Macchine Predittive per DevOps

Gregorio Piccoli, Michel Glèlè Ahanhanzo

Padova, 26 Novembre 2019

ZUCCHETTI SpA

Prima software house italiana, per storia e dimensione:

- 41 anni
- 800M Euro di fatturato
- 5500 persone
- 130000 clienti attivi
- Presenza in 50 paesi

Siamo una “software house”, non un “system integrator”.

ZUCCHETTI SpA

Tre grandi divisioni responsabili dei prodotti:

1. Gestionali: contabilità, acquisti, vendite, magazzino, ...
2. Risorse umane: paghe, stipendi, presenze, controllo accessi, …
3. Fiscale: dichiarazione dei redditi, 730, conservazione sostitutiva, ...

e con molti prodotti a contorno: Business Intelligence, robotica, IoT, sicurezza, …

Fatturazione elettronica

A partire dal 1 Gennaio 2019 tutte le fatture emesse in Italia dovranno essere spedite al MEF (Ministero dell' Economia e delle Finanze) per poter essere considerate valide.

Ogni anno in Italia vengono emesse circa 2.000.000.000 fatture.

La Zucchetti offre ai propri clienti il servizio integrato nei propri gestionali e dei servizi di spedizione per le terze parti.

Ogni giorno vengono processati dai nostri sistemi circa 600.000 documenti, 365 giorni all'anno per un totale previsto di circa 250.000.000 a fine anno.

Fatturazione elettronica

La Zucchetti è contemporaneamente produttrice del software e gestore del servizio.

La fabbrica è in contatto con le operazioni sul campo, questo è lo scenario indicato dal termine “DevOps”, contrazione delle due parole “Development” ed “Operations”.

Cosa cambia quando un unico attore si occupa sia del “development” che delle “operations”?

DevOps

“Development” + “Operations”: un unico attore che svolge i due ruoli.

Lo sviluppo opera per un sistema specifico e che controlla in ogni sua componente, non per una “universalità” non ben definita.

L’operatività quotidiana deve garantire che i sistemi siano attivi e funzionanti, ma anche segnalare i possibili miglioramenti allo sviluppo.

I sistemisti devono conoscere cosa potrebbe essere in grado di realizzare la “fabbrica”, i programmatori devono diventare un po’ sistemisti.

DevOps

Ciclo tradizionale: analisi, progettazione, sviluppo e test.

Si arriva quindi al rilascio: normalmente sono pacchetti distribuiti ai clienti.

Il software, uscito dalla produzione, viene gestito dai sistemisti.

Gli sviluppatori ricevono dei feedback dal servizio di assistenza, i sistemisti devono far funzionare al meglio quello che ricevono.

DevOps

Nel caso “DevOps” il programma non esce dalla “fabbrica” e viene gestito da sviluppatori e sistemisti.

I sistemisti possono richiedere e suggerire modifiche al prodotto in base a metriche oggettive, non sono costretti ad usare le sole “leve” delle configurazioni dell’hardware.

Gli sviluppatori ricevono un flusso continuo di informazioni sullo stato della procedura e sul suo funzionamento.

Il “Responsabile della qualità” diventa molto più importante.

DevOps

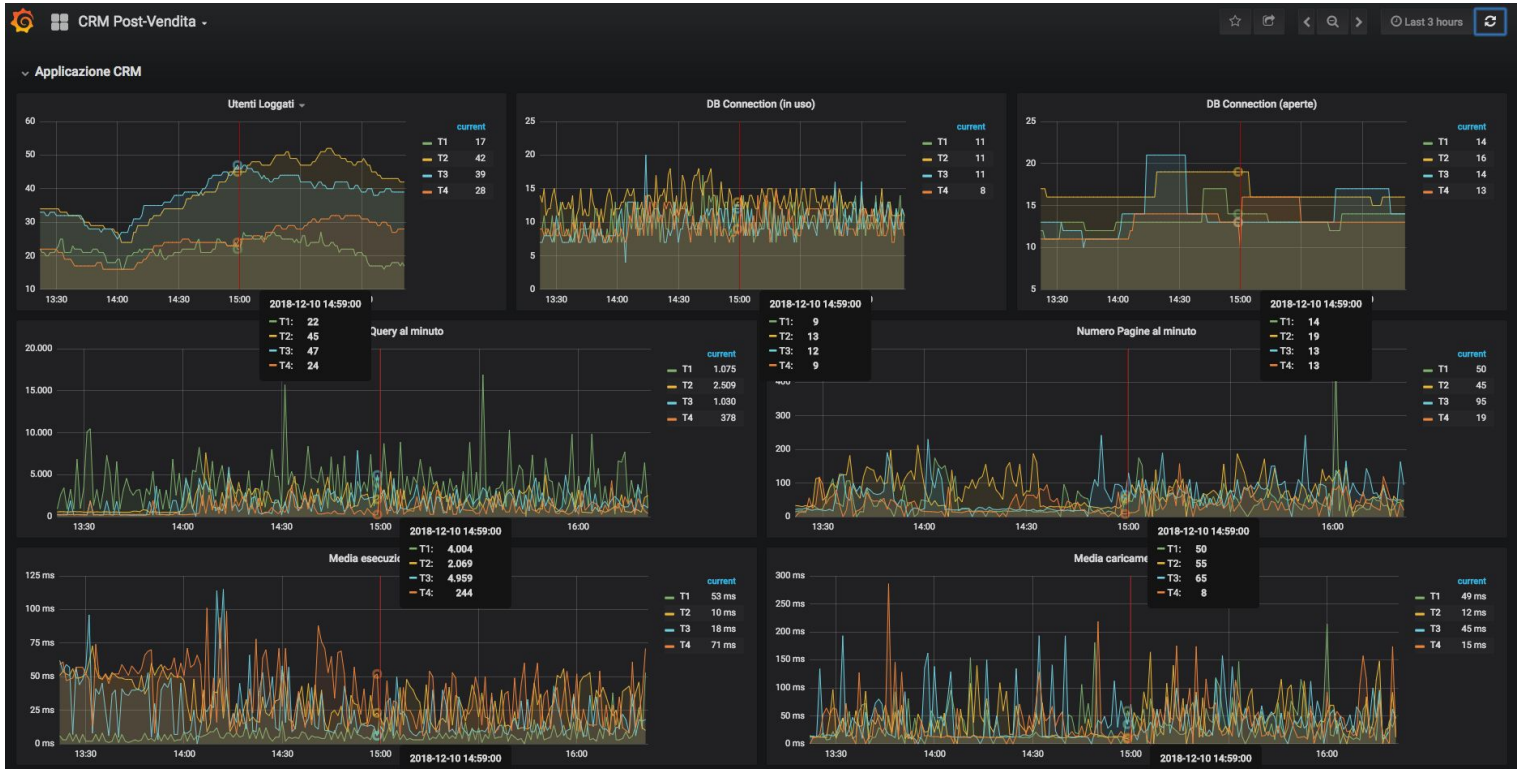
La velocità di rilascio, data dal controllo dell'unica installazione, porta a release molto più frequenti.

