ADULT CENSUS INCOME ML

Rao Francesco e Nesticò Bruno
14/02/2025

TABLE OF CONTENTS

INTRODUZIONE AL DATASET

- Il dataset **Adult Census Income,** presente su **Kaggle,** è stato estratto dal **Census Bureau degli Stati Uniti** nel 1994.
- Contiene informazioni su circa **48.842 individui** con vari dettagli come età, istruzione, occupazione, ecc.
- Obiettivo principale: **prevedere se una persona guadagna più o meno di $50.000 all'anno**.

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week	native_country	label
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50K
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50K

FEATURE DEL DATASET

Feature Numeriche

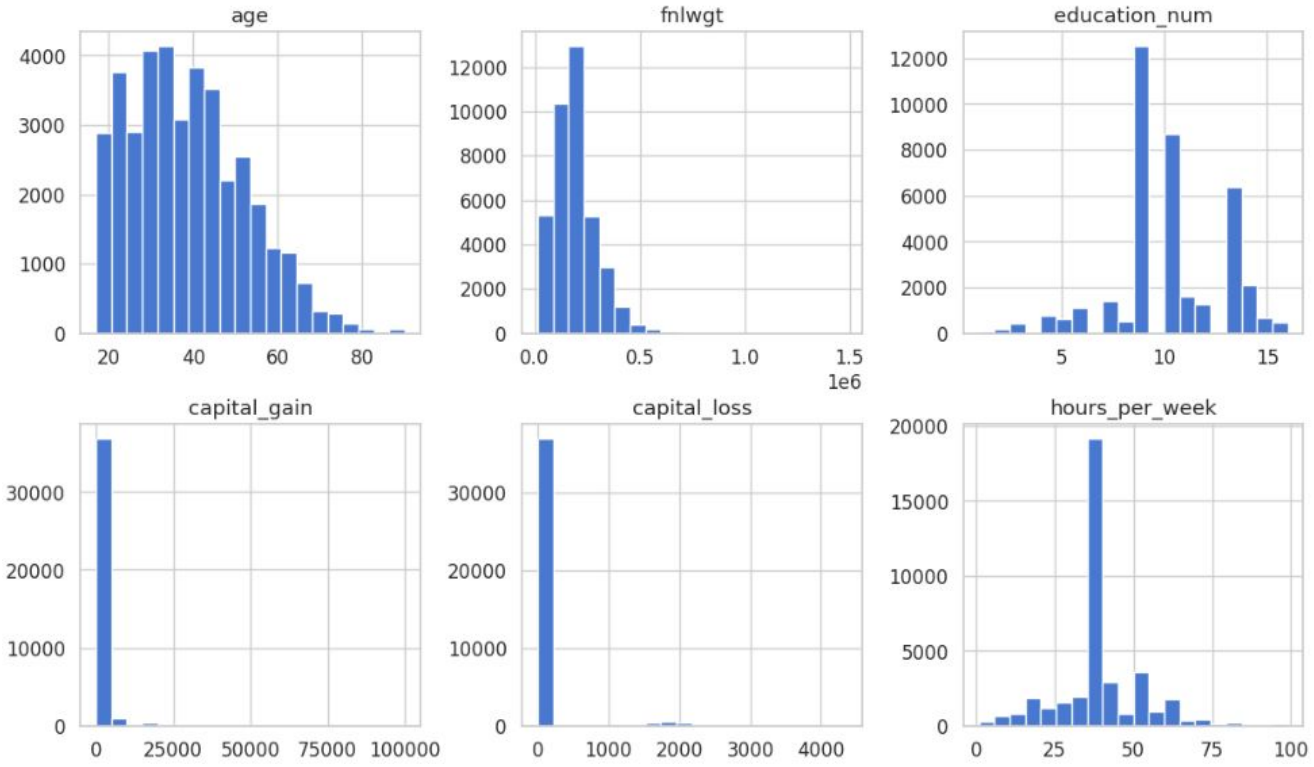
- **Age**: Età dell'individuo.
- **Hours-per-week**: Ore di lavoro settimanali.
- **Education-num**: Numero di anni di istruzione completati.
- **Capital-gain / Capital-loss**: Guadagni o perdite da investimenti o vendita di beni.
- **Fnlwgt (Final Weight)**:
    - Rappresenta il peso statistico associato a ciascun individuo. In altre parole, se un record ha un valore di fnlwgt pari a 100.000, questo significa che quell'individuo, o meglio le sue caratteristiche, rappresenta 100.000 persone nella popolazione reale. Questo peso è stato calcolato dal **Census Bureau** per compensare eventuali squilibri nel campionamento, assicurando che il campione rifletta in modo corretto la distribuzione demografica della popolazione.
    - non è una caratteristica intrinseca dell'individuo (come età, istruzione o occupazione), ma è un valore calcolato per bilanciare il campione. Inserirlo come predittore potrebbe introdurre rumore o distorcendo la reale interpretazione delle caratteristiche personali.

