

# 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 25 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 matplotlib

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

<b>1 概述</b>	<b>4</b>
<b>2 数据可视化</b>	<b>5</b>
2.1 解析文件 . . . . .	5
2.2 分组 . . . . .	6
2.3 绘制折线图 . . . . .	7
2.4 分析数据 . . . . .	7
<b>3 数据处理</b>	<b>8</b>
3.1 求的统计数据 . . . . .	8
3.2 绘制箱型图和小提琴图 . . . . .	8
<b>4 数据预处理</b>	<b>9</b>
4.1 去除丢失数据行 . . . . .	9
4.2 默认值替换 . . . . .	9
<b>5 数据合并</b>	<b>11</b>
5.1 通过 join 进行合并 . . . . .	11
<b>6 PCA</b>	<b>12</b>
6.1 绘制四维图像 . . . . .	12
6.2 实现 PCA . . . . .	13
<b>A 代码</b>	<b>14</b>

# 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国, 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 (Sepal width), 花瓣长度 (Petal length), 花瓣宽度 (Petal width), 还有一个变量是其所属的品种类别变量 (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.

6999x14 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
	String15	String15	String15	String15	String15	String15	String
1	ys0015561	宽带10M产品	6/25/2017	正常使用	正常	c10002109695	新用户
2	ys0023214	宽带4M产品	7/24/2017	正常使用	正常	c10002109697	新用户
3	ys0002301	宽带10M产品	6/25/2017	正常使用	正常	c10002109701	新用户
4	ys0022740	宽带10M产品	2/26/2018	正常使用	正常	c10002114105	新用户
5	ys0056409	宽带10M产品	1/3/2018	正常使用	正常	c10002114084	新用户
6	ys0003401	宽带10M产品	5/6/2017	正常使用	正常	c10002110913	新用户
7	ys0074044	宽带10M产品	10/29/2016	已停用	欠费停机	c10006305966	新用户
8	ys0000014	宽带10M产品	1/5/2018	正常使用	正常	c10002120172	新用户
9	ys0070152	宽带4M产品	7/15/2015	已停用	客户报停	c10002120173	新用户
10	ys0040259	宽带10M产品	8/25/2017	正常使用	正常	c10002120293	新用户
11	ys0074065	宽带10M产品	11/10/2017	正常使用	正常	c10002123835	新用户
12	ys0041355	宽带10M产品	5/27/2017	正常使用	正常	c10002125494	新用户
13	ys0057767	宽带10M产品	11/3/2016	已停用	欠费停机	c10007202177	新用户
14	ys0056459	宽带10M产品	3/6/2017	正常使用	正常	c10002125840	新用户
15	ys0000035	宽带10M产品	2/5/2018	正常使用	正常	c10002125908	新用户
16	ys0046632	宽带10M产品	9/16/2017	正常使用	正常	c10002126686	新用户
17	ys0130153	宽带10M产品	1/16/2018	正常使用	正常	c10002126687	新用户
18	ys0016491	宽带10M产品	11/29/2017	正常使用	正常	c10002205661	新用户
19	ys0045160	宽带10M产品	12/26/2017	正常使用	正常	c10002205657	新用户
20	ys0090756	宽带10M产品	11/4/2016	已停用	欠费停机	c10006370500	新用户
21	ys0027721	宽带10M产品	1/22/2018	正常使用	正常	c10002204543	新用户
22	ys0031693	宽带10M产品	5/13/2018	正常使用	正常	c10002204543	新用户
23	ys0007107	宽带10M产品	5/23/2017	正常使用	正常	c10002209922	新用户
24	ys0018042	宽带10M产品	4/16/2017	正常使用	正常	c10002209924	新用户
25	ys0046621	宽带10M产品	10/21/2017	正常使用	正常	c10002209927	新用户
26	ys0032342	宽带10M产品	8/20/2017	正常使用	正常	c10002209929	新用户
27	ys0125430	宽带10M产品	11/30/2017	正常使用	正常	c10002209946	新用户
28	ys0126100	宽带10M产品	7/4/2017	正常使用	正常	c10002217047	新用户
29	ys0007979	宽带10M产品	10/11/2017	正常使用	正常	c10002236312	新用户
30	ys0065099	宽带10M产品	6/11/2017	正常使用	正常	c10002236313	新用户
31	ys0043657	宽带10M产品	10/11/2017	正常使用	正常	c10002236316	新用户
32	ys0076592	宽带10M产品	3/16/2017	正常使用	正常	c10002236322	新用户
33	ys0037044	宽带4M产品	6/29/2017	正常使用	正常	c10002236332	新用户
34	ys0000370	宽带10M产品	2/11/2018	正常使用	正常	c10002236334	新用户
35	ys0002223	宽带10M产品	6/29/2017	正常使用	正常	c10002236335	新用户
36	ys0075274	宽带10M产品	3/21/2018	正常使用	正常	c10002236356	新用户
37	ys0064635	宽带4M产品	8/25/2017	正常使用	正常	c10002236337	新用户
38	ys0005004	宽带10M产品	10/6/2017	正常使用	正常	c10002239100	新用户
39	ys0023467	宽带10M产品	11/30/2017	正常使用	正常	c10002241607	新用户
40	ys0127407	宽带10M产品	12/23/2017	正常使用	正常	c10002247402	新用户
41	ys0062065	宽带10M产品	1/19/2018	正常使用	正常	c10002261307	新用户
42	ys0044083	宽带10M产品	8/25/2017	正常使用	正常	c10002262665	新用户
43	ys0044075	宽带10M产品	10/6/2017	正常使用	正常	c10002262670	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" => "star_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] = dict[data[i, :net_kind]]
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 String31	用户数量 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

