ML Part 2 tutorial

Dimensionality reduction & cross-validation

Jérôme Dockès & Nikhil Bhagwat

QLS course 2021-07-30

Problem setting

$$Y = f(X) + E \tag{1}$$

- $Y \in \mathbb{R}$: output (a.k.a. target, dependent variable) to predict
- $X \in \mathbb{R}^p$: features (a.k.a. inputs, regressors, descriptors, independent variables)
- $E \in \mathbb{R}$: unmodelled noise
- f: the function we try to approximate

Parameter estimation a.k.a. model fitting

Minimize a sum of:

- the empirical risk: error on training data
- a regularization term

Example: logistic regression (used today)

$$\underset{\beta, \beta_0}{\operatorname{argmin}} \frac{1}{2}\|\beta\|_2^2 + C \sum_{i=1}^{n} \log(\exp(-y_i\,(X_i^T\,\beta + \beta_0)) + 1) \qquad (2)$$

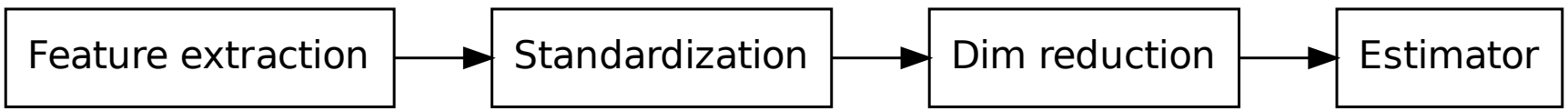
- β, β_0: parameters to be *estimated*
- C: hyperparameter, *chosen* prior to learning (controls amount of regularization)

scikit-learn "estimator API": `fit; predict`

```
estimator = LogisticRegression(C=1)
estimator.fit(X_train, y_train)
prediction = estimator.predict(X_test)
```

Dataset transformations

Typical pipeline



Example we will use today

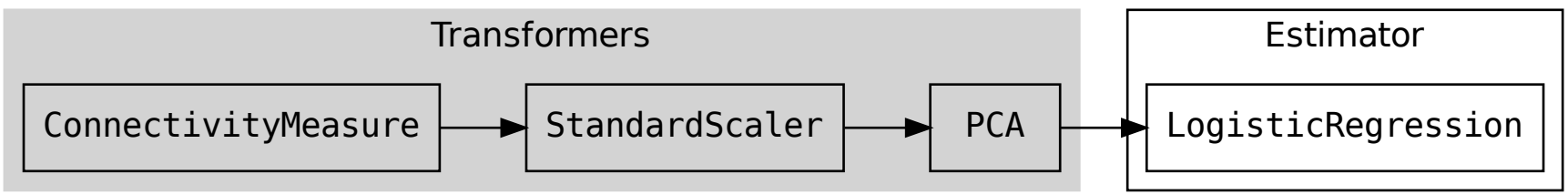
scikit-learn "transformer API": `fit; transform`

```
transformer = StandardScaler()
transformer.fit(X_train)
transformed_X = transformer.transform(X_train)
```

scikit-learn "transformer API": `fit; transform`

```
transformer = StandardScaler()
transformed_X = transformer.fit_transform(X_train)

transformed_X_test = transformer.transform(X_test)
```



Example: `preprocessing.StandardScaler`

`fit`:
Compute mean and standard deviation of each column

`transform`:
Subtract mean and divide by standard deviation

Example: `feature_selection.SelectKBest`

`fit`:

- Perform ANOVA for each column of X
- Remember the indices of the k columns with highest scores

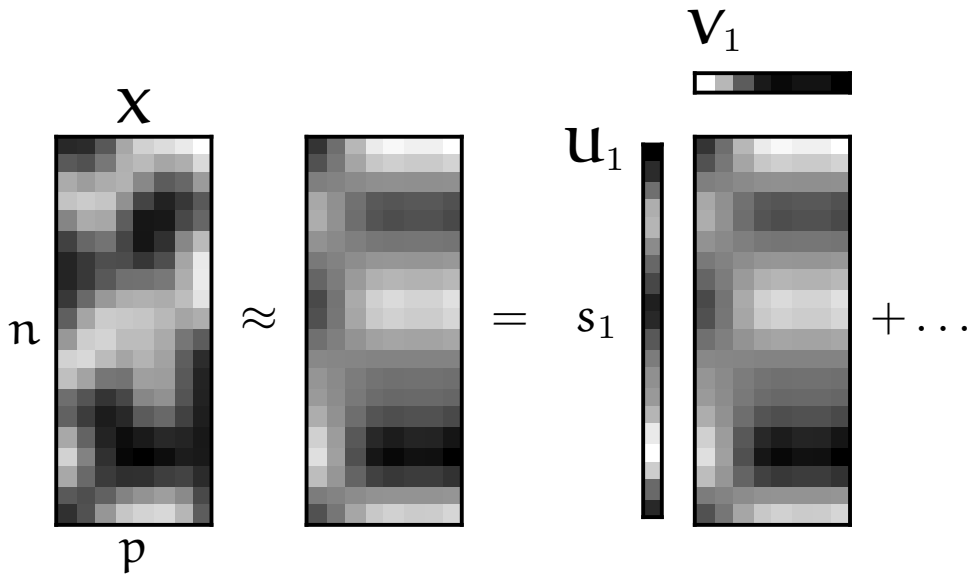
`transform`:

