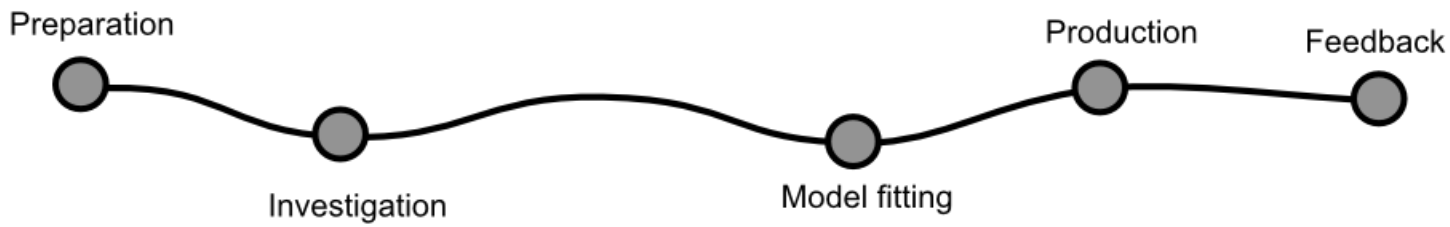
Predictive models design

Predictive models design

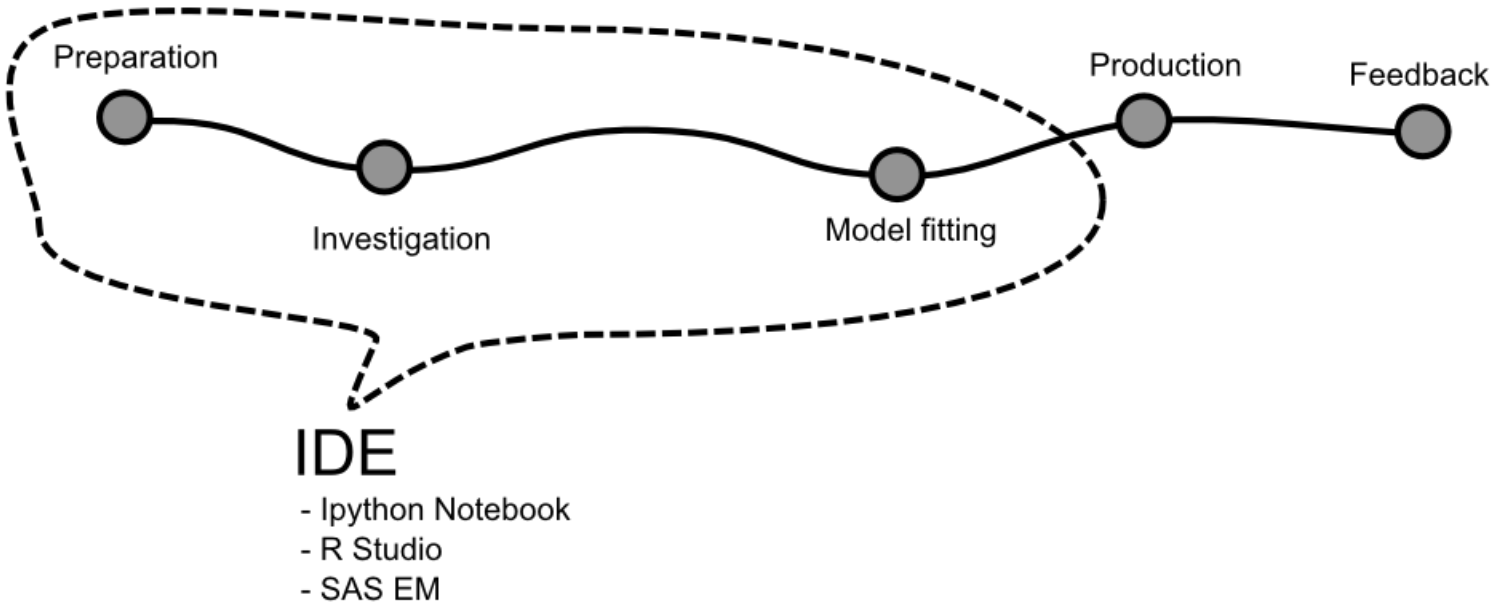
Predictive models design

Predictive models design

Predictive models design



Small data case



IP[y]: Notebook iris example (unsaved changes)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

In [1]:
```
from sklearn import datasets
iris = datasets.load_iris()
```

IP[y]: Notebook iris example (autosaved)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

In [1]:
```
from sklearn import datasets
iris = datasets.load_iris()
```

In [ ]:
```
import pandas as pd
pd.set_option('display.max_rows', 10)
iris_df = pd.DataFrame (iris.data, columns = iris.feature_names)
iris_df["target"] = iris.target
iris_df
```

IP[y]: Notebook iris example (unsaved changes)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

In [1]:
```
from sklearn import datasets
iris = datasets.load_iris()
```

In [2]:
```
import pandas as pd
pd.set_option('display.max_rows', 10)
iris_df = pd.DataFrame (iris.data, columns = iris.feature_names)
iris_df["target"] = iris.target
iris_df
```

Out[2]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

150 rows × 5 columns

IP[y]: Notebook iris example (unsaved changes)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

```
In [ ]: from sklearn import datasets
        iris = datasets.load_iris()
```

```
In [ ]: import pandas as pd
        pd.set_option('display.max_rows', 10)
        iris_df = pd.DataFrame (iris.data, columns = iris.feature_names)
        iris_df["target"] = iris.target
        iris_df
```

```
In [ ]: %matplotlib inline
        import seaborn as sns
        sns.pairplot(iris_df, hue="target", size=2.5)
```

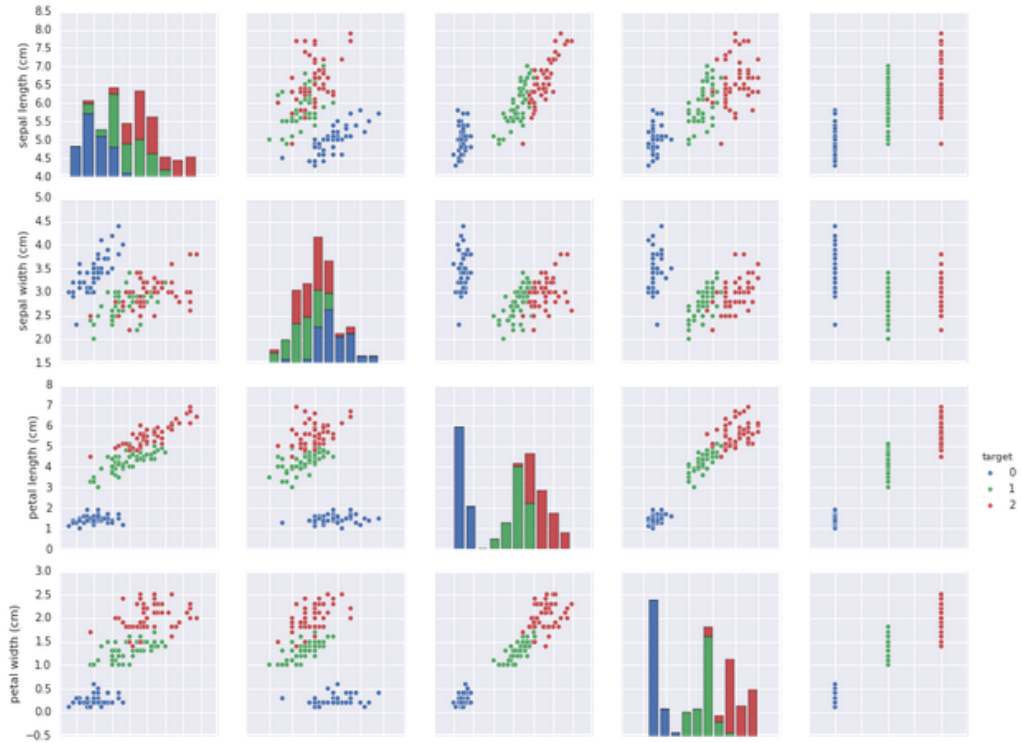
IP[y]: Notebook iris example (autosaved)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

In [4]:
```
%matplotlib inline
import seaborn as sns
sns.pairplot(iris_df, hue="target", size=2.5)
```

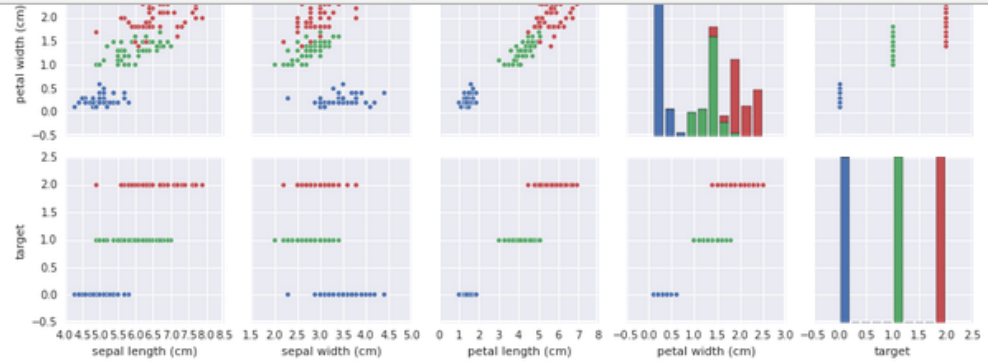
Out[4]:
```
<seaborn.axisgrid.PairGrid at 0x7fc3484a4e10>
```



IP[y]: Notebook iris example (unsaved changes)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

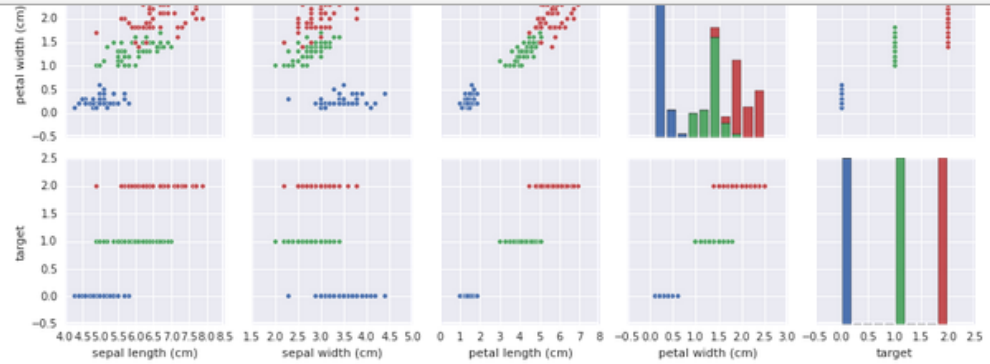


In [ ]:
```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
model = lr.fit(iris.data, iris.target)

predictions = model.predict(iris.data)
```

IP[y]: Notebook iris example (autosaved)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

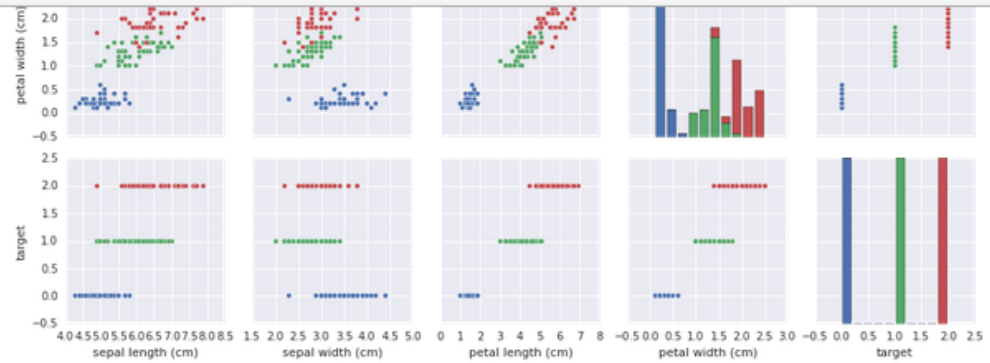


In [6]:
```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
model = lr.fit(iris.data, iris.target)

predictions = model.predict(iris.data)
```

IP[y]: Notebook iris example (autosaved)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None



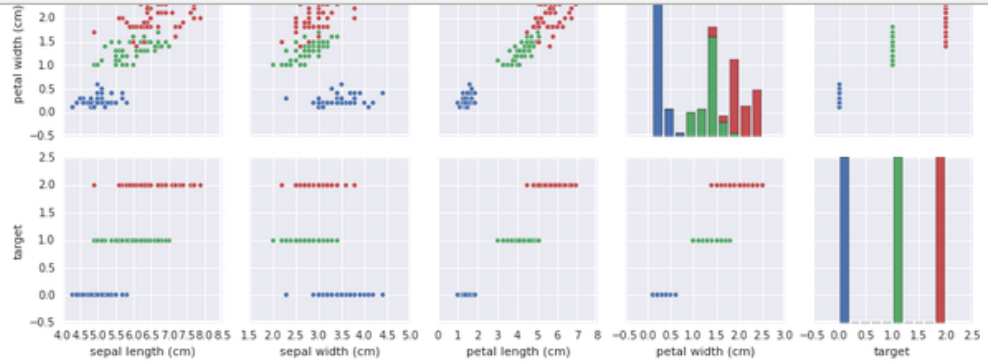
In [6]:
```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
model = lr.fit(iris.data, iris.target)

predictions = model.predict(iris.data)
```

In [ ]:
```
from sklearn.metrics import confusion_matrix
confusion = confusion_matrix (iris.target, predictions)
pd.DataFrame (confusion, columns=iris.target_names, index=iris.target_names)
```

IP[y]: Notebook iris example (unsaved changes)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None



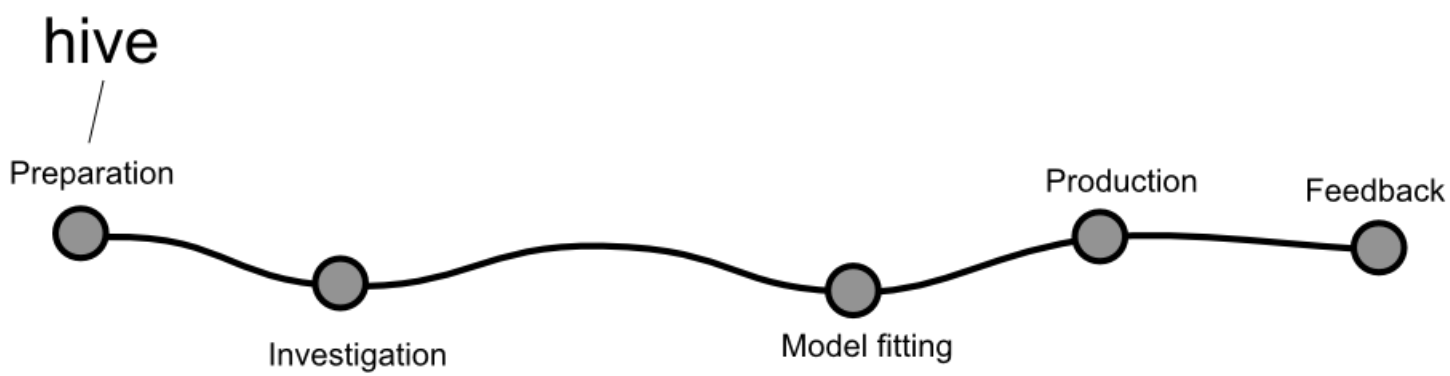
In [6]:
```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
model = lr.fit(iris.data, iris.target)

predictions = model.predict(iris.data)
```

In [7]:
```
from sklearn.metrics import confusion_matrix
confusion = confusion_matrix (iris.target, predictions)
pd.DataFrame (confusion, columns=iris.target_names, index=iris.target_names)
```

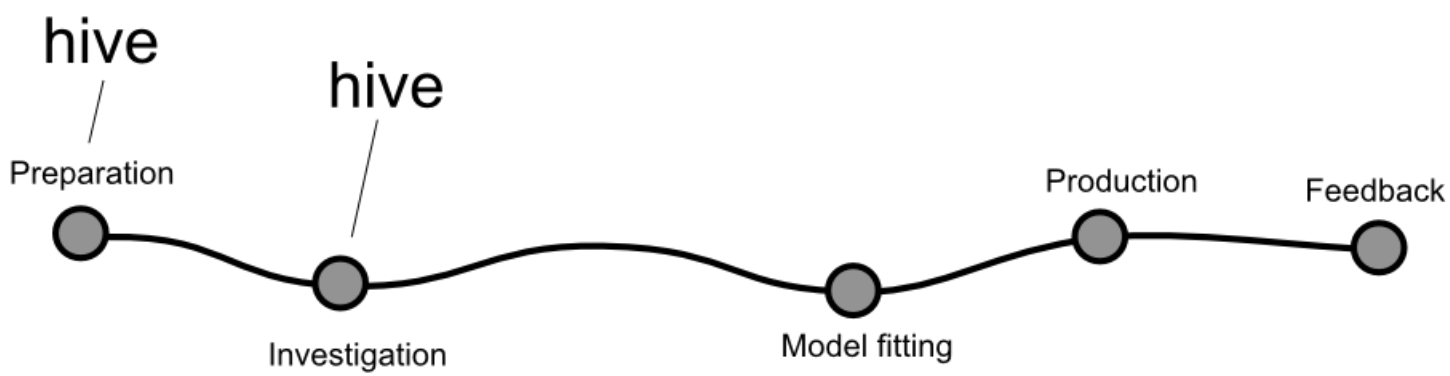
Out[7]:

