Università degli Studi di Salerno

Dipartimento di Informatica

Corso di Laurea Magistrale in Informatica

a.a. 2017/2018

Homework 3

Alessandra Orsi
Matricola: 0522500424

Prof. Carmine Gravino
Corso: Metriche e Qualità del Software

Sommario

1. Il Dataset

Il primo passo per fare l'analisi dei dati è quello di caricare il dataset (dataset n.6 nel mio caso):

```
## carico il dataset
dati <- read.table("/Users/Alessandra/Desktop/dataset6Nasa.csv", header = TRUE, sep = ";", dec = ",")
```

Successivamente bisogna differenziare le variabili, ovvero vi è la necessità di individuare le variabili dipendenti e le variabili indipendenti.
Come possiamo vedere nello snippet di codice che segue, vi è una singola variabile dipendente nel dataset n.6, ovvero *EffortOreUomo*, e 6 variabili indipendenti (*Methodology, Complexity, Experience, TotalLines, NewLines, DevelopedLines*).
Nel set di dati originale le variabili indipendenti sono 7 ma ho deciso di non caricare quella relativa all'ID dei record perché non avrebbe portato alcun contributo significativo alla mia analisi.

```
## le variabili che utilizzo: Dep = variabile dipendente e Ind2,Ind3,Ind4,Ind5,Ind6,Ind7 le var. indipendenti
Dep<-dati$EffortOreUomo
Ind2<-dati$Methodology
Ind3<-dati$Complexity
Ind4<-dati$Experience
Ind5<-dati$TotalLines
Ind6<-dati$NewLines
Ind7<-dati$DevelopedLines
```

Una volta caricate le variabili dipendenti e indipendenti, e prima di analizzarle e fornire le principali statistiche descrittive, bisogna verificare se i dati seguono una distribuzione normale (o Gaussiana); nel caso in cui ciò non è verificato, bisogna normalizzarli.
Quindi verifico la normalità della distribuzione nel seguente modo:

```
## test per la normalità della distribuzione
## Ipotesi nulla (H0): i dati seguono una distribuzione normale.
## p-value > 0.05: accetto H0
shapiro.test(Dep)
shapiro.test(Ind2)
shapiro.test(Ind3)
shapiro.test(Ind4)
shapiro.test(Ind5)
shapiro.test(Ind6)
shapiro.test(Ind7)
```

Il test di Shapiro-Wilk è un test per la verifica di ipotesi statistiche ed è considerato in letteratura uno dei test più potenti per la verifica della normalità, soprattutto per piccoli campioni (come nel mio caso).

L’esito del test è il seguente:

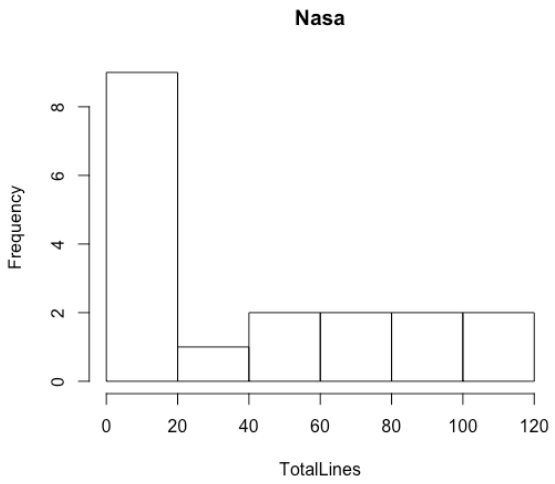
```
Console  Terminal ×
~/Desktop/
        Shapiro-Wilk normality test

data:  Dep
W = 0.83186, p-value = 0.004442

> shapiro.test(Ind2)

        Shapiro-Wilk normality test

data:  Ind2
W = 0.91055, p-value = 0.08798

> shapiro.test(Ind3)

        Shapiro-Wilk normality test

data:  Ind3
W = 0.86634, p-value = 0.0155

> shapiro.test(Ind4)

        Shapiro-Wilk normality test

data:  Ind4
W = 0.86543, p-value = 0.01497

> shapiro.test(Ind5)

        Shapiro-Wilk normality test

data:  Ind5
W = 0.85226, p-value = 0.009193

> shapiro.test(Ind6)

        Shapiro-Wilk normality test

data:  Ind6
W = 0.8366, p-value = 0.005243

> shapiro.test(Ind7)

        Shapiro-Wilk normality test

data:  Ind7
W = 0.8276, p-value = 0.003831
```

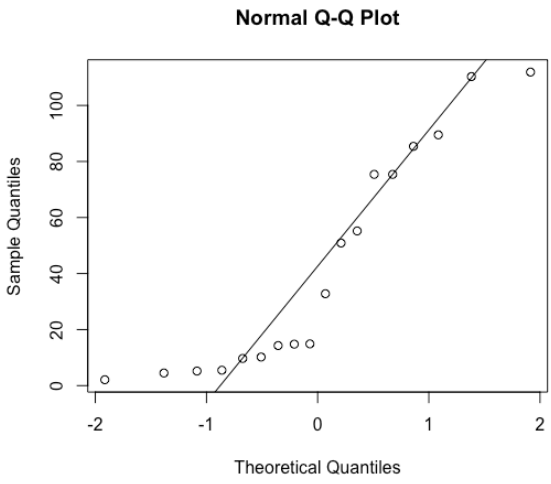
Poiché per molte variabili il **p-value**, ovvero il livello di significatività assegnato, ossia una misura di evidenza contro l’ipotesi nulla, è **< 0.05**, bisogna rigettare l’ipotesi nulla (H0: i dati seguono una distribuzione normale) e normalizzare tutti i dati applicando il logaritmo cercando di avvicinarci maggiormente ad una distribuzione Gaussiana.

Inoltre, ho ritenuto opportuno mostrare due tipi di grafici per ogni variabile (sia prima la normalizzazione e sia dopo la normalizzazione):

- Istogramma per mostrare la distribuzione di frequenza;
- Il Q-Q Plot, ovvero la rappresentazione grafica dei quantili di una distribuzione. Esso confronta la distribuzione cumulata della variabile osservata con la distribuzione cumulata della normale. Se la variabile osservata presenta una distribuzione normale, i punti di questa distribuzione congiunta si addensano sulla diagonale che va dal basso verso l'alto e da sinistra verso destra.

Prendiamo come esempio la variabile indipendente 5, ovvero *TotalLines*, e mostriamo i due grafici relativi ad essa:





Dall'istogramma possiamo ben vedere che non segue affatto una distribuzione normale, e anche osservando il Q-Q plot possiamo vedere che sono molti i punti che si discostano dalla diagonale.

Una volta normalizzati i dati, bisogna ricaricare le variabili e rieseguire lo Shapiro-Wilk test.

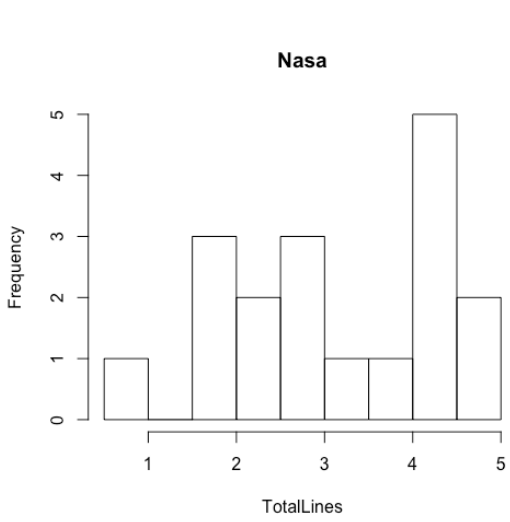
```
## cerco di avvicinarmi ad una distribuzione normale applicando il log ai dati
dati = log(dati)

## ricarico le variabili
Dep<-dati$EffortOreUomo
Ind2<-dati$Methodology
Ind3<-dati$Complexity
Ind4<-dati$Experience
Ind5<-dati$TotalLines
Ind6<-dati$NewLines
Ind7<-dati$DevelopedLines

## rifaccio lo shapiro test sui valori normalizzati
shapiro.test(Dep)
shapiro.test(Ind2)
shapiro.test(Ind3)
shapiro.test(Ind4)
shapiro.test(Ind5)
shapiro.test(Ind6)
shapiro.test(Ind7)
```

