قدم یار

قدم به قدم، همراه دانشجو...

WWW.GhadamYar.Com

جامع ترین و به‌روزترین پرتال دانشجویی کشور (پرتال دانش)

با ارائه خدمات رایگان، تحصیلی، آموزشی، رفاهی، شغلی و...

برای دانشجویان

۱) راهنمای ارتقاء تحصیلی. (کاردانی به کارشناسی، کارشناسی به ارشد و ارشد به دکتری)
۲) ارائه سوالات کنکور مقاطع مختلف سالهای گذشته، همراه پاسخ، به صورت رایگان
۳)معرفی روش‌های مقاله و پایان‌نامه نویسی و ارائه پکیج‌های آموزشی مربوطه
۴)معرفی منابع و کتب مرتبط با کنکورهای تحصیلی (کاردانی تا دکتری)
۵) معرفی آموزشگاه‌ها و مراکز مشاوره تحصیلی معتبر
۶)ارائه جزوات و منابع رایگان مرتبط با رشته‌های تحصیلی
۷)راهنمای آزمون های حقوقی به همراه دفترچه سوالات سالهای گذشته (رایگان)
۸)راهنمای آزمون های نظام مهندسی به همراه دفترچه سوالات سالهای گذشته (رایگان)
۹)آخرین اخبار دانشجویی، در همه مقاطع، از خبرگزاری های پربازدید
۱۰)معرفی مراکز ورزشی، تفریحی و فروشگاه‌های دارای تخفیف دانشجویی
۱۱)معرفی همایش‌ها، کنفرانس‌ها و نمایشگاه‌های ویژه دانشجویی
۱۲)ارائه اطلاعات مربوط به بورسیه و تحصیل در خارج و معرفی شرکت‌های معتبر مربوطه
۱۳)معرفی مسائل و قوانین مربوط به سربازی، معافیت تحصیلی و امریه
۱۴)ارائه خدمات خاص ویژه دانشجویان خارجی
۱۵)معرفی انواع بیمه‌های دانشجویی دارای تخفیف
۱۶)صفحه ویژه نقل و انتقالات دانشجویی
۱۷)صفحه ویژه ارائه شغل های پاره وقت، اخبار استخدامی
۱۸)معرفی خوابگاه‌های دانشجویی معتبر
۱۹)دانلود رایگان نرم افزار و اپلیکیشن های تخصصی و...
۲۰)ارائه راهکارهای کارآفرینی، استارت آپ و...
۲۱)معرفی مراکزتایپ، ترجمه، پرینت، صحافی و ... به صورت آنلاین
۲۲)راهنمای خریدآنلاین ارزی و معرفی شرکت های مطرح
۲۳) .....

WWW.GhadamYar.Ir WWW.PortaleDanesh.com WWW.GhadamYar.Org

۰۹۱۲ ۳۰ ۹۰ ۱۰۸ باما همراه باشید... ۰۹۱۲ ۰۹ ۰۳ ۸۰۱

www.GhadamYar.com

تحلیل های آماری با نرم افزار R

محمود مسلمان

m.mosalman@mail.sbu.ac.ir

تحلیل های آماری با نرم افزار R- محمود مسلمان

www.GhadamYar.com

فهرست

پیشگفتار ۳

فصل اول: مقدمات ۵

قراردادهای ابتدائی ۵

آشنایی با برخی توابع مقدماتی در R ۶

فصل دوم: ساختار داده ها در R ۸

بردار ۸

ایجاد بردارهای خاص ۸

انتخاب زیر مجموعه ای از بردارها ۹

ویژگی های یک بردار ۱۰

ماتریس ۱۱

انتخاب زیر مجموعه ای از یک ماتریس ۱۳

آرایه ۱۴

لیست ۱۶

تصحیح داده ها ۱۷

عامل و طبقه
۱۹۱۸

وارد کردن داده ها ۱۸

ذخیره داده ها ۱۹

فصل سوم: محاسبات ریاضی در R ۲۰

عملگر های ریاضی ۲۰

عملگرهای منطقی ۲۱

توابع مقدماتی ریاضی ۲۲

محاسبات ماتریسی ۲۴

حل دستگاه معادلات خطی ۲۹

مشتق ۳۰

انتگرال ۳۰

پیدا کردن ریشه های چند جمله ای ۳۱

احتمال و اعداد تصادفی ۳۱

فصل چهارم: نوشتن توابع در R ۳۴

چند عبارت و تابع مورد نیاز در برنامه نویسی ۳۵

حلقه ها در R ۳۷

فصل پنجم: آمار توصیفی و نمودار ها ۳۹

انواع داده ها: ۳۹

خلاصه کردن داده های کمی: ۴۰

نمودار شاخه و برگ ۴۳

هیستوگرام (بافت نگار) ۴۳

نمودار (QQ-Plot) یا چندک – چندک ۴۵

نمودار جعبه ای ۴۶

خلاصه کردن داده های رسته ای ۴۷

استفاده از جداول ۴۷

نمودار میله‌ای ۴۸

تحلیل های آماری با نرم افزار R - محمود مسلمان

www.GhadamYar.com

نمودار دایره‌ای ۵۰
تحلیل داده های رسته ای دو متغیره ۵۱
رسم داده های جدول ۵۲
داده های دو متغیره: کمی و رستهای ۵۳
فصل ششم: آزمون های آماری ۵۵
آزمون میانگین جامعه (یک نمونه ای) ۵۵
آزمون کلاسیک (t-test): ۵۵
آزمون رتبه ای ویلکاکسون: ۵۷
آزمون برابری واریانس های دو جامعه ۵۸
آزمون برابری میانگین های دو جامعه: ۵۹
آزمون t-استیودنت برای داده های جفتی (paired t-test) ۶۰
آزمون جفتی ویلکاکسون برای مقایسه میانگین های دو جامعه ۶۱
آزمون صفر بودن ضریب همبستگی ۶۲
آزمون نیکویی برازش خی- دو ۶۳
آزمون استقلال خی- دو ۶۴
آزمون برابری نسبت جامعه با یک نسبت فرضی (آزمون دقیق دو جمله ای) ۶۵
آزمون نسبت با استفاده از تقریب نرمال ۶۷
آزمون برابری نسبت های دو جامعه ۶۷
تحلیل واریانس ۶۸
فصل هفتم: رگرسیون ۷۰
رگرسیون خطی ساده ۷۰
فصل هشتم: شبیه سازی ۷۵
برآورد ماکزیمم درستنمایی ۷۵
قضیه حد مرکزی ۷۶
مراجع و مآخذ ۷۸

تحلیل های آماری با نرم افزار R- محمود مسلمان

www.GhadamYar.com

پیشگفتار

امروزه با گسترش روز افزون علم آمار و کاربردهای آن در سایر علوم لزوم آشنایی با نرم افزار های آماری که برای تجزیه و تحلیل داده ها و همچنین بسط و توسعه روشهای نوین آماری به کار می‌روند بیش از پیش قابل درک است. در این میان و در سالهای اخیر نرم افزار R از پیشرفت و محبوبیت قابل ملاحظه ای در بین پژوهشگران و محافل علمی دنیا برخوردار بوده است. کتابها و مقالات متعدد، سمینار های آموزشی در دانشگاه های معتبر جهان و سایت های اینترنتی گوناگون تنها برای گسترش و آموزش این نرم افزار بوجود آمده است. نرم افزار R بر اساس زبان آماری S ایجاد شده است. زبان آماری S در سال ۱۹۶۰ توسط John Chambers و همکارانش در لابراتوار Bell به منظور برنامه نویسی آماری برای تحلیل داده ها و مدل بندی پیشرفته ایجاد شد. بعدها نسخه تجاری S تحت عنوان S-PLUS وارد بازار شد. از آنجایی که S-PLUS یک محصول گران قیمت بود (البته این مشکل در ایران حل شده است!) دو آمار دان نیوزلندی، Ross Ihaka و Robert Gentleman از دانشگاه Auckland تصمیم گرفتند نسخه ارزانتری از S برای اهداف آموزشی بنویسند. اینکه حرف R قبل از حرف S است و اینکه اول اسامی هر دو نفر با R شروع می شود، از دلایل نامگذاری این نرم افزار به نام R بود. متن اصلی R در سال ۱۹۹۵ تحت لیسانس GPL عرضه شد و اعضای تیم توسعه به ۱۵ نفر افزایش یافت. نسخه 1.0.0 در ۲۹ فوریه سال ۲۰۰۰ عرضه شد. در حقیقت R یک ویرایش متن باز (Open Source) از S-PLUS است و به طور رایگان قابل دریافت است. برای دسترسی سریع و آسان کاربران همه نقاط جهان در سایت اینترنتی اکثر دانشگاههای معتبر دنیا یک بازتاب برای سایت اصلی CRAN ایجاد شده است که در ایران این بازتاب در سایت در دانشگاه فردوسی مشهد وجود دارد. آخرین نسخه R و پکیج های آن را می توانید از این آدرس دریافت کنید:

http://www.cran.um.ac.ir

نرم افزار R دارای مزایای منحصر بفردی است که آنرا از سایر نرم افزار های آماری متمایز کرده است:

- R یک نرم افزار متن باز (open source) و رایگان است.
- R به عنوان یک بسته آموزشی رایگان در مراکز آموزشی قابل استفاده است.
- توسعه پذیری و انعطاف R باعث دسترسی همگان در کوتاه ترین زمان به روش های نوین آماری شده است.

قابلیت های فنی R:

- کاربری و نگهداری داده ها به صورت مفید و مؤثر
- دارا بودن بسیاری از عملگر های لازم برای محاسبات ماتریسی و آرایه ای
- دارا بودن مجموعه کاملی از ابزار تجزیه و تحلیل داده ها

www.GhadamYar.com

- امکانات گرافیکی منحصر بفرد برای تحلیل داده ها و نمایش آنها
- زبان برنامه نویسی ساده با قابلیت های بروز رسانی بالا

بسیاری از متخصصان علوم آماری جهت معرفی روش های ابداعی خود برای تحلیل داده ها و غیره، نتایج مطالعات خود را به راحتی و بدون هیچ هزینه ای به صورت پکیج هایی برای R در اختیار سایرین قرار می دهند و همین امر R را به ابزار گسترش و پیشرفت سریعتر علم آمار تبدیل کرده است.

تحلیل های آماری با نرم افزار R- محمود مسلمان

www.GhadamYar.com

قراردادهای ابتدائی

- در R دستورات در پنجره R console و بعد از عملگر اعلان یعنی ">" وارد می‌شود.
- برای ذخیره یک ساختار داده باید به آن یک نام اختصاص داد. بدین منظور از عملگر تخصیص " = "، " -> " یا "<- " استفاده می‌شود.

```
> x=5
```

- یکی از ساده ترین دستورات R تایپ نام یک ساختار داده برای مشاهده محتویات آن است:

```
> x
[1] 5
```

عدد ۱ داخل براکت یعنی سطر نمایش دهنده از اولین عنصر x شروع می‌شود.

تابع print(x) نیز کار دستور فوق را انجام می‌دهد. از این دستور بیشتر برای نوشتن تابع و در حلقه ها استفاده می‌شود.

- نام یک ساختار (متغیر) باید با یک حرف شروع شود (A تا Z و a تا z) و می تواند شامل حروف، اعداد و نقطه باشد. برای مثال تمام اسامی زیر در R قابل قبول هستند.

```
Data, data1, data.12, Data.New
```

- R نسبت به کوچک یا بزرگ بودن حروف حساس است. بنابراین در R، 'X' و 'x' متفاوتند.
- R معمولاً از فاصله ها چشم پوشی می‌کند.

```
> 12    + 2
[1] 14
```

- اکثر دستورات در R به صورت تابع هستند. شکل کلی توابع به صورت زیر است:

function(شناسه‌ها)

شناسه های تابع، اعداد یا عباراتی هستند که چگونگی عملکرد تابع را کنترل می‌کنند. هر تابع تعدادی شناسه دارد که برخی لازم و برخی اختیاری هستند. در صورتیکه نام تابع را بنویسید، متن تابع روی صفحه نمایش ظاهر می‌شود.

www.GhadamYar.com

آشنایی با برخی توابع مقدماتی در R

- **تابع attach()**

مهمترین کاربرد این تابع ایجاد متغییر مستقل از یک ساختار ترکیبی داده است، مثلا اگر متغیر x ترکیب دو متغیر y و z باشد، با اجرای دستور attach(x)، متغیر های y و z به عنوان دو متغیر آزاد قابل دسترسی هستند.

- **تابع detach()**

این تابع عکس عملیات تابع attach() را انجام می‌دهد.

- **تابع help()**

برای استفاده از راهنمائی‌های R از تابع help() یا عملگر " ? " استفاده می‌شود. شناسه این تابع اغلب نام تابعی است که می خواهیم درباره آن اطلاعاتی کسب کنیم.

```
> help( نام تابع )
> ? نام تابع
```

- **تابع apropos()**

برای جستجوی یک عبارت در اسامی توابع از تابع apropos() یا عملگر " ?? " استفاده می‌شود. مثلاً برای مشاهده همه توابعی که دارای عبارت mean هستند از دستور زیر استفاده کنید:

```
> ??mean
```

- **تابع ls()**

لیست متغیر هایی را که تاکنون ساخته اید نمایش می‌دهد. برای مشاهده اسامی همه متغیرها از عبارت زیر استفاده کنید:

```
> ls()
```

برای مشاهده اسامی متغیر هایی که حرف m دارند از عبارت زیر استفاده کنید:

```
> ls(pat="m")
```

- **تابع rm()**

برای حذف کردن متغیر های ایجاد شده از این تابع استفاده می‌شود. به منظور حذف کردن همه متغیر ها از این عبارت استفاده کنید:

```
> rm(list=ls())
```

برای حذف متغیر x به این صورت عمل کنید:

```
> rm(x)
```


www.GhadamYar.com

- **تابع sink()**

این تابع خروجی یک برنامه یا عبارت را در یک فایل متن ذخیره می‌کند. برای استفاده از این تابع باید قبل از اجرای دستوراتی که مایل به ذخیره کردن خروجی آنها هستیم، یک نام برای فایل متن داخل " " به عنوان شناسه در تابع sink() قرار دهیم، بعد از اجرای دستورات دوباره تابع sink() را بدون شناسه اجرا می‌کنیم.

مثال:

```
> sink("list.txt")
> ls()
> sink()
```

در این مثال خروجی تابع ls() در فایل list.txt ذخیره می‌شود.

- **تابع source()**

اغلب مایلیم که لیستی از دستورات را از یک فایل خارجی بخوانیم و اجرا کنیم، فایل اسکریپت خود را در Notepad نوشته و آنرا ذخیره کنید. در نرم افزار R از منوی File گزینه 'Source R code' را انتخاب کنید. با این کار همه دستورات داخل فایل اسکریپت شما خوانده و اجرا می‌شود، همچنین می‌توانید از دستور زیر برای انجام این کار استفاده کنید:

```
> source("h:/analysis.txt")
```

اگر شما مایلید که تنها بخشی از دستورات داخل فایل اسکریپت اجرا شود می‌توانید از منوی File گزینه 'Display file' را انتخاب کنید، با این کار فایل اسکریپت در پنجره ای جدید باز می‌شود و شما می توانید دستورات مورد نظرتان را copy و paste کنید.

- **تابع data()**

در R تعداد زیادی ساختار داده از قبل وجود دارد که برای اهداف آموزشی و مثال ها استفاده می‌شود. برای مشاهده لیست این داده ها از تابع data() استفاده کنید. مثلاً یکی از این ساختارها Orange است که داده های مربوط به رشد درختان پرتقال را شامل می‌شود.

- **تابع library()**

لیست تمام پکیج های نصب شده در R را نمایش می‌دهد. برای بارگذاری یک پکیج خاص نیز از این دستور استفاده می‌شود. مثلاً برای بارگذاری پکیج foreign از دستور زیر استفاده کنید.

```
> library(foreign)
```

این پکیج برای خواندن داده هایی که توسط نرم افزار های Minitab، SAS، SPSS و ... ذخیره شده، استفاده می‌شود.

www.GhadamYar.com

بردار

ساده ترین نوع ساختار داده در R بردار است که مجموعه ای مرتب از مقادیر عددی ، کاراکتری و یا منطقی است. رایج ترین روش ساختن بردار ها استفاده از تابع c () می‌باشد. با استفاده از این تابع می‌توان مقادیر دلخواه را با هم ترکیب کرده، یک بردار ساخت.

مثال:

```
> x <- c(2,3,5,7)
> x
[1] 2 3 5 7
> y<-c(11,13,15,19)
> z<-c(x,y,20)
> z
[1]  2  3  5  7 11 13 15 19 20
```

یک بردار می‌تواند مجموعه ای از کاراکتر ها باشد:

```
> c("green", "blue sky", "-77")
[1] "green"  "blue sky"  "-77"
```

توجه کنید که "77-" یک کاراکتر است نه یک عدد، بنابراین نمی‌توان این بردار را در یک عدد ضرب کرد.

ایجاد بردارهای خاص

اغلب مایلیم که برداری به صورت دنباله ای از اعداد داشته باشیم، یا اینکه برداری شامل عناصر یکسان باشد. چندین روش برای ایجاد چنین بردارهایی وجود دارد.

مثال: برای ایجاد لیست اعداد بین ۱۰ و ۳۰ با استفاده از عملگر دنباله " : " داریم:

```
> x = 10:30
> x
[1] 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
[20] 29 30
```


www.GhadamYar.com

اعداد داخل براکت که در ابتدای هر سطر نمایش داده می‌شود نشان دهنده شماره عنصری از بردار است که سطر جدید با آن شروع می‌شود. چنانکه سطر اول با اولین عنصر بردار (۱۰) و سطر دوم با بیستمین عنصر بردار (۲۹) شروع می‌شود.

