Evaluation Metrics and Regularization Continued

CS145: Introduction to Data Mining

Spring 2024

Contents

1 Introduction

Evaluation metrics and regularization are two essential concepts in machine learning that play crucial roles in assessing model performance and preventing overfitting. Evaluation metrics provide quantitative measures of how well a model performs on a given task, while regularization techniques help to control model complexity and improve generalization to unseen data.

2 Evaluation Metrics

Evaluation metrics are used to assess the performance of machine learning models. The choice of metric depends on the problem type (e.g., classification, regression) and the specific goals of the application.

2.1 Classification Metrics

For binary classification problems, common metrics include:

Accuracy: The proportion of correct predictions out of the total number of predictions.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

Accuracy is a simple and intuitive metric, but it can be misleading when the classes are imbalanced. For example, if 95% of the instances belong to one class, a classifier that always predicts that class will have a 95% accuracy, even though it fails to classify any instances of the minority class.

Example: In a medical diagnosis task, if a model correctly identifies 80 patients with a disease (TP) and 90 healthy patients (TN), while misclassifying 15 healthy patients as having the disease (FP) and 15 patients with the disease as healthy (FN), the accuracy would be $\frac{80+90}{80+90+15+15} = 0.85$.

Precision: The proportion of true positive predictions out of the total positive predictions.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

Precision measures the classifier's ability to avoid false positives. It is useful when the cost of false positives is high, such as in spam email detection, where classifying a legitimate email as spam (false positive) is more harmful than letting a spam email pass through (false negative).

Example: Using the same medical diagnosis example, the precision would be $\frac{80}{80+15} \approx 0.84$. This indicates that when the model predicts a patient has the disease, it is correct 84% of the time.

Recall (Sensitivity): The proportion of true positive predictions out of the total actual positive instances.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

Recall measures the classifier's ability to find all positive instances. It is important when the cost of false negatives is high, such as in medical diagnosis, where failing to identify a disease (false negative) is more harmful than incorrectly identifying a healthy patient as having the disease (false positive).

Example: Again, using the medical diagnosis example, the recall would be $\frac{80}{80+15} \approx 0.84$. This means that the model correctly identifies 84% of all patients who actually have the disease.

F1 Score: The harmonic mean of precision and recall.

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

The F1 score provides a balanced measure of precision and recall. It is useful when both false positives and false negatives are important, and there is a need to find a balance between them. The F1 score reaches its best value at 1 and its worst value at 0.

Example: Using the precision and recall from the previous examples, the F1 score would be $2 \cdot \frac{0.84 \cdot 0.84}{0.84+0.84} \approx 0.84$. The F1 score is the harmonic mean of precision and recall, providing a balanced measure of the model's performance.

2.1.1 Precision-Recall Curves

Area Under the Precision-Recall Curve (AUC-PR) is a measure of the classifier's performance, calculated by plotting precision against recall at various threshold settings.

The PR curve shows the trade-off between precision and recall as the decision threshold varies. AUC-PR is more informative than AUC-ROC when the class distribution is highly imbalanced, and the focus is on the performance of the classifier on the minority class.

To compute a PR curve:

1. Sort the instances by their predicted probabilities (or scores) in descending order.
2. At each possible threshold, calculate the precision and recall.
3. Plot precision on the y-axis and recall on the x-axis.

A perfect classifier would have a PR curve that passes through the point (1, 1), indicating 100% precision and 100% recall. A random classifier would have a PR curve that is a horizontal line at the value of the positive class prevalence.

The Area Under the Precision-Recall Curve (AUC-PR) summarizes the PR curve into a single value. A higher AUC-PR indicates better performance. AUC-PR is computed using the trapezoidal rule for approximating the area under the curve.

Example: Consider a binary classification problem with 100 instances, 20 of which are positive. A classifier produces the following predicted probabilities:

Instance	Predicted Probability
1	0.95
2	0.90
⋮	⋮
99	0.10
100	0.05

To compute the PR curve, we would sort the instances by their predicted probabilities and calculate precision and recall at each threshold. For example, at a threshold of 0.90, we would have 2 true positives and 0 false positives, resulting in a precision of 1.0 and a recall of 0.1. Repeating this process for all thresholds and plotting the results would give us the PR curve.

2.1.2 AUC-ROC Curves

Area Under the Receiver Operating Characteristic Curve (AUC-ROC) is another measure of the classifier's ability to distinguish between classes, calculated by plotting the true positive rate against the false positive rate at various threshold settings.

The ROC curve shows the trade-off between the true positive rate (recall) and the false positive rate (1 - specificity) as the decision threshold varies. A perfect classifier would have an AUC-ROC of 1, while a random classifier would have an AUC-ROC of 0.5. AUC-ROC is useful when the class distribution is imbalanced, and the cost of false positives and false negatives is not well defined.

