用 GitHub Copilot 构建机器学习模型

让小白也能做数据分析

张 丹

微软MVP | R语言实践者

北京青萌数海科技有限公司 CTO

个人博客：http://fens.me

个人介绍

张丹，**R语言实践者**，北京青萌数海科技有限公司CTO，微软MVP。

10年以上互联网应用架构经验，在R、大数据、数据分析等方面有深厚的积累。精通量化投资交易策略，熟悉中国金融二级市场、交易规则和投研体系。 熟悉数据学科方法论，在海关、药监、外汇等监管科技领域均有落地项目。

著有《R的极客理想：量化投资篇》、《R的极客理想：工具篇》、《R的极客理想：高级开发篇》，图书英文版被CRC出版集团引进，在美国发行。个人博客：http://fens.me 。

背景介绍

用机器学习做数据分析，是一种普遍的智能模型建模的思路。机器学习基于结构化数据，以统计概率为算法基础，计算快，解释性好，但是上手的门槛不低，需要有统计学的知识，以及对业务的理解。利用GitHub Copilot, 补齐短板，让小白也能做数据分析。

- 01 · 机器学习建模关键步骤
- 02 · 让AI生成代码：统计概览
- 03 · 让AI生成代码：机器学习
- 04 · 让AI生成代码：可视化工具
- 05 · 大模型生成代码 VS 古法手写代码

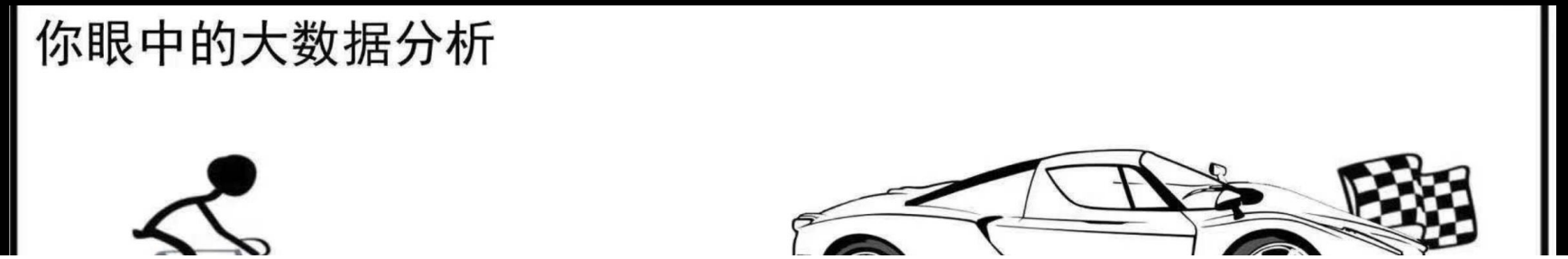
提问：

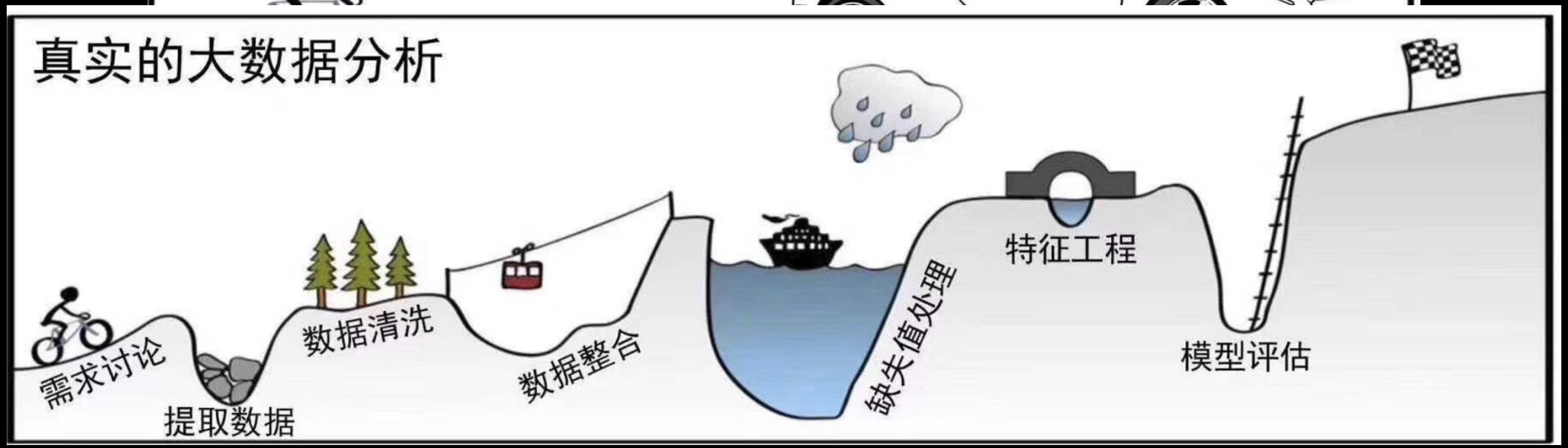
现场有多少人想学 **数据分析/机器学习**，但被建模门槛劝退？

机器学习建模关键步骤

机器学习模型建设步骤

数据分析并非快速直达结果，而是要经历一系列的问题解决和处理步骤





机器学习场景

适用于**结构化**、封闭领域任务，如分类、回归、聚类、推荐等，依赖明确的问题定义和静态数据分布。

典型场景：金融风控

进口冷冻牛肉风险识

	申报实例	商品编码		出口退税率	监管条件	检验检疫	更多信息
☐	牛肉	第1类 活动物;动物产品					
		02 肉及食用杂碎					
		0201300090 其他鲜或冷藏的去骨牛肉	[税目]	9%	4,7,A,B,x	R,S,P,Q	详情
		第4类 食品;饮料、酒及醋;烟草、烟草及烟草代用品的制品					
		16 肉、鱼、甲壳动物,软体动物及其他水生无脊椎动物的制品					
		1602509090 其他制作或保藏的牛肉,杂碎,血	[税目]	13%	A,B	R,S,P,Q	详情
☐	牛肉锤	第15类 贱金属及其制品					
		82 贱金属工具、器具、利口器、餐匙、餐叉及其零件					
		8205200000 手工锤子	[税目]	13%			详情
		8205510000 其他家用手工工具	[税目]	13%			详情
☐	牛肉刀	第15类 贱金属及其制品					
		82 贱金属工具、器具、利口器、餐匙、餐叉及其零件					
		8211920000 刃面固定的其他刀	[税目]	13%			详情
		8214900010 切菜刀等厨房用利口器	[税目]	13%	A	R	详情

Fold

cs Consignee	-		
No. Qualifier	-		
mporter City	PLYMOUTH		
Zip Code	48170		
stics Shipper	-		
No. Qualifier	-		
Exporter City	KARIYA		
Zip Code	448-8650		
Notify Party Name	AWTEC;AISIN WORLD CORP.OF AMERICA	Secondary Notify Party	-
Comm No.	;	Comm No. Qualifier	-
Notify Party Contact	;		
Notify Party Add.	14920 KEEL STREET;24330 GARNIER ST;;;	Notify Party City	PLYMOUTH;TORRANCE
Notify Party State	MI;CA	Zip Code	48170;90505
Notify Party Ctry	-		

数据概览

青萌数海

主要是对抽取的测试样本数据进行系统性描述和初步分析，目的是确保测试数据的质量、代表性和有效性。

了解数据结构

- 数据集共有多少条记录，多少个字段。
- 每个字段是数值型（如价格、年龄）、分类型（如性别、地区）、文本型还是日期时间型?

数据质量检查

- 哪些字段存在空值，缺失的比例有多高。
- 通过最大值、最小值或常识判断，是否存在明显不合理的数据（如年龄为200岁、价格为负数）
- 数据中是否存在完全重复的记录
- 同一字段的格式是否统一（如日期格式、单位等）

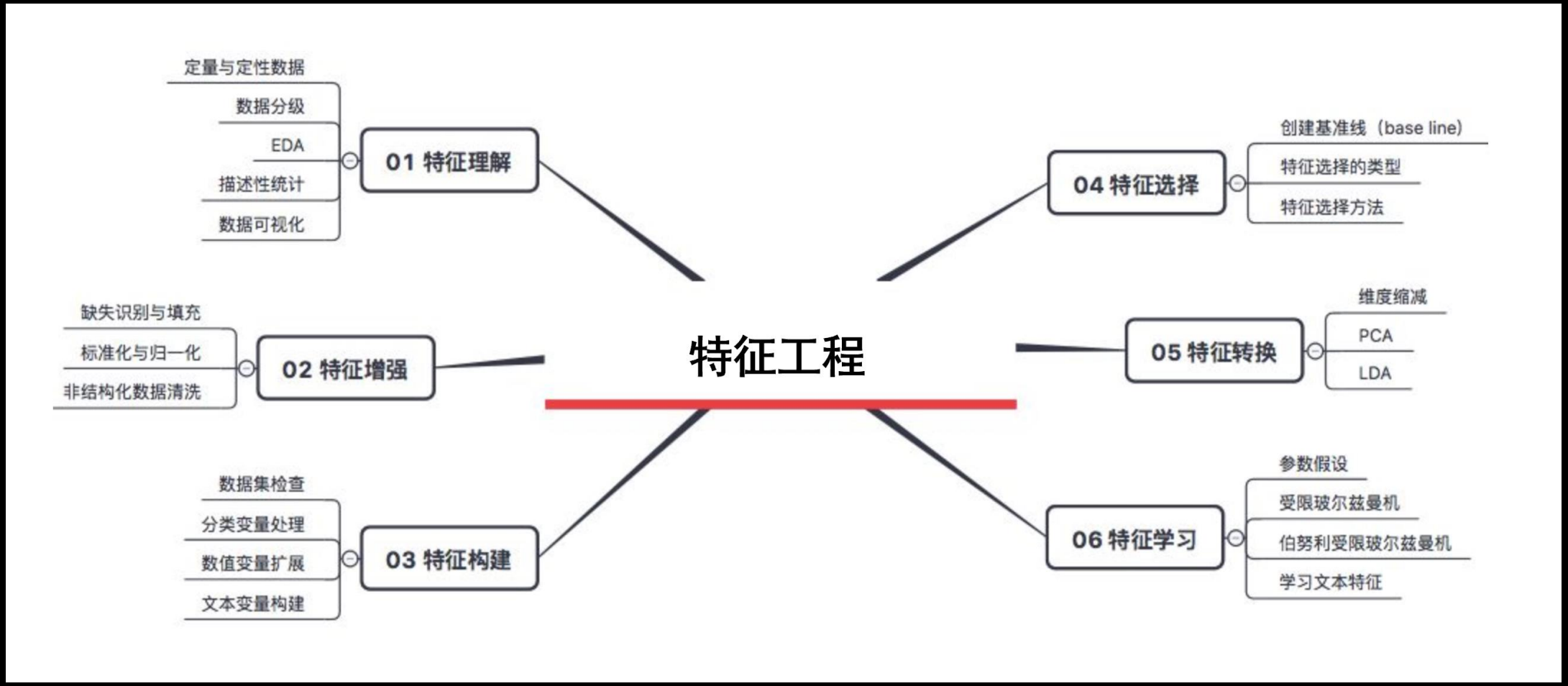
描述性统计分析

- 计算各数值型字段（如金额、数量、风险分值等）的均值、中位数、标准差、最小值、最大值
- 统计分类字段的频数分布
- 分析风险分值的整体分布情况（是否呈偏态分布、集中趋势等）

交叉维度分析

- 通过简单的相关性分析或交叉表，初步了解不同字段之间的关联。
- 分类目标的样本是否均衡？是否存在某些类别样本极少的情况?
- 数据随时间的变化趋势如何?

