Progetto Fondamenti Data Science e Machine Learning

Emanuele Bruno

Matricola: 0522501089

Analisi del dataset MIMIC-IV

Il progetto di corso sviluppato tratta l'analisi del dataset MIMIC-IV al fine di effettuare delle predizioni sulla morte ospedaliera e su situazioni quali la riammissione in ospedale entro un determinato lasso di tempo

Per svolgere il progetto ho utilizzato Jupyter notebook

Fase iniziale di preparazione

All'inizio è stato di fondamentale importanza consultare la documentazione del MIMIC disponibile sul sito stesso da cui è possibile ottenere il dataset (https://mimic.mit.edu/docs/iv/modules/)

Analizzando la documentazione ho deciso di concentrarmi principalmente sulle tabelle di ammissioni, pazienti e diagnosi in quanto mi permettono di aggiungere dati ad ogni ammissione e poter effettuare delle predizioni sulla singola ammissione che possono essere poi integrate su nuove ammissioni

Inizialmente ho lavorato su due notebook distinti, uno per le ammissioni e i pazienti e uno per le diagnosi

Ammissioni e pazienti (admpat.ipynb):

in questo notebook ho iniziato ad analizzare i dataset di ammissioni e pazienti, ho eliminato alcune feature ridondanti e ho unito i due dataset utilizzando come chiave le ammissioni.

Dopo aver unito I due dataset ho riorganizzato alcune feature più semplici come ad esempio il genere in modo da avere valori numerici, ho poi provveduto a calcolare l'età dei pazienti in quanto l'età è riferita all'ammissione in cui il paziente è stato ricoverato in ospedale.

Ho successivamente riordinato il dataset in modo da poter verificare per ogni ammissione se il paziente fosse stato successivamente riammesso in ospedale e ho dotato il dataframe di colonne apposite che segnalassero la riammissione entro 30/90/365 giorni.

Ho salvato il dataframe risultante come csv in modo da poter avere accesso ad esso successivamente.

	subject_id	hadm_id	admittime	dischtime	admission_type	admission_location	discharge_location	insurance	language	marital_status	ethnicity	h
0	10000019	25058216	2129-05-21 19:16:00	2129-05-23 18:30:00	ELECTIVE	NaN	HOME	Other	ENGLISH	NaN	WHITE	
1	10000032	22595853	2180-05-06 22:23:00	2180-05-07 17:15:00	URGENT	TRANSFER FROM HOSPITAL	HOME	Other	ENGLISH	WIDOWED	WHITE	
2	10000032	22841357	2180-06-26 18:27:00	2180-06-27 18:49:00	EW EMER.	EMERGENCY ROOM	HOME	Medicaid	ENGLISH	WIDOWED	WHITE	
3	10000032	29079034	2180-07-23 12:35:00	2180-07-25 17:55:00	EW EMER.	EMERGENCY ROOM	HOME	Medicaid	ENGLISH	WIDOWED	WHITE	
4	10000032	25742920	2180-08-05 23:44:00	2180-08-07 17:50:00	EW EMER.	EMERGENCY ROOM	HOSPICE	Medicaid	ENGLISH	WIDOWED	WHITE	
...	...	...	...	...	...	...	...	...	...	...	...	
523735	19999828	29734428	2147-07-18 16:23:00	2147-08-04 18:10:00	EW EMER.	PHYSICIAN REFERRAL	HOME HEALTH CARE	Other	ENGLISH	SINGLE	WHITE	
523736	19999828	25744818	2149-01-08 16:44:00	2149-01-18 17:00:00	EW EMER.	TRANSFER FROM HOSPITAL	HOME HEALTH CARE	Other	ENGLISH	SINGLE	WHITE	
523737	19999840	26071774	2164-07-25 00:27:00	2164-07-28 12:15:00	EW EMER.	EMERGENCY ROOM	HOME	Other	ENGLISH	WIDOWED	WHITE	
523738	19999840	21033226	2164-09-10 13:47:00	2164-09-17 13:42:00	EW EMER.	EMERGENCY ROOM	DIED	Other	ENGLISH	WIDOWED	WHITE	
523739	19999987	23865745	2145-11-02 21:38:00	2145-11-11 12:57:00	EW EMER.	EMERGENCY ROOM	REHAB	Other	ENGLISH	NaN	UNKNOWN	

523740 rows × 18 columns

Diagnosi (diag.ipynb):

anche il dataset delle diagnosi fa riferimento alle ammissioni è stato possibile utilizzarlo.

Prima di unire i dataset è stato utile analizzare i codici delle diagnosi che sono espressi in due versioni diverse, icd9 e icd10. Questi codici rappresentano la diagnosi che è stata assegnata al paziente e è possibile raggrupparle in delle macrocategorie a seconda del codice.

