FONDAMENTI DI INTELLIGENZA ARTIFICIALE

Project Documentation - TeachTuner

Sentiment Analysis for Education

Nocera Salvatore[1]* | **Mattia Fattoruso**[2]*

Students information

[1]Matricola: 0512117510

[2]Matricola: 0512118639

Correspondence

Dipartimento di Informatica, Università degli Studi di Salerno, Fisciano, SA, Italia

Contact details

s.nocera7@studenti.unisa.it

m.fattoruso11@studenti.unisa.it

Questo progetto applica tecniche di sentiment analysis al contesto educativo per automatizzare l'interpretazione dei feedback degli studenti. Il progetto prevede la raccolta di commenti tramite form personalizzati, la normalizzazione dei dati testuali e l'utilizzo di modelli di machine learning e deep learning per classificare le opinioni in sentiment positivi, negativi o neutri. I risultati sono presentati attraverso report intuitivi e visualizzazioni grafiche, offrendo uno strumento pratico e scalabile per migliorare la qualità dell'insegnamento e supportare decisioni informate basate sui dati.

1 | INTRODUCTION

TeachTuner è un progetto che esplora come la sentiment analysis possa essere applicata al contesto educativo. L'obiettivo è fornire a educatori e istituzioni uno strumento innovativo per raccogliere e analizzare feedback degli studenti, migliorando la qualità dell'insegnamento e ottimizzando le strategie didattiche. Utilizzando tecnologie di machine learning e deep learning, il progetto automatizza l'elaborazione del linguaggio naturale per classificare opinioni e sentimenti, favorendo decisioni informate e basate sui dati.

La documentazione che segue descrive in dettaglio le scelte metodologiche, le tecnologie utilizzate e le sfide affrontate, evidenziando i vantaggi di un sistema progettato per l'analisi emotiva nel settore educativo.

Link Repository Tutti i dettagli, il codice sorgente e la documentazione tecnica completa del progetto sono disponibili nella nostra repository ufficiale: `https://github.com/Pr1vat30/machine_learning_project.git`

1.1 | What is sentiment analysis

La sentiment analysis, nota anche come opinion mining, è un ramo dell'elaborazione del linguaggio naturale (NLP, Natural Language Processing) e della linguistica computazionale che si concentra sull'identificazione, l'estrazione e la quantificazione del tono emotivo e delle informazioni soggettive contenute nei dati testuali. Questo campo riveste

* Equally contributing authors.

un ruolo cruciale nella comprensione degli atteggiamenti, delle emozioni e delle opinioni espresse dagli individui nella comunicazione scritta o parlata. Analizzando il linguaggio, la sentiment analysis mira a classificare il testo come positivo, negativo o neutro e, in applicazioni più avanzate, a identificare emozioni specifiche come gioia, rabbia o tristezza.

Alla base della sentiment analysis vi è l'esigenza di comprendere il comportamento umano, le preferenze e gli atteggiamenti in un mondo sempre più dominato dalla comunicazione digitale. In un'epoca in cui i social media, le recensioni online e i feedback dei clienti sono centrali nelle interazioni, organizzazioni e ricercatori devono elaborare enormi volumi di dati testuali per trarre informazioni significative. La sentiment analysis colma questa lacuna automatizzando l'interpretazione del testo e offrendo un modo sistematico per monitorare l'opinione pubblica, valutare la soddisfazione dei clienti e analizzare le tendenze emotive nel tempo.

L'importanza della sentiment analysis trascende settori e discipline. Nel mondo del business, consente alle aziende di ottimizzare le strategie di marketing, migliorare le relazioni con i clienti e monitorare la percezione del brand. In politica, la sentiment analysis viene utilizzata per misurare l'opinione pubblica su politiche, campagne o candidati. Oltre alle sue applicazioni pratiche, lo studio della sentiment analysis contribuisce anche ai progressi nella linguistica computazionale e nell'intelligenza artificiale, ampliando le capacità delle macchine di comprendere il linguaggio umano.

I fondamenti teorici della sentiment analysis si basano sull'assunto che il linguaggio trasmetta non solo informazioni, ma anche segnali emotivi. Analizzando le proprietà semantiche e sintattiche del testo, gli algoritmi possono dedurre i sentimenti sottostanti. Ad esempio, la presenza di aggettivi come "eccellente" o "terribile", l'uso dei punti esclamativi o persino la disposizione sintattica delle frasi possono indicare un sentimento positivo o negativo.

Inoltre, la sentiment analysis tiene conto della natura contestuale del linguaggio, riconoscendo che il significato e il tono delle parole variano spesso a seconda del contesto. Per chiarire quanto detto riportiamo alcuni esempi:

- *"This course is amazing! The lectures are very engaging!"* -> è probabile che vanga classificata come positiva.
- *"The assignment was too difficult"* -> è probabile che vanga classificata come negativa.
- *"The exam was scheduled for Friday"* -> è probabile che vanga classificata come neutrale.

1.2 | How sentiment analysis work

Il funzionamento della sentiment analysis si basa su una combinazione di principi linguistici, modelli statistici e tecniche computazionali avanzate. Alla sua base, la sentiment analysis implica l'elaborazione e l'analisi dei dati testuali per classificarli in base al loro contenuto emotivo. Questo processo può essere suddiviso in diverse fasi fondamentali, ognuna delle quali svolge un ruolo cruciale nel trasformare il testo grezzo in intuizioni significative sul sentiment.

La prima fase della sentiment analysis è il data preprocessing, che consiste nella pulizia e preparazione dei dati testuali per l'analisi. I dati testuali raccolti da diverse fonti, come post sui social media, recensioni o email, sono spesso non strutturati e contengono rumore sotto forma di errori di battitura, gergo, emoji e informazioni irrilevanti.

Il preprocessing generalmente include passaggi come la tokenization, che suddivide il testo in singole parole o frasi; la rimozione di stopwords, ovvero parole comuni come "e" o "il" che non hanno un significato rilevante; e la standardizzazione del testo tramite la conversione in minuscolo o la riduzione delle parole alle loro radici (stemming).
Una volta preprocessato il testo, la fase successiva consiste nell'feature extraction, in cui il testo viene trasformato in

una rappresentazione numerica che può essere elaborata da algoritmi di machine learning. Questo può essere realizzato attraverso tecniche come il bag-of-words (BoW), il term frequency-inverse document frequency (TF-IDF) o le word embeddings. BoW e TF-IDF rappresentano il testo in base alla frequenza delle parole, mentre le word embeddings, come Word2Vec o GloVe, codificano le parole in vettori densi che catturano le relazioni semantiche.

Il cuore della sentiment analysis risiede nell'applicazione di algoritmi per classificare il testo in base al sentiment. Questi algoritmi possono variare da sistemi basati su regole a modelli di machine learning e, più recentemente, architetture di deep learning. I sistemi basati su regole si affidano a dizionari predefiniti di parole connotate dal punto di vista emotivo e a regole sintattiche per determinare la polarità del sentiment. Ad esempio, un approccio basato su regole potrebbe classificare una frase come positiva se contiene parole come "felice" o "straordinario". Sebbene semplice, questo metodo spesso fatica a gestire il linguaggio dipendente dal contesto e le espressioni complesse.

I modelli di machine learning, invece, utilizzano dataset etichettati per addestrare classificatori in grado di prevedere il sentiment. Algoritmi come la logistic regression, le support vector machines (SVMs) e il naive Bayes sono stati storicamente scelte popolari per la sentiment analysis. Questi modelli apprendono pattern dai dati riuscendo ad associare le caratteristiche del testo alle etichette di sentiment. Tuttavia, richiedono quantità significative di dati etichettati e spesso non riescono a cogliere le sfumature del linguaggio.

L'avvento del deep learning ha migliorato significativamente la precisione e la flessibilità della sentiment analysis. Le reti neurali, in particolare le recurrent neural networks (RNNs), le convolutional neural networks (CNNs) e le architetture basate su transformer, hanno dimostrato prestazioni eccezionali nella classificazione del sentiment.

La sentiment analysis può essere condotta a diversi livelli di granularità, tra cui:

- Document-level analysis, che valuta il sentiment complessivo di un testo, come un'intera recensione o articolo.
- Sentence-level analysis, che si concentra sul sentiment di singole frasi.
- Aspect-level analysis, che analizza il sentiment relativo a specifici attributi o caratteristiche all'interno del testo.

Nonostante i progressi effettuati, l'efficacia della sentiment analysis dipende dalla capacità di affrontare diverse sfide. L'ambiguità del linguaggio, la presenza di sarcasmo e ironia e la natura dinamica dei significati delle parole rappresentano ostacoli significativi. Inoltre, la necessità di elaborare dati multilingue e specifici per determinati domini richiede tecniche sofisticate e dataset di addestramento diversificati.

1.3 | Sentiment analysis in education

L'applicazione della sentiment analysis in ambito educativo si è dimostrata un approccio promettente per migliorare i risultati dell'apprendimento, favorire il coinvolgimento degli studenti e arricchire l'esperienza educativa. Attraverso l'analisi di dati testuali generati in contesti formativi — come feedback degli studenti, discussioni su forum online o contenuti accademici — questa tecnologia permette a educatori e istituzioni di comprendere meglio emozioni, atteggiamenti e bisogni degli studenti, offrendo una visione più completa del loro percorso formativo.

Uno dei principali impieghi della sentiment analysis è la valutazione dei feedback raccolti tramite sondaggi, valutazioni dei corsi o piattaforme digitali. Tradizionalmente, l'analisi manuale di questi dati è un processo lento e soggetto a errori, oltre che dispendioso in termini di tempo e risorse. La sentiment analysis automatizza questa attività, classificando i

commenti come positivi, negativi o neutri, e aiutando così a identificare punti di forza e aree di miglioramento in modo più efficiente ed oggettivo.

Un altro ruolo cruciale è il monitoraggio del benessere emotivo degli studenti. Analizzando testi provenienti da forum, social media o compiti, è possibile rilevare segnali di stress, frustrazione o disimpegno. Ad esempio, espressioni negative relative al carico di lavoro o alla difficoltà di un argomento potrebbero indicare la necessità di supporto aggiuntivo, come tutoraggi o sessioni di counseling. Questo approccio proattivo contribuisce a creare un ambiente di apprendimento più inclusivo e attento alle dimensioni emotive, spesso trascurate ma fondamentali per il successo formativo.

Nell'ambito dell'e-learning, questa tecnologia può personalizzare le esperienze formative in modo significativo. Studiando interazioni come post nei forum, consegne di compiti o chat, fornisce insight dettagliati sulle reazioni emotive degli studenti ai materiali didattici e alle metodologie di insegnamento. Questi dati permettono di modulare contenuti e risorse in base alle esigenze individuali, migliorando così sia il coinvolgimento sia i risultati di apprendimento.

La sentiment analysis è anche utile per analizzare le dinamiche delle interazioni in classe e le pratiche didattiche. Esaminando trascrizioni o registrazioni delle lezioni, è possibile valutare il tono emotivo e il livello di partecipazione degli studenti. Tendenze positive possono indicare strategie efficaci, mentre quelle negative potrebbero suggerire la necessità di modifiche, offrendo ulteriori strumenti per affinare i metodi pedagogici e creare un ambiente più coinvolgente.

Su scala più ampia, l'analisi di dataset come post sui social media o forum pubblici permette di misurare l'opinione pubblica su iniziative e riforme educative. I decisori politici, ad esempio, potrebbero utilizzarla per valutare l'impatto di un nuovo curriculum o di cambiamenti nei metodi di valutazione, assicurando che le decisioni siano allineate alle aspettative di studenti, genitori e insegnanti. Questo approccio basato sui dati può contribuire a costruire un sistema educativo più rispondente alle esigenze reali della comunità.

Tuttavia, l'applicazione della sentiment analysis nell'educazione non è priva di sfide. Questioni etiche e preoccupazioni legate alla privacy rappresentano ostacoli significativi. Le istituzioni, infatti, devono garantire che la raccolta e l'analisi dei dati rispettino le normative e i diritti degli studenti, adottando misure di sicurezza e trasparenza per proteggere le informazioni sensibili senza creare un senso di sorveglianza o limiti la libertà di espressione.

Inoltre, la natura ambigua del linguaggio richiede modelli in grado di cogliere sfumature specifiche. Ad esempio, una critica a un compito particolarmente impegnativo potrebbe esprimere impegno e determinazione anziché disinteresse, rendendo necessaria un'interpretazione accurata e contestualizzata. Per affrontare questa complessità, è fondamentale sviluppare algoritmi che integrino conoscenze specifiche, garantendo analisi più affidabili e significative.

2 | PROPOSED PROJECT

2.1 | Introduction

Nei seguenti paragrafi, descriveremo perché abbiamo deciso di affrontare il problema della sentiment analysis e perché ci siamo voluti focalizzare sul contesto educativo. Accenneremo inoltre alle prime scelte riguardanti come abbiamo deciso di strutturare il progetto e ad altre informazioni relative al problema in esame.

2.2 | Domain choices

2.1.1 - Why chose education

La scelta di applicare la sentiment analysis al contesto educativo nasce dalla crescente esigenza di comprendere meglio le percezioni, le opinioni e le esperienze degli studenti in relazione al materiale didattico, alle lezioni e agli approcci pedagogici adottati. In un panorama in cui la qualità dell'educazione è fortemente influenzata dalla capacità degli insegnanti e delle istituzioni di rispondere ai bisogni degli studenti, l'analisi automatizzata delle opinioni offre un'opportunità unica per raccogliere e sintetizzare informazioni critiche.

La sentiment analysis consente di identificare rapidamente tendenze, punti di forza e aree di miglioramento, riducendo il tempo e le risorse necessarie per l'analisi manuale del feedback. Inoltre, l'educazione rappresenta un campo in cui le emozioni e le percezioni svolgono un ruolo fondamentale nel determinare l'efficacia dei processi di apprendimento.

Concentrandoci su questo dominio applicativo, possiamo contribuire a migliorare la qualità del sistema educativo, fornendo ai docenti e agli amministratori strumenti per prendere decisioni basate sui dati. La nostra scelta riflette quindi la necessità di utilizzare tecnologie avanzate per migliorare l'efficienza e l'efficacia del settore educativo, valorizzando le opinioni degli studenti come fonte di informazioni strategiche.

2.1.2 - What we focus on

Nell'applicare la sentiment analysis al contesto educativo, abbiamo scelto di concentrarci su specifici aspetti che riguardano la valutazione di materiali didattici, lezioni e approcci pedagogici. Sebbene l'analisi delle emozioni e della salute mentale degli studenti sia un'area di interesse importante, abbiamo deliberatamente deciso di non focalizzarci su questo ambito, né sulla personalizzazione della didattica. Questi aspetti richiederebbero modelli altamente specializzati e l'adozione di misure etiche più rigorose, che esulano dagli obiettivi di questo progetto.

Invece, il nostro interesse si concentra sull'analisi delle recensioni e dei commenti relativi a contenuti come manuali, dispense e corsi online, nonché sul feedback riguardante le lezioni erogate e le metodologie di insegnamento. La scelta di questo focus deriva dalla volontà di offrire un supporto concreto agli educatori, fornendo loro strumenti per migliorare la qualità delle risorse e dei metodi utilizzati.

Questo approccio si distingue per la sua praticità e immediatezza, permettendo di identificare rapidamente aree di miglioramento e di adottare strategie che rispondano in modo diretto alle esigenze espresse dagli studenti. Tale scelta, inoltre, è in linea con la crescente importanza attribuita all'uso di dati per ottimizzare l'efficacia dell'educazione.

2.3 | Application choices

2.2.1 - Why chose web application

La scelta di sviluppare un'applicazione web come artefatto del progetto è stata determinata dalla necessità di garantire accessibilità, flessibilità e facilità d'uso a un vasto numero di utenti, tra cui studenti, docenti e amministratori. Le applicazioni web rappresentano un mezzo ideale per l'implementazione di strumenti basati su sentiment analysis, poiché consentono la raccolta in tempo reale di dati provenienti da diverse utenze; inoltre incentivano all'ammodernamento dei metodi didattici andando ad integrarle facilmente ad ogni tipo di contesto.

Rispetto ad altre tipologie di artefatti, come software desktop o applicazioni mobili, le applicazioni web offrono il vantaggio di essere indipendenti dalla piattaforma e di poter essere utilizzate su qualsiasi dispositivo dotato di un browser. Questo approccio consente inoltre un aggiornamento continuo delle funzionalità e l'implementazione di nuove metodologie di analisi senza la necessità di interventi complessi da parte degli utenti finali.

La nostra scelta riflette quindi l'importanza di fornire uno strumento versatile e scalabile, in grado di rispondere alle esigenze mutevoli del contesto educativo e di favorire la diffusione dell'innovazione tecnologica nel settore.

2.2.2 - What the web-app does

L'applicazione web sviluppata per questo progetto mira a offrire una soluzione pratica ed efficace per l'analisi delle opinioni e delle percezioni degli studenti, in linea con gli obiettivi identificati per il dominio applicativo. L'applicazione consente di raccogliere e analizzare automaticamente feedback relativi a materiali didattici, lezioni e approcci pedagogici, fornendo risultati chiari e sintetici sotto forma di report e metriche.

Questo strumento permette agli educatori di monitorare il sentiment generale degli studenti e di identificare eventuali aree di criticità, supportando un processo decisionale informato e basato sui dati.

L'applicazione è stata progettata per garantire un'esperienza utente intuitiva, rendendo semplice l'utilizzo delle funzionalità di analisi anche a personale non esperto di tecnologie avanzate. L'approccio adottato riflette la volontà di mettere a disposizione un sistema accessibile e altamente efficace per migliorare la qualità dell'educazione, valorizzando il ruolo centrale delle opinioni degli studenti nella progettazione di esperienze di apprendimento più coinvolgenti e significative.

2.4 | Methodology choices

2.3.1 - Why choose ML/DL

La scelta di focalizzarsi su approcci basati su machine learning (ML) e deep learning (DL) rispetto a metodi rule-based o algoritmi di ricerca tradizionali è stata guidata dalla complessità e dalla variabilità del linguaggio naturale utilizzato nei contesti educativi. Gli approcci rule-based, pur essendo semplici, risultano spesso inadeguati per gestire la polisemia, le espressioni idiomatiche e la complessità del feedback degli studenti, che può includere sarcasmo, ironia e ambiguità.

Al contrario, i modelli di ML e DL offrono la possibilità di apprendere direttamente dai dati, garantendo una maggiore capacità di adattamento alle sfumature linguistiche e ai contesti specifici. Le reti neurali profonde, in particolare, permettono di catturare relazioni semantiche e sintattiche complesse, migliorando significativamente la precisione dell'analisi del sentiment. Inoltre, l'utilizzo di tecnologie avanzate come BERT o GPT ha reso possibile lo sviluppo di modelli pre-addestrati che possono essere facilmente adattati al dominio educativo, riducendo i tempi e i costi di implementazione. La nostra scelta riflette quindi l'obiettivo di ottenere risultati accurati e affidabili, sfruttando il potenziale delle tecnologie più avanzate nel campo dell'elaborazione del linguaggio naturale.