شما می‌توانید همین عمل را با استفاده از دستور seq () انجام دهید:

```
> x = seq(10,30,by=1)
```

این دستور دنباله ای از اعداد بین ۱۰ و ۳۰ را با هر با افزایش ۱ واحد ایجاد می‌کند. همچنین می‌توانید با مشخص کردن تعداد عناصر مورد نظرتان نتیجه مشابهی را بدست آورید:

```
> x = seq(10,30,length=21)
```

یک دستور مفید دیگر rep () می‌باشد که برای تکرار یک عدد یا مجموعه ای از اعداد به کار می‌رود، برای مثال، جهت ۵ بار تکرار عدد ۳ داریم:

```
> rep(3,5)
[1] 3 3 3 3 3
```

شناسه های دستور rep ()، بردارها نیز می‌توانند باشند:

```
> x=c(1,2,3)
> rep(x,3)
[1] 1 2 3 1 2 3 1 2 3
```

انتخاب زیر مجموعه ای از بردارها

می‌توان با استفاده از دو براکت " [ ] " به عناصر مشخصی از یک بردار دست یافت:

مثال: فرض کنید یک بردار x به طول ۵ ساخته اید:

```
> x=c(4,7,8,11,16)
```

برای نمایش عنصر چهارم از بردار x بنویسید:

```
> x[4]
[1] 11
```

برای نمایش عنصر دوم و پنجم بردار x عبارت زیر را به کار برید:

```
> x[c(2,5)]
[1]  7 16
```


www.GhadamYar.com

اگر می‌خواهید همه عناصر بردار x به جز عناصر اول و سوم نمایش داده شود، قبل از تابع c() از علامت منفی "-" استفاده کنید:

```
> x[-c(1,3)]
[1]  7 11 16
```

برای تغییر عنصر چهارم از دستور زیر استفاده کنید:

```
> x[4]=20
> x
[1]  4  7  8 20 16
```

یک تابع سودمند برای دسترسی به عناصر دلخواه یک بردار تابع which() می‌باشد:

```
> x = c(10, 7, 9, 10, 10, 12, 9, 10)
```

برای نمایش موقعیت عناصری که برابر ۱۰ هست از دستور زیر استفاده کنید:

```
> which(x==10)
[1] 1 4 5 8
```

برای استخراج عناصر کمتر از ۱۰ بنویسید:

```
> x[which(x<10)]
[1] 7 9 9
```

ویژگی های یک بردار

هر بردار دارای سه ویژگی طول، حالت و نام است.

طول یک بردار بیانگر تعداد عناصر آن است و با تابع length() نشان داده می‌شود.

حالت بردار بیان کننده نوع عناصر آن است و یکی از انواع عددی، کاراکتری، مختلط و منطقی است. عناصر یک بردار همه باید دارای یک حالت باشند. برای نمایش حالت بردار از دستور mode() استفاده می‌شود.

مثال:

```
> x=c(1,3,6,7,11)
> length(x)
[1] 5
```


www.GhadamYar.com

```
> mode(x)
[1] "numeric"
```

عناصر یک بردار را می‌توان به صورت زیر نام گذاری کرد:

```
> a <- c(x = 1, y = 5.8, z = -77)
> a
    x     y     z
  1.0   5.8 -77.0
```

یک روش معادل نیز به صورت زیر است:

```
> a <- c(1, 5.8, -77)
> names(a) <- c("x", "y", "z")
```

ماتریس

ماتریس ها یا آرایه های دوطرفه از ساختارهای مهم در R هستند. در تحلیل داده ها، متغیرها به صورت ستون های ماتریس و موقعیت ها یا واحدهای آزمایشی به صورت سطرهای یک ماتریس معرفی می‌شوند. در R می‌توان هر نوع داده ای را در قالب ماتریس ها ذخیره کرد، برای مثال می‌توان ماتریسی از حروف یا کلمات را داشت.

برای تبدیل یک بردار به ماتریس از تابع dim() استفاده می‌شود. در حقیقت این تابع بعد ماتریس را مشخص می‌کند. به کمک این تابع یک بردار عددی به طول ۲ که بیانگر تعداد سطر ها و ستون های ماتریس است، به بعد تخصیص می‌دهیم.

مثال:

```
> x <- 1:12
> dim(x) <- c(3,4)
> x
[,1] [,2] [,3] [,4]
[1,] 1  4  7  10
[2,] 2  5  8  11
[3,] 3  6  9  12
```


www.GhadamYar.com

بدین ترتیب اعداد ۱ تا ۱۲ را در یک ماتریس ۴×۳ جای دادیم، توجه کنید که در این روش اعداد به صورت ستونی در ماتریس جای می‌گیرند.

برای ساختن ماتریس‌ها می‌توانید از تابع matrix() نیز استفاده کنید. تعداد سطرها و ستون ها در تابع matrix() با شناسه های nrow و ncol تعیین می‌شود. برای اینکه ماتریس به طور سطری پر شود، مقدار شناسه byrow را برابر T قرار دهید.

```
> matrix(1:12,nrow=3,byrow=T)
[,1] [,2] [,3] [,4]
[1,] 1 2 3 4
[2,] 5 6 7 8
[3,] 9 10 11 12
```

برخی توابع مفید در رابطه با ماتریس ها عبارتند از rownames() ، colnames() که برای نام گذاری سطرها و ستون‌های ماتریس استفاده می‌شود.

مثال:

```
> x<-matrix(1:9,nrow=3)
> rownames(x)<-LETTERS[1:3]
> x
  [,1] [,2] [,3]
A    1    4    7
B    2    5    8
C    3    6    9
```

بردار کاراکتری LETTERS یک متغیر تعریف شده در R است که شامل حروف بزرگ A تا Z می باشد، بردار حروف کوچک نیز letters نام دارد.

برای ترکیب بردار ها یا ماتریس ها به منظور ایجاد ماتریس جدید از دو تابع rbind() و cbind() استفاده می‌شود. تابع cbind() ماتریس ها(یا بردارها) را به صورت ستونی و تابع rbind() آنها را به صورت سطری ترکیب می‌کند.

www.GhadamYar.com

مثال:

```
> cbind(A=1:4,B=5:8,C=9:12)
     A B  C
[1,] 1 5  9
[2,] 2 6 10
[3,] 3 7 11
[4,] 4 8 12

> rbind(A=1:4,B=5:8,C=9:12)
  [,1] [,2] [,3] [,4]
A    1    2    3    4
B    5    6    7    8
C    9   10   11   12
```

انتخاب زیر مجموعه ای از یک ماتریس

انتخاب یک زیرمجموعه از یک ماتریس همانند انتخاب یک زیر مجموعه از بردارها است، با این تفاوت که ماتریس ها دارای دو بعد هستند. برای انتخاب درایه سطر i-ام و ستون j-ام ماتریس از عبارت زیر استفاده کنید.

نام ماتریس[i,j]

برای انتخاب چند سطر یا ستون از یک ماتریس می توان از تابع c() استفاده کرد.

مثال:

```
> x<-matrix(1:9,ncol=3)
> x
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```


www.GhadamYar.com

```
> x[3,2]
[1] 6
> x[1,]
[1] 1 4 7
> x[,-c(2,3)]
[1] 1 2 3
> x[x>5]
[1] 6 7 8 9
```

برای مشاهده بعد ماتریس، حالت ماتریس و طول ماتریس (تعداد درایه های ماتریس) از توابع dim()، mode() و length() استفاده کنید. همچنین از تابع dimnames() برای نامگذاری سطر ها و ستون های ماتریس استفاده می‌شود.

مثال:

```
> dimnames(x)<-list(paste("row",letters[1:3]),
+ paste("col",LETTERS[1:3]))
> x
      col A col B col C
row a     1     4     7
row b     2     5     8
row c     3     6     9
```

آرایه

آرایه تعمیم یافته ماتریس است. در آرایه ها مقدار ویژگی بعد می تواند عددی بیش از دو باشد.

فرم ساختن آرایه به صورت زیر است:

```
> array(data, dim)
```

که به جای data بردار داده ها مثلاً c(64,22,45,11,61,32) و به جای dim بعد آرایه را قرار دهید.

اگر در تابع array() داده ها معرفی نشوند، آرایه ای شامل مقادیر گم شده که با NA نشان داده می‌شوند، ساخته می‌شود.

www.GhadamYar.com

مثال:

```
> x=array(1:24,c(3,4,2))
> x
, , 1
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
, , 2
     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

تابع array() ابتدا مقدار اولین بعد را کامل می‌کند، سپس دومین بعد و به همین ترتیب بعد های دیگر را کامل می‌کند.

برای ساختن آرایه از بردارهایی که قبلاً ساخته اید، از تابع dim() استفاده کنید.

مثال:

```
> x=c(1:8,11:18,111:118)
> dim(x)=c(2,3,4)
> x
, , 1
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
, , 2
     [,1] [,2] [,3]
[1,]    7   11   13
[2,]    8   12   14
, , 3
     [,1] [,2] [,3]
[1,]   15   17  111
[2,]   16   18  112
```


www.GhadamYar.com

```
, , 4
     [,1] [,2] [,3]
[1,]  113  115  117
[2,]  114  116  118
```

اگر بعد دوم این آرایه را ثابت کنیم، یک ماتریس ۴×۲ خواهیم داشت:

```
> x[,2,]
     [,1] [,2] [,3] [,4]
[1,]    3   11   17  115
[2,]    4   12   18  116
```

لیست

اغلب مایلیم که مجموعه داده های کوچکتر را در یک ترکیب یکجا داشته باشیم، به این ترکیب لیست می‌گوییم. لیست ساختاری است که می‌تواند مقادیری با حالت های مختلف مانند مقادیر عددی و کاراکتری با داشته باشد.

برای ساختن لیست از تابع `list()` استفاده کنید.

به عنوان مثال اگر مقادیر فشار خون ۵ نفر قبل و بعد از مصرف داروی `xyz` را در بردار های `x1` و `x2` داشته باشیم، می‌توان این دو بردار و همچنین نام دارو را در یک لیست با هم داشت.

```
> x1=c(12,14,13,14,16)
> x2=c(12,13,10,11,15)
> mylist=list(before=x1,after=x2,drug.name="xyz")
> mylist
$before
[1] 12 14 13 14 16

$after
[1] 12 13 10 11 15

$drug.name
[1] "xyz"
```

دقت کنید که در داخل تابع `list()` عمل نامگذاری را نیز انجام دادیم.

www.GhadamYar.com

با استفاده از علامت $ می توان جزء خاصی از یک لیست را نشان داد:

```
> mylist$after
[1] 12 13 10 11 15
```

لیست ها هم مثل سایر ساختارها دارای دو ویژگی طول و حالت هستند. طول لیست تعداد مؤلفه های لیست است.

```
> length(mylist)
[1] 2
```

حالت یک لیست list است و ویژگی نام لیست، نام مؤلفه های یک لیست ار مشخص می کند. برای نمایش نام یک لیست از تابع names () استفاده کنید.

```
> names(mylist)
[1] "before" "after"
```

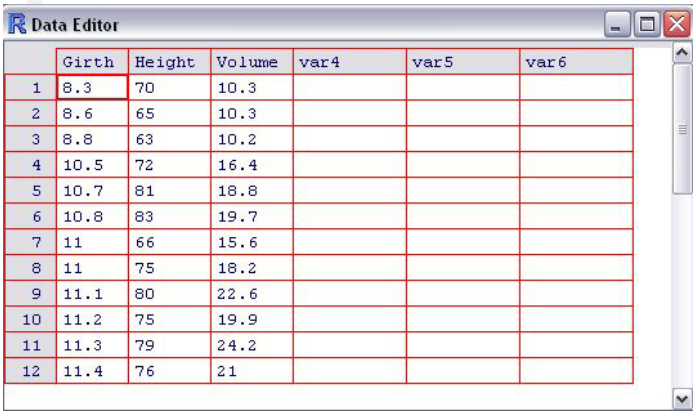
تصحیح داده ها

با استفاده از دستور data.entry() می‌توان داده ها را در یک فرم جدولی مشاهده کرد و به تصحیح آنها پرداخت. همچنین در این جدول می‌توان متغیر جدیدی را تعریف کرد و داده های دلخواه را به آن اختصاص دهیم. با بستن این پنجره، تغییرات به صورت خودکار اعمال می‌شود.

مثال:

```
> data(trees)
> data.entry(trees)
```

با اجرای دستور فوق پنجره Data Editor که در زیر ملاحظه می‌کنید، برای ویرایش ساختار داده trees ظاهر می‌شود.



	Girth	Height	Volume	var4	var5	var6
1	8.3	70	10.3			
2	8.6	65	10.3			
3	8.8	63	10.2			
4	10.5	72	16.4			
5	10.7	81	18.8			
6	10.8	83	19.7			
7	11	66	15.6			
8	11	75	18.2			
9	11.1	80	22.6			
10	11.2	75	19.9			
11	11.3	79	24.2			
12	11.4	76	21			

www.GhadamYar.com

عامل و طبقه

در تحلیل داده ها، در بسیاری از موارد با داده های کیفی از قبیل گروه خونی، جنسیت یا وضعیت تأهل سر و کار داریم. داده‌های کیفی مقادیر عددی ندارند، اما می‌توان آنها را به یک طبقه منسوب کرد. در R داده های کیفی به وسیله ساختاری به نام "عامل" (factor) نمایش داده می‌شود.

برای ساختن یک عامل از تابع factor() استفاده می شود.

برای مثال در مورد استعمال سیگار از ۵ نفر سؤال شده است.

```
> smoke=c("Yes","No","No","Yes","Yes")
> factor(smoke)
[1] Yes No  No  Yes Yes
Levels: No Yes
```

در اینجا دو سطح با نام های Yes و No از یک متغیر کیفی ایجاد می‌شود.

سطوح مورد نظر و تعداد سطوح را می توان با شناسه levels تعیین کرد، همچنین برای نام گذاری سطوح مختلف یک عامل از شناسه label استفاده می‌شود.

```
> factor(c(1,2,2,1,3,2,1),levels=c(1,2),label=c("Yes","No"))
[1] Yes  No   No   Yes  <NA> No   Yes
Levels: Yes No
```

وارد کردن داده ها

اغلب شما با مواردی برخورد می‌کنید که باید داده ها را از منابعی خارج از محیط R بازخوانی کنید. اگر داده ها در یک فایل متنی (text) ذخیره شده باشند می توانید با دستور read.table() آنها را بازخوانی کنید. این دستور شناسه‌های زیادی دارد که بسته به نحوه چیدمان داده‌ها در فایل متنی قابل استفاده هستند. فرم ساده این دستور به این صورت است که آدرس فایل متنی را داخل گیومه " " به عنوان شناسه در تابع read.table() قرار می‌دهیم.

```
> data = read.table("c:/data.txt")
```


www.GhadamYar.com

معمولاً فایل‌های حاوی داده اسامی متغیرها را در سطر اول خود دارند، که در این صورت می‌توان با به کار بردن شناسه header=True یا بطور خلاصه h=T این موضوع را به R اطلاع دهیم. نوع کاراکتر به کار رفته در فایل متنی برای جدا کردن داده ها را نیز می‌توان با شناسه sep مشخص کرد.

یک فایل داده به شکل زیر در نظر بگیرید:

height , weight
169, 61
167 , 69
...
163, 63

سطر اول شامل اسامی متغیر هاست و داده ها با علامت " , " از یک دیگر جدا شده اند پس با دستور زیر این فایل را در متغیر mydata ذخیره می‌کنیم:

```
> mydata = read.table("h:/hgtdata.dat", h=T, sep=",")
```

ذخیره داده ها

اگر مایلید چارچوب داده ای که در R ایجاد کرده‌اید را بر روی هارد خود ذخیره کنید تابع write.table() این امکان را به شما می‌دهد. این دستور نیز دارای گزینه های اختیاری متعددی است. برای ذخیره کردن یک چارچوب داده مطابق آنچه در قسمت قبل ایجاد کردیم از دستور زیر استفاده می‌کنیم:

```
> write.table(hgtdata, file="h:/hgtdata.dat", col.names=T, sep=",")
```

توجه کنید که دستور فوق از شناسه col.names برای بازیابی نام ستون ها در خط اول استفاده می‌کند.

www.GhadamYar.com

عملگر های ریاضی

عملگر های حسابی پایه در جدول زیر آمده اند، که ترتیب عمل آنها طبق ترتیب معمول است(توان- ضرب و تقسیم -جمع و تفریق). با استفاده از پرانتز ( ) می‌توان اولویت‌ها را تغییر داد، توجه داشته باشید که نباید از { } یا [ ] استفاده کرد، این علامات برای اهداف دیگری در R مورد استفاده قرار می‌گیرند.

عملگر	شرح
+	جمع
–	تفریق
*	ضرب
/	تقسیم
^	توان
%/%	خارج قسمت تقسیم
%%	باقیمانده تقسیم

جدول ۱- عملگر های ریاضی

در مثال های زیر نحوه استفاده از این عملگر ها نشان داده شده است:

```
> 2+4
[1] 6

> y=0
> x=4
> x*y^2
[1] 0
> x^4
[1] 256
```


www.GhadamYar.com

```
> z=5
> (x+y)*z/x
[1] 5

> 25%/%4
[1] 6

> 25%%4
[1] 1
```

عملگرهای منطقی

عملگر های منطقی برای شرطی کردن اجرای دستورات استفاده می شود.