Per poter ottenere le categorie delle diagnosi si è deciso di convertire i codici icd9 in icd10 e tale fine è stato usato uno script reperito su github, tale script tuttavia non è perfetto e ha commesso alcuni errori che però sono rimasti sotto il 2% di tutte le diagnosi.

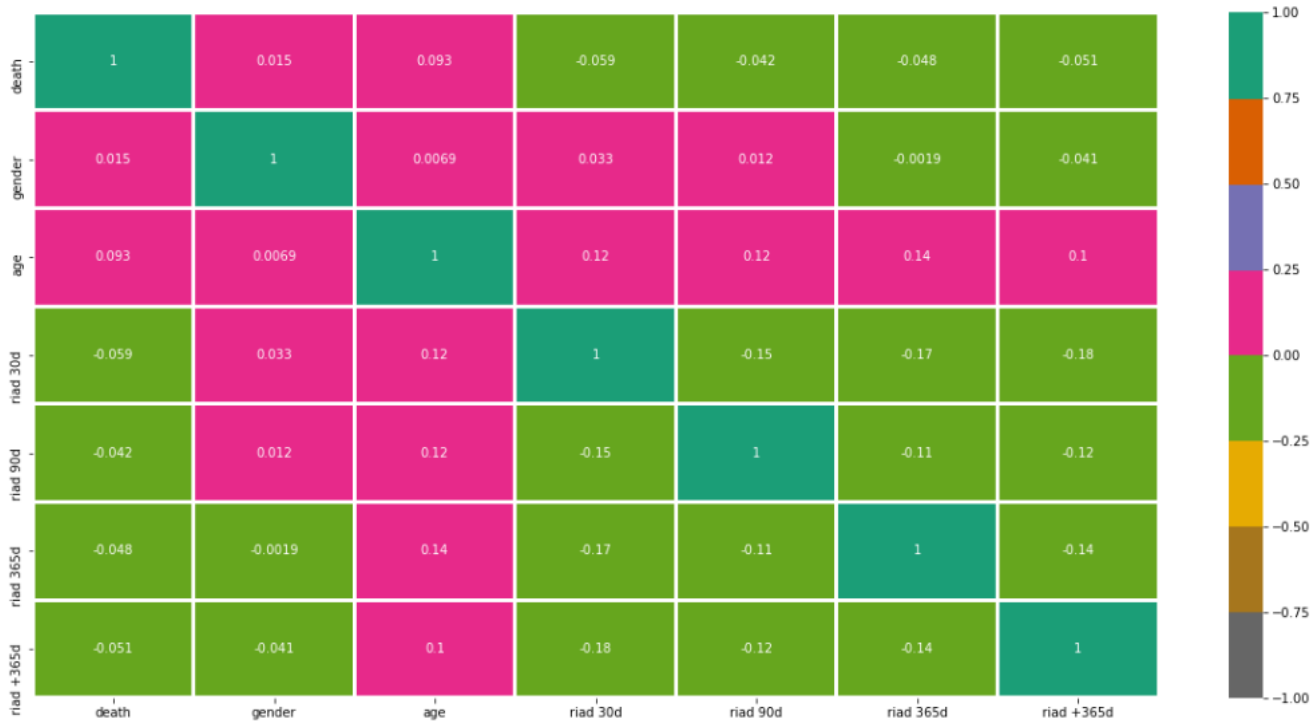
Una volta suddivise le diagnosi in categorie ho provveduto a raggruppare le tuple che si riferivano alla stessa ammissione in quanto in ogni ammissione potevano essere assegnate più diagnosi al paziente.

	hadm_id	icd_D_blood	icd_D_health_status	icd_D_pregnancy	icd_D_abnormal	icd_D_chromosomal	icd_D_infectious_parasitic	icd_D_musculoskeletal
0	20000019	1	1	0	0	1	2	0
1	20000024	1	1	0	1	0	0	1
2	20000034	1	6	0	5	0	0	1
3	20000041	0	4	0	0	0	0	1
4	20000055	0	2	0	0	0	0	0
...	...	...	...	...	...	...	...	...
521106	29999745	0	2	0	0	0	0	0
521107	29999785	0	4	0	0	0	0	0
521108	29999809	0	2	0	0	0	2	0
521109	29999828	0	1	0	1	0	0	0
521110	29999928	1	2	0	0	0	0	0

521111 rows × 25 columns

Exploratory data analysis

Nella fase di esplorazione dei dati ho analizzato le tabelle unite *admpat* e *diag* sulle quali avrei fatto le predizioni. Per analizzare i dati ho tracciato alcuni grafici che permettessero di dare una rappresentazione visiva un po' più intuitiva dei dati, oltre a questo è stata definita la seguente matrice di correlazione



La matrice di correlazione evidenzia come non esistano feature estremamente predittive, l'età risulta essere una delle più correlate agli attributi che vogliamo predire ma comunque non ha un valore molto significativo. La matrice ci mostra che tra i dati di un'ammissione le predizioni probabilmente non arriveranno a una precisione significativamente alta.