To compute an AUC-ROC curve:

1. Sort the instances by their predicted probabilities (or scores) in descending order.
2. At each possible threshold, calculate the TPR and FPR.
3. Plot TPR on the y-axis and FPR on the x-axis.

A perfect classifier would have an AUC-ROC curve that passes through the point (0, 1), indicating a 100% TPR and a 0% FPR. A random classifier would have an AUC-ROC curve that is a diagonal line from (0, 0) to (1, 1).

The Area Under the ROC Curve (AUC-ROC) summarizes the ROC curve into a single value. A higher AUC-ROC indicates better performance. AUC-ROC is computed using the trapezoidal rule for approximating the area under the curve.

Example: Using the same binary classification problem and predicted probabilities as in the PR curve example, we would compute the AUC-ROC curve by sorting the instances, calculating TPR and FPR at each threshold, and plotting the results. For example, at a threshold of 0.90, we would have a TPR of 0.1 and an

FPR of 0.0. Repeating this process for all thresholds and plotting the results would give us the AUC-ROC curve.

2.2 Regression Metrics

For regression problems, common metrics include:

Mean Squared Error (MSE): The average of the squared differences between the predicted and actual values.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5}$$

MSE penalizes larger errors more heavily than smaller errors due to the squaring of the differences. This makes MSE sensitive to outliers, as a few large errors can dominate the overall error. MSE is differentiable, which makes it suitable for optimization algorithms like gradient descent.

Example: If a model predicts housing prices with errors of \$5,000, \$10,000, and \$15,000 for three houses, the MSE would be $\frac{1}{3}(5000^2 + 10000^2 + 15000^2) \approx 1.08 \times 10^8$. MSE penalizes larger errors more heavily due to the squared term.

Root Mean Squared Error (RMSE): The square root of the MSE, which has the same units as the target variable.

$$\text{RMSE} = \sqrt{\text{MSE}} \tag{6}$$

RMSE is interpretable in the same units as the target variable, making it easier to understand the magnitude of the errors. Like MSE, RMSE is sensitive to outliers.

Mean Absolute Error (MAE): The average of the absolute differences between the predicted and actual values.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{7}$$

MAE treats all errors equally, regardless of their magnitude. This makes MAE more robust to outliers compared to MSE and RMSE. However, MAE is not differentiable at zero, which can make it less suitable for some optimization algorithms.

Example: Using the same housing price prediction example, the MAE would be $\frac{1}{3}(5000 + 10000 + 15000) = 10000$. MAE treats all errors equally and is more interpretable than MSE.

R-squared (R^2): The proportion of the variance in the target variable that is predictable from the input features.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{8}$$

where $\bar{y}$ is the mean of the actual values.

R^2 ranges from 0 to 1, with 1 indicating a perfect fit and 0 indicating a model that performs no better than predicting the mean of the target variable. R^2 can be negative if the model performs worse than predicting the mean. R^2 is scale-invariant, meaning that multiplying the target variable by a constant does not change the value of R^2. However, R^2 does not indicate whether the model's coefficients are statistically significant or whether the model is biased.

Example: If a model predicts housing prices with an R^2 of 0.75, this means that the model explains 75% of the variance in the target variable (housing prices) compared to a baseline model that always predicts the mean price.

2.3 Discussions

The choice of evaluation metric depends on the specific problem and the goals of the analysis. In classification tasks, accuracy is often the first metric considered, but it can be misleading when the classes are imbalanced. In such cases, precision, recall, F1 score, AUC-ROC, and AUC-PR provide more informative measures of the classifier's performance.

In regression tasks, MSE, RMSE, and MAE are commonly used to measure the magnitude of the errors. MSE and RMSE are sensitive to outliers, while MAE is more robust. R^2 provides a measure of the proportion of variance explained by the model, but it does not indicate the statistical significance of the model's coefficients or the presence of bias.

It is important to consider multiple evaluation metrics and to understand their strengths and limitations when assessing the performance of a machine learning model. Additionally, it is crucial to evaluate the model on a separate test set that was not used during training to get an unbiased estimate of its generalization performance.

2.4 Limitations

- Evaluation metrics provide a summary of the model's performance but do not give a complete picture of its behavior. It is possible for a model to have good performance according to one metric but poor performance according to another.
- Evaluation metrics are based on the assumption that the test data is representative of the real-world data the model will encounter. If the test data is not representative, the metrics may not accurately reflect the model's true performance.
- Some evaluation metrics, such as accuracy, can be misleading when the class distribution is imbalanced. It is important to consider the class distribution and the costs of different types of errors when selecting evaluation metrics.
- Evaluation metrics do not provide information about the model's interpretability, fairness, or robustness to adversarial attacks. These factors may be important in certain applications, such as healthcare or finance, where the model's decisions need to be explainable and unbiased.
- Evaluation metrics are based on the model's predictions on a finite sample of data. The metrics are subject to sampling variability, and the true performance of the model may differ from the estimated performance based on the sample.