Quindi i team devono organizzarsi con cicli di vita molto più rapidi, il che significa quasi sempre adottare metodologie agili.

La possibilità di rilasciare velocemente diventa essenziale: si possono fare più release al giorno!

Ci si scontra con la continuità del servizio (SLA).

GRAFANA


CRM Post-Vendita
Last 3 hours
Applicazione CRM
Utenti Loggati
current
T1 17
T2 42
T3 39
T4 28
DB Connection (in uso)
current
T1 11
T2 11
T3 11
T4 8
DB Connection (aperte)
current
T1 14
T2 16
T3 14
T4 13
2018-12-10 14:59:00
T1: 22
T2: 45
T3: 47
T4: 24
2018-12-10 14:59:00
T1: 9
T2: 13
T3: 12
T4: 9
2018-12-10 14:59:00
T1: 14
T2: 19
T3: 13
T4: 13
Query al minuto
current
T1 1.075
T2 2.509
T3 1.030
T4 378
Numero Pagine al minuto
current
T1 50
T2 45
T3 95
T4 19
2018-12-10 14:59:00
T1: 4.004
T2: 2.069
T3: 4.959
T4: 244
2018-12-10 14:59:00
T1: 50
T2: 55
T3: 65
T4: 8
Media esecuzi
Media caricame
current
T1 53 ms
T2 10 ms
T3 18 ms
T4 71 ms
current
T1 49 ms
T2 12 ms
T3 45 ms
T4 15 ms
13:30
14:00
14:30
15:00
16:00

GRAFANA

Il monitoraggio delle applicazioni diventa particolarmente importante perché ora ha due compiti:

1. Controllare la salute del sistema e verificare che le performance siano entro i livelli di SLA (Service Level Agreement)
2. Identificare quali punti deboli possono essere corretti dal team di sviluppo e fornire elementi per definire la scala delle priorità nelle migliorie e nelle nuove implementazioni

Si deve monitorare ogni cosa: le pagine servite, il numero degli utenti, la loro provenienza, l’uso della CPU, il disco, la rete ...

GRAFANA

Esempio di pagina di “alert”.

Se qualcosa è in condizione critica e richiede l’attenzione del sistemista l’indicatore prende un colore diverso.

Nel caso di situazioni estreme scattano avvisi come e-mail ai responsabili.

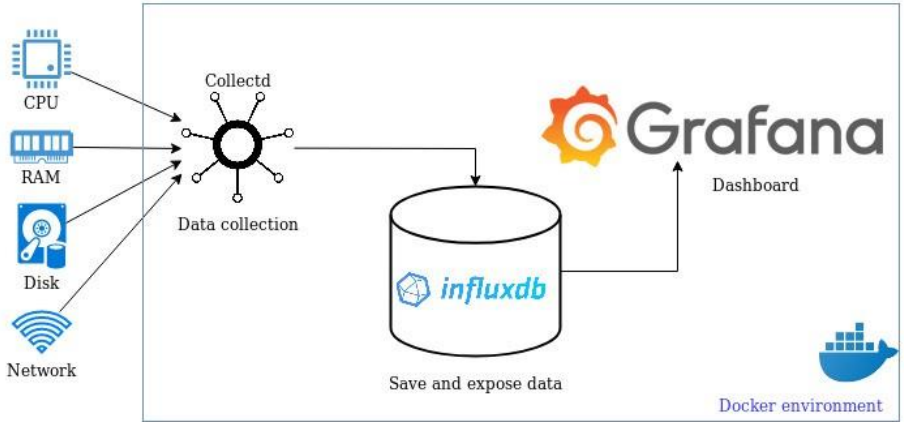
Reperibilità 24/7 dei sistemisti.

GRAFANA

L’architettura del “sistema” Grafana prevede degli agenti che raccolgono delle misure dalle varie macchine.

I dati vengono spediti a una istanza di InfluxDB, un database specializzato in serie temporali.

Grafana interroga InfluxDB presentando dashboard di grafici e notificando allarmi.

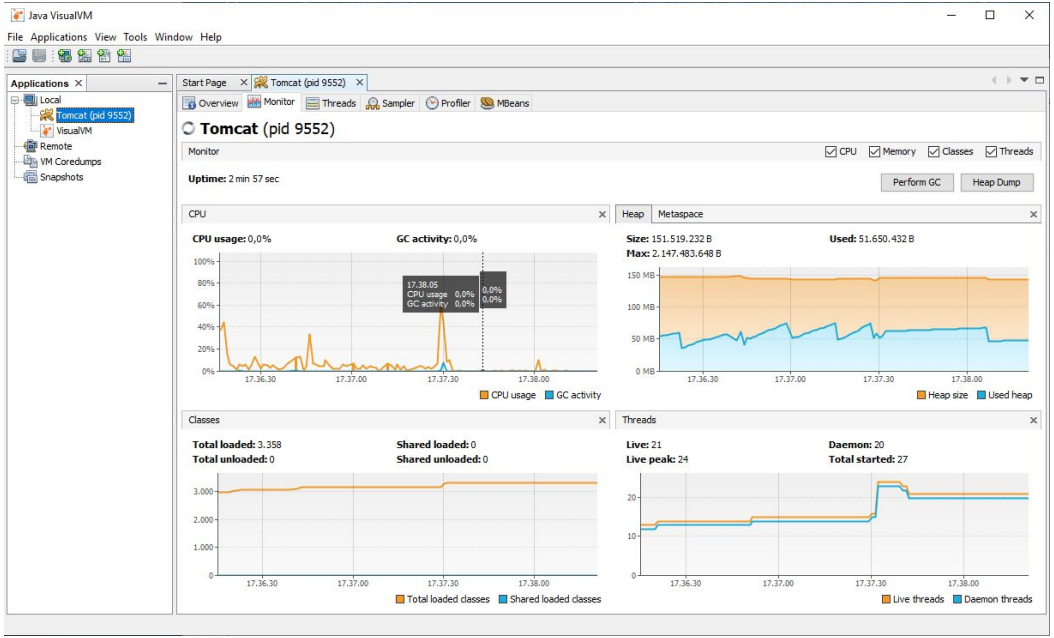


Java JMX

In procedure basate su Java i “sensori” sono spesso degli M-bean.

JMX è lo standard per il monitoraggio della Java Virtual Machine.

