Introduzione

Machine Learning

Il Machine Learning è un procedimento di **ottimizzazione**

- Definire una classe di **modelli** del problema parametrizzati su di un insieme di **parametri** Θ
- Definire una **metrica di valutazione**: una misura dell'**errore** dei vari modelli
- Modificare i parametri Θ per **minimizzare** l'errore sui **dati di training**

Classi di problemi di apprendimento

- Apprendimento supervisionato: input + output
  - Classificazione, assegnare un nuovo input ad una classe partendo da un insieme finito
  - Regressione, assegnare ad un nuovo input un valore atteso in linea con i valori visti finora
- Apprendimento non supervisionato: solo input
- Apprendimento con rinforzo: azioni e ricompense

Tecniche differenti

- Modi diversi per definire i modeli

Features

Feature = Caratteristica

Le features sono gli input del processo di apprendimento. Con l'approccio tradizionale le features vengono scelte manualmente, con l'approccio *deep* vengono forniti dati grezzi e una rete neurale sintetizza nuove features

Alberi di decisione

Train set: insieme di **esempi di allenamento**

$$\langle x^{(i)}, y^{(i)} \rangle$$

- $x^{(i)} \in X$: insieme degli input
- $y^{(i)} \in Y$: insieme degli ouput
- i indice dell'istanza dell'esempio di traning

Problema: "apprendere" la funzione che mappa $x^{(i)}$ a $y^{(i)}$

- Y discreto: problema di **classificazione**
- Y continuo: problema di **regressioen**

Bisogna partire da uno **spazio di funzioni** H all'interno del quale cercare la funzione che meglio approssima il problema

Albero di decisione: - Ogni nodo testa una feature X - Ogni arco uscente da un nodo corrisponde ad uno dei possibili valori **discreti** di X - Ogni foglia predice la risposta Y con una probabilità condizionata

Entropia

Data X variabile aleatoria e n i possibili valori di X

$$H(X) = -\sum_{i=1}^{n} P(X = i) \log_2 P(X = i)$$

Misura il *grado di impurità* dell'informazione

Informazione

Quantità media di informazione prodotta da una sorgente stocastica di dati

Definiamo $I(p) = -\log(p)$

Guadagno informativo

> Riduzione di entropia della variabile target Y conseguente alla conoscenza di qualche attributo X

Usato per definire il migliore attributo durante la costruzione di alberi di decisione

Costruzione Top Down induttiva

1. Assegnare al nodo corrente il *miglior* attributo X_i
2. Creare un nodo figlio per ogni possibile valore di X_i e propagare i dati versi i figli a seconda del loro valore
3. Per ogni figlio, se tutti i dati del training set del nodo hanno la stessa etichetta y marcare il nodo come foglia con etichetta y, altrimenti ripetere da (1)

Con gli attributi continui prendiamo le decisioni basandoci sulle soglie

Overfitting

> Una funzione $h \in H$ *overfitta* il traning set se esiste un'altra funzione per cui l'errore per il traning set è maggiore, ma per l'intero data set è minore

Per risolvere il problema si dividono i dati disponibili in due insiemi disgiunti, un training set e un validation set, utilizzato per misurare accuratezza e overfitting di h

Per evitare overfitting ci sono due tecniche - Early stopping: terminare la costruzione dell'albero quando il miglioramento del modello è statisticamente insignificante - Post pruning: Si sviluppa l'intero albero e si procede a *potarlo* all'indietro

Impurità di Gini

> Nozione alternativa di entropia, misura la probabilità che un generico elemento sia mal classificato in base alla classificazione corrente

Date m categorie, sia f_i la frazione dei dati con etichetta i.

$$I_G(f) = \sum_{i=1}^{m} f_i(1 - f_i) = 1 - \sum_{i=1}^{m} f_i^2$$

Per avere una valutazione della qualità dell'attributo i valori sono sommati in modo pesato per ogni nodo figlio.

Aspetti negativi

- Rischio elevato di overfitting
- Facile che gli alberi siano sbilanciati con classe dominanti

Gli alberi di decisione sono utilizzati nelle **random forest**, dove numerosi alberi vengono usati insieme per prendere decisioni

Approccio probabilistico

> Invece di calcolare $f : X \to Y$ possiamo calcolare $p : P(Y \mid X)$

Date n feature booleane, costruiamo una tabella con 2^n righe con tutte le possibili combinazioni tra i valori delle feature (distribuzione congiunta).

Dobbiamo stimare le probabilità di ogni combinazione di valori, con questa possiamo calcolare la probabilità di ogni evento con la regola di Bayes.

Il problema principale di questo approccio è che il numero di parametri da calcolare è esponenziale, e per ottenere valori significativi bisogna avere molti esempi per ogni combinazione.

