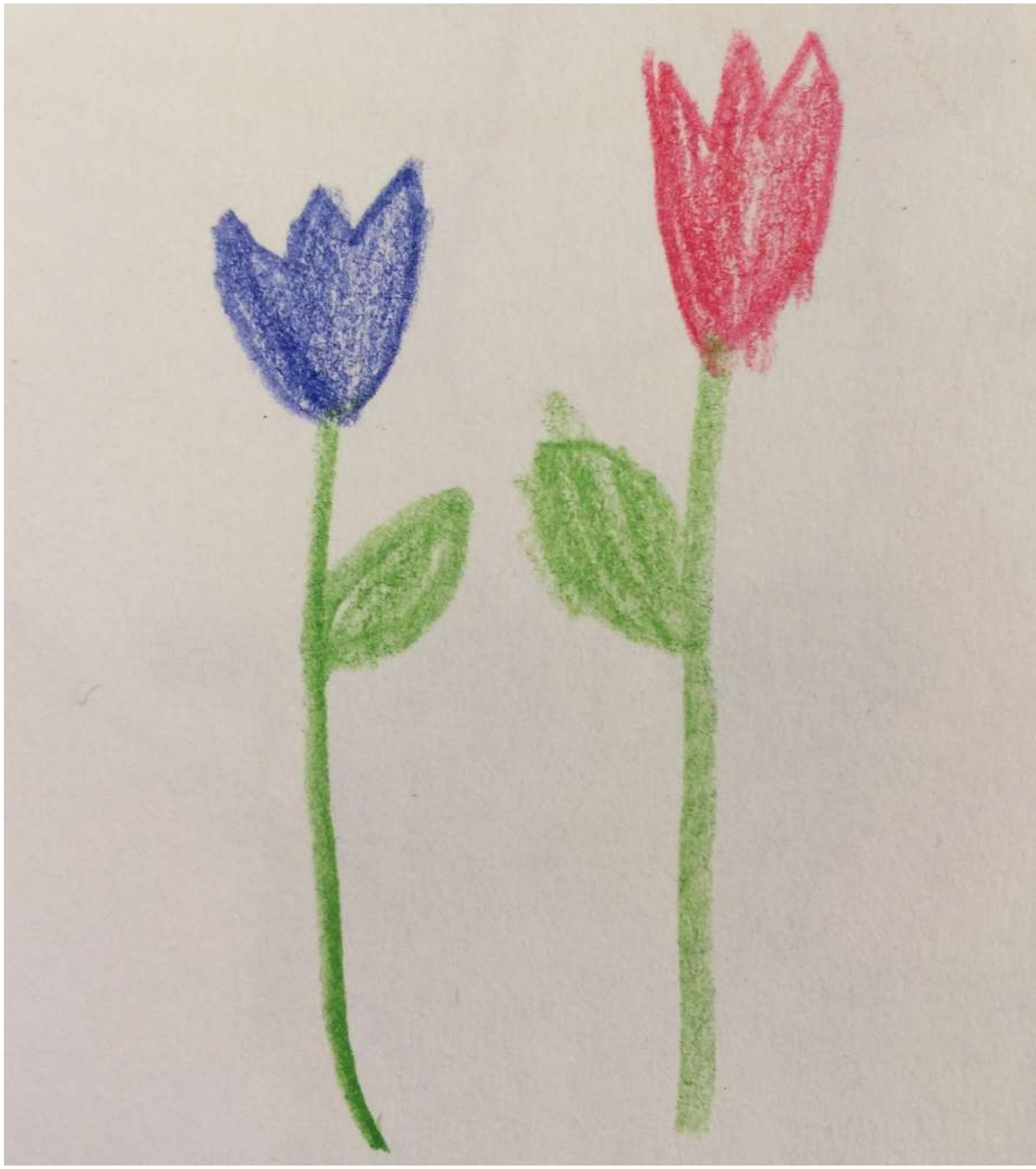
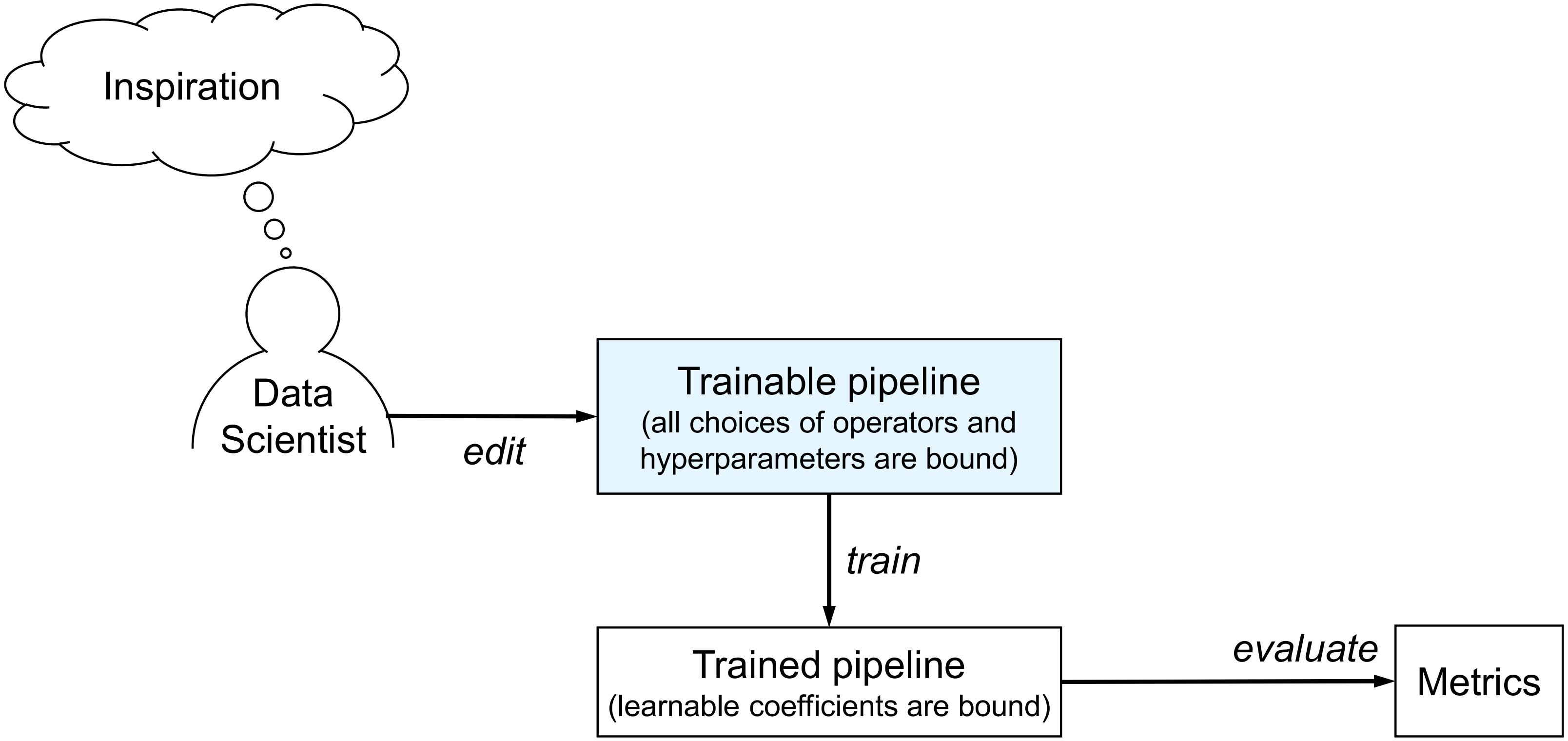
LALE: Library for Semi-Automated Data Science

