数据挖掘实验实验报告

实验一: 数据预处理

姓 名: 柴 博 文
学 号: 04194012
班 号: 大数据 1901

数据挖掘与机器学习
(秋季, 2021)

西安邮电大学
计算机学院
数据科学与大数据专业

2021 年 10 月 12 日

摘 要

本次实验使用 Julia 语言进行实现.

如果需要运行本项目代码, 请安装 python 以及 matplot

随后打开终端, 运行 Julia

安装 XLSX,CSV,DataFrames,Plots,Dates,Statistics

实验报告采用 LaTeX, 在 overleaf 上进行编写.

通过 DataFrames, CSV, XLSX 读取数据, PyPlots, Plots, StatsPlot 绘制图案.

本次实验代码均可以在github 仓库下找到.

目 录

1 概述

1、掌握数据探索统计特征计算、数据可视化等基本方法

2、掌握数据集缺失值、含噪数据的平滑处理、数据变换、数据集成等预处理方法。

3、掌握 PCA 主成分分析等降维方法

- **数据可视化**对某县广电宽带用户的 5000 条数据 (或者自己感兴趣的其他领域的数据) 进行探索，通过统计特征可视化进行数据分析，探索发现你感兴趣的知识。

- **数据处理**对北京西安的年薪数据 (或者自己感兴趣的其他领域的数据) 计算均值，方差等统计特征，绘制据箱体图和小提琴图等图，分析北京西安年薪的差异。

- **数据清洗**用'movie_metadata.csv' 数据集 (或者自己感兴趣的其他领域的数据) 进行案例分析，这个数据集包含了包括演员、导演、预算、总输入，以及 IMDB 评分和上映时间等信息，进行处理缺失数据，可以是添加默认值，删除不完整的行，异常值处理，重复数据处理，规范化数据类型等等。

- **数据集成**合并两个给定数据集：ReaderRentRecode.csv 和 ReaderInformation.csv(或者自己感兴趣的其他领域的数据)，其中两个数据集的共同点是具有相同的 num 属性，最终生成一个综合的数据集。

- **PCA** 使用鸢尾花数据集 (或者自己感兴趣的其他领域的数据)，这个数据集有 150 个样本，其中每个样本有五个变量，其中四个为特征变量，分别为萼片长度 (Sepal length)，萼片宽度 (Sepalwidth)，花瓣长度 (Petallength)，花瓣宽度 (Petalwidth)，还有一个变量是其所属的品种的类别变量 (Species)，这个鸢尾花内别共有 3 种类别分别是山鸢尾 (Iris-setosa)、变色鸢尾 (Iris-versicolor) 和维吉尼亚鸢尾 (Iris-virginica)，首先对 4 维的原始数据集实现可视化，可视化一组数据来观察数据分布，然后对数据集进行标准化 (归一化)，接着利用 PCA 主成分分析将数据降到二维。

2 数据可视化

2.1 解析文件

首先使用 Excel 讲旧版 Excel 格式的 xls 文件转换为 CSV 文件github

随后使用 CSV 读取文件内容, 并通过 DataFrame 解析格式以及类型

图1.

