UNIVERSITÉ DE RENNES 1

UMR IRISA

Master SIF / DSL

Project presentation

Jean-Marc Jézéquel / Mathieu Acher

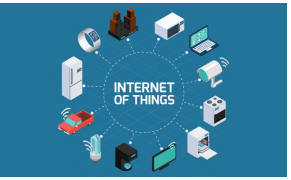
UNIVERSITÉ DE RENNES 1

UMR IRISA

JSON

CSV

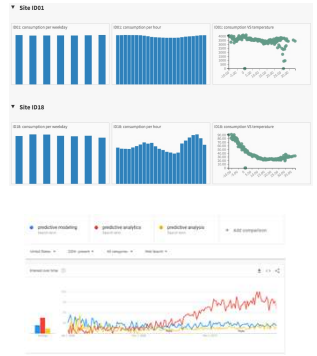
Machine learning



Web/IoT data

Tabular Data

Prediction model



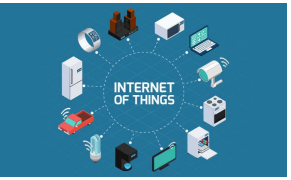
UNIVERSITÉ DE RENNES 1

UMR IRISA

JSON

CSV

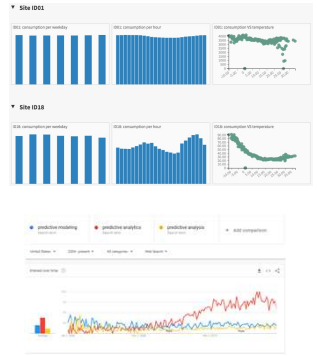
Machine learning



Web/IoT data

Tabular Data

Prediction model



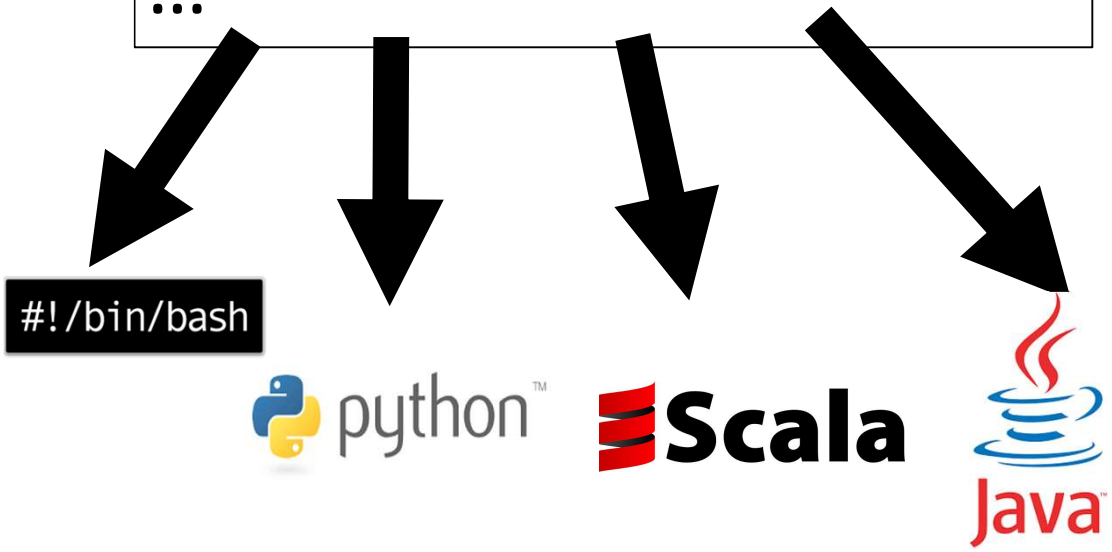
UNIVERSITÉ DE RENNES 1

UMR IRISA

Goals

- Finding functional/performance bugs
- Improve learning curve of APIs/libraries in a domain
- Choosing the good-enough/best morph for a given “task”
- Identifying good test suites/benchmarks for a domain
- Portoflio (“meta”) solution
  - eg can be run in parallel

Morph CSV
Morph JSON
Morph ML classification
Morph ML regression
…



UNIVERSITÉ DE RENNES 1

UMR IRISA

DSL for JSON

- Concepts
  - Load, Store JSON files
  - Select subset of objects, Projection (slice of objet)
    - Aka core relational algebra operators
  - Compute basic $\sum, \prod$ over fields
  - Print field value, #objects, #fields, depth, expressions…
  - Insert/modify/remove object/fields
- Services
  - Export to CSV
  - Interpreter
  - Compilers to (Java|Python|Julia) + JQ (for relevant subset)
    - https://stedolan.github.io/jq/

UNIVERSITÉ DE RENNES 1

UMR IRISA

DSL for CSV

- Concepts
  - Load, Store CSV files
  - Select subset of lines/column (cut)
    - Aka core relational algebra operators
  - Compute basic $\sum, \prod$ over fields
  - Print field value, #objects, #fields, expressions…
  - Insert/modify/remove lines/fields
- Services
  - Export to JSON
  - Interpreter
  - Compilers to (Java|Python|Julia) + bash (grep/cut/awk…)

