Machine Learning Challenge Report

Challenge 0: Classificazione del Dataset delle Startup

Dino Meng, SM3201466

Introduzione

In questo report si descrive un resoconto di un progetto di Machine Learning, nel quale si prepara un dataset contenente informazioni su 50 start-up statunitensi - tra cui varie spese, lo stato e il profitto - e lo si analizza usando i metodi dell'apprendimento supervisionato. In particolare si va a usare i metodi della regressione logistica, usando delle eventuali regolarizzazioni per investigare sui suoi effetti.

Keywords: Machine Learning, Regressione Logistica, Regolarizzazione in L1, L2, Elasticnet

---

Metodologia

Preparazione dei dati

La fase propedeutica - e la più cruciale - di questo progetto è la preparazione dei dati. Il dataset importato contiene sei variabili descrivente una startup, che sono:

- `R&D Spend`: Spese della startup su attivita *"Research & Development"*
- `Administration`: Costi di amministrazione della startup
- `Marketing Spend`: Costi di marketing della startup
- `State`: Lo stato in cui è basata la startup.
- `Profit`: Il profitto ricavato dalla startup

Facendo un'ispezione preliminare del dataset, non ci sono dei problemi particolari come valori mancanti, variabili sparse e outlier. Pertanto non sarà necessaria effettuare la bonifica del dataset presente.

In questo progetto la variable `State` diventerà la nostra *"target variable"*, ossia la variabile che il modello imparerà a predirre.

Quindi nei prossimi passaggi vogliamo trasformare il dataset per presetare un *problema di classificazione binaria*. In particolare scegliamo `California` e `Florida` come le classi da predirre.

Per la trasformazione abbiamo usato la seguente *pipeline*:

- Filtrare le righe che contengano `State` uguale a `California` o `Florida`
- Effettuare la *one-hot encoding* su `State`, ossia `State` diventa una variabile binaria che contiene `1` se è California, `0` altrimenti (`Florida`)
- Normalizzare le variabili numeriche per riportarli sulla stessa scala; in particolare per ogni variabile dividiamo tutti i numeri per il massimo assoluto della colonna, così da avere una variabile numerica compresa nell'intervallo $[0, 1]$

L'ultimo passaggio potrebbe rivelarsi cruciale per la fase della modellazione predittiva; infatti senza questa trasformazione il nostro modello potrebbe attribuire maggiore importanza alle variabili che presentano una scala più grande delle altre variabili.

Modellazione Predittiva

Metodo di Valutazione

Nella fase di valutazione del modello, abbiamo adottato il metodo *hold-out*, suddividendo in una maniera semplice il dataset in due parti:

- **75% del dataset** per il training del modello, da cui si ottiene la *train F1-score*.
- **25% del dataset** per il testing, da cui si ricava la *test F1-score*.

Il metodo *hold-out* ci permette di valutare rapidamente le prestazioni del modello. La *test F1-score* è stata privilegiata come metrica principale in quanto essa fornisce una buona stima (sia in termini di precisione che di *recall*) delle prestazioni del modello su dati non visti, aiutando a identificare casi di *overfitting* o *underfitting*.

Per un'analisi più approfondita, sono stati utilizzati strumenti aggiuntivi:

- **Matrice di confusione**: per visualizzare la distribuzione delle previsioni del modello rispetto alle classi reali (veri positivi, falsi positivi, ecc.).
- **Curva ROC (Receiver Operating Characteristic)**: per valutare il bilanciamento tra il tasso di veri positivi (TPR) e il tasso di falsi positivi (FPR) al variare della soglia di classificazione.

Scelta dei Modelli Predittivi

Adesso si descrivono i modelli usati, e come sono stati implementati.

Prima di tutto abbiamo usato la regressione logistica implementata dalla libreria *Scikit-Learn*, ottimizzata con l'algoritmo L-BFGS.

Dopodiché abbiamo implementato manualmente la regressione logistica ottimizzata con l'algoritmo del Gradient Descent, dove la funzione loss è espressa in termini di OLS (Ordinary Least Squares), nonché le sue regolarizzazioni.

Per regolarizzare il nostro modello abbiamo usato tre metodi: la regolarizzazione LASSO (o nota come L1), la regolarizzazione RIDGE (o nota come L2) e la regolarizzazione Elasticnet.