عملگر	شرح
>	بزرگتر
<	کوچکتر
<=	کوچکتر یا مساوی
≥	بزرگتر یا مساوی
==	مساوی
! =	نامساوی
!	نقیض
&	"و" منطقی
|	"یا" منطقی

جدول ۲- عملگر های منطقی

از تابع شرطی کننده `if()` که بعداً معرفی می شود می توان برای شرطی کردن با استفاده از عملگر های منطقی استفاده کرد:

```
> x=2
> y=5
```


www.GhadamYar.com

```
> if(x<y) z=x+1
> z
[1] 3
> if(y==4|x==2) w=x+y
> w
[1] 7

> a=c(2,4,7,11)
> a[a≥6]
[1]  7 11
```

توابع مقدماتی ریاضی

همه توابع رایج و شناخته شده ریاضی در R موجود است، در جدول زیر برخی از این توابع را مشاهده می کنید:

تابع	شـــــــرح
abs()	قدر مطلق
sum()	مجموع
prod()	حاصلضرب
sign()	نشانه
cumsum()	جمع تجمعی اعداد یک بردار
cumprod()	ضرب تجمعی اعداد یک بردار
exp()	تابع نمایی
floor()	جزء صحیح یک عدد
gamma()	تابع گاما
factorial()	تابع فاکتوریل
log()	لگاریتم طبیعی
log10()	لگاریتم در مبنای ۱۰
round()	گرد کردن اعداد
sqrt()	ریشه دوم عدد
22runk()	نزدیک ترین عدد صحیح به عدد
sin(),cos(),tan()	توابع مثلثاتی

www.GhadamYar.com

تابع	شرح
asin(),acos(),atan()	توابع معکوس مثلثاتی
sinh(),cosh(),tanh()	توابع مثلثاتی هیپربولیک
asinh(),acosh(),atanh()	وارون توابع مثلثاتی هیپربولیک
choose()	ترکیب
min(),max()	می نیمم و ماکزیمم
pmin(),pmax()	می نیمم و ماکزیمم موازی

جدول ۳- توابع رایج ریاضی در R

در مثالهای زیر نحوه استفاده از توابع فوق را مشاهده می‌کنید:

```
> abs(2-4)
[1] 2

> sum(1:10)
[1] 55

> floor(2.3)
[1] 2

> factorial(6)
[1] 720

> log(0)      #تعریف نشده
[1] -Inf

> round(sqrt(2),4)
[1] 1.4142

> cos(2*pi)
[1] 1

> pmin(c(1,4,9),c(2,3,10))
[1] 1 3 9

> choose(10,7)      # 10!/(7!*3!)
[1] 120
```

در R عبارت inf به معنی بینهایت است.

www.GhadamYar.com

محاسبات ماتریسی

یکی از قابلیت‌های نرم افزار R توانایی بالای آن در انجام محاسبات ماتریسی است. محاسبات برداری و ماتریسی در R به صورت عنصر به عنصر انجام می‌گیرد بنابراین برای انجام محاسباتی از قبیل جمع یا ضرب دو ماتریس باید ابعاد ماتریس‌ها برابر باشد. در بردارها، اگر طول یک بردار از بردار دیگر کمتر باشد، عناصر بردار با طول کمتر به طور متناوب تکرار می‌شوند تا زمانی که طول دو بردار برابر شود.

در مثال زیر بردار x که دارای طول ۳ است برای اینکه با بردار y با طول 6 جمع شود، به طور متناوب تکرار می شود .

```
> x=c(1,3,5)
> y=c(2,4,6,8,10,12)
> x+y
[1]  3  7 11  9 13 17
```

در جدول زیر چند تابع مورد نیاز برای محاسبات ماتریسی ذکر شده است.

تابع	شـــــرح
diag()	بدست آوردن قطر ماتریس و ساختن یک ماتریس قطری
eigen()	بدست آوردن مقادیر و بردارهای ویژه یک ماتریس
t()	محاسبه ترانهاده یک ماتریس (A^t)
det()	محاسبه دترمینان یک ماتریس ($\|A\|$)
ginv()[1]	محاسبه معکوس یک ماتریس (A^{-1})

جدول ۴ - توابع مربوط به محاسبات ماتریسی

[1] این دستور در پکیج MASS قرار دارد.

www.GhadamYar.com

مثال ها: **کاربرد عملگر های حسابی:**

ابتدا ماتریس های A و B را به صورت زیر می‌سازیم:

```
> A=matrix(c(1,0,4,2,-1,1,0,3,1),nrow=3)
> A
     [,1] [,2] [,3]
[1,]    1    2    0
[2,]    0   -1    3
[3,]    4    1    1
> r=c(1,3,5)
> s=c(2,4,6)
> t=c(-2,0,1)
> B=cbind(r,s,t)
> B
     r s  t
[1,] 1 2 -2
[2,] 3 4  0
[3,] 5 6  1
```

برای انجام جمع دو ماتریس داریم:

```
> A+B
     r s  t
[1,] 2 4 -2
[2,] 3 3  3
[3,] 9 7  2
```

ضرب :

```
> A*B
      r  s t
[1,]  1  4 0
[2,]  0 -4 0
[3,] 20  6 1
```


www.GhadamYar.com

توجه کنید که در دستور های فوق جمع و ضرب به صورت عنصر در عنصر انجام گرفت، برای انجام ضرب ماتریس‌ها به صورتی که در جبر خطی خوانده ایم و باید تعداد ستون های ماتریس اول برابر تعداد سطر های ماتریس دوم باشد از عملگر %*% استفاده می‌کنیم:

```
> A%*%B
      r  s  t
[1,]  7 10 -2
[2,] 12 14  3
[3,] 12 18 -7
```

انتظار داریم B×A متفاوت از A×B باشد:

```
> B%*%A
     [,1] [,2] [,3]
[1,]   -7   -2    4
[2,]    3    2   12
[3,]    9    5   19
```

از این عملگر همچنین برای انجام ضرب داخلی بردارها نیز استفاده می‌شود:

```
> c(0,2,3)%*%c(1,4,1)
     [,1]
[1,]   11
```

- **کاربرد توابع مقدماتی:**

اعمال توابع مقدماتی ریاضی بر روی یک ماتریس (بردار) باعث اعمال آن تابع بر روی تک تک درایه های ماتریس (بردار) می‌شود.

ماتریس F را به صورت زیر تعریف می‌کنیم:

```
> F=matrix(c(11,10,15,7,8,11,15,14,12),nrow=3)
```


www.GhadamYar.com

ریشه دوم درایه های F :

```
> sqrt(F)
         [,1]     [,2]     [,3]
[1,] 3.316625 2.645751 3.872983
[2,] 3.162278 2.828427 3.741657
[3,] 3.872983 3.316625 3.464102
```

گرد کردن حاصل لگاریتم طبیعی درایه های F:

```
> trunc(log(F))
     [,1] [,2] [,3]
[1,]    2    1    2
[2,]    2    2    2
[3,]    2    2    2
```

- **کاربرد توابع در محاسبات ماتریسی:**

ترانهاده ماتریس F:

```
> t(F)
     [,1] [,2] [,3]
[1,]   11   10   15
[2,]    7    8   11
[3,]   15   14   12
```

دترمینان ماتریس F:

```
> det(F)
[1] -158
```

برای مشاهده عناصر روی قطر اصلی ماتریس F از دستور زیر استفاده کنید:

```
> diag(F)
[1] 11  8 12
```


www.GhadamYar.com

برای ایجاد ماتریس قطری با عناصر مشخص روی قطر آن به صورت زیر عمل می‌کنیم:

```
> diag(c(5,10,15))
     [,1] [,2] [,3]
[1,]    5    0    0
[2,]    0   10    0
[3,]    0    0   15
```

برای ایجاد ماتریس واحد (I_4) داریم:

```
> diag(4)
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1
```

مقادیر ویژه ماتریس F:

```
> eigen(F)
$values
[1] 34.7014812 -4.6753398  0.9738586

$vectors
           [,1]       [,2]        [,3]
[1,] -0.5582404 -0.5330419 -0.60833622
[2,] -0.5395383 -0.4018480  0.79283377
[3,] -0.6302905  0.7445700  0.03662857
```

معکوس ماتریس F (ابتدا باید پکیج MASS را بارگذاری کنید) :

```
> library(MASS)
> ginv(F)
            [,1]       [,2]        [,3]
[1,]  0.36708861 -0.5126582  0.13924051
[2,] -0.56962025  0.5886076  0.02531646
[3,]  0.06329114  0.1012658 -0.11392405
```


www.GhadamYar.com

می‌توانیم صحت رابطه $(F^{-1})^{-1}=F$ را با R نشان دهیم:

```
> ginv(ginv(F))
     [,1] [,2] [,3]
[1,]   11    7   15
[2,]   10    8   14
[3,]   15   11   12
```

حل دستگاه معادلات خطی

برای حل دستگاه معادلات خطی از تابع solve() استفاده می‌کنیم. شناسه اول تابع ماتریس ضرایب و شناسه دوم بردار ستونی مقادیر معلوم است. اگر شناسه تابع solve() فقط نام یک ماتریس باشد، این تابع وارون ماتریس را محاسبه می‌کند.

مثال: دستگاه معادلات زیر را در نظر بگیرید:

$$\begin{cases} x+2y+3z=14 \\ 2x+y=4 \\ 3x-y+2z=7 \end{cases}$$

برای حل دستگاه فوق به صورت زیر عمل می‌کنیم:

```
> A=matrix(c(1,2,3,2,1,0,3,-1,2),nrow=3,byrow=T)
> b=c(14,4,7)
> solve(A,b)
[1] 1 2 3
```

که مقادیر ۱، ۲ و۳ را به ترتیب برای مجهول های x، y و z می‌دهد.

اگر ماتریس ضرایب دستگاه معادلات به صورت بالا مثلثی یا پایین مثلثی باشد می‌توان به ترتیب از توابع backsolve() و forwardsolve() برای حل دستگاه استفاده کرد.

www.GhadamYar.com

مشتق

برای مشتق گیری از یک عبارت ریاضی از تابع D() استفاده می‌شود. شناسه اول تابع، عبارتی است که می‌خواهیم از آن مشتق بگیریم و شناسه دوم نام متغیری است که می خواهیم نسبت آن مشتق بگیریم. در R برای تعریف یک عبارت ریاضی از تابع expression() استفاده می‌شود. در مثال های زیر نحوه مشتق گیری از عبارات ریاضی نشان داده شده است:

مثال:

مشتق عبارت $2x^3$:

```
> D(expression(2*x^3),"x")
2 * (3 * x^2)
```

مشتق عبارت $\ln(y)$:

```
> D(expression(log(y)),"y")
1/y
```

مشتق عبارت ae^{-bx}:

```
> D(expression(a*exp(-b * x)),"x")
-(a * (exp(-b * x) * b))
```

انتگرال

در R می توان با تابع integrate() انتگرال معین هر تابعی را محاسبه کرد.

مثال: برای محاسبه انتگرال $\int_0^2 2x\,dx$ به صورت زیر عمل کنید:

```
> fx=function(x) 2*x
```

ابتدا با دستور function() تابع fx را بر حسب x به صورت فوق تعریف کنید، حال با استفاده از تابع integrate() انتگرال معین را محاسبه کنید. شناسه های الزامی این تابع، عبارتی است که می‌خواهیم از آن انتگرال بگیریم و سپس حدود پایین و بالای انتگرال.

```
> integrate(fx,0,2)
4 with absolute error < 4.4e-14
```


www.GhadamYar.com

برای محاسبه انتگرال تابع چگالی نرمال استاندارد روی دامنه آن از دستور زیر استفاده کنید:

```
> integrate(dnorm,-Inf,Inf)
1 with absolute error < 9.4e-05
```

dnorm بیانگر تابع چگالی نرمال است که در بخش های بعدی آنرا معرفی می‌کنیم.

پیدا کردن ریشه های چند جمله ای

برای یافتن ریشه های حقیقی و مختلط چند جمله ای $a_0+a_1x+...+a_kx^k$ از تابع polyroot() استفاده می‌شود. شناسه این تابع بردار ضرایب چند جمله ای $(a_0, a_1, ..., a_k)$ است.

مثال:

معادله $x^3+2x-3=0$ با استفاده از تابع فوق به صورت زیر حل می‌شود:

```
> polyroot(c(-3,2,0,1))
[1]  1.0+0.000000i -0.5+1.658312i -0.5-1.658312i
```

ملاحظه می‌کنید که این معادله یک ریشه حقیقی و دو ریشه مختلط دارد.

احتمال و اعداد تصادفی

در R توابع متعددی برای محاسبه چندک ها، مقادیر احتمال توزیع های احتمال مختلف و تولید اعداد تصادفی وجود دارد. نام هر یک از این توابع با یکی از حروف زیر شروع می‌شود.

r : تولید اعداد تصادفی

p : محاسبه مقدار احتمال

d : محاسبه تابع چگالی

q : محاسبه چندک ها

هر یک از این حروف به همراه "کد توزیع" که در جدول ۳ آمده است، تابع مربوط به آن توزیع را می‌سازد.

جدول ۳ جزئیات هر توزیع و مقادیر پیش فرض پارامتر ها را نشان می‌دهد (اگر مقدار پیش فرض برای توزیعی تعریف نشده باشد به معنای آن است که کاربر حتما باید مقدار پارامتر را مشخص کند- مانند درجه آزادی توزیع t).

www.GhadamYar.com

نام توزیـــــع	کد توزیـــــع	پارامتر های لازم	مقادیر پیش فرض
دو جمله ای	binom	size,prob	
دو جمله ای منفی	nbinom	size,prob	
هنـــــدسی	geom	prob[۲]	
فوق هندسی	hyper	m,n,k	
پـــوآسون	pois	lambda	
یـــکنواخت	unif	min,max	0,1
نـــــرمال	norm	mean,sd	0,1
t- استودنت	t	df	
کـــی- دو	chisq	df	
F	f	df1,df2	
نـــمایی	exp	rate	1
گامـــــا	gamma	shape,scale	
بتـــــا	beta	shape1,shape2	
وایبل	weibull	shape,scale	
لگ- نرمال	lnorm	meanlog,sdlog	0,1
لجستیک	logis	location,scale	0,1
رتبه ویلکاکسون	wilcox	m,n	
کوشی	cauchy	location,scale	0,1

جدول ۵- مشخصات توزیع های شناخته شده در R

مثال: برای پیدا کردن نسبت افراد دارای قد کمتر از ۲۰۰ سانتیمتر در جامعه ای که توزیع قد، نرمال با میانگین ۱۷۰ و انحراف معیار ۱۵ است از دستور زیر استفاده می‌شود:

```
> pnorm(200, 170, 15)
[1] 0.9772499
```

در حقیقت این دستور مقدار $P(X\leq 200)$ را محاسبه می‌کند.

عبارت زیر صدک ۹۵-ام توزیع کی- دو با ۵ درجه آزادی را محاسبه می‌کند:

۲- پارامتر prob احتمال موفقیت است.

www.GhadamYar.com

```
> qchisq(.95,5)
[1] 11.07050
```

با توجه به پاسخ عبارت فوق احتمال اینکه یک متغیر تصادفی کی- دو با درجه آزادی ۵ از ۱۱/۰۷۰۵ کوچکتر باشد برابر ۰/۹۵ است.

```
> pchisq(11.0705,5)
[1] 0.95
```

برای محاسبه چندک های معروف توزیع نرمال استاندارد از دستور زیر استفاده کنید:

```
> alpha=c(.025,.05,.1)
> qnorm(1-alpha)
[1] 1.959964 1.644854 1.281552
```

برای تولید ۵ عدد تصادفی از توزیع یکنواخت در بازه صفر و یک از دستور زیر استفاده می‌شود:

```
> runif(5)
[1] 0.4133270 0.9950175 0.6801583 0.8881924
[5] 0.2649808
```

هر بار که از این دستور استفاده می کنید، نتیجه متفاوتی مشاهده می‌کنید چون داده ها به صورت تصادفی تولید می‌شوند.

عبارت زیر مقدار تابع چگالی هندسی با احتمال 0.2 را در نقطه ۵ محاسبه می‌کند:

```
> dgeom(5,.2)
[1] 0.065536
```

تحلیل های آماری با نرم افزار R- محمود مسلمان

www.GhadamYar.com

فصل چهارم: نوشتن توابع در R

R قبل از اینکه یک نرم افزار آماری باشد یک زبان برنامه نویسی آماری است. از اینرو محیط R این امکان را به شما می‌دهد که توابعی را که مکرراً به کار می‌برید، در یک تابع جدید (برنامه) ذخیره کنید تا شما و نیز افراد دیگر به راحتی از آن استفاده کنند. به این ترتیب هر بار که نیاز به اجرای این توابع داشته باشید می‌توانید برنامه خود را اجرا کنید. ساختار عمومی یک برنامه به صورت زیر است:

```
> function( شناسه ها ) {
+ متن برنامه
+ }
```

شناسه های تابع متغیر هایی هستند که در عملیات متن برنامه مورد استفاده قرار می‌گیرند. شناسه ها با یک کاما از هم جدا می‌شوند و قسمت متن برنامه شامل عبارت های تعریف شده در R است که با یک (;) یا یک خط جدید از هم جدا می‌شوند. متن برنامه داخل { } قرار می‌گیرد.

برنامه زیر مربع میانگین یک بردار را محاسبه می‌کند. با تعریف تابع ms می توان هر موقع که مایل به محاسبه مربع میانگین برای متغیر x بودیم به جای نوشتن mean (x) ^2 بنویسیم: ms (x).

```
> ms=function(x){
+ mean(x)^2
+ }

> a=1:10
> ms(a)
[1] 30.25
```

اگر متن برنامه تنها از یک دستور تشکیل شده باشد می‌توان از { } چشم پوشی کرد.