-

April 2021 user study

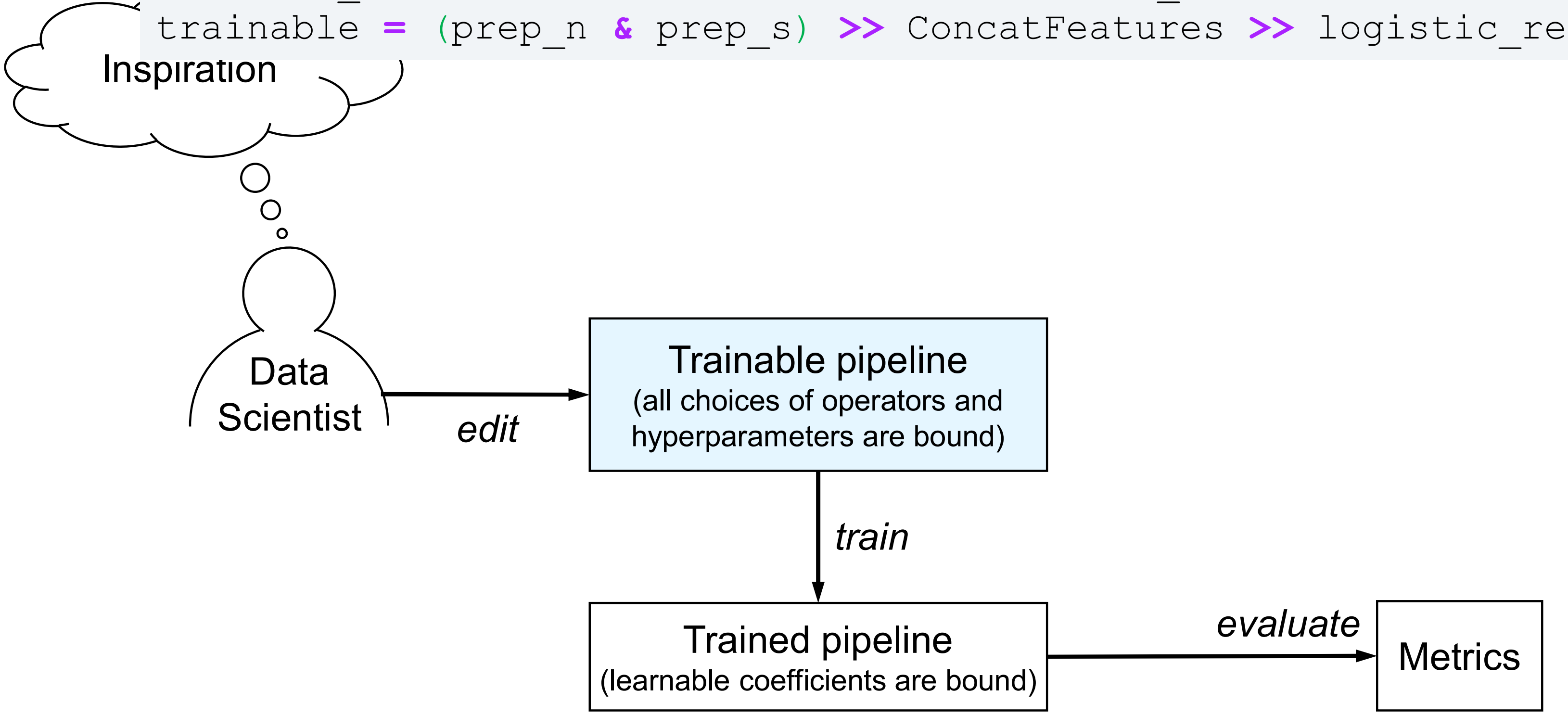


Manual Machine Learning

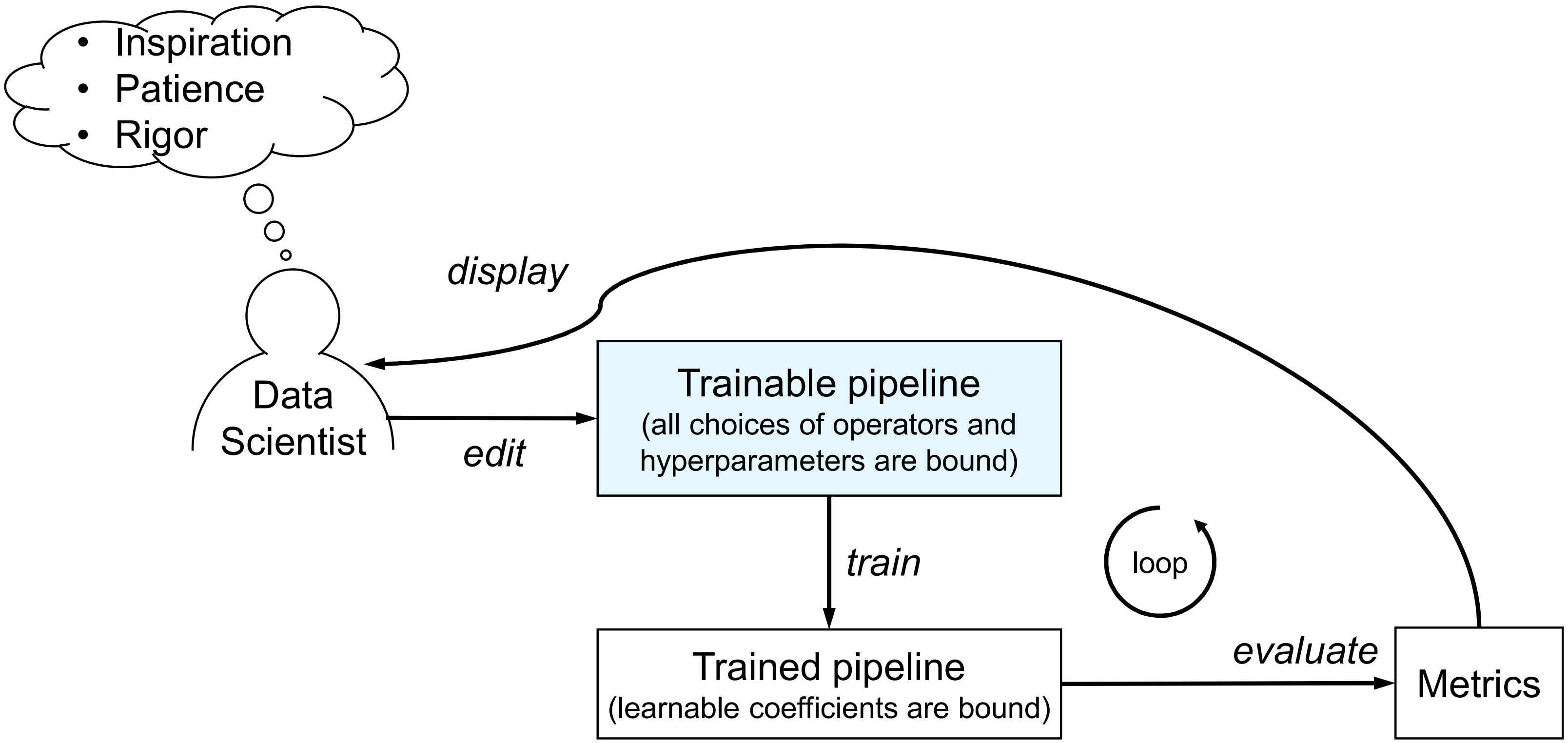

Inspiration
Data Scientist
edit
Trainable pipeline
(all choices of operators and hyperparameters are bound)
train
Trained pipeline
(learnable coefficients are bound)
evaluate
Metrics

Manual Machine Learning

```
prep_n = Project(columns={'type': 'number'}) >> StandardScaler()
prep_s = Project(columns={'type': 'string'}) >> OneHotEncoder()
logistic_regression = LogisticRegression(max_iter=1000)
