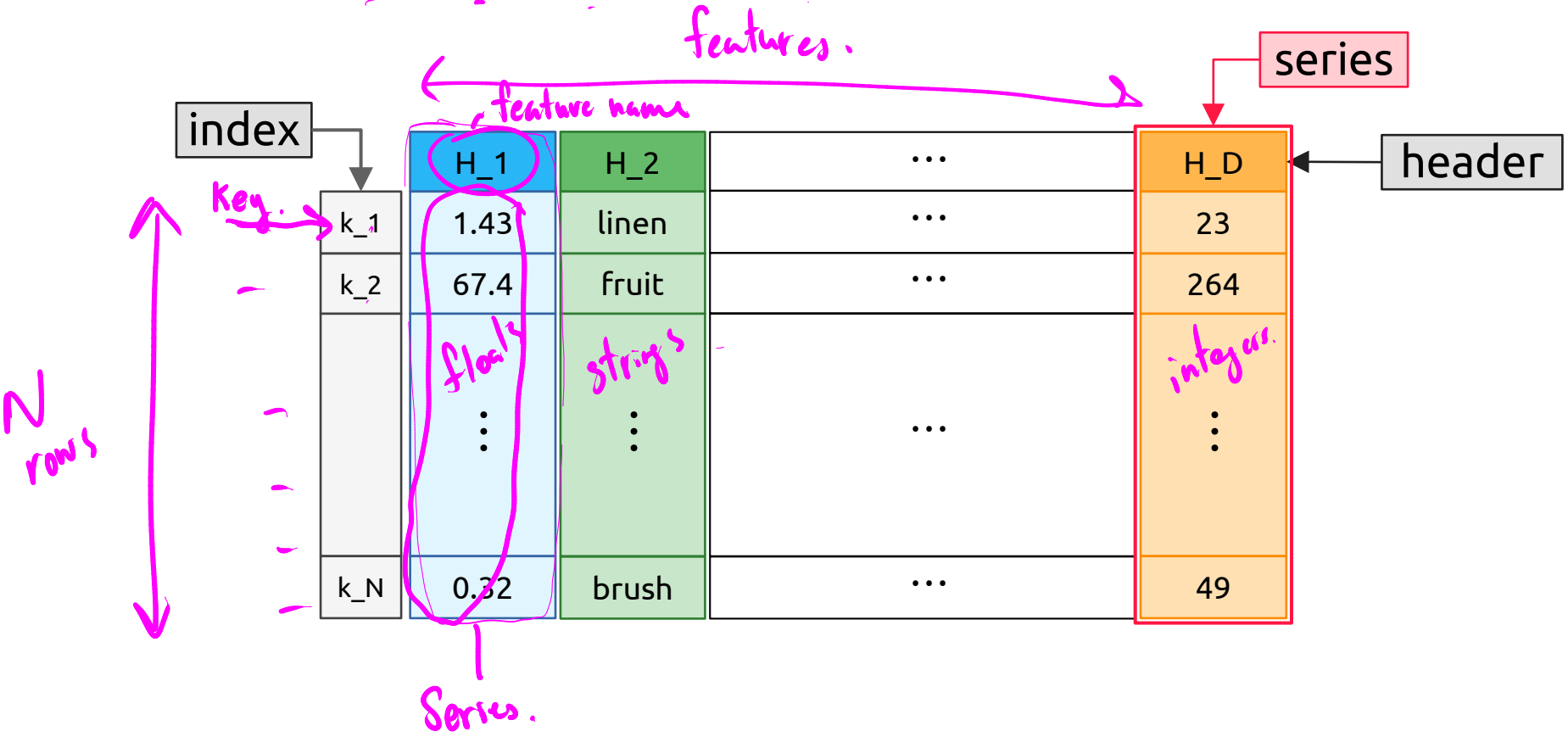
pandas
&
scikit-learn

pandas: A package for tabular data

- **DataFrame:** (index,{header:series})



Querying a DataFrame

- **Column selectors:** `[]`
  - `X["H1"]` … single column
  - `X[["H1","H2"]]` … multiple columns

- **Selecting rows by index:** `.loc[]`
  - `X.loc[k1]` … single row
  - `X.loc[[k1,k2]]` … multiple rows

- **`loc[]` also accepts a column selector**
  - `X.loc[k1,"H1"]`
  - `X.loc[k1,["H1","H2"]]`
  - `X.loc[[k1,k2],"H1"]`
  - `X.loc[[k1,k2],["H1","H2"]]`

Querying a DataFrame (cntd.)

