Домашна работа №1

на
Денис Тихогло
Софтуерно инженерство, 3 курс, 3 група, ф.н. 855256

- **Прочетете данните и ги запишете в data frame**

> setwd("~//doc//sem") – посочвам моята работна директория да е тази, която съдържа файла pokemon.csv;

> pokemons = read.csv("pokemon.csv" , header = TRUE) – четя съдържанието на файла pokemon.csv в датафрейм pokemons;

- **Генерирайте си подизвадка от 600 наблюдения. Запишете само редовете със съответните индекси в нов дейтафрейм.**

> set.seed(855256) – задавам състояние на генератора на случайни числа, използвайки своят факултетен номер;

> sample_vector = sample(1:705, 600, replace = FALSE) – генерирам вектор, който съдържа извадка без връщане от 600 наблюдения;

> pokemons600 = pokemons[pokemons$Number %in% sample_vector,] – записвам само редове с съответните индекси в нов датафрейм pokemons600

> attach(pokemons600) – атачвам датафрейма за да не указвам всеки път неговото име при достъпване на колона.

- **Изкарайте на екрана първите няколко (5-6) наблюдения;**

> head(pokemons600) – показва първите 5 реда от датафрейма.

- **Какъв вид данни са записани във всяка от променливите?**

Данните в колони Name, Type1 и Type2 са качественни, тъй като техните възможни значения са словесни и описателни; възможните значения са непресичащи се класове.
Данните в колона Number носи смисъл на идентификатора, затова също може да бъде разгладана като качественна характеристика.
Данните в колони Attack, Defense, Height и Weight са количественни, тъй като те приемат съдържателни числови значения.
Данните в колони Attack и Defense са дискретни, тъй като те могат да заемат само цели числови стойности.
Данните в колони Height и Weight са непрекъснати, тъй като те могат да заемат всички стойности – и цели и дробни.

- **Изведете дескриптивни статистики за всяка една променлива**

Статистика за поле Attack:

```
> summary(Attack)
   Min.    1st Qu.      Median       Mean      3rd Qu.      Max.
   5.00     53.00        73.00        74.44     93.00        165.00
```

Отговорът означава, че минималната стойност на атака в извадката е 5, първия квантил е 53, медианата (II-рия квантил) е 73, средното значение е 74.44, третия квантил е 93 и максималното значение в цялата извадка е 165.

Статистика за поле Defense:

```
> summary(Defense)
   Min.    1st Qu.      Median       Mean     3rd Qu.      Max.
   5.00     50.00        65.00        70.74     85.00        230.00
```

Отговорът означава, че минималната стойност на защита в извадката е 5, първия квантил е 50, медианата е 65, средното значение е 70.74 , третия квантил е 85 и максималното значение е 230.

Статистика за поле Height:

```
> summary(Height)
   Min.    1st Qu.     Median       Mean     3rd Qu.      Max.
   0.100   0.610       0.990       1.132     1.400        9.190
```

Отговорът означава, че минималната стойност на височина в извадката е 0.1, първия квантил е 0.61, медианата е 0.99, средното значение е 1.132, третия квантил е 1.4 и максималното значение е 9.190.

Статистика за поле Weight:

```
> summary(Weight)
   Min.    1st Qu.     Median         Mean      3rd Qu.     Max.
   0.10     9.50          28.00       57.27      62.25       950.00
```

Отговорът означава, че минималната стойност на теглото в извадката е 0.1, първия квантил е 9.5, медианата е 28, средното значение е 57.27, третия квантил е 62.25 и максималното значение е 950.

За количественни данни, можем да получим процентно разпределение на покемони в различни групи. Например, за променливата Type1:

```
> cbind(round(prop.table(table(Type1))*100, 2)) -
```

дава резултат, който показва, колко процента от покемони принадлежат към различни типове:

```
              [,1]
Bug           9.17
```

```
Dark          3.83
Dragon        3.50
Electric      4.67
Fairy         2.50
Fighting      3.50
Fire          6.17
Flying        0.17
Ghost         3.00
Grass         9.17
Ground        4.17
Ice           3.67
Normal       12.33
Poison        4.17
Psychic       5.67
Rock          5.83
Steel         3.17
Water        15.33
```

Вижда се, че най-голям брой са покемони с `Type1 = Water – 15.33` процента, а най-малък брой са покемони с `Type1 = Flying –  0.17` процента.