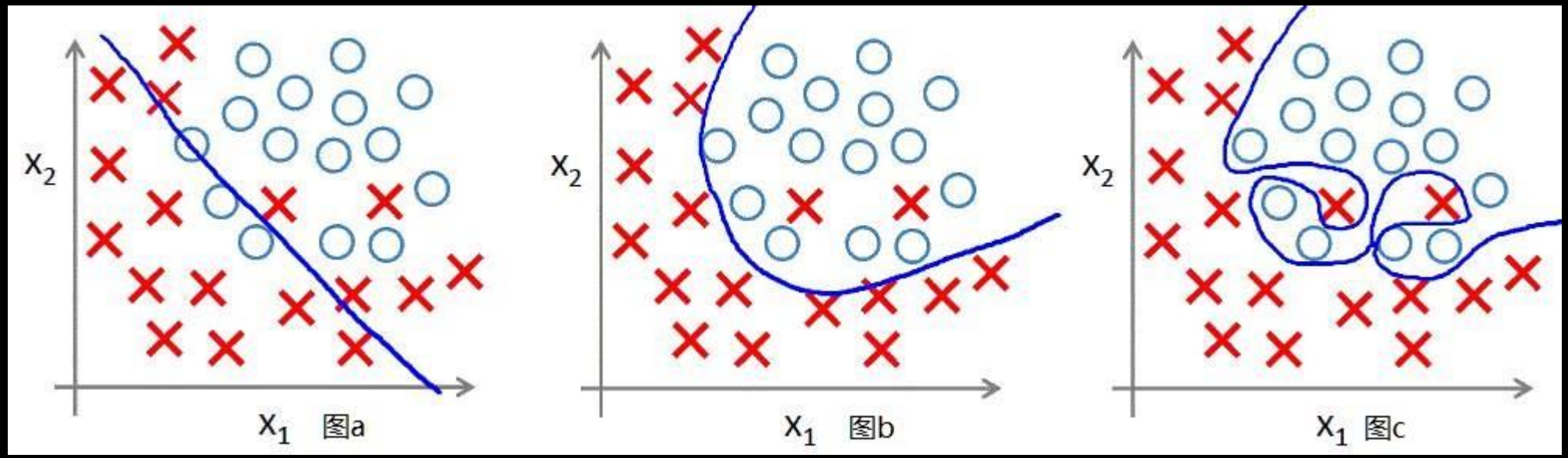
特征工程


特征工程
01 特征理解
定量与定性数据
数据分级
EDA
描述性统计
数据可视化
02 特征增强
缺失识别与填充
标准化与归一化
非结构化数据清洗
03 特征构建
数据集检查
分类变量处理
数值变量扩展
文本变量构建
04 特征选择
创建基准线（base line）
特征选择的类型
特征选择方法
05 特征转换
维度缩减
PCA
LDA
06 特征学习
参数假设
受限玻尔兹曼机
伯努利受限玻尔兹曼机
学习文本特征

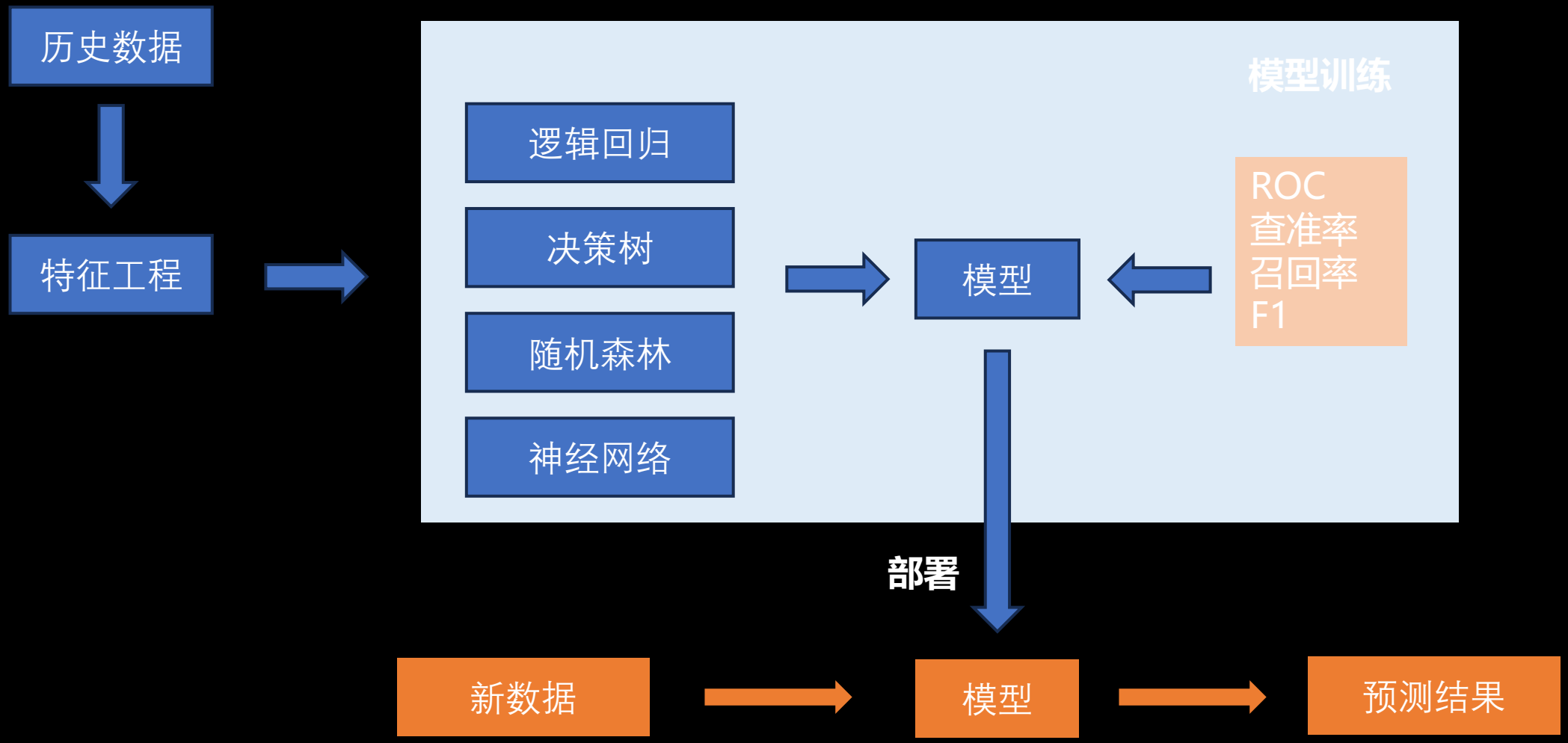
分类模型

根据数据样本上抽取出的特征，判定其属于有限个类别中的哪一个

- **进口冷冻牛肉风险甄别**：1、有问题，2、没问题
- 垃圾邮件识别：1、垃圾邮件 2、正常邮件



模型训练过程


历史数据
特征工程
模型训练
逻辑回归
决策树
随机森林
神经网络
模型
ROC
查准率
召回率
F1
部署
新数据
模型
预测结果

模型评估

青萌数海

分类评价

真实情况	预测结果	
	正例	反例
正例	TP (真正例)	FN (假反例)
反例	FP (假正例)	TN (真反例)

- 查准率(准确率)： $P = \frac{TP}{TP + FP}$
- 查全率(召回率)： $R = \frac{TP}{TP + FN}$

核心评价指标：

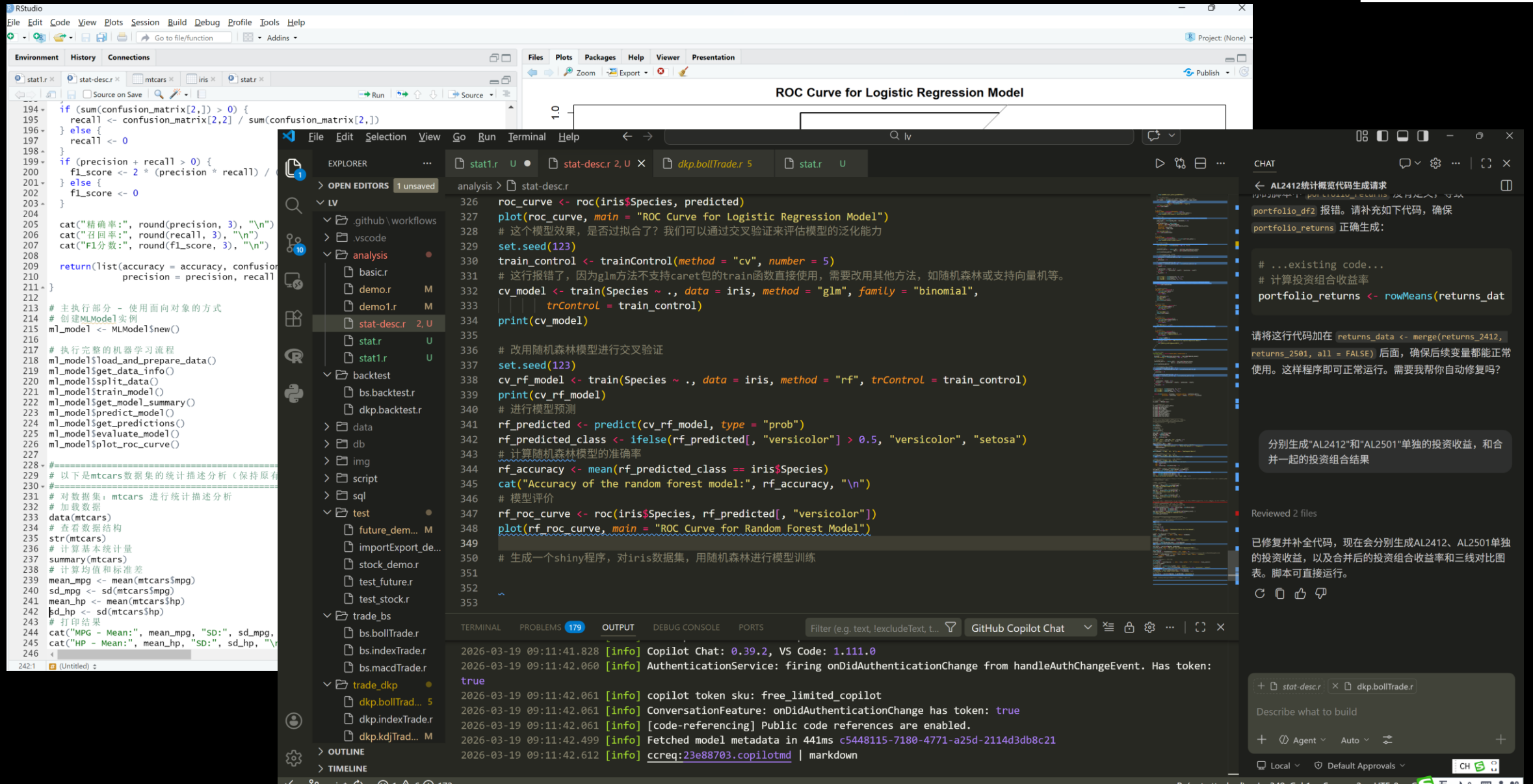
- **业务指标（查获率，立案率）**
- **技术指标（系统功能，系统性能）**

青萌
数海

让AI生成代码：统计概览

1. 独立用AI生成代码
2. 传统方式的分析方法

开发工具： VSCode, RStudio



对数据集介绍

数据集为 **mtcars**，是R语言中自带的数据集。

该数据提取自1974年《摩托趋势》美国杂志，包含32辆汽车（1973-1974年款）的燃油消耗数据以及汽车设计与性能的10个方面。

一个包含32个观测值和11个（数值型）变量的数据框。

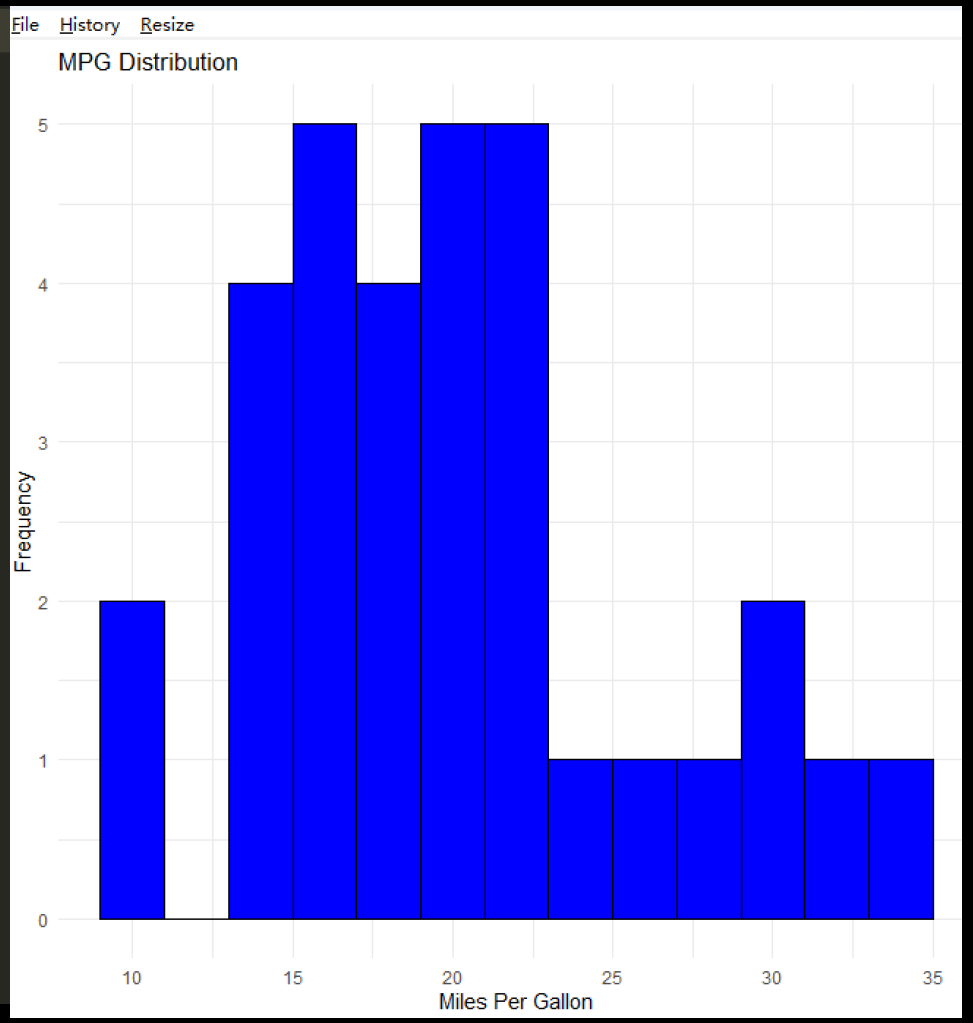
[, 1] mpg 英里/加仑（美国）
[, 2] cyl 气缸数量
[, 3] disp 排量（立方英寸）
[, 4] hp 总功率
[, 5] drat 后轴比
[, 6] wt 重量（1000磅）
[, 7] qsec 1/4英里耗时
[, 8] vs 发动机形状（0 = V型，1 = 直列型）
[, 9] am 变速器类型（0 = 自动，1 = 手动）
[,10] gear 前进挡位数
[,11] carb 化油器数量