Naive Bayes

Possiamo sfruttare la regola di Bayes per calcolare $P(Y \mid X)$, calcolando a priori $P(Y)$ e $P(X_1, X_2, \dots, X_n \mid Y)$.

A questo punto per ridurre la complessità **assumiamo** che, dato Y, gli X_i siano indipendenti tra loro.

Quindi abbiamo: - Bayes rule

$$P(Y = y_i \mid X_1 \dots X_n) = \frac{P(Y = y_i) \cdot P(X_1 \dots X_n \mid Y = y_i)}{P(X_1 \dots X_n)}$$

- Naive Bayes

$$P(Y = y_i \mid X_1 \dots X_n) = \frac{P(Y = y_i) \cdot \prod_j P(X_j \mid Y = y_i)}{P(X_1 \dots X_n)}$$

- Classificaione di un nuovo dato

$$Y^{new} = arg\ max_{y_i} P(Y = y_i) \cdot \prod_j P(X_j = x_j \mid Y = y_i)$$

Per la stima dei parametri: - Training

$$\pi_k = P(Y = y_k) \quad \forall y_k \in Y$$
$$\theta_{ijk} = P(X_i = x_{ij} \mid Y = y_k) \quad \forall x_{ij} \in X_i$$

- Classificazione di $a^{new} = \langle a_1 \dots a_n \rangle$

$$Y^{new} = arg\ max_{y_k} P(Y = y_k) \cdot \prod_i P(X_i = a_i \mid Y = y_k)$$
$$= arg\ max_k \pi_k \cdot \prod_i \theta_{ijk}$$

Tecniche generative e limiti di Naive Bayes

Con il dataset mnist notiamo che tramite la regola di Bayes generiamo una **distribuzione dei dati** data la categoria.

In alcuni casi qualche probabilità $P(X_i \mid Y)$ può essere 0, quindi rendere 0 il prodotto di tutta la formula

Inoltre non possiamo sempre supporre che gli eventi siano indipendenti tra loro dato Y

Classificazione di documenti (bag of words)

Indichiamo con $\theta_{i,word,\ell} = P(X_i = word \mid Y = \ell)$ la probabilità che nel documento di categoria ℓ la parola *word* appaia in posizione i

Assumiamo che tutti gli eventi siano indipendenti tra loro e che la probabilità sia indipendente dalla loro posizione (assunzioni discutibili) - Training

$$\pi_k = P(Y = y_k)$$
$$\theta_{ijk} = P(X_i = x_{ij} \mid Y = y_k)$$

- Classificazione di $a^{new} = \langle a_1 \dots a_n \rangle$ (sequenza di n parole)

$$Y^{new} = arg\ max_{y_k} P(Y = y_k) \cdot \prod_i P(X_i = a_i \mid Y = y_k)$$
$$= arg\ max_k \pi_k \cdot \prod_i \theta_{ijk}$$

- Maximum Likelihood Estimates: Possiamo passare al logaritmo quando calcoliamo Y^{new}, inoltre sapendo che $\theta_{ijk} = \theta_{i'jk} = \theta_{jk}$

$$Y^{new} = arg\ max_k \log(\pi_k) \cdot \sum_i n_j \cdot \log(\theta_{jk})$$

Con n_j numero di occorrenze della parola w_j nel documento da classificare

Vettore "spettrale"

Consideriamo dei vettori della stessa dimensione del vocabolario - Training

$$s_k = \langle \log(\theta_{jk}) \rangle_{j \in words}$$

Con θ_{jk} frequenza della parola j nei documenti di categoria k - Classificazione - Calcoliamo il vettore d per il nuovo documento - Classifichiamo il documento in base alla categoria il cui spettro è maggiormente correlato ad esso

$$arg\ max_k \underbrace{d \cdot s_k}_{correlazione} = arg\ max_k \sum_j d_j \cdot s_{jk}$$

Natura lineare di Naive Bayes

Indichiamo come **tecniche di classificazione lineari** gli algoritmi basati su una combinazione lineare delle feature. Ovvero che ogni caratteristica del dato è valutata indipendentemente dalle altre

Quello che dobbiamo stimare sono i pesi, ovvero uanto ogni caratteristica contibuisce al risultato

Naive Bayes è una tecnica di classificazione lineare

Naive Byes Gaussiano

Se le feature sono contine allora possiamo supporre che $P(X_i \mid Y)$ abbia una distrbuzione **gaussiana**, e usare le densità di probabilità

Distrubuzione Gaussiana

- Densità
$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$
- Valore medio: $E[X] = \mu$
- Varianza: $Var[X] = \sigma^2$
- Deviazione standard: $\sigma_X = \sigma$