4999×14 DataFrame

Row	计费对象	产品名称	产品到期时间	状态	停机类型	客户编号	用户类型	String
	String15	String15	String15	String15	String15	String15	String15	
1	ys8015561	宽带10M产品	6/25/2017	正使用	正常	c1002109695	新用户	
2	ys8023214	宽带4M产品	7/24/2017	正使用	正常	c1002109697	新用户	
3	ys8082301	宽带10M产品	6/25/2017	正使用	正常	c1002109701	新用户	
4	ys8022748	宽带10M产品	2/26/2018	正使用	正常	c1002114185	新用户	
5	ys8056409	宽带10M产品	1/3/2018	正使用	正常	c1002114884	新用户	
6	ys8083681	宽带10M产品	5/8/2017	正使用	正常	c1002118933	新用户	
7	ys8074844	宽带10M产品	10/29/2016	已停用	欠费停机	c1086305966	新用户	
8	ys8080014	宽带10M产品	1/5/2018	正使用	正常	c1002120172	新用户	
9	ys8070152	宽带4M产品	7/15/2015	已停用	客户报停	c1002120173	新用户	
10	ys8048259	宽带10M产品	8/25/2017	正使用	正常	c1002120293	新用户	
11	ys8074865	宽带10M产品	11/18/2017	正使用	正常	c1002123835	新用户	
12	ys8041355	宽带10M产品	5/27/2017	正使用	正常	c1002125494	新用户	
13	ys8057767	宽带10M产品	11/3/2016	已停用	欠费停机	c1007202177	新用户	
14	ys8056459	宽带10M产品	3/6/2017	正使用	正常	c1002125848	新用户	
15	ys8080035	宽带10M产品	2/5/2018	正使用	正常	c1002125908	新用户	
16	ys8046632	宽带10M产品	9/14/2017	正使用	正常	c1002126686	新用户	
17	ys8130153	宽带10M产品	1/16/2018	正使用	正常	c1002126687	新用户	
18	ys8016491	宽带10M产品	11/29/2017	正使用	正常	c1002203661	新用户	
19	ys8045160	宽带10M产品	12/26/2017	正使用	正常	c1002204537	新用户	
20	ys8096736	宽带10M产品	11/4/2016	已停用	欠费停机	c1008675505	新用户	
21	ys8027721	宽带10M产品	1/22/2018	正使用	正常	c1002204542	新用户	
22	ys8031693	宽带10M产品	3/13/2018	正使用	正常	c1002204543	新用户	
23	ys8087107	宽带10M产品	5/23/2017	正使用	正常	c1002209922	新用户	
24	ys8018042	宽带10M产品	4/16/2017	正使用	正常	c1002209924	新用户	
25	ys8046621	宽带10M产品	10/21/2017	正使用	正常	c1002209927	新用户	
26	ys8032342	宽带10M产品	8/28/2017	正使用	正常	c1002209929	新用户	
27	ys8125430	宽带20M产品	11/30/2017	正使用	正常	c1002209946	新用户	
28	ys8012610	宽带10M产品	7/4/2017	正使用	正常	c1002217047	新用户	
29	ys8087979	宽带10M产品	10/11/2017	正使用	正常	c1002236312	新用户	
30	ys8065899	宽带10M产品	6/11/2017	正使用	正常	c1002236313	新用户	
31	ys8043657	宽带10M产品	10/11/2017	正使用	正常	c1002236316	新用户	
32	ys8076592	宽带10M产品	3/14/2017	正使用	正常	c1002236322	新用户	
33	ys8037844	宽带4M产品	6/29/2017	正使用	正常	c1002236332	新用户	
34	ys8085178	宽带10M产品	1/11/2018	正使用	正常	c1002236334	新用户	
35	ys8082223	宽带10M产品	6/19/2017	正使用	正常	c1002236335	新用户	
36	ys8075274	宽带10M产品	3/21/2018	正使用	正常	c1002236336	新用户	
37	ys8044635	宽带4M产品	8/25/2017	正使用	正常	c1002236337	新用户	
38	ys8085804	宽带10M产品	10/5/2017	正使用	正常	c1002239189	新用户	
39	ys8023467	宽带10M产品	11/30/2017	正使用	正常	c1002241607	新用户	
40	ys8127407	宽带10M产品	12/23/2017	正使用	正常	c1002247402	新用户	
41	ys8062065	宽带10M产品	1/19/2018	正使用	正常	c1002261387	新用户	
42	ys8044083	宽带10M产品	8/25/2017	正使用	正常	c1002262665	新用户	
43	ys8046875	宽带10M产品	10/6/2017	正使用	正常	c1002262670	新用户	

julia>

图 1: 广电信息 CSV

```
quality =
    "lab1/julia/file/xian_guangdian.csv" |>
    CSV.File |>
    DataFrame |>
    data ->
        begin
            combine(nrow, groupby(select(data, :客户等级), :客户等级)) |>
            data -> rename(data, :nrow => "用户数量") |> println
            combine(nrow, groupby(select(data, [:客户等级, :网络类型]),
                                                [:客户等级, :网络类型]))
        end |>
        data ->
            rename(data,
                :nrow => :quantity,
                    :网络类型 => :net_kind,
```

```
                        :客户等级 => :user_level)
data = combine(groupby(quality, :net_kind), [:user_level, :quantity])
dict = Dict(
    "5星ABD客户" => "star_5ABD",
    "离线" => "out_link",
    "3星AB客户" => "star_3AB",
    "1星D客户" => "star_1D",
    "1星A客户" => "star_1A",
    "VIP商业个人客户" => "vip",
    "3星AD客户" => "start_3AD",
)
1:(data|>nrow) .|>
i -> begin
    data[i, :net_kind] =
        Dict(
            "农网用户" => "village",
            "城网用户" => "city",
            " " => "unknown"
        )[data[
            i,
            :net_kind,
        ]]
    data[i, :user_level] = dict[data[i, :user_level]]
end
gp = groupby(data, :net_kind)
gp |>
keys .|>
kind -> @df combine(gp[kind], [:user_level, :quantity]) plot(
    :user_level,
    :quantity,
    label = "$kind",
) |> fig -> savefig(fig, "lab1/julia/images/first_$kind")
```

2.2 分组

随后将数据根据客户等级进行分组, 总共有 7 组, 见图2.