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2
Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8
Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

Copliot：对数据集进行统计概览

对数据集mtcars进行统计描述分析

```r
# 统计概览
# 对数据集：mtcars 进行统计描述分析
# 加载数据
data(mtcars)
# 查看数据结构
str(mtcars)
# 计算基本统计量
summary(mtcars)
# 计算均值和标准差
mean_mpg <- mean(mtcars$mpg)
sd_mpg <- sd(mtcars$mpg)
mean_hp <- mean(mtcars$hp)
sd_hp <- sd(mtcars$hp)
# 打印结果
cat("MPG - Mean:", mean_mpg, "SD:", sd_mpg, "\n")
cat("HP - Mean:", mean_hp, "SD:", sd_hp, "\n")
# 画出mpg的分布图
library(ggplot2)
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram(binwidth = 2, fill = "blue", color = "black") +
  labs(title = "MPG Distribution", x = "Miles Per Gallon", y = "Frequency") +
  theme_minimal()
```



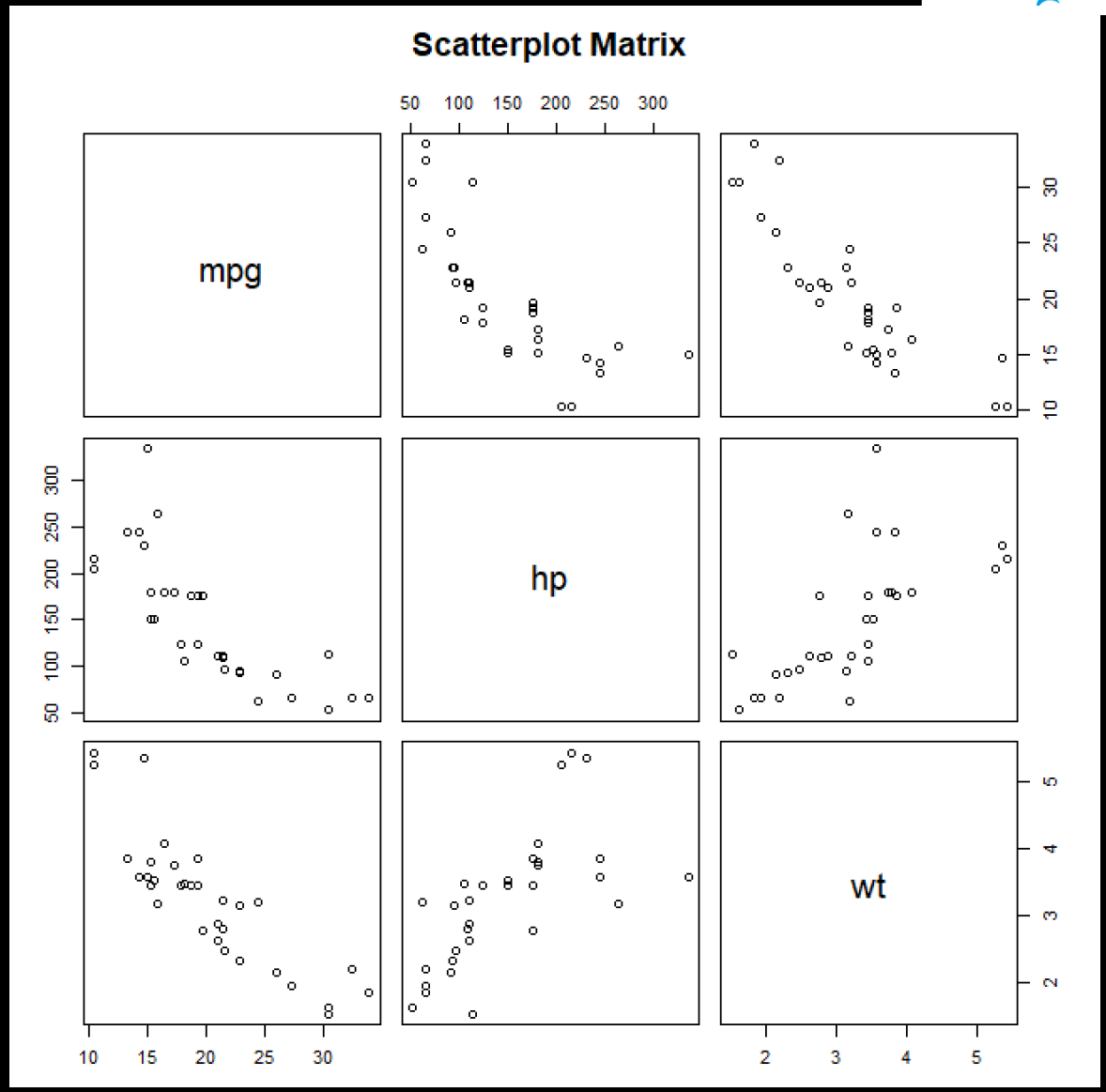
相关性图1

青萌数海

生成不同变量的相关系图

```r
# 生成不同变量的相关系图
pairs(mtcars[, c("mpg", "hp", "wt")], main = "Scatterplot Matrix")
```

[, 1] mpg 英里/加仑（美国）
[, 4] hp 总功率
[, 6] wt 重量（1000磅）



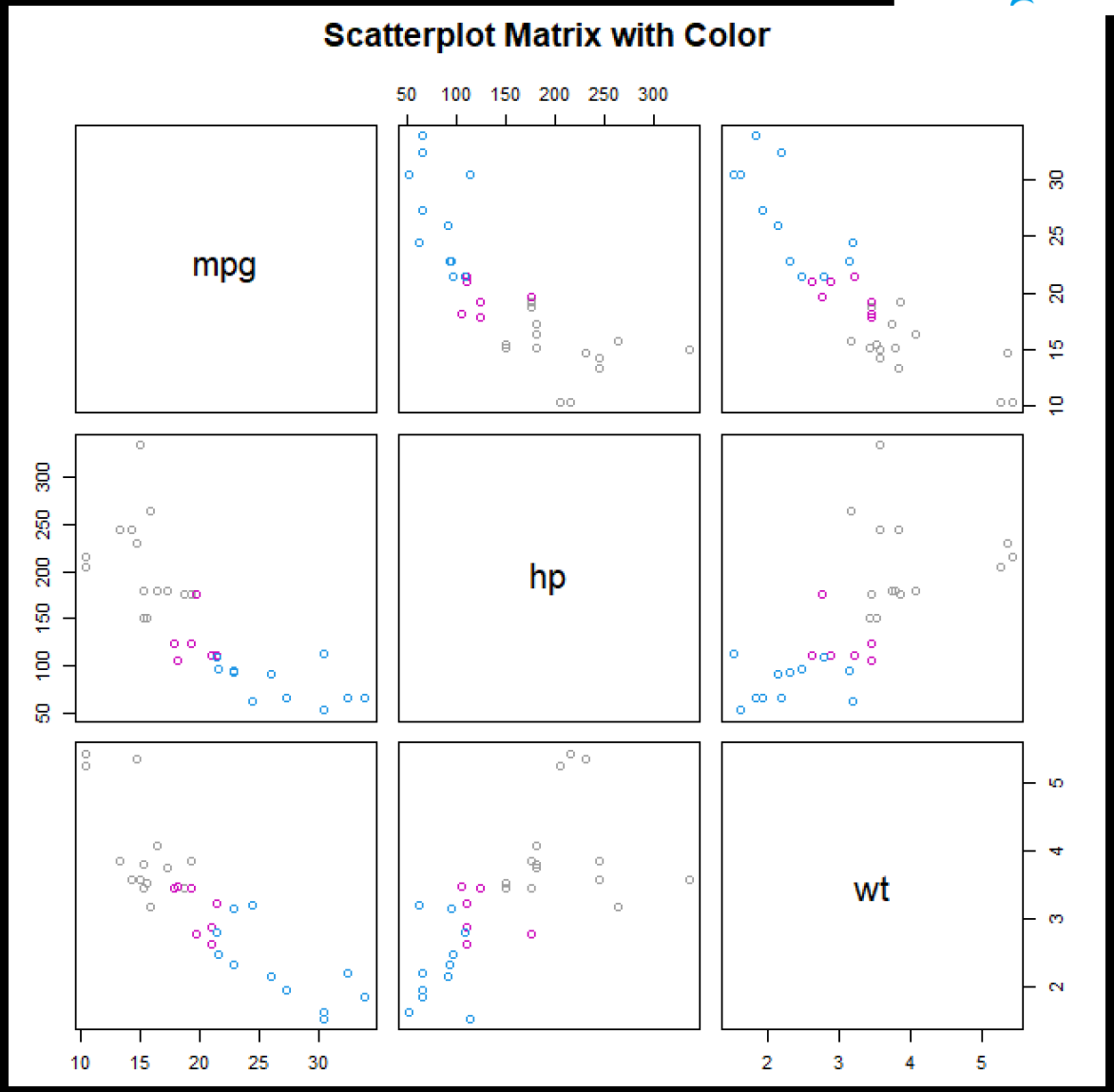
相关性图2

青萌数海

生成不同变量的相关系图，增加颜色

```r
# 生成不同变量的相关系图，增加颜色
pairs(mtcars[, c("mpg", "hp", "wt")],
    main = "Scatterplot Matrix with Color", col = mtcars$cyl)
```

[, 1] mpg 英里/加仑（美国）
[, 4] hp 总功率
[, 6] wt 重量（1000磅）



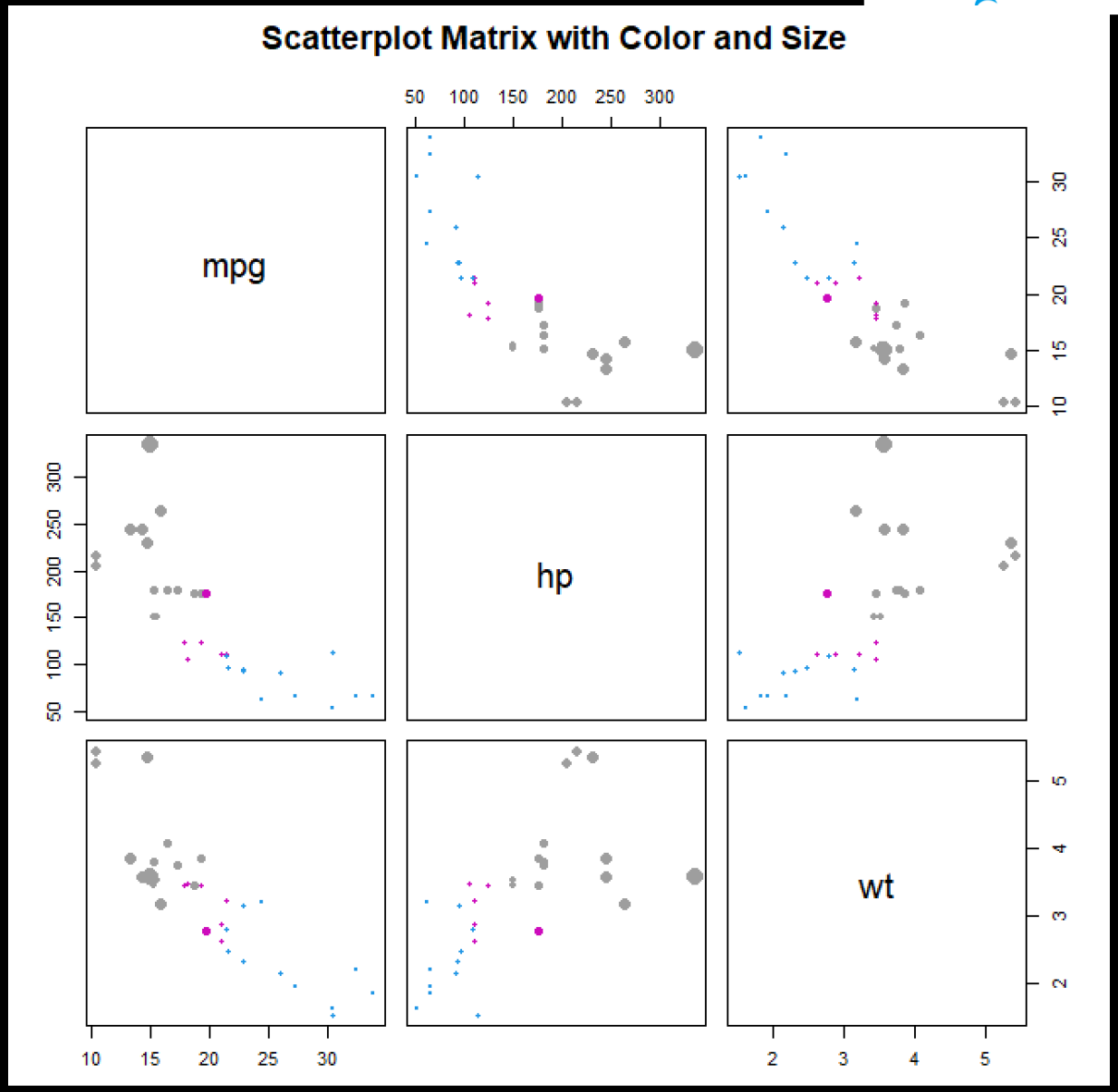
相关性图3

青萌数海

生成不同变量的相关系图，增加颜色和点的大小

```r
# 生成不同变量的相关系图，增加颜色和点的大小
pairs(mtcars[, c("mpg", "hp", "wt")],
    main = "Scatterplot Matrix with Color and Size",
    col = mtcars$cyl,
    pch = 16,
    cex = mtcars$hp / max(mtcars$hp) * 2)
```