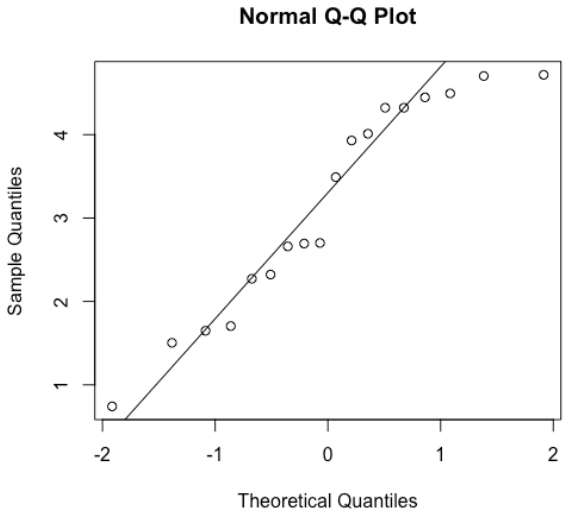
L’esito del secondo test è il seguente:

```
Console  Terminal
~/Desktop/
        Shapiro-Wilk normality test

data:  Dep
W = 0.89294, p-value = 0.04335

> shapiro.test(Ind2)

        Shapiro-Wilk normality test

data:  Ind2
W = 0.88085, p-value = 0.02697

> shapiro.test(Ind3)

        Shapiro-Wilk normality test

data:  Ind3
W = 0.8971, p-value = 0.05117

> shapiro.test(Ind4)

        Shapiro-Wilk normality test

data:  Ind4
W = 0.7251, p-value = 0.0001588

> shapiro.test(Ind5)

        Shapiro-Wilk normality test

data:  Ind5
W = 0.91444, p-value = 0.1031

> shapiro.test(Ind6)

        Shapiro-Wilk normality test

data:  Ind6
W = 0.94245, p-value = 0.3189

> shapiro.test(Ind7)

        Shapiro-Wilk normality test

data:  Ind7
W = 0.93686, p-value = 0.2558
```

Come possiamo notare, i valori sono nettamente migliorati. Inoltre ho rifatto l’istogramma ed il Q-Q plot per ciascuna variabile normalizzata, in modo tale da poter percepire anche un miglioramento visivo della distribuzione.

Prendiamo in considerazione come esempio sempre la variabile indipendente 5, ovvero *TotalLines*, e mostriamo i due nuovi grafici relativi ad essa:





Dall'istogramma possiamo vedere un netto cambiamento che ci mostra un avvicinamento ad una distribuzione normale; ancora più visibile nel Q-Q plot l'accostamento di quasi tutti i punti alla diagonale.

Inoltre, se volessimo confrontare i valori dei p-value dopo il test di Shapiro-Wilk nel caso dei dati normalizzati e in assenza di normalizzazione, otterremo esattamente la conferma che i p-value > 0.05 con i dati normalizzati sono nettamente aumentati, e quindi la distribuzione si avvicina sempre più a quella Gaussiana.

	p-value con dati normalizzati	**p-value con dati non normalizzati**
EffortOreUomo	0.04335	0.004442
Methodology	0.02697	0.0879
Complexity	0.05117	0.0155
Experience	0.0001588	0.0149
TotalLines	0.1031	0.0091
NewLines	0.3189	0.0052
DevelopedLines	0.2558	0.0038

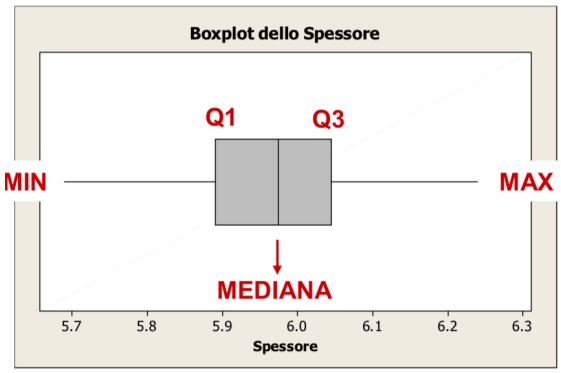
Adesso che è stata fatta la normalizzazione per migliorare la distribuzione, è possibile fornire le statistiche descrittive e fare il plot dei dati.

1.1 Statistiche descrittive e plot dei dati

La statistica descrittiva è un insieme di tecniche usate per descrivere le caratteristiche di base dei dati raccolti in un esperimento/studio.
Esse forniscono una sintesi semplice del campione e delle misure raccolte. Insieme alla semplice analisi grafica, costituisco la base iniziale di partenza di qualsivoglia analisi quantitativa dei dati.
I parametri calcolati per fornire le statistiche descrittive di ciascuna variabile sono i seguenti:

- Minimo: valore minimo;
- Massimo: valore massimo;
- 1° Quartile (Q1): il valore più vicino alla posizione 1⁄4 definisce Q1;
- 3° Quartile (Q3): il valore più vicino alla posizione 3⁄4 definisce Q2;
- Media: media dei valori;
- Mediana: la media dei due valori nella posizione centrale definisce la mediana;
- Deviazione Standard: stima della variabilità dei dati.

La rappresentazione grafica dei 5 parametri di sintesi (MIN, Q1, MEDIANA, Q3 e MAX) restituisce il cosiddetto BOXPLOT:



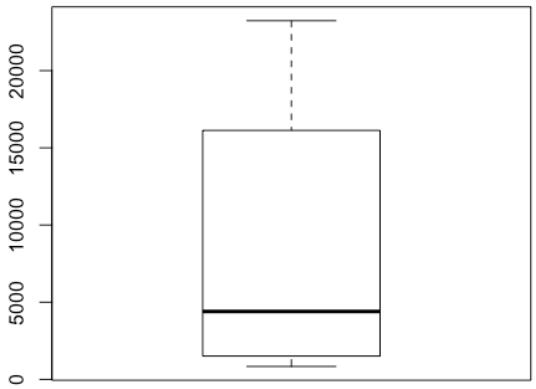
Questi parametri li ho ottenuti nel seguente modo:

```
## statistiche descrittive
summary(Dep)
summary(Ind2)
summary(Ind3)
summary(Ind4)
summary(Ind5)
summary(Ind6)
summary(Ind7)
# deviazione standard
sd(Dep)
sd(Ind2)
sd(Ind3)
sd(Ind4)
sd(Ind5)
sd(Ind6)
sd(Ind7)
```

I valori ottenuti sono i seguenti:

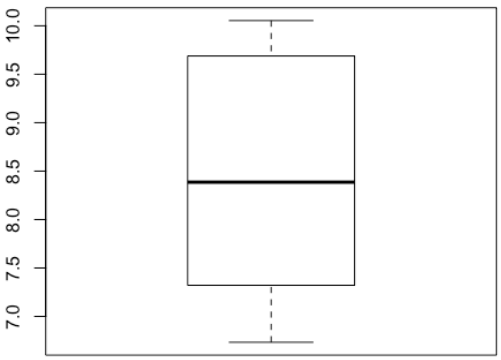
EffortOreUomo (Dep) non normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
840	1567	4402	8311	15910	23234	7681.946



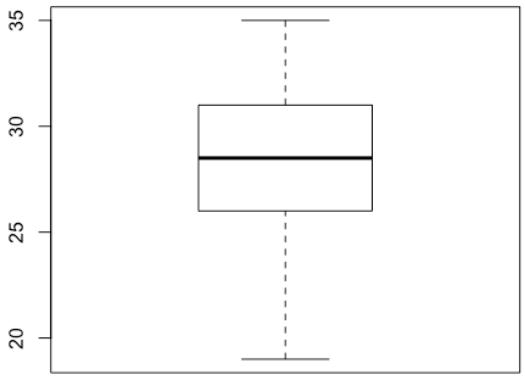
EffortOreUomo (Dep) normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
6.733	7.355	8.386	8.479	9.674	10.053	1.157558



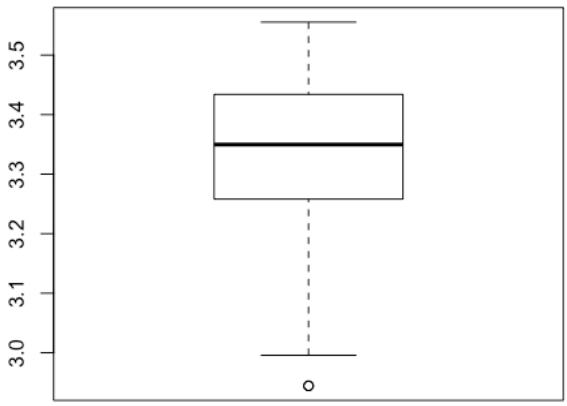
Methodology (Ind2) non normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
19.00	26.00	28.50	27.78	31.00	35.00	5.385772



