REPORT

FRAUD DETECTION

RILEVAMENTO TRANSAZIONI FRAUDOLENTE

Progetto:

MACHINE LEARNING

Autore:

ANTONIO BISOGNO [MAT. 0512116580]

Prof./Prof.ssa

GIUSEPPE POLESE, LOREDANA CARUCCIO

UNIVERSITÀ
DEGLI STUDI
DI SALERNO

INDICE

1. INTRODUZIONE

Il problema delle frodi nelle transazioni finanziarie rappresenta una delle sfide più rilevanti nell'ambito dell'applicazione del **Machine Learning**. Le transazioni fraudolente hanno un impatto significativo sia per gli istituti bancari sia per i clienti.

L'obiettivo di questo progetto è sviluppare un sistema in grado di distinguere tra transazioni legittime e fraudolente, sfruttando tecniche di Machine Learning.

Il sistema è stato implementato in Python, utilizzando librerie standard come ***scikit-learn***, ***pandas***, ***matplotlib***, e ***seaborn***.

È stata realizzata anche una piccola GUI interattiva in ***Tkinter*** per consentire la verifica in tempo reale di nuove transazioni.

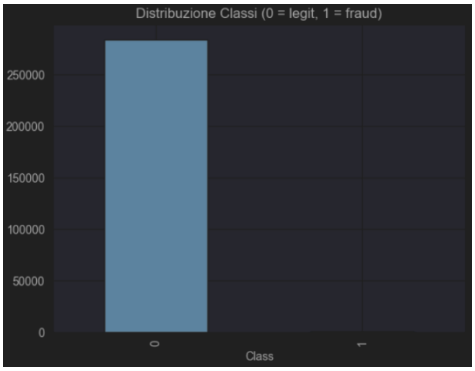
2. DATASET

2.1 FONTE

Il dataset utilizzato è il **Credit Card Fraud Detection Dataset**, pubblicato su Kaggle.com e fornito dal *Machine Learning Group* dell'*Université Libre de Bruxelles*.

2.2 CARATTERISTICHE PRINCIPALI

Il dataset raccoglie **284.807** transazioni effettuate con carte di credito europee, per un totale di **492 frodi** (0,172%), rilevate in due giorni.



Il dataset è pulito, non presenta valori mancanti. La *issue* principale riguarda lo sbilanciamento dei dati: esiste 1 frode ogni 578 transazioni.

Per motivi di riservatezza, alcune *features* non sono interpretabili direttamente, ma racchiudono relazioni nascoste nei dati originali. Ciò limita la *spiegabilità* dei modello ma aumenta la robustezza della predizione.

2.3 STRUTTURA

Contiene 31 caratteristiche:

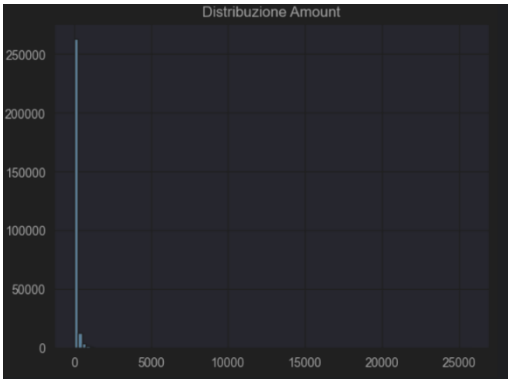
- 28 variabili di valore numerico (V1,…,V28) sono ottenute tramite la trasformazione del metodo ***PCA***;
- 2 variabili sono:
  - **Time** (tempo che intercorre tra la transazione corrente e la prima transazione);
  - **Amount** (importo della transazione);
- Classe:
  - 0 = Legit;
  - 1 = Fraud.