	setosa	versicolor	virginica
setosa	50	0	0
versicolor	0	45	5
virginica	0	1	49

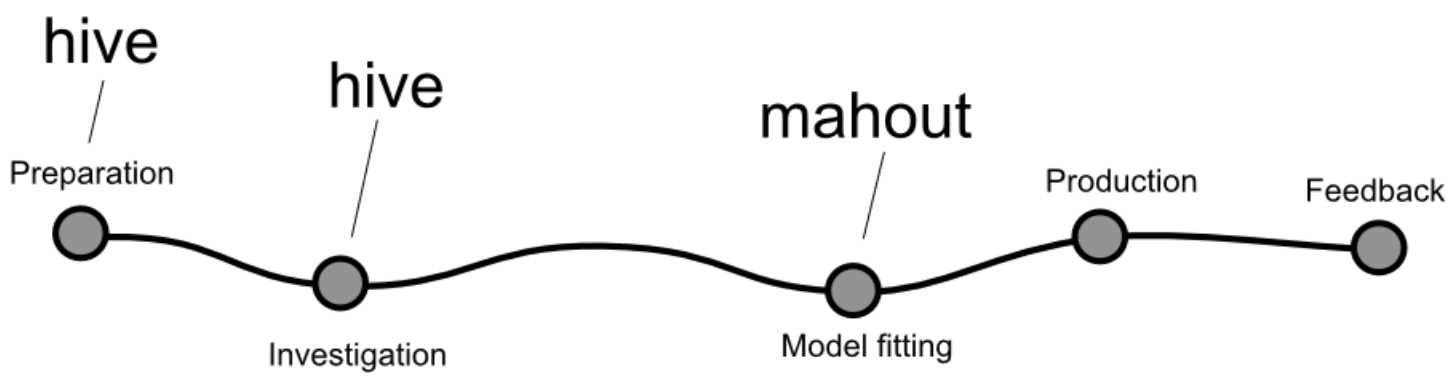
Big data case



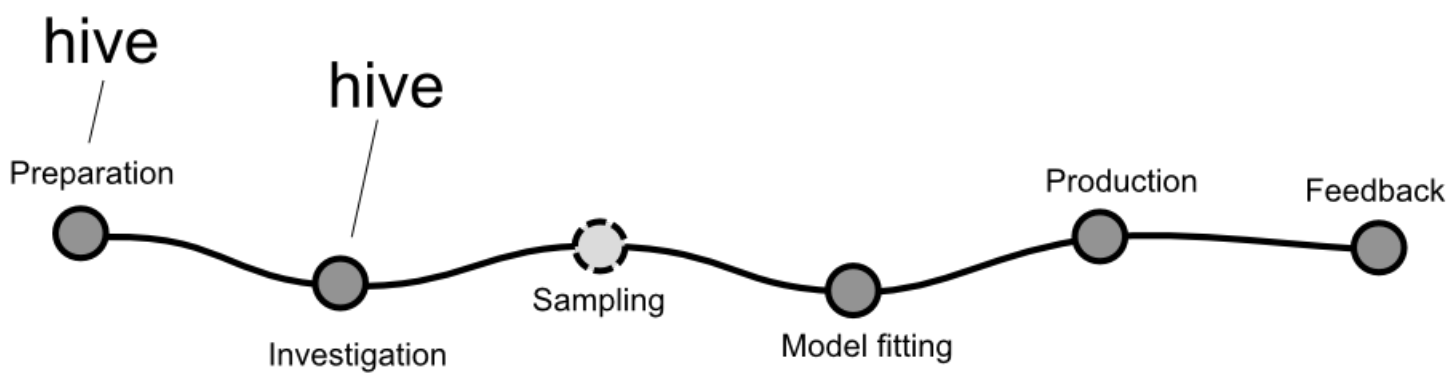
Big data case



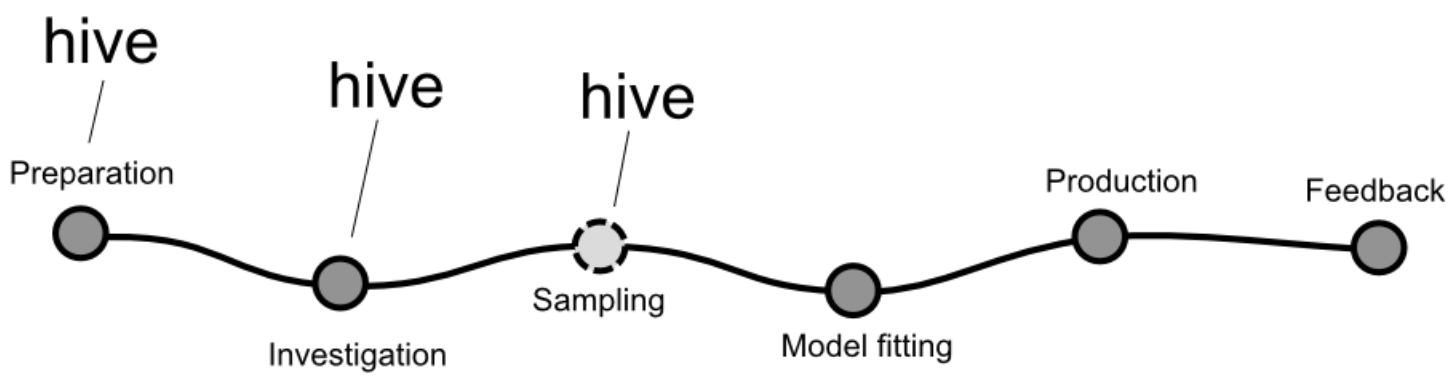
Big data case



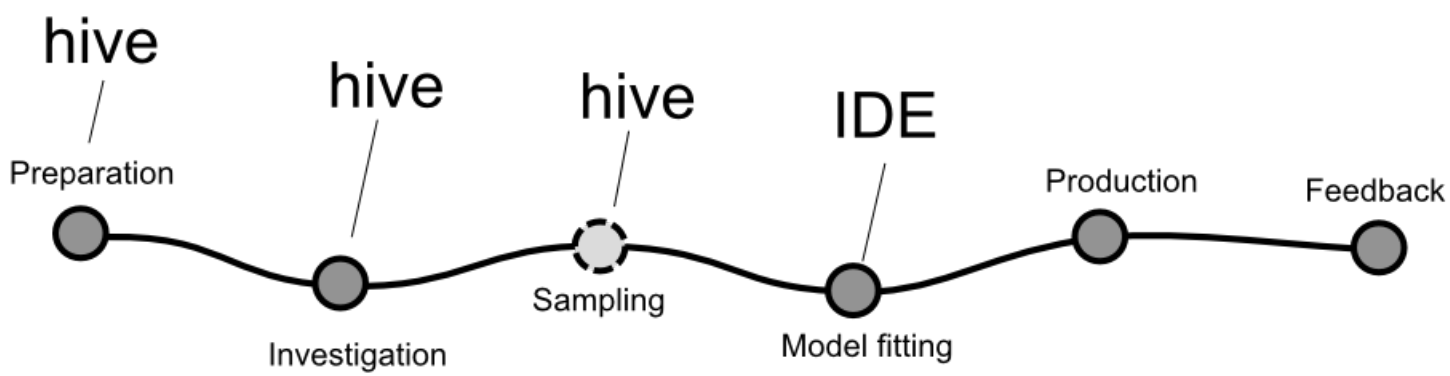
Big data case



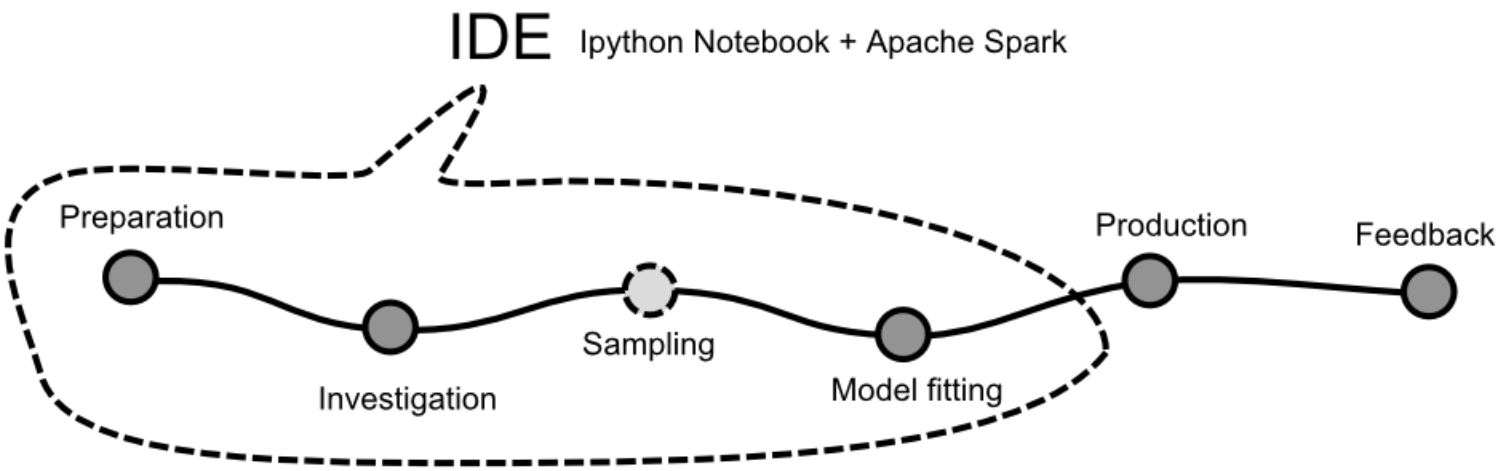
Big data case



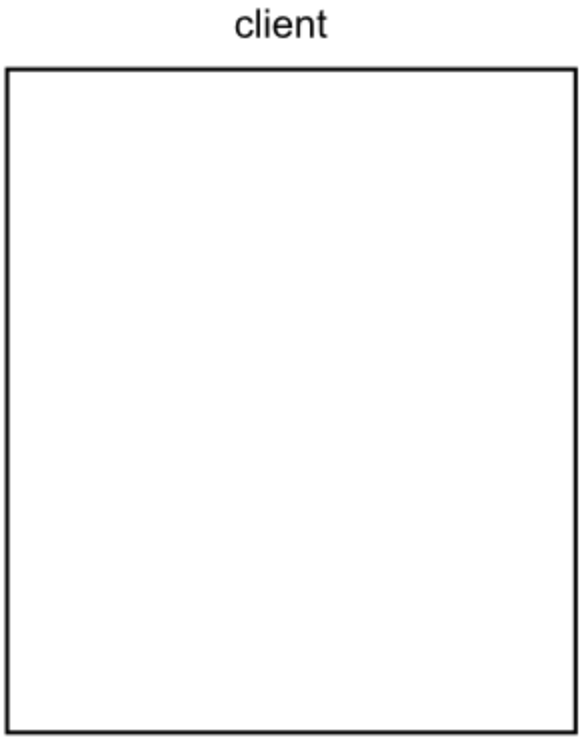
Big data case



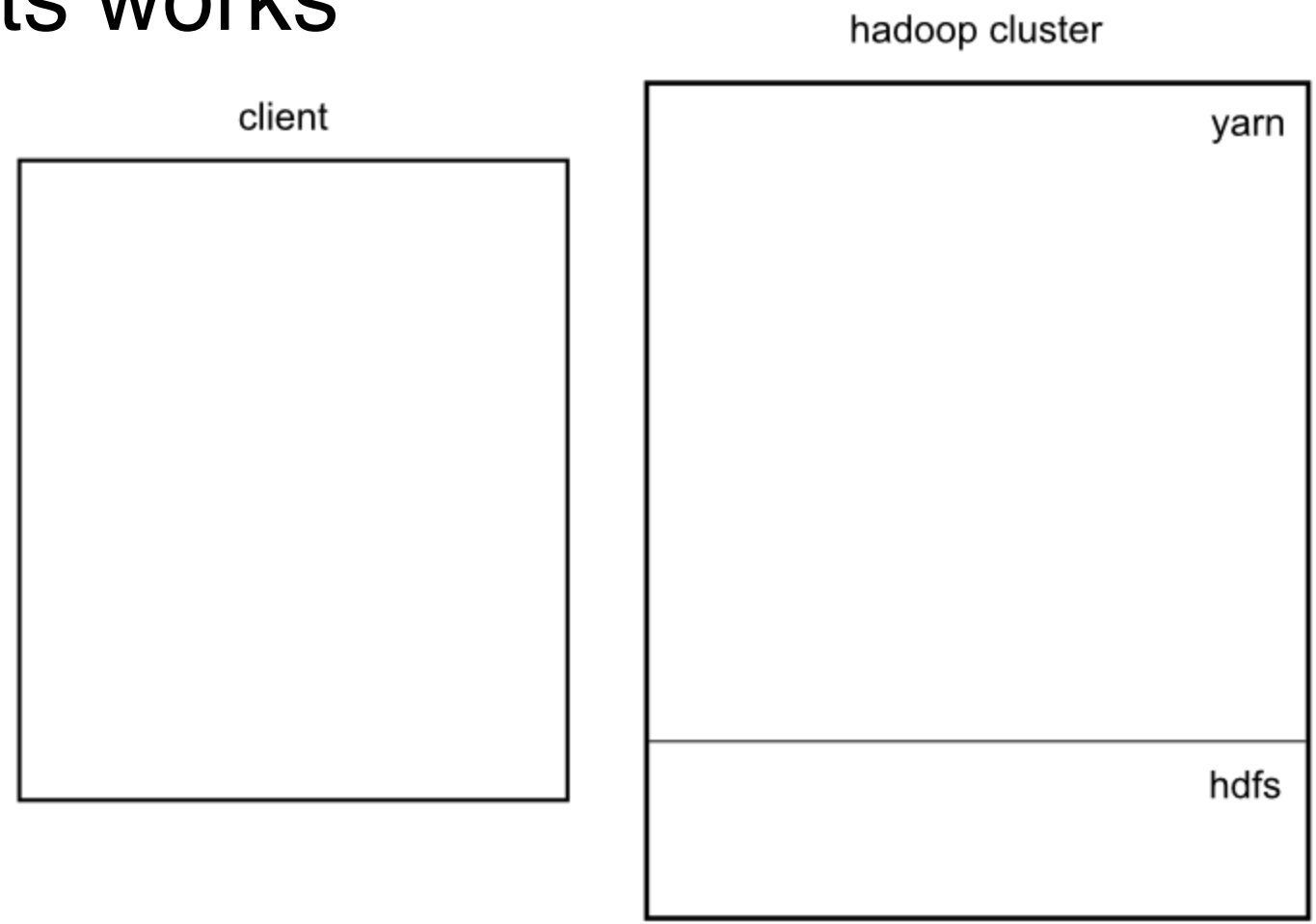
Apache Spark case



How does its works

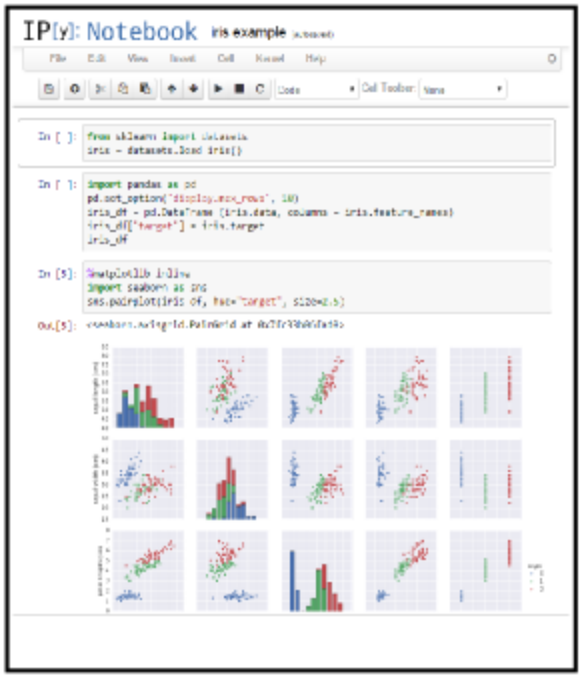


How does its works

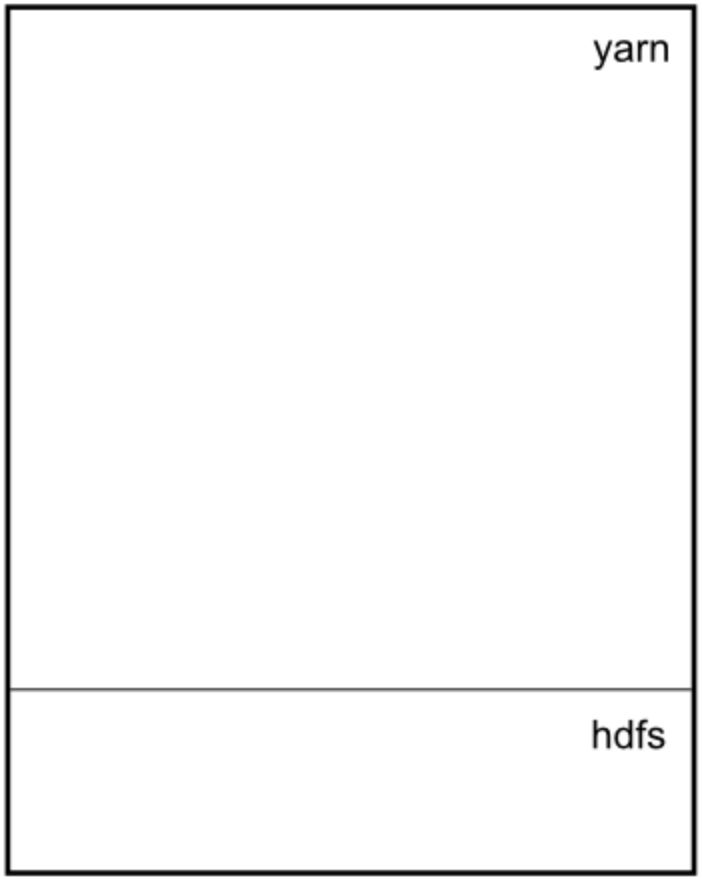


How does its works

client



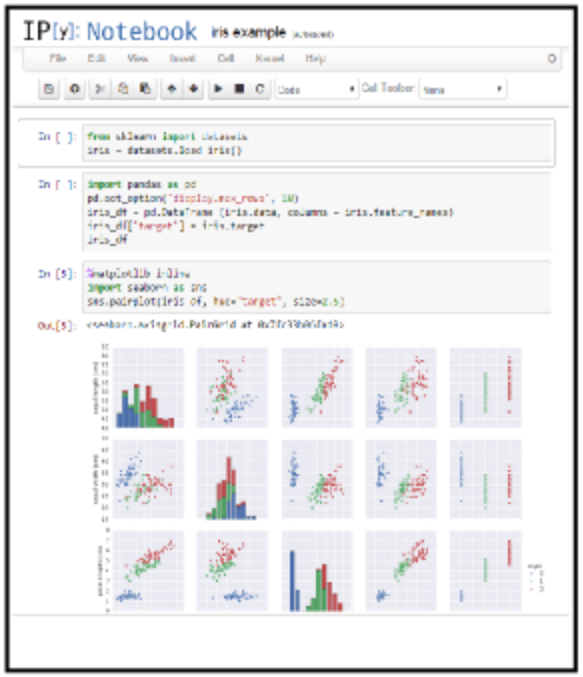
hadoop cluster



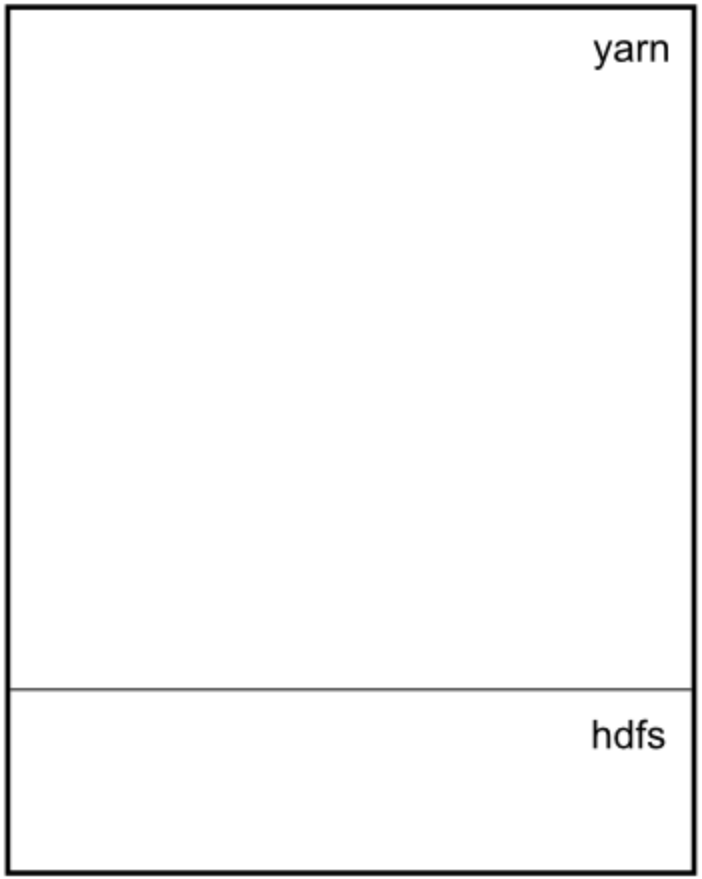
How does its works

```
sc = SparkContext( SparkConf() )
```

