




Lesson_5

Generalization in Machine Learning

Ali Aburas, PhD

What is Generalization in Machine Learning?

- 
- The goal of a good machine learning model is to generalize well from the training data to any data from the problem domain. This allows us to make predictions in the future on data the model has never seen. **“learning general concepts from specific examples”**
 - A spam email classifier is a great example of generalization in machine learning.
 - Suppose you have a training dataset containing emails labeled as either spam or not spam
 - and your goal is to build a model that can accurately classify incoming emails as spam or legitimate based on their content.
 - During the training phase, the machine learning algorithm learns from the set of labeled emails, extracting relevant features and patterns to make predictions.
 - The true test of the model's effectiveness lies when new emails arrive, the model needs to accurately classify them as spam or legitimate without prior exposure to their content. **This is where generalization comes in.**
 - ***Overfitting and underfitting*** are the two biggest causes for **poor performance of machine learning algorithms.**

Considering Regression models

- Here we have 3 Models, which have the following training and testing errors.

Assuming,

Model 1:

Training Error
= 2%

Testing Error
= 18%

Model 2:

Training Error
= 24%

Testing Error
= 28%

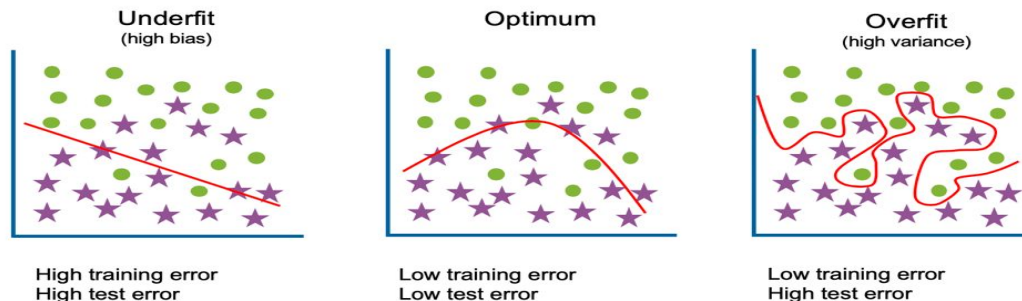
Model 3:

Training Error
= < 10%

Testing Error
= < 10%

Overfitting vs Underfitting

- What is overfitting?
 - A model is **overfitted** if it offers ideal predictions when tested against training data but fails against new, unidentified (validating) data.
- What is underfitting?
 - An **underfit** model performs poorly both on training and new (validating) data.
- Indicators of overfitting and underfitting: Bias and variance
 - **Bias** is the difference between the average prediction of our model and the correct value which we are trying to predict.
 - A model with **high bias** pays very little attention to the training data and oversimplifies the model.
 - **Variance** is the error rate of the testing data.
 - A model with **high variance** pays a lot of attention to training data and does not generalize on the data which it hasn't seen before.