2.3.2 - Pipeline step choices

Nello sviluppo dell'applicazione, particolare attenzione è stata dedicata alla scelta delle metodologie di preprocessing e dei modelli utilizzati, al fine di garantire risultati precisi e significativi nell'analisi del sentiment. Il preprocessing dei dati testuali ha incluso fasi fondamentali come la tokenizzazione, la rimozione di stopword, la lemmatizzazione e la gestione

dei caratteri speciali, per assicurare che i testi fossero adeguatamente normalizzati e pronti per l'analisi. Questi passaggi sono stati fondamentali per ridurre il rumore nei dati ed estrarre le caratteristiche linguistiche più rilevanti.

Per quanto riguarda i modelli, abbiamo scelto di utilizzare approcci classici di machine learning, come le Support Vector Machines (SVM) e la logistic regression, che offrono una solida capacità di catturare pattern nei dati strutturati e interpretabili. Questi modelli sono stati ulteriormente ottimizzati e addestrati su dataset specifici del dominio educativo, garantendo così una maggiore accuratezza nell'analisi delle opinioni relative ai materiali didattici e alle metodologie di insegnamento. La combinazione di un preprocessing rigoroso e di modelli classici ben consolidati ha permesso di ottenere un sistema performante, in grado di soddisfare le esigenze del progetto e di fornire risultati utili e affidabili.

3 | MACHINE LEARNING PIPELINE

Per lo sviluppo del progetto, abbiamo adottato il modello CRISP-DM (Cross-Industry Standard Process for Data Mining), una metodologia flessibile e consolidata per la gestione di progetti di data science. La scelta del CRISP-DM è stata motivata dalla natura del nostro progetto, che è di dimensioni relativamente ridotte e non coinvolge un numero elevato di figure professionali con competenze altamente differenziate. In questo contesto, la necessità di garantire una rigorosa socio-technical congruence tra i diversi attori non rappresenta una priorità assoluta.

3.1 | Business understanding

Prima di iniziare a lavorare con i dati, è opportuno effettuare un riepilogo delle scelte prese e delle decisioni effettuate fino a questo momento. Come descritto nel paragrafo 2.1, abbiamo deciso di focalizzarci sull'ambito educational, con particolare attenzione all'analisi dei commenti e delle recensioni fornite dagli studenti. Su tali commenti verrà applicata una document-level sentiment analysis, mentre eventuali miglioramenti o approfondimenti derivanti da livelli di granularità più elevati saranno lasciati alle considerazioni finali.

Si darà priorità allo sviluppo e al testing di modelli basati sul machine learning (ML) o deep learning (DL). Per ciascun modello, saranno dettagliate le caratteristiche specifiche adottate nel nostro caso di studio. Successivamente, ogni modello sarà valutato e confrontato con gli altri, al fine di individuare la soluzione più performante da implementare e utilizzare nell'applicazione finale.

Nel corso di questa analisi, prenderemo in considerazione anche l'impatto dei diversi tipi di word embedding sulle prestazioni dei modelli. Una volta completate tutte le analisi e le valutazioni necessarie, procederemo con lo sviluppo dell'applicazione vera e propria, integrando la soluzione più efficace emersa.

3.2 | Data understanding

I modelli che intendiamo testare saranno addestrati su un dataset contenente commenti e recensioni lasciati dagli studenti, relativi a diversi aspetti legati alla didattica. Sebbene sia prassi comune per le istituzioni raccogliere questo tipo di dati, tali informazioni non sono spesso rese pubbliche a causa di vincoli legati alla privacy. Questo limita la possibilità di reperire dataset già predisposti e di dimensioni adeguate per l'addestramento di un modello. Di conseguenza, si rende necessario esplorare altre fonti di dati, come forum o blog di discussione, che possano offrire un'alternativa valida.

Nel corso delle nostre ricerche, abbiamo individuato due piattaforme da cui è possibile ricavare feedback degli studenti su tematiche didattiche: Coursera e RateMyProfessor. Entrambi i siti sono molto popolari e contengono, in forma esplicita o implicita, commenti lasciati dagli studenti. Nello specifico, Coursera si concentra sulle opinioni relative ai corsi online seguiti, mentre RateMyProfessor raccoglie recensioni sui docenti e sui loro metodi di insegnamento.

La combinazione delle due fonti appare particolarmente promettente, in quanto in grado di coprire molte delle principali tematiche legate all'ambito dell'istruzione. Per tale motivo, abbiamo deciso di verificare l'esistenza di dataset, creati attraverso operazioni di web scraping su questi due siti, che includano i commenti degli studenti, unitamente a eventuali altre informazioni utili.

Sebbene avessimo potuto ricavare i dati direttamente tramite strumenti di web scraping sviluppati ad hoc, tale operazione avrebbe richiesto la realizzazione di programmi specifici per l'estrapolazione delle informazioni. Tuttavia, dato che il focus di questo progetto è la costruzione e la valutazione dei modelli, abbiamo deciso di non investire tempo in questa fase, concentrandoci invece sulle attività principali.

Questa scelta, tuttavia, ci impone di accettare i dataset così come sono stati creati dai rispettivi autori, limitando la possibilità di personalizzarne la struttura. Di conseguenza, sarà necessario intraprendere operazioni aggiuntive per adattarli alle esigenze specifiche del nostro caso d'uso. Non essendoci voluto molto per trovare due dataset, disponibili per il download e realizzati a partire dalle informazioni estrapolate dai siti precedentemente individuati, siamo passati direttamente alle fasi successive.

Il primo dataset, relativo alla piattaforma Coursera, è stato individuato su Kaggle e, per semplicità, faremo riferimento ad esso come coursera-reviews. Tale dataset contiene diverse informazioni riguardanti i corsi offerti sulla piattaforma, tra cui il commento testuale lasciato dagli utenti e la valutazione assegnata in termini di stelle (da 1 a 5). Questi due elementi rappresentano gli aspetti di maggiore interesse per il nostro progetto.

Il secondo dataset, relativo alla piattaforma RateMyProfessor, è stato individuato su Hugging Face e, da ora in poi, lo indicheremo come rate-my-professor. Esso include una serie di dati, tra cui la media delle valutazioni attribuite ai docenti e il dipartimento di appartenenza. Tuttavia, la nostra attenzione si concentra, ancora una volta, sui commenti lasciati dagli studenti, i quali non risultano pre-etichettati per quanto riguarda il sentiment.

Considerata la scarsa disponibilità di alternative facilmente accessibili, riteniamo opportuno partire dai due dataset individuati, e sottoporli ad una fase di raffinamento che comprenda operazioni di pulizia e rietichettatura, ove necessario. Al termine di questa fase, intendiamo combinare i due dataset raffinati, creandone uno più robusto e adeguato ai nostri obiettivi. Anche se non è focus di questo progetto, in futuro potrebbe essere utile esplorare ulteriori tecniche avanzate di web scraping o altre metodologie, al fine di migliorare la qualità complessiva del dataset a disposizione.

3.3 | Data engineering

3.3.1 - Datasets refine process

Elaborazione coursera-reviews Partiamo con l'elaborazione del primo dataset, composto da circa un milione di record, ciascuno contenente diverse feature. Tra queste, le informazioni di nostro interesse sono il comment e il rating.

Sebbene il dataset non presenti una label esplicita che indichi il sentiment espresso nei commenti, è possibile dedurlo con un'accuratezza accettabile basandosi sul valore del rating. Come indicato dal creatore del dataset, il rating è espresso su una scala da 1 a 5 stelle, e a ciascun valore è possibile associare una specifica polarità di sentiment:

- 1 stella –> estremamente negativo
- 2 stelle –> negativo
- 3 stelle –> neutro
- 4 stelle –> positivo
- 5 stelle –> estremamente positivo

Non disponendo di prove concrete sull'affidabilità di questa associazione, abbiamo deciso di condurre un'analisi manuale su un campione rappresentativo, verificando le prime 100 entry per ciascuna delle classi di rating (1-5).

Dai risultati ottenuti, l'associazione proposta appare accurata per le classi più estreme (1-2 e 4-5), poiché i commenti riflettono chiaramente la volontà dell'autore di esprimere giudizi positivi o negativi sul corso. Diversamente, per i commenti con rating pari a 3 stelle, l'identificazione della neutralità risulta più complessa, data la natura ambigua del linguaggio utilizzato. Nonostante queste incertezze, abbiamo deciso di accettare l'associazione proposta, anche in considerazione del fatto che i rating pari a 3 costituiscono una netta minoranza rispetto alle altre classi.

A questo punto, abbiamo creato una nuova label contenente il sentiment associato a ciascun rating (1-2 -> negative, 3 -> neutral, 4-5 -> positive), abbiamo rimosso le feature non rilevanti e salvato il dataset ottenuto. Un ulteriore passaggio necessario prima di passare al raffinamento del secondo dataset riguarda l'eliminazione dei commenti scritti in lingue diverse dall'inglese. Durante l'analisi delle entry del dataset, infatti, è emerso che, seppur in misura minima rispetto al totale, sono presenti commenti effettuati in diverse lingue.

Poiché il nostro progetto prevede l'utilizzo esclusivo della lingua inglese, tali entry saranno rimosse. L'operazione di rimozione sarà effettuata solo su questo dataset in quanto il secondo non presenta tale problematica. Per garantire coerenza con i nostri obiettivi, abbiamo scelto di eliminare i record anziché sostituirli o tradurli, considerando che la dimensione iniziale del dataset è ampiamente sufficiente per il nostro scopo.

Elaborazione rate-my-professor Il secondo dataset individuato è composto da circa 300.000 record. Anche in questo caso, solo alcune feature sono rilevanti per il nostro progetto, nello specifico i commenti testuali lasciati dagli studenti. A differenza del dataset precedente, qui non è presente alcun elemento che consenta di annotare sistematicamente il sentiment dei commenti. Di conseguenza, abbiamo deciso di ricorrere a metodi alternativi per l'etichettatura dei dati.

Data la grande quantità di record, l'annotazione manuale non è una soluzione praticabile. Per questo motivo, abbiamo optato per l'utilizzo di un modello pre-addestrato specifico per la classificazione dei testi. Poiché la nostra analisi prevede la classificazione dei sentimenti in tre categorie – negativo, positivo e neutro – abbiamo selezionato il modello Twitter-roBERTa-base for Sentiment Analysis, disponibile su Hugging Face.

Questo modello, chiamato formalmente TimeLM-21 nel relativo paper, oltre al fatto che è uno dei più utilizzati per la classificazione in tre classi, come dimostrato dal numero di download per mese, è risultato particolarmente adatto alle nostre esigenze per due principali motivi, ovvero:

1. Secondo i risultati del paper, ha superato altri modelli considerati state-of-the-art in task di sentiment analysis, dimostrando notevoli capacità di predizione.
2. È stato addestrato su un dataset di grande dimensione (circa 125M) aggiornato agli ultimi cinque anni, un periodo coerente con la diffusione di sondaggi didattici e strumenti simili.

Sebbene l'uso di un modello pre-addestrato introduca un margine di errore nella classificazione, riteniamo che questo non comprometta la qualità complessiva del nostro lavoro. Valutare con precisione l'impatto di tali errori richiederebbe un confronto con altri modelli equivalenti, ma ciò esula dagli obiettivi del nostro progetto.

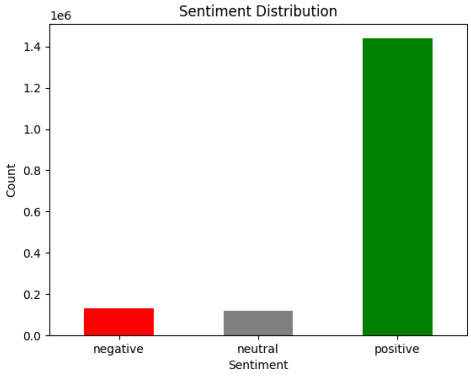
Inoltre, considerando la dimensione e l'eterogeneità del dataset usato per addestrarlo, riteniamo che il modello offra un'adeguata generalizzazione utilizzabile senza introdurre troppo errore per l'etichettatura del nostro dataset.

Anche qui, una volta completata la classificazione dei commenti, abbiamo eliminato tutte le feature superflue, conservando solo quelle rilevanti, e salvato il dataset risultante per le fasi successive. Infine, dopo aver raffinato i due dataset separatamente, li abbiamo combinati e siamo passati alla fase di analisi.

3.3.2 - Refine Dataset Analysis

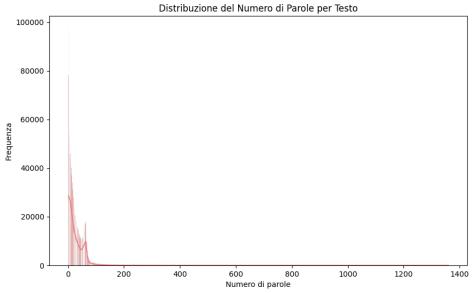
Una volta ottenuto il dataset, una delle prime e più rilevanti operazioni effettuate è stata l'analisi della distribuzione delle classi. La presenza di un dataset bilanciato è di fondamentale importanza, in quanto uno squilibrio significativo tra le classi potrebbe introdurre problemi di bias e varianza, compromettendo l'efficacia dell'addestramento dei modelli.

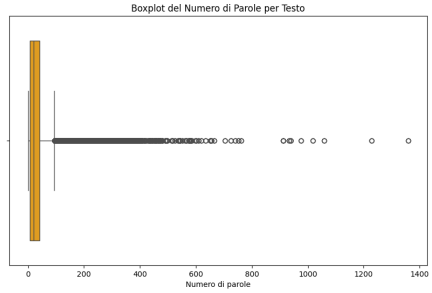
A tal proposito, è stato osservato che entrambi i dataset utilizzati nella fase precedente presentavano uno squilibrio a sfavore della classe positiva. Questo andamento si è confermato anche nel dataset unificato e raffinato.



Fortunatamente, oltre allo sbilanciamento, non sembrano esserci particolari problemi per quanto riguarda il livello di generalizzazione del dataset. Quest'ultima risulta evidente se osserviamo il word cloud, ma è stata anche riscontrata attraverso un'analisi più approfondita condotta usando gli n-gram.

Abbiamo infine analizzato alcune statistiche descrittive relative ai commenti, concentrandoci in particolare sulla lunghezza media dei commenti presenti nel dataset. Quest'analisi si è dimostrata utile per comprendere meglio le caratteristiche stilistiche e semantiche dei commenti, fornendo informazioni sufficienti per proseguire.





3.3.3 - Dataset balancing

Prima di eseguire operazioni come la lemmatizzazione o la rimozione delle stopword, è fondamentale bilanciare la distribuzione delle classi all'interno del dataset. Le strategie principali per affrontare questa problematica sono l'undersampling e l'oversampling.

Poiché il dataset è fortemente sbilanciato verso la classe positiva, è necessario adottare una combinazione di queste tecniche per ottenere un dataset processabile con le risorse hardware disponibili, mantenendo al contempo un campione rappresentativo e abbastanza generalizzato per ogni classe.

Operazioni preliminari Considerando che la classe positiva rappresenta una porzione significativamente maggiore rispetto alle altre classi, è necessario ridimensionarla, sia per la mole di dati eccessiva rispetto alla capacità di elaborazione delle nostre macchine, sia per evitare che i modelli sovrastimino tale classe.

Come prima operazione di undersampling, abbiamo effettuato una riduzione generale del volume del dataset sulla base delle informazioni raccolte nella precedente fase di analisi. In tale fase, infatti, è emerso che la maggioranza dei commenti ha una lunghezza inferiore alle 50 parole.

Pertanto, è stato deciso di eliminare quelli che, in senso lato, possono essere considerati degli outlier. Tale operazione ha contribuito a ridurre, anche se minimamente, le dimensioni del dataset originale in maniera controllata, senza comportare una significativa perdita di informazione.

Abbiamo, inoltre, ridotto i commenti "singleton" composti da una sola parola. Sebbene questi possano essere utili ai modelli per comprendere come influenzino il significato complessivo di una frase, tali commenti non rappresentano una casistica frequente nel contesto di recensioni o commenti reali. Per questo motivo, si è deciso di conservarne solo una parte, in modo tale da mantenere una base sufficiente per i modelli senza compromettere la rappresentatività

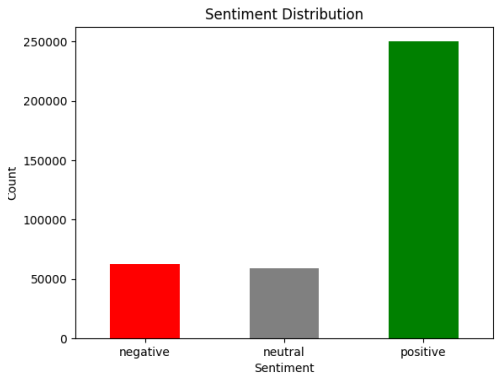
Grazie a queste operazioni, abbiamo concentrato l'attenzione sulla porzione più rilevante del dataset, rendendo il successivo undersampling meno impattante rispetto a un approccio diretto.

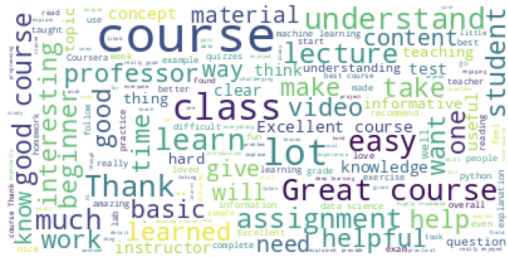
Undersampling Essendo le operazioni di undersampling e oversampling particolarmente delicate, in quanto possono facilmente generare problemi come bias, rumore, overfitting e underfitting, è essenziale definire una strategia efficace che ne consenta un uso combinato, bilanciato ed efficace.

L'obiettivo della fase di bilanciamento che ci siamo posti è raggiungere un dataset di circa 600 mila record, suddiviso in 480 mila per il training e 120 mila per il testing. Poiché le classi negativa e neutra contano attualmente circa 100 mila record complessivi, è stato deciso di ridurre la classe positiva a 250 mila record (attraverso undersampling) e di aumentare le altre due classi fino a circa 150 mila record ciascuna (attraverso oversampling). In questo modo riusciamo a mantenere una leggera predominanza della classe positiva, e contemporaneamente ad incrementare il peso delle classi meno rappresentate, ritrovandoci con una distribuzione complessiva del tipo 40% - 30% - 30%.

Per la riduzione della classe positiva, al fine di evitare somiglianze eccessive tra i dati che potrebbero ridurre la capacità di generalizzazione del modello, è stato scelto di adottare un campionamento casuale. Tale approccio è stato preferito a metodi più avanzati, come quelli basati sulla similarità tra le classi, per ragioni di semplicità ed efficienza computazionale oltre che per evitare possibile overfitting.