[, 1] mpg 英里/加仑（美国）
[, 4] hp 总功率
[, 6] wt 重量（1000磅）



相关性图3

青萌数海

从所有变量中找到相关性最高的变量对

```r
# 从所有变量中找到相关性最高的变量对，并画出它们的散点图
cor_matrix <- cor(mtcars[, sapply(mtcars, is.numeric)])
cor_matrix[lower.tri(cor_matrix, diag = TRUE)] <- NA
cor_pairs <- which(abs(cor_matrix) == max(abs(cor_matrix), na.rm = TRUE), arr.ind = TRUE)
var1 <- colnames(mtcars)[cor_pairs[1, 2]]
var2 <- colnames(mtcars)[cor_pairs[1, 1]]
cat("Most correlated variables:", var1, "and", var2, "\n")
```

[, 2] cyl 气缸数量
[, 3] disp 排量（立方英寸）

```
> cat("Most correlated variables:", var1, "and", var2, "\n")
Most correlated variables: disp and cyl
> cor_matrix
     mpg       cyl       disp         hp       drat         wt        qsec
mpg   NA -0.852162 -0.8475514 -0.7761684  0.6811719 -0.8676594  0.41868403
cyl   NA        NA  0.9020329  0.8324475 -0.6999381  0.7824958 -0.59124207
disp  NA        NA         NA  0.7909486 -0.7102139  0.8879799 -0.43369788
hp    NA        NA         NA         NA -0.4487591  0.6587479 -0.70822339
drat  NA        NA         NA         NA         NA -0.7124406  0.09120476
wt    NA        NA         NA         NA         NA         NA -0.17471588
qsec  NA        NA         NA         NA         NA         NA          NA
vs    NA        NA         NA         NA         NA         NA          NA
am    NA        NA         NA         NA         NA         NA          NA
gear  NA        NA         NA         NA         NA         NA          NA
carb  NA        NA         NA         NA         NA         NA          NA
             vs         am       gear        carb
mpg   0.6640389  0.5998324  0.4802848 -0.55092507
cyl  -0.8108118 -0.5226070 -0.4926866  0.52698829
disp -0.7104159 -0.5912270 -0.5555692  0.39497686
hp   -0.7230967 -0.2432043 -0.1257043  0.74981247
drat  0.4402785  0.7127111  0.6996101 -0.09078980
wt   -0.5549157 -0.6924953 -0.5832870  0.42760594
qsec  0.7445354 -0.2298609 -0.2126822 -0.65624923
vs           NA  0.1683451  0.2060233 -0.56960714
am           NA         NA  0.7940588  0.05753435
gear         NA         NA         NA  0.27407284
carb         NA         NA         NA          NA
>
```

请判断哪量车的性能最好

该数据提取自1974年《摩托趋势》美国杂志，包含32辆汽车（1973-1974年款）的燃油消耗数据以及汽车设计与性能的10个方面。
请判断哪量车的性能最好

```r
# 该数据提取自1974年《摩托趋势》美国杂志，包含32辆汽车（1973-1974年款）的燃油消耗数据以及汽车
# 请判断哪量车的性能最好

# 生成性能对比的方案
# 方案1：以mpg为性能指标，选择mpg最高的车
best_mpg <- mtcars[which.max(mtcars$mpg), ]
cat("Best performance based on MPG:\n")
print(best_mpg)
# 方案2：以hp为性能指标，选择hp最高的车
best_hp <- mtcars[which.max(mtcars$hp), ]
cat("Best performance based on HP:\n")
print(best_hp)
# 方案3：以wt为性能指标，选择wt最低的车
best_wt <- mtcars[which.min(mtcars$wt), ]
cat("Best performance based on Weight:\n")
print(best_wt)
```

[, 1] mpg 英里/加仑（美国）
[, 4] hp 总功率
[, 6] wt 重量（1000磅）

请判断哪量车的性能最好

重新评估性能，应该如何选择指标，如何计算？

```r
#[, 1] mpg 英里/加仑（美国）[, 2] cyl 气缸数量[, 3] disp 排量（立方英寸）[, 4] hp
# 重新评估性能，应该如何选择指标，如何计算？
# 方案4：综合考虑mpg、hp和wt，计算一个性能得分
# 这里我们可以使用一个简单的加权得分方法，假设mpg
# 的权重为0.5，hp的权重为0.3，wt的权重为0.2
mtcars$performance_score <- 0.5 * (mtcars$mpg / max(mtcars$mpg)) +
    0.3 * (mtcars$hp / max(mtcars$hp)) -
    0.2 * (mtcars$wt / max(mtcars$wt))
best_performance <- mtcars[which.max(mtcars$performance_score), ]
cat("Best performance based on combined score:\n")
print(best_performance)
```

```
> print(best_performance)
              mpg cyl disp  hp drat    wt qsec vs am gear carb
Lotus Europa 30.4   4 95.1 113 3.77 1.513 16.9  1  1    5    2
             performance_score
Lotus Europa         0.4937825
```

统计概览分析

青萌数海

```
> # 查看数据静态结构
> ds_screener(mtcarz)
-------------------------------------------------------------------------
|  Column Name  |  Data Type  |  Levels  |  Missing  |  Missing (%)  |
-------------------------------------------------------------------------
|      mpg      |   numeric   |    NA    |     0     |       0       |
|      cyl      |   factor    |   4 6 8  |     0     |       0       |
|     disp      |   numeric   |    NA    |     0     |       0       |
|      hp       |   numeric   |    NA    |     0     |       0       |
|     drat      |   numeric   |    NA    |     0     |       0       |
|      wt       |   numeric   |    NA    |     0     |       0       |
|     qsec      |   numeric   |    NA    |     0     |       0       |
|      vs       |   factor    |    0 1   |     0     |       0       |
|      am       |   factor    |    0 1   |     0     |       0       |
|     gear      |   factor    |   3 4 5  |     0     |       0       |
|     carb      |   factor    |1 2 3 4 6 8|     0     |       0       |
-------------------------------------------------------------------------

 Overall Missing Values           0
 Percentage of Missing Values     0 %
 Rows with Missing Values         0
 Columns With Missing Values      0
```

- 使用 descriptr 包
- 用 mtcarz 替代 mtcars：对cyl, vs,am,gear,carb 变量转factor类型

[, 1] mpg 英里/加仑（美国）
[, 2] cyl 气缸数量
[, 3] disp 排量（立方英寸）
[, 4] hp 总功率
[, 5] drat 后轴比
[, 6] wt 重量（1000磅）
[, 7] qsec 1/4英里耗时
[, 8] vs 发动机形状（0 = V型，1 = 直列型）
[, 9] am 变速器类型（0 = 自动，1 = 手动）
[,10] gear 前进挡位数
[,11] carb 化油器数量

单独变量统计概览：mpg 英里/加仑

```
> ds_summary_stats(mtcarz,mpg)
---------------------------------------------------- Variable: mpg ---------------

                      Univariate Analysis

 N                          32.00      Variance                  36.32
 Missing                     0.00      Std Deviation              6.03
 Mean                       20.09      Range                     23.50
 Median                     19.20      Interquartile Range        7.38
 Mode                       10.40      Uncorrected SS         14042.31
 Trimmed Mean               19.95      Corrected SS            1126.05
 Skewness                    0.67      Coeff Variation           30.00
 Kurtosis                   -0.02      Std Error Mean             1.07

                         Quantiles

              Quantile                                 Value

              Max                                      33.90
              99%                                      33.44
              95%                                      31.30
              90%                                      30.09
              Q3                                       22.80
              Median                                   19.20
              Q1                                       15.43
              10%                                      14.34
              5%                                       12.00
              1%                                       10.40
              Min                                      10.40

                      Extreme Values

                Low                                     High

  Obs                      Value       Obs                      Value
  15                       10.4        20                       33.9
  16                       10.4        18                       32.4
  24                       13.3        19                       30.4
   7                       14.3        28                       30.4
  17                       14.7        26                       27.3
```

输出分成了3个部分：Univariate Analysis（单变量分析），Quantiles（分位数），Extreme Values（极值）。

- Univariate Analysis（单变量分析），包括N（个数），Missing（缺失值），Mean（均值），Median（中位数），Mode（众数），Trimmed Mean（修正均值），Skewness（偏度），Kurtosis（峰度），Variance（方差），Std Deviation（标准差），Range(范围，最大-最小)，Interquartile Range(四分位数范围)，Uncorrected SS（未修正平方和），Corrected SS（修正平方和），Coeff Variation（变异系数，标准差/均值），Std Error Mean（标准误差均值）
- Quantiles（分位数），从最小值到最小值，按顺序排列，对应的数值。
- Extreme Values（极值），包括最小值前5个，最大值前5个。

多变量统计概览：


```
> # 统计概览：多变量统计
> ds_tidy_stats(mtcarz, mpg, disp, hp)
# A tibble: 3 × 16
  vars    min   max  mean t_mean median  mode range variance  stdev  skew kurtosis coeff_var    q1    q3 iqrange
  <chr> <dbl> <dbl> <dbl>  <dbl>  <dbl> <dbl> <dbl>    <dbl>  <dbl> <dbl>    <dbl>     <dbl> <dbl> <dbl>   <dbl>
1 disp   71.1 472   231.   228    196.  276.  401.   15361.  124.   0.420  -1.07        53.7 121.  326    205.
2 hp     52   335   147.   144.   123   110   283     4701.   68.6  0.799   0.275       46.7  96.5 180     83.5
3 mpg    10.4  33.9  20.1   20.0   19.2  10.4  23.5     36.3   6.03 0.672  -0.0220      30.0  15.4  22.8    7.38
>
```

分组统计

分组统计：把数据集中变量进行分组，再分别计算统计特征。

```
> # 分组统计：把数据集中变量进行分组，再分别计算统计特征。
> ds_group_summary(mtcarz,cyl,mpg)
                                    mpg by cyl
-------------------------------------------------------------------------------
|     Statistic/Levels|                  4|                  6|                  8|
-------------------------------------------------------------------------------
|                  Obs|                 11|                  7|                 14|
|              Minimum|               21.4|               17.8|               10.4|
|              Maximum|               33.9|               21.4|               19.2|
|                 Mean|              26.66|              19.74|               15.1|
|               Median|                 26|               19.7|               15.2|
|                 Mode|               22.8|                 21|               10.4|
|       Std. Deviation|               4.51|               1.45|               2.56|
|             Variance|              20.34|               2.11|               6.55|
|             Skewness|               0.35|              -0.26|              -0.46|
|             Kurtosis|              -1.43|              -1.83|               0.33|
|       Uncorrected SS|            8023.83|            2741.14|            3277.34|
|         Corrected SS|             203.39|              12.68|               85.2|
|      Coeff Variation|              16.91|               7.36|              16.95|
|      Std. Error Mean|               1.36|               0.55|               0.68|
|                Range|               12.5|                3.6|                8.8|
|  Interquartile Range|                7.6|               2.35|               1.85|
-------------------------------------------------------------------------------
```

[, 1] mpg 英里/加仑（美国）
[, 2] cyl 气缸数量

数据测量1

数据集变化分析:
范围，四分位范围，方差，标准差，
变异系数，标准误差

```
> # 数据集变化分析：范围，四分位范围，方差，标准差，变异系数，标准误差
> ds_measures_variation(mtcarz)
# A tibble: 6 × 7
  var    range    iqr  variance      sd coeff_var std_error
  <chr>  <dbl>  <dbl>     <dbl>   <dbl>     <dbl>     <dbl>
1 disp   401.   205.    15361.    124.        53.7   21.9
2 drat     2.17   0.84      0.286   0.535     14.9    0.0945
3 hp     283     83.5    4701.     68.6       46.7   12.1
4 mpg     23.5    7.38    36.3     6.03      30.0    1.07
5 qsec     8.4    2.01     3.19    1.79      10.0    0.316
6 wt       3.91   1.03     0.957   0.978     30.4    0.173
```

