

Введение в работу с данными

Гань Чжаолун

26 декабрь, 2024, Москва, Россия

Российский Университет Дружбы Народов

Цели и задачи работы

Цель лабораторной работы

Основной целью работы является специализированных пакетов Julia для обработки данных.

Процесс выполнения лабораторной работы

Используя Jupyter Lab, повторите примеры из раздела 7.2

Я повторю все задание 7.2 целиком

Выполните задания для самостоятельной работы

```
using RDatasets
iris = dataset("datasets", "iris")
```

150 rows x 5 columns

	SepalLength	SepalWidth	PetalLength	PetalWidth	Species
	Float64	Float64	Float64	Float64	Cat...
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa

```
# Построение графика:
plot(size=(500,500),leg=false)
x = iris[:, :SepalLength]
y = iris[:, :PetalLength]
scatter(x, y, markersize=3, title="Распределение признаков SepalLength и PetalLength",
        xlabel="SepalLength", ylabel="PetalLength", leg=false)
```

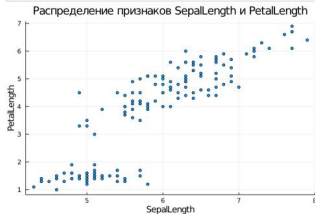


Рисунок 1. Код и результат Задания 1.1–1.2

Выполните задания для самостоятельной работы

1	5.0	1.4
4	4.9	1.4
5	4.7	1.3
5	4.9	1.4
4	5.1	1.4

cluster cluster
sepal.length petal.length

100 rows x 5 columns

```
X = matrix('iris', nrow=nrow(X), byrow=TRUE)
# Транспонирование матрицы с данными
```

Конвертация данных в матричный вид:

```
X = convert(Matrix(Float64, X))
```

Транспонирование матрицы с данными:

```
X = X'
```

2x150 LinearAlgebra.Adjoint{Float64,Array{Float64,2}}:

```
5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 - 6.8 6.7 6.7 6.3 6.5 6.2 5.9
1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 5.9 5.7 5.2 5.0 5.2 5.4 5.1
```

Задание количества кластеров на основе вида цветов:

```
k = length(unique(iris[, :Species]))
```

3

Определение K-среднего:

```
C = kmeans(X, k)
```

```
KmeansResult{Array{Float64,2},Float64,Int64}([5.874137931034483 6.839024390243902 5.007843137254902; 4.39310344827586
3 5.678048780487804 1.4921568627450983], [3, 3, 3, 3, 3, 3, 3, 3 - 2, 2, 1, 2, 2, 1, 2, 1], [0.01698577
4702035883, 0.0201230296040092, 0.13169165705497932, 0.16639753940791735, 0.008554402153016838, 0.1969857747020427,
0.1748289119569364, 0.00012302960399779295, 0.37796616685889717, 0.011691657054981874 - 0.0254193932183, 0.33785841
760854396, 0.5051991676575369, 0.05078524687686194, 0.01980963712077255, 0.24785841760854055, 0.5496819262782395, 0.3
4346817370612825, 0.48566329565733213, 0.5003715814506506], [58, 41, 51], [58, 41, 51], 53.80997864410648, 10, true)
```

Формирование фрейма данных:

```
iris_new = DataFrame(cluster = C.assignments,
SepalLength = iris[:, :SepalLength], PetalLength = iris[:, :PetalLength], Species = iris[:, :Species])
```

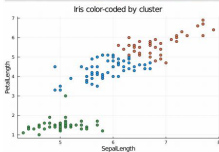
150 rows x 4 columns

cluster	SepalLength	PetalLength	Species
Int64	Float64	Float64	Cat...
1	3	5.1	setosa
2	3	4.9	setosa
3	3	4.7	setosa
4	3	4.6	setosa

