1. Introduzione

Questi appunti si rifanno alle lezioni 2023/2024 del corso Introduction to Machine Learning tenuto dalla docente Elisa Ricci, al libro 'Deep Learning' di Ian Goodfellow e Yoshua Bengio; ed infine al libro 'Hands on machine learning' di Aurélien Géron pubblicato da O'Reilly.

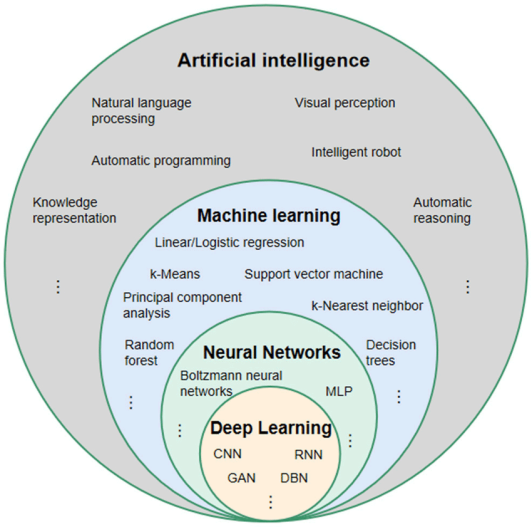

Figure 1: La relazione tra intelligenza artificiale, machine learning e deep learning.

1.1. Dataset

Il dataset è l'insieme dei dati disponibili per l'analisi. Su questo dataset si effettuano le operazioni di training e testing.

Il training set è il sottoinsieme del dataset utilizzato per addestrare il modello; mentre il test set è il sottoinsieme utilizzato per testare il modello. Il validation set è un sottoinsieme del training set utilizzato per regolare gli iperparametri del modello, prima della fase di testing.

Per generare questi sottinsiemi è necessario fare due assunzioni sui dati (*i.i.d. assumption*), ovvero che siano:

- **indipendenti** (non ci sia correlazione tra i dati del training set e del test set)
- **identicamente distribuiti** (prelevati dalla stessa distribuzione di probabilità p_{data})

1.2. Modello

L'obbiettivo, nel Machine Learning, è che il nostro modello performi bene su dati che non ha mai visto prima; questa abilità è detta *generalizzazione*.

Durante la fase di training, (durante la quale abbiamo accesso solo al training set) possiamo misurare l'errore

1.3. Underfitting e Overfitting

L'Underfitting si verifica quando il modello non ottiene buone prestazioni ne sul training set, ne sul test set.

L'Overfitting si verifica quando il modello ottiene buone prestazioni sul training set ma non sul test set.

1.4. The No Free Lunch Theorem

Contrariamente a quanto si possa pensare, non esiste un modello che sia il migliore in assoluto per tutti i problemi.

2. Regressione Lineare

Come suggerisce il nome, la regressione lineare è un modello che risolve un problema di regressione, ovvero dato un vettore $\boldsymbol{x} \in \mathbb{R}^n$ in input, restituisce un valore $y \in \mathbb{R}$ in output. L'output della regressione lineare è una funzione lineare dell'input.

Definiamo $\hat{y}$ come il valore che il nostro modello predice, definiamo dunque l'output come:

$$\hat{y} = \boldsymbol{w}^\top \boldsymbol{x}$$

Dove: $\boldsymbol{w}$ è un vettore di parametri.

Questi parametri, anche chiamati pesi, determinano il comportamento del sistema; in questo specifico caso si tratta del coefficiente per cui moltiplichiamo il vettore di input $\boldsymbol{x}$.

$$\hat{y} = \boldsymbol{w}^\top \boldsymbol{x} + b$$

Questa è una *affine function*, ovvero una funzione lineare con una traslazione (b è noto come *intercept term* o *bias*). Come si può notare, inoltre, l'equazione assomiglia molto a quella di una retta in due dimensioni: $y = mx + q$. Infatti per un grado $n = 1$ la regressione lineare è proprio una retta.

Facciamo un breve esempio pratico: supponiamo di avere un dataset con GDP per capita e un valore di soddisfazione della vita per ogni paese del mondo e volessimo costruire un modello che preveda quest'ultimo valore[1].

Prima di tutto plottiamo i dati:

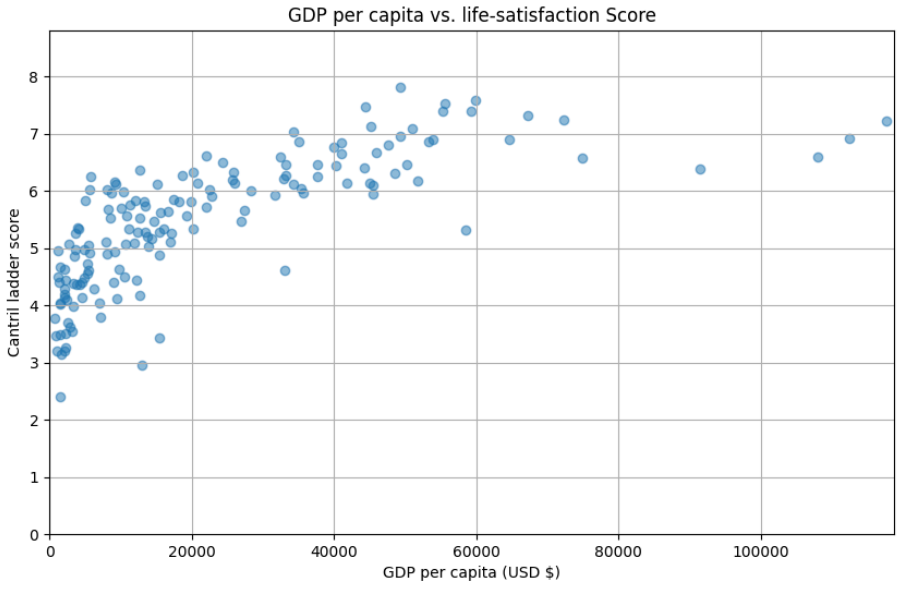

Figure 2: Plot dei dati GDP vs Life-satisfaction degli ultimi dati disponibili per ogni paese. (ex Austria)

Ora proviamo ad utilizzare la regressione lineare per prevedere il livello di soddisfazione della vita in Austria, che abbiamo escluso dal training set, dato il suo GDP per capita:

[1] Questo valore viene misurato con la scala di Cantril.

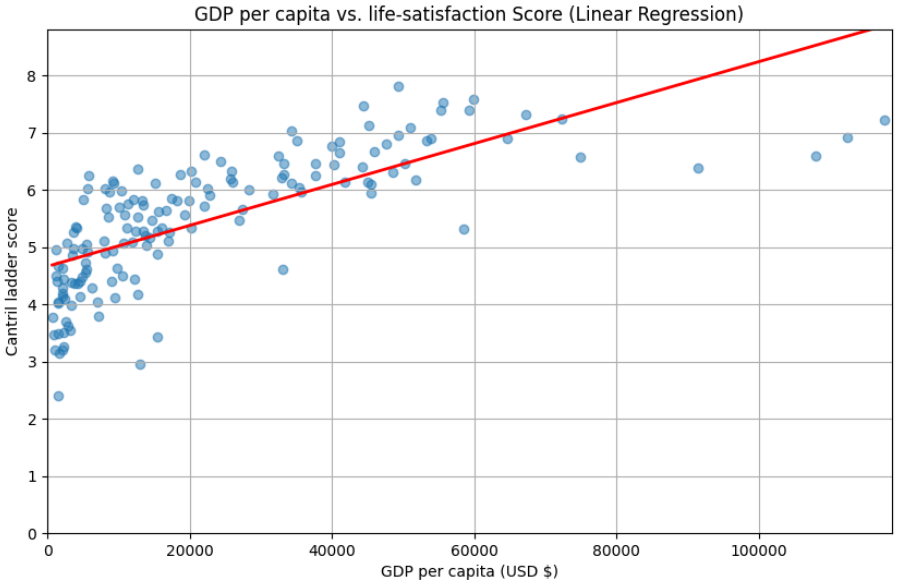

Figure 3: Plot dei dati GDP vs Life-satisfaction con la regressione lineare e grado 1

L'austria nel 2022 aveva un GDP per capita di $55,867 e un livello di felicità di 7,09. Il modello di regressione lineare ci dice che il livello di felicità previsto è di 6,66. Forse possiamo fare di meglio.

