UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA

Laurea triennale in Informatica

Fondamenti di Intelligenza Artificiale

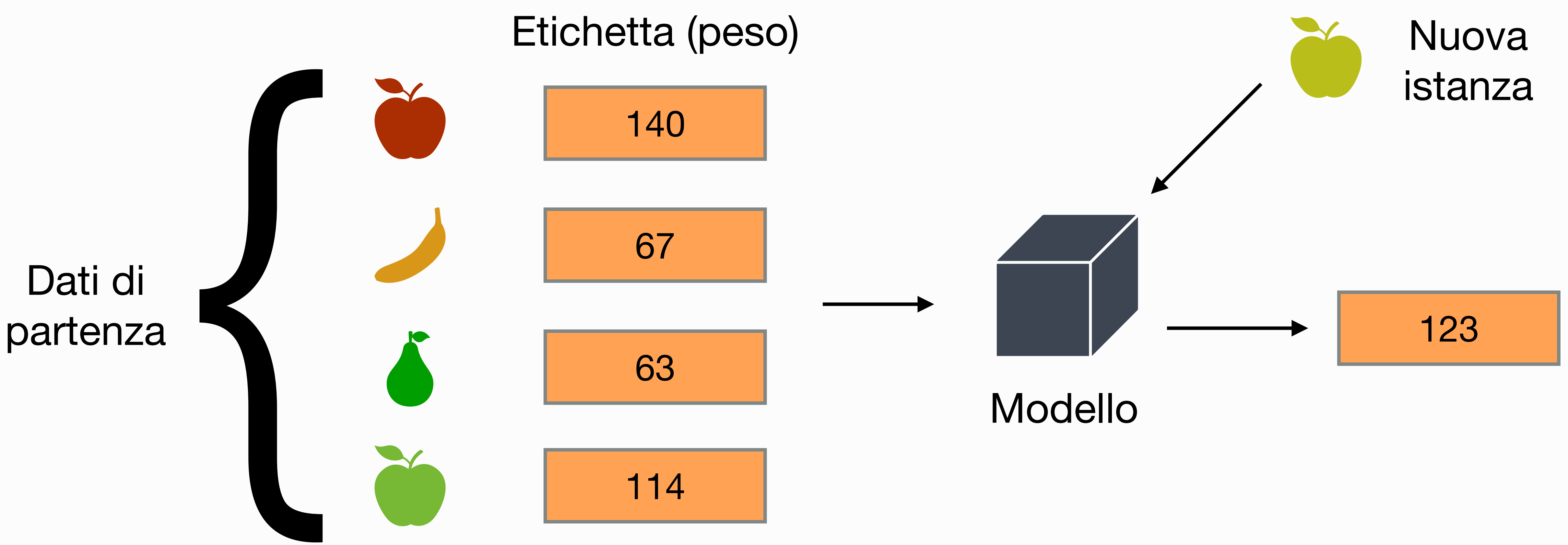
Lezione 17 - Regressione, regressori, e oltre

Regressione, Regressori, e Oltre

Problemi di regressione

Regressione: Task in cui l’obiettivo è predire il valore di una variabile numerica, chiamata variabile dipendente o di risposta, tramite l’utilizzo di un training set, ovvero un insieme di osservazioni per cui la variabile dipendente è nota.

I problemi di regressione sono istanze di problemi di apprendimento supervisionato.



Regressione, Regressori, e Oltre

Problemi di regressione

Regressione: Task in cui l'obiettivo è predire il valore di una variabile numerica, chiamata variabile dipendente o di risposta, tramite l'utilizzo di un training set, ovvero un insieme di osservazioni per cui la variabile dipendente è nota.

I problemi di regressione sono istanze di problemi di apprendimento supervisionato.

In maniera simile alla classificazione, un problema di regressione porta alla costruzione di un *modello*, ovvero di uno strumento che fa uso di un algoritmo di apprendimento, anche detto *regressore*, per predire i nuovi elementi sulla base del training set.

I regressori sono essenzialmente delle funzioni matematiche che cercano di descrivere i dati. Diversi regressori si distinguono tra di loro per via delle assunzioni fatte sui dati così come delle specifiche proprietà che portano alla regressione, ma anche del numero di variabili indipendenti (predittori) di cui disponiamo.

Come fatto per i problemi di classificazione, approfondiremo due regressori in particolare: la *regressione singola* e la *regressione multipla*. Più importante, però discuteremo come è possibile selezionare il *giusto* regressore in base al problema.

Anche per i problemi di regressione, è possibile parlare di *ensemble learning*, ovvero della combinazione di più regressori. Tuttavia, queste tecniche sono più complesse di quelle usate in classificazione, poiché non possono limitarsi a giudicare le classi generate, ma devono interpretare i valori numerici predetti da più modelli.

Regressione, Regressori, e Oltre

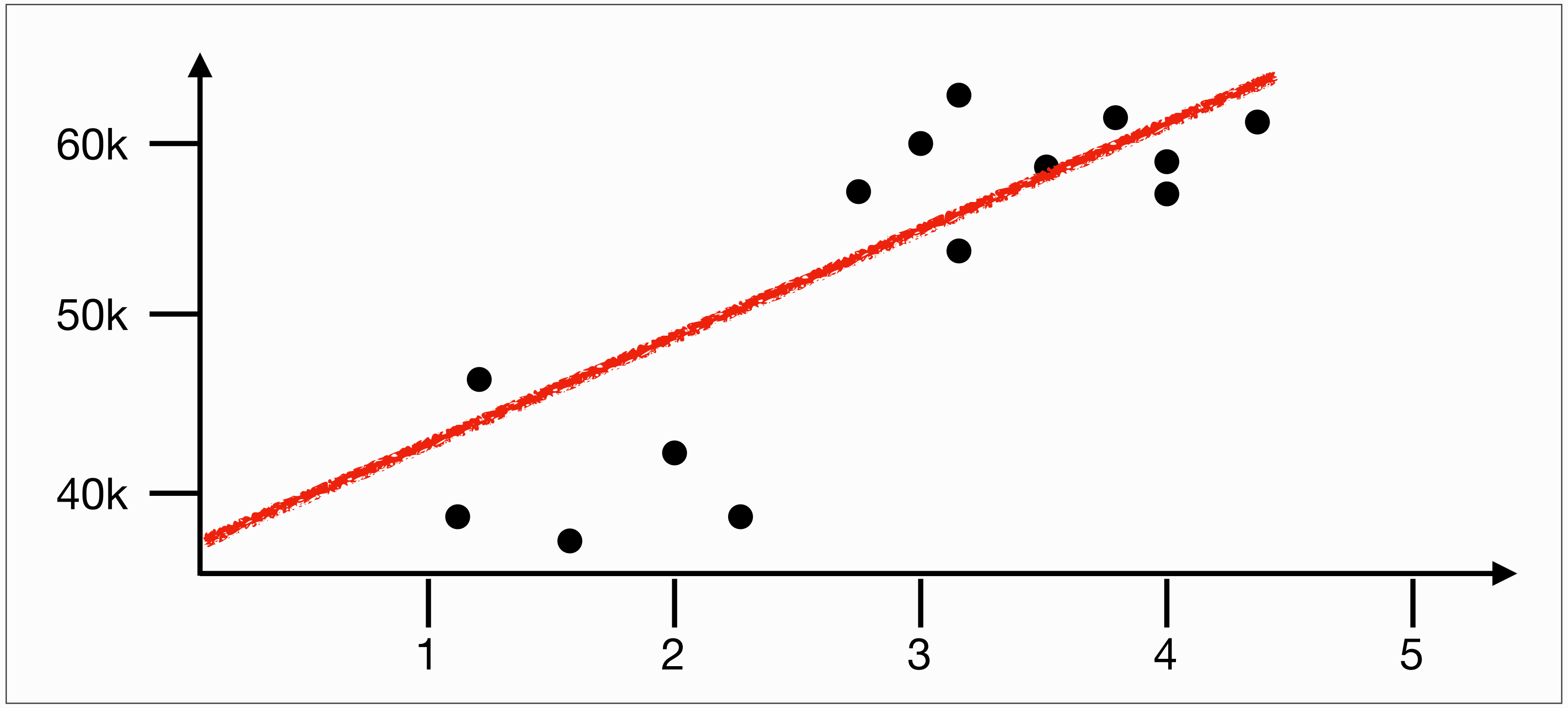
Regressione lineare singola e multipla

La differenza tra modelli singoli e multipli dipende dal numero di predittori che abbiamo a disposizione. Questo avrà un impatto sulla funzione che verrà generata.

Salario	Anni di esperienza
39.343	1,1
46.205	1,3
37.731	1,5
43.525	2,0
39.891	2,2
56.655	2,9
60.150	3,0
54.445	3,2
64.313	3,2
57.189	3,7
63.218	3,9
55.794	4,0
56.957	4,0
61.111	4,5

Supponiamo di dover predire il salario di un impiegato sulla base dei suoi anni di esperienza.

Questo è un problema risolvibile con una regressione lineare singola, in quanto abbiamo un unico predittore.



$$\tilde{y} = 32.375{,}85 + 6.995{,}31 \bullet anniDiEsperienza$$

La funzione verrà poi usata per predire nuovi dati.

Regressione, Regressori, e Oltre

Regressione lineare singola e multipla

Il punto quindi è quello di trovare una funzione che si adatti ai dati di training, cosicché possa essere efficace per predire il valore della variabile dipendente su nuovi dati.

