



UNIVERSITÀ DEGLI STUDI DI SALERNO

Dipartimento di Informatica

Corso di Reti Geografiche: Struttura, Analisi e Prestazioni

DOCUMENTO DI PROGETTO:

**Analisi delle Recensioni False
Mediante Machine Learning: Sviluppo
di un Bot per la Rilevazione
Automatica**

AUTORI

Agosto Piero

Anzalone Domenico

Abstract

Le recensioni false rappresentano un problema crescente nel mondo digitale, causando distorsioni significative nella percezione del pubblico verso prodotti e servizi. Spesso generate da bot, queste recensioni possono ingannare i consumatori, influenzandone le decisioni di acquisto con informazioni fuorvianti. La necessità di tutelare i consumatori da tali pratiche disoneste ha stimolato lo sviluppo di un modello capace di individuare e distinguere con precisione le recensioni false da quelle autentiche. Per affrontare questa sfida, il nostro modello utilizza tecniche avanzate di elaborazione del linguaggio naturale (NLP) e apprendimento automatico (Machine Learning), analizzando pattern linguistici e comportamenti anomali nelle recensioni. Questo approccio consente un'esaminazione approfondita dei contenuti, identificando con accuratezza quelli potenzialmente falsi. Il modello offre una risposta rapida ed efficace, contribuendo a creare un ambiente online più trasparente e affidabile. Grazie alla sua capacità di distinguere tra recensioni autentiche e false mediante l'utilizzo di queste tecnologie avanzate, aiuta i consumatori a prendere decisioni di acquisto più consapevoli, basate su valutazioni genuine.

Indice

Elenco delle Figure	iii
Elenco delle Tabelle	iv
1 Introduzione	1
1.1 Contesto e Obiettivi	1
1.2 Approccio	2
1.2.1 Algoritmi Utilizzati	2
1.2.2 Scelta dei Modelli	2
1.2.3 Testing	3
1.2.4 Implementazione del Modello	3
2 Stato dell'Arte nella Fake Review Detection	4
2.1 Introduzione alle Fake Reviews	4
2.1.1 Tipologie di Fake Reviews	5
2.2 Attuali approcci alla Fake Review Detection	6
2.2.1 Apprendimento Supervisionato Tradizionale	6
2.2.2 Selezione delle Caratteristiche e Ensemble Learning	6
2.2.3 Approcci basati su reti neurali	7
2.2.4 Reti Neurali Ricorrenti e Generative Adversarial Networks (GAN)	7

3 Metodologia	8
3.1 Introduzione all'approccio	8
3.2 Selezione e Utilizzo dei Tool	10
3.3 Analisi Esplorativa dei Dati	11
3.3.1 Descrizione del Dataset	11
3.3.2 Descrizione delle Features	12
3.4 Preparazione dei dati	14
3.5 Addestramento Modelli	18
3.6 Testing dei Modelli	21
3.7 FakeReviewDetector	22
3.7.1 Tecnologie Utilizzate	22
3.7.2 Implementazione	22
3.7.3 Funzionamento	24
4 Risultati	27
4.1 Introduzione	27
4.2 Risultati	29
4.2.1 Multinomial Naive Bayes	29
4.2.2 Random Forest Algorithm	30
4.2.3 Decision Tree Classifier	31
4.2.4 KNeighbors Classifier	32
4.2.5 Support Vector Classifier Model Results	33
4.2.6 Logistic Regression	34
4.3 Confronto risultati	35
5 Conclusioni e Sviluppi futuri	36
5.1 Sviluppi Futuri	36
Bibliografia	39

Elenco delle figure

3.1	Immagine che illustra le prime righe del dataset "raw_dataset"	11
3.2	Grafico che illustra la distribuzione della feature "category"	12
3.3	Grafico che illustra la distribuzione della feature "label"	13
3.4	Grafico che illustra la distribuzione della lunghezza della feature "text_"	14
3.5	Esempio di creazione di un bot	23
3.6	Funzione che gestisce il comando start	23
3.7	Funzione che gestisce la risposta alle recensioni	23
3.8	Pagina iniziale del bot	24
3.9	Esempio completo di conversazione	25
3.10	Elenco dei comandi del bot	26
4.1	Grafico contenente il confronto tra i modelli	35

Elenco delle tabelle

4.1	Multinomial Naive Bayes Model Results	29
4.2	Confusion Matrix for Multinomial Naive Bayes Model	29
4.3	Accuracy Score for Multinomial Naive Bayes Model	29
4.4	Random Forest Algorithm Model Results	30
4.5	Confusion Matrix for Random Forest Algorithm	30
4.6	Accuracy Score for Random Forest Algorithm	30
4.7	Decision Tree Classifier Results	31
4.8	Confusion Matrix for Decision Tree Classifier	31
4.9	Accuracy Score for Decision Tree Classifier	31
4.10	KNeighbors Classifier Algorithm Model Results	32
4.11	Confusion Matrix for KNeighbors Classifier Algorithm	32
4.12	Accuracy Score for KNeighbors Classifier Algorithm	32
4.13	Support Vector Classifier Model Results	33
4.14	Confusion Matrix for Support Vector Classifier	33
4.15	Accuracy Score for Support Vector Classifier	33
4.16	Logistic Regression Algorithm Model Results	34
4.17	Confusion Matrix for Logistic Regression Algorithm	34
4.18	Accuracy Score for Logistic Regression Algorithm	34

CAPITOLO 1

Introduzione

1.1 Contesto e Obiettivi

Le recensioni dei clienti rappresentano feedback preziosi forniti da coloro che hanno già acquistato e utilizzato un prodotto. Spesso, le persone si affidano ciecamente a queste recensioni per decidere se acquistare o meno un articolo. Tuttavia, si presenta il rischio delle cosiddette ‘recensioni false’ (Fake Reviews), create per incrementare artificialmente la popolarità di un prodotto o, al contrario, per denigrarne la qualità. Questa pratica disonesta, spesso perpetrata da venditori che impiegano terze parti e bot per generare recensioni ingannevoli, induce in errore i consumatori. Di conseguenza, diventa essenziale sviluppare strategie innovative per rilevare e contrastare tali recensioni fasulle. Il presente lavoro mira a sviluppare un sistema basato sulle tecniche di machine learning, capace di identificare con precisione ed efficacia le recensioni false. L’obiettivo è fornire agli utenti una visione affidabile e veritiera dei prodotti, garantendo così un’informazione trasparente e genuina sulle esperienze di acquisto.

1.2 Approccio

Nella fase iniziale del progetto, è stata condotta una ricerca approfondita per raccogliere dati disponibili online che presentavano caratteristiche di recensioni riguardanti la vendita di prodotti e servizi online. Questo passaggio fondamentale ha comportato un'accurata revisione di questi dati.

Per l'addestramento del modello, abbiamo deciso di concentrarci unicamente sul testo delle recensioni. Abbiamo utilizzato diverse tecniche di NLP per la pre-elaborazione dei dati, tra cui la conversione in stringa e in lowercase, la tokenizzazione e l'ottimizzazione del testo con lemming e stemming. Abbiamo poi diviso il dataset pre-processato in Train set e Test set, utilizzando l'80% dei dati (Train Set) per l'addestramento e il rimanente 20% (Test Set) per testare la sua efficacia.

1.2.1 Algoritmi Utilizzati

Per l'addestramento del modello abbiamo utilizzato diversi algoritmi di classificazione:

- **Multinomial Naive Bayes:** Adatto per la classificazione di testi;
- **Random Forest:** Ideale per analizzare le recensioni che presentano diverse caratteristiche e stili;
- **Decision Tree Classifier:** Utile per visualizzare e comprendere il processo decisionale;
- **KNeighbors Classifier:** Utile per identificare schemi non lineari;
- **Support Vector Classifier:** Ideale per distinguere tra recensioni autentiche e false quando esistono margini sottili di differenziazione;
- **Logistic Regression:** Efficace nel classificare dati binari.

1.2.2 Scelta dei Modelli

La scelta del modello da utilizzare è stata basata sulla performance di ciascuno algoritmo durante la fase di addestramento. In particolare abbiamo confrontato l'ac-

curatezza media di ogni modello per avere un risultato immediato sulla performance del modello in generale.

1.2.3 Testing

Una volta completato l’addestramento, abbiamo proceduto al test di tutti e sei i modelli sul Test set. Questo Test set è fondamentale per la valutazione dei modelli, poiché comprende dati che non sono stati utilizzati durante la fase di addestramento. Per testare i modelli, abbiamo impiegato il metodo predict su ogni istanza del Test set. Questo ci ha permesso di valutare le prestazioni di ogni modello nel classificare correttamente le recensioni.