- Index input to keep only the k selected columns

Example: `decomposition.PCA`

`fit:`
Compute Singular Value Decomposition of X

$$X = U S V^T \tag{3}$$

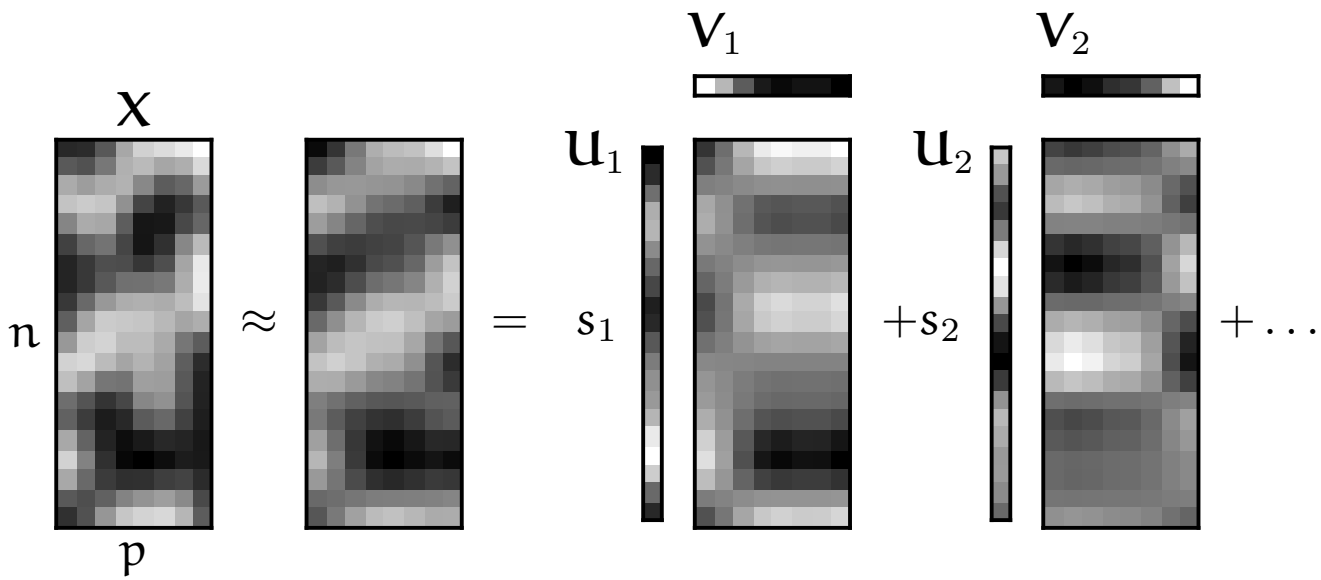


Explained variance: 0.53

Example: `decomposition.PCA`

`fit:`
Compute Singular Value Decomposition of X

$$X = U S V^T \tag{4}$$

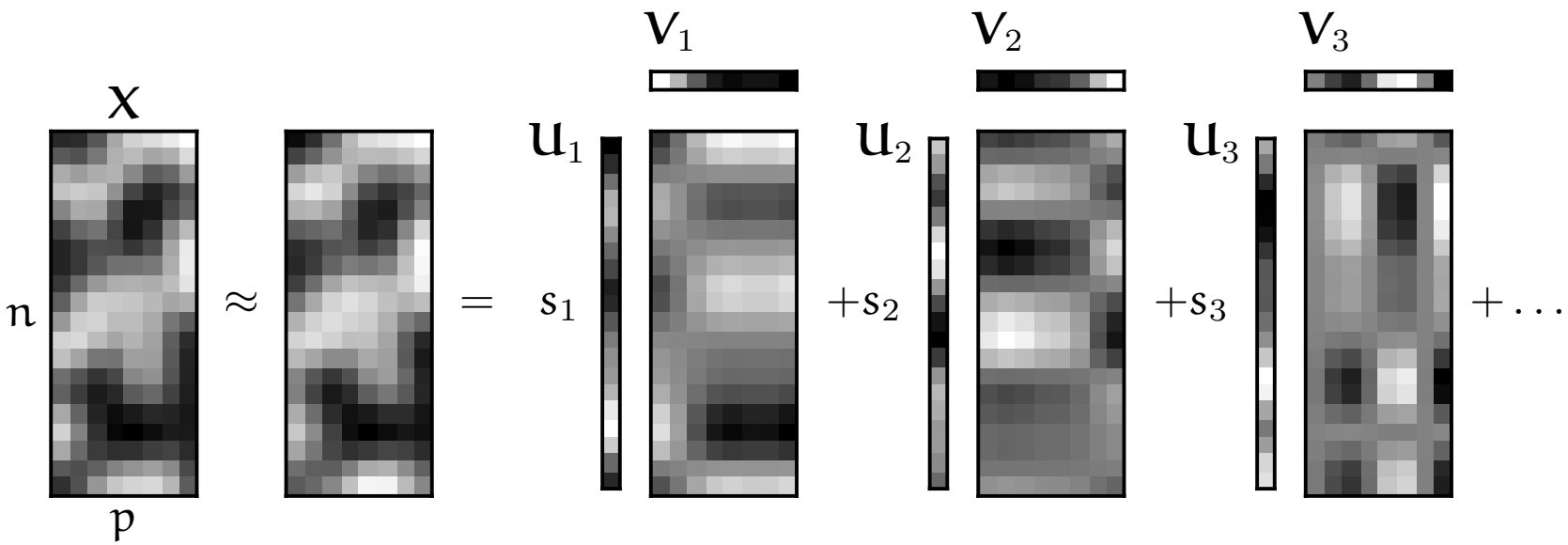


Explained variance: 0.84

Example: `decomposition.PCA`

`fit:`
