MLJ : Notes

Dainish Jabeen

January 12, 2023

1 General

MLJ is a machine learning toolbox for Julia, that wraps a large number of models and provides great tools such as resampling and evaluation [1] [2].

Models are structs storing hyperparameters.

Listing 1: Basics

```
#Split data randomly with seed (rng)
y, X = unpack(df,==(:col);rng=123);

df |> pretty #Output pretty version

#Load model
VAR = @load model pkg=""

var = VAR() #Default parameters

#evaluate model with cross validation
 via error measures

evaluate(var,X,y,resampling=CV(shuffle=true),measures=[rms])
```

1.1 Types

Each model has an expected variable type needed to train it:

target_scitype(model) provides type needed.

Listing 2: Change type

```
#Change type of var

y = coerce(df, :col=type,...)
```

2 Machines

Used to store training outcomes.

Listing 3: Machines

```

        mach = machine(model,X,y)

        #70:30 partition giving an index vector
        train, test = partition(eachindex(y),0.7)

        fit!(mach,train)
        yhat = predict(mach,X[text,:]);

        #error rate
        misclassification_rate(yhat, test)
```

Can also evaulate! machines.

2.1 Prediction

To predict with a given probability for a class use **broadcast**.

For a matrix of all classes: **pdf**.

For an output of a class: **predict_mode**.

2.2 Inspection

Two methods: fitted_params and report.

Fitted params: The learned parameters for a machine.

report: More detailed stats on machine.

3 Hyperparameter tuning

Check notebook Resampling(Julia-MLJ) for example.

- Create ensemble model
- Create ranges for parameters
- wrap in tuned model
- create mach, fit data
- report on best model
- mach will become the best model

learning_curve() gives a performance line for a tuning parameter.

4 Pipeline

Linear set of models chained together, that can be evaluated and used as a single model.

For example: Change var type |> tune parameter |> model |> train.

5 Resampling

Listing 4: Resampling

```
1
2         #Split data into sections of chosen percentage,
3         #       can also shuffle
4         holdout = Holdout(; fraction_train=0.7,
5                     shuffle=nothing,
6                     rng=nothing)
7
8         # Cross validate over number of folds
9         cv = CV(; nfolds=6,  shuffle=nothing, rng=nothing)
10
11        # For classification problems aims to retain the connection
12        #       between the predictors in train and test with the
13        #       response class level.
14
15        stratified_cv = StratifiedCV(; nfolds=6,
16                              shuffle=false,
17                              rng=Random.GLOBAL_RNG)
18
19
20        #CV for when observations are chronological and not expected
21        #       to be independent.
22
23        tscv = TimeSeriesCV(; nfolds=4)
```

References

[1] A. D. Blaom, F. Kiraly, T. Lienart, Y. Simillides, D. Arenas, and S. J. Vollmer, "MLJ: A julia package for composable machine learning," *Journal of Open Source Software*, vol. 5, no. 55, p. 2704, 2020. DOI: 10.21105/joss.02704. [Online]. Available: https://doi.org/10.21105/joss.02704.

[2] A. D. Blaom and S. J. Vollmer, *Flexible model composition in machine learning and its implementation in MLJ*, 2020. arXiv: 2012.15505 [cs.LG].