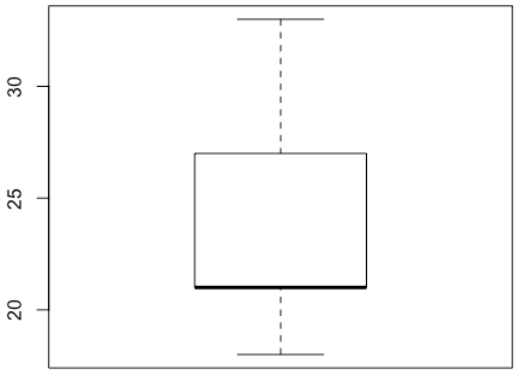
Methodology (Ind2) normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
2.944	3.258	3.350	3.305	3.434	3.555	0.2072579



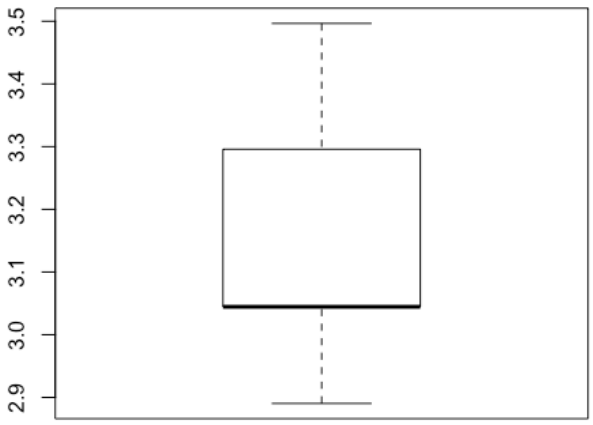
Complexity (Ind3) non normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
18.00	21.00	21.00	23.44	26.50	33.00	4.816909



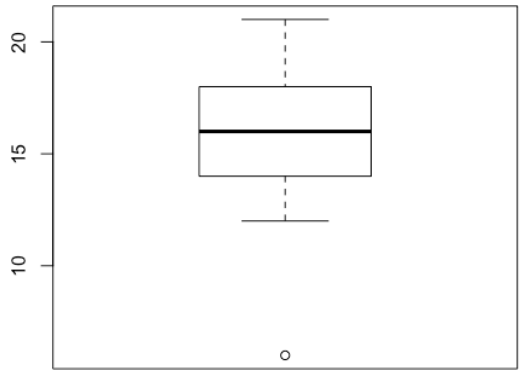
Complexity (Ind3) normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
2.890	3.045	3.045	3.136	3.277	3.497	0.194575



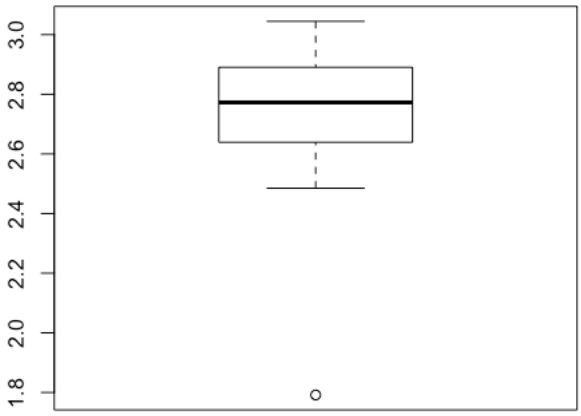
Experience (Ind4) non normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
6.00	14.50	16.00	15.83	17.50	21.00	3.365045



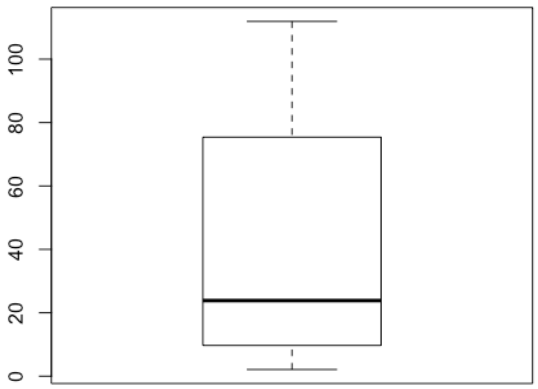
Experience (Ind4) normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
1.792	2.672	2.773	2.733	2.861	3.045	0.2734455



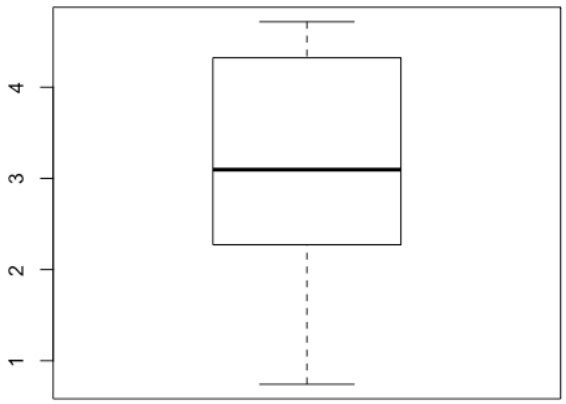
TotalLines (Ind5) non normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
2.100	9.825	23.850	42.667	75.400	111.900	39.28234



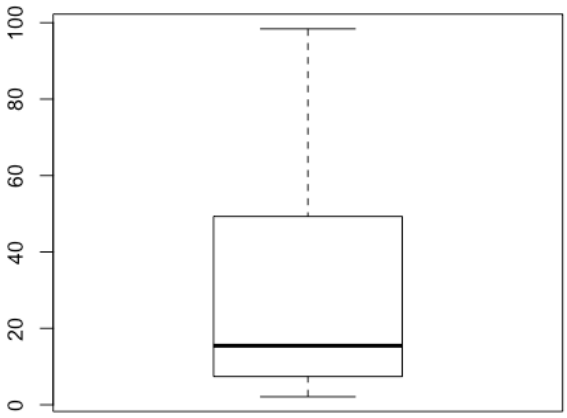
TotalLines (Ind5) normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
0.7419	2.2847	3.0959	3.1494	4.3228	4.7176	1.272582



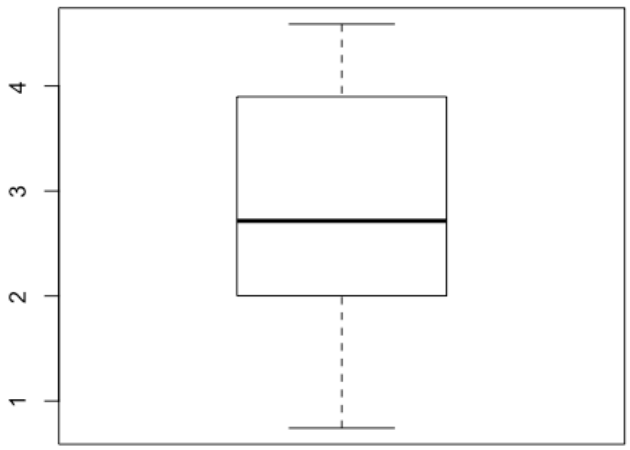
NewLines (Ind6) non normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
2.10	7.95	15.45	31.32	48.30	98.40	31.35087



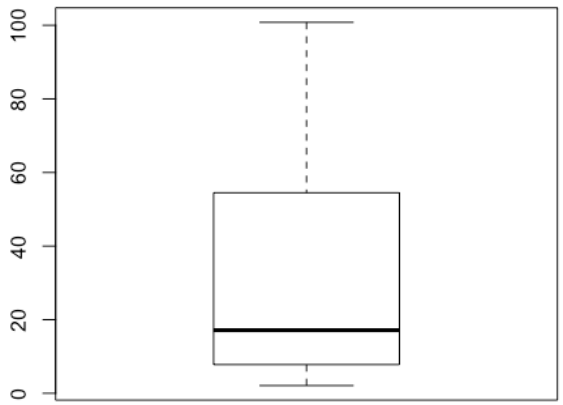
NewLines (Ind6) normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
0.7419	2.0666	2.7150	2.8393	3.8768	4.5890	1.231487



