

Census income prediction project

Nicolò Sansevrino 865889

June 16, 2024

Contents

1	Introduzione, dominio e obiettivi	3
2	Analisi esplorativa e Data visualization	4
2.1	Proprietà generali del dataset	4
2.2	Statistiche univariate	6
2.2.1	Distribuzione variabili	7
2.2.2	Boxplot	8
2.3	Statistiche bivariate	10
2.3.1	Distribuzione variabili in funzione del target	10
2.3.2	Boxplot bivariati	11
3	Preprocessing	13
3.1	Valori duplicati	13
3.2	Valori null e Nan	13
3.2.1	Valori mancati in native country	14
3.2.2	Valori mancanti in workclass e occupation	14
3.3	Gestione colonna 'target'	14
3.4	Codifica colonna 'sex'	15
3.5	Correlazione tra colonna education e education-num	15
3.6	Fusione colonne capital-gain e capital-loss	16
3.7	Gestione 'fnlwgt'	16
3.8	Outliers	17
4	Feature engineering	18
4.1	Codifica delle feature categoriche	18
4.2	Standardizzazione delle feature numeriche	18
4.3	Split in train e test	18
4.4	Ricampionamento del training set	18
4.4.1	SMOTE	19
4.4.2	TomekLinks	19
4.4.3	SMOTE + TomekLinks	20

5 Esperimenti sui modelli	21
5.1 Alberi decisionali	21
5.1.1 Esperimento 1	21
5.1.2 Esperimento 2	30
5.1.3 Esperimento 3	32
5.1.4 Esperimento 4	34
5.1.5 Conclusioni sugli esperimenti sui Decision Tree	36
5.2 Support Vector Machines	36
5.2.1 Esperimento 1	37
5.2.2 Esperimento 2	37
5.2.3 Esperimento 3	38
5.2.4 Conclusioni sugli esperimenti su svm	39
6 Tecniche di validazione	40
6.1 Tecniche di validazione sui Decision Tree	40
6.1.1 Stratified k-fold cross validation esperimento 1	40
6.1.2 Stratified k-fold cross validation esperimento 2	41
6.1.3 Stratified k-fold cross validation esperimento 3	42
6.1.4 Stratified k-fold cross validation esperimento 4	42
6.2 Tecniche di validazione su Support Vector Machines	43
6.2.1 Stratified k-fold cross validation esperimento 1	43
6.2.2 Stratified k-fold cross validation esperimento 2	44
6.2.3 Stratified k-fold cross validation esperimento 3	45
7 Analisi dei risultati	47
7.0.1 Analisi su Decision Trees	47
7.0.2 Analisi su Support Vector Machines	47
7.0.3 Confronto tra le due famiglie di modelli	47
8 Conclusioni	48

1 Introduzione, dominio e obiettivi

Il progetto mira a definire dei modelli di machine learning in grado di predire lo stato economico sociale sulla base di alcune caratteristiche, quali titolo di studio, età, paese di origine, ore lavorate alla settimana ecc. Il dataset (presente sia su Kaggle che su UCI) raccoglie dati inerenti a una porzione di cittadini statunitensi prelevati dal US Census Bureau per l'anno 1994 (in aggiornamento) disponibile al seguente link: Adult census income.

Il percorso seguito durante il lavoro e che verrà usato come scaletta per queste presentazione è il seguente:

- Analisi esplorativa e Data visualization: per costruire dei classificatori su qualsivoglia problema è necessario studiarne a fondo la natura, obiettivo di questo punto
- Preprocessing: il dataset viene sottoposto ad una serie di modifiche per permettere un migliore apprendimento dei modelli
- Feature engineering: similmente alla fase precedente, il dataset viene trattato con lo scopo di ottenerne varie versioni per condurre poi gli esperimenti
- Esperimenti sui modelli: in questa fase costruiamo i classificatori e condurremo diversi esperimenti. Questi riguarderanno classificatori di famiglie diverse, diverse versioni di modelli e allenati su diversi dataset
- Tecniche di validazione: usiamo tecniche di validazione note per osservare i risultati dei modelli
- Analisi dei risultati: commentiamo i risultati
- Conclusioni: commentiamo la natura del problema, limiti e possibilità

2 Analisi esplorativa e Data visualization

Il nostro dataset è etichettato con due valori: $\leq 50k$, $> 50k$. Queste due etichette indicano il guadagno annuale in migliaia di dollari di un lavoratore statunitense. Le feature del dataset sono:

- age: età della persona
- workclass: macro categoria di tipo di lavoratore (a livello burocratico)
- fnlwgt: peso assegnato dal US Census Bureau che indica quanto una certa istanza rappresenta la popolazione (questa colonna non verrà usata come una feature)
- education: livello di istruzione
- education-num: numero di anni spesi nel mondo dell'istruzione
- marital-status: stato civile
- occupation: macro categoria di lavoro (a livello di settore)
- relationship: ruolo familiare
- race: etnia di appartenenza
- sex: genere registrato
- capital-gain: entrate economiche al di fuori del lavoro (es: investimento)
- capital-loss: perdite economiche al di fuori del lavoro
- hours-per-week: numero di ore lavorate alla settimana
- native-country: paese di origine

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba
...
48837	39	Private	215419	Bachelors	13	Divorced	Prof-specialty	Not-in-family	White	Female	0	0	36	United-States
48838	64	NaN	321403	HS-grad	9	Widowed	NaN	Other-relative	Black	Male	0	0	40	United-States
48839	38	Private	374983	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband	White	Male	0	0	50	United-States
48840	44	Private	83891	Bachelors	13	Divorced	Adm-clerical	Own-child	Asian-Pac-Islander	Male	5455	0	40	United-States
48841	35	Self-emp-inc	182148	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	60	United-States

Come possiamo notare il dataset può essere ridondante e ci occuperemo di questo aspetto in seguito.

2.1 Proprietà generali del dataset

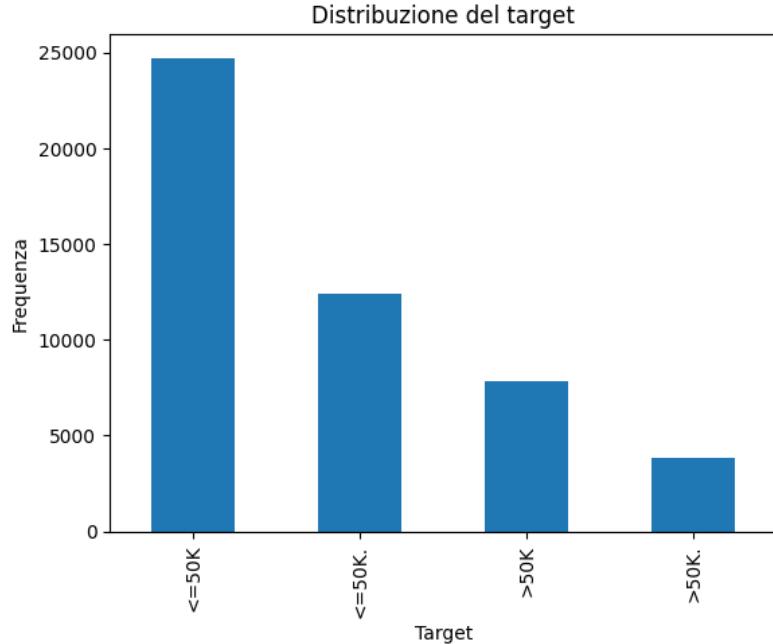
Nella tabella di seguito vengono sintetizzate alcune informazioni ottenibili attraverso le funzioni messe a disposizione dalla libreria Pandas.

index	name	role	type	demographic	Dtype
0	age	Feature	Integer	Age	int64
1	workclass	Feature	Categorical	Income	object
2	fnlwgt	Feature	Integer	None	int64
3	education	Feature	Categorical	Education Level	object
4	education-num	Feature	Integer	Education Level	int64
5	marital-status	Feature	Categorical	Other	object
6	occupation	Feature	Categorical	Other	object
7	relationship	Feature	Categorical	Other	object
8	race	Feature	Categorical	Race	object
9	sex	Feature	Binary	Sex	object
10	capital-gain	Feature	Integer	None	int64
11	capital-loss	Feature	Integer	None	int64
12	hours-per-week	Feature	Integer	None	int64
13	native-country	Feature	Categorical	Other	object
14	income	Target	Binary	Income	object

```
< class 'pandas.core.frame.DataFrame'>
RangeIndex: 48842 entries, 0 to 48841
dtypes: int64(6), object(9)
memory usage: 5.6+ MB
Valori duplicati: 29
Valori null: 963 in workclass, 966 in occupation, 274 in native country
```

Tuttavia quest ultimo dato è da prendersi con cautela: infatti nel dataset alcuni valori mancanti sono indicati con '?', mentre altri con ',' in ogni caso stanti ad indicare NaN, ma purtroppo la funzionalità di riconoscimento Nan integrata nella libreria Pandas non risulta in grado di rilevare questa differenza. La trattazione della differenza tra queste due notazioni verrà trattata più avanti.

Distribuzione variabile target:



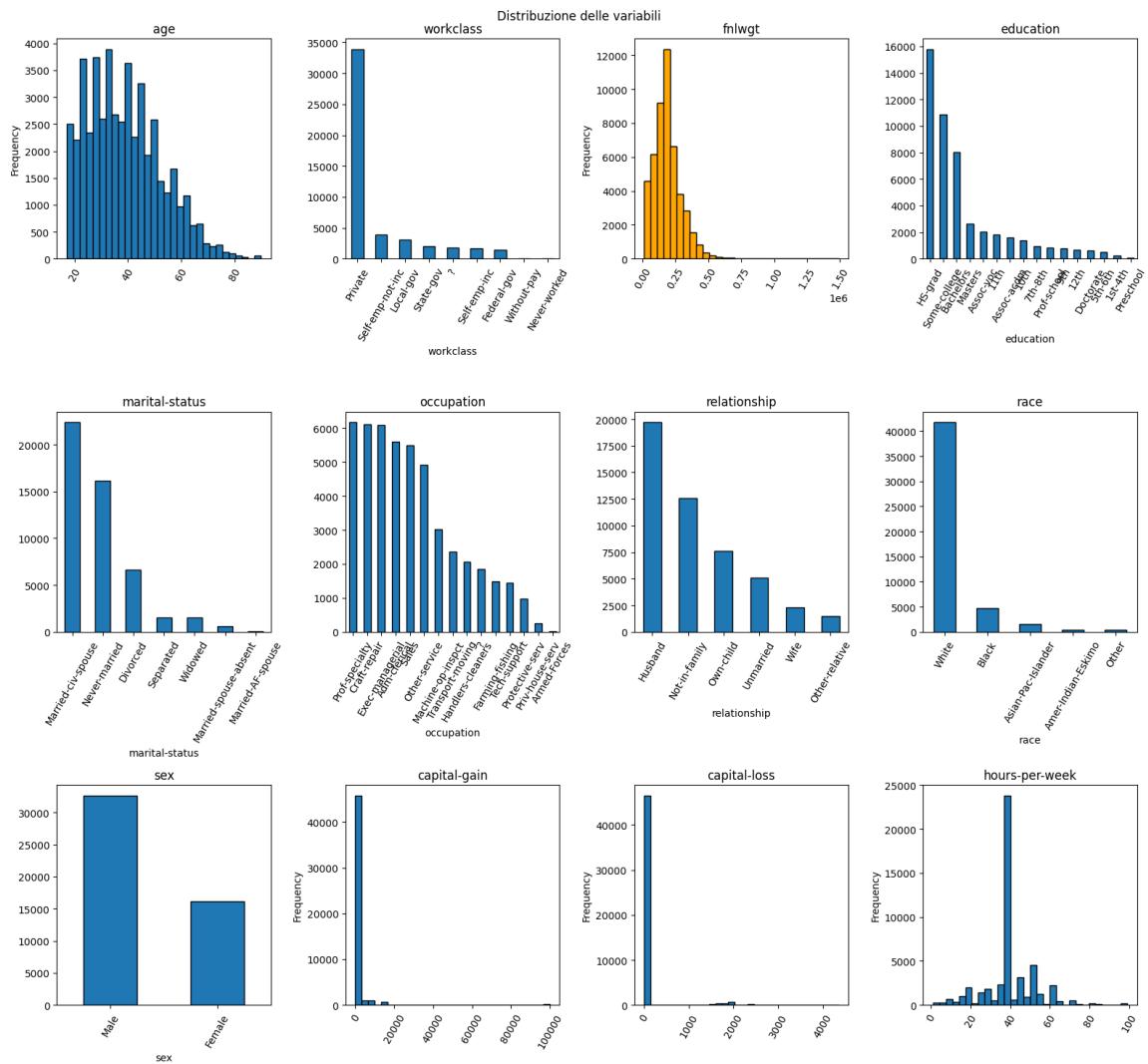
income	count
$\leq 50K$	24720
$\leq 50K.$	12435
$> 50K$	7841
$> 50K.$	3846

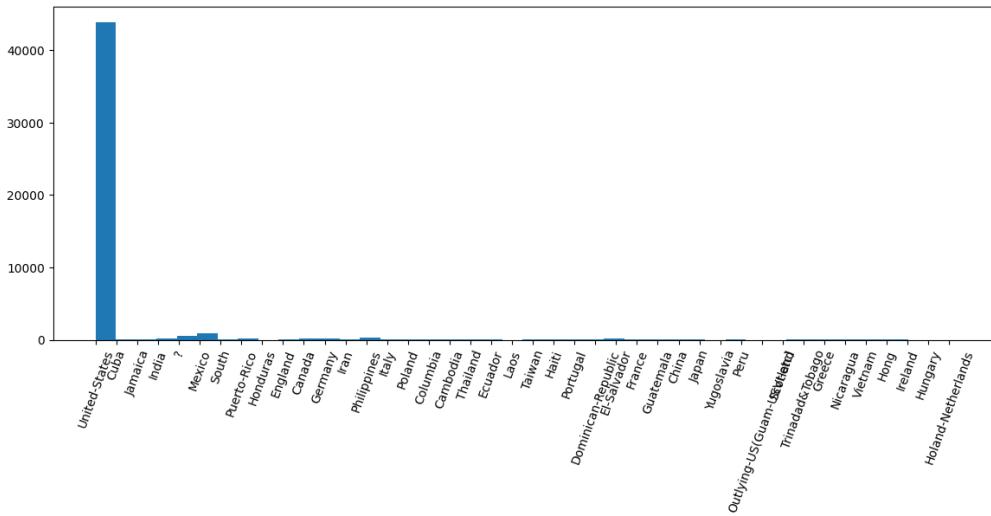
Notiamo che la colonna 'income' presenta più di due valori: accorperemo ' $\leq 50K$ ' e ' $\leq 50K.$ ' nello stesso valore. Lo stesso vale per ' $> 50K$ ' e ' $> 50K.$ ' Il nostro problema è binario: definiamo true quando il guadagno è maggiore di 50k, false quando è minore o uguale.

2.2 Statistiche univariate

Ora procediamo a visualizzare le statistiche del dataset. Il primo tipo di grafico che sceglieremo è quello che ci mostra la distribuzione individuale delle variabili. Dalla visualizzazione escludiamo 'education-num' poichè come vedremo più avanti contiene le stesse informazioni di 'education', 'native country' per rendere più pulita la realizzazione dei grafici. Infine 'fnlwgt' non è una feature, bensì un peso e non ha senso considerarlo come le altre feature, tuttavia lo visualizziamo per farci comunque un'idea.