trainable = (prep_n & prep_s) >> ConcatFeatures >> logistic_regression
```


Inspiration
Data Scientist
edit
Trainable pipeline
(all choices of operators and hyperparameters are bound)
train
Trained pipeline
(learnable coefficients are bound)
evaluate
Metrics

Iterated Manual Machine Learning



- Inspiration
- Patience
- Rigor

Data Scientist

display

edit

Trainable pipeline
(all choices of operators and hyperparameters are bound)

train

loop

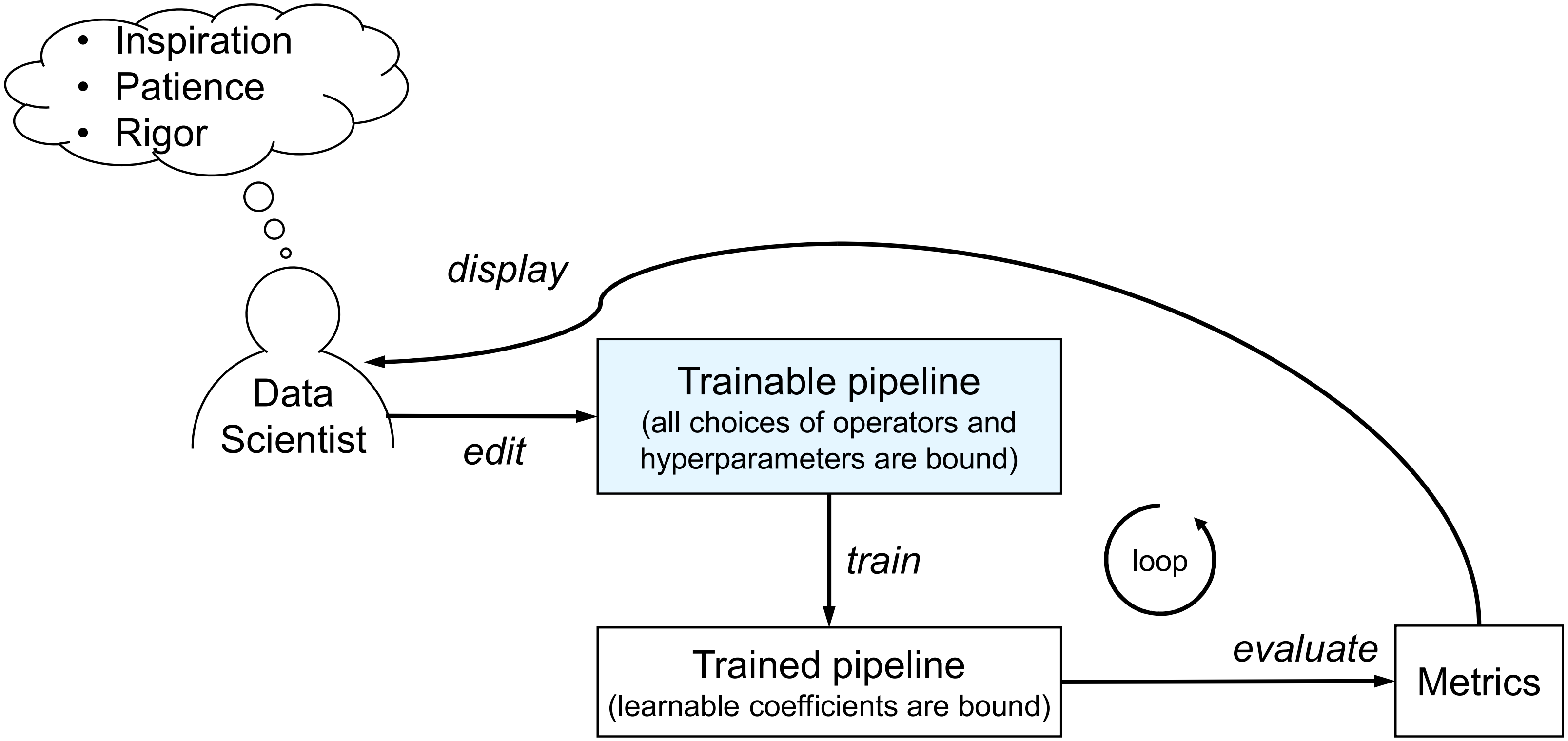
Trained pipeline
(learnable coefficients are bound)