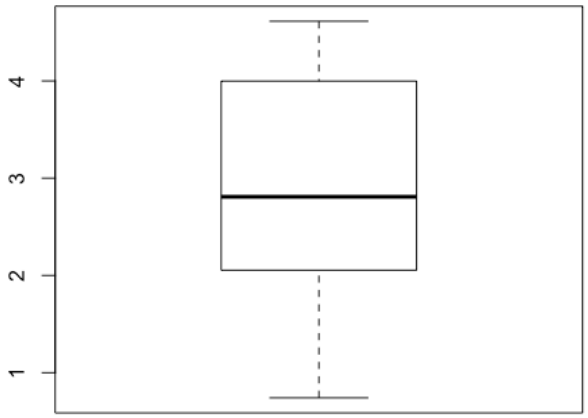
DevelopedLines (Ind7) non normalizzata

Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
2.100	8.275	17.150	35.256	52.500	100.800	35.10157

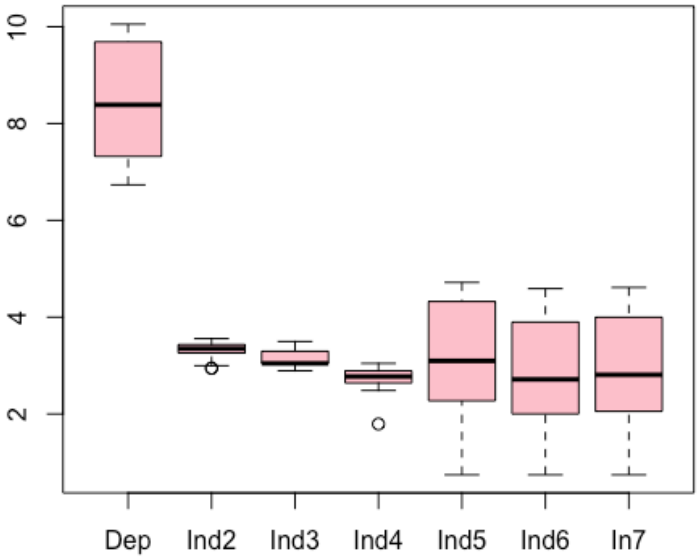


DevelopedLines (Ind7) normalizzata

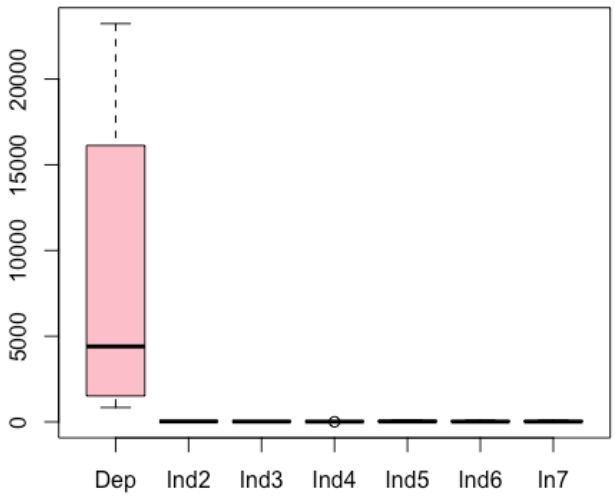
Min	Q1	Mediana	Media	Q3	Max	Dev.Standard
0.7419	2.1086	2.8087	2.9392	3.9585	4.6131	1.259931



Summary dati normalizzati



Summary dati non normalizzati



2. Costruzione e descrizione del modello di predizione

Per costruire un buon modello di predizione bisogna prima verificare le seguenti assunzioni per la regressione lineare:

- l'esistenza di una relazione lineare tra la variabile indipendente e la variabile dipendente (**linearità**);
- la varianza costante dei termini di errore per tutti i valori della variabile indipendente (**omoschedasticità**);
- la distribuzione normale degli errori (**normalità**);
- l'indipendenza statistica degli errori, in particolare, nessuna correlazione tra errori consecutivi (**indipendenza**).

Linearità

1) Pearson correlation

Ho calcolato il coefficiente di Pearson tra la mia unica variabile dipendente e ciascuna delle variabili indipendenti.
Lo scopo di tale calcolo sulle variabili è quello di verificare se vi è un'eventuale relazione di linearità tra esse.

Nella pratica si distinguono vari "tipi" di correlazione:

- se il coefficiente è > 0, allora le variabili sono correlate positivamente;
- se il coefficiente è = 0, allora le variabili non sono correlate;
- se il coefficiente è < 0, allora le variabili sono correlate negativamente.

Inoltre per la correlazione positiva (analogamente per la negativa), bisogna distinguere i seguenti casi:

1. se il coefficiente è compreso tra 0 e 0.3 allora si ha correlazione debole;
2. se il coefficiente è compreso tra 0.3 e 0.7 allora si ha correlazione moderata;
3. se il coefficiente è maggiore di 0.7 allora si ha correlazione forte.

Tale coefficiente l'ho calcolato nel seguente modo:

```
## pearson correlation
corre <- cor.test(Ind2, Dep, alternative = c("two.sided"), method = c("pearson"), exact = NULL, conf.level = 0.95)
corre
corre <- cor.test(Ind3, Dep, alternative = c("two.sided"), method = c("pearson"), exact = NULL, conf.level = 0.95)
corre
corre <- cor.test(Ind4, Dep, alternative = c("two.sided"), method = c("pearson"), exact = NULL, conf.level = 0.95)
corre
corre <- cor.test(Ind5, Dep, alternative = c("two.sided"), method = c("pearson"), exact = NULL, conf.level = 0.95)
corre
corre <- cor.test(Ind6, Dep, alternative = c("two.sided"), method = c("pearson"), exact = NULL, conf.level = 0.95)
corre
corre <- cor.test(Ind7, Dep, alternative = c("two.sided"), method = c("pearson"), exact = NULL, conf.level = 0.95)
corre
```

I risultati ottenuti sono i seguenti:

	Methodology	**Complexity**	**Experience**	**Total Lines**	**New Lines**	**Developed Lines**
EffortOreUomo	0.4269073 (corr. moderata)	-0.04403491 (corr. inversa)	-0.1057348 (corr. forte)	0.4491553 (corr. moderata)	0.393859 (corr. moderata)	0.4038885 (corr. moderata)

2) Test di multicollinarietà

Faccio una ulteriore verifica per vedere se vi è correlazione tra le variabili indipendenti:

```
##verifico se c'è correlazione tra le variabili indipendenti
cor(dati[,2:7],method = "pearson")
```

Questa funzione mi restituisce una matrice di correlazione tra le variabili indipendenti:

```
                Methodology Complexity Experience TotalLines   NewLines DevelopedLines
Methodology      1.00000000  0.1265869  0.3090462  0.1435968 0.07582448      0.1038577
Complexity       0.12658692  1.0000000  0.1820758  0.6096138 0.66863484      0.6612797
Experience       0.30904622  0.1820758  1.0000000  0.2174871 0.26727344      0.2447644
TotalLines       0.14359679  0.6096138  0.2174871  1.0000000 0.96735706      0.9832924
NewLines         0.07582448  0.6686348  0.2672734  0.9673571 1.00000000      0.9942744
DevelopedLines   0.10385770  0.6612797  0.2447644  0.9832924 0.99427438      1.0000000
```

Come possiamo osservare da questi valori, vi è alta dipendenza tra le variabili indipendenti *TotalLines, NewLines e DevelopedLines*, in quanto hanno come valore di correlazione un valore molto prossimo a 1.
Vedremo in seguito che dal modello verranno eliminate le due variabili *TotalLines* e *DevelopedLines* (quelle con p-value più alto) perché per un modello predittivo non va assolutamente bene avere correlazione tra variabili indipendenti.