In termini più pratici, generalizzando la funzione di regressione lineare, avremo:

$$\tilde{y} = \beta_0 + \beta_1 \bullet x_1$$

Questo indicherà l'inclinazione della retta

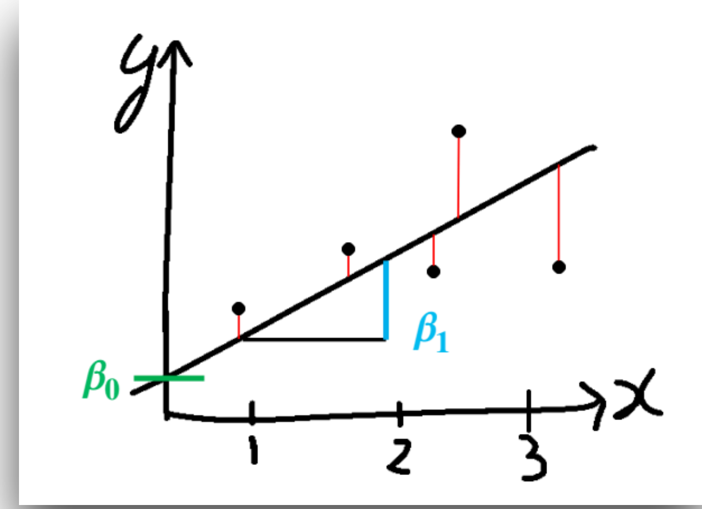
Per i più smemorati, l'intercetta β_0 indica il valore di y_i quando x_i è zero.

L'inclinazione indica la variazione di y_i quando x_i incrementa di una unità.

$$y = \tilde{y} + \epsilon$$

Questo indicherà il residuo

Il residuo è la differenza tra il valore reale ed il valore predetto dal modello di regressione.



Se la definizione di regressione lineare singola è abbastanza semplice, meno semplice è capire quando una retta si adatta bene ai dati di training.

Regressione, Regressori, e Oltre

Regressione lineare singola e multipla

Il punto quindi è quello di trovare una funzione che si adatti ai dati di training, cosicché possa essere efficace per predire il valore della variabile dipendente su nuovi dati.

In termini più pratici, generalizzando la funzione di regressione lineare, avremo:

$$\tilde{y} = \beta_0 + \beta_1 \bullet x_1$$

Questo indicherà l'inclinazione della retta

Per i più smemorati, l'intercetta β_0 indica il valore di y_i quando x_i è zero.

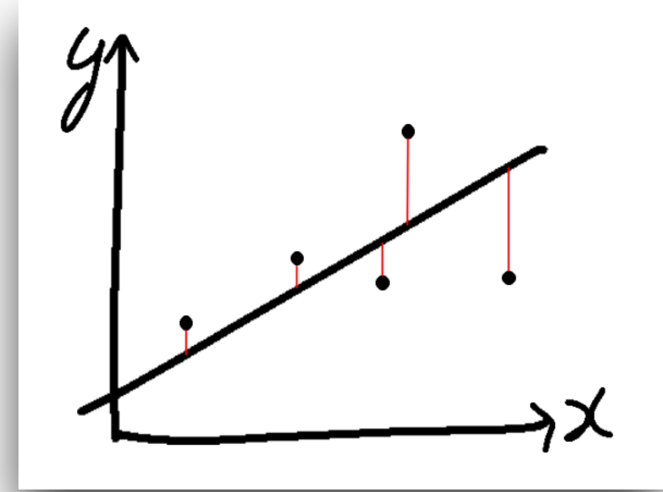
L'inclinazione indica la variazione di y_i quando x_i incrementa di una unità.

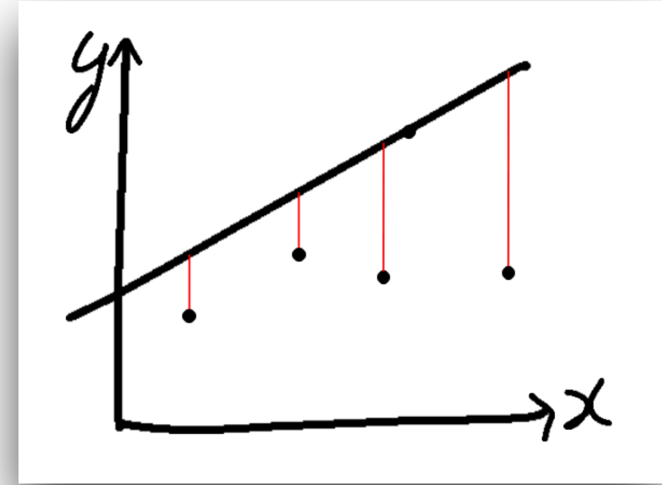
$$y = \tilde{y} + \epsilon$$

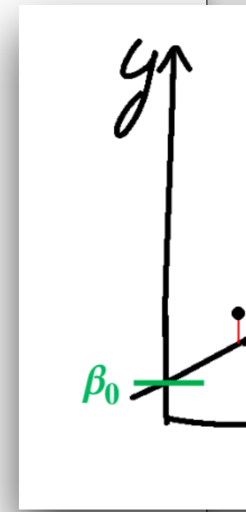
Questo indicherà il residuo

Il residuo è la differenza tra il valore reale ed il valore predetto dal modello di regressione.

Quale di queste due rette è migliore?







Regressione lineare singola e multipla

Il punto quindi è quello di trovare una funzione che si adatti ai dati di training, cosicché possa essere efficace per predire il valore della variabile dipendente su nuovi dati.

In termini più pratici, generalizzando la funzione di regressione lineare, avremo:

$$\tilde{y} = \beta_0 + \beta_1 \bullet x_1$$

Questo indicherà l'inclinazione della retta

Per i più smemorati, l'intercetta β_0 indica il valore di y_i quando x_i è zero.

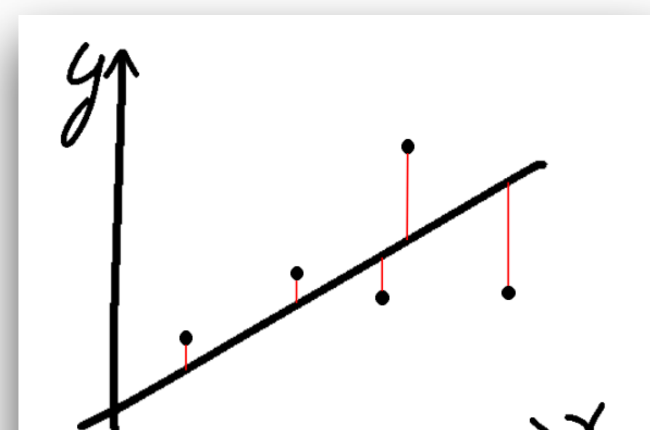
L'inclinazione indica la variazione di y_i quando x_i incrementa di una unità.

$$y = \tilde{y} + \epsilon$$

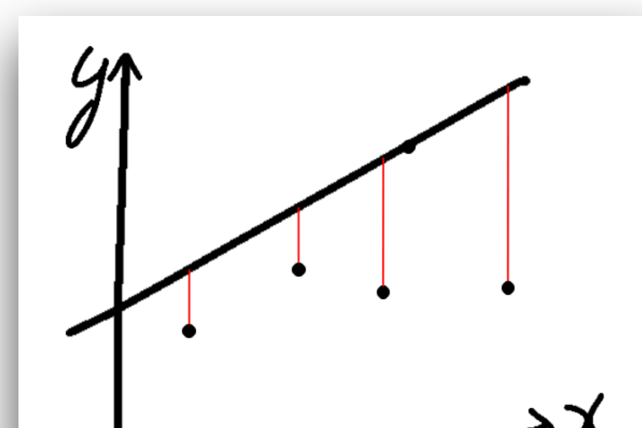
Questo indicherà il residuo

Il residuo è la differenza tra il valore reale ed il valore predetto dal modello di regressione.

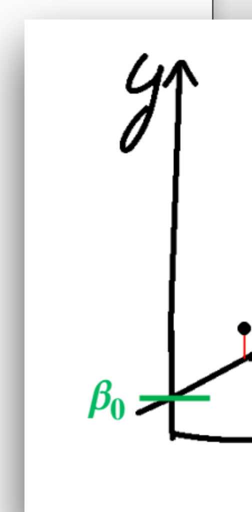
Quale di queste due rette è migliore?



Questa retta si adatta meglio



Qui la retta è lontana dai dati



Regressione lineare singola e multipla

A questo proposito, occorre nuovamente parlare di sperimentazione empirica e di metriche di valutazione.

Mean Absolute Error (MAE)

$$\frac{\sum_{i=1}^{n} |\tilde{y} - y|}{n}$$

La metrica MAE indica la differenza media osservata tra i valori predetti e i valori reali del test set.

Mean Squared Error (MSE)

$$\frac{\sum_{i=1}^{n} (\tilde{y} - y)^2}{n}$$

La metrica MSE indica l'errore quadratico medio commesso sui dati presenti nel test set.

Root Mean Squared Error (RMSE)

$$\sqrt{\frac{\sum_{i=1}^{n} (\tilde{y} - y)^2}{n}}$$

La metrica RMSE indica la radice quadrata dell'errore quadratico medio commesso sui dati presenti nel test set.

Regressione, Regressori, e Oltre

Regressione lineare singola e multipla

Il punto quindi è quello di trovare una funzione che si adatti ai dati di training, cosicché possa essere efficace per predire il valore della variabile dipendente su nuovi dati.

In termini più pratici, generalizzando la funzione di regressione lineare, avremo:

$$\tilde{y} = \beta_0 + \beta_1 \bullet x_1$$

$$y = \tilde{y} + \epsilon \quad -> \quad \epsilon = y - \tilde{y}$$

I modelli di regressione puntano a trovare la retta che si adatta meglio ai dati. Il più semplice criterio è quello della minimizzazione del residuo: più basso è il residuo, più la retta sarà vicina ai dati.

La somma dei residui è detta *funzione di perdita* (loss function). Molti delle tecniche di regressione mirano a minimizzare la funzione di perdita.