evaluate

Metrics

Iterated Manual Machine Learning

```
trained = trainable.fit(train_X, train_y)
accuracy_score(test_y, trained.predict(test_X))
```



- Inspiration
- Patience
- Rigor

Data Scientist

display

edit

Trainable pipeline
(all choices of operators and hyperparameters are bound)

train

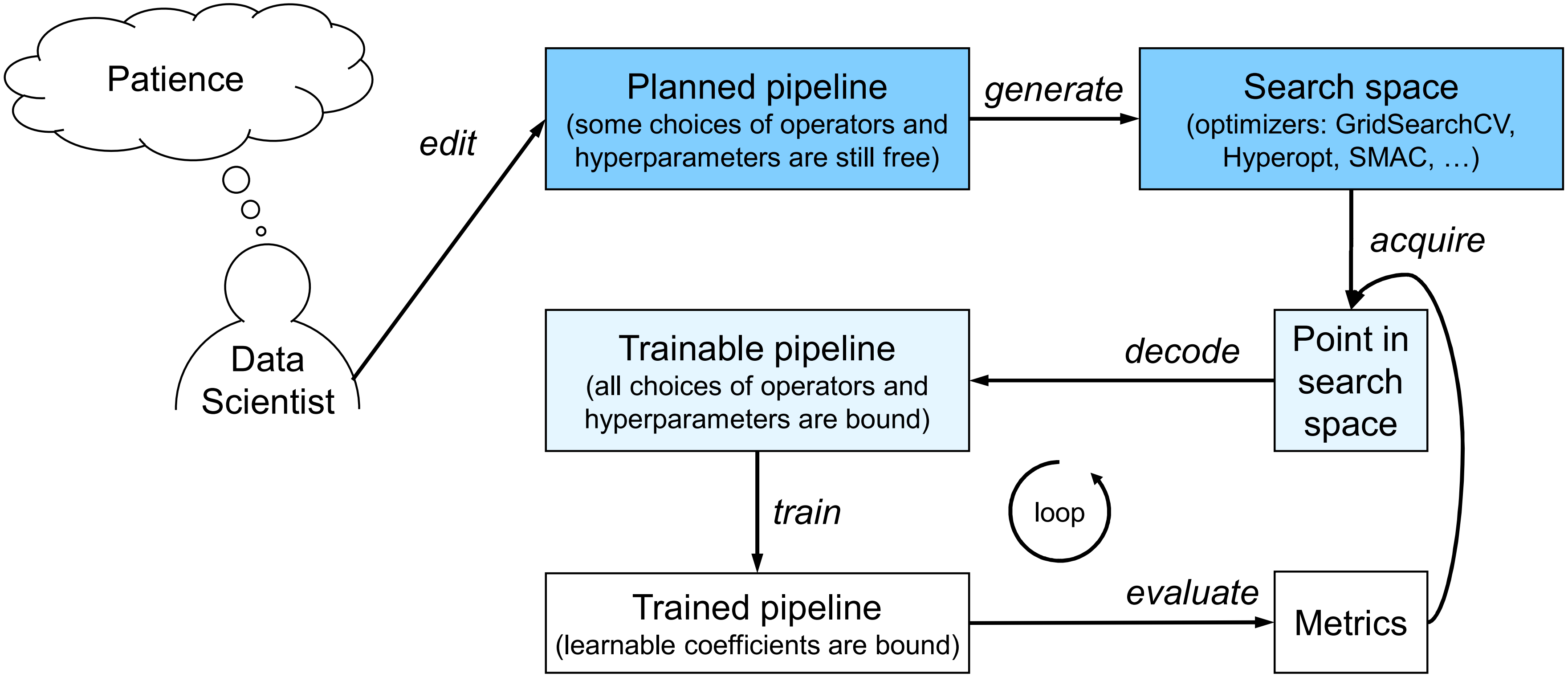
loop

Trained pipeline
(learnable coefficients are bound)

evaluate

Metrics

AutoML: Automated Machine Learning



Patience

Data Scientist

edit

Planned pipeline
(some choices of operators and hyperparameters are still free)

generate

Search space
(optimizers: GridSearchCV, Hyperopt, SMAC, …)

acquire

Point in search space

decode

Trainable pipeline
(all choices of operators and hyperparameters are bound)

train

Trained pipeline
(learnable coefficients are bound)