Predizione in hospital mortality

Per l'allenamento e la predizione si è deciso di suddividere il dataset in training (60%), validation (10%), test (30%).

Come modelli sono stati utilizzati l'SDGClassifier, il DcisionTreeClassifier, il LogisticRegressor, il RandomForestClassifier, il MultilayerPerceptronClassifier, il SupportVectorClassifier e il GaussianNaiveBayes. Allenando i modelli sul training set per provare a predirne poi le label i valori di precisione sono stati i seguenti.

```
In [250]: from sklearn.metrics import accuracy_score

          for i in range(len(models)):
              model = models[i](random_state=42)  # inizializzo il modello qui (non ancora addestrato)
              name = names[i]
              model.fit(X_train_ihm_scaled, y_train_ihm) # qui addestro il modello sui dati
              y_pred_ihm = model.predict(X_train_ihm_scaled)
              print(f'accuracy of {name}: ',accuracy_score(y_pred_ihm, y_train_ihm))

          accuracy of SGD Classifier:  0.9821023072543865
          accuracy of Decision Tree Classifier:  0.9998368866458137
          accuracy of RandomForestClassifier:  0.9998304900436888
          accuracy of Logistic Regressor:  0.9814914317514536
          accuracy of MLP Classifier:  0.9848592427702404
          accuracy of Support Vector Classifier:  0.9825052931882584
          accuracy of Gaussian NB:  0.745111396826006
```

Tali risultati non sono ovviamente poco significativi in quanto sono predizioni eseguite sul training set ma ci permettono di dare una prima valutazione delle prestazioni dei modelli utilizzati e ciò ci permette di effettuare una prima selezione sui metodi da portare avanti e approfondire. Tra i metodi esaminati si è scelto di abbandonare il support vector classifier e il gaussian naive bayes in quanto il primo non produce risultati migliori degli altri metodi nonostante impieghi un tempo significativamente maggiore per l'addestramento e il secondo produce scarsi risultati persino sul training set.

Come accennato sopra questi risultati sono comunque poco significativi perché le predizioni sono eseguite sul training set e dunque i modelli hanno già visto i dati durante l'allenamento e questo è il motivo di queste prestazioni elevate apparenti, per ridurre l'overfitting è stato quindi deciso di utilizzare la cross-validation in modo che una parte del training set alla volta fosse utilizzata come validazione su cui testare il modello.

A seguito della cross-validation la precisione come è immaginabile è diminuita e calcolando precision, recall e f1 score è risultato evidente che uno dei più grandi problemi al momento fosse lo sbilanciamento del dataset. Avendo un dataset fortemente sbilanciato i modelli tendono a predire quasi solamente una delle due classi e per ovviare al problema ho deciso di applicare la tecnica dell'oversampling.

Dopo l'oversampling addestrando i modelli sul nuovo training set bilanciato i valori di precision, recall e f1 score in cross-validation si sono alzati.

RANDOM FOREST PERFORMANCES

	Metric	pre-oversampling	post-oversampling
0	precision	0.484536	0.996635
1	recall	0.008399	0.985837
2	f1	0.016512	0.991207

LOGISTIC REGRESSION PERFORMANCES

	Metric	pre-oversampling	post-oversampling
0	precision	0.362667	0.842743
1	recall	0.048606	0.879741
2	f1	0.085723	0.860844

MULTILAYER PERCEPTRON PERFORMANCES

	Metric	pre-oversampling	post-oversampling
0	precision	0.321287	0.954951
1	recall	0.130272	0.966535
2	f1	0.185378	0.960708

Questi valori tuttavia sono sospettamente alti, e per verificarli ho utilizzato il validation set. ma prima ho provato a effettuare il tuning degli iperparametri che tuttavia non ha mostrato grandi possibilità di miglioramento se non per il multilayer perceptron.

Eseguendo la predizione sul validation set dopo l'allenamento sul training conferma che le prestazioni estremamente elvate che si potevano osservare non erano realistiche e attendibili e ci fornisce dei risultati più realistici che probabilmente saranno vicini ai risultati reali delle predizioni finali sul test set.