3) Test di linearità

Verifico l'ipotesi di linearità eseguendo la regressione lineare tra ogni variabile indipendente e la variabile dipendente nel seguente modo:

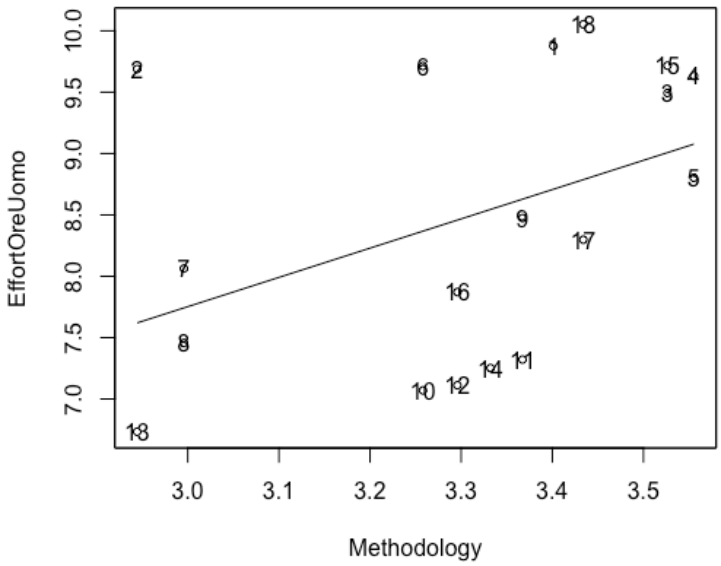
```
# test di linearità
sm.regression(Ind2, Dep, h=300,model="linear",test=TRUE, xlab="Methodology", ylab="EffortOreUomo")
text(Ind2, Dep)
sm.regression(Ind3, Dep, h=300,model="linear",test=TRUE, xlab="Complexity", ylab="EffortOreUomo")
text(Ind3, Dep)
sm.regression(Ind4, Dep, h=300,model="linear",test=TRUE, xlab="Experience", ylab="EffortOreUomo")
text(Ind4, Dep)
sm.regression(Ind5, Dep, h=300,model="linear",test=TRUE, xlab="TotalLines", ylab="EffortOreUomo")
text(Ind5, Dep)
sm.regression(Ind6, Dep, h=300,model="linear",test=TRUE, xlab="NewLines", ylab="EffortOreUomo")
text(Ind6, Dep)
sm.regression(Ind7, Dep, h=300,model="linear",test=TRUE, xlab="DevelopedLines", ylab="EffortOreUomo")
text(Ind7, Dep)
```

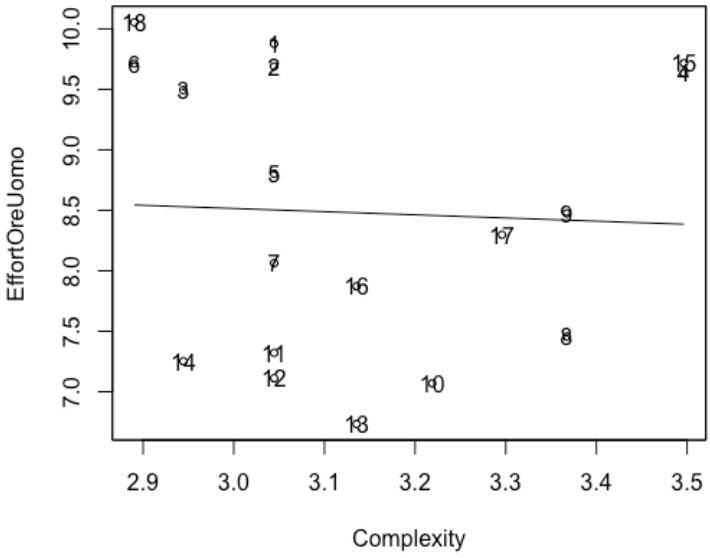
Ho ottenuto dei valori di significatività ed i risultati sono i seguenti:

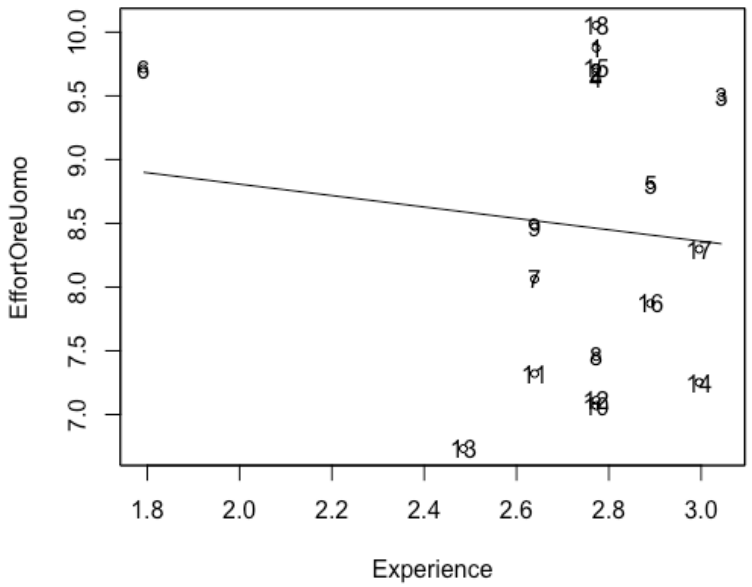
	Methodology	**Complexity**	**Experience**	**Total Lines**	**New Lines**	**Developed Lines**
EffortOreUomo	0.161	0.019	0.307	0.998	0.343	0.573

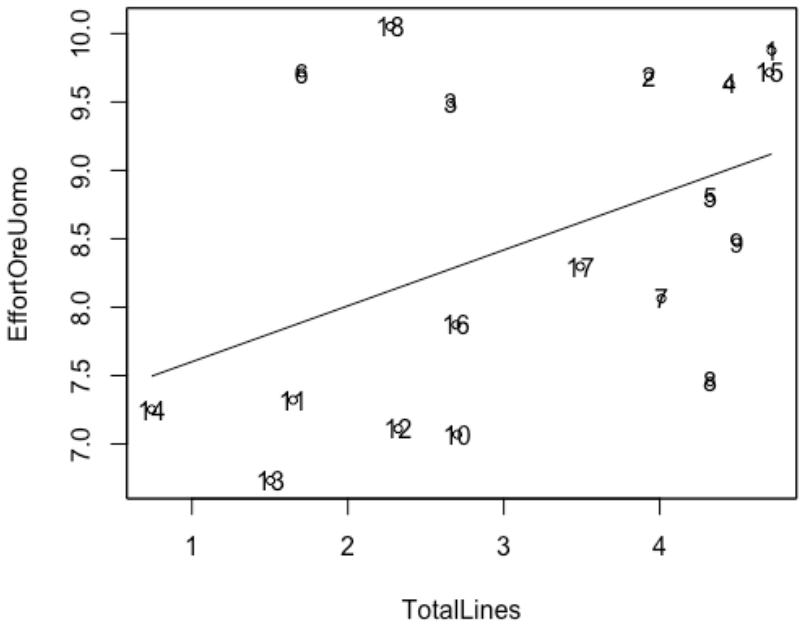
Come possiamo notare, la significatività per alcune coppie è bassa, mentre per *EffortOreUomo-TotalLines* abbiamo una significance molto alta (quasi prossima ad 1).

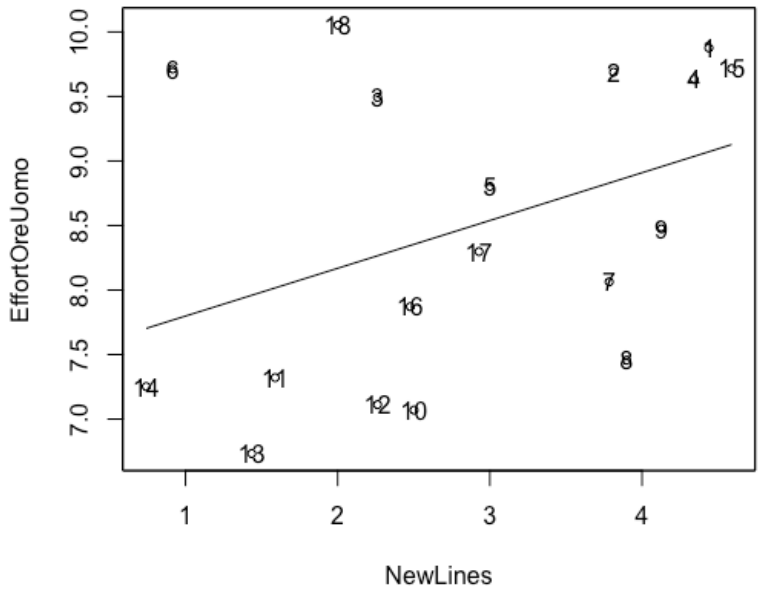
Vediamo graficamente nelle prossime pagine quanto appena detto:

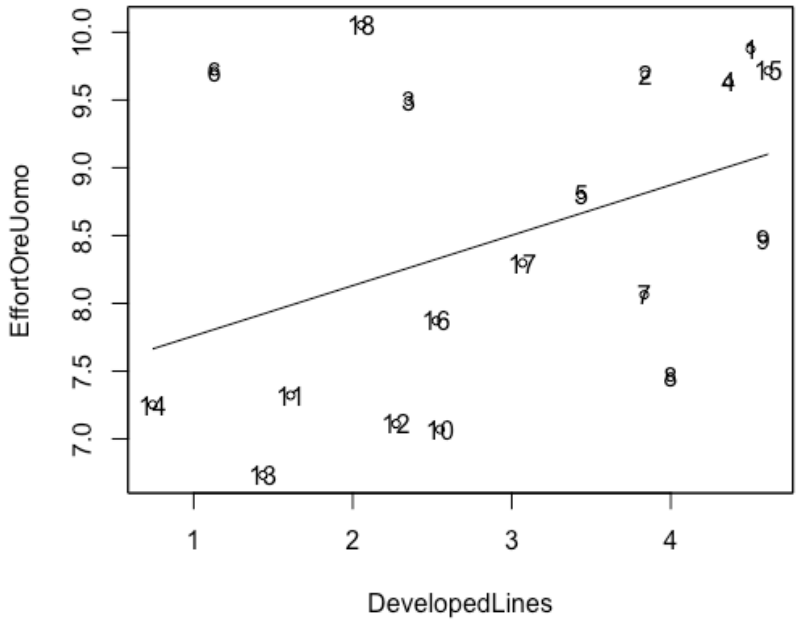

EffortOreUomo
10.0
9.5
9.0
8.5
8.0
7.5
7.0
3.0
3.1
3.2
3.3
3.4
3.5
Methodology


EffortOreUomo
10.0
9.5
9.0
8.5
8.0
7.5
7.0
2.9
3.0
3.1
3.2
3.3
3.4
3.5
Complexity


EffortOreUomo
10.0
9.5
9.0
8.5
8.0
7.5
7.0
1.8
2.0
2.2
2.4
2.6
2.8
3.0
Experience


EffortOreUomo
10.0
9.5
9.0
8.5
8.0
7.5
7.0
1
2
3
4
TotalLines

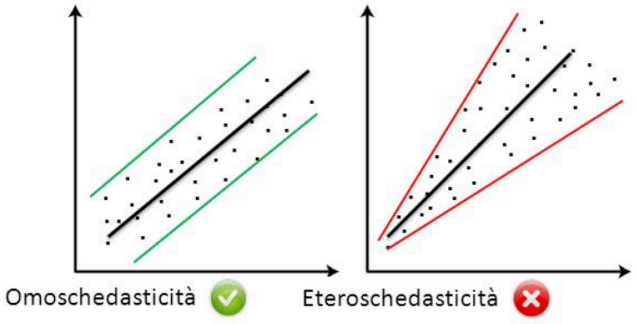

EffortOreUomo
NewLines
7.0
7.5
8.0
8.5
9.0
9.5
10.0
1
2
3
4


EffortOreUomo
DevelopedLines
7.0
7.5
8.0
8.5
9.0
9.5
10.0
1
2
3
4

4) Breusch-Pagan test

Tale test viene effettuato per verificare se la distribuzione dei dati è omoschedastica o eteroschedastica.
L'omoschedasticità è una condizione ideale nella quale si trova una funzione di dati rappresentabili graficamente come dispersi in maniera abbastanza omogenea al di sopra o al di sotto di una linea retta; l'eteroschedasticità, invece, è esattamente il concetto opposto e quindi i dati si distribuiscono non in maniera omogenea ma seguono un determinato pattern vicino la linea retta.

Esempio:



Ho effettuato il seguente test con la funzione bptest:

```
# perform Breusch-Pagan test per l'omoschedasticità
bptest(Dep ~ Ind2 + Ind3 + Ind4 + Ind5 + Ind6 + Ind7)
```

Ho ottenuto i seguenti risultati:

- BP = 8.3044
- df (grado di libertà) = 6
- p-value = 0.2166

Il p-value è > 0.05 e indica che l'ipotesi nulla (la varianza non cambia per i residui) può essere rigettata e mostra l'esistenza dell'eteroschedasticità, confermata ulteriormente dall'esecuzione del comando **gvlma** in R dopo aver costruito il modello con la Stepwise Linear Regression.

5) Stepwise linear regression

La Stepwise linear regression è un metodo per costruire modelli predittivi in cui vi sono più variabili indipendenti e la scelta delle variabili predittive viene effettuata da una procedura automatica.
Gli indicatori di bontà del modello di predizione sono i seguenti:

- R^2: è un indicatore per la correttezza del modello. Esso varia tra 0 e 1. Un valore prossimo ad 1 indica che la variazione della variabile dipendente è ben rappresentata dal modello costruito;
- F-statistic: è un indicatore usato in combinazione con il p-value per determinare se il modello si adatta bene (valore alto di F-statistic e p-value basso sono preferibili);
- p-value;
- t-value: Il t-value indica la significatività delle variabili. Se il t-value è maggiore di 1,5 la variabile considerata e significativa per il modello.

Ho costruito il modello nel seguente modo:

```
## stepwise linear regression
fm <- lm(data=dati, Dep   ~ Ind2 + Ind3  + Ind4 + Ind5 + Ind6 + Ind7)
summary(fm)
```

Ho ottenuto i seguenti risultati:

	Methodology	Complexity	Experience	Total Lines	New Lines	Developed Lines
t-value	3.128	-2.553	-2.298	0.570	1.481	-1.002

- Multiple R-squared = 0.6883
- Adjusted R-squared = 0.5183
- F-statistic = 4.048
- p-value = 0.02177

Possiamo dire che i Multiple R-squared e Adjusted R-squared sono modesti ma non prossimi ad 1, quindi questo significa che la variazione della variabile dipendente rispetto al modello è pressoché buona ma non ottima.
La F-statistic è nella norma (più alta che bassa) e il p-value è piccolo quindi posso dire che il modello si adatta abbastanza bene, ma non possiamo dire che sia un buon modello dati i valori nella loro totalità.
Il t-value ha valori non buoni, e l'unica variabile che assume un valore molto significativo è Methodology.

Come detto prima, posso avere la conferma dell'assunzione di eteroschedasticità dei miei dati tramite il seguente comando applicato al modello appena descritto:

```
gvlma(fm)
```

Ottengo il seguente output:

```
Call:
 gvlma(x = fm)

                     Value p-value                Decision
Global Stat        0.8712389  0.9287 Assumptions acceptable.
Skewness           0.0002601  0.9871 Assumptions acceptable.
Kurtosis           0.2970702  0.5857 Assumptions acceptable.
Link Function      0.2221202  0.6374 Assumptions acceptable.
Heteroscedasticity 0.3517885  0.5531 Assumptions acceptable.
```

Avendo costruito il modello, è importante andare a verificare quali variabili indipendenti sono utili per il mio modello e quali devono essere automaticamente scartare; questa procedura l'ho fatta tramite questo snippet di codice:

```
stepsel <-step(fm, direction="both")
### restituisce la formula con le variabili selezionate
formulas <-  stepsel$call$formula
formu <- toString(formulas)
cat("Formula: ", formu, "\n")
```