2.2.1 Distribuzione variabili



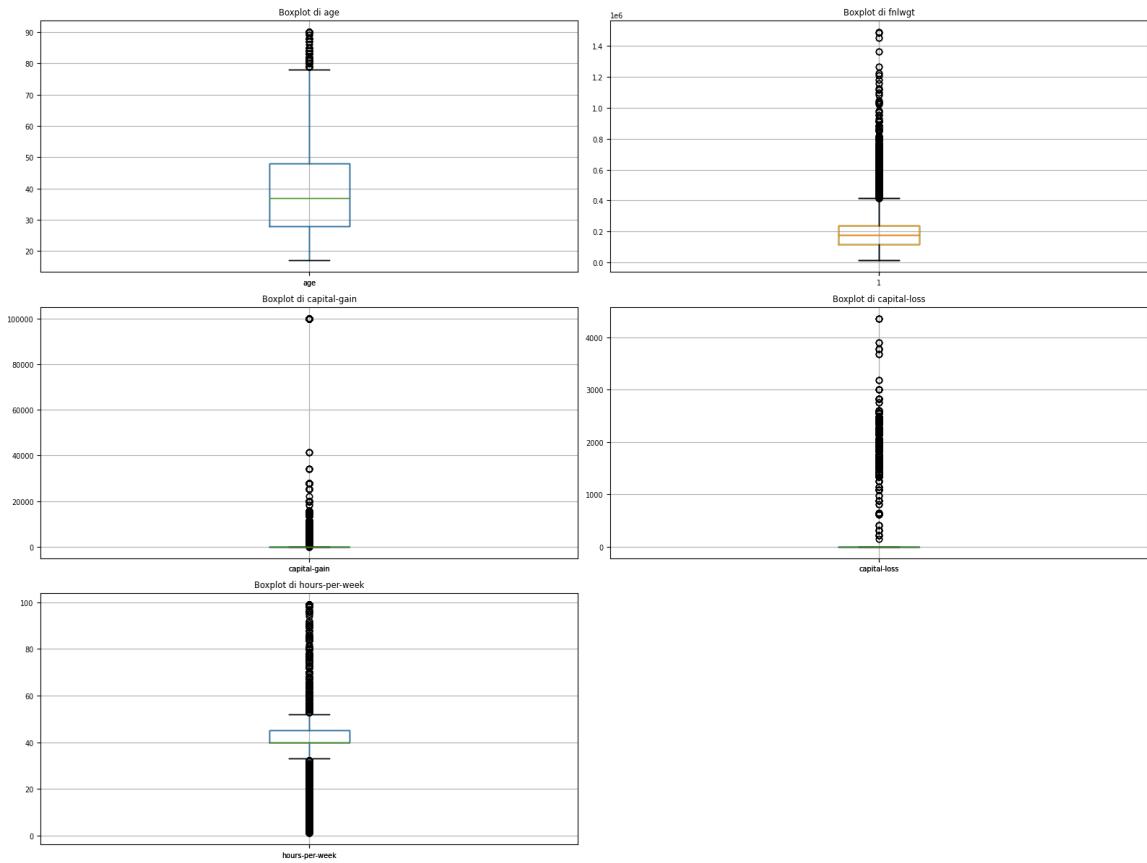


Alcune considerazioni che si possono fare in merito alla distribuzione dei dati:

- la fascia di età maggiormente rappresentata va dai 18 ai 50 anni circa
- i privati sono i tipi di contratti maggiormente diffusi
- il titolo di studio più comune è il diploma
- lo stato civile più rappresentato è quello di persone sposate
- l'etnia più rappresentata è quella bianca
- il genere più rappresentato è quello maschile
- è difficile guadagnare (o perdere) tanto capitale all'esterno del mondo del lavoro (comunque sono entrambi 0 nella maggior parte dei casi)
- la maggior parte delle persone lavora 40 ore a settimana
- il paese di provenienza più rappresentato sono gli Stati Uniti

2.2.2 Boxplot

Continuiamo ora con i boxplot per la visualizzazione di eventuali outliers. Per quanto riguarda i boxplot osserviamo solo i dati di natura numerica. Inoltre non ha senso osservare 'education-num' perché è la codifica numerica del titolo di studio, però osserviamo 'fnlwgt' prestando attenzione a pensarlo come a un peso anzichè come a una feature.



Grazie ad un'apposita funzione possiamo riassumere che gli outliers sono i seguenti:

- outliers in age: 216 and their medium weight is: 155606
- outliers in fnlwgt: 1453 and their medium weight is: 515040
- outliers in capital-gain: 4035 and their medium weight is: 187001
- outliers in capital-loss: 2282 and their medium weight is: 188025
- outliers in hours-per-week: 13496 and their medium weight is: 185293

Inoltre: Average weight of dataset is: 189664 , +- 105604

Delle quattro feature studiate si può dire che:

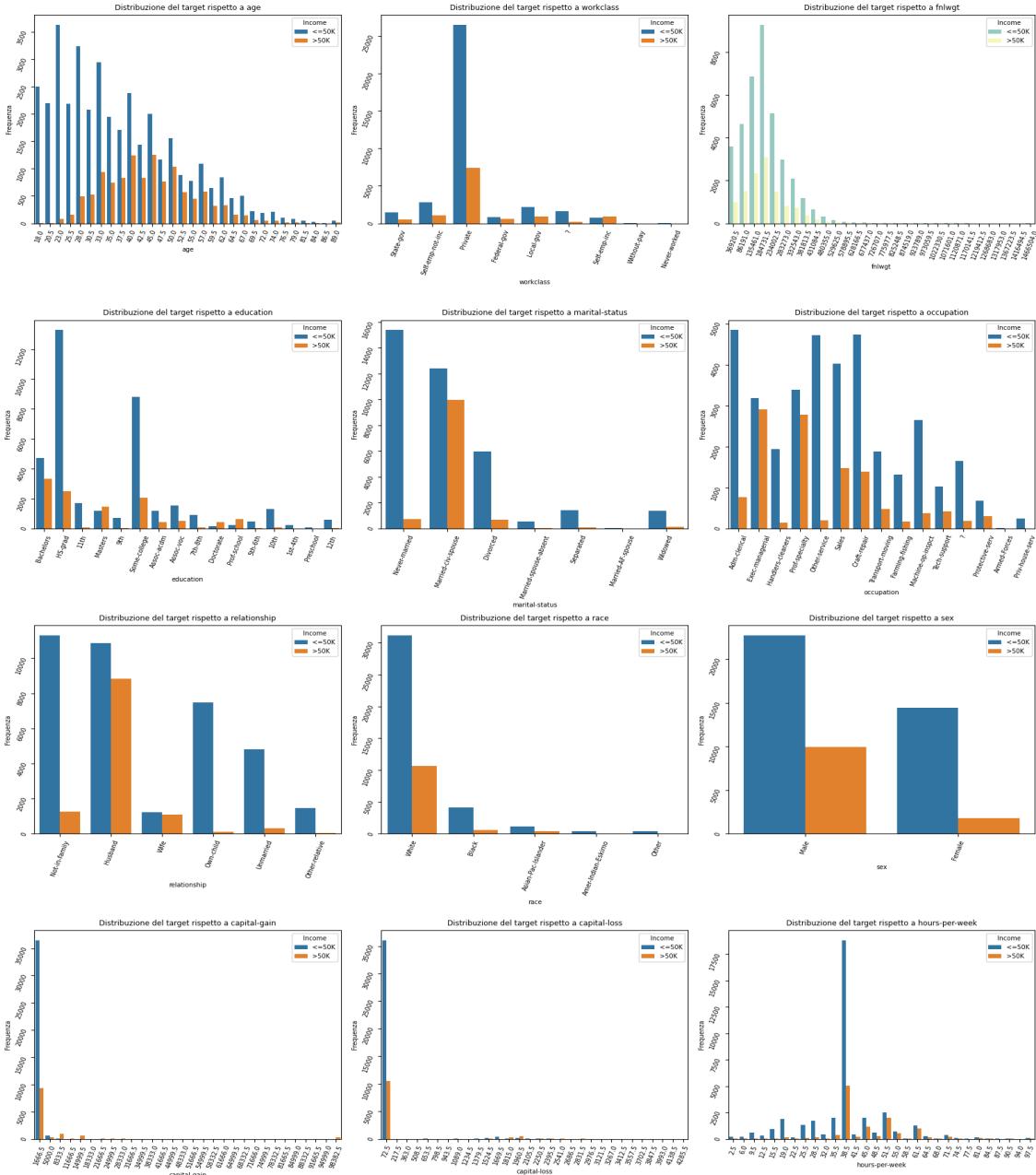
- l'età media è di circa 38 anni e sono presenti outliers superiori
- i capital loss e gain medi sono vicini a 0, ma esistono outliers superiori
- le ore medie lavorate sono 40, ma esistono eccezioni sia superiori che inferiori
- il valore medio di fnlwgt è 189000, ma gli outliers (solo superiori) hanno un valore molto grande

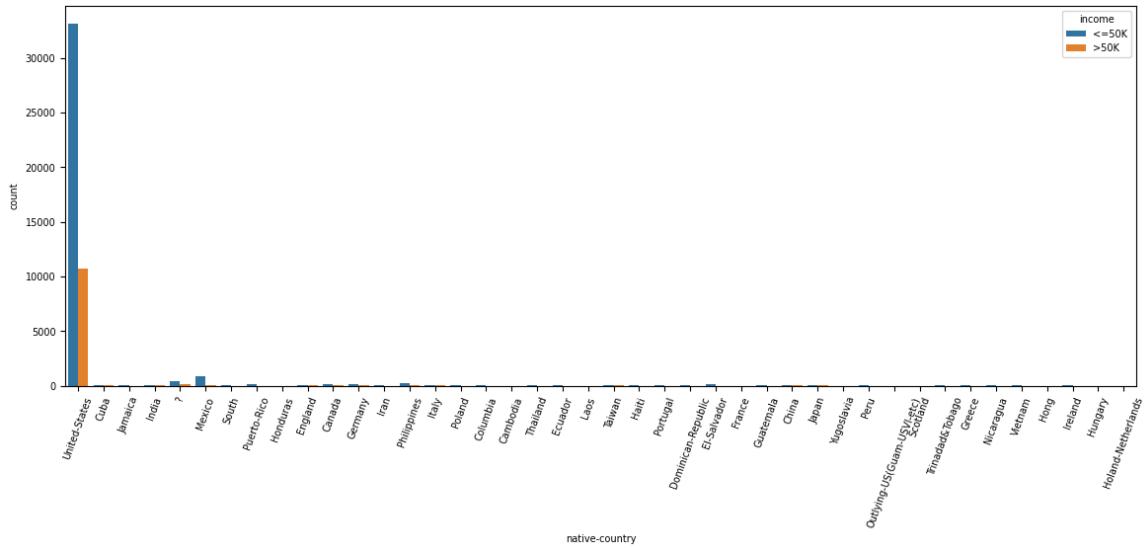
Gli outliers di age, capital-gain, capital-loss e hours-per-week hanno un peso che non si discosta molto dalla media del peso del dataset. Anche la stessa colonna fnlwgt ha degli outliers, ma ricordiamoci che fnlwgt non è una feature: eliminare gli outliers di fnlwgt significherebbe distorcere la rappresentazione della popolazione. Gli outliers delle altre colonne sono molti e sarà necessario gestirli.

2.3 Statistiche bivariate

Ora osserviamo la relazione che c'è tra la colonna target 'income' e le altre colonne del dataset. Nel fare ciò escludiamo nuovamente 'education-num' e riportiamo invece fnlwgt per verificare la distribuzione del target in rapporto ai pesi.

2.3.1 Distribuzione variabili in funzione del target





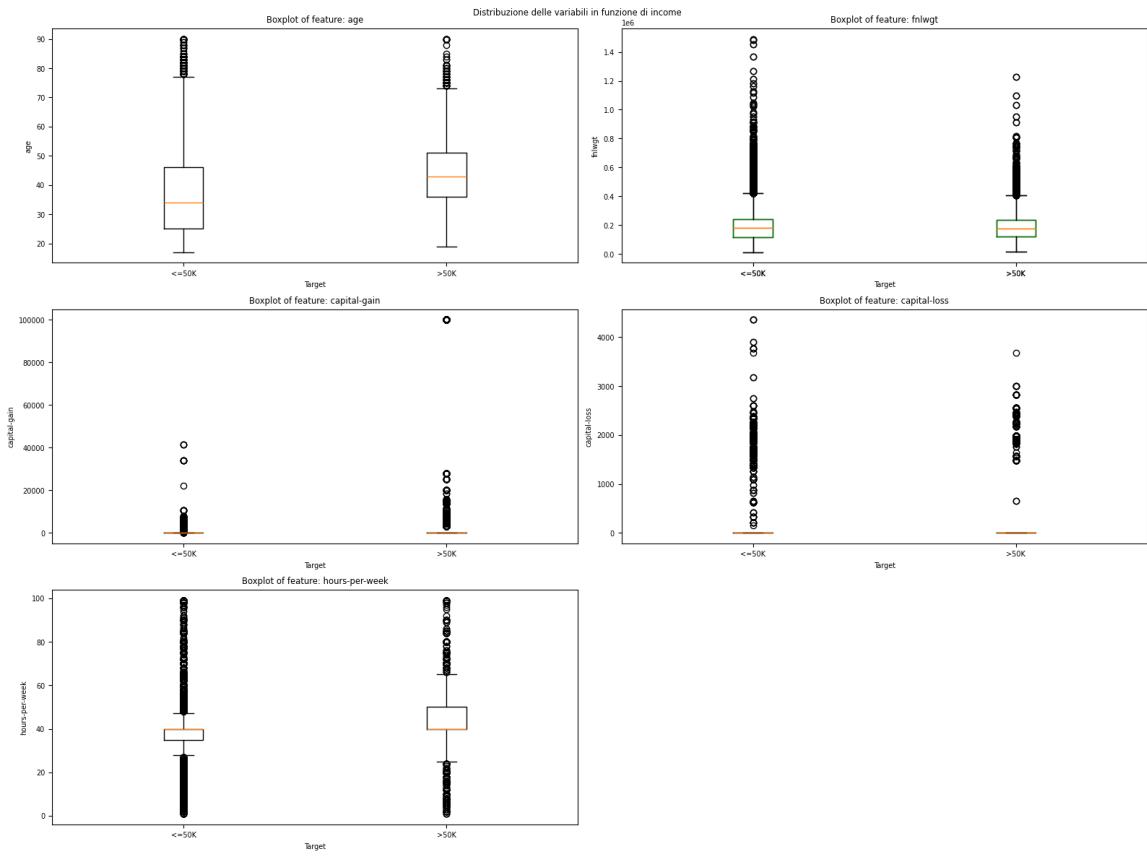
Si possono fare una serie di considerazioni in merito al rapporto tra la soglia di guadagno individuale e le condizioni personali:

- l'età è un fattore molto correlato al guadagno, ciò che emerge è che a guadagnare meno di 50.000 dollari l'anno sono giovani ed anziani, mentre le fasce di età intermedie (presumibilmente in carriera) presentano più persone che guadagnano almeno tale cifra.
- chi lavora in proprio (es: freelancer, ha uno studio proprio) è più probabile che guadagni almeno 50.000 dollari l'anno, mentre per gli altri tipi è molto più probabile guadagnare meno.
- esistono più i laureati (magistrale), dottorati e insegnanti scolastici che guadagnano più di 50.000 dollari l'anno rispetto a quelli che ne guadagnano meno. La categoria dei laureati triennali tende a guadagnare meno di 50.000 dollari l'anno, tuttavia è la categoria con maggiore equilibrio.
- gli insegnanti specializzati e le posizioni manageriali sono le categorie più equilibrate tra chi guadagna più e meno di 50.000 dollari l'anno, mentre le categorie meno equilibrate sono quelle che lavorano in ambito amministrativo, clericale e "altri servizi"
- le persone sposate hanno una probabilità vicina al 50% di guadagnare almeno 50.000 dollari l'anno, negli altri casi meno
- circa un individuo di etnia bianca su 4 guadagna almeno 50.000 dollari l'anno. Per le altre etnie (eccetto asian-pac-island) tale rapporto è più piccolo
- circa un maschio su 3 guadagna almeno 50.000 dollari l'anno, mentre per le donne il rapporto è di circa 1 su 10.

Si noti che in merito a fnlwgt i valori del target seguono curve simili: ciò significa che le istanze del dataset sono pesate a prescindere dal loro guadagno.

2.3.2 Boxplot bivariati

Ora procediamo con dei boxplot in cui includiamo il confronto con la colonna target



Emerge che l'età media di chi guadagna almeno 50.000 dollari l'anno è maggiore di chi ne guadagna meno e che chi guadagna di più è anche più probabile che lavori più ore della media. Abbondano gli outliers.

Infine anche in questo caso il target rispetto a fnlwgt è equidistribuito.

3 Preprocessing

Il dataset presenta alcuni valori mancanti, duplicati e ridondanti. Inoltre è sbilanciato pertanto occorre adottare strategie di bilanciamento del target. Possiamo eseguire alcune ottimizzazioni:

- gestione delle righe duplicate
- gestione delle righe con valori null e nan
- verifica del contenuto delle colonne 'education' ed 'education num' (potrebbero contenere le stesse informazioni)
- codifica dei due generi (Male e Female) in 0 e 1.
- accorpamento dei valori del target
- fusione di 'capital gain' e 'capital loss'
- gestione di 'final weight'
- gestione degli outliers

Le colonne con valori mancanti sono: workclass, occupation e native-country. Tutte queste colonne sono di tipo Categorico.

Seguirà anche il bilanciamento del dataset

3.1 Valori duplicati

Le righe sono duplicate e non abbiamo chiavi primarie o criteri univoci per distinguerle, quindi in assenza di altre informazioni il modo più opportuno per gestire questi valori duplicati è eliminarli. Forma del dataset prima dell'eliminazione delle righe duplicate: (48842, 15) Forma del dataset dopo l'eliminazione delle righe duplicate: (48794, 15)

3.2 Valori null e Nan

Per prima cosa osserviamo che esistono due notazioni nel dataset, ovvero ',' e ',?,'. Non essendo indicata la differenza tra le due verranno trattate entrambe come dati ignoti. Modifichiamo la notazione del dataset e uniamole in una notazione unica, ovvero ',' in modo che la libreria Pandas riconosca come Nan questi valori.

Dopo aver apportato queste modifiche la conta dei Nan è cambiata: 2799 in workclass, 2809 in occupation, 856 in native country.

Ora procediamo a gestire questi valori mancanti. Useremo approcci diversi.

Ragioniamo sulla feature 'native country'. Come possiamo osservare il valore più frequente è 'Stati Uniti'. Il dataset raccoglie informazioni presenti nel US Census Bureau in merito a cittadini che lavorano negli Stati Uniti di origine statunitense e di altre origini. Un possibile modo per gestire il valore mancante è imputare il valore con quello più frequente nel dataset (Stati Uniti), tuttavia questa assunzione può intrudurre bias. Infatti i dati sono relativi a persone che lavorano negli Stati Uniti, ma non per forza di origine statunitense.

