Immagine che contiene testo, moneta

Descrizione generata automaticamente

|  |
| --- |
| Immagine che contiene testo  Descrizione generata automaticamente  Scuola Politecnica e delle Scienze di Base  Corso di Laurea Magistrale in Ingegneria Informatica |

|  |
| --- |
| **Big Data Engineering**  Anno Accademico 2021/2022 |

Professore:

**Sperlì Giancarlo**

Studenti:

**Lamboglia Anna M63001219**

**Perrot Raffaella M63001135**

**Ricchiuti Fausto M63001144**

Sommario

[Big Data Engineering I](#_Toc105151525)

[Capitolo 1 3](#_Toc105151526)

[1.1 Descrizione della challenge 3](#_Toc105151527)

[1.2 Dataset utilizzato 3](#_Toc105151528)

[1.3 Strumenti Utilizzati 5](#_Toc105151529)

[1.3.1 MongoDB 5](#_Toc105151530)

[1.3.2 Python 5](#_Toc105151531)

[1.3.3 Spark 6](#_Toc105151532)

[1.3.4 Excel 7](#_Toc105151533)

[Capitolo 2 8](#_Toc105151534)

[2.1 Trend delle carriere universitarie 8](#_Toc105151535)

[Capitolo 3 9](#_Toc105151536)

[3.1 Analisi dei Fuorisede 9](#_Toc105151537)

[3.1.1 Immatricolati Fuorisede totali e divisi per ateneo 9](#_Toc105151538)

[3.1.2 Fuorisede per area di provenienza 12](#_Toc105151539)

[3.2 Creazione dei dataset 12](#_Toc105151540)

[Capitolo 4 14](#_Toc105151541)

[4.1 Predizione dei trend di iscrizione 14](#_Toc105151542)

[4.1.1 Creazione del dataset 14](#_Toc105151543)

[4.1.2 Modello regressivo 18](#_Toc105151544)

[4.1.3 Dataset di test 18](#_Toc105151545)

[Conclusioni 20](#_Toc105151546)

# Capitolo 1

### Descrizione della challenge

L’obiettivo della challenge è quello di effettuare uno studio sui trend nelle università in Italia.

In particolare, ci si è focalizzati su tre punti:

* Analizzare e descrivere i trend delle carriere universitarie. Sono state estratte informazioni significative e interessanti utilizzando dati demografici, ambiti di studio, background personale e opportunità di lavoro;
* Analizzare e descrivere i trend del decentramento attraverso le scelte delle sedi universitarie. Sono state analizzate le scelte degli studenti fuorisede;
* Prevedere il trend di iscrizione in base ai dati storici e fattori come l’istruzione, la demografia o altri fattori che si possono scoprire.

### Dataset utilizzato

Al fine di risolvere i tre punti sopraelencati, è stata svolta una ricerca per poter trovare dei dati che siano quanto più consoni possibile alle analisi da effettuare.

In particolare, sono stati scaricati quattro file *.csv* messi a disposizione dal MIUR mediante questo link : <https://bit.ly/3zhmMf1> e dall’ISTAT mediante questo link: <https://bit.ly/3NftiqI> .

I quattro file in questione sono i seguenti ed è possibile trovarli all’interno della seguente repository github (INSERIRE IL LINK) :

* *Immatricolati\_Anno\_Gruppo.csv*. Il file è composto da cinque colonne*:*

1. *AnnoA,* l’anno accademico;
2. *GruppoNOME*, nella qualesono definite le tipologie di università;
3. *Sesso;*
4. *Imm*, nella quale sono indicati il numero di immatricolati;
5. *Note* per definire delle note particolari.

Tale file permette di comprendere il numero di immatricolati, dato l’anno, il tipo di università e il sesso.

Il file ha una dimensione pari a 96 Kb e presenta 360 righe;

* *Iscritti\_Anno\_Gruppo.csv.* Il file è composto da cinque colonne*:*

1. *\_id;*
2. *AnnoA,* l’anno accademico;
3. *GruppoNOME*, nella quale sono definite le tipologie di università;
4. *Sesso;*
5. *Isc*, nella quale sono indicati il numero di iscritti.

Tale file permette di comprendere il numero di iscritti di un determinato gruppo universitario, dato l’anno e il sesso.

Il file ha una dimensione pari a 89 Kb e presenta 330 righe;

* *Immatricolati\_Residenza\_Ateneo.sv*. Il file è composto da otto colonne*:*

1. *\_id;*
2. *AnnoA,* l’anno accademico;
3. *AteneoCOD,* il codice associato ad un determinato ateneo;
4. *SedeP*, ilcodice relativo ad una determinata provincia;
5. *ResidenzaR,* la regione di residenza;
6. *ResidenzaP,* la provincia di residenza;
7. *Imm*, nella quale sono indicatiil numero di immatricolati.

