Федеральное государственное бюджетное образовательное учреждение высшего образования  
Национальный исследовательский университет «МЭИ»

Научно-исследовательская работа

Тема

«Многомерная линейная регрессия»

Студент Черток В.Д.  
Группы А-05-14

Научный руководитель Шаграев А.Г.

Москва 2017

Оглавление

**1. Введение1**

1.1. Линейная регрессия2

1.2. Вычисление сумм и средних для большого количества вещественных чисел 2

1.3. Метод Уэлфорда для взвешенных сумм произведений4

1.4. Многомерная линейная регрессия 5

**2. Основная часть.7**

2.1. Анализ7

2.2. Проектирование7

2.3. Реализация9

2.4. Тестирование14

**3. Сравнение с аналогами15**

**4. Заключение15**

**5. Используемая литература16**

1. Введение

Многомерная линейная регрессия [1] является одним из классических методов восстановления функциональных зависимостей по эмпирическим данным.

В настоящей работе реализуется вычислительно устойчивый метод решения задачи восстановления линейной регрессии, позволяющий работать со взвешенными выборками и поддерживающий обновление моделей при поступлении новых данных.

Для решения нормальной системы МНК используется метод LDL-разложения. [2].

Для составления матрицы нормальной системы МНК используется метод Уэлфорда [3], модифицированный для использования взвешенных объектов и оптимизированный применительно к матричным вычислениям.

Реализации всех предложенных методов выполнены на языке C++ в виде консольного приложения в среде Microsoft Visual Studio Community 2015.

1.1. Линейная регрессия

Мы будем рассматривать задачу восстановления линейной регрессии как задачу машинного обучения, в которой множество объектов совпадает с множеством m-мерных вещественных векторов , множество ответов совпадает с множеством вещественных чисел , классом решающих функций является множество линейных преобразований , а функционалом потерь является квадратичная функция потерь.

Для большей общности мы будем рассматривать задачу взвешенной линейной регрессии, то есть, считать, что также определена некоторая весовая функция . Тогда для решающей функции и целевой функции величина функционала потерь определяется следующим образом: если – обучающая выборка, то

1.2. Вычисление сумм и средних для большого количества вещественных чисел.

Пусть имеется некоторый набор вещественных чисел

и соответствующие им веса

Нас будет интересовать вопрос вычисления суммы

и среднего

Введем также обозначение

Известно, что точность вычислений по формуле с использованием «наивного» алгоритма зависит от очередности слагаемых, и наилучшая точность достигается в том случае, когда величины отсортированы в порядке возрастания абсолютных значений [4]. Абсолютная величина погрешности для любого «наивного» метода суммирования растет линейно с ростом количества слагаемых. Метод Кахана [5] лишен этого недостатка: абсолютная величина погрешности при суммировании этим методом не зависит от количества слагаемых.

Тогда

поскольку

1.3. Метод Уэлфорда для взвешенных сумм произведений

Пусть теперь также имеется последовательность величин , которым соответствуют те же веса.

Обозначим:

Легко видеть, что

Для обновления отношения можно использовать формулу (1):

1.4. Многомерная линейная регрессия.

Рассмотрим теперь классическую задачу многомерной линейной регрессии.

Пусть даны

Известно, что вектор , являющийся решением этой задачи оптимизации, находится из системы нормальных уравнений МНК:

Элементы матрицы системы вычисляются по следующим формулам:

а элементы вектора в правой части системы по формулам

При непосредственном использовании этих формул могут, естественно, возникать проблемы, аналогичные тем, что возникают в случае решения задачи одномерной линейной регрессии. Попробуем поэтому применить метод Уэлфорда и для этой задачи.

Будем решать центрированную задачу наименьших квадратов. Формально, составим задачу с центрированной матрицей системы МНК и центрированным вектором правых частей:

Известно, что в такой задаче свободный коэффициент будет равен нолю. Если вектор является решением модифицированной задачи МНК

то предсказание в модифицированной задаче будет осуществляться по формуле

При этом будет предсказываться величина .

Таким образом, решение исходной задачи совпадает с решением модифицированной задачи за тем исключением, что в исходной задаче необходимо ввести свободный коэффициент , величина которого определяется формулой

Заметим теперь, что для вычисления элементов матрицы можно использовать полученные выше формулы, аналогичные формулам Уэлфорда:

Если обозначить через и соответственно последовательности значений i-го и -го признаков, то

Аналогично, для элементов вектора в правой части модифицированной системы можно получить:

**2. Основная часть.**

2.1. Анализ

В рамках поставленной задачи требуется написать программу обладающую следующим функционалом:

1) Консольное приложение, получающее параметры в виде аргументов командной строки.