- **Selecting rows with a conditional**
  - `X.loc[boolean_mask]`
  - `X.loc[boolean_mask,column_selector]`

… syntactic sugar: `X[<boolean mask>]`

- **Ordered rows (integer index)**
  - `X[slice]`
  - `X.loc[slice,column_selector]`

- **Ordered rows and columns:** `.iloc[]`
  - `X.iloc[row_slice]`
  - `X.iloc[row_slice,col_slice]`

Loading and saving data

Single object files

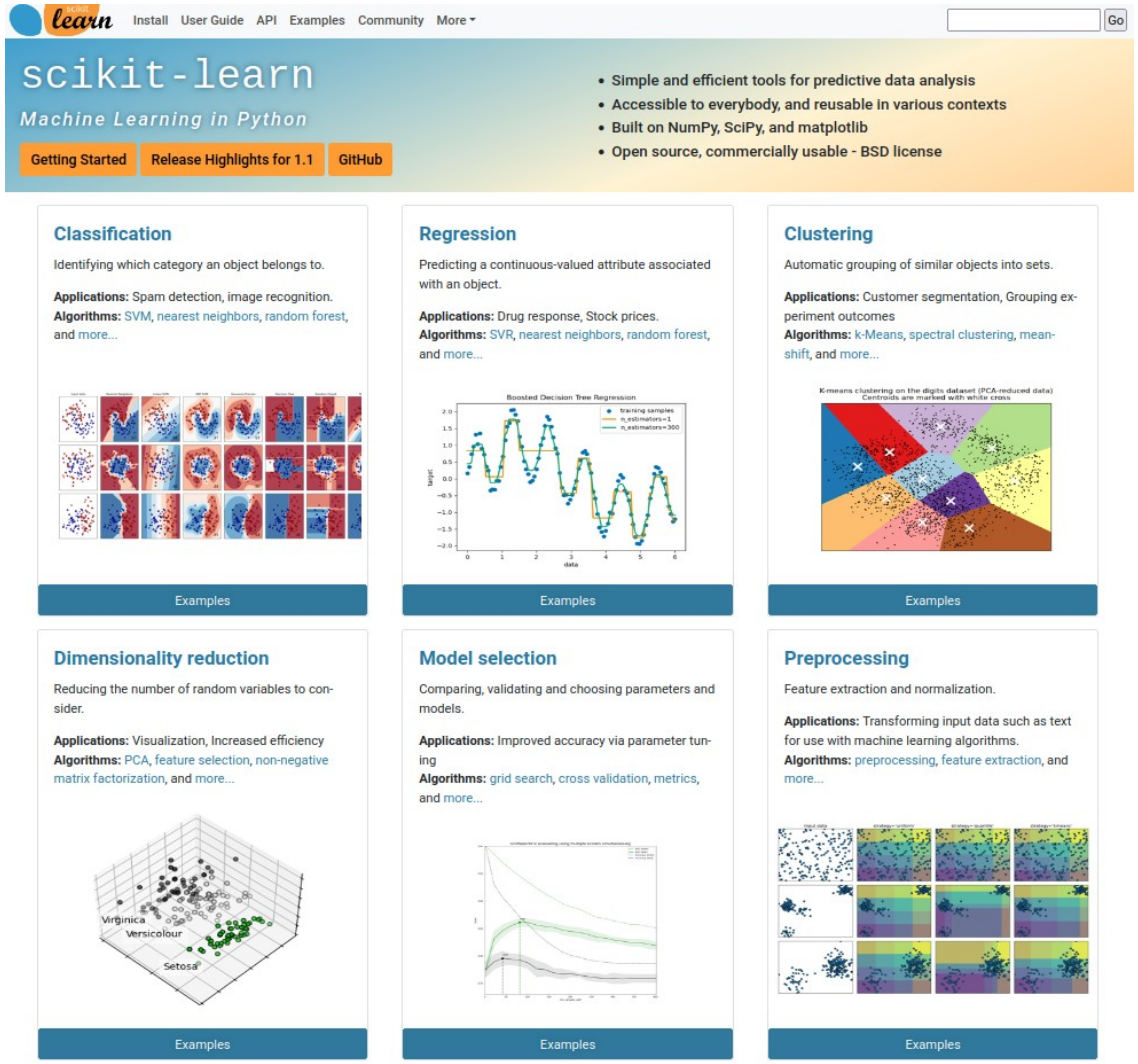
	text	pickle
numpy	`np.savetxt(filename,A)` `A = np.loadtxt(filename)`	`np.save(filename,A)` `A = np.load(filename)`
pandas	`DF.to_csv(filename)` `DF = pd.read_csv(filename)`	`DF.to_pickle(filename)` `DF = pd.read_pickle(filename)`