Tale file permette di comprendere la residenza degli immatricolati associati ad una determinata università.

Il file ha una dimensione pari a 1.805 Kb e presenta 32000 righe;

* *Popolazione Province.csv*. Il file è composto da tredici colonne:

1. *ITTER107*, id del territorio;
2. *Territorio*, la provincia o la regione;
3. *TIPO\_DATO15*, il codice del periodo di riferimento;
4. *Tipo dato*, popolazione del primo gennaio;
5. *ETA1*, classe di età (totale);
6. *Classe di età*, totale;
7. *SEXISTAT1*, il codice per identificare il sesso;
8. *Sesso*;
9. *CITTADINANZA*, la sigla della nazionalità;
10. *Cittadinanza*, la sigla della nazionalità;
11. *TIME*, l’anno di riferimento;
12. *Seleziona periodo*, l’anno di riferimento;
13. *Value*, la popolazione.

Il file ha una dimensione pari a 502 Kb e presenta 4051 righe.

### Strumenti Utilizzati

Per effettuare le analisi sono stati utilizzati i seguenti strumenti:

* *Mongo DB;*
* *Python;*
* *Pyspark;*
* *Excel.*

#### 1.3.1 MongoDB

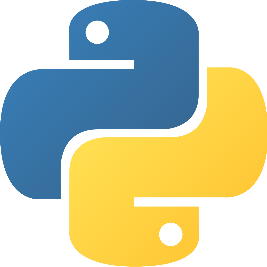
Per effetuare l’analisi del dataset è stato deciso di utilizzare *MongoDB*. Esso è un DBMS non relazionale sviluppato in C++, open-source, document-oriented e scalabile. Esso è stato realizzato in maniera tale da avere alte prestazioni, sia in lettura che in scrittura. Tra i vantaggi, si osserva che le letture più consistenti possono essere distribuite in più server replicati e le interrogazioni sono pi`u semplici e veloci grazie all’approccio ai documenti che rende possibile la rappresentazione di relazioni gerarchiche complesse attraverso documenti nidificati e array. Le caratteristiche principali di MongoDB sono le seguenti:

* **Database document-oriented:** i dati vengono archiviati sotto forma di documenti in formato JSON;
* **Supporto completo agli indici**: indicizzazione di qualsiasi attributo;
* **Replicazione**: facilità nella replicazione dei dati attraverso la rete e alta scalabilità;
* **Sharding**: scalabilità orizzontale senza compromettere nessuna funzionalità.



#### 1.3.2 Python

Python è un linguaggio di programmazione di "alto livello", orientato a oggetti, adatto, tra gli altri usi, a sviluppare applicazioni distribuite, scripting, computazione numerica e system testing.



#### 1.3.3 Spark

Per poter lavorare sui dati ed applicare tecniche di Machine Learning è stata utilizzato *Pyspark*, ossia una libreria che consente l’utilizzo in Python di Apache Spark.

*Apache Spark* è un framework di elaborazione parallela open source, che supporta l’elaborazione in memoria al fine di migliorare le prestazioni delle applicazioni che analizzano Big Data.

Le caratteristiche principali di Spark sono le seguenti:

* **Fast processing:** Spark contiene Resilient Distributed Dataset (RDD), che consentono di risparmiare tempo necessario per le operazioni di lettura e scrittura, permettendo di ottenere un tempo di elaborazione più veloce rispetto ad Hadoop, di uno o due ordini di grandezza;
* **In-Memory computing:** in Spark i dati sono salvati nella RAM (Random Access Memory), in questo modo è possibile accedere a quest’ultimi più velocemente ed accelerare la velocità della fase di analytics;
* **Flexible**: Spark supporta diversi linguaggi e consente agli sviluppatori di realizzare applicazioni in Java, Scala, R o Python;
* **Fault tolerance.** Spark contiene Resilient Distributed Dataset (RDD), progettati per gestire il guasto di un qualsiasi nodo worker del cluster. In questo modo, si garantisce che la perdita di dati sia pressocché pari a zero.
* **Better Analytics**: Spark ha a disposizione una ricca serie di funzionalità, come la possibilità di effettuare query SQL oppure applicare algoritmi di machine learning, che consentono di migliorare le performance in fase di analisi.

#### 1.3.4 Excel

*Microsoft Excel* è un programma prodotto da Microsoft, dedicato alla produzione ed alla gestione di fogli elettronici. È parte della suite di software di produttività personale Microsoft Office, ed è disponibile per i sistemi operativi Windows e macOS. È il programma per la produzione e gestione di fogli elettronici più utilizzato.



# Capitolo 2

### Trend delle carriere universitarie

Per risolvere il primo punto definito dalla challenge è stato deciso di utilizzare il database NOSQL MongoDB. In particolare, per poter analizzare i dati delle carriere sono stati utilizzati i 2 file Immatricolati\_Anno\_Gruppo.csv, Iscritti\_Anno\_Gruppo.csv.

