

Инструмент для сравнения Jupyter ноутбуков в интегрированной среде разработки DataSpell

Отчет по производственной практике
Божнюк Александр Сергеевич
группа 19.Б11-мм

Консультант: DataSpell разработчик А. В. Вокин
Научный руководитель: ст. преп. С.Ю. Сартасов

Введение

- В наше время все большую популярность набирают ноутбуки Jupyter;
- Компанией JetBrains была создана IDE для Data Science и работы с ноутбуками Jupyter - DataSpell;
- В IntelliJ Platform существует инструмент для сравнения файлов (Diff Tool), который хорошо справляется со своей задачей;
- Однако в случае сравнения ноутбуков Jupyter есть проблема: отображается результат сравнения для JSON представления, вместо обычного представления ноутбуков. Это неудобно для пользователя и приводит к проблемам при работе с инструментом.

Проблема

Clipboard vs Editor

Side-by-side viewer Do not ignore Highlight words 34 differences

YouTrack Assignee Prediction.py (/Users/anastasia.miller/Downloads)

```
381 457 # ('tfidf', TfidfTransformer(sublinear_tf=True, norm='l2')) ✓
382 458 # ], verbose=True)),
383 459
384 468 # ('developer_numbers', Pipeline([
385 461 # ('developer_numbers', developer_number_feature_extractor),
386 462 # ], verbose=True)),
387 463 ('issue_features', Pipeline([
388 464 ('issue_features', issue_features),
389 465 ('scale', StandardScaler())
390 466 ]), verbose=True)),
391 467 # ('issue_features', Pipeline([
392 468 # ('json_to_text', IssueToTextTransformer()),
393 469 # ('vectorize_text', CountVectorizer(analyzer=split,
394 # vocabulary=KEYWORDS)),
395 470 # ('add_normalized_counts', FeatureUnion([
396 471 # ('id', FunctionTransformer(validate=True,
397 # accept_sparse=True)),
398 472 # ('frequency', TfidfTransformer(use_idf=False))
399 473 # ])),
400 474 # ('densify', DenseTransformer()),
401 475 # ('scale', StandardScaler())
402 476 # ], verbose=True)),
403 477
404 478 # ('correlation_features', PPipeline([
405 479 # ('correlation_features', correlation_feature_extractor),
406 480 # ('scale', StandardScaler())
407 481 # ], verbose=True))
408 482
409 483 # ], verbose=True)),
410 484 ('clf', cclf),
411 485
412 486
413 487
414 488
415 489
416 490
417 491
418 492
419 493
420 494
421 495
422 496
```

evaluate_model()
right_dev = 0
res = clf.predict_proba(test_x)
count = 0
result = []
for expected, actual in zip(res, test_y):
proba_1 = expected[1]

Цели и задачи

Целью данной работы является создание инструмента сравнения для ноутбуков Jupyter в интегрированной среде разработки DataSpell.

Для достижения цели были поставлены следующие задачи:

1. Провести обзор существующих подходов к алгоритмам сравнения;
2. Провести обзор конкурентов, имеющих свою реализацию инструмента сравнения файлов;
3. Провести обзор платформы IntelliJ;
4. Реализовать инструмент сравнения файлов (Diff Tool) для Jupyter ноутбуков и интегрировать его в DataSpell. При этом разобраться с такими сопутствующими задачами, как сравнение выводов ячеек и сравнение ячеек с кодом Markdown;
5. Реализовать инструмент слияния (Merge Tool) для интеграции с системами контроля версий.

Обзор конкурентов

Вопросы, которые были рассмотрены во время обзора конкурентов:

- Сравнение ячеек с кодом на языке Markdown;
- Сравнение выводов ячеек;
- Дополнительные возможности.

Рассмотренные конкуренты:

- Библиотека nbtime;
- Visual Studio Code;
- DagsHub;
- Curvenote;
- ReviewNB for GitHub.

Обзор конкурентов: результаты

- Сравнение ячеек с кодом Markdown: отключение обработки текста и последующее сравнение содержимого;
- Сравнение выводов ячеек: принцип было/стало;
- Дополнительные возможности: сравнение метаданных и свертка ячеек, в которых не произошло изменений.

Обзор платформы IntelliJ

Платформа IntelliJ - проект, предоставляющий инфраструктуру для разработки IDE.

Составляющие платформы IntelliJ, которые необходимы для дальнейшего хода работы:

- **Документ (Document)**: позволяет работать с содержимым файла как с обычным текстом;
- **Редактор (Editor)**: настройка и использование различной функциональности, связанной с IDE;
- **Точки расширения (Extension Points)**: расширение функциональности одних плагинов другими.

