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

年 月 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](https://github.com/lovebaihezi/lab/tree/main/data-process/lab1/julia)下找到.

[c]

# 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。

# 数据可视化

## 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]("https://github.com/lovebaihezi/lab/blob/main/data-process/lab1/julia/file/xian_guangdian.csv")

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](#fig:csvinfo).

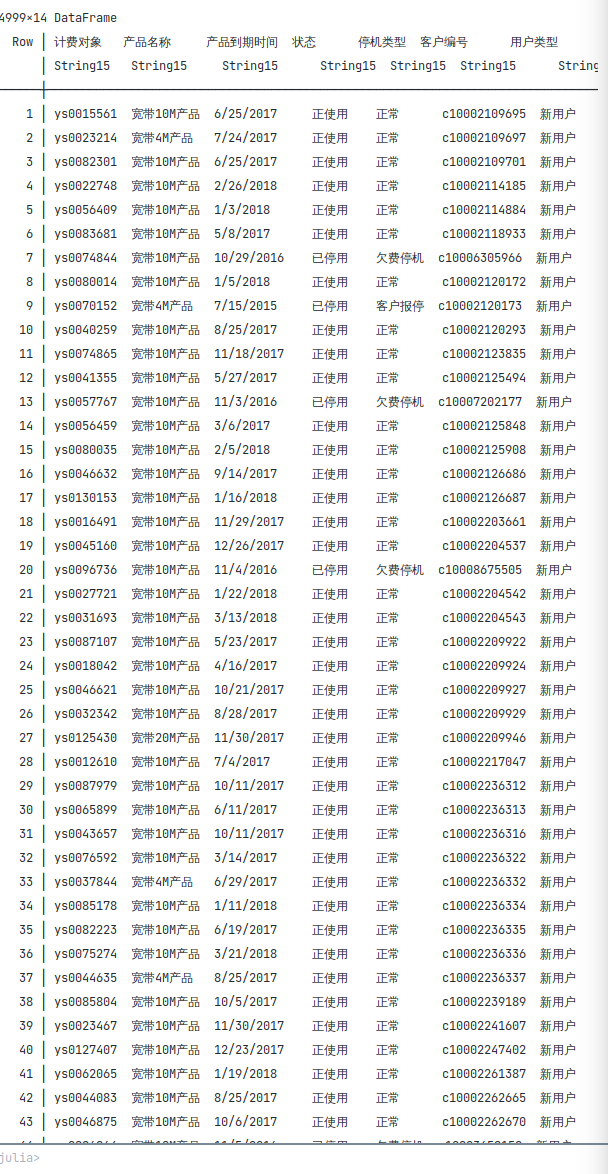


Figure : 广电信息CSV

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")

## 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](#fig:customgroup).

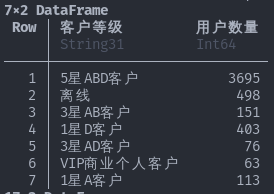


Figure : 分组结果图

再将每组一网络类型进行分组, 图[3](#fig:netgroup).