client

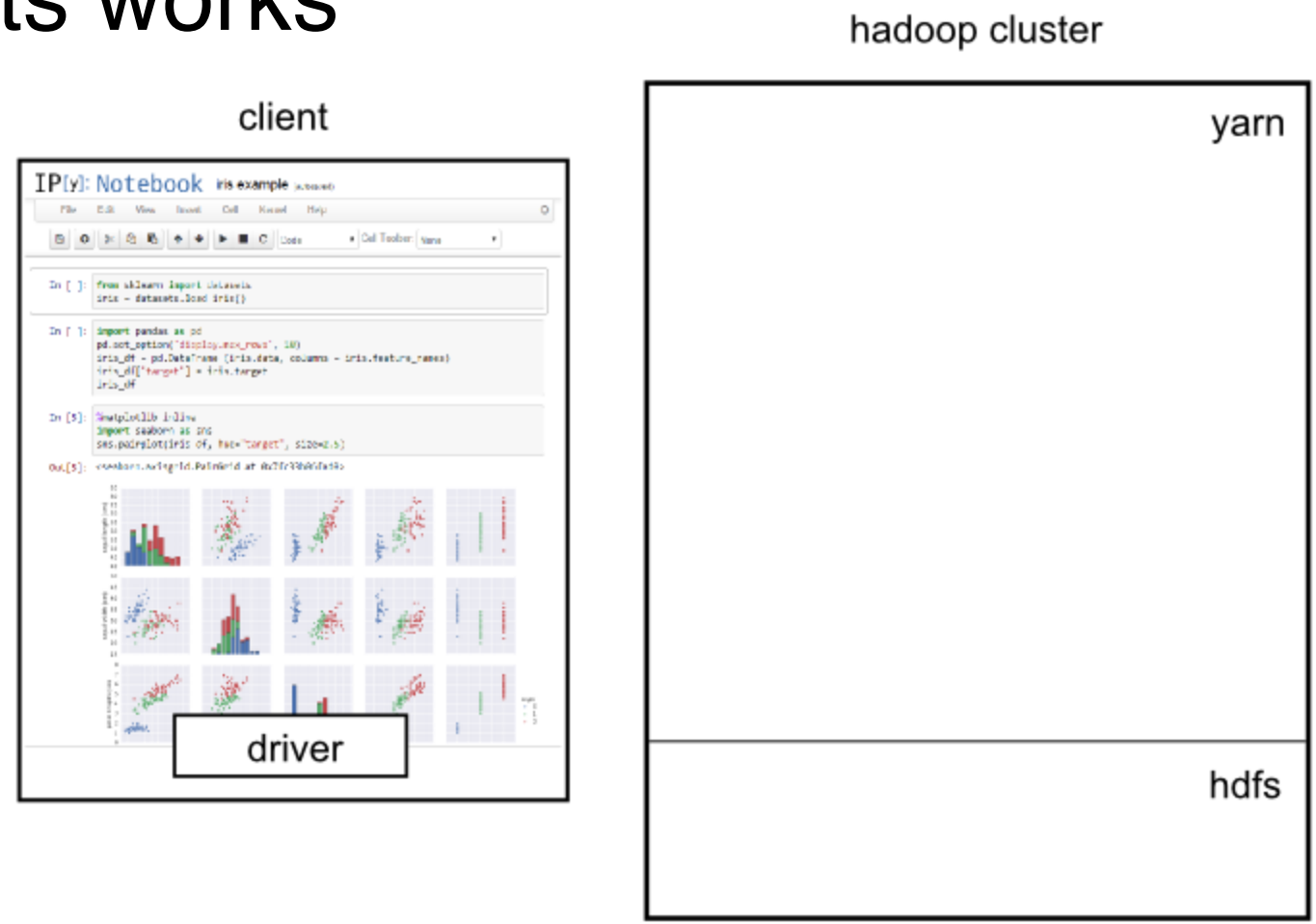


hadoop cluster



How does its works

```
sc = SparkContext( SparkConf() )
```



How does its works

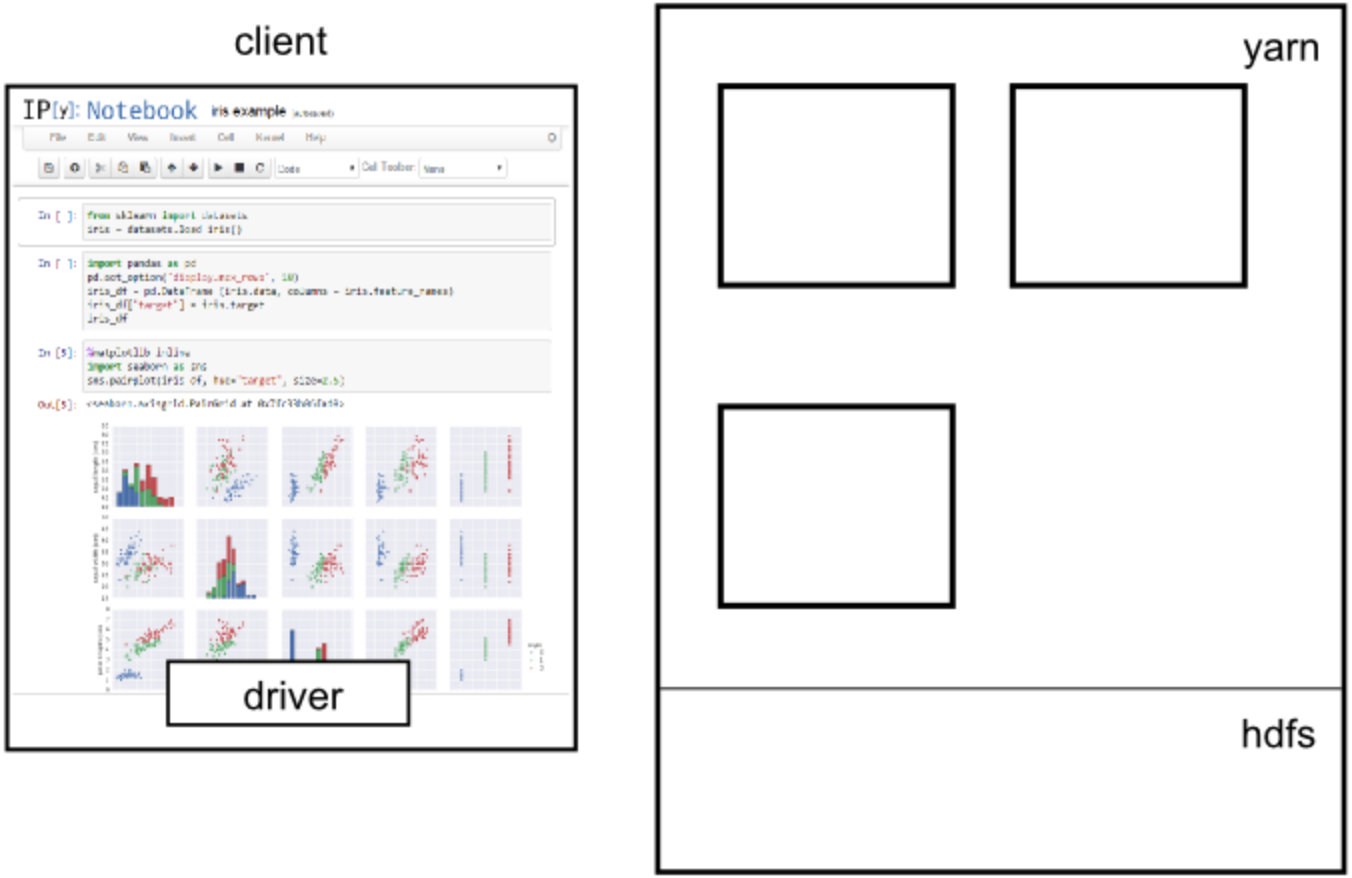
```
sc = SparkContext( SparkConf() )
```

How does its works

```
sc = SparkContext( SparkConf() )
```

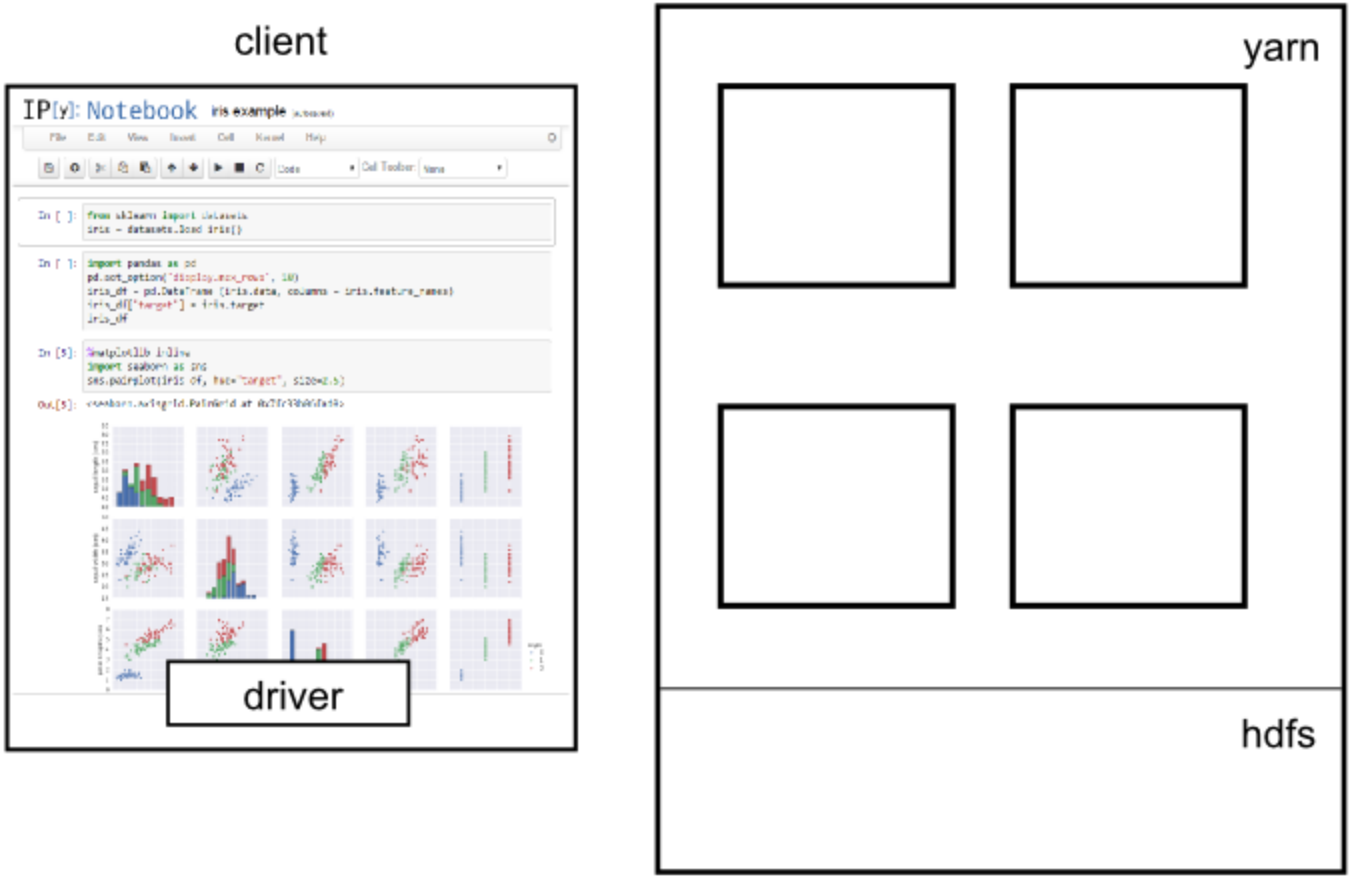
How does its works

```
sc = SparkContext( SparkConf() )
```



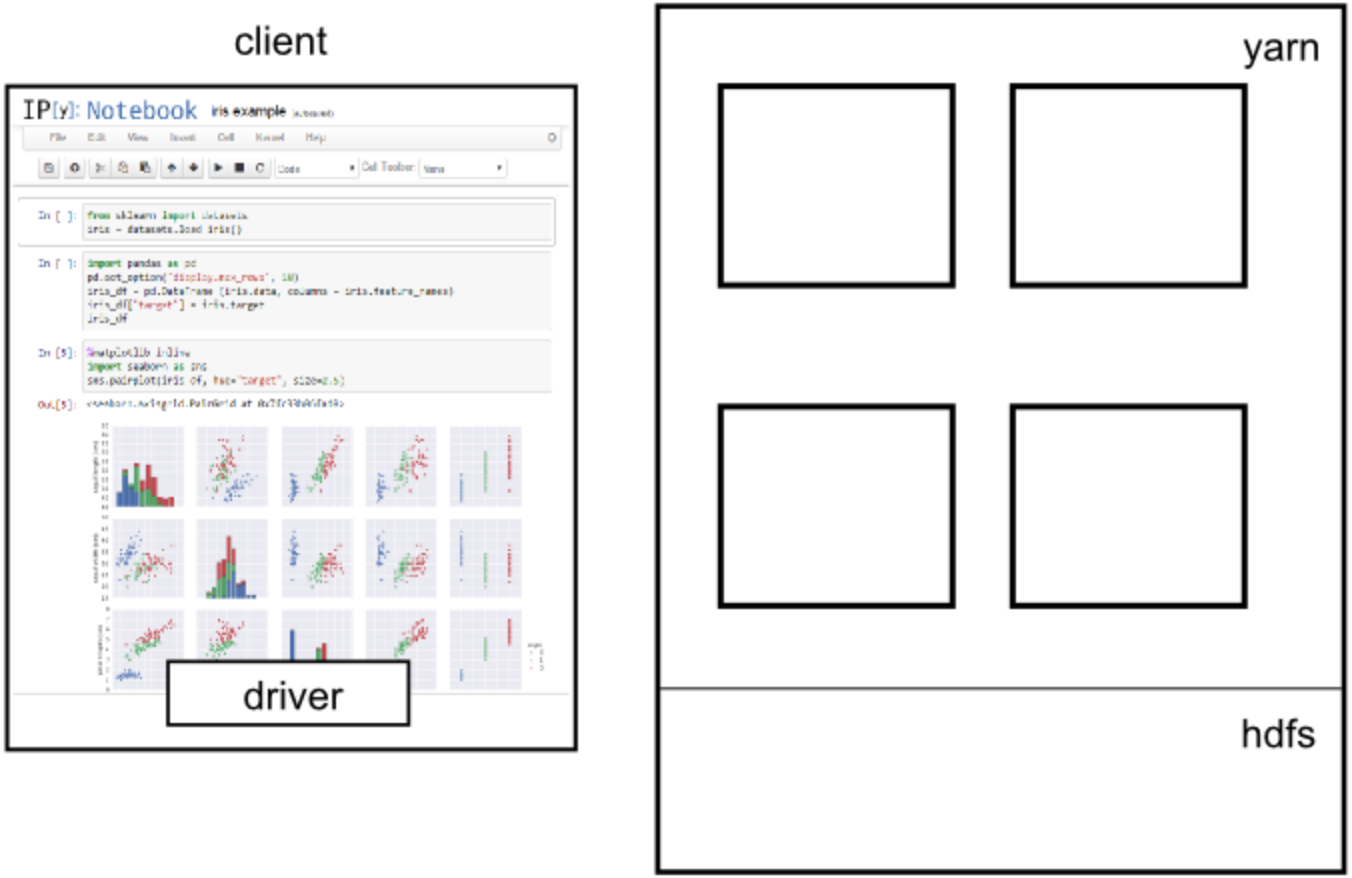
How does its works

```
sc = SparkContext( SparkConf() )
```