Calcolare i parametri della funzione

β_0

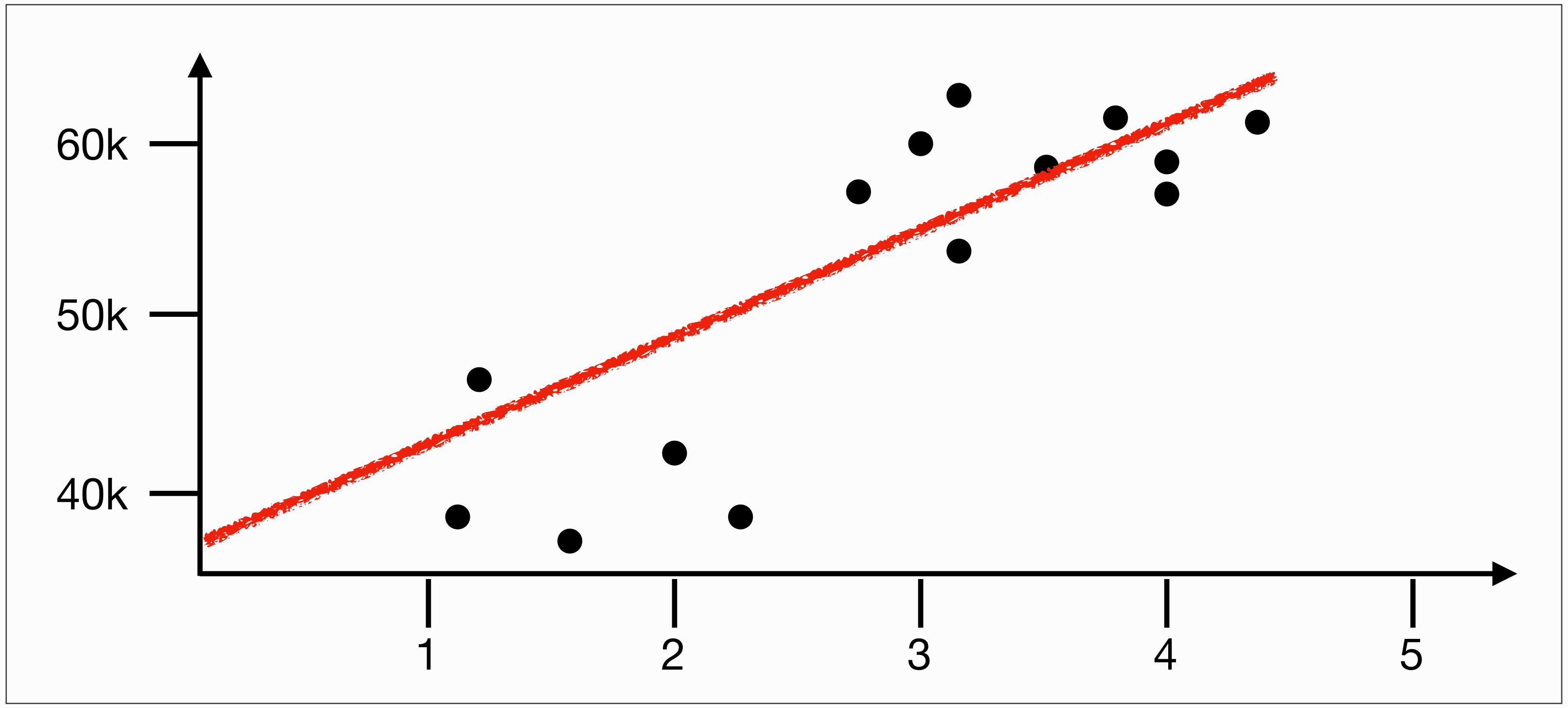
β_1

Il metodo più conosciuto è quello dei minimi quadrati (least squared), che avete visto o vedrete più nel dettaglio nei corsi di Calcolo Scientifico e Statistica ed Analisi dei Dati.

L'idea dietro al calcolo è relativamente semplice ed intuitiva.

Regressione, Regressori, e Oltre

Regressione lineare singola e multipla



1. Per ogni punto noto *(x, y)* dell'insieme *N* di punti, calcolare x^2 e xy.
2. Sommare tutti gli x, y, x^2, e xy. Questo ci porta ad avere: $\sum x$, $\sum y$, $\sum x^2$, $\sum xy$;
3. Calcolare l'inclinazione della retta, tramite la formula:

$$\beta_1 = \frac{N \bullet \sum xy - \sum x \bullet \sum y}{N \bullet \sum (x^2) - (\sum x)^2}$$

4. Calcolare l'intercetta della retta, tramite la formula: $\beta_0 = \dfrac{\sum y - \beta_1 \bullet \sum x}{N}$
5. Mettere tutto insieme nella versione: $\tilde{y} = \beta_0 + \beta_1 \bullet x_1$

Regressione lineare singola e multipla

Facciamo un esempio:

Salario	Anni di esperienza
39.343	1,1
46.205	1,3
37.731	1,5
43.525	2,0
39.891	2,2
56.655	2,9
60.150	3,0
54.445	3,2
64.313	3,2
57.189	3,7
63.218	3,9
55.794	4,0
56.957	4,0
61.111	4,5

Step 1 →

y	x	x^2	xy
39.343	1,1	1,21	43.277,3
46.205	1,3	1,69	60.066,5
37.731	1,5	2,25	56.596,5
43.525	2,0	4	87.050,0
39.891	2,2	4,84	87.760,2
56.655	2,9	8,41	164.299,5
60.150	3,0	9	180.450,0
54.445	3,2	10,24	172.224,0
64.313	3,2	10,24	205.801,6
57.189	3,7	13,69	211.599,3
63.218	3,9	15,21	246.550,2
55.794	4,0	16	223.176,0
56.957	4,0	16	227.828,0
61.111	4,5	20,25	274.999,5

Regressione lineare singola e multipla

Facciamo un esempio:

Salario	Anni di esperienza
39.343	1,1
46.205	1,3
37.731	1,5
43.525	2,0
39.891	2,2
56.655	2,9
60.150	3,0
54.445	3,2
64.313	3,2
57.189	3,7
63.218	3,9
55.794	4,0
56.957	4,0
61.111	4,5

Step 1 →

Step 2 →

y	x	x^2	xy
39.343	1,1	1,21	43.277,3
46.205	1,3	1,69	60.066,5
37.731	1,5	2,25	56.596,5
43.525	2,0	4	87.050,0
39.891	2,2	4,84	87.760,2
56.655	2,9	8,41	164.299,5
60.150	3,0	9	180.450,0
54.445	3,2	10,24	172.224,0
64.313	3,2	10,24	205.801,6
57.189	3,7	13,69	211.599,3
63.218	3,9	15,21	246.550,2
55.794	4,0	16	223.176,0
56.957	4,0	16	227.828,0
61.111	4,5	20,25	274.999,5
736.527	40,5	133,03	2.241.678,6

Regressione, Regressori, e Oltre

Regressione lineare singola e multipla

Facciamo un esempio:

y	x	x^2	xy
39.343	1,1	1,21	43.277,3
46.205	1,3	1,69	60.066,5
37.731	1,5	2,25	56.596,5
43.525	2,0	4	87.050,0
39.891	2,2	4,84	87.760,2
56.655	2,9	8,41	164.299,5
60.150	3,0	9	180.450,0
54.445	3,2	10,24	172.224,0
64.313	3,2	10,24	205.801,6
57.189	3,7	13,69	211.599,3
63.218	3,9	15,21	246.550,2
55.794	4,0	16	223.176,0
56.957	4,0	16	227.828,0
61.111	4,5	20,25	274.999,5
736.527	40,5	133,03	2.241.678,6

Step 3: $\beta_1 = \dfrac{N \bullet \sum xy - \sum x \bullet \sum y}{N \bullet \sum (x^2) - (\sum x)^2}$

$$\beta_1 = \frac{14 \bullet 2.241.678 - 40{,}5 \bullet 736.527}{14 \bullet 133{,}03 - 1.640{,}25}$$

$$\beta_1 = 6.995{,}31$$

Regressione lineare singola e multipla

Facciamo un esempio:

y	x	x^2	xy
39.343	1,1	1,21	43.277,3
46.205	1,3	1,69	60.066,5
37.731	1,5	2,25	56.596,5
43.525	2,0	4	87.050,0
39.891	2,2	4,84	87.760,2
56.655	2,9	8,41	164.299,5
60.150	3,0	9	180.450,0
54.445	3,2	10,24	172.224,0
64.313	3,2	10,24	205.801,6
57.189	3,7	13,69	211.599,3
63.218	3,9	15,21	246.550,2
55.794	4,0	16	223.176,0
56.957	4,0	16	227.828,0
61.111	4,5	20,25	274.999,5
736.527	40,5	133,03	2.241.678,6

Step 3: $\beta_1 = \dfrac{N \bullet \sum xy - \sum x \bullet \sum y}{N \bullet \sum (x^2) - (\sum x)^2}$

$$\beta_1 = \frac{14 \bullet 2.241.678 - 40,5 \bullet 736.527}{14 \bullet 133,03 - 1.640,25}$$

$$\beta_1 = 6.995,31$$

Step 4: $\beta_0 = \dfrac{\sum y - \beta_1 \bullet \sum x}{N}$

$$\beta_0 = \frac{736.572 - (6.995,31 \bullet 40,5)}{14}$$

$$\beta_0 = 32.375,85$$

Regressione, Regressori, e Oltre

Regressione lineare singola e multipla

Facciamo un esempio:

y	x	x^2	xy
39.343	1,1	1,21	43.277,3
46.205	1,3	1,69	60.066,5
37.731	1,5	2,25	56.596,5
43.525	2,0	4	87.050,0
39.891	2,2	4,84	87.760,2
56.655	2,9	8,41	164.299,5
60.150	3,0	9	180.450,0
54.445	3,2	10,24	172.224,0
64.313	3,2	10,24	205.801,6
57.189	3,7	13,69	211.599,3
63.218	3,9	15,21	246.550,2
55.794	4,0	16	223.176,0
56.957	4,0	16	227.828,0
61.111	4,5	20,25	274.999,5
736.527	40,5	133,03	2.241.678,6