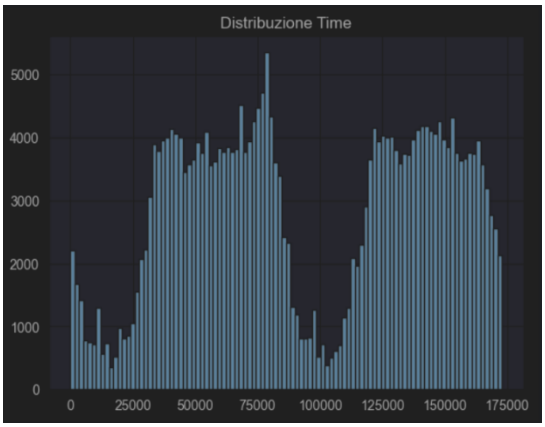
3. ANALISI DEI DATI

3.1 ANALISI ESPLORATIVA

Le classi non sono bilanciate: il rischio è che un modello che predice sempre "*Legit*" ottenga una **Accuracy** superiore al 99% ma senza alcuna utilità pratica.

La distribuzione del parametro **Amount** presenta valori molto variabili (da 0 a oltre 25.000), con forti asimmetrie e *outlier*.





La distribuzione del parametro **Time** mostra pattern legati all'attività oraria (picchi durante il giorno, cali durante la notte)

Le features **V1,…,V28** mostrano distribuzioni gaussiane centrate intorno allo zero, tipiche della PCA.

3.2 PULIZIA DEI DATI

Non sono presenti *missing values*.

Non sono stati rimossi gli *outlier* poiché le transazioni fraudolente stesse potrebbero apparire come *outlier* rispetto ai dati "*Legit*".

3.3 PREPROCESING

- **Scaling**: Amount e Time sono state standardizzate con *StandardScaler*, per portarle su una scala confrontabile con V1,…,V28;

- **Train–test split**: dati divisi in 80% training e 20% test tramite `stratify = y`, per mantenere la stessa proporzione di "Fraud" e "Legit" in entrambi i set;
- **Class imbalance**: gestito tramite il parametro `class_weight="balanced"`, che ribilancia automaticamente il contributo delle classi durante l'allenamento.

```
# INPUT
X = df[['Time', 'Amount']]
y = df['Class']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42, stratify=y
)
✓ [48] 91ms
```

4. MODELLI E METRICHE

4.1 MODELLI SCELTI

- **Logistic Regression**:
  - Modello lineare interpretabile, adatto a problemi binari;
  - Utile come baseline

```
# INPUT
log_reg = LogisticRegression(class_weight='balanced', max_iter=1000, random_state=42)
log_reg.fit(X_train, y_train)

y_pred_lr = log_reg.predict(X_test)
y_proba_lr = log_reg.predict_proba(X_test)[:,1]
✓ [49] 86ms
```

- **Random Forest Classifier**:
  - Modello ensemble basato su molti alberi decisionali;

- Robusto al rumore e capace di catturare relazioni non lineari;

```
# INPUT
rf = RandomForestClassifier(class_weight='balanced', n_estimators=100, random_state=42)
rf.fit(X_train, y_train)

y_pred_rf = rf.predict(X_test)
y_proba_rf = rf.predict_proba(X_test)[:,1]
✓ [51] 21s 692ms
```

Logistic Regression è stato scelto per la sua semplicità, rapidità di training e interpretabilità.

Random Forest è stato scelto perché garantisce maggiore Accuracy e Recall e la gestione di dataset sbilanciati.

4.2 METRICHE DI VALUTAZIONE

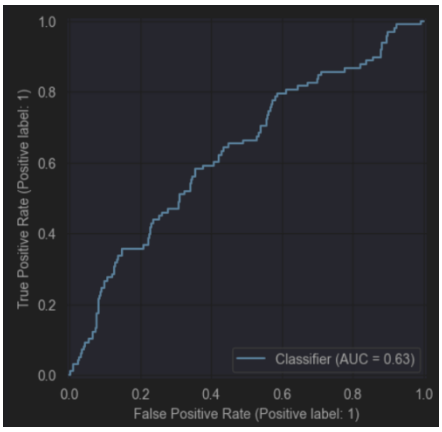
- **Confusion Matrix**: visualizza True Positive, False Positive, True Negative e False Negative;
- **Precision**: rapporto tra transazioni identificate come frodi e quelle effettivamente fraudolente;
- **Recall**: capacità del modello di individuare tutte le frodi;
- **F1-Score**: media armonica tra Precision e Recall;
- **ROC AUC**: misura la capacita discriminativa del modello;
- **Precision-Recall Curve**: particolarmente utile con dataset sbilanciati.