Innanzitutto è stato creato un DB contente una collection per ogni file .csv in modo da poter effettuare delle projection dei dati di interesse.

IscrizioniCollection = Database['Iscrizioni']

ImmatricolazioniCollection = Database['Immatricolazioni']

Innanzitutto è stato creato un DB contente una collection per ogni file .csv in modo da poter effettuare delle projection dei dati di interesse

# Capitolo 3

### Analisi dei Fuorisede

Nel seguente capitolo è mostrato l’approccio utilizzato per risolvere il task legato all’analisi ed alla descrizione dei trend del decentramento, considerando le scelte effettuate dagli studenti *fuorisede*.

Per poter analizzare i dati è stato utilizzato il file *Immatricolati\_Residenza\_Ateneo.csv*, composto da otto colonne:

1. *\_id;*
2. *AnnoA*, l’anno accedemico;
3. *AteneoCOD,* il codice identificativo dell’ateneo*;*
4. *AteneoNOME;*
5. *SedeP*, id della provincia dell’ateneo;
6. *ResidenzaR*, regione di residenza;
7. *ResidenzaP*, provincia di residenza;
8. *Imm*, numero popolazione*.*

E’ stato creato un DB contenente una collection per il file .csv:

FuorisedeCollection = Database['Fuorisede']

#### 3.1.1 Immatricolati Fuorisede totali e divisi per ateneo

Immagine che contiene testo

Descrizione generata automaticamenteIn questo step, l’obiettivo è stato di valutare il numero totale di immatricolati fuorisede, per ogni anno e per ogni ateneo. Di seguito, si riporta il codice utilizzato:

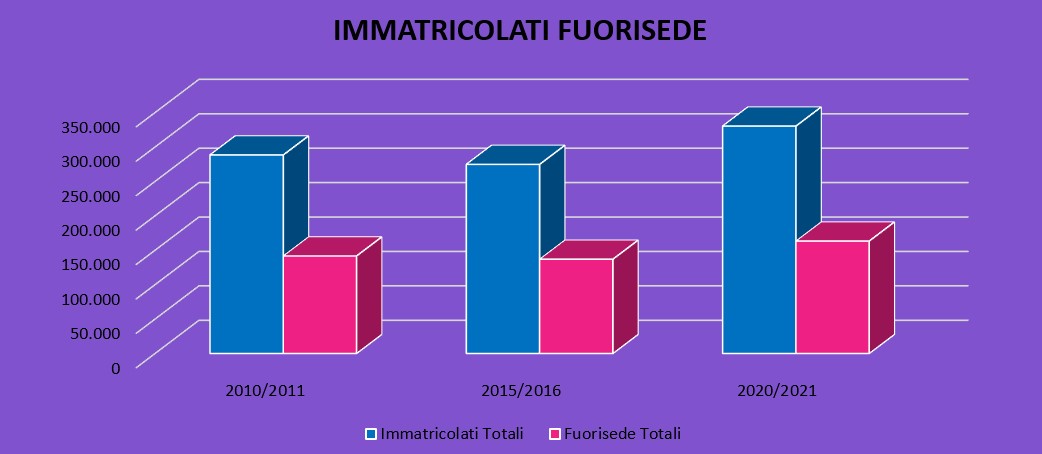
Successivamente, sono state effettuate delle operazioni in python per:

* Eliminare le righe in cui il nome della *Regione* o della *Provincia* è incluso nel nome dell’*Ateneo*;
* Eliminare le università di tipo telematico.

Immagine che contiene testo

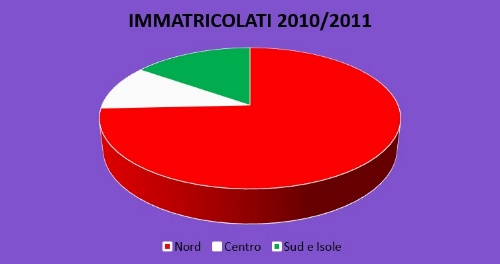
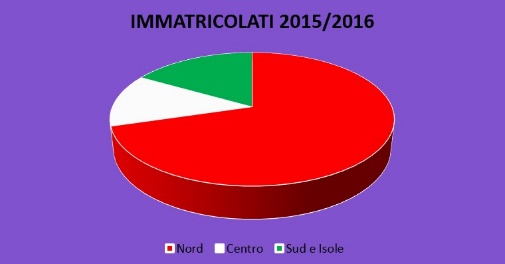
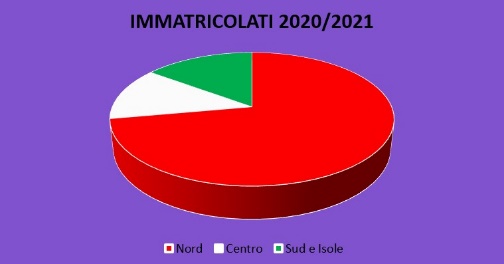
Descrizione generata automaticamente