FEATURE DEL DATASET

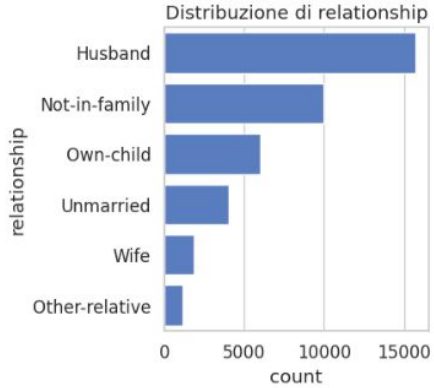
Feature Categoriche

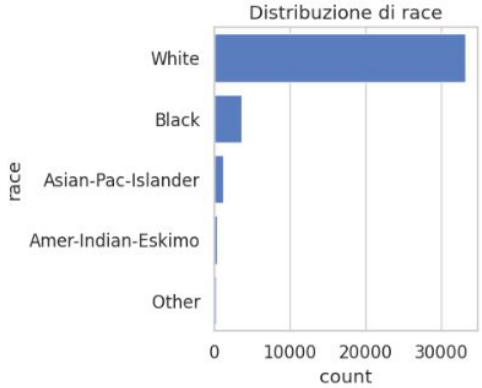
- **Workclass**: Tipo di impiego (*Privato, Pubblico, Autonomo, etc.*).
- **Education**: Livello di istruzione (*Diploma, Laurea, Master, etc.*).
- **Marital-status**: Stato civile (*Single, Sposato, Divorziato*).
- **Occupation**: Tipo di lavoro (*Tecnico, Manager, Operaio, etc.*).
- **Relationship**: Ruolo familiare (*Coniuge, Figlio, etc.*).
- **Native-country**: Paese di origine.
- **Sex**
- **Race**

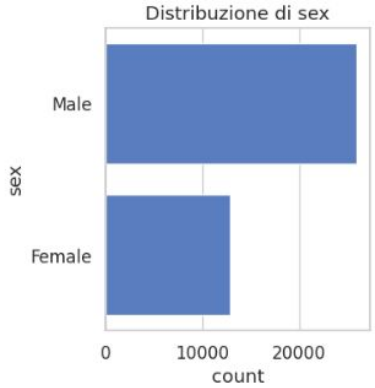
ANALISI FEATURE DEL DATASET

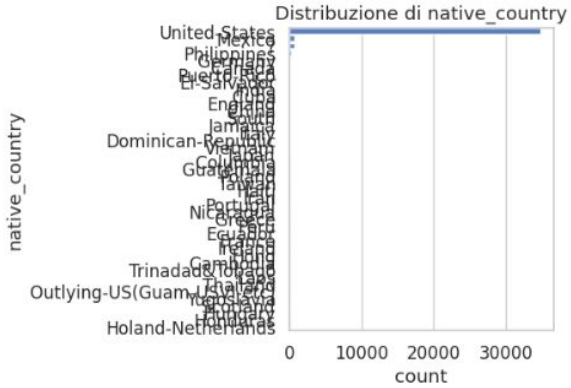


ANALISI FEATURE DEL DATASET

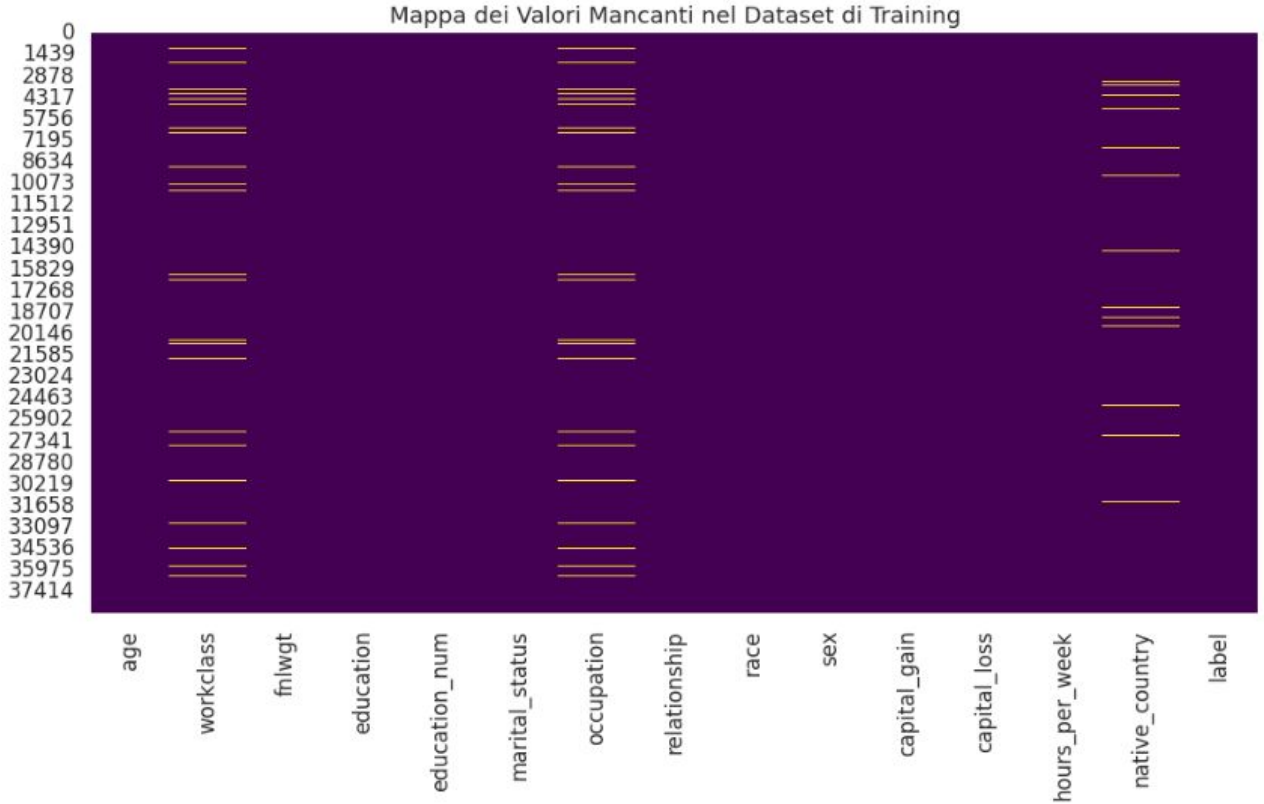








GESTIONE DATI MANCANTI



GESTIONE DATI MANCANTI

Il dataset presentava valori mancanti contrassegnati con il carattere "?" tutti di natura **categorica**.

Per affrontare questa problematica, è stata implementata una strategia di imputazione:

- **Per le variabili numeriche: (definita nel codice ma inutile nel caso specifico)**
  - sostituzione con la mediana per la sua robustezza agli outlier e la capacità di preservare la distribuzione originale dei dati.

- **Per le variabili categoriche:**
  - sostituzione con la moda per mantenere le frequenze relative delle categorie più comuni.

FEATURE ENGINEERING: RIMOZIONE DI FEATURE

- **fnlwgt**: è una caratteristica intrinseca dell'individuo (come età, istruzione o occupazione), ma è un valore calcolato per bilanciare il campione. Inserirlo come predittore potrebbe introdurre rumore o distorcendo la reale interpretazione delle caratteristiche personali.

- Le variabili **native_country** e **race** sono state eliminate a causa della distribuzione estremamente sbilanciata e per evitare potenziali bias discriminatori.
- Le variabili **capital_gain** e **capital_loss** sono state rimosse a causa della loro distribuzione fortemente sbilanciata verso lo zero, dando un contributo nullo alla predizione e introducendo solo potenziale rumore.

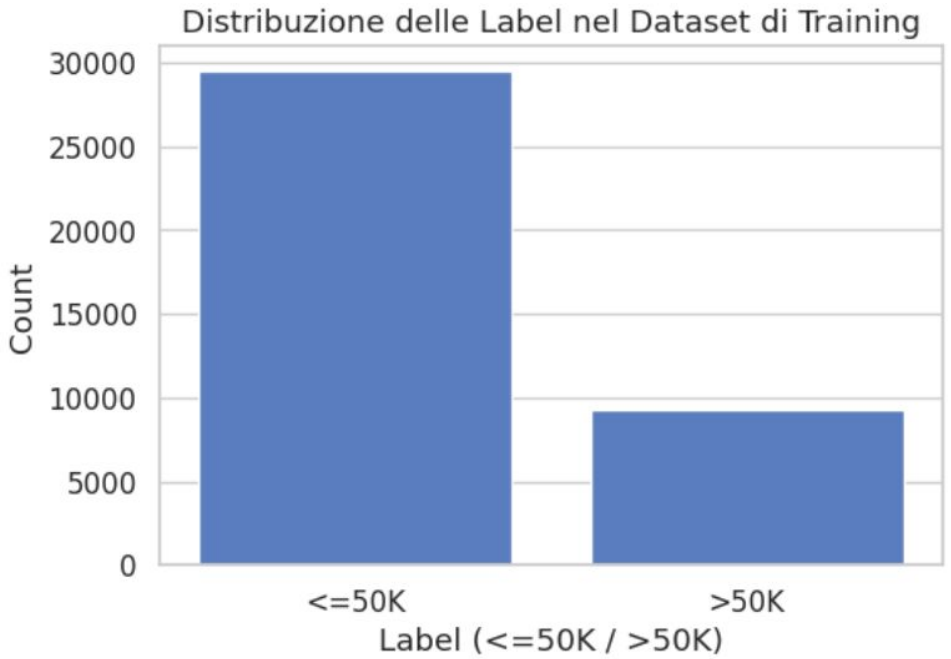
```
# Rimozione delle feature, se presenti
for col in ['native_country', 'race', 'capital_gain', 'capital_loss', 'fnlwgt']:
    if col in train_df.columns:
        train_df.drop(columns=[col], inplace=True)
    if col in test_df.columns:
        test_df.drop(columns=[col], inplace=True)
```

BILANCIAMENTO DEI DATI

Il DataSet è stato bilanciato utilizzando una combinazione di undersampling e oversampling (SMOTE).

Prima è stato eseguito un **random undersampling** parziale della classe maggioritaria, poi si è applicato **SMOTE** per portare il dataset a bilanciamento completo.

SMOTE è stato usato per generare nuovi esempi sintetici creando punti intermedi tra l'istanza esistente e i suoi vicini più prossimi della stessa classe.



BINNING E ONE HOT ENCODING

Il **binning** è una tecnica che trasforma **valori numerici continui** in **categorie discrete** raggruppandoli in intervalli (bin).

Perché si usa?

✔ **Rende i dati più interpretabili** (es. "Giovane" invece di età esatta).
✔ **Riduce la sensibilità agli outlier**.
✔ **Può migliorare le prestazioni di alcuni algoritmi** (es. Alberi di Decisione).

Esempio nel codice

Nel codice, applichiamo il binning a due feature:

1. **age** (età) → Categorizzata in gruppi di 5 anni.
2. **hours_per_week** (ore lavorative) → Categorizzata in gruppi di 5 ore.