L'output è il seguente:

```
> stepsel <-step(fm, direction="both")
Start:  AIC=-2.74
Dep ~ Ind2 + Ind3 + Ind4 + Ind5 + Ind6 + Ind7

       Df Sum of Sq     RSS     AIC
- Ind5  1    0.2100  7.3103 -4.2195
- Ind7  1    0.6481  7.7484 -3.1719
<none>               7.1003 -2.7442
- Ind6  1    1.4156  8.5159 -1.4718
- Ind4  1    3.4098 10.5102  2.3155
- Ind3  1    4.2077 11.3080  3.6325
- Ind2  1    6.3158 13.4161  6.7095

Step:  AIC=-4.22
Dep ~ Ind2 + Ind3 + Ind4 + Ind6 + Ind7

       Df Sum of Sq     RSS     AIC
- Ind7  1    0.4954  7.8057 -5.0394
<none>               7.3103 -4.2195
- Ind6  1    1.2109  8.5212 -3.4607
+ Ind5  1    0.2100  7.1003 -2.7442
- Ind4  1    3.5612 10.8715  0.9239
- Ind3  1    5.1956 12.5060  3.4450
- Ind2  1    6.9042 14.2145  5.7501

Step:  AIC=-5.04
Dep ~ Ind2 + Ind3 + Ind4 + Ind6

       Df Sum of Sq     RSS     AIC
<none>               7.8057 -5.0394
+ Ind7  1    0.4954  7.3103 -4.2195
+ Ind5  1    0.0572  7.7484 -3.1719
- Ind4  1    3.0804 10.8861 -1.0519
- Ind3  1    4.9563 12.7620  1.8098
- Ind2  1    6.4743 14.2800  3.8328
- Ind6  1    9.3125 17.1182  7.0958
> ### restituisce la formula con le variabili selezionate
> formulas <-  stepsel$call$formula
> formu <- toString(formulas)
> cat("Formula: ", formu, "\n")
Formula:  ~, Dep, Ind2 + Ind3 + Ind4 + Ind6
```

Come possiamo vedere sono state eliminate le due variabili indipendenti TotalLines (Ind5) e DevelopedLines (Ind7) che non solo erano correlate tra loro (aspetto negativo) come abbiamo mostrato prima, ma la loro eliminazione ha contribuito al miglioramento dell'AIC del modello, in quanto più piccolo è e più buono è il modello di predizione.

6) Normalità dei residui

Bisogna verificare se gli errori sono distribuiti normalmente o meno; la normalità dei residui l'ho testata tramite lo Shapiro-Wilk test applicato ai residui:

```
# normalità dei residui ottenuti con il modello
shapiro.test(resid(fm)) ##sono normalmente distribuiti perchè p-value > 0.05
```

Eseguendo questa funzione, ottengo un **p-value pari a 0.6889**.
Poiché la mia ipotesi nulla è che i residui sono normalmente distribuiti e il mio p-value è > 0.05 non posso rifiutare tale ipotesi e quindi posso confermare che sono normalmente distribuiti.

7) Test di Durbin-Watson

Il test di Durbin-Watson viene utilizzato per verificare la presenza di correlazione seriale tra i residui. Il valore della statistica di Durbin-Watson varia da 0 a 4.
Come regola generale, i residui non sono correlati se l'indice di Durbin-Watson è circa 2 (intervallo accettabile è 1,50 - 2,50).
Ho effettuato tale test sui dati nel seguente modo:

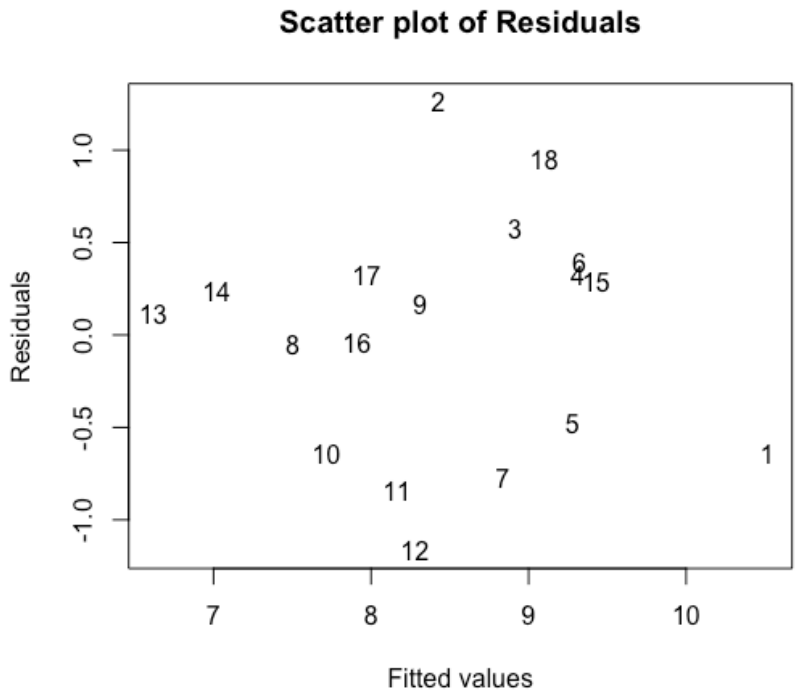
```
## test di Durbin-Watson
dw<-dwtest(fm,data=dati) ##sono correlati
```

L'output è il seguente:

```
        Durbin-Watson test

data:  fm
DW = 1.4855, p-value = 0.09841
```

Come possiamo vedere, il DW non ricade nell'intervallo accettabile e quindi posso già dire che i miei residui sono correlati tra loro.

Di seguito lo scatter-plot dei residui:



8) Distanza di Cook

I valori di distanza di Cook possono essere usati per identificare gli outlier (o punti influenti) del modello predittivo. Ogni punto che ha distanza superiore a 3 × (4 / n), dove n rappresenta il numero totale delle osservazioni, sono immediatamente rimossi dall'analisi dei dati. D'altra parte, i punti con distanza superiore a 4 / n ma inferiore a 3 × (4 / n) vengono rimossi e la stabilità del modello viene testato nuovamente osservando l'effetto della loro rimozione sul modello.
Se la bontà dell'adattamento del modello migliora (cioè un R-squared più alto) allora sono esclusi gli outlier.

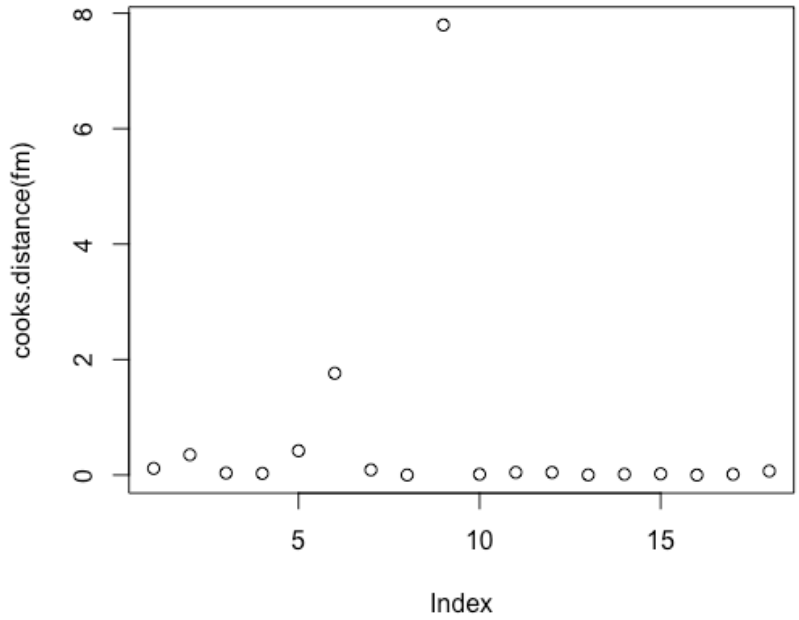
Nel mio caso, ho verificato la presenza degli outlier nel seguente modo:

```
#distanza di cook
cooks.distance(fm)
plot(cooks.distance(fm))
plot(density(cooks.distance(fm)))
```

Ho ottenuto il seguente risultato:

```
> cooks.distance(fm)
          1           2           3           4           5           6           7           8
1.151866e-01 3.509047e-01 3.560999e-02 2.836432e-02 4.206996e-01 1.763453e+00 8.976172e-02 6.762508e-04
          9          10          11          12          13          14          15          16
7.795961e+00 1.330762e-02 4.541842e-02 4.525876e-02 2.902384e-03 1.400821e-02 2.150870e-02 4.048412e-05
         17          18
1.320389e-02 6.880161e-02
```

Come possiamo notare, l'osservazione 6 e la osservazione 9 hanno la maggiore distanza di Cook.
Per conferma, faccio il plot:



Sia la osservazione 6 e sia la osservazione 9 hanno distanza > di 3 * (4/18), quindi potrei togliere direttamente questi due outlier dall'analisi perché ininfluenti, ma sia per motivi di tempo e anche per la mancata conoscenza della provenienza dei dati, ho preferito lasciarli inalterati.

3. Validazione del modello

La validazione, nel senso ampio del termine, corrisponde al processo di determinazione del grado in cui il modello corrisponde al sistema reale, o almeno rappresenta con precisione il documento di specifica del modello.

