OOP Programming and Scripting in Python

Keywords:

- ML = Machine Learning,
- SKLEARN = sci-kit learning (sklearn)
- SKANN = scit-kit neural networks (sknn)
- ORANGE = SW toolkit Orange (machine learning tool con interfaccia visuale)
- REFLECTION = programmazione riflessiva (pattern di SW eng., metaclassi, ecc.)
- BIOINFO = Bioinformatics,
- BIOMED = biomedicine,
- BIOPY = Biopython (libreria Python per la bioinformatica e la biomedicina)
- ETC = Altro

Elenco temi per progettini

- Implementazione della classe DeepNetwork, come wrapper per implementare reti neurali artificiali multilayer

  [keywords: ML, SKLEARN, SKANN]

  Bisogna definire la sintassi (in formato testo) per la descrizione di una deep network e implementare la classe. Sulla base della descrizione esterna (tipicamente da leggere da file), la classe deve poter generare un'istanza di ANN multilayer. I layer della ANN devono anche poter essere costituiti da CNN (convolutional neural networks) o autoencoder.

  PS Non si pretende di realizzare uno strumento di grande generalità. Ovviamente tutto dipende dall'espressività della descrizione associata alla rete da generare automaticamente. Ai fini del progettino ci si concentrerà sugli aspetti più semplici.

- Implementazione della classe imgANNClassifier

  [keywords: ML, SKLEARN, SKANN]

  La classe dovrà essere in grado di caricare da file dataset composti da varie immagini (scegliere almeno tre dataset dal repository Kaggle), le trasforma in un formato 3DArray per i dati e 1DArray per i target, e poi addestra una rete neurale a partire dagli input così ottenuti.

  NB Il formato dei dati è 3DArray perché sulla riga 0 c'è la prima immagine 2D, sulla riga 1 la seconda, e così via. Si può comunque optare anche per un formato 2DArray, in cui ogni immagine viene "linearizzata" con una semplice istruzione numpy.reshape.

- Implementazione di un widget per il tool Orange (https://orange.biolab.si)

  [keywords: ML, ORANGE]

  Bisogna definire un widget per poter inserire tra le scelte di Orange anche un'applicazione sviluppata esternamente al progetto stesso. Quale sia l'applicazione non è molto

importante. La difficoltà principale qui sta nell'utilizzare la programmazione basata su eventi e le raccomandazioni degli sviluppatori di Orange per consentire l'embedding di applicazioni esterne dentro il toolkit.

NB Per realizzare questo progettino occorre avere almeno un'idea di base sulle tecniche di programmazione basata sugli eventi.

- Implementazione di alcuni pattern noti di ingegneria del software

  [keywords: REFLECTION]

  A differenza di altri linguaggi applicativi della stessa classe, Python è altamente configurabile e consente in particolare di cambiare il comportamento di default nella creazione e gestione degli oggetti.

  Variante 1. Debugging SW pattern. Occorre creare un decoratore che consenta di specificare –se premesso a una definizione di classe– quali metodi devono essere sottoposti a debug, e per ogni metodo se si adotta il debugger standard (da realizzare anch'esso) oppure un debugger alternativo. Occorre sviluppare due versioni del decoratore: una che usa soltanto delle classi e l'altra che usa le function closure.

  PS Function closure in Python: (per es.) https://sahandsaba.com/python-decorators.html

  Variante 2. Delegation. Occorre sviluppare una classe server le cui istanze fanno da "front end" al codice cliente. L'istanza della classe server in sé non è in grado di fare nulla, però può delegare altro codice a svolgere il compito richiesto. Perché ciò accada occorre istruire l'istanza su come instradare le richieste. In pratica l'istanza deve avere informazioni sulle funzioni da invocare per svolgere i compiti richiesti. Tutto questo in run time.

  NB Su richiesta, possono essere scelti anche altri pattern di ingegneria del software da implementare.

- Studio di dataset di biomedicina (gene expression per breast cancer)

  [keywords: ML, BIOINFO, BIOMED]

  Gli studi di espressione genica sono molto utilizzati in biomedicina. Un dominio tipico di studio è quello in cui si vuole individuare la correlazione tra espressioni geniche e tumori alla mammella. Per fare questo occorre organizzare opportuni esperimenti di apprendimento automatico. In questo caso l'input sarà costituito da espressioni geniche ottenute mediante microarray, e si vuole predire l'insorgenza o meno di ricadute entro 3 o 5 anni.

  Variante 1: Analisi univariata. Bisogna effettuare esperimenti e comparare i risultati ottenuti mediante tecniche classiche con quelli ottenuti utilizzando i diagrammi phidelta.

  Variante 2: Analisi multivariata. Per lo studio si useranno tecniche classiche (es. Lasso, Cox, Elastic Net).

Variante 3: Analisi Multivariata. In alternativa si possono usare tecniche che utilizzano le reti neurali artificiali come encoder capaci di mettere in evidenza eventuali combinazioni di feature utili per la predizione.

In tutti I casi occorre scrivere il codice Python per effettuare esperimenti.

- Studio di dataset di biomedicina (altre tematiche relative a espressione genica)

  [keywords: ML, BIOINFO, BIOMED]

  Idem come sopra, con focalizzazione su dataset relativi a tematiche simili ma diverse (es. espressione genica per altri tipi di tumore o per malattie multifattoriali)

  Può generare molte varianti.

- Encoding PSSM di proteine

  Bisogna generare una classe wrapper per ricerche su repository di bioinformatica. Un'istanza della classe prende in ingresso un proteina target. Il metodo "search" cerca in uno tra i possibili repository selezionati e restituisce la matrice PSSM. La classe da implementare deve poter agevolmente gestire i vari parametri che si possono impostare per modificare la ricerca e i soi risultati. [keywords: BIOPY]

- Analisi di serie storiche economiche (es. indici di borsa)

  [keywords: ML, SKLEARN, SKNN]

  Lo studio delle serie storiche è un argomento importante di economia finanziaria. Spesso a partire dai dati grezzi vengono calcolate misure ritenute di utilità per effettuare predizioni.

  Variante 1. In questo caso occorre scrivere alcune classi per facilitare l'utente nell'analisi delle serie realizzata con strumenti classici (es. ARMA models, ARIMA models, ecc.).

  Variante 2. Preprocessing dei dati. In questo caso occorre concentrarsi sulle attività di preprocessing finalizzate a generare I campioni da utilizzare (es. varie medie mobili, up trend, down trend, ecc.).

- Attribuzione di linee di credito a clienti privati da parte di enti finanziatori

  [keywords: ML, SKLEARN, SKNN]

  I sistemi di supporto alle decisioni per decidere se conedere o meno un finanziamento a chi ne fa richiesta sono oggi strumenti fondamentali per gli enti erogatori.

  Si vuole studiare e realizzare un piccolo sistema che decide se erogare o meno il credito sulla base delle informazioni disponibili sull'utente. La classe principale (di interfaccia

verso l’operatore) utilizzerà a sua volta un sistema basato su reti neurali artificiali per prendere una decisione. Il risultato sarà un valore float in [0,1] oppure [-1,+1].

Per portare a termine il lavoro occorrerà reperire su Internet (es. UCI, Kaggle o altro repository online) uno o più dataset su cui effettuare le sperimentazioni.