#数据集数值分析：均值，修
正均值，中位数，众数

```
> #数据集数值分析：均值，修正均值，中位数，众数
> ds_measures_location(mtcarz)
# A tibble: 6 × 5
  var     mean trim_mean median   mode
  <chr>  <dbl>     <dbl>  <dbl>  <dbl>
1 disp   231.      228    196.   276.
2 drat     3.60      3.58   3.70   3.07
3 hp     147.      144.   123    110
4 mpg     20.1      20.0   19.2   10.4
5 qsec    17.8      17.8   17.7   17.0
6 wt       3.22      3.20   3.32   3.44
```

数据测量2

数据集分位数分析：从最小值到最大值排序

```
> # 数据集分位数分析：从最小值到最大值排序
> ds_percentiles(mtcarz)
# A tibble: 6 × 12
  var     min  per1  per5 per10     q1 median     q3  per95  per90  per99    max
  <chr> <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
1 disp   71.1  72.5  77.4  80.6  121.   196.   326    449    396    468.   472
2 drat    2.76  2.76  2.85  3.01   3.08   3.70   3.92   4.31   4.21   4.78   4.93
3 hp     52    55.1  63.6  66     96.5  123    180    254.   244.   313.   335
4 mpg    10.4  10.4  12.0  14.3   15.4   19.2   22.8   31.3   30.1   33.4   33.9
5 qsec   14.5  14.5  15.0  15.5   16.9   17.7   18.9   20.1   20.0   22.1   22.9
6 wt      1.51  1.54  1.74  1.96   2.58   3.32   3.61   5.29   4.05   5.40   5.42
```

极值分析，最大值，最小值

```
> # 极值分析
> ds_extreme_obs(mtcarz,mpg)
# A tibble: 10 × 3
   type  value index
   <chr> <dbl> <int>
 1 high   33.9    20
 2 high   32.4    18
 3 high   30.4    19
 4 high   30.4    28
 5 high   27.3    26
 6 low    10.4    15
 7 low    10.4    16
 8 low    13.3    24
 9 low    14.3     7
10 low    14.7    17
```

类别变量频率表

查看数据集中类别变量的频率表。

```
> # 查看数据集中类别变量的双向表。
> ds_cross_table(mtcarz, cyl, gear)
   Cell Contents
 |---------------|
 |     Frequency |
 |       Percent |
 |       Row Pct |
 |       Col Pct |
 |---------------|

 Total Observations:  32

-------------------------------------------------------------------------
|               |                          gear                         |
-------------------------------------------------------------------------
|           cyl |             3 |             4 |             5 |   Row Total |
-------------------------------------------------------------------------
|             4 |             1 |             8 |             2 |          11 |
|               |         0.031 |         0.25  |         0.062 |             |
|               |          0.09 |          0.73 |          0.18 |        0.34 |
|               |          0.07 |          0.67 |           0.4 |             |
-------------------------------------------------------------------------
|             6 |             2 |             4 |             1 |           7 |
|               |         0.062 |         0.125 |         0.031 |             |
|               |          0.29 |          0.57 |          0.14 |        0.22 |
|               |          0.13 |          0.33 |           0.2 |             |
-------------------------------------------------------------------------
|             8 |            12 |             0 |             2 |          14 |
|               |         0.375 |         0     |         0.062 |             |
|               |          0.86 |          0    |          0.14 |        0.44 |
|               |           0.8 |          0    |           0.4 |             |
-------------------------------------------------------------------------
| Column Total |            15 |            12 |             5 |          32 |
|               |         0.468 |         0.375 |         0.155 |             |
-------------------------------------------------------------------------
```

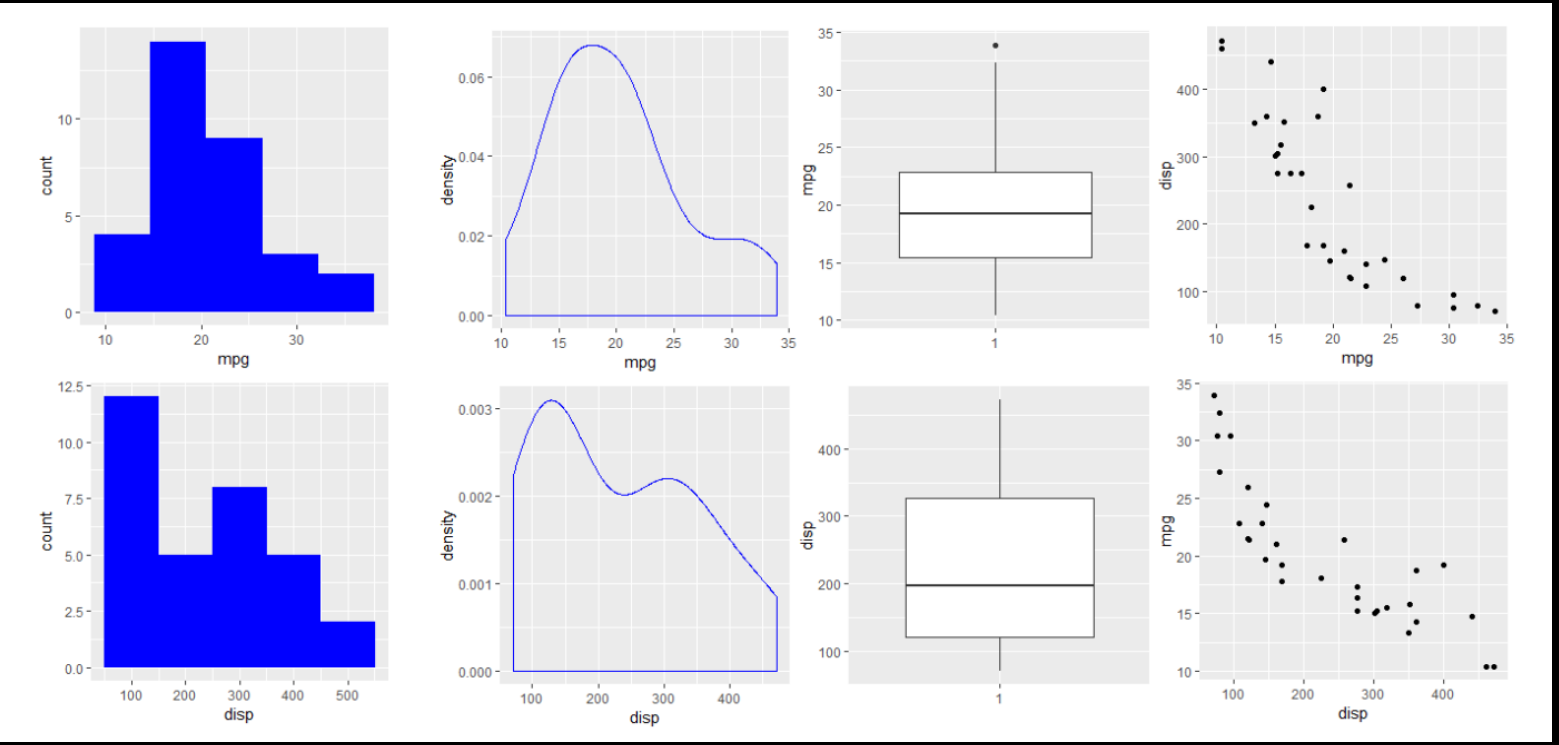
类别变量频率表

查看数据集中类别变量的双向表。

```
> # 类别变量的双向表
> ds_twoway_table(mtcarz, cyl, gear)
Joining with `by = join_by(cyl, gear, count)`
# A tibble: 8 × 6
  cyl   gear  count percent row_percent col_percent
  <fct> <fct> <int>   <dbl>       <dbl>       <dbl>
1 4     3         1  0.0312      0.0909      0.0667
2 4     4         8  0.25        0.727       0.667
3 4     5         2  0.0625      0.182       0.4
4 6     3         2  0.0625      0.286       0.133
5 6     4         4  0.125       0.571       0.333
6 6     5         1  0.0312      0.143       0.2
7 8     3        12  0.375       0.857       0.8
8 8     5         2  0.0625      0.143       0.4
```

可视化连续型数据

```
> # 可视化连续型数据
> ds_plot_histogram(mtcars, mpg, disp)
> ds_plot_density(mtcars, mpg, disp)
> ds_plot_box_single(mtcars, mpg, disp)
> ds_plot_scatter(mtcars, mpg, disp)
```

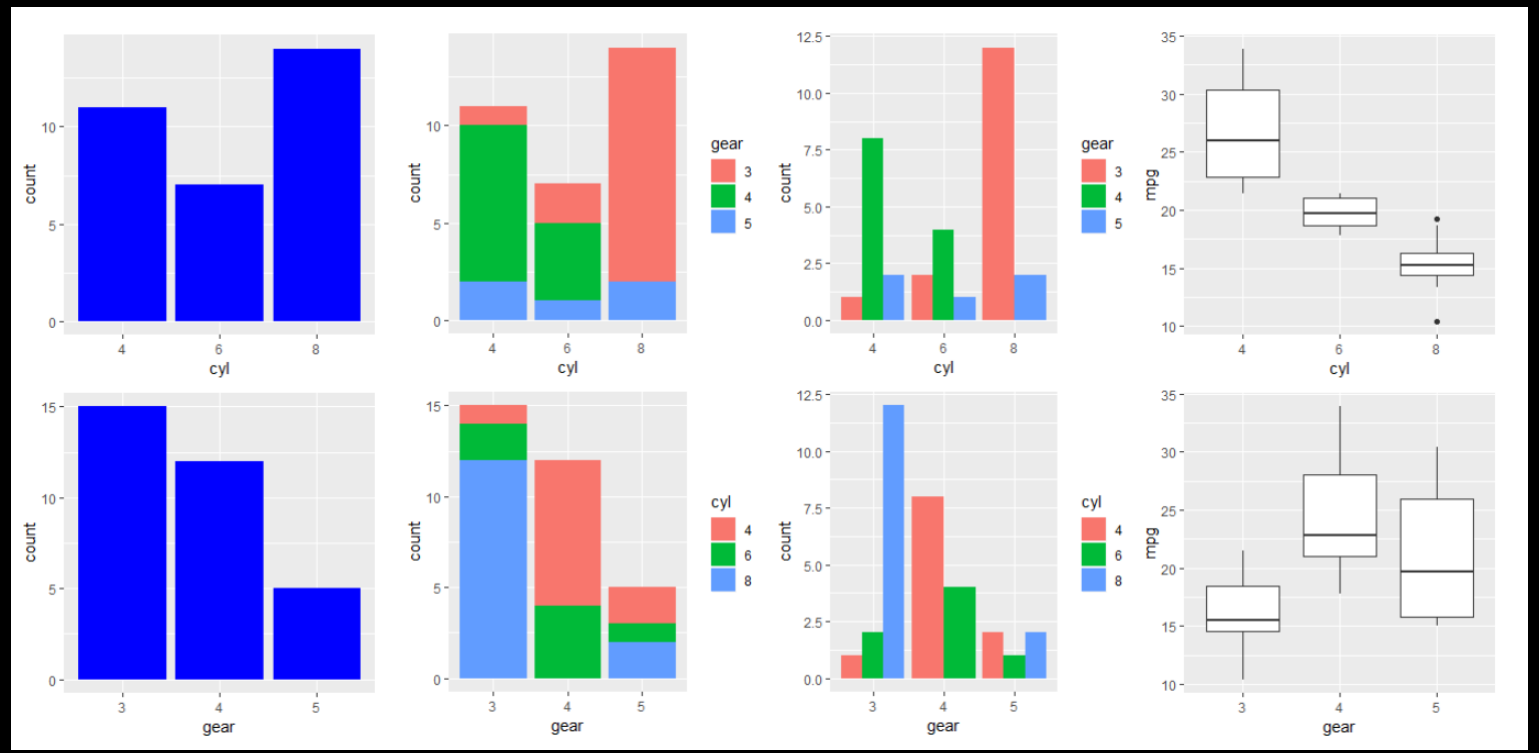


[, 1] mpg 英里/加仑（美国）
[, 3] disp 排量（立方英寸）

可视化类别型数据

青萌数海

```
> #可视化类别型数据
> ds_plot_bar(mtcarz,cyl, gear)
> ds_plot_bar_stacked(mtcarz, cyl, gear)
> ds_plot_bar_grouped(mtcarz, cyl, gear)
> ds_plot_box_group(mtcarz, cyl, gear, mpg)
```



