

The maximum number of estimates for K-Fold CV is K. For example, a 5-Fold CV will generate 5 evaluation metrics. Is 5 enough?



Doing a random sub-sampling multiple times.

Test sets may have some overlapping data points between iterations.

Can iterate as many as we want.

Cross-validated performances should be

indicative of real performance.

We can choose the **best** model

among our list of models based on the CV results.

A QUICK NOTE

A model has two types of parameters.

Regular parameters (we just call "parameters"), and

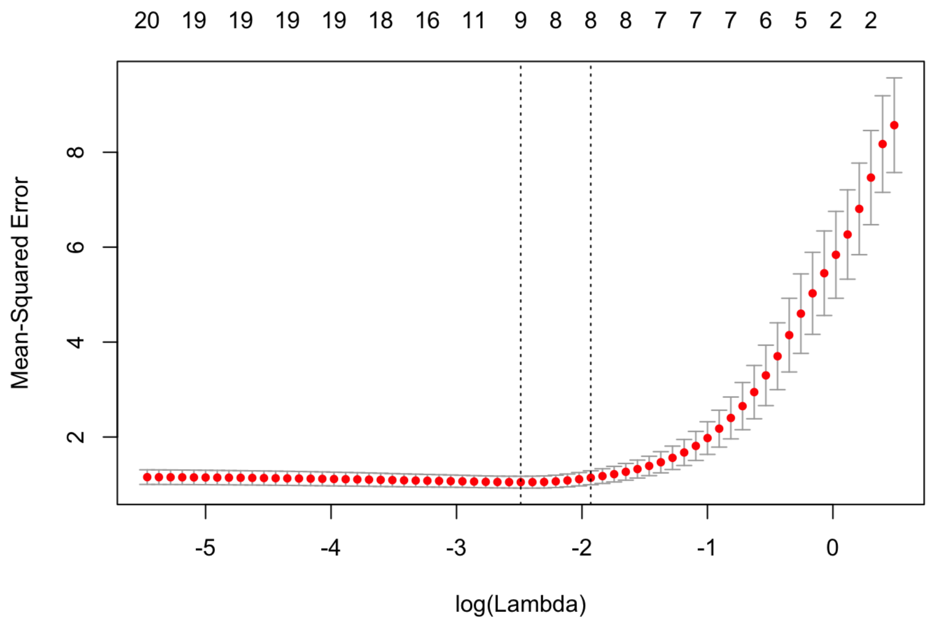
Hyperparameters, which we cannot estimate during training,

such as λ, α in Elastic-Net.

If models have different hyperparameters,

for now, let's just treat them **different** models.

Example of Choosing the best λ in Elastic-Net



https://web.stanford.edu/~hastie/glmnet/glmnet_beta.html

Questions?