Рисунок 2. Код и результат Задания 1.3–1.6

Выполните задания для самостоятельной работы

```
qwebya(cjnsrsw~tjdnle)
ftrjei(,tjns cojoi-coqeq pA cjnsrsw,_)
ljwreji(,bsewtjwudgr,_)
kjrweji(,sebwjtjwudgr,_)
ewq
sewrswi(cjnsrsw~tjdnle, kwtja, lwtja, wwtjkwetjse=q)
lwtja = tjns'wem'cjnsrswi[1':bsewtjwudgr]
kwtja = tjns'wem'cjnsrswi[1':sebwjtjwudgr]
tjns'wem'cjnsrswi = tjns'wem[tjns'wem[1':cjnsrswi]'== 7':]
zox t = 1:k
cjnsrswi~tjdnle = bjtjof{redwq = zjwse}
```



```
unique_species = unique(iris[,1:Species])
species_figure = plot(legend = false)
for species in unique_species
  iris_sp = iris[iris[,1:Species]==species,]
  x = iris_sp[,Sepal.Length]
  y = iris_sp[,Petal.Length]
  scatter(iris_sp, x, y)
end
xlabel("Sepal.Length")
ylabel("Petal.Length")
title("Iris color-coded by species")
display(species_figure)
```

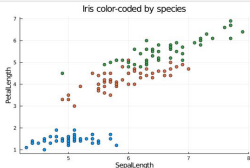


Рисунок 3. Код и результат Задания 1.7–1.8

Выполните задания для самостоятельной работы

```
[0'e4e883e88501f34e' 0'0432242e4ae4e32a' 0'22e0a84e7335235e]

λ = X * w0 + 0'I * κωquq(I000):
βιγuρ(w0)
w0 = κωquq(3)
X = κωquq(0' 22e' 3)

[0'e4e883e88501f33312' 0'04523a3244453330e' 0'228033585e3423e' -0'0074022a38033334433]

βιγuρ(w)
w = γγuεwελ'λεδισεσγou'ωoεfj(x' λ)

γγuεwελ'λεδισεσγou'ωoεfj (δewεγic γnuvεfγou λγγu γ ωεfγoq)

εuq
    κεfγau X / λ
    X = γuεc(x' X5)
    X5 = ouεs(I000)
    γnuvεfγou γγuεwελ'λεδισεσγou'ωoεfj(x' λ)

[0'e4e883e88501f33312' 0'04523a3244453330e' 0'228033585e3423e' -0'0074022a38033334433]

βιγuρ(w)
s = γγad(x' λ)
% εοιγε νεγυα γγad
νεγυα ηγγfγλ9εγγcε2εfεε

Pkg.add("GLM")
using GLM, DataFrames

Resolving package versions...
No Changes to ~/.julia/environments/v1.5/Project.toml`
No Changes to ~/.julia/environments/v1.5/Manifest.toml`

x1 = X[:,1]
x2 = X[:,2]
x3 = X[:,3]

data = DataFrame{y = y, x1 = X1, x2 = X2, x3 = X3};

log(fracmcsim) ~ x1 + x2 + x3); data)

y = 1 + x1 + x2 + x3

coefficients:

Coeff. Std. Error t Pr(>|t|) Lower 95% Upper 95%
[Intercept] -0.20140559 0.00310028 -64.45 0.0004 -0.00748934 0.00487815
x1 0.4646052 0.00305025 211.21 1e-39 0.4640952 0.4652073
x2 0.04625398 0.00292113 12.94 1e-24 0.0360991 0.0469084
x3 0.0460319 0.00289713 160.74 1e-39 0.0331975 0.0489054
```

Рисунок 4. Код и результат Задания 2.1

Выполните задания для самостоятельной работы

X3	0'228033	0'0030813	180'10	<10-20	0'227312	0'204001
X5	0'045238	0'0035813	15'30	<10-24	0'034001	0'000000
X7	0'000000	0'0030000	517'51	<10-24	0'000000	0'000000
(Intercept)	-0'0010402	0'0037005	-0'42	0'0004	-0'0014003	0'0004012
	coef	std_err	t	p(> t)	coef	std_err

