Лабораторная работа № 7. Введение в работу с данными

Компьютерный практикум по статистическому анализу данных

Демидова Е. А.

24 ноября 2024

Российский университет дружбы народов, Москва, Россия

Информация

Докладчик

- Демидова Екатерина Алексеевна
- студентка группы НКНбд-01-21
- Российский университет дружбы народов
- https://github.com/eademidova

Введение

Цель работы

Основной целью работы является специализированных пакетов Julia для обработки данных.

Задачи

1. Используя Jupyter Lab, повторите примеры.
2. Выполните задания для самостоятельной работы.

Выполнение лабораторной работы

Выполнение примеров

```
using CSV, DataFrames, DelimitedFiles, Plots, Statistics, GLM
```

Считывание данных

```
# Считывание данных и их запись в структуру:
P = CSV.File("programminglanguages.csv") |> DataFrame

# Функция определения по названию языка программирования года его создания:
function language_created_year(P,language::String)
loc = findfirst(P[:,2].==language)
return P[loc,1]
end
# Пример вызова функции и определение даты создания языка Python:
language_created_year(P,"Python")
# Пример вызова функции и определение даты создания языка Julia:
language_created_year(P,"Julia")

# Функция определения по названию языка программирования
# года его создания (без учёта регистра):
function language_created_year_v2(P,language::String)
loc = findfirst(lowercase.(P[:,2]).==lowercase.(language))
return P[loc,1]
end

language_created_year_v2(P,"julia")

# Построчное считывание данных с указанием разделителя:
Tx = readdlm("programminglanguages.csv", ',')
```

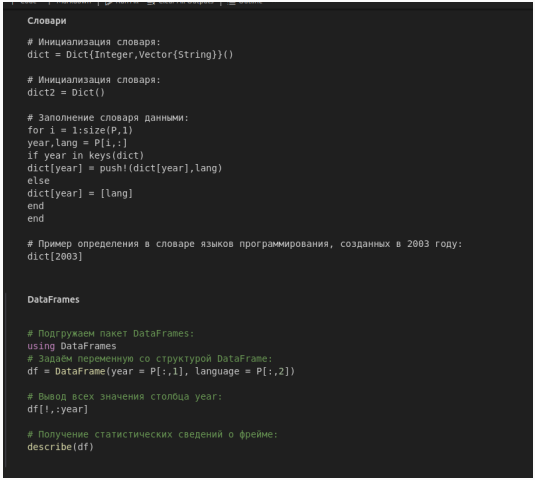
Запись данных в файл

```
# Запись данных в CSV-файл:
CSV.write("programming_languages_data2.csv", P)
Можно задать тип файла и разделитель данных:
# Пример записи данных в текстовый файл с разделителем ',':
writedlm("programming_languages_data.txt", Tx, ',')
# Пример записи данных в текстовый файл с разделителем '-':
writedlm("programming_languages_data2.txt", Tx, '-')

# Построчное считывание данных с указанием разделителя:
P_new_delim = readdlm("programming_languages_data2.txt", '-')
```

Рис. 1: Примеры

Выполнение примеров



Словари

```
# Инициализация словаря:
dict = Dict{Integer,Vector{String}}()

# Инициализация словаря:
dict2 = Dict()

# Заполнение словаря данными:
for i = 1:size(P,1)
year,lang = P[i,:]
if year in keys(dict)
dict[year] = push!(dict[year],lang)
else
dict[year] = [lang]
end
end

# Пример определения в словаре языков программирования, созданных в 2003 году:
dict[2003]
```

