



ALMA MATER STUDIORUM  
UNIVERSITÀ DI BOLOGNA

# Progetto del Corso

## Data Analytics


*Master in Computer Science (Second cycle degree)*  
*University of Bologna*

---

**Prof. Marco Di Felice Prof. Giuseppe Lisanti**

Department of Computer Science and Engineering, University of Bologna

marco.difelice3@unibo.it, giuseppe.lisanti@unibo.it

A solid red horizontal bar spanning the width of the slide.

# Progetto del Corso

---

- L'esame del corso di Data Analytics, a.a. 2023/2024, consiste di due prove: **prova orale** e **progetto**, entrambe **OBBLIGATORIE**.
- Le prove maturano voti distinti, il voto finale si calcola come media delle due prove:

$$\text{Voto}_{\text{FINALE}} = 0.5 * \text{Voto}_{\text{PROGETTO}} + 0.5 \text{Voto}_{\text{ORALE}}$$

- Le due prove avvengono contestualmente (nella stessa data). A valle della discussione del progetto, è previsto l'esame orale con domande di teoria sugli argomenti trattati durante il corso.



# Progetto del Corso

---

- Il progetto può essere svolto **singolarmente** o in gruppi di massimo **DUE** unità.
- Sono previsti i seguenti appelli:  
**22 Gennaio 2024, 16 Febbraio 2024, inizio Giugno 2024, inizio Luglio 2024, fine Luglio 2024, prima metà Settembre 2024**
- La consegna su Virtuale deve avvenire almeno 3 giorni prima della data di appello.
- E' obbligatorio iscriversi su Almaesami per partecipare all'appello.



# Progetto del Corso

---

- La consegna avviene tramite **VIRTUALE**.
- Occorre consegnare i sorgenti ed una relazione (in italiano o in inglese) con le seguenti sezioni di massima:
  1. Introduzione
  2. Metodologia
  3. Implementazione
  4. Risultati



# Progetto del Corso

---

- A valle della consegna, occorre preparare una **presentazione con slide** per la discussione del progetto: durante la presentazione, può essere chiesta una demo.
- Tutti i membri del gruppo devono essere presenti durante la discussione (**IN PRESENZA**), e devono conoscere il 100% del progetto svolto. La ripartizione del lavoro deve essere equa.
- La prova orale è indipendente dal progetto e verte su **TUTTI** gli argomenti del corso (anche quelli non trattati dal progetto).



# Progetto del Corso

---

- Il progetto consiste nella realizzazione di uno **studio di data analytics**, implementando **TUTTE le fasi della pipeline** viste a lezione:
  - Data Acquisition
  - Data Visualization
  - Data Preprocessing (quelle necessarie per il task)
  - Modeling (con tuning degli iperparametri)
  - Performance Evaluation



# Progetto del Corso

- Il progetto prevede lo sviluppo di **funzionalità indipendenti, e con complessità crescente.**
- Ogni funzionalità è associata ad un **punteggio massimo** che si può conseguire se tali funzionalità sono state sviluppate nel modo corretto (a discrezione dei docenti).
- Il punteggio massimo che si può conseguire è **30 e lode.**
- Il progetto può essere ritenuto insufficiente (in tale caso, si devono ripetere la consegna e discussione seguente).
- In tutti gli altri casi, **NON sono previste riconsegne della stessa traccia. Si può ripetere l'esame, ma su traccia differente.**

# Progetto del Corso

---

- **FUNZIONALITA' 1.** Il progetto prevede l'utilizzo di tecniche di ML supervised tradizionali non deep (Random Forest, LR, SVM, KNN) → **PUNTEGGIO MASSIMO OTTENIBILE: 25**
- **FUNZIONALITA' 2.** Il progetto prevede l'utilizzo di tecniche di ML supervised basate su reti neurali ed architettura Feed-Forward → **PUNTEGGIO MASSIMO OTTENIBILE: 28**
- **FUNZIONALITA' 3.** Il progetto prevede l'utilizzo di tecniche di ML supervised con modelli deep per *Tabular* Data (**TabNet & TabTransformer**) → **PUNTEGGIO MASSIMO OTTENIBILE: 30**





# Progetto del Corso

---

- Nel caso della FUNZIONALITA' 3, far riferimento ai **modelli deep per Tabular Data** disponibili qui:

[https://github.com/manujosephv/pytorch\\_tabular](https://github.com/manujosephv/pytorch_tabular)

# Progetto del Corso

---

- Occorre strutturare il Progetto in due **componenti software distinte ed autonome dal punto di vista dell'esecuzione**:
  - **Training Module**: questa componente effettua il pre-processamento, l'addestramento degli algoritmi implementati (tra quelli previsti nelle FUNZIONALITA' 1, 2 e 3) ed il tuning degli iper-parametri.
  - **Inference Module**: questa componente istanzia le tecniche implementate (tra quelle previste nelle FUNZIONALITA' 1, 2 e 3), usando la configurazione migliore degli iper-parametri identificata durante la fase di Training. Inoltre, questa componente DEVE implementare un API descritta nelle slide a seguire.

