Metodi e strumenti utilizzati

Machine Learning

Il Machine Learning, o apprendimento automatico, è un campo di studio che si occupa di sviluppare algoritmi per i calcolatori che sono in grado di migliorare automaticamente grazie all'esperienza acquisita tramite l'utilizzo dei dati. Gli algoritmi di Machine Learning creano modelli matematici a partire da dati di esempio, chiamati "dati di training", in modo da poter fare predizioni o prendere decisioni senza essere esplicitamente programmati per farlo. Ci sono tre categorie di approcci di Machine Learning: l'apprendimento supervisionato, l'apprendimento non supervisionato e l'apprendimento per rinforzo.

L'apprendimento supervisionato consiste nell'insegnare al calcolatore una regola generale che mappi gli input e gli output desiderati. L'algoritmo di apprendimento è fornito di dati di input di esempio e degli output corrispondenti e, attraverso successive iterazioni, costruisce un modello matematico che può essere usato per predire l'output associato a un nuovo input.

L'apprendimento non supervisionato, invece, non fornisce all'algoritmo di apprendimento le etichette desiderate. In questo caso, l'algoritmo di apprendimento deve estrarre le informazioni significative dai dati di input senza conoscere a priori l'output desiderato.

L'apprendimento per rinforzo prevede che un programma interagisca con un ambiente dinamico in cui deve raggiungere un obiettivo specifico, come ad esempio guidare un veicolo o vincere un gioco contro un avversario. Durante l'interazione, il programma riceve un feedback sotto forma di premio e cerca di massimizzarlo, in modo da imparare a raggiungere l'obiettivo prefissato.

Computer Vision

La Computer Vision è un campo interdisciplinare che si occupa della capacità dei computer di acquisire conoscenza da immagini o video, cercando di replicare il funzionamento dell'apparato visivo umano. La Computer Vision utilizza metodi per l'acquisizione e l'analisi di immagini digitali, in modo da estrarre dati multidimensionali dal mondo reale e produrre informazioni numeriche o simboliche, come decisioni. La Computer Vision si avvale di conoscenze di geometria, fisica, statistica e teoria dell'apprendimento per descrivere il mondo in modo sensato, producendo pensieri che possono portare alla corretta linea d'azione.

Visual Studio Code

Visual Studio Code è un editor di codice sorgente sviluppato da Microsoft per Windows, Linux e macOS, che supporta il debugging, il controllo Git integrato, la Syntax Highlighting, l'IntelliSense, lo Snippet e il refactoring del codice. Visual Studio Code supporta molteplici linguaggi e funzionalità aggiuntive grazie alla possibilità di installare dei plugin disponibili attraverso un repository centrale. Nel testo, si fa riferimento all'utilizzo di Visual Studio Code con il supporto a Python per il lavoro descritto.

Pandas

La libreria software open source Pandas è stata sviluppata per il linguaggio di programmazione Python ed è utilizzata per la manipolazione e l'analisi dei dati. Con Pandas è possibile effettuare operazioni su tabelle numeriche e serie temporali grazie alle sue strutture dati. Il nome "Pandas" deriva dal termine econometrico "Panel Data", che indica un insieme di dati contenenti osservazioni sugli stessi individui durante più periodi di tempo.

Cuda

[da https://developer.nvidia.com/cuda-zone]

CUDA® è una piattaforma di calcolo parallelo e un modello di programmazione sviluppato da NVIDIA per il calcolo generale su unità di elaborazione grafica (GPU). Con CUDA, gli sviluppatori possono aumentare significativamente la velocità delle applicazioni di calcolo sfruttando la potenza delle GPU.

Nelle applicazioni con accelerazione GPU, la parte sequenziale del carico di lavoro viene eseguita sulla CPU - ottimizzata per le prestazioni single-threaded - mentre la parte computazionalmente intensiva dell'applicazione viene eseguita su migliaia di core GPU in parallelo. Quando si utilizza CUDA, gli sviluppatori programmano in linguaggi popolari come C, C++, Fortran, Python e MATLAB ed esprimono la parallelismo attraverso estensioni sotto forma di poche parole chiave di base.