По аналогичния начин постъпвам и за променливата `Type2`:

`> cbind(round(prop.table(table(Type1))*100, 2))` – дава резултат

```
               [,1]
               50.67
Bug             0.50
Dark            2.67
Dragon          1.67
Electric        0.83
Fairy           2.83
Fighting        2.33
Fire            1.33
Flying         12.50
Ghost           1.50
Grass           2.17
Ground          4.17
Ice             1.33
Normal          0.67
Poison          4.67
Psychic         3.83
```

```
Rock           2.00
Steel          2.50
Water          1.83
```

Вижда се, че най-голям брой покемони в извадката изобщо нямат `Type2` - такива са `50.67` процента. Най-малък брой са покемони с `Type2 = Bug` - те са `0.` процента.

Променливите `Number` и `Name` са уникални за всеки покемон и нямат статистически храктерисики.

- **Изведете редовете на най-високия и на най-лекия покемон**

`> pokemons600[Height == max(Height), ]` - редове с най-високия покемон. Резултатът е

```
    Number    Name Type1  Type2 Attack Defense Height Weight
208    208 Steelix Steel Ground     85     200   9.19    400
```

`> pokemons600[Weight == min(Weight), ]` - редове с най-лекия покемон. Те са `3` на брой:

```
    Number    Name Type1  Type2 Attack Defense Height Weight
92      92  Gastly Ghost Poison     35      30    1.3    0.1
93      93 Haunter Ghost Poison     50      45    1.6    0.1
659    659 Flabébé Fairy            38      39    0.1    0.1
```

- **Изведете редовете на покемоните с общ брой точки за атака и защита над 220**

```
> pokemons600[Attack + Defense > 220,]
```

Резултатите са:

	Number	Name	Type1	Type2	Attack	Defense	Height	Weight
76	76	Golem	Rock	Ground	120	130	1.40	300.0
91	91	Cloyster	Water	Ice	95	180	1.50	132.5
99	99	Kingler	Water		130	115	1.30	60.0
112	112	Rhydon	Ground	Rock	130	120	1.91	120.0
127	127	Pinsir	Bug		125	100	1.50	55.0
149	149	Dragonite	Dragon	Flying	134	95	2.21	210.0
205	205	Forretress	Bug	Steel	90	140	1.19	125.8
208	208	Steelix	Steel	Ground	85	200	9.19	400.0
212	212	Scizor	Bug	Steel	130	100	1.80	118.0
213	213	Shuckle	Bug	Rock	10	230	0.61	20.5
232	232	Donphan	Ground		120	120	1.09	120.0
248	248	Tyranitar	Rock	Dark	134	110	2.01	202.0
289	289	Slaking	Normal		160	100	2.01	130.5
306	306	Aggron	Steel	Rock	110	180	2.11	360.0
324	324	Torkoal	Fire		85	140	0.51	80.4
348	348	Armaldo	Rock	Bug	125	100	1.50	68.2
376	376	Metagross	Steel	Psychic	135	130	1.60	550.0
377	377	Regirock	Rock		100	200	1.70	230.0
379	379	Registeel	Steel		75	150	1.91	205.0
383	383	Groudon	Ground		150	140	3.51	950.0
384	384	Rayquaza	Dragon	Flying	150	90	7.01	206.5
408	408	Rampardos	Rock		165	60	1.60	102.5
443	443	Garchomp	Dragon	Ground	130	95	1.91	95.0
448	448	Hippowdon	Ground		112	118	2.01	300.0
462	462	Rhyperior	Ground	Rock	140	130	2.39	282.8
468	468	Leafeon	Grass		110	130	0.99	25.5
481	481	Dialga	Steel	Dragon	120	120	5.41	683.0
484	484	Regigigas	Normal		160	110	3.71	420.0
489	489	Arceus	Normal		120	120	3.20	320.0
522	522	Gigalith	Rock		135	130	1.70	260.0
530	530	Conkeldurr	Fighting		140	95	1.40	87.0
560	560	Carracosta	Water	Rock	108	133	1.19	81.0
584	584	Escavalier	Bug	Steel	135	105	0.99	33.0
593	593	Ferrothorn	Grass	Steel	94	131	0.99	110.0
607	607	Haxorus	Dragon		147	90	1.80	105.5
620	620	Bisharp	Dark	Steel	125	100	1.60	70.0
627	627	Durant	Bug	Steel	109	112	0.30	33.0
637	637	Zekrom	Dragon	Electric	150	120	2.90	345.0
642	642	Chesnaught	Grass	Fighting	107	122	1.60	90.0
669	669	Doublade	Steel	Ghost	110	150	0.84	4.5
685	685	Tyrantrum	Rock	Dragon	121	119	2.49	270.0
702	702	Xerneas	Fairy		131	95	3.00	215.0
703	703	Yveltal	Dark	Flying	131	95	5.79	203.0
704	704	Diancie	Rock	Fairy	100	150	0.71	8.8
705	705	Volcanion	Fire	Water	110	120	1.70	195.0

