Model Selection

Introduction

- Model selection is the process of deciding which algorithm and model architecture is best suited for a particular task or dataset.
- It entails contrasting various models, assessing their efficiency, and choosing the one that most effectively addresses the issue at hand.
- The choice of an appropriate machine learning model is crucial since there are various levels of complexity, underlying assumptions, and capabilities among them.
- A model's ability to generalize to new, untested data may not be as strong as its ability to perform effectively on a single dataset or problem. Finding a perfect balance between the complexity of models & generalization is therefore key to model selection.

Model selection processes

- **Problem formulation: Clearly express the issue at hand**, including the kind of predictions or task that you'd like the model to carry out (for example, classification, regression, or clustering).

- **Candidate model selection:** Pick a group of models that are appropriate for the issue at hand. These models can include straightforward methods like decision trees or linear regression as well as more sophisticated ones like deep neural networks, random forests, or support vector machines.

- **Performance evaluation:** Establish measures for measuring how well each model performs. Common measurements include area under the receiver's operating characteristic curve (AUC-ROC), recall, F1-score, mean squared error, and accuracy, precision, and recall. The type of problem and the particular requirements will determine which metrics are used.

- **Training and evaluation:** Each candidate model should be trained using a subset of the available data (the training set), and its performance should be assessed using a different subset (the validation set or via cross-validation). The established evaluation measures are used to gauge the model's effectiveness.

- **Model comparison:** Evaluate the performance of various models and determine which one performs best on the validation set. Take into account elements like data handling capabilities, interpretability, computational difficulty, and accuracy.

- **Hyperparameter tuning:** Before training, many models require that certain hyperparameters, such as the learning rate, regularisation strength, or the number of layers that are hidden in a neural network, be configured. Use methods like grid search, random search, and Bayesian optimization to identify these hyperparameters' ideal values.

- **Final model selection:** After the models have been analyzed and fine-tuned, pick the model that performs the best. Then, this model can be used to make predictions based on fresh, unforeseen data.

Model selection techniques

Techniques used for selecting models are:

1. **Train-Test Split:** With this strategy, the available data is divided into two sets: a training set & a separate test set. This method offers a quick and easy way to evaluate a model's performance using hypothetical data.
2. **Cross-Validation:** A resampling approach called cross-validation divides the data into various groups or folds. Several folds are used as the test set & the rest folds as the training set, and the models undergo training and evaluation on each fold separately.
3. **Random Search:** A set distribution for hyperparameter values is sampled at random as part of the random search hyperparameter tuning technique. Random search only investigates a portion of the hyperparameter field. When a thorough search is not possible due to the size of the search space, this strategy can be helpful.

- **Bayesian optimization:** It models the relationship between the performance of the model and the hyperparameters using a probabilistic model. It intelligently chooses which set of hyperparameters to investigate next by updating the probabilistic model and iteratively assessing the model's performance. When the search space is big and expensive to examine, Bayesian optimization is especially effective.
- **Model averaging:** This technique combines forecasts from various models to get a single prediction. Model averaging can increase overall prediction accuracy by lowering the bias and variation of individual models.
- **Information Criteria:** Information criteria offer a numerical assessment of the trade-off between model complexity and goodness of fit. These criteria discourage the use of too complicated models and encourage the adoption of simpler models that adequately explain the data.

- **Domain Expertise & Prior Knowledge:** Prior understanding of the problem and the data, as well as domain expertise, can have a significant impact on model choice.

- **Model Performance Comparison:** Using the right assessment measures, it is vital to evaluate the performance of various models. Depending on the issue at hand, these measurements could include F1-score, mean squared error, accuracy, precision, recall.

Ensemble classifiers

- The basic idea behind **the Ensemble Method is to combine a collection of models** in hopes that their unity will achieve better performance rather than designing a new model from scratch. This collection of models is referred to as a model ensemble.

Example :Let’s say you're building a spam filter.

Instead of training just one model, you build an ensemble:

- Model A — a Decision Tree trained on word frequency
- Model B — a Logistic Regression model trained on sender behavior
- Model C — a k-Nearest Neighbors model trained on email structure

Now, to classify an incoming email, each model gives its prediction (spam or not spam), and the ensemble aggregates them .

Combining classifiers

What types of classifiers are to be combined?

- **Homogeneous combination approach:** classifiers deriving from the same classification algorithm

Suppose you are using Decision Tree classifiers to detect spam emails.

  - Decision Tree 1 on emails from Gmail users
  - Decision Tree 2 on emails from Yahoo users
  - Decision Tree 3 on emails from corporate accounts
  - Each classifier is trained on a different subset of the data but uses the same algorithm (Decision Tree).

You then combine their outputs .This is a homogeneous ensemble, because all classifiers are of the same type.

