Departamento de Informática
Universidad Carlos III de Madrid

Bachelor in Robotics Engineering

Machine Learning
Course 2024-2025

Tutorial 3: Data Pre-processing and Evaluation

Use additional tools for data processing and Cross-validation. Analyze differences between ML Performance and Task Performance

October 13, 2025

- The tutorial aims to understand the importance of pre-processing data and perform the right evaluation to ensure robust and reliable models in machine learning.
- Additionally, the tutorial will be used to distinguish between Task and ML performance.
- The tutorial can be completed on Linux, Windows, or Mac.
- It is essential to complete the exercises in sequence.

1 Introduction

In the previous tutorial, we learned to create a simple ML agent to solve the Taxi game. However, we did not take into account some basic considerations when solving an ML problem:

- Was the data used to train the model useful? Are there other features that would increase the model's performance? Are all features important?
- Was the model correctly evaluated? Was the obtained performance reliable or dependent on the split performed between train and test data?
- Did a better performance in the test data mean a better performance of the agent when deployed? Could a model with worse accuracy reach the goal faster or better?

In this tutorial, you will learn to answer these questions to ensure that the learned model is robust and reliable, which is an important feature in robotics. We will continue using Decision Trees with some additional features, such as Cross-validation or feature generation and selection.

2 Instructions

You should answer each question in the given order, indicating the exercise number. **The document should not contain screenshots with code**. We strongly recommend you make tables when comparing different algorithm parameters. When training the algorithms, pay attention to:

- Use function *train_test_split* from scikit-learn to divide the data between train and test data (80% for training, 20% for test).

- Use function *cross_val_score* to perform the cross validation process[1].
- Refer to this link to manage the tree-related questions[2].

2.1 New material

Download the folder from Aula Global called *maps*. This folder will contain the new map configurations for this tutorial. Place the folder in the folder where you develop the code for the tutorials, and consider it when loading the maps in your code.

3 Exercises

Exercise 1: Feature generation

1. Choose any of the new map configurations and execute the code with the model learned in the previous tutorial. Is the taxi capable of picking up the passenger and dropping him in the desired location? Why or why not?
2. Think about some features that could make the state representation independent of the map configuration used. Explain which features are selected and how you would obtain them. Afterward, implement the necessary code to obtain all those features in each timestep.
3. Divide the 10 maps into training and test maps (with a ratio of 80% and 20% respectively, that is 8 for training and 2 for test). Collect data for each map as done in Tutorial 1. Try to keep the mentioned ratios also in the collected data (Keep in mind: You will need more data than the one collected for Tutorial 1). Explain the collection process.
4. After generating all data, it is time to select which data is better. We will use feature selection for this task[3]. Select two methods for feature selection and obtain the selected features. Which features are kept? Why do you think the eliminated features were not chosen? Keep the 4 datasets (the original one, the one with additional features, and the two selection methods).

Exercise 2: Decision Tree with Cross Validation

Up to this point, all parameters have been chosen based on a train-test split, but this is not the common method in machine learning. The split may create a bias in data distribution, causing problems with the robustness and reliability of the models trained. To solve this problem, we normally use cross-validation, which divides the training data into several chunks. Then, all buckets except one are used to train, and the unused one is used to validate the process. This is done iteratively with all the chunks.

1. Select a new set of parameters considering the same attributes as in the previous tutorial, and perform a training process with the addition of cross-validation. Take into account that now you will have an additional parameter: the dataset used to train the models. How many actions were correctly predicted? How many were incorrect? What is the precision, recall, and accuracy of the model created? Use a table to compare these results.
2. Select the three best models for each dataset type and perform the prediction of the test data. Which model is better? Compare the models in a table.
3. Select the three best models from the previous point and retrain the models with all the training data with no cross-validation (as in the last tutorial). Predict the test data again. Do the results differ from the models with cross-validation? Why or why not?

[1]https://scikit-learn.org/stable/modules/cross_validation.html
[2]https://scikit-learn.org/stable/auto_examples/tree/plot_unveil_tree_structure.html#sphx-glr-auto-examples-tree-plot-unveil-tree-structure-py
[3]https://scikit-learn.org/stable/modules/feature_selection.html

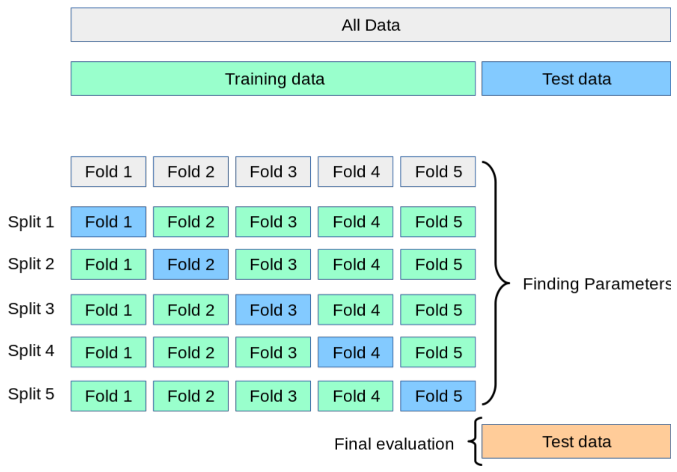

Figure 1: Cross validation process

Exercise 3: Deploy the agent

1. After selecting the three best models to solve the problem, it is time to deploy the agent. Modify the game loop for each of the three best models and compare their performance in the training maps. Are these models capable of reaching the goal? If not, why not? Does the best model (better test metrics) solve the problem the fastest? Why or why not? How many maps are solved?

2. Use these models in the test maps. Do your features make the problem general enough to solve these new configurations? Why or why not?

4 Files to Submit

All the lab assignments **must** be done in groups of 2 people. You must submit a .zip file containing the required material through Aula Global before the following deadline: **Sunday, 26th October 2025 at 23:55**. The name of the zip file must contain the last 6 digits of both students' NIA, `i.e., tutorial3-123456-234567.zip`

The zip file must contain the following files:

1. A **PDF** document with (6 page limit without cover page):
   - Cover page with the names and NIAs of both students.
   - Answers to all the questions appearing in the Exercises section.
   - Description of the features generated and how they are obtained.
   - Explanation of the decisions made to select the best models.
   - Conclusions.

2. Include in a folder called "material" the following files:
   - The files used to implement all functionalities of the tutorial,
   - The data generated for the tutorial.

Please, **be very careful and respect the submission rules**.