7×2 DataFrame

Row	客户等级	用户数量
	String31	Int64
1	5星ABD客户	3695
2	离线	498
3	3星AB客户	151
4	1星D客户	403
5	3星AD客户	76
6	VIP商业个人客户	63
7	1星A客户	113

图 2: 分组结果图

再将每组一网络类型进行分组, 图3.

7×2 DataFrame

Row	user_level	quantity
	String31	Int64
1	star_5ABD	2192
2	out_link	313
3	star_3AB	88
4	star_1D	204
5	start_3AD	40
6	vip	57
7	star_1A	74

7×2 DataFrame

Row	user_level	quantity
	String31	Int64
1	star_5ABD	1501
2	out_link	185
3	star_3AB	62
4	star_1D	197
5	start_3AD	36
6	vip	6
7	star_1A	39

3×2 DataFrame

Row	user_level	quantity
	String31	Int64
1	star_5ABD	2
2	star_3AB	1
3	star_1D	2

图 3: 分组结果图

2.3 绘制折线图

然后将每组画到折线图之上, 图4

2.4 分析数据

通过该次结果可以看出, 在办理了广电业务的客户之中,5 星 ABD 客户数目远远多余其他客户, 而且明显城区用户多余农村用户

但是低级用户和高级用户的数量几乎差不多, 而且最关键的是两个图的趋势是相似的, 说明农村和城市对于网络的需求是很一致的

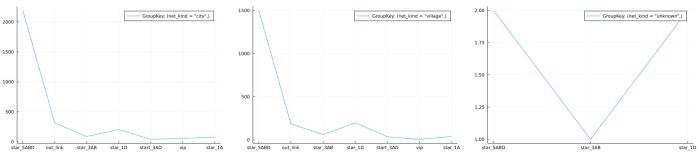


图 4: 城市居民, 农村, 未登记

3 数据处理

3.1 求的统计数据

使用 XLSX 将文件内容读入, 并使用 DataFrame 对数据进行类型判断并转换位 DataFrame 类型随后使用统计模块中的统计方法求数据的均值, 方差, 标准差, 协方差矩阵, 图5 在使用 Plots 进行绘图, 图6

```
file_path = "lab1/julia/file/xian_beijing_salary.xlsx"
salarys = DataFrame(XLSX.readdata(file_path,
                "Sheet1!C3:D14"), :auto) .|> identity
salary = [salarys.x1, salarys.x2]
println("mean:$(salary .|> Statistics.mean)")
println("var:$(salary .|> Statistics.var)")
println("std:$(salary .|> Statistics.std)")
println("cov:$(salary |> Statistics.cov)")
println("cor:$(salary .|> Statistics.cor)")
violin(["Xi'an"], salarys.x1, label = "Xi'an")
violin!(["Beijing"], salarys.x2, label = "Beijing")
boxplot(["Xi'an"], salarys.x1, label = "Xi'an")
boxplot!(["Beijing"], salarys.x2, label = "Beijing")
```

