

IN550 Machine Learning

Training, validazione e test

Riduzione delle feature

Vincenzo Bonifaci

Training set e test set

Separiamo **a caso** i dati di esempio a nostra disposizione in due insiemi:



Molti metodi di apprendimento richiedono di selezionare **iperparametri**:

- η e T negli algoritmi basati su Gradient Descent
- K nei metodi K -Nearest Neighbor
- λ nei metodi con regolarizzazione
- ...

Quale metodologia per selezionare i valori degli iperparametri?

Ricerca esaustiva su una griglia (*grid search*)

Si discretizzano tramite una griglia dei possibili intervalli per ognuno degli iperparametri, ad esempio se gli iperparametri sono η e T si potrebbe considerare:

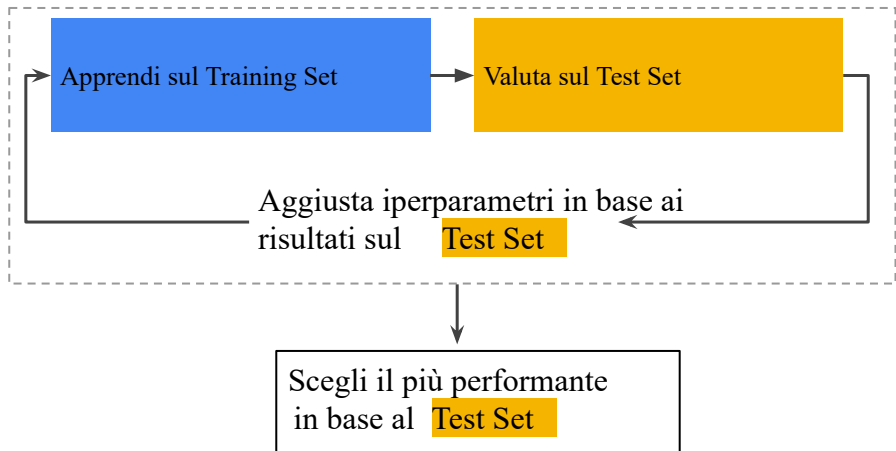
- $\eta \in \{0.0003, 0.001, 0.003, 0.01, 0.03, 0.1\} = \Theta_1$
- $T \in \{50, 100, 200, 500, 1000, 2000\} = \Theta_2$

Si valuta ogni combinazione $(\eta, T) \in \Theta_1 \times \Theta_2$ applicando l'algoritmo di apprendimento e si usa la migliore

In generale si valuta ogni combinazione $\theta \in \Theta$ per qualche griglia Θ

Ma su quale insieme di dati?

Un possibile schema di lavoro?

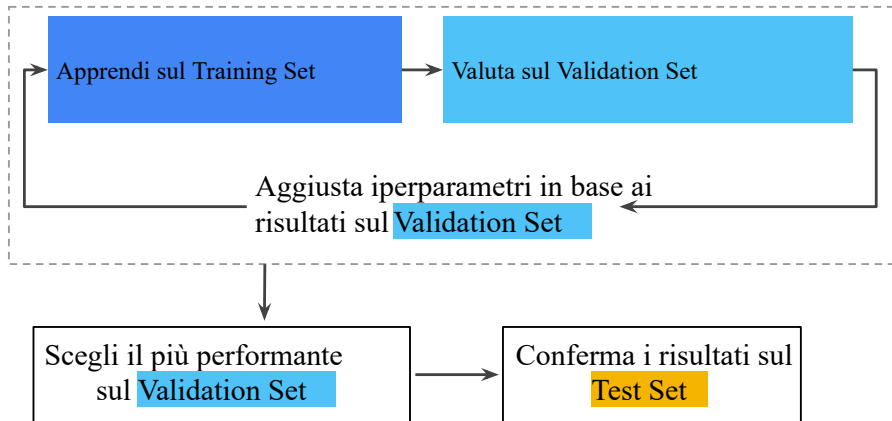


Il validation set

Separiamo **a caso** i dati di esempio a nostra disposizione in **tre** insiemi:



Uso di un validation set



- Validation set e test set devono essere **sufficientemente** grandi da poter stimare con la precisione desiderata il rischio atteso
- Compatibilmente col punto precedente, il training set dovrebbe essere **il più grande possibile**

Teorema (Taglia del test set)

Sia h un'ipotesi e si consideri una funzione costo a valori in $[0, C]$. Allora per ogni $\delta \in (0, 1)$, con probabilità almeno $1 - \delta$ sulla scelta di un test set T di cardinalità m_T si ha

$$|\text{RE}_T(h) - \text{RA}(h)| \leq C \sqrt{\frac{\log(2/\delta)}{2m_T}}$$

Interpretazione: l'errore con cui il test set stima il rischio atteso decresce con la radice quadrata della dimensione del test set

\Rightarrow per una stima di $\text{RA}(h) \pm 1\%$ è sufficiente m_T dell'ordine di 10,000

Dimensionamento dei vari insiemi in pratica

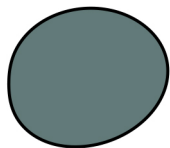
- Per dataset **piccoli** o **medi** ($\approx 100 - 10,000$ osservazioni) è spesso usata una suddivisione 70%-30% oppure 60%-20%-20%:
70% per il training set, 30% per il test set, o
60% per il training set, 20% per il validation set, 20% per il test set
- Per dataset **grandi** ($\approx 100,000 - 1,000,000$ osservazioni e oltre) può essere sufficiente mantenere validation set e test set intorno alle 10,000 o 100,000 osservazioni

Esempio: 1,000,000 osservazioni

98% training set, 1% validation set, 1% test set può essere accettabile

Validazione incrociata (*cross-validation*)

Se i dati scarseggiano, possiamo fare a meno del validation set?



■ training
■ validation



fold 1



fold 2



fold 3

Validazione incrociata (*cross-validation*)

K intero positivo (valore spesso utilizzato: $K = 10$)

K-fold Cross Validation (K-CV)

- 1 Partiziona il training set S in K sottoinsiemi (*fold*) S_1, \dots, S_K
- 2 Ripeti per $i = 1, \dots, K$:
 - Apprendi un'ipotesi h_i usando tutti i dati in S **tranne quelli in S_i**
 - Calcola il rischio empirico di h_i usando i dati in S_i : $RE_{S_i}(h_i)$
- 3 Restituisci, come stima del rischio atteso dell'ipotesi appresa su S , il rischio medio di validazione:

$$RA(h) \approx \frac{1}{K} \sum_{i=1}^K RE_{S_i}(h_i)$$

Avvertenza: la validazione incrociata è un'euristica spesso adottata in pratica, ma raramente è supportata da risultati teorici

Validazione incrociata per la selezione di un modello

La validazione incrociata può essere combinata con la ricerca degli iperparametri su una griglia Θ

K -CV per la selezione degli iperparametri

Input: training set S , griglia di iperparametri Θ , algoritmo A , intero K

- 1 Partiziona il training set S in K sottoinsiemi (*fold*) S_1, \dots, S_K
- 2 Per ogni $\theta \in \Theta$:
 - Per $i = 1, \dots, K$: $h_{i,\theta} = A(S \setminus S_i; \theta)$
 - $\text{risk}(\theta) = \frac{1}{K} \sum_{i=1}^K \text{RE}_{S_i}(h_{i,\theta})$
- 3 $\theta^* = \text{argmin}_{\theta} \text{risk}(\theta)$
- 4 $h^* = A(S; \theta^*)$

Riduzione delle feature

Perché ridurre le feature?

