Lezione 1 (Russo Russo) - 13 ottobre

per evitare di utilizzare 'pip install' che potrebbe rompere tutto il progetto si utilizza la creazione di un ambiente isolato.

metodo 1

python quando è installato fornisce venv per creare una cartella tramite pip tutte le cose installate verranno installate dentro quella cartella, vantaggio di non dover installare software esterno problema: costretto a utilizzare versione python del sistema operativo, aggiornando il SO e python di conseguenza si potrebbero avere problemi con le librerie installate Vantaggio non c'è bisogno di installare tool di terze parti

```
#create environment (nella directory corrente)
python -m venv my_env
#attivare la cartella (tutte le installazioni vengono
inserite nella cartella di interesse)
source ./my_env/bin/activate
#installare libreria
pip install scikit-learn
#una volta finito chiudere terminale o utilizzare il
comando:
deactivate
```

Buona pratica avere file di testo '.txt' e installare le librerie tramite comando

```
pip install -r requirements.txt
```

in modo che se si butta l'environment si ri-installano facilmente con un solo comando

metodo 2 (migliore)

consente di avere ambiente più isolato (ognuno ha il suo interprete python, non si aggiorna insieme al SO). conda è il software utilizzato per fare questo. Per installare conda ci sono diverse distribuzione (più famosa AnaConda, problema pesante.. installa insieme diverse librerie; si installa da google).

Una volta installata sul pc si ha il comando conda. Si possono eseguire seguenti comandi:

```
#Create env (posso selezionare la versione di python che
voglio)
conda create -n ENVNAME python=3.10
#attivarlo per installarci dentro le varie librerie
conda activate ENVNAME
#installare cose
```

```
conda install jupyterlab
conda install scikit-learn
#per avviare jupyter
jupyter lab
#per uscire
conda deactivate
```

dopo aver installato conda si sta già in un ambiente di base per evitare e quindi utilizzare conda se serve in un altro env utilizzare il comando:

```
conda config --set auto_activate_base false
```

importare ed esportare ambiente

Una volta creato un ambiente può essere esportato, questo viene scritto in un file che conterrà nome dell'ambiente, librerie ecc. A partire da questo può essere ricreato in automatico l'ambiente con le stesse librerie e stesse cose.

```
#Export to file
conda env export > ENV.yml
#Cross-platform export
conda env export --from-history > ENV.yml

#Import
conda env create -n ENVNAME --file ENV.yml
```

metodo 3 (pigri)

I precedenti buoni per avere librerie sul pc. Applicazione tramite browser in cui non si scarica nulla sul pc, si utilizza Google Colab per eseguire codice sui server di Google (gratis con account google, può connettersi con drive e github per caricare notebooks). Non si può usare per calcoli pesanti. https://colab.research.google.com Permette uso di GPU (utili per addestrare reti neurali). da cambia tipo runtime è possibile selezionare l'utilizzo della GPU.

Numpy & Pandas

librerie per lavorare con i dati. numpy numerical oython pensata per far lavorare oython cion collezioni di dati numerici. famosa per ndarray (array multidimensionale che contine enumeri), importante perché in python per memorizzare array di numeri si crea una lista (struttura dati poco efficiente), ndarray particolarmente efficienti (tutte funzionalità offerte implementate in C o C++, in modo efficiente). si importa per convenzione come

```
import numpy as np
```

posso creare array A

```
import numpy as np
A=np.array([1,2,3])
print(A) #stampa array
```

permette anche di fare operazioni vettoriali direttamente come `A+B` per somma componente per componente.

Si può creare matrice creando lista di liste.

Si può creare vettore di zeri `np.zeros(size)`, o matrice di zeri `np.zeros((righe, colonne))`. per sapere dimensione si usa `A.shape`.

È utile a volte cambiare dimensioni del nostro oggetto. Per fare questo si può utilizzare `A.reshape(righe, colonne)`

Per fare il trasposto di una matrice utilizzare `A.transpose()`

Per la moltiplicazione tra matrici `np.matmul(mat1, mat2)`.

come in python si può indicizzare un elemento, si può anche assegnare un valore ad un'intera slice (A[1:10]=1.0, assegno 1.0 a tutti gli elementi da 1 a 10).

Pandas

fornisce strutture dati e funzioni per lavorare con dati non necessariamente numerici. fornisce Series e dataframe

`import pandas as pd`

Series array unidimensionale etichettato può contenere dati di tipo diverso, elementi all'interno della stessa serie il tipo deve essere uniforme. I dati contenuti sono etichettati (etichette prendono nome di indice). simili a ndarray ma con etichette. quando creo una serie posso passare valori e indice.

Dataframe array multidimensionale (ogni colonna è una series, deve contenere elementi swllo stesso tipo. le varie colonne possono contenere tipi diversi). può essere creato tramite dizionario.

posso prendere una colonna utilizzando nome delle colonna

loc e iloc sono due funzioni per selezionare elementi della colonna o intere colonne (anche più colonne insieme)

obj.loc[righe da prendere, colonne da prendere] se seleziono più colonne ottengo di nuovo un dataframe basato su etichette iloc è uguale ma funziona con indici e non etichette.

Con pandas è facile leggere file csv (contenenti dataset)

pd.read_csv("nome_file.csv")

Pandas rende semplice preprocessamento e analisi dei dati.