- **Колко на брой покемони имат първичен или вторичен тип `Dragon` или `Flying` и са високи над един метър?**

```
> nrow(pokemons600[(Type1 == "Dragon" | Type1 == "Flying" | Type2 == "Dragon" | Type2 == "Flying") & Height > 1, ])
```

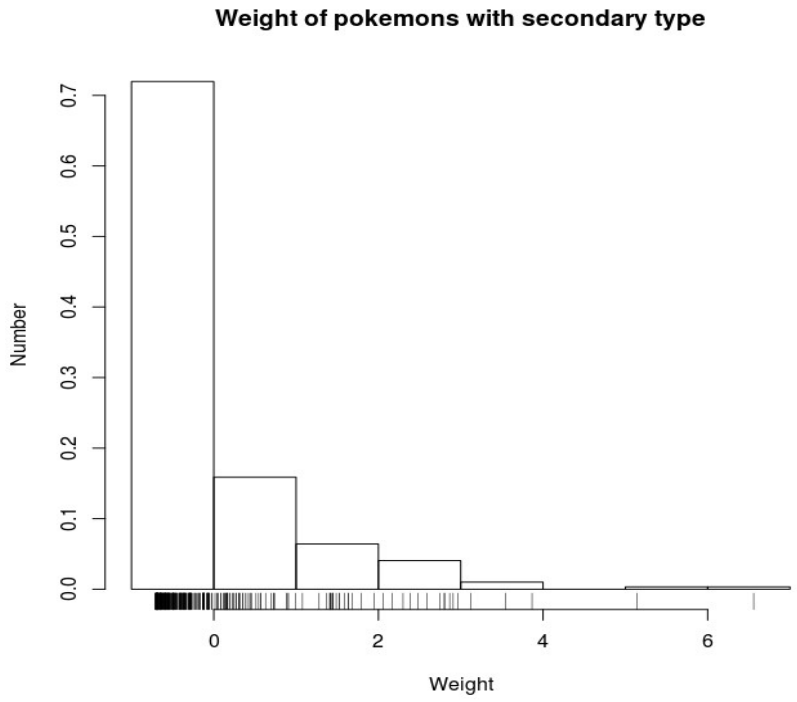
За моята извадка резултатът е 60.

- **Направете хистограма на теглото само на покемоните с вторичен тип и нанесете графика на плътността върху нея. Симетрично ли са разположени данните?**

`> type2pokemons = pokemons600[Type2 != "",]` – записвам в датафрейм `type2pokemons` само тези редове, в които стойността на `Type2` не е празен низ.

`> hist(scale(type2pokemons$Weight), probability=TRUE, main = "Weight of pokemons with secondary type", xlab = "Weight", ylab="Number")` – рисувам скалирана и нормирана хистограма.