In particolare, sono stati presi come riferimento tre anni accademici: 2010/2011, 2015/2016, 2020/2021 e di seguito si riportano degli istogrammi che mostrano il numero di *fuorisede* negli anni:



Sono state selezionate quatto università:

* Bologna, poichè è l’università con il più alto numero di studenti Fuorisede;
* Milano, ‘rappresentante’ delle università del Nord ;
* Roma La Sapienza, ‘rappresentante’ delle università del Centro ;
* Napoli Federico II, ‘rappresentante’ delle università del Sud.

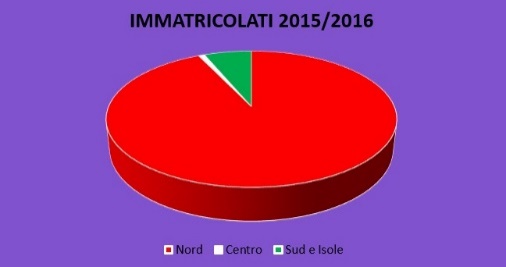
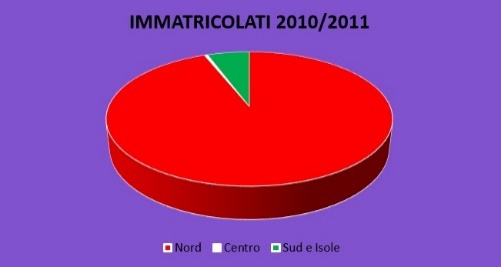
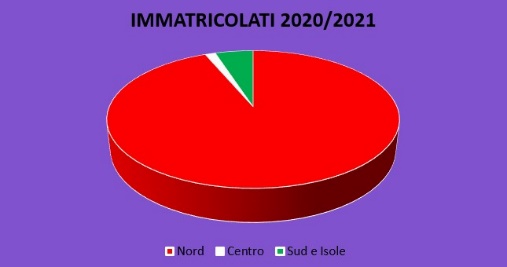
Bologna:

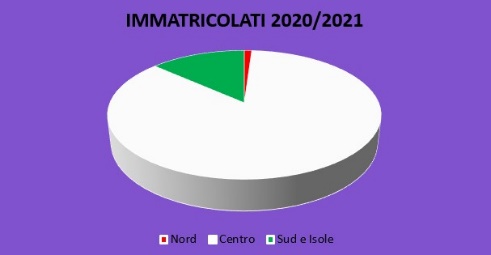
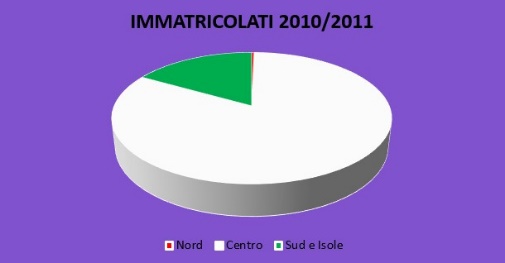
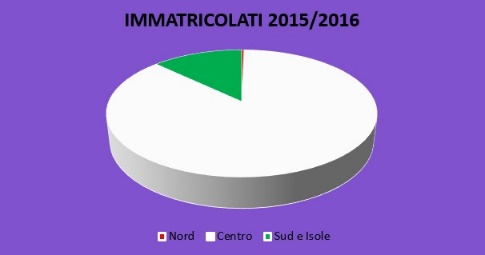
Milano:

Immagine che contiene testo, clipart

Descrizione generata automaticamenteImmagine che contiene testo, clipart

Descrizione generata automaticamenteImmagine che contiene testo, clipart

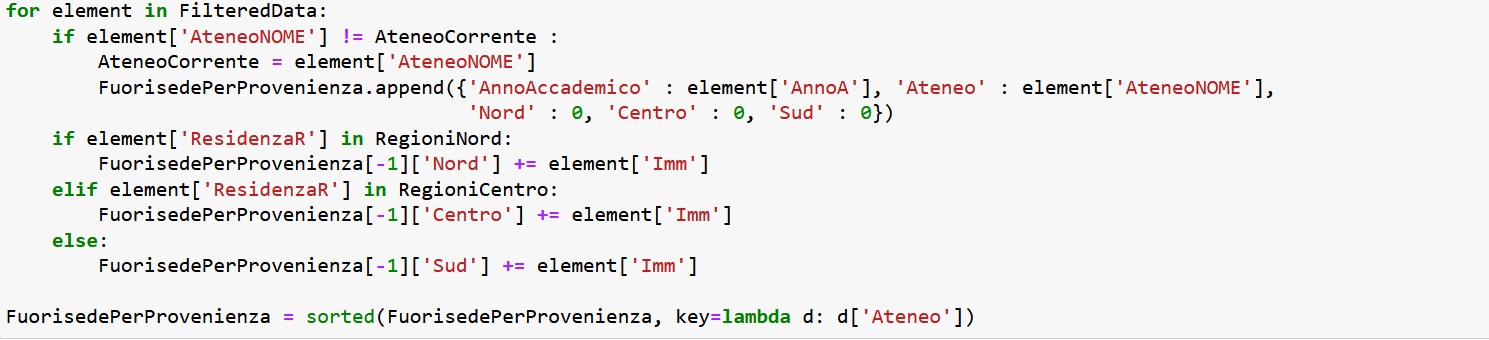
Descrizione generata automaticamenteRoma:

Napoli:

#### 3.1.2 Fuorisede per area di provenienza

Si è voluto, inoltre, analizzare i Fuorisede per area di provenienza.

Immagine che contiene testo

Descrizione generata automaticamenteSi riporta il codice python nel quale le regione vengono innanzitutto divise le regioni in regioni del Nord, del Centro e del Sud:

### 3.2 Creazione dei dataset

Come è stato fatto nel capitolo precedente per gli iscritti e gli immatricolati, anche per i fuorisede sono stati creati tre fogli excel:

* *Fuorisede per A.A*.. Il quale ha due colonne:

1. AnnoAccedemico ;
2. Fuorisede.

* *Fuorisede per Ateneo*. Il quale ha tre colonne:

1. AnnoAccedemico ;
2. Ateneo ;
3. Fuorisede.

* *Fuorisede Per Provenienza*. Il quale ha cinque colonne:

1. AnnoAccademico ;
2. Ateneo ;
3. Nord ;
4. Centro ;
5. Sud.

# Capitolo 4

### 4.1 Predizione dei trend di iscrizione

Nel seguente capitolo è mostrato l’approccio utilizzato per risolvere il task legato alla predizione dei trend di iscrizione degli studenti all’università nei prossimi anni. A tale scopo, si è ritenuto necessario l’utilizzo della liberia Python *Pyspark*, in modo da potersi avvalere del componente Mllib di Apache Spark, che permette di effettuare operazioni di Machine Learning.

Inoltre, è stato effettuato un confronto tra le performance fornite da una soluzione che utilizza *Pyspark* ed i risultati ottenuti mediante la libreria Python *Scikit-Learn*.

#### 4.1.1 Creazione del dataset

Al fine di risolvere il task di cui al precedente paragrafo, appare necessaria la fase di *creazione del dataset*, effettuata cercando di comprendere quali possano essere le *features* di interesse per l’analisi in oggetto.

Pertanto, a valle di un confronto tra gli autori, sono state valutate le *features* di maggiore interesse, di seguito riportate:

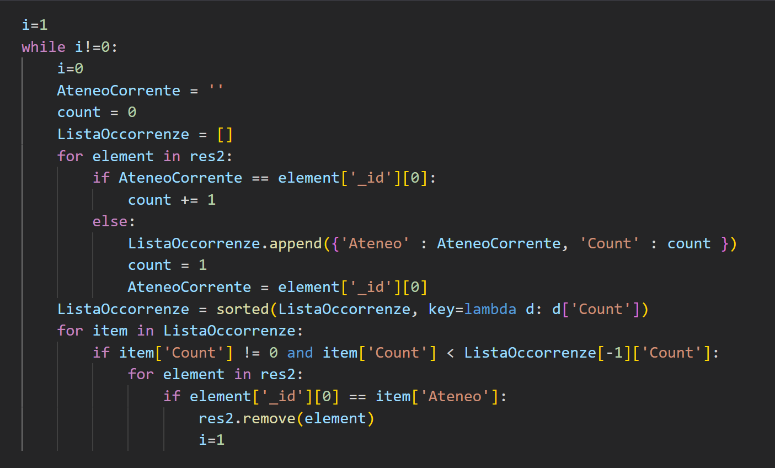
* *AnnoAccademico*: indica l’anno accademico oggetto di analisi;
* *Ateneo*: riporta l’ateneo a cui si riferiscono le elaborazioni;
* *PopolazioneP*: indica il numero di abitanti nella provincia di cui fa parte l’ateneo analizzato;
* *AteneiP*: indica il numero di atenei presenti in una determinata provincia;
* *PopolazioneR*: indica il numero di abitanti nella regione di cui fa parte l’ateneo analizzato;
* *AteneiR*: indica il numero di atenei presenti in una determinata provincia;
* *Fuorisede*: numero di studenti di un determinato ateneo, residenti in un’altra provincia rispetto all’ateneo stesso, ottenuti mediante le elaborazioni descritte nei precedenti capitoli;
* *Immatricolati:* numero di studenti iscritti al primo anno in un determinato ateneo per l’*AnnoAccademico* di riferimento.