Il toolkit CUDA di NVIDIA fornisce tutto il necessario per sviluppare applicazioni con accelerazione GPU. Il toolkit CUDA include librerie accelerate su GPU, un compilatore, strumenti di sviluppo e il runtime CUDA.

Random forest

Da [https://www.ibm.com/topics/random-forest#:~:text=Random%20forest%20is%20a%20commonly,both%20classification%20and%20regression%20problems.]



La random forest è un algoritmo di apprendimento automatico comunemente usato, marchiato da Leo Breiman e Adele Cutler, che combina l'output di più alberi decisionali per raggiungere un singolo risultato. La sua facilità d'uso e la flessibilità hanno alimentato la sua adozione, in quanto gestisce sia problemi di classificazione che di regressione.

Alberi decisionali

Dato che il modello di random forest è composto da più alberi decisionali, sarebbe utile iniziare descrivendo brevemente l'algoritmo dell'albero decisionale. Gli alberi decisionali partono da una domanda di base, come ad esempio "Dovrei fare surf?". Da lì, è possibile porre una serie di domande per determinare una risposta, come "C'è un'onda di lungo periodo?" o "Il vento soffia a riva?". Queste domande costituiscono i nodi decisionali dell'albero, agendo come mezzo per suddividere i dati. Ogni domanda aiuta un individuo a giungere a una decisione finale, che sarebbe indicata dal nodo foglia. Le osservazioni che soddisfano i criteri seguiranno il ramo "Sì" e quelle che non li soddisfano seguiranno il percorso alternativo. Gli alberi decisionali cercano di trovare la miglior suddivisione per i dati e vengono tipicamente addestrati attraverso l'algoritmo Classification and Regression Tree (CART). Metriche come l'impurità di Gini, il guadagno di informazione o l'errore quadratico medio (MSE) possono essere utilizzati per valutare la qualità della suddivisione.

Questo albero decisionale è un esempio di un problema di classificazione, dove le etichette di classe sono "fare surf" e "non fare surf".

Sebbene gli alberi decisionali siano comuni algoritmi di apprendimento supervisionato, possono essere soggetti a problemi come il bias e l'overfitting. Tuttavia, quando più alberi decisionali formano un insieme nell'algoritmo di random forest, predicono risultati più accurati, in particolare quando i singoli alberi non sono correlati tra loro.

L'algoritmo della random forest

L'algoritmo della random forest è un'estensione del metodo di bagging in quanto utilizza sia il bagging che la casualità delle caratteristiche per creare una foresta di alberi decisionali non correlati. La casualità delle caratteristiche, anche nota come bagging delle caratteristiche o "il metodo del sottospazio casuale", genera un sottoinsieme casuale di caratteristiche che assicura una bassa correlazione tra gli alberi decisionali. Questa è una differenza chiave tra gli alberi decisionali e le foreste casuali. Mentre gli alberi decisionali considerano tutte le possibili suddivisioni delle caratteristiche, le foreste casuali selezionano solo un sottoinsieme di quelle caratteristiche.

Se torniamo all'esempio "dovrei fare surf?", le domande che potrei porre per determinare la previsione potrebbero non essere così esaustive come il set di domande di qualcun altro. Tenendo conto di tutta la potenziale variabilità dei dati, possiamo ridurre il rischio di sovradattamento, di bias e di varianza complessiva, ottenendo previsioni più precise.

Come funziona

Gli algoritmi delle foreste casuali hanno tre iperparametri principali che devono essere impostati prima dell'allenamento. Questi includono la dimensione del nodo, il numero di alberi e il numero di caratteristiche campionate. Da lì, il classificatore della foresta casuale può essere utilizzato per risolvere problemi di regressione o di classificazione.