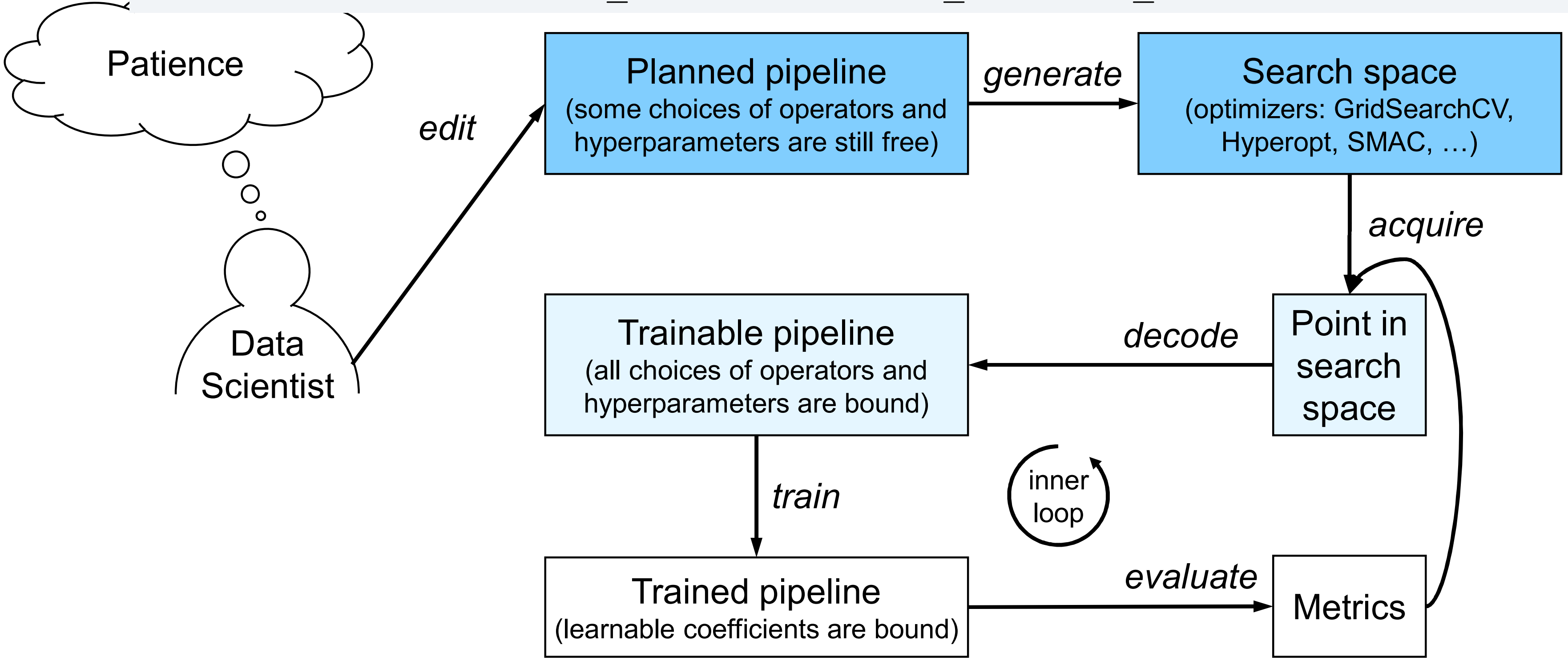
evaluate

Metrics

loop

AutoML: Automated Machine Learning

```
clf = LogisticRegression | DecisionTreeClassifier | KNeighborsClassifier
planned = (prep_n & prep_s) >> ConcatFeatures >> clf
trained = planned.auto_configure(train_X, train_y, optimizer=Hyperopt)
```