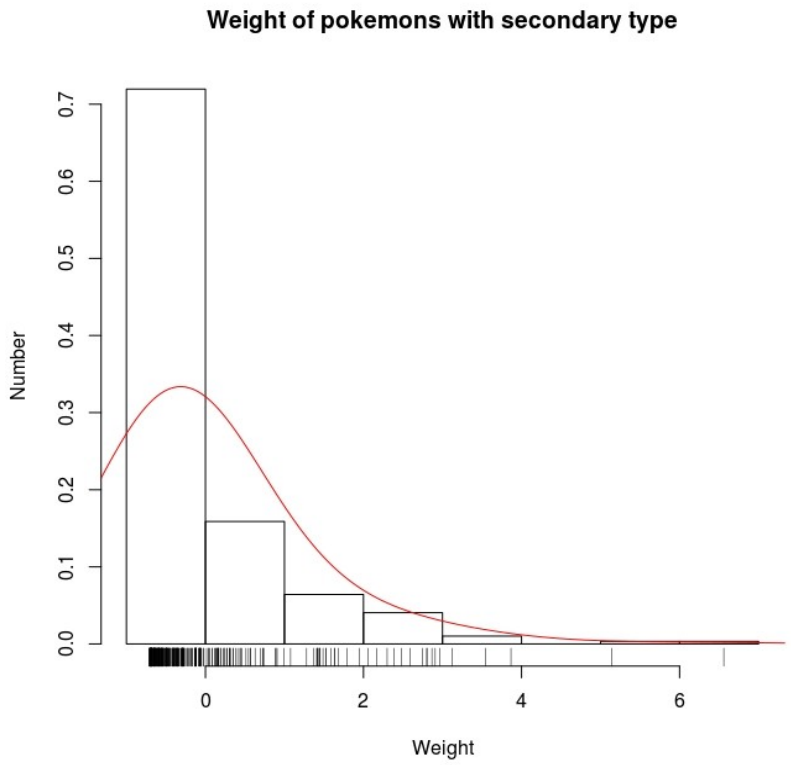
Командата `scale(type2pokemons$Weight)` дава информация, че `0` по абсцисната ос отговаря на стойност `66.76` – средно значение на извадката, а една стъпка по тази ос е равна на `93.95`. `probability=TRUE` задава, че това е хистограма на относителни честоти и лицето на пространство под хистограмата е равно на `1`.



`> rug(jitter(scale(type2pokemons$Weight)))` – отбелязва, къде точно се намират наблюденията.

На хистограмата се вижда, че, очевидно, данните не са разположени симетрично. Повече от 70% от покемони се намират наляво от средното, и всички те са в рамки на една стъпка. Тези които се намират по-дясно от центъра, са по-малко на брой, но те са по-разпръснати.

`> lines(density(scale(type2pokemons$Weight), bw = 1), col = 'red')` – чертае линия на плътността върху съществуващата хистограма.



Върхът на линията се намира наляво от средното (нула). Освен това, форма на линията говори, че налице е дясна (положителна) асиметрия. Това се съгласува с факта, че имаме по-голямо струпване на данни около по-малките значения на Weight. За проверка може да се използва функцията skewness() от пакета „moments“:

```
> skewness(type2pokemons$Weight)
```

Резултатът е 2.747286 , което говори за значителната дясна асиметрия.

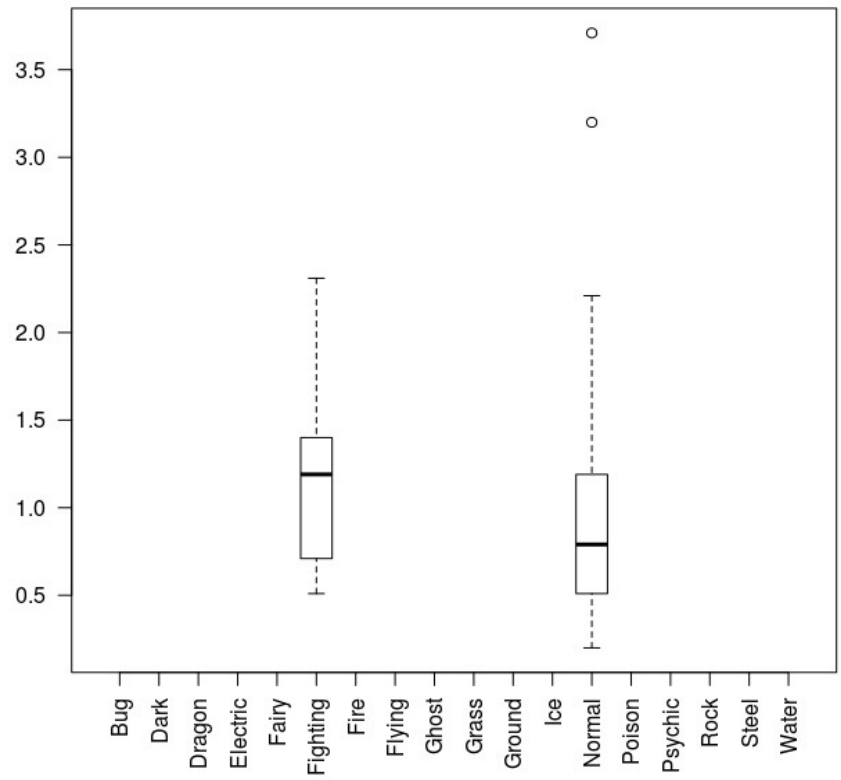
- **За покемоните с първичен тип "Normal" или "Fighting" изследвайте съвместно променливите Type1 и Height с подходящ графичен метод.**

```
> pokemonsNF = pokemons600[Type1 == "Normal" | Type1 == "Fighting", ]
```

– записвам в датафрейм pokemonsNF покемони с първичен тип "Normal" или "Fighting".