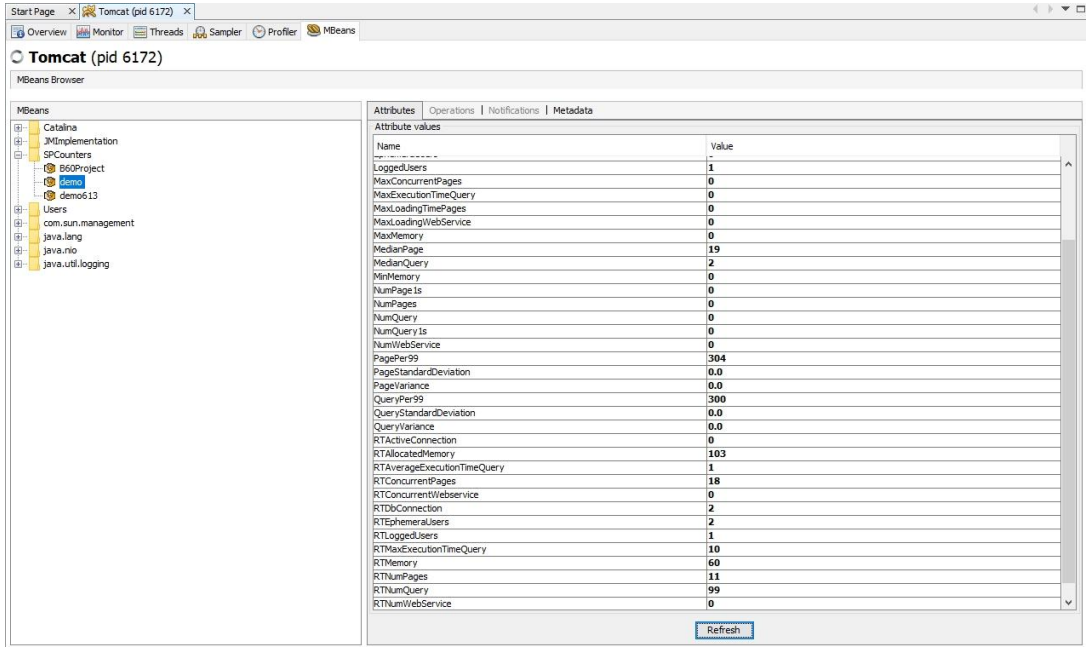
Nella distribuzione ufficiale di Java troviamo VisualJVM e JMC (Java mission control).



Java JMX

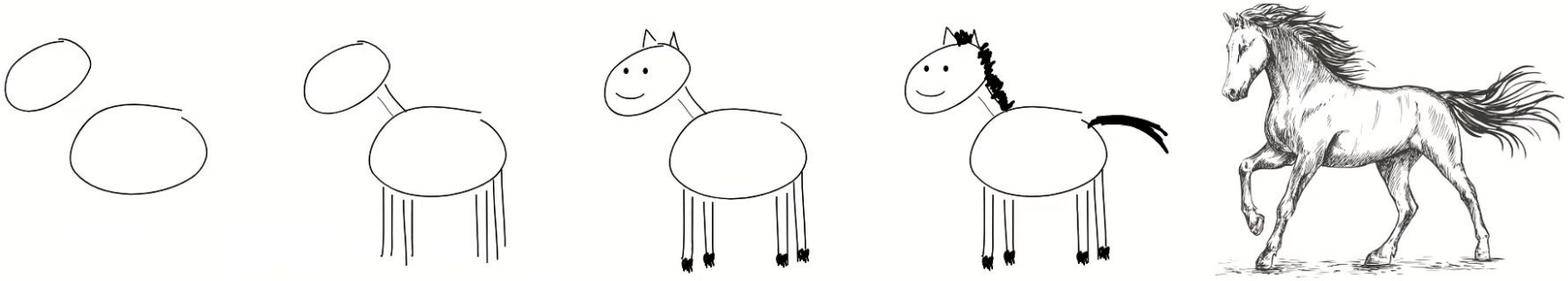
Le applicazioni possono riportare dei dati ai sistemisti creando degli M-bean per esporre il loro stato interno.

Abbiamo sviluppato dei bean per il numero di utenti loggati, le connessioni al database, lo stato del pooler, i tempi di esecuzione delle query …



Performance delle applicazioni in architettura Web

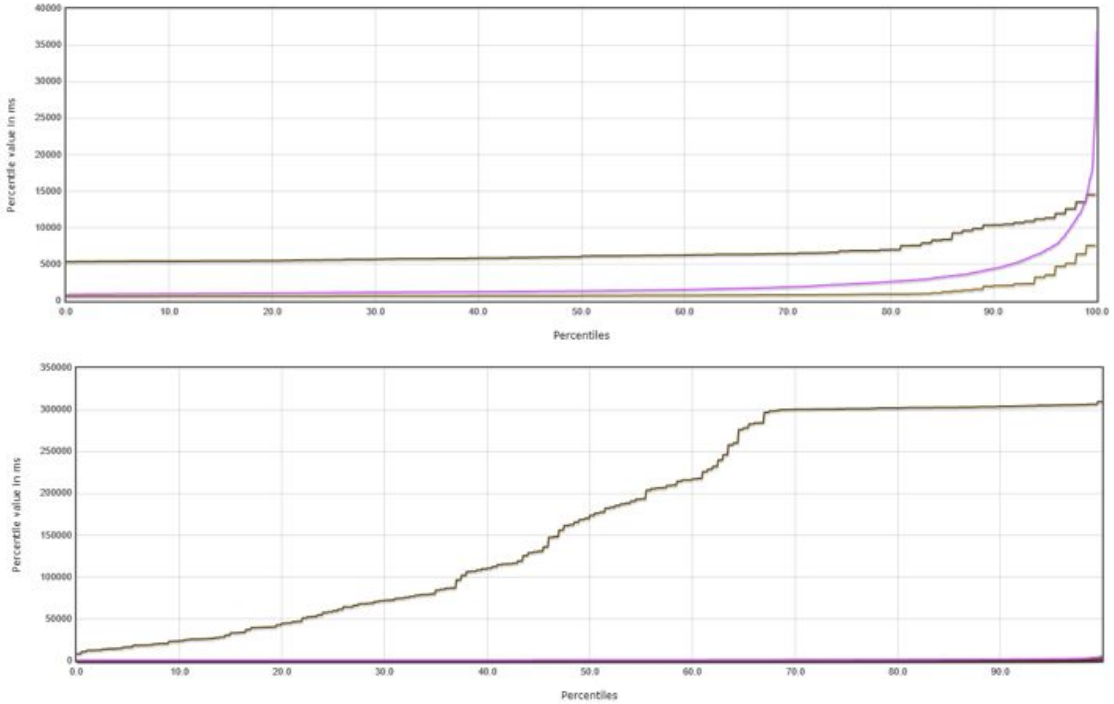
Fixing web performance is as easy as drawing a horse



Java JMeter

“Prevenire è meglio di curare”: i sistemi possono essere messi artificialmente sotto stress.

JMeter è uno strumento della Apache Foundation ben integrato nello stack Java per verificare il comportamento delle applicazioni Web.