Torniamo sulla formula della regressione lineare, possiamo generalizzarla come:

$$\hat{y} = b + w_1 x_1 + w_2 x_2 + \ldots + w_n x_n$$

Dalla formula generalizzata capiamo che la regressione lineare può funzionare anche in più dimensioni, non solo con una variabile indipendente; ed in questo caso si dice "multivariata". Per esempio con 2 variabili indipendenti avremo un piano. Dunque se aggiungessimo lo Human Freedom Index come feature, avremmo un modello tridimensionale:

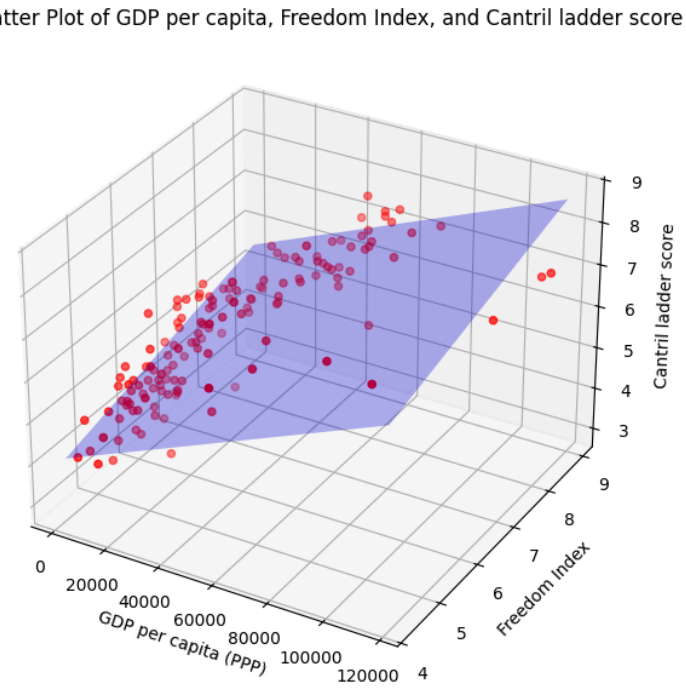

Figure 4: Plot dei dati GDP e Freedom vs Life satisfaction in 3D

In questo caso la predizione del modello per l'Austria è di 6,80, più vicina al valore reale.

3. Polinomial Regression

Nel caso in cui non avessimo altre features a disposizione, o la distribuzione dei dati non fosse lineare, potremmo utilizzare una regressione polinomiale. Nel caso preso in esempio, abbiamo visto come la retta non fosse in grado di generalizzare particolarmente bene i dati. Prima di procedere con la pratica vediamo la formula della regressione polinomiale, anche se è abbastanza intuitiva e non ci dovrebbe essere nulla da spiegare:

$$\hat{y} = wx + b \underset{polinomiale}{\longrightarrow} \hat{y} = b + \sum_{i=1}^{n} w_i x^i$$

Applicando la regressione polinomiale con grado del polinomio: $n = 2$ all'esempio visto in precedenza otteniamo:

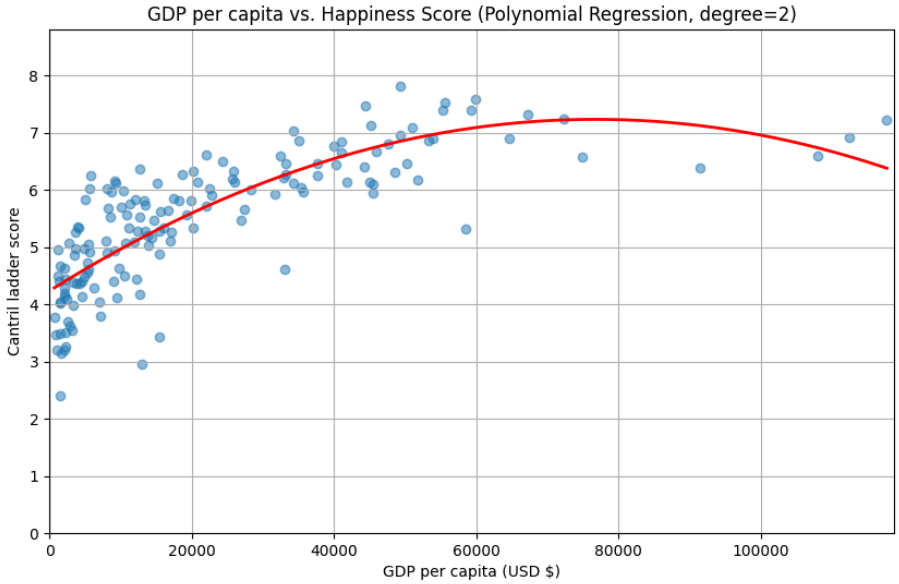

Figure 5: Plot dei dati GDP vs Life satisfaction con la regressione lineare e grado 2