Una volta definite le *features* oggetto del dataset, bisogna comprendere come è possibile popolarlo.

E’ possibile notare che i file .csv visti nei paragrafi precedenti, permettono di popolare, in alcuni casi anche a valle di opportune elaborazioni, le colonne *AnnoAccademico, Ateneo, AteneiP, AteneiR, Fuorisede* ed *Immatricolati.*

Infatti, per quanto concerne la creazione della colonna *Fuorisede*,è stata eseguita la seguente query in MongoDB e sono state effettuate opportune elaborazioni in *Python*:

Figura 1: Query ed elaborazioni in Python (1/2)



*Immagine che contiene testo, screenshot, schermo

Descrizione generata automaticamente*

*Immagine che contiene testo

Descrizione generata automaticamente*

Figura 2: Query ed elaborazioni in Python (2/2)

Figura 3:

Per quanto riguarda la creazione delle colonne *AteneiP, AteneiR*, è stato elaborato un algoritmo che, a partire dagli atenei, riuscisse ad effettuare un conteggio di atenei per regione e per provincia, così come descritto dal seguente codice:

Immagine che contiene testo

Descrizione generata automaticamente

Figura 4: Conteggio degli Atenei

Immagine che contiene testo

Descrizione generata automaticamente

Figura 5: Creazione del Dataset

Inoltre, per poter popolare anche le colonne *PopolazioneP* e *PopolazioneR*,è stato necessario introdurre il file *Popolazione Province.csv,* già descritto nel capitolo introduttivo ed anche in questo caso, come nei precedenti paragrafi, è stata creata una collezione all’interno del database *KPMG,* denominata *ISTAT*, contenente i dati del file .csv.

Successivamente, è stato necessario effettuare una query al database MongoDB e poi è sono state effettuate successive elaborazioni in Python:

Immagine che contiene testo

Descrizione generata automaticamente

Figura 6: Query sulla collection ISTAT

Immagine che contiene testo

Descrizione generata automaticamente

Figura 7: Elaborazione sulla popolazione

A valle delle precedenti operazioni, una volta che i dati sono stati ricavati, è stato creato un nuovo file excel denominato *RegressionDataset.xlsx*, utilizzato per la fase di training del modello regressivo:

Immagine che contiene tavolo

Descrizione generata automaticamente

Figura 8: File excel risultante

#### 4.1.2 Modello regressivo

Sulla base di quanto descritto nel precedente paragrafo, è stato ottenuto il *dataset* da utilizzare per poter applicare un modello di *Machine Learning* utilizzando la libreria Python *Pyspark*.

Per cui, dato che il problema da risolvere riguarda la previsione dell’andamento degli immatricolati per gli anni futuri, è stato deciso di applicare un *modello regressivo*. Tale approccio presuppone preliminarmente un *preprocessing* dei dati, in quanto sono presenti due colonne che presentano dati di tipo *string*, ossia l’*AnnoAccademico* e *Ateneo*.

Quindi, per ovviare a tale problematica, una volta caricato il dataset è stato applicato il processo di *One Hot Encoding*, trasformando preliminarmente il tipo *string* in un array all’interno delle colonne denominate *AnnoAccademico\_Array* e *Ateneo\_Array*, tramite le righe di codice Python mostrate in figura:

Immagine che contiene testo

Descrizione generata automaticamente

Figura 9: Creazione delle colonne AnnoAccademico\_Array e Ateneo\_Array

Successivamente, è stato applicato il *One Hot Encoding* sulle colonne appena create:

Immagine che contiene testo

Descrizione generata automaticamente

Figura 10: One Hot Encoding

Infine, per terminare la fase di preprocessing, sono state rimosse le colonne *AnnoAccademico*, *Ateneo, AnnoAccademico\_Array* e *Ateneo\_Array.*

A questo punto, è necessario definire le colonne *features* e *label*:

Immagine che contiene testo

Descrizione generata automaticamente

Figura 11: Creazione del Dataframe

Ottenendo dunque:

Immagine che contiene tavolo

Descrizione generata automaticamente

Figura 12: Aggiunta delle colonne Features e Label

Una volta ottenute le colonne *features* e *label*, è stato effettuato lo split del dataset in *train*, *validation* e *test*:

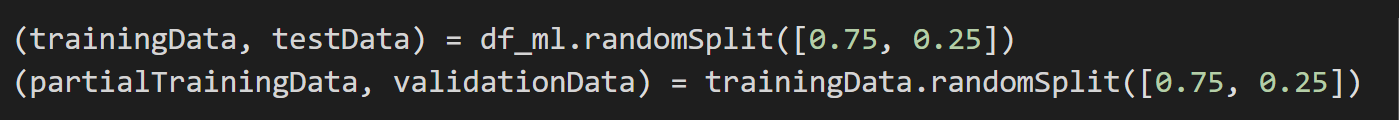


Figura 13: Split del dataset in Train, Validation e Test

In tal senso, è’ stato dunque suddiviso il dataset nel seguente modo :

* **75%**: *training*, di cui è stata effettuata la seguente ulteriore suddivisione per ottimizzare gli iperparametri:
* 75%: *training*;
* 25%: *validation*.
* **25%**: *test*.