- **Heterogeneous combination approaches**: if the fusion panel contains different classifiers

suppose you're again trying to detect spam emails, but this time you use:

  - A Naive Bayes classifier trained on the frequency of spam words
  - A Support Vector Machine (SVM) trained on email metadata
  - A Logistic Regression model trained on email sender behavior

Each classifier uses a different learning algorithm and may be even different feature sets.
combine their outputs to get the final spam/not-spam prediction.
This is a heterogeneous ensemble, because the classifiers are of different types.

- Techniques to perform ensemble decision trees:
    1. Bagging
    2. Boosting

Bagging :

- Is used when our goal is to reduce the variance of a decision tree.
- Each tree is built from a subset of the training dataset by **random sampling the training dataset with replacement.**
- This technique is also called bootstrapping
- When training the decision trees on the bootstrapped samples, the goal is to build very deep overfitting trees. Each tree thus has very high variance. As multiple models are used for the final prediction in the ensemble, the overall outcome is a model with low bias. This is because if many overfitting models are averaged out, the result is a model that does not overfit to any one sample so the ensemble has low bias and lower variance. So, while each tree is a robust modelling of a data sample, the united model ensemble is more powerful.

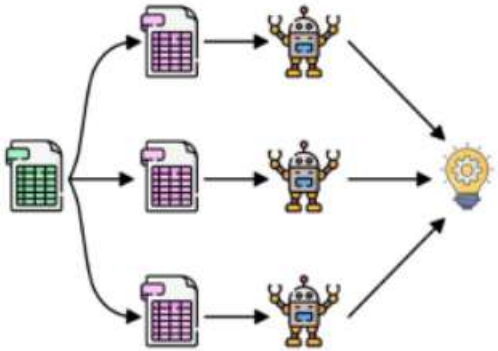
Steps involved in bagging

1. **Selection of Subset**: Bagging starts by choosing a random sample, or subset, from the entire dataset.
2. **Bootstrap Sampling**: Each model is then created from these samples, called Bootstrap Samples, which are taken from the original data with replacement. This process is known as row sampling.
3. **Bootstrapping**: The step of row sampling with replacement is referred to as bootstrapping.
4. **Independent Model Training**: Each model is trained independently on its corresponding Bootstrap Sample. This training process generates results for each model.
5. **Majority Voting**: The final output is determined by combining the results of all models through majority voting. The most commonly predicted outcome among the models is selected.
6. **Aggregation**: This step, which involves combining all the results and generating the final output based on majority voting, is known as aggregation.

Boosting:

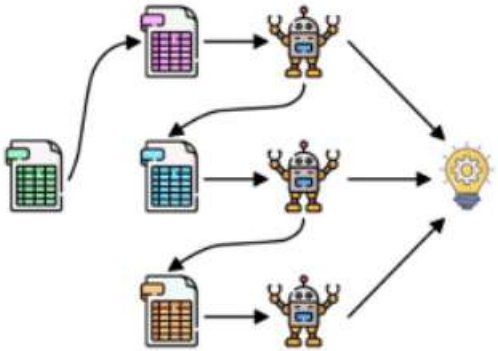
- Boosting means that each tree is dependent on prior trees.
- The algorithm learns by fitting the residual of the trees that preceded it.
- Thus, boosting in a decision tree ensemble tends to improve accuracy with some small risk of less coverage.
- In boosting each tree is dependent on prior trees.
- A boosting algorithm combines multiple simple models (also known as weak learners or base estimators) to generate the final output. It is done by building a model by using weak models in series.

Bagging



Parallel

Boosting



Sequential

Random forest

- The output of multiple decision trees to reach a single result
- It handles both classification and regression problems.
- The algorithm’s strength lies in its ability to handle complex datasets and mitigate overfitting.
- It can handle the data set containing ***continuous variables,*** as well as ***categorical variables***