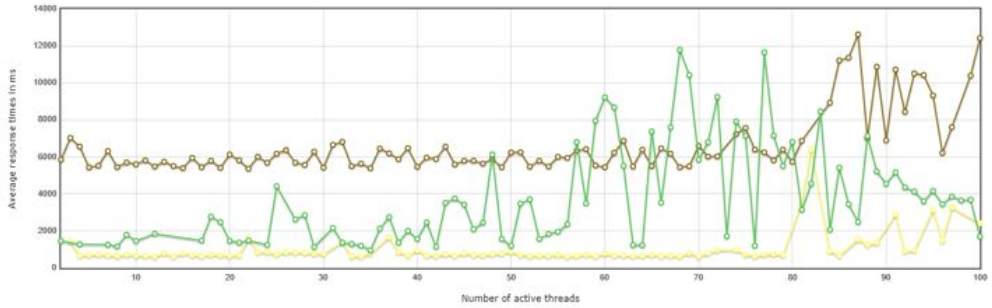


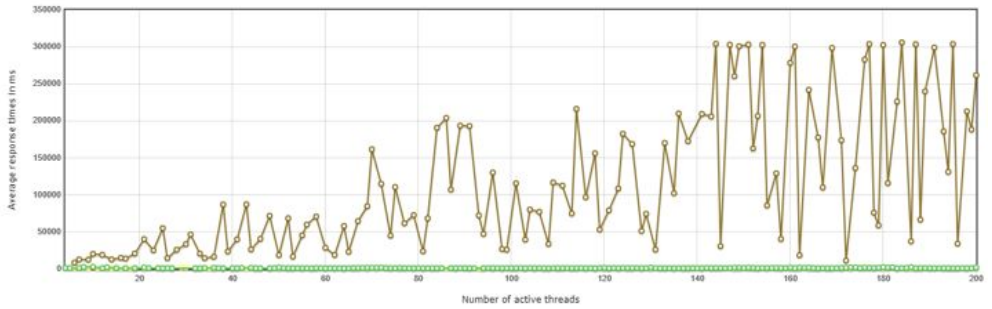
Java JMeter

Nelle immagini a lato si vede la risposta del sistema all’aumentare dei thread, quindi degli utenti concorrenti.

L’immagine a fianco mostra un sistema che fallisce.

Le immagini della slide precedente mostrano la tipica distribuzione dei tempi di esecuzione.





Prevedere

I dati dal monitoraggio sono alla base del “DevOps”, le informazioni certe permettono di migliorare costantemente l’applicazione.

Le misure dal campo trasferiscono alla linea di produzione i dati per guidare l’ottimizzazione del codice.

Dai dati grezzi è possibile ricavare nuovi dati, in presenza di “regole” è possibile eseguire delle previsioni.

Poter produrre valori che anticipano quello che avverrà può aiutare a prevenire i problemi o interpretare meglio i dati del monitoraggio.

Prevedere: regressione e classificazione

Esistono due tipi fondamentali di previsione:

1. **Regressione**: a partire da dei dati ne viene prodotto uno nuovo numerico e che assume dei valori di tipo continuo.
2. **Classificazione**: a partire da dei dati ne viene prodotto uno nuovo numerico o stringa con valori di tipo discreto.

I dati usati per produrre il nuovo valore sono chiamati “predittori”.

Esistono molteplici algoritmi e metodi sia di regressione che di classificazione come la regressione lineare, regressione logistica, le reti neurali, le support vector machine, gli alberi decisionali, le “random forest”, il “naive bayes”, etc.

Prevedere: qualità della previsione

Nel contesto del “DevOps” è molto importante la qualità della previsione.

Un dato può provocare degli allarmi, i quali attivano delle risposte che possono andare dal limitare accessi alle risorse fino ad interventi umani a qualunque ora del giorno (e della notte).

Ogni calcolo dovrebbe avvenire solo quando è possibile “certificare” la qualità del risultato.

Le previsioni dovrebbero essere corredate delle motivazioni per giustificare il valore predetto e quindi permettere agli operatori di valutare se accettare quanto indicato o rigettare l’indicazione e procedere per altre vie.

Regressione lineare

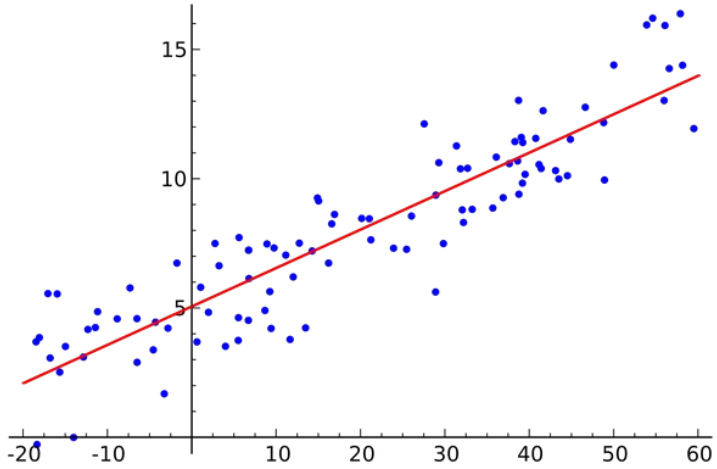
La regressione lineare è il metodo più classico di previsione di un valore numerico: sottende una “regola” determinata da una linea retta.

La formula di previsione è:

$$y = a*x+b$$

dove i parametri **a** e **b** vengono determinati con il metodo dei minimi quadrati.

Inventata nel 1800 da Legendre e Gauss.



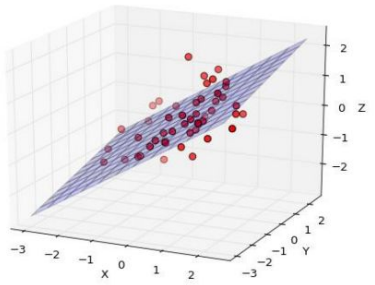
Regressione lineare

La regressione lineare può essere estesa a più predittori. La formula diventa:

$$y = a_1 * x_1 + a_2 * x_2 + \ldots + b$$

Per determinare i parametri a_i e **b** si prendono dei dati di “addestramento” e si cercano i valori per cui la somma del quadrato degli errori (distanza tra retta e valore reale) è minima.