3 Regularization

Regularization is a technique used to prevent overfitting in machine learning models by adding a penalty term to the loss function. The penalty term discourages the model from learning overly complex or extreme parameter values, thereby improving the model's generalization performance on unseen data.

3.1 L1 Regularization (Lasso)

L1 regularization, also known as Lasso (Least Absolute Shrinkage and Selection Operator), adds the sum of the absolute values of the model's parameters to the loss function:

$$\mathcal{L}_{\text{L1}}(\boldsymbol{w}) = \mathcal{L}(\boldsymbol{w}) + \lambda \sum_{i=1}^{d} |w_i| \tag{9}$$

where $\boldsymbol{w} = (w_1, \ldots, w_d)$ are the model's parameters, $\mathcal{L}(\boldsymbol{w})$ is the original loss function, and λ is the regularization strength hyperparameter.

L1 regularization has the effect of shrinking some of the parameter values to exactly zero, resulting in a sparse model. This can be useful for feature selection, as the non-zero parameters correspond to the most important features.

The geometric interpretation of L1 regularization is that it constrains the parameter vector to lie within a diamond-shaped region centered at the origin. The corners of the diamond correspond to sparse solutions where some of the parameters are exactly zero.

L1 regularization is particularly useful when the number of features is large, and there is a belief that only a small subset of the features are relevant for the task. By setting some of the parameters to zero, L1 regularization effectively performs feature selection, making the model more interpretable and reducing its complexity.

However, L1 regularization has some limitations:

1. When there are multiple correlated features, L1 regularization tends to select only one of them arbitrarily, leading to unstable feature selection.
2. L1 regularization can lead to biased parameter estimates, especially when the regularization strength is high.
3. The loss function with L1 regularization is not differentiable at the origin, which can make optimization more challenging.

3.2 L2 Regularization (Ridge)

L2 regularization, also known as Ridge regression or Tikhonov regularization, adds the sum of the squared values of the model's parameters to the loss function:

$$\mathcal{L}_{\text{L2}}(\boldsymbol{w}) = \mathcal{L}(\boldsymbol{w}) + \lambda \sum_{i=1}^{d} w_i^2 \tag{10}$$

L2 regularization encourages the model to learn smaller parameter values, but unlike L1 regularization, it does not result in sparse solutions. Instead, it helps to distribute the importance more evenly among the features.

The geometric interpretation of L2 regularization is that it constrains the parameter vector to lie within a circular region centered at the origin. The radius of the circle is determined by the regularization strength λ.

L2 regularization is useful when there are many features that are believed to be relevant for the task, and the goal is to prevent the model from overfitting by keeping the parameter values small. L2 regularization is also differentiable, making it easier to optimize compared to L1 regularization.

However, L2 regularization also has some limitations:

1. L2 regularization does not perform feature selection, as it does not set any parameters exactly to zero. This can make the model less interpretable when there are many irrelevant features.
2. L2 regularization can lead to high bias when the regularization strength is high, as it heavily constrains the parameter values.

3. L2 regularization is sensitive to the scale of the features, so it is important to standardize the features before applying L2 regularization.

3.3 Elastic Net Regularization

Elastic Net regularization combines both L1 and L2 penalties:

$$\mathcal{L}_{\text{Elastic Net}}(\boldsymbol{w}) = \mathcal{L}(\boldsymbol{w}) + \lambda_1 \sum_{i=1}^{d} |w_i| + \lambda_2 \sum_{i=1}^{d} w_i^2 \tag{11}$$

where λ_1 and λ_2 control the strength of the L1 and L2 penalties, respectively.

Elastic Net regularization can be useful when there are multiple correlated features, as it can select groups of correlated features together. The L1 penalty promotes sparsity, while the L2 penalty helps to distribute the importance among the selected features.

The geometric interpretation of Elastic Net regularization is that it constrains the parameter vector to lie within a region that is a combination of the diamond-shaped region of L1 regularization and the circular region of L2 regularization. The exact shape of the region depends on the values of λ_1 and λ_2.

Elastic Net regularization strikes a balance between the feature selection property of L1 regularization and the stability of L2 regularization. It can be useful when there are many correlated features, and the goal is to select a subset of them while maintaining stable parameter estimates.