برنامه زیر هیچ شناسه ای ندارد و فقط متن Welcome را روی صفحه نمایش می‌دهد.

```
> wel=function() print("Welcome")
> wel()
[1] "Welcome"
```


www.GhadamYar.com

برای شناسه های یک تابع می توان مقدار پیش فرض تعیین کرد. در این صورت اگر هنگام فراخوانی تابع مقداری برای شناسه تعیین نشود، تابع از مقدار پیش فرض استفاده می کند.

در برنامه زیر برای متغیر x مقدار پیش فرض "R" را در نظر می‌گیریم:

```
> myprog = function(x = "R") {
+ print(paste("I use", x, "for statistical computation."))
+ }

> myprog()
[1] "I use R for statistical computation."

> myprog("SPSS")
[1] "I use SPSS for statistical computation."
```

چند عبارت و تابع مورد نیاز در برنامه نویسی

عبارت if()

شکل کلی این عبارت به صورت زیر است:

if(شرط) نتیجه شرط

یا

if(شرط) نتیجه شرط else ...

شرط در عبارت if یک گزاره منطقی است که عبارت if ابتدا آنرا ارزیابی می‌کند. اگر این گزاره منطقی درست باشد، عبارتی که در نتیجه شرط نوشته شده است، اجرا می شود. در صورتی که عبارت منطقی نادرست باشد و عبارت if به همراه عبارت else نوشته شده باشد، عبارت بعد از else اجرا می شود.

مثال: تابع kind را برای تشخیص زوج یا فرد بودن یک عدد به صورت زیر تعریف می‌کنیم:

```
> kind=function(x){
+ if(x%%2==0) print("x is couple")
+ else print("x is odd")
+ }
```


www.GhadamYar.com

```
> kind(5)
[1] "x is odd"

> kind(1000)
[1] "x is couple"
```

اگر باقیمانده تقسیم x بر ۲ برابر صفر باشد عبارت "x is couple" به معنی زوج بودن x و در غیر اینصورت عبارت "x is odd" به معنی فرد بودن x نمایش داده می‌شود.

عبارت ifelse()

گاهی اوقات مایلیم در صورت درست بودن شرط، عملیاتی انجام گیرد و در صورت درست نبودن شرط عکس آن عملیات صورت پذیرد. تابع ifelse() این امکان را برای شما فراهم می‌کند.

مثال:

```
> x = c(6:-4)
```

اگر بخواهیم جذر عناصر x را محاسبه کنیم، برای عناصری که مقدار منفی دارند پیغام خطا داده می شود:

```
> sqrt(x)
 [1] 2.45 2.24 2.00 1.73 1.41 1.00 0.00  NaN  NaN  NaN  NaN
Warning message:
In sqrt(x) : NaNs produced
```

اگر از عبارت ifelse() به صورت زیر استفاده کنید، پیغام خطا ظاهر نمی شود:

```
> sqrt(ifelse(x >= 0, x, NA))
 [1] 2.45 2.24 2.00 1.73 1.41 1.00 0.00   NA   NA   NA   NA
```

در R مقدار گم شده با NA نمایش داده می‌شود. (Not Available)

www.GhadamYar.com

حلقه ها در R

زبان R شامل چند حلقه از جمله for، while و repeat است که اغلب از آنها در برنامه نویسی استفاده می‌شوند.

مثال: در اینجا حلقه ای با ۵ تکرار برای مقادیر i از یک تا ۵ اجرا می‌شود. مربع هر مقدار نمایش داده می‌شود:

```
> for(i in 1:5) print(i^2)
[1] 1
[1] 4
[1] 9
[1] 16
[1] 25
```

تابع زیر با استفاده از حلقه for مقدار فاکتوریل x را محاسبه می‌کند.

```
> fac1=function(x){
+ f=1
+ if(x<2) return(1)
+ for(i in 2:x){
+ f=f*i }
+ f }
```

برای محاسبه فاکتوریل اعداد ۰ تا ۵ داریم:

```
> sapply(0:5,fac1)
[1]   1   1   2   6  24 120
```

با استفاده از حلقه while نیز یک تابع فاکتوریل تعریف می‌کنیم:

```
> fac2=function(x){
+ f=1
+ t=x
+ while(t>1){
+ f=f*t
+ t=t-1}
+ return(f) }
```


www.GhadamYar.com

```
> sapply(0:5,fac2)
[1]   1   1   2   6  24 120
```

در پایان برای نشان دادن نحوه استفاده از حلقه repeat یک تابع فاکتوریل بر اساس آن تعریف می‌کنیم:

```
> fac3=function(x){
+ f=1
+ t=x
+ repeat{
+ if(t<2) break
+ f=f*t
+ t=t-1 }
+ return(f) }

> sapply(0:5,fac3)
[1]   1   1   2   6  24 120
```

تحلیل های آماری با نرم افزار R- محمود مسلمان

www.GhadamYar.com

فصل پنجم: آمار توصیفی و نمودار ها

اولین قدم در تجزیه و تحلیل داده ها به دست آوردن خلاصه ای از داده ها می‌باشد. خلاصه کردن و توضیح خصوصیات مهم مجموعه داده ها را معمولاً آمار توصیفی می‌نامند. این مبحث شامل فشرده کردن داده ها در قالب جداول، نمایش آنها به صورت نموداری و محاسبه شاخصهای عددی گرایش به مرکز و شاخصهای پراکندگی است. قبل از شروع به تجزیه و تحلیل داده ها باید انواع داده ها را بشناسید تا روش مناسب برای توصیف هر نوع داده را به کار برید.

انواع داده ها:

- **داده های کمی**

گسسته : متغیر ها فقط مقادیر صحیح را می‌گیرند (۰، ۱، ۲ و ...). مانند: تعداد فرزندان، ترم تحصیلی

پیوسته: متغیر ها هر عدد حقیقی را می‌گیرند (اغلب در یک بازه معین). مانند: وزن و قد و سن افراد

- **داده های کیفی (غیر عددی- رسته ای)**

اسمی: مقادیر رسته ای غیر رتبه ای مانند: گروه خونی، رنگ چشم

رتبه ای: مقادیر رسته‌ای رتبه‌ای مانند سطح سواد،

تحلیل های آماری با نرم افزار R- محمود مسلمان

www.GhadamYar.com

خلاصه کردن داده های کمی:

در جدول زیر توابع مهم برای خلاصه کردن داده ها را مشاهده می‌کنید.

تابع	شرح
min(x)	کوچکترین مقدار x
max(x)	بزرگترین مقدار x
range(x)	فاصله کوچکترین و بزرگترین مقدار x
IQR(x)	برد میان چارکی مقادیر x
mean(x)	میانگین مقادیر x
median(x)	میانه مقادیر x
var(x)	واریانس مقادیر x
sd(x)	انحراف معیار مقادیر x
cor(x,y)	همبستگی بین x و y
quantile(x,p)	p امین چندک x
cov(x,y)	کوواریانس بین x و y

جدول ۶- توابع مهم برای خلاصه کردن داده‌ها

دستورات زیر نحوه محاسبه میانگین، انحراف معیار، واریانس و میانه را برای ۵۰ داده تصادفی از توزیع نرمال استاندارد نشان می‌دهد.

```
> x = rnorm(50)
> mean(x)
[1] 0.03301363
> sd(x)
[1] 1.069454
> var(x)
[1] 1.143731
> median(x)
[1] -0.08682795
```

اگر شما دستورات فوق را اجرا کنید دقیقاً این نتایج را مشاهده نمی‌کنید، چون داده های تصادفی شما متفاوت است.

اگر در داده ها مقدار گمشده موجود باشد، نمی‌توان خلاصه‌های آماری را برای آن محاسبه کرد. برای حل این مشکل می‌توان با شناسه na.rm=T مقادیر گمشده را نادیده گرفت و آماره مورد نظر را محاسبه کرد.

www.GhadamYar.com

```
> y=scan()
1: 1 5 2 4 NA
6:
Read 5 items
> mean(y)
[1] NA
> mean(y,na.rm=T)
[1] 3
```

با استفاده از تابع quantile() می‌توان چندکهای مهم را بدست آورد.

```
> quantile(x)
        0%        25%        50%        75%       100%
-2.60741896 -0.54495849 -0.08682795 0.70018536 2.98872414
```

همانطور که ملاحظه می‌کنید این تابع به طور پیش فرض مقادیر ماکزیمم، مینیمم و چارک های اول و دوم و سوم را محاسبه می کند. از تابع quantile() می توان برای محاسبه هر چندک دلخواه استفاده کرد. مثلاً برای محاسبه دهک‌های متغیر x از دستور زیر استفاده کنید:

```
> pvec = seq(0,1,0.1)
> pvec
[1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
> quantile(x,pvec)
         0%        10%        20%        30%        40%
-2.60741896 -1.07746896 -0.70409272 -0.46507213 -0.29976610
        50%        60%        70%        80%        90%
-0.08682795  0.19436950  0.49060129  0.90165137  1.31873981
       100%
 2.98872414
```

تفاضل چارک سوم و چارک اول، برد میان چارکی نامیده می‌شود و توسط تابع IQR() قابل محاسبه است.

```
> IQR(x)
[1] 1.245144
```


www.GhadamYar.com

تابع summary() یک نمایش خلاصه از متغیرهای عددی ارائه می‌کند. برای مثال از مجموعه داده Loblolly مربوط به رشد درختان کاج استفاده می‌کنیم.

```
> attach(Loblolly)
> names(Loblolly)
[1] "height" "age"    "Seed"
> summary(height)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   3.46   10.47   34.00   32.36   51.36   64.10
```

ملاحظه می‌کنید که نتیجه اجرای تابع summary() برای متغیر عددی height مقادیر مینیمم، چارک اول، میانه، میانگین، چارک سوم و ماکزیمم است. تابع summary() را همچنین می‌توان بر روی یک چارچوب داده نیز اعمال کرد.

```
> summary(Loblolly)
     height           age             Seed
 Min.   : 3.46   Min.   : 3.0   329    : 6
 1st Qu.:10.47   1st Qu.: 5.0   327    : 6
 Median :34.00   Median :12.5   325    : 6
 Mean   :32.36   Mean   :13.0   307    : 6
 3rd Qu.:51.36   3rd Qu.:20.0   331    : 6
 Max.   :64.10   Max.   :25.0   311    : 6
                                (Other):48
```

ملاحظه می‌کنید که برای متغیر سن (age) مقادیر ذکر شده فوق مشخص شد. متغیر seed مربوط به نوع بذر است و از آنجایی که یک متغیر اسمی است، تنها فراوانی مربوط به هر سطح آن مشخص می‌شود.

www.GhadamYar.com

نمودار شاخه و برگ

روشهای مختلفی برای نمایش گرافیکی داده‌ها وجود دارد. اگر مجموعه داده‌ها نسبتاً کوچک باشد نمودار شاخه و برگ برای مشاهده شکل توزیع، بسیار کارآمد است. فرض کنید اعداد زیر امتیازات کسب شده توسط هر بازیکن در یک تیم بسکتبال باشد.

۵ ۲ ۸ ۴ ۱۴ ۳۱ ۲۸ ۶ ۰ ۰ ۰ ۲ ۱۳ ۴ ۱۲ ۱۴ ۲۳ ۱۶ ۳ ۲

رسم نمودار شاخه برگ برای این داده‌ها ساده است، داده را با تابع scan () به صورت زیر وارد کنید:

```
> scores=scan()
1: 2 3 16 23 14 12 4 13 2 0 0 0 6 28 31 14 4 8 2 5
21:
Read 20 items
```

با تابع stem () نمودار شاخه و برگ را رسم کنید:

```
> stem(scores)

  The decimal point is 1 digit(s) to the right of the |

  0 | 000222344568
  1 | 23446
  2 | 38
  3 | 1
```

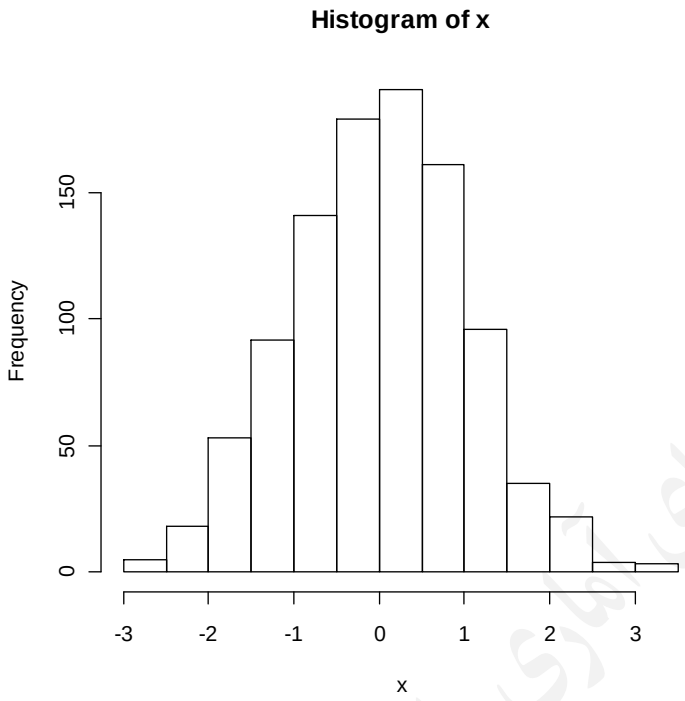
عدد سمت چپ خط را به عنوان شاخه و عدد سمت راست را به عنوان برگ در نظر می‌گیریم.

هیستوگرام (بافت نگار)

با رسم هیستوگرام یک متغیر، می‌توان شکل تقریبی توزیع آن را مشاهده کرد. در هیستوگرام فراوانی مشاهدات هر رده مشخص می‌شود.

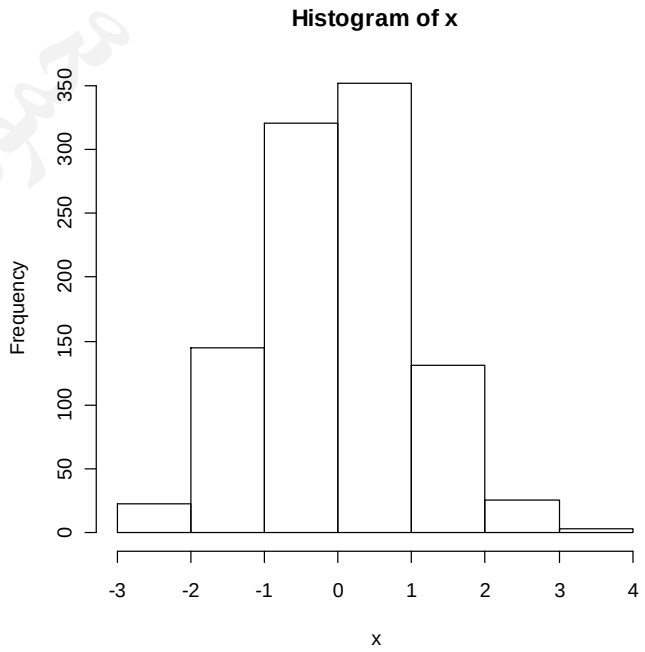
```
> x=rnorm(1000)
> hist(x)
```


www.GhadamYar.com



این تابع تعداد دسته ها را به طور خود کار انتخاب می‌کند. این انتخاب به گو نه‌ای است که ضمن حفظ اطلاعات جزئی، هموار ترین نمودار ممکن نیز رسم می‌شود. اگر مایل به انتخاب تعداد دسته ها هستید، می‌توانید از شناسه `nclass` استفاده کنید. در مثال زیر تعداد دسته ها ۷ تعیین شده است.

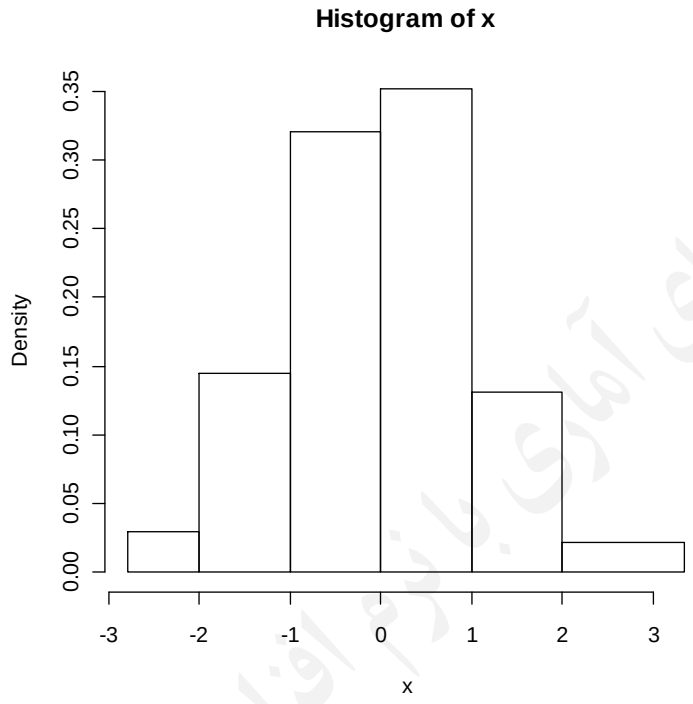
```
> hist(x,nclass=7)
```



www.GhadamYar.com

برای مشخص کردن مرز رده ها در این تابع می‌توان از شناسه breaks استفاده کرد. این شناسه را برابر برداری قرار دهید که مقادیر آن مرز دسته ها و طول آن برابر دسته ها به اضافه ۱ باشد.

```
> hist(x,breaks=c(min(x),-2,-1,0,1,2,max(x)))
```



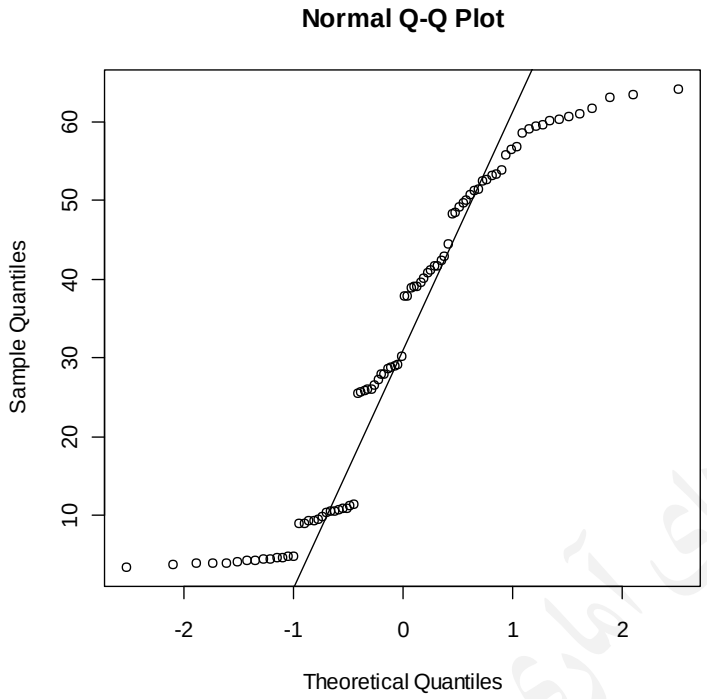
نمودار (QQ-Plot) یا چندک – چندک

در نمودار Q-Q Plot، چندک های یک توزیع احتمال خاص در برابر چندک های نمونه رسم می‌شود. با استفاده از این نمودار می‌توان توزیع دو مجموعه را با هم مقایسه کرد و همچنین تعیین کرد که یک مجموعه داده از توزیع خاصی پیروی می‌کند یا نه؟ اگر توزیع فرض شده برای داده ها نرمال باشد، برای رسم نمودار چندک- چندک یا نمودار احتمال نرمال از تابع qqnorm() استفاده کنید.