Step 3: $\beta_1 = \dfrac{N \bullet \sum xy - \sum x \bullet \sum y}{N \bullet \sum (x^2) - (\sum x)^2}$

$$\beta_1 = \frac{14 \bullet 2.241.678 - 40{,}5 \bullet 736.527}{14 \bullet 133{,}03 - 1.640{,}25}$$

$$\beta_1 = 6.995{,}31$$

Step 4: $\beta_0 = \dfrac{\sum y - \beta_1 \bullet \sum x}{N}$

$$\beta_0 = \frac{736.572 - (6.995{,}31 \bullet 40{,}5)}{14}$$

$$\beta_0 = 32.375{,}85$$

Step 5:

$$\tilde{y} = 32.375{,}85 + 6.995{,}31 \bullet anniDiEsperienza$$

Regressione lineare singola e multipla

Applichiamo adesso la funzione sui dati di partenza:

$$\tilde{y} = 32.375{,}85 + 6.995{,}31 \bullet anniDiEsperienza$$

Salario	Anni di esperienza	Salario stimato	Errore
39.343	1,1	40.070,70	-727,7
46.205	1,3	41.469,75	4.735,25
37.731	1,5	42.868,82	-5.137,82
43.525	2,0	46.366,47	-2.841,47
39.891	2,2	47.765,53	-4.736.662
56.655	2,9	52.662,25	3.992,75
60.150	3,0	53.361,78	6.788,22
54.445	3,2	54.760,84	-315,84
64.313	3,2	54.760,84	9.552,16
57.189	3,7	58.258,50	-1.069,5
63.218	3,9	59.657,56	3.560,44
55.794	4,0	60.357,09	-4.563,09
56.957	4,0	60.357,09	-3.400,09
61.111	4,5	63.854,75	-2.743,75

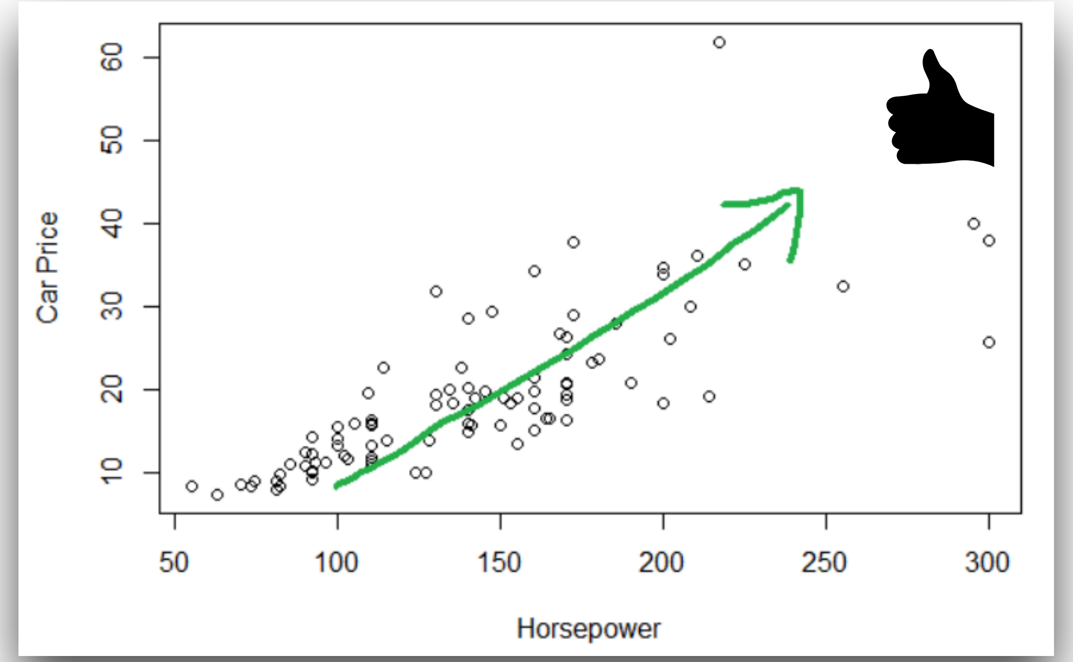
Sulla base degli errori potremo poi calcolare le metriche di valutazione (MAE, MSE, ecc.) e confrontare diversi modelli.

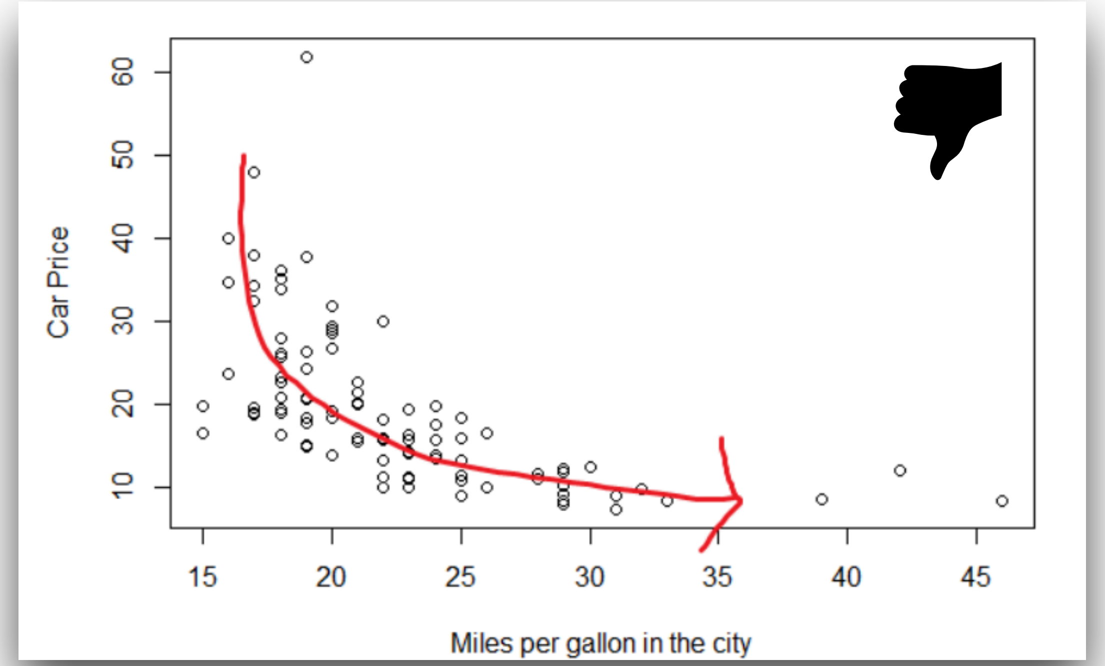
Regressione, Regressori, e Oltre

Regressione lineare singola e multipla

Sebbene sia semplice e talvolta efficace, la regressione lineare può essere utilizzata solo in determinati contesti. In particolare, il suo utilizzo assume:

Linearità dei dati. La relazione tra variabile indipendente X e variabile dipendente Y deve essere lineare, ovvero può essere rappresentata tramite una funzione lineare.



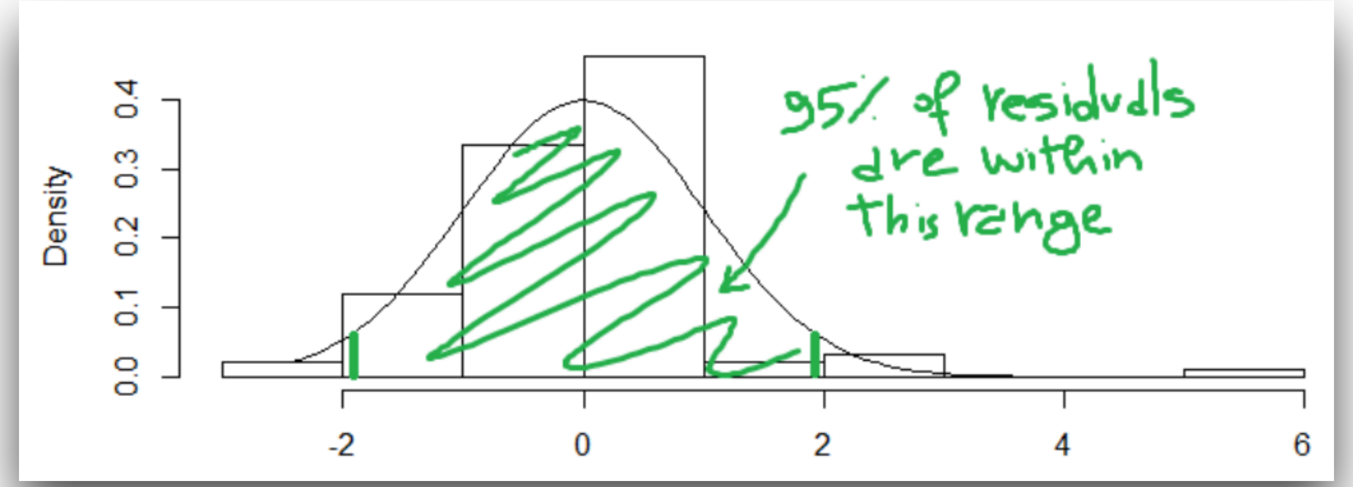


Regressione lineare singola e multipla

Sebbene sia semplice e talvolta efficace, la regressione lineare può essere utilizzata solo in determinati contesti. In particolare, il suo utilizzo assume:

Linearità dei dati. La relazione tra variabile indipendente X e variabile dipendente Y deve essere lineare, ovvero può essere rappresentata tramite una funzione lineare.

Normalità dei residui. Gli errori residui devono essere normalmente distribuiti.