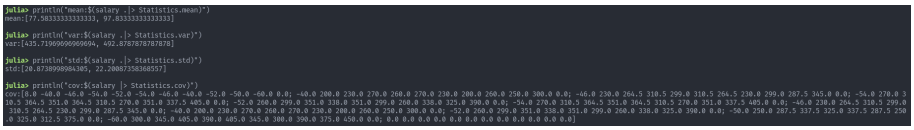


图 5: 均值, 方差, 标准差, 协方差

3.2 绘制箱型图和小提琴图

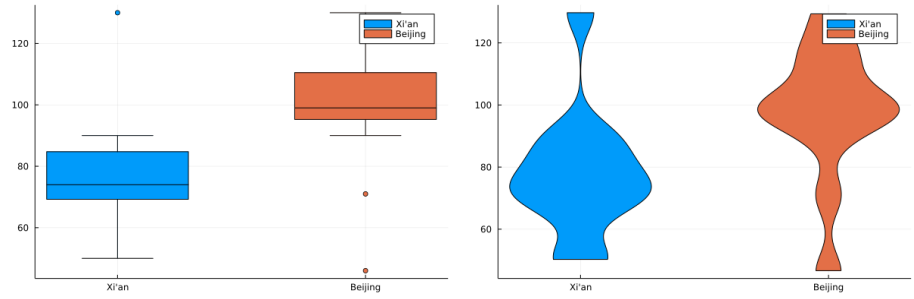
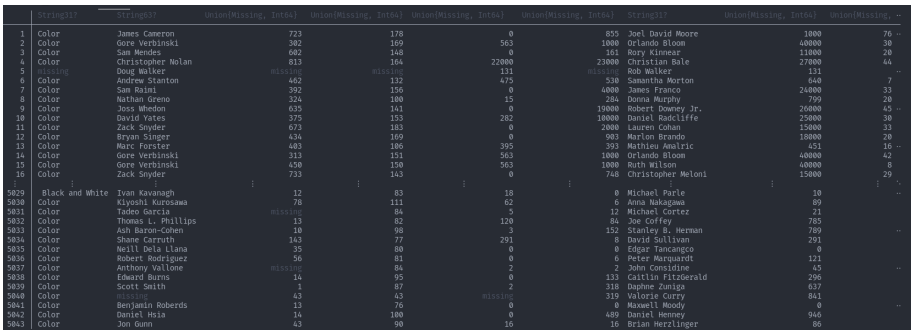


图 6: 箱型图和小提琴图

4 数据预处理

4.1 去除丢失数据行

在读取完数据后, 通过 DataFrame 转换之后
可以通过 DataFrame 得知每一列都会有值是丢失的: 图7



	String31?	String63?	Union{Missing, Int64}	Union{Missing, Int64}	Union{Missing, Int64}	Union{Missing, Int64}	String31?	Union{Missing, Int64}	Union{Missing, …
1	Color	James Cameron	723	178	0	855	Joel David Moore	1000	76 …
2	Color	Gore Verbinski	302	169	563	1000	Orlando Bloom	40000	30
3	Color	Sam Mendes	602	148	0	161	Rory Kinnear	11000	20
4	Color	Christopher Nolan	813	164	22000	23000	Christian Bale	27000	44
5	missing	Doug Walker	missing	missing	131	missing	Rob Walker	131	…
6	Color	Andrew Stanton	462	132	475	530	Samantha Morton	640	7
7	Color	Sam Raimi	392	156	0	4000	James Franco	24000	33
8	Color	Nathan Greno	324	100	15	284	Donna Murphy	799	20
9	Color	Joss Whedon	635	141	0	19000	Robert Downey Jr.	26000	45 …
10	Color	David Yates	375	153	282	10000	Daniel Radcliffe	25000	30
11	Color	Zack Snyder	673	183	0	2000	Lauren Cohan	15000	33
12	Color	Bryan Singer	434	169	0	903	Marlon Brando	18000	20
13	Color	Marc Forster	403	106	395	393	Mathieu Amalric	451	16 …
14	Color	Gore Verbinski	313	151	563	1000	Orlando Bloom	40000	42
15	Color	Gore Verbinski	450	150	563	1000	Ruth Wilson	40000	8
16	Color	Zack Snyder	733	143	0	748	Christopher Meloni	15000	29
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮ ⋱
5029	Black and White	Ivan Kavanagh	12	83	18	0	Michael Parle	10	…
5030	Color	Kiyoshi Kurosawa	70	111	62	6	Aona Nakagawa	89	
5031	Color	Tadeo Garcia	missing	84	5	12	Michael Cortez	21	
5032	Color	Thomas L. Phillips	13	82	120	84	Joe Coffey	785	
5033	Color	Ash Baron-Cohen	10	98	3	152	Stanley B. Herman	709	…
5034	Color	Shane Carruth	143	77	291	8	David Sullivan	291	
5035	Color	Neill Dela Llana	35	80	0	0	Edgar Tancangco	0	
5036	Color	Robert Rodriguez	56	81	0	6	Peter Marquardt	121	
5037	Color	Anthony Vallone	missing	84	2	2	John Considine	45	…
5038	Color	Edward Burns	14	95	0	133	Caitlin FitzGerald	296	
5039	Color	Scott Smith	1	87	2	318	Daphne Zuniga	637	
5040	Color	missing	43	43	missing	319	Valorie Curry	841	
5041	Color	Benjamin Roberds	13	76	0	0	Maxwell Moody	0	…
5042	Color	Daniel Hsia	14	100	0	489	Daniel Henney	946	
5043	Color	Jon Gunn	43	90	16	16	Brian Herzlinger	86	