2) Программа должна обрабатывать входной файл, представляющий из себя построчно записанные объекты и ответы.

3) Программа должна иметь возможность выгружать обученную модель во внешний файл, и загружать обученную модель из внешнего файла для предсказания.

4) Предусмотреть возможность последующего добавления других алгоритмов решения.

2.2. Проектирование

Для реализации функциональных требований нам понадобится 4 входных параметра:

1) Режим - обучение или предсказание;

2) Путь входного файла;

3) Путь (куда сохранять)/(откуда загружать) обученную модель;

4) Режим обучения (при выбранном в первом пункте «обучении») - позволит задать алгоритм, которым будет обучаться модель, или Путь куда сохранять предсказания (при выбранном в первом пункте «предсказании»);

Для хранения введенной информации введем класс Dataset. Возложим на него функции чтения информации из файла, и передачи её классу, реализующему обучающий алгоритм.

Для реализации обучающего алгоритма по методу Уэлфорда введем класс WelfordSolver. Наличие обучающего алгоритма в отдельном классе позволит в будущем при реализации другого обучающего алгоритма, написать другой класс с таким же интерфейсом.

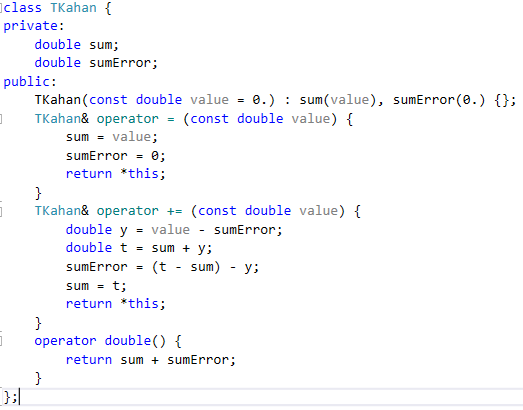
Для решения СЛАУ методом LDL-разложения введем отдельный класс LDL предоставляющий свои методы классу, реализующему обучающий алгоритм. Это позволит в дальнейшем реализовать другой алгоритм решения СЛАУ, написав для него другой класс с тем же интерфейсом.

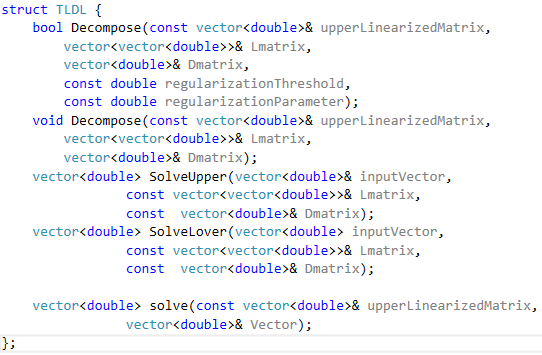
Для хранения обученной модели введем класс LinearModel, при обучении этот класс будет получать информацию от класса, реализующего обучающий алгоритм. При предсказании, он будет загружать модель из внешнего файла и подсчитывать функционал ошибки.

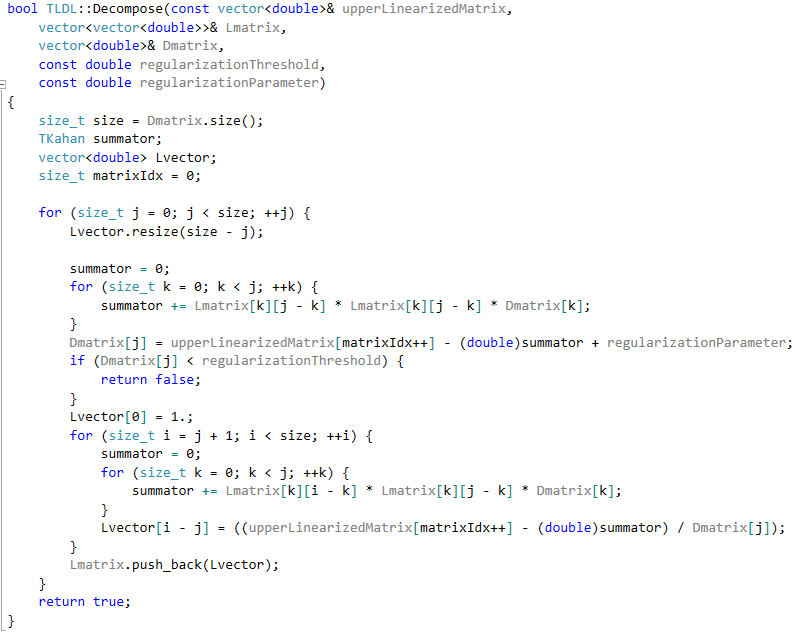
Для выполнения точного сложения больших последовательностей дробных цифр реализуем класс Kahan. Он будет использоваться повсеместно при операциях сложения.