Assumere che le persone di origine non nota non siano statunitensi è un'assunzione più sensata che assumere lo siano in virtù dello sviluppo avanzato del paese rispetto a quelli presenti nel dataset. Per questo motivo eliminare le istanze di origine non nota distorce meno il dataset rispetto ad imputare 'Stati Uniti' come paese nativo.

Per quanto riguarda invece le colonne 'workclass' e 'occupation' è osservabile (eccetto in rari casi) che i valori null sono presenti sulle stesse righe. Essendo le due feature legate, è normale che l'assenza

di un'informazione si riverberi sull'assenza dell'altra.

A differenza della colonna 'native country' non c'è solo la possibilità che il dato sia ignoto, ma anche che non esista: è possibile per una persona essere (attualmente) senza lavoro, non è possibile non avere un paese di origine.

Questo potrebbe suggerire che una persona è attualmente senza lavoro, quindi conserveremo questa informazione.

3.2.1 Valori mancati in native country

Procediamo a gestire le righe con valori mancanti rimuovendole.

Dimensione del dataset prima: (48794, 15)

Dimensione del dataset dopo: (47938, 15)

3.2.2 Valori mancati in workclass e occupation

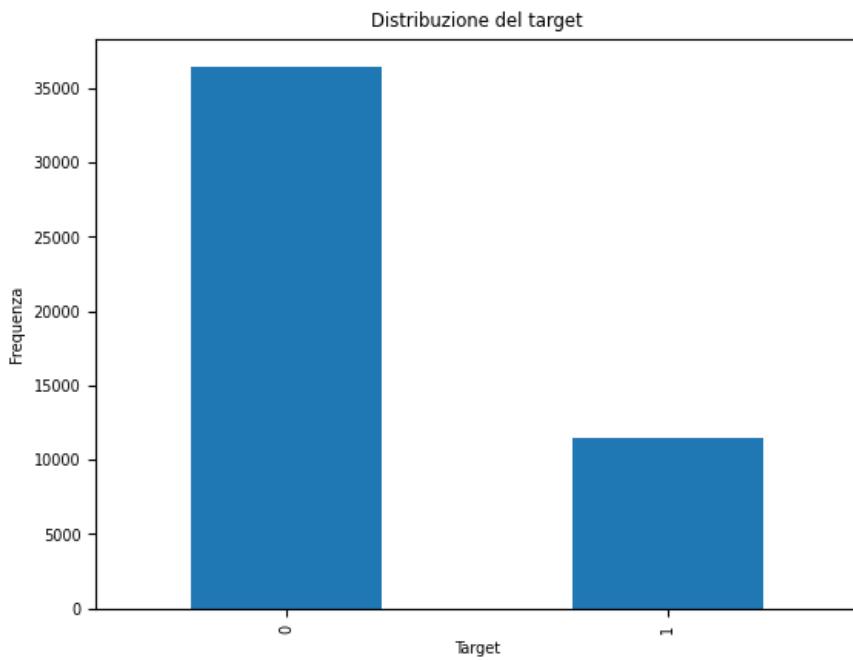
Per quanto riguarda 'workclass' e 'occupation' la questione è diversa. Infatti non è noto se i valori mancanti si riferiscono a una persona inoccupata, oppure se il dato è effettivamente sconosciuto.

Per eliminare valori Nan sostituiremo i valori assenti con la stringa 'unknown'. Però c'è un problema, infatti i valori Nan nella colonna 'workclass' e nella colonna 'occupied' differiscono di 10. Studiando queste 10 righe separatamente scopriamo che la differenza numerica è spiegata dal fatto che esistono persone nel dataset che non hanno mai lavorato. Per queste istanze sostituiremo 'Never-worked' nella colonna 'occupation'.

Infine sostituiamo i valori mancanti con la stringa 'unknown' nelle celle delle colonne 'workclass' e 'occupation'.

3.3 Gestione colonna 'target'

Innanzitutto rinominiamo la colonna 'income' con 'target' che è più convenzionale, poi siccome nella colonna target i valori sono ' $> 50K.$ ', ' $> 50K$ ', ' $\leq 50k.$ ' e ' $\leq 50k$ ', accorpiamo i quattro valori in due rinominando ' $> 50K.$ ' in ' $> 50K$ ' e ' $\leq 50k.$ ' in ' $\leq 50k$ '. Dopodichè convertiamo i tipi della colonna target da stringa a booleana secondo il criterio deciso in precedenza. Infine osserviamo i valori della colonna target dopo aver eseguito questa piccola operazione di miglioramento.



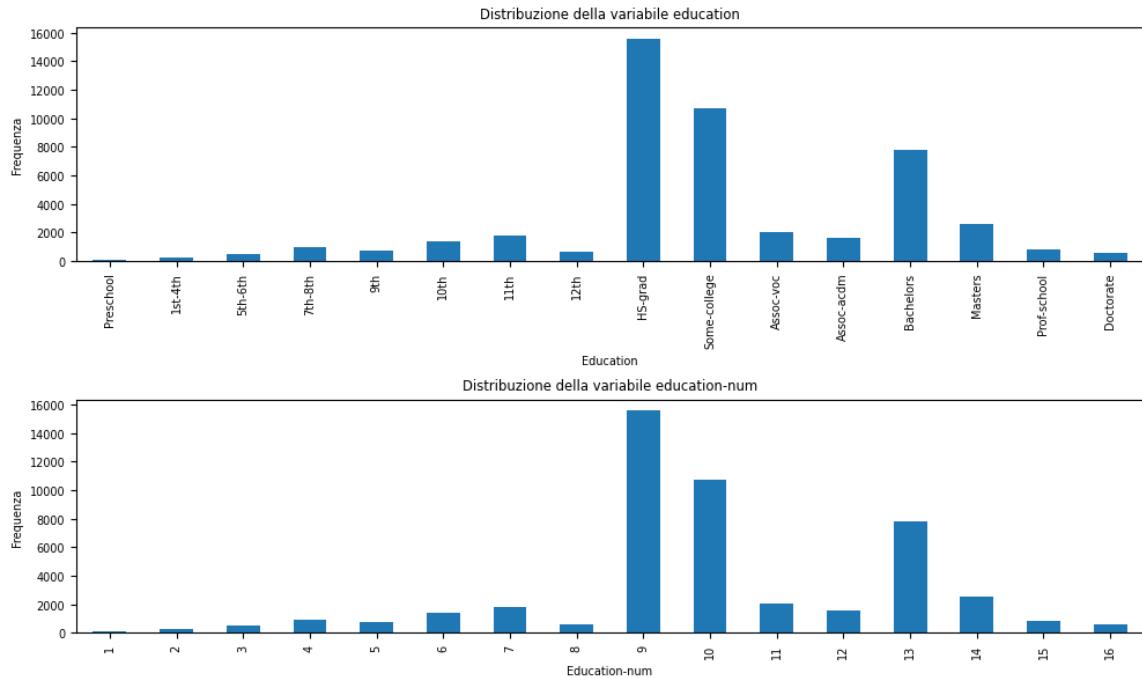
3.4 Codifica colonna 'sex'

Codifichiamo 'female' con 0 e 'male' con 1

3.5 Correlazione tra colonna education e education-num

Prima di procedere ad ottenere la correlazione tra education e education-num occorre stabilire il corretto ordine del livello di educazione. Se il contenuto di education-num è più intuitivo (nel senso che l'ordine è quello numerico), per il contenuto di education occorre informarsi sul sistema scolastico americano. L'ordine corretto dal livello minore a quello maggiore quindi è: Preschool, 1st-4th, 5th-6th, 7th-8th, 9th, 10th, 11th, 12th, HS-grad, Some-college, Assoc-voc, Assoc-acdm, Bachelors, Masters, Prof-school e infine Doctorate.

Osserviamo su un grafico la corrispondenza.



Trasformiamo in tipo numerico il contenuto della colonna `education` associando ad ogni livello di istruzione un numero da 1 a 16 come la colonna `education-num`.

Osserviamo la correlazione tra le due colonne: se è uguale a 1 significa che le due colonne sono identiche, pertanto potremo eliminare una delle due per evitare ridondanza tra i dati.

	education	education-num
education	1.000000	1.000000
education-num	1.000000	1.000000

Procediamo ad eliminare la colonna '`education`', mentre manteniamo '`education-num`' poichè il tipo numerico è più semplice da gestire.

3.6 Fusione colonne capital-gain e capital-loss

Il prossimo passo consiste nell'unire due colonne semanticamente simili, in una che sia in grado di racchiudere gli stessi concetti. In tal modo preserviamo informazione riducendo il dataset di una colonna.

3.7 Gestione '`fnlwgt`'

Final weight non è una feature normale, bensì un peso assegnato dall'US Census Bureau che indica quanto un record del dataset sia rappresentativo della popolazione. Pertanto non può essere trattata come una feature normale, ma contiene dati preziosi che possono essere usati in fase di addestramento dei modelli.

3.8 Outliers

Come abbiamo potuto osservare gli outliers sono presenti in misura considerevole. La gestione di questi ultimi non può essere liquidata ad una semplice eliminazione delle istanze 'eccezionali' in quanto se nella colonna 'age' abbiamo un paio di centinaia di età straordinarie, invece nella colonna 'hours-per-week' queste sono 13 migliaia. Eliminare tutte queste righe potrebbe comportare l'eliminazione di dati importanti.

Invece ciò che si farà sarà considerare l'uso di modelli robusti agli outliers.

4 Feature engineering

Vediamo adesso alcune pratiche di feature engineering. In ordine:

- codifichiamo con OneHotEncoder le feature categoriche
- standardizziamo le feature numeriche
- dividiamo in test e training set
- bilanciamo il dataset

4.1 Codifica delle feature categoriche

Per ognuna delle colonne non numeriche convertiamo la feature in formato One Hot. Il dataset non presenterà più le colonne non numeriche di partenza, bensì solo le loro versioni codificate. Questo comporterà un numero di colonne abbastanza elevato, pertanto sarà necessario provvedere a ridurre la dimensionalità.

4.2 Standardizzazione delle feature numeriche

Portiamo le feature numeriche ad una scala standard. Il processo viene applicato anche alla colonna 'fnlwgt', tuttavia nel range 0-1 in quanto verrà impiegata successivamente e potrebbe essere più conveniente standardizzarla in questo modo già adesso.

NON applichiamo invece questa standardizzazione alle colonne ottenute come risultato della codifica One Hot perché la loro media e varianza sono state già modificate in precedenza

4.3 Split in train e test

Siamo alla fase di splitting del dataset in train e test set.

4.4 Ricampionamento del training set

Siccome il dataset è sbilanciato occorre usare tecniche di ricampionamento al fine di ottenere pari rappresentanza per le due classi target.

Il training set originale si presenta con:

istanze appartenenti alla classe 0: 25518

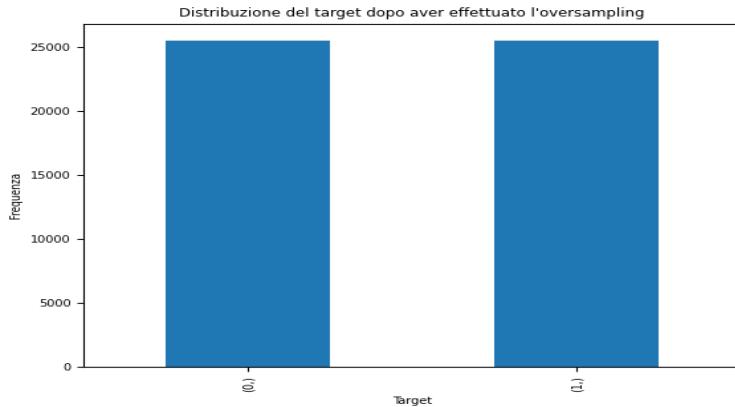
istanze appartenenti alla classe 1: 8038

Tutte le tecniche che seguono sono state applicate sul training set

4.4.1 SMOTE

Procederemo con la prima di ricampionamento che è una tecnica di oversampling chiamata 'SMOTE' da applicare sul training set. Quando usiamo questa tecnica rinunciamo al campo fnlwgt, poichè i valori generati sono sintetici e potrebbero essere mal rappresentativi della realtà.

L'applicazione di tale tecnica porta al seguente risultato:



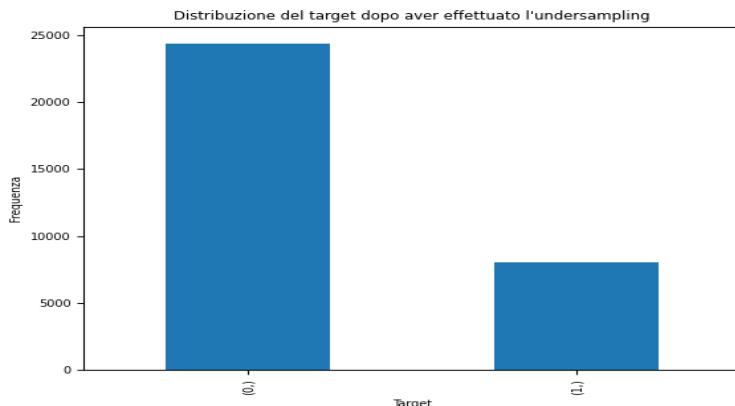
istanze appartenenti alla classe 0: 25518

istanze appartenenti alla classe 1: 25518

Ora il training set è bilanciato, però proviamo anche un'altra tecnica che potrebbe permetterci di non fare ricorso a dati sintetici, ovvero una tecnica di undersampling.

4.4.2 TomekLinks

Un altro metodo per bilanciare il dataset è attraverso l'undersampling: questo metodo elimina i campioni presenti in misura maggiore, andando a ridurre il numero di istanze.



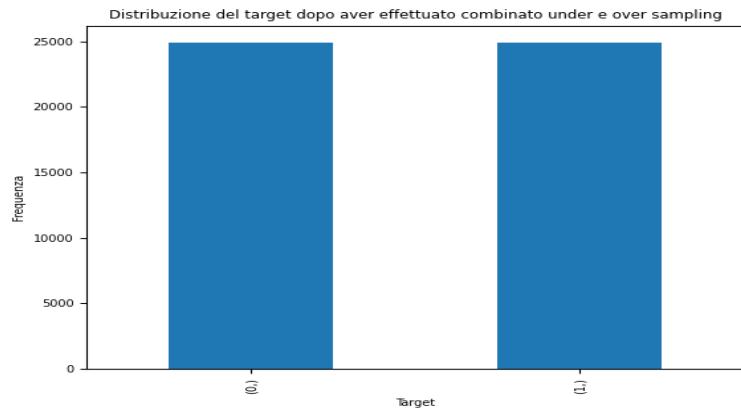
istanze appartenenti alla classe 0: 24393

istanze appartenenti alla classe 1: 8038

Questo metodo ha contribuito a modifiche del dataset, ma sicuramente non ha risolto il nostro problema, quindi procederemo con un'ultima tecnica che mette assieme le due appena viste.

4.4.3 SMOTE + TomekLinks

Infine proviamo una combinazione dei due metodi. Anche in questo caso dovremo rinunciare a fnlwgt



istanze appartenenti alla classe 0: 24907

istanze appartenenti alla classe 1: 24907

Essendo una combinazione delle due tecniche viste prima è plausibile che questo metodo raccolga i benefici di entrambi, quindi sarà questo il metodo scelto per proseguire con i modelli.

5 Esperimenti sui modelli

5.1 Alberi decisionali

Il primo modello scelto sono gli alberi decisionali. Gli alberi decisionali sono modelli molto robusti che presentano diversi vantaggi rispetto al nostro dataset:

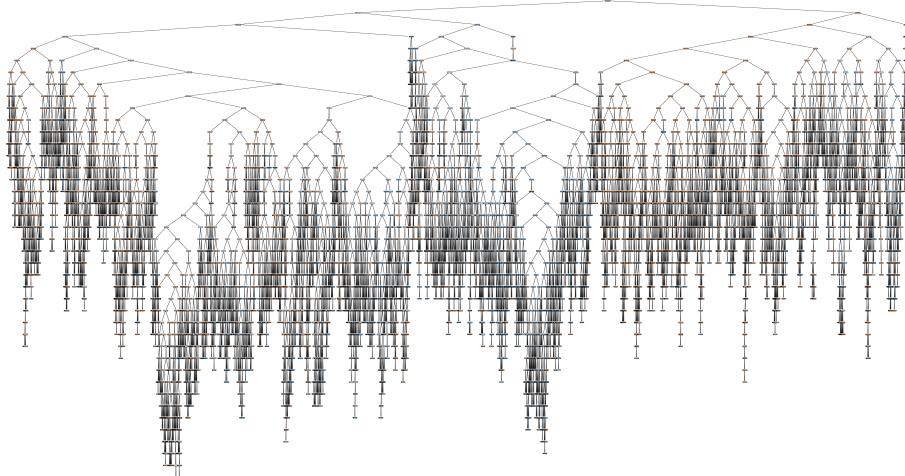
- in primis, sono di facile lettura e capire cosa determina il risultato del nostro problema può essere molto interessante da un punto di vista sociale
- la loro velocità di addestramento è notevole dato il nostro dataset di modeste ma neanche così piccole dimensioni

Permangono alcune criticità, ovvero il dataset è sbilanciato, ma a tal proposito svolgeremo degli esperimenti che coinvolgono tecniche di bilanciamento di quest'ultimo.