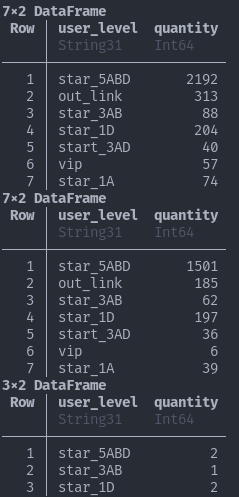
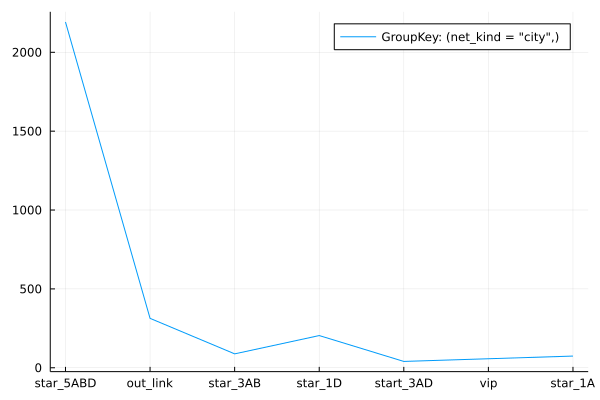
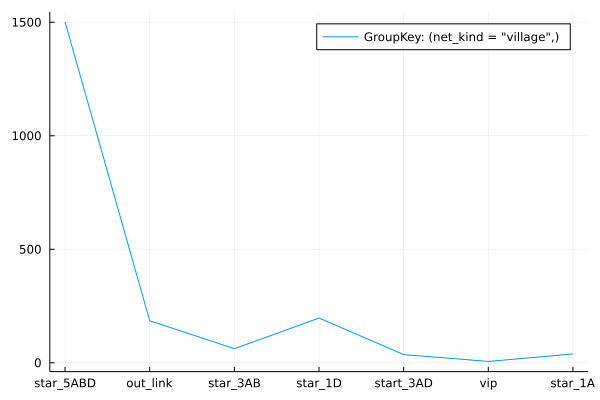
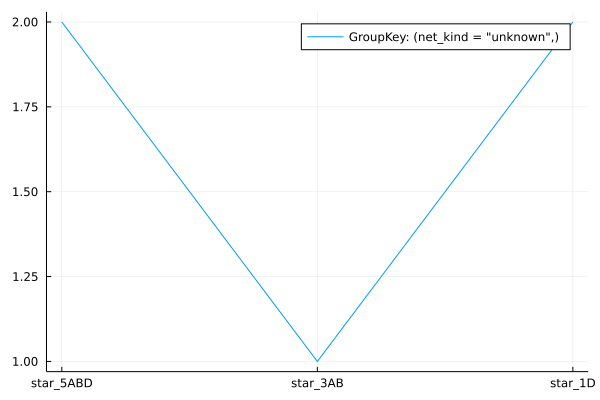


Figure : 分组结果图

## 绘制折线图

然后将每组画到折线图之上, 图[6](#fig:city)

## 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

# 数据处理

## 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[7](#fig:mean) 在使用Plots进行绘图,图[9](#fig:violin_box)

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")

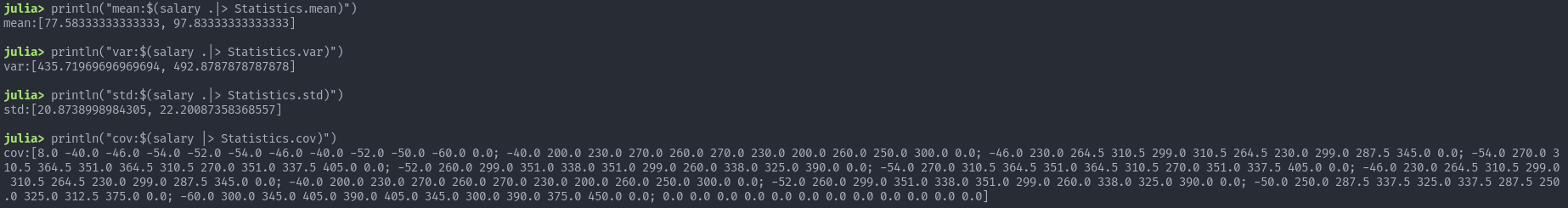
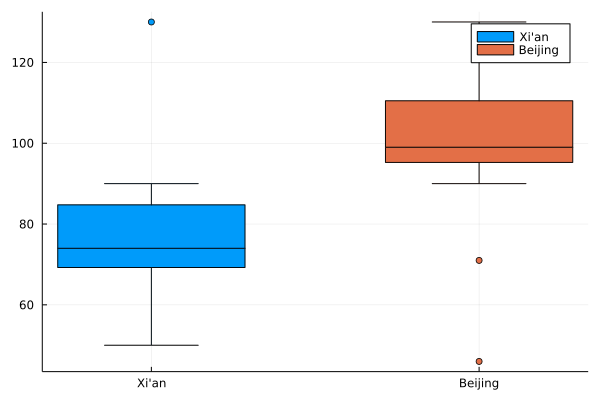
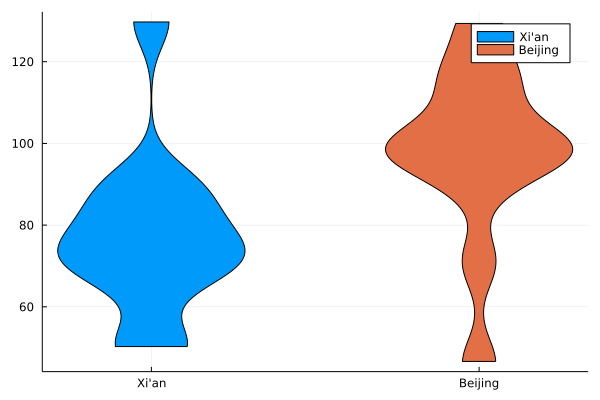