Ora il modello predice un valore di 7,01 per l'Austria, più vicino al valore reale. Proviamo con gradi ancora più alti:

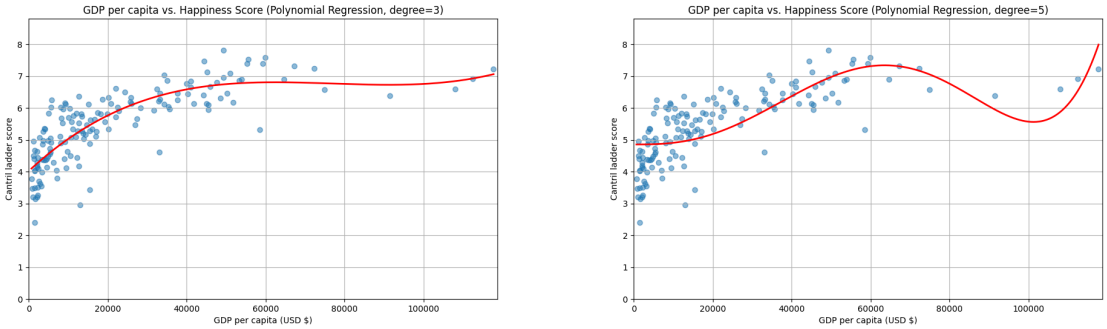

Figure 6: Plot dei dati GDP vs Life satisfaction con la regressione lineare e grado 3 e 5

Con un grado 3 il modello predice un valore di 6,79, mentre con un grado 5 il modello predice un valore di 7,21.

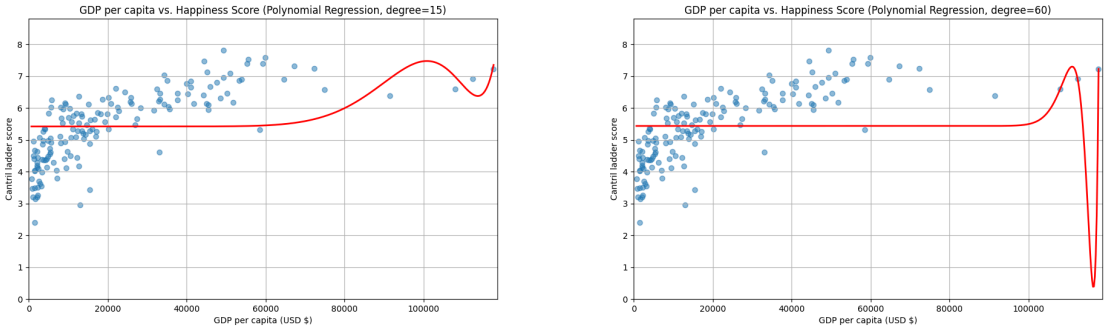

Figure 7: Plot dei dati GDP vs Life satisfaction con la regressione lineare e grado 15 e 60

Se alzialiamo ulteriormente il grado del polinomio, il modello non migliorerà; anzi dalla tabella di seguito e dalle immagini precedenti dovrebbe essere chiaro come avere un grado del polinomio più alto non implichi che il modello generalizzi meglio.

grado	1	2	3	5	15	20	30	40	50	60
predizione	6.66	7.01	6.79	7.21	5.43	5.43	5.43	5.43	5.43	5.43
errore	±0.44	±0.09	±0.31	±0.11	±1.67	±1.67	±1.67	±1.67	±1.67	±1.67

Quindi come possiamo capire quale grado del polinomio è il migliore?

Il grado del polinomio è un **iperparametro** del modello; e per essere determinato correttamente viene utilizzato un **validation set**.

3.1. Regulaization / Regolarizzazione

La regolarizzazione è una qualsiasi modifica che apportiamo al modello per ridurre l'errore di generalizzazione (ma non il training error).

Il comportamento dell'algoritmo è influenzato infatti, non solo dalla capacità del modello (spazio delle ipotesi); ma anche dall'identità delle funzioni utilizzate. Per esempio, la regressione lineare ha uno spazio delle ipotesi composto esclusivamente da funzioni lineari e, nel caso non ci sia relazione lineare tra i dati (*e.g.* $\sin(x)$), non sarà in grado di generalizzare bene.