Имаме един количествен признак (Height) и един качествен признак (Type1). За изследване на влияние на качествения признак върху количетсвения, ще нарисувам два boxplot-а, представящи данни на Height поотделно по групи на Type1.

```
boxplot(formula = Height ~ Type1, data = pokemonsNF)
```



Наляво наблюдаваме boxplot-а за "Fighting" подгрупата,надясно – за "Normal" подгрупата. За подгрупата "Normal" има по-голямо разнообразие на височина в извадката, в сравнение с "Fighting". За

тази подгрупа е наблюдавна най-висока и най-ниска стойност на височина.Освен това, тя има два outlier-а нагоре, само те са причина най-високата стойост да принадлежи на "Normal" подгрупата.

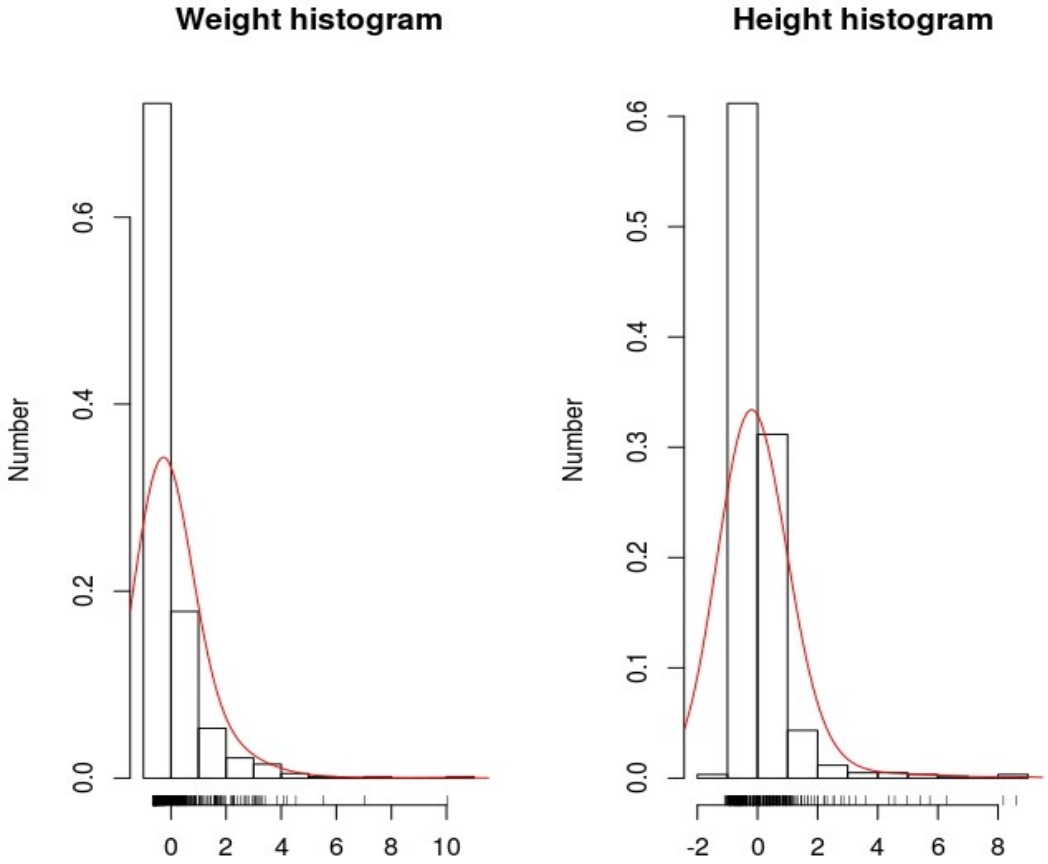
Лявата кутия се намира малко по-горе от дясната (медиана на лявата е примерно на едно ниво с горната граница на дясната). Следователно, 50% данни от тип "Fighting" приемат същата или по-голяма стойност от 50% данни от тип "Normal".
На графика се вижда, че трети квантил, медианата, втори квантил и минималната стойност за подгрупата "Normal" приемат по-малки стойности от съответните стойности на подгрупата "Fighting".
В лявата кутия медианата е по-близо към горната част на кутия. Това означава, че данните с подгрупа "Fighting" са разположени по-гъсто между медианата и третия квантил (то ест между 50% и 75% на вариационния ред)относително данните между третия квантил и медианата (то ест между 25% и 50% на вариационния ред). За дясната кутия ("Normal" подгрупа) ситуацията е противоположна – данните от 25% до 50% са разположени по-гъсто от данните от 50% до 75% на вариационния ред.
Виждаме, че и за двата типа покемони кутиите са разположени доста по-близо към минималната стйност, а не максималната. Това дава подсказка, че и в двата случая има дясна асиметрия.