L'algoritmo della foresta casuale è composto da una collezione di alberi decisionali, e ogni albero nell'insieme è costituito da un campione di dati tratto da un set di allenamento con sostituzione, chiamato campione di bootstrap. Di quel campione di allenamento, un terzo viene messo da parte come dati di test, noti come campione fuori dalla borsa (oob), a cui torneremo in seguito. Un'altra istanza di casualità viene quindi iniettata attraverso il bagging delle caratteristiche, aggiungendo maggiore diversità al dataset e riducendo la correlazione tra gli alberi decisionali. A seconda del tipo di problema, la determinazione della previsione varierà. Per un compito di regressione, gli alberi decisionali individuali verranno mediati, mentre per un compito di classificazione, una maggioranza di voti - ossia la variabile categorica più frequente - darà come risultato la classe prevista. Infine, il campione oob viene utilizzato per la convalida incrociata, finalizzando quella previsione.

Benefici e sfide del random forest

Ci sono diversi vantaggi e sfide chiave che l'algoritmo random forest presenta quando utilizzato per problemi di classificazione o regressione. Alcuni di essi includono:

Principali vantaggi

Riduzione del rischio di overfitting: Le alberi di decisione corrono il rischio di overfitting poiché tendono ad adattarsi strettamente a tutti i campioni all'interno dei dati di formazione. Tuttavia, quando ci sono un robusto numero di alberi di decisione in un random forest, il classificatore non sovrastimerà il modello poiché la media di alberi scorrelati riduce la varianza complessiva e l'errore di previsione.

Fornisce flessibilità: poiché il random forest può gestire sia compiti di regressione che di classificazione con un elevato grado di precisione, è un metodo popolare tra i data scientist. Inoltre, la feature bagging rende il classificatore random forest uno strumento efficace per stimare i valori mancanti poiché mantiene l'accuratezza quando una parte dei dati è mancante.

Facile determinazione dell'importanza delle feature: il random forest rende facile valutare l'importanza delle variabili, o il contributo, al modello. Ci sono alcuni modi per valutare l'importanza della feature. L'importanza di Gini e la diminuzione media dell'impurità (MDI) vengono solitamente utilizzati per misurare quanto diminuisce l'accuratezza del modello quando una determinata variabile viene esclusa. Tuttavia, l'importanza permutazione, nota anche come diminuzione media dell'accuratezza (MDA), è un'altra misura di importanza. MDA identifica la diminuzione media dell'accuratezza permutando casualmente i valori delle feature nei campioni out-of-bag.

Principali sfide

Processo che richiede tempo: poiché gli algoritmi random forest possono gestire grandi set di dati, possono fornire previsioni più accurate, ma possono essere lenti nell'elaborazione dei dati in quanto computano i dati per ogni singolo albero decisionale.

Richiede più risorse: poiché i random forest elaborano set di dati più grandi, richiedono più risorse per archiviare quei dati.

Più complesso: la previsione di un singolo albero decisionale è più facile da interpretare rispetto a una foresta di essi.

Dataset risulato:

Come anticipato nel primo capitolo il dataset utilizzato per il mio caso di studio è il risultato dell’unione dei 2 dataset Student engagement dataset e DAiSEE.

Le immagini al loro interno sono state inizialmente elaborate attraverso la libreria py-feat per ottenere le misure delle Action Units.

Prima di poter effettuare delle predizioni è necessaria la creazione di un oggetto Detector fornito dalla libreria.

\begin{minted}[bgcolor=bg]{python}

device = "cuda" if torch.cuda.is\_available() else "cpu"

return Detector(

face\_model="retinaface",

landmark\_model="mobilefacenet",

au\_model="xgb",

emotion\_model="resmasknet",

facepose\_model="img2pose",

device=device,

)

\end{ minted }

Come è possibile notare è possibile nel codice, durante la creazione dell’oggetto Detector è possibile specificare il parametro

Py-feat permette di estrarre i valori delle Action units attraverso il metodo