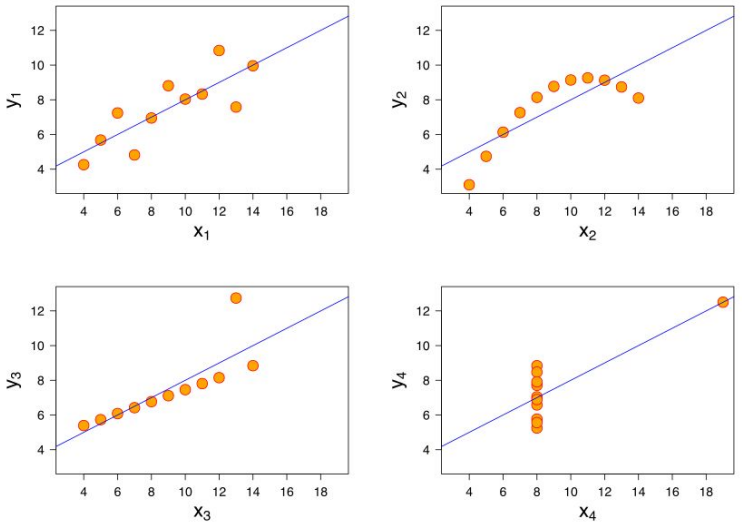
Per prevedere il valore della variabile **y** si applica ai valori attuali delle variabili x_i la formula qui riportata che contiene solo moltiplicazioni e somme.



Regressione lineare

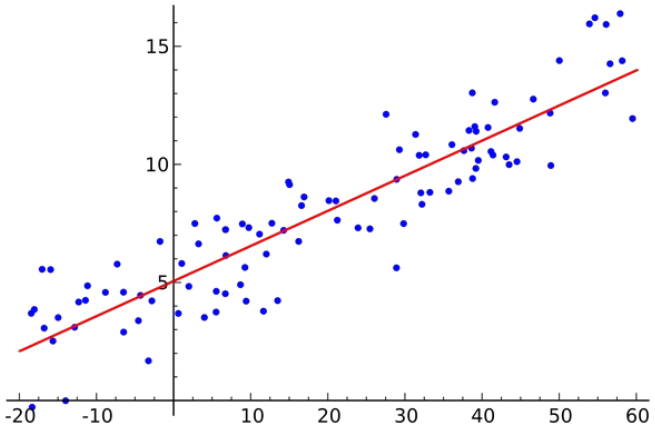
Quando si può applicare la regressione lineare? Il quartetto di Anscombe ci presenta i casi estremi.

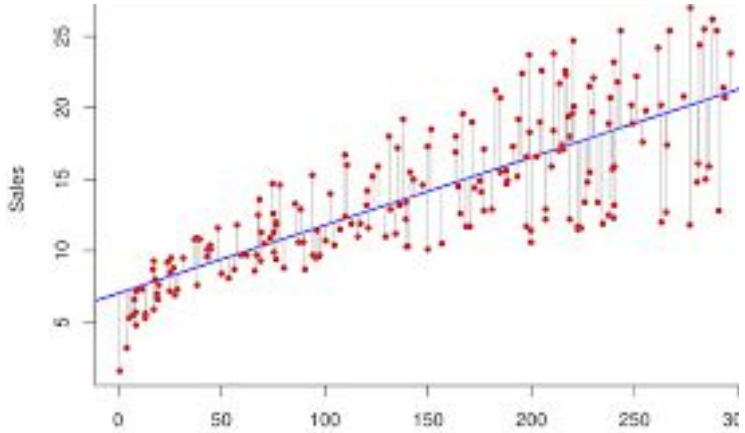
1. Dati ottimali per la Regressione Lineare
2. Dati non lineari
3. Outlier che sfalsa la regola
4. Dati che non si prestano in alcun modo all'analisi con la Regressione Lineare



Regressione lineare

La Regressione Lineare è applicabile in entrambi i casi?





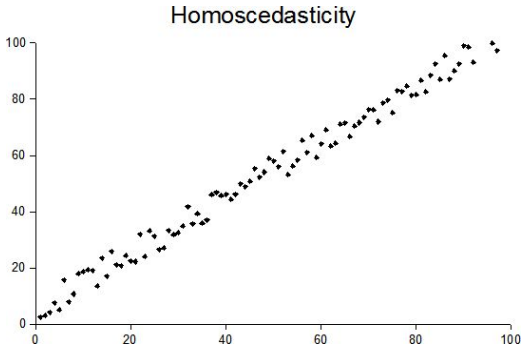
Regressione lineare

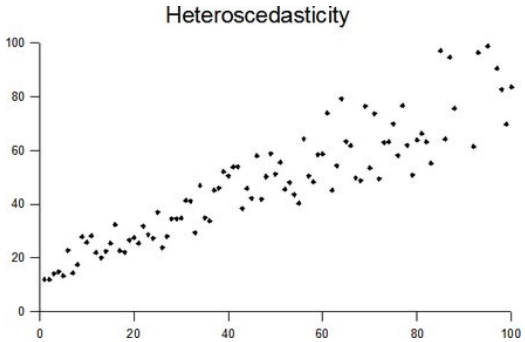
Un importante problema con la RL è la varianza degli scostamenti:

Omoschedasticità: le variabili hanno la stessa varianza, gli errori di previsione sono omogenei.

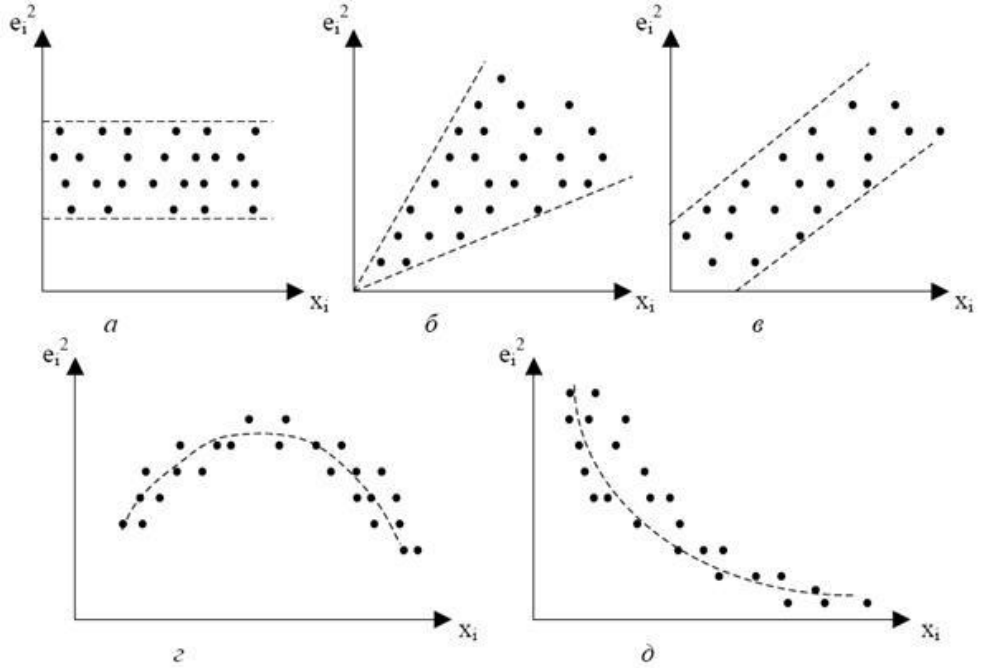
Eteroschedasticità: le variabili aleatorie non hanno la stessa varianza, gli errori dipendono dalla posizione.