Figure : 均值, 方差, 标准差, 协方差

## 绘制箱型图和小提琴图

# 数据预处理

## 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[10](#fig:movie_metadata)

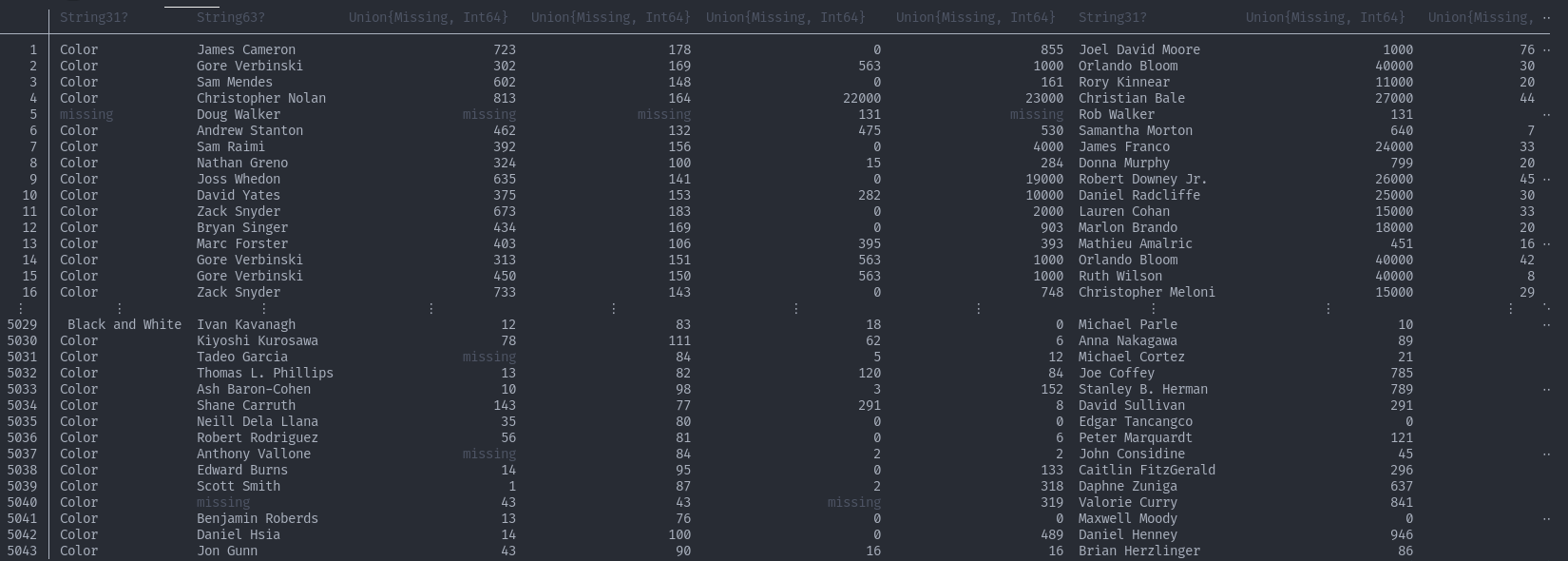


Figure : 电影元数据