Polymorphic CSV

```
f1 = "/tmp/test.csv”
n1 = #rows f1
```

```
wc –l /tmp/test.csv
```

```
#!/bin/bash
```

```
import csv
a = open('/tmp/test.csv', 'rt')
a_read = csv.reader(a)
print(sum(1 for row in a_read))
```

python

```
public static void processCsv(Reader iCvs, Writer oCvs, String COL_NAME_SUM) throws IOException {
  CSVPrinter printer = null;
  try {
    printer = new CSVPrinter(oCvs, CSVFormat.DEFAULT.withRecordSeparator(NL));
    List<String> oCvsHeaders;
    List<String> oCvsRecord;
    CSVParser records = CSVFormat.DEFAULT.withHeader().parse(iCvs);
    Map<String, Integer> irHeader = records.getHeaderMap();
    oCvsHeaders = new ArrayList<String>(Arrays.asList((irHeader.keySet()).toArray(new String[0])));
    oCvsHeaders.add(COL_NAME_SUM);
    printer.printRecord(oCvsHeaders);
    for (CSVRecord record : records) {
      oCvsRecord = record2list(record, oCvsHeaders, COL_NAME_SUM);
      printer.printRecord(oCvsRecord);
    }
  }
  finally {
    if (printer != null) {
      printer.close();
    }
  }
  return;
}
```

Java

UNIVERSITÉ DE RENNES 1

UMR IRISA

DSL for ML Classification

- Concepts
  - https://en.wikipedia.org/wiki/Statistical_classification
    - Typically uses CSV file as input eg *https://en.wikipedia.org/wiki/Iris_flower_data_set*
  - Evaluation strategy either:
    - dataset is split in two (training/test), with user defined % training
    - cross-validation (provide means to parameterize it)
  - Predictive variables and target variable can be specified
    - By default, all variables are predictive except last column of the CSV (target)
  - Specify what to calculate: accuracy and/or recall and/or f1
  - Which algorithm(s) to use
    - e.g., classification tree or SVM for scikit-learn
- Services
  - Interpreter (ie classification using « random » algorithm)
    - Useful to provide baselines
  - Compilers to Python/scikit-learn + (R | Julia)

UNIVERSITÉ DE RENNES 1

UMR IRISA

Example of scikit-learn code to be generated

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.metrics import accuracy_score
# Using pandas to import the dataset
df = pd.read_csv("iris.csv")

# Learn more on pandas read_csv :
#    https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html
# pandas input in general :
#     https://pandas.pydata.org/pandas-docs/stable/reference/io.html

# Spliting dataset between features (X) and label (y)
X = df.drop(columns=["variety"])
y = df["variety"]

# pandas dataframe operations :
#     https://pandas.pydata.org/pandas-docs/stable/reference/frame.html

# Spliting dataset into training set and test set
test_size = 0.3
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=test_size)

# scikit-learn train_test_split :
#     https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
# Other model selection functions :
#     https://scikit-learn.org/stable/modules/classes.html#module-sklearn.model_selection

# Set algorithm to use
clf = tree.DecisionTreeClassifier()

# scikit-learn DecisionTreeClassifier :
#     https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier
# Other scikit-learn tree algorithms :
#     https://scikit-learn.org/stable/modules/classes.html#module-sklearn.tree
# Use the algorithm to create a model with the training set
clf.fit(X_train, y_train)

# Compute and display the accuracy
accuracy = accuracy_score(y_test, clf.predict(X_test))
print(accuracy)
```

UNIVERSITÉ DE RENNES 1

UMR IRISA

DSL for ML Regression

- Concepts
  - Evaluation strategy either:
    - dataset is split in two (training/test), with user defined % training
    - cross-validation (provide means to parameterize it)
  - Predictive variables and target variable can be specified
    - By default, all variables are predictive except last column of the CSV (target)
  - Specify what to calculate: mean relative error…
  - Which algorithm(s) to use
    - e.g., regression tree or SVM for scikit-learn
- Services
  - Interpreter (ie regression using « random » algorithm)
    - Useful to provide baselines
  - Compilers to Python/scikit-learn + (R | Julia)

Tasks

- Choose a sub project among JSON/CSV/ML-classif/ML-reg (Now)
  - Work in groups of up to 2 -> each subproject should be taken at least once
    - Working alone is still possible
    - Insert group composition into spreadsheet by Sept. 13th
      - https://bit.ly/3q3cskW
- Build a first version of your metamodel (sept 15th)
  - scan of hand written diagram or pdf is good enough at that stage
  - Be ready to present it on Sept. 22th
- Build concrete syntax + parser
  - Be ready to present it on Oct. 3th
- Build interpreter
- Build compiler #1
- Build compiler #2
- Make sure to interoperate with 2 complementary sub-projects
  - Show test case, if ok bonus for all 3 teams.