La regolarizzazione LASSO consiste in aggiungere nella loss la somma dei pesi in valore assoluto, caratterizzata dal coefficiente di penalizzazione $\lambda \geq 0$.

$$\mathcal{L}_{\text{LASSO}} = \mathcal{L} + \lambda \sum_n |w_n|$$

La regolarizzazione RIDGE è analoga alla regolarizzazione LASSO, solo che i pesi sommati sono elevati al quadrato.

$$\mathcal{L}_{\text{RIDGE}} = \mathcal{L} + \lambda \sum_n |w_n|^2$$

La regolarizzazione Elasticnet consiste nell'effettuare una media pesata tra la regolarizzazione RIDGE e LASSO, caratterizzata dal parametro di combinazione convessa $\gamma \in [0, 1]$.

$$\mathcal{L}_{\text{ELASTICNET}} = \mathcal{L} + \gamma(\lambda \sum_n |w_n|) + (1 - \gamma)(\lambda \sum_n |w_n|^2)$$

Nel nostro caso abbiamo scelto i parametri $\lambda = 0.1$, $\gamma = 0.5$.

Risultati

Riportiamo la performance di ogni modello, quantificata dalla *accuracy* sul *test dataset*, in formato tabulare.

Modello	**Test F1-score***
Regressione Logistica (Sklearn)	0.3333
Regressione Logistica (LASSO)	0.6666
Regressione Logistica (RIDGE)	0.6666
Regressione Logistica (ElasticNet)	0.6666

* *La test F1-score è troncata fino alla quarta cifra decimale.*

Come possiamo vedere, la regressione logistica regolarizzata produce una prestazione migliore della regressione logistica implementata da Scikit-Learn.

Inoltre riportiamo la matrice di confusione delle predizioni riportate dalla regressione logistica secondo *Scikit-Learn* (fig. 1):

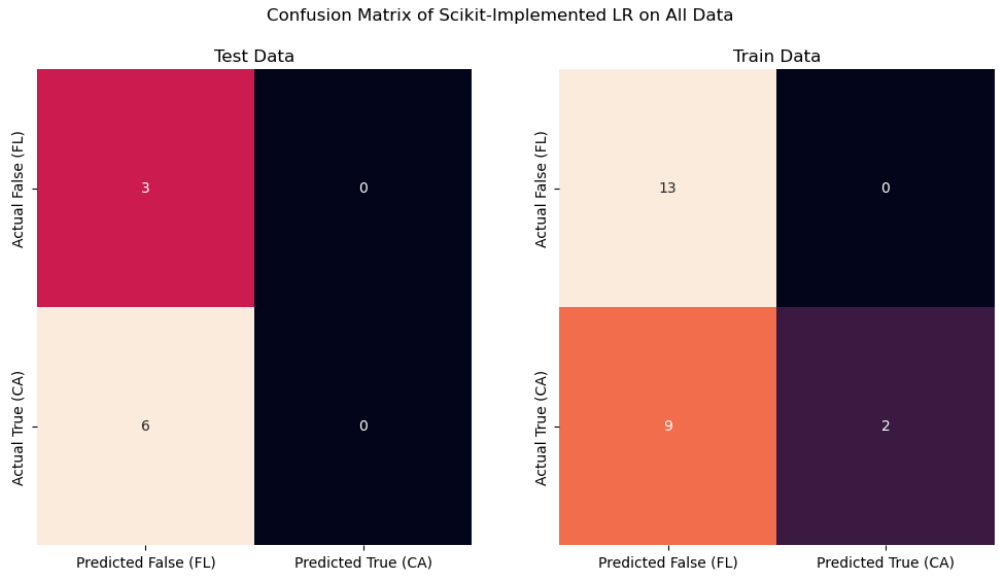

Figure 1: Scikit-Learn Implemented LR Confusion Matrixes

Invece notiamo che le matrici di confusioni delle predizioni riportate delle regressioni logistiche regolarizzate sono tutte uguali (fig. 2).

Inoltre riportiamo la curva ROC delle regressioni logistiche regolarizzate (fig. 3)

Alla fine, riportiamo anche i *scatterplot* (diagrammi di dispersione) delle variabili numeriche trasformate, per dare un panorama completo del nostro dataset (fig. 4).