Inoltre, sono stati applicati tre algoritmi di regressione differenti: *Linear Regressor*, *Random Forest Regressor* e *GBT Regressor*. In particolare, per il *Random Forest Regressor*, sono stati impostati un *numTrees* pari a 1000 ed una *maxDepth* pari a 3, mentre per il *GBT Regressor* sono stati impostati un *maxIter* pari a 100, ed una *stepSize* di 0.01, ottenendo i seguenti risultati:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Linear Regressor*** | | ***Random Forest Regressor*** | | ***GBT Regressor*** | |
| **MAE [Immatricolati]** | ***R 2*** | **MAE [Immatricolati]** | ***R 2*** | **MAE [Immatricolati]** | ***R 2*** | |
| **Validation** | *≃ 160* | ***-0.021*** | *≃ 139* | ***0,127*** | *≃ 104* | ***0,222*** | |
| **Test** | *≃ 161* | ***0.004*** | *≃ 143* | ***0.148*** | *≃ 109* | ***0.261*** | |
| **Re-Train** | *≃ 163* | ***-0.024*** | *≃ 144* | ***0.168*** | *≃ 101* | ***0.391*** | |

Figura 14: Tabella di confronto dei vari algoritmi

Come è possibile notare, i risultati migliori sono stati ottenuti applicando il *GBT Regressor*, dunque sarà quest’ultimo l’algoritmo utilizzato per la predizione finale.

#### 4.1.3 Dataset di test

Una volta creato il dataset e scelto il modello, la fase successiva è la definizione del *dataset di test*. Infatti, sebbene dati come *Anno Accademico* ed *Ateneo* possano essere semplici da definire, per quanto riguarda i dati della *popolazione* e dei *fuorisede*, risulta complesso definire come debbano essere popolate le corrispondenti colonne.

La soluzione che è stata utilizzata consiste nell’applicazione della seguente formula per calcolare l’incremento o decremento annuale della popolazione e dei fuorisede:

In figura, è riportato un esempio del calcolo che è stato effettuato:

Immagine che contiene testo, verde, segnale, puntamento

Descrizione generata automaticamente

Figura 15:

Di seguito, viene riportato il codice Python che è stato scritto per poter autoimatizzare questo calcolo:

Immagine che contiene testo

Descrizione generata automaticamente

Figura 16: Calcolo della media degli scarti annuali

A questo punto, è stato applicato l’algoritmo di predizione sulle istanze di test create, in modo da predire l’andamento degli immatricolati dal 2021/2022, fino al 2025/2026 per ogni singolo ateneo.

Nelle seguenti figure vengono riportati i risultati relativi alle immatricolazioni ottenute dalla predizione\*(*inserire una nota che dice che questi risultati sono quelli ottenuti con Sklearn come poi verrà spiegato nel paragrafo successivo*), in particolare è stato deciso di vedere in primis l’andamento totale delle immatricolazioni in Italia, e dall’istogramma in Figura (DA INSERIRE) è possibile notare come l’andamento risulti essere crescente, e poi valutare per singolo ateneo, e per semplicità sono stati riportati gli istogrammi relativi agli atenei di Bologna, Milano, Napoli Federico II è Roma La Sapienza, utilizzati come riferimento anche negli scorsi capitoli, mostrando un andamento crescente in tutti e quattro i casi. E’ stato voluto mostrare però come questo andamento non sia identico in tutti gli atenei, anzi come mostrato in Figura (DA INSERIRE), l’ateno di Bari presenta un trend di immatricolazioni decrescente per i prossimi 5 anni.

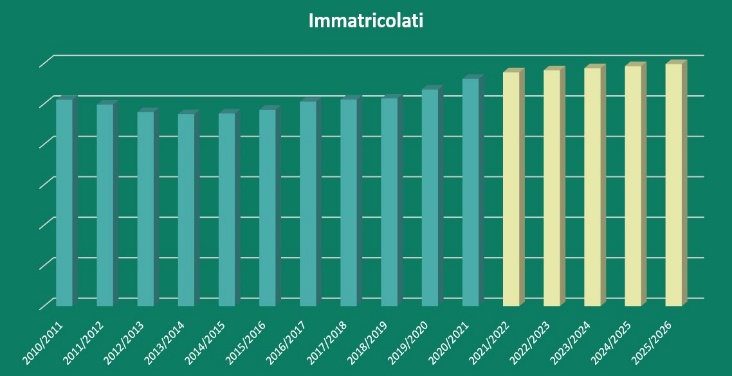


Figura 17: Trend delle immatricolazioni totali



Figura 18: Trend degli Atenei presi in considerazione

Immagine che contiene tavolo

Descrizione generata automaticamente

Figura 19: Trend di immatricolazioni per l’ateneo di Bari

E’ anche importante chiarire che sono stati scelti questi cinque atenei per facilità di documentazione, ma sono state generate le predizioni per tutti gli atenei.