Основная идея

- С каждым файлом ноутбука связаны 2 документа;
- Первый: содержит JSON-представление;
- Второй: содержит “ноутбучное” представление;
- На вход инструмента сравнения приходит первый документ - нужно подменить его на второй.

```
#%% md
***Божнюк Александр Сергеевич, 271 группа***
#%%
from math import sin, cos
import matplotlib.pyplot as plt
```


Инструмент сравнения платформы

- **DiffRequest** - запрос, который содержит в себе информацию для сравнения;
- **DiffViewer** - компонент визуализации результатов сравнения;
- **DiffContext** - контекст, хранящий в себе информацию о текущем проекте и настройках;
- **DiffTool**: на основе **DiffContext** и **DiffRequest** создает **DiffViewer**.
- **Основная цель**: подмена запроса, чтобы в нем хранились нужные документы;
- Для этого был реализован свой **DiffTool: JupyterDiffTool**.

Результат подмены

Boundary Value Problem17.ipynb (/Users/Alexander.Bozhnyuk/projects/notebooks/notebooks)
Божнюк Александр Сергеевич, 2/1 группа

2

Задание:

```
Sy'' + 2xy'-y = 2(x^2+1)cos(x)$  
$0 <= x <= 0.5$  
$y(0) = 1111111111$  
$y(0) = 10000000$  
$y(0.5) = 0.5sin(0.5)$
```

4

Код решения (Python3)

```
from math import sin, cos  
import matplotlib.pyplot as plt  
  
def bvp(N, a, b, y0, yN, p, q, f):  
    h = (b - a) / N  
    x = [a + k * h for k in range(0, N + 1)]  
    L = [-1, 0] # we don't use L[0]  
    K = [-1, y0] # we don't use K[0]  
    # L[k] and K[k] evaluation  
    for j in range(2, N + 1):  
        ap = 1 - p(x[j - 1]) * h / 2  
        bp = h * h * q(x[j - 1]) - 2  
        cp = 1 + p(x[j - 1]) * h / 2  
        fp = h * h * f(x[j - 1])  
        lc = -cp / (ap * L[j - 1] + bp)  
        kc = (-ap * K[j - 1] + fp) / (ap * L[j - 1] + bp)  
        L.append(lc)  
        K.append(kc)  
    # y[k] evaluation  
    y = [yN]
```

Boundary Value Problem16.ipynb (/Users/Alexander.Bozhnyuk/projects/notebooks/notebooks)
Божнюк Александр Сергеевич, 271 группа

2

Задание:

```
Sy'' + 2xy'-y = 2(x^2+1)cos(x)$  
$0 <= x <= 0.5$  
$y(0) = 0$  
$y(0.5) = 0.5sin(0.5)$
```

4

Код решения (Python3)

```
In 1 1 from math import sin, cos  
2 import matplotlib.pyplot as plt  
  
In 2 1 def bvp(N, a, b, y0, yN, p, q, f):  
2 h = (b - a) / N  
3 x = [a + k * h for k in range(0, N + 1)]  
4 L = [-1, 0] # we don't use L[0]  
5 K = [-1, y0] # we don't use K[0]  
6 # L[k] and K[k] evaluation  
7 for j in range(2, N + 1):  
8 ap = 1 - p(x[j - 1]) * h / 2  
9 bp = h * h * q(x[j - 1]) - 2  
10 cp = 1 + p(x[j - 1]) * h / 2  
11 fp = h * h * f(x[j - 1])  
12 lc = -cp / (ap * L[j - 1] + bp)  
13 kc = (-ap * K[j - 1] + fp) / (ap * L[j - 1] + bp)  
14 L.append(lc)  
15 K.append(kc)  
16 # y[k] evaluation  
17 y = [yN]
```

Исправление визуальных проблем

- Внесение изменений в **модуль визуализации ноутбуков** (требовалось адекватная работа в режиме сравнения);
- **Определение своих DiffViewer** для более гибкой настройки визуализации (например, решение проблемы с сравнением ячеек с кодом на языке Markdown).