Multiple object files

```
import pickle
```

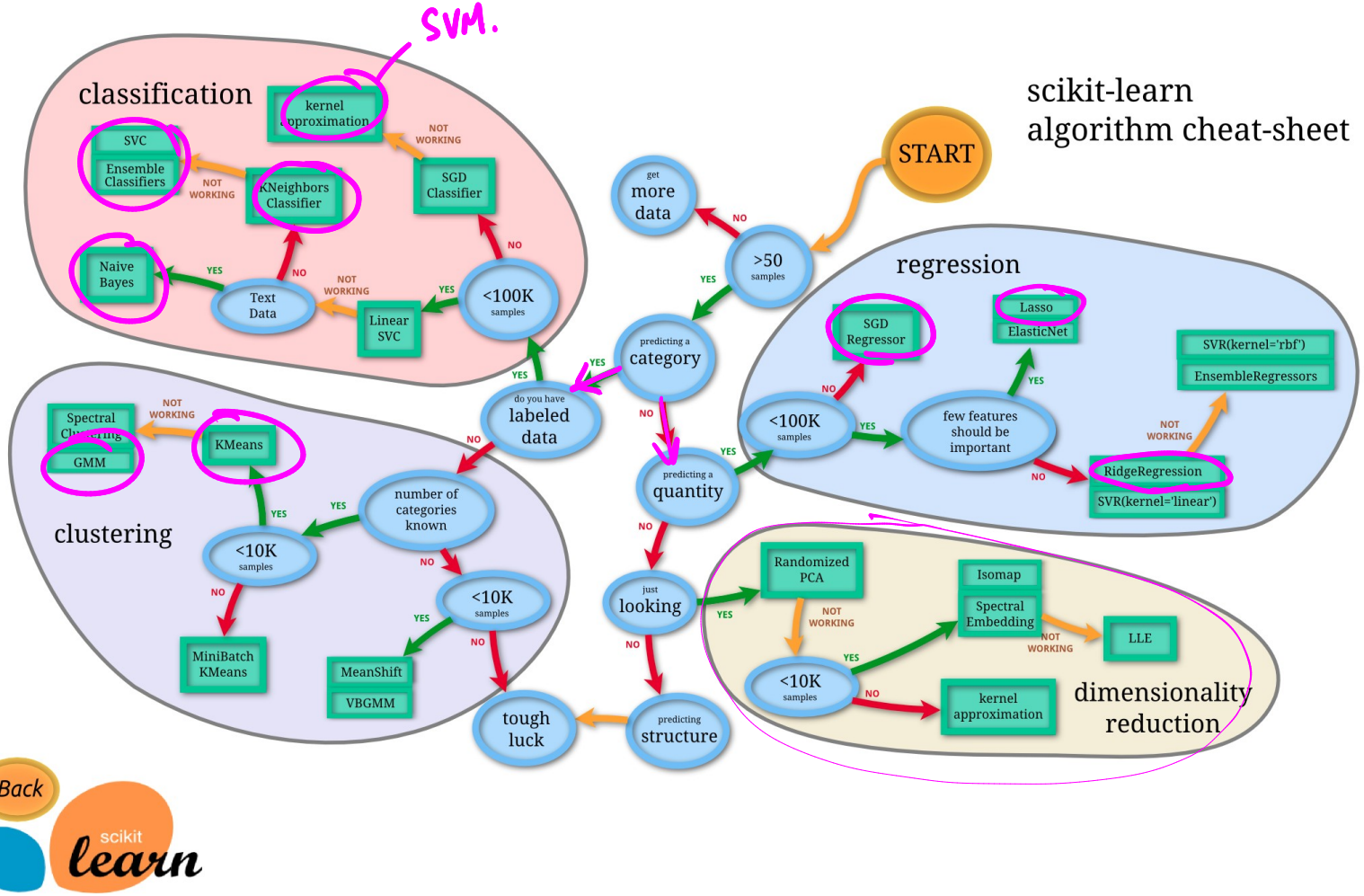
```
with open("mypickle.pkl","wb") as f:
    pickle.dump((A,D),f)
```

```
with open("mypickle.pkl", "rb") as f:
    Anew, Dnew = pickle.load(f)
```