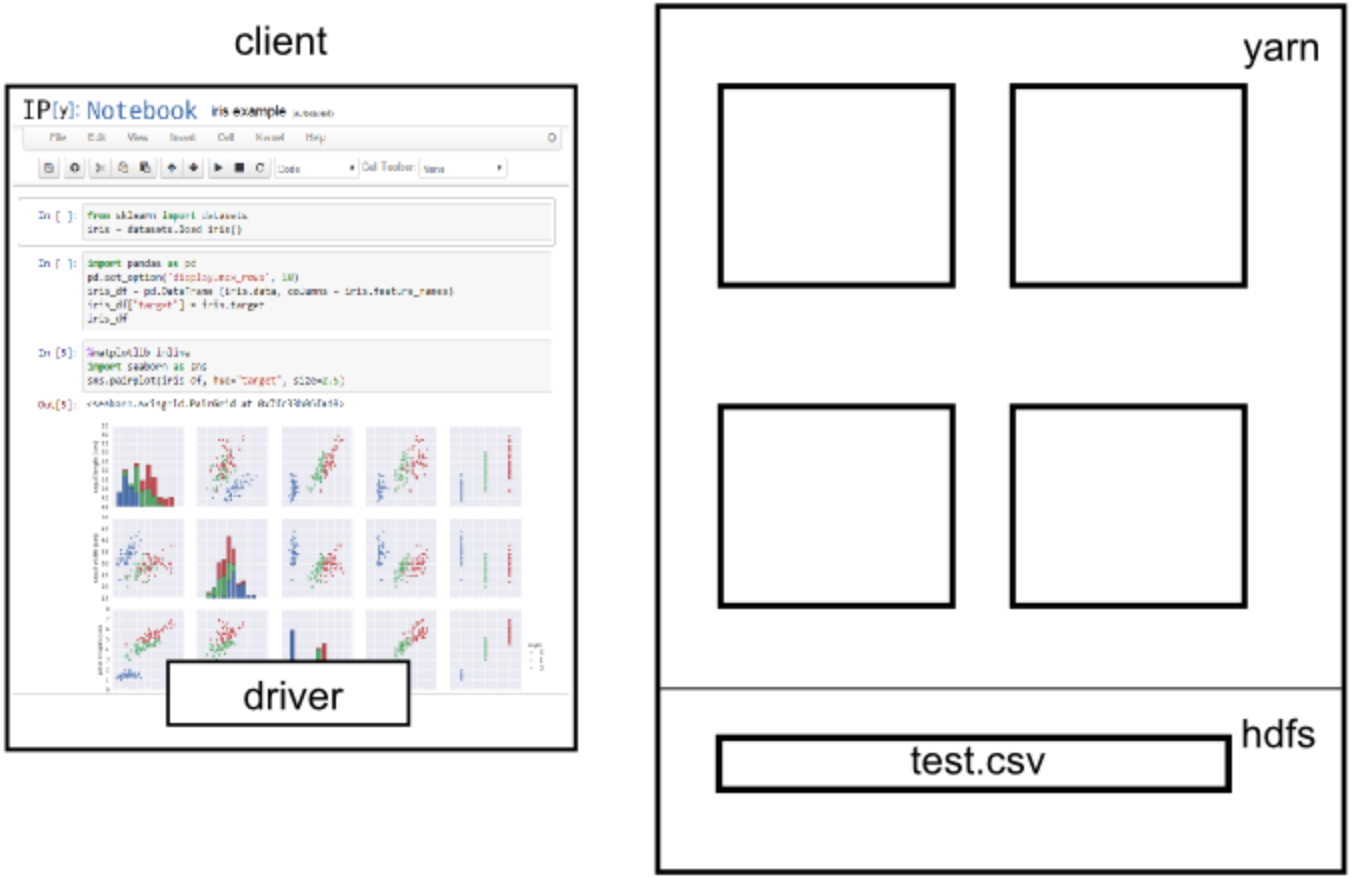
How does its works

```
a_rdd = sc.textFile("test.csv")
```



How does its works

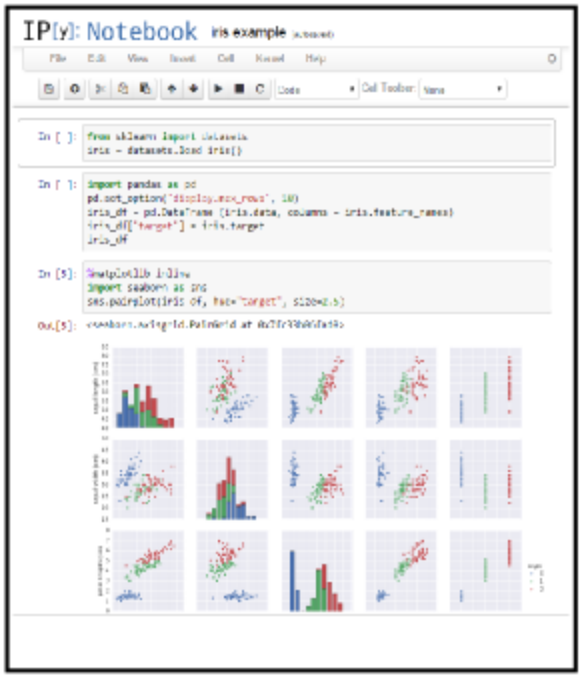
```
a_rdd = sc.textFile("test.csv")
```



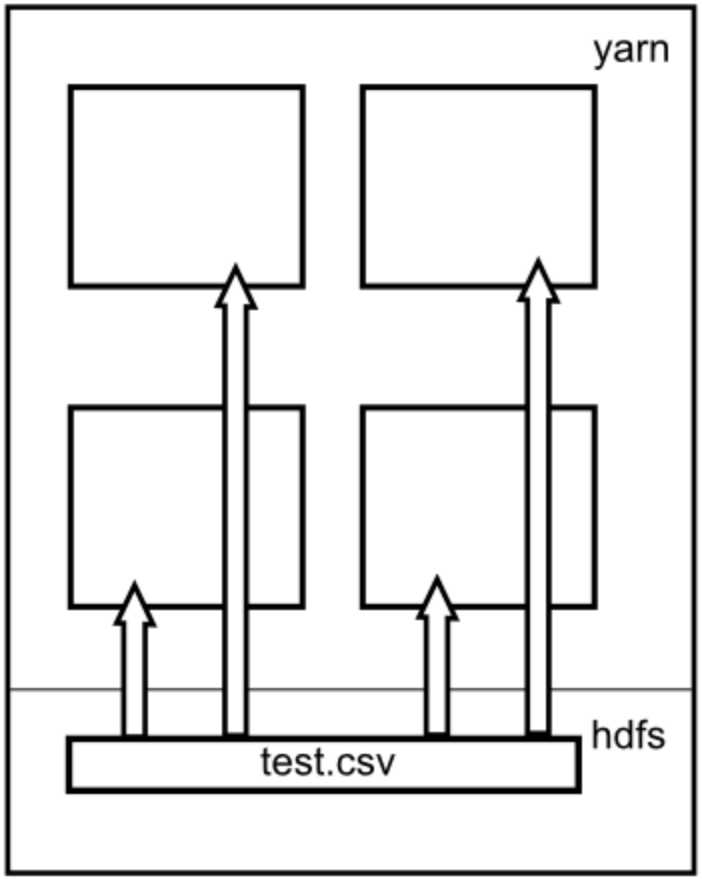
How does its works

```
a_rdd = sc.textFile("test.csv")
```

client



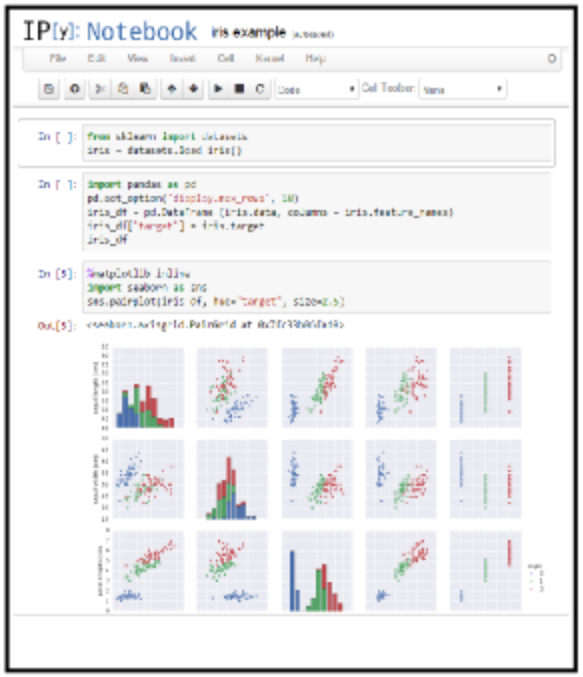
hadoop cluster



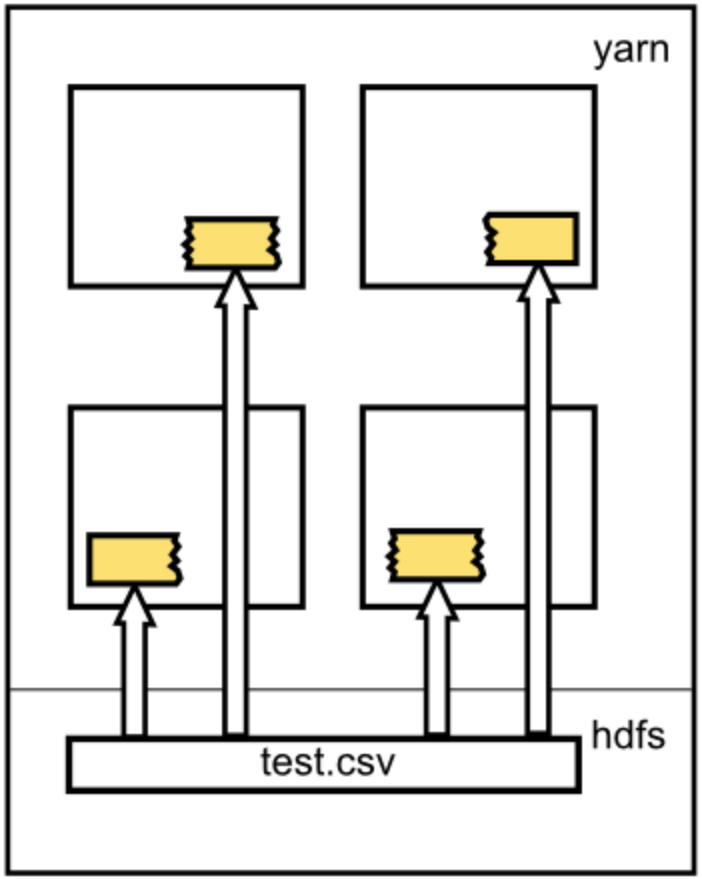
How does its works

```
a_rdd = sc.textFile("test.csv")
```

client



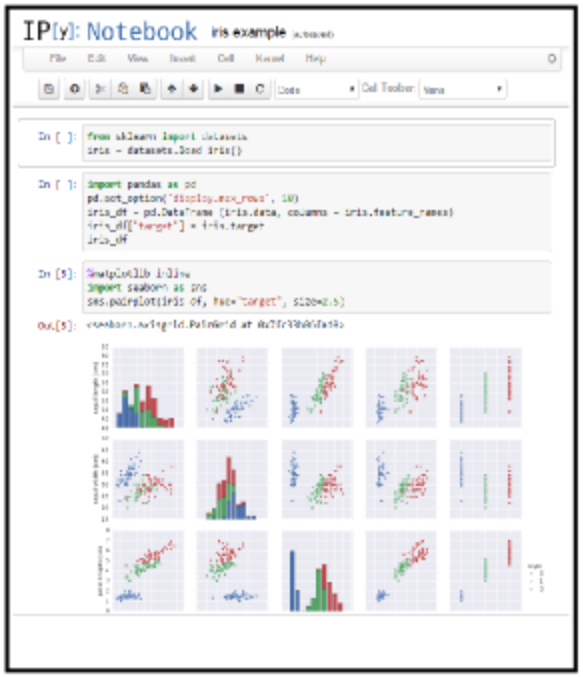
hadoop cluster



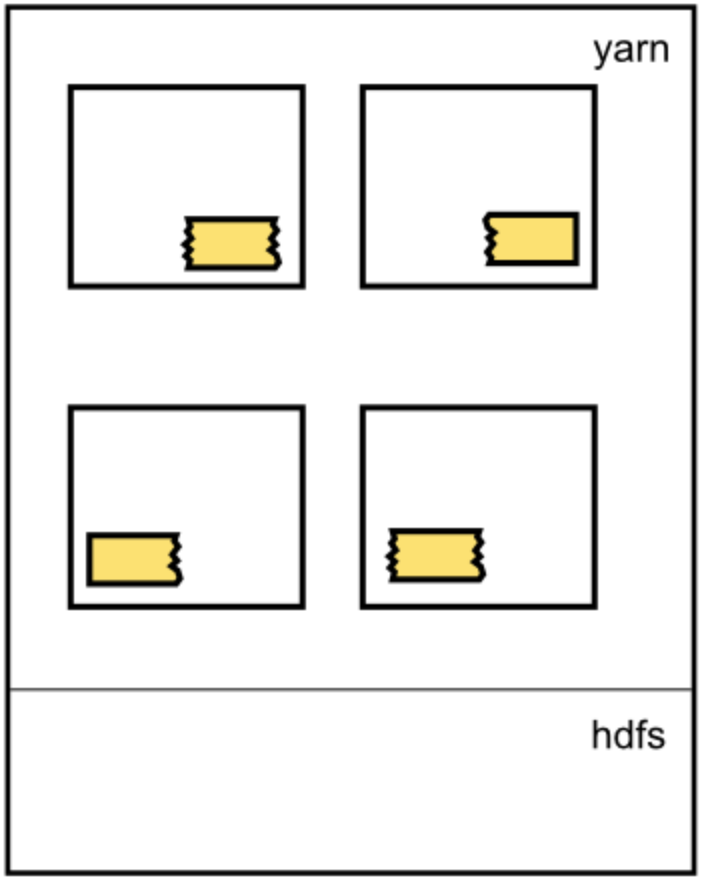
How does its works

```
b_rdd = a_rdd.map (....)
c_rdd = b_rdd.reduce (....)
e_rdd = c_rdd.join (a_rdd)
```

client



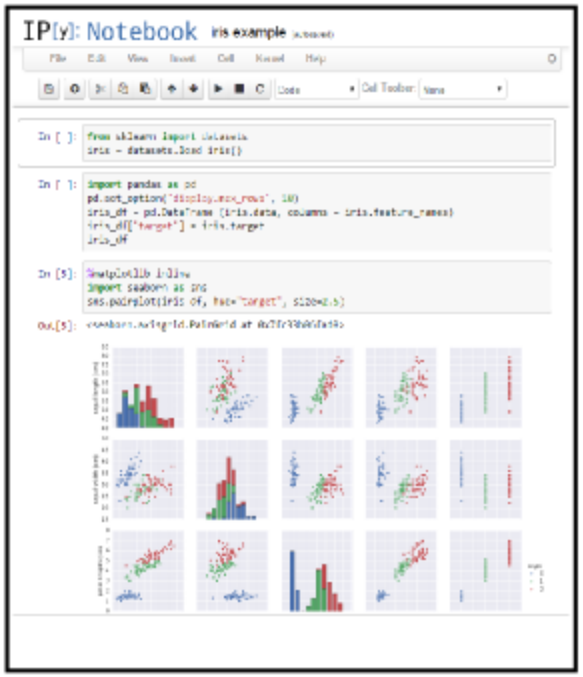
hadoop cluster



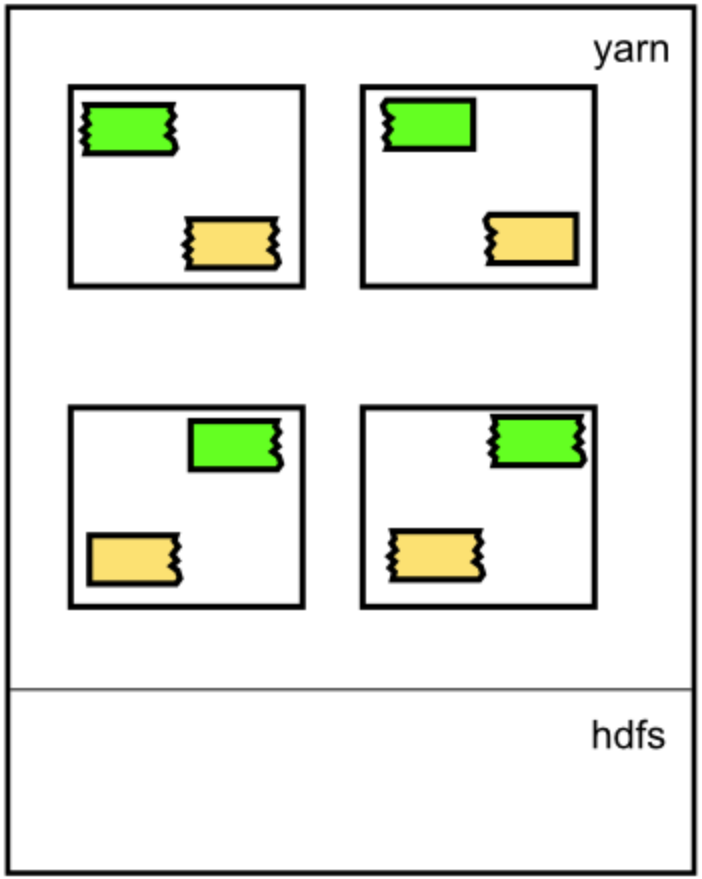
How does its works

```
b_rdd = a_rdd.map (....)
c_rdd = b_rdd.reduce (....)
e_rdd = c_rdd.join (a_rdd)
```

