🏁 Wrap-up quiz 7

This quiz requires some programming to be answered.

Open the dataset `bike_rides.csv` with the following commands:

```
import pandas as pd

cycling = pd.read_csv("../datasets/bike_rides.csv", index_col=0,
                      parse_dates=True)
cycling.index.name = ""
target_name = "power"
data, target = cycling.drop(columns=target_name), cycling[target_name]
data
```

A detailed description of this dataset is given in the appendix. As a reminder, the problem we are trying to solve with this dataset is to use measurements from cheap sensors (GPS, heart-rate monitor, etc.) in order to predict a cyclist power. Power can indeed be recorded via a cycling power meter device, but this device is rather expensive.

Instead of using blindly machine learning, we will first introduce some flavor of classic mechanics: the Newton's second law.

$$P_{meca} = (\frac{1}{2}\rho.SC_x.V_a^2 + C_r.mg.\cos\alpha + mg.\sin\alpha + ma)V_d$$

where ρ is the air density in kg.m^{-3}, S is frontal surface of the cyclist in m^2, C_x is the drag coefficient, V_a is the air speed in m.s^{-1}, C_r is the rolling coefficient, m is the mass of the rider and bicycle in kg, g is the standard acceleration due to gravity which is equal to 9.81 m.s^{-2}, α is the slope in radian, V_d is the rider speed in m.s^{-1}, and a is the rider acceleration in m.s^{-2}.

This equation might look a bit complex at first but we can explain with words what the different terms within the parenthesis are:

- the first term is the power that a cyclist is required to produce to fight wind
- the second term is the power that a cyclist is required to produce to fight the rolling resistance created by the tires on the floor
- the third term is the power that a cyclist is required to produce to go up a hill if the slope is positive. If the slope is negative the cyclist does not need to produce any power to go forward
- the fourth and last term is the power that a cyclist requires to change his speed (i.e. acceleration).

We can simplify the model above by using the data that we have at hand. It would look like the following.

$$P_{meca} = \beta_1 V_d^3 + \beta_2 V_d + \beta_3 \sin(\alpha) V_d + \beta_4 a V_d$$

This model is closer to what we saw previously: it is a linear model trained on a non-linear feature transformation. We will build, train and evaluate such a model as part of this exercise. Thus, you need to:

- create a new data matrix containing the cube of the speed, the speed, the speed multiplied by the sine of the angle of the slope, and the speed multiplied by the acceleration. To compute the angle of the slope, you need to take the arc tangent of the slope (`alpha = np.arctan(slope)`). In addition, we can limit ourself to positive acceleration only by clipping to 0 the negative acceleration values (they would correspond to some power created by the braking that we are not modeling here).

- using the new data matrix, create a linear predictive model based on a `sklearn.preprocessing.StandardScaler` and a `sklearn.linear_model.RidgeCV`;
- use a `sklearn.model_selection.ShuffleSplit` cross-validation strategy with only 4 splits (`n_splits=4`) to evaluate the generalization performance of the model. Use the mean absolute error (MAE) as a generalization performance metric. Also, pass the parameter `return_estimator=True` and `return_train_score=True` to answer the subsequent questions. Be aware that the `ShuffleSplit` strategy is a naive strategy and we will investigate the consequence of making this choice in the subsequent questions.

Question

What is the mean value of the column containing the information of $\sin(\alpha)V_d$?