Supponiamo di avere d variabili di input e m osservazioni

Ridurre il numero di predittori (d) può essere importante per:

- Tenere sotto controllo la **varianza** (specialmente quando $d > m$)
- Migliorare l'**interpretabilità** del modello

- *Selezione*: selezioniamo un **sottoinsieme** di $d' < d$ predittori
- *Shrinkage (regolarizzazione)*: penalizziamo modelli in cui **tanti** predittori hanno **grande** influenza sulla predizione
- *Proiezione*: proiettiamo i d predittori su un **sottospazio** d' -dimensionale con $d' < d$

NB. Qui illustreremo i metodi nel contesto della regressione lineare, ma i principi sono generali

Metodi di selezione

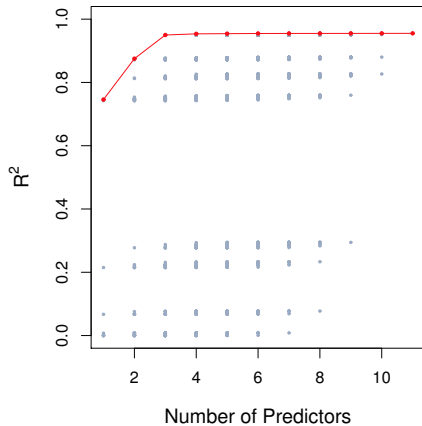
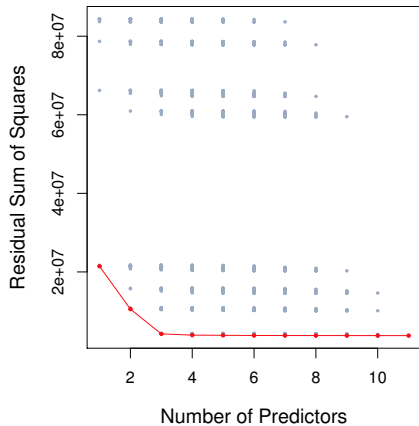
Best Subset Selection

- 1 $h_0 \leftarrow$ miglior ipotesi **costante**, cioè con 0 predittori
Per esempio, nella regressione lineare,

$$h_0(x) = \frac{1}{m} \sum_{i=1}^m y^{(i)}$$

- 2 Per $k = 1, 2, \dots, d$,
 - Apprendi tutte le $\binom{d}{k}$ ipotesi consistenti di k predittori
 - $h_k \leftarrow$ ipotesi col miglior **rischio empirico** tra queste
- 3 Restituisci l'ipotesi $h \in \{h_0, h_1, \dots, h_d\}$ col minimo **rischio atteso** stimato sul validation set (o con la validazione incrociata)

Esempio di Best Subset Selection



- Il numero totale di ipotesi considerate è 2^d :
inapplicabile se d è molto grande
- Se d è grande, lo spazio di ricerca è enorme e può dare luogo ad **overfitting**
- Per questo motivo, un'alternativa è costituita dai metodi di *selezione passo-passo*

Selezione passo-passo in avanti

- Inizia con un'ipotesi senza predittori
- Aggiungi un predittore alla volta: quello che fornisce il **massimo incremento** della qualità del fit
- Restituisci l'ipotesi col minimo rischio atteso stimato (sul validation set) tra tutte le $d + 1$ ipotesi costruite

Forward Stepwise Selection

- 1 $h_0 \leftarrow$ miglior ipotesi costante, con 0 predittori
 - Per $k = 0, \dots, d - 1$, considera tutte le $d - k$ ipotesi con un predittore **in più** di h_k
 - $h_k \leftarrow$ ipotesi col miglior **rischio empirico** tra queste
- 2 Restituisci l'ipotesi $h \in \{h_0, h_1, \dots, h_d\}$ col minimo **rischio atteso** stimato sul validation set (o con la validazione incrociata)

Vantaggi e svantaggi di Forward Stepwise Selection

- Ipotesi esplorate: $d + 1$ anziché 2^d
- **Non** garantisce il miglior modello tra i 2^d possibili

Esempio: Credit dataset

| k | Best Subset | Forward Stepwise |
|-----|----------------------------------|-----------------------------------|
| 1 | rating | rating |
| 2 | rating, income | rating, income |
| 3 | rating, income, student | rating, income, student |
| 4 | cards, income, student, limit | rating, income, student, limit |

Metodi di shrinkage e regolarizzazione

Regolarizzazione

Come cercare automaticamente un equilibrio tra bias e varianza?