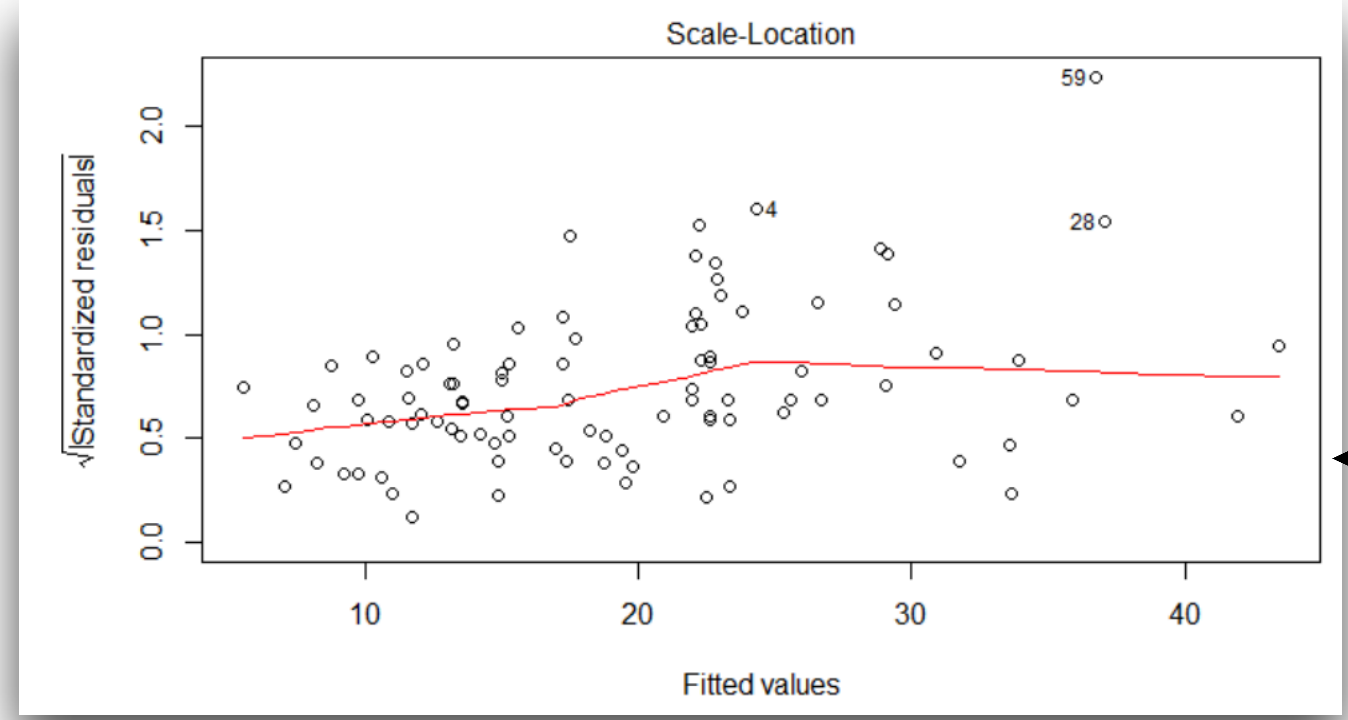
Regressione lineare singola e multipla

Sebbene sia semplice e talvolta efficace, la regressione lineare può essere utilizzata solo in determinati contesti. In particolare, il suo utilizzo assume:

Linearità dei dati. La relazione tra variabile indipendente X e variabile dipendente Y deve essere lineare, ovvero può essere rappresentata tramite una funzione lineare.

Normalità dei residui. Gli errori residui devono essere normalmente distribuiti.

Omoschedasticità. Gli errori residui devono avere una varianza costante.



Questa può essere verificata andando a plottare i residui standardizzati vs i valori predetti.

Se la proprietà è soddisfatta, vedrete un trend orizzontale piuttosto che punti sparsi nello spazio.

Regressione lineare singola e multipla

Sebbene sia semplice e talvolta efficace, la regressione lineare può essere utilizzata solo in determinati contesti. In particolare, il suo utilizzo assume:

Linearità dei dati. La relazione tra variabile indipendente X e variabile dipendente Y deve essere lineare, ovvero può essere rappresentata tramite una funzione lineare.

Normalità dei residui. Gli errori residui devono essere normalmente distribuiti.

Omoschedasticità. Gli errori residui devono avere una varianza costante.

Indipendenza degli errori. Gli errori residui devono essere indipendenti per ogni valore di X. Un test statistico particolarmente utile è noto come *Durbin-Watson*: quando gli errori sono indipendenti, il valore del test sarà vicino a 2.

Regressione, Regressori, e Oltre

Regressione lineare singola e multipla

Sebbene sia semplice e talvolta efficace, la regressione lineare può essere utilizzata solo in determinati contesti. In particolare, il suo utilizzo assume:

Linearità dei dati. La relazione tra variabile indipendente X e variabile dipendente Y deve essere lineare, ovvero può essere rappresentata tramite una funzione lineare.

Normalità dei residui. Gli errori residui devono essere normalmente distribuiti.

Omoschedasticità. Gli errori residui devono avere una varianza costante.

Indipendenza degli errori. Gli errori residui devono essere indipendenti per ogni valore di X. Un test statistico particolarmente utile è noto come *Durbin-Watson*: quando gli errori sono indipendenti, il valore del test sarà vicino a 2.

Quando si parla di statistica, il linguaggio di programmazione ed ambiente di sviluppo R è quello più appropriato, poiché implementa già molte delle funzioni e dei test menzionati.

Giusto come esempio, il modello di esempio sarebbe implementato come:

```
regressore <- lm(salario~anniDiEsperienza, data=dataset)
```

Una guida per l'implementazione della regressione in R è disponibile qui: https://www.scribbr.com/statistics/linear-regression-in-r/

Regressione lineare singola e multipla

Ma cosa succede se non ho un'unica variabile indipendente? In questo caso, parliamo di regressione lineare multipla.

$$\tilde{y} = \beta_0 + \beta_1 \bullet x_1 + \beta_2 \bullet x_2 + \ldots + \beta_m \bullet x_m$$

$$y = \beta_0 + \beta_1 \bullet x_1 + \beta_2 \bullet x_2 + \ldots + \beta_m \bullet x_m + \epsilon$$

I parametri rappresentano l'effetto che la variazione di un'unita delle variabili indipendenti hanno sulla variabile dipendente

Come nel caso precedente, il metodo dei minimi quadrati rappresenta il modello più semplice di risoluzione di un problema di regressione multipla.

La differenza sta nel fatto che verrà utilizzata una forma matriciale per rappresentare le variabili dipendenti ed indipendenti, oltre che non vorremo più cercare una singola retta di regressione, ma un piano che meglio interpola i dati di training.

La cosa importante da sapere è che per la regressione lineare multipla valgono gli stessi vincoli di linearità e omoschedasticità della regressione lineare singola.

Altra cosa fondamentale: avendo più variabili, potremmo ricadere nel problema della multicollinearità —> è perciò importante eliminare le variabili ridondanti!

Machine learning: Cosa viene dopo?

Nella ultime lezioni, abbiamo visto come risolvere problemi di classificazione e regressione. Ma il machine learning è anche (e soprattutto?) altro…

Sebbene questo sia un corso di fondamenti di intelligenza artificiale, è bene sapere cosa c'è al di là dei modelli di base che abbiamo trattato - anche perché non tutto può essere risolto con i modelli tradizionali.

Considerate questa parte come un **NON argomento di esame**: l'obiettivo è incuriosirvi, non appesantirvi ulteriormente!

Apprendimento per rinforzo

Deep Learning

Quantum Machine Learning

Oggi come oggi, sentite costantemente parlare di deep learning e, in effetti, i modelli di deep learning sono molto più potenti dei modelli classici.

Ma non c'è solo il deep learning! L'apprendimento per rinforzo, l'active learning e altro rappresentano dei miglioramenti che possono risolvere problemi più efficientemente dei modelli di deep learning.

Infine, nei prossimi anni la parola "deep learning" verrà probabilmente sostituita con la parola "quantum machine learning"…

Apprendimento per Rinforzo

La caratteristica principale dell'apprendimento per rinforzo sta nel fatto di ricevere feedback, in forma di ricompense (reward), durante l'esecuzione dell'agente.

L'idea del rinforzo sta nell'utilizzo di tecniche di esplorazione dell'ambiente per il miglioramento della conoscenza già acquisita —> *Cosa ci ricorda tutto questo?*

L'ottimizzazione! L'apprendimento per rinforzo si avvicina all'apprendimento umano poiché non si basa solamente sulle etichette assegnate dal progettista, ma è in grado di ottimizzare tali etichette sulla base di quanto ha imparato.

Nella pratica, ogni azione ha un impatto sull'ambiente e ogni ambiente fornisce una ricompensa che guida l'algoritmo di apprendimento —> se l'algoritmo farà qualcosa di positivo, allora sarà premiato; se sbaglierà, allora la sua ricompensa sarà minore, cosicché possa capire il suo errore ed aggiornare la sua conoscenza.

> **Apprendimento per rinforzo:** Task in cui l'obiettivo è imparare i comportamenti corretti partendo da una conoscenza pregressa ma migliorandola iterativamente tramite interazioni con l'ambiente e ottimizzazione di una funzione reward.

In altri termini, il modello è associato ad una *funzione obiettivo* che deve essere massimizzata. La funzione obiettivo è una rappresentazione numerica delle azioni dell'agente sull'ambiente.

Ma come passiamo dalla funzione obiettivo al miglioramento?

Apprendimento per Rinforzo

Per capirlo, torniamo al problema di decidere se andare a giocare o meno a tennis sulla base delle condizioni climatiche.