client



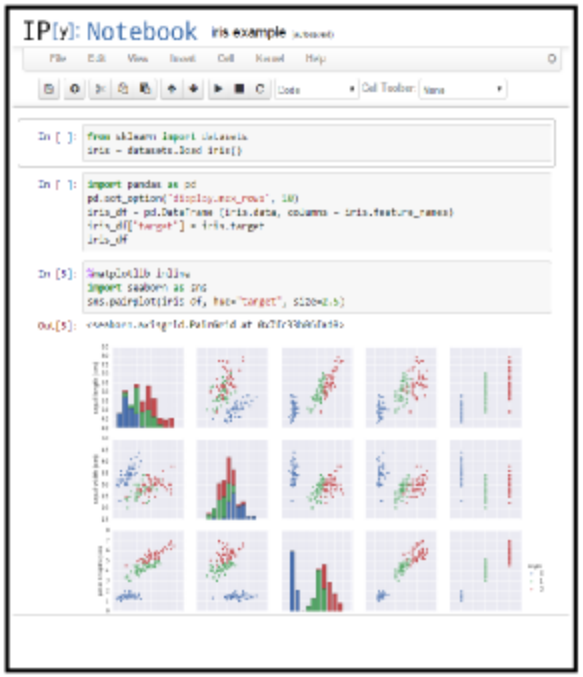
hadoop cluster



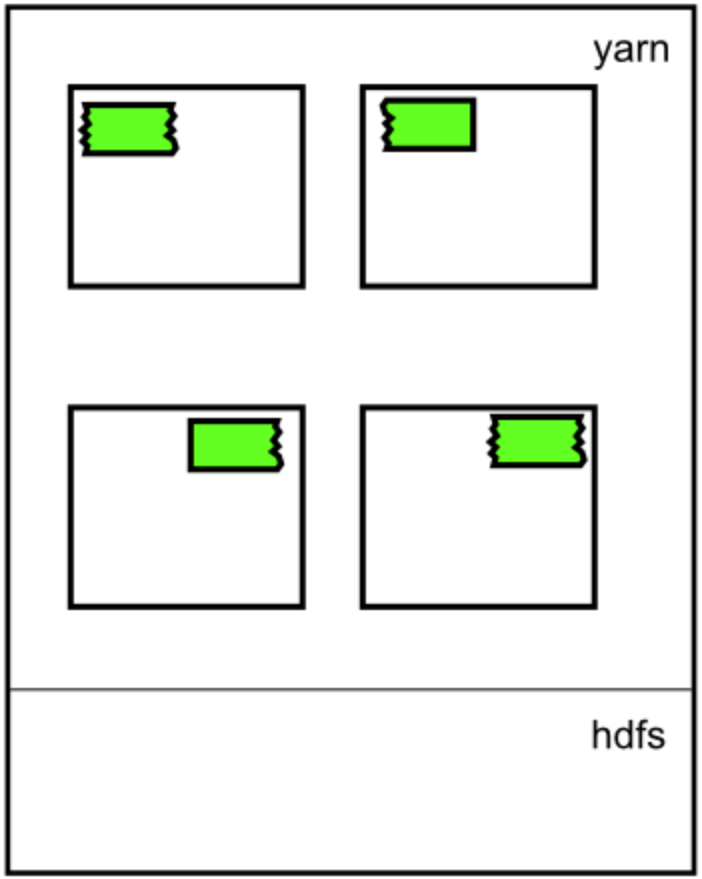
How does its works

```
e_list = e_rdd.collect()
```

client

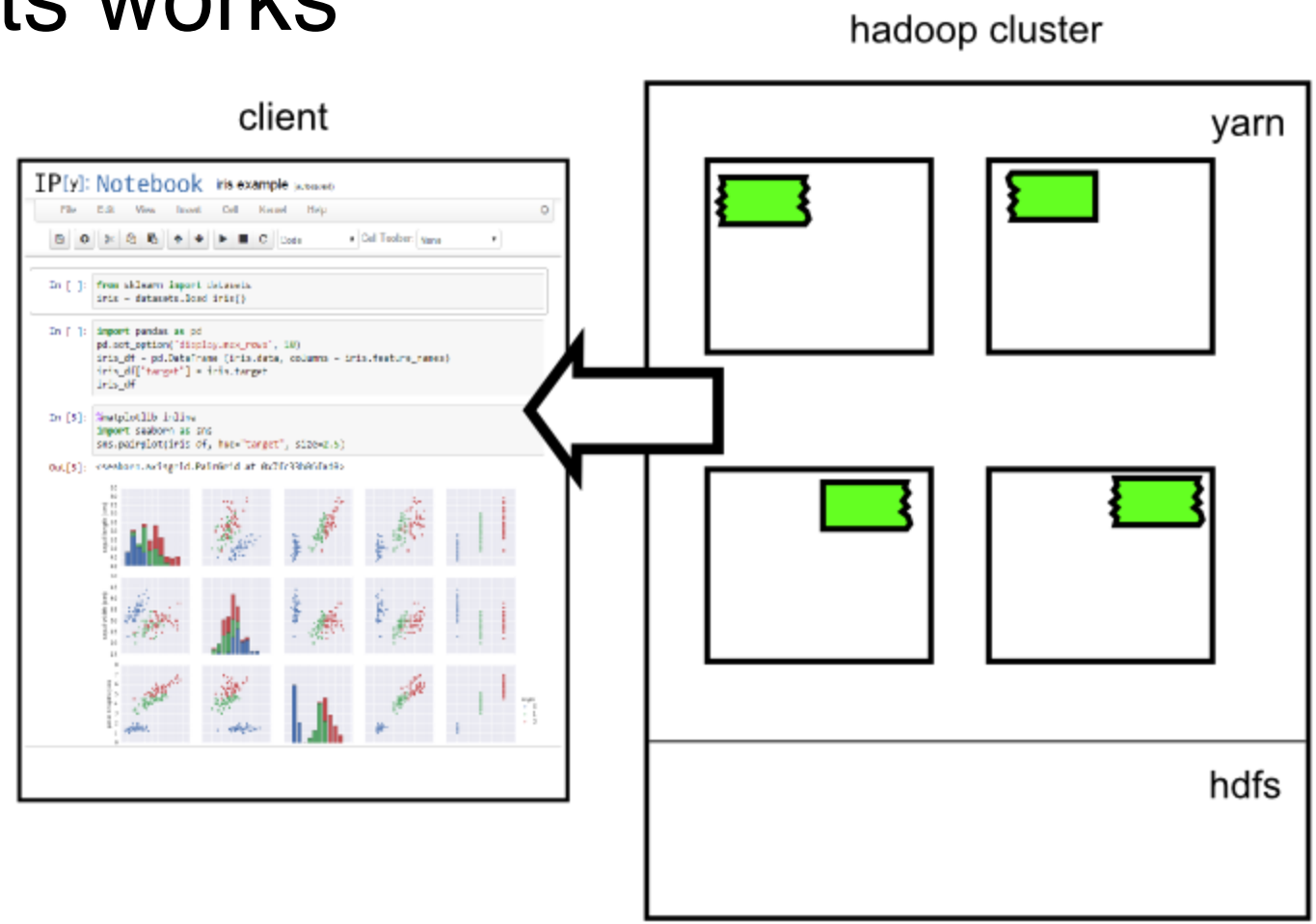


hadoop cluster



How does its works

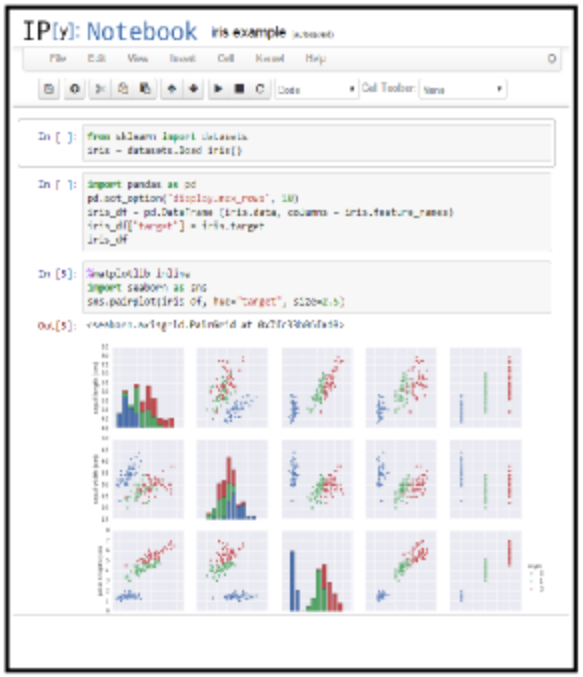
```
e_list = e_rdd.collect()
```



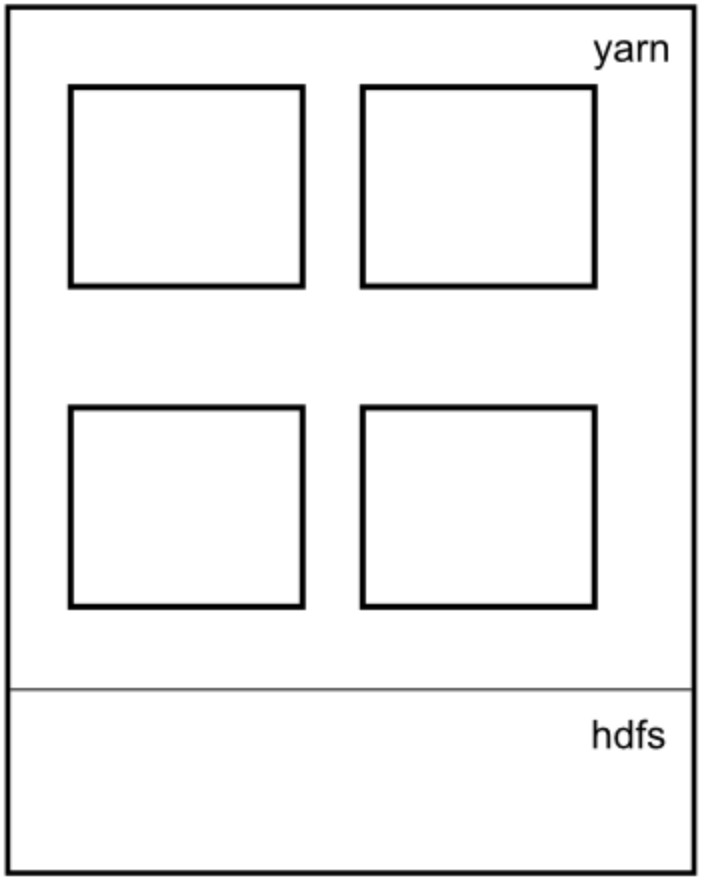
How does its works

```
a_rdd = sc.parallelize(a_list)
```

client

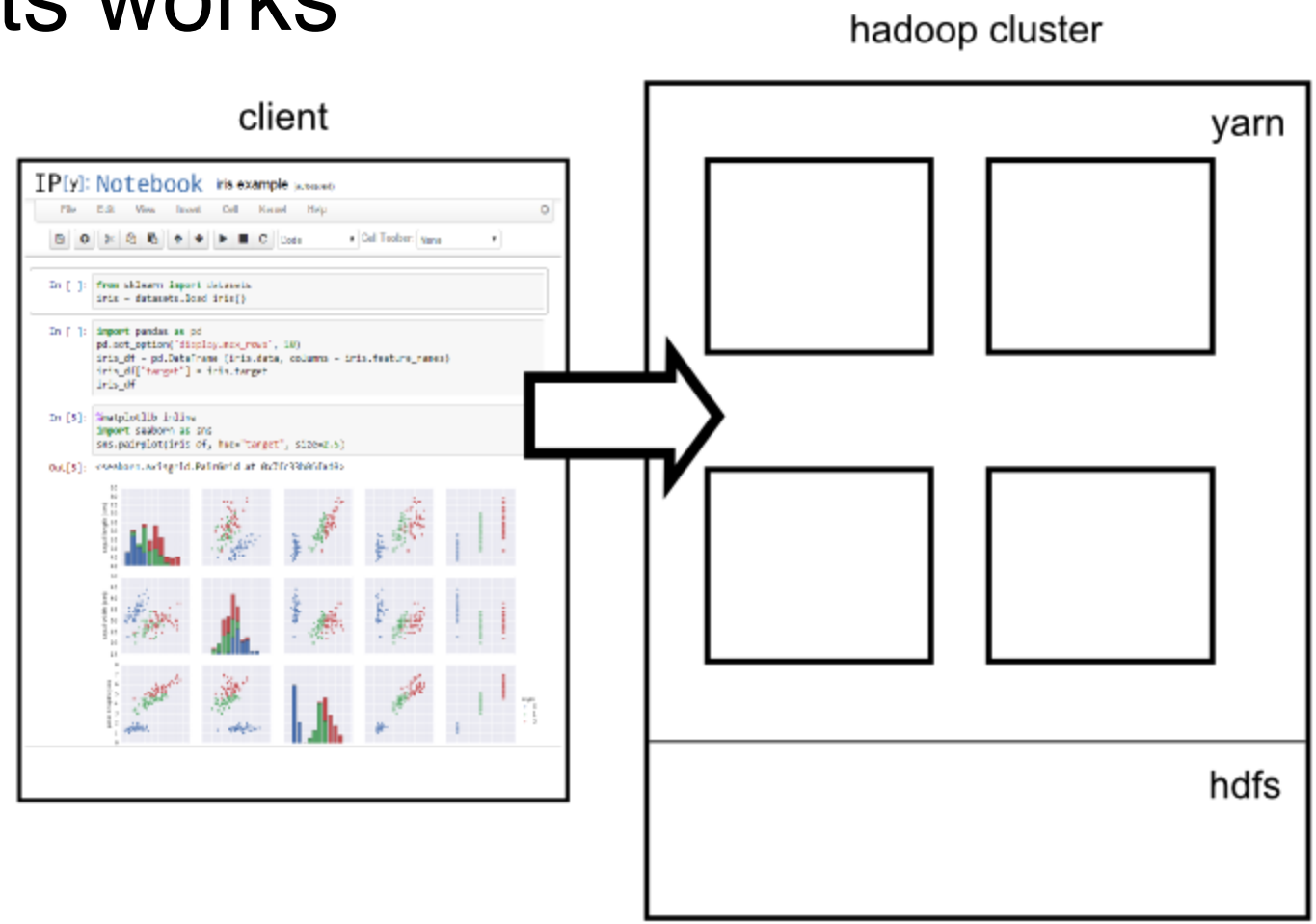


hadoop cluster



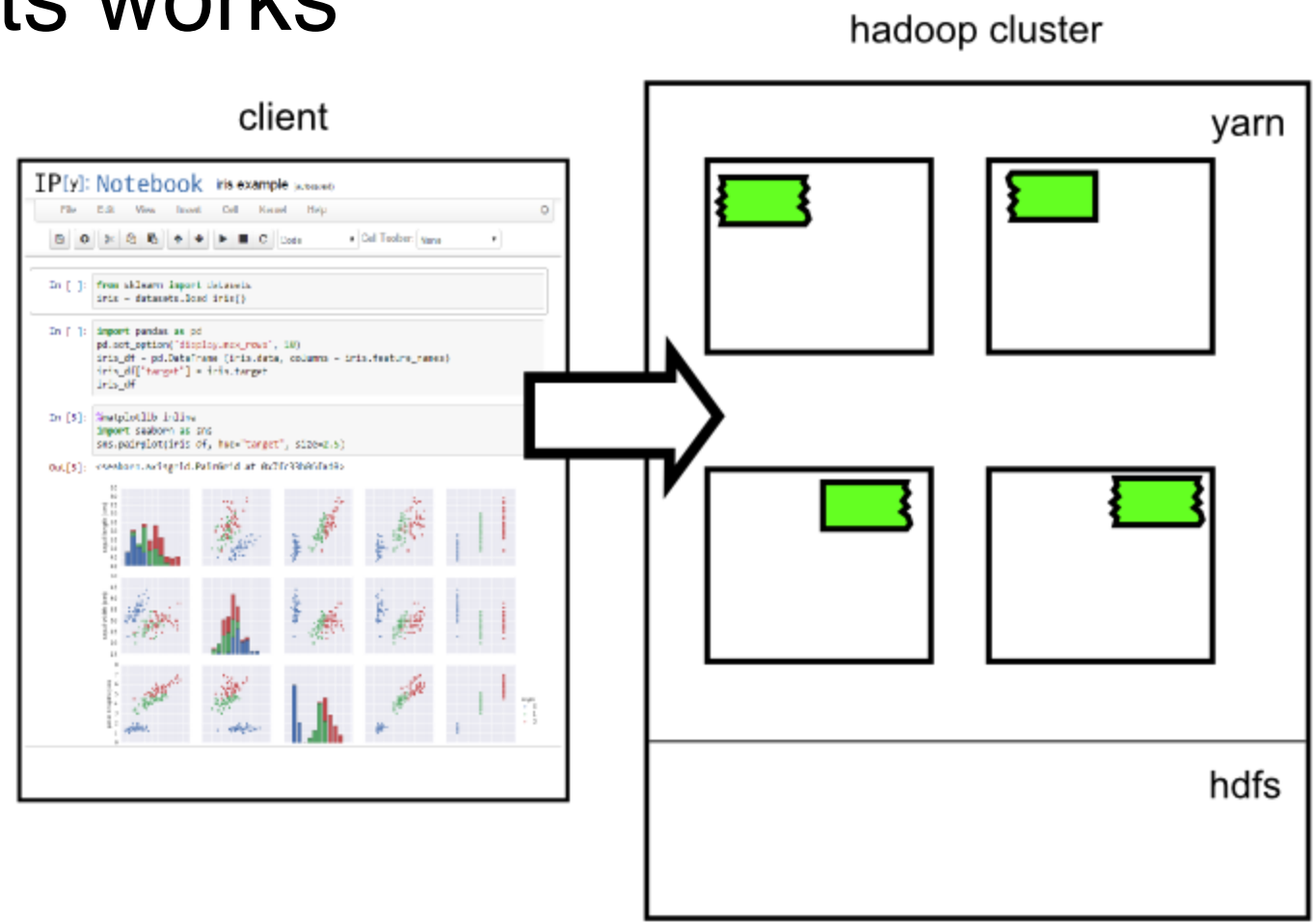
How does its works

```
a_rdd = sc.parallelize(a_list)
```