图 7: 电影元数据

为了将来数据处理的正常话, 这里将导演空的一栏都去掉

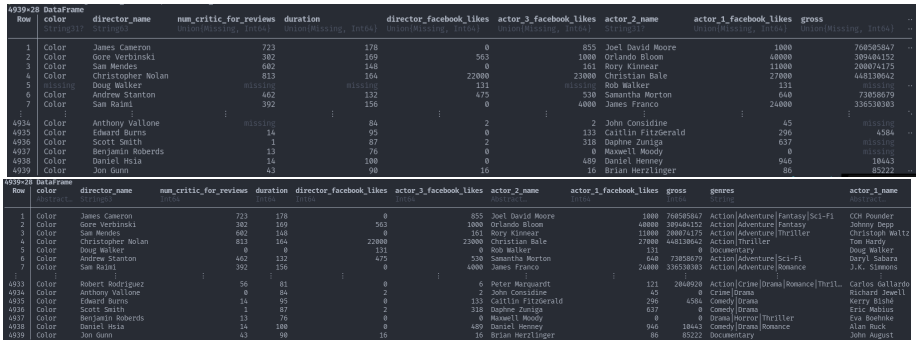
4.2 默认值替换

同时其他的值变为其默认值

名称	默认值
color	Color
num critic for reviews	0
duration	0
director facebook likes	0
actor 3 facebook likes	0
actor 2 name	
actor 1 facebook likes	0
gross	0
genres	
actor 1 name	
movie title	
num voted users	0
cast total facebook likes	0
actor 3 name	
facenumber in poster	0
plot keywords	
movie imdb link	

num user for reviews	0
language	
country	
content rating	PG-0
budget	0
title year	0
actor 2 facebook likes	0
imdb score	0
aspect ratio	0
movie facebook likes	0

于是就对每一列进行一次变化: 如果每一行的数据是 missing, 就将其替换为默认值, 见图4.2



Row	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes	actor_2_name	actor_1_facebook_likes	gross
1	Color	James Cameron	723	178	0	855	Joel David Moore	1000	760505847
2	Color	Gore Verbinski	302	169	563	1000	Orlando Bloom	40000	309404152
3	Color	Sam Mendes	602	148	0	161	Rory Kinnear	11000	200074175
4	Color	Christopher Nolan	813	164	22000	23000	Christian Bale	27000	448130642
5	missing	Doug Walker	missing	missing	131	missing	Rob Walker	131	missing
6	Color	Andrew Stanton	462	132	475	530	Samantha Morton	640	73058679
7	Color	Sam Raimi	392	156	0	4000	James Franco	24000	336530303
4934	Color	Anthony Vallone	missing	84	2	2	John Considine	45	missing
4935	Color	Edward Burns	14	95	0	133	Caitlin FitzGerald	296	4584
4936	Color	Scott Smith	1	87	2	318	Daphne Zuniga	637	missing
4937	Color	Benjamin Roberds	13	76	0	0	Maxwell Moody	0	missing
4938	Color	Daniel Hsia	14	100	0	489	Daniel Henney	946	10443
4939	Color	Jon Gunn	43	90	16	16	Brian Herzlinger	86	85222

Row	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes	actor_2_name	actor_1_facebook_likes	gross	genres	actor_1_name
1	Color	James Cameron	723	178	0	855	Joel David Moore	1000	760505847	Action|Adventure|Fantasy|Sci-Fi	CCH Pounder
2	Color	Gore Verbinski	302	169	563	1000	Orlando Bloom	40000	309404152	Action|Adventure|Fantasy	Johnny Depp
3	Color	Sam Mendes	602	148	0	161	Rory Kinnear	11000	200074175	Action|Adventure|Thriller	Christoph Waltz
4	Color	Christopher Nolan	813	164	22000	23000	Christian Bale	27000	448130642	Action|Thriller	Tom Hardy
5	Color	Doug Walker	0	0	131	0	Rob Walker	131	0	Documentary	Doug Walker
6	Color	Andrew Stanton	462	132	475	530	Samantha Morton	640	73058079	Action|Adventure|Sci-Fi	Daryl Sabara
7	Color	Sam Raimi	392	156	0	4000	James Franco	24000	336530303	Action|Adventure|Romance	J.K. Simmons
4933	Color	Robert Rodriguez	56	81	0	6	Peter Marquardt	121	2040920	Action|Crime|Drama|Romance|Thril…	Carlos Gallardo
4934	Color	Anthony Vallone	0	84	2	2	John Considine	45	0	Crime|Drama	Richard Jewell
4935	Color	Edward Burns	14	95	0	133	Caitlin FitzGerald	296	4584	Comedy|Drama	Kerry Bishé
4936	Color	Scott Smith	1	87	2	318	Daphne Zuniga	637	0	Comedy|Drama	Eric Mabius
4937	Color	Benjamin Roberds	13	76	0	0	Maxwell Moody	0	0	Drama|Horror|Thriller	Eva Boehnke
4938	Color	Daniel Hsia	14	100	0	489	Daniel Henney	946	10443	Comedy|Drama|Romance	Alan Ruck
4939	Color	Jon Gunn	43	90	16	16	Brian Herzlinger	86	85222	Documentary	John August