# Progetto del Corso

---

- In VIRTUALE, troverete due Sezioni distinte di consegna:
  - **Consegna Training Module**
    - File .zip contenente tutto il codice del modulo: organizzazione e strutturazione su diversi file sono a discrezione dello studente.
  - **Consegna Test Module**
    - File .zip contenente:
      1. file di codice `test.py`, che DEVE implementare le 4 funzioni Python descritte nella slide seguente (esattamente con gli stessi nomi)
      2. eventuali altri file di codice o di dati utilizzati da `test.py` per caricare gli iperparametri, i modelli pre-addestrati ed istanziare le tecniche di processamento/modeling



# Progetto del Corso

---

- Metodi da implementare in `test.py`

**getName()** → restituisce un identificativo dello studente o del gruppo (es. combinazione di matricole)

## INPUT:

Nessuno

## OUTPUT:

Stringa identificativa della consegna, a discrezione degli studenti (no nomi scurrili)

# Progetto del Corso

- Metodi da implementare in `test.py`

**`preprocess(df, clfName)`** → effettua il pre-processing dei dati di test forniti in input. Il metodo deve caricare i parametri determinati durante la fase di TRAINING, per le tecniche di preprocessing scelte.

## INPUT:

Dataframe Pandas contenente i dati di TEST. I dati hanno la stessa struttura (numero e nomi attributi) di quelli usati per il TRAINING.

`clfName`: stringa che identifica la tecnica di ML da utilizzare, e che può assumere i valori indicati nella slide successiva

## OUTPUT:

Dataframe Pandas ottenuto come risultato del pre-processamento.

# Progetto del Corso

---

- Metodi da implementare in `test.py`

Il campo **clfName** è una stringa con i seguenti valori ammissibili:

- 'LR' → identifica regressore *Linear Regression* (F1)
- 'RF' → identifica regressore *Random Forest Regressor* (F1)
- 'KNR' → identifica regressore basato su *KNN* (F1)
- 'SVR' → identifica regressore *Support Vector Regressor* (F1)
- 'FF' → identificare regressore con reti neurali, architettura *Feed Forward* (F2)
- 'TB' → identificare regressore con reti neurali, architettura *TabNet* (F3)
- 'TF' → identificare regressore con reti neurali, architettura *Tab Transformer* (F3)



# Progetto del Corso

---

- Metodi da implementare in `test.py`  
**`load(clfName)`** → istanzia la tecnica di ML con nome `clfName`

## INPUT:

`clfName`: stringa che identifica la tecnica di ML da utilizzare, e che può assumere i valori indicati nella slide successiva

## OUTPUT:

Oggetto Python relativo all'istanza dell'algoritmo di ML, con iper-parametri determinati durante la fase di TRAINING. Se l'algoritmo non è stato implementato, deve restituire un valore `None`.

# Progetto del Corso

- Metodi da implementare in `test.py`

**`predict(df, clfName, clf)`** → esegue il modello ML (oggetto `clf`) sui dati di TEST preprocessati (`df`)

## INPUT:

`df`: dataframe di TEST, ottenuto come output del metodo di preprocess descritta precedentemente

`clfName`: stringa che identifica la tecnica di ML da utilizzare

`clf`: istanza dell'algoritmo di ML, ottenuto come output del metodo di load descritto precedentemente

## OUTPUT:

Dizionario Python contenente i valori di prestazioni dell'algoritmo di ML quando eseguito sui dati di TEST preprocessati (vedere slide successiva)





# Progetto del Corso

---

- Metodi da implementare in `test.py`

## Chiavi del dizionario (output della predict)

- 'mse': valore del Mean Squared Error
- 'mae': valore del Mean Absolute Error
- 'mape': valore del Mean Absolute Percentage Error
- 'r2score': valore di R2 score



# Progetto del Corso

---

- Cosa viene fornito:
  - **Un singolo dataset.** A discrezione degli studenti come/se suddividerlo in dataset di train/validation.
- Cosa NON viene fornito (ma a disposizione dei docenti):
  - **Dataset di test «privato»**
  - **Sistema automatico di valutazione delle consegne.** Il sistema automatico esegue il modulo di inferenza sul dataset di test privato e ricava le metriche di prestazioni, per tutte le tecniche implementate tra le Funzionalità 1, 2 e 3.

# Progetto del Corso

- Es. di flusso di esecuzione del Modulo di INFERENZA da parte del nostro sistema automatico di valutazione delle consegne:

```
for clfName in CLF_NAME_LIST: ← Lista degli acronimi,  
                                vedi slide 15  
    name = getName()  
    dfProcessed = preprocess(df_test, clfName)  
    clf = load(clfName)  
    perf = predict(dfProcessed, clfName, clf)
```

- Viene fornito in VIRTUALE un esempio di consegna (per toy dataset), con i due moduli di TEST e TRAINING.



# Progetto del Corso

---

- E' assolutamente **VIETATO**, pena **NON VALUTAZIONE** della consegna:
  1. Rinominare i nomi dei metodi precedentemente indicati
  2. Ri-addestrare i modelli dentro i metodi `load` e `predict`
  3. Effettuare il tuning degli iper-parametri sui dati di test dentro i metodi `load` e `predict`



# Progetto del Corso

---

**DATASET** Task: **REGRESSIONE**

**OBIETTIVO:** Riconoscere l'anno in cui è stata pubblicata una canzone a partire dalle feature della sua traccia audio.

Variabili Input: < N feature audio di una canzone >

Variabile Output: < Anno in cui è stata pubblicata la canzone >