Random forest algorithm

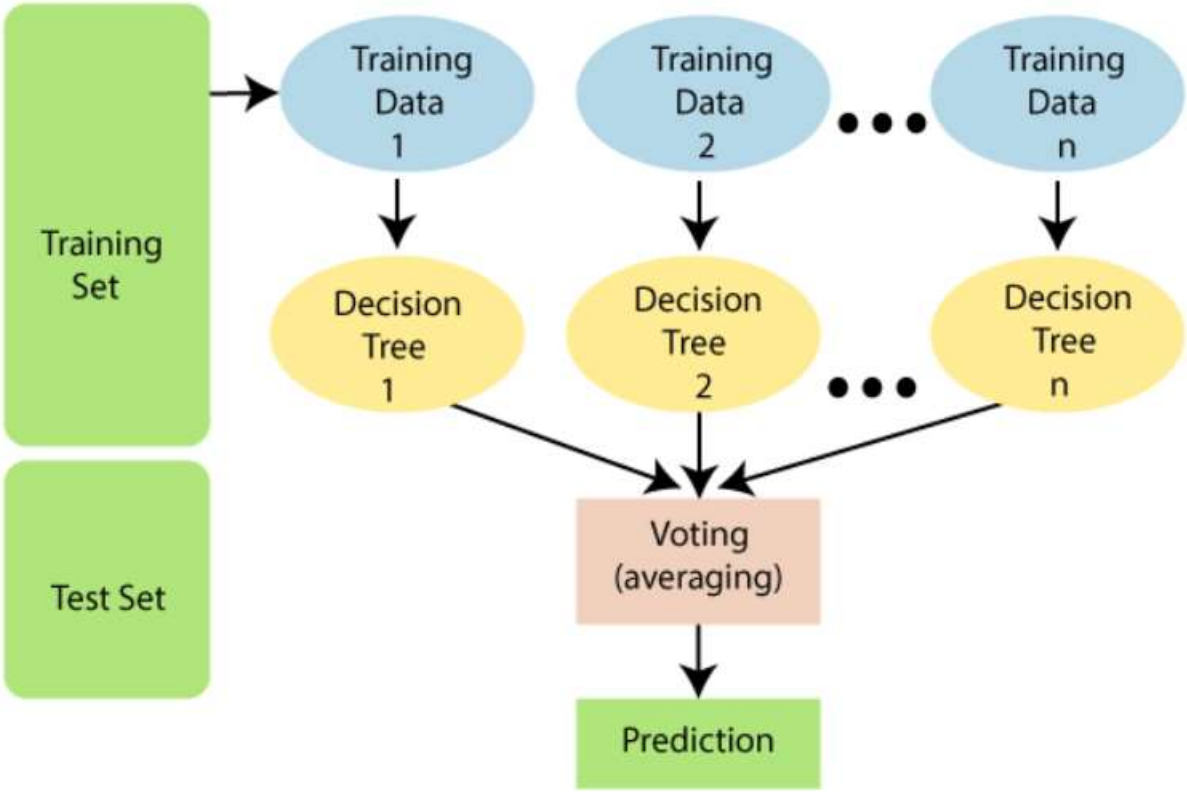
Step 1: In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on ***Majority Voting or Averaging*** for Classification and regression, respectively.

- For Classification:
  - Each tree votes for a class label.
  - The class with the most votes is chosen.
  - This is called Majority Voting.
- For Regression:
  - Each tree gives a numeric prediction.
  - The average of all tree outputs is taken.
  - This is called Averaging.


Training Set
Test Set
Training Data 1
Training Data 2
Training Data n
Decision Tree 1
Decision Tree 2
Decision Tree n
Voting (averaging)
Prediction

Features

- **Diversity:** Not all attributes/variables/features are considered while making an individual tree; each tree is different.
- **Immune to the curse of dimensionality:** Since each tree does not consider all the features, the feature space is reduced.
- **Parallelization:** Each tree is created independently out of different data and attributes. This means we can fully use the CPU to build random forests.
- **Train-Test split:** In a random forest, we don’t have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.
- **Stability:** Stability arises because the result is based on majority voting/ averaging.

Advantages

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.
- Random Forest is capable of performing both Classification and Regression tasks

Application

1. **Banking:** Used for identifying loan risk.
2. **Medicine:** Disease trends and risks of the disease can be identified.
3. **Land Use:** We can identify the areas of similar land use by this algorithm.
4. **Marketing:** Marketing trends can be identified using this algorithm.

Evaluation Metrics for Classifiers

- **Accuracy**
- **Precision, Recall, and F1 Score**
- **Confusion Matrix**
- **ROC Curve & AUC (Area Under the Curve)**

Validation Methods

Holdout Validation

- Split data into **training** and **test** sets (e.g., 70/30 or 80/20).
- Simple but can be data-dependent.

K-Fold Cross-Validation

- Split data into *k* parts, train on *k-1* and test on 1, repeated *k* times.
- Commonly used: **k=5** or **k=10**.
- **More stable estimates** of performance.

Validation Methods

Stratified K-Fold Cross-Validation

- Ensures that **class proportions are preserved** in each fold.
- Especially useful for **imbalanced datasets**.

Leave-One-Out Cross-Validation (LOOCV)

- Special case of K-fold where k = number of data points.
- Very thorough but **computationally expensive**.

Bootstrap Sampling

- Samples are drawn **with replacement**.
- Provides confidence intervals for evaluation metrics

Visualization Tools

- **Learning Curves:** Plot training vs. validation error across iterations.
- **Precision-Recall Curve:** Useful for evaluating performance on **imbalanced classes**.
- **Calibration Curves:** For checking the **quality of predicted probabilities**.