However, Elastic Net regularization also has some limitations:

1. Elastic Net regularization introduces two hyperparameters (λ_1 and λ_2), which can make hyperparameter tuning more challenging compared to L1 or L2 regularization alone.
2. Like L1 regularization, Elastic Net regularization can lead to biased parameter estimates when the regularization strength is high.
3. The optimal values of λ_1 and λ_2 depend on the specific problem and the characteristics of the data, so it may require extensive experimentation to find the best combination.

3.4 Regularization in Neural Networks

In the context of neural networks, regularization techniques are often applied to the weights of the network. Some common regularization methods for neural networks include:

1. L1 and L2 regularization on the weights, as described above.
2. Dropout: During training, randomly set a fraction of the activations to zero, which helps prevent the network from relying too heavily on any single feature or neuron.
   Dropout can be interpreted as a form of ensemble learning, where each training iteration corresponds to training a different subnetwork with a random subset of the neurons. At test time, the predictions of all possible subnetworks are averaged, which helps to reduce overfitting and improve generalization.
3. Early stopping: Monitor the model's performance on a validation set during training and stop training when the performance on the validation set starts to degrade, indicating overfitting.
   Early stopping helps to prevent the model from overfitting to the training data by limiting the number of training iterations. It is a simple and effective regularization technique that does not require modifying the model architecture or the loss function.
4. Batch normalization: Normalize the activations of each layer to have zero mean and unit variance, which can help stabilize training and reduce the sensitivity to the choice of initialization.
   Batch normalization helps to reduce the internal covariate shift, which is the change in the distribution of the activations due to the updates of the previous layers' parameters. By normalizing the activations,

batch normalization allows the model to learn more stable representations and reduces the risk of overfitting.

The choice of regularization method and strength depends on the specific problem and the characteristics of the dataset. It is often necessary to experiment with different regularization techniques and hyperparameter values to find the best combination for a given task.

3.5 Discussions

Regularization is a powerful technique for preventing overfitting and improving the generalization performance of machine learning models. L1, L2, and Elastic Net regularization are commonly used regularization methods that add penalty terms to the loss function based on the magnitude of the model's parameters.

L1 regularization promotes sparsity and can be useful for feature selection, while L2 regularization encourages smaller parameter values and helps to distribute the importance among the features. Elastic Net regularization combines the strengths of L1 and L2 regularization and can be useful when there are multiple correlated features.

In neural networks, additional regularization techniques such as dropout, early stopping, and batch normalization can be used to prevent overfitting and improve generalization. These techniques help to reduce the model's reliance on specific features or neurons, limit the number of training iterations, and stabilize the training process.

The effectiveness of regularization depends on the choice of the regularization method and the appropriate setting of the hyperparameters. It is important to experiment with different regularization techniques and to use cross-validation to select the best hyperparameter values for a given problem.

Regularization should be used in conjunction with other techniques such as feature scaling, cross-validation, and model selection to ensure the best possible performance and generalization of the machine learning model.

3.6 Limitations

In summary, regularization is a powerful technique for preventing overfitting and improving the generalization performance of machine learning models. However, it is important to be aware of its limitations and to use it in combination with other techniques to ensure the best possible results. The choice of the regularization method and the setting of the hyperparameters should be based on a careful analysis of the problem and the characteristics of the dataset, and the performance of the regularized model should be evaluated using appropriate metrics and cross-validation techniques.

1. Regularization introduces additional hyperparameters (e.g., λ for L1 and L2 regularization) that need to be tuned. Hyperparameter tuning can be time-consuming and computationally expensive, especially when dealing with large datasets or complex models.
2. Regularization techniques are not guaranteed to prevent overfitting in all cases. If the model is too complex or the regularization strength is not set appropriately, the model may still overfit to the training data.
3. Regularization can lead to increased bias in the model's predictions, especially when the regularization strength is high. This is because regularization constrains the model's parameters and limits its ability to fit the training data perfectly.
4. The effectiveness of regularization depends on the choice of the regularization method and the characteristics of the dataset. Some regularization methods may work well for certain types of data but not for others. It is important to experiment with different regularization techniques and to compare their performance using appropriate evaluation metrics.

5. Regularization does not address other sources of overfitting, such as data leakage or improper model selection. It is important to use regularization in conjunction with other techniques to ensure the best possible performance and generalization of the machine learning model.
6. Regularization can make the model's predictions more difficult to interpret, especially when using L1 regularization or dropout. This is because regularization can lead to sparse or noisy parameter estimates, which may not have a clear interpretation in terms of the original features.
7. Regularization may not be effective when dealing with very high-dimensional datasets (e.g., images or text data) or when the number of features is much larger than the number of samples. In such cases, other techniques such as feature extraction, dimensionality reduction, or transfer learning may be more appropriate.