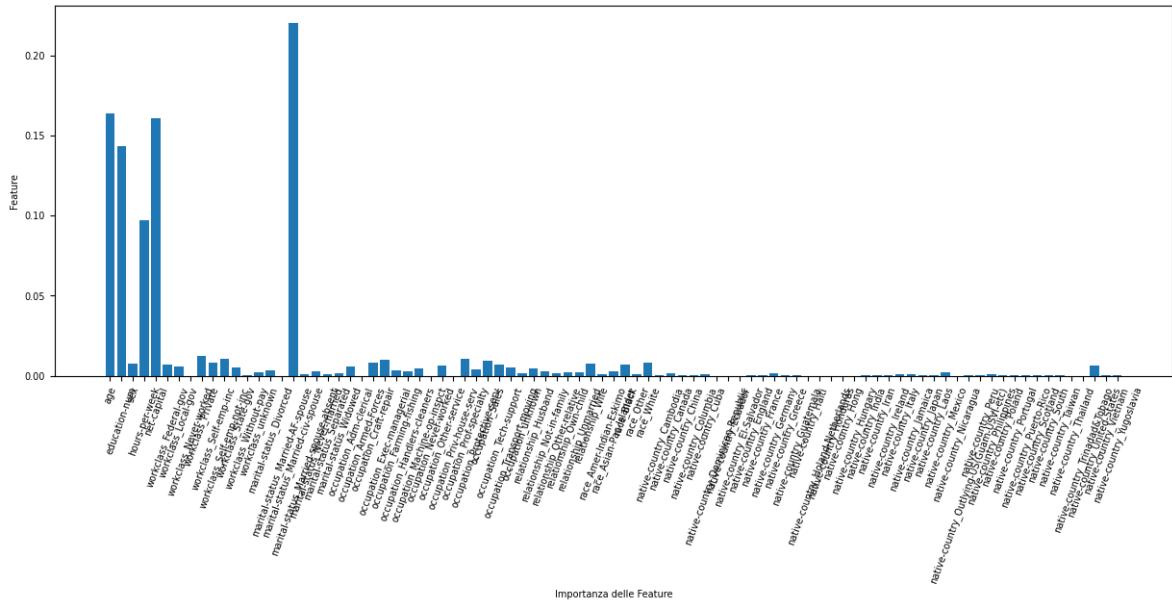
Però l'aggiunta di fnlwgt costituisce per il modello un aiuto dal quale apprendere in modo leggermente facilitato.

5.1.1 Esperimento 1

In questo caso useremo il dataset 'liscio', ovvero senza bilanciamento. Faremo uso del campo fnlwgt per dare un diverso peso alle istanze. Mostriamo la struttura dell'albero: osserviamo che sono presenti un grandissimo numero di livelli e ramificazioni, indice di un probabile overfitting.



Visualizziamo le feature in ordine di importanza

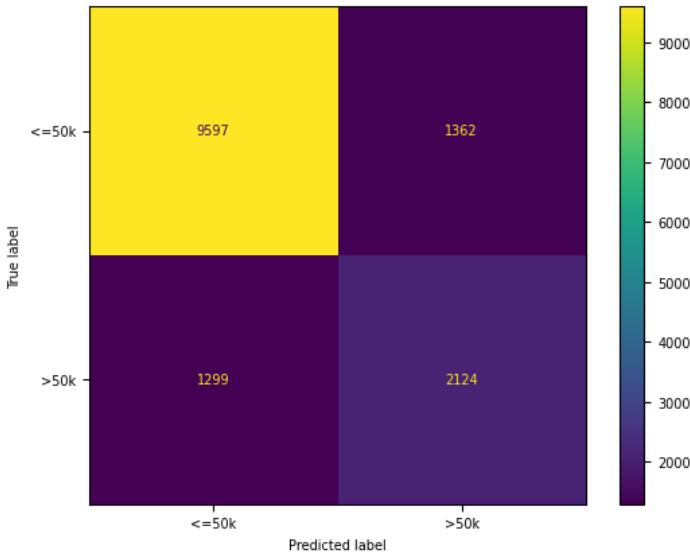


Le feature più importanti sono:

- marital-status-Married-civ-spouse: 0.220197
 - age: 0.163777
 - net-capital: 0.160829
 - education-num : 0.143627
 - hours-per-week: 0.097208

Questi numeri ci indicano un albero che è stato allenato eccessivamente. Infatti la accuracy sul training set supera di gran lunga quella sul test set. Inoltre la struttura è pesante: il numero di nodi e di livelli di profondità è altissimo, con la conseguenza che l'immagine è illegibile. Procederemo ad effettuare il pruning.

Studiamo le prime metriche come accuracy e confusion matrix.



Da quanto possiamo osservare, il modello è più bravo a riconoscere i casi appartenenti alla fascia di chi guadagna meno di 50k\$ l'anno. Questo risultato non sorprende, dato che nel dataset rappresenta la classe di maggioranza. Infatti assegna 1362 istanze alla classe '> 50K' che apparterebbero all'altra classe, mentre 1299 istanze predette come appartenenti alla classe ' $\leq 50k$ ' appartengono alla classe opposta.

Le istanze confuse sono in quantità simile, però poichè il dataset è sbilanciato questo significa che circa 1 istanza della classe di maggioranza su 10 viene male assegnata, mentre per la classe di minoranza questo è vero per più di 1 istanza su 3 circa.

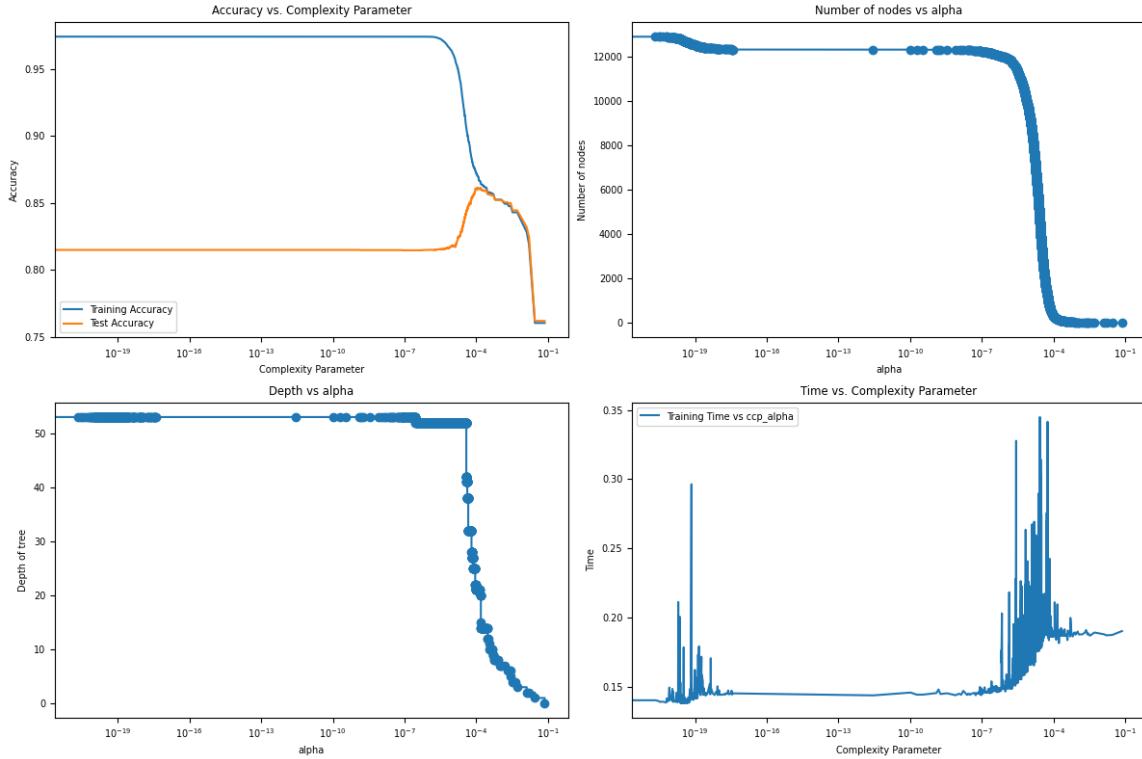
Questo vuol dire che il modello ha sviluppato capacità predittive distorte. Per finire osserviamo il report:

	precision	recall	f1-score	support
$\leq 50k$	0.88	0.88	0.88	10959
$> 50K$	0.61	0.62	0.61	3423
accuracy			0.81	14382
macro avg	0.75	0.75	0.75	14382
weighted avg	0.82	0.81	0.82	14382

Analizziamo i risultati:

- per questo primo esperimento non guarderemo l'accuracy per via dello sbilanciamento del dataset
- come commentato poc'anzi sotto alla confusion matrix, abbiamo dei valori di precision distorti
- poichè esistono sia 'falsi positivi' che 'falsi negativi', abbiamo una recall simile alla precision
- come risultato di media tra precision e recall, abbiamo un f1-score in linea con quanto detto poco fa
- notiamo che queste metriche cambiano quando passiamo dalla valutazione di tipo 'macro' alla valutazione di tipo 'weighted', questo sempre in virtù dello sbilanciamento delle classi, dando una valutazione più rappresentativa delle prestazioni complessive del modello

Ora si procede con la raccolta degli alpha per poi eseguire il pruning dell'albero. Mostriamo cosa succede alle statistiche degli alberi al variare di alpha:



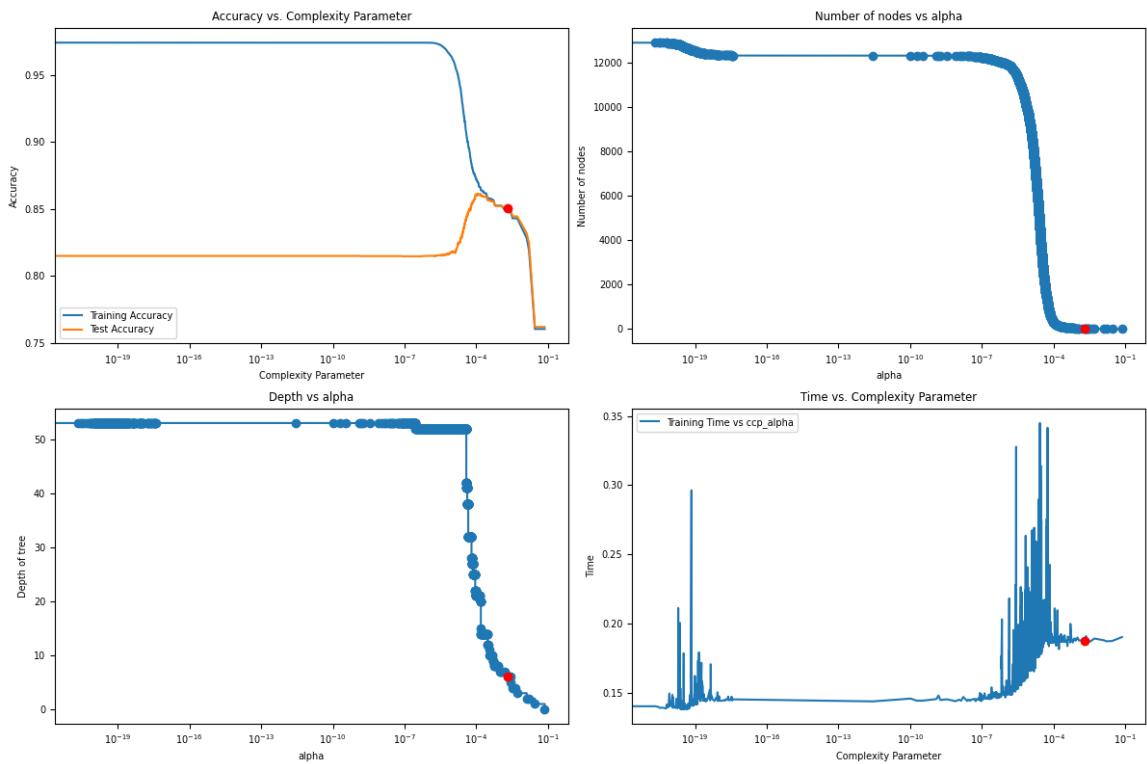
Misuriamo come cambiano i valori di accuracy su training set e test set al variare di alfa: da quello che possiamo vedere, per gli alfa più bassi abbiamo livelli di test accuracy che discostano abbastanza dalla training accuracy, indice del fatto che continuiamo ad avere modelli troppo ben allenati, tuttavia quando alfa raggiunge valori un po' più alti si comincia ad avere modelli promettenti.

Osserviamo come il crescere dell'alfa impatti sulla struttura degli alberi: il numero di nodi cala mano a mano e così anche il numero di livelli. Imponendo un fattore limitativo di crescita agli alberi, andremo ad ottenere strutture meno ramificate e profonde, che pertanto impareranno in misura più limitata dal training set, riducendo il rischio di overfitting.

Un altro aspetto interessante da osservare è il tempo di addestramento: esso tende ad aumentare all'aumentare dell'alfa. Com'è possibile che un albero più piccoli impieghi più tempo ad essere addestrato? Poco chiari anche i picchi.

Ora che abbiamo fatto queste analisi andremo a selezionare il modello con ccp alpha che più ci soddisfa. In particolare prima restringiamo alla zona del grafico per cui la test accuracy supera la training accuracy, dopodiché in questo sotto-intervallo prendiamo l'ultimo modello con l'accuracy più alta. In questo modo recuperiamo un albero con il numero di nodi più basso possibile, con un numero basso di livello di profondità con l'accuracy più alta possibile, con la certezza di aver prevenuto overfitting e con un tempo di addestramento competitivo.

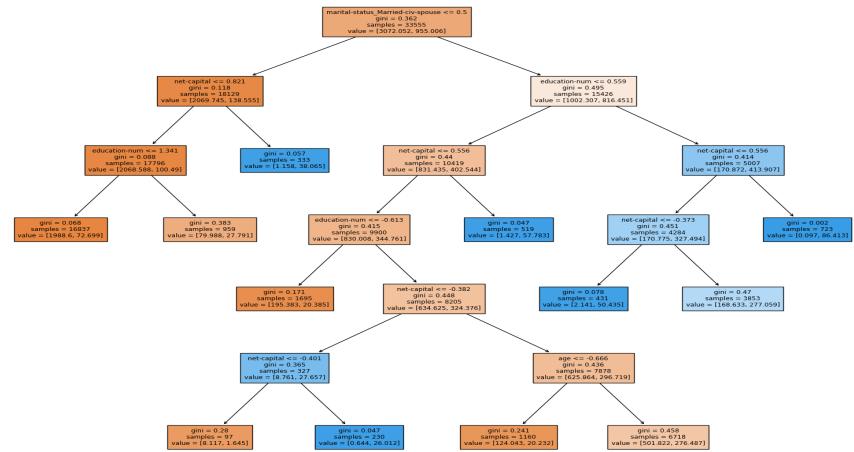
Studiamo il prototipo più nel dettaglio:



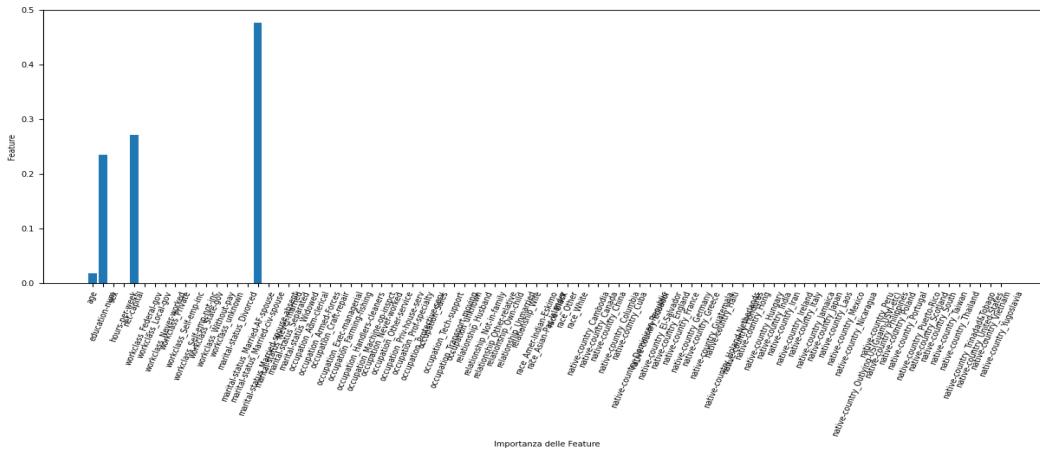
Questi sono i dettagli:

- Training accuracy of 3164' model : 0.8494159017761355
- Test accuracy of 3164' model: 0.8505771102767348
- Cpp-alpha of 3164' model: 0.0020348904888528724
- Number of nodes of 3164' model: 23
- Depth of 3164' model: 6
- Time took by training of 3164' model: 0.19736599922180176

Di seguito possiamo osservare l'albero risultante.