DataFrames

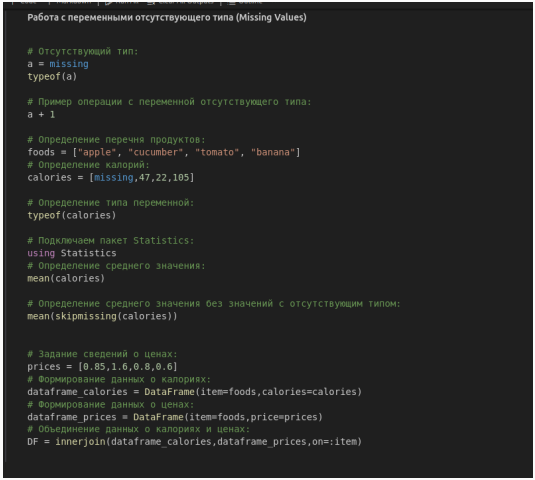
```
# Подгружаем пакет DataFrames:
using DataFrames
# Задаём переменную со структурой DataFrame:
df = DataFrame(year = P[:,1], language = P[:,2])

# Вывод всех значения столбца year:
df[!,:year]

# Получение статистических сведений о фрейме:
describe(df)
```

Рис. 2: Примеры

Выполнение примеров



Работа с переменными отсутствующего типа (Missing Values)

```
# Отсутствующий тип:
a = missing
typeof(a)

# Пример операции с переменной отсутствующего типа:
a + 1

# Определение перечня продуктов:
foods = ["apple", "cucumber", "tomato", "banana"]
# Определение калорий:
calories = [missing,47,22,105]

# Определение типа переменной:
typeof(calories)

# Подключаем пакет Statistics:
using Statistics
# Определение среднего значения:
mean(calories)

# Определение среднего значения без значений с отсутствующим типом:
mean(skipmissing(calories))


# Задание сведений о ценах:
prices = [0.85,1.6,0.8,0.6]
# Формирование данных о калориях:
dataframe_calories = DataFrame(item=foods,calories=calories)
# Формирование данных о ценах:
dataframe_prices = DataFrame(item=foods,price=prices)
# Объединение данных о калориях и ценах:
DF = innerjoin(dataframe_calories,dataframe_prices,on=:item)
```

Рис. 3: Примеры

Выполнение примеров

```
# Загрузка данных:
houses = CSV.File("houses.csv") |> DataFrame

# Построение графика:
using Plots
plot(size=(500,500),leg=false)
x = houses[:,:sq__ft]
y = houses[:,:price]
scatter(x,y,markersize=3)

# Фильтрация данных по заданному условию:
filter_houses = houses[houses[:,:sq__ft].>0,:]
# Построение графика:
x = filter_houses[:,:sq__ft]
y = filter_houses[:,:price]
scatter(x,y)

# Подключение пакета Statistics:
using Statistics
# Определение средней цены для определённого типа домов:
combine(groupby(filter_houses,[:type]),filter_houses->mean(filter_houses[:,:price]))

using Clustering
# Добавление данных :latitude и :longitude в новый фрейм:
X = filter_houses[!,[:latitude,:longitude]]
# Конвертация данных в матричный вид:
X = Matrix(X)

# Транспонирование матрицы с данными:
X = X'

# Задание количества кластеров:
k = length(unique(filter_houses[!,:zip]))

# Определение K-среднего:
C = kmeans(X,k)

# Формирование фрейма данных:
df = DataFrame(cluster = C.assignments,city = filter_houses[!,:city],
latitude = filter_houses[!,:latitude],longitude =
                    filter_houses[!,:longitude],zip = filter_houses[!,:zip])


clusters_figure = plot(legend = false)
for i = 1:k
    clustered_houses = df[df[!,:cluster].== i,:]
    xvals = clustered_houses[:,:latitude]
    yvals = clustered_houses[:,:longitude]
    scatter!(clusters_figure,xvals,yvals,markersize=4)
end
xlabel!("Latitude")
ylabel!("Longitude")
title!("Houses color-coded by cluster")
display(clusters_figure)

unique_zips = unique(filter_houses[!,:zip])
zips_figure = plot(legend = false)
for uzip in unique_zips
    subs = filter_houses[filter_houses[!,:zip].==uzip,:]
    x = subs[!,:latitude]
    y = subs[!,:longitude]
    scatter!(zips_figure,x,y)
end
xlabel!("Latitude")
ylabel!("Longitude")
title!("Houses color-coded by zip code")
display(zips_figure)
```

