Università degli studi di
Milano-Bicocca

Text Mining and Search

Final Project

Twitter US Airline Sentiment Classification

Autori:
Lorenzo Mauri - 807306 - l.mauri28@campus.unimib.it
Alessandro Vincenzi - 800608 - a.vincenzi3@campus.unimib.it
Emanuele Rebesco - 864006 - e.rebesco@campus.unimib.it
June 8, 2021

Contents

Abstract

Grazie ad un'analisi mirata del testo presente sulle piattaforme social, siamo spesso in grado di generare un valido prospetto che sintetizzi il pensiero dei consumatori e che fornisca inoltre delle metriche od informazioni utili per monitorare la qualità dei servizi offerti.
Questo lavoro è volto all'analisi del giudizio e delle opinioni dei passeggeri di diverse compagnie aeree tramite l'elaborazione di una vasta collezione di recensioni proveniente da Twitter.
Vengono difatti impiegate algoritmi di Machine Learning (come *Neural Network*, *Random Forest* e *Support Vector Machine*) con lo scopo classificare la polarità delle recensioni e viene presentata una metodologia per la risoluzione dello sbilanciamento di classe. In questo specifico caso la *Text Augmentation* permette di migliorare in media l'accuracy dei classificatori di circa il 6%.

1 Introduzione

Generalmente, viaggiare in aereo, è il metodo più comodo e rapido per percorrere lunghe tratte, spesso ostacolate da mari e montagne. Infatti il viaggio regala un notevole vantaggio in termini temporali rispetto ad ogni altro mezzo di trasporto convenzionale offrendo la possibilità di risparmiare tempo prezioso e permettendo di evitare difficoltà logistiche che potrebbero compromettere la buona riuscita del viaggio. Purtroppo però la comodità non coincide con la convenienza per tutte le compagnie aeree, difatti l'esperienza dipende principalmente dal rapporto che si instaura tra i passeggeri e il vettore, a partire dal check-in e terminando con il check-out. Volare rende il viaggio verso destinazioni lontane più gestibile, ma presenta anche alcuni inconvenienti, in quanto il successo dell'esperienza di volo dipende principalmente dall'organizzazione complessiva del vettore.
Il fine del progetto è quello di svolgere un task di classificazione, attribuendo l'esatto sentimento espresso da ciascuna recensione, prodotta all'interno della piattaforma social Twitter. Grazie a questi dati è possibile creare dei modelli altamente specializzati in grado di determinare se un particolare tweet sia negativo, positivo o neutro, ma anche di scoprire quale compagnia aerea sia la più apprezzata così come la più detestata e perfino comprendere quali siano i problemi più diffusi e che più preoccupano i passeggeri. Ciò avrebbe un forte impatto sulla crescita delle compagnie stesse, in quanto aiuterebbe loro a comprendere le proprie carenze tramite feedback utili e stimolanti.

1.1 Descrizione formale del problema

Il task in esame è un problema di classificazione **Single-Label Multi-Class**.

Dato un insieme di oggetti

$$D = x_1, x_2, \ldots \tag{1}$$

e un insieme di categorie o classi

$$C = c_1, c_2, \ldots, c_n \tag{2}$$

l'obiettivo è quello di associare ciascuna istanza appartenente a D (1) ad *una e una sola* classe in C (2) , cioè vale la relazione

$$h : D \to C \tag{3}$$

E' importante notare che in questo caso vale $n > 2$, cioè esistono 3 classi.

2 Datasets

Il dataset, proveniente dalla piattaforma Kaggle [1], è una collezione di **14485 recensioni** in lingua inglese espresse dai passeggeri riguardanti 6 diverse compagnie aeree americane.
I dati forniti, postati nel Febbraio 2015, sono stati estratti dal social network Twitter.

Il dataset è composto dalle seguenti 15 variabili:

1. **tweet_id** : Codice identificativo del tweet
2. **airline_sentiment** : Polarità dei tweet: positiva, negativa, neutra
3. **airline_sentiment_confidence** : Confidenza attribuita alla polarità assegnata
4. **negativereason** : Ragione principale per la quale un commento è negativo

[1] https://www.kaggle.com/

5. **negativereason_confidence** : Confidenza attribuita alla causa identificata nei commenti negativi
6. **airline** : Nome della compagnia aerea
7. **airline_sentiment_gold** : Variabile ignota
8. **name** : Nome utente degli account Twitter
9. **negativereason_gold** : Variabile ignota
10. **retweet_count** : Somma dei retweet per ogni tweet
11. **text** : Messaggio testuale contenuto nel tweet
12. **tweet_coord** : Variabile ignota
13. **tweet_created** : data di creazione del tweet
14. **tweet_location** : luogo di creazione del tweet
15. **user_timezone** : ore locali

Per la nostra analisi, sono state utilizzate principalmente 2 variabili, quali: **text** e **airline_sentiment**.

3 Approccio Metodologico

In questo paragrafo vengono esposti i procedimenti seguiti e il loro dettagliato sviluppo per lo svolgimento di uno studio accurato e minuzioso. I temi affrontati vengono riassunti in 4 sottoparagrafi, ovvero: Pre-processing, Data Augmentation, Text Representation e Data Modeling.

3.1 Pre-processing

Durante la fase di Pre-processing, sono state adottate numerose operazioni di pulizia del testo per facilitare il successivo lavoro di modellazione. Nello specifico, evidenziamo:

- **Normalizzazione** : processo che ha il compito di eliminare il superfluo e i rumori all'interno del testo, per produrre un'unica forma canonica ideale. Viene attuato quanto segue:

  1. Rimozione dei tag e degli hashtag (@abc ; abc)
  2. Rimozione delle emoji
  3. Rimozione degli URL
  4. Rimozione della punteggiatura
  5. Rimozione dei numeri
  6. Rimozione degli spazi bianchi
  7. Rimozione delle recensioni vuote
  8. Rimozione di termini orfani dovuti ad acronimi o contrazioni
  9. Case Folding - lower case

- **Tokenization** : fondamentale step che frammenta i testi in singoli elementi, chiamati token. In particolare, nella libreria *nltk* è possibile sfruttare uno specifico *tokenizzatore* che tenga conto di caratteri propri della sintassi comunemente utilizzata nei corpi testo dei *tweet*, ad esempio hashtags, qualora si vogliano preservare.

- **Rimozione stop-words** : vengono eliminate tutte quelle parole che non contribuiscono al trasporto di informazione aggiuntiva e che si presentano con ritmo frequente. E' necessario tale smaltimento perchè la loro presenza spesso compromette i risultati dei modelli di classificazione.

- **Lemmatization** : operazione che si occupa di ridurre le varie forme flesse delle parole alla loro forma canonica

- **Stemming** : netto troncamento alla radice della parola, che genera forme lessicali generiche in maniera più rude

4 Sbilanciamento di classe

Successivamente alle operazioni descritte in precedenza, il dataset viene bilanciato tramite metodi di oversampling e undersampling.
Di seguito le tecniche adottate :

1. **Data Augmentation** : si tratta di una tecnica di *oversampling*, descritta in modo più approfondito nella sezione successiva.
   E' comunque importante evidenziare che questo metodo, basato sulla sostituzione di sinonimi nelle recensioni, deve essere applicato necessariamente prima di operazioni come lo stemming e la lemmatization onde evitare risultati errati.

2. **Campionamento casuale semplice** : si tratta di una tecnica di *undersampling* tramite la quale è possibile ridurre la classe maggioritaria.

Vengono creati due diversi insiemi di dati al fine di valutare il contributo apportato da ciascuna tecnica alle metriche di performance dei classificatori.

4.1 Data Augmentation

4.1.1 Introduzione alla metodologia

In termini generali, con *Data Augmentation* si intende una tecnica che ha lo scopo di espandere artificialmente le dimensioni di un dataset contenente una determinata tipologia di oggetti, in questo caso stringhe di testo, creando varianti dell'oggetto stesso. In altre parole per ciascun oggetto si genereranno $n - varianti$, secondo uno o più criteri di base.
Adottando una o più tecniche di *Data Augmentation*, ci si aspetta in linea di principio, uno scenario modellistico tendenzialmente migliorativo, in particolare:

1. Aumento dei valori di *accuratezza* e *recall*, e in generale una migliore performance;
2. Miglioramento capacità di prevenire l'*overfitting*, migliore gestione dell'*errore di generalizzazione*;
3. Allontanamento della condizione di squilibrio tra classi;

In questo contesto, per esempio, non è ragionevole aumentare i dati utilizzando tecniche equivalenti alla *signal trasformation* come nei casi in cui si operi con set di immagini o audio, in quanto all'interno del testo l'ordine esatto dei caratteri può formare un significato sintattico e semantico ben definito. Alla luce di ciò, il modo migliore per aumentare il volume dei dati sarebbe quindi quello di usare nuove formulazioni per le frasi.

Tuttavia, passare in rassegna manualmente il dataset rende un simile approccio irrealistico e costoso a causa del volume di campioni contenuto nel set di dati stesso. Di conseguenza, la scelta più naturale per l'aumento dei dati è operare una sostituzione di parole con i loro **sinonimi**.[1]

4.1.2 Sostituzione lessicale tramite thesaurus

In termini computazionali, sono stati presi in considerazione differenti approcci per la costruzione dei record artificiali. Risultano interessanti :

- Sostituzione sinonimi in lingua inglese basata sulla libreria Python *nlpaug*, in cui viene utilizzato internamente un thesaurus fornito da *nltk*.[2]
- Sostituzione sinonimi in lingua inglese basata su thesaurus esterno via web (http://paraphrase.org), contenente 169.6 milioni di parafrasi (PPDB) in lingua inglese.

Nel primo caso il thesaurus a sua volta è ricavato da WordNet, dove ogni sinonimo di una parola o di una frase è classificato in base alla vicinanza semantica al più frequente significato. Nel secondo caso le parafrasi in PPDB sono classificate utilizzando un modello di regressione supervisionata. Per determinare quante e quali parole sostituire, viene utilizzato un approcio probabilistico per tutte le parole sostituibili (le stopwords sono escluse), con *distribuzione geometrica*. (nel secondo la distribuzione è a scelta dell'utente). Nello specifico, vengono estratte r parole dal set di tutte le parole sostituibili. La probabilità legata al numero r di parole estratte è determinata da una distribuzione geometrica con parametro p in cui $P[r] \sim p^r$. A sua volta l'indice s di un sinonimo scelto data una parola, è determinato da un'altra distribuzione geometrica in cui $P[s] \sim q^s$. In questo modo, la probabilità di un sinonimo scelto diminuisce quando si allontana dal significato più frequente. [1]

4.2 Text Representation

Durante questa fase viene costruito un peso da assegnare alle parole nei documenti, per scoprire quali siano le più discriminanti. Viene considerata la frequenza dei termini sia all'interno del singolo documento, sia nell'intero corpus di documenti. L'informazione congiunta del TF (Term Frequency)

[2]Soluzione attualmente implementata.

con quella del IDF (Inverse Document Frequency) permette di ottenere il peso

$$w_{d,t} = (\frac{tf_{t;d}}{max_{ti} tf_{ti;d}}) * log_{10}(\frac{N}{df_t})$$

assegnato alle parole. Combinando queste misure, il peso ricavato esprime l'importanza di una parola in un documento. Esso aumenta al crescere del numero di occorrenze nel documento, d'altra parte diminuisce al crescere della sua frequenza all'interno dell'intera collezione. Successivamente viene eseguito un mapping per ridurre la dimensionalità della matrice term-context, eseguendo una selezione di feature latenti.
La *Singular Value Decomposition* genera una matrice

$$\underset{7080\times400}{\mathrm{X}} \tag{4}$$

nel caso del dataset not-augmented e una matrice

$$\underset{10680\times400}{\mathrm{X}} \tag{5}$$

nel caso del dataset con data augmentation.

4.3 Data Modeling

La parte di addestramento dei modelli prevede uno split dei due diversi dataset secondo le proporzioni usuali : 80% per il training set e il restante 20% per il test set.
Nelle sezioni seguenti riportiamo alcune delle caratteristiche dei calssificatori utilizzati.

4.3.1 Feed-Forward Neural Network

Durante l'impiego di una rete neurale, è stato optato una Feed-Forward Neural Network con i seguenti parametri:

- numero di hidden layer : 1
- unità input layer : 128
- unità hidden layer : 64
- unità output layer : 3

- regolarizzazioni : dropout con $p = 0.8$
- learning rate : 0.001
- funzioni di attivazione : relu e softmax
- funzione di perdita : sparse categorical crossentropy

La struttura di questa rete è stata ponderata secondo vari criteri, tra cui: è stata assegnata come funzione di attivazione nei layers intermedi la Relu, invece come funzione di attivazione nei layers esterni la Softmax; inoltre è stato annesso il layer di dropout tra il layer di input e l'hidden layer in modo tale da generalizzare il modello e quindi evitare overfitting, ma anche ottenere una rappresentazione indipendente dei neuroni.

4.3.2 Random Forest

Per *Random Forest* si intende un classificatore d'insieme (*ensemble method*) ottenuto dall'aggregazione tramite bagging di alberi di decisione[2] [3]. Gli algoritmi *Random forest* si pongono come soluzione che minimizza l'overfitting del training-set rispetto agli alberi di decisione.
Una classe di alberi di decisione di grandezza arbitraria, presenta un valore infinito di *VC-dimension*, rendendo il metodo prono all'overfitting. Al fine di limitare la possibilità di incorrere in tale fenomeno, si può certamente ridurre la dimensione dell'albero di decisione, o alternativamente e altrettanto efficaciemente generare appunto un ensemble di alberi decisionali. Un random forest pertanto è un classificatore costituito da una collezione di alberi decisionali, in cui ogni albero è costruito applicando un algoritmo A su di un training set S e un vettore addizionale casuale θ, dove è campionato *i.i.d.* La previsione del Random Forest è ottenuta con un voto a maggioranza sulle previsioni dei singoli alberi. [4].
Il numero di stimatori scelto nel caso in esame risulta essere: 50

4.3.3 Support Vector Machines

Per *Support Vector Machines* (chiamate anche *Support-Vector-Network*) si intende una famiglia di modelli ad apprendimento supervisionato, e relativi algoritmi associati, per l'analisi di problemi di classificazione e regressione [5]. Più specificatamente, i paradigmi SVM trovano applicazione

nell'implementazione di predittori lineari, in spazi di features ad alta dimensionalità. I modelli basati sul paradigma SVMs sono tra i metodi di predizione più robusti essendo basati sul framework statistico della VC-theory[3] [4].
A titolo di esempio si consideri un insieme di campioni appartenente ad un training set, ciascuno contrassegnato come appartenente a una di due categorie. Un algoritmo di addestramento SVM costruirà un modello che assegna nuovi campioni a una categoria o all'altra, rendendo il modello un *classificatore binario lineare non-probabilistico* (sebbene alcuni metodi quali il Platt scaling[4], utilizzino il paradigma SVM in un contesto di classificazione probabilistica). In generale SVM mappa campioni di un training set a punti nello spazio in modo da massimizzare l'ampiezza del divario tra le due categorie. I nuovi campioni vengono quindi mappati nello stesso spazio e si prevede che appartengano a una categoria in base a quale lato del divario cadono. Questo tipo di condizione è detta ad *"ampio margine"*. Da questo si evince come Il paradigma SVM affronti il problema della *sample-complexity*. Più formalmente, per *ampio margine* intendiamo la condizione per cui il training set, è separato da un *mezzo-spazio* in cui i punti considerati giacciono nel lato corretto dell'*iperpiano* e al contempo siano distanti dal limite dell'iperpiano stesso, in una condizione di massima distanza [4]. Limitare l'output dell'algoritmo al solo caso in cui l'elemento separatore sia in una condizione di ampio margine, porta ad una diminuzione della *sample-complexity* anche in caso di spazi di features ad infinita dimensionalità[4].
È importante notare altresì che:

1. Oltre al caso di classificazione lineare, i modelli basati su paradigma SVM possono efficientemente performare anche nel caso di classificazioni non-lineari, utilizzando quella che viene definita comunemente *kernel-trick*.

2. Nel caso di dati non contrassegnati quando l'approccio non supervisionato è necessario, la variante Support Vector Clustering SVC [6], consente di applicare la statistica dei vettori di supporto, propria delle SVMs, al fine di categorizzare tali dati determinando un raggruppamento naturale dei dati stessi.

[3]https://en.wikipedia.org/wiki/Vapnik-Chervonenkis_theory
[4]https://en.wikipedia.org/wiki/Platt_scaling

5 Risultati e Valutazioni

In questa sezione vengono e i risultati, ovvero le metriche di performance dei classificatori per ciascun dataset considerato (con e senza data augmentation).
Essendo il problema di classificazione di tipo **Single-Label Multi-Class**, la metrica utilizzata per la valutazione dei modelli è l'**accuracy**, cioè il rapporto tra il numero delle istanze correttamente classificate e il totale preso in considerazione.
La formula 6 mostra il calcolo della metrica.

$$A = \frac{\sum_{c_i \in \varsigma} \Lambda_{ii}}{\sum_{c_i, c_j \in \varsigma} \Lambda_{ij}} \tag{6}$$

Nella Tabella 1 vengono quindi riportate le misure di accuracy per ogni dataset utilizzato, mentre nelle tabelle successive (Tabella 2 e Tabella 3) le metriche di precision, recall e f1-score dei modelli *Random Forest* e *Support Vector Machine*, sempre per ciascun dataset.

Nella sezione successiva,invece, si procederà con la discussione ed interpretazione dei risultati ottenuti.

Table 1: Misure di accuracy

	Not Augmented	Augmented
	accuracy	accuracy
NN	0.6977	0.7374
RF	0.63	0.71
SVM	0.71	0.77

Table 2: Dataset **not** Augmented

	RF			SVM		
	precision	recall	f1-score	precision	recall	f1-score
positivi	0.77	0.61	0.68	0.82	0.70	0.76
negativi	0.57	0.71	0.63	0.71	0.71	0.71
neutri	0.59	0.58	0.58	0.63	0.73	0.67

Table 3: Dataset Augmented

	RF			SVM		
	precision	recall	f1-score	precision	recall	f1-score
positivi	0.82	0.71	0.76	0.84	0.78	0.81
negativi	0.68	0.75	0.71	0.77	0.74	0.76
neutri	0.65	0.67	0.66	0.69	0.77	0.73

6 Conclusioni e Futuri sviluppi

Esaminando i risultati appena ottenuti, è possibile affermare che il problema di classificazione viene risolto in maniera soddisfacente da tutti i modelli implementati.
Come è possibile notare in Tabella 1 le tecniche di over/undersampling applicate migliorano in media l'accuracy dei classificatori del 6%
Da una più approfondita analisi emerge una maggiore difficoltà nel classificare istanze appartenenti alla classe neutra rispetto alle restanti, rendendo piuttosto contenuta l'accuracy globale. In Tabella 3 e Tabella 2 infatti la classe neutra possiede valori di *precision*, *recall* e *f1-score* più bassi.
Questo fatto è giustificato dal fatto che le recensioni o commenti sono raramente caratterizzati da una completa neutralità, tanto che possono esistere testi etichettati come neutri ma con diversi gradi di positività o negatività.

Un futuro progetto potrebbe quindi cercare di migliorare le prestazioni di classificazione proprio su quest'ultima categoria, applicando l'ottimizzazione degli iperparameteri o ricercando diverse tecniche e/o algoritmi.

References

[1] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," 2016.

[2] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[3] A. De Mauro, *Big data analytics: guida per iniziare a classificare e interpretare dati con il machine learning*, 8th ed. Milano: Apogeo, 2019.

[4] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning - From Theory to Algorithms.* Cambridge University Press, 2014.

[5] V. V. Cortes Corinna, "The random subspace method for constructing decision forests," *Machine Learning*, vol. 20, no. 3, pp. 1573–0565, 1995.

[6] A. Ben-Hur, D. Horn, H. Siegelmann, and V. Vapnik, "Support vector clustering," *Journal of Machine Learning Research*, vol. 2, pp. 125–137, 11 2001.