Compute Singular Value Decomposition of X

$$X = U S V^T \tag{5}$$



Explained variance: 0.97

Example: `decomposition.PCA`

`fit:`
Compute Singular Value Decomposition of $\mathbf{X}$

$$\mathbf{X} = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^{\mathsf{T}}$$

store $\mathbf{V}$

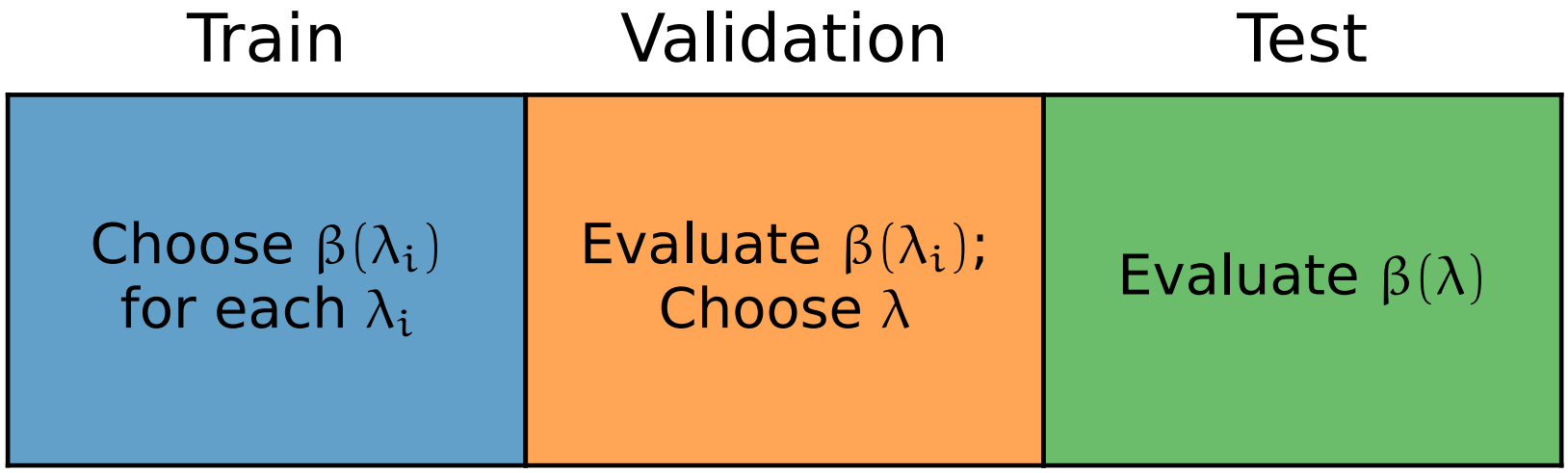
`transform:`
Compute coordinates in the basis $\mathbf{V}$: simply multiply by $\mathbf{V}^{\mathsf{T}}$