Рис. 4: Примеры

Выполнение примеров

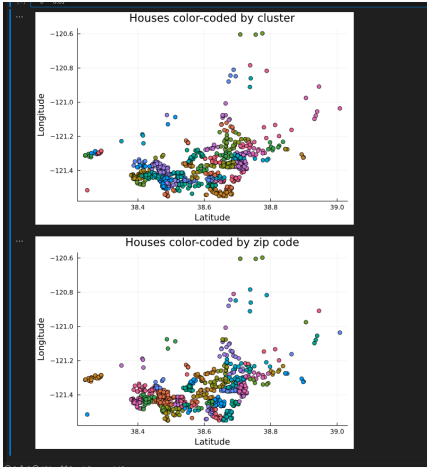


Рис. 5: Примеры

Выполнение примеров

```
using NearestNeighbors

knearest = 10
id = 70
point = X[:,id]

# Поиск ближайших соседей:
kdtree = KDTree(X)
idxs, dists = knn(kdtree, point, knearest, true)

# Все объекты недвижимости:
x = filter_houses[!,:latitude];
y = filter_houses[!,:longitude];
scatter(x,y)
# Соседи:
x = filter_houses[idxs,:latitude];
y = filter_houses[idxs,:longitude];
scatter!(x,y)
# # Фильтрация по районам соседних домов:
# cities = filter_houses[idxs,:city]
```

[30] ✓ 0.0s

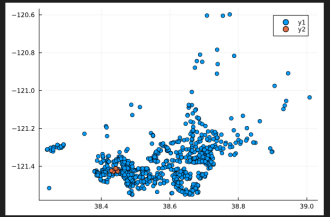


Рис. 6: Примеры

Выполнение примеров

```
# Фрейм с указанием площади и цены недвижимости:
F = filter_houses[!,[:sq__ft,:price]]
y = filter_houses[!,:price]
# Конвертация данных в массив:
# F = convert(Array{Float64,2},F)'
F = Matrix(F)'
y = y'

using MultivariateStats

# Приведение типов данных к распределение для PCA:
M = fit(PCA, F)

# Выделение значений главных компонент в отдельную переменную:
Xr = reconstruct(M, y)
# Построение графика с выделением главных компонент:
scatter(F[1,:],F[2,:])
scatter!(Xr[1,:],Xr[2,:])
```