1.2.4 Implementazione del Modello

Una volta aver concluso tutte le fasi precedenti, il modello è stato implementato sviluppando FakeReviewDetector, un bot al quale è possibile inviare una recensione tramite messaggio e lui utilizzando il modello implementato in precedenza la analizzerà fornendo come risposta una indicazione sulla sua autenticità.

CAPITOLO 2

Stato dell'Arte nella Fake Review Detection

2.1 Introduzione alle Fake Reviews

In questa era di Internet, le recensioni effettuate per valutare un prodotto, un ristorante, un negozio e così via, sono molto importanti e d'aiuto per farsi un'idea prima di prendere una decisione. Le recensioni positive portano un grande guadagno a livello finanziario, mentre quelle negative tendono a portare un effetto negativo sulle finanze. Per questo anche le aziende ascoltano le opinioni degli utenti, migliorando i loro prodotti e servizi nelle aree nei quali essi sono più criticati.

I social media ti permettono di postare liberamente feedback senza obbligazioni né limiti. In cambio però, questo porta alcune compagnie ad utilizzare i social per promuovere ingiustamente i loro prodotti oppure a criticare i loro rivali. Spesso questo viene fatto tramite persone o bot utilizzati appositamente a questo scopo. Le recensioni pubblicate in questo modo sono considerate fake. L'attività principale che viene associata all'individuazione delle fake reviews è la classificazione delle recensioni come autentiche o false.

2.1.1 Tipologie di Fake Reviews

Le tipologie più comuni di Fake Reviews sono:

- **Recensioni Positive Falsificate:** Gli stessi venditori o creatori di prodotti possono pubblicare recensioni positive fingendo di essere clienti soddisfatti per aumentare la reputazione del loro prodotto;
- **Recensioni Negative Falsificate:** Ad esempio i concorrenti possono pubblicare recensioni negative per danneggiare la reputazione di un prodotto o di un marchio, anche se basate su false informazioni;
- **Recensioni di Bot Automatici:** I bot possono generare recensioni generiche che non forniscono informazioni specifiche o dettagliate sul prodotto, rendendole difficili da distinguere come genuine o false;
- **Recensioni di Clienti non esistenti:** Venditori o terze parti possono creare account falsi per simulare recensioni provenienti da clienti inesistenti, aumentando artificialmente la popolarità del prodotto;
- **Recensioni di Acquisti Non Verificati:** Alcune piattaforme consentono agli utenti di recensire un prodotto anche senza averlo acquistato effettivamente, consentendo la diffusione di recensioni false senza la necessità di una transazione reale.

La tipologia su cui ci concentreremo è quella relativa alle recensioni di bot automatici per le quali è molto difficile per una persona distinguerle da una scritta da un umano, infatti generalmente esse sono molto simili. Per questo introdurre modelli efficienti per riconoscere le recensioni false in maniera automatica è obbligatorio. A questo scopo, il machine learning gioca un ruolo significativo. Ad esempio, il machine learning supervisionato è molto popolare per svolgere questa attività; esso richiede che i dati siano etichettati per classificare le fake reviews da quelle autentiche sulla base di caratteristiche specifiche. Il machine learning permette di separare le recensioni false da quelle autentiche rivelando pattern di testo nascosti che l'occhio umano non è in grado di riconoscere.

2.2 Attuali approcci alla Fake Review Detection

La seguente revisione della letteratura offre un’analisi dettagliata degli approcci utilizzati per l’individuazione delle recensioni false, con un focus su metodologie basate sia su tecniche di apprendimento statistico tradizionale che su avanzate reti neurali. L’obiettivo è fornire una panoramica generale delle metodologie attualmente in uso e dei progressi raggiunti in questo campo.

L’evoluzione verso l’impiego di reti neurali ha evidenziato vantaggi e limitazioni specifiche di ciascun approccio. Tuttavia, la scelta del modello dipende fortemente dal contesto specifico, dal dataset e dalle caratteristiche considerate.

2.2.1 Apprendimento Supervisionato Tradizionale

Gli approcci di apprendimento supervisionato hanno costituito il nucleo di numerosi studi. Jindal e Liu (2008) hanno proposto un algoritmo di apprendimento basato su un modello di insieme che utilizza unigrammi e bigrammi come caratteristiche predittive, impiegando algoritmi quali Naïve Bayes, Random Forest e Support Vector Machine (SVM) [1]. Altri studi hanno introdotto un modello basato su Sparse Additive Generative Model per rilevare recensioni false in un ambiente cross-domain. L’efficienza di Support Vector Network (SVN) nella classificazione di recensioni false in diversi domini è stata esaminata da Hernandez-Castaneda (2014) [2].

2.2.2 Selezione delle Caratteristiche e Ensemble Learning

La selezione delle caratteristiche è stata identificata come un elemento chiave per migliorare le prestazioni dei modelli. In alcuni studi hanno esteso il lavoro precedente introducendo un modello di apprendimento basato su ensemble, utilizzando tecniche avanzate di selezione delle caratteristiche. Mani et al. (2018) hanno sottolineato l’importanza dell’ensemble learning nel rilevare recensioni false, anche se il loro modello non ha superato gli algoritmi di deep learning [3].

2.2.3 Approcci basati su reti neurali

L'introduzione di metodi basati su reti neurali ha rappresentato un passo significativo nella ricerca. Alcuni studi hanno provato a proporre un modello basato su caratteristiche semantiche del Linguaggio (SLM) utilizzando la caratteristica di similarità coseno per rilevare recensioni false. Xing et al. (2019) hanno impiegato un approccio semi-supervisionato positivo e non etichettato, utilizzando reti neurali per la rappresentazione del documento e modelli multi-task [4].

2.2.4 Reti Neurali Ricorrenti e Generative Adversarial Networks (GAN)

L'uso di reti neurali ricorrenti (RNN) è stato ampiamente esplorato. Degli studi hanno utilizzato una rete neurale ricorrente gated (GRNN) ottenendo miglioramenti significativi nella rilevazione di recensioni false. Gli approcci basati su GAN sono stati proposti con il modello FakeGAN, che ha affrontato la scarsità di dati generando recensioni fintizie. Cao et al. (2020) hanno combinato caratteristiche dettagliate e generali attraverso reti neurali, mostrando un'efficacia promettente, anche se con una complessità computazionale ancora fin troppo più elevata [5].

CAPITOLO 3

Metodologia

3.1 Introduzione all'approccio

In questo progetto, abbiamo adottato una serie di scelte strategiche mirate. La prima di queste riguarda la selezione dei dati. Abbiamo condotto una ricerca approfondita per raccogliere alcuni dei dataset disponibili online che presentavano caratteristiche di recensioni riguardo la vendita di prodotti online. Tra questi, solo un numero limitato includeva un'indicazione sull'autenticità delle recensioni. Questo passaggio fondamentale ha comportato un'accurata revisione di questi dataset, verificando l'affidabilità e il metodo di calcolo dell'indicatore di autenticità.

Una volta identificati e raccolti i dataset che soddisfacevano i nostri criteri, li abbiamo fusi in un unico dataset, denominato "raw_dataset". Questo termine è stato scelto per sottolineare la natura grezza dei dati, che necessitano di ulteriori elaborazioni e ottimizzazioni prima di essere impiegati nel nostro modello.

La seconda decisione cruciale ha riguardato la selezione delle caratteristiche (features) da utilizzare per l'addestramento del modello. Dopo un'analisi preliminare e considerando le attuali ricerche nel campo, abbiamo deciso di concentrarci unicamente sul testo delle recensioni per vari motivi:

- **Rilevanza Informativa:** Il testo delle recensioni è una miniera di informazioni linguistiche e semantiche essenziali per identificare le false recensioni. Caratteristiche come l’uso di determinate parole, la struttura delle frasi e l’impiego di esagerazioni o frasi stereotipate possono rivelare pattern nascosti che gli algoritmi di machine learning sono in grado di rilevare, andando oltre la capacità di analisi umana.
- **Analisi Linguistica (NLP):** Le recensioni false tendono a presentare tratti linguistici distintivi, identificabili attraverso tecniche di Natural Language Processing (NLP). Ciò rende l’analisi testuale un approccio particolarmente efficace.
- **Semplificazione del Modello:** Limitarsi al testo aiuta a prevenire l’overfitting, mantenendo il modello più semplice e interpretabile. Questo approccio riduce la complessità e facilita la manutenzione del modello.
- **Accessibilità dei Dati:** Il testo delle recensioni è facilmente reperibile e disponibile. Altre tipologie di dati, come informazioni utente o contesti di acquisto, possono essere limitati da restrizioni sulla privacy e normative.
- **Scalabilità e Riuso:** Un modello focalizzato sul testo è più facilmente adattabile a diverse piattaforme di recensioni, senza necessità di modifiche per adattarsi a set di dati non testuali.
- **Considerazioni Etiche:** L’utilizzo di dati personali degli utenti potrebbe sollevare questioni di privacy ed etica. Pertanto, l’analisi testuale rappresenta una scelta prudente e rispettosa delle normative vigenti.

Queste scelte metodologiche sono state dettate dalla volontà di sviluppare un modello efficace, riusabile, eticamente responsabile e tecnicamente sostenibile per l’identificazione delle false recensioni online.

3.2 Selezione e Utilizzo dei Tool

Per lo sviluppo del back-end di questo progetto, abbiamo scelto strumenti che si sono dimostrati efficaci e affidabili. Di seguito, i dettagli e le motivazioni dietro la scelta di ciascuno:

- **Python 3.11.4:** Questo linguaggio di programmazione è stato scelto per la sua versatilità e per la vasta comunità che lo supporta. Python è rinomato per la sua sintassi chiara e leggibile, che facilita la manutenzione e l'aggiornamento del codice. Inoltre, offre un eccellente supporto per il machine learning e l'analisi dei dati, con un'ampia gamma di librerie dedicate.
- **Pandas:** Una libreria essenziale per l'analisi dei dati in Python. Pandas offre strutture dati potenti e flessibili, come DataFrame, che semplificano la manipolazione e l'analisi dei dataset complessi. È stata utilizzata per gestire e preparare i dati prima dell'addestramento del modello, consentendo operazioni efficienti di pulizia, selezione e trasformazione dei dati.
- **Scikit-learn:** Questa libreria è stata utilizzata per l'addestramento dei modelli di machine learning principalmente perché offre una vasta gamma di algoritmi efficienti e facili da usare, sia per compiti di classificazione che di regressione, clustering, riduzione della dimensionalità, e selezione del modello. Inoltre, la sua documentazione è estremamente dettagliata e user-friendly.
- **Matplotlib:** Questa libreria è stata impiegata per la creazione e la personalizzazione di grafici e visualizzazioni di dati. Matplotlib è uno strumento estremamente versatile che supporta una vasta gamma di tipi di grafici, permettendo di visualizzare complessi pattern nei dati in modo intuitivo. Questo ha facilitato sia l'analisi esplorativa dei dati sia la presentazione dei risultati del modello.
- **Jupyter Notebook:** Sono stati utilizzati per documentare e presentare il flusso di lavoro di analisi. Questo strumento è ideale per combinare codice eseguibile, output visivi e testo narrativo in un unico documento. Ciò ha reso possibile non solo eseguire analisi in tempo reale, ma anche condividere procedure e risultati in modo interattivo e didattico.

3.3 Analisi Esplorativa dei Dati

Questa sezione si dedica all’analisi esplorativa dei dati raccolti, con l’obiettivo di identificare le caratteristiche chiave e i pattern rilevanti per il nostro studio. Il notebook ‘data_presentation’ è stato strutturato per fornire una panoramica esaustiva del dataset. Tale sezione integra codice, testo in formato markdown e rappresentazioni grafiche per una più chiara esposizione dei dati.

3.3.1 Descrizione del Dataset

Il dataset impiegato nel nostro studio, denominato "raw_dataset", rappresenta un’aggregazione di diverse raccolte di dati riguardanti recensioni online. Questo dataset è particolarmente esteso, comprendendo 40.432 istanze o righe, ognuna rappresentante una recensione unica, e si articola in 4 features o colonne, che rappresentano le diverse caratteristiche di ciascuna recensione. Sotto è rappresentata un’immagine che illustra le prime righe, permettendo di osservare direttamente la sua struttura e le caratteristiche delle colonne 3.1.

category	rating	label	text_
Home_and_Kitchen	5.0	CG	Love this! Well made, sturdy, and very comfor...
Home_and_Kitchen	5.0	CG	love it, a great upgrade from the original. I...
Home_and_Kitchen	5.0	CG	This pillow saved my back. I love the look and...
Home_and_Kitchen	1.0	CG	Missing information on how to use it, but it i...
Home_and_Kitchen	5.0	CG	Very nice set. Good quality. We have had the s...

Figura 3.1: Immagine che illustra le prime righe del dataset "raw_dataset"

Una caratteristica distintiva del nostro dataset è l’assenza totale di valori nulli in tutte le sue colonne, un aspetto cruciale che indica la completezza e l’affidabilità dei dati raccolti. Tale completezza è fondamentale per garantire l’accuratezza dei modelli di machine learning che verranno successivamente addestrati su questi dati.

Per quanto riguarda i tipi di dati, il dataset utilizza principalmente tre formati: float64 e int64 per le variabili numeriche, e il tipo ‘object’, impiegato prevalentemente per gestire dati di tipo testuale, come le stringhe. Questa diversità nei tipi di

dati riflette la varietà delle informazioni raccolte, da valutazioni numeriche a testi descrittivi.

In termini di spazio di archiviazione, il dataset si dimostra sorprendentemente compatto, occupando solamente 1.52 MB di memoria. Questa efficienza nello spazio di archiviazione facilita la manipolazione, l'analisi e la condivisione dei dati, rendendo il nostro dataset un eccellente candidato per applicazioni di machine learning in ambienti con risorse computazionali limitate.

3.3.2 Descrizione delle Features

Inizialmente, il dataset "raw_dataset" comprende quattro features principali:

- **Category:** Questa feature rappresenta la macro categoria del prodotto recensito. La distribuzione delle categorie fornisce un'indicazione della varietà di prodotti e servizi inclusi nel dataset. Un grafico della distribuzione delle categorie aiuta a visualizzare questo aspetto 3.2.

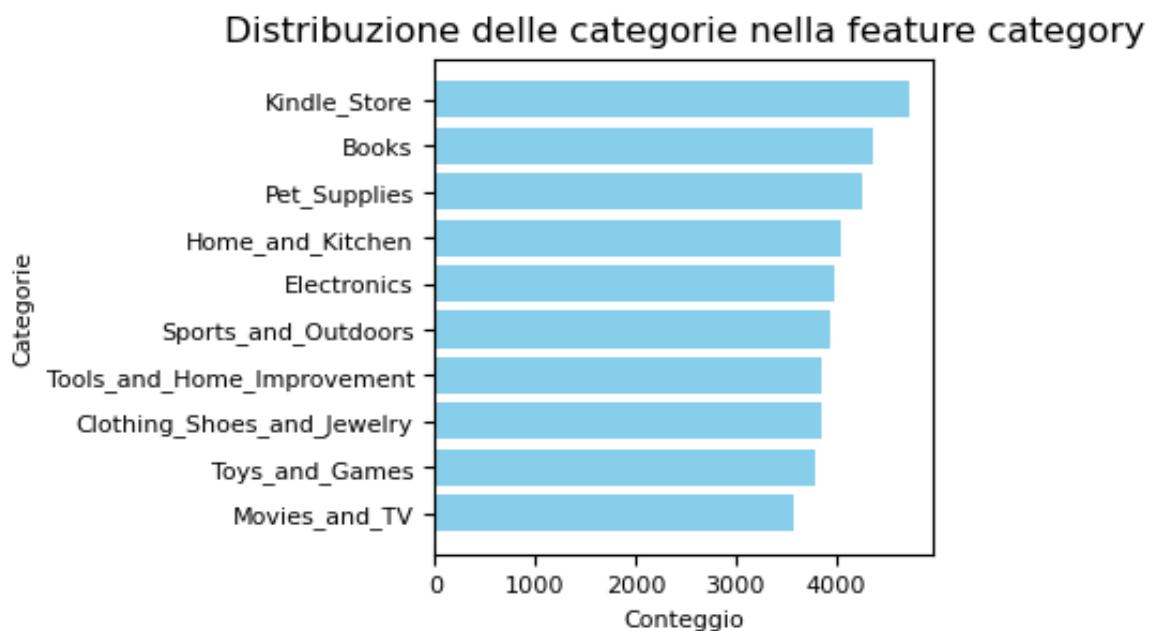


Figura 3.2: Grafico che illustra la distribuzione della feature "category"

- **Rating:** Questa feature è una valutazione numerica che va da 1 a 5 e riflette il giudizio dell'utente sul prodotto recensito. Il rating fornisce una prima impressione generale della recensione.

- **Autenticità (o Label):** Questa è la prima feature chiave del progetto. Rappresenta un valore booleano che indica se una recensione è stata generata da un computer (CG) o è originale (OR). Abbiamo mirato a costruire un dataset bilanciato in termini di questa feature per evitare problemi di bias comuni nel machine learning, come il sovrapprendimento su una classe predominante. Un grafico che rappresenta questo equilibrio tra recensioni CG e OR è essenziale per dimostrare la validità del dataset per l'addestramento del modello 3.3.

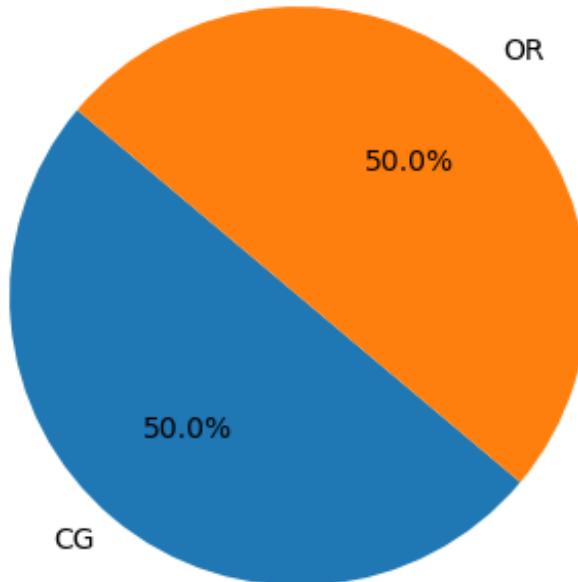


Figura 3.3: Grafico che illustra la distribuzione della feature "label"

- **Text:** La seconda feature di fondamentale importanza è il testo della recensione. Questa è l'unica feature che verrà utilizzata per prevedere l'autenticità di una recensione, permettendo al modello di apprendere a distinguere tra recensioni CG e OR. Nella fase di esplorazione dei dati, abbiamo elaborato statistiche sulla lunghezza delle recensioni, per ottenere una visione approfondita di questa caratteristica. Un grafico che mostra la distribuzione delle lunghezze delle recensioni può rivelare differenze significative tra le recensioni CG e OR 3.4.

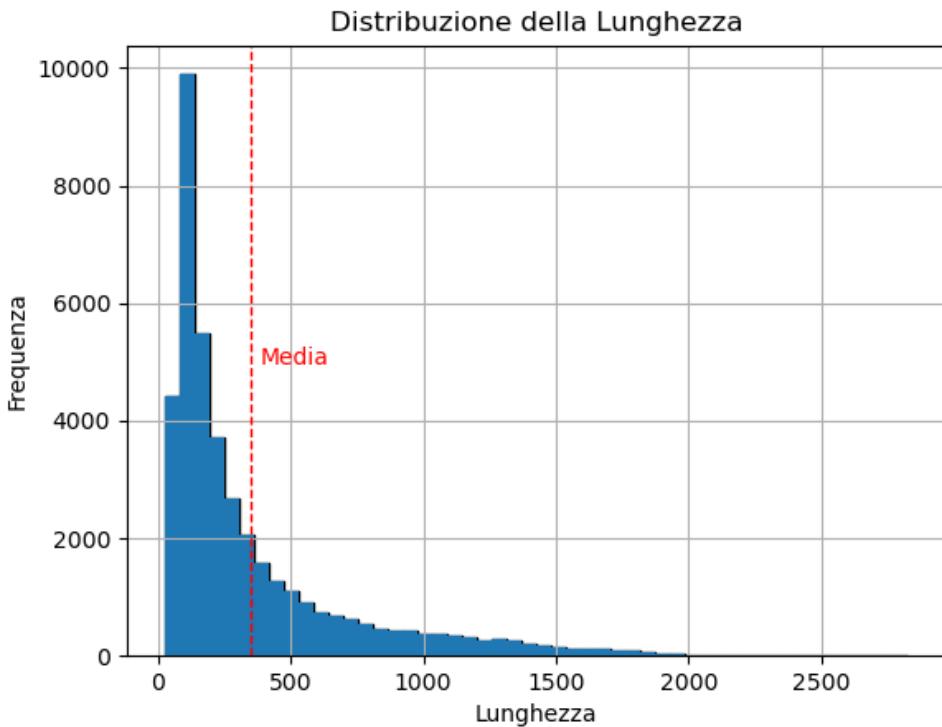


Figura 3.4: Grafico che illustra la distribuzione della lunghezza della feature "text_"

3.4 Preparazione dei dati

Introduzione alla Pre-Elaborazione Nel nostro progetto di rilevamento di recensioni false, abbiamo riconosciuto l'importanza della pre-elaborazione dei dati come passo fondamentale nel preparare i dati per l'addestramento di un modello di Machine Learning (ML). Questo processo, cruciale nel campo dell'NLP, ci ha permesso di pulire e standardizzare il testo, migliorando significativamente la qualità delle informazioni fornite al modello. Il nostro obiettivo principale è stato ridurre il rumore nei dati e aumentare la precisione nella classificazione delle recensioni. Per facilitare il monitoraggio e la comprensione di questo processo, abbiamo creato un file Jupyter Notebook. In questo notebook, abbiamo accuratamente suddiviso le varie fasi di pre-elaborazione, esattamente come descritte in questo documento. Ciò offre un riferimento immediato e chiaro, mettendo in evidenza i benefici di ogni fase e facilitando la riproducibilità e la revisione del nostro approccio da parte di altri ricercatori o membri del team.

Operazioni di Base

- **Conversione in Stringa:** Abbiamo convertito ogni entry nella colonna 'text_' in stringa, un passo essenziale per gestire correttamente i dati testuali, specialmente considerando che potevano essere stati importati con formati non uniformi.
- **Conversione in Lowercase:** Abbiamo trasformato ogni lettera nella colonna "text_" in minuscolo. In NLP, la normalizzazione del testo come questa è cruciale per ridurre la varianza causata dalle differenze tra maiuscole e minuscole.

Analisi Testuale e Pulizia

- **Tokenizzazione:** Abbiamo suddiviso il testo in singole parole o token. Questa operazione è stata cruciale per analizzare la struttura e i contenuti delle recensioni, consentendoci di identificare schemi linguistici o usi insoliti del linguaggio che potrebbero indicare una recensione falsa.
- **Rimozione delle Stop Words:** Abbiamo eliminato parole comuni che aggiungevano poco significato al testo. Questo ha permesso al nostro modello di concentrarsi su parole più rilevanti, fondamentali per identificare l'autenticità o la falsità delle recensioni.
- **Rimozione dei Numeri e della Punteggiatura:** Abbiamo rimosso numeri e segni di punteggiatura, poiché spesso non contribuivano al rilevamento delle recensioni false. Tuttavia, abbiamo considerato che in certi contesti la punteggiatura potrebbe essere un indicatore di recensioni false, ma abbiamo comunque deciso di rimuoverla dopo alcuni test di performance, in ambito di velocità di predizione del modello.

Ottimizzazione del Testo per ML Abbiamo ridotto la variabilità del testo convertendo le parole nelle loro forme base o radici. Questi processi, fondamentali nell'NLP, ci hanno aiutato a concentrarci sul contenuto essenziale delle recensioni, minimizzando le variazioni linguistiche superficiali.

- **Lemming:** Questo processo ha coinvolto la trasformazione delle parole nelle loro forme di lemma, che rappresentano il modo più significativo e di base di una parola. A differenza dello stemming, la lemmatizzazione considera il contesto e trasforma la parola nella sua forma base significativa. Per esempio, la parola "running" viene trasformata in "run", e "better" in "good". Questo è stato particolarmente utile nel nostro progetto per mantenere il significato contestuale delle parole, consentendoci di analizzare il testo in modo più preciso e rilevante. Abbiamo utilizzato algoritmi avanzati di lemmatizzazione che considerano la morfologia delle parole, garantendo così che la forma base sia una parola valida nella lingua target.
- **Stemming:** Al contrario, lo stemming è un metodo più semplice e rudimentale che taglia le estremità delle parole per arrivare alla loro forma radicale, senza tenere conto del contesto. Per esempio, "running", "runner", "runs" possono essere tutti ridotti a "run". Lo stemming è utile per ridurre la complessità dei dati testuali e per diminuire il numero di varianti di una stessa parola nel testo. Tuttavia, la forma risultante può non essere una parola valida. Ad esempio, lo stemming di "better" potrebbe risultare in "bett", che non è una parola riconoscibile. Nonostante questa limitazione, lo stemming si è rivelato efficace nel nostro progetto per ridurre la complessità linguistica e velocizzare il processo di analisi.

Entrambi i processi hanno giocato un ruolo cruciale nel nostro lavoro di pre-elaborazione, poiché hanno semplificato la struttura linguistica delle recensioni e ci hanno permesso di concentrarci sui contenuti chiave per il rilevamento di recensioni false. La combinazione di lemmatizzazione e stemming ha migliorato notevolmente l'efficacia del nostro modello di machine learning, fornendo un input testuale più pulito e coerente.

Benefici nel Rilevamento di Recensioni False Queste tecniche hanno ridotto la complessità del testo e migliorato l'efficienza nell'elaborazione, facilitando il rilevamento di schemi tipici delle recensioni false. L'analisi del sentimento si è rivelata utile per identificare eccessi emotivi o incongruenze tra il punteggio assegnato e il senti-

mento espresso nel testo, evidenziando tentativi di manipolazione emotiva. Questi metodi di pre-elaborazione sono stati fondamentali per il successo del nostro progetto, permettendoci di affinare il nostro approccio nel rilevamento delle recensioni false e di sfruttare al meglio le potenzialità dell’NLP e del machine learning.

3.5 Addestramento Modelli

In questa sezione, descriveremo in dettaglio l'intero processo attraverso cui abbiamo addestrato diversi modelli di machine learning, utilizzando tecniche avanzate di Natural Language Processing (NLP). Il nostro obiettivo è stato quello di valutare e identificare il modello più performante, basandoci sull'efficacia dell'algoritmo utilizzato.

È importante sottolineare che ogni fase di questo processo è stata documentata e implementata in un apposito file Jupyter Notebook, creato appositamente per permettere un'analisi approfondita e un confronto diretto con il codice. Questo approccio ci ha permesso non solo di testare diverse tecniche di NLP, ma anche di valutare in modo trasparente e riproducibile i risultati ottenuti.

Bag-of-Words Dopo aver pulito il testo, il nostro prossimo passo è stato convertirlo in un formato comprensibile ai modelli di machine learning. Per farlo, abbiamo implementato il concetto di "Bag-of-Words" (BoW). In questo approccio, abbiamo trasformato il testo in un vettore numerico, dove ogni elemento indica la presenza o la frequenza di una parola all'interno del vocabolario complessivo del nostro set di dati. Questo metodo ci ha permesso di semplificare il testo, focalizzandoci sulla frequenza delle parole, che si è rivelata un indicatore efficace del sentimento o del tono generale di una recensione.

Questi passaggi sono stati fondamentali nel nostro lavoro di NLP e nell'analisi del sentimento, perché ci hanno consentito di convertire testi complessi in dati strutturati e analizzabili. In particolare, per la rilevazione di recensioni false o l'analisi del sentimento, è stato cruciale capire quali parole fossero più frequentemente associate a recensioni positive, negative o ingannevoli.

TF-IDF Successivamente abbiamo implementato la trasformazione TF-IDF, Term Frequency-Inverse Document Frequency, per affinare la nostra capacità di analizzare e interpretare testi, in particolare nell'analisi del sentimento e nella rilevazione di recensioni false. Abbiamo combinato due concetti fondamentali: la frequenza del termine (TF), che misura quanto spesso una parola appare in un documento, e l'in-

verse document frequency (IDF), che valuta l’importanza di una parola nel contesto dell’intero dataset. Questo ci ha permesso di dare maggior peso alle parole rare, bilanciando così la loro frequenza con la loro unicità.

Con la trasformazione TF-IDF, abbiamo trasformato il testo in una rappresentazione numerica, dove parole uniche e significative nei documenti specifici hanno ottenuto un punteggio più alto. Questo approccio è stato particolarmente efficace per la nostra analisi, poiché ha aiutato i nostri modelli di machine learning a concentrarsi su parole o frasi chiave che potrebbero indicare il tono o l’autenticità di una recensione. Invece di affidarci a un vasto vocabolario, ci siamo concentrati su un numero più ristretto di parole chiave, aumentando la precisione della nostra classificazione.

Definizione pipeline addestramento Nel nostro progetto, abbiamo significativamente semplificato il processo di elaborazione del testo e addestramento del modello grazie all’integrazione della classe Pipeline di scikit-learn nel nostro flusso di lavoro. Questa integrazione ha permesso di definire, in ogni pipeline, tre operazioni distinte. Ognuna di queste incorpora due fasi precedentemente menzionate: la conversione in Bag-of-Words e la trasformazione TF-IDF. Inoltre, ogni pipeline include una specifica funzione di classificazione. Di conseguenza, abbiamo creato sei pipeline diverse, ciascuna basata su un differente algoritmo di classificazione. Sebbene la struttura di ogni pipeline sia identica, ciò che le differenzia è esclusivamente l’algoritmo di classificazione impiegato.

Prima di implementare le sei pipeline, abbiamo diviso il dataset pre-processato in due parti: il Train set e il Test set, utilizzando un rapporto di 80:20. Questo significa che l’80% dei dati è stato destinato all’addestramento del modello (Train set), mentre il rimanente 20% è stato utilizzato per testare la sua efficacia (Test set). Tale divisione è cruciale per garantire che il modello sia allenato su una porzione sostanziale del dataset, permettendogli di apprendere in modo efficace, e allo stesso tempo assicurare che ci sia una quota sufficiente di dati non visti per validare in modo imparziale la sua accuratezza e generalizzabilità.

Dopo aver suddiviso il dataset, abbiamo proceduto con l’addestramento di ogni pipeline. In questa fase, abbiamo utilizzato il metodo fit fornito dalla libreria scikit-learn su ciascuna delle sei pipeline. Questo metodo permette al modello di ‘impa-

rare' dai dati di addestramento. Attraverso il fit, il modello analizza i pattern e le caratteristiche presenti nel Train set, adattando i suoi parametri interni per poter successivamente effettuare predizioni accurate sul Test set e, in scenari reali, su nuovi dati non ancora esaminati.

Classificatori utilizzati Proseguendo nella descrizione del nostro progetto, approfondiamo gli algoritmi di classificazione utilizzati in ciascuna delle sei pipeline, ognuno selezionato per le sue peculiarità e potenzialità nell'ambito del rilevamento delle recensioni false:

- **Algoritmo Multinomial Naive Bayes:** Questo algoritmo è particolarmente adatto per la classificazione di testi, grazie alla sua capacità di gestire un gran numero di parole. È efficace nel riconoscere schemi e tendenze nelle recensioni, consentendoci di identificare quelle false con un'elevata precisione.
- **Algoritmo Random Forest:** Questo algoritmo utilizza un insieme di alberi decisionali per migliorare la precisione della classificazione. È robusto contro il sovra-adattamento e può gestire sia dati categorici che continui, rendendolo ideale per analizzare le recensioni che presentano diverse caratteristiche e stili.
- **Algoritmo Decision Tree Classifier:** Basato sulla costruzione di alberi decisionali, questo algoritmo è utile per visualizzare e comprendere il processo decisionale. Pur essendo semplice, può essere molto efficace nel distinguere recensioni autentiche da quelle false, specialmente quando le differenze sono ben definite.
- **Algoritmo KNeighbors Classifier:** Questo algoritmo di classificazione basato sui vicini più prossimi è particolarmente utile per identificare schemi non lineari. La sua efficacia nel rilevamento delle recensioni false risiede nella capacità di riconoscere gruppi di recensioni simili, facilitando l'identificazione di quelle potenzialmente false.
- **Algoritmo Support Vector Classifier:** Questo algoritmo è rinomato per la sua efficienza nella classificazione di dati complessi e di alta dimensionalità. È

particolarmente utile per distinguere tra recensioni autentiche e false quando esistono margini sottili di differenziazione.

- **Algoritmo Logistic Regression:** Nonostante la sua semplicità, l'algoritmo di regressione logistica si rivela efficace nel classificare dati binari, come il caso delle recensioni false e autentiche. È particolarmente utile per la sua capacità di fornire probabilità, offrendo una misura della sicurezza con cui una recensione può essere classificata come falsa.

Ognuno di questi algoritmi è stato scelto per la sua capacità di analizzare e interpretare diversi aspetti delle recensioni, garantendo così un approccio robusto al rilevamento delle recensioni false.

3.6 Testing dei Modelli

Sulla base di questa struttura, una volta completato l'addestramento, abbiamo proceduto al test di tutti e sei i modelli sul Test set, che rappresenta il 20% del dataset precedentemente diviso. Questo Test set è fondamentale per la valutazione dei modelli, poiché comprende dati che non sono stati utilizzati durante la fase di addestramento. Ciò assicura che, in fase di test, ogni modello si confronti con questi dati come se fossero completamente nuovi, offrendo una valutazione realistica della loro capacità di generalizzazione e della loro efficacia nel riconoscere recensioni false in situazioni non incontrate precedentemente. Per testare i modelli, abbiamo impiegato il metodo predict su ogni istanza del Test set. Questo ci ha permesso di valutare le prestazioni di ogni modello nel classificare correttamente le recensioni. La misura chiave che abbiamo scelto per comparare i modelli è stata l'accuratezza media, ovvero la percentuale di predizioni corrette rispetto al totale delle predizioni effettuate. Abbiamo optato per l'accuratezza come unica metrica di paragone per diverse ragioni. Prima di tutto, l'accuratezza fornisce un'intuizione immediata e comprensibile sulla performance del modello in termini generali, rendendola una metrica facilmente interpretabile. Inoltre, in un contesto dove le classi (recensioni autentiche vs false) sono relativamente bilanciate, l'accuratezza diventa un indicatore affidabile delle prestazioni del modello.

3.7 FakeReviewDetector

Una volta aver addestrato il modello a distinguere le recensioni vere da quelle generate da bot, lo abbiamo implementato sviluppando FakeReviewDetector. Esso è un bot Telegram al quale è possibile inviare una recensione sottoforma di messaggio. Questo sarà elaborato dal bot che sarà in grado di dire all'utente se quella recensione è autentica o meno.

3.7.1 Tecnologie Utilizzate

Lo sviluppo di FakeReviewDetector ha implicato l'utilizzo di varie tecnologie:

- **Linguaggio di programmazione:** il linguaggio scelto per programmare il bot è stato Python, poiché grazie alle sue numerose librerie e integrazioni con i framework, risulta essere un linguaggio semplice, potente e che permette di ottimizzare lo sviluppo;
- **Framework:** il framework utilizzato è TeleBot, una libreria per Python che facilita lo sviluppo di bot per Telegram. Con esso, è possibile gestire le richieste degli utenti, inviare messaggi, rispondere a comandi e interagire con gli utenti attraverso il proprio bot. L'API di Telegram offre una serie di funzionalità che consentono di interagire con la piattaforma.

3.7.2 Implementazione

La fase iniziale è quella relativa alla creazione del bot.

Telegram è una piattaforma di messaggistica che offre diverse funzionalità, tra le quali BotFather. Esso è un bot, che svolge il ruolo di "padre" di tutti i bot presenti su Telegram e che permette di creare, gestire e personalizzare i propri bot. Quindi il nostro bot è stato creato attraverso BotFather, il quale, alla fine della fase di creazione restituisce un token, indispensabile per l'autenticazione quando si inviano richieste API al bot.

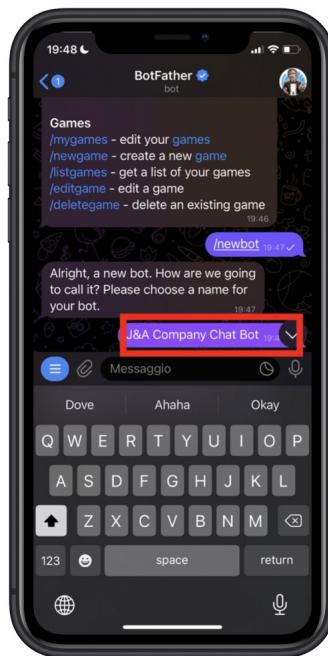


Figura 3.5: Esempio di creazione di un bot

Dopo la fase di creazione si è passati allo sviluppo vero e proprio, semplificato con l'utilizzo di Telebot.

```
#Function that handles the response to the start command
@bot.message_handler(commands=["start"])
def start(message):
    bot.reply_to(message, "Hi! Welcome to FakeReviewDetector, send me a review to check it")
```

Figura 3.6: Funzione che gestisce il comando start

```
#Function that handles the review sent in the form of a message
@bot.message_handler()
def message(message):
    recensione = message.text
    prediction = isReviewReal(recensione)
    if prediction == True:
        bot.reply_to(message,"The review you submitted is true and was therefore written by a human")
    else:
        bot.reply_to(message,"You submitted a bot-written review, which means that it is not authentic")
```

Figura 3.7: Funzione che gestisce la risposta alle recensioni

Come si può notare dalle figure sopra, per gestire i messaggi inviati dall'utente si utilizza il bot message handler, con la relativa funzione che ci permette di gestire i vari casi.

3.7.3 Funzionamento

Fin'ora abbiamo descritto le tecnologie utilizzate e come è avvenuto lo sviluppo del bot FakeReviewDetector. In questa sezione verrà illustrato nel dettaglio il suo funzionamento.

Innanzitutto per utilizzare il bot bisognerà scaricare l'applicazione Telegram e cercare FakeReviewDetector o @frdetectionbot.

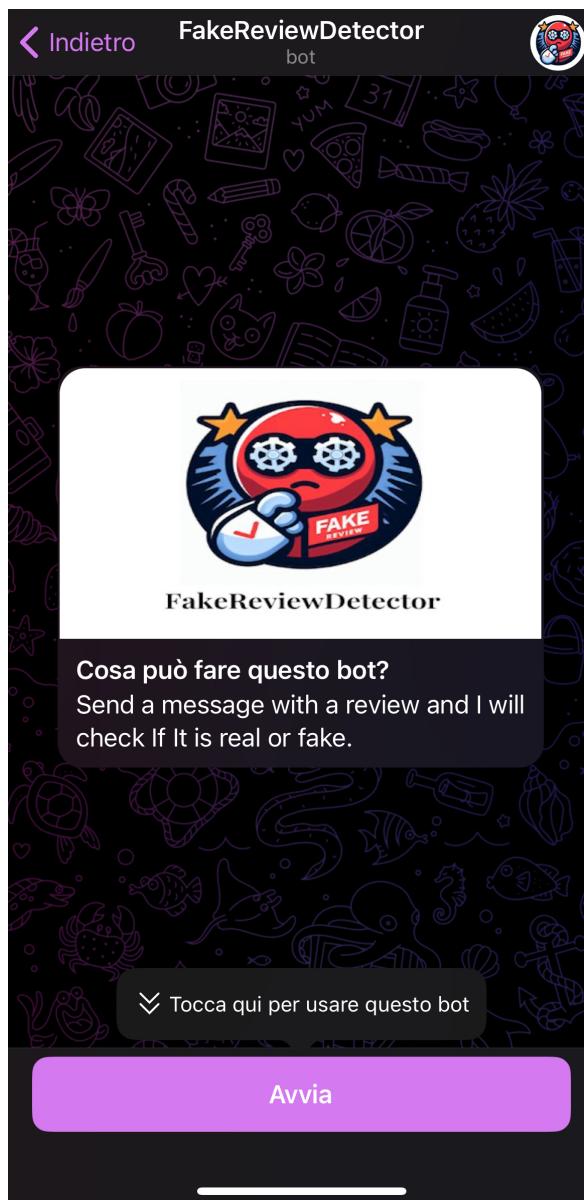


Figura 3.8: Pagina iniziale del bot

Inizialmente il bot si presenta in questo modo e cliccando sul pulsante avvia è possibile iniziare la conversazione:

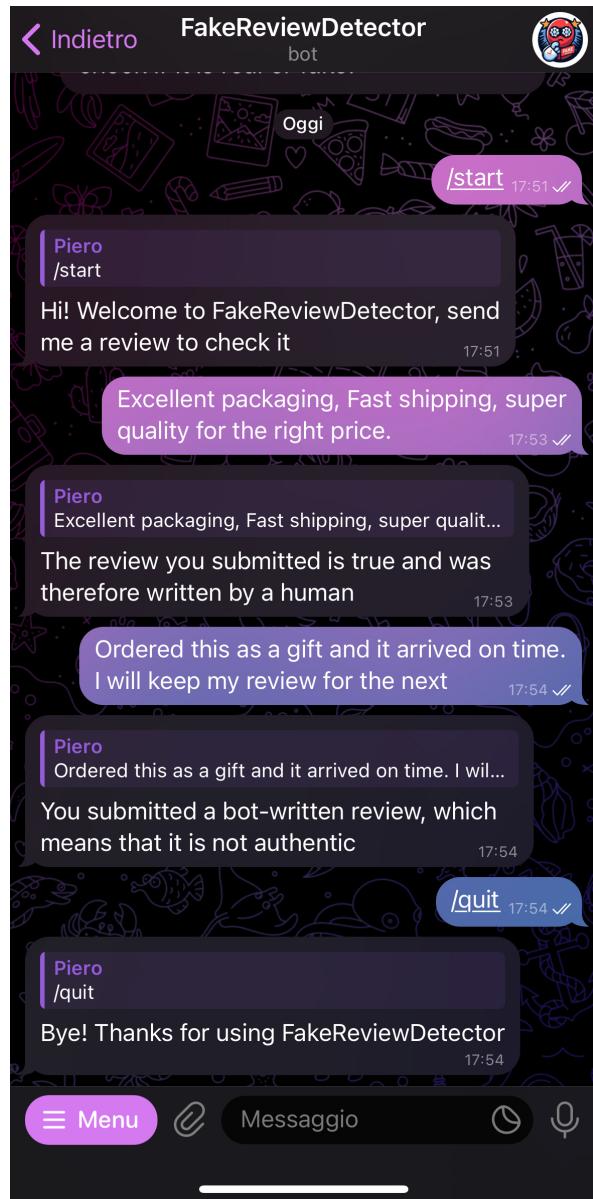


Figura 3.9: Esempio completo di conversazione

Come si può notare, date due recensioni diverse, il bot riesce a rispondere all’utente identificando quale delle due è autentica e quale è falsa.

Inoltre è possibile cliccare sul pulsante menù per vedere l'elenco dei comandi disponibili:

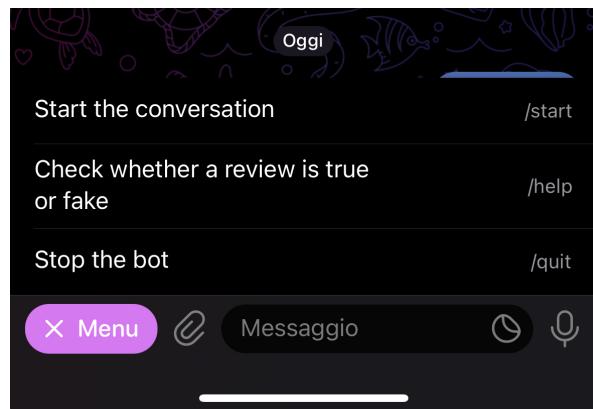


Figura 3.10: Elenco dei comandi del bot

CAPITOLO 4

Risultati

4.1 Introduzione

In questo capitolo, ci focalizziamo sui risultati ottenuti dal testing dei sei modelli di machine learning sviluppati per il rilevamento di recensioni false. Per garantire un confronto equo e omogeneo tra i vari modelli, abbiamo adottato un approccio standardizzato nella fase di testing.

Prima di tutto, è importante sottolineare che per tutti i sei modelli è stato utilizzato il 20% del dataset complessivo. Questa divisione è stata mantenuta costante sia durante la fase di addestramento che in quella di testing. Tale scelta metodologica assicura che ogni modello sia stato valutato in condizioni identiche, permettendo così un confronto più diretto e significativo delle loro prestazioni.

Data la natura specifica del problema del rilevamento di recensioni false, abbiamo deciso di utilizzare l'accuratezza come unica metrica per il confronto dei modelli. Questa decisione si basa sul fatto che, in un contesto dove l'obiettivo è identificare correttamente le recensioni autentiche e quelle false, l'accuratezza fornisce una misura immediata e intuitiva dell'efficacia del modello nel raggiungere tale obiettivo. Essa misura la proporzione di predizioni corrette (sia positive che negative) rispetto

al totale delle predizioni, offrendo quindi una visione generale dell'efficacia del modello.

Di seguito, presenteremo dettagliatamente i risultati di ciascun modello. Saranno mostrate delle tabelle che riassumono i risultati ottenuti, includendo non solo l'accuracy ma anche altre metriche di performance. Queste metriche aggiuntive, pur non essendo state utilizzate per il confronto diretto tra i modelli, possono fornire informazioni preziose per coloro che desiderano esplorare ulteriormente questo campo o continuare il lavoro da noi iniziato. In aggiunta, sarà presentato un grafico che illustra visivamente il confronto tra i vari modelli, permettendo una comprensione immediata delle loro prestazioni relative.

4.2 Risultati

Di seguito sono presentate una serie di tabelle con i risultati ottenuti per ogni modello:

4.2.1 Multinomial Naive Bayes

Class	Precision	Recall	F1 Score	Support
CG	0.83	0.88	0.85	4036
OR	0.88	0.82	0.85	4051
Average	0.85	0.85	0.85	8087

Tabella 4.1: Multinomial Naive Bayes Model Results

	Predicted CG	Predicted OR
Actual CG	3566	470
Actual OR	742	3309

Tabella 4.2: Confusion Matrix for Multinomial Naive Bayes Model

Metric	Score
Accuracy Score	0.8501

Tabella 4.3: Accuracy Score for Multinomial Naive Bayes Model

4.2.2 Random Forest Algorithm

Class	Precision	Recall	F1 Score	Support
CG	0.82	0.88	0.85	4036
OR	0.87	0.81	0.84	4051
Average	0.85	0.85	0.85	8087

Tabella 4.4: Random Forest Algorithm Model Results

	Predicted CG	Predicted OR
Actual CG	3558	478
Actual OR	768	3283

Tabella 4.5: Confusion Matrix for Random Forest Algorithm

Metric	Score
Accuracy Score	0.8459
Model Prediction Accuracy	84.59%

Tabella 4.6: Accuracy Score for Random Forest Algorithm

4.2.3 Decision Tree Classifier

Class	Precision	Recall	F1 Score	Support
CG	0.74	0.75	0.75	4036
OR	0.75	0.74	0.74	4051
Average	0.74	0.74	0.74	8087

Tabella 4.7: Decision Tree Classifier Results

	Predicted CG	Predicted OR
Actual CG	3043	993
Actual OR	1073	2978

Tabella 4.8: Confusion Matrix for Decision Tree Classifier

Metric	Score
Accuracy Score	0.7445
Model Prediction Accuracy	74.45%

Tabella 4.9: Accuracy Score for Decision Tree Classifier

4.2.4 KNeighbors Classifier

Class	Precision	Recall	F1 Score	Support
CG	0.54	0.97	0.70	4036
OR	0.86	0.18	0.30	4051
Average	0.70	0.57	0.50	8087

Tabella 4.10: KNeighbors Classifier Algorithm Model Results

	Predicted CG	Predicted OR
Actual CG	3919	117
Actual OR	3320	731

Tabella 4.11: Confusion Matrix for KNeighbors Classifier Algorithm

Metric	Score
Accuracy Score	0.5750
Model Prediction Accuracy	57.5%

Tabella 4.12: Accuracy Score for KNeighbors Classifier Algorithm

4.2.5 Support Vector Classifier Model Results

Class	Precision	Recall	F1-Score	Support
CG	0.90	0.87	0.89	4036
OR	0.88	0.90	0.89	4051
Average/Total	0.89	0.89	0.89	8087

Tabella 4.13: Support Vector Classifier Model Results

Actual \ Predicted	CG	OR
CG	3514	522
OR	391	3660

Tabella 4.14: Confusion Matrix for Support Vector Classifier

Metrica	Valore
Accuracy Score	0.887
Accuracy di Predizione	88.71%

Tabella 4.15: Accuracy Score for Support Vector Classifier

4.2.6 Logistic Regression

Class	Precision	Recall	F1 Score	Support
CG	0.88	0.86	0.87	4036
OR	0.86	0.88	0.87	4051
Average	0.87	0.87	0.87	8087

Tabella 4.16: Logistic Regression Algorithm Model Results

	Predicted CG	Predicted OR
Actual CG	3451	585
Actual OR	487	3564

Tabella 4.17: Confusion Matrix for Logistic Regression Algorithm

Metric	Score
Accuracy Score	0.8674
Model Prediction Accuracy	86.74%

Tabella 4.18: Accuracy Score for Logistic Regression Algorithm

4.3 Confronto risultati

Di seguito un grafico che mostra un confronto tra i vari modelli, utilizzando come unica metrica l'accuratezza.

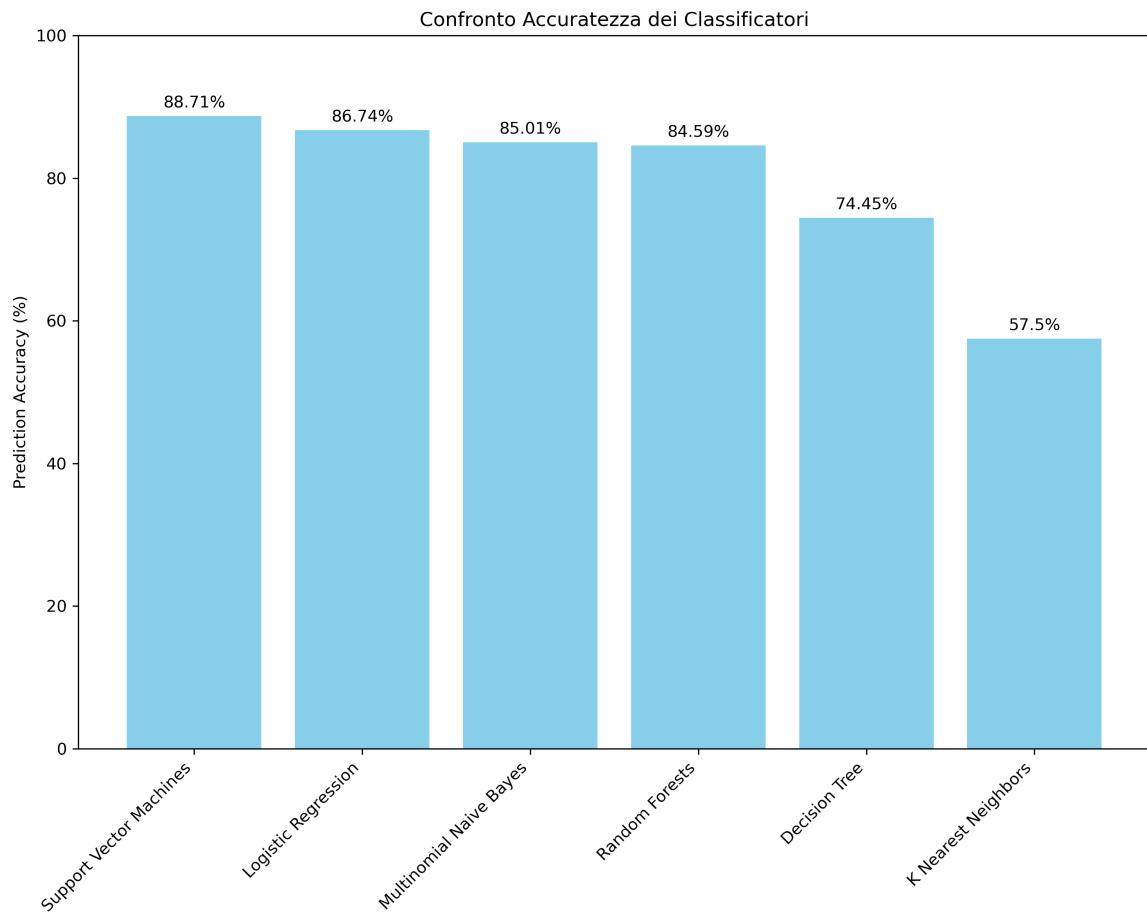


Figura 4.1: Grafico contenente il confronto tra i modelli

CAPITOLO 5

Conclusioni e Sviluppi futuri

In conclusione, il presente lavoro ha affrontato con successo la sfida di sviluppare un modello mirato ad identificare recensioni false generate da bot. Attraverso l'impiego di diverse tecniche di elaborazione del linguaggio naturale (NLP), il nostro modello ha dimostrato una buona accuratezza nel distinguere recensioni autentiche da quelle create in maniera artificiale ed è stato fondamentale nell'analizzare la complessità e la diversità del linguaggio umano nell'ambito delle valutazioni online.

La crescente diffusione di recensioni false online rappresenta un problema per i consumatori, che spesso si affidano a tali opinioni per prendere una decisione sull'acquisto di prodotti e servizi. Questo modello si propone come uno strumento importante in questo contesto, offrendo agli utenti un mezzo per filtrare le recensioni e garantire una valutazione basata esclusivamente su esperienze autentiche. Inoltre, avendo implementato il modello in un bot su Telegram, esso si propone come un tool user-friendly, utilizzabile da chiunque e su qualsiasi dispositivo.

5.1 Sviluppi Futuri

Gli sviluppi futuri di FakeReviewDetector possono assumere diverse direzioni per incrementare ulteriormente l'efficacia del sistema. In particolare, abbiamo identificato

alcune possibili strade che possono essere intraprese da coloro interessati a proseguire questo lavoro:

- **Addestramento su Ulteriori Features del Dataset:** Introdurre la possibilità di addestrare il modello su più caratteristiche del dataset, come il rating e la categoria del prodotto. Questo potrebbe fornire una visione più completa delle recensioni, migliorando la capacità del sistema di rilevare anomalie e pattern sospetti nelle recensioni.
- **Testing su dati reali:** Uno dei prossimi passi nel miglioramento del nostro modello di machine learning potrebbe essere l'introduzione di una fase di testing su dati reali, non visti durante l'addestramento e non appartenenti al dataset usato durante l'addestramento. Questo approccio mira a valutare la capacità del modello di generalizzare le sue predizioni in ambienti reali, fornendo un'indicazione affidabile della sua performance effettiva. L'uso di un set di test costituito da esempi reali e attuali potrebbe essere molto utile per identificare come il modello si comporta di fronte a nuove informazioni, permettendoci di rilevare eventuali limiti nelle sue capacità predittive. Questo test ci potrebbe aiutare a comprendere meglio la robustezza del modello, le sue potenziali vulnerabilità e a identificare aree per miglioramenti futuri.
- **Affinamento delle tecniche NLP:** Continuare a raffinare i modelli di NLP per migliorare la capacità di comprendere il linguaggio umano in modo più profondo. Ciò potrebbe includere l'addestramento su set di dati più ampi o più specifici, o l'uso di modelli di linguaggio di ultima generazione.
- **Integrazione del modello all'interno di un e-commerce:** Il modello può essere integrato come funzionalità all'interno di un e-commerce. Questo garantirà l'identificazione della veridicità di una recensione, rendendo le recensioni un mezzo affidabile per scegliere se acquistare un prodotto o meno.
- **Ospitare il bot su una piattaforma di Hosting:** Un primo obiettivo è quello di ospitare il bot su una piattaforma di hosting. Questo approccio garantirà la continuità operativa assicurando l'esecuzione continua del servizio e consentendo agli utenti di accedervi in qualsiasi momento, la scalabilità e la gestione

del traffico per gestire efficacemente i picchi di traffico e garantire una risposta rapida agli utenti, oltre agli aggiornamenti.

In definitiva, FakeReviewDetector ha il potenziale per evolversi in diverse direzioni, innovandosi e adattandosi alle esigenze degli utenti per mantenere una buona rilevanza nel tempo.

Bibliografia

- [1] N. Jindal, B. Liu, and E.-P. Lim, "Finding unusual review patterns using unexpected rules," in *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, ser. CIKM '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 1549–1552. [Online]. Available: <https://doi.org/10.1145/1871437.1871669> (Citato a pagina 6)
- [2] Hernández-Castañeda, H. Calvo, A. Gelbukh, and J. Flores, "Cross-domain deception detection using support vector networks," *Soft Computing*, vol. 21, 02 2017. (Citato a pagina 6)
- [3] A. Srivastava, M. Singh, and P. Kumar, "Supervised semantic analysis of product reviews using weighted k-nn classifier," in *2014 11th International Conference on Information Technology: New Generations*, 2014, pp. 502–507. (Citato a pagina 6)
- [4] X. Deng, R. Smith, and G. Quintin, "Semi-supervised learning approach to discover enterprise user insights from feedback and support," 2020. (Citato a pagina 7)
- [5] N. Cao, S. Ji, D. K. Chiu, M. He, and X. Sun, "A deceptive review detection framework: Combination of coarse and fine-grained features," *Expert Systems with Applications*, vol. 156, p. 113465, 2020. [Online]. Available:

<https://www.sciencedirect.com/science/article/pii/S095741742030289X> (Citato
a pagina 7)