Patience

Data Scientist

edit

Planned pipeline
(some choices of operators and hyperparameters are still free)

generate

Search space
(optimizers: GridSearchCV, Hyperopt, SMAC, …)

acquire

Point in search space

decode

Trainable pipeline
(all choices of operators and hyperparameters are bound)

train

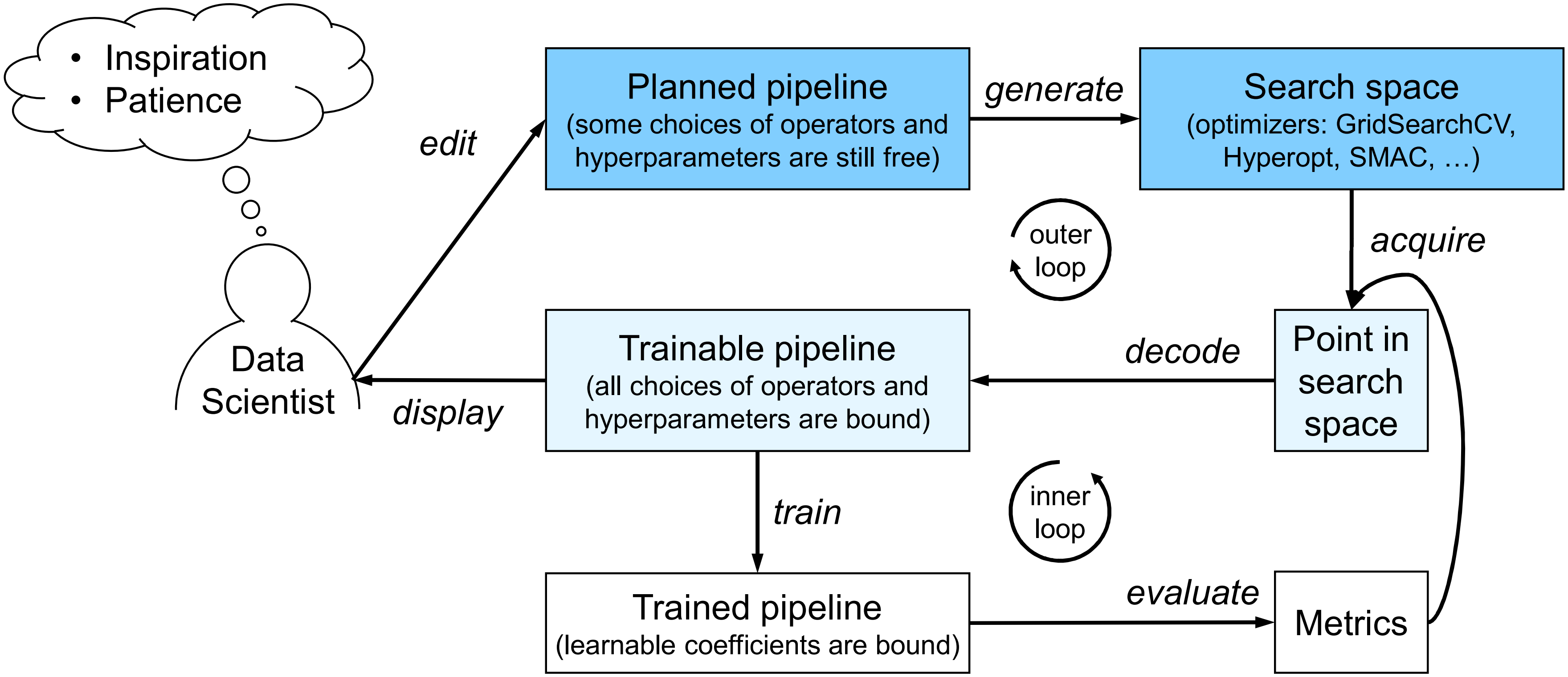
inner loop

Trained pipeline
(learnable coefficients are bound)

evaluate

Metrics

Iterated AutoML


- Inspiration
- Patience

Data Scientist

edit

Planned pipeline
(some choices of operators and hyperparameters are still free)

generate

Search space
(optimizers: GridSearchCV, Hyperopt, SMAC, …)

acquire

outer loop

display

Trainable pipeline
(all choices of operators and hyperparameters are bound)