Giocare	Meteo	Temperatura	Umidità
NO	Soleggiato	Caldo	Elevata
NO	Soleggiato	Caldo	Elevata
SI	Nuvoloso	Caldo	Elevata
SI	Piovoso	Mite	Elevata
SI	Piovoso	Freddo	Normale
NO	Piovoso	Freddo	Normale
SI	Nuvoloso	Freddo	Normale
NO	Soleggiato	Mite	Elevata
SI	Soleggiato	Freddo	Normale
SI	Piovoso	Mite	Normale
SI	Soleggiato	Mite	Normale
SI	Nuvoloso	Mite	Elevata
SI	Nuvoloso	Caldo	Normale
NO	Piovoso	Mite	Elevata

Al seguente input:

Meteo = Pioggia; Temperatura = Caldo; Umidità = Elevata.

un semplice classificatore bayesiano risponderebbe con “NO”.

Ma se, invece, la risposta corretta fosse “SI”, allora potremmo assegnare un punteggio negativo al comportamento del modello:

reward = -10;

Che implicazioni ha questo punteggio?

La cosa più semplice da pensare sarebbe quella di modificare l’etichetta da “NO” a “SI” nei casi in cui il dataset ha, come predittori, gli stessi che hanno causato l’errore…

Ma possiamo subito renderci conto che questa non è un’opzione valida, per due valide ragioni.

Regressione, Regressori, e Oltre

Apprendimento per Rinforzo

Per capirlo, torniamo al problema di decidere se andare a giocare o meno a tennis sulla base delle condizioni climatiche.

Giocare	Meteo	Temperatura	Umidità
NO	Soleggiato	Caldo	Elevata
NO	Soleggiato	Caldo	Elevata
SI	Nuvoloso	Caldo	Elevata
SI	Piovoso	Mite	Elevata
SI	Piovoso	Freddo	Normale
NO	Piovoso	Freddo	Normale
SI	Nuvoloso	Freddo	Normale
NO	Soleggiato	Mite	Elevata
SI	Soleggiato	Freddo	Normale
SI	Piovoso	Mite	Normale
SI	Soleggiato	Mite	Normale
SI	Nuvoloso	Mite	Elevata
SI	Nuvoloso	Caldo	Normale
NO	Piovoso	Mite	Elevata

Al seguente input:

Meteo = Pioggia; Temperatura = Caldo; Umidità = Elevata.

un semplice classificatore bayesiano risponderebbe con “NO”.

Già in questo piccolo esempio, non esiste nessuna entry che corrisponde esattamente alla decisione presa!

Più in generale, può mai bastare un errore per modificare l’intera conoscenza acquisita fino a quel momento?

quella di modificare l’etichetta da “NO” a “SI” nei casi in cui il dataset ha, come predittori, gli stessi che hanno causato l’errore…

Ma possiamo subito renderci conto che questa non è un’opzione valida, per due valide ragioni.

Apprendimento per Rinforzo

Senza entrare troppo nel dettaglio, potremmo associare ad ogni azione una probabilità che indichi quanto questa possa portare ad un miglioramento della ricompensa.

Giocare	Meteo	Temperatura	Umidità
NO	Soleggiato	Caldo	Elevata
NO	Soleggiato	Caldo	Elevata
SI	Nuvoloso	Caldo	Elevata
SI	Piovoso	Mite	Elevata
SI	Piovoso	Freddo	Normale
NO	Piovoso	Freddo	Normale
SI	Nuvoloso	Freddo	Normale
NO	Soleggiato	Mite	Elevata
SI	Soleggiato	Freddo	Normale
SI	Piovoso	Mite	Normale
SI	Soleggiato	Mite	Normale
SI	Nuvoloso	Mite	Elevata
SI	Nuvoloso	Caldo	Normale
NO	Piovoso	Mite	Elevata

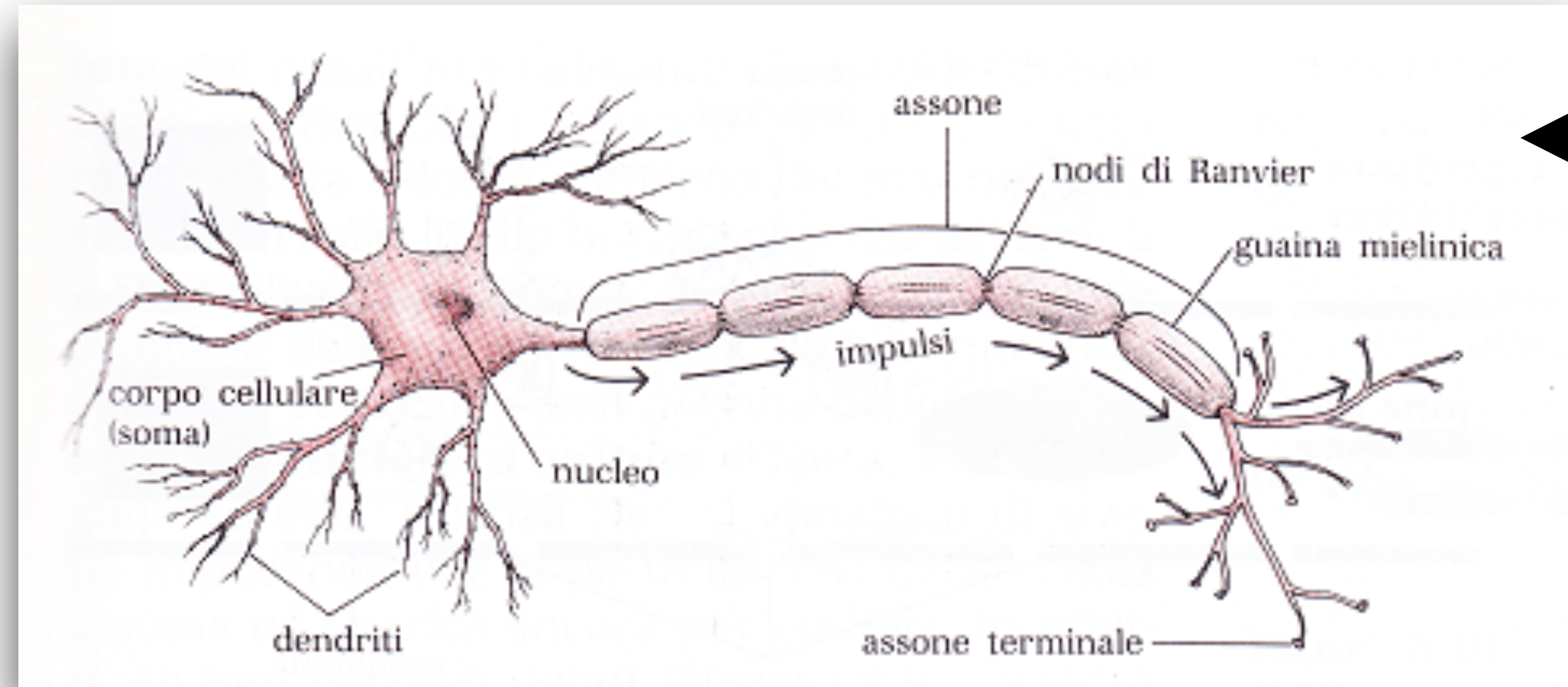
Sulla base di questa informazione, potremmo poi definire una *politica di azione*, ovvero una strategia di aggiornamento delle probabilità e delle azioni che l'agente dovrà effettuare.

Ad esempio, una possibile politica di azione prevede l'utilizzo degli algoritmi greedy (ricordate?), cosicché l'agente effettui l'azione che massimizzi, in un dato momento, la probabilità di reward.

L'aggiornamento delle probabilità è spesso effettuato tramite i cosiddetti processi decisionali di Markov, i metodi di Monte Carlo o attraverso i metodi statistici di apprendimento differenziale temporale.

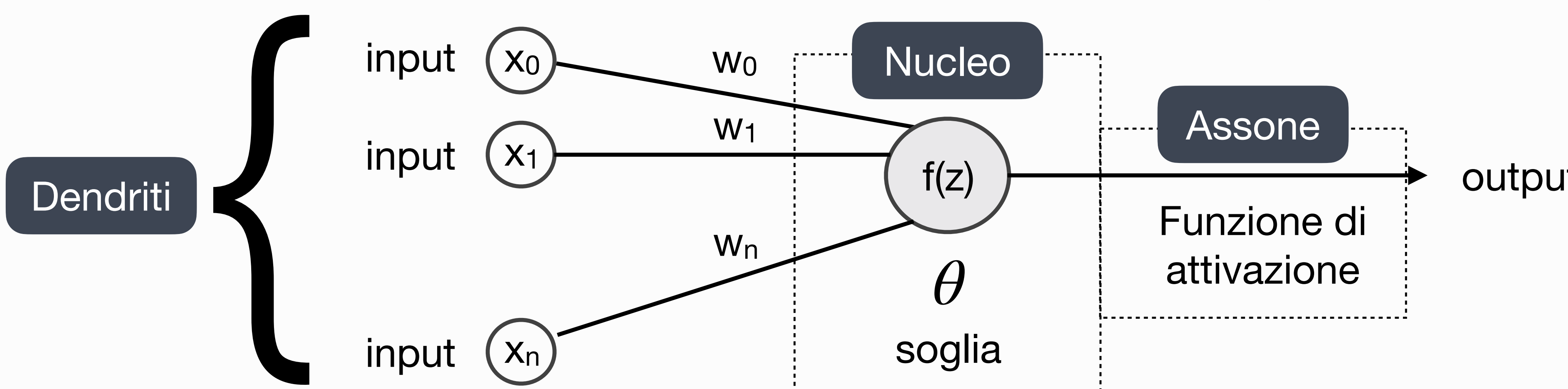
Questi modelli hanno un costo notevolmente maggiore a quelli di base, ma sono di norma più efficaci ed adattivi.

Deep Learning



Ricordate il neurone?