L'**One-Hot Encoding (OHE)** è una tecnica per convertire variabili **categoriche** in variabili **numeriche binarie** (0/1), rendendo così feature categoriche processabili dai modelli di ML usati.

MODELING: DECISION TREE

Decision Tree (Albero di Decisione)

🔍 Cos'è?

Un **Decision Tree** suddivide i dati in base a domande a risposta binaria ("Sì/No"), creando una struttura ad albero.

✅ Pro e Contro

✔ Interpretabile e semplice da visualizzare.
✔ Adatto a feature sia numeriche che categoriali.
❌ Sensibile agli outlier e al **overfitting** (se l'albero è troppo profondo).

🛠️ Nel nostro caso:

- Ottimo per catturare **relazioni non lineari** tra le variabili (es. "Ore lavorate + Titolo di studio").
- Ma può sovradattarsi ai dati di training, riducendo la generalizzazione.

MODELING: RANDOM FOREST

Random Forest (Foresta Casuale)

🔍 Cos'è?

Un **Random Forest** combina più **alberi di decisione**, ognuno addestrato su **diversi sottoinsiemi** di dati e feature. Il risultato finale è la **maggioranza delle previsioni** dei singoli alberi.

✅ Pro e Contro

✔ Più robusto e accurato rispetto a un singolo albero.
✔ Meno sensibile al **overfitting** grazie all'aggregazione di più modelli.
❌ Più lento da addestrare rispetto al Decision Tree.

🛠️ Nel nostro caso:

- Ideale per **dataset con molte feature categoriali**.
- Migliore rispetto a un singolo Decision Tree.

MODELING: LOGISTIC REGRESSION

Logistic Regression (Regressione Logistica)

🔍 Cos'è?

Un **modello statistico** che utilizza una funzione logistica per stimare la probabilità di appartenenza a una classe (nel nostro caso, **≤50K o >50K**).

✅ Pro e Contro

✔ Ottimo per dati **linearmente separabili**.
❌ Non funziona bene con **relazioni non lineari**.

🛠️ Nel nostro caso:

- Nonostante la non chiara natura dei nostri dati in termini di separabilità lineare si è rivelato un modello molto capace per la modellazione del nostro dataset..

MODELING: K-NEAREST NEIGHBORS

K-Nearest Neighbors (KNN)

🔍 Cos'è?

KNN classifica un dato **in base ai suoi "K" vicini più simili**, misurando la distanza nello spazio delle feature.

✅ Pro e Contro

✔ Semplice e senza necessità di un vero "addestramento".
✔ Adattabile a distribuzioni complesse.
❌ Lento con dataset grandi (**trova i vicini per ogni predizione**).

🛠️ Nel nostro caso:

- Potrebbe risultare efficace per la possibile presenza di **gruppi di persone** con caratteristiche simili.

MODELING: NAÏVE BAYES

Naïve Bayes

🔍 Cos'è?

Un modello basato sulla **Teorema di Bayes**, che assume che tutte le feature siano **indipendenti** (ipotesi "naïve").

✅ Pro e Contro

✔ Veloce e funziona bene con **dataset sbilanciati**.
✔ Utile con dati testuali e categorici.
❌ L'ipotesi di indipendenza non è sempre realistica.

🛠️ Nel nostro caso:

- L’ipotesi di indipendenza potrebbe non essere valida per ogni feature del nostro dataset, tuttavia è stato interessante includerlo per confrontarlo ad altri modelli.

MODELING: ENSEMBLE LEARNING

Ensemble Learning: Majority Vote

🔍 Cos'è?

Un **modello ensemble** che combina le previsioni di più algoritmi, scegliendo la **classe più votata**.

✅ Pro e Contro

✔ Aumenta l’accuratezza combinando più modelli.
✔ Bilancia i punti deboli di ogni singolo algoritmo.
❌ Può essere più difficile da interpretare dei singoli modelli usati.

🛠️ Nel nostro caso:

- Prendiamo i **risultati di tutti i modelli (escludendo Naive Bayes per le scarse performance)** e scegliamo la classe più predetta.
- Miglioriamo la **robustezza e la generalizzazione** rispetto all'uso di un solo modello.