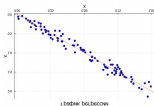
COLLECTOR:

$$\lambda = J + x_I + x_S + x_T$$

```
ju(ξozuπy(λ - x1 + x3 + x3), q0c0)
```

```
qwcw = DwcwLwwo(λ = λ', xI = xI', xS = xS', xJ = xJ)!
```

```
x3 = x[1:3]
```

$$x_5 = x[1:5]$$
$$XJ = X(:, J)$$


```

k:=gcd(gcd(m^2-n^2, gcd(m+n, m-n)), gcd(m^2-n^2, gcd(m+n, m-n)))
m:=m^2-k^2
n:=m^2-k^2
m^2-p:=gcd(m^2-n^2, gcd(m+n, m-n))
k:=5*k+1
g:=m^2-n^2

```

Рисунок 5. Код и результат Задания 2.2

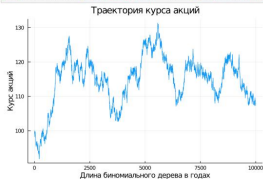
Выполните задания для самостоятельной работы

```
S = 100
T = 1
n = 10000
sigma = 0.3
r = 0.08

h = T / n # длина одного периода
u = exp(r*h + sigma * sqrt(h))
d = exp(r*h - sigma * sqrt(h))
p = (exp(r*h) - d)/(u - d)

j = 0
stockTree = []
append!(stockTree, S)
for i in 1:n
    k = rand()
    if k < p
        append!(stockTree, S * (u^(i - j)) * (d ^ j))
    else
        append!(stockTree, S * (u^(i - j)) * (d ^ (j + 1)))
        j = j + 1
    end
end

using Plots
plot!(stockTree, title="Траектория курса акций", xlabel="Длина бинаomialного дерева в годах", ylabel="Курс акций", leg=false)
```



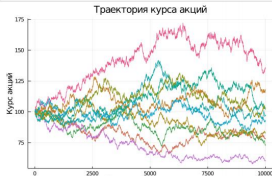
```
function createStock(S::Int64, T::Int64, n::Int64, sigma::Float64, r::Float64)
    # S - начальная цена акции
    # T - время бинаomialного дерева в годах
    # n - длина одного периода
    # sigma - волатильность акции
    # r - безрисковая процентная ставка

    h = T / n # длина одного периода
    u = exp(r*h + sigma * sqrt(h))
    d = exp(r*h - sigma * sqrt(h))
    p = (exp(r*h) - d)/(u - d)
    stockTree = []
    append!(stockTree, S)
    j = 0
    for i in 1:n
        k = rand()
        if k < p
            append!(stockTree, S * (u^(i - j)) * (d ^ j))
        else
            j = j + 1
            append!(stockTree, S * (u^(i - j)) * (d ^ j))
        end
    end
    return stockTree
end
```

Рисунок 6. Код и результат Задания 3.1

Выполните задания для самостоятельной работы

```
for i in 1:10
    IJulia.clear_output(true)
    traj = createPath(100, 1, 10000, 0.3, 0.08)
    if i == 1
        p = plot(traj, title="Траектория курса акций", xlabel="Длина бинаomialного дерева в годах",
            ylabel="Курс акций", leg=false)
    end
    p = plot!(traj)
    display(p)
end
```



```
using Base.Threads
Threads.@threads for i in 1:10
    IJulia.clear_output(true)
    traj = createPath(100, 1, 10000, 0.3, 0.08)
    if i == 1
        g = plot(traj, title="Траектория курса акций", xlabel="Длина бинаomialного дерева в годах",
            ylabel="Курс акций", leg=false)
    end
    g = plot!(traj)
    display(g)
end
```

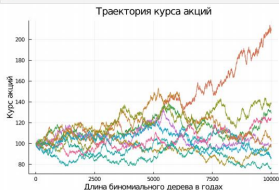


Рисунок 7. Код и результат Задания 3.2

Выводы по проделанной работе

Я освоил специализированные пакеты для решения задач в непрерывном и дискретном времени