Nonostante l'undersampling comporti una significativa perdita di informazione, questa scelta è stata dettata dalla limitata capacità di calcolo disponibile e dalla necessità di garantire un dataset bilanciato e rappresentativo.





Oversampling Come visto ridurre esclusivamente il numero di elementi della classe positiva fino a renderla uniforme con le altre comporterebbe una drastica diminuzione della dimensione complessiva del dataset, aumentando il rischio di bias e compromettendo la capacità del modello di apprendere pattern significativi.

Per tale motivo, dopo le operazioni di undersampling, abbiamo applicato una serie di operazioni di oversampling per aumentare la presenza delle classi negativa e neutra, cercando di limitare l'introduzione di rumore e di overfitting.

Tra le tecniche disponibili, abbiamo considerato metodi come SMOTE per generare nuovi dati a partire da quelli esistenti. Tuttavia, data la ridotta dimensione delle classi negativa e neutra, l'applicazione di tali tecniche avrebbe generato campioni troppo simili tra loro, aumentando il rischio di overfitting.

Abbiamo quindi optato per un approccio basato sulla generazione di campioni sintetici utilizzando Large Language Models (LLM). In particolare, abbiamo sfruttato il tool Ollama per eseguire alcuni modelli localmente. Considerando le capacità hardware dei nostri dispositivi, abbiamo installato e testato tramite Ollama i seguenti modelli:

- Deepseek-r1 (7B) -> recente modello con capacità di reasoning simili a quelle di GPT-4o
- Llama3.1 (8B) -> modello specializzato in comprensione del linguaggio, coerenza e contestualizzazione
- Mistral (7B) -> modello ottimizzato per velocità e leggerezza, ideale per scenari che richiedono basse latenze

Per deicidere quale di questi modelli utilizzare per la generazione dei commenti, li abbiamo confrontati e valutati in base a tre criteri principali, ovvero: 1) varietà lessicale utilizzata per generare le risposte. 2) tempo di elaborazione necessario per generare l'output. 3) coerenza del contenuto rispetto alle indicazioni fornite.

Per ciascun modello, sono stati quindi iterati 5 volte gli stessi prompt e valutati i risultati. I test hanno evidenziato che:

- Deepseek-r1 richiede tempi di elaborazione maggiori per generare l'output e, pur producendo contenuti coerenti, non rispetta il formato richiesto. Tale comportamento sembra derivare dalle sue capacità di reasoning, le quali, tuttavia, non sono state esplorate a sufficienza. Pertanto, è stato escluso dal processo.
- Llama e Mistral hanno performato in modo simile per tempi di risposta, contenuto e formato, tuttavia:
  - Mistral ha generato commenti negativi con maggiore varietà lessicale.
  - Llama si è dimostrato più efficace nella generazione di commenti neutri.

Abbiamo quindi deciso di utilizzare Mistral per generare commenti negativi e Llama per quelli neutri.

Approccio generale La generazione di commenti sintetici è stata affrontata testando due approcci distinti e procedendo successivamente con quello più efficace. Entrambi condividono una base comune, ovvero il modo in cui interagiamo con il modello linguistico (LLM) e il tentativo di introdurre la maggiore varietà possibile nelle risposte generate.

Entrambi gli approcci prevedono un'interrogazione iterativa del modello, al quale viene inviato un prompt contenente:

- Esempi di input forniti e output attesi (few-shot learning)
- Richiesta del contenuto e modalità per generarlo
- Descrizione della struttura e del formato dell'output

In virtù di quanto detto, tutti i prompt utilizzati sarranno strutturati più o meno nel seguente modo:

```
f""" Here are some example reviews of lessons. Please note the format is a single, short sentence:
Example for Neutral: [Neutral example sentence].
Example for Positive: [Positive example sentence].
Example for Negative: [Negative example sentence].

Now, please generate a review based on the following instructions:
{instruction_or_prompt}

Respond ONLY with the new sentence and MAX {word_limit} words. """
```

Approccio basato su dati esistenti Questo approccio prevedeva la generazione di nuovi commenti con un sentiment specifico, partendo dalle entry già presenti nel dataset originale. L'obiettivo era "trasformare" parte dei commenti positivi scartati durante l'undersampling in commenti negativi o neutri, aumentandone sia il numero che la varietà, mantenendo al contempo un'aderenza al dataset originale.

Inizialmente, questa strategia ha funzionato discretamente: per le prime centinaia di iterazioni, il modello generava copie di commenti positivi già esistenti, trasformandoli in negativi tramite l'aggiunta di negazioni o la sostituzione di avverbi. Tuttavia, dopo un certo numero di iterazioni, il modello tendeva a effettuare modifiche ripetitive o a creare strutture sintattiche molto simili tra loro.

Per affrontare questo problema, abbiamo variato periodicamente (circa ogni 2.500 risposte) alcuni parametri del prompt. In particolare, abbiamo variato sia il numero che la complessità degli esempi forniti. Purtroppo, queste modifiche non hanno avuto un impatto rilevante sulla generazione di nuovi record, soprattutto per i commenti neutri.

La difficoltà maggiore risiedeva nella generazione di commenti neutri: mentre per i commenti negativi il modello poteva semplicemente aggiungere negazioni o sostituire parole con i loro contrari, per i commenti neutri era necessario ristrutturare completamente la frase.Di conseguenza, abbiamo deciso di concentrare l'attenzione unicamente sui commenti negativi, rimandando la generazione di commenti neutri a una strategia alternativa.

Dopo aver generato circa 15.000 commenti negativi, ne abbiamo estratto casualmente un campione di 1.000 per valutarne la varietà. Grazie al legame con il dataset originale, il contenuto lessicale si è rivelato sufficientemente variegato, sebbene con alcune ridondanze: parole come course o lecture risultavano eccessivamente frequenti.

Questa problematica, già osservata durante l'analisi preliminare del dataset, se ignorata, rischia di introdurre rumore aggiuntivo . Inoltre, la questione della generazione di commenti neutri restava irrisolta.

La combinazione di queste problematiche ci ha portato a stabilire una nuova strategia, questa volta non più basata sui dati già disponibili ma basata su una lista di istruzioni da noi creata ed integrata nella struttura prima vista.

Approccio basato su lista istruzioni Per affrontare le limitazioni del primo approccio, abbiamo adottato una nuova strategia non basata sui dati esistenti, ma su prompt personalizzati integrati nella struttura precedentemente descritta.

In questo approccio, invece di fornire un commento da trasformare, il modello riceveva istruzioni per generare liberamente una recensione su un tema specifico. Questi temi, scelti da una lista di circa 30 argomenti legati alla didattica e al mondo dell'educazione, sono stati progettati per coprire un ampio spettro di contenuti e ridurre il rischio di generare commenti simili.

Ad ogni iterazione, il prompt includeva uno degli argomenti scelto casualmente dalla lista, permettendo al modello di generare risposte significativamente diverse tra loro. Questa casualità, unita alla rotazione periodica degli esempi e delle informazioni incluse nel prompt, ha garantito una generazione più efficiente, variegata e qualitativamente superiore rispetto al precedente approccio.

Grazie a questa strategia, siamo riusciti anche a generare commenti neutri con maggiore successo. Modificando

leggermente gli esempi forniti e le istruzioni, pur mantenendo invariata la lista di argomenti, il modello è stato in grado di produrre commenti neutri seguendo le strutture fornite.

Generazione dei commenti negativi e neutri Con quest'ultima strategia, siamo arrivati a generare circa 50.000 commenti negativi. Mentre cambiamenti agli esempi forniti non hanno prodotto significativi cambiamenti nel comportamento del modello, le variazioni alla lista di istruzioni hanno avuto un impatto significativo sulle risposte generate, introducendo una maggiore varietà sia negli argomenti sia nella sintassi utilizzata, rimanendo comunque controllata.

Per quanto riguarda i commenti neutri, il problema principale è legato all'ambiguità intrinseca del linguaggio, che rende difficile definire chiaramente un commento neutro. Questo è un aspetto noto della sentiment analysis, paragonabile alla sfida posta dall'ironia o da altre forme retoriche. Tecniche avanzate, come l'aspect-based sentiment analysis (ABSA), possono identificare la polarità di specifici aspetti di una frase e determinare se il commento sia neutro, ma tali tecniche richiedono una raccolta dati più complessa e un addestramento articolato.

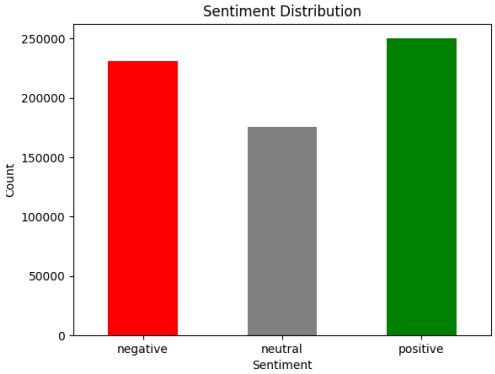
Nel nostro progetto, avendo optato per una sentiment analysis a livello di documento, abbiamo dovuto trovare il modo di rendere evidenti alcune caratteristiche strutturali per simulare l'analisi della polarità.

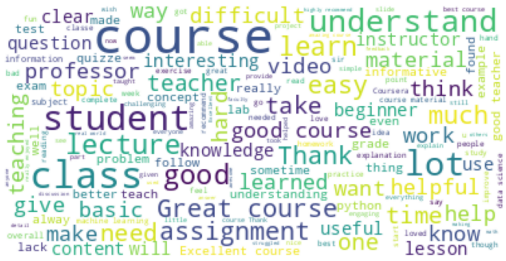
Fondamentale in questo è stato il few shot learning, in quanto fornendo esempi di commenti neutri con una chiara separazione degli aspetti (ad esempio, tramite l'uso di connettivi come but, however o while), il modello è stato in grado di generare nuovi commenti strutturati in modo simile.

Abbiamo quindi proseguito con la generazione di commenti neutri, variando regolarmente la lista delle istruzioni fino a raggiungere un totale di 50.000 esempi. A questo punto, abbiamo concluso le operazioni di oversampling, integrando quanto prodotto con il dataset principale. Successivamente, siamo passati all'analisi del risultato complessivo ottenuto dalle operazioni di bilanciamento effettuate.

3.3.4 - Balanced Dataset Analysis

Con le operazioni effettuate, abbiamo ottenuto un dataset completo, sufficientemente generalizzato e ben distribuito tra le tre classi di sentiment. Concludiamo questa prima fase di preprocessing con un'analisi del dataset finale, che una volta finito, utilizzeremo per l'addestramento dei modelli.





Confrontando la distribuzione e il word cloud del dataset elaborato fino a questo punto con quello di partenza, si osserva un netto miglioramento nella distribuzione, nonostante la rimozione di una parte significativa dei dati originali.

Abbiamo mantenuto una buona generalizzazione, evitando interventi invasivi durante l'oversampling. In particolare, il rumore e l'overfitting sono stati controllati accuratamente, grazie alla selezione precisa delle istruzioni iterate in ogni ciclo. Eventuali criticità non derivano dunque dalle scelte adottate, ma dalla natura degli strumenti utilizzati.

L'obiettivo iniziale delle operazioni di bilanciamento era raggiungere circa 150 mila elementi per le classi negative e neutral, mantenendo una leggera prevalenza della classe positiva per garantirne una rappresentanza adeguata.

Mentre la generazione di entry negative è risultata soddisfacente, le entry neutre hanno sollevato fin dal primo momento delle perplessità. Queste non sono state causate dalle modalità di generazione adottate, ma piuttosto dalla capacità del LLM nel comprendere appieno le nostre richieste e seguire gli esempi forniti.

Per assicurarci che non ci fosse stato qualche problema, abbiamo deciso di rietichettare le entry neutre utilizzando lo stesso modello impiegato per il dataset rate-my-pofessor. Il risultato finale mostra che la maggior parte delle entry generate è effettivamente neutra, sebbene una parte sia stata riclassificata come positiva o negativa.

Questa situazione ha portato a una distribuzione non perfettamente bilanciata. Tuttavia, riteniamo che tale discrepanza sia accettabile e non giustifichi ulteriori modifiche. Procederemo quindi alle fasi successive dell'analisi.

3.3.5 - Data preprocessing

Il preprocessing è una fase fondamentale nell'elaborazione di un dataset utilizzato per elaborazione di linguaggio naturale. Consiste nell'applicare una serie di trasformazioni ai dati per migliorarne la qualità e renderli adatti all'analisi.

Questa fase è cruciale per eliminare elementi di disturbo, uniformare i dati e facilitare l'identificazione di pattern rilevanti. Un preprocessing accurato non solo riduce il rumore nei dati, ma aumenta l'efficacia e la precisione dei modelli di machine learning o deep learning. Nel nostro caso, al dataset generato, abbiamo effettuato le seguenti trasformazioni:

1. Lowercase: convertire tutto il testo in minuscolo è una tecnica di base per evitare che variazioni nella capitalizzazione, come "Buono" e "buono", vengano interpretate come termini differenti dal modello.
2. Clean text spaces and digits: rimuovere spazi ridondanti e disordinati garantisce un testo strutturato correttamente, migliorando il processo di tokenizzazione e l'efficienza dei successivi passaggi. Ciò vale anche per i numeri, i quali non rappresentano informazioni rilevanti per il modello.
3. Remove punctuation: la punteggiatura spesso non aggiunge informazioni rilevanti in un contesto di Sentiment Analysis. Rimuoverla consente di focalizzarsi esclusivamente sul contenuto semantico del testo.
4. Remove long lines: le righe molto lunghe possono contenere rumore, come descrizioni prolisse o informazioni irrilevanti. Eliminandole, si semplifica il dataset e si concentra l'analisi su dati più rilevanti. Questa scelta è stata presa anche in considerazione della media della lunghezza delle recensioni, la quale ha evidenziato una scarsa presenza di commenti lunghi più di 50 parole ciascuna.
5. Remove stopwords: le stopwords sono parole molto frequenti e comuni, come articoli, congiunzioni o preposizioni (ad esempio: "e", "ma", "di"). Questi termini, pur essendo utili a livello sintattico, non apportano valore semantico

all'analisi. La loro rimozione riduce il volume di dati elaborati dal modello e permette di concentrarsi sulle parole che hanno un impatto diretto sull'identificazione del sentimento.

6. Lemmatization: la lemmatizzazione è una fase chiave per uniformare il vocabolario del testo. Questo processo riduce le parole alla loro forma base o lemma, considerando il loro significato. Ad esempio, "camminando", "camminava" e "camminato" vengono ricondotti a "camminare". Ciò permette di eliminare informazioni che non influenzano il contenuto semantico e aiuta il modello a riconoscere i concetti, piuttosto che le sole occorrenze superficiali delle parole. Questa operazione è essenziale per migliorare la coerenza e ridurre la complessità del dataset.
7. Remove short lines: le righe molto brevi contengono raramente informazioni utili, come commenti privi di contesto o espressioni generiche. La loro eliminazione garantisce un dataset più pulito e rappresentativo. In questa trasformazione andiamo principalmente ad eliminare valori nulli generati dalle fasi precedenti.

3.3.6 - Data embedding

Nel campo del Natural Language Processing (NLP), la rappresentazione del testo è uno step fondamentale per trasformare i dati testuali, intrinsecamente non strutturati, in un formato numerico che i modelli di machine learning o deep learning possano interpretare. Poiché i modelli lavorano esclusivamente con numeri, convertire parole o frasi in vettori numerici consente di analizzare, elaborare e trarre informazioni significative dal linguaggio naturale.

tecniche di rappresentazione del testo mirano a preservare il significato semantico, le relazioni sintattiche e i contesti delle parole. Le metodologie più semplici includono il bag of words (BoW) e il TF-IDF (Term Frequency-Inverse Document Frequency) , che si basano su contatori di frequenze per costruire vettori di rappresentazione. Tuttavia, queste tecniche ignorano l'ordine delle parole e i loro significati contestuali.

Con l'avanzamento dell'NLP, tecniche più sofisticate come i word embeddings sono state sviluppate per superare i limiti dei metodi tradizionali. I word embeddings mappano le parole in uno spazio vettoriale continuo in cui la similarità semantica tra le parole è preservata. Questo approccio ha rivoluzionato il campo, poiché permette di rappresentare le parole in modo denso e ricco di significato. Di seguito elenchiamo le tecniche di embedding che abbiamo deciso di utilizzare per valutare quanto le performance dei modelli dipendano da esse

TF-IDF Embedding La tecnica TF-IDF (Term Frequency-Inverse Document Frequency) è uno degli approcci più conosciuti e utilizzati per rappresentare il testo in modo numerico. Il suo obiettivo è assegnare a ogni parola un peso che rifletta la sua importanza in un documento specifico, tenendo conto anche della sua rilevanza rispetto all'intero corpus. In altre parole, TF-IDF permette di identificare quanto una parola contribuisce al significato di un documento, escludendo termini comuni che appaiono ovunque, come articoli o congiunzioni.

TF-IDF combina due concetti chiave: Term Frequency (TF), che misura quanto frequentemente una parola appare in un documento specifico; Inverse Document Frequency (IDF), che valuta quanto una parola è rara nel corpus complessivo, penalizzando i termini troppo comuni. Il risultato è una rappresentazione in cui ogni documento è descritto da un vettore di parole pesate in base alla loro importanza relativa.

TF-IDF è semplice da implementare e particolarmente efficace per applicazioni di machine learning tradizionali, come SVM o Naive Bayes. È ampiamente utilizzato per classificare documenti, recuperare informazioni (come nei motori di ricerca) o rilevare spam. La sua leggerezza lo rende ideale per scenari in cui le risorse hardware sono limitate.

TF-IDF presenta anche alcune criticità. Non considera il contesto delle parole: termini ambigui come "banca" (che può riferirsi a un fiume o a un istituto finanziario) vengono trattati allo stesso modo, indipendentemente dalla frase. Inoltre, non riesce a cogliere relazioni semantiche o sinonimi. Queste limitazioni lo rendono inadatto a compiti più complessi, come la generazione di testo o l'analisi avanzata del sentiment, dove tecniche più sofisticate sono preferibili.

Word2Vec Embedding Word2Vec è una delle tecniche di word embedding più rivoluzionarie nella storia dell'elaborazione del linguaggio naturale. Sviluppata da Google nel 2013, si basa su reti neurali per rappresentare ogni parola come un vettore in uno spazio continuo, dove parole simili si trovano vicine. Questo approccio consente di catturare sia il significato semantico delle parole sia le loro relazioni contestuali.

Word2Vec utilizza due metodi principali per generare queste rappresentazioni:

- CBOW (Continuous Bag of Words), che prevede una parola basandosi sul contesto circostante. Ad esempio, dato un frammento come "Il gatto — sul divano", il modello può prevedere "dorme".
- Skip-Gram, che lavora in modo opposto, cercando di prevedere il contesto circostante partendo da una parola specifica. Ad esempio, partendo da "cane", il modello può prevedere parole come "abbaia" o "giardino".