How does its works

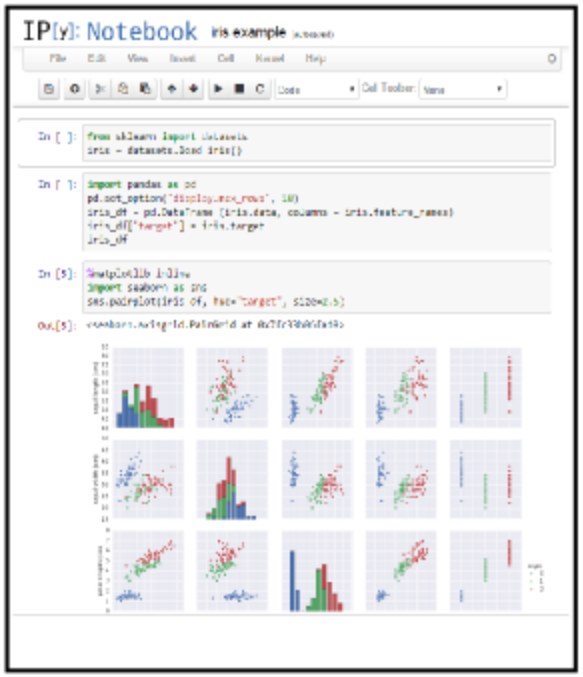
```
a_rdd = sc.parallelize(a_list)
```



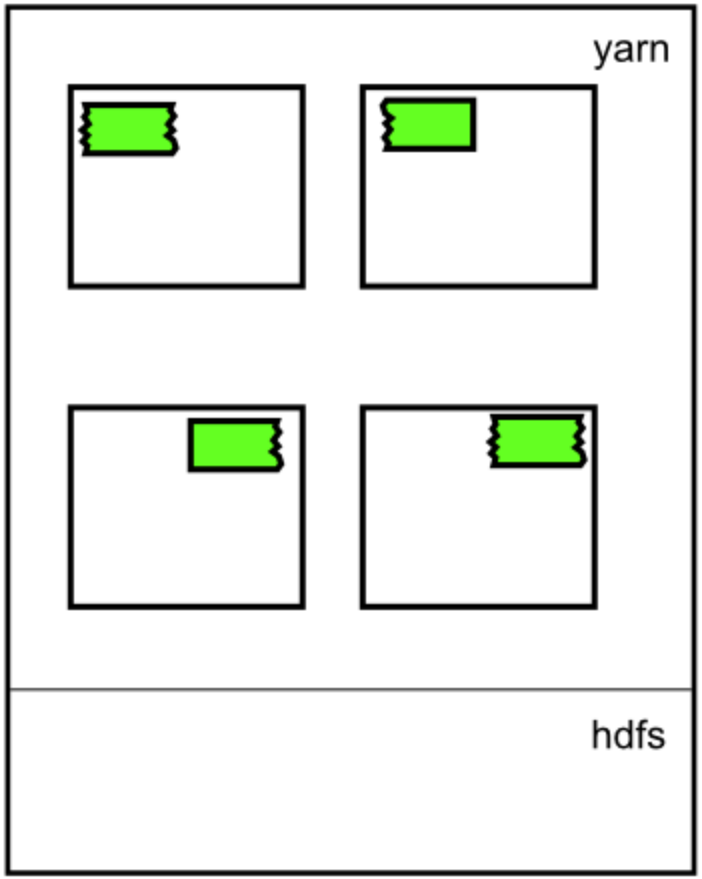
How does its works

```
a_rdd.saveAsTextFile("test.csv")
```

client



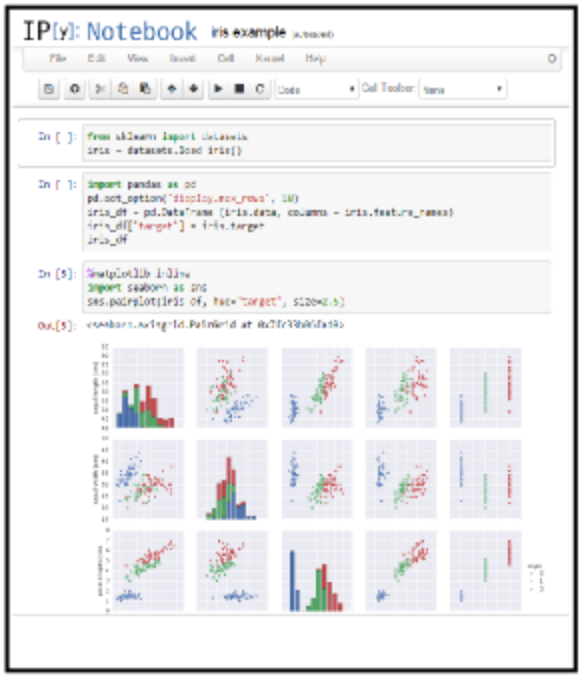
hadoop cluster



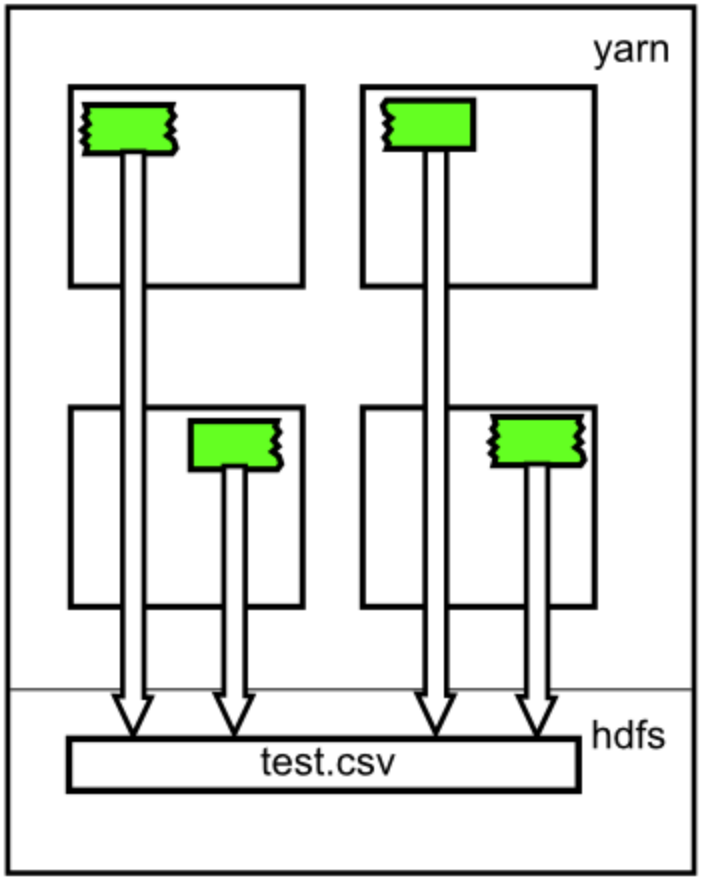
How does its works

```
a_rdd.saveAsTextFile("test.csv")
```

client



hadoop cluster



Apache Spark
in
Large Scale Machine Learning

LSML issues

- Too much samples to classify
- Training data does not fit in memory
- Too much training samples
- Too much models to train

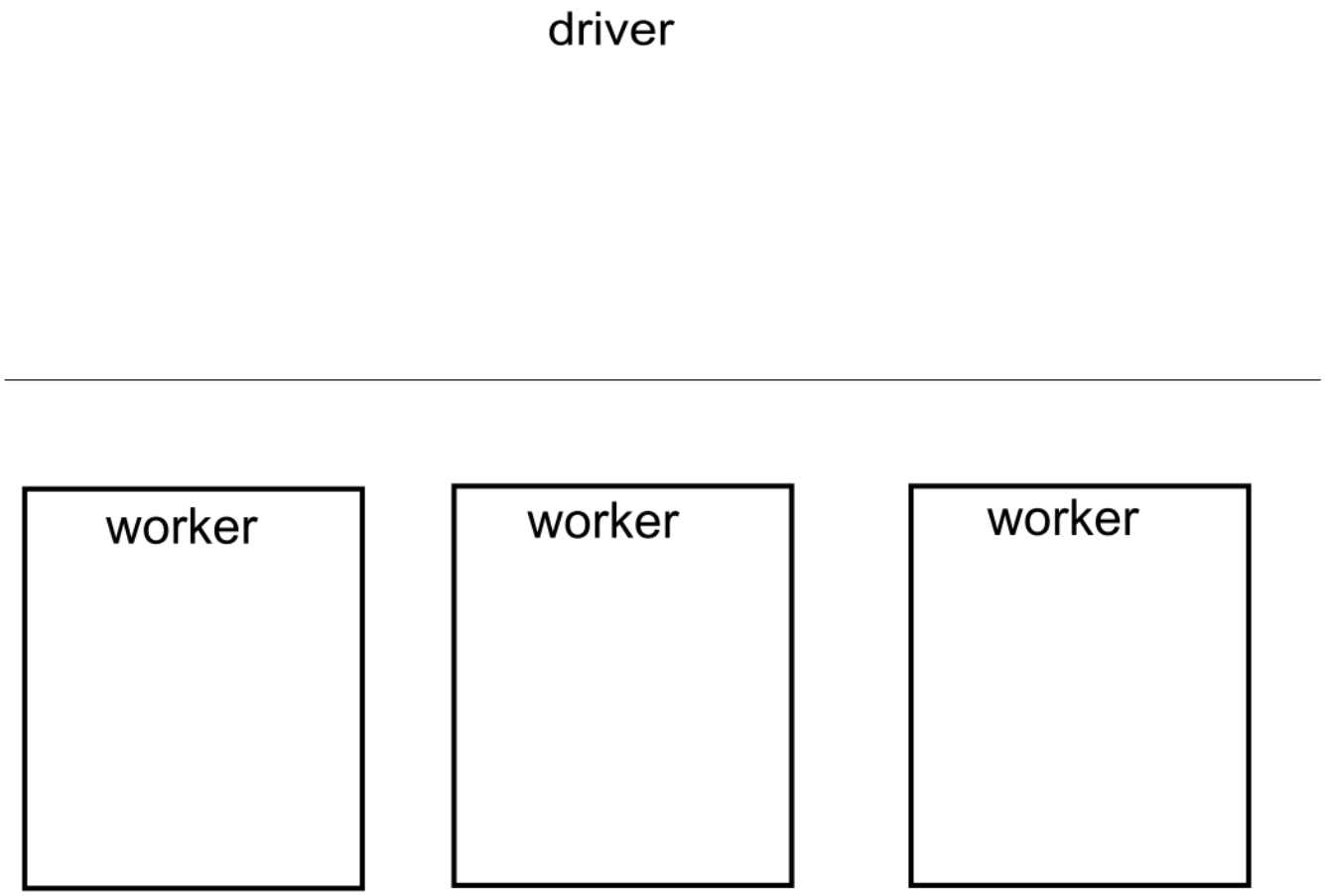
Why not MLLib?

- MLLib is less stable
- too few algorithms comparing to scikit-learn
- ML pipelines are not so mature than in scikit-learn
  - e. g. there is no simple way to use logistic regression for feature selectic
- MLLib python API falls behind Java/Scala API
- MLLib is actively developed and may be feasible choice in near future

Spark + scikit-learn = ?

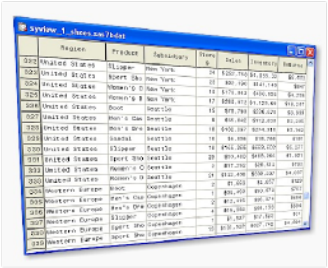
- Parallel training
  - meta-parameter grid search
  - parallel one-vs-rest for multi-class models
  - same features but different targets
  - parallel bagging and ensembles
  - parallel learning for multi-step classification
- Parallel prediction

Distributed learning



Distributed learning

dataset



driver

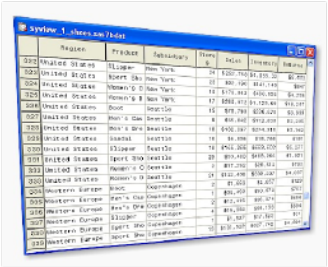
worker

worker

worker

Distributed learning

dataset



driver

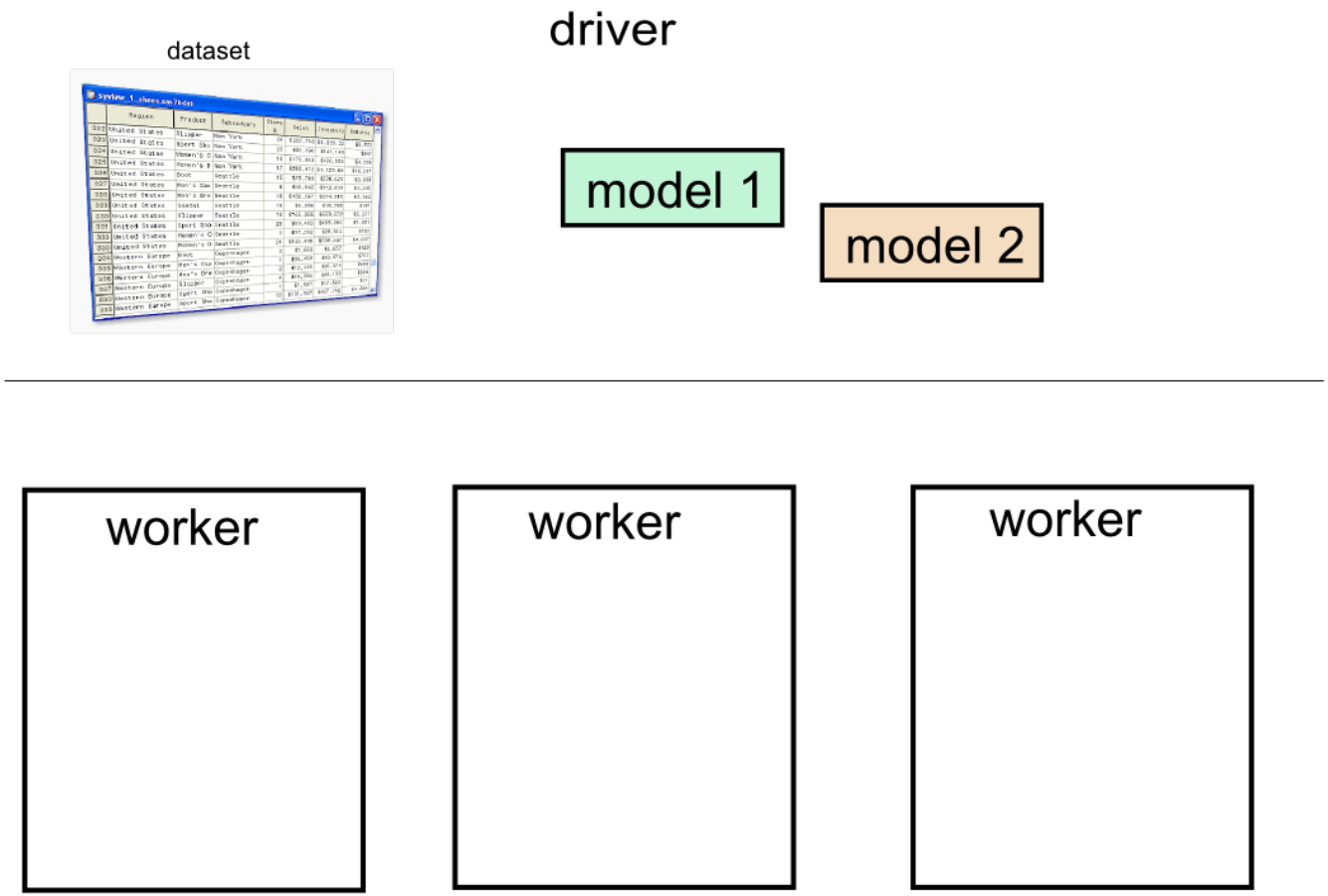
model 1

worker

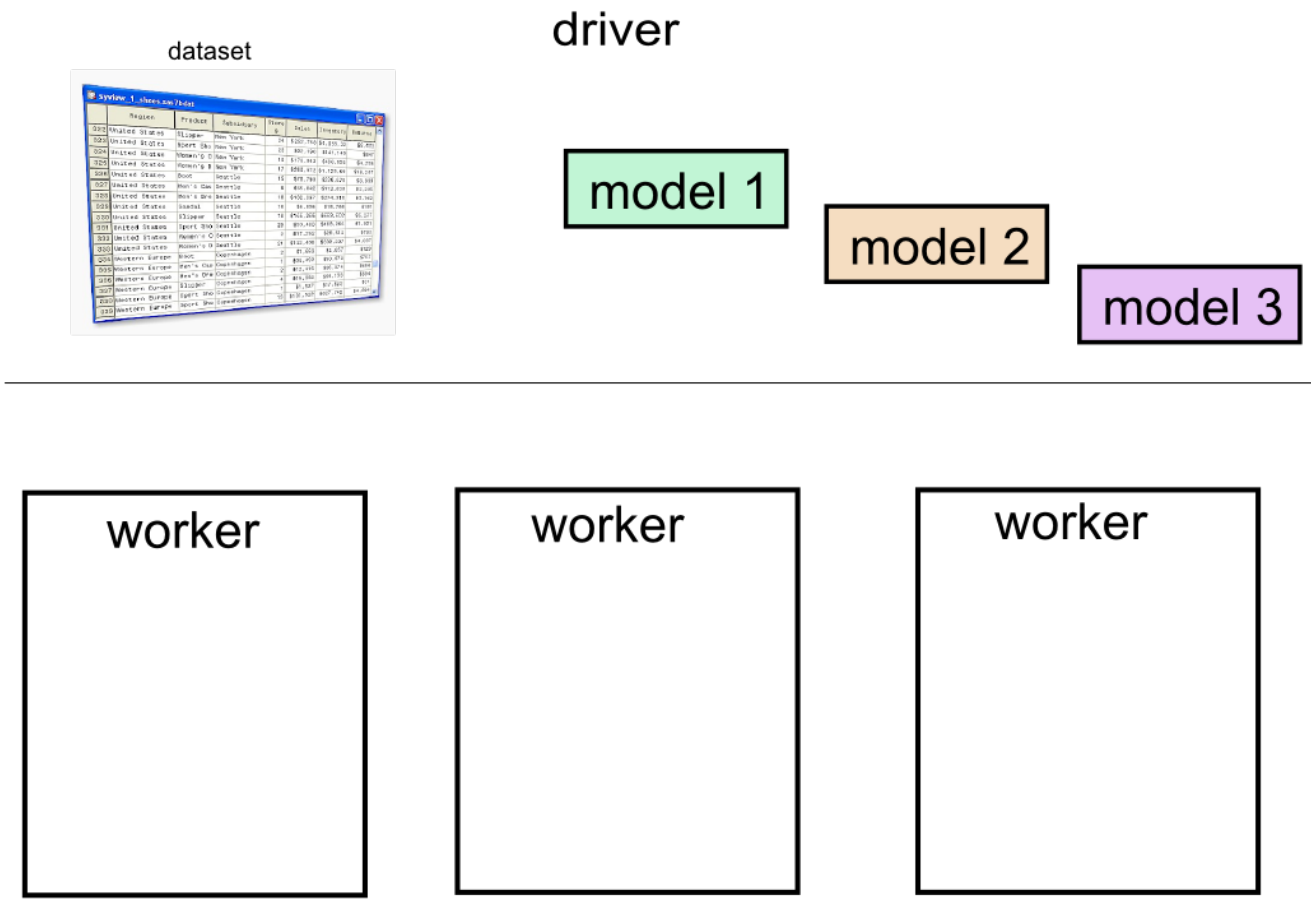
worker

worker

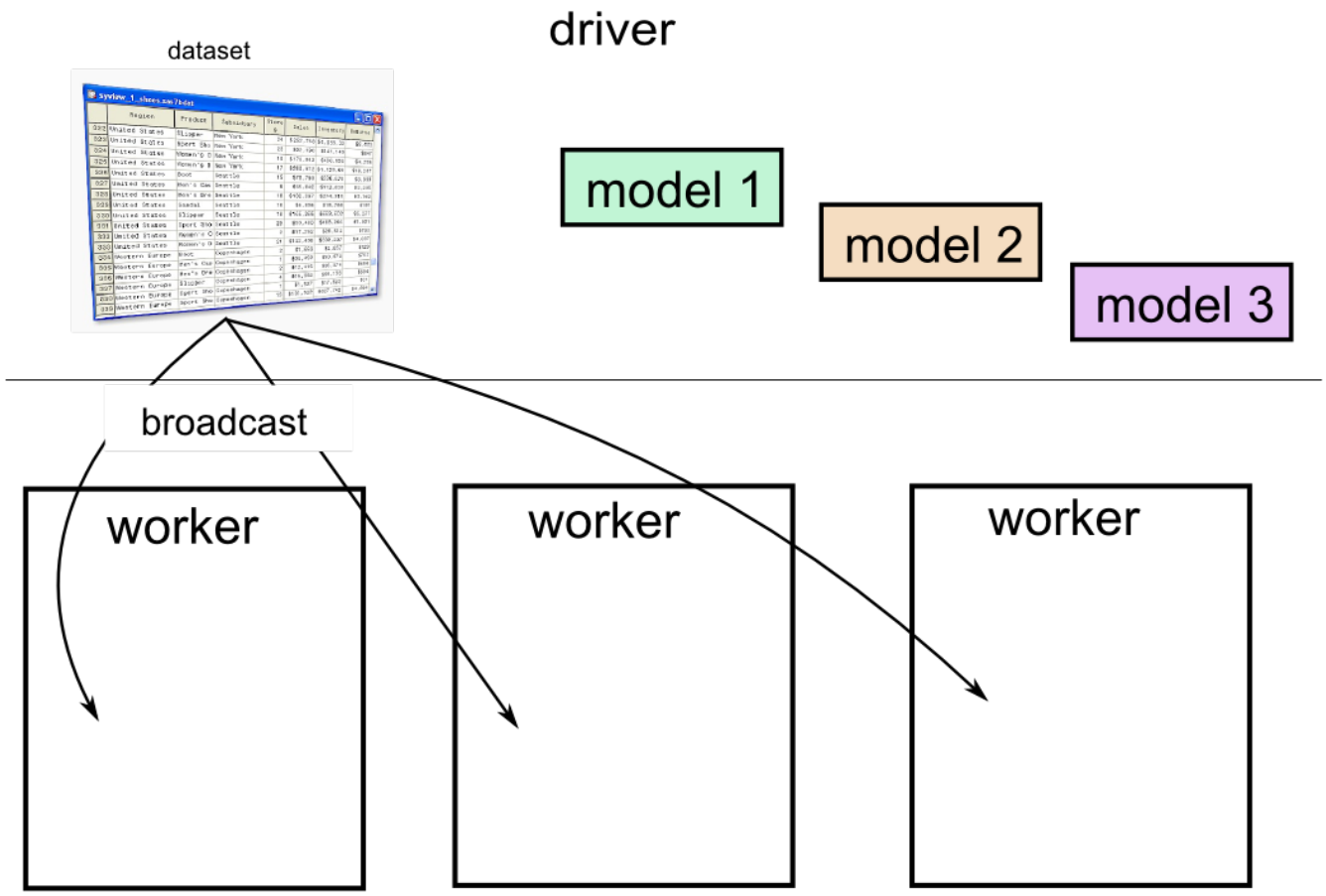
Distributed learning



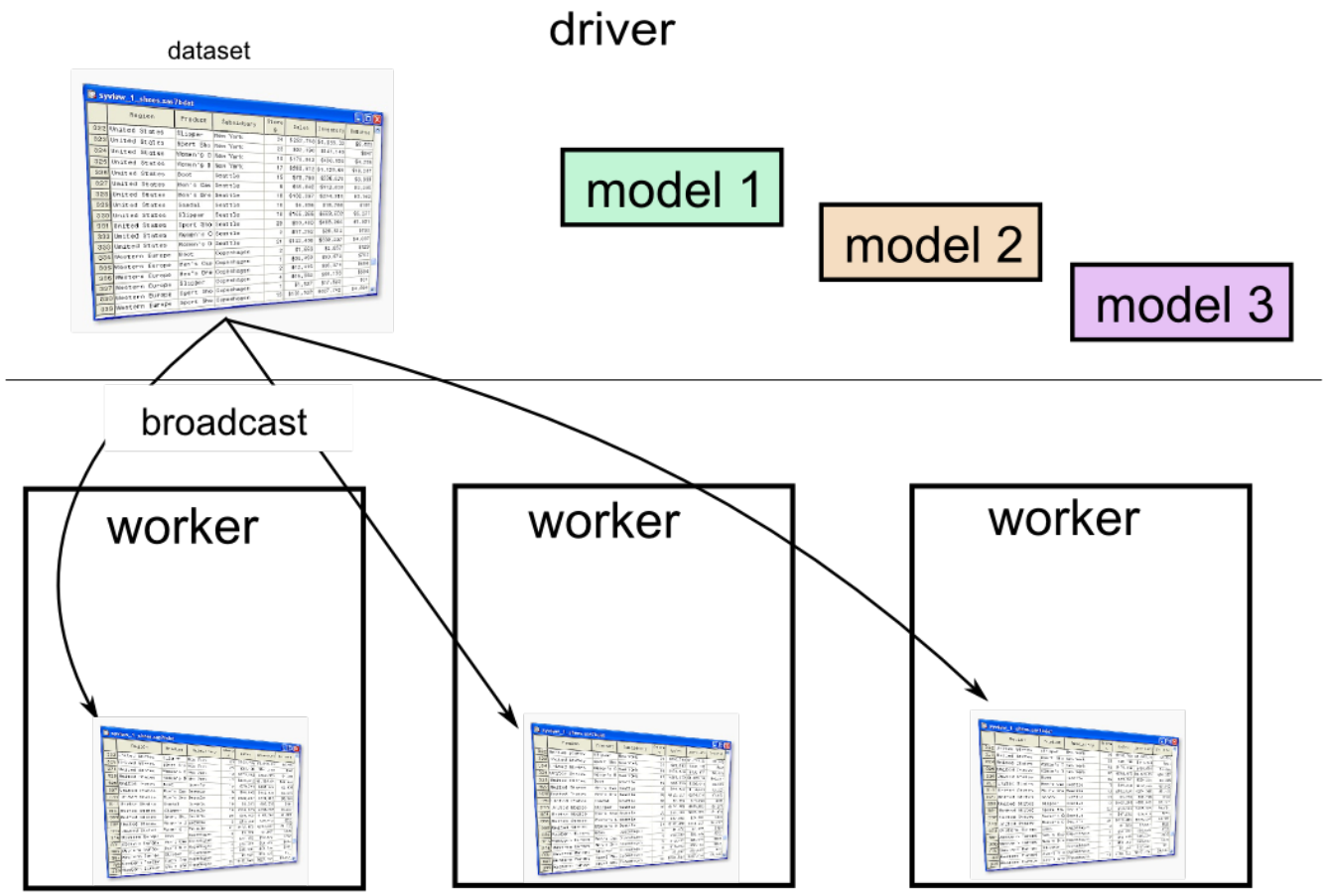
Distributed learning



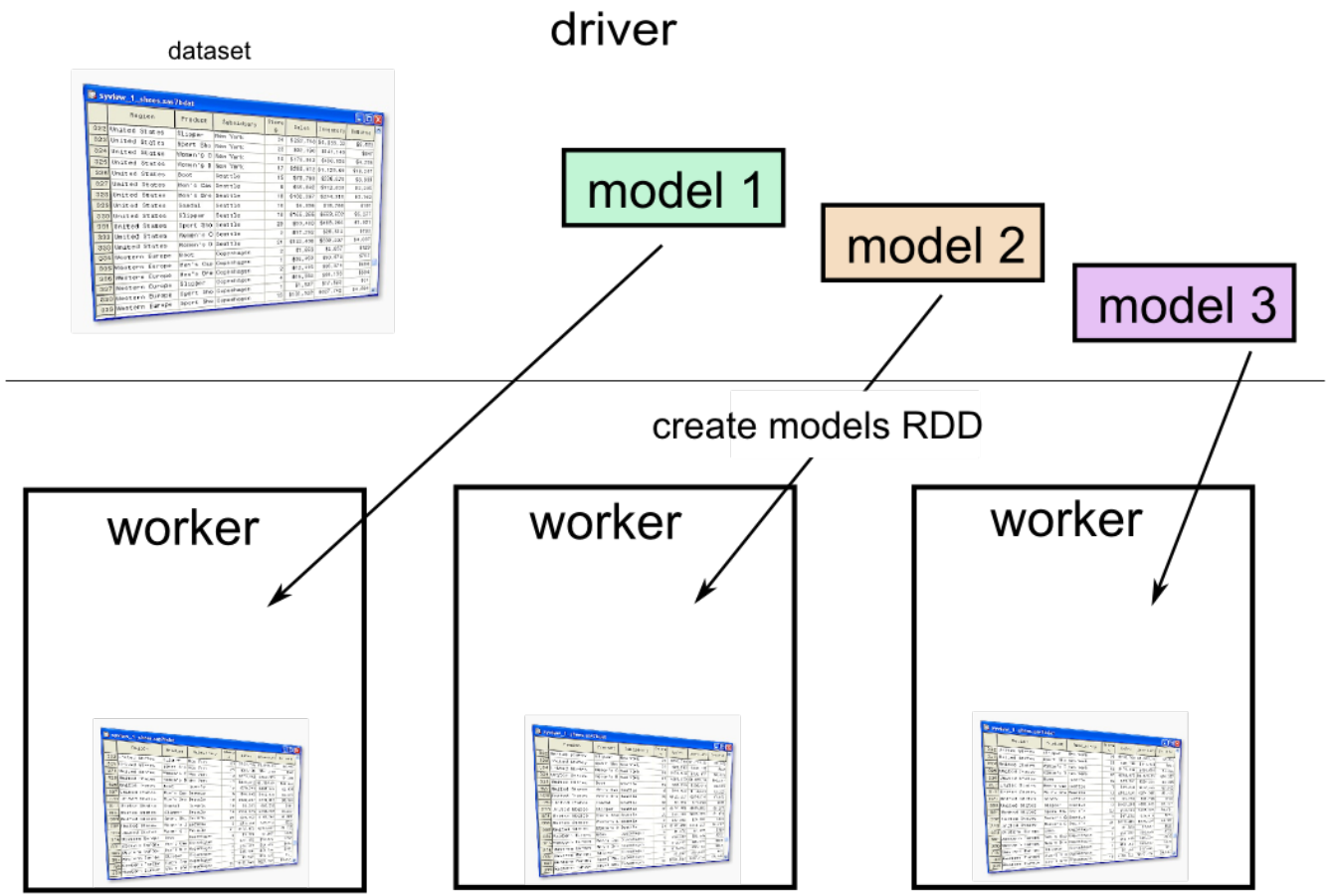
Distributed learning



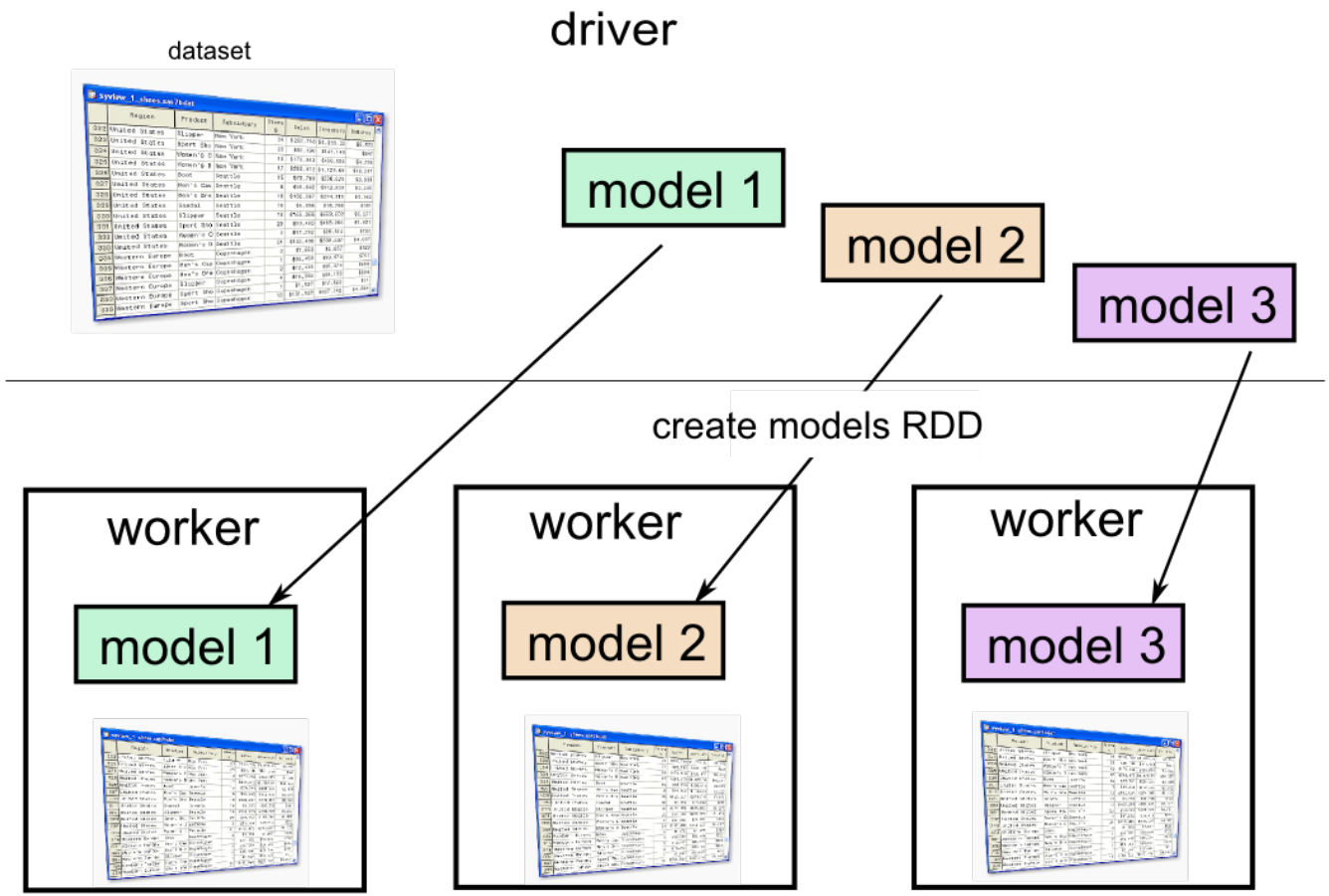
Distributed learning



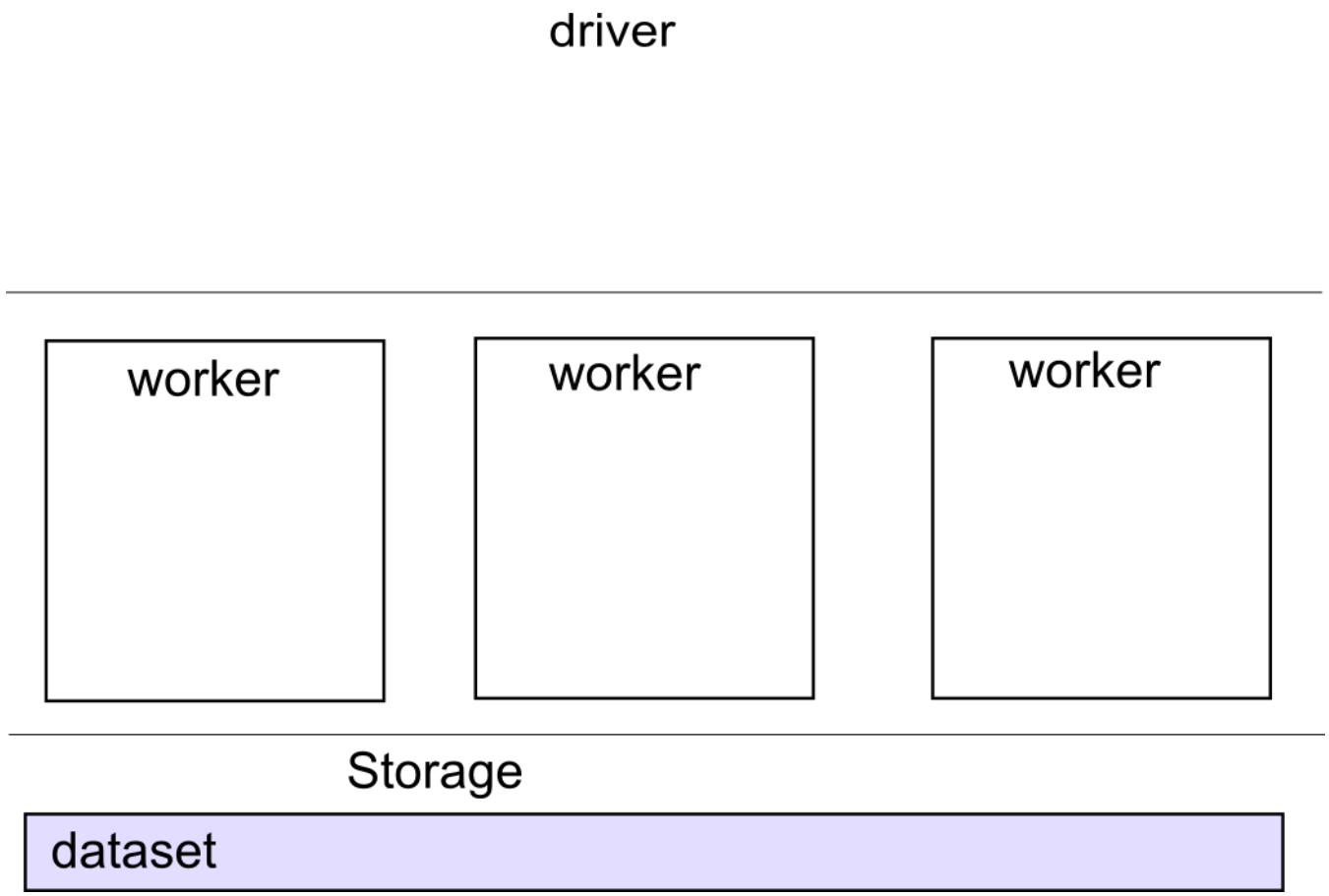
Distributed learning



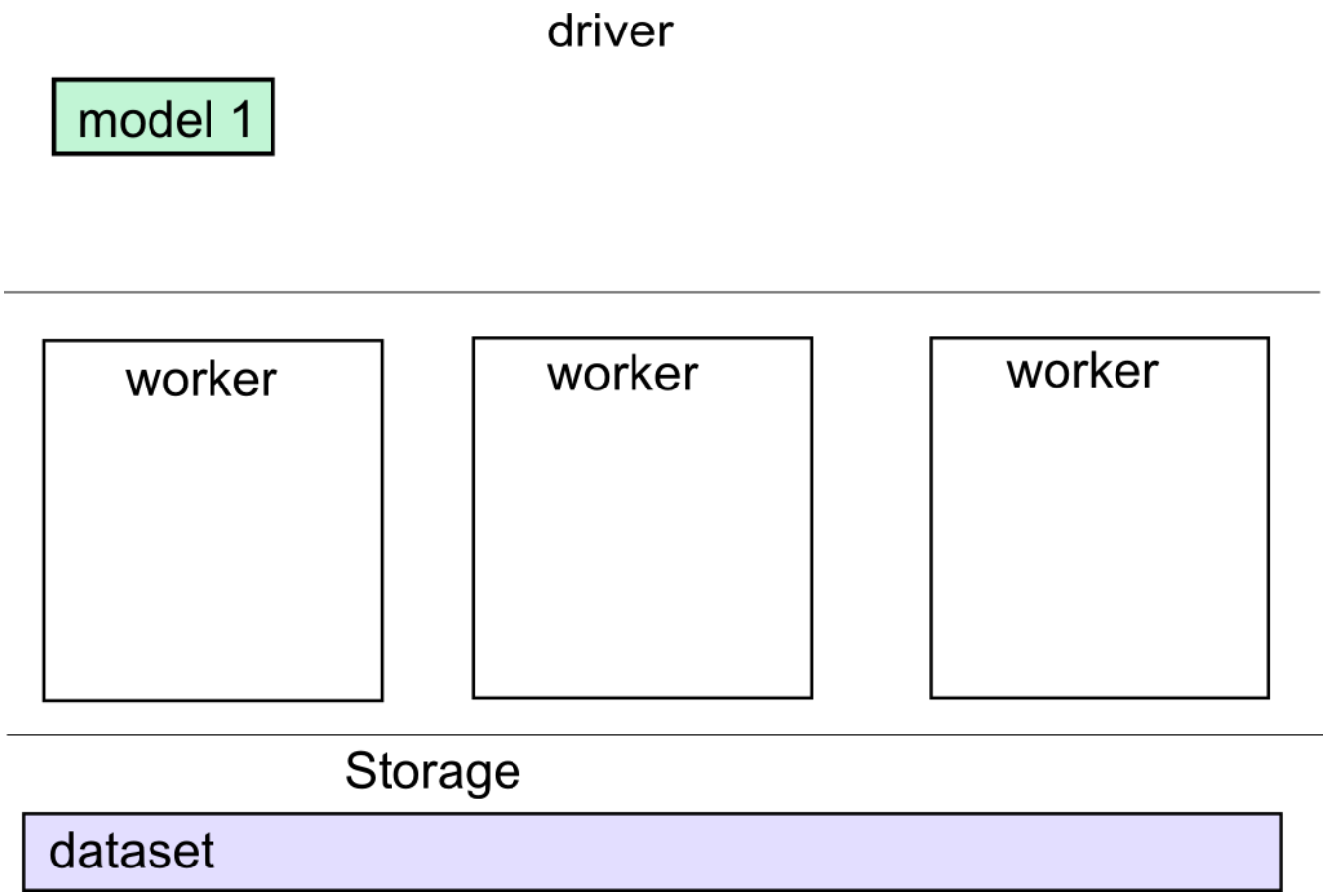
Distributed learning



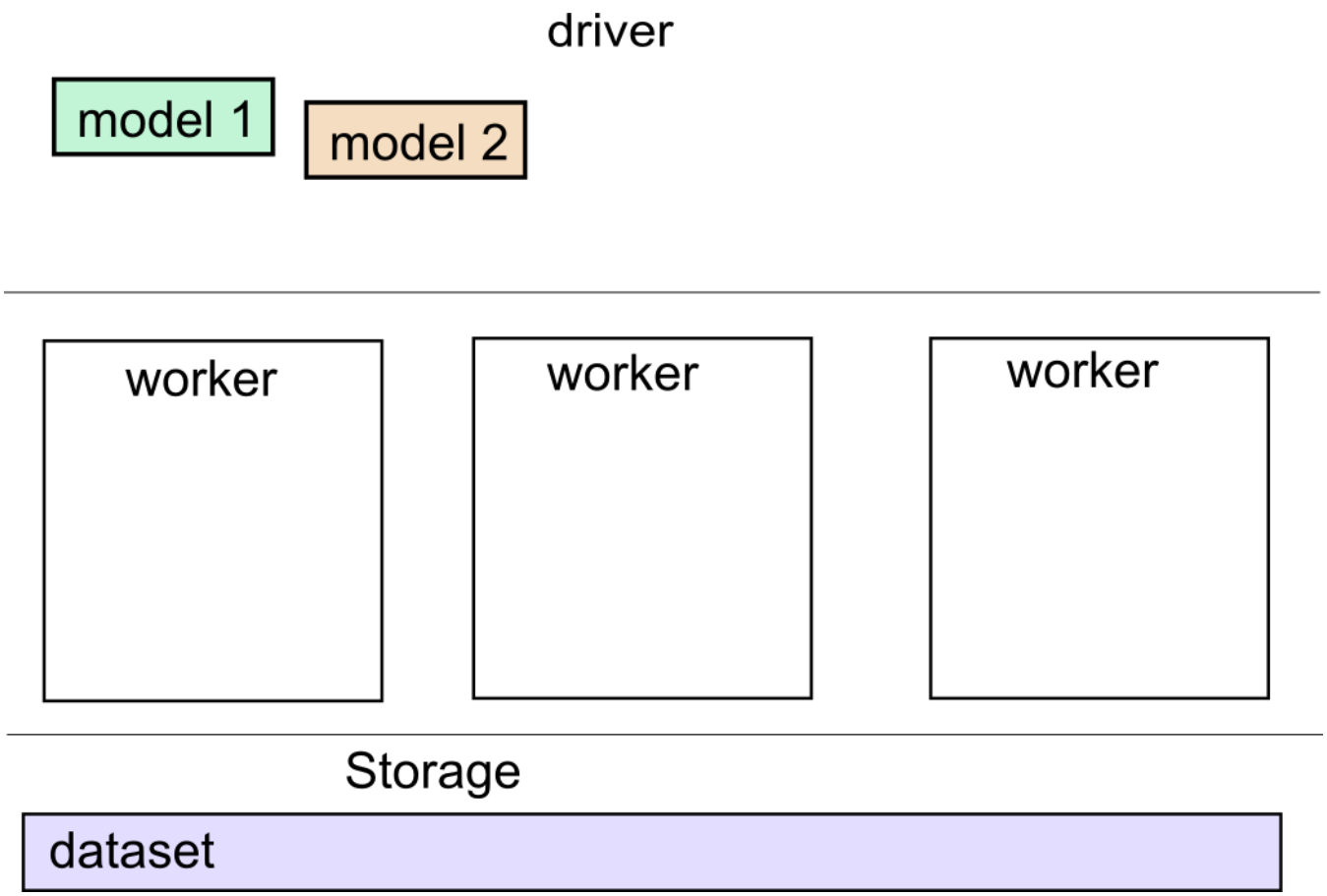
Distributed prediction



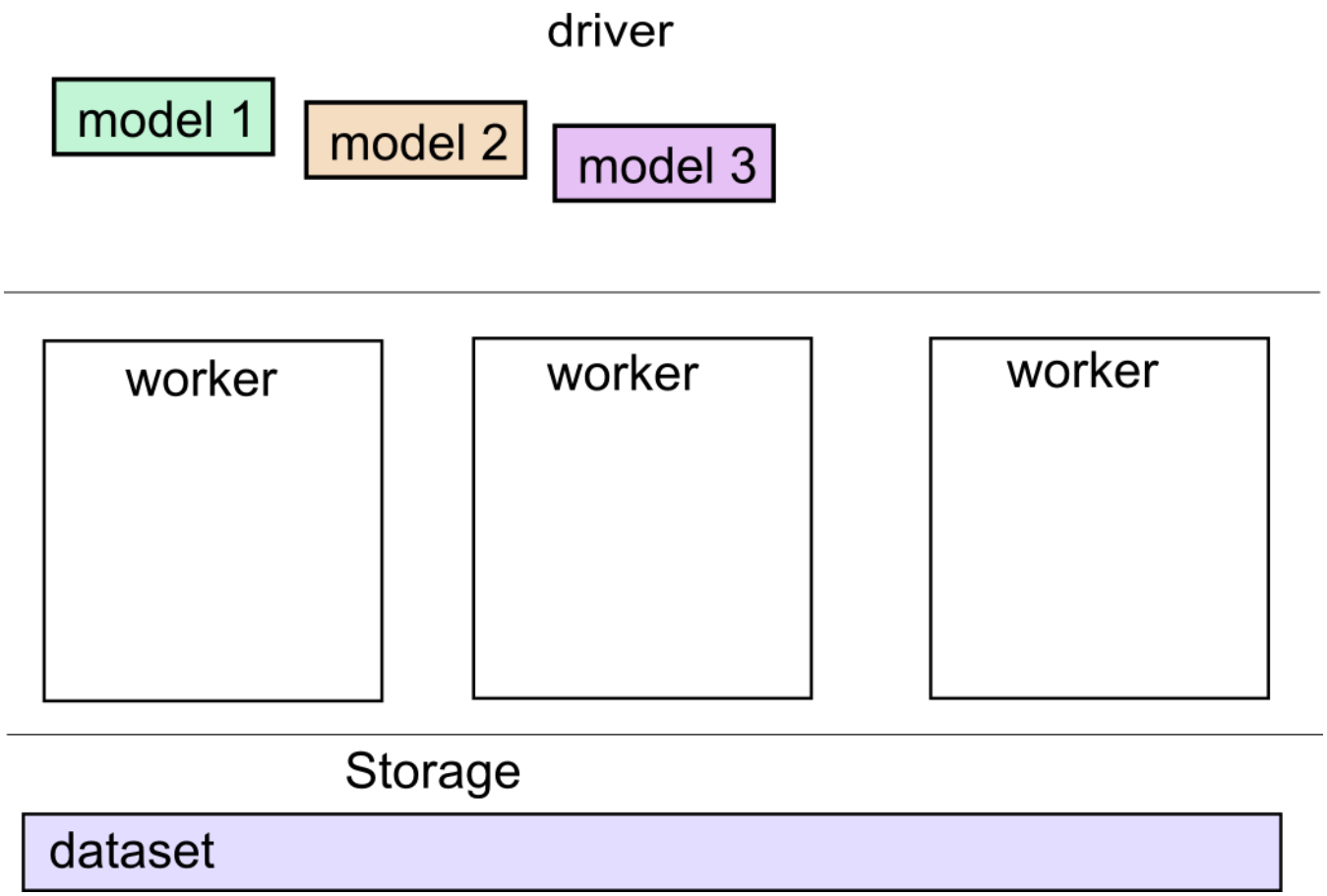
Distributed prediction



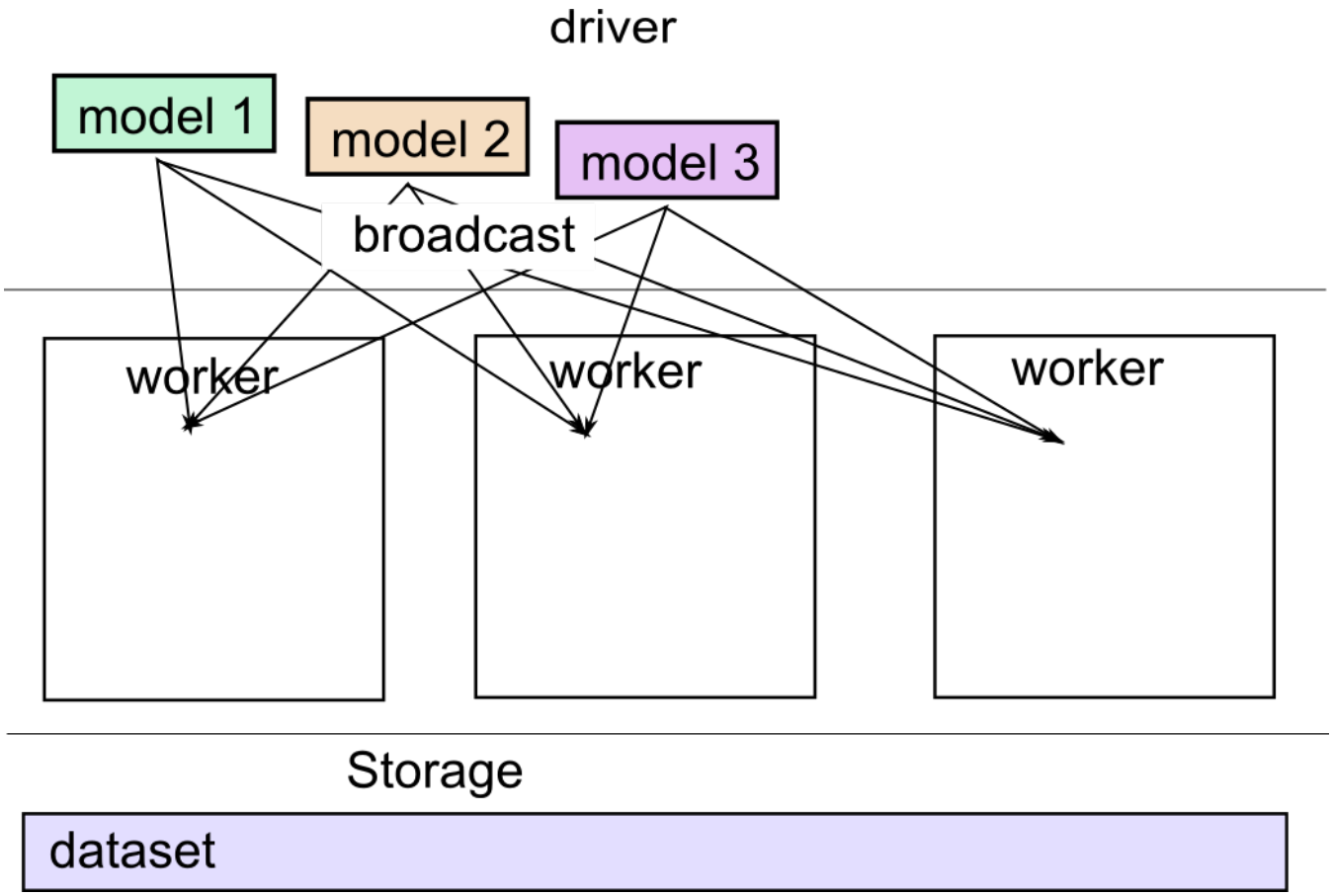
Distributed prediction



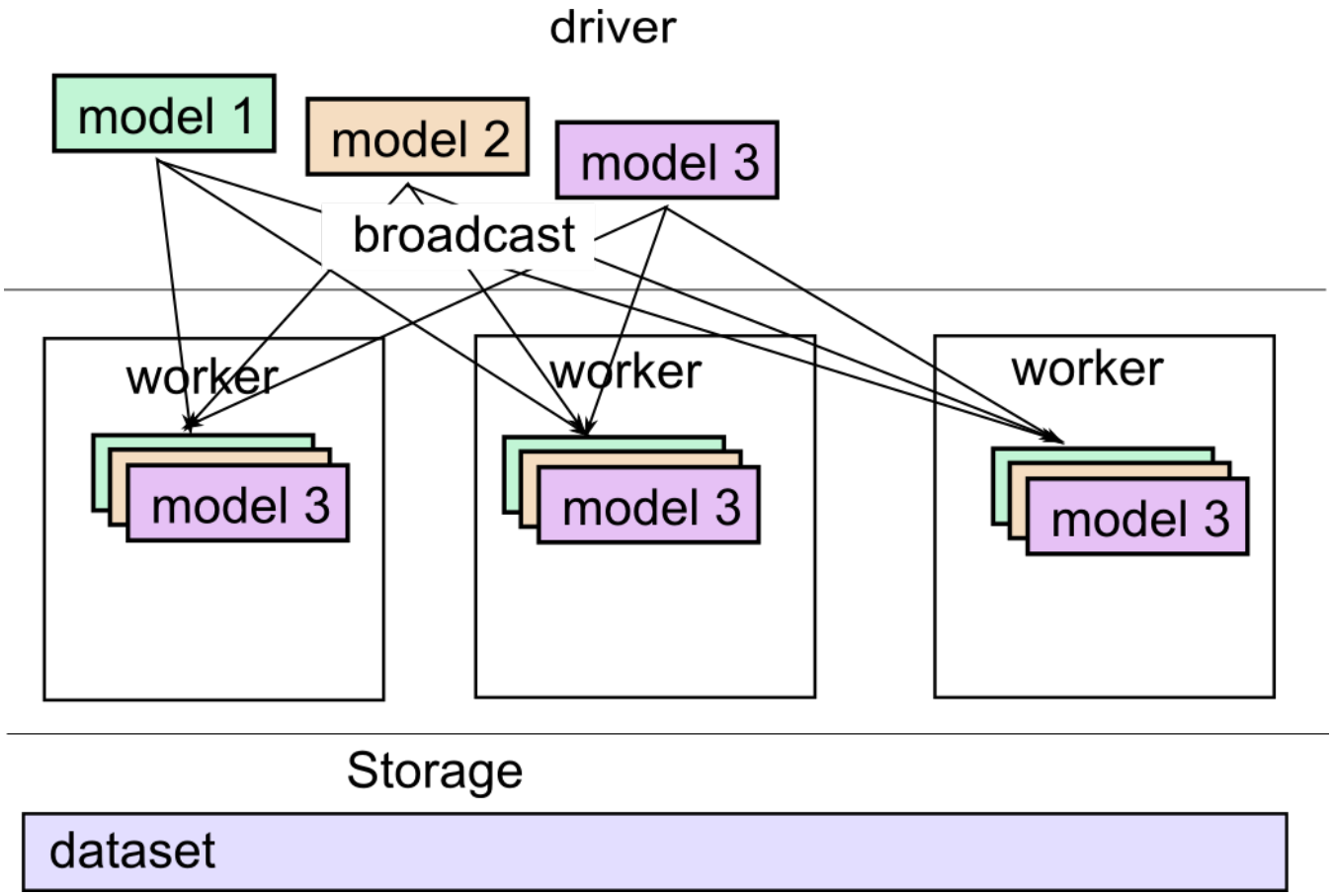
Distributed prediction



Distributed prediction



Distributed prediction



Distributed prediction