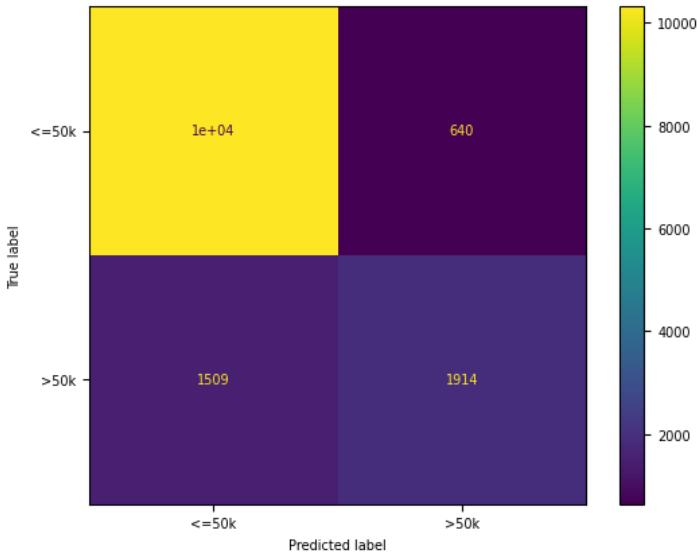
Quanto ottenuto è di dimensioni ridotte ed è decisamente più leggibile. Abbiamo ottenuto lo scopo di questo esperimento, ovvero ottenere un risultato facilmente interpretabile, riuscendo inoltre a non perdere accuratezza. Notiamo che le feature usate maggiormente per effettuare lo splitting sono lo stato civile, net-capital ed education-num, come mostrato anche nel grafico sottostante.



Le feature più importanti in questo caso sono:

- marital-status-Married-civ-spouse: 0.476448
 - net-capital: 0.270670
 - education-num: 0.234856
 - age: 0.018025
 - native-country-France: 0.000000

Notiamo dei cambiamenti interessanti: ora l'albero da un peso molto maggiore alla feature marital-status-married-civ-spouse, è leggermente più importante net-capital, mentre age è diventata una feature poco considerata. Infine la quinta feature è quinta per ordine di importanza, ma è approssimabile a zero, questo significa che il modello ora usa meno feature per prendere le sue decisioni. Osserviamo la confusion matrix:



Rispetto all'albero precedentemente questo albero pare essere più in grado di classificare correttamente le istanze della classe di maggioranza a discapito però della sua capacità di valutare le istanze della classe minoranza. Come conseguenza abbiamo una recall più sbilanciata in favore della classe di maggioranza. L'accuracy è leggermente migliorata, ma anche qui bisogna ricordarsi che abbiamo un dataset sbilanciato.

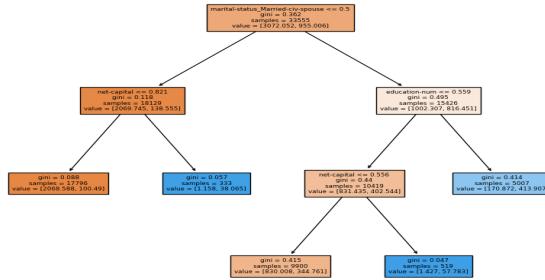
Osserviamo ora il report:

	precision	recall	f1-score	support
$\leq 50k$	0.87	0.94	0.91	10959
$> 50K$	0.75	0.56	0.64	3423
accuracy			0.85	14382
macro avg	0.81	0.75	0.77	14382
weighted avg	0.84	0.85	0.84	14382

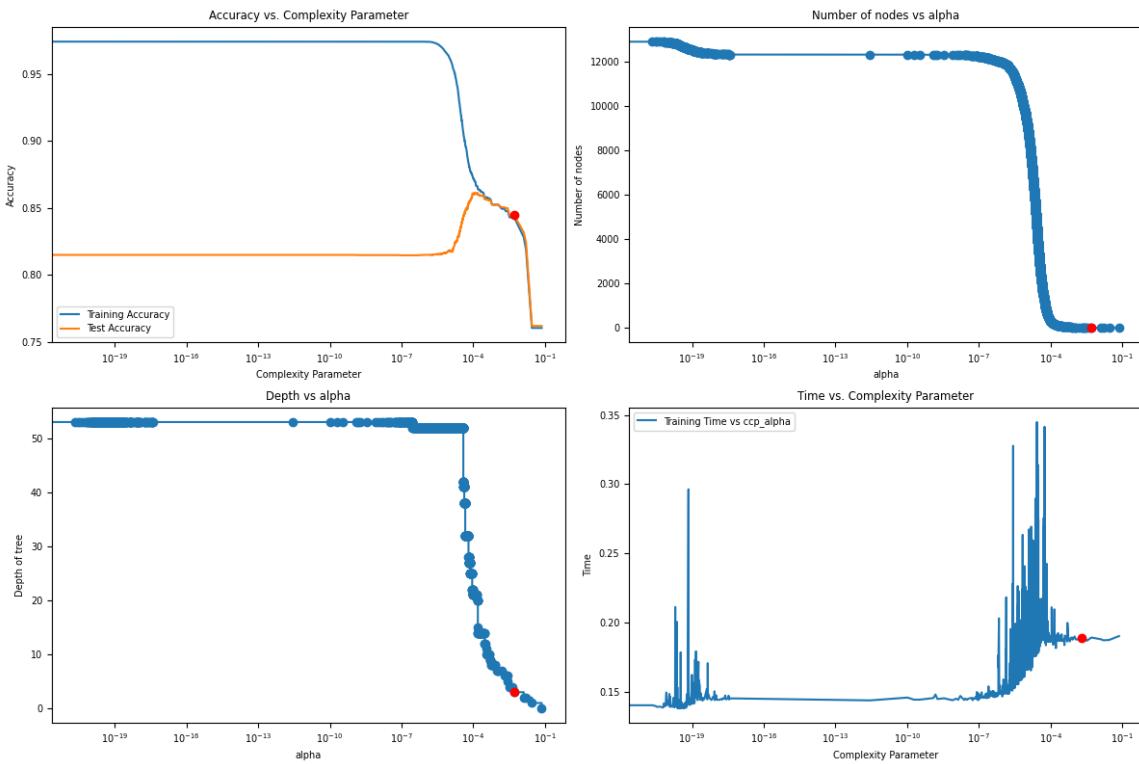
Analizziamo i risultati:

- come prima, ignoriamo l'accuracy
- è cambiata la precision: ora il modello è in grado di prevedere molto bene le istanze della classe di maggioranza e come conseguenza migliora anche la precision della classe di minoranza
- è cambiata la recall diventando più sbilanciata, questo è dato dal fatto che ora molte istanze della classe di minoranza vengono classificate come appartenenti alla classe di maggioranza
- come conseguenza anche l'f1 score è più sbilanciato
- come prima, la metrica globale weighted, tenendo conto dello sbilanciamento delle classi è più alta della metrica macro

Ancora più piccolo Poichè l'obiettivo di questo esperimento è quello di ottenere un albero leggibile, decidiamo di voler cercare un albero ancora più semplice anche a discapito degli altri aspetti. Come vedremo comunque manteremo un'accuracy alta, quindi l'albero resta affidabile.



Per farlo, scelgo 'a mano' dal file di testo generato un modello che presumibilmente farà al caso nostro. Non è possibile scegliere l'albero più piccolo in assoluto perchè sarebbe quello in cui tale albero degenera in un nodo singolo, perciò vorrei un via di mezzo che presumibilmente sarà attorno 3171, che come vedremo poi ci soddisferà.

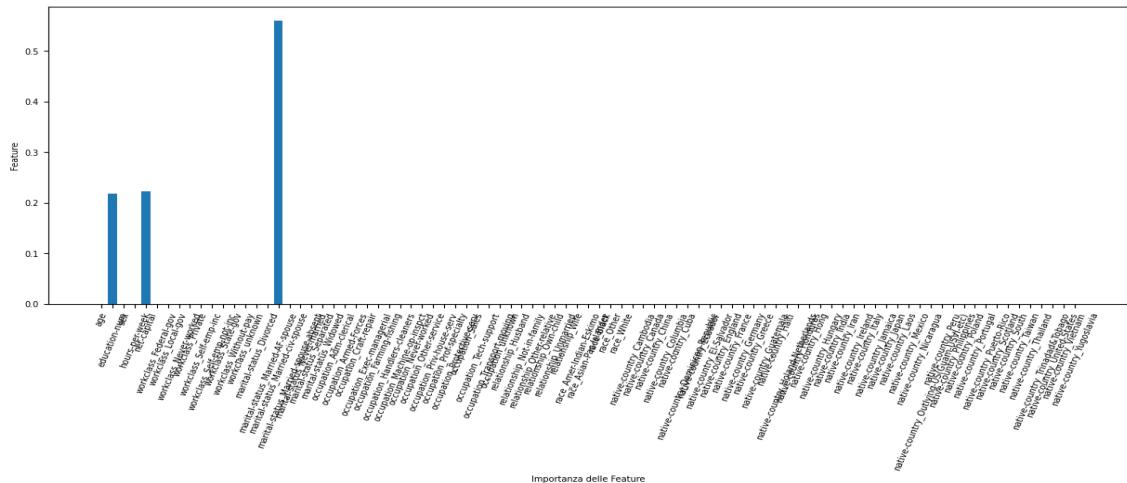


- Training accuracy of 3171' model : 0.8430385028012874
- Test accuracy of 3171' model: 0.8445278820748158
- Cpp alpha of 3171' model: 0.00519810783445844
- Number of nodes of 3171' model: 9
- Depth of 3171' model: 3

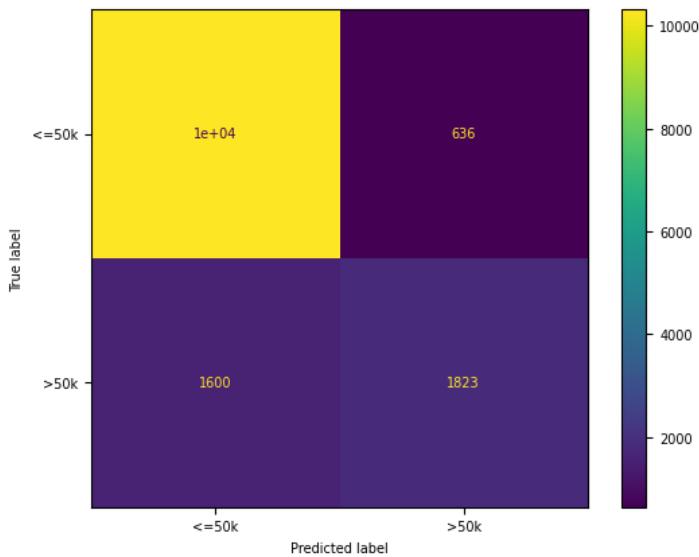
- Time took by training of ' model: 0.19095516204833984

Importanza delle feature:

- marital-status-Married-civ-spouse: 0.559853
- net-capital: 0.222680
- education-num: 0.217467
- age: 0.000000
- native-country-France: 0.000000



Si riconferma il trend mostrato in precedenza: le feature con maggiore importanza sono lo stato civile, net-capital e education-num. Quello che succede continuando a rimpicciolare l'albero è che viene dato sempre maggior rilievo alle feature che già avevano un'importanza maggioritaria a dispetto di quelle che invece erano secondarie.



L'albero è più semplice e più approssimativo nelle sue valutazioni: se le istanze appartenenti alla classi di maggioranza registrano gli stessi numeri dell'albero precedente, è anche vero che invece le istanze appartenenti alla classi di minoranza invece sono molto più mal classificate di prima, ora più della metà delle istanze della classe di minoranza vengono erroneamente classificate nel modo sbagliato.

	precision	recall	f1-score	support
$\leq 50k$	0.87	0.94	0.90	10959
$> 50K$	0.74	0.53	0.62	3423
accuracy			0.84	14382
macro avg	0.80	0.74	0.76	14382
weighted avg	0.84	0.84	0.84	14382

Si amplia la forbice tra le due classi: tutte le metriche hanno valori simili riferendosi alla classe di maggioranza e valori più bassi riferendosi alla classe di minoranza. Anche le metriche globali di weighted e macro registrano valori più bassi.

Conclusioni: pare che un modello più semplice diventi più approssimativo, cioè tenda a classificare più frequentemente nella classe di maggioranza tutte le istanze. Come conseguenza continuiamo ad avere un'accuracy discreta, ma questo non vuol dire che il modello sia allenato propriamente.

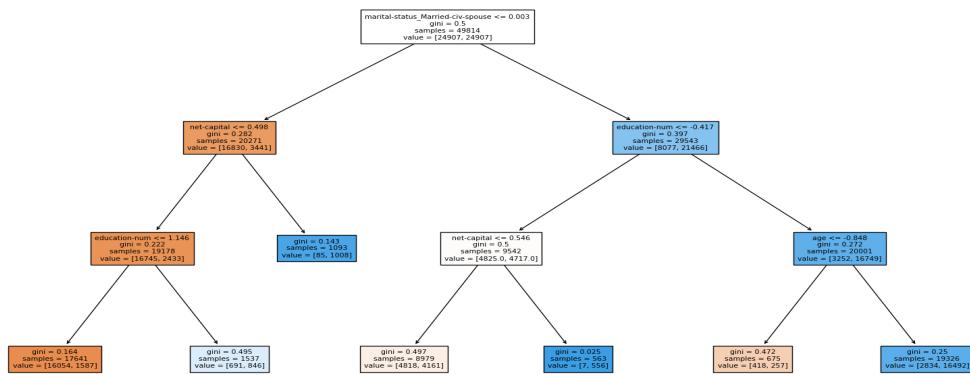
5.1.2 Esperimento 2

Da questo punto in poi viene presentata solo la versione finale dei modelli degli esperimenti a fine di brevità.

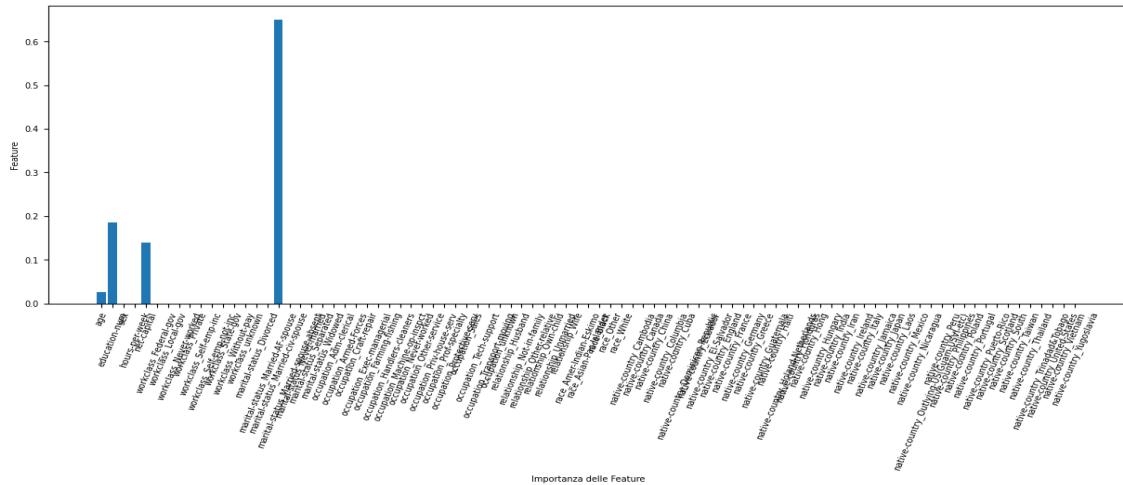
In questo secondo caso esperimento andremo ad allenare nuovamente degli alberi di decisioni, però su un dataset diverso, ovvero quello ottenuto tramite la tecnica di oversampling SMOTE combinato con la tecnica di undersampling TomekLinks. Non useremo tuttavia fnlwgt per i motivi spiegati in precedenza.

L'obiettivo di questo esperimento è verificare l'impatto di un dataset bilanciato sull'apprendimento dei modelli.