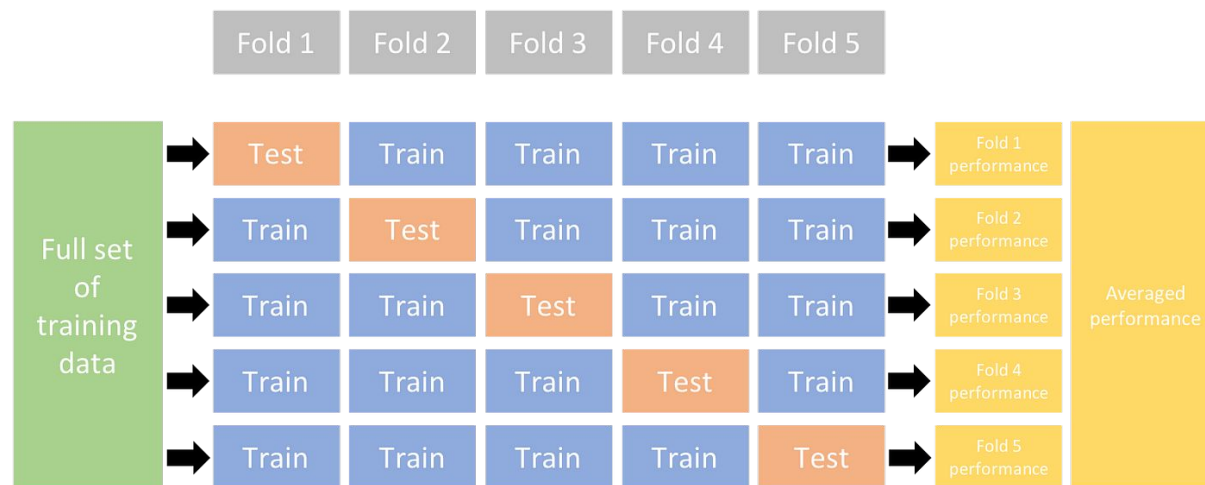
Techniques for Generalization

—

Cross-Validation

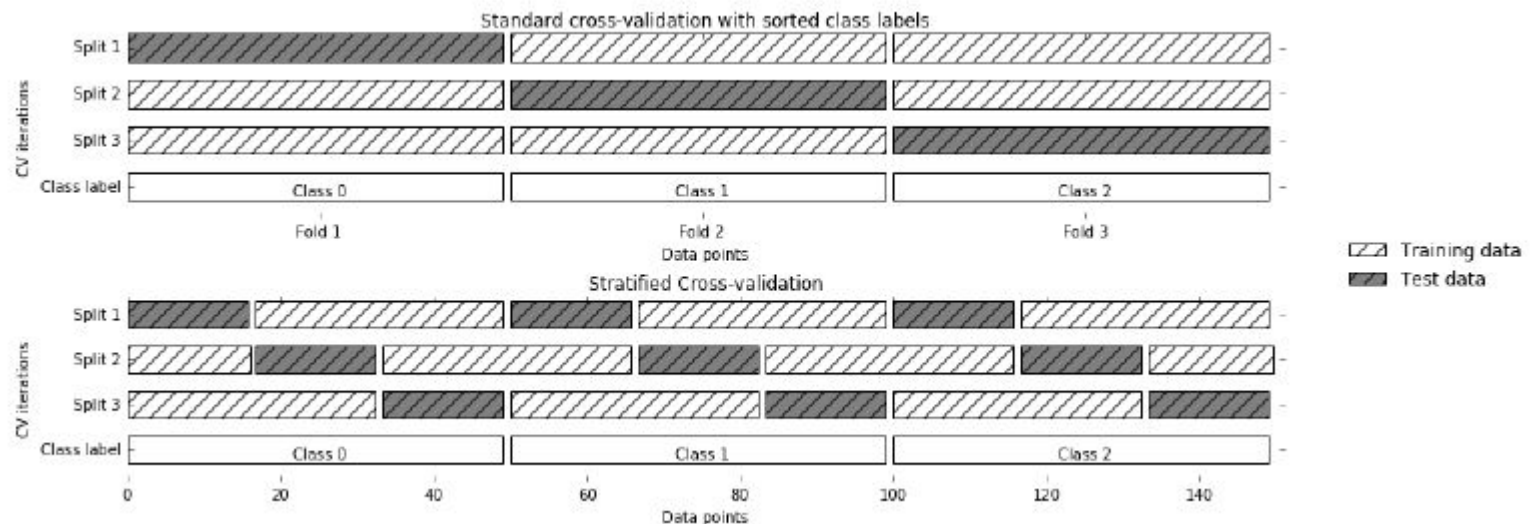
- Cross-Validation

- It involves splitting the available data into multiple subsets, training the model on a portion of the data, and evaluating its performance on the remaining test set.
- Cross-validation provides a more robust estimation of the model's generalization ability, helping in model selection.



Stratified k-Fold Cross-Validation

- In stratified cross-validation, we split the data such that the proportions between classes are the same in each fold as they are in the whole dataset.
- For example, if 90% of your samples belong to class A and 10% of your samples belong to class B, then stratified cross-validation ensures that in each fold, 90% of samples belong to class A and 10% of samples belong to class B.
- In the case of only 10% of samples belonging to class B, using standard k-fold cross-validation it might easily happen that one fold only contains samples of class A.



Feature Engineering

- **Feature engineering** is the process of using domain knowledge to extract features from raw data that make machine learning algorithms work more efficiently.
 - High-quality features often lead to improved model accuracy.
 - Effective feature engineering can reduce the complexity of models, making them faster and more efficient.
- **Techniques for Feature Engineering**
 - **Handling Missing Values:** Missing data can significantly impair model performance
 - Imputation: Replacing missing values with the mean, median, or mode of the column.
 - Deletion: Removing rows or columns with missing values.
 - **Encoding Categorical Data:** Machine learning models require numerical input, but many datasets contain categorical variables.
 - Label Encoding: Assigning each category a unique number.
 - One-Hot Encoding: Creating binary columns for each category.
 - Target Encoding: Replacing categories with the mean target value for each category.
- **Feature scaling** ensures that all features contribute equally to the model's performance
 - **Normalization:** Scaling features to a range of $[0, 1]$.
 - **Standardization:** Scaling features to have zero mean and unit variance.
- **Feature Creation:** Creating new features can provide additional predictive power
 - **Interaction Features:** Combining two or more features to capture their interaction.
 - **Polynomial Features:** Creating polynomial terms to model non-linear relationships.