为了将来数据处理的正常话,这里将导演空的一栏都去掉

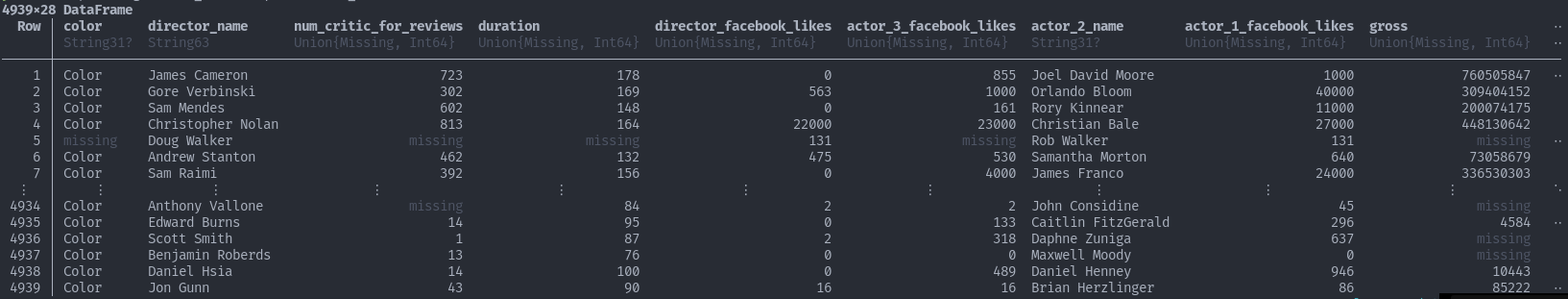
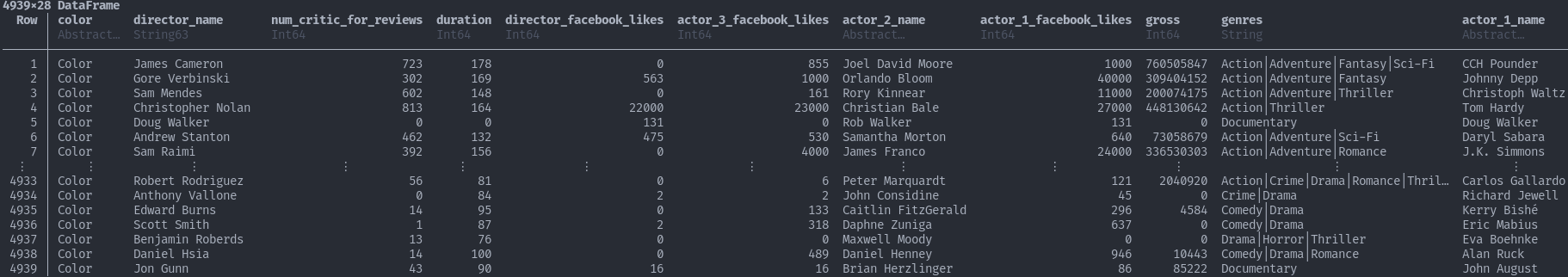
## 默认值替换

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[12](#fig:missing)

[fig:missing]  