Potremmo modificare il criterio di ottimizzazione per la regressione lineare includendo un termine regolarizzatore (denotato con $\Omega(w)$) nella funzione di costo.

Nello specifico caso del weight decay il rego è uguale a: $\Omega(w) = \boldsymbol{w}^\top \boldsymbol{w}$. Dunque il criterio sarà:

$$J(\boldsymbol{w}) = \text{MSE}_{\text{train}} + \lambda \boldsymbol{w}^\top \boldsymbol{w}$$

in questo modo minimizziamo una somma che comprende sia l'errore quadratico medio sul training set, sia il termine di regolarizzazione. In questo caso il termine λ è un iperparametro che regola l'importanza del termine di regolarizzazione. Con $\lambda = 0$ il modello si comporta come una regressione lineare standard, mentre con $\lambda > 0$ il modello tenderà a preferire pesi più piccoli, da questo il nome weight decay.

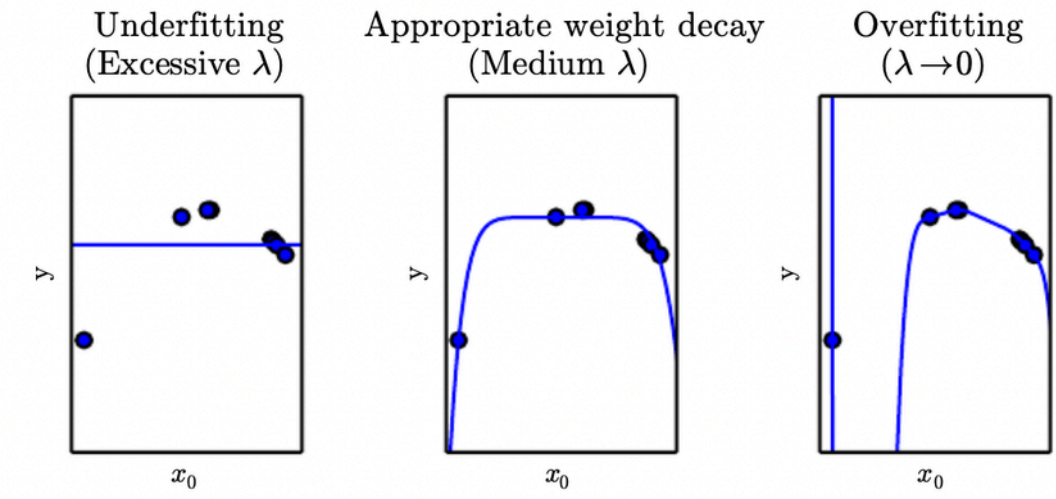

Figure 8: "Il modello utilizzato ha solo funzioni di grado 9, mentre il dataset è generato da una funzione quadratica."

Nel campo del Machine Learning esistono diverse varianti per quanto riguarda le tecniche di regolarizzazione.

Famiglia delle L^p norme; generalizzata con la formula:

$$\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}$$

con n ad indicare le dimensioni e $p \in [1, +\infty)$.

- La norma 1 è banalmente la somma dei valori assoluti dei componenti.
- La norma 2 o Norma Euclidea, è la radice quadrata della somma dei quadrati dei valori:

$$\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$$

.

3.2. Dataset Augmentation

Il miglior modo per avere un modello che generalizza bene è trainarlo su più dati e, spesso, il dataset a disposizione non è abbastanza grande. Un modo per risolvere questo problema sono le tecniche di dataset augmentation. Questo approccio è molto efficace con le task di classificazione, object recognition e speech recognition. Com'è facile immaginare per quanto concerne l'object recognition, possiamo ruotare, scalare, e traslare le immagini; per lo speech recognition possiamo aggiungere rumore alle registrazioni.

L'iniezione di rumore è alla base di alcuni modelli unsupervised, come il denoising autoencoder. La noise injection può inoltre essere implementata negli hidden layer.

3.3. Hyperparameters

Gli iperparametri sono parametri che non vengono appresi durante il training, ma che influenzano il comportamento del modello.

Molti modelli di Machine Learning hanno iperparametri, per quanto riguarda la regressione lineare, di base, ha solo il grado del polinomio. Il grado del polinomio, come abbiamo visto precedentemente determina la capacità del modello.

Allo stesso modo λ nella regolarizzazione è un iperparametro.