# PRESENTAZIONE DEL PROGETTO DI MACHINE LEARNING + DEEP LEARNING

A partire da un dataset di immagini con caratteri scritti a mano, l’obiettivo del progetto è quello di classificare i pazienti in disgrafici o non disgrafici.

Abbiamo organizzato il progetto in due file ipynb:

* Il file ***Features\_extraction***, in cui ci occupiamo di fare l’estrazione delle features.
* Il file ***Classification***, in cui ci occupiamo della classificazione.

## FEATURES EXTRACTION

### IMPORT

Contiene tutti gli import necessari

### COSTRUZIONE DATASET

In questa sezione carichiamo il file **users** che contiene i dati dei pazienti. Il file è organizzato nel seguente modo: ogni riga rappresenta un paziente, caratterizzato da un ID, una label “diag” che indica se è disgrafico o meno, il sesso, la mano con cui scrive e l’età.

Dopodiché abbiamo creato una funzione encoder che trasforma i valori delle colonne nominali “diag”, “sex” e “hand” in valori numerici binari

#### SALVATAGGIO DI TUTTE LE IMMAGINI CONVERTITE

Utilizziamo la funzione **get\_Session** per accedere, a partire dall’id di uno user, ai dati ricavati a partire dalla sessione che lo user ha svolto per rilevare la disgrafia (ad esempio spazio, pressione, coordinate, ecc)

Abbiamo definito una funzione **view\_image** che ci permette di costruire l’immagine a partire dalle coordinate contenute nella sessione estratta grazie alla funzione precedente

Dopodiché, tramite la funzione **save\_image**, salviamo queste immagini nel drive

Infine, tramite un ciclo for prendiamo ognuna di queste immagini e le salviamo

### ESTRAZIONE FEATURES CON CNN

Abbiamo deciso di effettuare l’estrazione delle features utilizzando due architetture deep già implementate: ResNet50 e VGG16.

Per entrambe le reti, abbiamo definito una funzione **model\_features\_extraction** che itera su tutte le immagini, e per ognuna estrae le features. Ciò che otteniamo in output è un dataframe in cui ogni riga rappresenta uno user e ogni colonna rappresenta la feature estratta.

Per entrambi i modelli abbiamo ignorato l’ultimo layer relativo alla classificazione.

Infine, sempre per entrambi i modelli abbiamo salvato i rispetti file contenenti le features estratte.

### APPLICAZIONE DELLA DATA AUGMENTATION AL DATASET ORIGINALE

In questa sezione, dato che il nostro dataset era limitato perché costituito solamente da 120 immagini, abbiamo deciso di applicare la data augmentation, con l’obiettivo di capire se con questa tecnica potevamo ottenere delle prestazioni migliori.

Abbiamo applicato:

* Flipping orizzontale e verticale
* Rotation
* Cropping
* Abbiamo aggiunto rumore gaussiano

### ESTRAZIONE FEATURES CON DATA AUGMENTATION

In questa sezione abbiamo eseguito gli stessi passaggi effettuati nella sezione “Estrazione features” senza data augmentation.

Ai file contenenti l’estrazione delle features dovevamo aggiungere una colonna contenente le labels. Quindi ci siamo costruiti un dataframe **df\_diag** con una colonna **diag** contenente le rispettive labels.

Ogni file relativo alla features extraction è costituito da 720 righe. Abbiamo notato che per come abbiamo costruito questo file, gruppi di 6 righe corrispondono allo stesso user.

Quindi, abbiamo costruito **df\_diag** in modo che le prime 6 righe sono relative allo user00006, le seconde 6 righe sono relative allo user00007 e così via

## CLASSIFICATION

### IMPORT

Importiamo le librerie necessarie.

### CLASSIFICAZIONE CON MODELLI DI ML (SENZA E CON DATA AUGMENTATION)

Abbiamo usato i seguenti classificatori:

* Logistic Regression con l1 e l2 penalty
* Decision Tree
* Random Forest
* SVM

Per ogni classificatore abbiamo calcolato l’accuracy utilizzando per **random state** 100 valori randomici compresi tra 1 e 100.

Il nostro obiettivo era quello di calcolare un’accuracy media.

Inoltre, nel caso specifico di:

* Logistic Regression: abbiamo fatto variare anche l’iperparametro C
* Random Forest: abbiamo fatto variare anche l’iperparametro n\_estimators
* SVM: abbiamo fatto variare anche il tipo di kernel

### CONCATENAZIONE DELLE FEATURES OTTENUTE SENZA DATA AUGMENTATION E CON DATA AUGMENTATION

Abbiamo provato ad utilizzare i classificatori dandogli in input le features ottenute sia senza data augmentation che con data augmentation. Abbiamo applicato questo procedimento solo a ResNet50, che è stata la rete che ha ottenuto le prestazioni migliori.

### CLASSIFICAZIONE CON RETE NEURALE PRE-ADDESTRATA SUL DATASET I-AM

Per effettuare la classificazione abbiamo sfruttato una rete neurale pre-addestrata su un dataset composto da forme di testo scritte a mano in inglese.

Abbiamo eliminato il layer di output della rete caricata, e abbiamo aggiunto;

* Un layer denso
* Un layer di output con funzione di attivazione sigmoide

Inizialmente, abbiamo congelato i pesi del modello caricato e abbiamo addestrato sul nostro dataset solo i layer che abbiamo aggiunto noi.

Abbiamo addestrato la rete su 70 epoche, e per ogni epoca ci siamo stampati l’accuracy e la loss.

Successivamente, abbiamo scongelato anche i pesi del modello caricato e lo abbiamo addestrato nuovamente su 40 epoche. In questo modo i pesi sono stati aggiornati partendo da un’inizializzazione non randomica, ma dall’inizializzazione relativa all’addestramento sul dataset IAM.