Ecco un neurone artificiale

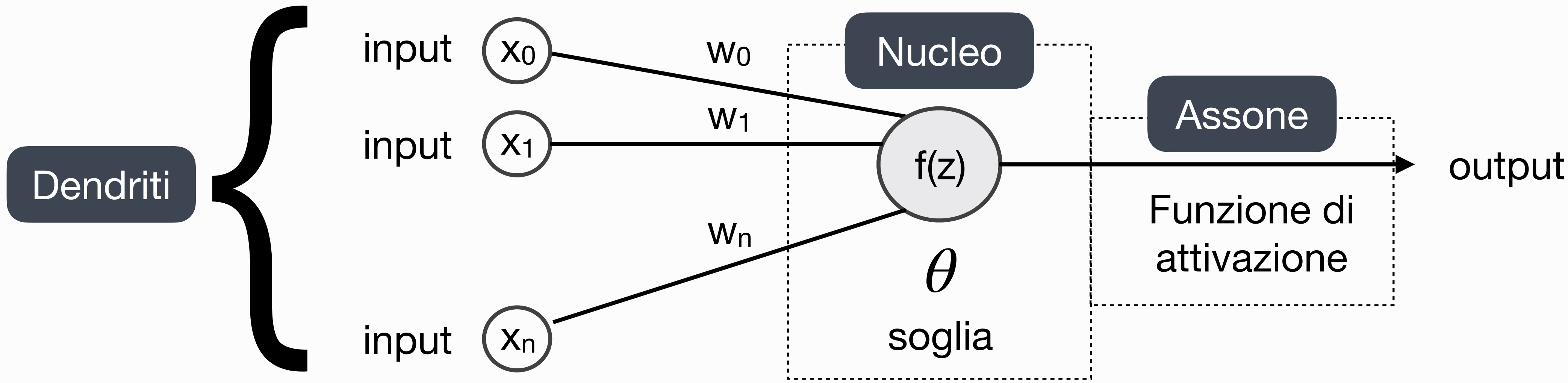


$x = [x_0, x_1, \ldots, x_n]$

$w = [w_0, w_1, \ldots, w_n]$

$y = f(x \bullet w^T)$

Deep Learning



$x = [x_0, x_1, \ldots, x_n]$ $w = [w_0, w_1, \ldots, w_n]$ $y = f(x \bullet w^T)$

Se $f(z) > \theta \Longrightarrow f(z) - \theta > 0$. In questo caso, il neurone è attivato ed il suo segnale verrà propagato

Una rete di questo tipo è chiamata *rete neurale artificiale* e rappresenta la base per rappresentare reti ancora più complesse.

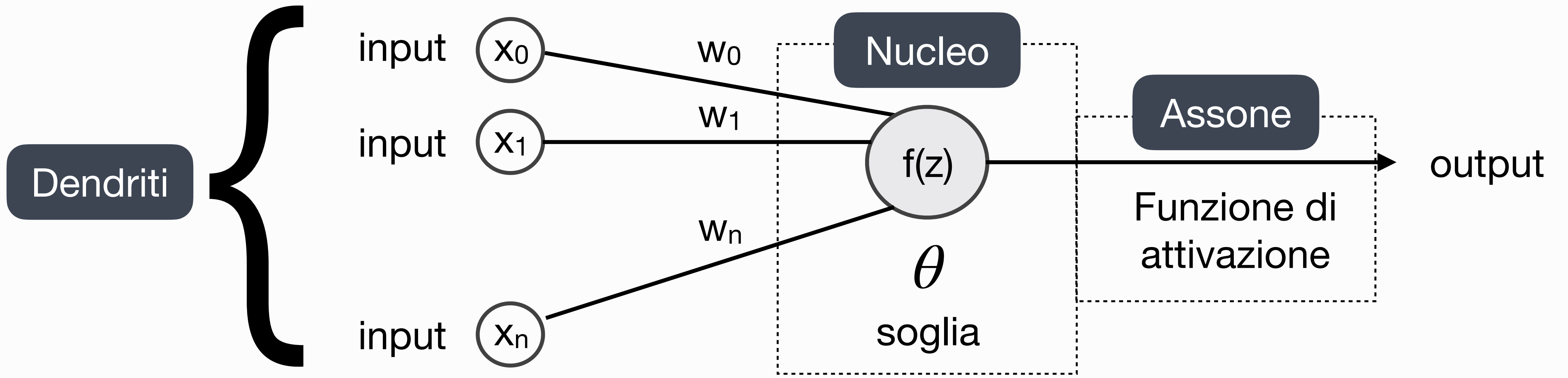
Ma perché dovremmo usare una rete neurale se abbiamo classificatori o regressori più semplici ed intuitivi?

Già guardando la forma del neurone artificiale, possiamo renderci conto che questo **NON presenta** alcuna informazione su ciò che dobbiamo predire —> un tipico caso d'uso è rappresentato dai problemi per cui non conosciamo nulla sulla variabile di output.

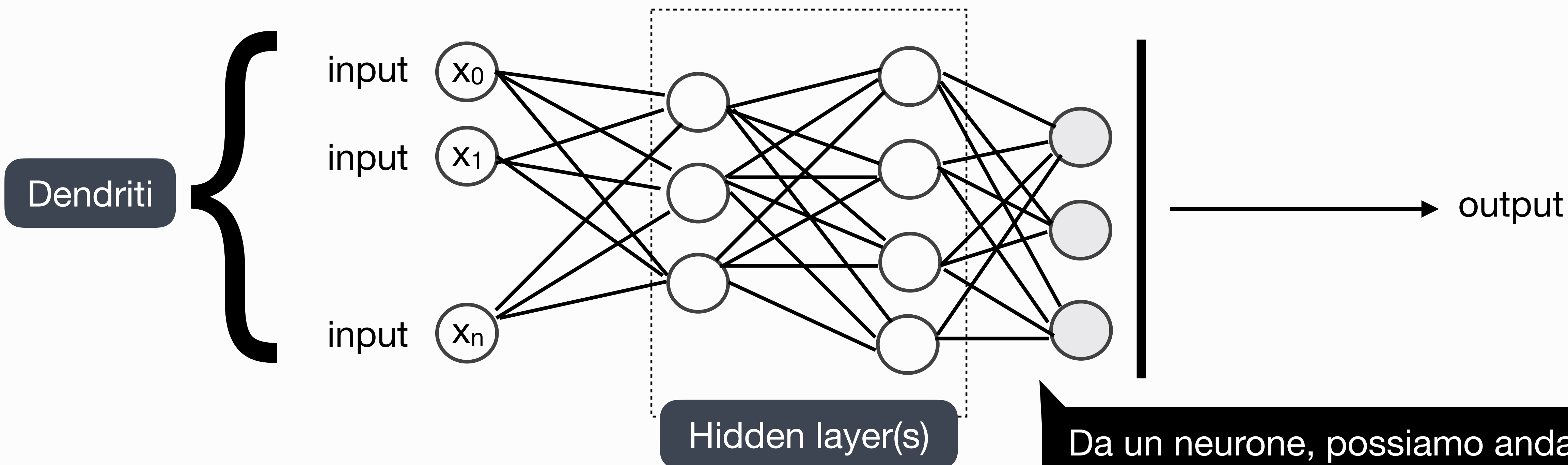
L'esempio classico dell'auto a guida autonoma: come potremmo mai utilizzare un decision tree o un classificatore bayesiano?

Regressione, Regressori, e Oltre

Deep Learning



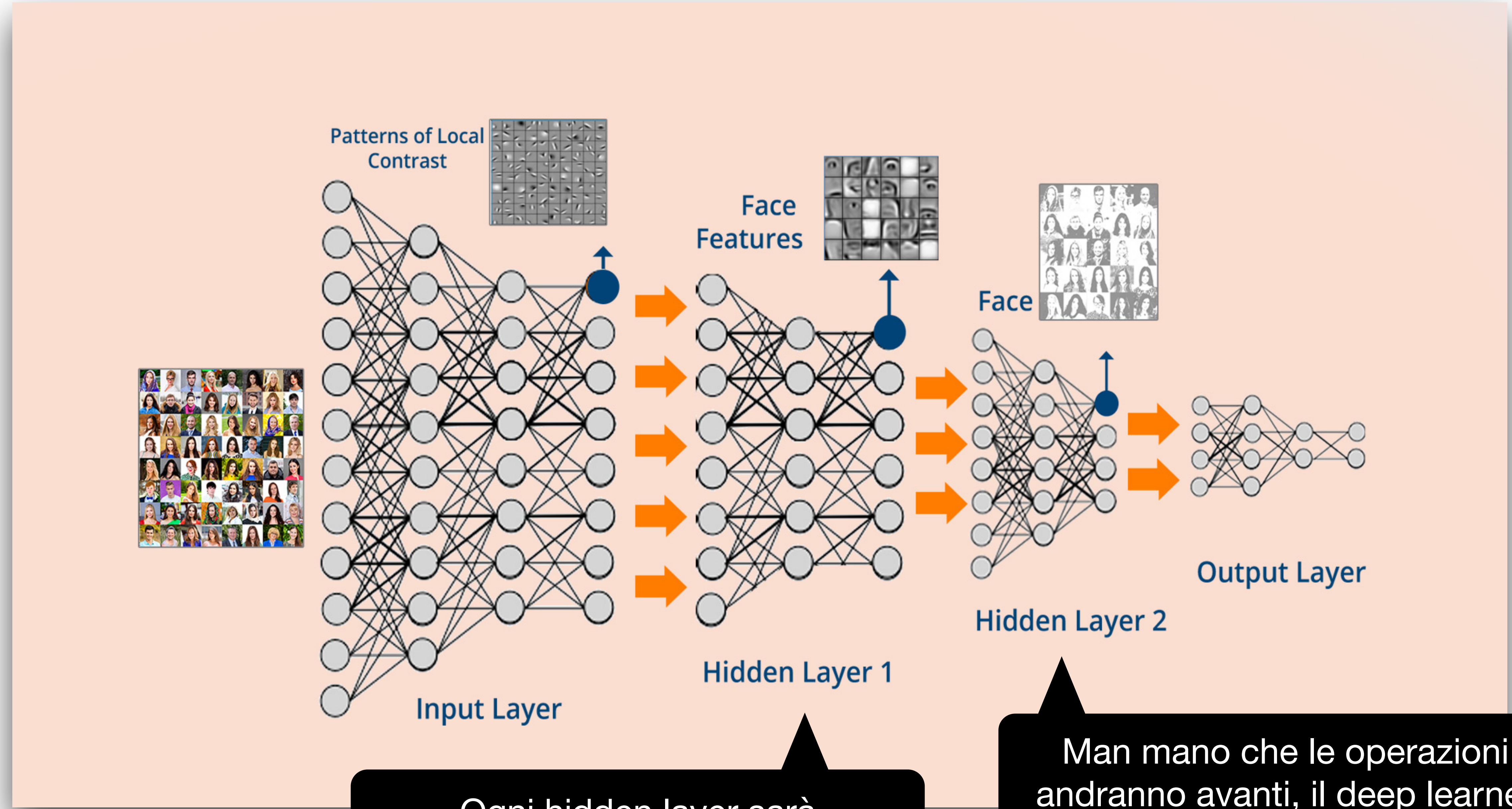
$x = [x_0, x_1, \ldots, x_n]$ $w = [w_0, w_1, \ldots, w_n]$ $y = f(x \bullet w^T)$