METRICHE DI VALUTAZIONE

1. Accuracy

Misura la percentuale di previsioni corrette sul totale.

✅ **Buona quando i dati sono bilanciati**, ma **può essere ingannevole se una classe è molto più frequente dell'altra**.

2. Precision

Misura quanti dei casi che il modello ha classificato come positivi sono davvero positivi.

✅ **Utile quando vogliamo evitare troppi falsi positivi.**

METRICHE DI VALUTAZIONE

3. Recall (Sensibilità o Tasso di veri positivi)

Misura quanti dei positivi reali sono stati effettivamente trovati dal modello.

✅ **Importante quando vogliamo ridurre i falsi negativi.**

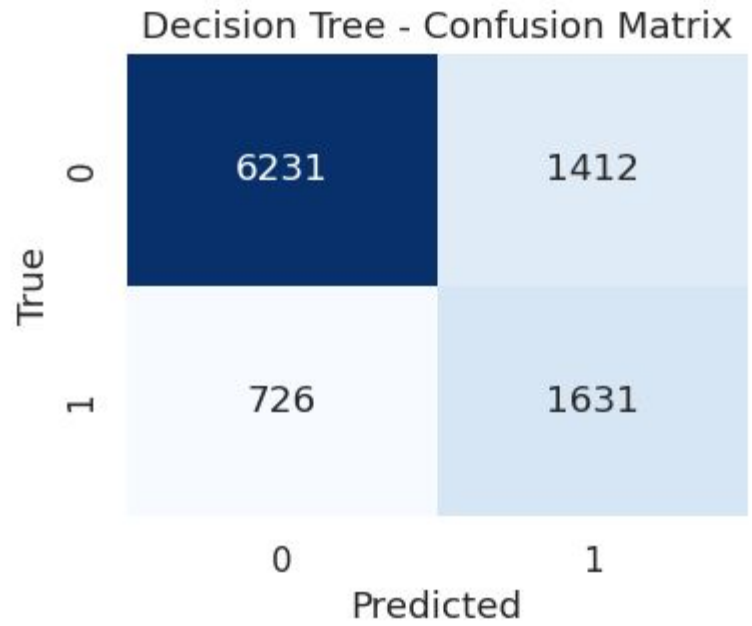
4. F1-score

È una media armonica tra Precision e Recall, utile quando vogliamo un compromesso tra le due.

✅ **Indicato quando c'è un disequilibrio tra le classi e vogliamo considerare sia FP che FN.**