Aspetto rivoluzionario di Word2Vec è la capacità di catturare relazioni semantiche attraverso i vettori. Parole con significati simili si trovano vicine nello spazio vettoriale, mentre relazioni come sinonimia o analogia emergono naturalmente. Ad esempio, Word2Vec è in grado di rappresentare relazioni come "re" e "regina".

Word2Vec supera le limitazioni delle tecniche precedenti come TF-IDF, producendo rappresentazioni dense che catturano meglio le relazioni tra parole. È ideale per task come il clustering di documenti, la traduzione automatica e la classificazione testuale. Inoltre, la sua efficienza e flessibilità ne hanno fatto una pietra miliare nell'NLP.

Nonostante i suoi punti di forza, Word2Vec ha alcune debolezze. Le rappresentazioni generate sono statiche: ogni parola ha un unico vettore, indipendentemente dal contesto in cui si trova. Ad esempio, "banca" avrà lo stesso embedding sia che si parli di un fiume, sia che ci si riferisca a un istituto finanziario. Questo limite lo rende inadatto a task che richiedono una comprensione più contestuale, per cui modelli come BERT risultano più efficaci.

BERT Pre-trained Embedding BERT (Bidirectional Encoder Representations from Transformers) rappresenta un passo avanti rispetto a Word2Vec, introducendo un nuovo livello di comprensione contestuale. Sviluppato da Google AI nel 2018, BERT sfrutta l'architettura Transformer per analizzare il contesto delle parole in modo bidirezionale. Ciò significa che il modello tiene conto di tutte le parole in una frase, sia quelle a sinistra che quelle a destra, per determinare il significato di un termine specifico.

L'embedding BERT si distingue per il suo approccio all'addestramento, che include due tecniche principali:

- Masked Language Modeling (MLM): Durante l'addestramento, alcune parole nella frase vengono mascherate, e il modello impara a prevederle basandosi sul contesto. Ad esempio, nella frase "Il gatto [MASK] sul divano", BERT impara che la parola mancante potrebbe essere "dorme".
- Next Sentence Prediction (NSP): BERT apprende anche le relazioni tra frasi consecutive, una capacità fondamentale per attività come la risposta a domande o la comprensione testuale.

BERT rappresenta un punto di svolta per l'elaborazione del linguaggio naturale grazie alla sua capacità di cogliere il

contesto globale delle parole. Questa caratteristica lo rende ideale per compiti complessi, come l'analisi del sentiment, la traduzione automatica e la generazione di testo. Inoltre, grazie al pre-training su enormi quantità di dati, BERT può essere facilmente adattato a compiti specifici attraverso il fine-tuning.

Nonostante le sue potenzialità, BERT richiede risorse computazionali significative per l'addestramento e l'inferenza. Questo lo rende meno accessibile per progetti con risorse hardware limitate. Inoltre, la sua complessità lo rende meno pratico per applicazioni semplici o in tempo reale, dove modelli più leggeri possono essere più appropriati.

3.3.7 - Data Quality

Come in qualsiasi progetto di machine learning, le scelte effettuate durante lo sviluppo possono introdurre una serie di problematiche, tra cui rumore, bias, underfitting e overfitting. Questi problemi possono compromettere l'efficacia del modello e la qualità dei risultati ottenuti. In questa descrizione dettagliata, esploreremo tutti i possibili fattori e le scelte che hanno potuto introdurre queste problematiche, giustificando le motivazioni dietro tali scelte.

Rumore nei Dati

Il rumore nei dati si riferisce alla presenza di informazioni irrilevanti, distorte o non pertinenti che possono influenzare negativamente l'addestramento del modello. Nel contesto del progetto, il rumore può derivare da diverse fonti, tra cui:

1. **Commenti in Lingue Diverse**
   - **Descrizione**: Il progetto prevede l'analisi di commenti in inglese, ma il dataset iniziale potrebbe contenere commenti in altre lingue. Sebbene questi commenti siano stati rimossi durante il preprocessing, la loro presenza iniziale potrebbe aver introdotto rumore nei dati.
   - **Motivazione**: La decisione di rimuovere i commenti in lingue diverse dall'inglese è stata presa per garantire coerenza nel linguaggio analizzato. Poiché il modello è stato progettato per lavorare con l'inglese, l'inclusione di commenti in altre lingue avrebbe potuto introdurre rumore e compromettere l'accuratezza del modello.
   - **Impatto**: La rimozione dei commenti in lingue diverse ha ridotto il rumore, ma potrebbe aver limitato la generalizzazione del modello in contesti multilingue.
2. **Caratteri Speciali e Punteggiatura**
   - **Descrizione**: La presenza di caratteri speciali, emoji o punteggiatura non standard può introdurre rumore. Nel progetto, la punteggiatura è stata rimossa durante il preprocessing.
   - **Motivazione**: La rimozione della punteggiatura è stata effettuata per semplificare il testo e ridurre il rumore. La punteggiatura, infatti, non aggiunge informazioni rilevanti per l'analisi del sentiment.
   - **Impatto**: Questa scelta ha ridotto il rumore, ma potrebbe aver rimosso alcune informazioni contestuali, come l'uso di punti esclamativi per enfatizzare un sentimento positivo o negativo.
3. **Commenti Troppo Brevi o Lunghi**
   - **Descrizione**: Commenti molto brevi (ad esempio, una sola parola) o molto lunghi possono contenere informazioni irrilevanti o ridondanti. Nel progetto, i commenti troppo brevi sono stati rimossi.
   - **Motivazione**: I commenti troppo brevi sono stati rimossi perché spesso non contengono informazioni sufficienti per un'analisi significativa del sentiment. D'altra parte, i commenti troppo lunghi potrebbero contenere informazioni ridondanti o irrilevanti.
   - **Impatto**: La rimozione dei commenti troppo brevi ha ridotto il rumore, ma potrebbe aver eliminato alcuni

commenti utili. I commenti troppo lunghi, se non adeguatamente filtrati, potrebbero ancora introdurre rumore.

4. **Generazione di Dati Sintetici**
   - **Descrizione**: Per bilanciare il dataset, il progetto ha utilizzato tecniche di oversampling, inclusa la generazione di commenti sintetici utilizzando modelli linguistici come Mistral e Llama.
   - **Motivazione**: La generazione di dati sintetici è stata necessaria per bilanciare il dataset, che era fortemente sbilanciato a favore della classe positiva. senza perdere troppa informazione. Questo approccio ha permesso di aumentare il numero di commenti negativi e neutri.
   - **Impatto**: Sebbene questa tecnica abbia aiutato a bilanciare il dataset, c'è il rischio che i dati sintetici introducano pattern artificiali che il modello potrebbe imparare a riconoscere, portando a overfitting.

Bias nei Dati e nel Modello

Il bias si riferisce a distorsioni sistematiche nei dati o nel modello che possono portare a previsioni inaccurate o ingiuste. Nel contesto del progetto, il bias può manifestarsi in diverse forme:

1. **Bias nel Dataset**
   - **Descrizione**: Il dataset utilizzato potrebbe non essere totalmente rappresentativo della popolazione target. Ad esempio, se il dataset è composto principalmente da commenti di studenti universitari, il modello potrebbe non generalizzare bene a contesti educativi diversi, come le scuole superiori o l'istruzione online per adulti.
   - **Motivazione**: Il dataset è stato estratto da piattaforme come Coursera e RateMyProfessor, che sono prevalentemente utilizzate da studenti universitari. Questa scelta è stata dettata dalla disponibilità dei dati.
   - **Impatto**: Il modello potrebbe essere biased verso il contesto universitario, limitando la sua applicabilità.
2. **Bias nelle Etichette**
   - **Descrizione**: Nel progetto, i commenti sono stati etichettati come positivi, negativi o neutri basandosi sui rating degli studenti. Tuttavia, questa associazione potrebbe non essere sempre accurata, specialmente per i commenti con rating intermedi (ad esempio, 3 stelle).
   - **Motivazione**: L'uso dei rating come proxy per il sentiment è stato scelto per semplificare il processo di etichettatura, dato che i dataset non erano pre-etichettati.
   - **Impatto**: Questa scelta potrebbe introdurre un bias nel modello, che potrebbe imparare a classificare erroneamente i commenti neutri.
3. **Bias nel Modello Pre-addestrato**
   - **Descrizione**: L'uso di modelli pre-addestrati, come Twitter-roBERTa, potrebbe introdurre un bias se il modello è stato addestrato su dati non rappresentativi del contesto educativo.
   - **Motivazione**: Il modello Twitter-roBERTa è stato scelto per la sua efficacia nella classificazione del sentiment in tre classi (positivo, negativo, neutro). Tuttavia, questo modello è stato addestrato su tweet, che potrebbero non riflettere le sfumature linguistiche dei feedback degli studenti.
   - **Impatto**: Il modello potrebbe essere biased verso il linguaggio informale e colloquiale dei social media, limitando la sua capacità di cogliere le sfumature specifiche dei contesti educativi.

Overfitting

L'overfitting si verifica quando un modello impara troppo bene i dati di addestramento, catturando anche il rumore e le peculiarità specifiche del dataset, ma fallisce nel generalizzare a nuovi dati. Nel contesto del progetto, l'overfitting potrebbe essere introdotto da diverse scelte:

1. **Complessità del Modello**
   - **Descrizione**: L'uso di modelli di deep learning, come le reti neurali, può aumentare il rischio di overfitting, specialmente se il modello è troppo complesso rispetto alla quantità di dati disponibili.
   - **Motivazione**: Nel progetto, sono stati utilizzati modelli come SVM e logistic regression, che sono meno soggetti a overfitting rispetto alle reti neurali. Tuttavia, la loro configurazione potrebbe aumentare il rischio di overfitting.
   - **Impatto**: Se il modello è troppo complesso, potrebbe imparare a riconoscere pattern specifici del dataset di addestramento, compromettendo la sua capacità di generalizzare.
2. **Generazione di Dati Sintetici**
   - **Descrizione**: La generazione di dati sintetici per bilanciare il dataset potrebbe introdurre pattern artificiali che il modello potrebbe imparare a riconoscere.
   - **Motivazione**: La generazione di dati sintetici è stata necessaria per bilanciare il dataset, ma potrebbe introdurre rumore e pattern non rappresentativi della realtà.
   - **Impatto**: Se i dati sintetici non sono sufficientemente vari e rappresentativi, il modello potrebbe overfittare su questi pattern artificiali.

Underfitting

L'underfitting si verifica quando un modello è troppo semplice e non riesce a catturare le relazioni complesse nei dati, portando a prestazioni scarse sia sui dati di addestramento che su quelli di test. Nel contesto del progetto, l'underfitting potrebbe essere introdotto da:

1. **Scelta di Modelli Troppo Semplici**
   - **Descrizione**: Sebbene i modelli come SVM e logistic regression siano meno soggetti a overfitting, potrebbero non essere sufficientemente complessi per catturare le sfumature del linguaggio naturale nei feedback degli studenti.
   - **Motivazione**: La scelta di modelli più semplici è stata dettata dalla necessità di evitare overfitting e di garantire un buon bilanciamento tra complessità e generalizzazione.
   - **Impatto**: Se il modello è troppo semplice, potrebbe non catturare relazioni complesse, portando a underfitting.
2. **Riduzione Eccessiva del Dataset**
   - **Descrizione**: Durante il bilanciamento del dataset, è stata effettuata una riduzione significativa della classe positiva attraverso l'undersampling. Se questa riduzione è troppo aggressiva, potrebbe portare a una perdita di informazioni importanti.
   - **Motivazione**: L'undersampling è stato necessario per bilanciare il dataset, ma una riduzione eccessiva potrebbe compromettere la qualità dei dati.
   - **Impatto**: Se il dataset è troppo ridotto, il modello potrebbe non avere sufficienti informazioni per apprendere relazioni complesse, portando a underfitting.

3.4 | Model creation

In questa sezione vengono descritti i modelli implementati per affrontare il problema della sentiment analysis. Ogni modello è stato selezionato e sviluppato per sfruttare specifiche caratteristiche dei dati e del task, al fine di garantire un'analisi accurata ed efficiente. Tra i modelli proposti, figurano algoritmi classici e consolidati, come il Multinomial Naive Bayes, il Support Vector Machine e la Logistic Regression, affiancati da approcci più avanzati basati su reti neurali. Ogni metodo viene descritto dettagliatamente, con un focus sulle motivazioni alla base della scelta, le definizioni formali e i passaggi fondamentali per il loro funzionamento.

3.4.1 - Multinomial Naive Bayes

Overview Il Multinomial Naive Bayes (MNB) è un algoritmo probabilistico che si è dimostrato particolarmente efficace nella classificazione di dati testuali, come nel caso della sentiment analysis. Si basa sul principio del Naive Bayes, utilizzando il teorema di Bayes per calcolare la probabilità che un testo appartenga a una determinata classe in base alle sue caratteristiche. La principale differenza tra il Naive Bayes generico e la sua variante multinomiale risiede nell'assunzione di una distribuzione multinomiale per i dati. Questa caratteristica rende l'MNB particolarmente adatto a lavorare con dati discreti, come le parole che compongono un testo.

Nel contesto dell'elaborazione del linguaggio naturale, l'MNB modella la frequenza delle parole nei testi per identificare modelli che caratterizzano le diverse classi. È particolarmente utile quando si lavora con dati ad alta dimensionalità e con distribuzioni sparse, entrambi tratti distintivi dei testi. Grazie alla sua semplicità ed efficienza computazionale, il modello è largamente utilizzato per applicazioni come la classificazione testuale e la sentiment analysis.

Formal Definition L'obiettivo principale del Multinomial Naive Bayes è quello di determinare a quale classe appartiene un determinato testo (ad esempio, positivo, negativo o neutro) sulla base delle parole che lo compongono. Il modello esegue questa classificazione analizzando la probabilità che il testo in esame sia associato a ciascuna classe, tenendo conto della frequenza delle parole sia all'interno del testo stesso sia nei dati di addestramento.

Per fare ciò, l'MNB calcola una serie di probabilità. In primo luogo, considera la distribuzione iniziale delle classi nel dataset, ovvero quanto frequentemente ogni classe compare nei dati di addestramento. Successivamente, stima quanto è probabile che ogni parola del testo appaia nei documenti di una specifica classe. Infine, queste informazioni vengono combinate per determinare la probabilità complessiva che il testo appartenga a ciascuna classe.

Un aspetto importante del Multinomial Naive Bayes è la capacità di gestire la presenza di parole che non sono mai apparse nei dati di addestramento, evitando che queste causino problemi nel calcolo delle probabilità. Questo è possibile grazie all'applicazione di tecniche di smoothing, come il Laplace Smoothing, che aggiungono una piccola correzione ai conteggi delle parole per evitare valori nulli.

Fundamental Steps Quando il Multinomial Naive Bayes viene utilizzato per classificare un nuovo testo, il processo segue una serie di passaggi chiari e ben definiti:

1. Calcolo delle probabilità iniziali delle classi: Il primo passo consiste nel determinare quanto frequentemente ogni classe (ad esempio, positivo, negativo o neutro) compare nei dati di addestramento. Questo passaggio stabilisce una sorta di punto di partenza per il calcolo delle probabilità.
2. Stima della probabilità delle parole nelle classi: Successivamente, il modello analizza quanto è probabile che ciascuna

parola presente nel testo appartenga a una determinata classe. Questo viene fatto analizzando la frequenza con cui ogni parola compare nei documenti di ciascuna classe durante la fase di addestramento.

3. Applicazione di tecniche di smoothing: Per evitare che parole nuove o poco comuni, non presenti nei dati di addestramento, compromettano l'efficacia del modello, si applicano correzioni ai conteggi delle parole. Questo garantisce che anche le parole rare abbiano una probabilità minima associata.
4. Determinazione della probabilità complessiva delle classi: Utilizzando le probabilità calcolate nei passaggi precedenti, il modello combina tutte le informazioni disponibili per stimare la probabilità complessiva che il testo appartenga a ciascuna classe.
5. Assegnazione della classe: Infine, il modello assegna al testo la classe che ha la probabilità complessiva più alta.

Motivazioni scelta Abbiamo scelto di implementare il Multinomial Naive Bayes per una serie di motivi che ne evidenziano l'efficacia e l'idoneità per la sentiment analysis:

- Adattabilità ai dati testuali: L'MNB è particolarmente adatto alla natura dei dati testuali, poiché tratta in modo efficace la frequenza delle parole. Questo lo rende ideale per applicazioni in cui le parole stesse costituiscono le caratteristiche principali.
- Efficienza computazionale: Grazie alla sua semplicità, l'MNB è un modello leggero che può essere addestrato e utilizzato per fare previsioni in modo molto rapido, anche con dataset di grandi dimensioni o in contesti che richiedono analisi in tempo reale.
- Gestione della sparsa distribuzione dei dati: I dati testuali spesso presentano una distribuzione sparsa, in cui molte parole appaiono raramente nei documenti. Il Multinomial Naive Bayes gestisce bene questa caratteristica, concentrandosi sulle parole più frequenti e significative.
- Riduzione del rischio di overfitting: Il modello, basandosi sull'assunzione di indipendenza tra le caratteristiche (le parole), evita di catturare relazioni complesse che potrebbero portare a un sovradattamento ai dati di addestramento. Questo lo rende più robusto quando applicato a dati nuovi.
- Compatibilità con tecniche di embedding: L'MNB si integra perfettamente con tecniche di rappresentazione testuale come il TF-IDF, che pesano le parole in base alla loro importanza relativa nel corpus. Questa combinazione permette al modello di distinguere meglio le parole più significative e di ignorare quelle irrilevanti, migliorando le prestazioni complessive nella classificazione.

3.4.2 - Support Vector Machine

Overview Il Support Vector Machine (SVM) è un potente algoritmo di apprendimento supervisionato ampiamente utilizzato per compiti di classificazione, inclusa la sentiment analysis. L'idea alla base dell'SVM è rappresentare i dati come punti in uno spazio multidimensionale, dove ogni dimensione corrisponde a una caratteristica del dato (ad esempio, una parola in un testo). L'obiettivo principale dell'SVM è trovare un iperpiano che separi in modo ottimale i punti appartenenti a classi diverse.

Nel caso di dati facilmente separabili, il modello cerca di massimizzare il margine, ovvero la distanza tra l'iperpiano e i punti più vicini di ciascuna classe, noti come vettori di supporto. Questo margine massimo garantisce una separazione più robusta e generalizzabile. Tuttavia, quando i dati non sono linearmente separabili (come accade spesso nei problemi reali), l'SVM utilizza tecniche avanzate, come le funzioni kernel, per trasformare i dati in uno spazio di dimensioni superiori. In questo spazio trasformato, diventa più semplice trovare una separazione lineare tra le classi. Questa capacità di adattarsi anche a dati complessi rende l'SVM estremamente versatile ed efficace.

What is a Kernel? Un elemento fondamentale del Support Vector Machine è il concetto di kernel. Il kernel è una funzione che consente di trasformare i dati in uno spazio a dimensioni più elevate, dove la separazione tra le classi può diventare più semplice. Questa trasformazione è particolarmente utile quando i dati non possono essere separati linearmente nello spazio originale.