به عنوان مثال برای بررسی نرمال بودن متغیر height از مجموعه داده Loblolly به صورت زیر عمل کنید.

```
> data(Loblolly )
> attach(Loblolly )
> qqnorm(height)
> qqline(height)
```


www.GhadamYar.com



تابع qqline() یک خط بر اساس توزیع نرمال برازش می‌دهد. هر چه نقاط نمودار به این خط نزدیک‌تر باشند، توزیع داده ها به نرمال نزدیک‌تر است.

نمودار جعبه ای

نمودار جعبه ای نشان دهنده چند خصیصه مهم از توزیع داده‌های یک بعدی است. میانه، دامنه، برد میان چارکی، پراکندگی، چاولگی و نقاط دور افتاده را می‌توان در یک نمودار جعبه‌ای مشاهده کرد. برای ساختن یک نمودار جعبه ای از تابع boxplot() استفاده می‌شود. به عنوان مثال عبارت زیر نمودار جعبه ای را برای height رسم می کند.

```
> boxplot(height)
```

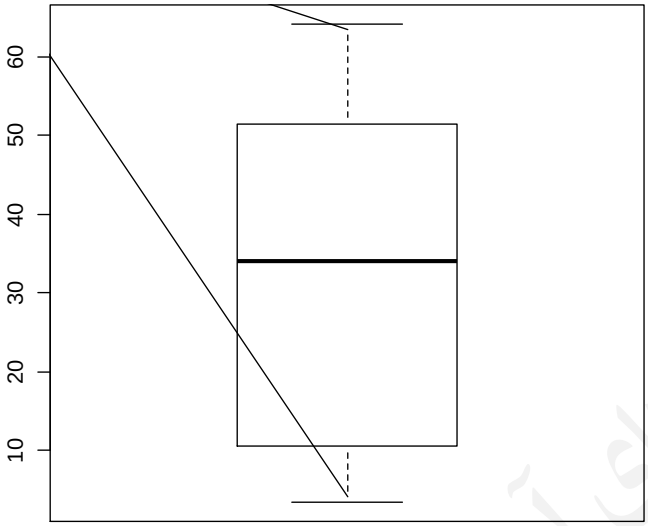
خط افقی داخل جعبه، میانه داده‌ها و ارتفاع جعبه دامنه میان چارکی (IQR) است. خط عمودی نقطه چین را "ویسکر" می‌نامند. نقطه بالایی و پایینی ویسکرها به ترتیب به صورت زیر تعریف می‌شوند.

$\min\{\max(height), Q_3+1.5IQR\}$

$\max\{\min(height), Q_1-1.5IQR\}$

نقاط خارج از ویسکرها، نقاط دور افتاده هستند.

www.GhadamYar.com



خلاصه کردن داده‌های رسته ای

اغلب داده‌های رسته‌ای را در قالب جداول نمایش می‌دهند. این داده ها را می‌توان با نمودار میله‌ای یا دایره‌ای نیز نمایش داد.

استفاده از جداول

با استفاده از تابع `table()` می‌توان داده‌های رسته را به صورت جدولی مشاهده کرد. فرم ساده این دستور به صورت `table(x)` است که در آن `x` متغیر رسته‌ای است.

مثال: در یک تحقیق که در مورد استعمال سیگار از افراد سؤال شده است، داده‌های زیر بدست آمده است:

بله، خیر، خیر، بله، بله

داده ها را با دستور `c()` وارد و با دستور `table()` خلاصه می‌کنیم:

```
> x=c("Yes","No","No","Yes","Yes")
> table(x)
x
 No Yes
  2   3
```

در واقع دستور `table()` فراوانی هر رسته را مشخص می‌کند.

www.GhadamYar.com

نمودار میله‌ای

ارتفاع میله‌ها در یک نمودار میله‌ای، مقادیر متناظر با فراوانی رسته‌ها را مشخص می کند، بطوریکه بلندترین میله نشانگر رسته‌ای است که بیشترین مقدار (فراوانی) را دارد. نمودار میله‌ای را می توان بر اساس فراوانی مطلق و نیز فراوانی نسبی (درصد) رسم کرد.

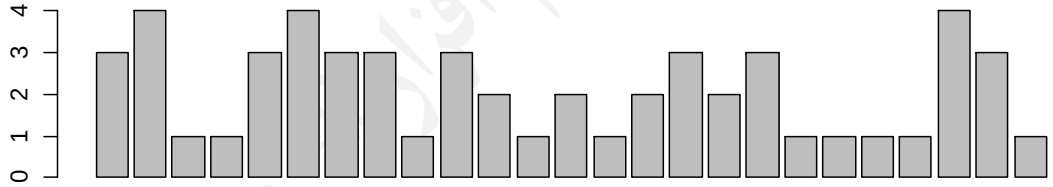
فرض کنید در مورد نحوه آمدن به دانشگاه از ۲۵ دانشجو سؤال کرده‌ایم. رسته ها عبارتند از ۱: اتوبوس، ۲: تاکسی، ۳: اتومبیل شخصی ۴: پیاده. داده‌های زیر بدست آمده است:

۳،۴،۱،۱،۳،۴،۳،۳،۱،۳،۲،۱،۲،۱،۲،۳،۲،۳،۱،۱،۱،۱،۴،۳،۱

نمودار میله‌ای مربوط به فراوانی مطلق و فروانی نسبی داده ها را رسم می کنیم:

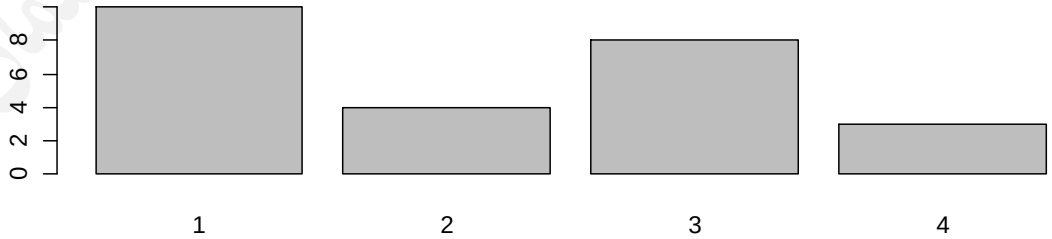
```
> uni=scan()
1:  3 4 1 1 3 4 3 3 1 3 2 1 2 1 2 3 2 3 1 1 1 1 4 3 1
26:
Read 25 items
> barplot(uni)
```

این فرم صحیح نیست



این نمودار مناسب نیست، ما ۴ تا رده داشتیم نه ۲۵ تا!

```
> tuni=table(uni)
> barplot(tuni)
```

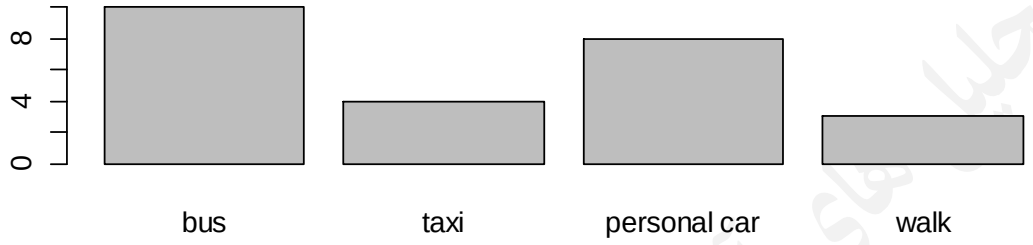


نمودار میله ای - فراوانی مطلق

www.GhadamYar.com

برای اینکه به جای شماره رسته ها نام هر رسته را روی نمودار مشاهده کنیم با تابع names () اسامی رسته ها را مشخص می‌کنیم.

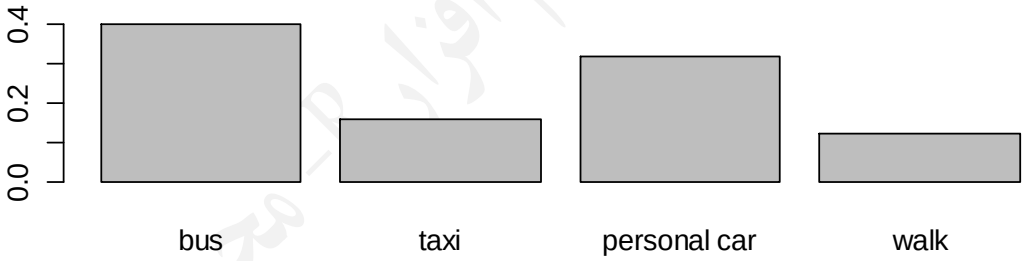
```
> names(tuni)=c("bus","car","personal car","walk")
> barplot(tuni)
```



نمودار میله‌ای با نمایش اسامی رسته ها

برای رسم نمودار میله‌ای بر حسب فراوانی نسبی (درصد) داده ها از دستور زیر استفاده می‌کنیم:

```
> barplot(tuni/length(uni))
```



نمودار میله‌ای – فراوانی نسبی

دستور زیر جدول فراوانی نسبی داده‌ها را ایجاد می‌کند:

```
> tuni/length(uni)
       bus      taxi personal car      walk
      0.40      0.16         0.32      0.12
```

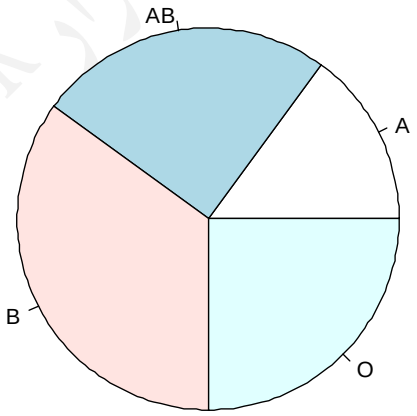

www.GhadamYar.com

نمودار دایره‌ای

داده های رسته‌ای را می‌توان با نمودار دایره‌ای نیز نشان داد. برای رسم یک نمودار دایره‌ای از تابع `pie()` استفاده می‌شود.

مثال: نمونه ای ۲۰ تایی از گروه خونی مراجعه کنندگان به یک درمانگاه در دست داریم. با دستورات زیر نمودار دایره ای گروه خونی این افراد رسم می‌شود.

```
> blood=sample(c("A" , "B" , "AB" , "O"),20,replace=T)
> blood
 [1] "AB" "B"  "B"  "B"  "A"  "A"  "O"  "O"  "B"  "B"  "O"
[12] "B"  "AB" "O"  "B"  "A"  "AB" "AB" "O"  "AB"
> tblood=table(blood)
> tblood
blood
 A AB  B  O
 3  5  7  5
> pie(tblood)
```



در این مثال داده های خود را به صورت یک نمونه تصادفی در نظر گرفتیم. برای تولید نمونه تصادفی از تابع `sample()` استفاده می‌شود. پارامتر اول این تابع، جامعه‌ای است که می‌خواهیم از آن نمونه بگیریم که در این مثال بردار اسامی ۴ نوع گروه خونی موجود است؛ پارامتر دوم حجم نمونه مورد نیاز است. شناسه `replace` را برابر `True` قرار می‌دهیم تا نمونه گیری تصادفی ساده با جایگذاری انجام شود. مانند دستور رسم نمودار میله‌ای در اینجا نیز نمودار را برای جدول فراوانی رسم می‌کنیم.

www.GhadamYar.com

تحلیل داده‌های رسته ای دو متغیره

تابع `table()` داده‌های دو متغیره را مانند داده‌های یک متغیره خلاصه می‌کند. فرض کنید یک تحقیق در مورد استعمال سیگار و میزان مطالعه دانشجویان انجام و داده های زیر به دست آمده است.

دانشجو	استعمال سیگار	میزان مطالعه
۱	بله	کمتر از ۵ ساعت
۲	خیر	۱۰-۵ ساعت
۳	خیر	۱۰-۵ ساعت
۴	بله	بیشتر از ۱۰ ساعت
۵	خیر	بیشتر از ۱۰ ساعت
۶	بله	کمتر از ۵ ساعت
۷	بله	۱۰-۵ ساعت
۸	بله	کمتر از ۵ ساعت
۹	خیر	بیشتر از ۱۰ ساعت
۱۰	بله	۱۰-۵ ساعت

می توان این داده‌ها را با دو بردار وارد R کرد و سپس از دستور `table()` استفاده کرد.

```
> smoke=c("Y","N","N","Y","N","Y","Y","Y","N","Y")
> study=c(1,2,2,3,3,1,2,1,3,2)
> table(smoke,study)
     study
smoke 1 2 3
    N 0 2 2
    Y 3 2 1
```

امکان وجود برخی روابط در جدول مشاهده می‌شود.

www.GhadamYar.com

برای مشاهده فراوانی‌های نسبی از دستور prop.table() استفاده می‌شود، البته با فراخوانی جدول فراوانی که قبلاً آن را ذخیره کرده‌ایم و یک عدد برای مشخص کردن فراوانی‌های سطری (۱) یا فراوانی‌های ستونی (۲) که مقدار پیش فرض آن فراوانی کلی را مشخص می‌کند.

```
> tmp=table(smoke,study)
```
ذخیره کردن جدول

```
> prop.table(tmp,1)
```
جمع سطرها برابر ۱ می شود

```
     study
smoke          1         2         3
    N 0.0000000 0.5000000 0.5000000
    Y 0.5000000 0.3333333 0.1666667
```

```
> prop.table(tmp,2)
```
جمع ستون ها برابر ۱ می شود

```
     study
smoke          1         2         3
    N 0.0000000 0.5000000 0.6666667
    Y 1.0000000 0.5000000 0.3333333
```

```
> prop.table(tmp)
```
جمع همه اعداد برابر ۱ می شود

```
     study
smoke   1   2   3
    N 0.0 0.2 0.2
    Y 0.3 0.2 0.1
```

رسم داده‌های جدول

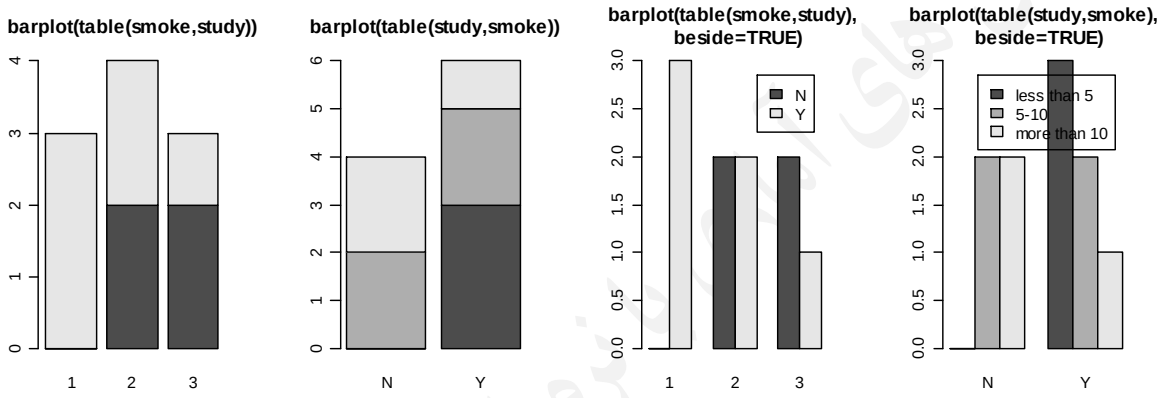
ممکن است بخواهید داده‌های خلاصه شده در جداول را به صورت نموداری مشاهده کنید. در مثال قبل می‌توانید متغیر میزان مطالعه را برای هر بله و خیر، یا متغیر استعمال سیگار را برای هر سطح از میزان مطالعه رسم کرد. در هر دو مورد می‌توان از barplot استفاده کرد.

```
> barplot(table(smoke,study))
> barplot(table(study,smoke))
> smoke=factor(smoke)
```


www.GhadamYar.com

```
> barplot(table(smoke,study),beside=TRUE,legend.text=T)

> barplot(table(study,smoke),
+ beside=TRUE,
+ ,legend.text=c("less than 5","5-10","more than 10"))
```



۴نوع نمودار میله ای برای داده های یکسان

اهمیت ترتیب متغیر ها در ایجاد جدول در این نمودارها مشخص می شود. در حقیقت تابع barplot هر سطر از داده‌های جدول را رسم می‌کند، که این کار هم به صورت انباشته (حالت پیش فرض) و هم به صورت کنار هم (با اضافه کردن شناسه beside=TRUE) انجام می‌گیرد. شناسه legend.text برای اضافه کردن راهنما به نمودار استفاده می‌شود. می توانید نام‌های هر رنگ را تغییر دهید، اما اگر یک متغیر factor با اسامی سطرها در تابع table() باشد حالت پیش فرض legend.text=T ساده تر است (مانند نمودار سوم).

داده‌های دو متغیره: کمی و رسته‌ای

فرض کنید داده‌های کمی برای چندین رسته در اختیار دارید. مثلاً در یک آزمایش دارویی، داده‌هایی را برای گروه تیمار و داده‌هایی برای گروه شاهد در اختیار دارید.

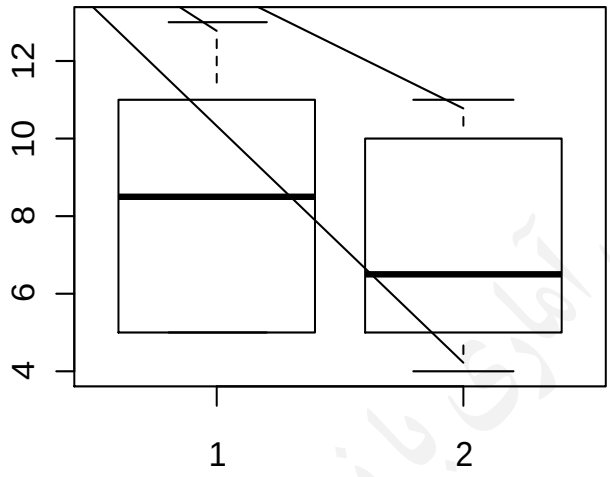
گروه تیمار: ۵، ۵، ۵، ۱۳، ۷، ۱۱، ۱۱، ۹، ۸، ۹

گروه شاهد: ۱۱، ۸، ۴، ۵، ۹، ۵، ۱۰، ۵، ۴، ۱۰

www.GhadamYar.com

می‌توان داده‌ها را به صورت جداگانه خلاصه کرد و نیز با هم مقایسه کرد، اما چگونه می‌توان داده‌ها را با هم مشاهده کرد؟ یک نمودار جعبه‌ای موازی (دو طرفه) برای این منظور مناسب است.

```
> x=c(5,5,5,13,7,11,11,9,8,9)
> y=c(11,8,4,5,9,5,10,5,4,10)
> boxplot(x,y)
```



با مقایسه دو نمودار می بینیم که متغیر y (گروه شاهد، ۲) کمتر از متغیر x (گروه تیمار، ۱) به نظر می‌رسد.

البته ممکن است شما داده‌ها را به صورت یک دسته اعداد و یک متغیر که رسته هر عدد را مشخص می‌کند، به صورت زیر دریافت کنید:

مقدار:	۵	۵	۵	۱۳	۷	۱۱	۱۱	۹	۸	۹	۱۱	۸	۴	۵	۹	۵	۱۰	۵	۴	۱۰
گروه:	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۲	۲	۲	۲	۲	۲	۲	۲	۲	۲

در این حالت برای ایجاد یک نمودار جعبه ای موازی با استفاده از مدلبندی به صورت زیر عمل می کنیم:

```
> amount=scan()
1: 5 5 5 13 7 11 11 9 8 9 11 8 4 5 9 5 10 5 4 10
21:
Read 20 items
> group=scan()
1: 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
21:
Read 20 items
> boxplot(amount~group)
```


www.GhadamYar.com

تعداد زیادی از پژوهشها مشتمل بر دو نمونه یا بیشتر است، اما گاهی ممکن است محقق از یک نمونه واحد از مشاهدات برای استنباط در مورد فقط یک جامعه استفاده کند.

آزمون میانگین جامعه (یک نمونه ای)

در R برای انجام آزمون برابری میانگین یک جامعه با یک مقدار مفروض به روش کلاسیک از تابع `t.test()` و برای انجام آزمون ناپارامتری از تابع `wilcox.test()` استفاده می‌شود. لازم است قبل از انجام این آزمون‌ها با تحلیل گرافیکی داده‌ها (نمودار جعبه ای، هیستوگرام، `Q-Q plot` و...) نوع آزمون آماری مناسب انتخاب شود.

آزمون کلاسیک (t-test):

`t-test` یک نمونه‌ای، آزمون می‌کند که آیا میانگین یک جامعه برابر مقدار مفروضی است یا نه؟ به عبارت دیگر فرض صفر در این آزمون $H_0: \mu=\mu_0$ و فرض مقابل $H_1: \mu\neq\mu_0$ می باشد.

فرض هایی که برای استفاده از این آزمون وجود دارد:

- مشاهدات دارای توزیع نرمال باشند.
- مشاهدات مستقل از هم باشند.

برای انجام آزمون t-استیودنت از تابع `t.test()` استفاده می‌شود، شناسه‌های اختیاری و پر کاربرد این تابع عبارتند از:

- شناسه `conf.level` : مقدار این شناسه سطح اطمینان آزمون است که به طور قراردادی برابر ۹۵٪ می‌باشد.
- شناسه `alternative` : به وسیله این شناسه نوع فرض مقابل آزمون تعیین می‌شود. مقادیر این شناسه عبارتند از:

۱- `"two.sided"` برای فرض مقابل دوطرفه $\mu \neq \mu_0$.

۲- `"greater"` برای فرض مقابل یکطرفه $\mu > \mu_0$.

۳- `"less"` برای فرض مقابل یکطرفه $\mu < \mu_0$.

مقدار پیش فرض این شناسه `"two.sided"` است.

www.GhadamYar.com

- شناسه mu : مقدار این شناسه همان مقدار مفروض (μ_0) در فرض $H_0: \mu=\mu_0$ است که بطور پیش فرض برابر صفر است.

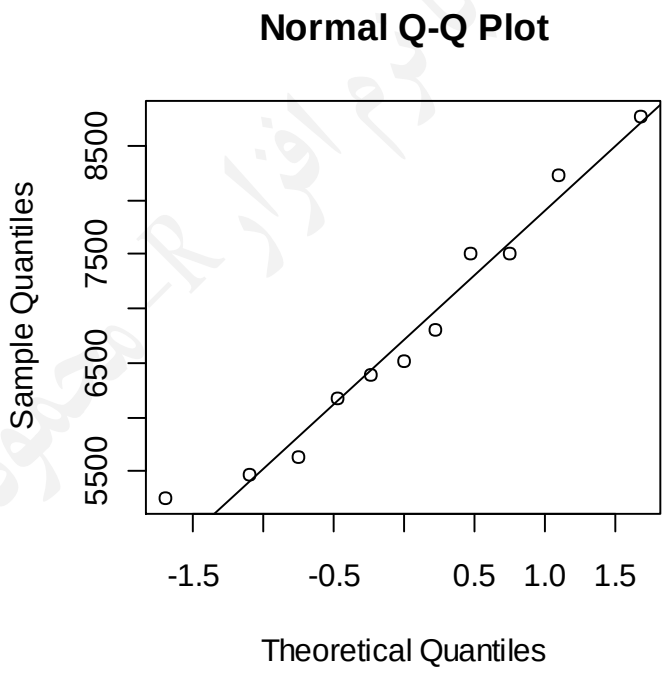
مثال: داده های زیر مقدار انرژی دریافتی روزانه ۱۱ مرد را بر حسب کیلو ژول نشان می‌دهد.

```
> energy = c(5260,5470,5640,6180,6390,6515,6805,7515,7515,8230,8770)
```

مایلیم بررسی کنیم که آیا انرژی دریافتی روزانه مردان با مقدار توصیه شده توسط پزشکان یعنی ۷۷۲۵ کیلو ژول برابر است یا خیر؟

قبل از انجام آزمون با رسم Q-Q plot نرمال بودن داده ها را بررسی می‌کنیم.

```
> qqnorm(energy)
> qqline(energy)
```



مشاهده می‌شود که داده‌ها تقریباً از توزیع نرمال پیروی می‌کنند، پس می‌توان از آزمون t- استیودنت استفاده کرد. با استفاده از تابع t.test() داریم:

www.GhadamYar.com

```
> t.test(energy,mu=7725)

        One Sample t-test

data:  energy
t = -2.8208, df = 10, p-value = 0.01814
alternative hypothesis: true mean is not equal to 7725
95 percent confidence interval:
 5986.348 7520.925
sample estimates:
mean of x
 6753.636
```

مقدار p-value در خروجی تابع محاسبه می‌شود که در این مثال کوچکتر از سطح معنی داری آزمون یعنی ۰/۰۵ است، بنابراین با اطمینان ۹۵٪ فرض صفر $H_0: \mu=۷۷۲۵$ رد می‌شود.

فاصله اطمینان ۹۵٪ مربوط به میانگین جامعه نیز در خروجی دستور آمده است که برابر (۵۹۸۶/۳۴۸, ۷۵۲۰/۹۲۵) است. اگر مایل باشیم فاصله اطمینان ۹۰٪ را بدست آوریم به صورت زیر عمل می‌کنیم:

```
> t.test(energy,mu=7725,conf.level=0.9)$conf.int
[1] 6129.492 7377.781
attr(,"conf.level")
[1] 0.9
```

که فاصله اطمینان ۹۰٪ برابر (6129.492 ,7377.781) بدست می‌آید.

آزمون رتبه ای ویلکاکسون:

آزمون رتبه ای ویلکاکسون، معادل ناپارامتری آزمون t می‌باشد. برای اینکه بدانیم میانه یک جامعه برابر یک مقدار ویژه‌ای است یا نه، از این آزمون استفاده می‌کنیم. پس در مواردی که فرض نرمال بودن داده ها برقرار نباشد می‌توان از این آزمون استفاده کرد.

مثال: در یک تحقیق در مورد مدت زمان استفاده افراد از تلفن همراه، داده‌های زیر به دست آمده است:

۱۲/۸	۳/۵	۲/۹	۹/۴	۸/۷	۰/۷	۰/۲	۲/۸	۱/۹	۲/۸	۳/۱	۱۵/۸

ابتدا با نمودار شاخه و برگ توزیع شکل توزیع داده‌ها را بررسی می‌کنیم:

www.GhadamYar.com

```
> x=c(12.8,3.5,2.9,9.4,8.7,0.7,0.2,2.8,1.9,2.8,3.1,15.8)
> stem(x)

  The decimal point is 1 digit(s) to the right of the |

  0 | 01233334
  0 | 99
  1 | 3
  1 | 6
```

توزیع داده‌ها چاوله به نظر می‌رسد، پس t-test قابل استفاده نیست. در عوض آزمونی برای میانه انجام می‌شود. فرض صفر اینکه میانه برابر ۵ و فرض مقابل اینکه میانه بزرگتر از ۵ باشد. برای انجام این آزمون با استفاده از تابع wilcox.test() به صورت زیر عمل می‌کنیم:

```
> wilcox.test(tel,mu=5,alt="greater")

        Wilcoxon signed rank test with continuity correction

data:  tel
V = 39, p-value = 0.5156
alternative hypothesis: true location is greater than 5

Warning message:
In wilcox.test.default(tel, mu = 5, alt = "greater") :
  cannot compute exact p-value with ties
```

با توجه به اینکه p-value خیلی کوچک نیست، H_0 رد نمی شود.

آزمون برابری واریانس‌های دو جامعه

برای آزمون برابری واریانس‌های دو جامعه از تابع var.test() استفاده می‌شود. شناسه‌های این تابع بردار مشاهدات نمونه از جامعه اول و بردار مشاهدات نمونه از جامعه دوم هستند. مانند تابع t.test() شناسه‌های اختیاری conf.level و alternative نیز در این تابع کاربرد دارد. از این تابع در مثال های بعدی استفاده خواهد شد.

www.GhadamYar.com

آزمون برابری میانگین‌های دو جامعه:

زمانی که می‌خواهیم میانگین‌های دو جامعه را مقایسه کنیم از آزمون‌های دو نمونه‌ای استفاده می‌کنیم که در این آزمون‌ها فرض صفر به صورت $H_0: \mu_1=\mu_2$ می باشد که μ_1 و μ_2 میانگین‌های دو جامعه مختلف هستند.

برای انجام آزمون t- استیودنت دو نمونه‌ای نیز همانند حالت یک نمونه‌ای از تابع `t.test()` استفاده می‌شود. ولی در این حالت شناسه دیگری به نام `var.equal` بکار می‌رود. مقدار قراردادی این شناسه F است یعنی R به طور قراردادی واریانس‌های دو جامعه را متفاوت فرض می‌کند. در اجرای آزمون t دو نمونه‌ای مقدار شناسه mu را برابر صفر است (اختلاف دو میانگین تحت فرض صفر). اگر فرض صفر آزمون $H_0: \mu_1-\mu_2=k$ باشد، شناسه mu برابر k خواهد بود. برای انجام آزمون ناپارامتری ویلکاکسون نیز از تابع `wilcox.test()` همانند حالت یک نمونه‌ای استفاده می‌شود.

مثال: مدت زمان طی شده تا بهبود بیماران پس از مصرف یک دارو (به روز) ثبت شده است. یک گروه شاهد نیز برای این آزمایش در نظر گرفته شده است. داده ها به صورت زیر است:

```
> drug=c(15,10,13,7,9,8,21,9,14,8)
> placebo=c(15,14,12,8,14,7,16,10,15,12)
```

ابتدا فرض برابری واریانس‌ها را آزمون می‌کنیم:

```
> var.test(drug,placebo)

        F test to compare two variances

data:  drug and placebo
F = 1.9791, num df = 9, denom df = 9, p-value = 0.3237
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.491579 7.967821
sample estimates:
ratio of variances
          1.979094
```

p-value کوچک نیست، پس فرض برابر بودن واریانس ها را می‌پذیریم.

برای آزمون اینکه اثر بخشی دارو سریعتر از دارونماست به این صورت عمل کنید:

www.GhadamYar.com

```
> t.test(drug,placebo,alt="less",var.equal=T)

        Two Sample t-test

data:  drug and placebo
t = -0.5331, df = 18, p-value = 0.3002
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
     -Inf 2.027436
sample estimates:
mean of x mean of y
     11.4      12.3
```

بنابر این فرض صفر را می‌پذیریم، یعنی سرعت اثر بخشی دارو و دارو نما برابر است!

آزمون t- استیودنت برای داده های جفتی (paired t-test)

اگر اندازه‌گیری بر روی واحدهای آزمایشی یکسان دو بار انجام شود از آزمون‌های جفتی یا زوج شده استفاده می‌کنیم، مثلاً برای بررسی تأثیر دارو بر روی فشار خون افراد، قبل و بعد از مصرف آن و مسائلی از این قبیل، از آزمون t زوج شده استفاده می‌کنیم. برای اجرای این نوع آزمون باید شناسه paired در تابع t.test برابر T قرار گیرد.

مثال: برای ارتقای بی طرفی در تصحیح اوراق امتحانی دانشجویان، هر برگه امتحانی توسط دو مصحح مختلف تصحیح می‌شود. داده‌های زیر نمره‌های ۱۰ دانشجو که توسط هر دو مصحح ثبت شده را نشان می‌دهد، آیا تفاوتی بین دو مصحح وجود دارد؟

مصحح ۱:	۳	۰	۵	۲	۵	۵	۴	۴	۵	۵
مصحح ۲:	۲	۱	۴	۱	۴	۳	۳	۲	۳	۵

با توجه به داده‌ها وجود اختلاف بین دو مصحح واضح است. اجرای یک آزمون جفتی حقیقت را روشن می‌کند. با قبول فرض نرمال بودن داده‌ها داریم:

www.GhadamYar.com

```
> x=c(3,0,5,2,5,5,5,4,4,5)
> y=c(2,1,4,1,4,3,3,2,3,5)
> t.test(x,y,paired=T)

        Paired t-test

data:  x and y
t = 3.3541, df = 9, p-value = 0.008468
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.3255550 1.6744450
sample estimates:
mean of the differences
                      1
```

که به رد فرض صفر منجر می‌شود، یعنی با دو مصحح کاملاً متفاوت مواجهیم!

آزمون جفتی ویلکاکسون برای مقایسه میانگین‌های دو جامعه

برای انجام این آزمون نیز کافی است شناسه parired را در تابع wilcox.test برابر T قرار دهید.

```
> wilcox.test(x,y,paired=T)

        Wilcoxon signed rank test with continuity correction

data:  x and y
V = 41.5, p-value = 0.02316
alternative hypothesis: true location shift is not equal to 0

Warning messages:
1: In wilcox.test.default(x, y, paired = T) :
  cannot compute exact p-value with ties
2: In wilcox.test.default(x, y, paired = T) :
  cannot compute exact p-value with zeroes
```

ملاحظه می کنید که این آزمون نیز معنی دار است (فرض صفر رد می شود).

www.GhadamYar.com

آزمون صفر بودن ضریب همبستگی

در فصل‌های قبل تابع `cor()` برای برآورد نقطه‌ای ضریب همبستگی را معرفی کردیم. در این قسمت تابع `cor.test()` برای آزمون عدم همبستگی دو متغیر معرفی می‌شود. این تابع نیز دارای شناسه `alternative` است که مقادیر آن قبلاً معرفی شد و همچنین شناسه `method` که برای انتخاب نوع ضریب همبستگی استفاده می‌شود. مقادیر ممکن این شناسه عبارتند از:

- `"pearson"` یا `"p"` : ضریب همبستگی پیرسون
- `"kendall"` یا `"k"` : ضریب همبستگی کندال
- `"spearman"` یا `"s"` : ضریب همبستگی اسپیرمن

مقدار پیش فرض این شناسه `"pearson"` است، یعنی اگر ما نوع ضریب همبستگی را مشخص نکنیم، نرم افزار R به طور خود کار ضریب همبستگی پیرسون را در نظر می‌گیرد.