- **Изследвайте съвместно променливите Height и Weight с подходящ графичен метод**.

Сравнение на две количественни величини може да бъде направено, например,с помощта на де хистограми с линнии на плътност. За да имаме еднакви скали по абсцисната ос ще центрираме и норомираме данните с функция scale().

```
> par(mfrow = c(1, 2)) – за едновременно показване на две хистограми
> hist(scale(Weight), probability = TRUE, right = FALSE, main =
"Weight histogram", xlab = "", ylab = "Number") – хистограма на
    относителни честоти на теглото.
> rug(jitter(scale(Weight))) – отбелязване на точни наблюдения
> lines(density(scale(Weight),bw = 1), col = 'red') – линия на
    плътността
> hist(scale(Height), probability = TRUE, right = FALSE, main =
"Height histogram", xlab = "", ylab = "Number") – хистограма на
    относителни честоти на височината.
```

`> rug(jitter(scale(Height)))` – отбелязване на точни наблюдения
`> lines(density(scale(Height),bw = 1), col = 'red')` – линия на плътността



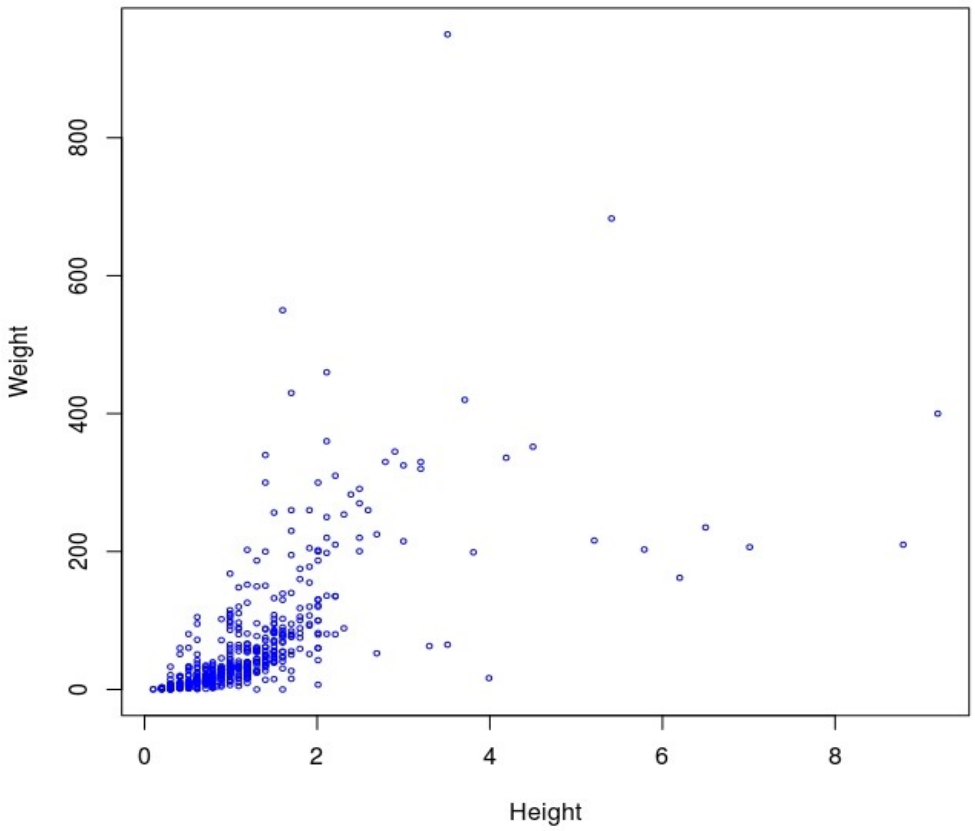
Графиците изглеждат доста подобно. За двете променливи повече от 60% от наблюдения се намират наляво от средното, и то в рамките на една стъпка. Останалите данни се намират отдясно от средното значение (нула) и са по-разпръснати. Линни на плътност също изглеждат приличащо, и двете променливи имат положителна асиметрия. Следователно има вероятност, че съществува положителна корелация между теглото и височина.
Сега да намерим числеността значение на корелацията:

```
> cor(Height, Weight)
```