Da un neurone, possiamo andare deep, formando una rete neurale multistrato

Regressione, Regressori, e Oltre

Deep Learning



Ogni hidden layer sarà specializzato in qualcosa

Man mano che le operazioni andranno avanti, il deep learner avrà le “idee più chiare”

I più interessati, possono dare un’occhiata qui: https://www.slideshare.net/ashraybhandare/deep-learning-cnn-and-rnn.

Quantum Machine Learning

"The dream has come true", dice qualcuno… la tecnologia quantum è ormai dietro le porte e presto potremo sfruttare a pieno le potenzialità dei quantum computer. Alcuni, addirittura, già nominano il 21° secolo come la *"quantum era"*.

Il quantum computing si basa su due idee di base: (1) *superposition*, i q-bit possono assumere stati diversi allo stesso momento; (2) *entanglement*, i q-bit possono essere fortemente connessi anche in assenza di interazioni fisiche dirette.

Oggi come oggi, siamo in attesa della cosiddetta *quantum supremacy*, ovvero il momento in cui un programma interamente su quantum computing sarà in grado di risolvere problemi che nessun altro programma tradizionale può risolvere in nessun ragionevole ammontare di tempo.

Il machine learning rappresenta un problema particolarmente interessante per il quantum computing. Un motivo su tutti: la fase di addestramento è molto costosa, soprattutto quando parliamo di deep learning.

E quindi, possiamo usare delle componenti quantistiche per velocizzare le attività di addestramento dei modelli di machine learning?

Il *quantum-enhanced machine learning* è la branca che si occupa di come ingegnerizzare architetture quantistiche che possano supportare l'addestramento dei modelli di machine learning. Non entreremo nel dettaglio, chiaramente…

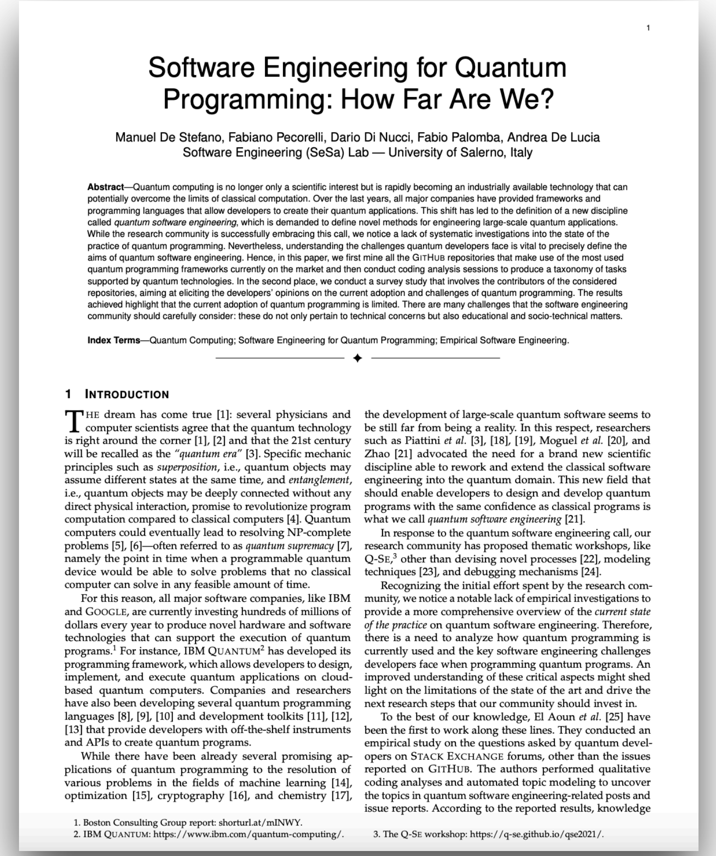
Ma è vero anche il contrario…

Quantum Machine Learning

Quantum machine learning è anche associato all'utilizzo di algoritmi di machine learning che analizzano e predicono dati quantistici.

Sebbene tutto questo sia ancora qualcosa di molto preliminare, la vera sfida è rappresentata dalla possibilità di combinare componenti tradizionali e quantistiche per l'analisi dei dati o, addirittura, creare modelli completamente basati su q-bit (il cosiddetto *fully quantum machine learning*).

Come ultima nota, vale la pena notare che la tecnologia quantistica è usata anche per i problemi di ricerca: ad esempio, un noto algoritmo è il *quantum annealing*.



1

Software Engineering for Quantum Programming: How Far Are We?

Manuel De Stefano, Fabiano Pecorelli, Dario Di Nucci, Fabio Palomba, Andrea De Lucia
Software Engineering (SeSa) Lab — University of Salerno, Italy

Abstract—Quantum computing is no longer only a scientific interest but is rapidly becoming an industrially available technology that can potentially overcome the limits of classical computation. Over the last years, all major companies have provided frameworks and programming languages that allow developers to create their quantum applications. This shift has led to the definition of a new discipline called *quantum software engineering*, which is demanded to define novel methods for engineering large-scale quantum applications. While the research community is successfully embracing this call, we notice a lack of systematic investigations into the state of the practice of quantum programming. Nevertheless, understanding the challenges quantum developers face is vital to precisely define the aims of quantum software engineering. Hence, in this paper, we first mine all the GitHub repositories that make use of the most used quantum programming frameworks currently on the market and then conduct coding analysis sessions to produce a taxonomy of tasks supported by quantum technologies. In the second place, we conduct a survey study that involves the contributors of the considered repositories, aiming at eliciting the developers' opinions on the current adoption and challenges of quantum programming. The results achieved highlight that the current adoption of quantum programming is limited. There are many challenges that the software engineering community should carefully consider: these do not only pertain to technical concerns but also educational and socio-technical matters.

Index Terms—Quantum Computing; Software Engineering for Quantum Programming; Empirical Software Engineering.

1 INTRODUCTION

THE dream has come true [1]: several physicians and computer scientists agree that the quantum technology is right around the corner [1], [2] and that the 21st century will be recalled as the *"quantum era"* [3]. Specific mechanic principles such as *superposition*, i.e., quantum objects may assume different states at the same time, and *entanglement*, i.e., quantum objects may be deeply connected without any direct physical interaction, promise to revolutionize program computation compared to classical computers [4]. Quantum computers could eventually lead to resolving NP-complete problems [5], [6]—often referred to as *quantum supremacy* [7], namely the point in time when a programmable quantum device would be able to solve problems that no classical computer can solve in any feasible amount of time.

For this reason, all major software companies, like IBM and GOOGLE, are currently investing hundreds of millions of dollars every year to produce novel hardware and software technologies that can support the execution of quantum programs.[1] For instance, IBM QUANTUM[2] has developed its programming framework, which allows developers to design, implement, and execute quantum applications on cloud-based quantum computers. Companies and researchers have also been developing several quantum programming languages [8], [9], [10] and development toolkits [11], [12], [13] that provide developers with off-the-shelf instruments and APIs to create quantum programs.

While there have been already several promising applications of quantum programming to the resolution of various problems in the fields of machine learning [14], optimization [15], cryptography [16], and chemistry [17], the development of large-scale quantum software seems to be still far from being a reality. In this respect, researchers such as Piattini *et al.* [3], [18], [19], Moguel *et al.* [20], and Zhao [21] advocated the need for a brand new scientific discipline able to rework and extend the classical software engineering into the quantum domain. This new field that should enable developers to design and develop quantum programs with the same confidence as classical programs is what we call *quantum software engineering* [21].

In response to the quantum software engineering call, our research community has proposed thematic workshops, like Q-SE,[3] other than devising novel processes [22], modeling techniques [23], and debugging mechanisms [24].

Recognizing the initial effort spent by the research community, we notice a notable lack of empirical investigations to provide a more comprehensive overview of the *current state of the practice* on quantum software engineering. Therefore, there is a need to analyze how quantum programming is currently used and the key software engineering challenges developers face when programming quantum programs. An improved understanding of these critical aspects might shed light on the limitations of the state of the art and drive the next research steps that our community should invest in.

To the best of our knowledge, El Aoun *et al.* [25] have been the first to work along these lines. They conducted an empirical study on the questions asked by quantum developers on STACK EXCHANGE forums, other than the issues reported on GITHUB. The authors performed qualitative coding analyses and automated topic modeling to uncover the topics in quantum software engineering-related posts and issue reports. According to the reported results, knowledge

1. Boston Consulting Group report: shorturl.at/mINWY.
2. IBM QUANTUM: https://www.ibm.com/quantum-computing/.
3. The Q-SE workshop: https://q-se.github.io/qse2021/.

Qiskit

Cirq

Q#

QuantuMoonlight

UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA

Laurea triennale in Informatica

Fondamenti di Intelligenza Artificiale

Lezione 17 - Regressione, regressori, e oltre