Università degli Studi di Salerno

Dipartimento di Informatica

Esame di Reti Geografiche: Struttura, Analisi e Prestazioni

ClickBait Detection with Machine Learning

Docente

Prof.ssa Delfina Malandrino

Studenti

Cardaropoli Giuseppe 0522501310

Fasulo Antonio 0522501312

Anno Accademico 2021/2022

Indice

1 Introduzione ed Obiettivi

1.1 Problema

Oggi giorno esistono molte testate giornalistiche che fondano il loro guadagno sulla pubblicazione di fake news o articoli con titoli clickbait. Molti utenti, per pigrizia o per far avvalere i propri ideali, non approfondiscono le notizie lette e danno tutto per vero. Inoltre, le testate giornalistiche pubblicano articoli con titoli clickbait per poter ottenere un maggior numero di visualizzazioni così da massimizzare il guadagno. Quindi, le domande che ci siamo posti sono le seguenti:

- *"C'è un modo per individuare se il titolo di un articolo è clickbait o meno?"*.
- *"Quali sono le testate giornalistiche che pubblicano il maggior numero di articoli con titoli clickbait?"*

1.2 Fake news vs Clickbait

Bisogna fare distinzione tra fake news e clickbait. Con il termine ***Fake news*** indichiamo articoli di giornale o pubblicazioni sui social network che contengono informazioni inventate, ingannevoli o distorte. Il loro obiettivo è quello di disinformare, creare scandali oppure attirare click. Un esempio di fake news è la figura 1.

Figura 1: Esempio fake news

Invece, secondo l'**Oxford Dictionary**, con il temine ***Clickbait*** indichiamo qualsiasi "contenuto il cui scopo principale è attrarre l'attenzione e spingere i lettori a cliccare sul link di una determinata pagina web". La pratica del clickbaiting è attuata da giornalisti e creator dei social network. Questi pubblicano contenuti, di qualsiasi tipo, con titoli accattivanti, così da "rubare" click ai competitor e massimizzare i guadagni. Un esempio di articolo clickbait è la figura 2

Figura 2: Esempio articolo con titolo clickbait

Quindi quello che differenzia un clickbait da una fake news è il contenuto. Nelle fake news il contenuto è puramente inventato. Invece, nel clickbait il contenuto è veritiero ma non coerente con il titolo.

1.3 Stato dell'arte

Datasets: in letteratura sono stati creati diversi dataset sia di fake news che articoli clickbait. Un esempio sono i seguenti:

- **Clickbait Dataset**: `https://www.kaggle.com/amananandrai/clickbait-dataset`
- **Fake News Dataset**: `https://www.kaggle.com/c/fake-news/data?select=train.csv`

Il primo si riferisce solo ed esclusivamente ai clickbait mentre il secondo è orientato verso le fake news.

Articoli: in letteratura troviamo diversi articoli sulle tematiche citate prima, però, tra quelli analizzati abbiamo preso in considerazione i seguenti articoli:

- "FakeDetector: Effective Fake News Detection with Deep Diffusive Neural Network"
- "Clickbait Detection using Multiple Categorization Techniques"

1.4 Obiettivi

Lo scopo del progetto è quello di progettare un modello di machine learning in grado di riconoscere, con un elevata accuratezza, se il titolo di un articolo è clickbait o meno. Per raggiungere questo obiettivo verranno seguiti questi step:

1. Allenare e testare diversi modelli di machine learning (Dummy Classifier, Naive Bayes, Random Forest, SVM, Logistic Regression e XGBoost) nel classificare titoli di articoli in "clickbait" e "non-clickbait". Tra i modelli verrà scelto il migliore, ovvero quello più accurato.

2. Progettare un nostro modello. Allenarlo e testarlo utilizzando la stessa metodologia dello step precedente;

3. Confrontare il nostro modello (step 2) con il miglior competitor (step 1). Il migliore sarà scelto considerando l'accuratezza;

4. Testare il migliore dei modelli su articoli pubblicati dalle più importanti testate giornalistiche internazionali. Ne sono state selezionate 30. Tra queste spiccano il Wall Street Journal, The New York Times, The Guardian e cosi via. In questo modo individueremo quali sono le testate giornalistiche che pubblicano il maggior numero di articoli con titoli clickbait.

Nella prima e nella seconda fase utilizzeremo lo stesso dataset labellato, descritto nella sezione 3.

2 Strumenti utilizzati

Tutte le fase di training e testing sono state eseguite sulla medesima macchina con:

- CPU: AMD Ryzen 7 3700;
- GPU: NVIDIA RTX 3070;
- RAM: 32 GB.

Le componenti hardware che ha influenzato maggiormente i risultati ottenuti sono la GPU, perché abbiamo allenato e testato il nostro modello sfruttando i suoi cuda core, e la RAM, perché ci ha permesso di lavorare con batch di campioni più grandi.
Utilizzare la stessa configurazione ci ha permesso di non falsificare i risultati ottenuti. Come linguaggio di programmazione è stato utilizzato Python, precisamente la versione 3.8.10. Abbiamo scelto questo linguaggio poiché ci sono moltissime librerie di supporto per il machine learning, come ad esempio TensorFlow, Scikit-learn, Pandas e NumPy.
Invece, come IDE per lo sviluppo abbiamo utilizzato PyCharm. Tutti gli script, i risultati ottenuti, il dataset e le librerie utilizzate si possono trovare nella repository GitHub del progetto al seguente link: `https://github.com/findSaul0/Clickbait-Detection`.

3 Dataset Utilizzato

3.1 Struttura del Dataset

Il dataset utilizzato in questo progetto è il seguente: `https://www.kaggle.com/amananandrai/clickbait-dataset`. Tale dataset contiene 32000 campioni, ovvero titoli di articoli già labellati in clickbait e non-clickbait. A questo dataset sono stati aggiunti ulteriori 20000 campioni appartenenti a questo dataset: `https://zenodo.org/record/5530410/files/clickbait17-train-170630.zip?download=1`. Il tutto per un totale di circa 52000 campioni. In questa fase preliminare il dataset contiene solo due colonne:

- **headline**: il titolo dell'articolo;
- **clickbait**: un'etichetta che vale 1 se l'articolo è clickbait, 0 altrimenti.

3.2 Feature Engineering

È stata svolta un'attività di **Feature Engineering** per individuare delle feature che potevano essere utili durante la classificazione dei titoli in clickbait e non-clickbait. Le feature individuate sono le seguenti:

- **question**: vale 1 se il titolo è una domanda, ovvero contiene ? e/o inizia con una **question word** (le 5 W della lingua inglese), 0 altrimenti;
- **exclamation**: vale 1 se il titolo contiene un punto esclamativo, 0 altrimenti;
- **starts_with_num**: vale 1 se il titolo inizia con un numero, 0 altrimenti;
- **headline_words**: il numero di parole del titolo.

Queste feature sono state calcolate per ogni campione del dataset e sono state aggiunte a quest'ultimo come colonne. Le feature che abbiamo considerato sono alcune delle 19 feature considerate in questo lavoro: "Ensemble Learning Approach for Clickbait Detection Using Article Headline Features".

3.3 Data Cleaning

Prima di utilizzare il dataset si è ritenuto opportuno effettuare una procedura di "pulizia" del testo.

1. I titoli sono stati convertiti in lowercase;
2. Sono state rimosse dai titoli le stopwords, ovvero tutte quelle parole che non hanno particolare importanza all'interno del testo. Tali parole, sono anche scartate dai motori di ricerca. Ovviamente, essendo i titoli in lingua inglese, abbiamo considerato le stopwords inglesi;
3. Segni di punteggiatura, numeri, link ed altri caratteri non-alfabetici sono stati rimossi.

Il dataset finale è il seguente:

	headline	clickbait	question	exclamation	starts_with_num	headline_words
0	trey gowdy just humiliated adam schiff in fron...	1	0	0	0	10
1	60 netflix titles leaving in july 2020	1	0	0	1	7
2	learn how to make a green grape taste like a j...	1	0	1	0	22
3	the new july netflix titles are here and there...	1	0	0	0	13
4	the courts say sex discrimination laws protect...	1	0	0	0	19
...	...	...	...	...	...	...
52167	to make female hearts flutter in iraq throw a ...	0	0	0	0	10
52168	british liberal democrat patsy calton 56 dies ...	0	0	0	0	9
52169	drone smartphone app to help heart attack vict...	0	0	0	0	12
52170	netanyahu urges pope benedict in israel to den...	0	0	0	0	9
52171	computer makers prepare to stake bigger claim ...	0	0	0	0	9

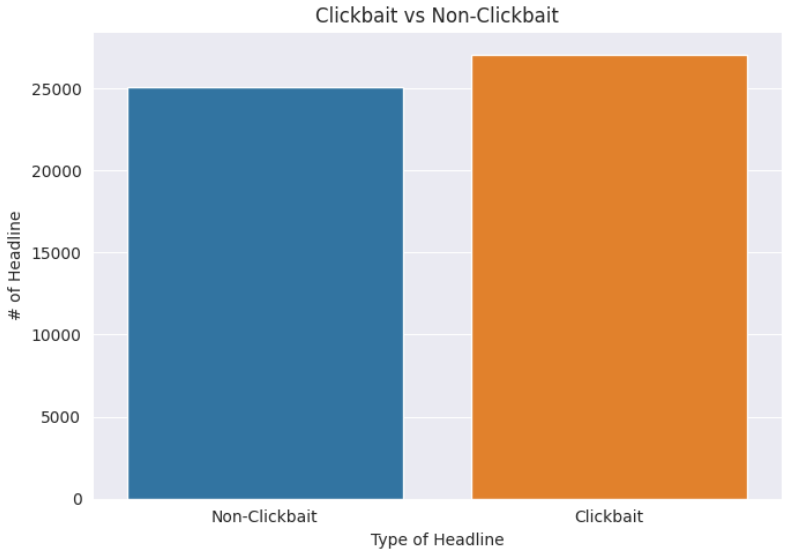
Si tratta di un file con estensione .csv di cui Sono mostrati i primi e gli ultimi 5 campioni con il relativo testo, label e features.

4 Exploratory Data Analysis

Una volta effettuato il feature engineering e il data cleaning del dataset abbiamo effettuato un analisi esplorativa di quest'ultimo così da individuare: la distribuzione dei campioni in base alle classi, la distribuzione in base alla feature e le parole più comuni.

4.1 Distribuzione in classi

Nel seguente grafico è riportata la distribuzione dei campioni nelle classi clickbait e non-clickbait.

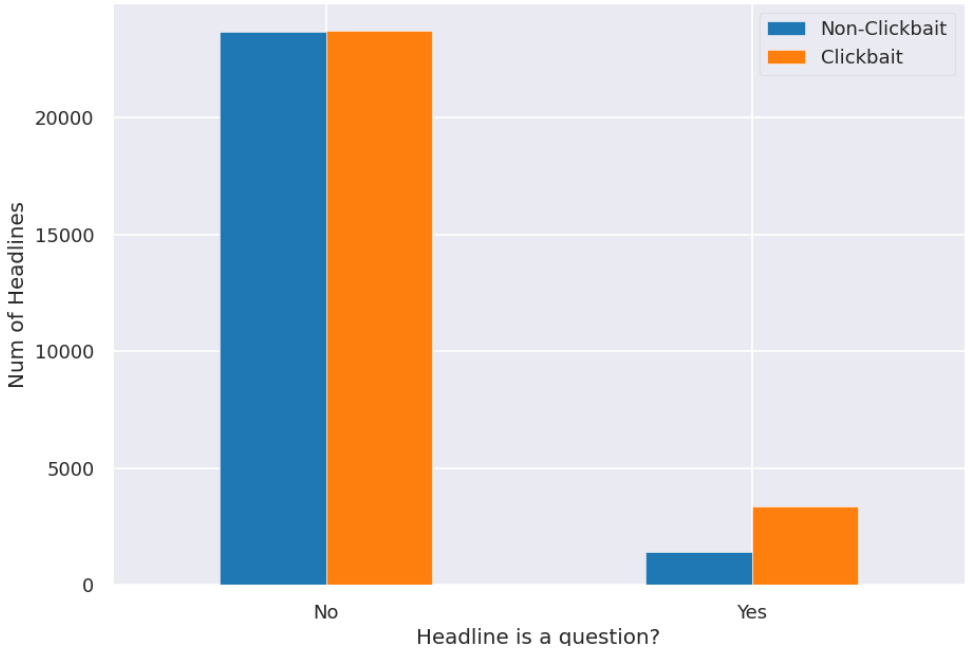


Possiamo notare che la distribuzione è abbastanza equilibrata con una leggera prevalenza di titoli clickbait.

4.2 Distribuzione per feature

4.2.1 Feature "*question*"

Nel seguente grafico a barre è mostrata la distribuzione dei campioni considerando la feature "***question***".



Possiamo notare come ci sono il doppio di titoli clickbait che sono domande rispetto ai titoli non-clickbait.

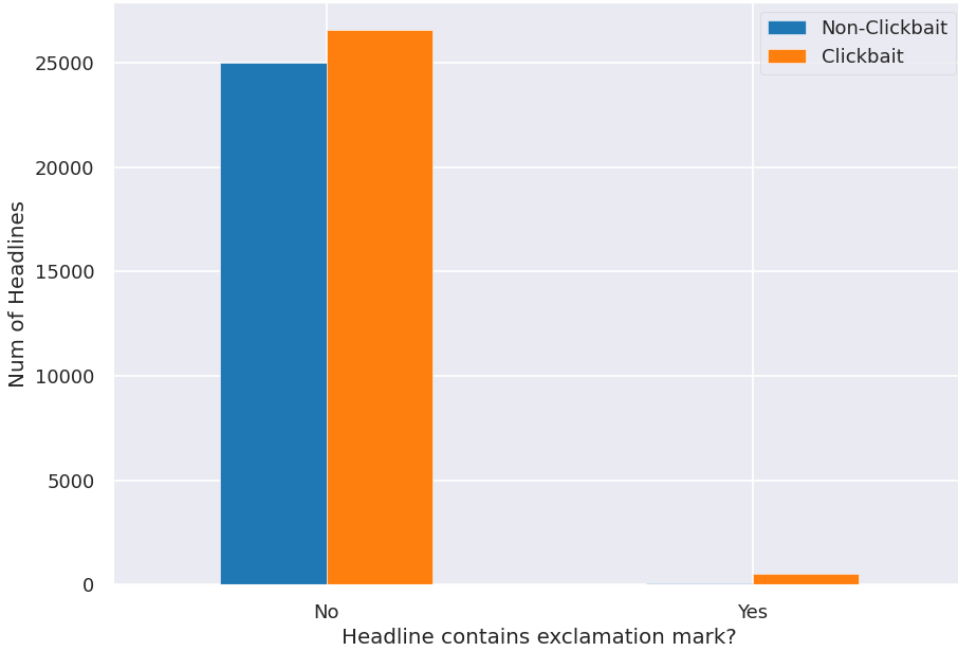
Esempi:

- **Clickbait question**: "*Which Golden Globe Reaction Face Are You Based On Your Zodiac Sign*".
- **Non-Clickbait question**: "*Smartphone From Dell? Just Maybe*".

Ciò che differenzia le question non-clickbait è la presenza di una mini risposta già nel titolo. Inoltre, la maggior parte delle question clickbait sono della forma "Quale X sei in base al tuo Y?". Dove X può essere ad esempio "attore" ed Y "mese di nascita".

4.2.2 Feature "*exclamation*"

Invece, nel seguente grafico a barre consideriamo la feature "***exclamation***".



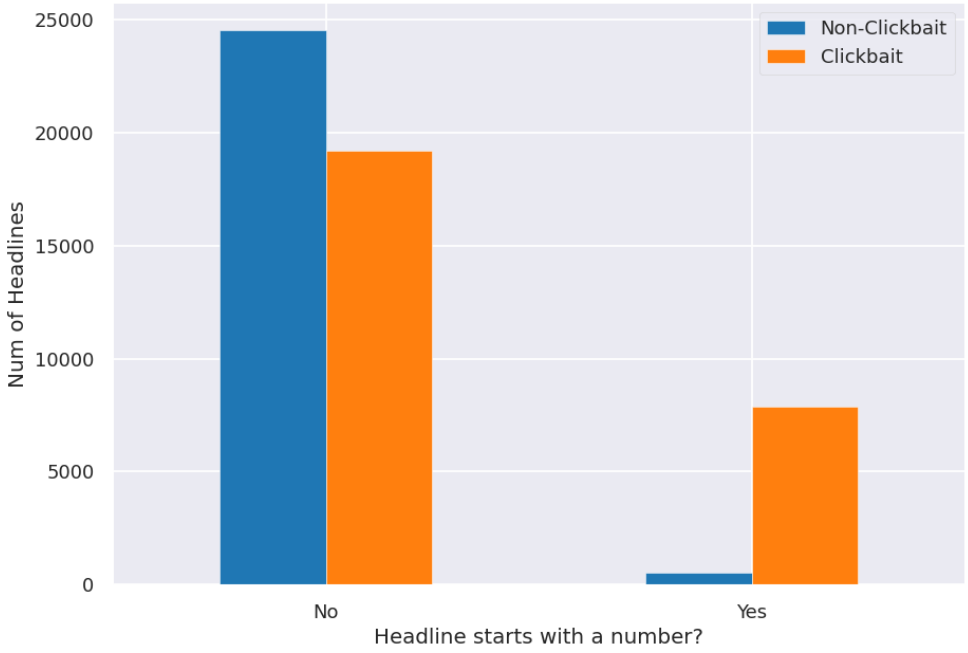
Si può notare che ci sono pochissimi (quasi zero) titoli non-clickbait che sono esclamazioni. Inoltre, ci sono più titoli clickbait che contengono un punto esclamativo rispetto a quelli non-clickbait.

Esempi:

- **Clickbait exclamation**: "*Finally! A Crossword Filled With Star Wars Puns*".
- **Non-Clickbait exclamation**: "*2010 FIFA World Cup: arrivederci Italia!*".

4.2.3 Feature "*starts_with_number*"

In quest'altro grafico a barre mostriamo la distribuzione dei campioni in base alla feature "*starts_with_number*"



Possiamo notare come ci sono molti titoli clickbait che iniziano con un numero, rispetto a quelli non-clickbait. Inversamente, ci sono più titoli non-clickbait che non iniziano con un numero.

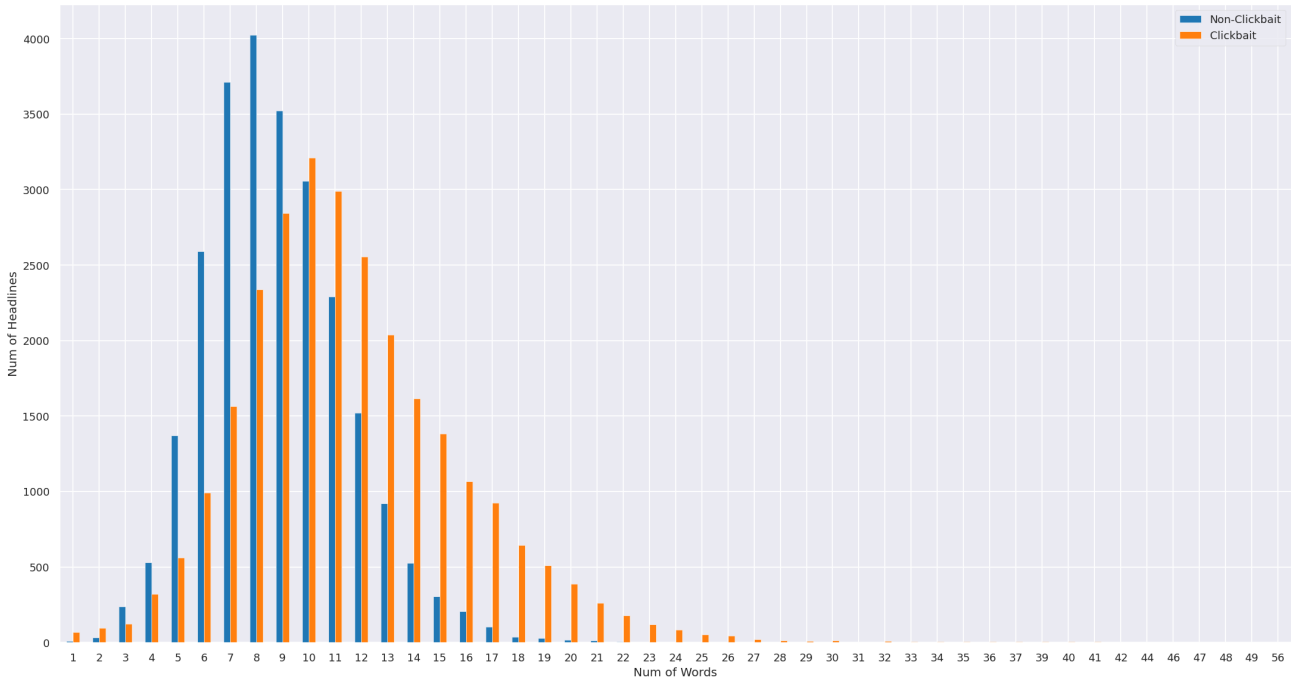
Esempi:

- **Clickbait starts_with_number**: "*17 Pictures Hot People Will Never Understand*".
- **Non-Clickbait starts_with_number**: "*2008 Taipei AMPA: IT industry will pulse the growth of automotive industry*".

I numeri con cui iniziano i titoli non-clickbait sono principalmente date.

4.2.4 Feature "*headline_words*"

Infine, in questo grafico a barre abbiamo considerato la feature "***headline_words***".

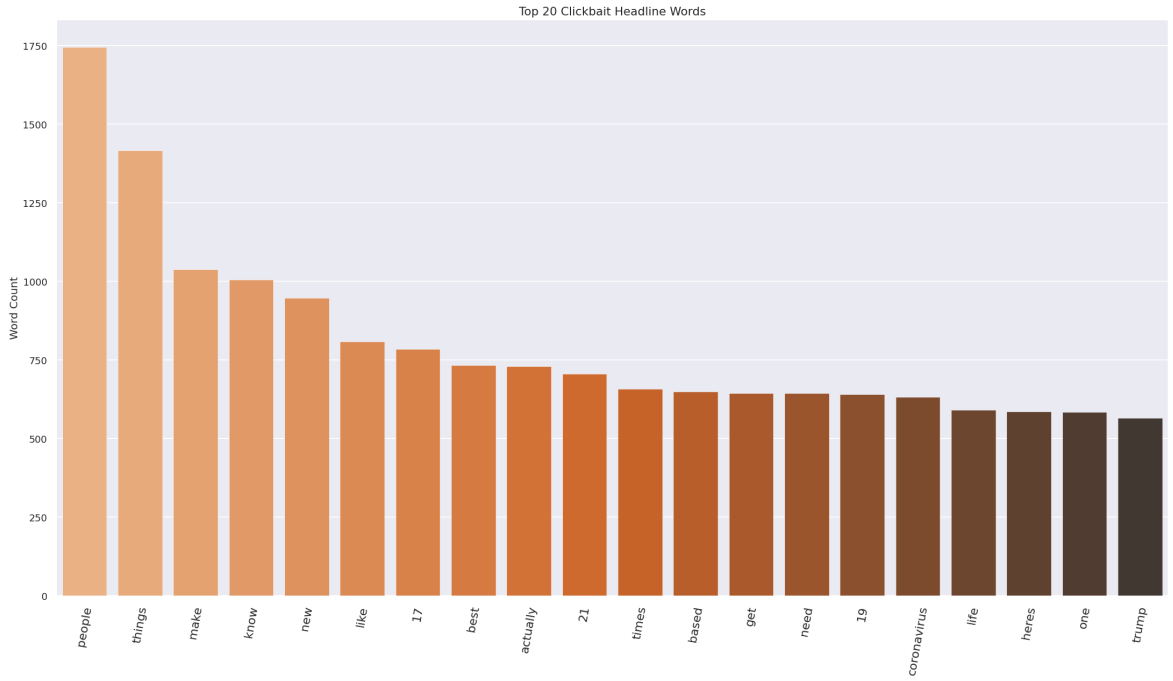


Possiamo notare che per i titoli con lunghezza da 3 a 10 parole sono principalmente non-clickbait. Invece, i titoli con 11 parole sono quasi equamente distribuiti con una piccola prevalenza di titoli clickbait. Da 11 parole in su, man mano che il numero di parole aumenta, aumenta anche il distacco tra clickbait e non-clickbait.

4.3 Statistiche sulle singole parole

4.3.1 Frequenza delle parole nei titoli clickbait

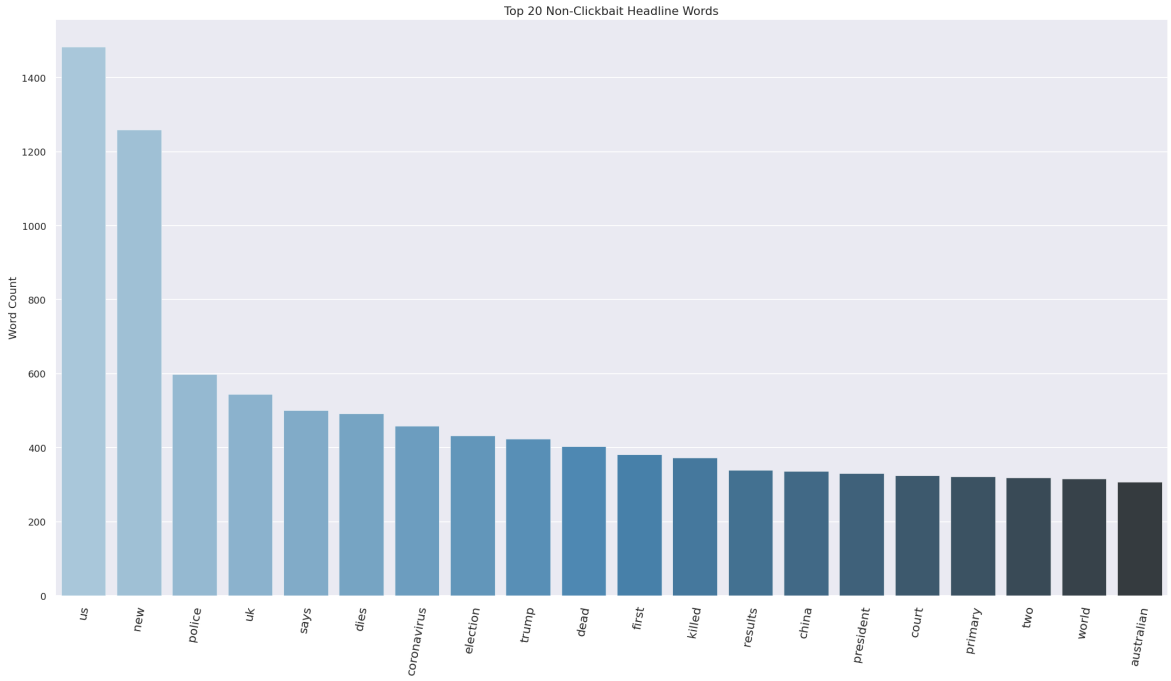
Nella precedente sezione abbiamo analizzato la distribuzione dei titoli nella loro interezza. In questa sezione ci concentriamo sulle singole parole così da individuare quelle più comuni nei titoli clickbait e in quelli non-clickbait.



Nel precedente grafico sono mostrate le 20 parole più frequenti nei titoli clickbait e la loro frequenza. Possiamo notare che le parole più frequenti sono ***people***, ***things*** e ***make***.

4.3.2 Frequenza delle parole nei titoli non-clickbait

Invece, nel seguente grafico sono mostrate le 20 parole più frequenti nei titoli non-clickbait e la loro frequenza.

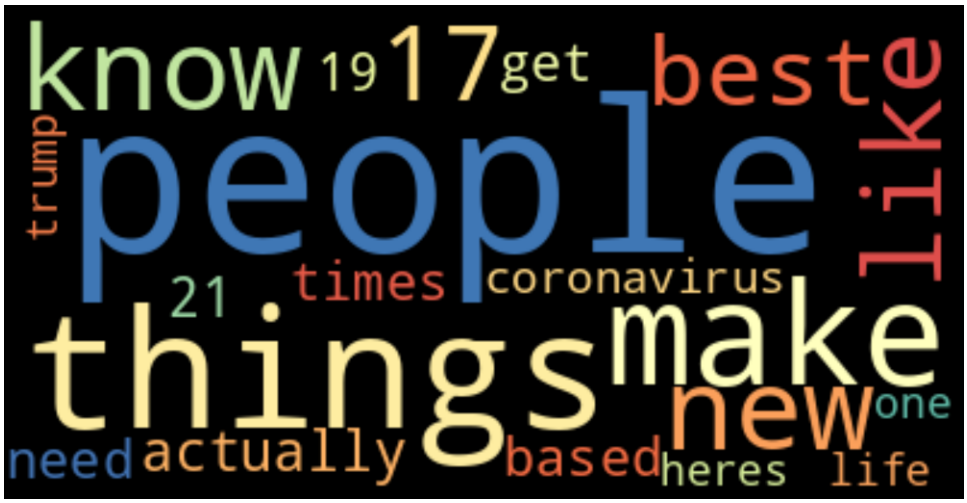


Le parole più frequenti sono ***us***, ***new*** e ***police***. Inoltre, si può notare come alcune parole siano in entrambe le top 20 ma con frequenza diversa. Ad esempio, le parola ***coronavirus*** è al 7° posto tra le non-clickbait ed al 17° tra le clickbait, oppure la parola ***trump*** è al 9° posto tra le non-clickbait ed al 20° tra le clickbait.

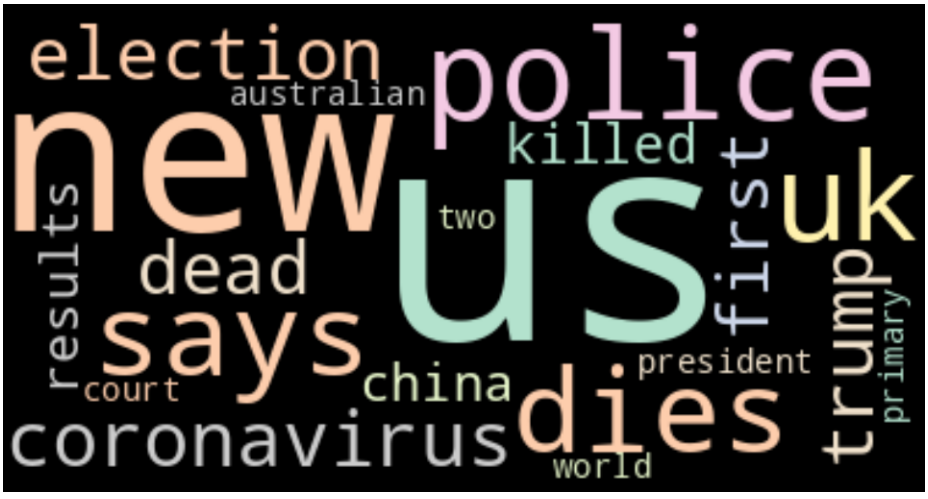
4.4 Word Clouds

Infine, di seguito sono riportare due word cloud. Mostrano la frequenza delle parole nei titoli clickbait e non-clickbait. Più è grande la parola maggiore sarà la sua frequenza.

Word Cloud Clickbait



Word Cloud Non-Clickbait



5 Confronto dei modelli

dataset precedente descrito sono stati addestrati e testati i seguenti modelli: Dummy Classifier, Naive Bayes, Random Forest, SVM, Logistic Regression e XGBoost.
Per ognuno di questi modelli abbiamo diviso il dataset in due set: training set e test set. Il training set conteneva il 80% dei campioni, invece il restante 20% fa parte del test set. Per ognuno dei modelli abbiamo calcolato l'accuratezza e prodotto la matrice di confusione. La matrice di confusione mostra il numero di:

- **Veri negativi**: titoli non-clickbait classificati come non-clickbait;
- **Falsi positivi**: titoli non-clickbait classificati come clickbait;
- **Falsi negativi**: titoli clickbait classificati come non-clickbait;
- **Veri positivi**: titoli clickbait classificati come clickbait.

5.1 Dummy Classifier

Un **Dummy Classifier** non è un vero e proprio classificatore, in quanto non utilizza le feature dei dati per classificarli, bensì adotta una "strategia". Infatti, la parola "***dummy***" significa "***fittizio***". Alcune strategie sono:

- **Più frequente**: viene predetta sempre la classe con maggiore frequenza. Questa è la strategia che abbiamo utilizzato;
- **Uniforme**: le previsioni avvengono uniformemente a caso;
- **Costante**: viene predetta sempre la stessa classe.

Questo classificatore viene utilizzato semplicemente come confronto rispetto per gli altri. Infatti, nessun classificatore dovrebbe essere meno accurato di questo.

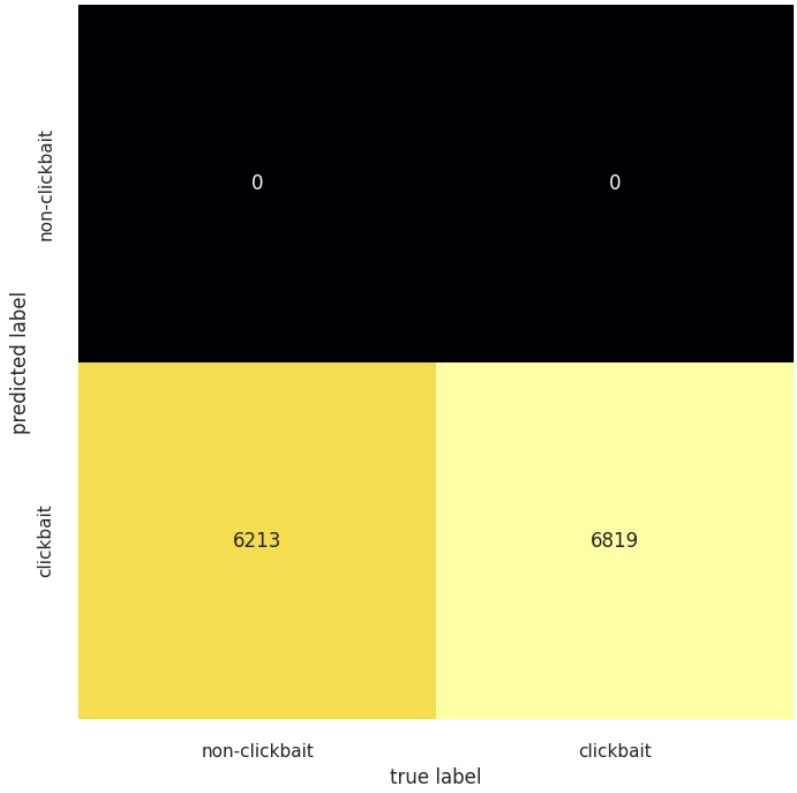


Figura 3: Matrice di confusione Dummy Classifier

L'accuratezza del Dummy Classifier è del **52.3%**.

5.2 Naive Bayes

I classificatori bayesiano sono una famiglia di classificatori che basano il loro funzionamento sul **teorema di Bayes**. Quest'ultimo è utilizzato per calcolare la probabilità di un evento conoscendo la probabilità di un altro evento che è accaduto. Nei classificatori bayesiani viene calcolata l'appartenenza di un campione ad una classe dato il vettore di feature.
Un'importante assunzione che viene fatta è che le feature danno lo stesso contributo. Il **Naive Bayes** è un classificatore binario che fa un'ulteriore assunzione "ingenua" (naive), ovvero che le feature sono indipendenti tra di loro e non si influenzano. Di seguito è riportata la matrice di confusione.

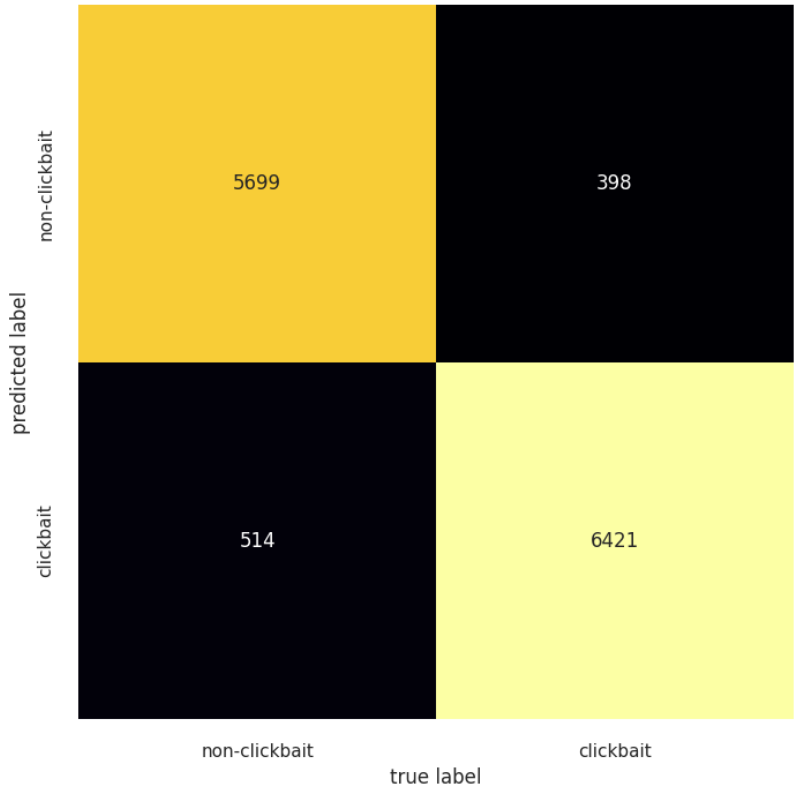


Figura 4: Matrice di confusione Naive Bayes

L'accuratezza del Naive Bayes è del **93%**.

5.3 Random Forest

Le **Random Forest** sono un modello di apprendimento d'insieme. Possono essere utilizzate sia per la classificazione che la regressione. Una random forest è data dall'aggregazione, tramie bagging, di **alberi decisionali**. Un albero decisionale è un grafo di decisioni e delle possibili conseguenze. Può essere utilizzato per classificare: viene fornito alla radice un campione in input, questo attraverserà tutto l'albero fino a raggiungere una foglia la cui etichetta indicherà la classe. Invece, in una random forest lo stesso campione viene dato in input a tutti gli alberi della foresta. Il risultato è dato dalla maggioranza dei risultati dei singoli alberi. Di seguito è riportata la matrice di confusione.

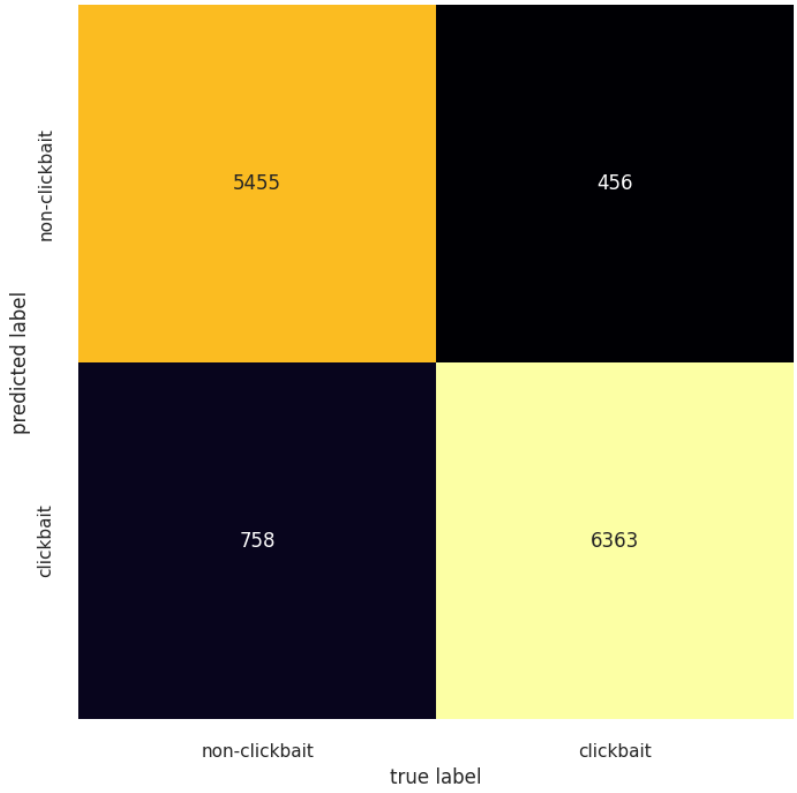


Figura 5: Matrice di confuzione Random Forest

L'accuratezza del Random Forest è del **90.6%**.

5.4 SVM

L'algoritmo **Support Vector Machine** è usato nel machine learning supervisionato per risolvere problemi di classificazione e di regressione. Questi sono detti macchine a vettori di supporto. Il processo di addestramento riceve in input un dataset di training composto da molti dati, nel nostro caso frasi clickbait. Ogni riga di questo dataset è un dato identificato da un vettore x_i con n caratteristiche, nel nostro caso le features sono 4 (question, exclamation, start_with_num, headline_words), e da un'etichetta (labels) che nel nostro caso rappresenta l'appartenenza della frase analizzata al gruppo di frasi clickbait o al gruppo di frasi non-clickbait.
L'algoritmo attraverso l'utilizzo di una specifica coppia di oggetti **(X,y)**, dove X rappresenta una matrice composta dai vettori x_i e y rappresenta il vettore contenente le etichette (label), individua un iperpiano in grado di dividere il set di dati in due classi (clickbait

e non-clickbait). Dobbiamo dire che ogni vettore x_i è l'insieme delle caratteristiche di un dato e individua un punto nello spazio a n dimensioni. Il confine che separa le classi è detto **confine decisionale (decision boundary)**.
Nel nostro caso dato che le caratteristiche analizzate sono numerose l'iperpiano generato dall'algoritmo sarà un piano generato nello spazio multidimensionale. Una questione importante da specificare è il fatto che i punti più vicini all'iperpiano sono maggiormente soggetti all'aggiunta dei nuovi dati, inoltre una piccola variazione dell'iperpiano può modificare la loro classificazione. Invece, i punti più lontani dall'iperpiano hanno maggiore probabilità di essere classificati correttamente dall'algoritmo. Per questo ragione sono detti **support vector**. Di seguito possiamo visualizzare la matrice di confusione ricavata da tale modello.

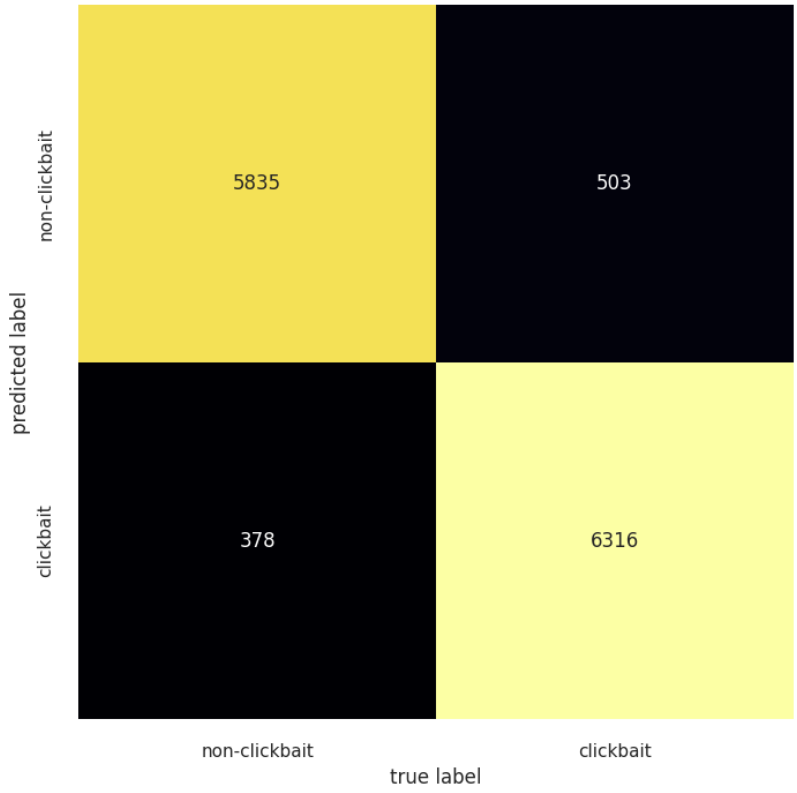


Figura 6: Matrice di confusione SVM

L'accuratezza ricavata del Support Vector Machine è del **93.2%**.

5.5 Logistic Regression

Anche il modello di **Logistic Regression** è assimilabile ad un classificatore binario. Il Logistic Regression calcola la somma pesata (**wighted sum**) delle feature, con l'aggiunta di un termine denominato **bias**, ma il risultato è processato da una **funzione logistica** invece di essere restituito direttamente. La funzione logistica è un caso particolare, appartenente a una classe di funzioni chiamati sigmoidi. Infatti si tratta di particolari funzioni con un andamento a 'S'. Queste funzioni potemmo definirle come dei "mappatori": dato un qualsiasi input, l'output sarà sempre compresa tra due valori (clickbait e non-clickbait). Visualizziamo adesso la matrice di confusione ricavata dalla Logistic Regression.

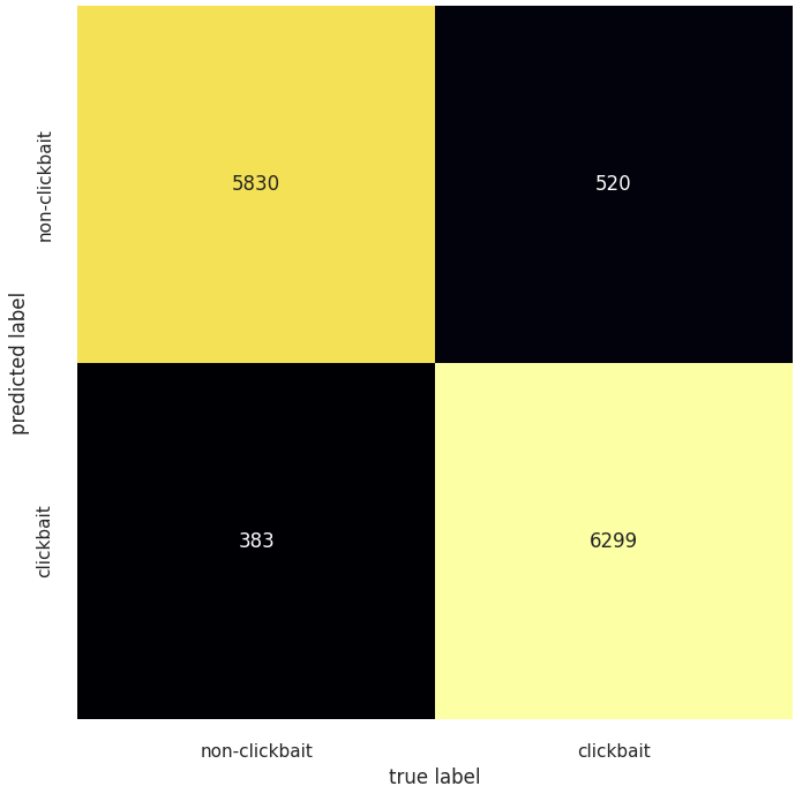


Figura 7: Matrice di confusione del Logistic Regression

L'accuratezza ricavata dal modello Logistic Regression è del **93%**.

5.6 XGBoost

XGBoost è un algoritmo di machine learning basato su un isieme di algoritmi di apprendimento ad albero decisionale e utilizza un framework per il **gradient boosing** (il processo

di gradient boosting riduce al minimo gli errori attraverso l'algoritmo di discesa del gradiente.). L'algoritmo XGBoost è il risultato di un progetto di ricerca organizzato presso l'Università di Washington. Carlos Guestrin e Tianqi Chen hanno presentato il loro lavoro alla conferenza SIGKDDD del 2016. Da allora, XGBoost è diventato un rinomato algoritmo che ha rivoluzionato il panorama dell'apprendimento automatico.
XGBoosting è una versione significativamente più efficiente del gradient boosting, da cui il suo nome: Extreme Gradient Boosting è una combinazione ideale di tecniche di ottimizzazione hardware e software per ottenere risultati superiori utilizzando risorse di calcolo minime in brevi periodi.Visualizziamo adesso la matrice di confusione ricavata dall'algoritmo XGBoost.

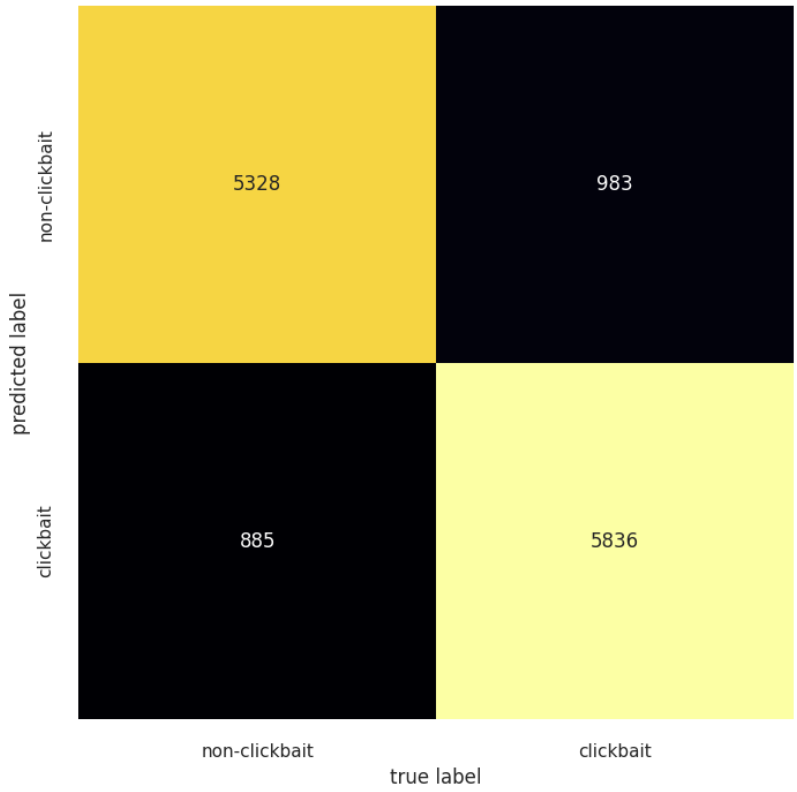


Figura 8: Matrice di confusione del XGBoost

L'accuratezza del modello XGBoost si aggira intorno al **88.3%**

5.7 Risultati del confronto

Nella seguente tabella sono riportati per ogni modello i valori di:

- **accuracy**: indica con quanta probabilità il modello fornisce una previsione corretta;
- **precision**: la percentuale di veri positivi classificati correttamente;
- **recall** (**sensibility**): la percentuale di veri positivi sul totale di tutti i possibili positivi;
- **F1 score**: una metrica calcolata a partire dalla precision e dalla recall in questo modo

$$F1 = 2 \times \frac{precision \times recall}{precision + recall}$$

	Accuracy	**Precision**	**Recall**	**F1 Score**
Dummy Classifier	0.523250	0.523250	1.0	0.687018
Naive Bayes	0.930018	0.925883	0.931633	0.933692
Random Forest	0.906844	0.893333	0.933421	0.912937
SVM	0.932320	0.943523	0.926088	0.934724
Logistic Regression	0.930709	0.942681	0.923742	0.933116
XGBoost	0.883412	0.868323	0.855843	0.862038

Possiamo notare come SVM sia il migliore modello. Infatti, possiede i valori di accuracy, precision, recall e F1 score maggiori rispetto a tutti gli altri.

6 Implementazione di una rete neurale LSTM

Le implementazioni dei precedenti modelli sono delle implementazioni standard fornite dalla libreria Scikit-learn. Per questo motivo abbiamo deciso di cercare un modello più adatto alla classificazione di titoli in clickbait e non clickbait. L'obiettivo è quello di ottenere un'accuratezza maggiore rispetto al migliore dei precedenti modelli, ovvero SVM con un'accuratezza del **93.2%**. La nostra attenzione si è concentrata sulle reti neurali, in particolar modo sulle reti neurali LSTM. Per implementare tale modello abbiamo utilizzato la libreria Tensorflow. In questo caso i campioni sono stati divisi in:

- **Training set**: contiene l'80% dei campioni. Questo set verrà utilizzato per allenare il modello;
- **Testing set**: contiene il 10% dei campioni. Questo set verrà utilizzato per testare il modello su campioni differenti da quelli con cui è stato allenato;
- **Validation seti**: contiene il 10% dei campioni. Questo set verrà utilizzato per validare il modello e controllare la sua accuratezza dopo ogni epoca del training.

Questa suddivisione è necessaria per allenare il modello ed evitare che questo vada il **overfitting**, ovvero che si adatti ai campioni su cui è stato allenato. Un modello che va in overfitting lavora molto bene (fin troppo) con i campioni su cui è stato allenato ma molto male su nuovi campioni.

6.1 Reti RNN e LSTM

Le reti neurali furono progettate intorno alla metà del secolo scorso, con l'intento di simulare, per struttura e comportamenti, il sistema neurale umano composto da un numero considerevole di neuroni. Nelle reti neurali abbiamo dei neuroni computazionali disposti in diversi layer e connessi tra loro con dei collegamenti pesati.
L'input dato alla rete viene processato, layer dopo layer, dai vari neuroni fino ad arrivare all'output layer che restituisce un output. I pesi della rete vengono aggiornati durante l'allenamento andando a propagare l'errore all'indietro (**algoritmo backpropagation**).
La rete neurale proposta è una rete **Long short-term memory** (**LSTM**), ovvero un particolare tipo di **Rete Neurale Ricorrente** (**RNN**).
Iniziamo col spiegare che cos'è una RNN. Una rete neurale ricorrente è un particolare tipo di rete neurale dove i neuroni sono collegati tra loro in un loop. Questa interconnessione permette di utilizzare uno dei layer come memoria di stato. In questo modo il comportamento della rete dipenderà anche dalle informazioni che sono state ricevute in istante di

tempo precedenti. Ciò consente a queste reti di lavorare su sequenze di dati dipendenti tra loro. La figura 9 mostra la struttura di una semplice RNN. Il problema delle RNN è

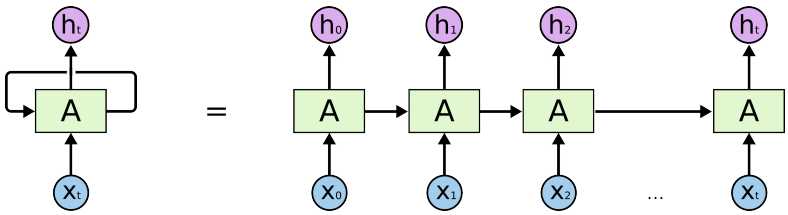


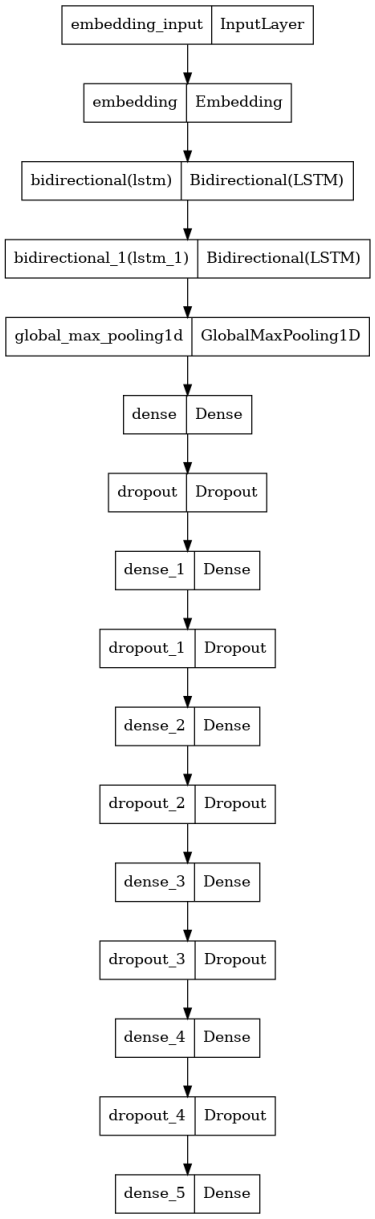
Figura 9: Struttura reti RNN

la loro poca memoria. Non riescono a ricordare informazioni processate molto tempo fa. Questi problemi vengono superati dalle reti neurali LSTM. Queste hanno una **memoria a lungo termine** che permette di ricordare le informazioni per lunghi periodi di tempo.

6.2 Struttura del Modello

La struttura del modello è quella mostrata a sinistra. Come in ogni rete neurale abbiamo:

- **Input Layer**: qui troviamo i neuroni di input che prendono l'input e lo forniscono ai neuroni dello strato successivo;
- **Hidden Layer**: sono una serie di layer "nascosti". I neuroni di questi strati prendono l'input da quelli dell'input layer, lo processano e lo forniscono all'output layer;
- **Output Layer**: qui abbiamo i neuroni di output che forniscono l'output finale.



6.3 Funzionamento

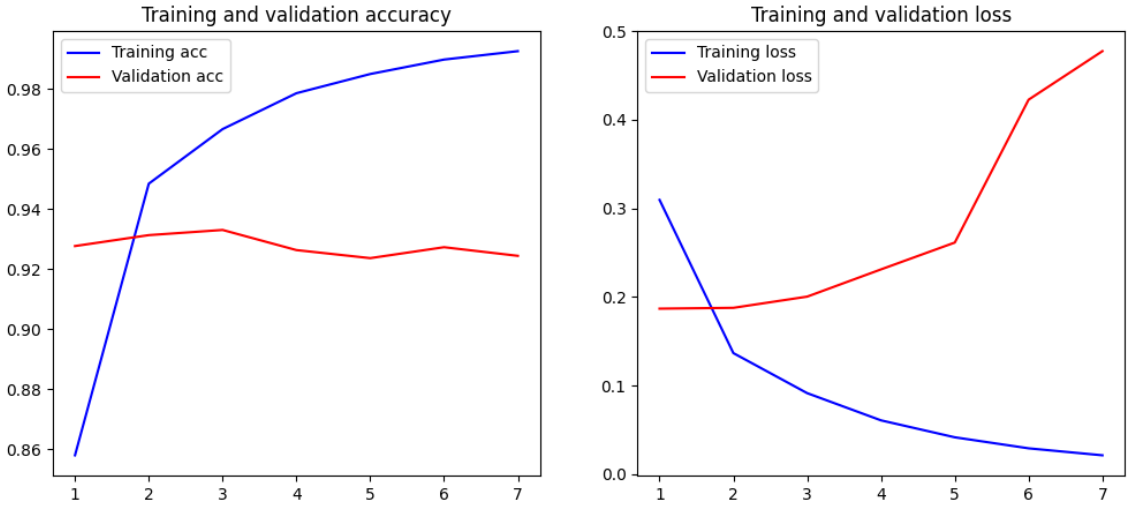
Alla rete neurale verrà fornita una stringa in input opportunamente suddivisa come una sequenza di parole. Questa sequenze verrà processata tenendo memoria di tutte le parole che compongono la frase. La funzione di attivazione utilizzata è la funzione **Sigmoide** che restituisce un valore reale compreso tra 0 ed 1. Se l'output della rete è 1 allora il titolo fornito in input è classificato come clickbait, invece se l'output è 0 allora è classificato come non-clickbait.

6.4 Training

Per il training è stato utilizzato il medesimo dataset con cui sono stati addestrati i precedenti modelli, ovvero quello descritto nella sezione 3. Il modello è stato allenato per 20 epoche con batch di 512 campioni. L'addestramento è avvenuto utilizzando una NVIDIA RTX 3070 che, grazie ai suoi 5888 cuda cores, ha permesso di ridurre notevolmente i tempi. Durante ogni epoca ne è stata calcolata l'accuratezza e la perdita. Invece, al termine di ogni epoca è stata eseguita una fase di validazione del modello. Anche durante la validazione abbiamo calcolato l'accuratezza e la perdita. La validazione è necessaria per controllare le performance del modello non sui dati di training e prima del testing. Inoltre, sono state definite due **callback** che vengono invocate al termine di ogni epoca:

- **EarlyStopping**: permette di terminare il training prematuramente nel caso in cui l'accuratezza non migliora considerevolmente per un tot di epoche. Questa callback consente di evitare l'overfitting;
- **ModelCheckpoint**: permette di salvare i pesi del modello. Nel nostro caso salviamo i pesi solamente della migliore epoca.

Come **algoritmo di ottimizzazione** è stato utilizzato **Adam**. Si tratta di un estensione dell'algoritmo della discesa del gradiente e permette di aggiornare iterativamente i pesi della rete durante il training.
Invece, la **loss function** considerata è la **binary cross entropy**, che permette di propagare l'errore di classificazione all'interno della rete. Di seguito sono riportati i risultati del training e della validazione:



Nel grafico a sinistra viene confrontato l'andamento dell'accuratezza del modello in fase di training e di validazione. Possiamo notare come al termine del training l'accuratezza raggiunge quasi il 99%, questo perché il modello ha osservato sempre i medesimi campioni ed è diventato sempre più accurato nel classificarli. Invece, per quanto riguarda la validazione, il modello ha mantenuto un accuratezza che si aggirava intorno al 93%.
Invece, nel grafico di destra viene confrontata la perdita del modello in fase di training e di validazione. Possiamo notare come la perdita diminuisce molto velocemente con il passare delle epoche del training.

6.5 Testing Modello

Al termine del training, il modello allenato ed i suoi pesi sono stati salvati per poi essere ricaricati per il testing. Durante il testing sono stati forniti al modello il 10% dei campioni totali, ovviamente si tratta di campioni che non ha mai visto durante le fasi precedenti. Di seguito è riportata la matrice di confusione:

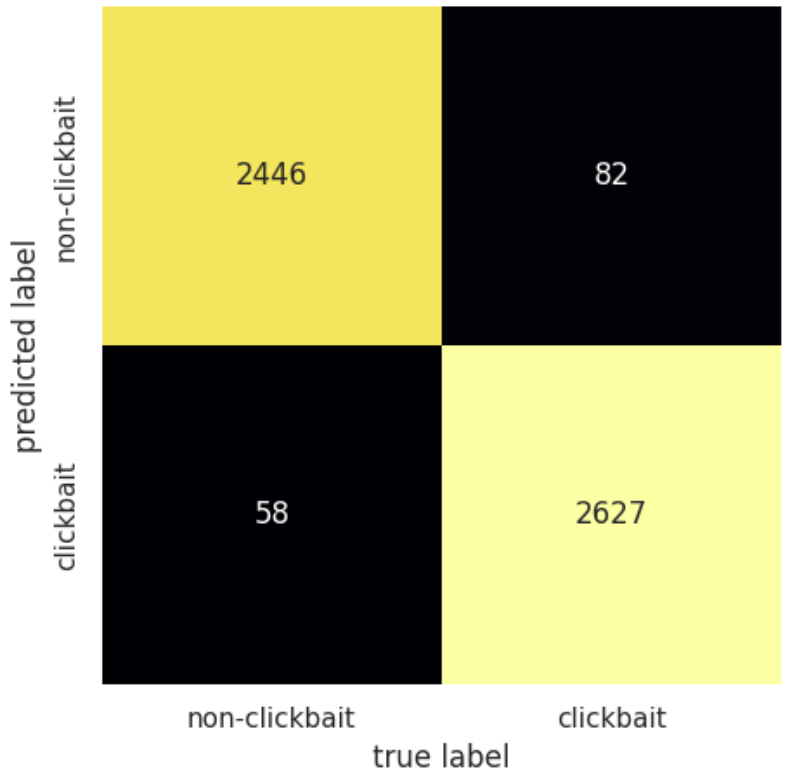


Figura 10: Confusion Matrix LSTM

L'accuratezza del modello è del **97.2%**. Infatti possiamo notare il basso numero di falsi positivi (**82**) e falsi negativi (**58**).

6.6 Considerazioni

Nella seguente tabella sono riportati i valori di accuracy, precision, recall ed F1 score dei modelli confrontati nella sezione 5 e delle reti LSTM.

	Accuracy	**Precision**	**Recall**	**F1 Score**
Dummy Classifier	0.523250	0.523250	1.0	0.687018
Naive Bayes	0.930018	0.925883	0.931633	0.933692
Random Forest	0.906844	0.893333	0.933421	0.912937
SVM	0.932320	0.943523	0.926088	0.934724
Logistic Regression	0.930709	0.942681	0.923742	0.933116
XGBoost	0.883412	0.868323	0.855843	0.862038
Reti LSTM	0.971801	0.979308	0.965863	0.972539

Possiamo notare come le reti LSTM abbiano ottenuto risultati maggiori rispetto a quelli di SVM. L'obiettivo è stato raggiunto, abbiamo individuato un modello migliore di SVM nel classificare titoli di articoli in clickbait e non-clickbait.

7 Testing "*In-The-Wild*"

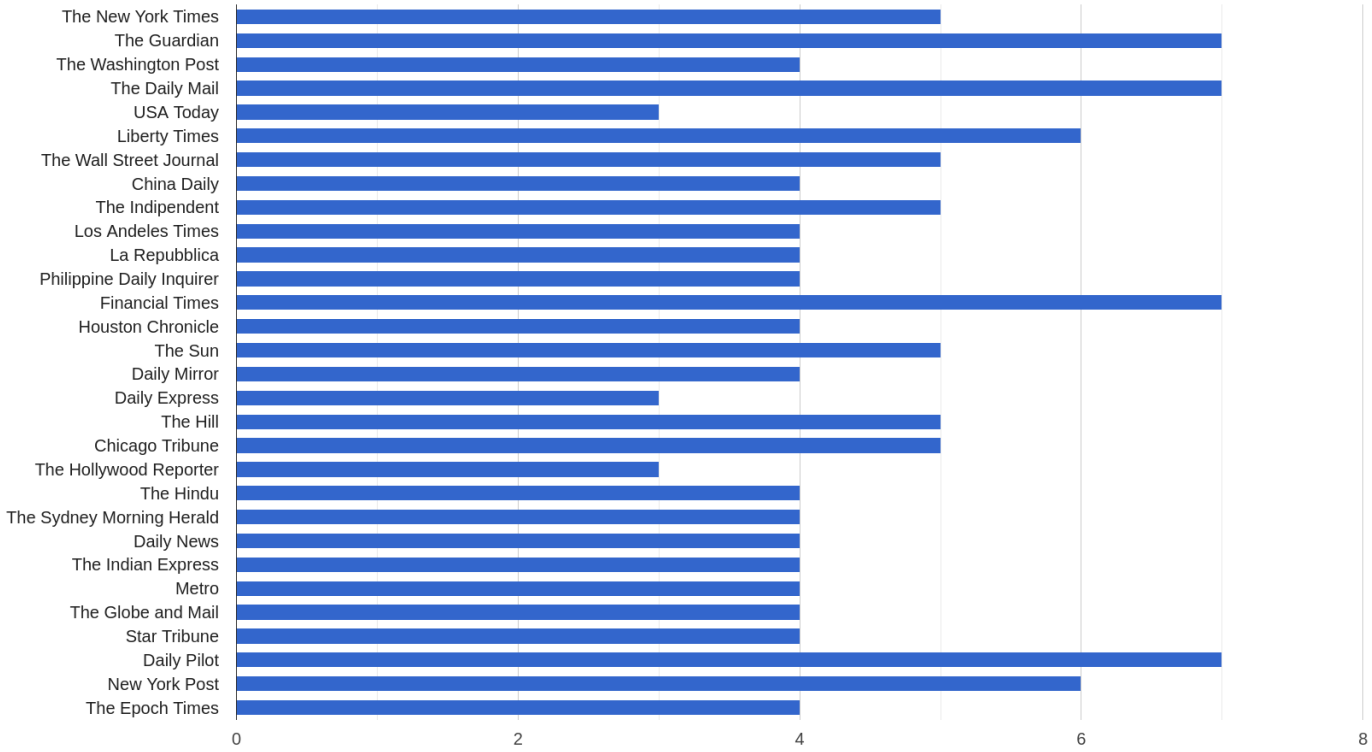
Una volta aver allenato e testato le reti LSTM sul dataset descritto nella sezione 3, abbiamo deciso di adoperare una tipologia di testing chiamata "***In-The-Wild***", così da testare il comportamento delle reti LSTM sul campo.

7.1 Dati Raccolti

Per questa tipologia di testing abbiamo ritenuto necessario costruire un dataset da zero. Sono state selezionate ben 30 testate giornalistiche in lingua inglese. La scelta è avvenuta tenendo conto della **circolazione** di tali giornali. Per ognuna di queste testate sono stati raccolti 10 titoli di articoli presenti sulla pagina principale, quindi per un totale di 300 titoli. La raccolta è avvenuta nel medesimo giorno (18 Dicembre 2021). I principali topic di questi articoli riguardano la variante Omicron, le festività Natalizie e le tensioni tra Russia e Ucraina.
Questi dati così "grezzi" sono stati "raffinati" affinché il dataset avesse le stesse caratteristiche di quello con cui è stato addestrato originariamente il modello. Tutti i titoli sono stati privati delle *stop-words* e di qualsiasi carattere non alfanumerico. Inoltre, i titoli sono stati convertiti in *lowercase*. Infine, per ogni campione sono state calcolate le feature ***question***, ***exclamation***, ***starts_with_num*** e ***headline_words*** (sezione 3.2).

7.2 Risultati Ottenuti

Il nuovo dataset costruito è stato dato in pasto al modello. Per ogni titolo, il modello ha fornito in output 0 (**non-clickbait**) oppure 1 (**clickbait**). Di seguito sono riportati i risultati ottenuti.



Testata Giornalistica	**Articoli clickbait (su 10)**
The New York Times	5
The Guardian	7
The Washington Post	4
The Daily Mail	7
USA Today	3
Liberty Times	6
The Wall Street Journal	5
China Daily	4
The Indipendent	5
Los Angeles Times	4
La Repubblica	4
Philippine Daily Inquirer	4
Financial Times	7
Houston Chronicle	4
The Sun	3
Daily Mirror	4
Daily Express	3
The Hill	5
Chicago Tribune	5
The Hollywood Reporter	3
The Hindu	4
The Sydney Morning Herald	4
Daily News	4
The Indian Express	4
Metro	4
The Globe and Mail	4
Star Tribune	4
Daily Pilot	7
New York Post	6
The Epoch Times	4

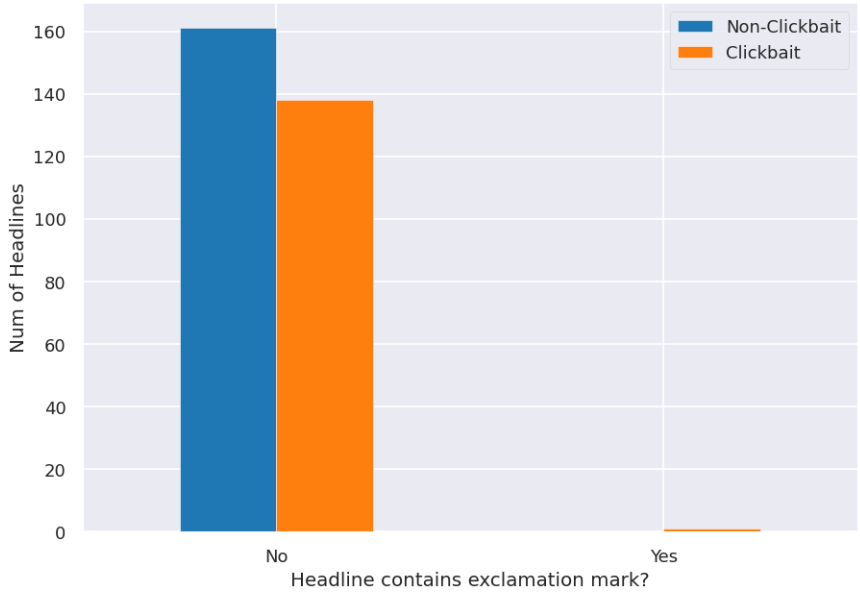
Possiamo notare che in media le riviste considerate hanno 4 articoli con titoli clickbait su 10. Le testate giornalistiche con la maggiore frequenza di articoli con titoli clickbait sono The Guardian, The Daily Mail, Financial Times e Daily Pilot con 7 su 10. Invece, le testate giornalistiche con il minor numero di articoli con titoli clickbait sono USA Today, The Sun,

Daily Express e The Hollywood Reporter con 3 su 10. Ovviamente questi risultati non sono pienamente rappresentativi del fenomeno del clickbaiting poiché bisognerebbe considerare un timescale molto più ampio. Bisognerebbe selezionare ogni giorno e per ogni testata giornalistica 10 articoli per un intero anno. In questo modo potremmo determinare qual è la testata giornalistica che nell'anno X ha pubblicato un maggior numero di articoli con titoli clickbait.

7.3 Analisi dei Risultati

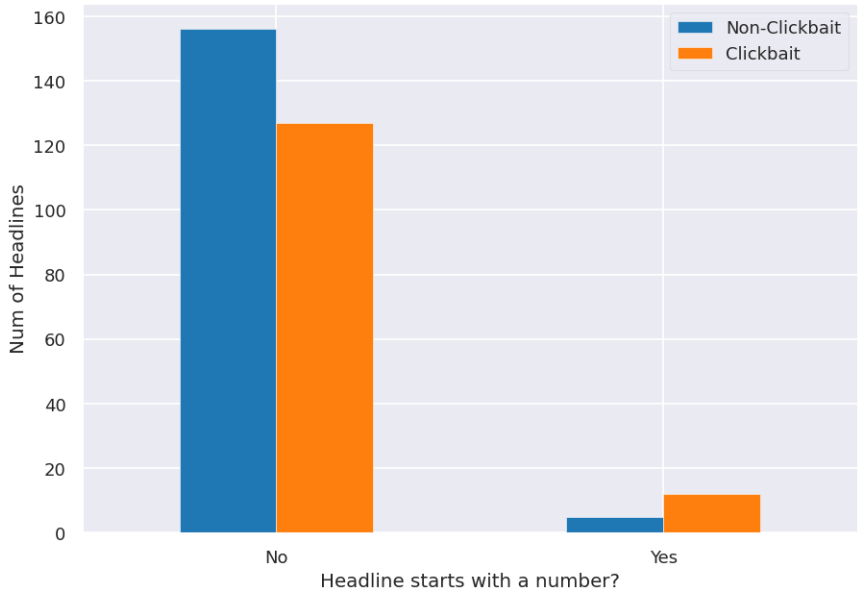
Una volta aver classificato i campioni del testing in-the-wild, li abbiamo analizzati. Di seguito sono riportati diversi grafici che mostrano la distribuzione dei campioni in base al valore delle feature (sezione 3.2).

feature "*exclamation*"



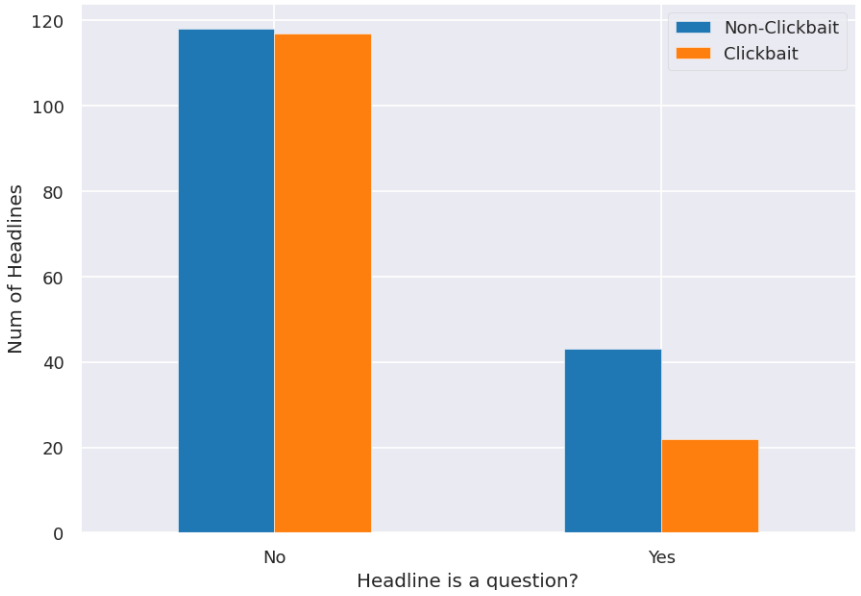
Non possiamo confermare l'utilità della feature *exclamation* ma nemmeno la sua inutilità, questo perché c'è un singolo campione che termina con un punto esclamativo ed è stato classificato come clickbait.

feature "*starts_with_num*"



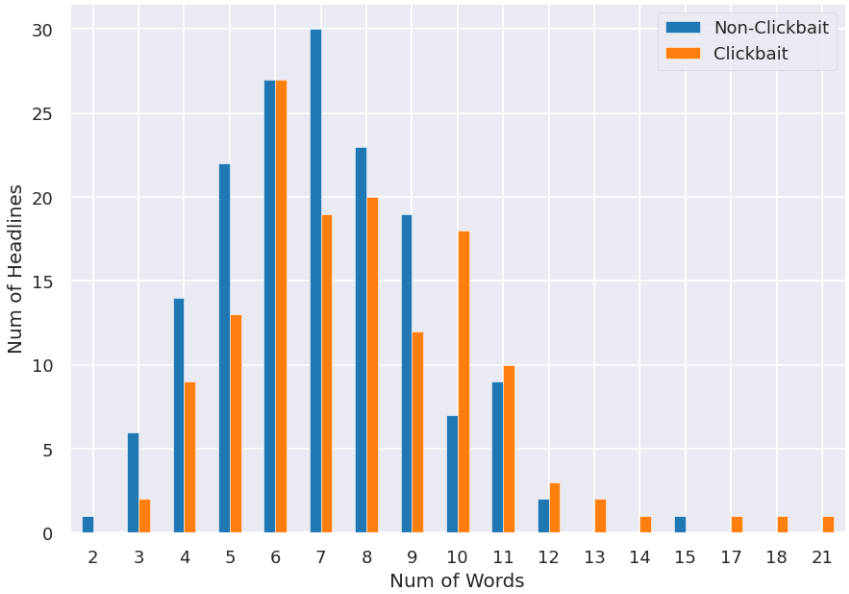
Più del doppio dei titoli che iniziano con un numero sono stati classificati come clickbait. In particolar modo, tutti questi erano articoli nel formato "*X things ...*". Possiamo affermare che questa feature è caratterizzante dei titoli clickbait.

feature "*question*"



In questo caso molti titoli che sono domande sono stati classificati come non-clickbait. Dunque la feature ***question*** non caratterizza i titoli clickbait.

feature "*headline_words*"



Possiamo notare come i titoli con lunghezza pari a 6 si distribuiscono equamente in clickbait e non clickbait. Inoltre, i titoli clickbait risultano essere meno frequenti nel range 3-9. Invece, i titoli con lunghezza maggiore di 10 risultano essere prevalentemente clickbait. Si può dedurre come più sia lungo il titolo maggiore è la probabilità che questo sia clickbait. Sulle medie lunghezze c'è un bilanciamento.

8 Lavori Futuri

Da questo progetto iniziale, in particolar modo dal modello proposit, è possibile sviluppare nuovi progetti. Di seguito sono riportare alcune idee.

8.1 Estensione Browser

Un'estensione browser che è possibile attivare sulla pagina web di una testata giornalistica. Questa riconoscerà automaticamente i titoli presenti nella pagina e mostrerà a schermo con quanta percentuale un determinato titolo può essere clickbait o meno.

8.2 Bot Telegram

Un Bot Telegram in grado di riconoscere se il titolo di un articolo dato in input è clickbait o meno. Il titolo può essere fornito in maniera testuale oppure mediante una foto scattata con lo smartphone. Nel secondo caso verrà prelevato il testo dall'immagine.

8.3 Nuove Feature

Si potrebbe ripetere l'esperimento considerando tutte o altre delle feature utilizzare nel lavoro: "Ensemble Learning Approach for Clickbait Detection Using Article Headline Features".

8.4 Estendere l'ambito

Oltre a considerare gli articoli di giornale potremmo estendere il lavoro considerando qualsiasi contenuto pubblicato sul web. Ad esempio, su Youtube ci sono creator che caricano video con titoli clickbait così da attivare gli utenti. Potrebbe essere interessante classificare i titoli del video Youtube in clickbait o meno. Anche in questo caso si potrebbe progettare un'estensione browser oppure un bot a cui dare in input l'url del video.