RISULTATI

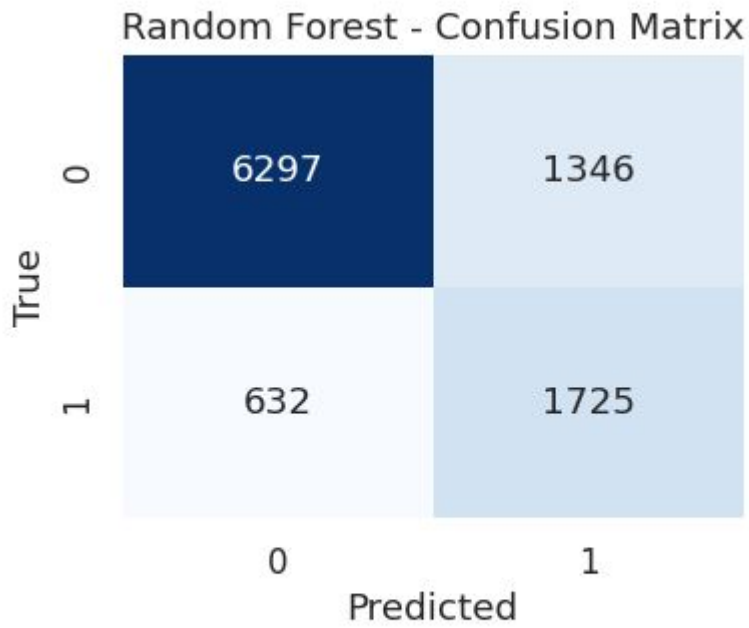
--- Decision Tree ---



Accuracy: 0.79
Precision: 0.81
Recall: 0.79
F1 Score: 0.79

RISULTATI

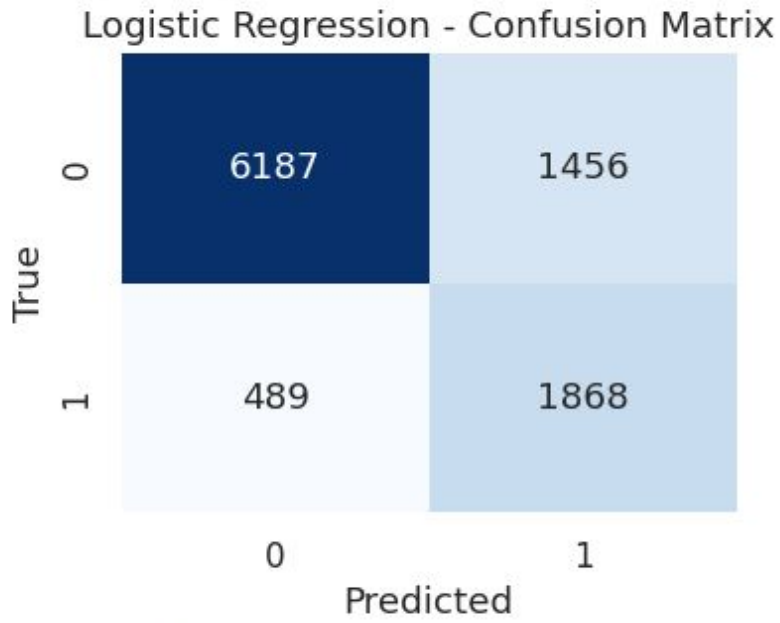
--- Random Forest ---



Accuracy: 0.80
Precision: 0.83
Recall: 0.80
F1 Score: 0.81

RISULTATI

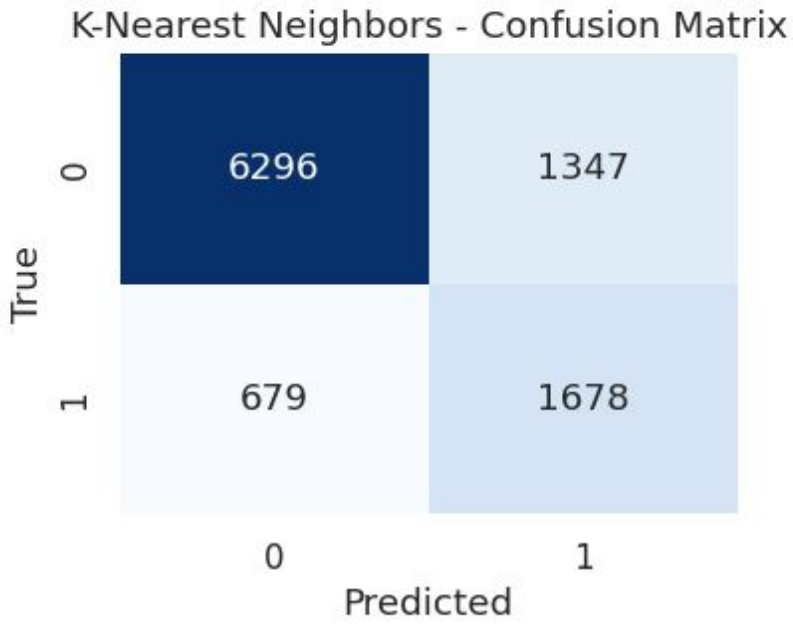
```
--- Logistic Regression ---
```




```
Accuracy: 0.81
Precision: 0.84
Recall: 0.81
F1 Score: 0.82
```