Di solito l'eteroschedasticità si presenta in fenomeni moltiplicativi e si "cura" con una trasformazione logaritmica.





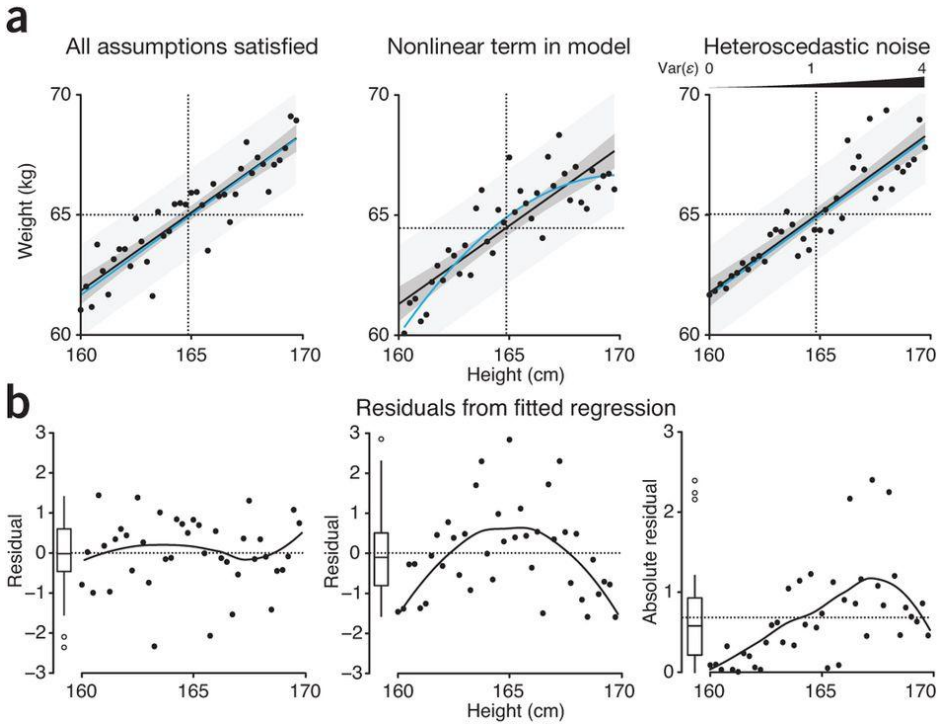
Regressione lineare



Regressione lineare

L’osservazione dei residui (scostamenti tra retta e valori reali) permette di rilevare questi problemi.

Quando il grafico dei residui non è una palla di punti informe (distribuzione normale) vuol dire che la regressione lineare non è il sistema di previsione corretto.

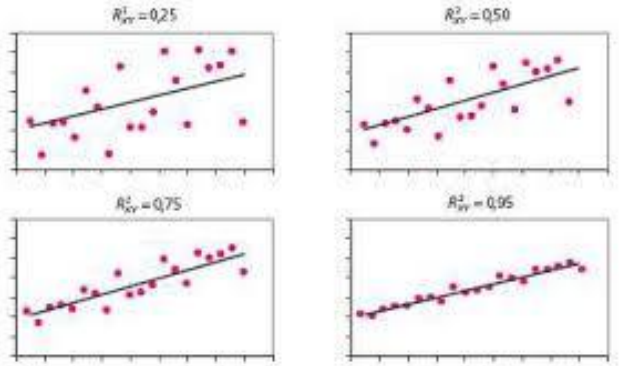


Regressione lineare

Anche quando tutti i vincoli sono rispettati possiamo avere qualità diverse!

Coefficiente R^2 che nasce dal rapporto tra gli errori rispetto alla retta e gli errori rispetto alla media della variabile **y**.

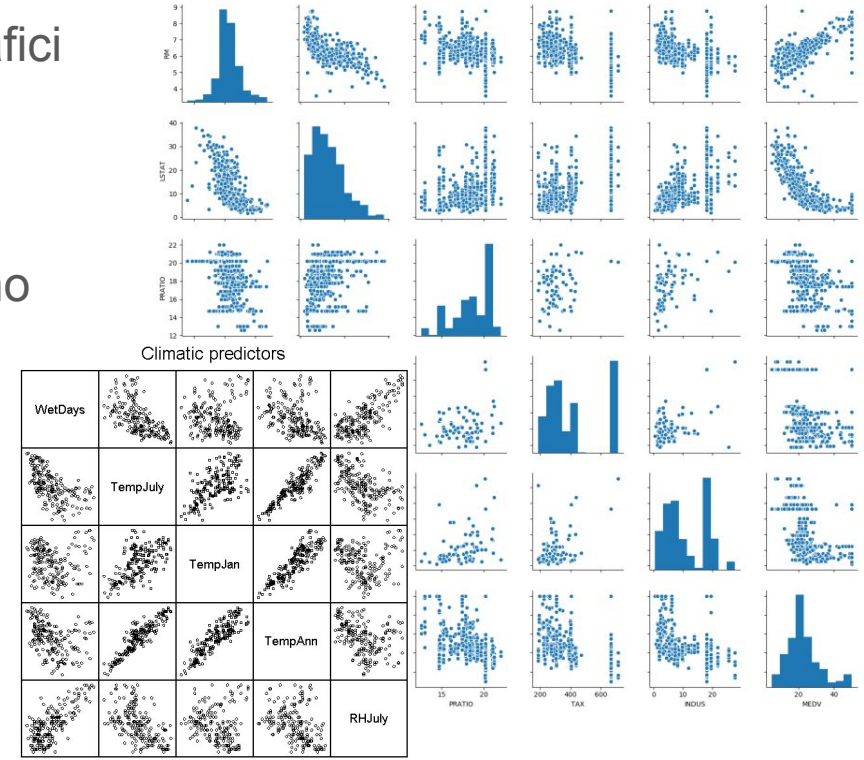
Un coefficiente vicino a 1 indica punti molto aderenti alla retta, un coefficiente vicino allo 0 indica punti dispersi.



Casi reali

Dati presi da 6 e 5 variabili, matrice dei grafici di correlazione.

Da questa matrice si può osservare facilmente che solo alcune variabili possono prevedere il valore di un’altra con la regressione lineare.