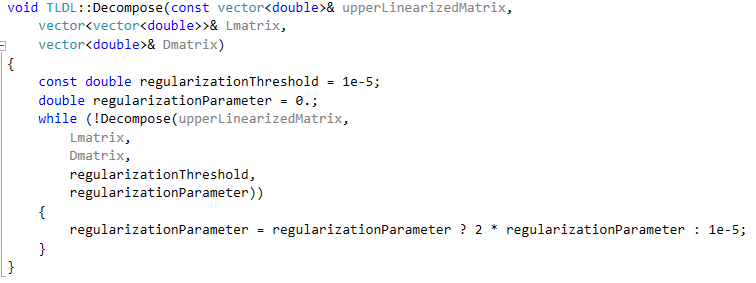
2.3. Реализация

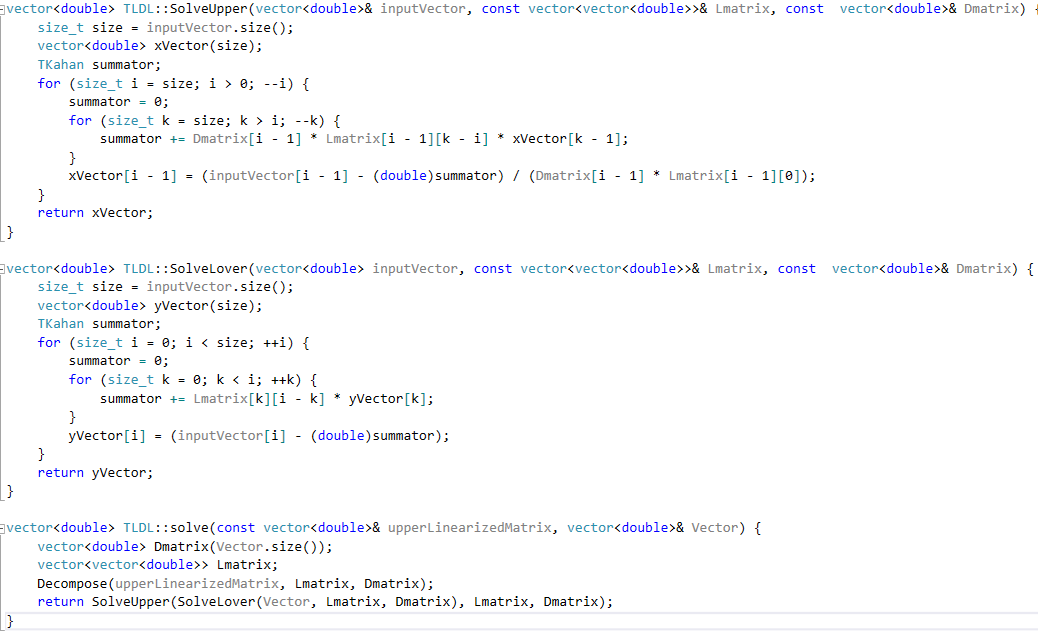


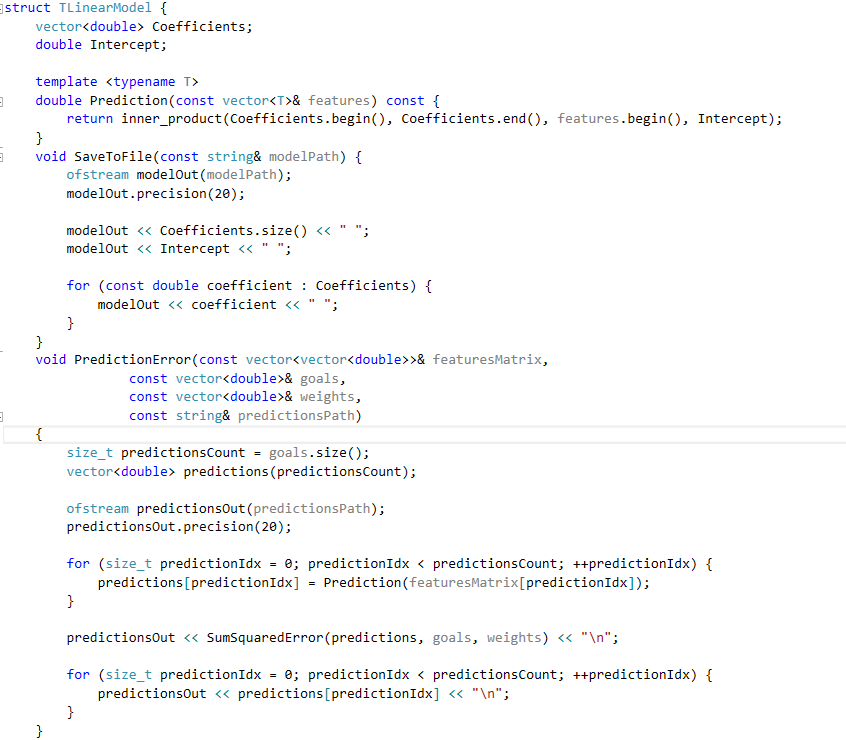


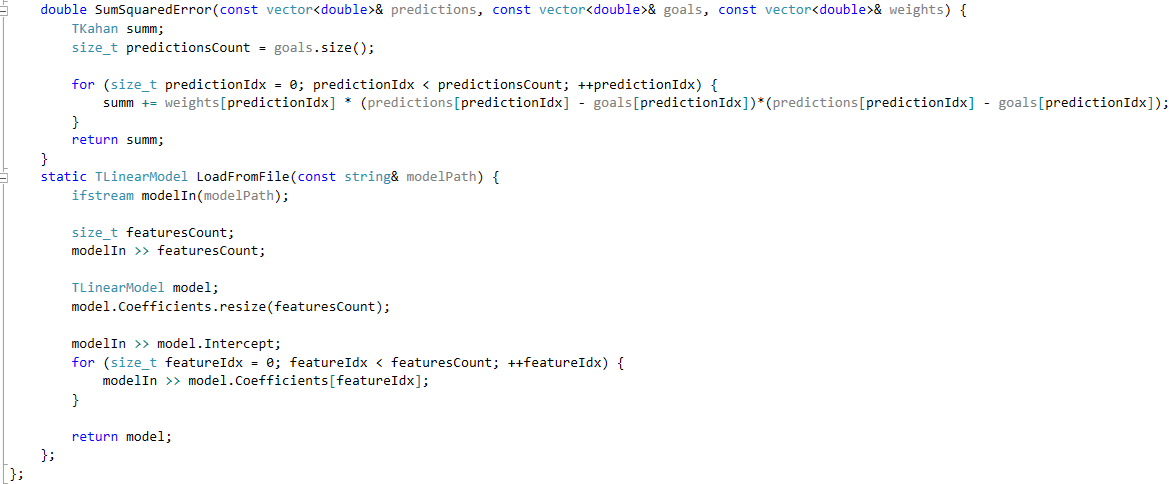


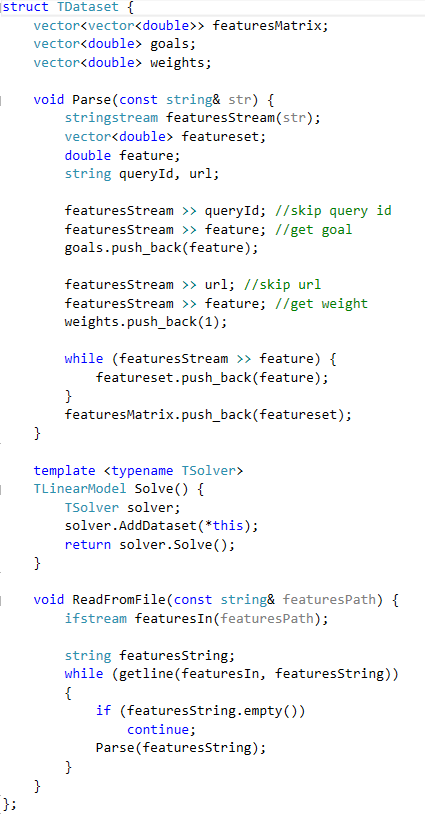


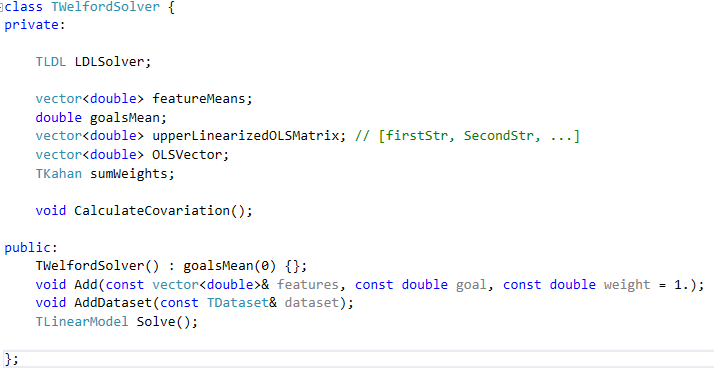


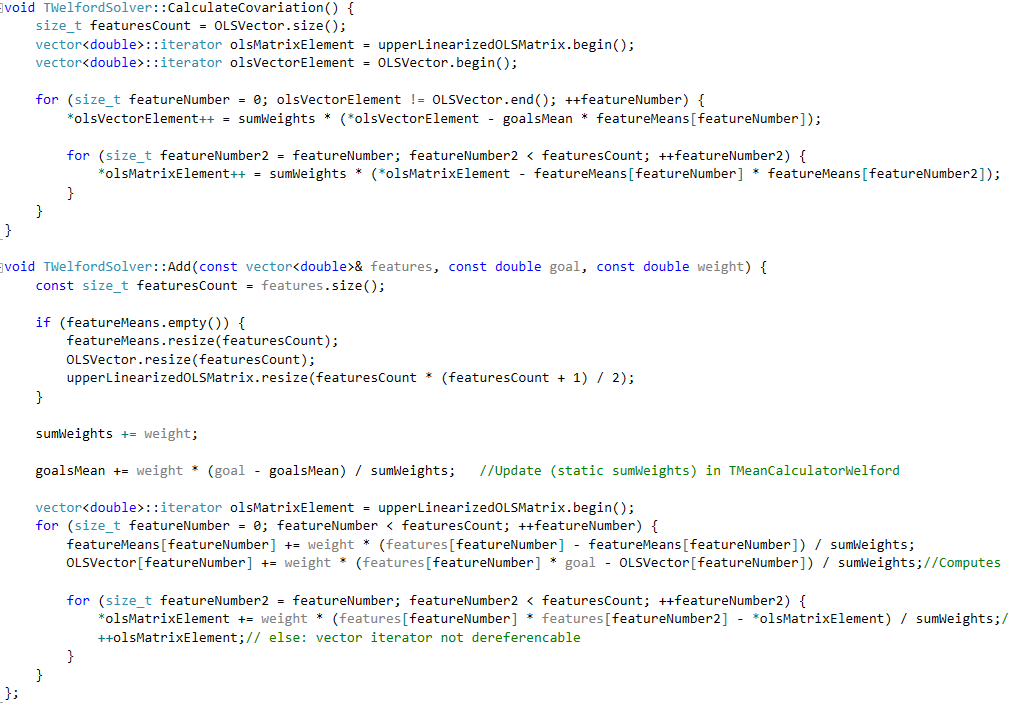


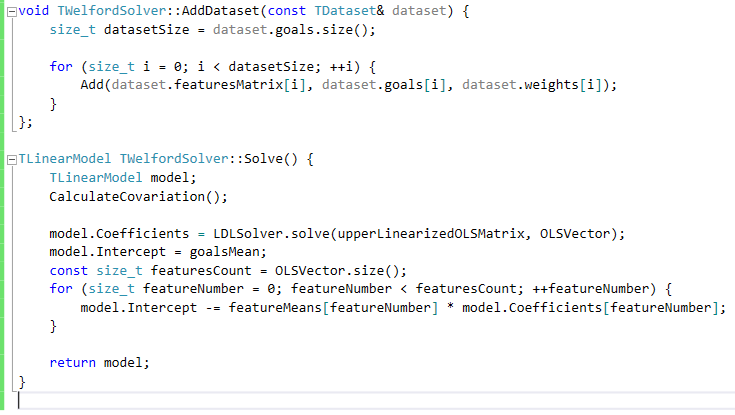




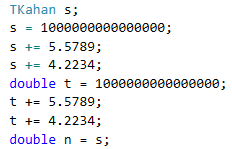


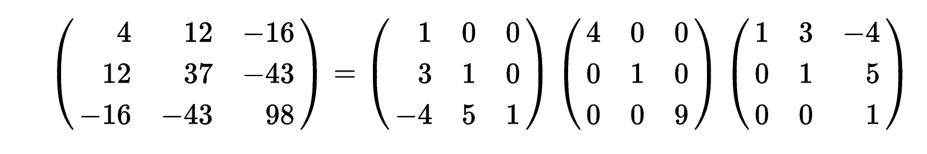
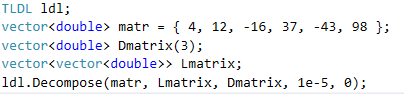


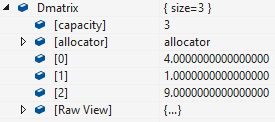
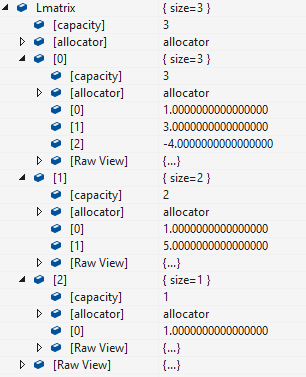




2.4. Тестирование

Результаты п.3 показывают корректность работы алгоритма.  
Дополнительно проверим корректность реализации алгоритма Кахана:  
  
  
Очевидно, что алгоритм Кахана исправил ошибку округления возникшую при сложении двух чисел выходящих за разрядную сетку.