Una *funzione di regolarizzazione* è una funzione $R : \mathcal{H} \rightarrow \mathbb{R}_+$

$R(h)$ è una qualche misura di complessità dell'ipotesi h

Regularized Loss Minimization (RLM)

Dato un insieme di esempi S , cerca una regola di predizione h che minimizzi il rischio empirico di h su S , più $R(h)$:

$$\min_{h \in \mathcal{H}} \text{RE}_S(h) + R(h)$$

Se l'ipotesi h è codificata dai coefficienti $w \in \mathbb{R}^{d+1}$, scriveremo anche $R(w)$ (in tal caso R è vista come funzione definita su \mathbb{R}^{d+1} anziché su \mathcal{H})

Regressione Ridge

$w = (w_0, w_1, \dots, w_d)$ include il termine costante w_0

$\omega = (w_1, w_2, \dots, w_d)$ non lo include

Regolarizzazione ℓ_2 (di Tikhonov)

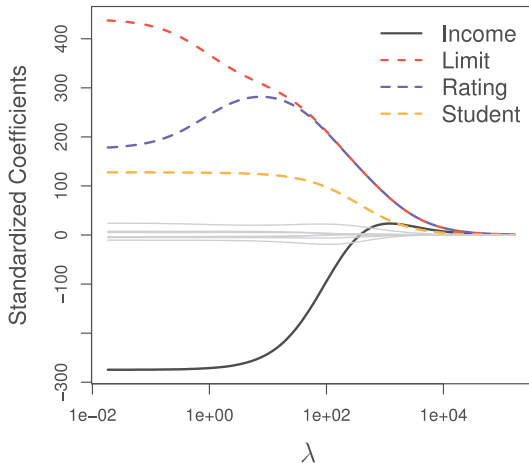
$$R(w) = \lambda(w_1^2 + w_2^2 + \dots + w_d^2) = \lambda \|\omega\|_2^2 \quad (\lambda \geq 0)$$

Regressione Ridge

$$\min_{w \in \mathbb{R}^{d+1}} \text{RE}_S(h) + \lambda \|\omega\|^2 = \min_{w \in \mathbb{R}^{d+1}} \frac{1}{m} \|Xw - y\|^2 + \lambda \|\omega\|^2$$

- Per $\lambda = 0$, coincide con il metodo dei minimi quadrati
- Per $\lambda \rightarrow \infty$, i coefficienti w_k tendono a 0
- Ammette soluzione in forma chiusa: $w^* = (X^\top X + \lambda m \cdot I_0)^{-1} X^\top y$

Regression Ridge



Valori dei coefficienti di una regressione ridge in funzione di λ

Regressione regolarizzata e standardizzazione delle variabili

Nella regressione regolarizzata è importante *standardizzare* i predittori, centrandonli sulla media e scalandonli per la deviazione standard:

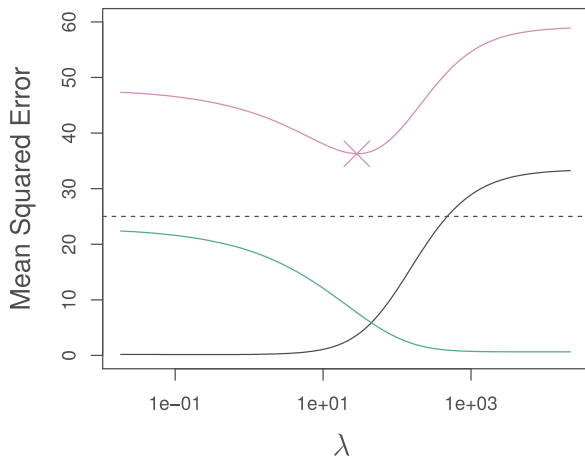
Standardizzazione di una variabile

Se v è un predittore, con esempi di valore v_1, \dots, v_m e media \bar{v} , poniamo

$$v \leftarrow \frac{v - \bar{v}}{\sqrt{\frac{1}{m} \sum_{i=1}^m (v_i - \bar{v})^2}}$$