[, 1] mpg 英里/加仑（美国）
[, 2] cyl 气缸数量
[,10] gear 前进挡位数

青萌
数海

让AI生成代码：机器学习

1. 独立用AI生成代码
2. 传统方式的分析方法
3. 让AI理解代码产生的逻辑，近一步提升代码质量

对数据集介绍

数据集为 **iris**，是R语言中自带的数据集。

这是著名的（费希尔或安德森）鸢尾花数据集，记录了3种鸢尾花各50朵样本的花萼长度、花萼宽度、花瓣长度和花瓣宽度的测量数据（单位：厘米）。三种鸢尾花分别为：刚毛鸢尾、变色鸢尾和维吉尼亚鸢尾。

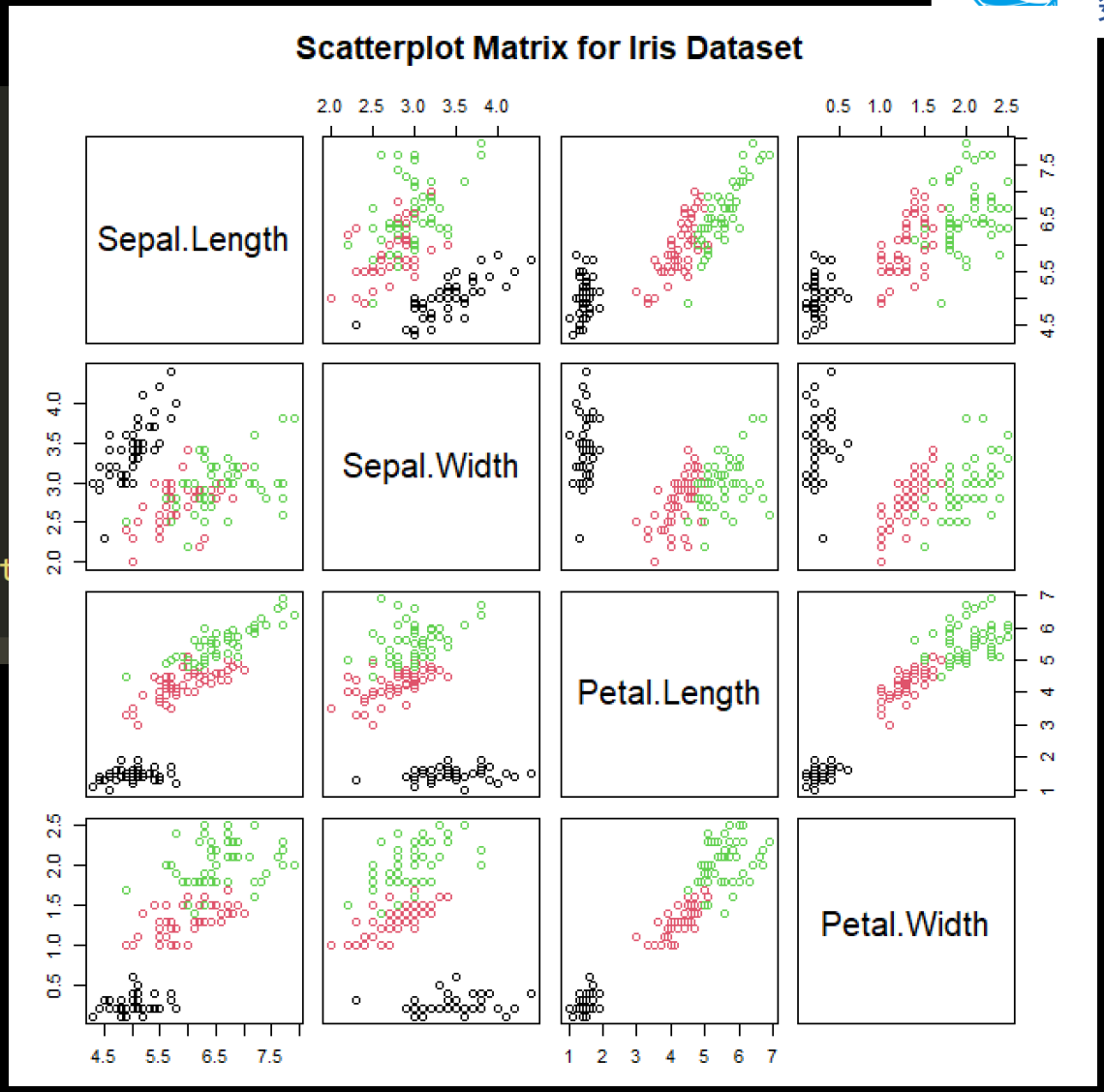
英文	中文
Sepal.Length	花萼长度
Sepal.Width	花萼宽度
Petal.Length	花瓣长度
Petal.Width	花瓣宽度
Species	品种

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
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
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa
15	5.8	4.0	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa
20	5.1	3.8	1.5	0.3	setosa
21	5.4	3.4	1.7	0.2	setosa
22	5.1	3.7	1.5	0.4	setosa
23	4.6	3.6	1.0	0.2	setosa
24	5.1	3.3	1.7	0.5	setosa
25	4.8	3.4	1.9	0.2	setosa
26	5.0	3.0	1.6	0.2	setosa
27	5.0	3.4	1.6	0.4	setosa
28	5.2	3.5	1.5	0.2	setosa
29	5.2	3.4	1.4	0.2	setosa

对数据集介绍

青萌
数海

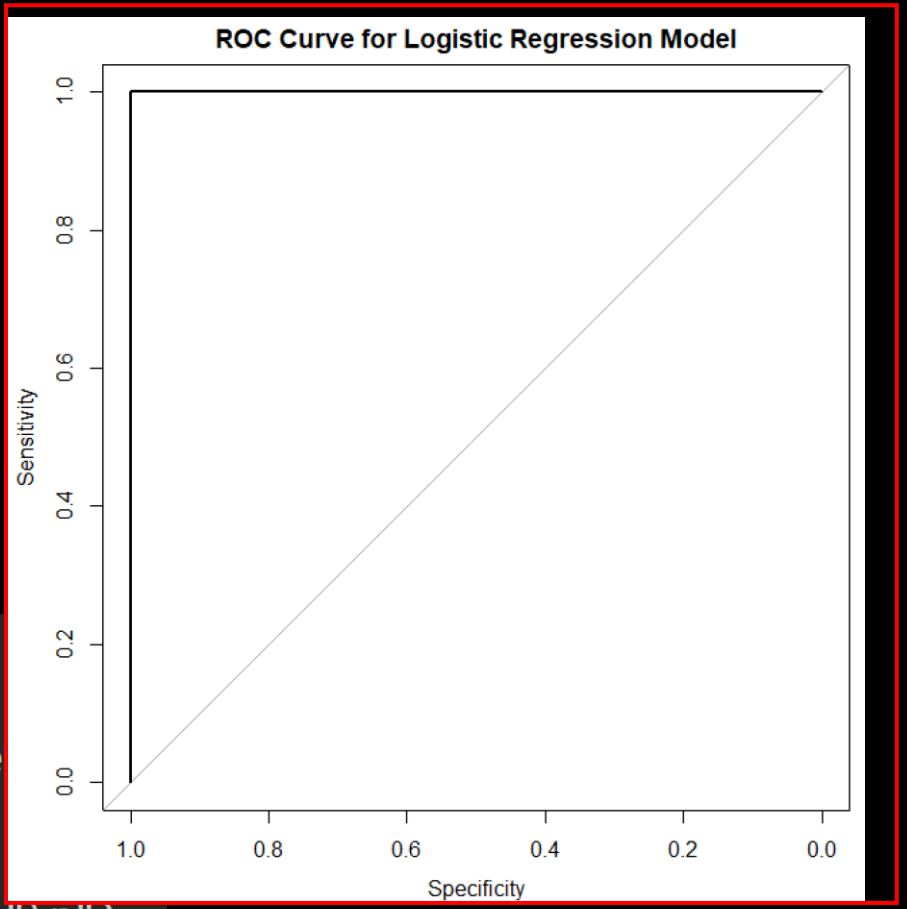
```r
#=============================================
# 新的例子
#=============================================
# 取数据集iris，进行统计描述分析
data(iris)
# 统计概览
summary(iris)
# 相关性分析
cor(iris[, 1:4])
# 画出散点图矩阵
pairs(iris[, 1:4], main = "Scatterplot Matrix for Iris Dat
    col = iris$Species)
```



构建一个逻辑回归模型


```r
# 构建一个逻辑回归模型，预测花的种类
model <- glm(Species ~ ., data = iris, family = binomial)
summary(model)
# 对iris数据集进行预测
predicted <- predict(model, type = "response")
predicted_class <- ifelse(predicted > 0.5, "versicolor", "setosa")
# 计算预测的准确率
accuracy <- mean(predicted_class == iris$Species)
cat("Accuracy of the logistic regression model:", accuracy, "\n")
# 模型评价
library(pROC)
roc_curve <- roc(iris$Species, predicted)
plot(roc_curve, main = "ROC Curve for Logistic Regression Model")
```

```
> # 对iris数据集进行预测
> predicted <- predict(model, type = "response")
> predicted_class <- ifelse(predicted > 0.5, "versicolor", "se
> # 计算预测的准确率
> accuracy <- mean(predicted_class == iris$Species)
> cat("Accuracy of the logistic regression model:", accuracy, "\n")
Accuracy of the logistic regression model: 0.6666667
```



对数据集介绍

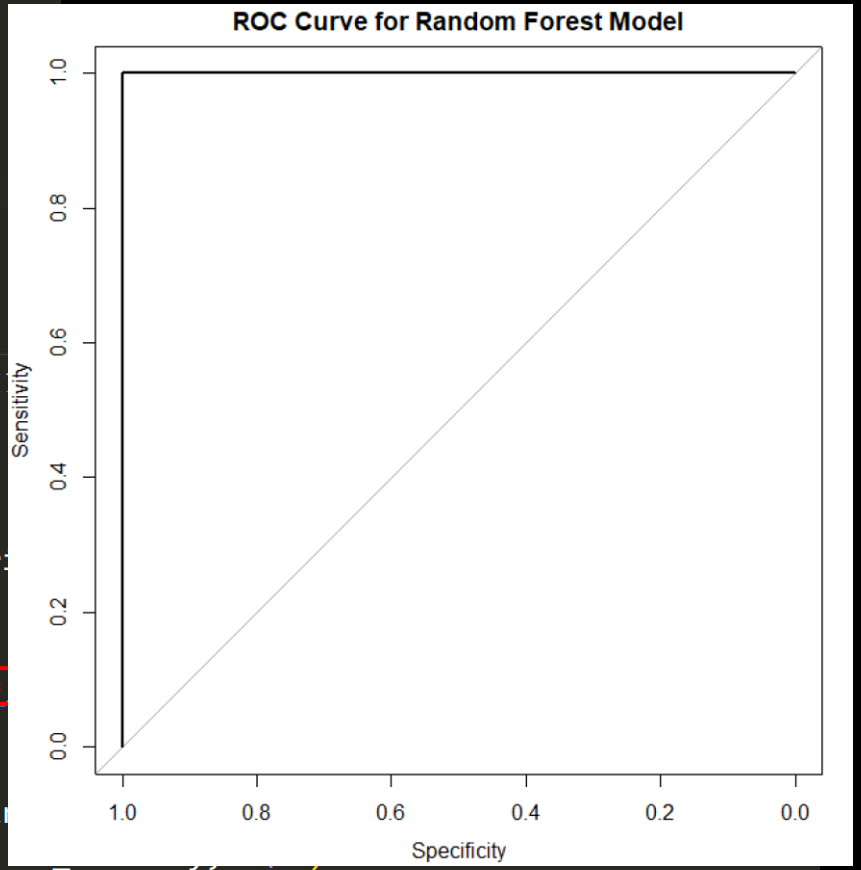
青萌数海

```r
# 模型评价
library(pROC)
roc_curve <- roc(iris$Species, predicted)
plot(roc_curve, main = "ROC Curve for Logistic Regression Model")
# 这个模型效果，是否过拟合了？我们可以通过交叉验证来评估模型的泛化能力
library(caret)
set.seed(123)
train_control <- trainControl(method = "cv", number = 5)
# 这行报错了，因为glm方法不支持caret包的train函数直接使用，需要改用其他方法，如随机森林或支持向量机等。
cv_model <- train(Species ~ ., data = iris, method = "glm", family = "binomial",
                  trControl = train_control)
print(cv_model)

# 改用随机森林模型进行交叉验证
```

TERMINAL PROBLEMS 51 OUTPUT DEBUG CONSOLE PORTS R Interactive

```
 Min.   : NA   Min.   : NA
 1st Qu.: NA   1st Qu.: NA
 Median : NA   Median : NA
 Mean   :NaN   Mean   :NaN
 3rd Qu.: NA   3rd Qu.: NA
 Max.   : NA   Max.   : NA
 NA's   :1     NA's   :1
Error: Stopping
In addition: Warning messages:
1: model fit failed for Fold1: parameter=none Err
ls,  :
  glm models can only use 2-class outcomes

2: model fit failed for Fold2: parameter=none Err
```

```r
# 改用随机森林模型进行交叉验证
library(randomForest)
set.seed(123)
cv_rf_model <- train(Species ~ ., data = iri
print(cv_rf_model)
# 进行模型预测
rf_predicted <- predict(cv_rf_model, type =
rf_predicted_class <- ifelse(rf_predicted[,
# 计算随机森林模型的准确率
rf_accuracy <- mean(rf_predicted_class == ir
cat("Accuracy of the random forest model:",
# 模型评价
rf_roc_curve <- roc(iris$Species, rf_predicted[, "versicolor"])
plot(rf_roc_curve, main = "ROC Curve for Random Forest Model")
```



Deepseek 的方案

青萌数海

Chat: 使用R语言逻辑回归，进行模型训练和预测的代码

r

复制 下载

```r
# 加载必要的包
library(dplyr)
library(ggplot2)

# 准备数据 - 将iris数据集改为二分类问题（只取两个物种）
data(iris)
iris_binary <- iris %>%
  filter(Species != "virginica") %>%  # 只保留setosa和versicolor
  mutate(Species_binary = ifelse(Species == "setosa", 0, 1))  # 将物种转换为0/1

# 查看数据结构
str(iris_binary)
head(iris_binary)

# 划分训练集和测试集
set.seed(123)  # 设置随机种子以确保结果可重复
train_index <- sample(1:nrow(iris_binary), size = 0.7 * nrow(iris_binary))
train_data <- iris_binary[train_index, ]
test_data <- iris_binary[-train_index, ]

# 训练逻辑回归模型
# 使用Species_binary作为目标变量，花萼和花瓣测量值作为特征
logit_model <- glm(Species_binary ~ Sepal.Length + Sepal.Width +
                   Petal.Length + Petal.Width,
                   data = train_data,
                   family = binomial(link = "logit"))

# 在测试集上进行预测
# 预测概率
test_data$pred_prob <- predict(logit_model, newdata = test_data, type = "response")

# 根据概率阈值(0.5)进行分类
test_data$pred_class <- ifelse(test_data$pred_prob > 0.5, 1, 0)

# 查看预测结果
head(test_data[, c("Species", "Species_binary", "pred_prob", "pred_class")])

# 计算模型准确率
accuracy <- mean(test_data$pred_class == test_data$Species_binary)
cat("模型准确率:", round(accuracy * 100, 2), "%\n")

# 创建混淆矩阵
confusion_matrix <- table(实际值 = test_data$Species_binary,
                          预测值 = test_data$pred_class)
print(confusion_matrix)

# 计算精确率、召回率和F1分数
precision <- confusion_matrix[2,2] / sum(confusion_matrix[,2])
recall <- confusion_matrix[2,2] / sum(confusion_matrix[2,])
f1_score <- 2 * (precision * recall) / (precision + recall)

cat("精确率:", round(precision, 3), "\n")
cat("召回率:", round(recall, 3), "\n")
cat("F1分数:", round(f1_score, 3), "\n")
```

代码重构

Chat:代码写法太散了，封装成功能函数，包括数据加载，数据处理，模型训练，模型预测，模型评估，几个函数。

我已经将代码重构为函数形式，将散乱的代码封装成几个功能函数：

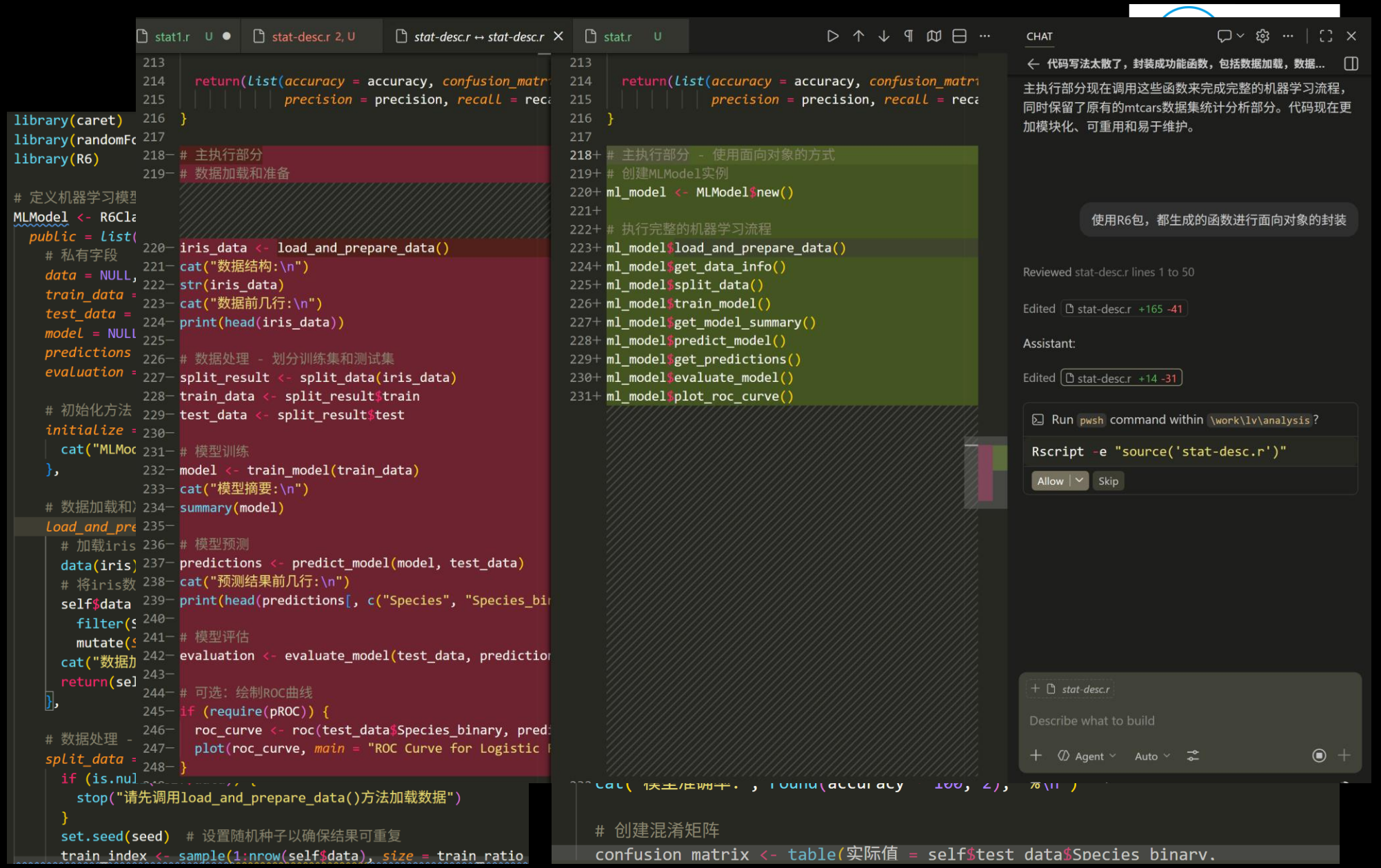
1. `load_and_prepare_data()` - 数据加载和准备：加载iris数据集并转换为二分类问题
2. `split_data()` - 数据处理：划分训练集和测试集
3. `train_model()` - 模型训练：训练逻辑回归模型
4. `predict_model()` - 模型预测：对测试数据进行预测
5. `evaluate_model()` - 模型评估：计算准确率、混淆矩阵、精确率、召回率和F1分数

主执行部分现在调用这些函数来完成完整的机器学习流程，同时保留了原有的mtcars数据集统计分析部分。代码现在更加模块化、可重用和易于维护。

```r
# 函数：数据加载和准备
load_and_prepare_data <- function() {
  # 加载iris数据集
  data(iris)
  # 将iris数据集改为二分类问题（只取两个物种）
  iris_binary <- iris %>%
    filter(Species != "virginica") %>%  # 只保留setosa和versicolor
    mutate(Species_binary = ifelse(Species == "setosa", 0, 1))  # 将物种转换
  return(iris_binary)
}

# 函数：数据处理 - 划分训练集和测试集
split_data <- function(data, train_ratio = 0.8, seed = 123) {
  set.seed(seed)  # 设置随机种子以确保结果可重复
  train_index <- sample(1:nrow(data), size = train_ratio * nrow(data))
  train_data <- data[train_index, ]
  test_data <- data[-train_index, ]
  return(list(train = train_data, test = test_data))
}

# 函数：模型训练
train_model <- function(train_data) {
  # 训练逻辑回归模型
  # 使用Species_binary作为目标变量，花萼和花瓣测量值作为特征
  logit_model <- glm(Species_binary ~ Sepal.Length + Sepal.Width +
                       Petal.Length + Petal.Width,
                     data = train_data,
                     family = binomial(link = "logit"))
  return(logit_model)
}

# 函数：模型预测
predict_model <- function(model, test_data, threshold = 0.5) {
  # 预测概率
  pred_prob <- predict(model, newdata = test_data, type = "response")
  # 根据概率阈值进行分类
  pred_class <- ifelse(pred_prob > threshold, 1, 0)
  # 返回预测结果
  predictions <- data.frame(
```