LOGISTIC REGRESSOR VALIDATION PERFORMANCES

	Metric	validation
0	precision	0.086218
1	recall	0.848875
2	f1	0.156537

RANDOM FOREST VALIDATION PERFORMANCES

	Metric	validation
0	precision	0.441860
1	recall	0.081458
2	f1	0.137557

MULTILAYER PERCEPTRON VALIDATION PERFORMANCES

	Metric	validation
0	precision	0.177677
1	recall	0.418006
2	f1	0.249361

Il multilayer perceptron eseguendo il tuning degli iperparametri e impostando i parametri indicati come migliori ha poi permesso di ottenere dei risultati migliori

MULTILAYER PERCEPTRON VALIDATION PERFORMANCES

	Metric	validation
0	precision	0.119963
1	recall	0.702036
2	f1	0.204912

I valori ottenuti con tutti e tre i modelli sono evidentemente non ottimali ma probabilmente se tra questi un ospedale dovesse affidare la predizione di un evento delicato come la morte di una persona probabilmente sarebbe meglio il logistic regressor o il multilayer perceptron in quanto hanno ottenuto una recall elevata che mostra come una maggior parte dei casi positivi sia stata classificata correttamente dal modello, nonostante per fare ciò siano stati predetti molti falsi positivi.

Applicando le predizioni sul test set abbiamo la conferma di valori simili a quelli ottenuti nel validation set.

LOGISTIC REGRESSOR FINAL PERFORMANCES

	Metric	final
0	precision	0.085407
1	recall	0.849281
2	f1	0.155206

```
print("{} veri positivi".format(cm_lr_final[1,1]))
print("{} veri negativi".format(cm_lr_final[0,0]))
print("{} falsi positivi".format(cm_lr_final[0,1]))
print("{} falsi negativi".format(cm_lr_final[1,0]))
```

```
2361 veri positivi
128270 veri negativi
25283 falsi positivi
419 falsi negativi
```

MULTILAYER PERCEPTRON FINAL PERFORMANCES

	Metric	final
0	precision	0.121900
1	recall	0.714388
2	f1	0.208263

```
print("{} veri positivi".format(cm_mlp_final[1,1]))
print("{} veri negativi".format(cm_mlp_final[0,0]))
print("{} falsi positivi".format(cm_mlp_final[0,1]))
print("{} falsi negativi".format(cm_mlp_final[1,0]))
```

```
1986 veri positivi
139247 veri negativi
14306 falsi positivi
794 falsi negativi
```

Predizione riadmission in 30 days

Partendo dallo stesso dataset utilizzato per la predizione della mortalità ospedaliera ma selezionando alcune feature diverse ho seguito un modello molto simile per la predizione delle riammissioni entro 30 giorni. I set di train, validation e test sono stati costruiti in modo analogo e si sono presentati problemi di sottorappresentazione di una delle due classi come per la

morte ospedaliera (nonostante in misura minore), si è quindi scelto di adottare nuovamente l'oversampling che ha portato a risultati migliori nella predizione.

La validazione del modello ha mostrato comunque una discesa delle prestazioni e osservate nel training set e risultati della validazione sono stati i seguenti.

RANDOM FOREST VALIDATION PERFORMANCES

	Metric	validation
0	precision	0.449444
1	recall	0.094979
2	f1	0.156818

LOGISTIC REGRESSOR VALIDATION PERFORMANCES

	Metric	validation
0	precision	0.269855
1	recall	0.733654
2	f1	0.394576

In questo caso è più difficile definire quale dei due potrebbe essere preferibile, ma probabilmente considerando l'f1 score che media tra precision e recall il logistic regressor rimane preferibile e applicandolo sul test set confermiamo i risultati ottenuti nella validazione.

LOGISTIC REGRESSOR FINAL PERFORMANCES

	Metric	final
0	precision	0.273248
1	recall	0.719590
2	f1	0.396089

Predizione riadmission in 90/365 days

La predizione delle riammissioni in 90 e 365 giorni è stata effettuata in maniera analoga alla predizione per la riammissione entro 30 giorni e dato che entrambe le classi erano ancora meno rappresentate è stato importante effettuare l'oversampling.

Nonostante tutto però anche qui i risultati delle predizioni una volta validati non sono stati particolarmente significativi e il logistic regressor è rimasto il metodo che performava meglio.

Conclusioni

Come era intuibile dalla matrice di correlazione i risultati ottenuti non sono mai stati davvero significativi e le predizioni sono state poco precise, per poter effettuare delle predizioni migliori sarebbe stato importante avere correlazioni maggiori ma dopotutto è comprensibile che sia

difficile predire con precisione quale sarà il risultato di un ricovero in ospedale solo dai pochi dati disponibili all'ammissione di un paziente.