Questo non era necessario nella regressione lineare, perché lì i coefficienti erano *equivarianti* rispetto alla scala: moltiplicare v per c scalava il coefficiente corrispondente di $1/c$

Regressione Ridge



Bias² (nero), varianza (verde) e MSE di test (viola) per una regressione ridge in funzione di λ

Regolarizzazione ℓ_1

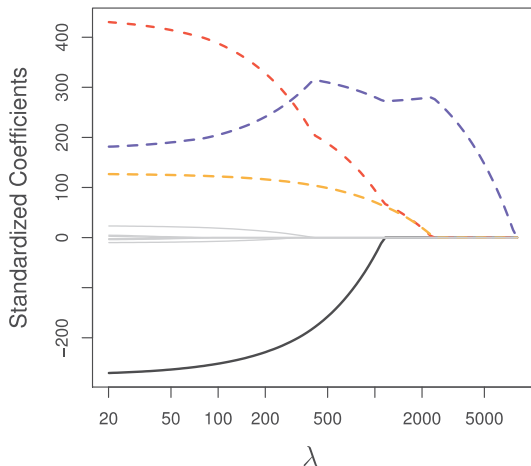
$$R(w) = \lambda(|w_1| + |w_2| + \dots + |w_d|) = \lambda \|\omega\|_1 \quad (\lambda \geq 0)$$

Regressione LASSO

$$\min_{w \in \mathbb{R}^{d+1}} \text{RE}_S(h) + \lambda \|\omega\|_1 = \min_{w \in \mathbb{R}^{d+1}} \frac{1}{m} \|Xw - y\|^2 + \lambda \|\omega\|_1$$

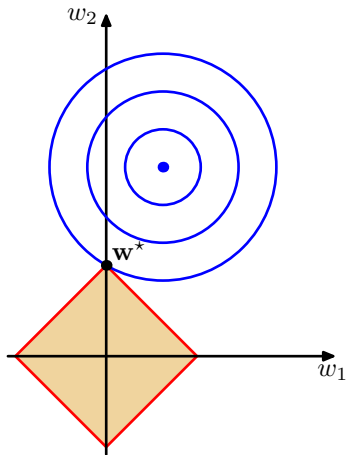
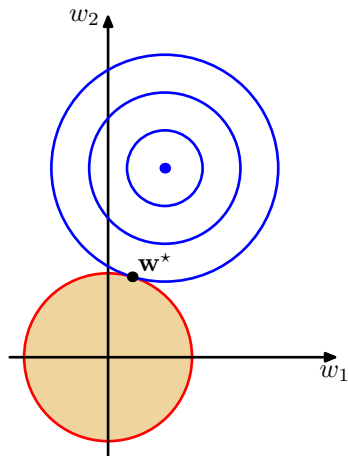
- Per $\lambda = 0$, coincide con il metodo dei minimi quadrati
- Per $\lambda \rightarrow \infty$, i coefficienti w_k tendono a 0
- Per λ crescente, alcuni coefficienti diventano **esattamente** pari a 0
(\Rightarrow incentiva modelli *sparsi*)