图 8: 处理 missing

5 数据合并

5.1 通过 join 进行合并

读取数据表, 通过 join 表上的 num 列对两张表进行合并, 图9

10×4 DataFrame

Row	num	sex	institution	category
	Int64	String1	String7	String7
1	1	m	xupt	teacher
2	2	w	xupt	teacher
3	3	m	xupt	student
4	4	w	xupt	student
5	5	m	xupt	student
6	6	m	xupt	student
7	7	w	xupt	student
8	8	w	xupt	teacher
9	9	m	xupt	teacher
10	10	w	xupt	teacher

10×4 DataFrame

Row	num	name	book	date
	Int64	String7	String63	Float64
1	1	Tom	gone with the wind	3.1
2	2	Anna	The scarlet letter	4.3
3	3	Jerry	The adventures of Tom Sawyer	3.18
4	4	Cyning	Tales of two cities	5.21
5	5	Peter	Pride and Prejudice	5.3
6	6	Kevien	Uncle Tom's Cabin	4.27
7	7	June	The old man and the sea	3.3
8	8	Angel	Le Comte de Monte-Cristo	5.8
9	9	Tony	The Adventures of Alice in Wonde…	5.9
10	10	May	Gulliver's Travels	5.19

10×7 DataFrame

Row	num	sex	institution	category	name	book	date
	Int64	String1	String7	String7	String7	String63	Float64
1	1	m	xupt	teacher	Tom	gone with the wind	3.1
2	2	w	xupt	teacher	Anna	The scarlet letter	4.3
3	3	m	xupt	student	Jerry	The adventures of Tom Sawyer	3.18
4	4	w	xupt	student	Cyning	Tales of two cities	5.21
5	5	m	xupt	student	Peter	Pride and Prejudice	5.3
6	6	m	xupt	student	Kevien	Uncle Tom's Cabin	4.27
7	7	w	xupt	student	June	The old man and the sea	3.3
8	8	w	xupt	teacher	Angel	Le Comte de Monte-Cristo	5.8
9	9	m	xupt	teacher	Tony	The Adventures of Alice in Wonde…	5.9
10	10	w	xupt	teacher	May	Gulliver's Travels	5.19

图 9: 数据合并

```
["lab1/julia/file/4ReaderInformation.csv",
     "lab1/julia/file/4ReaderRentRecode.csv"] .|>
CSV.File .|>
DataFrame |>
dates -> begin
    println(dates...)
    innerjoin(dates..., on = :num) |>
    file -> begin
        file |> println
        CSV.write("lab1/julia/file/join.csv", file)
    end
end
```

6 PCA

6.1 绘制四维图像

可视化四维数据, 我对数据依据种类进行了分组, 对每组进行了绘图

其中横轴, 数轴, 纵轴, 颜色分别代表: 萼片长度 (Sepal length), 萼片宽度 (Sepalwidth), 花瓣长度 (Petallength), 花瓣宽度 (Petalwidth)