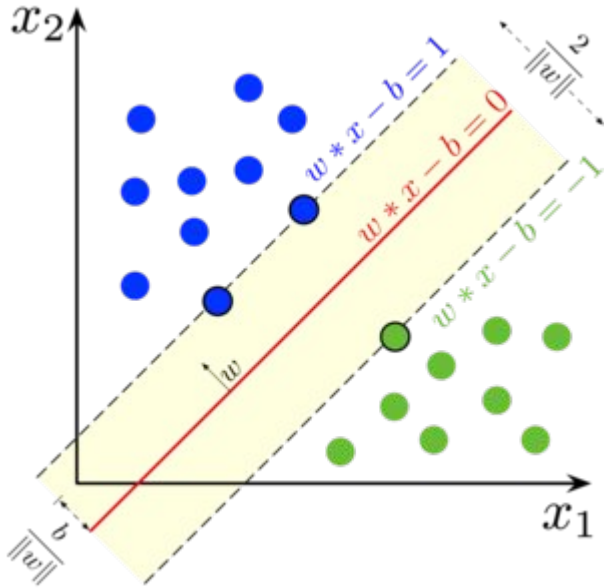
Support vector machine

Le SVM sono algoritmi per classificare inventati da V. Vapnik nel 1990.

Nell'esempio vediamo due predittori, $\mathbf{x}_1$ e $\mathbf{x}_2$, e la classe da predire è il colore verde o il colore blu dei punti.

La formula delle SVM è la stessa della RL! Le SVM cercano il piano che separa meglio le categorie nello spazio.

Le SVM identificano i punti del "vettore di supporto".

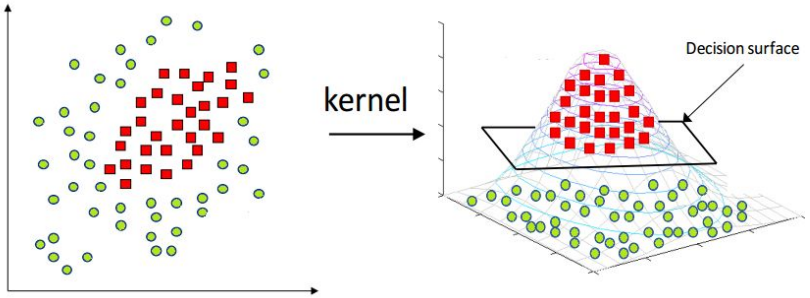


Support vector machine

Le SVM possono operare anche con problemi non lineari, utilizzando il “trucco del kernel”.

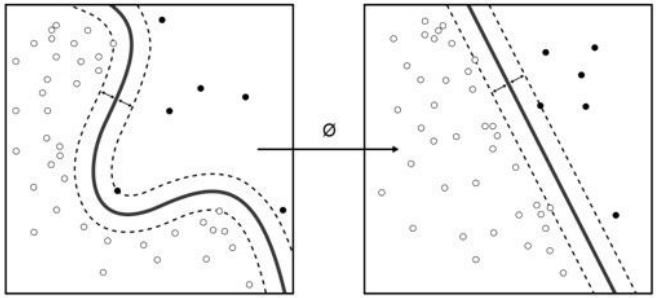
Si applica una trasformazione dello spazio, spesso aumentando le dimensioni, portando i punti in un sistema che è separabile da un (iper-)piano.

Le SVM non soffrono della “maledizione della dimensionalità”



Support vector machine

Esempio di trasformazione polinomiale. In questo caso i punti sono stati spostati in uno spazio della stessa dimensione di quello iniziale e posizionati in modo che una linea potesse separarli.



Le SVM sono confrontabili con le reti neurali ad uno strato con algoritmi di addestramento molto più semplici e deterministici.

Le SVM possono essere usate anche per la regressione.

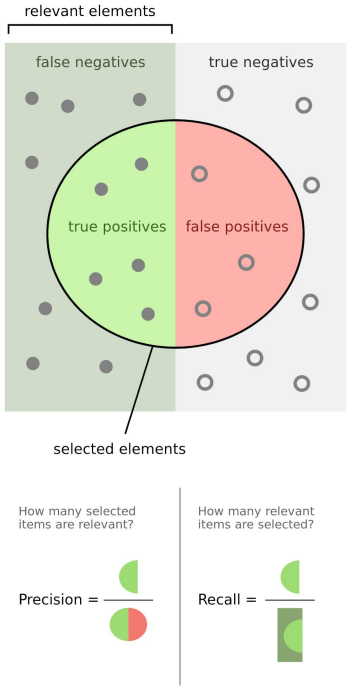
Precision e Recall

La qualità dei sistemi di classificazione viene valutata da varie misure, le principali sono “Precision” e “Recall”.

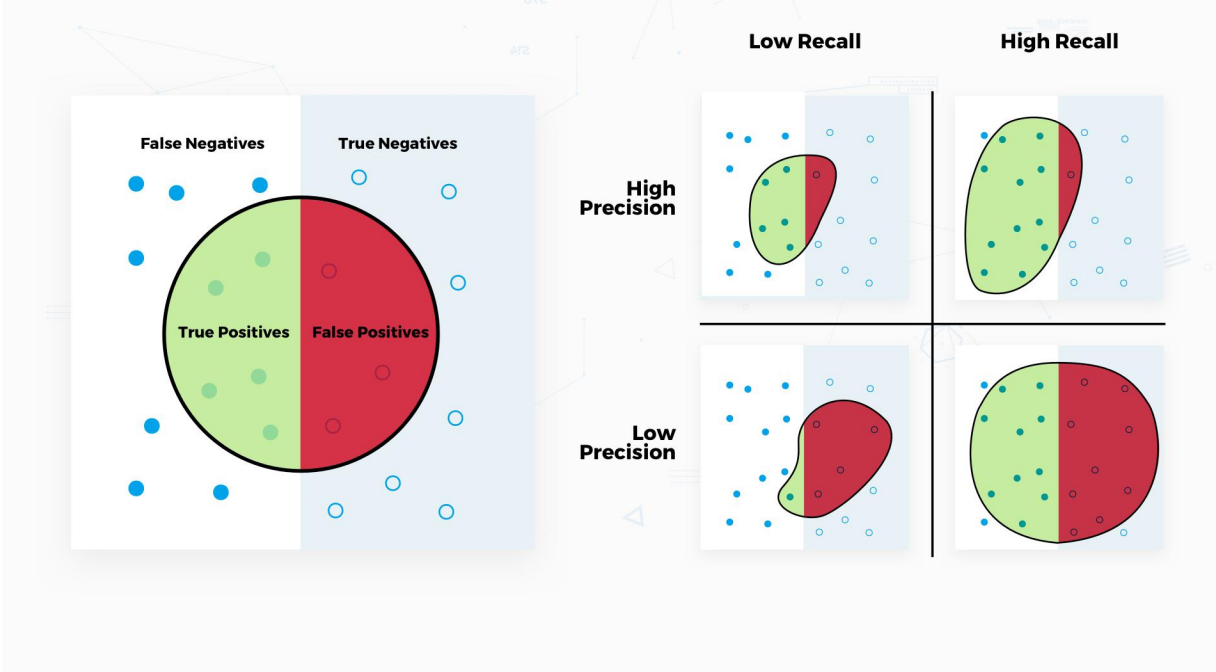
Precision: quanti di quelli che dico essere veri lo sono davvero?

