LELEC2870: Practical Sessions

Notations and vocabulary

- a scalar: A number represented with an upper-case letter when defining bounds on a series, or a lower-case one otherwise e.g. x in $1..X$
- a vector: A 1-dimensional array that will always be written with a **bold lower-case letter**
- a matrix: A 2-dimensional array that will always be written with a **bold upper-case letter**
- a tensor: A N-dimensional array that will always be written with a **bold upper-case letter**. While it uses the same notation as the matrix, it should always be clear which one is meant.
- an iteration: A time-measure for *iterative* algorithms. It denotes a pass over a part of the training dataset.
- an epoch: A time-measure for *iterative* algorithms. It denotes *one* pass over the complete training dataset. An epoch is composed of at least 1 iteration, but most of the time of multiple iterations especially when large datasets are involved.

Session 2

Objectives

Last session, you implemented linear regression between the target values and a few features of a dataset. This second session will focus on the behaviour one should adopt when confronted with a new dataset with a larger number of features. Indeed, preprocessing must be applied to the data received to ensure that machine learning models can be trained on this data with the available computational resources and that these models produce the best performance. Vous allez d'abord vous assurer que les données puisent être ingérées par un modèle de régression linéaire et ensuite appliquer différentes méthodes de Feature Selection apprises au cours afin d'entrainer des régresseurs toujours plus performants. We provide you a dataset for the topic covered in this session. This dataset can be found on the Moodle page of this course.

Preprocessing

Before plunging into the training of models with new data, one must think about the following themes:

1. **Sanity of data**: Are all the values in the dataset in a format that can be interpreted by the model we want to train?
2. **Redundancy of data**: do all features bring new information of is there repetition between features?
3. **Relevance of data** : Does a feature bring information about the target?
4. **Balance of influences**: Will the orders of magnitude of the numerical values of the features influence the training of my models?

The initial data may contain flaws with respect to one or more of these themes. Careful inspection of the data is therefore necessary in order to ensure that the models that will be trained do not suffer from biases that can be avoided.

Feature Selection

Filter methods When using filter methods, the best features are selected according to their respective score in some statistical test(s) e.g. Pearson's Correlation, Anova, χ^2... Those methods are thus independent from machine learning algorithms.

Wrapper method Wrapper methods for feature selection are iterative algorithms that either uses an increasing (`Forward`) or decreasing (`Backward`) number of features. For each set of selected features a model is trained the performances of those models and compared to each other. The set that generated the model with the best performances will then be the starting point of the next iteration. The forward and backward algorithms are depicted below.

- Forward selection

    1. Choose a significance level (SL).
    2. Fit regression models considering one feature at a time. Select the feature with the lowest p-value.
    3. Fit all possible models with one extra feature added to the previously selected feature(s).
    4. Again, select the feature with a minimum p-value. if $p\_value < SL$ then go to Step 3, otherwise terminate the process.

- Backward elimination

    1. Choose a significance level (SL).
    2. Fit a model with all the features.
    3. Consider the feature with the highest p-value. If the $p\_value > SL$ then go to Step 4, otherwise terminate the process.
    4. Remove the feature being considered.
    5. Fit a model with the remaining features. Resume to Step 3.

1 Preprocessing

1.1 Sanity verification

1.1.1 Missing values

It is quite common when facing a new dataset to notice missing values (`NaN`) that will prevent the proper training of the machine learning models. A common solution to this problem is simply to drop the data samples containing `NaN`s. Your first task in this session is to **remove those entries** from the dataset.

1.1.2 Data format

Some features are encoded in a format that can not be interpreted mathematically e.g. for categorical features. **Handle this problem** by converting such features in the dataset into numerical ones.

1.1.3 Outliers

Some of the samples in the dataset may contain values of features that are not representative of the general distribution of the other samples *i.e.* **outliers**. **Remove those samples** from the dataset.

1.2 Redundancy verification

Observe the correlation matrix between features. Do you find redundant features ? Can you explain how such features affect a model such as a `Linear regressor` or a `KNN regressor`? In both case, Is it important to keep or drop redundant features?

1.3 Relevance verification

Observe the correlation matrix between features and the target. Which features do you think are more likely to be selected when performing Feature Selection for a linear model? Do you think your answer will also hold for other regression models? Why?

1.4 Balance of Influences

Observe the modified dataset once again: are all features of the same order of magnitude? Does this variation affect a linear regressor? And a KNN regressor? How would you solve this?

2 Feature Selection

2.1 Dataset Splitting and baseline performance

Using the `sklearn.model_selection` module, **split your dataset** into train set and test set such that 30 percents of your data is used for testing.

Paste your implementation of the `compute_rmse` method you wrote last session in the jupyter notebook.

Implement a method that trains a linear regressor on the given training set, using the specified features, and predicts the target values of a test set. This model will be considered as the **baseline model** from now on. What is the value of its RMSE?

2.2 Filtering method

Select relevant features using the filtering method. To do so, define a correlation threshold between the features and the target and filter out features with a correlation lower than this threshold. Then you are asked to train a linear model on your training set using the selected features (**use the method you implemented in the previous section**.

Compare the RMSE of this new model with the baseline? Did the performances improve?

2.3 Wrapper method

For this last part, you are expected to use the implementation of `Forward` and `Backward` Wrapper methods provided in the `sklearn.feature_selection` module. Then, for each selection method, **train** a linear model and **compare** their performances (RMSE) with respect to the baseline model.