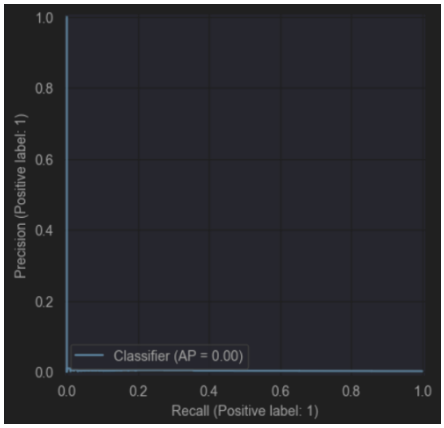
5. ANALISI DEI RISULTATI

5.1 LOGISTIC REGRESSION

Il modello riesce a catturare una buona parte delle frodi grazie a un Recall discreto.

Nonostante non sia molto preciso a causa dei molti falsi positivi, è stato considerato che è meglio identificare più transazioni come sospette piuttosto che perderne qualcuna.



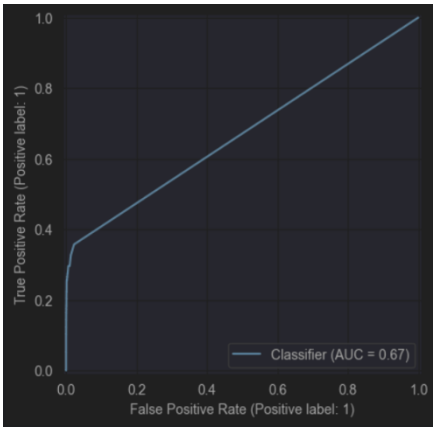


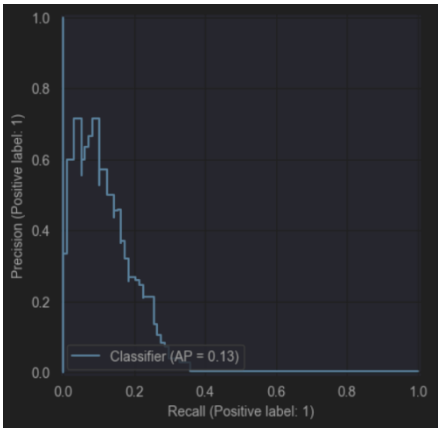
CLASSE	PRECISION	RECALL	F1-SCORE	ROC AUC
0	1.00	0.98	0.99	0.97
1	0.06	0.92	0.11	0.97

5.2 RANDOM FOREST

Il modello presenta un Recall molto alto e una Precision migliore rispetto alla Logistic Regression. Tuttavia richiede maggiore complessità e tempi di training più lunghi.

Modello adatto ad applicazioni reali grazie al buon compromesso tra Recall e Precision.





CLASSE	PRECISION	RECALL	F1-SCORE	ROC AUC
0	1.00	1.00	1.00	0.96
1	0.96	0.76	0.85	0.96

ROC Curve e Precision–Recall Curve mostrano una chiara superiorità della Random Forest. Mentre la Logistic Regression resta utile solo come baseline ma meno adatta a sistemi di rilevamento reali.

6. CONCLUSIONI

Il progetto Fraud Detection ha mostrato come il Machine Learning possa essere applicato con successo a un problema reale e complesso come il rilevamento delle frodi.

La pipeline ha incluso tutte le fasi: analisi, preprocessing, gestione dello sbilanciamento, addestramento, valutazione e prototipazione dell'applicativo.

I risultati confermano che modelli più sofisticati come la Random Forest superano approcci lineari come la Logistic Regression.

L'interfaccia Tkinter consente una semplice interazione con il modello e rappresenta un primo passo verso un'applicazione reale.

6.1 SVILUPPI FUTURI

- Bilanciamento avanzato dei dati: utilizzo di tecniche di oversampling, come ***SMOTE***;
- Ottimizzazione degli iperparametri: ***Grid Search*** o ***Random Search*** per migliorare le prestazioni
- Nuovi algoritmi: **Gradient Boosting** (***XGBoost***, ***LightGBM***);
- **Feature Engineering** per creare nuove variabili informative a partire da quelle esistenti;
- Utilizzo di un **API WEB** per integrare il modello in applicazioni bancarie reali.