Visualizziamo l'albero:

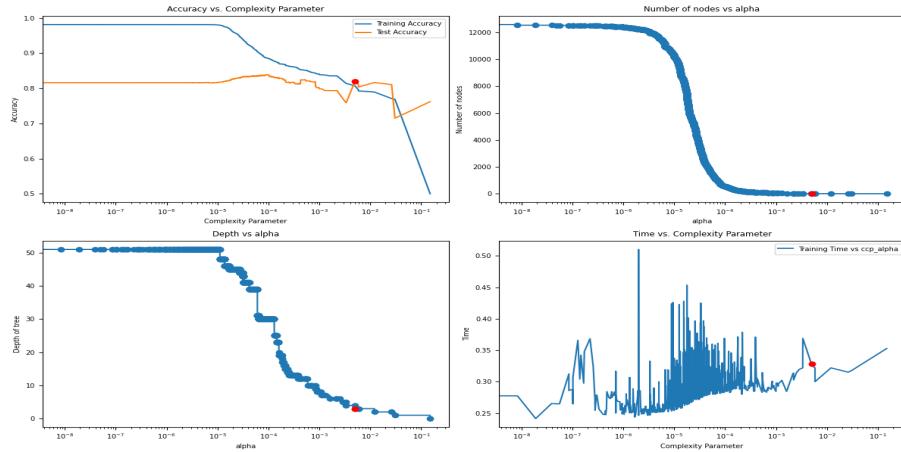


Osserviamo le feature per importanza:



- marital-status-Married-civ-spouse: 0.650207
- education-num: 0.184899
- net-capital: 0.139485
- age: 0.025410
- native-country-France 0.000000

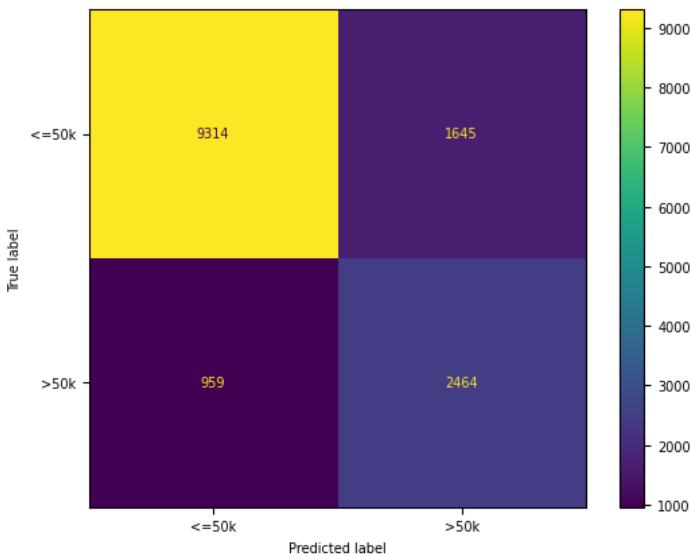
Studiamo i dettagli di questo albero:



- Training accuracy of 2903' model : 0.8068414501947244
- Test accuracy of 2903' model: 0.8189403420942846
- Cpp-alpha of 2903' model: 0.0050121831599254235
- Number of nodes of 2903' model: 13
- Depth of 2903' model: 3
- Time took by training of 2903' model: 0.2833702564239502

Possiamo dire che molte caratteristiche dell'albero siano molto simili al secondo prototipo del primo esperimento.

Ora osserviamo la confusion matrix:



Rispetto al secondo albero del primo esperimento abbiamo dei cambiamenti: ora le classificazioni errate si sono spostate dall'essere per lo più nella classe di minoranza ad essere per lo più nella classe di maggioranza. Come conseguenza abbiamo le recall più equilibrata, tuttavia adesso le precision sono sbilanciate per entrambe le classi.

	precision	recall	f1-score	support
$\leq 50k$	0.91	0.85	0.88	10959
$> 50K$	0.60	0.72	0.65	3423
accuracy			0.82	14382
macro avg	0.75	0.78	0.77	14382
weighted avg	0.83	0.82	0.82	14382

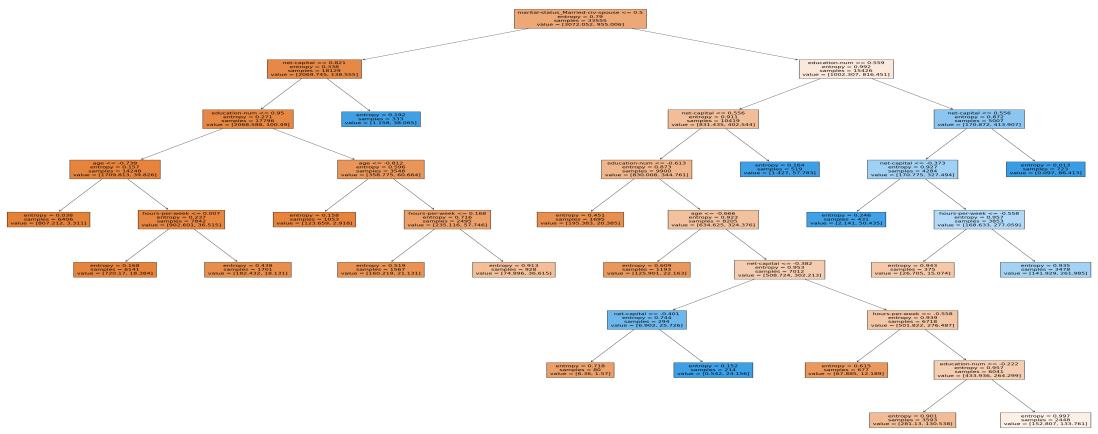
Rispetto a prima abbiamo dei cambiamenti interessanti: gli errori che commessi dall'albero ora sono più proporzionati nelle due classi. È più alta la precision per la classe di maggioranza, ma è più bassa per quella di minoranza. Questo si traduce inoltre in una recall più equilibrata. Questo modello classifica erroneamente circa 1 istanza su 6 tra quelle appartenenti alla classe di maggioranza e meno di 1 su 3 circa tra quelle appartenenti alla classe di minoranza. L'accuracy, che ora è un dato a cui possiamo guardare con più affidabilità, vale 0.82.

Tutto sommato non c'è stato un forte miglioramento, tuttavia il modello offre prestazioni più equilibrate tra le classi, frutto dell'avere usato un dataset bilanciato.

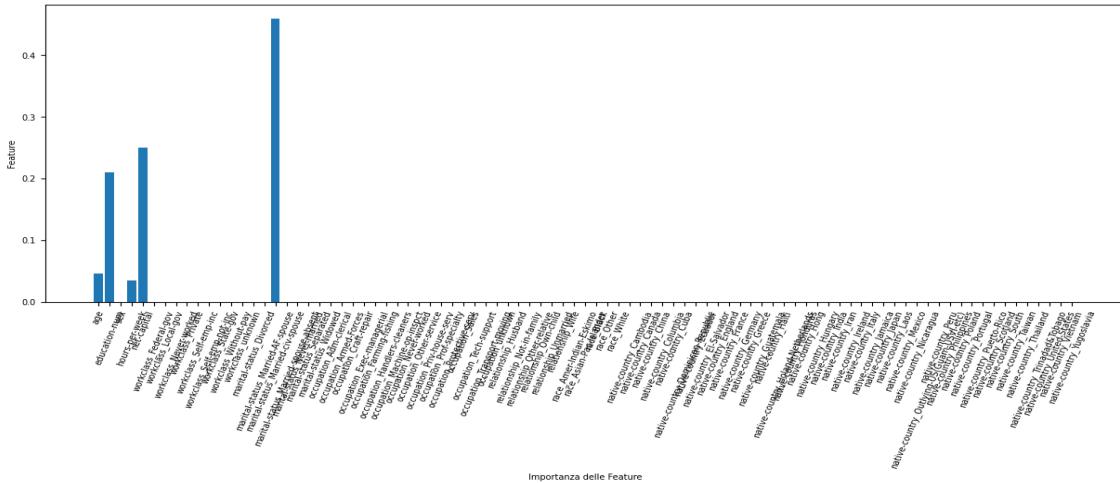
5.1.3 Esperimento 3

In questo esperimento giochiamo con il criterio di splitting, che finora è stato Gini. Ora usiamo Entropy.

Recuperiamo il secondo albero dell'esperimento 1 e ripetiamo l'esperimento con lo stesso dataset cambiando solo criterio

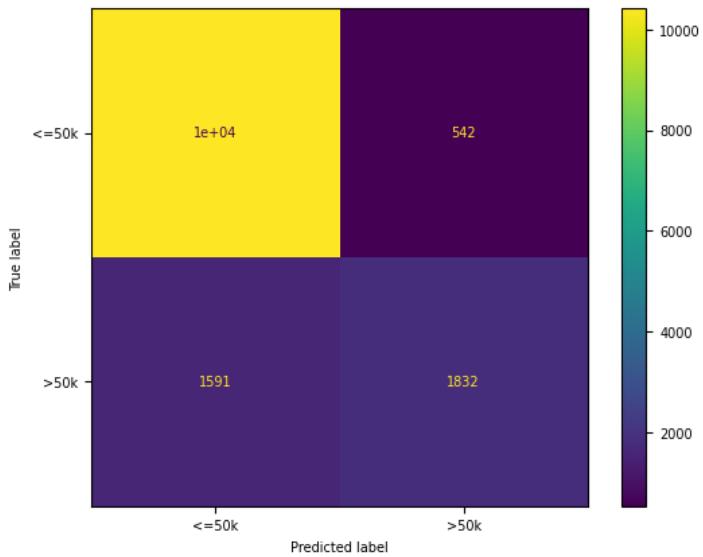


Anche le feature più importanti non sono più comprensibili a noi:



- marital-status-Married-civ-spouse: 0.459508
- net-capital: 0.249924
- education-num: 0.210334
- age: 0.046066
- hours-per-week: 0.034168

In questo esperimento, le feature sono più pesate in modo più equo.
Passiamo a osservare la matrice di correlazione:



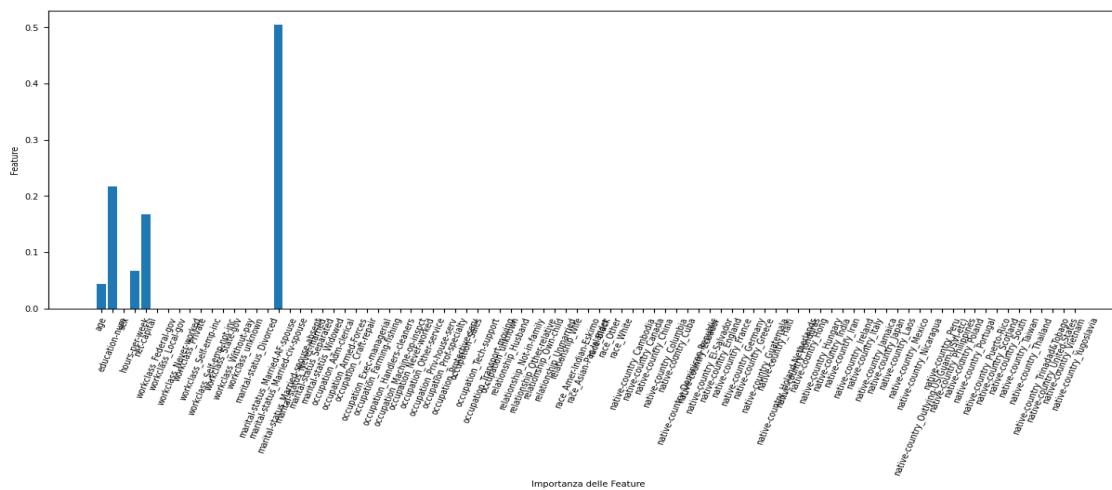
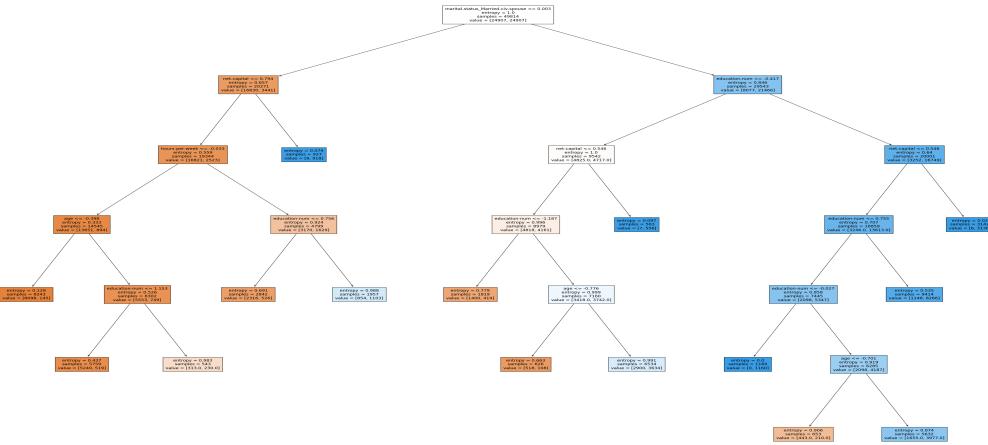
Questa matrice presenta fortissime somiglianze con il secondo modello indotto nel primo esperimento. Studiamo il report.

	precision	recall	f1-score	support
$\leq 50k$	0.87	0.95	0.91	10959
$> 50K$	0.77	0.54	0.63	342
accuracy			0.85	14382
macro avg	0.82	0.74	0.77	14382
weighted avg	0.84	0.85	0.84	14382

Anche il report è quasi identico. Questo vuol dire che il criterio di splitting tra Gini e Entropy non ha grossa influenza per questo problema. L'unica differenza è nella struttura dell'albero, che ora è più grosso.

5.1.4 Esperimento 4

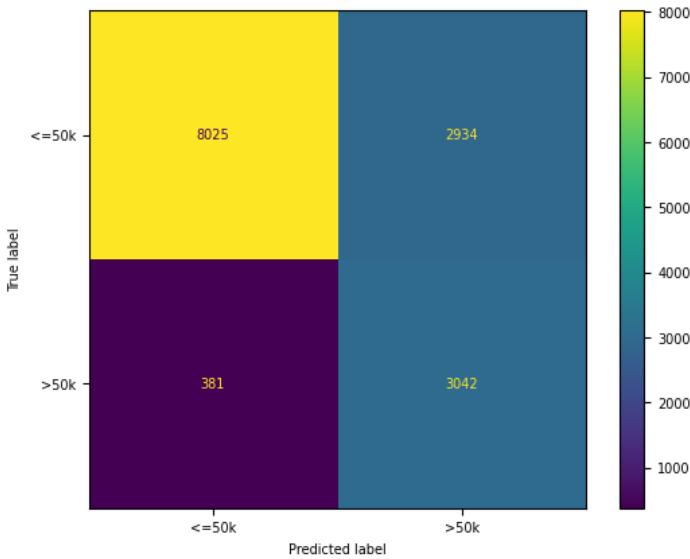
Ripetiamo infine lo stesso esperimento dell'esperimento 2, ma anche qui cambiamo il criterio di splitting e usiamo Entropy. Come prima cosa osserviamo le dimensioni dell'albero:



- marital-status-Married-civ-spouse: 0.504672
- education-num: 0.216840
- net-capital: 0.167616
- hours-per-week: 0.066900
- age: 0.043973

Qui la tendenza a dare maggiore rilievo alle feature già importanti è molto meno evidente. Non c'è un cambiamento molto evidente da questo punto di vista.

Osserviamo la confusion matrix:



Rispetto all'esperimento 3, il modello ora confonde in misura maggiore le istanze appartenenti alla classe di maggioranza rispetto a quelle della classe di minoranza. Studiamo il report:

	precision	recall	f1-score	support
$\leq 50k$	0.95	0.73	0.83	10959
$> 50K$	0.51	0.89	0.65	3423
accuracy			0.77	14382
macro avg	0.73	0.81	0.74	14382
weighted avg	0.85	0.77	0.79	14382

Come conseguenza possiamo osservare una precision maggiore nella classe di maggioranza, ma quasi invariata in quella di minoranza. Invece è migliorata la recall della classe di minoranza, ma non è salita la recall della classe di maggioranza.

5.1.5 Conclusioni sugli esperimenti sui Decision Tree

I pattern più interessanti da studiare sono quelli tra l'esperimento 1 e l'esperimento 2 e tra l'esperimento 3 e l'esperimento 4.

L'impatto che ha il bilanciamento del dataset non è indifferente, anzi è decisamente importante. Si è notata una differenza molto più notevole negli esperimenti in cui cambia il dataset rispetto a quelle in cui cambia il criterio di splitting. Tuttavia il cambiamento tra l'esperimento 3 e l'esperimento 4 non ricalca allo stesso modo le differenze tra l'esperimento 1 e l'esperimento 2

5.2 Support Vector Machines

La terza famiglia di classificatori che studiamo è quella delle Support Vector Machines.

- capacità di resistenza agli outliers

- capacità di lavorare in spazi ad altà dimensionalità

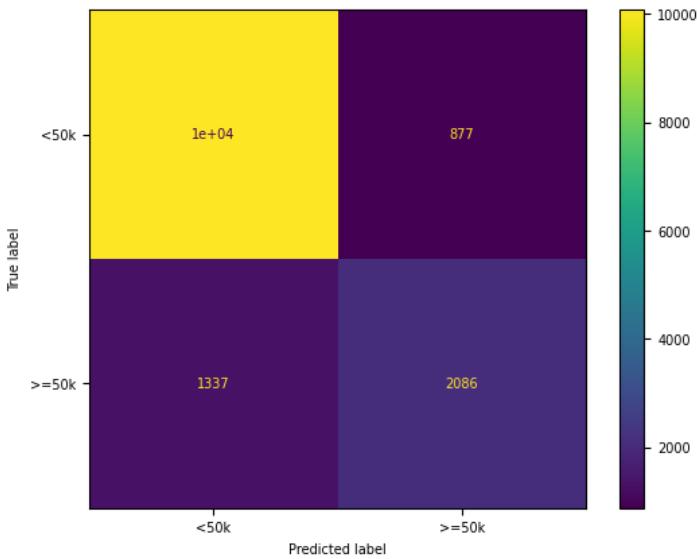
Purtroppo essendo questi classificatori costosi da allenare e il nostro dataset non particolarmente ridotto, il tempo di addestramento non sarà modestissimo.