نرمال بودن توزیع توأم نمونه ها برای انجام این آزمون به روش پیرسون الزامی است. ضرایب همبستگی کندال و اسپیرمن که بر اساس رتبه ها بدست می آیند، بر خلاف ضریب همبستگی اسپیرمن نسبت به نقاط دور افتاده و نرمال نبودن توزیع حساس نیستند.

مثال: داده‌های زیر عمق آبهای زیر زمینی در دو فصل تابستان و زمستان را برای ۹ منطقه نشان می‌دهد.

```
> summer=c(5.68,5.6,5.43,5.9,5.94,5.7,5.66,5.98,5.85)
> winter=c(5.36,5.41,5.39,5.85,5.82,5.73,5.58,5.89,5.78)
```

می خواهیم بدانیم که آیا یک همبستگی بین میزان عمق آب تابستان و زمستان در آبهای زیر زمینی وجود دارد یا نه؟

```
> cor(summer,winter)
[1] 0.8820102
```

پس یک همبستگی نسبتاً قوی مثبت بین داده‌ها وجود دارد. تعجبی نیست اگر در مناطقی که در تابستان دارای عمق آب بیشتری هستند در زمستان عمق آب آنها بیشتر شود. اگر بخواهیم یک سطح معنی‌داری برای میزان ضریب همبستگی مشخص کنیم با استفاده از تابع `cor.test()` داریم:

www.GhadamYar.com

```
> cor.test(summer,winter)

        Pearson's product-moment correlation

data:  summer and winter
t = 4.9521, df = 7, p-value = 0.001652
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.5259984 0.9750087
sample estimates:
      cor
0.8820102
```

مقدار p-value نشان می‌دهد که همبستگی شدیداً معنی‌دار است.

آزمون نیکویی برازش خی- دو

آزمون نیکویی برازش بررسی می‌کند که آیا داده‌ها از یک توزیع مشخص پیروی می‌کنند یا نه؟ آزمون نیکویی برازش خی- دو امکان آزمون برای داده‌های رسته ای که دارای مدلی به صورت احتمالات مشخص برای هر رسته می‌باشد، فراهم می‌کند. در پرتاب تاس، ۶ رسته (وجه تاس) می‌توانند هم شانس فرض شوند. در مورد توزیع حروف در یک متن برخی حروف را می‌توان دارای احتمال بیشتری نسبت به برخی دیگر برای مشاهده در نظر گرفت.

مثال: یک تاس را ۱۵۰ مرتبه پرتاب کرده‌ایم و توزیع زیر برای برآمدهای مشاهده شده است، آیا این تاس سالم است؟

وجه تاس	۱	۲	۳	۴	۵	۶
تعداد برآمد	۲۲	۲۱	۲۲	۲۷	۲۲	۳۶

مطمئناً شما شک می‌کنید، که اگر یک تاس سالم باشد، احتمال مشاهده هر وجه باید ۱/۶ باشد. در ۱۵۰ پرتاب باید انتظار داشته باشیم هر وجه تقریباً ۲۵ بار مشاهده شود. ولی در اینجا تعداد مشاهدات عدد ۶ برابر ۳۶ است. آیا این مسأله تصادفی است یا شاید چیز دیگری...؟

www.GhadamYar.com

کلید پاسخ به این سؤال بررسی میـزان فاصـله مشـاهدات از مقـدار مـورد انتظـار اسـت و ایـن میـزان بـا آمـاره خـی- دو بـه دست می‌آید.

$$\chi^2 = \sum_{i=1}^{n} \frac{(f_i - e_i)^2}{e_i}$$

که f_i فراوانی مشاهده شده و e_i فراوانی مورد انتظار رسته i ام است.

تابع مورد استفاده برای انجام آزمون خی دو chisq.test() است که شناسه اول آن بردار فراوانی مشاهدات و شناسه دوم p با مقدار احتمالات مربوط به هر رسته است:

```
> freq=c(22,21,22,27,22,36)
> probs=rep(1/6,6)
> chisq.test(freq,p=probs)

        Chi-squared test for given probabilities

data:  freq
X-squared = 6.72, df = 5, p-value = 0.2423
```

فرض صفر آزمون این است که هر رسته i ام دارای احتمال p_iاست(در مثال ما $p_i=1/6$) و فرض مقابل اینکه حداقل یکی از رسته ها دارای احتمال مشخص شده نباشد.

p-value برابر 0.2423 است، بنابراین هیچ دلیلی برای رد فرض صفر مبنی بر سالم بودن تاس نداریم.

آزمون استقلال خی- دو

آماره‌ای که در مثال فوق معرفی شد را می‌توان برای آزمون استقلال دو سطر در یک جدول توافقی به کار برد. فرض صفر این است که سطر ها مستقل اند و فرض مقابل اینکه سطر ها وابسته اند.

مثال: در یک تحقیق شدت جراحات وارده در تصادفات بر اساس اینکه سرنشینان از کمربند ایمنی استفاده کرده اند یا نه، جدول بندی شده است:

		شدت جراحات			
		هیچ	ناچیز	متوسط	زیاد
کمربند ایمنی	بله	۱۲۸۱۳	۶۴۷	۳۵۹	۴۲
	خیر	۶۵۹۶۳	۴۰۰۰	۲۶۴۲	۳۰۳

www.GhadamYar.com

آیا دو متغیر رسته ای از یکدیگر مستقل‌اند، یا کمربند ایمنی تأثیری بر شدت جراحات وارده دارد؟ برای این آزمون نیز از تابع chisq.test () استفاده می شود. سطوح متغیرهای رسته‌ای را می توان به صورت یک ماتریس یا چارچوب داده که حداقل دارای دو سطر و دو ستون است، به تابع chisq.test () معرفی کرد:

```
> yes=c(12813,647,359,42)
> no=c(65963,4000,2642,303)
> belt=rbind(yes,no)
```

که با فراخوانی ماتریس belt جدول توافقی ظاهر می‌شود:

```
> belt
     [,1] [,2] [,3] [,4]
yes 12813  647  359   42
no  65963 4000 2642  303
```

برای اجرای آزمون این ماتریس را در تابع chisq.test () قرار می‌دهیم:

```
> chisq.test(belt)

        Pearson's Chi-squared test

data:  belt
X-squared = 59.224, df = 3, p-value = 8.61e-13
```

آزمون به شدت معنی‌دار است، پس کمربندها را ببندیم !

آزمون برابری نسبت جامعه با یک نسبت فرضی (آزمون دقیق دو جمله ای)

برای آزمون $H_0: p=p_0$ با استفاده از توزیع دوجمله‌ای، از تابع binom.test () استفاده می‌شود. شناسه‌های الزامی این تابع تعداد موفقیت‌ها در نمونه و اندازه نمونه است. از شناسه‌های اختیاری این تابع می توان شناسه p برای مشخص کردن مقدار p_0 در فرض صفر و شناسه alternative برای مشخص کردن نوع فرض مقابل، را نام برد. مقدار پیش فرض شناسه p، ۰/۵ است.

www.GhadamYar.com

مثال: از یک نمونه تصادفی ۱۰۰ نفری در مورد استفاده از ماشین ظرفشویی سؤال کرده‌ایم و ۴۲ نفر از آنها پاسخ "بله" داده‌اند، آیا داده‌ها از این فرضیه که نسبت واقعی استفاده کنندگان از ماشین ظرفشویی ۵۰٪ است، پشتیبانی می‌کنند؟

```
> binom.test(42,100)

        Exact binomial test

data:  42 and 100

number of successes = 42, number of trials = 100, p-value =
0.1332

alternative hypothesis: true probability of success is not
equal to 0.5

95 percent confidence interval:

 0.3219855 0.5228808

sample estimates:

probability of success

                  0.42
```

پس فرض $H_0: p=0.5$ در مقابل فرض $H_1: p\neq 0.5$ رد نمی شود.

برای آزمون فرض $H_0: p=0.4$ در مقابل فرض $H_1: p>0.4$ به صورت زیر عمل کنید:

```
> binom.test(42,100,p=0.4,alt="g")


        Exact binomial test


data:  42 and 100

number of successes = 42, number of trials = 100, p-value = 0.3775

alternative hypothesis: true probability of success is greater
than 0.4

95 percent confidence interval:

 0.3364797 1.0000000

sample estimates:

probability of success

                  0.42
```

دلیل کافی برای رد فرض صفر وجود ندارد.

www.GhadamYar.com

آزمون نسبت با استفاده از تقریب نرمال

برای انجام آزمون برابری نسبت یک جامعه با یک نسبت فرضی و نیز برابری نسبت های دو جامعه با استفاده از تقریب نرمال از تابع `prop.test()` استفاده می شود.

آزمون برابری نسبت های دو جامعه

برای انجام آزمون برابری نسبت های دو جامعه از تابع `prop.test()` استفاده می‌شود. شناسه‌های الزامی این تابع دو بردار، یک شامل تعداد موفقیت ها در دو نمونه و دیگری شامل اندازه نمونه ها است. شناسه‌های اختیاری این تابع عبارتند از:

- `"alt"` : برای تعیین نوع فرض مقابل
- `"conf.level"` : برای تعیین سطح اطمینان فواصل اطمینان
- `"correct"` : استفاده از تصحیح پیوستگی، مقدار پیش فرض این شناسه TRUE است، یعنی آزمون با تصحیح پیوستگی انجام می گیرد.

مثال: در یک آزمایش پزشکی، تعداد ۱۱۰۳۷ نفر به عنوان گروه آزمایشی و ۱۱۰۳۴ نفر به عنوان گروه شاهد انتخاب شدند. به افراد گروه آزمایشی داروی آسپرین و به افراد گروه کنترل دارونما داده شده است. پس از مدتی ۱۰۴ نفر از گروه آزمایشی و ۱۸۹ نفر از گروه شاهد دچار حمله قلبی می شوند. هدف از این تحقیق بررسی اثر آسپرین بر کاهش حمله قلبی است. در این مثال بهتر است فرض برابری نسبت ها (بی اثر بودن آسپرین) در مقابل فرض تأثیر مثبت آسپرین آزمون شود. بدین منظور از عبارت زیر استفاده کنید:

```
> prop.test(x,n,alt="l")

        2-sample test for equality of proportions with
        continuity correction

data:  x out of n
X-squared = 24.4291, df = 1, p-value = 3.855e-07
alternative hypothesis: less
95 percent confidence interval:
 -1.000000000 -0.005082393
sample estimates:
    prop 1     prop 2
0.00942285 0.01712887
```


www.GhadamYar.com

مقدار p-value بسیار کوچک است لذا فرض صفر در این آزمون رد می‌شود. یعنی آسپرین در جلوگیری از حمله قلبی مؤثر است.

اکنون فرض کنید که پژوهشگران قبل از آزمایش عقیده دارند احتمال حمله قلبی در صورت مصرف آسپرین $p_{01}=0.01$ و در صورت مصرف دارونما $p_{02}=0.02$ است. برای آزمون درستی این ادعا از عبارت زیر استفاده کنید:

```
> prop.test(x,n,p=c(0.01,0.02))$p.value
[1] 0.09023416
```

آزمون در سطح ۰/۱ معنی دار است. در این دستور چون فقط مایل به مشاهده مقدار p-value بودیم، بعد از تابع prop.test() با استفاده از کاراکتر $ و عبارت p.value مستقیماً به مقدار p-value دست یافتیم.

تحلیل واریانس

برای مقایسه میانگین‌های k جامعه ($k\geq 2$) از تحلیل واریانس (ANOVA) استفاده می‌شود.

مثال: در یک همایش علمی به ۳۰۰ مقاله رسیده نمره‌ای داده می‌شود. مقالات رسیده توسط ۶ داور بررسی می‌شوند. دبیرخانه همایش می‌خواهد مطمئن شود که هر داور از شاخص یکسانی برای تصحیح مقالات استفاده کرده است. در یک نمونه تصادفی، ۲۷ نمره که توسط ۳ داور به مقالات داده شده است را در اختیار داریم. فرض‌های این آزمون را می توان به صورت زیر بازنویسی کرد:

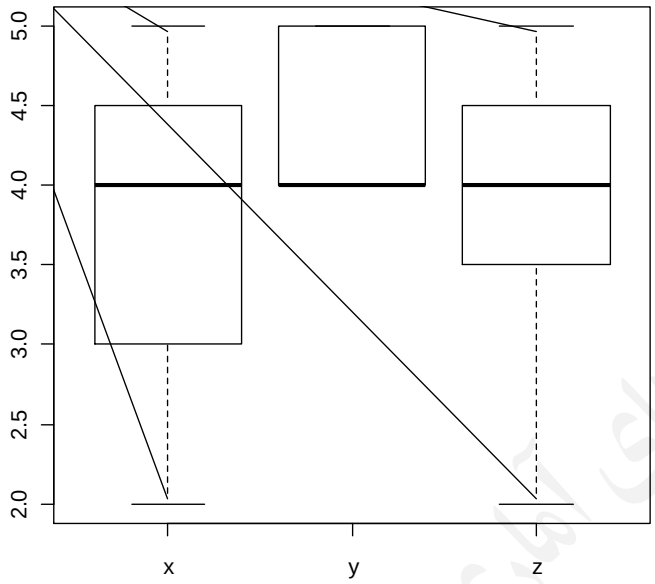
$$H_0: \mu_1 = \mu_2 = \mu_3$$

$$H_1: \text{حداقل یک } \mu_i \neq \mu_j$$

داده‌ها را به صورت زیر در R وارد می کنیم:

```
> x = c(4,3,4,5,2,3,4,5)
> y = c(4,4,5,5,4,5,4,4)
> z = c(3,4,2,4,5,5,4,4)
> scores = data.frame(x,y,z)
> boxplot(scores)
```


www.GhadamYar.com



از این نمودار مشخص می‌شود که داور ۲ متفاوت از داورهای ۱ و ۳ است.

تابع مورد نیاز برای انجام آزمون فرض آنالیز واریانس یکطرفه `oneway.test()` است، این تابع داده‌ها را به فرم متفاوتی می‌پذیرد. در حقیقت داده‌ها باید به صورت یک متغیر اصلی شامل نمرات و یک فاکتور برای مشخص کردن داور باشد. تابع `stack()` این کار را برای ما انجام می‌دهد:

```
> scores = stack(scores)
> names(scores)
[1] "values" "ind"
```

نمرات در متغیر `values` و شماره داور در متغیر کیفی (فاکتور) `ind` ثبت می شود. برای اجرای `oneway.test` باید فرمول مدل را به صورت زیر در نظر بگیریم:

```
> oneway.test(values ~ ind, data=scores, var.equal=T)

        One-way analysis of means

data:  values and ind
F = 1.1308, num df = 2, denom df = 21, p-value = 0.3417
```

مقدار ۰/۳۴ برای `p-value` یعنی فرض صفر (برابر بودن میانگین نمرات داده شده توسط سه داور) را می‌پذیریم.

با استفاده از تابع `anova()` نیز می‌توانید به نتایج بالا دست یابید.

```
> anova(lm(values~ind,data=scores))
```


www.GhadamYar.com

در این فصل به معرفی روشهای رگرسیونی برای برازش مدلهای خطی ساده می‌پردازیم.