[69] ✓ 0.1s

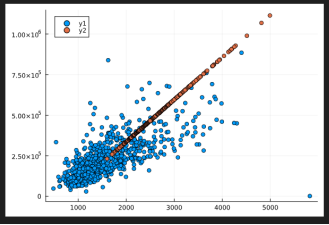


Рис. 7: Примеры

Выполнение примеров

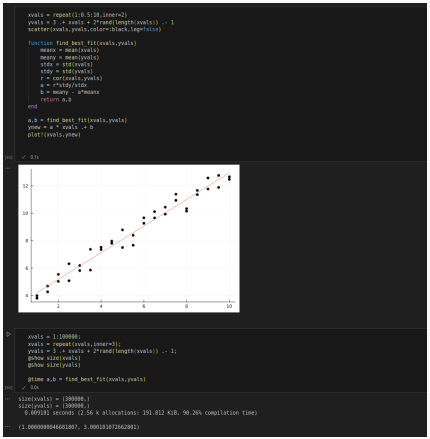


Рис. 8: Примеры

Выполнение заданий для самостоятельной работы

Рис. 9: Задание 1

Выполнение заданий для самостоятельной работы

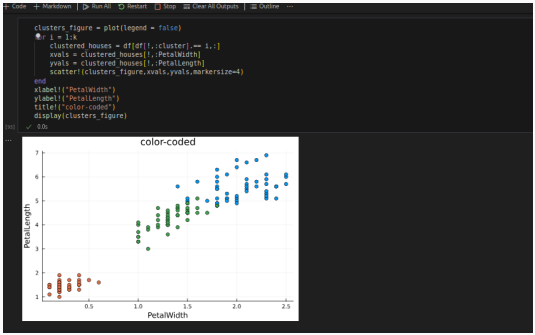


Рис. 10: Задание 1

Выполнение заданий для самостоятельной работы

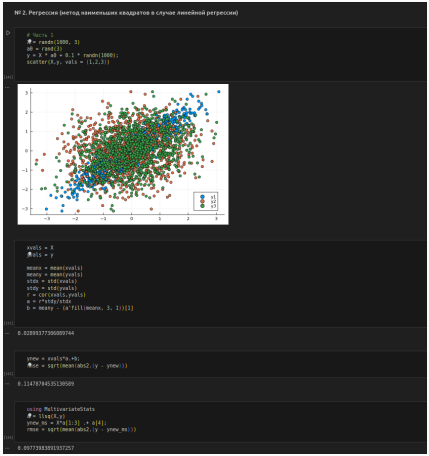


Рис. 11: Задания 2

Выполнение заданий для самостоятельной работы

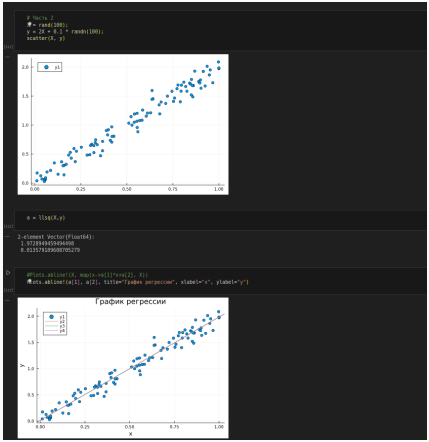


Рис. 12: Задание 2

Выполнение заданий для самостоятельной работы

Рис. 13: Задание 3

Выполнение заданий для самостоятельной работы

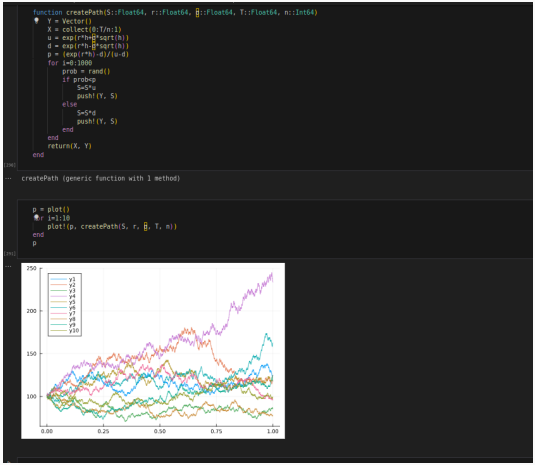


Рис. 14: Задание 3

Выполнение заданий для самостоятельной работы

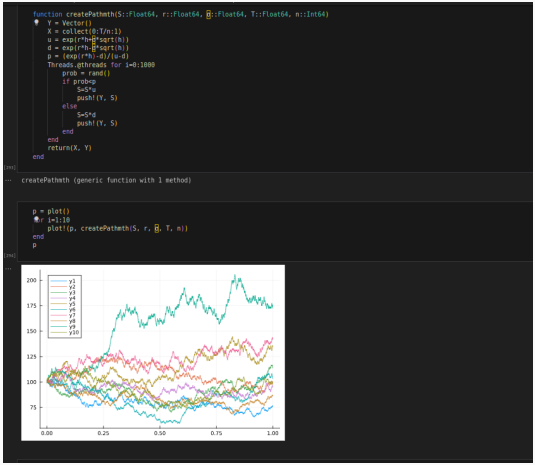


Рис. 15: Задание 3

Выводы

Выводы

В результате выполнения работы освоили использование специализированных пакетов Julia для обработки данных.

Список литературы

1. JuliaLang [Электронный ресурс]. 2024 JuliaLang.org contributors. URL: https: //julialang.org/ (дата обращения: 11.10.2024).
2. Julia 1.11 Documentation [Электронный ресурс]. 2024 JuliaLang.org contributors. URL: https://docs.julialang.org/en/v1/ (дата обращения: 11.10.2024).