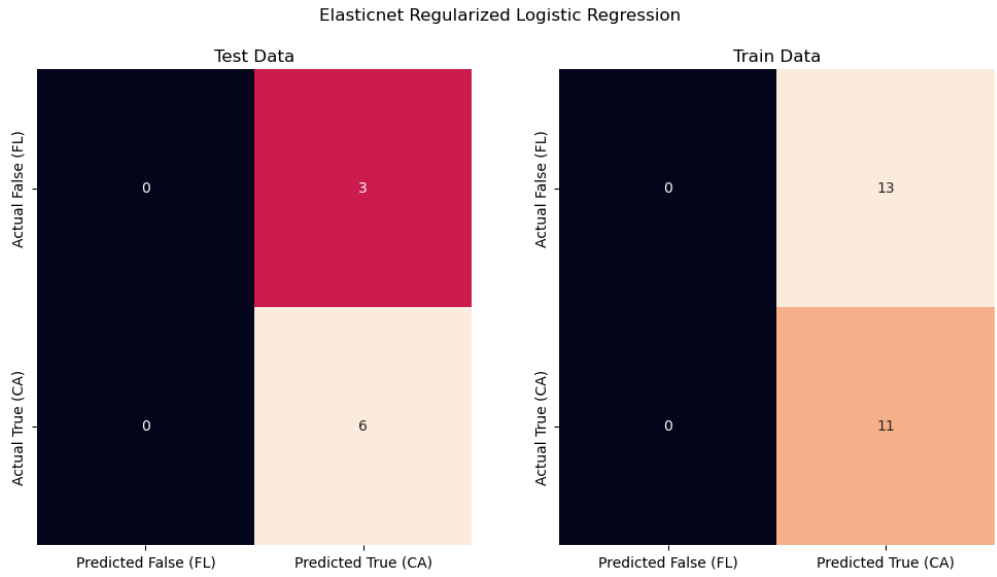

Figure 2: Manually Implemented Elasticnet-regularized LR Confusion Matrixes

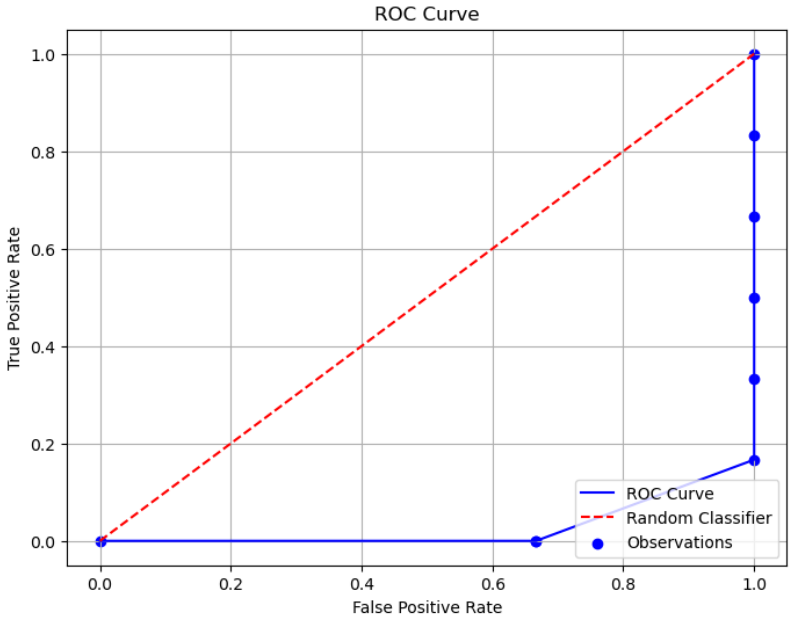

Figure 3: ROC Curve of Manually Implemented Elasticnet-regularized LR

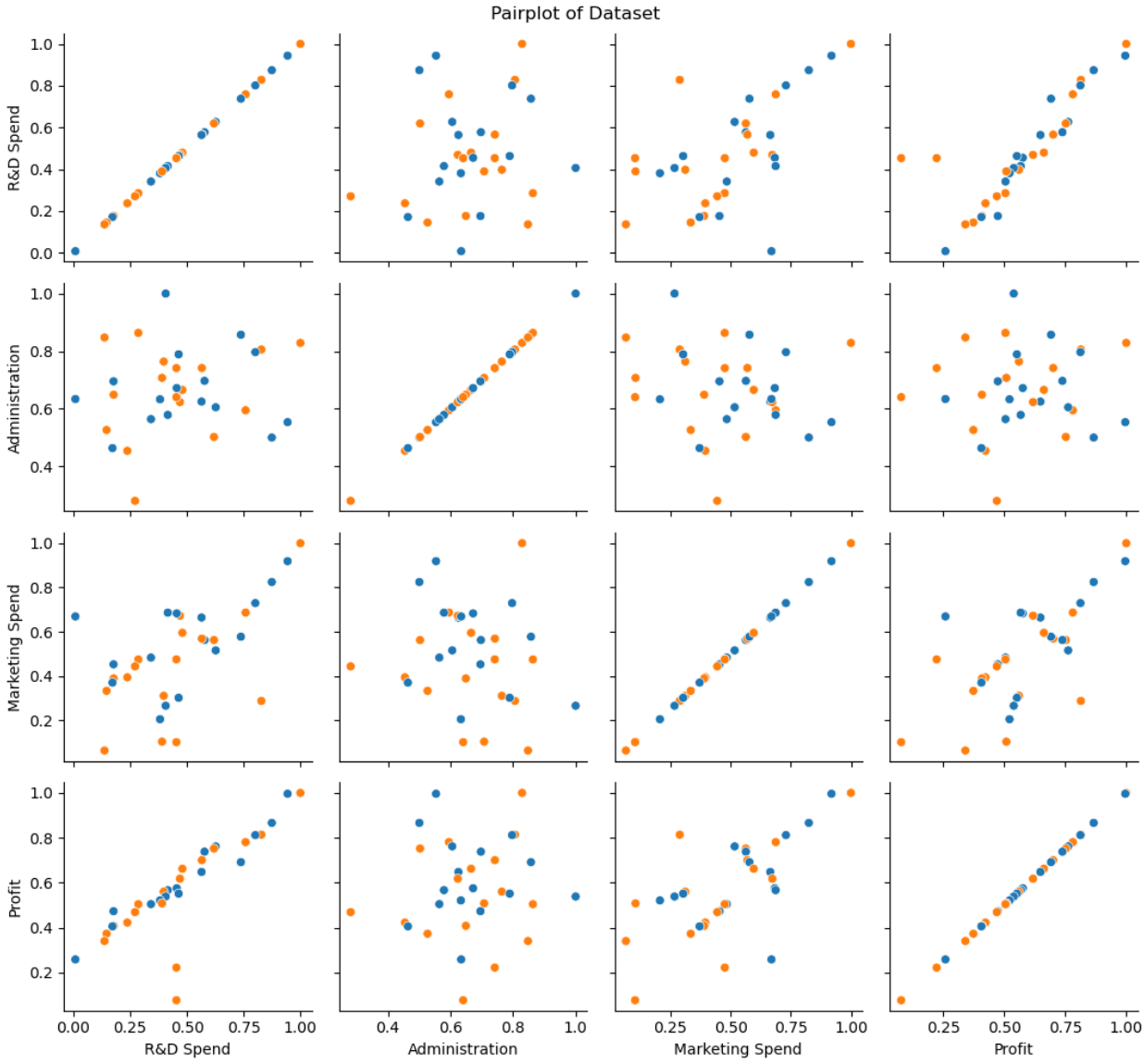

Figure 4: Dataset Scatterplots

Discussione

Come prima osservazione vediamo che c'è un comportamento differente dei modelli a seconda dell'uso della regolarizzazione. In assenza di regolarizzazione, il modello tendeva a classificare le istanze prevalentemente come negative; invece con la regolarizzazione, il modello tendeva a classificare le istanze come positive.

Questo suggerisce che, in assenza di regolarizzazione, la regressione logistica era in *overfit* sulle istanze negative. Tuttavia, con analisi approfondite vediamo che i modelli regolarizzati erano potenzialmente in stato di *underfit*: infatti presentavano comunque un test F1-score basso sia nella fase di training che di validazione; inoltre, vediamo che l'area sotto la curva ROC (AOC) ha un valore nettamente inferiore AOC $\ll 0.5$ (cfr. fig. 3), che suggerisce una debole potenza predittiva della regressione logistica regolarizata.

Una delle possibili cause di questo problema è lo spazio di dati in sè: essa ha un buon numero di variabili (6) ma un numero limitato di istanze (solo 50), comportando così una difficoltà intrinseca nell'apprendimento efficace. Inoltre, dai diagrammi di dispersione (cfr. fig. 3) si evince che il problema non è neanche linearmente separabile, ampliando ulteriormente le difficoltà nell'apprendimento dei modelli.

Conclusione

In un progetto successivo si può mitigare le problematiche descritte nella discussione usando tecniche più raffinate per trasformare il dataset.

Per esempio si potrebbe usare metodi non supervisionati per ridurre la dimensionalità del dataset (come PCA o T-SNE), trovare trasformazioni opportune per il dataset che rende il problema linearmente separabile o anche usare algoritmi di Machine Learning più complessi (come reti neurali, modelli Ensemble, ...).