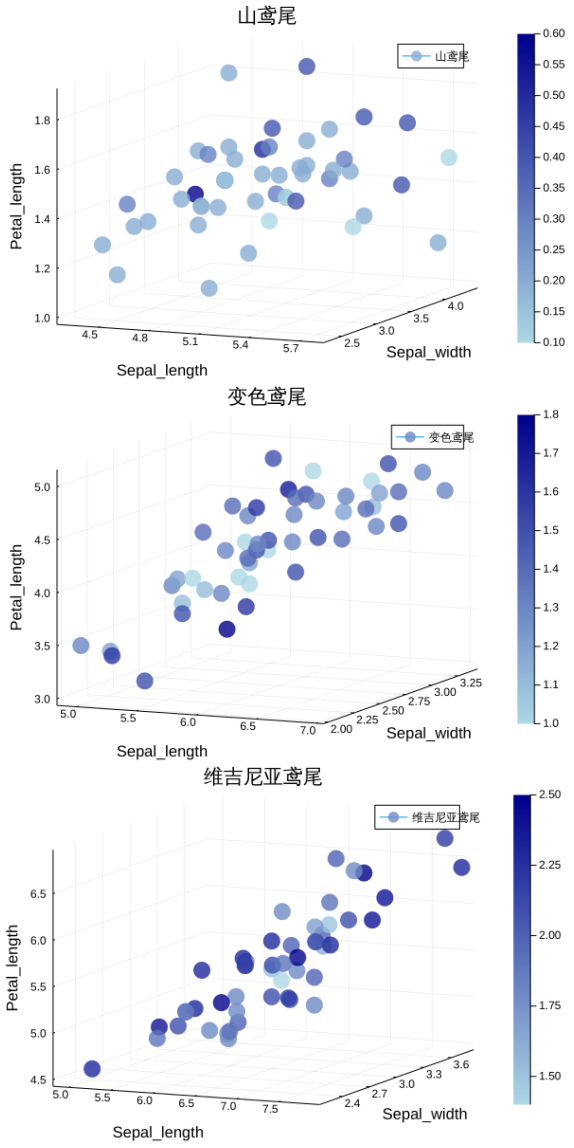


图 10: 山鸢尾花, 变色鸢尾花, 维吉尼亚鸢尾花

6.2 实现 PCA

PCA 实现步骤

首先是算法思路设有 n 条 d 维数据。

1. 将原始数据按列组成 n 行 d 列矩阵 X
2. 将 X 的每一列 (代表一个属性) 进行零均值化，即减去这一列的均值
3. 求出协方差矩阵 $\frac{1}{m}XX^T$
4. 求出协方差矩阵的特征值及对应的特征向量
5. 将特征向量按对应特征值大小从上到下按行排列成矩阵，取前 k 行组成矩阵 P
6. $Y = PX$ 即为降维到 k 维后的数据

```
data = copy(mat)
data = select!(data, Not([:Species])) |> Matrix
(values, vectors) = data |> Statistics.cov |> eigen
p = last(sortperm(values), 2) |> x -> vectors[:, x]
data * p
```

```
150×2 Matrix{Float64}:
 5.64133  -2.82714
 5.14517  -2.79595
 5.17738  -2.62152
 5.0036   -2.76491
 5.64865  -2.78275
 6.06251  -3.23145
 5.23262  -2.69045
 5.48513  -2.88486
 4.74393  -2.62338
 ⋮
 4.7398   -6.92542
 5.5907   -8.07467
 5.61823  -7.93073
 5.50214  -7.45536
 4.9397   -7.03701
 5.39324  -7.27539
 5.4306   -7.41297
 5.03184  -6.90101
```