Feature Scaling



- There are two common ways to get all attributes to have the same scale: *min-max* scaling and *mean normalization*.
 - *min-max scaling*: subtracting the min value and dividing by the max minus the min.

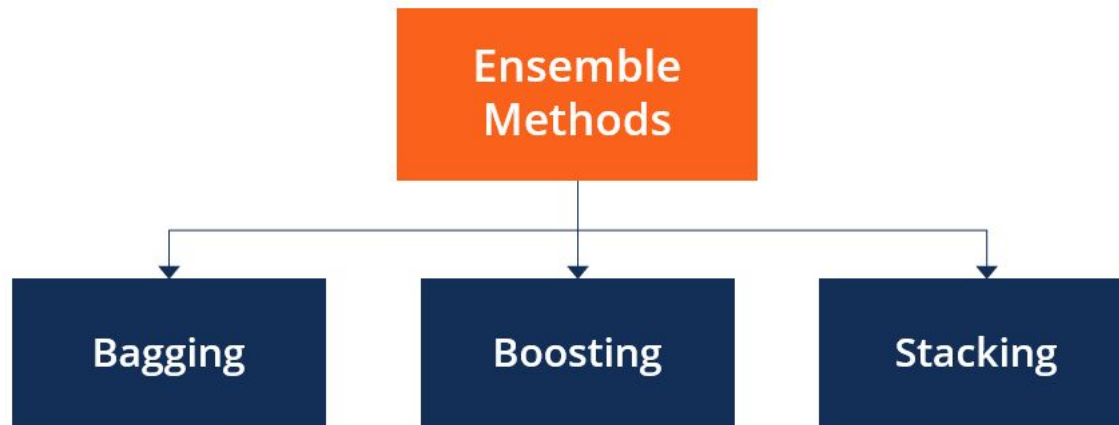
$$X_{\text{new}} = (X - X_{\text{min}}) / (X_{\text{max}} - X_{\text{min}})$$

- **Mean normalization**: This method is more or less the same as the previous method but here instead of the minimum value, we subtract each entry by the mean value of the whole data and then divide the results by the difference between the minimum and the maximum value.

$$X_{\text{new}} = (X - X_{\text{mean}}) / (X_{\text{max}} - X_{\text{min}})$$

Ensemble Methods

- **Ensemble Methods** combine predictions from multiple models to make more accurate, robust predictions.
- Techniques like *bagging*, *boosting*, and *random forests* create diverse models and aggregate their predictions, leading to more robust and generalized outcomes.



Regularization

- **Regularization:**

- This technique combats overfitting by adding a **penalty term to the model's loss function**, discouraging overly complex models and promoting simpler and more generalized representations.
- Techniques like L1 and L2 regularization (also known as ridge and lasso regression) help to control model complexity and prevent overfitting.

$$L1 : \quad J(\theta) = \frac{1}{m} \sum_{i=1}^m \text{Cost}(h_{\theta}(x^{(i)}), y^{(i)}) \quad s. t. \quad \|\theta\|_1 \leq C$$

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m \text{Cost}(h_{\theta}(x^{(i)}), y^{(i)}) + \frac{\lambda}{m} \sum_{j=1}^n |\theta_j|$$

$$L2 : \quad J(\theta) = \frac{1}{m} \sum_{i=1}^m \text{Cost}(h_{\theta}(x^{(i)}), y^{(i)}) \quad s. t. \quad \|\theta\|_2^2 \leq C^2$$

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m \text{Cost}(h_{\theta}(x^{(i)}), y^{(i)}) + \frac{\lambda}{2m} \sum_{j=1}^n \theta_j^2$$

m = number of samples, n = number of features

L1 and L2



- What exactly is L1 and L2 regularization?
 - L1 regularization, also known as LASSO regression adds the absolute value of each coefficient as a penalty term to the loss function.
 - L2 regularization, also known as Ridge regression adds the squared value of each coefficient as a penalty term to the loss function.

Choosing λ



- In both ridge and LASSO regression, we see that the larger our choice of the regularization parameter λ , the more heavily we penalize large values in θ
 - If λ is close to zero, we recover the MSE, i.e. ridge and LASSO regression is just ordinary regression.
 - If λ is sufficiently large, the MSE term in the regularized loss function will be insignificant and the regularization term will force *bridge* and *LASSO* to be close to zero
- To avoid ad-hoc choices, we should select λ using cross-validation.