Recall: quanti di quelli che sono veri non sono stato capace di trovare?

Queste due misure si possono combinare in una unica misura, la F-Measure, che indica la bontà generale del sistema di classificazione.



Precision e Recall

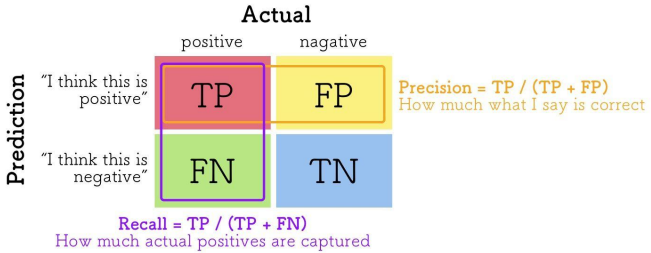


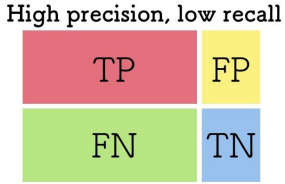
Confusion Matrix

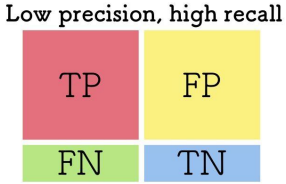
Lo strumento per valutare questi indici è la “Confusion Matrix”, che riporta i veri positivi, i falsi positivi, i veri negativi e i falsi negativi.

Dal confronto tra questi numeri nascono “precision”, “recall” e parecchi altri indicatori.

		Actual class	
		Cat	**Non-cat**
Predicted class	**Cat**	5 True Positives	2 False Positives
	Non-cat	3 False Negatives	3 True Negatives







Confusion Matrix

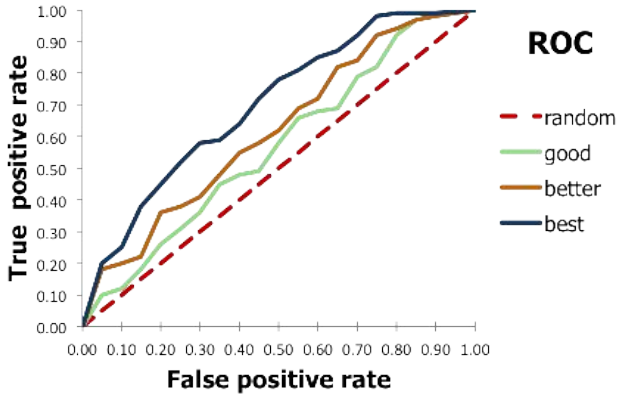
		True condition			
	Total population	Condition positive	Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$	F_1 score $= 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$

Curva ROC

Gli algoritmi di classificazione spesso possono essere calibrati.

La curva ROC (Receiver Operating Characteristic) mostra il comportamento del sistema riportando il tasso di veri positivi al variare del tasso di falsi positivi.

Più il sistema è buono e più la curva si avvicina alla costante 1, la retta che passa per i punti (0,0) e (1,1) corrisponde alle scelte a caso.



Aggiornamento dei parametri “On-Line”

Gli algoritmi proposti hanno un momento di addestramento e uno di applicazione delle formule per effettuare le previsioni.

Un tema interessante che può essere affrontato è l’addestramento “on-line” direttamente sul flusso dei dati. Poter aggiornare i parametri direttamente sul campo apre nuove possibilità ma pone anche nuovi problemi, in particolare sulla qualità della previsione.

Sia la Regressione Lineare che le Support Vector Machines hanno la versione dell’addestramento applicabile direttamente sul flusso di dati.

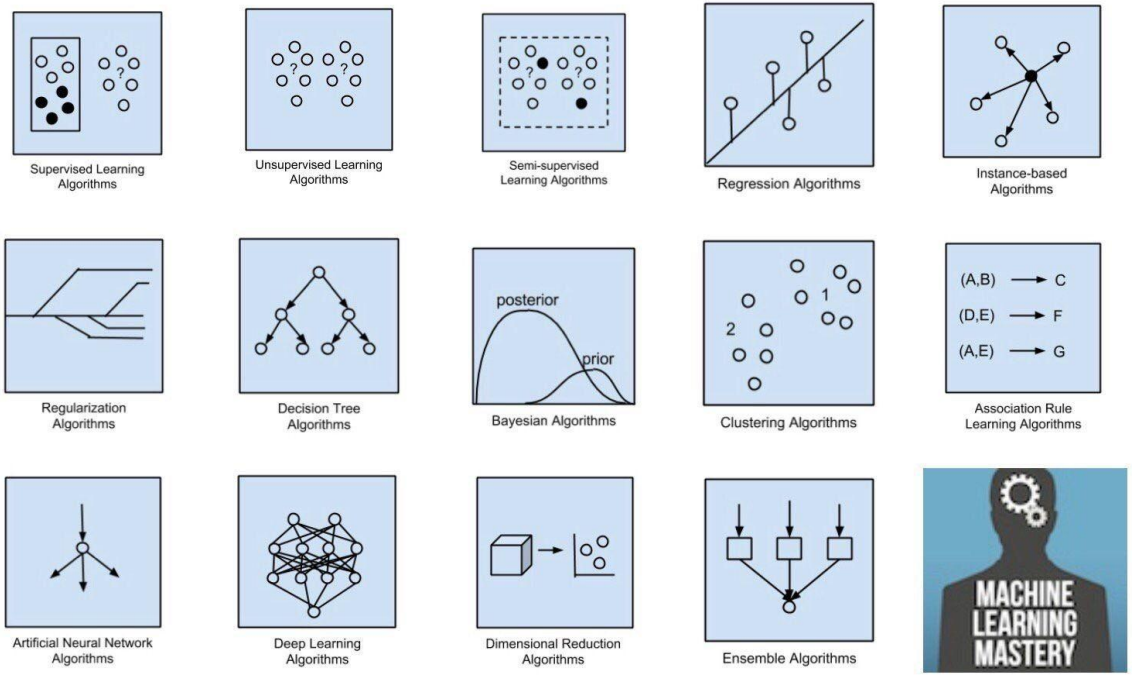
Algoritmi alternativi di Regressione e Classificazione

Regressione Lineare e Support Vector Machine sono gli algoritmi più semplici ed immediati per il tema del progetto, il proponente dispone di librerie in Javascript già pronte e le mette a disposizione dei gruppi che realizzeranno il capitolato.

Altri metodi che potrebbero essere presi in considerazione:

1. Alberi di decisione
2. Random Forest
3. Reti Neurali
4. Naive Bayes classifier
5. Regressione Logistica

Algoritmi alternativi di Regressione e Classificazione



Domande?

gregorio.piccoli@zucchetti.it