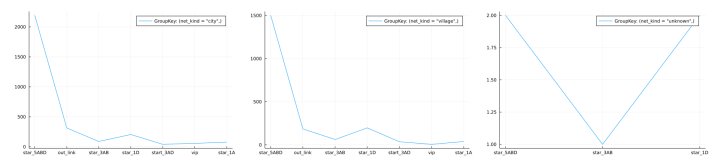


图 4: 城市居民, 农村, 未登记





## 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

	String1	String2	numcriticforreviews	numcriticforreviews	numcriticforreviews	numcriticforreviews	String3	numcriticforreviews	numcriticforreviews
1	Color	James Cameron	723	178	0	855	Joel David Moore	1800	76
2	Color	Gore Verbinski	382	169	563	1800	Orlando Bloom	40000	38
3	Color	Sam Mendes	682	148	0	161	Rory Kinnear	11800	20
4	Color	Christopher Nolan	813	164	22000	23000	Christian Bale	27000	44
5	Color	Doug Walker	615	131	0	131	Rob Walker	131	7
6	Color	Andrew Stanton	462	132	475	530	Samantha Morton	640	33
7	Color	Sam Raimi	392	126	0	4000	James Franco	24000	20
8	Color	Nathan Greno	324	100	15	284	Donna Murphy	799	20
9	Color	Joss Whedon	635	141	0	13000	Robert Downey Jr.	20000	42
10	Color	David Yates	375	153	282	10000	Daniel Radcliffe	25000	38
11	Color	Zack Snyder	671	183	0	2000	Lauren Cohan	15000	23
12	Color	Bryan Singer	434	159	0	393	Mathieu Amalric	18000	20
13	Color	Mark Forster	483	185	395	393	Mathieu Amalric	451	16
14	Color	Gore Verbinski	313	151	563	1800	Orlando Bloom	40000	42
15	Color	Gore Verbinski	450	159	563	1000	Ruth Wilson	40000	8
16	Color	Zack Snyder	733	143	0	748	Christopher Meloni	15000	29
5829	Black and White	Ivan Kavanagh	12	83	18	0	Michael Parle	18	7
5830	Color	Kiyoshi Kurosawa	78	111	62	5	Anna Malasawa	89	21
5831	Color	Taddeo Garcia	84	84	5	12	Michael Cortez	21	785
5832	Color	Thomas L. Phillips	13	82	120	84	Joe Coffey	785	789
5833	Color	Ash Baron-Cohen	18	98	3	152	Stanley B. Herman	291	0
5834	Color	Shane Carruth	143	77	291	0	David Sullivan	121	45
5835	Color	Neill LaBute	35	80	0	0	Eduar Tenebraga	296	887
5836	Color	Robert Rodriguez	56	81	0	6	Peter Marquardt	861	0
5837	Color	Anthony Vellone	1	84	2	218	Stephen Zupiga	946	0
5838	Color	Edward Burns	14	95	0	133	Caillie Fitzgerald	861	0
5839	Color	Scott Smith	1	87	2	218	Valerie Curry	861	0
5840	Color	Benjamin Roberts	43	43	0	0	Maxwell Moody	0	0
5841	Color	Daniel Hsia	13	76	0	489	Daniel Henry	946	0
5842	Color	Daniel Hsia	14	180	0	18	Brian Herzlinger	946	0
5843	Color	Jon Gunn	43	90	10	18	Brian Herzlinger	946	0

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	0
actor 1 facebook likes	0
gross	0
genres	0
actor 1 name	0
movie title	0
num voted users	0
cast total facebook likes	0
actor 3 name	0
facenumber in poster	0
plot keywords	0
movie imdb link	0