- a) about -3
- b) about -0.3
- c) about -0.03
- d) about -0.003

Select a single answer

Question

On average, the Mean Absolute Error on the test sets obtained through cross-validation is closest to:

- a) 20 Watts
- b) 50 Watts
- c) 70 Watts
- d) 90 Watts

Select a single answer

Hint: pass `scoring="neg_mean_absolute_error"` to the `cross_validate` function to compute the (negative of) the requested metric. Hint: it is possible to replace the negative acceleration values by 0 using `data["acceleration"].clip(lower=0)`

Question

Given the model $P_{meca} = \beta_1 V_d^3 + \beta_2 V_d + \beta_3 \sin(\alpha) V_d + \beta_4 a V_d$ that you created, inspect the weights of the linear models fitted during cross-validation and select the correct statements:

- a) $\beta_1 < \beta_2 < \beta_3$
- b) $\beta_3 < \beta_1 < \beta_2$
- c) $\beta_2 < \beta_3 < \beta_1$
- d) $\beta_1 < 0$
- e) $\beta_2 < 0$
- f) $\beta_3 < 0$
- g) $\beta_4 < 0$
- h) All βs are > 0

Select all answers that apply

Now, we will create a predictive model that uses all `data`, including available sensor measurements such as cadence (the speed at which a cyclist turns pedals measured in rotation per minute) and heart-rate (the number of beat per minute of the heart of the cyclist while exercising). Also, we will use a non-linear regressor, a

[sklearn.ensemble.HistGradientBoostingRegressor](). Fix the number of maximum iterations to 1000 (`max_iter=1_000`) and activate the early stopping (`early_stopping=True`). Repeat the previous evaluation using this regressor.

> **Question**
>
> On average, the Mean Absolute Error on the test sets obtained through cross-validation is closest to:
>
> - a) 20 Watts
> - b) 40 Watts
> - c) 60 Watts
> - d) 80 Watts
>
> *Select a single answer*

> **Question**
>
> Comparing both the linear model and the histogram gradient boosting model and taking into consideration the train and test MAE obtained via cross-validation, select the correct statements:
>
> - a) the generalization performance of the histogram gradient-boosting model is limited by its underfitting
> - b) the generalization performance of the histogram gradient-boosting model is limited by its overfitting
> - c) the generalization performance of the linear model is limited by its underfitting
> - d) the generalization performance of the linear model is limited by its overfitting
>
> *Select all answers that apply*
>
> Hint: look at the values of the `train_score` and the `test_score` collected in the dictionaries returned by the `cross_validate` function.

In the previous cross-validation, we made the choice of using a `ShuffleSplit` cross-validation strategy. It means that randomly selected samples were selected as testing set ignoring any time dependency between the lines of the dataframe.

We would like to have a cross-validation strategy that takes into account the groups defined by each individual date. Each group corresponds to a bike ride.

> **Question**
>
> How many bike rides are stored in the dataframe `data`? Do not hesitate to look at the hints.
>
> - a) 2
> - b) 3
> - c) 4
> - d) 5
>
> *Select a single answer*
>
> Hint: You can check the unique day in the `DatetimeIndex` (the index of the dataframe `data`). Indeed, we assume that on a given day the rider went cycling at most once per day. Hint: You can access to the date and time of a `DatetimeIndex` using `df.index.date` and `df.index.time`, respectively.

We would like to have a cross-validation strategy that evaluates the capacity of our model to predict on a completely new bike ride: the samples in the validation set should only come from rides not present in the training set. Therefore, we can use a `LeaveOneGroupOut` strategy: at each iteration of the cross-validation, we will keep a bike ride for the evaluation and use all other bike rides to train our model.

Thus, you concretely need to:

- create a variable called `group` that is a 1D numpy array containing the index of each ride present in the dataframe. Therefore, the length of `group` will be equal to the number of samples in `data`. If we had 2 bike rides, we would expect the indices 0 and 1 in `group` to differentiate the bike ride. You can use **`pd.factorize`** to encode any Python types into integer indices.
- create a cross-validation object named `cv` using the **`sklearn.model_selection.LeaveOneGroupOut`** strategy.
- evaluate both the linear and histogram gradient boosting models with this strategy.