图 11: PCA 结果

结果: 图11

附录 A　代码

```
using XLSX;
using CSV;
using DataFrames;
using Plots;
using Dates;
using Statistics;
using StatsPlots
import PyPlot;
using LinearAlgebra

function free()
    quality =
        "lab1/julia/file/xian_guangdian.csv" |>
        CSV.File |>
        DataFrame |>
        data ->
            begin
                combine(nrow, groupby(select(data, :客户等级), :客户等级)) |>
                data -> rename(data, :nrow => "用户数量") |> println
                combine(nrow, groupby(select(data, [:客户等级, :网络类型]), [:客户等级, :网络类型]))
            end |>
            data ->
                rename(data, :nrow => :quantity, :网络类型 => :net_kind, :客户等级 => :user_level)
    data = combine(groupby(quality, :net_kind), [:user_level, :quantity])
    dict = Dict(
        "5星ABD客户" => "star_5ABD",
        "离线" => "out_link",
        "3星AB客户" => "star_3AB",
        "1星D客户" => "star_1D",
        "1星A客户" => "star_1A",
        "VIP商业个人客户" => "vip",
        "3星AD客户" => "start_3AD",
    )
    1:(data|>nrow) .|>
    i -> begin
        data[i, :net_kind] =
            Dict("农网用户" => "village", "城网用户" => "city", " " => "unknown")[data[
                i,
                :net_kind,
            ]]
        data[i, :user_level] = dict[data[i, :user_level]]
    end
    gp = groupby(data, :net_kind)
    gp |>
    keys .|>
    kind -> @df combine(gp[kind], [:user_level, :quantity]) plot(
        :user_level,
        :quantity,
```

```
        label = "$kind",
    ) |> fig -> savefig(fig, "lab1/julia/images/first_$kind")
end

function draw_plot()
    file_path = "lab1/julia/file/xian_beijing_salary.xlsx"
    salarys = DataFrame(XLSX.readdata(file_path, "Sheet1!C3:D14"), :auto) .|> identity
    salary = [salarys.x1, salarys.x2]

    println("mean:$(salary .|> Statistics.mean)")
    println("var:$(salary .|> Statistics.var)")
    println("std:$(salary .|> Statistics.std)")
    println("cov:$(salary |> Statistics.cov)")
    println("cor:$(salary .|> Statistics.cor)")

    violin(["Xi'an"], salarys.x1, label = "Xi'an")
    violin!(["Beijing"], salarys.x2, label = "Beijing") |>
    fig -> savefig(fig, "lab1/julia/images/violin")
    boxplot(["Xi'an"], salarys.x1, label = "Xi'an")
    boxplot!(["Beijing"], salarys.x2, label = "Beijing") |>
    fig -> savefig(fig, "lab1/julia/images/box")

end

function map_transform()
    file_path = "lab1/julia/file/3movie_metadata.csv"
    movie_metadata = file_path |> CSV.File |> DataFrame
    dropmissing!(movie_metadata, :director_name)
    dict = Dict(
        :color => "Color",
        :num_critic_for_reviews => 0,
        :duration => 0,
        :director_facebook_likes => 0,
        :actor_3_facebook_likes => 0,
        :actor_2_name => "",
        :actor_1_facebook_likes => 0,
        :gross => 0,
        :genres => "",
        :actor_1_name => "",
        :movie_title => "",
        :num_voted_users => 0,
        :cast_total_facebook_likes => 0,
        :actor_3_name => "",
        :facenumber_in_poster => 0,
        :plot_keywords => "",
        :movie_imdb_link => "",
        :num_user_for_reviews => 0,
        :language => "",
        :country => "",
        :content_rating => 0,
        :budget => 0,
```

```
        :title_year => 0,
        :actor_2_facebook_likes => 0,
        :imdb_score => 0,
        :aspect_ratio => 0,
        :movie_facebook_likes => 0,
    )
    dict |>
    keys .|>
    key -> transform!(
        movie_metadata,
        :,
        key => (col -> col .|> each -> if ismissing(each)
            Dict[key]
        else
            each
        end) => key,
    )
end

function join_compine()
    ["lab1/julia/file/4ReaderInformation.csv", "lab1/julia/file/4ReaderRentRecode.csv"] .|>
    CSV.File .|>
    DataFrame |>
    dates -> begin
        println(dates...)
        innerjoin(dates..., on = :num) |>
        file -> begin
            file |> println
            CSV.write("lab1/julia/file/join.csv", file)
        end
    end
end

function self_pca()
    mat = "lab1/julia/file/5iris.csv" |> CSV.File |> DataFrame
    gp = groupby(mat, :Species)
    names = [:Sepal_length, :Sepal_width, :Petal_length, :Petal_width, :Species]
    @df gp[1] plot(
        :Sepal_length,
        :Sepal_width,
        :Petal_length,
        zcolor = reverse(:Petal_width),
        m = (10, 0.8, :blues, Plots.stroke(0)),
        fontfamily = "Yahei",
        xlabel = "Sepal_length",
        ylabel = "Sepal_width",
        zlabel = "Petal_length",
        title = "山鸢尾",
        label = "山鸢尾",
        w = 0,
    )
```

```
    @df gp[2] plot(
        :Sepal_length,
        :Sepal_width,
        :Petal_length,
        zcolor = reverse(:Petal_width),
        m = (10, 0.8, :blues, Plots.stroke(0)),
        fontfamily = "Yahei",
        xlabel = "Sepal_length",
        ylabel = "Sepal_width",
        zlabel = "Petal_length",
        title = "变色鸢尾",
        label = "变色鸢尾",
        w = 0,
    )
    @df gp[3] plot(
        :Sepal_length,
        :Sepal_width,
        :Petal_length,
        zcolor = reverse(:Petal_width),
        m = (10, 0.8, :blues, Plots.stroke(0)),
        fontfamily = "Yahei",
        xlabel = "Sepal_length",
        ylabel = "Sepal_width",
        zlabel = "Petal_length",
        title = "维吉尼亚鸢尾",
        label = "维吉尼亚鸢尾",
        w = 0,
    )
    data = copy(mat)
    data = select!(data, Not([:Species])) |> Matrix
    (values, vectors) = data |> Statistics.cov |> eigen
    p = last(sortperm(values), 2) |> x -> vectors[:, x]
    data * p
end

free();
draw_plot();
map_transform();
join_compine();
self_pca();
```