Так же проверим корректность алгоритма Холецкого LDL разложения матрицы:  
Ожидаемый результат:  
  




3. Сравнение с аналогами

Написанный алгоритм будем сравнивать с моделью «linear\_model.LinearRegression» из библиотеки «scikit-learn» для языка программирования Python.

Для проведения эмпирического исследования были взяты пять задач из коллекции UCI: cpu\_act, delta\_ailerons, kin8nm, machine\_cpu и puma32H. Эти задачи являются регрессионными и при этом не содержат номинальных признаков.  
Обучение производилось на всем множестве объектов, в качестве оценки обобщающей способности алгоритма использовалась ошибка всем множестве объектов.  
Ошибка вычислялась по формуле:  
Предобработка данных не производилась.

|  |  |  |
| --- | --- | --- |
| Набор данных | Welfrod C++ | LinearRegression Python |
| cpu\_act | 751888.84194471943192 | 751888.84194471862 |
| delta\_ailerons | 0.00021071364605677539244 | 0.0002107136460567752 |
| kin8nm | 333.59783195609014683 | 333.59783195609089 |
| machine\_cpu | 726920.11520360922441 | 726920.11520360992 |
| puma32H | 5.8288266996378448326 | 5.8288266996378439 |

\*Программная реализация работы с библиотекой scikit-learn в приложении.

4. Заключение

В результате проделанной работы был реализован алгоритм многомерной линейной регрессии с использованием алгоритмов Кахана и Уэлфорда.

Практическое сравнение показало, что алгоритм показывает сопоставимые, и в ряде случаев даже превосходящие результаты по сравнению с алгоритмом библиотеки scikit-learn.

К плюсам реализованного алгоритма можно отнести то, что он устойчив к разбросам масштабов входных параметров и не требует строгой предварительной регуляризации обучающей выборки.

5. Список использованной литературы

1. Дрейпер Н., Смит Г. Прикладной регрессионный анализ. – М.: Диалектика. – 2007.
2. Krishnamoorthy A., Menon D. Matrix inversion using Cholesky decomposition // Proc. Alg. Arch. Arrangements Applicat., 2013
3. Welford B. P. Note on a Method for Calculating Corrected Sums of Squares and Products // Taylor & Francis, Ltd. – Vol. 4, No. 3. – 1962.
4. Higham N. J. The accuracy of floating point summation // SAM Journal on Scientific Computing – T. 14 (4)
5. Kahan, W. Further remarks on reducing truncation errors // Communications of the ACM. – 1965.

5. Приложение

Программная реализация работы с библиотекой scikit-learn на ЯП Python в среде разработки Jupyter.  