رگرسیون خطی ساده

مثال: گفته می‌شود بیشترین تعداد ضربان قلب یک شخص و سن او توسط رابطه زیر به هم مربوط می شود:

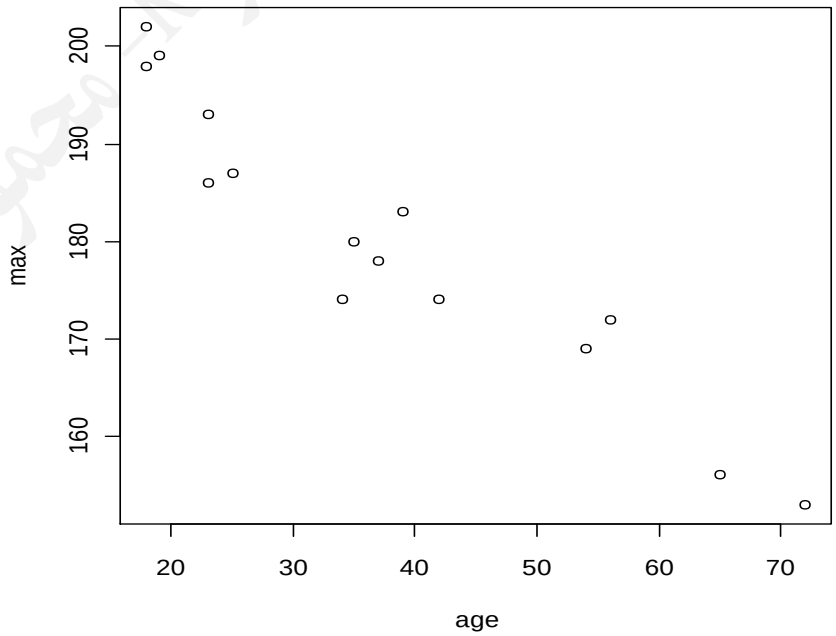
$$max = 220 - age$$

صحت این رابطه به طور تجربی ثابت شده است، در یک تحقیق بیشترین تعداد ضربان قلب ۱۵ نفر با سنین مختلف ثبت شده است، می‌خواهیم معادله خط رگرسیونی را پیدا کنیم.

```
> age=c(18,23,25,35,65,54,34,56,72,19,23,42,18,39,37)
> max=c(202,186,187,180,156,169,174,172,153,199,193,174,198,183,178)
```

نمودار پراکنش داده‌ها (Scatter Plot) را با تابع plot() رسم می‌کنیم.

```
> plot(age,max)
```



نمودار پراکنش یک روند خطی با شیب منفی را نشان می دهد.

www.GhadamYar.com

برای برازش خط رگرسیونی از تابع lm() که مخفف linear model است، استفاده می‌کنیم. شناسه این تابع، فرمول مدلی است که می‌خواهیم برازش دهیم و در این مثال max~age است، age به عنوان متغیر مستقل و max به عنوان متغیر وابسته.

```
> linreg=lm(max~age)
```

نتیجه این تابع را در متغیر linreg ثبت کردیم. برای مشاهده نتایج این برازش از تابع summary استفاده می‌کنیم. خروجی این تابع شامل ضرایب رگرسیونی، معیار R^2، مقدار p-value مربوط به آزمون معنی داری مدل، SSE و ... می‌باشد.

```
> summary(linreg)

Call:
lm(formula = max ~ age)

Residuals:
    Min      1Q  Median      3Q     Max 
-8.9258 -2.5383  0.3879  3.1867  6.6242 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 210.04846    2.86694   73.27  < 2e-16 ***
age          -0.79773    0.06996  -11.40 3.85e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.578 on 13 degrees of freedom
Multiple R-squared: 0.9091,     Adjusted R-squared: 0.9021 
F-statistic:   130 on 1 and 13 DF,  p-value: 3.848e-08
```

مقادیر p-value مربوط به مقدار ثابت و شیب خط نشان دهنده معنی دار بودن آنهاست.

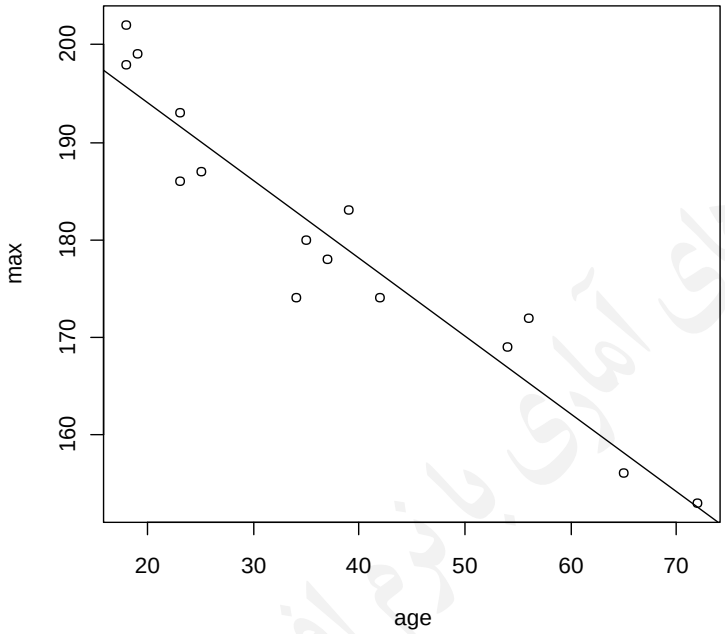
اسامی مقادیری که متغیر linreg ذخیره شده است:

```
> names(linreg)
 [1] "coefficients"  "residuals"     "effects"       "rank"         
 [5] "fitted.values" "assign"        "qr"            "df.residual"  
 [9] "xlevels"       "call"          "terms"         "model"        
```


www.GhadamYar.com

برای نمایش خط رگرسیونی پس از رسم نمودار پراکنش با استفاده از تابع `abline()` به صورت زیر عمل کنید:

```
> plot(age,max)
> abline(linreg)
```



از سه روش زیر می توان به مقادیر فیت شده یا $\hat{y}$ دست یافت:

```
> pred = predict(linreg)
> pred = fitted(linreg)
> pred = linreg$fitted
```

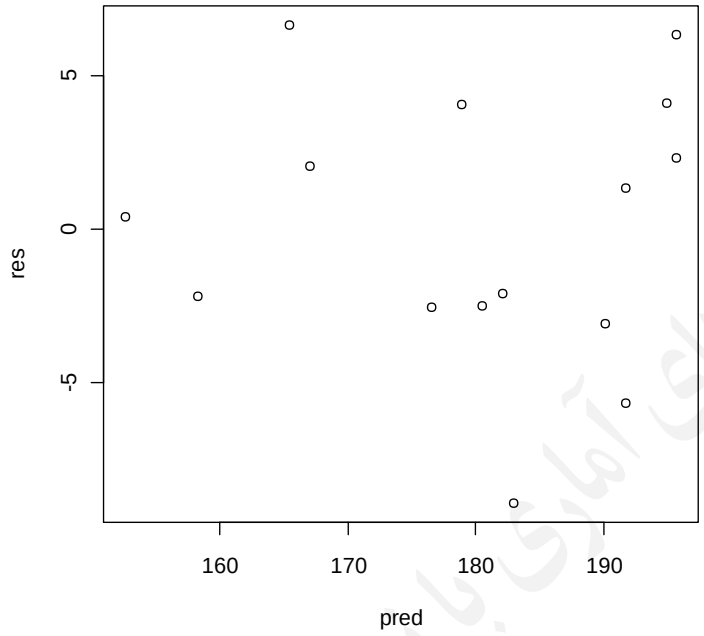
برای دسترسی به مقادیر باقیمانده ها یعنی $y - \hat{y}$ ، می‌توانید یکی از دستورات زیر را به کار ببرید:

```
> res = max - pred
> res = residuals(linreg)
> res = linreg$residuals
```

مقادیر فیت شده را در مقابل مقادیر باقیمانده‌ها رسم می‌کنیم:

www.GhadamYar.com

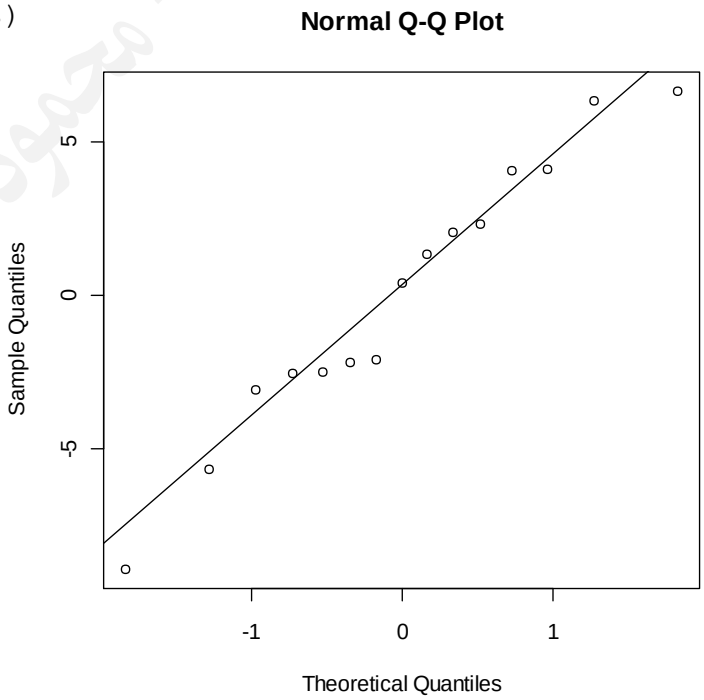
```
> plot(pred,res)
```



باقیمانده‌ها به طور تصادفی در اطراف صفر پراکنده شده‌اند و از روند خاصی پیروی نمی‌کنند، این امر برای مناسبت مدل مطلوب است.

فرض نرمال بودن باقیمانده ها را با نمودار Q-Q plot بررسی می‌کنیم.

```
> qqnorm(res)
> qqline(res)
```



www.GhadamYar.com

برای مشاهده فواصل اطمینان مربوط به پارامترهای مدل از دستور زیر استفاده کنید:

```
> confint(linreg,level=.95)
                 2.5 %      97.5 %
(Intercept) 203.854813 216.2421034
age          -0.948872  -0.6465811
```

تحلیل های آماری با نرم افزار R- محمود مسلمان

www.GhadamYar.com

از R می‌توان برای شبیه سازی مسائل مختلف آماری به منظور درک بهتر قضایا استفاده کرد.

برآورد ماکزیمم درستنمایی

اکثر توابع بهینه سازی در R توابع مینیمم کننده هستند و معمولاً باید مسائل را به صورت مسئله مینیمم سازی باز نویسی کرد. مثلاً برای ماکزیمم کردن یک تابع درستنمایی باید منفی تابع درستنمایی را مینیمم کنید. توابع زیادی برای این کار وجود دارد: nlm، optim، optimize، nlminb و constrOpim. تابع nlminb () هم برای مسائل تک پارامتری و هم برای مسائل چند پارامتری قابل استفاده است و همچنین قابلیت بالایی برای محدود کردن پارامترها دارد. کمترین چیزی که برای استفاده از این تابع لازم داریم، عبارت است از یک مقدار اولیه برای پارامتر و یک تابع برای مینیمم کردن.

مثال: فرض کنید ۲۰ مشاهده از یک توزیع نمایی با پارامتر λ در اختیار داریم (این داده‌ها را شبیه سازی می‌کنیم). مایلیم برآورد ماکزیمم درستنمایی را برای λ بیابیم. می دانیم تابع چگالی توزیع نمایی به صورت زیر است:

$$f(x|\lambda) = \lambda e^{-\lambda x} \quad \lambda \geq 0$$

بنابراین تابع log-likelihood را به صورت زیر داریم:

$$l(\lambda) = \sum_{i=0}^{20} (\ln \lambda - \lambda x_i)$$

$$= n \ln \lambda - \lambda \sum_{i=0}^{20} x_i$$

که در R می‌خواهیم این تابع را مینیمم کنیم.

```
> x = rexp(20, rate = 4)
>  n = length(x)
>  nllhood = function(lambda) {
+  -1 * (n * log(lambda) - lambda * sum(x))
+  }
>  fit = nlminb(6, nllhood)
```


www.GhadamYar.com

برای مشاهده مقدار برآورد λ از عبارت زیر استفاده کنید:

```
> fit$par
[1] 4.366605
```

ملاحظه می‌کنید که مقدار برآورد به مقدار اصلی پارامتر (۴) نزدیک است.

قضیه حد مرکزی

فرض کنید $X_1, X_2, \ldots, X_n$ یک نمونه تصادفی از توزیعی با میانگین و واریانس μ و σ^2 باشد، به طوری که $\sigma^2 < \infty$.
اگر n به اندازه کافی بزرگ باشد، دو نتیجه تقریبی زیر را داریم:

الف- $T_n = X_1 + X_2 + \ldots + X_n$ تقریباً دارای توزیع $N(n\mu, n\sigma^2)$ است.

ب- $\bar{X} = \frac{X_1 + X_2 + \ldots + X_n}{n}$ تقریباً دارای توزیع $N\left(\mu, \frac{\sigma^2}{n}\right)$ است.

اکنون با استفاده از R صحت این قضیه را برای توزیع یکنواخت تحقیق می‌کنیم.

مثال: برای توزیع یکنواخت $T = \sum_{i=0}^{n} X_i$ را در نظر بگیرید، که در آن $X_i \sim U(0,1)$. بر طبق قضیه حد مرکزی برای n به اندازه کافی بزرگ داریم:

$$T \approx N\left(n\left(\frac{1}{2}\right), n\left(\frac{1}{12}\right)\right)$$

اکنون برای n های ۳، ۵، ۲۰ و ۵۰ و با ۱۰۰۰۰ تکرار این مثال را شبیه سازی می‌کنیم:

ابتدا پنجره رسم نمودار را برای رسم همزمان ۴ نمودار (به ازای مقادیر مختلف n) آماده کنید:

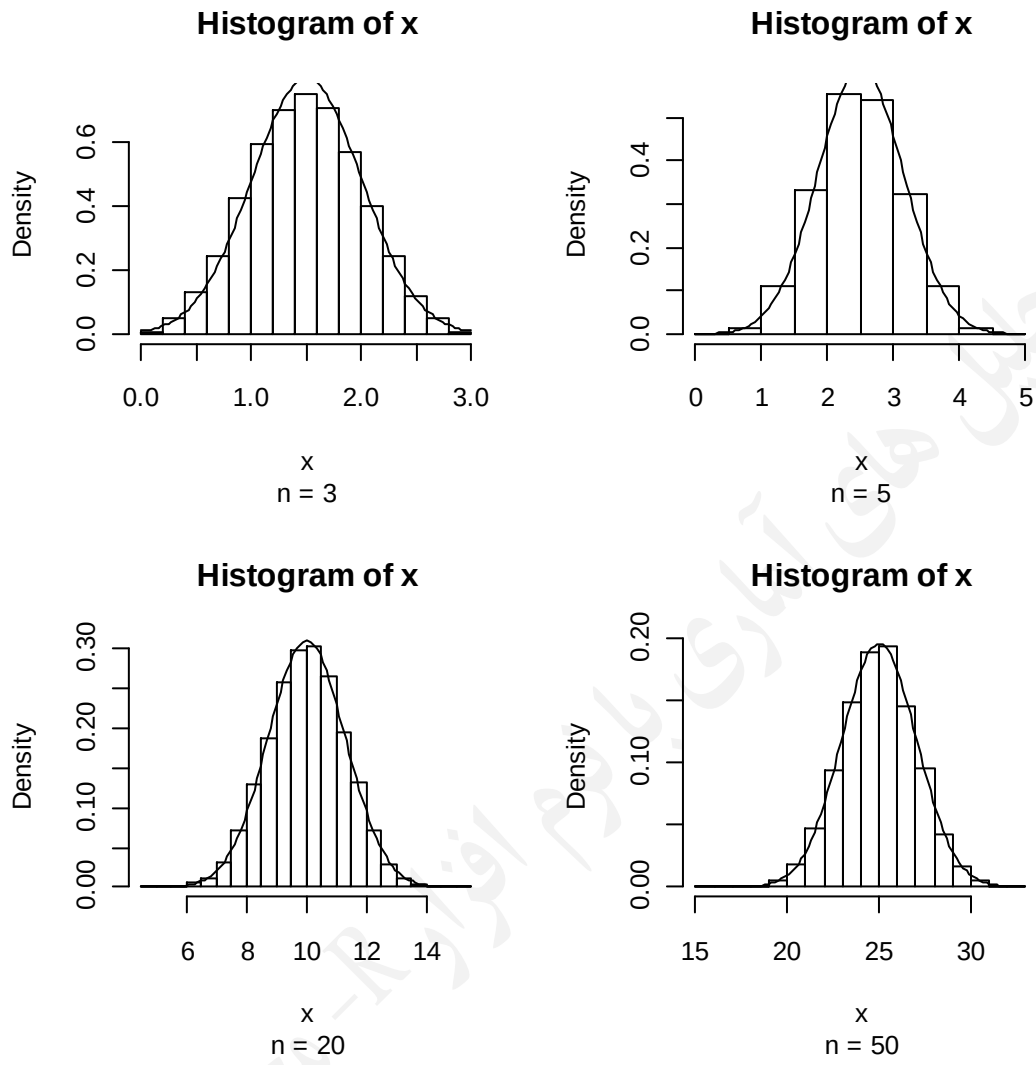
```
> par(mfrow=c(2,2))
```

سپس دستورات زیر را ۴ بار تکرار کنید (هر بار پس از تغییر مقدار n):

```
> n = 3                    # این مقدار را هر بار به ۵، ۲۰ و ۵۰ تغییر دهید
> x = c()
> for(r in 1:10000){ x[r] = sum(runif(n)) }
> hist(x,freq=F)
> curve(dnorm(x,n/2,sqrt(n/12)),add=T)
```

در دستورات فوق از تابع curve() برای رسم چگالی نرمال روی هیستوگرام داده‌ها استفاده شد. شناسه add=T برای رسم منحنی جدید روی منحنی قبلی به کار می‌رود.

www.GhadamYar.com



همانطور که در شکل بالا مشاهده می‌کنید هرچه n بزرگتر می‌شود، منحنی زنگی شکل نرمال تطبیق بیشتری بر لبه‌های هیستوگرام پیدا می‌کند، به نظر می‌رسد برای n مساوی ۲۰ و ۵۰ شرایط قضیه حد مرکزی برقرار است و منحنی چگالی نرمال به خوبی لبه‌های هیستوگرام را پوشش می‌دهد.

www.GhadamYar.com

مراجع و مآخذ

1. Chambers, J .*Introductory Statistics with R* .New York: Springer, 2008.
2. Crawley, M, J .*Statistics: An Introduction using R* .New York: Wiley, 2005.
3. Karim, Ehsan " *Basic Statistics and R: An Introductory Tutorial* ".2008.
4. Verzani, J .*Using R for Introductory Statistics* .Florida: Chapman & Hall/CRC Press, 2005.

۵. بهبودیان, جواد. *آمار و احتمال مقدماتی*. مشهد: دانشگاه امام رضا(ع), ۱۳۸۳.

۶. نواب پور, حمید رضا; یزدانی, افسانه; ایزدی, غلامرضا. *محاسبات آماری با کامپیوتر*. تهران: انتشارات دانشگاه پیام نور, ۱۳۸۸.

تحلیل های آماری با نرم افزار R- محمود مسلمان

www.GhadamYar.com