5.2.1 Esperimento 1

Nel primo esperimento usiamo una Support Vector Machine con kernel lineare e il parametro di regolarizzazione $C=1$. Inoltre suggeriamo un sample weight al modello grazie al campo fnlwgt. Il dataset usato è quello normale. In questo esperimento si vuole solo osservare come si adatta il modello al problema in condizioni "normali".

Il tempo di addestramento è di circa 30 secondi.

Questi sono i risultati:



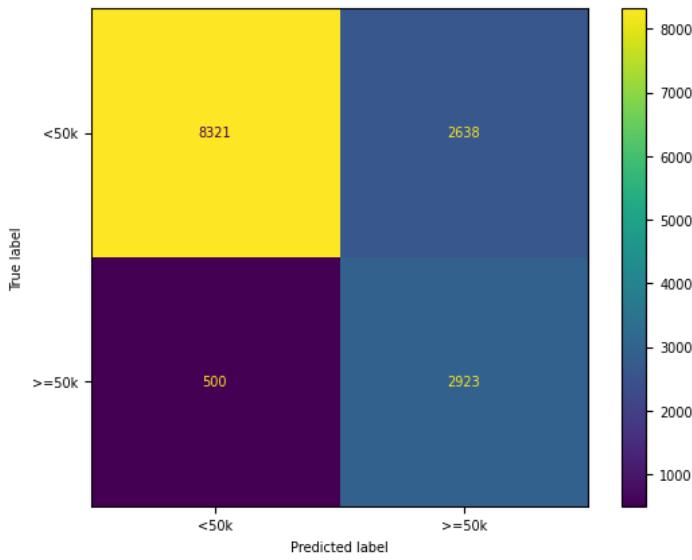
	precision	recall	f1-score	support
$\leq 50k$	0.88	0.92	0.90	10959
$> 50K$	0.71	0.61	0.65	3423
accuracy			0.85	14382
macro avg	0.79	0.76	0.78	14382
weighted avg	0.84	0.85	0.84	14382

I risultati ricordano molto quelli trovati alla fine del primo esperimento con i Decision Tree. Da quanto possiamo osservare si ripropongono gli stessi problemi, ovvero gli errori commessi dal modello riguardano in misura maggiore le istanze appartenenti alla classe di minoranza, rispetto a quelle appartenenti alla classe di maggioranza.

5.2.2 Esperimento 2

In questo secondo esperimento usiamo nuovamente lo stesso kernel, mentre cambiamo dataset e usiamo quello su cui abbiamo applicato le tecniche di ricampionamento. Sempre con un parametro

di regolarizzazione C=1 vediamo come si comporta il modello.
Il tempo di addestramento è di circa 1 minuto e 30 secondi.



In questo secondo esperimento usiamo nuovamente lo stesso kernel, mentre cambiamo dataset e usiamo quello su cui abbiamo applicato le tecniche di ricampionamento. Sempre con un parametro di regolarizzazione C=1 vediamo come si comporta il modello.

	precision	recall	f1-score	support
$\leq 50k$	0.94	0.76	0.84	10959
$> 50K$	0.53	0.85	0.65	3423
accuracy			0.78	14382
macro avg	0.73	0.81	0.75	14382
weighted avg	0.84	0.78	0.80	14382

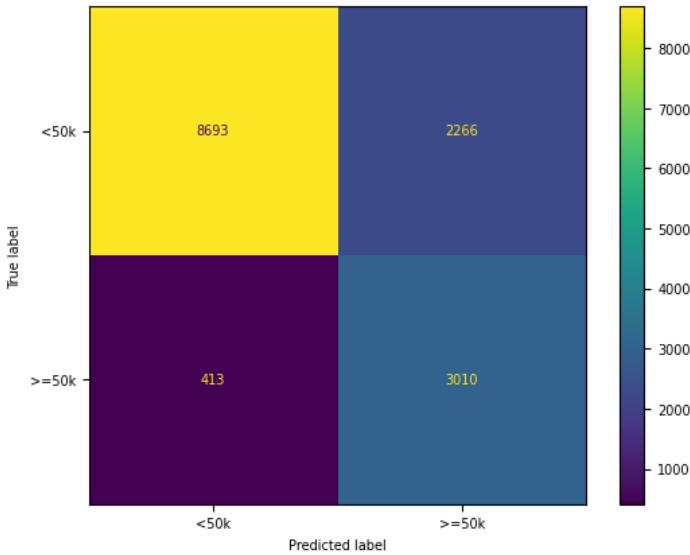
Ora la situazione è speculare a quanto trovato alla fine del secondo esperimento sui Decision Tree, ovvero gli errori di classificazione sono più equamente distribuiti tra le classi e seguono un andamento proporzionale. La differenza rispetto a quell'esperimento è che in questo caso ad essere favorita è la classe di minoranza.

5.2.3 Esperimento 3

Per finire usiamo il kernel radiale, ancora il dataset normale e ancora il parametro di regolarizzazione C=1.

Questo è solo uno dei tanti possibili esperimenti che è possibile condurre con le Support Vector Machine, ma per questioni di brevità non verranno riportati tutti gli esperimenti condotti.

Il tempo di addestramento è di circa 1 minuto e 30 secondi.



	precision	recall	f1-score	support
$\leq 50k$	0.95	0.79	0.87	10959
$> 50K$	0.57	0.88	0.69	3423
accuracy			0.81	14382
macro avg	0.76	0.84	0.78	14382
weighted avg	0.86	0.81	0.82	14382

Questo esperimento è molto in linea con il precedente. Ciò che possiamo concludere è che la scelta del kernel ha un impatto sulle capacità predittive del modello non trascurabili, infatti le performance sono salite. Oltre a questo non c'è niente di particolare da evidenziare.

5.2.4 Conclusioni sugli esperimenti su svm

I modelli, ottenuti con svariati esperimenti, dimostrano una abilità a lavorare con il problema non secondaria agli alberi.

Osservando i risultati del primo esperimento possiamo immediatamente concludere che si ripresentano le stesse problematiche di alcuni esperimenti visti con i Decision Tree, ovvero un'alta accuracy, ma su cui non poter affidamento e valori di recall e precision variabili. Dalla confusion matrix del primo esperimento deduciamo nuovamente che il modello apprende bene quando si tratta della classe di maggioranza, ma non apprende altrettanto bene se si tratta della classe di minoranza.

Dal secondo esperimento invece non possiamo non ricondurci ai risultati degli esperimenti sui dei Decision Tree condotti con dataset bilanciati. Esattamente come con gli alberi, il modello diventa più equilibrato nel riconoscere le classi, cioè classifica in misura corretta ed errata istanze di entrambe le classi in misura simile. Anche per questo osserviamo nell'esperimento 2 rispetto all'1 recall più simili e precision molto diverse.

Sulle orme del secondo esperimento, il terzo ed ultimo condotto sulle SVM che differisce solo nell'utilizzo del kernel porta risultati estremamente simili, ma leggermente migliori. Il kernel usato in questo caso è quello radiale e riesce a migliorare leggermente le performance sotto ogni aspetto.

6 Tecniche di validazione

Per procedere con la validazione dei modelli faremo uso della tecnica stratified k-fold. Avendo a che fare con un dataset sbilanciato è opportuno assicurarsi un'equa distribuzione tra le classi. Con la tecnica stratified k-fold riusciamo a preservare questa informazione ottenendo un risultato più affidabile.

6.1 Tecniche di validazione sui Decision Tree

Grazie a delle funzioni appositamente definite, proiedo con il generare intervalli di confidenza ottenuti su metriche quali accuracy, precision, recall, f1-score ecc e a crearcì un simil report come quelli visti finora, ma costruiti appunto su intervalli anzichè valori precisi.

6.1.1 Stratified k-fold cross validation esperimento 1

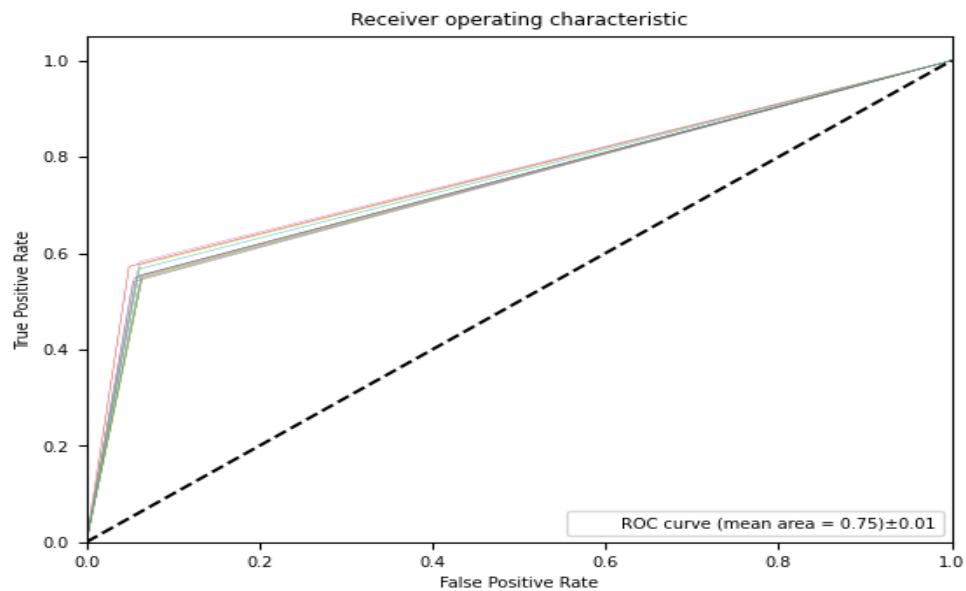
I risultati sono i seguenti:

	precision	recall	f1-score	support
$\leq 50k$	(0.868, 0.875)	(0.938, 0.943)	(0.903, 0.907)	(3647.35, 3648.05)
$> 50K$	(0.739, 0.756)	(0.546, 0.575)	(0.629, 0.651)	(1145.87, 1146.33)
macro avg	(0.805, 0.815)	(0.744, 0.758)	(0.766, 0.779)	(4793.5, 4794.10)
weighted avg	(0.838, 0.846)	(0.846, 0.853)	(0.838, 0.846)	(4793.5, 4794.10)

Inoltre l'intervallo dell'accuracy vale: (0.846, 0.853)

e l'intervallo dei tempi di addestramento: (0.2666, 0.3269).

Infine sono stati raccolti i dati relativi alla curva ROC e all'AUC score. Questi dati sono stati aggregati al fine di verificare comportamenti uniformi nei classificatori dei vari esperimenti:



Ora che abbiamo delle stime affidabili e dato uno sguardo al grafico possiamo concludere che i risultati restano fortemente influenzati dallo sbilanciamento tra le classi.

I modelli si riconfermano molto abili a predire le istanze appartenenti alla classe di maggioranza, ma non altrettanto a predire le istanze della classe di minoranza.

6.1.2 Stratified k-fold cross validation esperimento 2

Procediamo con la validazione dell'esperimento 2

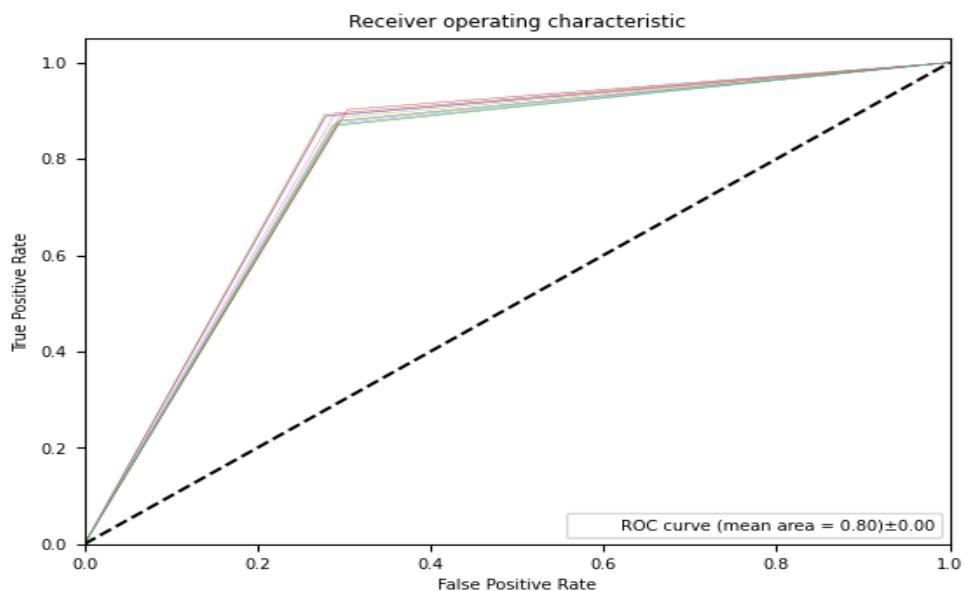
	precision	recall	f1-score	support
$\leq 50k$	(0.935, 0.957)	(0.693, 0.758)	(0.805, 0.835)	(3647.35, 3648.05)
$> 50K$	(0.475, 0.527)	(0.828, 0.904)	(0.625, 0.64)	(1145.87, 1146.33)
macro avg	(0.715, 0.732)	(0.79, 0.801)	(0.715, 0.737)	(4793.5, 4794.10)
weighted avg	(0.836, 0.843)	(0.743, 0.775)	(0.762, 0.788)	(4793.5, 4794.10)

Inoltre l'intervallo dell'accuracy è: (0.743, 0.775)

e l'intervallo dei tempi di addestramento: (0.4244, 0.4436).

Questi dati sono abbastanza sorprendenti, infatti contraddicono l'esperimento 2 e senza apparente motivo si discostano sparecchio dai risultati sperimentali. Soprattutto la precision della classe di minoranza è molto più bassa. I modelli così validati paiono essere completamente diversi dal modello dell'esperimento 2.

Osserviamo la curva ROC:



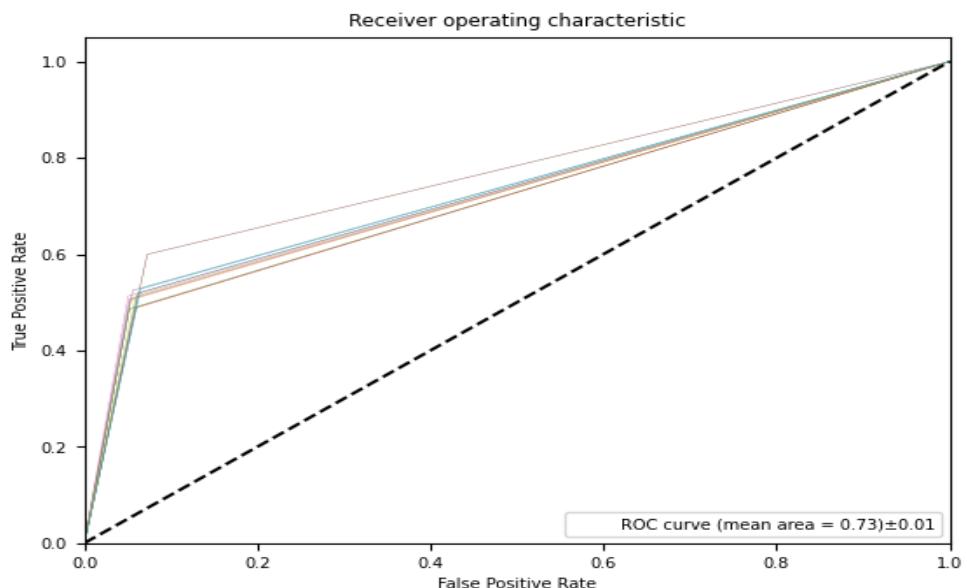
Nonostante i risultati diversi da quanto previsto, la curva ROC non sorprende. Infatti si nota che il rapporto tra i True Positive e i False Positive è più equilibrato. Anche questo deve essere una conseguenza del bilanciamento del dataset. Infine si nota che l'AUC score è aumentato, indice di classificatori più performanti.

6.1.3 Stratified k-fold cross validation esperimento 3

Osserviamo i risultatati della validazione dell'esperimento 3

	precision	recall	f1-score	support
$\leq 50k$	(0.86, 0.867)	(0.95, 0.96)	(0.904, 0.909)	(3647.35, 3648.05)
$> 50K$	(0.767, 0.801)	(0.503, 0.535)	(0.612, 0.636)	(1145.87, 1146.33)
macro avg	(0.815, 0.832)	(0.73, 0.744)	(0.759, 0.772)	(4793.5, 4794.10)
weighted avg	(0.84, 0.849)	(0.847, 0.855)	(0.835, 0.844)	(4793.5, 4794.10)

Inoltre l'intervallo dell'accuracy è: (0.847, 0.855)
e l'intervallo dei tempi di addestramento: (0.261, 0.288).
Osserviamo la curva ROC:



Si riconferma il trend sperimentale: i risultati delle tecniche di validazione dell'esperimento 3 sono estremamente simili a quelle della validazione dell'esperimento 1.