In [4]:
```
import numpy as np
A=np.array([1,2,3])
print(A) #stampa array
```

```
[1 2 3]
```

Scikit-learn

Libreria open source (è possibile vedere algoritmo) per machine learning. Fornisce strumenti per fare reprocessing dei dati, valutare algoritmi ecc.. Tutti gli algoritmi utilizzati vengono chiamati `stimatori`, tutti implementano un'interfaccia comune:

- fit: per addestrare modello in base a dati di addestramento, su questi dati vengono stabiliti dati del modello
- predict: prende nuovi dati e fornisce dati del modello

Esempio

Vedremo un esempio per vedere come funziona e capire come funziona progetto di ML (problema predire prezzi mediani abitazioni delle case della california in base al distretto).

Come affrontare problema

1. Capire esattamente problema che si sta andando a risolvere (perché ci stanno chiedendo di risolvere il problema?), a seconda dello scopo cambia il modo di affrontare il problema di ML.

- informazioni data a potenziali acquirenti, non è molto grave se sbaglio.
- informazioni date a persone che acquistano case per avere un guadagno e decidere se farlo o no in base al prezzo richiesto: serve avere stima accurata, può portare a scelte di investimento sbagliate. Conseguenza dell'errore si riflettono su alcune scelte che faccio nello studio della soluzione.

produco prodotto in fabbrica, ML classsifica prodotto in buono o non buono.
Classificatore fa errori di due tipi:

- prodotto buono come scarto
- prodotto non buono come buono

Se si tratta di farmaci secondo caso più grave. Posso tener conto di questa cosa con una funzione di costo, posso dare un peso all'errore per cercare di minimizzare errore più grave. In questo caso utilizzaremo 'Radice quandrata dell'errore quadratico medio'

In [4]:
```
# Scikit-Learn ≥0.20 is required
import sklearn
assert sklearn.__version__ >= "0.20"
```

```
 #verifico versione libreria compatibile con le altre che sto utilizzando

#pezzo di codice che scarica dataset, lo decomprime e lo scarica nella ca

#check se dataset già esiste non lo scaricare

#A questo punto vado a fare regressione prima di andare a capire che algo
#importo libreria pandas per importare e leggere csv mettendo risultato i
#funzione housing.head() mi dà 5 righe di preview del dataframe.

#Per avere sintesi delle colonne si usa funzione housing.info(), che mi d
#molti perché magari un sensore non funziona in un certo tempo).

#C'è poi una colonna target, quella che vogliamo predire

#housing["nome_colonna"].value_counts() mi dice per ogni valore quante op

#housing.describe() dà una serie di valori che mi fanno capire come varia

#housing.hist(..) serve a graficare il dataset e capire come sono distrib

#Più valori nulli si hanno peggio è per addestrare modello
```

Costruzione testing set

test_size mi dice percentuale dei dati che andanno nel test set tra 0 e 1 nell'esempio 20%

`train_set` ,

`test_set = train_test_split(housing, test_size=0.2, random_state=42)`

`random_state` è il seed serve a rendere ripetibile l'esecuzione

Secondo metodo

Nel dato di partenza si possono avere dati raccolti da un campione non uniformemente distribuito magari ho una minoranza che esprime un'opinione se genero un training set o testing set scompare completamente la presenza dell'informazione su questa minoranza, crea un problema se non ho un dataset molto grande. Per evitare questo posso fare un sampling stratificato. invece di prenderlo a caso suddivido i dati in base a un certo attributo in modo da prendere dati mantenendo la proporzione presente nel dataset originale in base a quell'attributo.

Creo un attributo in cui vado ad aggregare i distretti in base al valore del reddito, utilizzo poi la classe StratifiedShuffleSplit per utilizzare poi la funzione split specificando tra i parametri la variabile costruita. In questo modo vengono costruiti un nuovo train set e test set e confronta le proporzioni ottenute con quelle iniziali

Prima di passare all'algoritmo del ML ragiono sul training set. lo metto in housing.

Distribuzione geografica dei dati

per prima cosa posso vedere come sono distribuiti i dati che ho geograficamente. Posso vedere poi quanto costano le case dei vari distretti (vedo che costano di più sulla costa)

Correlazione

per capire di più se ci sono correlazioni tra le diverse colonne posso usare la funzione corr che restituisce una matrice. A me interessa la colonna che mi dice come sono correlate le altre rispetto al costo quindi seleziono la colonna del costo. Se vicino a 1 aumentando quel valore aumennta l'altro. -1 aumentando quel valore diminuisce l'altro. 0 scorrelato.

Feauturing engeenering

Fare questa analisi mi consente di capire che ci possono essere relazioni tra attributi che mi consentono di capire qualcosa in più. posso cercare di capire se creare nuovi attributi (creando nuove colonne) può essere utile. Una volta create queste colonne per vedere se sono utili vado a calcolare di nuovo la correlazione.

nota

Se questa analisi l'avessi fatta con i dati del test set avrei barato perché avrei preso questi nuovi attributi in modo tale che funzionino con il test set (e non con altri insiemi)

nota

la colonna da predire va scartata nel testing set e viene messa in una nuova variabile per conservarla.

Gestione valori mancanti

Ho 3 strategie:

1. Posso scartare righe con valori mancanti (se sono poche queste righe può essere buon approccio), posso decidere di scartare colonna se mancano tanti valori in quella colonna
2. cerco di rimpiazzare valori mancnati con stime
3. sostituisco valori mancanti con valore di default (SimpleImputer(strategy=...))

In [ ]: