**RELAZIONE MACHINE LEARNING – TEO**

Ho un’idea, un progetto o un prodotto che voglio realizzare. Non ho molti soldi, quello che voglio proporre è interessante, potrebbe funzionare, ma ho pochi soldi e necessito di un buon capitale giusto per avviare il tutto. Come faccio? Il 28 aprile 2009, è nata la piattaforma di fundrising chiamata Kickstarter, realizzata per aiutare artisti, musicisti, designer ed altri creatori a trovare il supporto e le risorse necessarie per dare concretezza ai loro progetti ed idee. Migliaia di progetti creativi sono stati avviati grazie a questa piattaforma, dove chiunque può dare una quota di denaro, in cambio di un riconoscimento dei fondatori del progetto, il prodotto stesso, oppure una parte dell’intero progetto come percentuale di equity. Ogni artista, regista, designer, sviluppatore e creatore di Kickstarter ha un controllo creativo completo sul proprio lavoro e l'opportunità di condividerlo con una vivace community di sostenitori. Al lancio del nostro progetto, ancora prima di aspettare la conclusione dello stesso, in base ad alcune caratteristiche, possiamo dire se avrà successo? Oppure otterrà un fallimento di raccolta?

INTRODUZIONE

Oltre 10 milioni di persone, provenienti da tutti i continenti, hanno sostenuto un progetto su Kickstarter.

HOLDOUT E VALUTAZIONE MODELLI

Il dataset a nostra disposizione lo abbiamo suddiviso, secondo il metodo holdout, in 67% per la parte di training set e 33% per la parte di test set. Abbiamo utilizzato il training set per addestrare i nostri classificatori, per poi passare ad una validazione successiva con il test set.

Abbiamo deciso di utilizzare come modelli: Random Forest Learner, NBTree (3.7), Bayesnet (3.7) K2, Bayesnet (3.7) TAN, NaiveBayes (3.7), Decision Tree Learner, J48, A1DE.

Ad ogni modello, abbiamo impostato gli attributi in modo da renderli performanti con i nostri modelli, utilizzando un flow di nodi per modificare la tipologia di attributi.

CROSS-VALIDATION

Ad ogni classificatore, in fase di addestramento del training set, abbiamo utilizzato un processo di cross-validation, su 5 cicli, in modo tale da permettere un’ ulteriore garanzia di validazione dei modelli, per assicurarsi di come i risultati di un’analisi statistica siano generalizzati su un dataset indipendente.

Abbiamo deciso, a causa di un problema di capacità computazionale del nostro dataset, di scegliere 5 cicli, per non aumentare enormemente il tempo di calcolo del metodo di cross-validation.

Abbiamo osservato e raccolto in una tabella i valori di precision, recall, f-measure, accuracy ed error di ogni modello di classificazione.

Andiamo a fare le seguenti considerazioni, in base ai risultati che abbiamo ottenuto, considerando l’obiettivo del nostro problema, ossia effettuare un’ottima predizione di successo o fallimento di un progetto:

* Notiamo dei valori di precision non troppo alti, ma che comunque ci permettono di considerare i modelli NBTree (3.7), BayesNet(3.7) TAN, J48 e A1DE come quelli più interessanti per questa metrica di valutazione;
* Per quanto riguarda l’error, possiamo notare che il classificatore decision tree learner, presenta il più alto valore di error. Gli altri modelli di classificazione si attestano su dei valori simili.
* Per quanto riguarda l’accuracy possiamo constatare dei valori alti per quanto riguarda i classificatori con Bayes, in particolare NaiveBayes (3.7), BayesNet (3.7) TAN, inoltre anche il modello A1DE presenta un’accuracy leggermente migliore;
* Per quanto riguarda la metrica di recall, i modelli si attestano intorno allo stesso valore, eccetto il modello random forest learner che presenta una recall piuttosto bassa, oltre anche al decision tree learner.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | RECALL | PRECISION | ERROR | F-MEASURE | ACCURACY |
| RANDOM FOREST LEARNER | 0.402 | 0.585 | 0.371 | 0.402 | 0.629 |
| **NBTREE (3.7)** | 0.532 | 0.598 | 0.343 | 0.532 | 0.657 |
| BAYESNET (3.7) K2 | 0.532 | 0.598 | 0.343 | 0.532 | 0.657 |
| **BAYESNET (3.7) TAN** | 0.518 | 0..608 | 0.342 | 0.518 | 0.658 |
| NAIVEBAYES (3.7) | 0.532 | 0.598 | 0.343 | 0.532 | 0.657 |
| DECISION TREE LEARNER | 0.475 | 0.493 | 0.412 | 0.475 | 0.588 |
| **J48** | 0.512 | 0.591 | 0.35 | 0.512 | 0.65 |
| **A1DE** | 0.533 | 0.603 | 0.341 | 0.533 | 0.659 |
| SIMPLELOGISTIC (3.7) | 0.165 | 0.556 | 0.3933 | 0.254 | 0.6067 |
| LOGISTIC (3.7) | 0.164 | 0.556 | 0.39224 | 0.254 | 0.60776 |

I modelli che abbiamo evidenziato sono quelli che mostrano delle performance più interessanti.

Abbiamo selezionato come modelli di classifazione NBTree (3.7), BayesNet (3.7) TAN, J48 e A1DE, ponendo l’attenzione sulle metriche di valutazione di precision e f-measure.

Focalizzandoci sul nostro obiettivo di classificare un progetto come successo (1), sono stati testati i modelli precedentemente selezionati applicando l’algoritmo di Adaptive Boosting, per verificare la reazione delle metriche di valutazione dei nostri classificatori al fine di un’ottimizzazione in favore dei record classificati in modo errato.

[mettere un rimando alla nostra fonte esterna, in particolare a qualcosa su wikipedia, mettendo due parole in più nel testo].

Di seguito mostriamo i valori delle metriche, successivamente al metodo di Adaptive Boosting:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | RECALL | PRECISION | ERROR | F-MEASURE | ACCURACY | AUC |
| **NBTREE (3.7)** | 0.667 | 0.64 | 0.354 | 0.653 | 0.646 | 0.797 |
| **BAYESNET (3.7) TAN** | 0.666 | 0.546 | 0.359 | 0.6 | 0.641 | 0.699 |
| **J48** | 0.749 | 0.599 | 0.304 | 0.665 | 0.696 | 0.779 |
| **A1DE** | 0.66 | 0.525 | 0.379 | 0.585 | 0.621 | 0.68 |

Possiamo constatare un leggero abbassamento dei valori di accuracy dei nostri modelli, con un aumento dei risultati di recall e di precision, che hanno condizionato anche i valori di f-measure. In particolare, possiamo notare come i valori di f-measure dei modelli J48 e NBTree (3.7) risultano essere nettamente migliorati, rendendoli i modelli più idonei per il nostro obiettivo. Inoltre, i valori di precision di questi ultimi, risultano superiori rispetto a BayesNet (3.7) e A1DE.

Inoltre, per un ulteriore confronto, sono stati rappresentati i vari modelli con la ROC curve, che ci ha permesso di osservare che i classificatori NBTree (3.7) e J48 risultano avere una parabola più alta rispetto agli altri due classificatori. Come ulteriore conferma dell’affermazione precedente, l’AUC (area sottesa della curva) presenta valori più alti nei modelli prima citati (J48 = 0.779 e NBTree (3.7) = 0.797). Osserviamo, ponendo l’attenzione sul valore dell’AUC, che il modello NBTree (3.7) presenta un valore più alto.

Decidiamo, quindi, di considerare il modello NBTree (3.7) come quello più adatto per il nostro obiettivo di predizione.

Di seguito rappresentiamo i valori della ROC curve:

APPENDICE SPIEGAZIONE FEATURE:

Id = l’id del progetto sulla piattaforma di kickstarter;

name = il nome del progetto che andrà ad apparire sulla piattaforma di kickstarter. È composto da una serie di parole che descrivono il progetto;

category = la categoria nel quale il progetto andrà ad essere posizionato.

main\_category = la categoria di una campagna;

currency = la valuta della moneta con il quale è stato lanciato il progetto;

deadline = il giorno che viene posto come termine della campagna del progetto;

goal = l’obiettivo di raccolta a livello economico del mio progetto;

launched = la data del lancio della nostra campagna del progetto;

pledged = l’ammontare di denaro che è stato raccolto dalla campagna;

state = lo stato nel quale il progetto è terminato, in particolare può essere live, canceled, life, success, failed

backers = il numero di investitori di ogni singolo progetto;

country = la nazione dalla quale è arrivato il caricamento del progetto;

usd\_pledged = l’ammontare di denaro raccolto in USD (con la conversione effettuata da KS);

usd\_pledged\_real = l’ammontare di denaro raccolto in USD (con la conversione di fixed.io api);

usd\_goal\_real = l’obiettivo di raccolta in USD.

FEATURE CREATION

Una volta selezionate le features più adatte per il nostro obiettivo di predizione, abbiamo deciso di creare da queste ultime delle features. Partendo dalle date di lancio e di termine della nostra campagna, abbiamo calcolato la differenza di tempo tra il termine e l’inizio della campagna stessa. Abbiamo così creato l’attributo project\_duration, con il numero di giorni di presenza della campagna su kickstarter, considerando che probabilmente più giorni una campagna rimane attiva, più ci potrebbero sostenitor, e quindi investitori. Dal giorno di lancio, abbiamo estrapolato:

* launched\_month, che ci permette di sapere in quale mese il progetto è stato lanciato;
* launch\_day, che ci permette di sapere il giorno della settimana nel quale il progetto è stato lanciato;
* weekend, che ci permette di sapere se la campagna è iniziata nel weekend o meno;

Considerando che i progetti presentano un ampio tasso di sottoscrizioni in tempi brevi dal lancio, dato che il progetto con il passare del tempo, viene superato da progetti più recenti, quindi obbligando chi ha lanciato la campagna a pagare kickstarter per mantere il progetto nelle prime posizioni di ricerca.

VALUTAZIONE MODELLI