I due tipi di validazione che ho adottato sono:

- K-fold cross validation;
- Hold-out cross validation.

3.1 K-fold cross validation

La K-fold cross validation consiste nella suddivisione del dataset totale in k parti di uguale numerosità e, ad ogni passo, la k-esima parte del dataset diventa il validation dataset, mentre la restante parte costituisce il training dataset.
Così, per ognuna delle k parti si allena il modello, evitando quindi problemi di overfitting, ma anche di campionamento asimmetrico (e quindi affetto da bias) del training dataset, tipico della suddivisione del dataset in due sole parti. In altre parole, si suddivide il campione osservato in gruppi di egual numerosità, si esclude iterativamente un gruppo alla volta e lo si cerca di predire con i gruppi non esclusi, e ciò al fine di verificare la bontà del modello di predizione utilizzato.

Ho implementato questo tipo di validazione nel seguente modo:

```
## k-fold cross validation
# define training control
#k=8 in base a MAE e l'altro valore
train_control <- trainControl(method="cv", number=8)
# train the model
model <- train(EffortOreUomo ~ Methodology + Complexity + Experience + NewLines,
               data=dati, trControl=train_control, method="lm", na.action = na.pass)
model$resample
print(model)
```

Analizziamo l'output che segue:

```
18 samples
 4 predictor

No pre-processing
Resampling: Cross-Validated (8 fold)
Summary of sample sizes: 16, 16, 16, 16, 15, 16, ...
Resampling results:

  RMSE       Rsquared   MAE
  0.8693528  0.9617704  0.8218127
```

Ho scelto un k = 8 in quanto non solo mi ha migliorato il valore di R-squared (molto prossimo a 1) che mi descrive la bontà del modello e della predizione in termini di Effort, ma anche i valori di MAE (Mean Absolute Error) e RMSE (Root Mean Square Deviation) che devono essere il più piccoli possibili, ed infatti soddisfano tale caratteristica.

3.2 Hold-out cross validation

Nella Hold-out cross validation, vengono assegnati casualmente dati a due set d0 e d1, di solito chiamati rispettivamente training set e test set.
La dimensione di ciascuno dei set è arbitraria, sebbene in genere il test set sia più piccolo del training set, e l'obiettivo è quello di fare training su d0 e testing su d1.

Ho implementato questo tipo di validazione nel seguente modo:

```
## hold out cross validation
set.seed(1)
in_train <- createDataPartition(dati$EffortOreUomo, p = 4/5, list = FALSE)
training <- dati[ in_train,]
testing  <- dati[-in_train,]
nrow(dati)
nrow(training)
nrow(testing)
lda_fit <- train(EffortOreUomo ~ Methodology + Complexity + Experience + NewLines, data = dati, method = "lm")
lda_fit
lda_fit$resample
```

Analizziamo l'output che segue:

```
> nrow(training)
[1] 16
> nrow(testing)
[1] 2
> lda_fit <- train(EffortOreUomo ~ Methodology + Complexity
> lda_fit
Linear Regression

18 samples
 4 predictor

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 18, 18, 18, 18, 18, 18, ...
Resampling results:

  RMSE      Rsquared   MAE
  1.101938  0.4905087  0.9387246
```

Con questo tipo di validazione ho ottenuto dei risultati peggiori per il mio modello, infatti come possiamo notare ho un R-squared relativamente basso e anche gli errori sono cresciuti rispetto a quelli ottenuti con la validazione precedente.

4. Alcuni confronti

Il confronto che sto per andare a fare riguarda l'accuratezza ottenuta con il modello di predizione ottenuto e l'accuratezza della Media e della Mediana.
Ho selezionato 4 progetti dal dataset e ho predetto il loro l'effort con il modello.
Infine, ho valutato l'absolute residuals (AR = |actual effort - predicted effort|) confrontandolo con l'errore commesso nell'uso di media e mediana.

Di seguito i risultati:

Methodology	Complexity	Experience	NewLines	EffortOreUomo
3.40	3.04	2.77	4.439	9.88
3.37	3.37	2.64	4.127	8.47
3.30	3.14	2.89	2.477	7.87
3.43	3.30	3.00	2.929	8.30

Predizione	Errore	Errore media	Errore mediana
140826613.47	140807077.74733794	4.0592	4.45487
21025993.16	21021223.64453051	1.0090	1.08762
267980.63	265363.0644118125	1.83859	1.67531
647484.41	43460.5376061777	1.1960	1.08980

I risultati sono stati tutti riportati a valori reali applicando la funzione esponenziale lì dove era necessario.
Vediamo i calcoli in dettaglio:

1) Predizioni calcolate con il modello

Predizione 1 = exp((3.40 *3.3797 + 3.04 * (-3.6245) + 2.77 * (-1.8820) + 4.439 * 2.7207) + intercetta) =
= exp((3.40*3.3797 + 3.04 * (-3.6245) + 2.77 * (-1.8820) + 4.439 * 2.7207) + 11.4264) =
= exp ((11.49098 – 11.01848 – 5.21314 + 12.0771873) + 11.4265) =
= exp(18.76304) = 140826613.47

Predizione 2 = exp((3.37*3.3797 + 3.37 * (-3.6245) + 2.64 * (-1.8820) + 4.127 * 2.7207) + intercetta) =
= exp((11.389589 -12.214565 – 4.96848 + 11.2283289) + 11.4264) =
= exp(16.86127) = 21025993.16

Predizione 3 = exp((3.30*3.3797 + 3.14 * (-3.6245) + 2.89 * (-1.8820) + 2.477 * 2.7207) + intercetta) =
= exp((11.15301 – 11.38093 – 5.43898 + 6.7391739) + 11.4264) =
= exp(12.49867) = 267980.63

Predizione 4 = exp((3.43*3.3797 + 3.30 * (- 3.6245) + 3.00 * (-1.8820) + 2.929 * 2.7207) + intercetta) =
= exp((11.592371 – 11.96085 – 5.646 + 7.9689303) + 11.4264) =
= exp(13.38085) = 647484.41

2) Calcolo degli errori di predizione

Errore = abs(exp(effort) - predizione)

Errore 1 = abs(exp(9.88) - 140826613.47) = (19535.72266206552 - 140826613.47) = 140807077.74733794

Errore 2 = abs(exp(8.47) - 21025993.16) = (4769.515469491679 - 21025993.16) = 21021223.64453051

Errore 3 = abs(exp(7.87) - 267980.63) = (2617.565588187496 - 267980.63) = 265363.0644118125

Errore 4 = abs(exp(8.30) - 647484.41) = (4023.872393822313 - 647484.41) = 43460.5376061777

3) Calcolo degli errori della media

Errore media = exp(abs(media var.dipendente– effort))

Errore media 1 = exp(abs(8.479 - 9.88)) = 4.0592
Errore media 2 = exp(abs(8.479 - 8.47)) = 1.0090
Errore media 3 = exp(abs(8.479 - 7.87)) = 1.83859
Errore media 4 = exp(abs(8.479 - 8.30)) = 1.1960

4) Calcolo degli errori della mediana

Errore mediana= exp(abs(mediana var.dipendente – effort))

Errore mediana 1 = exp(abs(8.386 - 9.88)) = 4.45487
Errore mediana 1 = exp(abs(8.386 - 8.47)) = 1.08762
Errore mediana 1 = exp(abs(8.386 - 7.87)) = 1.67531
Errore mediana 1 = exp(abs(8.386 - 8.30)) = 1.08980

	Errore	**Errore media**	**Errore mediana**
Media	40534281.24847161	2.0256974999999997	2,0769
Deviazione Standard	58515992.404452	1.2136940905841	1.3936514249804

È nettamente visibile quanto l'errore predetto si discosti dalla baseline della media e mediana. Ci eravamo già resi conto della non bontà del modello di predizione dai risultati della stepwise linear regression.

Riferimenti

[1] Estimating the Effort to Develop Screen Mockups (http://elearning.informatica.unisa.it/el-platform/pluginfile.php/27095/mod_resource/content/1/Estimating%20the%20Effort%20to%20Develop%20Screen%20Mockups.pdf).

[2] How To Estimate Model Accuracy in R Using The Caret Package (https://machinelearningmastery.com/how-to-estimate-model-accuracy-in-r-using-the-caret-package/).

[3] Hold-out validation in R (https://stackoverflow.com/questions/22972854/how-to-implement-a-hold-out-validation-in-r).