Результат после исправлений

| Boundary Value Problem16.ipynb (/Users/Alexander.Bozhnyuk/projects/notebooks/notebooks) | Boundary Value Problem17.ipynb (/Users/Alexander.Bozhnyuk/projects/notebooks/notebooks) |
|---|--|
| <pre>Божнюк Александр Сергеевич, 271 группа # Задание: \$y'' + 2xy' - y = 2(x^2+1)\cos(x)\$ \$0 \le x \le 0.5\$ \$y(0) = 0\$ \$y(0.5) = 0.5\sin(0.5)\$</pre> | <pre>Божнюк Александр Сергеевич, 271 группа # Задание: \$0 \le x \le 0.5\$ \$y(0) = 3\$ \$y(0.5) = 0.5\sin(0.5)\$</pre> |
| <h3>Код решения (Python3)</h3> | <h3>Код решения (Python3)</h3> |
| <pre>from math import sin, cos import matplotlib.pyplot as plt def bvp(N, a, b, y0, yN, p, q, f): h = (b - a) / N x = [a + k * h for k in range(0, N + 1)] L = [-1, 0] # we don't use L[0] K = [-1, y0] # we don't use K[0] # L[k] and K[k] evaluation for j in range(2, N + 1): ap = 1 - p(x[j - 1]) * h / 2 bp = h * h * q(x[j - 1]) - 2 cp = 1 + p(x[j - 1]) * h / 2 fp = h * h * f(x[j - 1]) lc = - cp / (ap * L[j - 1] + bp) kc = (-ap * K[j - 1] + fp) / (ap * L[j - 1] + bp) L.append(lc) K.append(kc) # y[k] evaluation y = [yN] for j in range(N - 1, 0, -1): y.insert(0, L[j + 1] * y[0] + K[j + 1]) v.insert(0, y0)</pre> | <pre>1 1 from math import sin, cos 2 2 import matplotlib.pyplot as plt 1 1 def bvp(N, a, b, y0, yN, p, q, f): 2 2 h = (b - a) / N 3 3 x = [a + k * h for k in range(0, N + 1)] 4 4 a = 3 5 5 L = [-1, 0] # we don't use L[0] 6 6 K = [-1, y0] # we don't use K[0] 7 7 # L[k] and K[k] evaluation 8 8 for j in range(2, N + 1): 9 9 ap = 1 - p(x[j - 1]) * h / 2 10 10 bp = h * h * q(x[j - 1]) - 2 11 11 cp = 1 + p(x[j - 1]) * h / 2 12 12 13 13 fp = h * h * f(x[j - 1]) 14 14 15 15 kc = (-ap * K[j - 1] + fp) / (ap * L[j - 1] + bp) 16 16 L.append(lc) 17 17 K.append(kc) 18 18 19 19 # y[k] evaluation 20 20 v = [vN]</pre> |

Сравнение выводов ячеек

Для сравнения выводов ячеек требовалось сделать следующее:

- **Извлечение информации** о выводах ячеек в удобном для сравнения виде;
- **Сравнение** извлеченной информации при помощи модуля сравнения IntelliJ. Определение того, какие выводы ячеек были вставлены/удалены/изменены;
- Основываясь на результатах сравнения, **перекрасить необходимые компоненты**, отвечающие за визуализацию выводов ячеек.

Все это было реализовано через переопределенные **DiffViewer**.

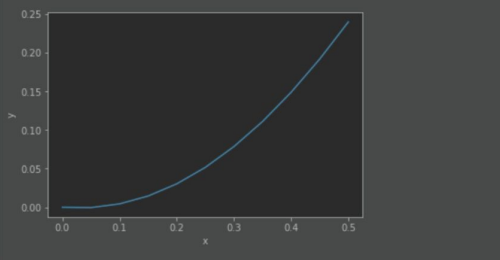
Результат сравнения выводов ячеек

```
Boundary Value Problem17.ipynb (/Users/Alexander.Bozhnyuk/projects/notebooks/notebooks)
a = 0
b = 0.5
y0 = 0
s = 'right'
yN = 0.5 * sin(0.5)
g = 'right'
p = lambda x: 2 * x
q = lambda _: -1
f = lambda x: 2 * (x + 1) * cos(x)

x1, y1 = bvp(N1, a, b, y0, yN, p, q, f)
print("x1:", x1)
print("y1:", y1)

x2, y2 = bvp(N2, a, b, y0, yN, p, q, f)
graph_plot(x2, y2)
```

x2: [0.0, 0.05, 0.1, 0.15000000000000002, 0.2, 0.25, 0.30000000000000004, 0.35000000000000003, 0.4, 0.45, 0.5]
y2: [0, -0.00043521737454653864, 0.0043610135406924, 0.014565665484516896, 0.030297640587335105, 0.051615263280707195, 0.07851491088919928, 0.11093079990486837, 0.1487359141745262, 0.19174403057425282, 0.2397127693021015]