decode

Point in search space

train

inner loop

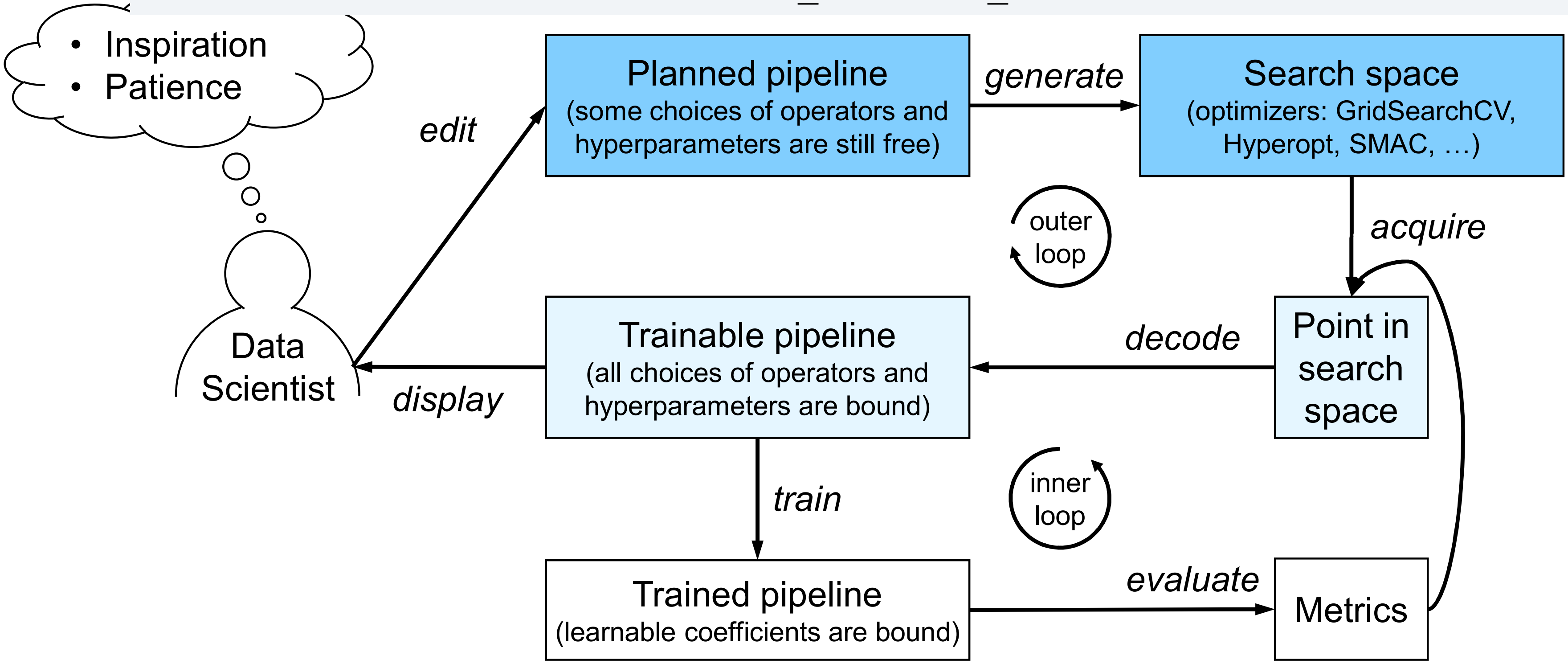
Trained pipeline
(learnable coefficients are bound)

evaluate

Metrics

Iterated AutoML

```
pipeline.visualize()
pipeline.pretty_print(ipython_display=True)
Grove = XGBClassifier.customize_schema(n_estimators=schemas.Int(min=2, max=8))
```



- Inspiration
- Patience

Data Scientist

edit

Planned pipeline
(some choices of operators and hyperparameters are still free)

generate

Search space
(optimizers: GridSearchCV, Hyperopt, SMAC, …)

acquire

outer loop

Point in search space

decode

Trainable pipeline
(all choices of operators and hyperparameters are bound)

display

train

inner loop

Trained pipeline
(learnable coefficients are bound)

evaluate

Metrics

LALE at a Glance

<table>
<tr><th></th><th>API</th><th>Description</th></tr>
<tr><td rowspan="3">manual ML</td><td><code>op2 = op1(hyperparams)</code></td><td>initialize and make trainable</td></tr>
<tr><td><code>op3 = op2.fit(train_X, train_y)</code></td><td>train</td></tr>
<tr><td><code>y_pred = op3.predict(test_X)</code></td><td>predict</td></tr>
<tr><td rowspan="3">pipeline combinators</td><td><code>op3 = op1 >> op2</code></td><td>pipe (add dataflow edge)</td></tr>
<tr><td><code>op3 = (op1 & op2) >> ConcatFeatures</code></td><td>union (separate transforms)</td></tr>
<tr><td><code>op3 = op1 | op2   # often with | NoOp</code></td><td>choice (AutoML picks one)</td></tr>
<tr><td rowspan="3">semi-automated ML</td><td><code>op3 = op1.auto_configure(X, y, optimizer)</code></td><td>run AutoML</td></tr>
<tr><td><code>op.visualize()</code></td><td>display graphically</td></tr>
<tr><td><code>op.pretty_print()</code></td><td>display as Python code</td></tr>
</table>

See also LALE's online documentation for the hyperparameter schemas of over 180 operators (from scikit-learn, imblearn, AIF360, Snap ML, …), e.g.:

```
from lale.lib.lale import ConcatFeatures, NoOp, Hyperopt
```