RISULTATI

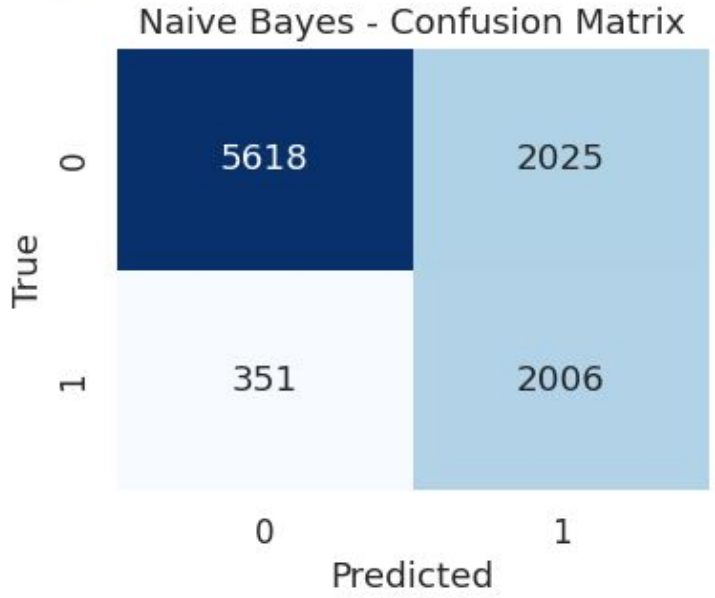
```
--- K-Nearest Neighbors ---
```




```
Accuracy: 0.80
Precision: 0.82
Recall: 0.80
F1 Score: 0.81
```

RISULTATI

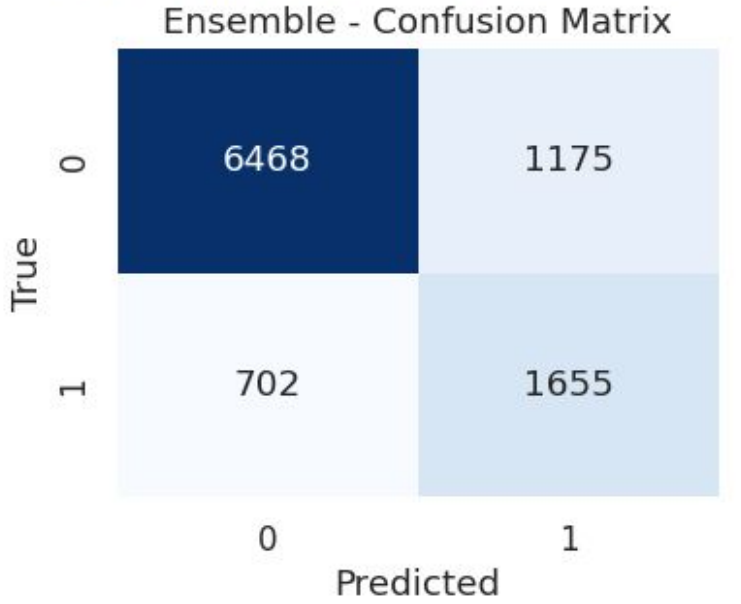
--- Naive Bayes ---



Accuracy: 0.76
Precision: 0.84
Recall: 0.76
F1 Score: 0.78

RISULTATI

```
--- Ensemble Majority Vote (senza Naive Bayes) ---
```




```
Ensemble Accuracy: 0.81
Ensemble Precision: 0.83
Ensemble Recall: 0.81
Ensemble F1 Score: 0.82
```

CONCLUSIONI E POSSIBILI MIGLIORAMENTI

Il progetto ha dimostrato l'efficacia di un approccio sistematico alla predizione del reddito, combinando tecniche avanzate di preprocessing, feature engineering e modellazione ensemble. I risultati ottenuti suggeriscono la possibilità di predire con buona accuratezza la classe di reddito di un individuo basandosi su caratteristiche demografiche e professionali.

Possibili **direzioni future** includono:

- L'esplorazione di tecniche più avanzate di feature selection, come la creazione di **nuove feature**.
- L’incremento delle **dimensioni** del dataset
- L'implementazione di algoritmi più avanzati come **deep learning**
- L'incorporazione di **feature temporali** per catturare **trend di carriera**