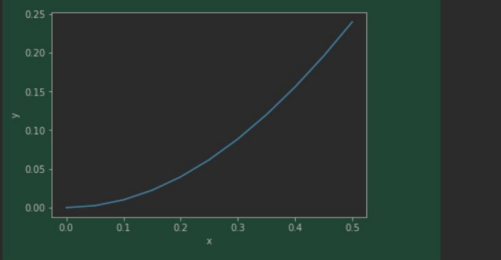


```
Boundary Value Problem16.ipynb (/Users/Alexander.Bozhnyuk/projects/notebooks/notebooks)
v = 'right'
yN = 0.5 * sin(0.5)
g = 'right'
p = lambda x: 2 * x
q = lambda _: -1
f = lambda x: 2 * (x * x + 1) * cos(x)

x1, y1 = bvp(N1, a, b, y0, yN, p, q, f)
print("x1:", x1)
print("y1:", y1)

s = 'left'
v = '1111'
graph_plot(x1, y1)
```

x1: [0.0, 0.05, 0.1, 0.15000000000000002, 0.2, 0.25, 0.30000000000000004, 0.35000000000000003, 0.4, 0.45, 0.5]
y1: [0, 0.00248846368914255, 0.00996447303514985, 0.02239065364860176, 0.0397048502033616, 0.06182035804444585, 0.08862624656644053, 0.1199877733411355, 0.15574688768054243, 0.19572282203057856, 0.2397127693021015]



Инструмент слияния

- Основная идея не отличается от **Diff Tool**;
- Реализация своего **MergeTool**: **JupyterMergeTool**, в нем производить подмену **MergeRequest**;
- Реализация своего **MergeView**, для этого требовалось произвести рефакторинг в модуле сравнения платформы. Это позволило внести правки в визуализацию.

Результат: инструмент сравнения

Merge Revisions for /Users/Alexander.Bozhnyuk/projects/notebooks/notebooks/Boundary Value Problem13.ipynb

Apply non-conflicting changes: >> Left >>> All <<< Right <<< Do not ignore Highlight words 2 changes. 3 conflicts.

| Your version | Result | Changes from server (revision d0b2c29607b63333b746cda2aa32b... |
|---|---|---|
| <pre>y.insert(0, L[j + 1] * y[0] + K[j + 1]) y.insert(0, y0) return (x, y)</pre> | 19 19 20 20 21 21 | <pre>y.insert(0, L[j + 1] * y[0] + K[j + 1]) y.insert(0, y0) return (x, y)</pre> |
| <pre>def graph_plot(x, y): plt.plot(x1, y1) plt.xlabel("x1111111") plt.ylabel("y1111111") plt.show()</pre> | 1 1 2 2 3 3 4 4 5 5 6 6 | <pre>def graph_plot(x, y): plt.plot(x1, y1) plt.xlabel("x0000000") plt.ylabel("y0000000") plt.show()</pre> |
| <pre>N1 = 10 N2 = 20 s = 'left' a = 0 b = 0.5 y0 = 0 yN = 0.5 * sin(0.5) p = lambda x: 2 * x q = lambda _: -10000000 f = lambda x: 2 * (x * x + 1) * cos(x)</pre> | 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 9 9 10 10 | <pre>N1 = 10 N2 = 20 s = 'base' a = 0 b = 0.5 y0 = 0 yN = 0.5 * sin(0.5) p = lambda x: 2 * x q = lambda _: -10000000 f = lambda x: 2 * (x * x + 1) * cos(x)</pre> |
| <pre>x1, y1 = bvp(N1, a, b, y0, yN, p, q, f) print("x:1111111", x1) print("y:1111111", y1) graph_plot(x1, y1)</pre> | 1 1 2 2 3 3 4 4 | <pre>x1, y1 = bvp(N1, a, b, y0, yN, p, q, f) print("x:0000000", x1) print("y:0000000", y1) graph_plot(x1, y1)</pre> |
| <pre>f = 0 for i in range(0, 21, 2): if (abs(y1[i // 2] - y2[i]) > f): f = abs(y1[i // 2] - y2[i]) print(f)</pre> | 1 1 2 2 3 3 4 4 5 5 | <pre>x2, y2 = bvp(N2, a, b, y0, yN, p, q, f) graph_plot(x2, y2) # Погрешность f = 0 for i in range(0, 21, 2): if (abs(y1[i // 2] - y2[i]) > f): f = abs(y1[i // 2] - y2[i]) print(f)</pre> |

Accept Left Accept Right Cancel Apply

Результаты

1. Проведен обзор проекта Jupyter и алгоритмов для сравнения файлов;
2. Проведен обзор конкурентов;
3. Проведен обзор платформы IntelliJ;
4. Реализован инструмент сравнения ноутбуков. Исправлены многочисленные проблемы, связанные с визуализацией ноутбуков Jupyter в режиме сравнения. Реализованы система сравнения ячеек с кодом на языке Markdown и система сравнения выводов ячеек;
5. Реализован инструмент слияния веток.