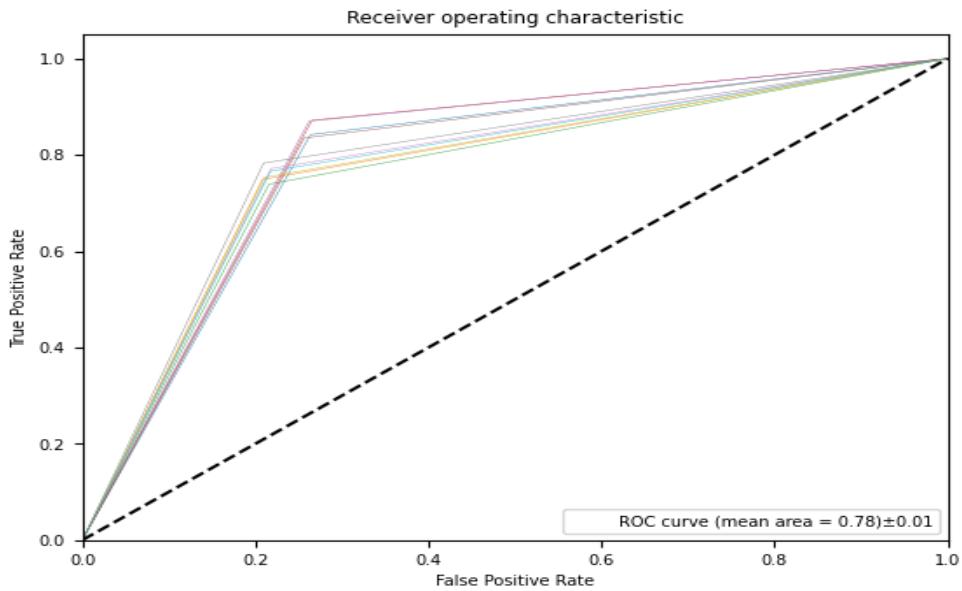
Verifichiamo se anche con i modelli del quarto esperimento questa relazione persiste

6.1.4 Stratified k-fold cross validation esperimento 4

Concludiamo con la validazione dei modelli studiati dall'esperimento 4. I risultati sono i seguenti:

	precision	recall	f1-score	support
$\leq 50k$	(0.949, 0.96)	(0.706, 0.735)	(0.813, 0.829)	(3647.35, 3648.05)
$> 50K$	(0.492, 0.51)	(0.875, 0.907)	(0.636, 0.646)	(1145.87, 1146.33)
macro avg	(0.725, 0.731)	(0.802, 0.809)	(0.725, 0.737)	(4793.5, 4794.10)
weighted avg	(0.843, 0.849)	(0.753, 0.769)	(0.771, 0.785)	(4793.5, 4794.10)

Inoltre l'intervallo dell'accuracy è: (0.753, 0.769).
e l'intervallo dei tempi di addestramento:(0.431, 0.458).
Osserviamo la curva ROC:



Come i risultati delle tecniche dell'esperimento 3 mostrava somiglianze evidenti con i risultati delle tecniche dell'esperimento 1, così anche i risultati delle tecniche dell'esperimento 4 mostrano più somiglianze con i risultati delle tecniche dell'esperimento 2.

Come per l'esperimento 2 rispetto all'1, i risultati delle tecniche di validazione dell'esperimento 4 sono più equilibrati e la curva ROC più bilanciata rispetto all'esperimento 3.

L'AUC score è aumentato e l'accuracy è diminuita.

6.2 Tecniche di validazione su Support Vector Machines

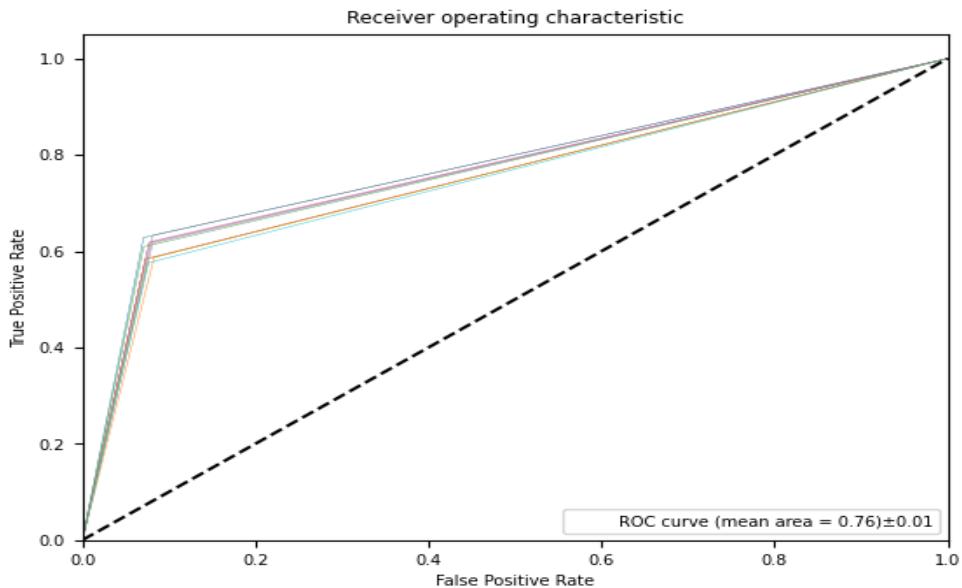
Anche per questa famiglia di modelli la tecnica di validazione usata resta la stessa. La motivazione di fondo è che non cambia la natura del dataset, quindi rimangono valide tutte le motivazioni alla base delle scelte fatte per validare gli alberi.

6.2.1 Stratified k-fold cross validation esperimento 1

Iniziamo con la prima validazione: usiamo le funzioni già viste in passato per fare i calcoli sulle metriche comuni ai modelli usati. Questi sono i risultati:

	precision	recall	f1-score	support
$\leq 50k$	(0.878, 0.886)	(0.92, 0.926)	(0.899, 0.905)	(3647.35, 3648.05)
$> 50K$	(0.703, 0.722)	(0.591, 0.621)	(0.643, 0.666)	(1145.87, 1146.33)
macro avg	(0.791, 0.803)	(0.757, 0.772)	(0.771, 0.785)	(4793.5, 4794.10)
weighted avg	(0.837, 0.846)	(0.843, 0.852)	(0.838, 0.847)	(4793.5, 4794.10)

Inoltre l'intervallo dell'accuracy è: (0.843, 0.852)



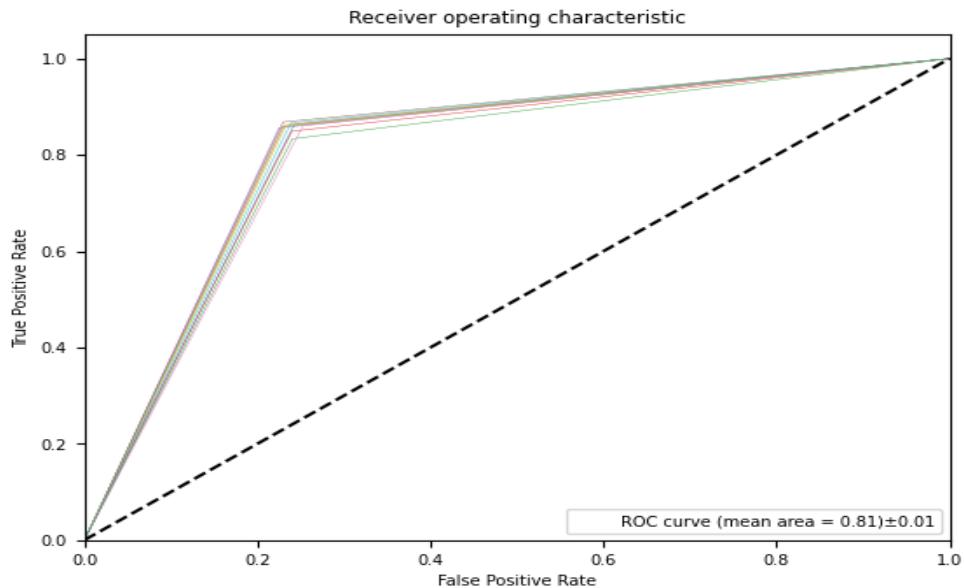
I risultati sono quelli previsti: le misure come accuracy, AUC score, ma anche precision, recall ecc sono quasi identici a quelli ottenuti validando con la medesima tecnica i Decision Tree del primo esperimento. Il comun denominatore è il dataset, infatti è possibile assumere che a causare in entrambi i casi uno sbilanciamento di questo tipo nei risultati sia un dataset sbilanciato.

6.2.2 Stratified k-fold cross validation esperimento 2

Procediamo con la verifica del secondo modello. Ricordiamo che in questo caso cambia solo il dataset. Stiamo usando quello su cui abbiamo effettuato ricampionamento con SMOTE+TomekLinks. Non cambia il parametro di complessità, né il kernel.

	precision	recall	f1-score	support
$\leq 50k$	(0.942, 0.947)	(0.758, 0.77)	(0.841, 0.849)	(3647.35, 3648.05)
$> 50K$	(0.526, 0.54)	(0.851, 0.865)	(0.651, 0.664)	(1145.87, 1146.33)
macro avg	(0.735, 0.743)	(0.806, 0.816)	(0.746, 0.756)	(4793.5, 4794.10)
weighted avg	(0.843, 0.85)	(0.781, 0.791)	(0.795, 0.804)	(4793.5, 4794.10)

Inoltre l'intervallo dell'accuracy è: (0.781, 0.791)



Quello che possiamo vedere ricalca molto i risultati ottenuti tramite validazione degli alberi del secondo esperimento. La differenza tra le Support Vector Machines validate in questo esperimento e quelle validate nell'esperimento precedente è la stessa che c'è tra gli alberi validati appartenenti agli esperimenti 1 e 2.

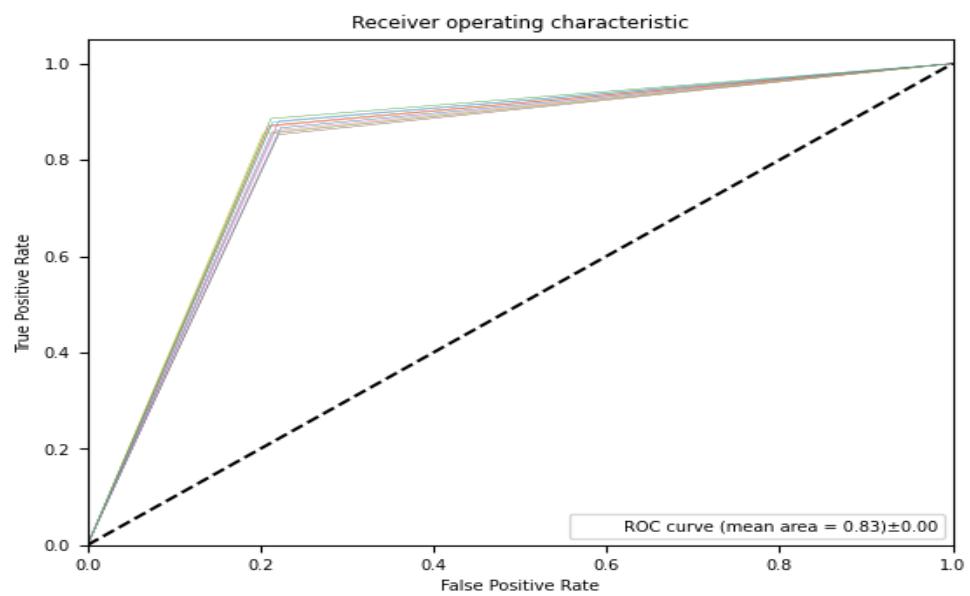
Avendo un dataset più bilanciato la recall è la curva sono più equilibrate, mentre la precision assume valori distanzi per le due classi.

6.2.3 Stratified k-fold cross validation esperimento 3

Infine validiamo gli ultimi tipi di Support Vector Machines, ovvero quelle con kernel radiale. Questi modelli dovrebbero dare risultati simili alla validazione precedente, ma dando alle metriche valori leggermente più alti.

	precision	recall	f1-score	support
$\leq 50k$	(0.947, 0.953)	(0.781, 0.79)	(0.857, 0.863)	(3647.35, 3648.05)
$> 50K$	(0.554, 0.565)	(0.859, 0.876)	(0.675, 0.686)	(1145.87, 1146.33)
macro avg	(0.751, 0.758)	(0.822, 0.831)	(0.766, 0.774)	(4793.5, 4794.10)
weighted avg	(0.853, 0.86)	(0.801, 0.809)	(0.813, 0.82)	(4793.5, 4794.10)

Inoltre l'intervallo dell'accuracy è: (0.801, 0.809).



I dati sono quelli previsti. Notiamo come la scelta del kernel abbia un impatto non trascurabile sulle prestazioni del modello.

7 Analisi dei risultati

7.0.1 Analisi su Decision Trees

Dalle tecniche di validazione possiamo concludere con una buona confidenza che i risultati degli esperimenti seguono un andamento preciso: a fare la differenza è soprattutto il bilanciamento del dataset.

Invece proprio il bilanciamento del dataset pare indubbiamente una tecnica efficace a migliorare le prestazioni dei modelli: infatti passiamo da un AUC score di 0.75 ± 0.01 a un AUC score di 0.80 ± 0.00 dall'esperimento 1 all'esperimento 2 e da un AUC score di 0.73 ± 0.01 a un AUC score di 0.78 ± 0.01 dall'esperimento 3 all'esperimento 4. Qua non guardiamo l'accuracy (che sarebbe comunque in diminuzione) perchè non ha senso fare il confronto su modelli addestrati con dataset sbilanciati per cui l'accuracy sarebbe fuorviante.

Un altro aspetto da considerare è la presenza di fnlwgt: essa non pare pesare più di tanto nelle valutazioni.

7.0.2 Analisi su Support Vector Machines

Similmente a quanto osservato con i Decision Tree, ciò che impatta maggiormente è il bilanciamento del dataset.

Il bilanciamento del dataset permette di passare da un AUC score di 0.76 ± 0.01 a 0.81 ± 0.01 .

In questi esperimenti anzichè concentrarci sull'impatto della PC, è stato preferito concentrarsi sull'impatto degli opportuni iperparametri, infatti l'ultimo esperimento ha coinvolto un kernel diverso e il suo impatto è stato concreto: l'accuracy è salita dal 78 – 79% al 80 – 81%, oltre ad aver aumentato in generale tutte le altre metriche.

Altri esperimenti sono stati condotti con diversi kernel o parametri di complessità, ma per questioni di brevità e di tempo di calcolo non è sempre stato possibile condurli o riportarli tutti.

7.0.3 Confronto tra le due famiglie di modelli

Il primo dato che va notato è il tempo di addestramento. Non perchè sia il dato più importante, ma perchè a parità di prestazioni abbiamo ordini di grandezza totalmente diversi: laddove gli alberi impiegavano meno di un secondo a terminare l'addestramento, le SVM impiegavano minuti rendendo difficile condurre esperimenti.

In secondo luogo, come citato poco fa ci sono le prestazioni. Queste sono molto simili tra le due famiglie di modelli. Non solo anche le altre metriche come precision, recall ed f1-score sono molto simili. Laddove usiamo un dataset bilanciato avremo certe misure e laddove non lo usiamo ne otteniamo di altre in entrambe le famiglie in molto modo simile.

Con le Support Vector Machines perdiamo la leggibilità delle feature. Sarebbe possibile mantenere qualche informazione, ma solo in parte e le difficoltà comportate dai lunghi tempi di addestramento non permettono un agevole lavoro.

8 Conclusioni

La natura di questo problema è complessa. I dati coinvolti sono molti e varia natura. Per risolvere questo problema si sono tentate diverse strade: bilanciare il dataset, usare dei Decision Tree con scopo di classificazione ed infine delle Support Vector Machine.

Gli esperimenti condotti ci permettono di concludere che esistono delle strade per la risoluzione di questo problema più o meno efficaci. Le strade qua percorse sono poche e incomplete, ma per questioni di brevità non è stato possibile percorrerle tutte.

I primi tentativi hanno riguardato una regolarizzazione del dataset, che è passata sotto varie forme, dalla feature selection, alla standardizzazione passando per il bilanciamento.

Altri tentativi hanno coinvolto una scelta di classificatori: i primi i Decision Tree, leggibili e veloci, ma che non rispondevano a tutte le nostre esigenze, le seconde le Support Vector Machines che sono più portate alla natura del nostro problema, resistono agli outliers, ma sono costose da addestrare. Proprio questo punto ha costituito un limite notevole in quanto non è stato possibile sperimentare più kernel o aumentare il parametro di regolarizzazione. Inoltre per questioni di brevità non è stato possibile mantenere altri esperimenti condotti su un'altra famiglia di modelli, ovvero i classificatori Naïve Bayes a cui qua faremo riferimento solo in questa occasione. Gli esperimenti condotti con i classificatori bayesiano hanno coinvolto solo le colonne categoriche del dataset codificate con One Hot Encoding, ma comunque non hanno raggiunto le performance degli alberi e delle Support Vector Machines.

Possibilità future potrebbero coinvolgere ciò che in questo progetto è stato solo accennato o iniziato. Sicuramente sperimentando con altri classificatori e diversi parametri è possibile aumentare l'affidabilità dei modelli, ma difficilmente si può portare un problema che coinvolge dati su persone a convergenza, data la trascuratezza di fattori personali che potrebbero portare i loro dati a non rientrare in schemi pre-configurati e prevedibili e quindi non essere facilmente classificabili.