Ad esempio, immagina un insieme di punti distribuiti in modo tale che sia impossibile tracciarvi una linea retta che separi le classi. Applicando una funzione kernel, i punti vengono "spostati" in uno spazio superiore, dove quella stessa separazione lineare diventa possibile. In altre parole, il kernel consente al modello di trovare soluzioni in spazi più complessi senza dover calcolare esplicitamente le trasformazioni. Questo approccio rende l'SVM adatto a una vasta gamma di problemi, anche molto complessi.

Quando invece i dati sono già separabili linearmente, si può utilizzare un kernel lineare che lascia i dati nel loro spazio originale, permettendo al modello di trovare una separazione semplice ed efficace.

Fundamental Steps Il processo seguito da un SVM per classificare i dati può essere suddiviso in tre passaggi principali:

1. Rappresentazione nello spazio delle caratteristiche: I dati vengono rappresentati come vettori in uno spazio multidimensionale, dove ogni dimensione corrisponde a una caratteristica (ad esempio, la presenza o la frequenza di una parola in un testo).
2. Individuazione dell'iperpiano ottimale: Durante la fase di addestramento, l'SVM cerca di individuare l'iperpiano che separa meglio i dati appartenenti a classi diverse. Il criterio chiave per questa separazione è la massimizzazione del margine, ossia la distanza tra l'iperpiano e i punti più vicini di ciascuna classe. Questo garantisce una classificazione robusta e generalizzabile.
3. Classificazione di nuovi dati: Una volta individuato l'iperpiano ottimale, l'SVM utilizza questa separazione per classificare i nuovi dati. La posizione di ciascun punto rispetto all'iperpiano determina la classe a cui appartiene.

Why choose it? Abbiamo deciso di implementare il Support Vector Machine per diversi motivi che ne evidenziano l'idoneità al nostro progetto:

- Separazione ottimale tra le classi: L'SVM è progettato per trovare la separazione migliore tra classi, rendendolo ideale per compiti come la sentiment analysis, dove spesso le classi (positivo, negativo, neutro) hanno confini complessi.
- Efficacia con dati ad alta dimensionalità: I dati testuali, come quelli utilizzati nella sentiment analysis, sono caratterizzati da un gran numero di caratteristiche (ad esempio, ogni parola rappresenta una dimensione). L'SVM eccelle in questi contesti, poiché può gestire efficacemente spazi a molte dimensioni.
- Robustezza al rumore: I dati testuali spesso contengono rumore, come parole non informative, errori di battitura o contenuti ambigui. L'SVM è noto per la sua capacità di resistere a questi problemi, concentrandosi solo sui punti più rilevanti, i vettori di supporto.
- Compatibilità con le tecniche di word embedding: Quando i testi vengono trasformati in rappresentazioni numeriche dense, come quelle generate da tecniche di embedding (ad esempio, Word2Vec), l'SVM è in grado di sfruttare queste rappresentazioni per migliorare la classificazione. Le proprietà semantiche catturate dagli embedding si combinano con la capacità dell'SVM di trovare separazioni ottimali, migliorando le performance complessive.

3.4.3 - Logistic Regression

Overview La Logistic Regression è un algoritmo di classificazione lineare ampiamente utilizzato per prevedere la probabilità che un'osservazione appartenga a una determinata classe. Nonostante il termine "regressione" possa far pensare a un modello di tipo predittivo-continuo, si tratta a tutti gli effetti di un modello di classificazione.

Il cuore della Logistic Regression è rappresentato dalla funzione logistica, conosciuta anche come funzione sigmoide. Questa funzione trasforma la combinazione lineare delle variabili di input in una probabilità compresa tra 0 e 1. In altre parole, il modello calcola la probabilità che un'osservazione appartenga a una specifica classe (ad esempio, sentiment positivo). Una volta calcolata questa probabilità, si utilizza una soglia predefinita (generalmente 0.5) per assegnare l'osservazione a una classe: se la probabilità supera la soglia, l'osservazione viene classificata come appartenente alla classe positiva, altrimenti a quella negativa.

La Logistic Regression è particolarmente efficace quando le classi sono separabili linearmente o quando esiste una relazione logaritmica tra le variabili di input e la variabile target.

Come funziona Il funzionamento della Logistic Regression può essere riassunto in pochi passaggi chiave. Per iniziare, il modello combina le variabili di input in un'espressione lineare (cioè somma pesata delle caratteristiche più un termine di bias). Questa combinazione lineare rappresenta il "punteggio grezzo" per ciascuna classe. Successivamente, il punteggio viene trasformato in una probabilità utilizzando la funzione logistica, che "comprime" il valore in un intervallo compreso tra 0 e 1.

Una volta calcolata la probabilità per una determinata classe, il modello prende una decisione: se questa probabilità è superiore a una soglia prestabilita (tipicamente 0.5), assegna l'osservazione alla classe positiva. Al contrario, se è inferiore, l'osservazione viene classificata come negativa.

La Logistic Regression utilizza un processo iterativo per ottimizzare i pesi associati alle variabili di input, in modo da ridurre al minimo gli errori nelle previsioni. Durante questa fase di addestramento, il modello apprende quali caratteristiche sono più rilevanti per distinguere tra le classi.

Fundamental Steps Il processo di classificazione della Logistic Regression può essere suddiviso in quattro fasi:

1. Costruzione del modello: Il primo passaggio consiste nel definire una combinazione lineare delle variabili di input. Questa rappresenta il punto di partenza per calcolare la probabilità che un'osservazione appartenga a una determinata classe.
2. Definizione della funzione di perdita: Il modello utilizza una funzione di perdita per valutare quanto le sue previsioni si discostano dai valori effettivi. Una delle funzioni più comuni è la Log Loss (o Binary Cross-Entropy), che penalizza fortemente le previsioni errate.
3. Ottimizzazione dei pesi: Durante l'addestramento, il modello utilizza tecniche di ottimizzazione, come il Gradient Descent, per modificare i pesi associati alle variabili di input. L'obiettivo è minimizzare la funzione di perdita e migliorare la capacità del modello di effettuare previsioni accurate.
4. Classificazione: Una volta addestrato, il modello utilizza la funzione logistica per calcolare la probabilità che un nuovo dato appartenga a una determinata classe. In base a questa probabilità e alla soglia scelta, il modello classifica l'osservazione.

Why choose it? La Logistic Regression rappresenta una scelta solida e ben motivata per diversi motivi, che ne evidenziano l'efficacia e la praticità, specialmente nel contesto della sentiment analysis:

- Semplicità e interpretabilità: È un modello lineare facile da comprendere e implementare. La Logistic Regression non solo classifica i dati, ma fornisce anche una probabilità associata a ogni previsione, permettendo di valutare il livello di confidenza del modello nelle sue decisioni. Inoltre, la semplicità della formula rende il modello particolarmente trasparente, facilitando l'interpretazione dei risultati.
- Efficienza nei dati linearmente separabili: Se le classi sono separabili linearmente, la Logistic Regression offre una soluzione semplice ed efficace. Questo la rende una scelta ideale quando i dati mostrano confini chiari tra sentiment positivo, negativo o neutro.
- Velocità di addestramento e inferenza: La Logistic Regression è un modello relativamente semplice dal punto di vista computazionale, il che garantisce tempi di addestramento e di inferenza rapidi. Questa caratteristica è particolarmente utile per applicazioni che richiedono analisi in tempo reale, come la sentiment analysis di commenti o recensioni.
- Interpretabilità dei pesi: I pesi associati alle variabili di input forniscono informazioni preziose sull'importanza relativa di ciascuna caratteristica. Ad esempio, in un task di sentiment analysis, i pesi possono aiutare a identificare quali parole influenzano maggiormente la classificazione di un testo come positivo o negativo.
- Compatibilità con tecniche di rappresentazione testuale: La Logistic Regression lavora bene con rappresentazioni come il TF-IDF, che assegna un peso alle parole in base alla loro rilevanza nel corpus. Questa combinazione permette di ottenere modelli più precisi e mirati per la sentiment analysis.

3.4.4 - Feedforward Neural Network

Overview Una rete neurale feedforward (FFNN) è uno dei tipi più semplici di architetture di rete neurale utilizzate nel deep learning. È composta da strati di neuroni connessi in modo che le informazioni fluiscano in una sola direzione, dallo strato di input allo strato di output, senza alcun ciclo di feedback. Ogni neurone in uno strato è connesso a ogni neurone nel prossimo strato, e la forza di queste connessioni è determinata dai pesi che vengono appresi durante l'addestramento.

A differenza delle reti neurali ricorrenti (RNN) o delle reti di memoria a lungo termine (LSTM), che sono progettate per gestire dati sequenziali, le reti feedforward sono generalmente utilizzate per compiti come la classificazione, la regressione e altri tipi di apprendimento supervisionato in cui i dati non sono sequenziali.

Nonostante la loro semplicità, le FFNN possono ottenere prestazioni impressionanti su una vasta gamma di compiti, specialmente se combinate con tecniche come la regolarizzazione, la normalizzazione e funzioni di attivazione avanzate.

Architettura L'architettura di una rete neurale feedforward è composta dai seguenti elementi:

- Strato di Input: Questo è il primo strato della rete, dove i dati di input vengono alimentati nella rete. Ogni nodo nello strato di input rappresenta una caratteristica dei dati di input.
- Strati Nascosti: Tra lo strato di input e quello di output, ci sono uno o più strati nascosti. Ogni strato nascosto è composto da neuroni che trasformano gli input ricevuti dallo strato precedente. I neuroni in questi strati applicano una trasformazione lineare (una somma ponderata degli input) seguita da una funzione di attivazione, che determina l'output del neurone.
- Strato di Output: L'ultimo strato produce l'output. In un compito di classificazione, questo solitamente implica l'uso di una funzione di attivazione softmax per produrre una distribuzione di probabilità tra le classi possibili.

- Funzioni di Attivazione: Queste funzioni vengono applicate alle somme ponderate degli input per ogni neurone in uno strato nascosto per introdurre non linearità, il che permette alla rete di modellare relazioni complesse nei dati.
- Pesi e Bias: La forza delle connessioni tra i neuroni è rappresentata dai pesi, che vengono appresi durante l'addestramento. Ogni neurone ha anche un termine di bias che aiuta a spostare la funzione di attivazione.

Il vantaggio principale delle reti neurali feedforward è che sono facili da comprendere e implementare. Tuttavia, possono avere difficoltà a catturare relazioni complesse nei dati rispetto a modelli più sofisticati come LSTM o reti neurali convoluzionali (CNN).

Why We Choose It Alcune caratteristiche chiave delle reti feedforward sono:

- Nessun Ciclo di Feedback: A differenza delle RNN, le FFNN elaborano ogni input in modo indipendente e non hanno la capacità di mantenere informazioni di stato tra i vari passaggi temporali. Questo le rende adatte per compiti in cui la sequenza dei dati non è così importante quanto i singoli punti dati.
- Addestramento End-to-End: L'intera rete viene addestrata simultaneamente tramite backpropagation, dove i pesi e i bias vengono aggiornati per minimizzare la funzione di perdita.
- Architettura Semplice: L'architettura feedforward è semplice e diretta, rendendola facile da utilizzare come modello di base prima di passare a modelli più complessi.

Feedforward Implemented Le reti neurali feedforward possono essere particolarmente efficaci per compiti di classificazione del testo, soprattutto quando il testo è rappresentato come vettori utilizzando tecniche di embedding.

Questi embeddings convertono i dati di testo in forma numerica che può essere elaborata dalla rete feedforward. Di seguito esploreremo come le FFNN vengono implementate utilizzando queste diverse tecniche di embedding, concentrandoci su tre implementazioni specifiche: TF-IDF, Word2Vec e BERT embeddings.

Rete Feedforward per Embedding TF-IDF

```
def _create_feedforward_tfidf(self, input_shape):
    self.model = Sequential([
        Input(shape=(input_shape,)),
        Dense(128, activation='relu'),
        Dropout(0.5),
        Dense(64, activation='relu'),
        Dropout(0.5),
        Dense(3, activation="softmax")
    ])
```

Lo strato di input prende un vettore derivato dalla trasformazione TF-IDF del testo, che rappresenta l'importanza delle parole all'interno di un documento rispetto a un corpus. Gli strati nascosti apprendono le relazioni non lineari tra le caratteristiche di input, mentre il dropout viene utilizzato per prevenire l'overfitting. Lo strato finale utilizza una funzione di attivazione softmax per produrre una distribuzione di probabilità tra le tre classi.

Rete Feedforward per Embedding Word2Vec

```
def _create_feedforward_word2vec(self, input_shape):
    self.model = Sequential([
        Input(shape=(input_shape,)),

        # Primo blocco
        Dense(256, activation=None),
        BatchNormalization(),
        ReLU(),
        Dropout(0.3),

        # Secondo blocco
        Dense(128, activation=None),
        BatchNormalization(),
        ReLU(),
        Dropout(0.3),

        # Terzo blocco
        Dense(64, activation=None),
        BatchNormalization(),
        ReLU(),
        Dropout(0.4),

        # Output
        Dense(3, activation="softmax")  # 3 classi
    ])
```

In questo caso, gli embedding Word2Vec vengono utilizzati per rappresentare le parole in uno spazio continuo e denso. La rete contiene più strati per catturare la complessità dei dati, con l'uso di funzioni di attivazione ReLU e la normalizzazione del batch per migliorare la stabilità dell'addestramento. Il dropout viene applicato per prevenire l'overfitting.

Rete Feedforward per Embedding BERT

```
def _create_feedforward_bert(self, input_shape):
    self.model = Sequential([
        Input(shape=(input_shape,)),

        # Primo blocco
        Dense(512),
        LeakyReLU(negative_slope=0.1),
        BatchNormalization(),
        Dropout(0.3),

        # Secondo blocco
        Dense(256),
        LeakyReLU(negative_slope=0.1),
        BatchNormalization(),
        Dropout(0.3),

        # Terzo blocco
```

```
        Dense(128),
        LeakyReLU(negative_slope=0.1),
        BatchNormalization(),
        Dropout(0.4),

        # Quarto blocco
        Dense(64),
        LeakyReLU(negative_slope=0.1),
        Dropout(0.4),

        # Output
        Dense(3, activation='softmax')
    ])
```

BERT è utilizzato per generare embedding contestuali che catturano il significato delle parole in base al loro contesto. Gli strati di LeakyReLU e Batch Normalization vengono utilizzati per migliorare l'apprendimento, mentre il dropout aiuta a prevenire l'overfitting. Anche qui lo strato di output utilizza la funzione di attivazione softmax.

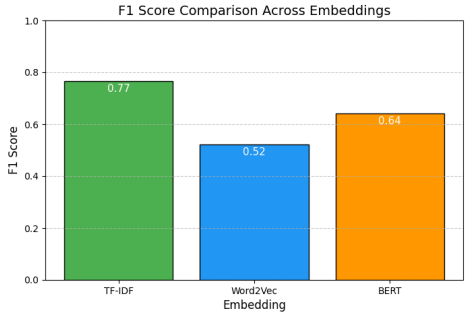
3.5 | Model evaluation

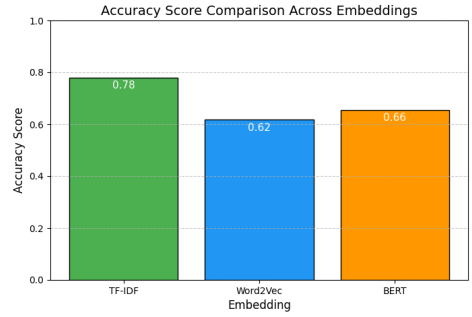
In questo paragrafo analizzeremo le prestazioni dei diversi modelli implementati, concentrandoci su tre metriche principali: Matrice di Confusione, Learning Curve e ROC-Curve.

- Matrice di Confusione: è uno strumento fondamentale per valutare le prestazioni del modello, fornendo una rappresentazione chiara del numero di predizioni corrette e sbagliate per ogni classe. Consente di distinguere i True Positive, False Positive, True Negative e False Negative, fornendo una visione dettagliata delle aree in cui il modello eccelle o necessita miglioramenti.
- Learning Curve : consente di osservare l'impatto di bias e varianza sul modello. Essa rappresenta l'andamento delle prestazioni in fase di training e testing al variare della dimensione del set di dati utilizzato per l'addestramento. Per ogni intervallo di crescita del dataset, viene calcolato uno score basato su una metrica specifica; nel nostro caso, utilizzeremo l'accuracy, che misura la percentuale di previsioni corrette rispetto al totale. Questa analisi è utile per identificare se il modello è sotto-addestrato (high bias) o sovra-addestrato (high variance).
- ROC-Curve : traccia la relazione tra il True Positive Rate (sensibilità) e il False Positive Rate, permettendo di individuare il compromesso ottimale tra i due. La curva consente di valutare le prestazioni del modello indipendentemente dalla soglia di classificazione scelta, calcolando l' AUC (Area Under the Curve) , un indicatore della capacità del modello di distinguere tra le classi. Una AUC più vicina a 1 indica un modello più efficace, mentre valori vicini a 0.5 suggeriscono prestazioni vicine a quelle di un classificatore casuale.

Per ogni modello analizzeremo innanzitutto le performance generali basandoci sulle metriche fondamentali F1-Score e Accuracy. Successivamente, approfondiremo il comportamento di ciascun modello in relazione alle diverse tipologie di codifiche adottate, evidenziando i punti di forza e le eventuali criticità.

3.5.1 - Prestazioni del Multinomial Naive Bayes Discutiamo adesso delle prestazioni rigistrate:

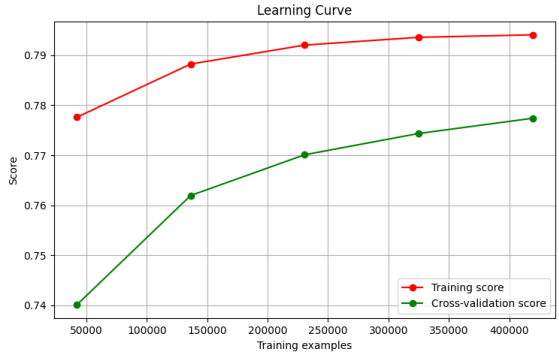




È evidente che il Naive Bayes ottiene i migliori risultati in entrambe le metriche utilizzando la codifica TF-IDF, mentre ottiene i risultati peggiori con Word2Vec. Nei paragrafi successivi, esamineremo più nel dettaglio il comportamento del modello con ciascuna delle codifiche.

TF-IDF - Learning Curve

Per valutare le performance generali del modello e identificare eventuali situazioni di overfitting o underfitting, è stata analizzata la learning curve, che mostra l'andamento dell'accuracy sia in fase di training che di testing al variare della dimensione del dataset di addestramento.



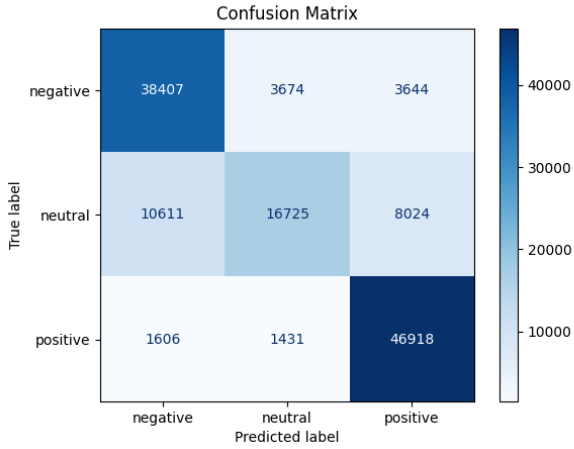
Esaminando il grafico delle learning curve, si osserva che l'andamento delle prestazioni del modello in fase di training e testing è del tutto normale.

- Il training score migliora gradualmente all'aumentare degli esempi di training, indicando che il modello apprende efficacemente i pattern dei dati.
- Anche il cross-validation score mostra un miglioramento costante, segnalando una buona capacità di generalizzazione su dati non visti.
- Il distacco tra le due curve è normale e atteso: il training score è tipicamente più alto poiché il modello è addestrato direttamente su quei dati, mentre il cross-validation score riflette la performance su dati non visti. Questo divario

non è eccessivo, suggerendo che il modello non è in overfitting. Con l'avanzare del training, le due curve tendono a convergere, indicando che il modello sta raggiungendo un equilibrio tra apprendimento e generalizzazione.

TF-IDF - Matrice di confusione

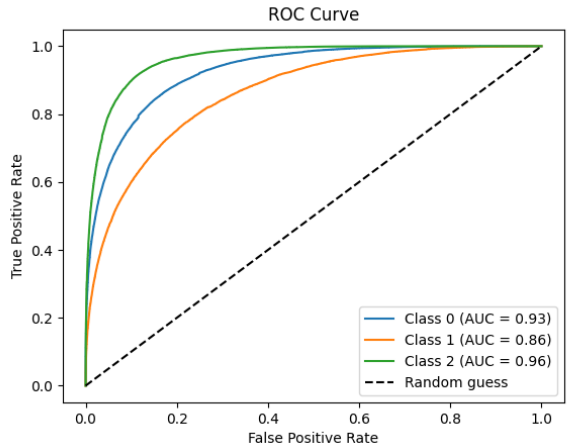
Il modello dimostra una buona capacità di classificare correttamente la maggior parte delle istanze per ciascuna classe, come evidenziato dai valori più elevati presenti sulla diagonale principale della matrice. Questo indica una concordanza significativa tra le etichette reali e quelle predette.



- Classe Positiva: Il modello ottiene risultati particolarmente buoni nella predizione del sentiment positivo, con un numero limitato di errori di classificazione. Nello specifico, si osservano 1606 falsi negativi (istanze positive classificate come negative) e 1431 falsi neutrali (istanze positive classificate come neutre).
- Classe Neutra: Come spesso accade nei compiti di analisi del sentiment, la classe neutra presenta una maggiore difficoltà di classificazione. Difatti si nota come il modello classifichi in maniera errata circa 30% delle etichette neutrali, predicendole come etichette negative.
- Classe Negativa: Anche per la classe negativa, il modello mostra una buona performance complessiva, nonostante la presenza di circa 7.000 dati erroneamente classificati come neutri o positivi. Va sottolineato che il modello ottiene buoni risultati nella classificazione del sentiment negativo anche perché tende spesso a classificare le etichette neutre come negative. Questo suggerisce che esista una soglia piccola tra i sentimenti neutro e negativo, che può influire sulla precisione della classificazione.

TF-IDF - ROC Curve

L'analisi della curva ROC (Receiver Operating Characteristic) e dei valori di AUC (Area Under the Curve) fornisce una valutazione approfondita delle prestazioni del modello di classificazione, confermando e ampliando quanto osservato nella matrice di confusione. Di seguito, viene presentata un'analisi dettagliata per ciascuna delle tre classi:



Dal grafico della curva ROC, si evince che il modello performi in maniera eccellente nella classificazione delle classi positive e negative. La curva ROC per queste due classi cresce rapidamente, mantenendo un False Positive Rate (FPR) basso. Questo indica che il modello è in grado di ottenere un numero elevato di Veri Positivi (True Positives) senza incrementare significativamente il numero di Falsi Positivi (False Positives), dimostrando una capacità discriminativa molto elevata per queste categorie.

Per quanto riguarda la Classe Neutra (Classe 1), le prestazioni del modello sono meno robuste rispetto alle altre due classi. Come già osservato nella matrice di confusione, questa categoria presenta maggiori difficoltà nella classificazione.

Dal grafico della curva ROC, ciò è evidente dalla crescita meno rapida della curva, che richiede un numero più elevato di Falsi Neutrali (False Positives) per raggiungere un buon numero di Veri Neutrali (True Positives). Le difficoltà nella classificazione della Classe Neutra potrebbero essere attribuite a una maggiore sovrapposizione con le altre classi in termini di feature o a una minore distinzione nei dati.

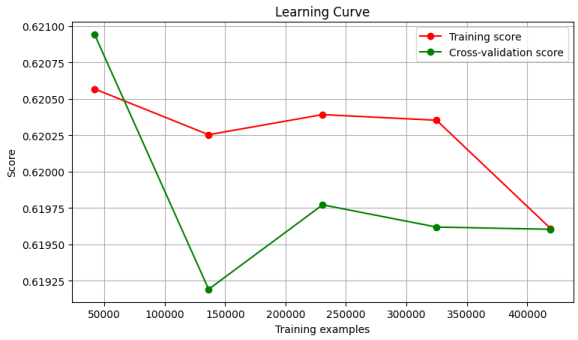
TF-IDF - Conclusioni

Il Multinomial Naive Bayes si adatta particolarmente bene a rappresentazioni sparse e discrete come quelle prodotte dal metodo TF-IDF. La natura lineare e interpretabile di TF-IDF permette al modello di sfruttare le assunzioni di indipendenza condizionale tra le caratteristiche, risultando in una classificazione più precisa.

In particolare, TF-IDF consente al MNB di rappresentare efficacemente l'importanza relativa dei termini in un documento rispetto al corpus e di seguire l'assunzione multinomiale, in cui ogni termine contribuisce in modo indipendente alla classificazione. Questo rende TF-IDF particolarmente adatto per il MNB.

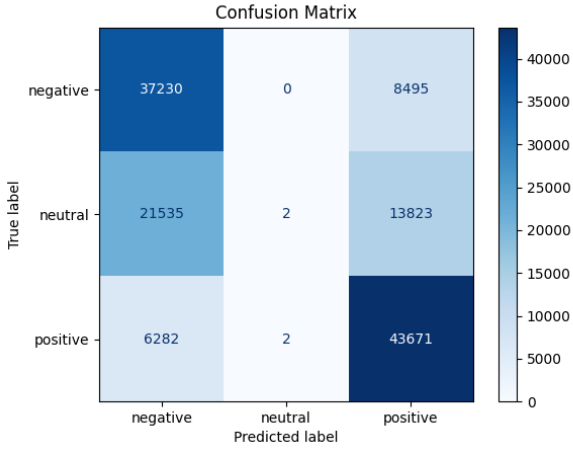
Word2Vec - Learning Curve

Dal grafico è facilmente intuibile che il modello presenta seri problemi sia in fase di training che di testing.



- Il training score mostra che il modello non riesce ad apprendere efficacemente dai dati di training. Man mano che il numero di esempi di training aumenta, le prestazioni del modello peggiorano, anziché migliorare. Questo comportamento è anomalo e indica che il modello non sta catturando i pattern sottostanti dei dati.
- Anche il testing score conferma che il modello non è in grado di generalizzare su dati non visti. All'aumentare dei dati di training, il validation score diminuisce ulteriormente, segnalando che il modello non solo non apprende dai dati, ma diventa anche meno efficace nel fare previsioni su nuovi dati.
- Il divario tra il training score e il validation score è molto piccolo. Questo potrebbe inizialmente sembrare un segnale positivo, ma in realtà è un chiaro sintomo di underfitting. Quando entrambe le curve sono basse e vicine tra loro, significa che il modello non è in grado di apprendere né dai dati di training né di generalizzare su dati non visti.
- L' underfitting osservato è con tutta probabilità causato dall'utilizzo di un embedding troppo complesso (Word2Vec) rispetto alla natura del modello.

Word2Vec - Matrice di confusione

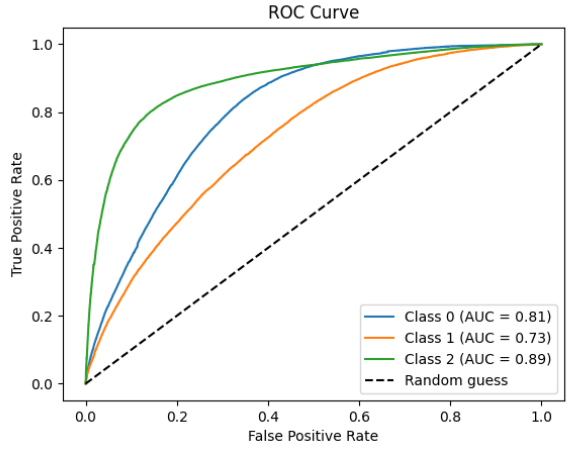


Con la Matrice di Confusione si evidenziano ulteriori problematiche di questo Embedding con il modello MNB. Soprattutto per quanto riguarda la classe "Neutral". In particolare:

- Classe Positiva: Il modello ottiene risultati discreti nella predizione del sentiment positivo. Il numero di falsi negativi (istanze positive classificate come negative o neutre) è limitato, indicando che il modello è in grado di identificare correttamente la maggior parte dei casi positivi. Non si può dire lo stesso per quanto riguarda i falsi positivi (istanze predette positive erroneamente), difatti il dato preoccupante sono gli 8.495 commenti negativi ritenuti positivi.
- Classe Neutra : La classe neutra presenta maggiori difficoltà di classificazione. Su circa 40.000 dati, il modello riesce a classificare correttamente la classe neutra solo 2 volte , un risultato estremamente basso. Circa il 60% dei dati con sentimento neutro vengono erroneamente classificati come negativi, mentre il 39.9% viene classificato come positivi. Solo lo 0.005% dei dati neutri viene classificato correttamente. Questo comportamento suggerisce che il modello non è in grado di distinguere efficacemente la classe neutra dalle altre due classi, probabilmente a causa dell'incompatibilità della codifica utilizzata con il modello probabilistico.
- Classe Negativa: Il modello ottiene risultati simili a quelli della classe positiva nella predizione del sentiment negativo. Il numero di falsi negativi (istanze negative classificate come positive o neutre) è contenuto, indicando una buona capacità del modello di identificare correttamente i casi negativi. Mentre come per la classe positiva, spiccano i falsi positivi in quanto circa il 33% dei dati predetti negativi appartengono in realtà alla classe neutra.

Word2Vec - ROC Curve

La ROC Curve è conferma i problemi esaminati nelle due analisi precedenti, in quanto le performance generali del modello anche in questa metrica non sono eccellenti.



Dal grafico della curva ROC, si evince che il modello dimostra prestazioni relativamente solide nella classificazione delle classi positive (2) e negative (1). La curva ROC per queste due classi cresce rapidamente, mantenendo un False Positive Rate (FPR) basso. Questo indica che il modello è in grado di ottenere un numero elevato di Veri Positivi (True Positives) senza incrementare significativamente il numero di Falsi Positivi (False Positives), dimostrando una discreta capacità discriminativa per queste categorie.

Per quanto riguarda la classe neutra (0), la curva ROC è molto vicina alla linea di random guess, che rappresenta un modello che classifica in modo casuale. Questo comportamento conferma le scarse prestazioni del modello nella classificazione della classe neutra, come già osservato nella matrice di confusione. La vicinanza alla random guess indica che il modello non è in grado di distinguere efficacemente la classe neutra dalle altre due classi, risultando in una classificazione quasi casuale.

I risultati della curva ROC confermano quanto osservato nella matrice di confusione: il modello performa discretamente nella classificazione delle classi positive e negative, ma ha serie difficoltà con la classe neutra. La curva ROC della classe neutra, essendo vicina alla random guess, riflette l'incapacità del modello di apprendere pattern significativi per questa categoria, portando a una classificazione inefficace.

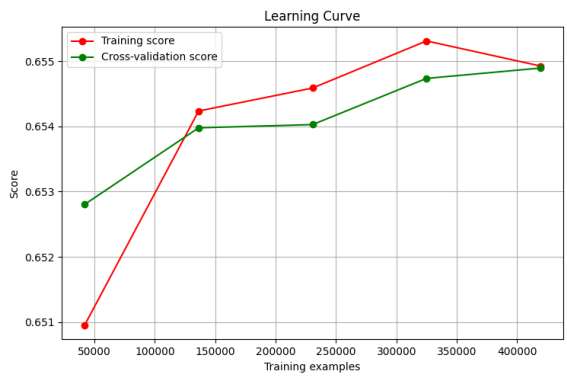
Word2Vec - Conclusioni

Possiamo quindi concludere che l'embedding Word2Vec non è efficiente con il modello MNB per il nostro task di classificazione per le seguenti principali motivazioni:

- Word2Vec genera vettori densi e continui che catturano relazioni semantiche tra le parole. Tuttavia, il Multinomial Naive Bayes (MNB) è progettato per lavorare con vettori discreti basati sulla frequenza delle parole (ad esempio, conteggi di termini o TF-IDF).
- Per utilizzare Word2Vec con MNB, è necessario scalare i vettori continui in una forma discreta, il che comporta una perdita di informazioni. Questa trasformazione rende difficile per il MNB sfruttare appieno le relazioni semantiche catturate da Word2Vec.
- La classe neutra, essendo spesso ambigua e meno distinta rispetto alle classi positiva e negativa, richiede una rappresentazione più ricca e sfumata, che Word2Vec potrebbe non fornire in modo efficace dopo la scalatura.

BERT - Learning Curve

Si può notare come il modello, nonostante diverse problematiche, riesca ad imparare dai dati in quando man mano che aumentano, aumenta anche lo score.



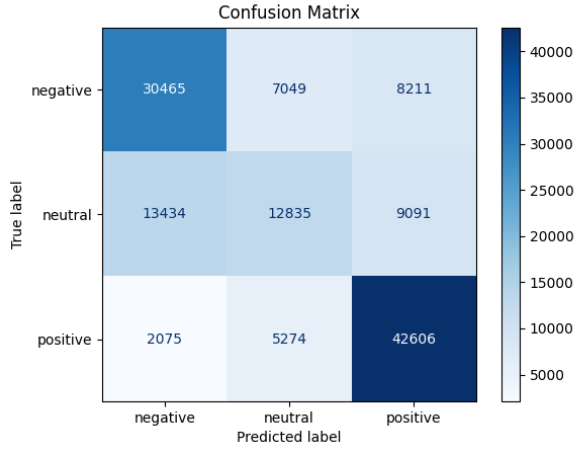
- Il Training Score inizialmente presenta valori relativamente bassi quando il numero di esempi di addestramento è limitato. Con l'aumento del numero di esempi, il punteggio di addestramento migliora gradualmente, mostrando

una crescita significativa tra i 50.000 e i 150.000 campioni. Tuttavia, nonostante tale crescita, il punteggio raggiunto si stabilizza su valori relativamente bassi, il che potrebbe indicare un possibile underfitting del modello.

- Anche il Testing Score inizia con valori bassi, ma migliora gradualmente con l'aumentare del numero di esempi di addestramento. Dopo una fase iniziale di adattamento, il testing score tende a convergere verso il training score, indicando che il modello sta generalizzando in modo simile a quanto avvenuto durante la fase di addestramento, ma su dati non visti. Tuttavia, è importante contestualizzare questa osservazione rispetto al punteggio complessivo, che risulta essere relativamente basso
- Per quanto riguarda il divario tra le due curve, man mano che il numero di esempi di training aumenta, il training score supera il testing score, ma il distacco rimane ridotto. Questo indica che il modello sta generalizzando e non è in overfitting. Tuttavia, il problema risiede nel punteggio finale al quale entrambe le curve convergono. Infatti, tale punteggio è basso, suggerendo che il modello è molto probabilmente in underfitting

BERT - Matrice di Confusione

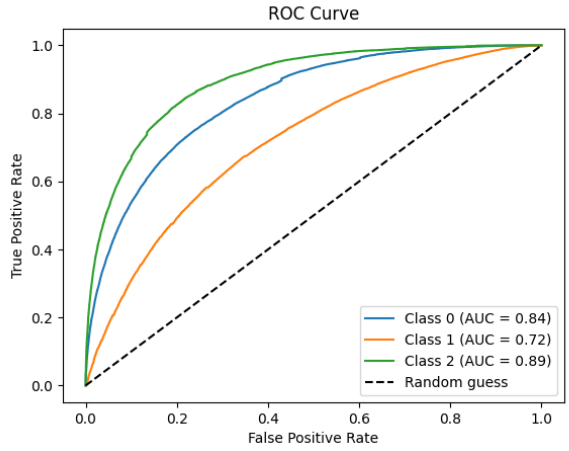
La matrice di confusione evidenzia una performance decente del modello con predizioni di classi positive e negative mentre una prestazione pessima con predizioni di quella neutrale.



- Classe Positiva: Il modello ottiene buone performance per quanto riguarda il numero di predizioni positive correttamente classificate, infatti circa l'85% dei dati con etichetta positiva sono stati classificati correttamente. Tuttavia, il problema risiede nei falsi positivi, in particolare nei 8211 dati negativi classificati come positivi. Questo rappresenta un dato critico per il modello, che potrebbe essere considerato poco affidabile, in quanto il 13% delle etichette predette come positive sono in realtà negative.
- Classe Neutra: Le predizioni per la classe neutra risultano essere le peggiori. La maggior parte dei dati con etichetta neutra viene erroneamente classificata come negativa, mentre solo una piccola parte viene correttamente classificata. Ciò che evidenzia la scarsa affidabilità del modello è che il 48% delle etichette predette come neutrali sono errate. Inoltre, solo il 36% delle etichette effettivamente neutrali vengono correttamente classificate.
- Classe Negativa: Le predizioni per la classe negativa seguono una distribuzione simile a quella della classe positiva, ma con un numero maggiore di falsi negativi per le classi neutrali.

BERT - ROC Curve

Dal grafico della curva ROC, si evince che il modello performa in modo efficace nella classificazione delle classi positive e negative. La curva ROC per queste due classi cresce rapidamente, mantenendo un False Positive Rate (FPR) basso.



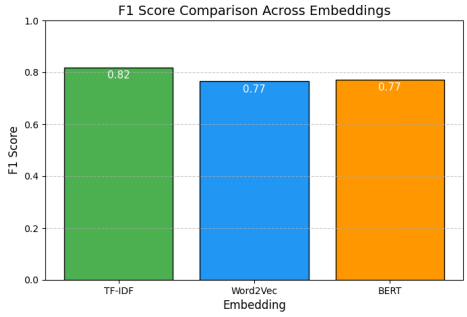
Per quanto riguarda la classe neutra (1), la sua curva ROC è quella più vicina alla random guess tra le tre. Questo comportamento conferma le difficoltà del modello nel classificare correttamente la classe neutra, come già evidenziato nella matrice di confusione.

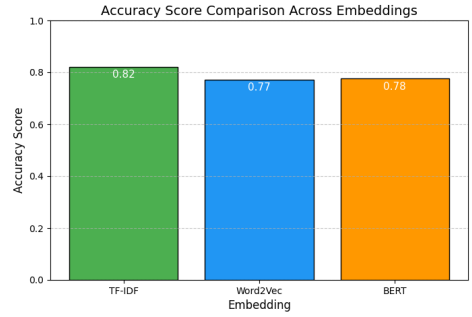
I risultati della curva ROC confermano quanto osservato nella matrice di confusione: il modello performa discretamente con le classi positiva e negativa, ma incontra notevoli difficoltà con la classe neutra.

BERT - Conclusioni

Nonostante il modello ottenga buoni risultati con le predizioni riguardanti le classi positive e negative, le predizioni sulla classe neutra presentano troppi errori per essere considerate affidabili in un contesto reale. Ciò è dovuto a motivazioni simili a quelle analizzate con Word2Vec, in quanto anche BERT produce vettori densi e continui che si basano su relazioni semantiche tra parole.
La natura ambigua della classe neutra, unita alla complessità dei vettori generati da BERT, rende difficile per il modello distinguere efficacemente questa categoria dalle altre.

3.5.2 - Prestazioni della Support Vector Machine Discutiamo adesso delle prestazioni registrate:

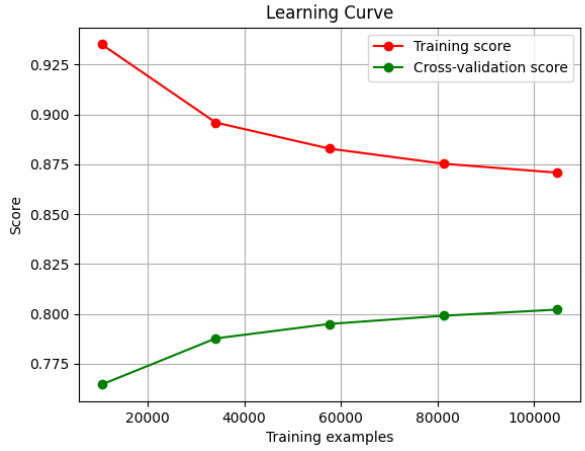




Support Vector Machine ottiene i suoi migliori risultati con la codifica TF-IDF, nei prossimi paragrafi, come fatto per Multinomial Naive Bayes, andremo ad analizzare il comportamento del modello con ognuno degli embedding per cercare di comprendere le motivazioni dietro questi grafici.

TF-IDF - Learning Curve

Esaminando la Curve Learning è possibile notare come l'apprendimento in fase di training e di testing sia differente.



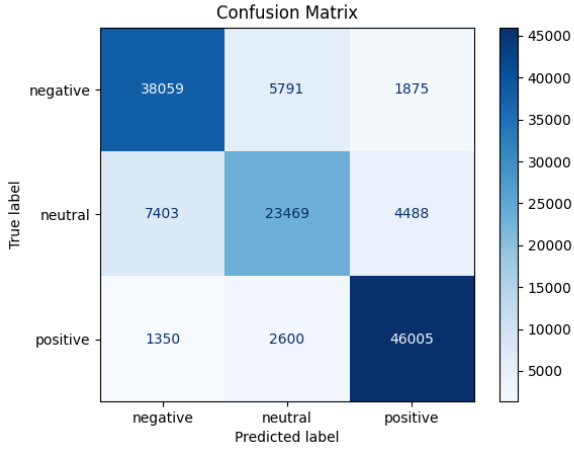
- Inizialmente, con pochi dati a disposizione, il Training Score è molto elevato perché l'SVM tende a overfittare, adattandosi eccessivamente ai dati di training e trovando un iperpiano complesso che "memorizza" i dati piuttosto che generalizzare.
- Con l'aumento dei dati di training, il punteggio diminuisce gradualmente, indicando che il modello sta imparando a generalizzare meglio, passando da un iperpiano strettamente adattato ai dati ad un margine più ampio e robusto, in grado di identificare pattern generali.
- Testing Score mostra una curva strettamente crescente, un segnale positivo che indica come il modello stia miglio-

rando la sua capacità di predizione e generalizzazione man mano che aumentano i dati di training. Non si notano difetti particolari in questa curva, il che suggerisce che l'SVM sta apprendendo in modo efficace.

- Inizialmente, il divario tra le due curve è molto accentuato, un chiaro sintomo di overfitting: con pochi dati, il modello performa molto bene sul training set ma non generalizza bene su dati non visti. Man mano che i dati aumentano, il divario si riduce significativamente, indicando che il modello sta raggiungendo un buon equilibrio tra fitting e generalizzazione, migliorando le sue prestazioni sia in fase di training che di testing

TF-IDF - Matrice di Confusione

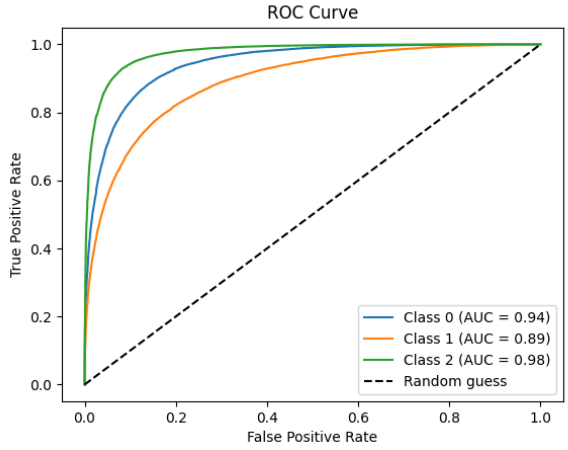
Il modello è in grado di classificare correttamente la maggior parte delle etichette, come evidenziato dai valori predominanti lungo la diagonale principale della matrice di confusione. Questo indica che le previsioni del modello coincidono spesso con le etichette reali, dimostrando una buona capacità di classificazione.



- Classe Positiva: il modello ottiene risultati ottimi per quanto riguarda questa classe, in quanto su 49.955 dati con etichetta positiva il modello ne ha predetti 46.005 in maniera corretta ovvero circa il 92%. Per quanto riguarda i falsi negativi sono solo 3950, mentre i falsi positivi ammontano a 6363 di cui però la maggior parte neutral.
- Classe Negativa: Anche per la classe negativa, il modello mostra prestazioni solide. Su 45.725 dati con etichetta negativa, il modello ne ha predetti correttamente 38.059, corrispondenti a circa l'83%. I falsi positivi (dati negativi classificati erroneamente come positivi o neutrali) sono 7.403, mentre i falsi negativi (dati non negativi classificati come negativi) ammontano a 8753 di cui maggior parte appartenenti a classe neutra.
- Classe Neutrale: Per la classe neutrale, il modello ha predetto correttamente 23.469 dati su 30.419. I falsi positivi (dati non neutrali classificati come neutrali) sono 8391, mentre i falsi negativi (dati neutrali classificati erroneamente) sono 11.891. Ciò evidenzia come la classificazione del sentimento neutrale sia più complessa rispetto alle altre due

TF-IDF - ROC Curve

Dal grafico seguente, è possibile confermare le valutazioni dedotte dalle analisi precedenti, poiché offre una visione complessiva della performance del modello nel predire le diverse classi.



Nel dettaglio, il modello dimostra una notevole capacità di discriminare la classe 0 (Negativa), con un'area sotto la curva (AUC) pari a 0.94, indicando un elevato tasso di veri positivi (TPR) associato a un contenuto tasso di falsi positivi (FPR). Anche per la classe 1 (Neutro), il modello raggiunge un livello di accuratezza soddisfacente, con un AUC di 0.89, sebbene leggermente inferiore rispetto alla classe 0. La classe 2 (Positiva) ottiene un AUC di 0.98, un valore prossimo alla perfezione, con una curva ROC che si avvicina notevolmente alla forma ideale, passando per i punti (0,0) e (1,1). Ciò evidenzia una capacità eccezionale del modello nel classificare correttamente questa categoria.

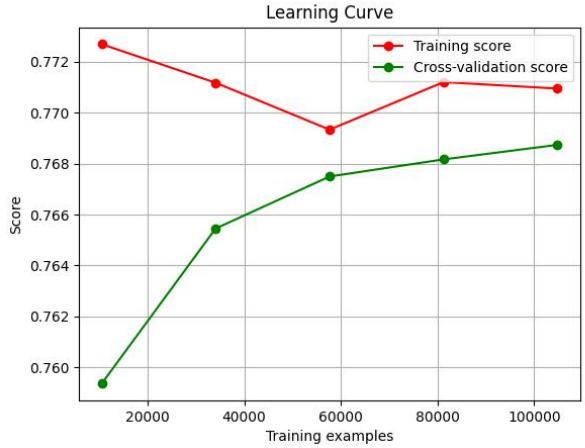
TF-IDF - Conclusioni

Il modello ha dimostrato risultati eccellenti, raggiungendo un ottimo equilibrio tra bias e varianza. La combinazione di TF-IDF con le Support Vector Machine (SVM) si è rivelata particolarmente efficace grazie alla natura sparsa e ad alta dimensionalità delle rappresentazioni TF-IDF.

SVM identifica facilmente un iperpiano di separazione ottimale in spazi lineari, sfruttando al meglio la struttura sparsa di TF-IDF. Questa sinergia ha contribuito molto alle prestazioni del modello, garantendo accuratezza e robustezza.

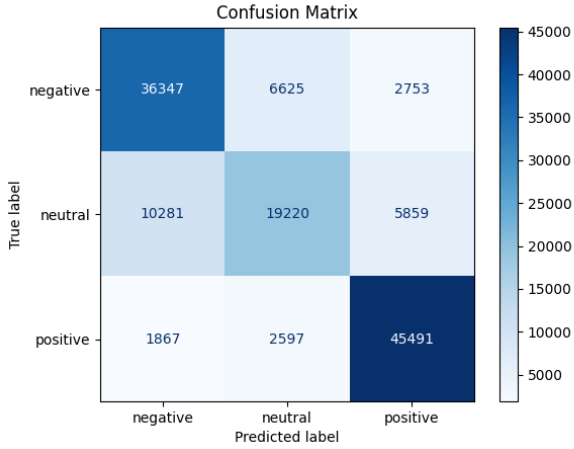
Word2Vec - Learning Curve

Esaminando tale learning curve possiamo ottenere diverse informazioni:



- Training Score: Dopo una fase iniziale in cui il modello manifesta una tendenza alla memorizzazione dei dati, le performance non mostrano miglioramenti significativi all'aumentare degli esempi di training. Tale comportamento evidenzia una limitata capacità di apprendimento e una difficoltà nel generalizzare durante l'addestramento.
- Testing Score : Sebbene si registri un incremento marginale delle prestazioni con l'aggiunta di dati, il punteggio finale rimane contenuto (0.76). Questo suggerisce che il modello non riesce a sfruttare appieno le informazioni.
- Divario tra Training e Testing Score : La ridotta distanza tra le due curve, accompagnata dalla convergenza verso valori bassi, conferma la presenza di underfitting . Il modello, probabilmente troppo semplice o basato su feature non ottimali, non è in grado di cogliere le relazioni sottostanti nei dati.

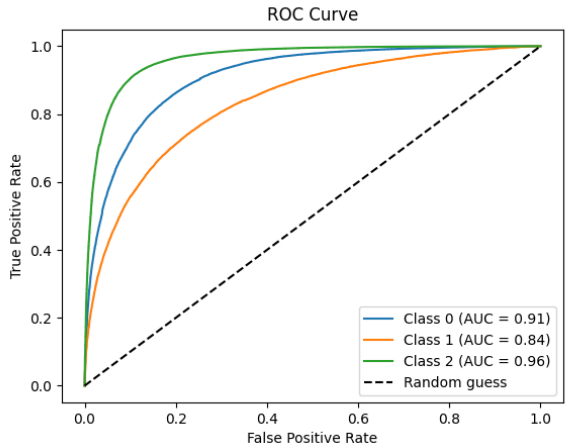
Word2Vec - Matrice di Confusione



- La classe positiva dimostra le prestazioni più robuste, con 45.491 predizioni corrette. Il modello presenta un'elevata precisione (88.5%), sebbene commetta errori significativi nel classificare 2.597 esempi neutrali e 1.867 esempi negativi come positivi.
- La classe neutra presenta le performance più critiche, con soli 19.220 esempi correttamente classificati. La precisione, pari circa al 53,8%, è relativamente bassa, e il modello commette numerosi errori: 10.281 esempi sono stati erroneamente classificati come negativi e 5.859 come positivi. Questi errori suggeriscono che il modello fatica a distinguere adeguatamente i dati neutri, probabilmente a causa di una difficoltà nel comprendere e sfruttare i pattern distintivi nei dati embeddati.
- Per la classe negativa , il modello raggiunge 36.347 predizioni corrette, con una precisione accettabile (78.5%). Tuttavia, si può notare confusione con la classe neutra in quanto molti neutrali sono stati identificati come negativi.

Word2Vec - ROC Curve

Nonostante i difetti evidenziati nella Learning Curve, il grafico conferma una giusta solidità del modello nelle performance di classificazione di classi positive (2) e negative (0) rispetto a quella neutra la quale ha una curva comunque buona ma non a livello delle altre due. Ciò era già evidente nella matrice di confusione, dove il modello otteneva le peggiori performance nella classificazione della classe neutra, spesso confusa con quella negativa.



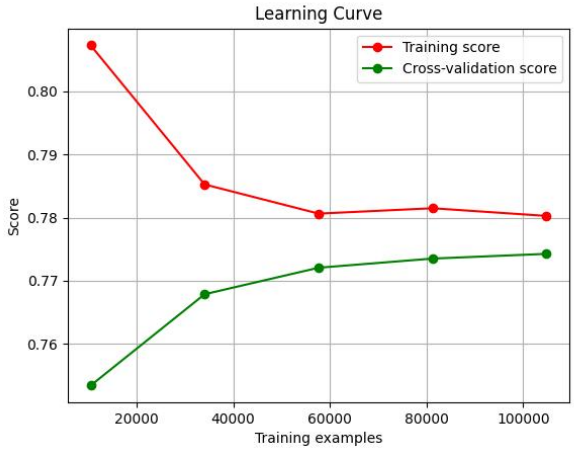
Word2Vec - Conclusioni

L'implementazione di SVM con rappresentazioni vettoriali Word2Vec dimostra un'adeguata capacità di gestione delle relazioni semantiche tra i termini. Tuttavia, le performance complessive rimangono inferiori rispetto all'approccio basato su codifica TF-IDF, con un divario particolarmente marcato nella classificazione della classe neutra.

I vettori Word2Vec, per loro natura, presentano variazioni di scala significative tra le dimensioni. Poiché SVM ottimizza i margini di decisione basandosi su distanze geometriche, è stato necessario applicare un'operazione di standardizzazione. Questo processo, seppur tecnicamente corretto, comprime la distribuzione originale degli embedding, attenuando differenze sottili ma semanticamente rilevanti per la classe neutra.

BERT - Learning Curve

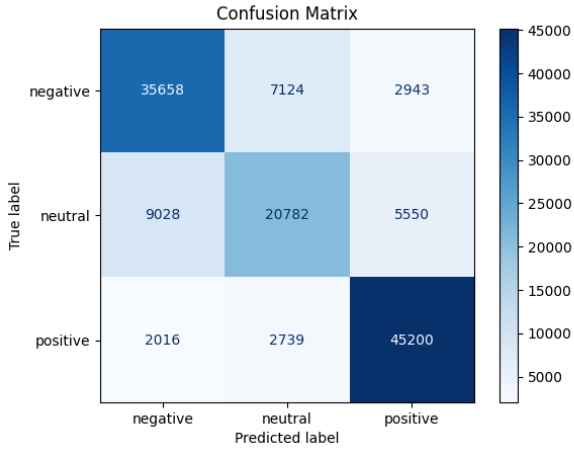
La curva di apprendimento del modello basata su rappresentazioni vettoriali estratte da BERT, evidenzia dinamiche significative nel bilanciamento tra capacità di generalizzazione e adattamento ai dati di training.



- Il training score parte da un valore elevato (0.80 con 20k esempi), indicando overfitting iniziale. Diminuisce progressivamente fino a 0.76 con 100k esempi, segnalando riduzione dell'overfitting e incapacità di apprendere relazioni complesse. Il decadimento riflette un bias elevato: il modello (SVM lineare) è troppo semplice per gestire la ricchezza semantica degli embedding BERT.
- Il Testing Score Parte da 0.76 (20k esempi) e migliora gradualmente fino a 0.78 (100k esempi), dimostrando una migliore generalizzazione all'aumentare dei dati. L'incremento (+0.02) è marginale nonostante il dataset quintuplichi, suggerendo che il modello non è in grado di imparare dai dati in maniera efficiente
- La distanza tra le curve inizialmente ampia indica sovradattamento. Con l'aumento dei dati, il gap si riduce, segnalando un migliore bilanciamento bias-varianza. Tuttavia, il calo del training score e il miglioramento limitato del validation score rivelano underfitting, infatti il modello non coglie appieno la complessità degli embedding. Quindi la convergenza riflette saturazione nell'apprendimento piuttosto che prestazioni ottimali.

BERT - Matrice di confusione

Dalla matrice di confusione è possibile confermare le criticità emerse nell'analisi della curva di apprendimento, in particolare la prestazione insoddisfacente sulla classe neutra rispetto alle altre due classi.



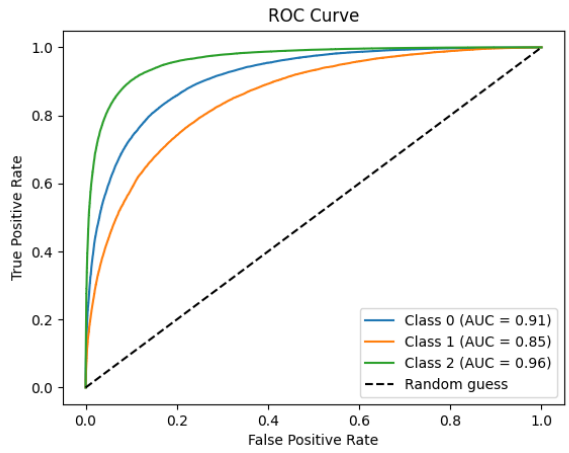
- Classe Positiva : Il modello dimostra ottime performance nel riconoscimento della classe positiva, con un 90% di predizioni corrette dei dati effettivamente positivi. Tuttavia, produce un 15% di falsi positivi , principalmente attribuibili a esempi neutri erroneamente classificati come positivi.
- Classe Neutra : Rappresenta la categoria più problematica, con solo il 59% di accuratezza . Gli errori sono distribuiti in modo significativo: il 25% degli esempi neutri viene classificato come negativo e il 16% come positivo , evidenziando una confusione sistematica con entrambe le altre classi.
- Classe Negativa : Le performance sono simili a quelle della classe positiva, ma con un' accuratezza leggermente inferiore . Il 15% degli esempi negativi è erroneamente classificato come neutro , sottolineando una sovrapposizione tra queste due categorie.