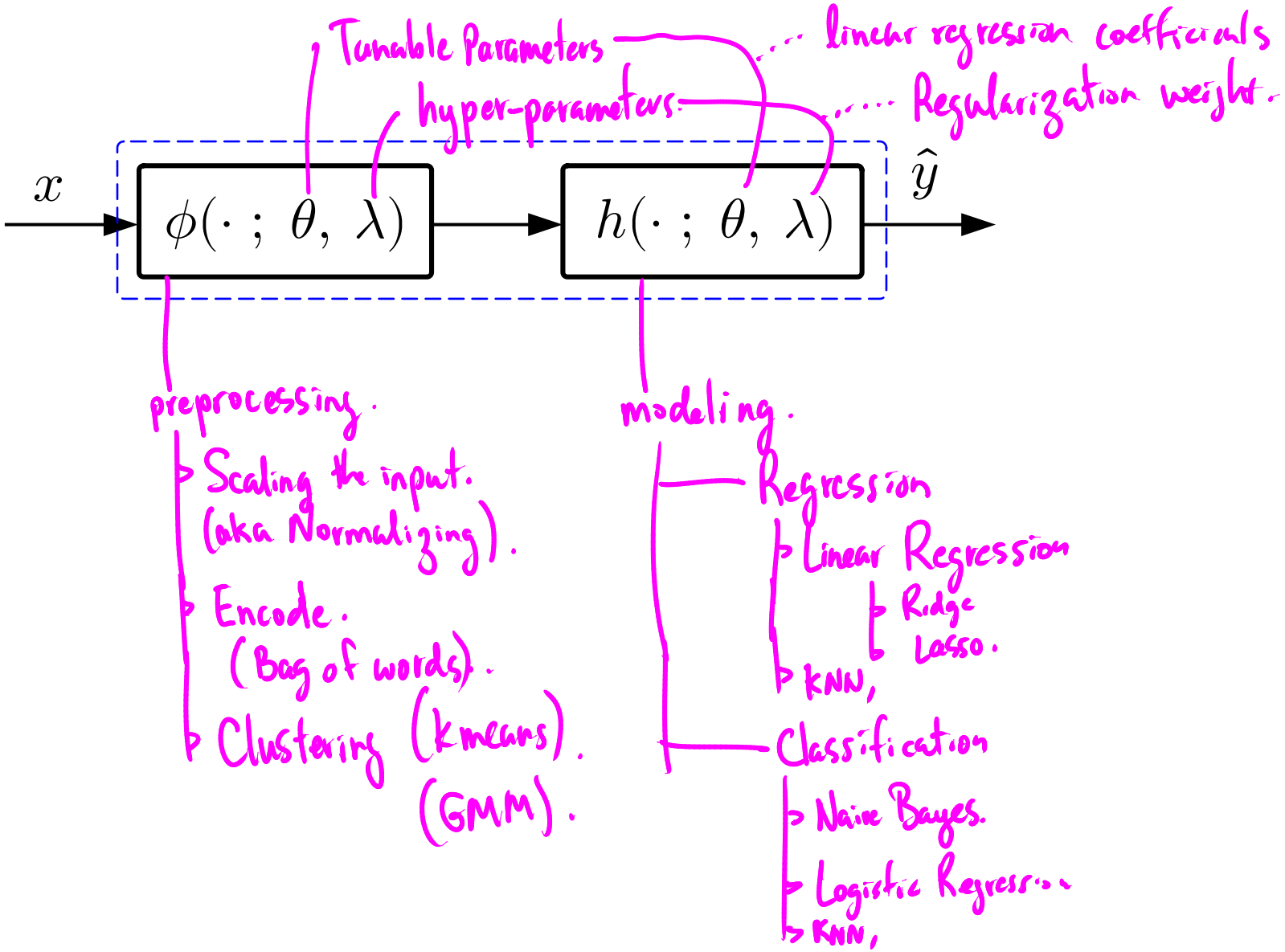


https://scikit-learn.org

- Open source
- Nice API
- Extensible
- Used in industrial R&D



https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html


Tunable Parameters
hyper-parameters
linear regression coefficients
Regularization weight.
x
$\phi(\cdot\,;\,\theta,\,\lambda)$
$h(\cdot\,;\,\theta,\,\lambda)$
$\hat{y}$
preprocessing.
Scaling the input. (aka Normalizing).
Encode. (Bag of words).
Clustering (kmeans). (GMM).
modeling.
Regression
Linear Regression
Ridge
Lasso.
KNN,
Classification
Naive Bayes.