Regressione LASSO

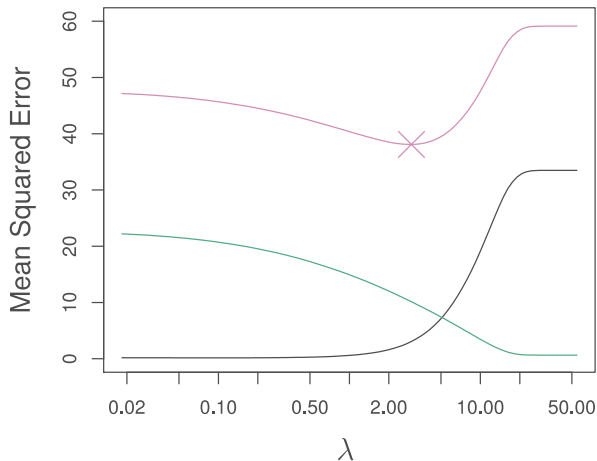


Valori dei coefficienti di una regressione LASSO in funzione di λ

Ridge ($\lambda \|w\|_2^2$) vs. LASSO ($\lambda \|w\|_1$)



Regressione LASSO



Bias² (nero), varianza (verde) e MSE di test (viola) per una regressione LASSO in funzione di λ

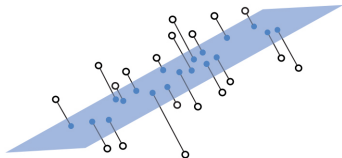
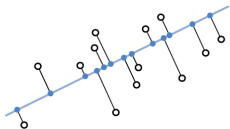
Metodi di regressione regolarizzata in scikit-learn

| | Iperparametri | Interfaccia scikit-learn |
|-----------------|--------------------|-----------------------------|
| Ridge (diretto) | λ | Ridge |
| Ridge (SGD) | η, T, λ | SGDRegressor(penalty='l2') |
| LASSO (SGD) | η, T, λ | SGDRegressor(penalty='l1') |

Metodi di proiezione

Parleremo dei metodi di proiezione nel contesto dell'apprendimento non supervisionato

Esempio tipico: Principal Component Analysis



Debugging dell'apprendimento

Esempio di scenario

- Filtro spam. Avete scelto un insieme di 100 parole da utilizzare come feature (invece di utilizzare 50000+ parole della lingua).
- La regressione logistica regolarizzata, implementata tramite discesa del gradiente, ottiene un errore di test del 20%, che per la vostra applicazione è **inaccettabile**
- Cosa fate ora?

Messa a punto dell'algoritmo di apprendimento

- Approccio possibile: provare a migliorare l'algoritmo in diversi modi
 - Ottenendo più esempi di training
 - Provando un insieme di feature più piccolo
 - Provando un insieme di feature più grande
 - Usando feature diverse: intestazione mail vs. corpo mail
 - Aumentando il numero di iterazioni di Gradient Descent
 - Usando un metodo di ottimizzazione del secondo ordine
 - Usando un diverso valore del coefficiente di regolarizzazione λ
 - Usando una Support Vector Machine anziché la regressione logistica
- Questo approccio potrebbe funzionare, ma richiede molto tempo, e rischia di diventare una questione di fortuna

Diagnosi di bias e varianza

Approccio preferibile:

- Diagnosi del problema incontrato dall'apprendimento
- Correzione del problema

Nel nostro scenario, l'errore di test (20%) è troppo alto

Supponiamo che sospettiate che il problema sia uno tra i seguenti due:

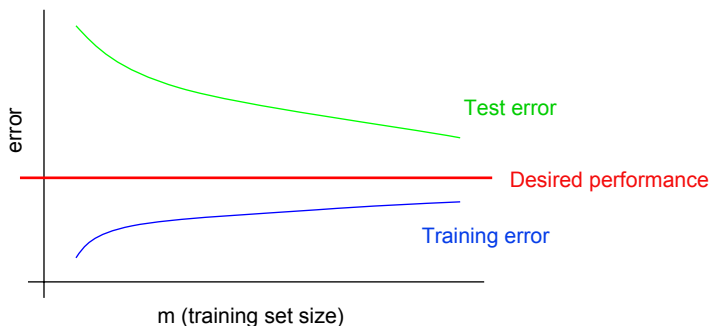
- Overfitting (**varianza** alta)
- Troppe poche feature per una corretta classificazione dello spam (**bias** alto)