Chaining transformations

Use `sklearn.pipeline.Pipeline`

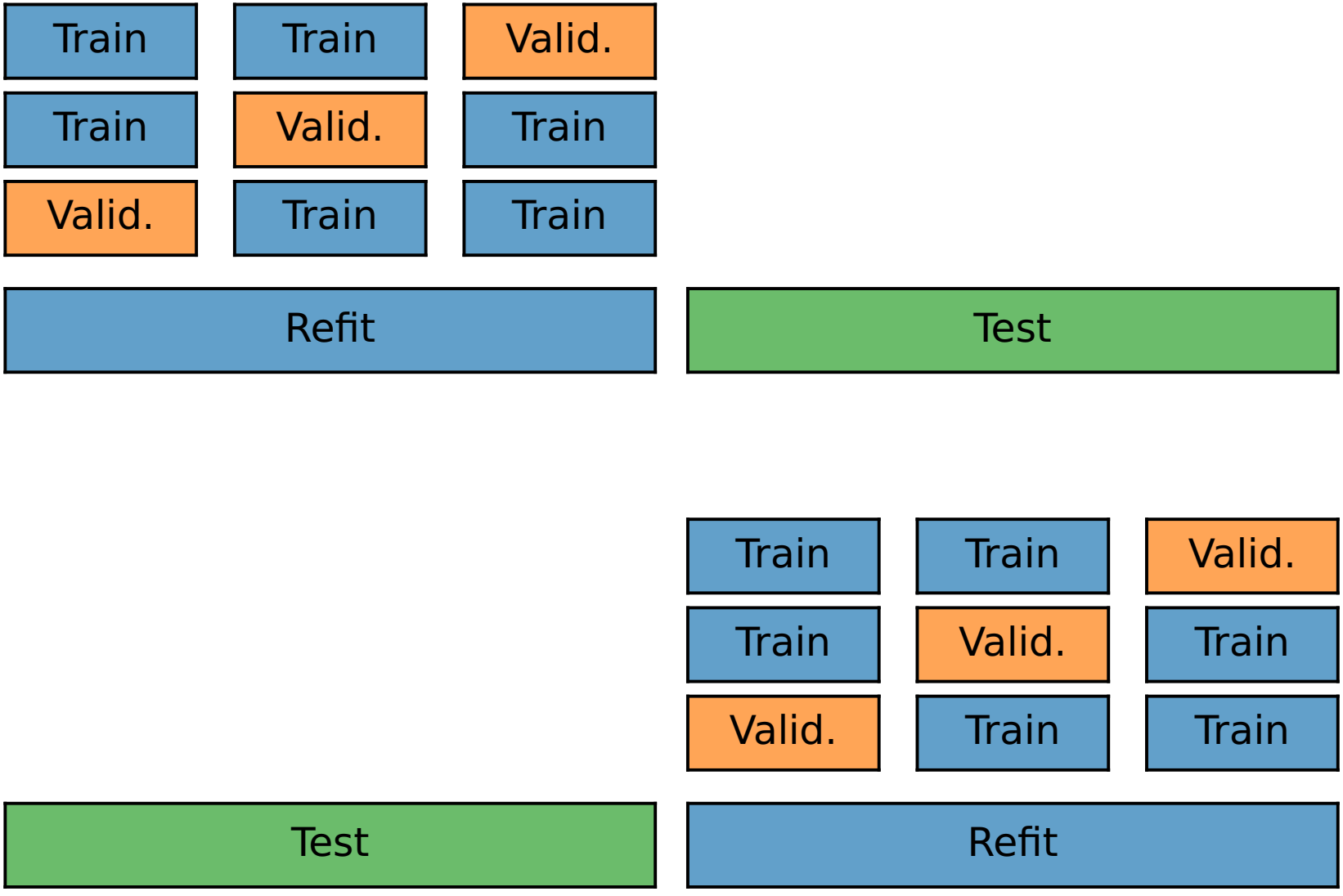
```
pipe = make_pipeline(feat_extraction,
                     standardization,
                     dim_reduction,
                     estimator)
pipe.fit(X, y)
```

Parameters, hyperparameters, evaluation



The whole pipeline must be fitted on "Train" only (including the transformers)!

Nested cross-validation



see `sklearn.model_selection.GridSearchCV` and `sklearn.model_selection.cross_validate`

Let’s start the exercises

https://github.com/neurodatascience/course-materials-2021/tree/master/lectures/30-July/12-intro-to-machine-learning-part-2/in-class-tutorials