Logistic Regression
KNN,

scikit-learn API

Estimators: .... fit : Runs the training algorithm to compute θ

Transformers

$$\Phi = \phi(\mathbf{X}\ ;\ \theta\ , \lambda)$$

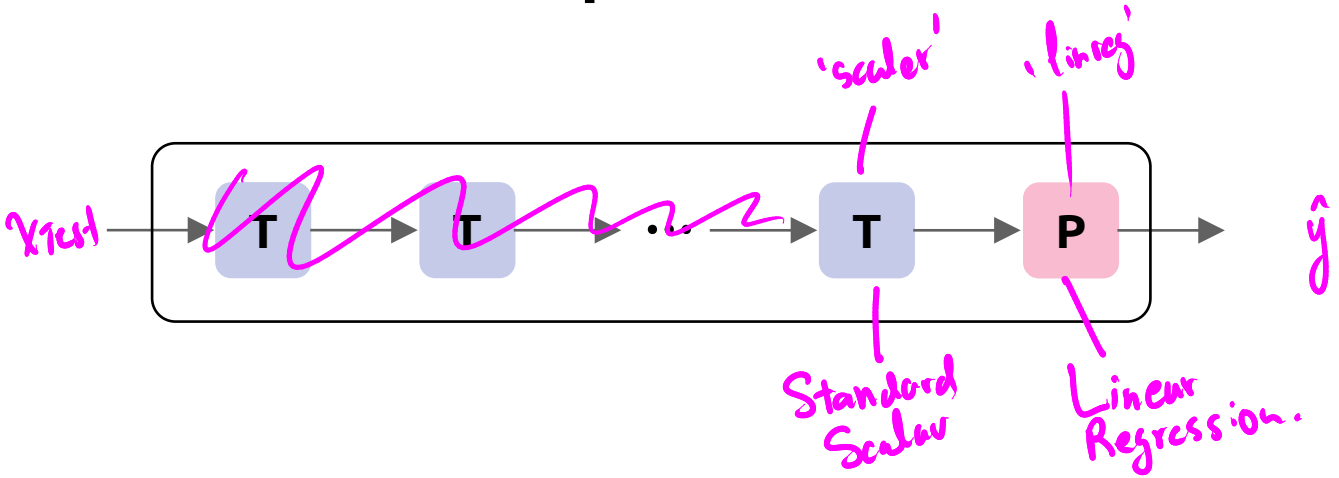
.transform(...)
.fit_transform(...)

Predictors

$$\hat{\mathbf{Y}} = h(\Phi\ ;\ \theta\ , \lambda)$$

.predict(...)
.fit_predict(...)

Pipelines



- A pipeline is an estimator of the same type as its downstream-most component.
- Strings together transformers and predictors.
- Clearly separates fitting and predicting functionality.