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

6599x26 DataFrame										
Row	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes	actor_2_name	actor_1_facebook_likes	gross	imdb_score
1	Color	James Cameron	723	178	0	855	Joel David Moore	1800	764585967	
2	Color	Gore Verbinski	362	169	563	1000	Orlando Bloom	40000	309404152	
3	Color	Sam Mendes	682	148	0	161	Rory Kinnear	11000	208074175	
4	Color	Christopher Nolan	813	164	22000	23000	Christian Bale	27000	448138062	
5	Color	Doug Walker	0	0	131	0	Rob Walker	131	0	
6	Color	Andrew Stanton	462	132	475	520	Samantha Morton	640	73058670	
7	Color	Sam Raimi	392	156	0	4000	James Franco	24000	336530303	
4934	Color	Anthony Vallone	0	84	2	2	John Considine	45	0	
4935	Color	Edward Burns	14	95	0	133	Caillin Fitzgerald	296	4584	
4936	Color	Scott Smith	1	87	2	318	Daphne Zuniga	637	0	
4937	Color	Benjamin Roberts	13	76	0	0	Maxwell Moody	0	0	
4938	Color	Daniel Heise	14	100	0	489	Daniel Henney	946	18443	
4939	Color	Jon Gunn	43	98	16	16	Brian Herzlinger	86	85222	

6599x26 DataFrame											
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	1800	764585967	Action Adventure Fantasy Sci-Fi	Coln Pounder
2	Color	Gore Verbinski	362	169	563	1000	Orlando Bloom	40000	309404152	Action Adventure Fantasy	Johnny Depp
3	Color	Sam Mendes	682	148	0	161	Rory Kinnear	11000	208074175	Action Adventure Thriller	Christopher Waltz
4	Color	Christopher Nolan	813	164	22000	23000	Christian Bale	27000	448138062	Action Thriller	Tom Hanks
5	Color	Doug Walker	0	0	131	0	Rob Walker	131	0	Documentary	Doug Walker
6	Color	Andrew Stanton	462	132	475	520	Samantha Morton	640	73058670	Action Adventure Sci-Fi	Daryl Sabara
7	Color	Sam Raimi	392	156	0	4000	James Franco	24000	336530303	Action Crime Drama Romance Thrill	Jay Simons
4933	Color	Robert Rodriguez	56	81	0	6	Peter Marquardt	121	2848920	Action Crime Drama Romance Thrill	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	Caillin Fitzgerald	296	4584	Comedy Drama	Kerry Riche
4936	Color	Scott Smith	1	87	2	318	Daphne Zuniga	637	0	Comedy Drama	Eric Mabius
4937	Color	Benjamin Roberts	13	76	0	0	Maxwell Moody	0	0	Drama Horror Thriller	Eric Robison
4938	Color	Daniel Heise	14	100	0	489	Daniel Henney	946	18443	Comedy Drama Romance	Alan Rick
4939	Color	Jon Gunn	43	98	16	16	Brian Herzlinger	86	85222	Documentary	Jon 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
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)

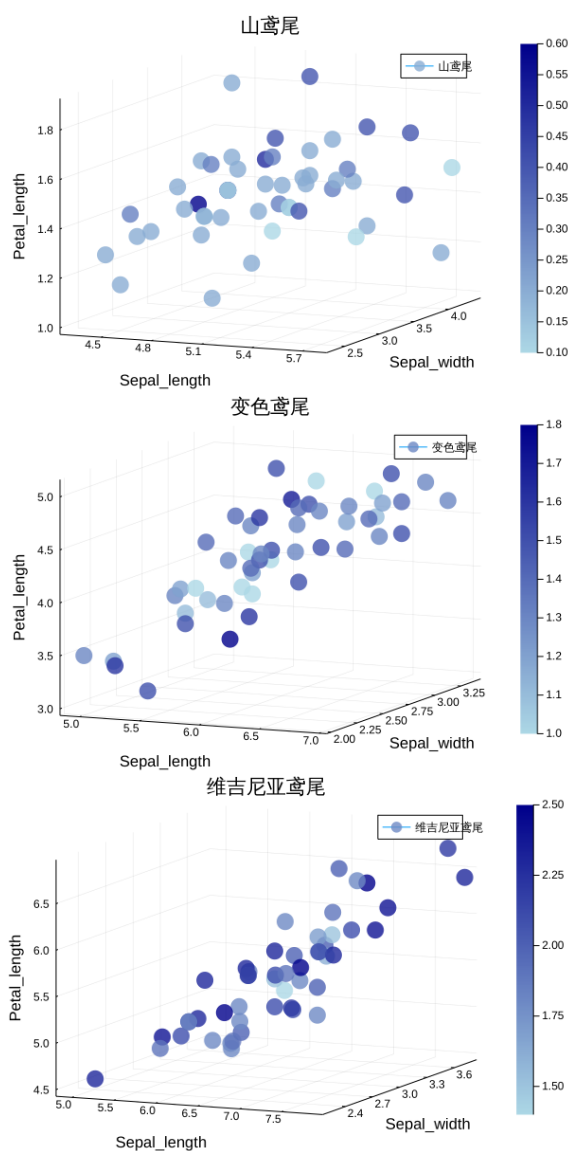


图 10: 山鸢尾花, 变色鸢尾花, 维吉尼亚鸢尾花

## 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,
    )
end

```

```

        :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,
    )
end

```



```

@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();

```