Question

Using the previous evaluations (with the `LeaveOneGroupOut` strategy) and looking at the train and test errors for both models, select the correct statements:

- a) the generalization performance of the gradient-boosting model is limited by its underfitting
- b) the generalization performance of the gradient-boosting model is limited by its overfitting
- c) the generalization performance of the linear model is limited by its underfitting
- d) the generalization performance of the linear model is limited by its overfitting

Select all answers that apply

Question

In this case we cannot compare cross-validation scores fold-to-fold as the folds are not aligned (they are not generated by the exact same strategy). Instead, compare the mean of the cross-validation test errors in the evaluations of the **linear model** to select the correct statement.

When using the `ShuffleSplit` strategy, the mean test error:

- a) is greater than the `LeaveOneGroupOut` mean test error by more than 3 Watts, i.e. `ShuffleSplit` is giving over-pessimistic results
- b) differs from the `LeaveOneGroupOut` mean test error by less than 3 Watts, i.e. both cross-validation strategies are equivalent
- c) is lower than the `LeaveOneGroupOut` mean test error by more than 3 Watts, i.e. `ShuffleSplit` is giving over-optimistic results

Select a single answer

Question

Compare the mean of the cross-validation test errors in the evaluations of the **gradient-boosting model** to select the correct statement.

When using the `ShuffleSplit` strategy, the mean test error:

- a) is greater than the `LeaveOneGroupOut` mean test error by more than 3 Watts, i.e. `ShuffleSplit` is giving over-pessimistic results
- b) differs from the `LeaveOneGroupOut` mean test error by less than 3 Watts, i.e. both cross-validation strategies are equivalent
- c) is lower than the `LeaveOneGroupOut` mean test error by more than 3 Watts, i.e. `ShuffleSplit` is giving over-optimistic results

Select a single answer

Question

Compare more precisely the errors estimated through cross-validation and select the correct statement:

- a) in general, the standard deviation of the train and test errors increased using the `LeaveOneGroupOut` cross-validation
- b) in general, the standard deviation of the train and test errors decreased using the `LeaveOneGroupOut` cross-validation

Select a single answer

Now, we will go more into details by picking a single ride for the testing and analyse the predictions of the models for this test ride. To do so, we can reuse the `LeaveOneGroupOut` cross-validation object in the following manner:

```
cv = LeaveOneGroupOut()
train_indices, test_indices = list(cv.split(data, target, groups=groups))[0]

data_linear_model_train = data_linear_model.iloc[train_indices]
data_linear_model_test = data_linear_model.iloc[test_indices]

data_train = data.iloc[train_indices]
data_test = data.iloc[test_indices]

target_train = target.iloc[train_indices]
target_test = target.iloc[test_indices]
```

Now, fit both the linear model and the histogram gradient boosting regressor models on the training data and collect the predictions on the testing data. Make a scatter plot where on the x-axis, you will plot the measured powers (true target) and on the y-axis, you will plot the predicted powers (predicted target). Do two separated plots for each model.

Question

By analysing the plots, select the correct statements:

- a) the linear regressor tends to under-predict samples with high power
- b) the linear regressor tends to over-predict samples with high power
- c) the linear regressor makes catastrophic predictions for samples with power close to zero
- d) the histogram gradient boosting regressor tends to under-predict samples with high power
- e) the histogram gradient boosting regressor tends to over-predict samples with high power
- f) the histogram gradient boosting makes catastrophic predictions for samples with power close to zero

Select all answers that apply

Now select a portion of the testing data using the following code:

```
time_slice = slice("2020-08-18 17:00:00", "2020-08-18 17:05:00")

data_test_linear_model_subset = data_linear_model_test[time_slice]
data_test_subset = data_test[time_slice]
target_test_subset = target_test[time_slice]
```

It allows to select data from 5.00 pm until 5.05 pm. Used the previous fitted models (linear and gradient-boosting regressor) to predict on this portion of the test data. Draw on the same plot the true targets and the predictions of each model.

Question

By using the previous plot, select the correct statements:

- a) the linear model is more accurate than the histogram gradient boosting regressor
- b) the histogram gradient boosting regressor is more accurate than the linear model

Select a single answer