代码重构

Chat:使用R6包，对生成的函数进行**面向对象**的封装。



让AI生成代码：可视化工具

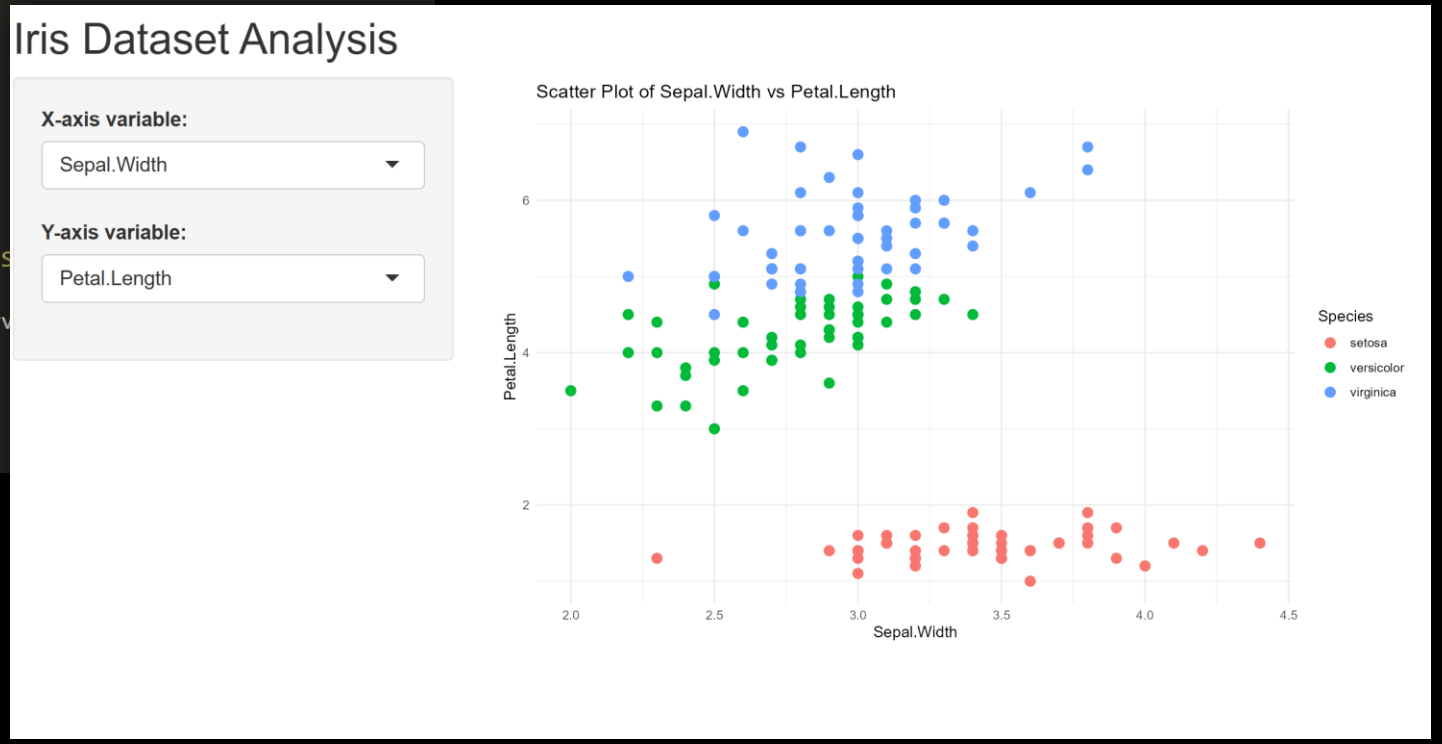
青萌
数海

可视化工具

青萌数海

Chat:生成一个shiny程序，对iris数据集进行交互式分析。

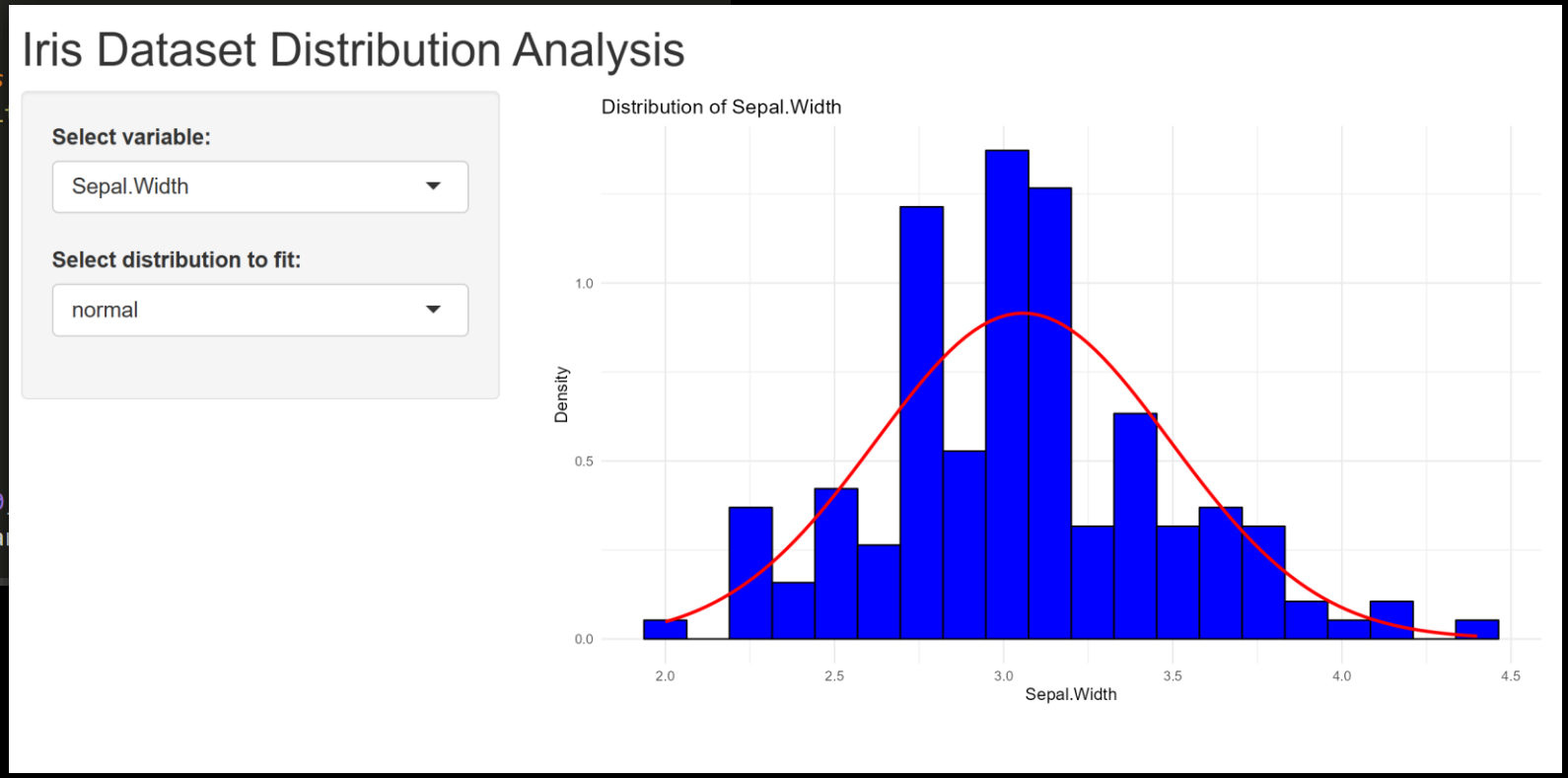
```r
# 生成一个shiny程序，对iris数据集进行交互式分析
library(shiny)
library(ggplot2)
ui <- fluidPage(
  titlePanel("Iris Dataset Analysis"),
  sidebarLayout(
    sidebarPanel(
      selectInput("xvar", "X-axis variable:", choices = names(iris)[1:4]),
      selectInput("yvar", "Y-axis variable:", choices = names(iris)[1:4], selected = names(iris)[2])
    ),
    mainPanel(
      plotOutput("scatterPlot")
    )
  )
)
server <- function(input, output) {
  output$scatterPlot <- renderPlot({
    ggplot(iris, aes_string(x = input$xvar, y = input$yvar, color = "S
      geom_point(size = 3) +
      labs(title = paste("Scatter Plot of", input$xvar, "vs", input$yv
      theme_minimal()
  })
}
shinyApp(ui = ui, server = server)
```



可视化工具

生成一个shiny程序，可以对iris数据集进行交互式分析，生成不同变量的分布图并可以对标准的多种分布，进行拟合和比较。

```r
# 生成一个shiny程序，可以对iris数据集进行交互式分析，生成不同变量的分布图并可以对标准的多种分布，进行拟合和比较
library(shiny)
library(ggplot2)
library(fitdistrplus)
ui <- fluidPage(
  titlePanel("Iris Dataset Distribution Analysis"),
  sidebarLayout(
    sidebarPanel(
      selectInput("var", "Select variable:", choices
      selectInput("dist", "Select distribution to fi
    ),
    mainPanel(
      plotOutput("distPlot")
    )
  )
)
server <- function(input, output) {
  output$distPlot <- renderPlot({
    var_data <- iris[[input$var]]
    ggplot(iris, aes_string(x = input$var)) +
      geom_histogram(aes(y = ..density..), bins = 20
      labs(title = paste("Distribution of", input$va
      theme_minimal() +
```

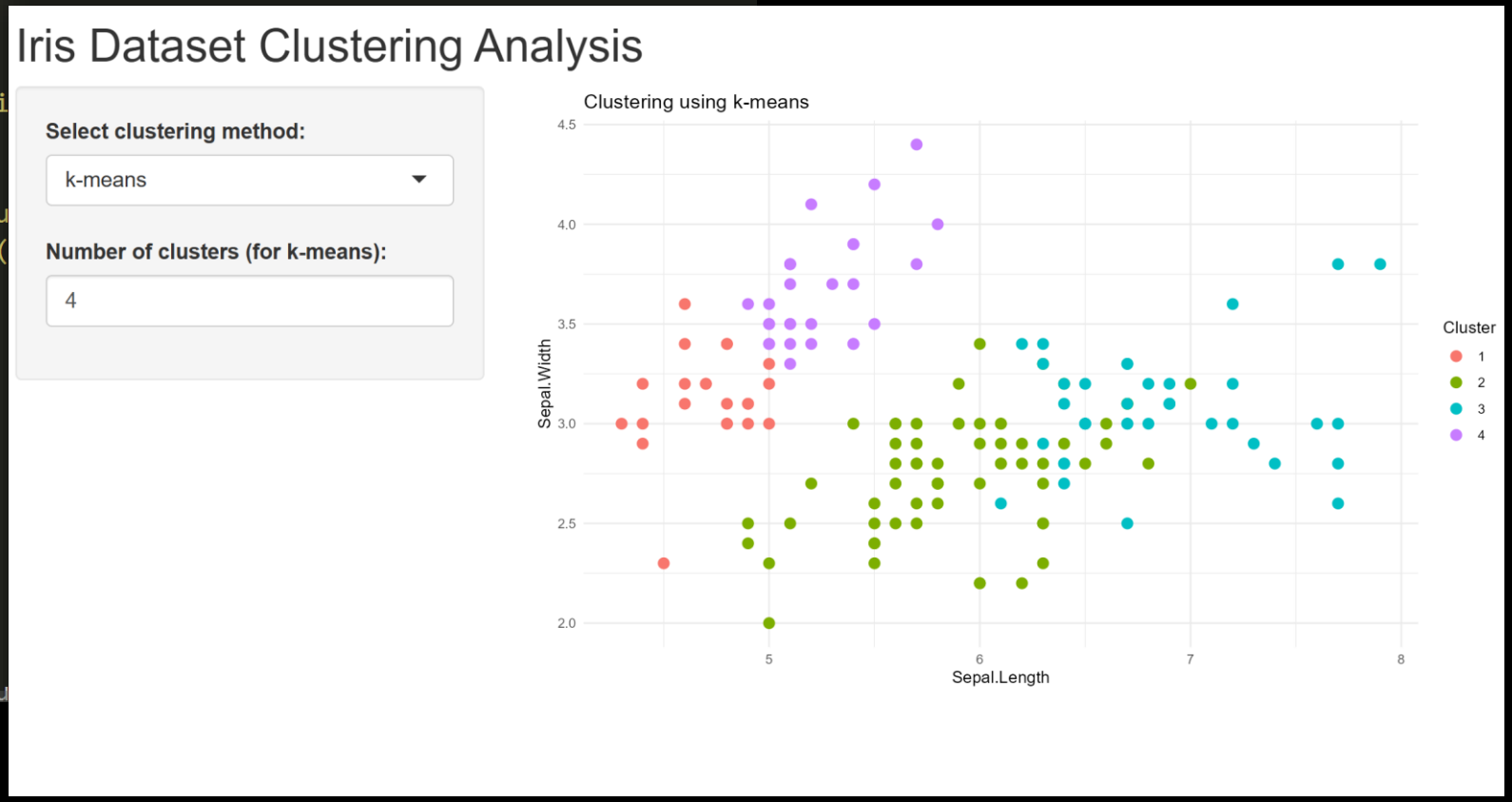


可视化工具

青萌数海

Chat:# 生成一个shiny程序，可以对iris数据集进行交互式分析，提供不同的聚类方法（如k-means、dbscan、层次聚类等），并可视化聚类结果

```r
# 生成一个shiny程序，可以对iris数据集进行交互式分析，提供不同的聚类方法（如k-means、dbscan、层次聚类等
library(shiny)
library(ggplot2)
library(cluster)
library(dbscan)
ui <- fluidPage(
  titlePanel("Iris Dataset Clustering Analysi
  sidebarLayout(
    sidebarPanel(
      selectInput("clust_method", "Select clu
      numericInput("k", "Number of clusters (
    ),
    mainPanel(
      plotOutput("clusterPlot")
    )
  )
)
server <- function(input, output) {
  output$clusterPlot <- renderPlot({
    data <- iris[, 1:4]
    if (input$clust_method == "k-means") {
      set.seed(123)
      clusters <- kmeans(data, centers = inpu
```



青萌
数海

大模型生成代码 和 古法手写代码

- 生产效率：从“分钟级”到“秒级”
- 代码质量：从“绝对可控”到“需审核的敏捷”
- 思维方式：从“记忆型”到“意图型”
- 创造能力：从“经验驱动”到“模式匹配”

代码能力对比

青萌数海

维度	古法手写代码	大模型生成代码
启动成本	需要从零开始思考语法、API 调用方式。即使是很简单的功能（数据库读取），也需要逐字敲出。	**自然语言即代码**：你只需要用注释描述意图，AI 瞬间生成代码块。
重复劳动	反复编写样板代码（如循环、异常处理、图表美化）。	**消除样板代码**：AI 擅长填充这些固定模式，让你专注于核心业务逻辑。
查文档时间	遇到不熟悉的库(如R6)，需要停下来翻阅文档或google之类。	**内置知识库**：AI已基于海量文档训练，直接调用最合适的函数，省去上下文切换时间。
逻辑严谨性	开发者完全理解每一行代码的含义和副作用，能精确控制边界条件。	**黑盒风险**：生成的代码可能逻辑正确，但在特定业务场景下可能存在隐含错误。
算法优化	取决于程序员自身的算法功底，可能会写出低效但能跑的代码。	**最佳实践集成**：AI 倾向于生成相对高效、符合 规范的代码，有时甚至会给出你没想到的优化方案。
解决未知问题	面对从未见过的问题，程序员需要创造性拆解，一步步构建解决方案。	**依赖训练数据**：对于常见问题表现出色；对于极其冷门或全新的算法，可能会“一本正经地胡说八道”。
代码风格	带有强烈的个人印记（命名习惯、排版风格）。	**风格趋同化**：倾向于生成行业内最通用的写法，个性化较弱。
架构设计	擅长从零构建大型系统的顶层架构。	目前更适合**函数级别**或**模块级别**的代码生成，复杂架构仍需人类主导。

未来趋势

有了Copilot，让写代码的过程发生了巨大变化，重点从“怎么写代码”转移到了“怎么问问题”和“怎么理解结果”。

数据分析领域：**非标准化的分析任务**，**不确定性的业务理解**，还是有很大的提升空间。

- 有经验的人能力放大
- 跨行的人降低门槛
- 搬砖直接被替代

公司介绍

青萌数海

北京青萌数海科技有限公司 是一家以技术为驱动力的数据分析公司，为中央政府提供数据分析服务，帮助国家级客户提升业务监管和执法能力，业务主要涉及**国际贸易犯罪、反洗钱、反欺诈**、行业标准制定，经济运行分析等。公司长期服务于：海关，外管，药监等政府机构及相关行业。 http://dataqm.cn/

- 专注于探索未知领域，数据与业务相结合，可解释，可落地的数据分析。
- 数据分析师团队，国内外名校教育经历及国内知名企业从业经历。
- 非常具有挑战性，以落地为核心点。

谢谢您的聆听

如果你和我一样，欢迎加入我的团队!