Diagnosi:

- Varianza alta: l'errore di training sarà **sostanzialmente inferiore** dell'errore di test
- Bias alto: l'errore di training sarà **alto**, quasi quanto quello di test

Bias vs. varianza e curve di apprendimento

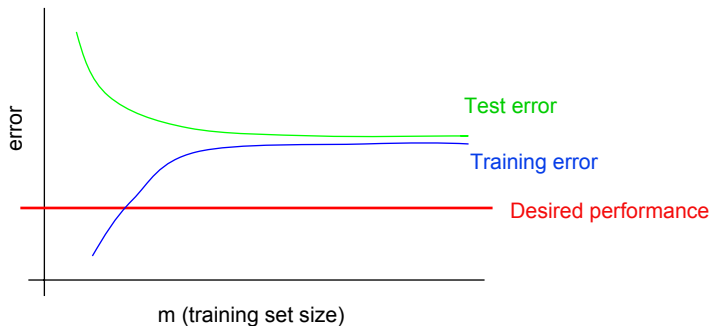
Curva di apprendimento tipica quando la **varianza** è alta:



- L'errore di test continua a decrescere con m crescente. Ciò suggerisce che un training set più grande possa essere utile
- Grande gap tra errore di test ed errore di training

Bias vs. varianza e curve di apprendimento

Curva di apprendimento tipica quando il **bias** è alto:



- Già l'errore di training è troppo alto!
- Piccolo gap tra errore di test ed errore di training

La diagnosi ci dice come intervenire

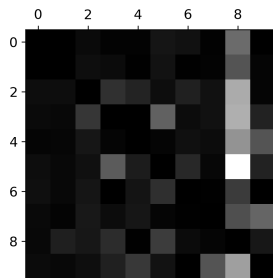
Nel nostro scenario, si può intervenire:

- Ottenendo più esempi di training Diminuisce la varianza
- Provando un insieme di feature più piccolo Diminuisce la varianza
- Provando un insieme di feature più grande Diminuisce il bias
- Usando l'intestazione della mail Diminuisce il bias
- Aumentando il numero di iterazioni di Gradient Descent Corregge l'algoritmo di ottimizzazione
- Usando un metodo del secondo ordine Corregge l'algoritmo di ottimizzazione
- Usando un diverso valore del coefficiente di regolarizzazione λ Aumentare λ diminuisce la varianza
- Usando una Support Vector Machine Corregge l'obiettivo dell'ottimizzazione

Analisi degli errori

Esempio: classificazione cifre MNIST

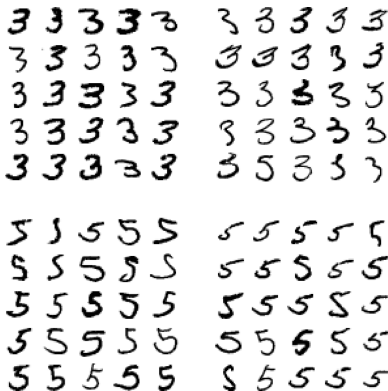
Analizziamo la **matrice di confusione** per capire quali classi vengono confuse con quali altre (**nota**: la diagonale è stata azzerata)



- Molte cifre (ad es., tanti 5) vengono facilmente confuse con degli 8
- I veri 8 vengono per lo più classificati correttamente
- Sembra utile aggiungere esempi che sembrano 8 ma non lo sono

Analisi degli errori

Analizziamo **errori individuali** per capire perché il classificatore fallisce



- Il classificatore sembra troppo sensibile a traslazione e rotazione
- Ciò suggerisce di centrare e “raddrizzare” le immagini

- Il tempo investito nella diagnosi dell'algoritmo di apprendimento è tempo ben investito
- La diagnosi di bias e varianza è molto utile e può essere semplice
- L'analisi degli errori può fornire altre informazioni utili