Резултатът е 0.66, което съответства на средна по сила правопропорционалност.

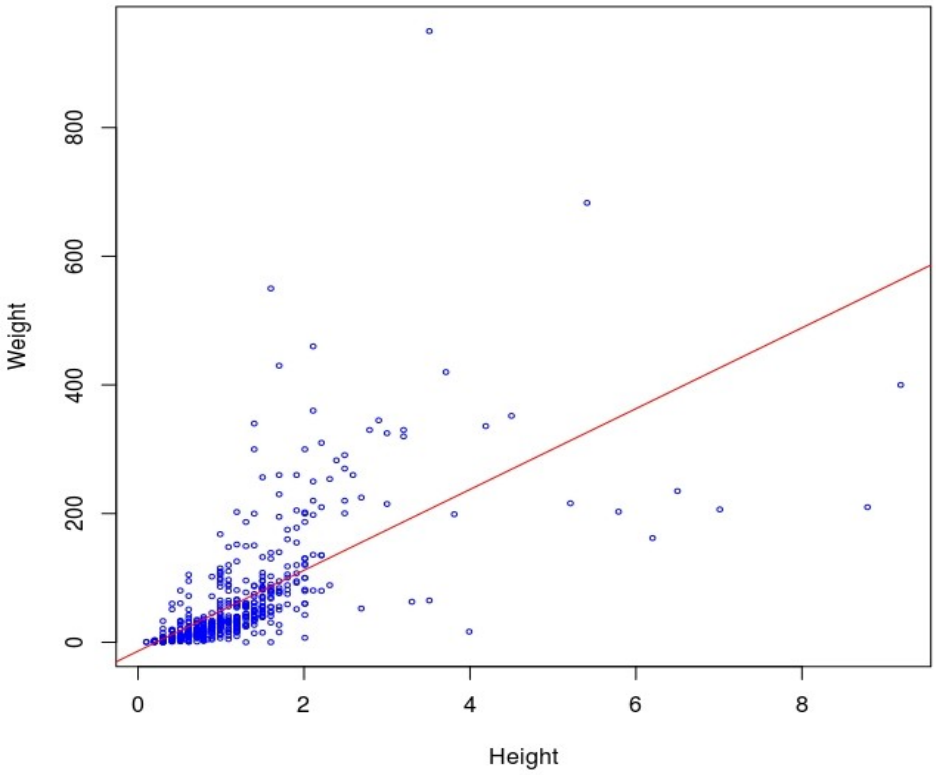
Да построим корелационно поле. (На този график за удобство теглото ще е зависима променлива от височината, но няма особена разлика, тъй като корелацията не носи никаква информация за причинно-следствена връзка)

`> plot(Height,Weight, col="blue", cex=0.5)` – Scatter plot за Height и Weight



На графика се вижда, че в повечето случаи увеличаването на височина и на тегло са право пропорционални.
Да начертаем регресионната права върху получения график.

`> abline(lm(Weight ~ Height), col="red")` – добавя регресионна линия върху корелационно поле.



Червената права съответства на изградения линеен модел; тя минава възможно най-близо към повечето точки (по метода на най-малките квадрати).

`> result_model = lm(Weight ~ Height)` – записваме получената линейна модел в променливата `result_model`.

Сега `result_model$coefficients[1]` е равно на `-13.83627` и това е свободния член, а `result_model$coefficients[2]` е равно на `62.81799` и това е ъгловия коефициент на линейния модел.

Следователно, покемон с височина `2.1` метра на базата на линейния модел се очаква да е с тегло `2.1 * 62.81799 - 13.83627 = 118.0815` килограма