#### 4.1.4 Confronto con scikit-learn

Nello scorso paragrafo, sono stati riportati i risultati delle performances dei regressori testati con Pyspark, ossia del *Linear Regressor, Random Forest Regressor* e *GBT Regressor* ottenuti con i vari split in train, validation e test.

Sebbene il MAE(Mean Absolute Error) ottenuto fosse abbastanza basso, ciò che ha messo in dubbio la bontà dei risultati ottenuti è stato l’indice R^2 abbastanza vicino allo 0.

Dunque al fine di ottenere dei risultati più soddisfacenti è stato deciso di applicare le stesse considerazioni fatte in precedenza in Pyspark, utilizzando la libreria *Scikit-learn*.

In particolare è stata replicata la stessa fase di preprocessing dei dati ed è stato suddiviso il dataset sempre in Train, Validation e Test.

E’ stato deciso di riportare un esempio di risultati differenti ottenuti utilizzando lo stesso algoritmo ed anche un confronto tra i migliori algoritmi che si sono trovati in entrambi i casi.

Prima di tutto riportiamo il caso in cui è stato utilizzato lo stesso regressore con gli stessi iperparametri, in particolare è stato provato il *Random Forest Regressor* con un *numTrees* pari a 1000 ed una *maxDepth* pari a 3, ottenendo i seguenti risulati e tempi di training :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***PySpark (RFRegressor)*** | | | ***Sklearn (RFRegressor)*** | | |
| **MAE [Immatricolati]** | ***R 2*** | **Time[s]** | **MAE [Immatricolati]** | ***R 2*** | **Time[s]** |
| **Validation** | *≃ 139* | ***0,127*** | *≃ 2.843* | *≃ 882* | ***0,894*** | *≃ 1.54* |
| **Test** | *≃ 143* | ***0.148*** | *≃ 2.843* | *≃ 1063* | ***0.842*** | *≃ 1.54* |
| **Re-Train** | *≃ 144* | ***0.168*** | *≃ 4.897* | *≃ 1050* | ***0.840*** | *≃ 2.69* |

Come è possibile osservare, i risultati risultano abbastanza differenti, infatti, sebbene utilizzando Pyspark si riescano dei valori di MAE più bassi, d’altro canto il valore R ^2 risulta essere molto basso ed i tempi di training più alti rispetto all’utilizzo di SKLearn.

Il secondo caso preso in esempio è un confronto tra le migliori soluzioni trovate utilizzando entrambe le librerie, in particolare è stato confrontato il GBTRegressor utilizzato in Pyspark, con un Extratrees Regressor con 400 stimatori, ottenendo i seguenti risultati :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***PySpark (GBTRegressor)*** | | | ***Sklearn (ExtraTreesRegressor)*** | | |
| **MAE [Immatricolati]** | ***R 2*** | **Time[s]** | **MAE [Immatricolati]** | ***R 2*** | **Time[s]** |
| **Validation** | *≃ 104* | ***0,222*** | *≃ 29.013* | *≃ 131* | ***0,996*** | *≃ 1.949* |
| **Test** | *≃ 109* | ***0.261*** | *≃ 33.16* | *≃ 181* | ***0.991*** | *≃ 1.* *949* |
| **Re-Train** | *≃ 101* | ***0.391*** | *≃ 29.799* | *≃ 156* | ***0.993*** | *≃ 2.01* |

Qui ancor di più si nota la differenza soprattutto a livello di tempo per il training tra i due algoritmi regressivi, infatti il tempo per addestrare un GBTRegressor in Spark risulta di circa 30 secondi mentre utilizzando un algoritmo che presenta performance abbastanza simili, se non migliori se si considera l’R^2, sono di circa 2 secondi.

Tuttavia, questo confronto non vuole far intendere che utilizzare SKLearn sia una soluzione milgliore rispetto a Pyspark, semplicemente in questo caso, avendo un numero di dati non eccessivo e la possibilità di testare algoritmi che nella libreria pyspark non sono ancora presenti, è stato deciso di scegliere i risultati dati dall’Extratrees Regressor per mostrare l’andamento del trend di immatricolazioni per i vari atenei.

# Conclusioni