Accuratezza, Precisione e Richiamo

- Accuratezza (Accuracy) = Istanza classificate correttamente
$$Accuratezza = \frac{TP + TN}{All}$$
- Precisione (Precision) = Precisione sui positivi
$$Precisione = \frac{TP}{TP + FP}$$
- Richiamo (Recall) = Percentuale dei positivi classificati come tali
$$Richiamo = \frac{TP}{TP + FN}$$

Modello

- Training
$$\mu_{ik}, \quad \sigma_{ik}, \quad \pi_k = P(Y = y_k)$$
- Classificazione di $a^{new} = \langle a_1 \dots a_n \rangle$ (sequenza di n parole)
$$Y^{new} = arg\ max_{y_k} P(Y = y_k) \cdot \prod_i P(X_i = a_i \mid Y = y_k)$$
$$= arg\ max_k \pi_k \cdot \prod_i N(a_i \mid \mu_{ik}, \sigma_{ik})$$
  - μ_{ik} è il valore medio di X_i per i dati con etichetta $Y = y_k$
  - σ^2_{ik} è la varianza di X_i per istanze con etichetta $Y = y_k$

Regressione e funzione logisitica

L'idea alla base è di cercare direttamente $P(Y \mid X)$ senza passare da $P(Y)$ e $P(X \mid Y)$

Nel caso di Naive Bayes la forma che assume $P(Y \mid X)$ è:

$$P(Y = 1 \mid X = \langle x_1 \dots x_n \rangle) = \frac{1}{1 + exp(w_0 + \sum_i w_i x_i)}$$

Nota: Usando i logaritmi possiamo classificare X guardando se $w_0 + \sum_i w_i x_i > 0$

Con la Regressione Logistica assumiamo che:

$$P(Y = 1 \mid X = \langle x_1 \dots x_n \rangle) = \frac{1}{1 + exp(w_0 + \sum_i w_i x_i)}$$

E cerchiamo di stimare i parametri w_i direttamente

Nota: La funzione $\sigma(x) = \frac{1}{1+e^{-x}}$ è detta **funzione logistica**

A questo punto sappaimo che

$$P(y = 1 \mid x, w) = \sigma(w_0 + \sum_i w_i x_i)$$

Per trovare i parametri w_i possiamo passimizzare il Log Likelihood, ovvero

$$\sum_\ell (y^\ell \cdot \log P(Y = 1 \mid x^\ell, w) + (1 - y^\ell) \cdot \log P(Y = 0 \mid x^\ell, w))$$

Non esistendo una soluzione analitica per trovare i w_i usiamo la tecnica del gradiente

Tecnica del gradiente

Con metodi lineari il minimo locale è anche globale, per le reti neurali non è detto

Solitamente si misura l'errore tramite **Mini-Batch**, ovvero calcolo gradiente ed errore su un sottoinsieme di dati, buon compromesso

Per la Regressione Logistica

- Probabilità che istanza ℓ abbia etichetta $Y = 1$

$$P(Y = 1 \mid x^\ell, w) = \sigma(w_0 + \sum_i w_i x_i^\ell) = \alpha^\ell$$

- Log Likelihood $l(w)$

$$\sum_\ell \log P(Y = y^\ell \mid x^\ell, w) =$$
$$= \sum_\ell y^\ell \log(\alpha^\ell) + (1 - y^\ell) \log(1 - \alpha^\ell)$$

- Gradiente

$$\frac{\partial l(w)}{\partial w_i} = \sum_\ell x_i^\ell \cdot (y^\ell - \alpha^\ell)$$

- Derivata della funzione logistica (molto piatta)

$$\frac{\partial \sigma(z)}{\partial z} = \sigma(z)(1 - \sigma(z))$$

- Training

$$w_i \leftarrow w_i + \mu \sum_\ell x_i^\ell \cdot (y^\ell - P(Y = y^\ell \mid x_i, w_i))$$

  - Frequentemente si aggiunge $-\mu\lambda|w_i|$ (una regolarizzazione) per tenere i parametri w_i vicini a 0 e ridurre l'overfitting

Metodi Generativi e Discriminativi

- Discriminativi
  - L'obbiettivo è apprendere una frontiera che separa le classi
- Generativi
  - L'obbiettivo è apprendere la distribuzione dei dati

Esempio

Problema: Stimare $P(Y \mid X)$

- Classificatore Generativo (Native Bayes)
  - Si assume distrubuzione per $P(X \mid Y)$ e $P(X)$
  - Si stimano i relativi parametri sui dati di training
  - Si utilizza Bayes per inferire $P(Y \mid X)$
- Classificatore Discriminativo (Regressione Logistica)
  - Si assume una distribuzione per $P(Y \mid X)$
  - Si stimano i relativi parametri sui dati di training

Conclusione

I metodi lineari sono molto veloci quando il dataset ha una distribuzione **poco sparsa**, ovvero diventa plausibile dividere le varie classi con **iperpiani**