# 数据合并

## 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[13](#fig:combine)

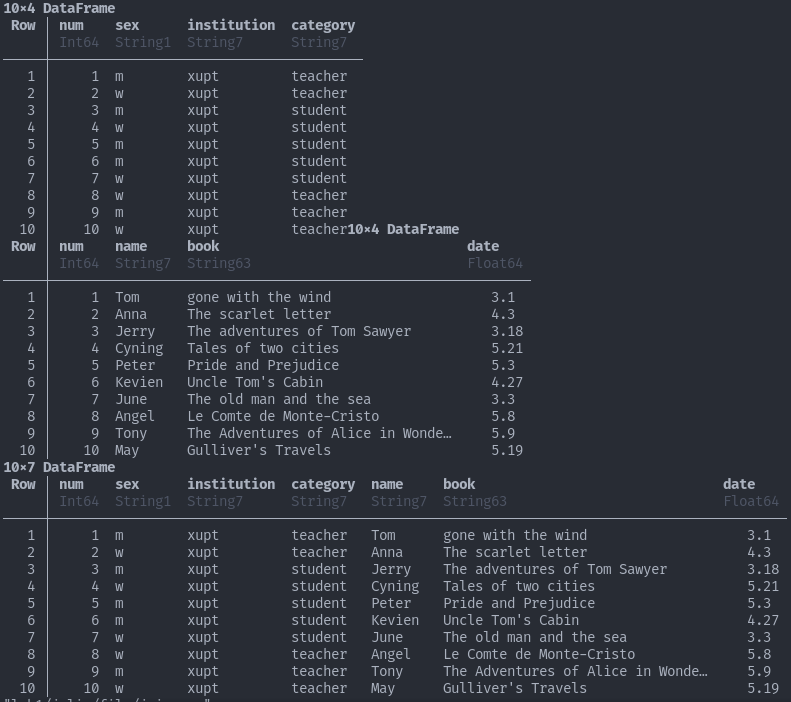


Figure : 数据合并

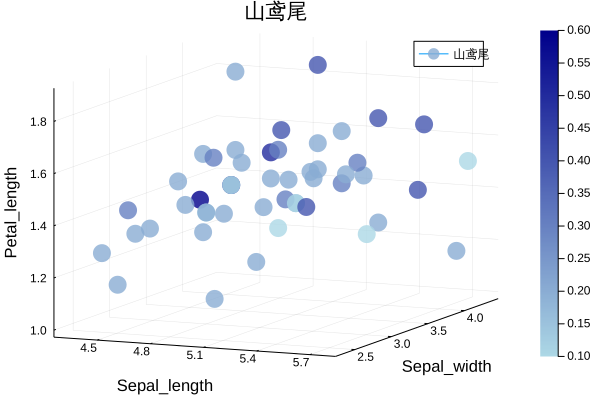
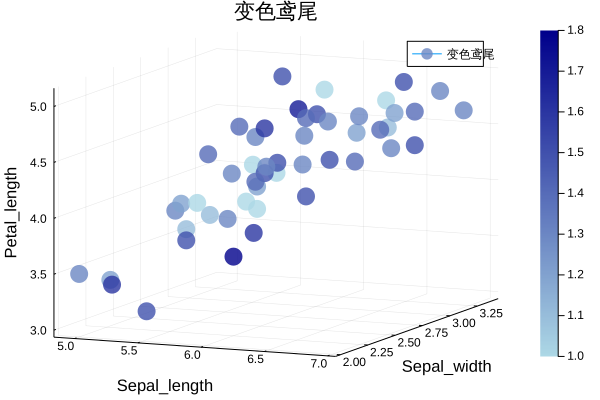
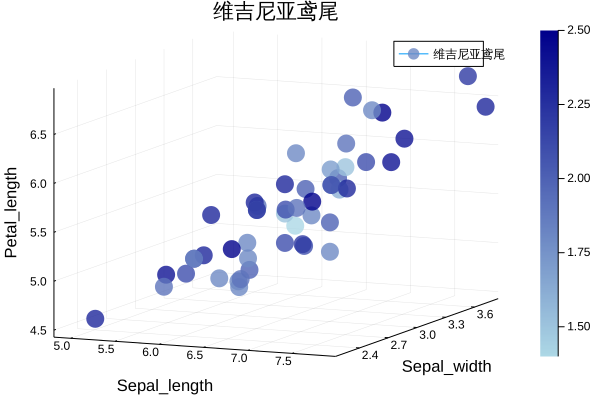
["lab1/julia/file/4ReaderInformation.csv",  
 "lab1/julia/file/4ReaderRentRecode.csv"] .|>  
CSV.File .|>  
DataFrame |>  
dates -> begin  
 println(dates...)  
 innerjoin(dates..., on = :num) |>  
 file -> begin  
 file |> println  
 CSV.write("lab1/julia/file/join.csv", file)  
 end  
end

# PCA

## 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)

## 实现PCA

PCA实现步骤

首先是算法思路 设有 条 维数据。

1. 将原始数据按列组成 行 列矩阵
2. 将 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
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 行组成矩阵
6. 即为降维到 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



Figure : PCA结果

结果:图[14](#fig:matrix)

# 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();