Il modello presenta difficoltà nel distinguere tra la classe neutra e quella negativa, con 9.028 esempi di classe neutra erroneamente classificati come negativi e 7.124 esempi negativi classificati come neutri.

Questa confusione tra le classi, unita alla bassa accuratezza sulla classe neutra, riflette i limiti emersi nell'analisi della curva di apprendimento, che indicano la presenza di underfitting e una difficoltà nell'estrarre pattern complessi dai dati.

BERT - ROC Curve

a curva ROC ottenuta risulta simile a quella precedentemente analizzata con l'uso delle embedding Word2Vec, mostrando un miglioramento marginale nella classificazione della classe neutra.



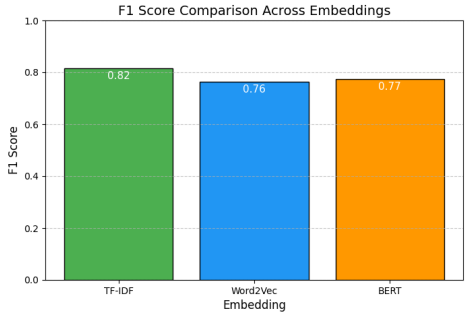
Tale miglioramento è evidente anche confrontando le rispettive matrici di confusione: con le embedding BERT, il modello riesce a distinguere in modo leggermente più efficace la classe neutra rispetto all'analisi con Word2Vec. Per quanto riguarda le prestazioni con classe positiva e negativa, esse rimangono perlopiù invariate.

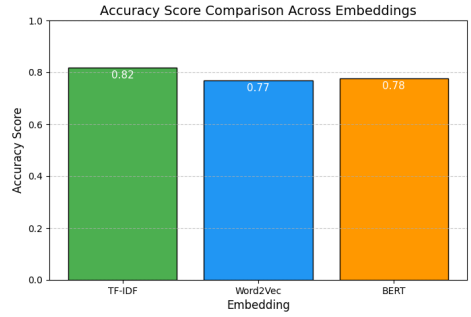
BERT - Conclusioni

Il modello basato su embedding BERT ottiene risultati complessivamente simili a quelli ottenuti in precedenza con Word2Vec, mostrando solo un lieve miglioramento nella rilevazione della classe neutra. Tuttavia, tale miglioramento risulta insufficiente per raggiungere le performance ottenute con TF-IDF.

Le difficoltà nella classificazione della classe neutra possono essere attribuite a problematiche già evidenziate con Word2Vec, in particolare a un possibile underfitting causato dalla standardizzazione dei valori dei vettori numerici, che porta a una perdita di informazioni rilevanti per il modello.

3.5.3 - Prestazioni di Regressione Logistica Discutiamo adesso delle prestazioni registrate:

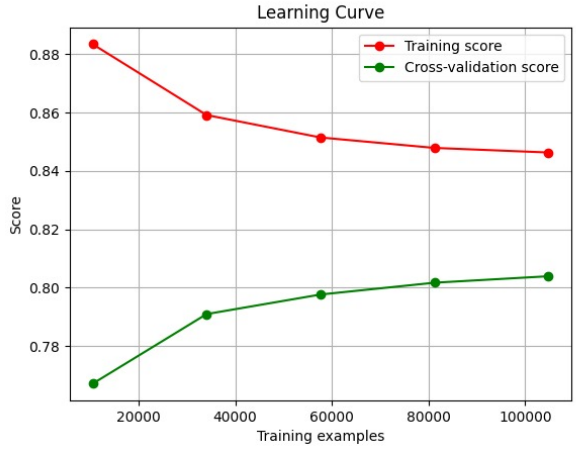




Come i due precedenti modelli anche la regressione logistica presenta migliori prestazioni con l'utilizzo di codifica TF-IDF, mantenendo comunque buone prestazioni anche con le altre due tipologie di codifica.

TF-IDF - Learning Curve

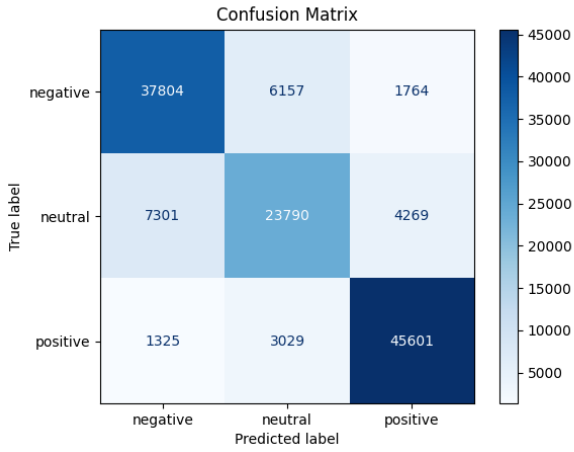
Il grafico mostra come il modello riesca ad ottenere un ottimo score sia in fase di addestramento che di testing:



- Il Training Score evidenzia inizialmente un momento di overfitting, probabilmente dovuto alla limitata quantità di dati disponibili. Questo comportamento, tuttavia, non è preoccupante, poiché analizzando la curva nei momenti successivi si nota un calo dello score del modello. Questo calo indica una riduzione della varianza e un miglioramento della capacità del modello di generalizzare
- Il Testing Score mostra una curva in costante crescita, indicando che il modello continua ad apprendere man mano che viene esposto a nuovi dati. Questo comportamento evidenzia la capacità del modello di ridurre progressivamente il bias con l'aumento del volume di dati, migliorando così la sua performance generale.
- Il divario tra le curve inizialmente è molto ampio, il che si aggiunge ai suggerimenti di un possibile overfitting iniziale.

Tuttavia, con l'aumento del dataset, tale divario tende a ridursi. Questo comportamento indica che, con il crescere della quantità di dati, il modello è in grado di bilanciare meglio il bias e la varianza, migliorando progressivamente la sua capacità di generalizzazione.

TF-IDF - Matrice di confusione

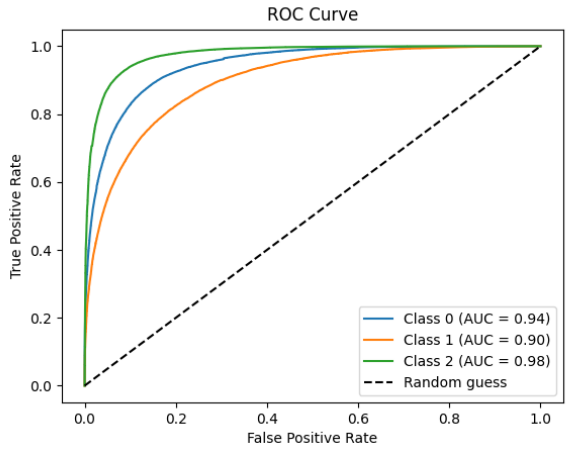


- Classe Positiva : il modello ottiene ottimi risultati nell'identificazione della classe positiva, difatti l'88% delle predizioni di sentiment Positivo sono corrette e il 91% delle effettive etichette positive sono state identificate correttamente. Gli errori seppur minimi si concentrano sull'errata classificazione dell'etichetta neutrale, in quanto l'8% delle predizioni di etichetta positiva sono in realtà etichetta neutra.
- Classe Neutra : risulta essere quelle più difficile da predire correttamente in quanto il 32% delle etichette neutre vengono classificate erroneamente come positive o negative. Solo il 68% delle etichette realmente neutrali vengono classificate tali. In particolare il 18% delle etichette neutrali predette sono in realtà etichette negative
- Classe Negativa : risultati simili a quelli della classe positiva, presenta maggiori errori soprattutto quando si trattano dati con etichetta neutra. Difatti il 15% delle etichette negative predette sono in realtà etichette neutrali.

Dalla Matrice di confusione possiamo quindi dedurre che il modello ha buone performance di predizione, ma presenta alcuni problemi riguardanti la classe neutra. In quanto si fa confusione tra etichetta negativa e neutrale.

TF-IDF - ROC Curve

La ROC curve evidenzia le ottime performance del modello su quasi tutte le classi.



In particolare, la classe 2 (Positiva) raggiunge una prestazione molto vicina al valore massimo. Analogamente, anche la classe 0 (Negativa) ottiene risultati eccellenti. Infine, la classe neutra, coerentemente con quanto osservato nelle analisi precedenti, risulta avere la AUC inferiore rispetto alle altre, pur mantenendo comunque una performance molto buona.

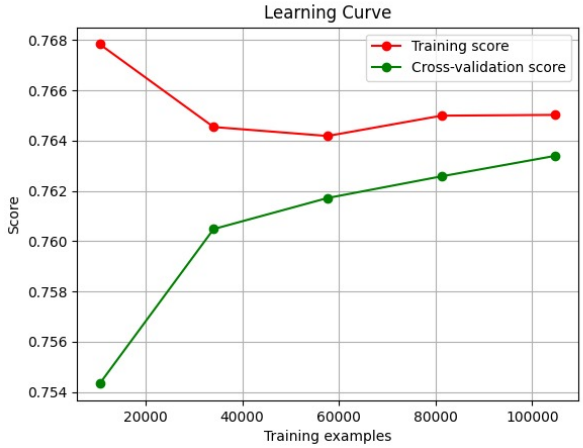
TF-IDF - Conclusioni

Il modello LR con codifica TF-IDF riesce ad ottenere risultati competitivi comparabili con gli altri modelli. Il motivo di queste ottime performance dipende principalmente dalla capacità della codifica TF-IDF di catturare l'importanza relativa delle parole in un documento, riducendo l'influenza delle parole comuni e enfatizzando quelle più distintive.

Inoltre, la regressione logistica, pur essendo un modello relativamente semplice, beneficia della sua natura lineare e della capacità di gestire efficacemente i dati ad alta dimensione, come quelli derivanti dalla codifica TF-IDF. Questo permette al modello di fare previsioni rapide e precise, mantenendo una buona capacità di generalizzazione.

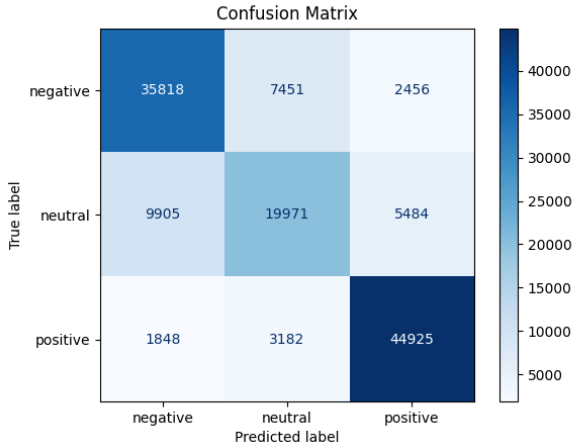
Word2Vec - Learning Curve

Il grafico risulta piuttosto simile a quello analizzato nella codifica precedente, con uno score più basso, in particolare:



- Il Training Score evidenzia un iniziale overfitting, tipico quando si dispone di un numero limitato di dati. Successivamente, si osserva un calo dello score, che coincide con una riduzione della varianza, indicando un miglioramento del modello. Ciò suggerisce che il modello stia imparando a generalizzare meglio, riducendo la tendenza a memorizzare i dati specifici del training set e migliorando così la sua capacità di adattarsi a nuovi dati.
- Il Testing Score mostra una curva in crescita costante, indicando una continua ottimizzazione nella riduzione del bias del modello. Questo comportamento riflette un miglioramento nella generalizzazione, poiché il modello diventa più preciso nel fare previsioni su nuovi dati non visti durante il training.
- La distanza tra le due curve evidenzia inizialmente un overfitting, con una grande separazione tra le curve di addestramento e validazione. Con l'aumento dei dati, le curve si avvicinano, indicando un miglior bilanciamento tra bias e varianza. Tuttavia, lo score rimane basso, suggerendo un possibile underfitting.

Word2Vec - Matrice di confusione



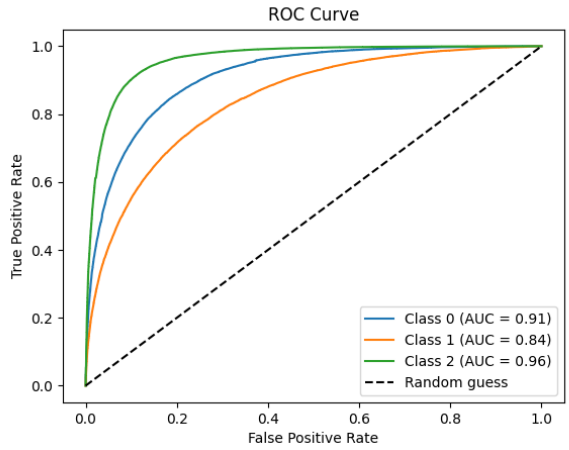
- Classe positiva : Il modello ottiene buoni risultati nella classificazione dei dati positivi. Su un totale di 49.955 elementi etichettati come positivi, ne ha identificati correttamente 44.925, con una percentuale di circa l'89%. Gli errori includono 5.030 dati positivi erroneamente classificati come negativi o neutrali, e 7.940 dati non positivi scambiati per positivi. Quindi, circa il 10% delle etichette effettivamente positive vengono classificate erroneamente, mentre il 15% delle etichette predette come positive sono in realtà negative o neutrali.
- Classe neutra : Il modello presenta difficoltà nel riconoscere correttamente questa classe, poiché il 28% delle etichette neutre viene erroneamente classificato come negativo e il 15% come positivo. Inoltre circa il 34% di etichette predette neutrali sono in realtà positive o negative.
- Classe negativa : Il modello performa generalmente bene nella predizione di questa classe, nonostante alcune difficoltà con la classe neutra, in cui il 20% delle sue predizioni sono errate, classificando erroneamente come negativi elementi che in realtà sono neutrali. Il modello riesce a classificare correttamente il 78% delle etichette effettivamente negative.

La matrice di confusione mostra che il modello presenta significative difficoltà nella classificazione del sentiment neutro. Infatti, gli errori riguardanti questa classe rappresentano la maggior parte degli errori complessivi del modello: circa 26.022 errori, che corrispondono all'85% del totale degli errori commessi.

Questo conferma l'underfitting osservato nelle analisi precedenti, poiché il modello non riesce a comprendere adeguatamente i dati relativi alla classe neutra, impedendo una corretta generalizzazione.

Word2Vec - ROC Curve

La ROC curve mostra che il modello ottiene ottimi risultati su quasi tutte le classi.



In particolare, la classe 2 (Positiva) si avvicina molto al massimo delle prestazioni. Anche la classe 0 (Negativa) raggiunge risultati eccellenti. Tuttavia, la classe neutra, come già evidenziato nelle analisi precedenti, presenta una AUC inferiore rispetto alle altre, sebbene mantenga comunque una performance buona.

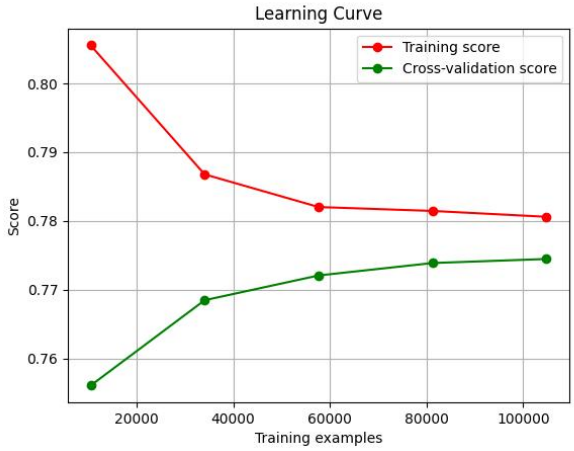
Word2Vec - Conclusioni

Il modello di Regressione Logistica affiancato all'embedding Word2Vec ottiene buoni risultati, ma non supera le performance ottenute con la codifica basata su TF-IDF. Le analisi indicano che ciò è dovuto a un problema di underfitting, in quanto il modello non è in grado di trattare in modo efficiente i dati codificati tramite Word2Vec.

Questo problema è legato alla necessità di standardizzare i dati embeddati, un aspetto che limita l'efficacia della regressione logistica nel gestire rappresentazioni dense e non lineari come quelle prodotte da Word2Vec.

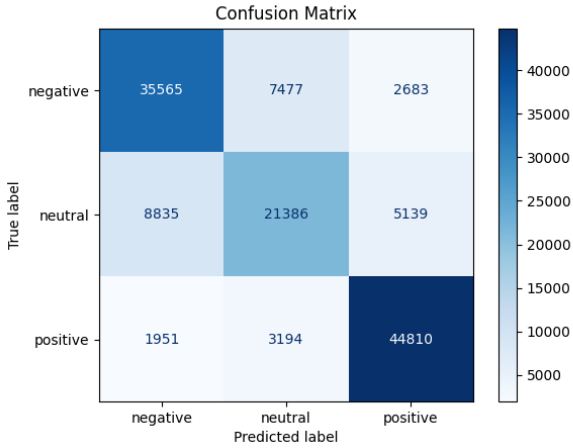
BERT - Learning Curve

La curva di apprendimento della regressione logistica con BERT embedding risulta essere piuttosto simile a quella analizzata nello studio del modello con embedding Word2Vec.



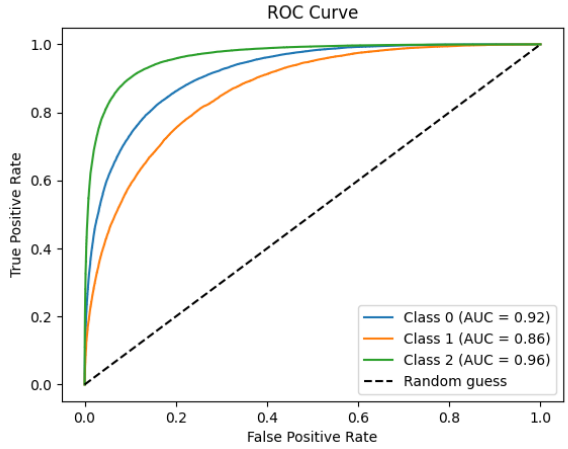
- Il Training Score iniziale suggerisce la presenza di un'elevata varianza nel modello, riconducibile a una limitata disponibilità di dati di addestramento.Con l'incremento della dimensione del dataset, si osserva una progressiva riduzione della varianza.
- Il modello, infatti, acquisisce la capacità di identificare relazioni sottostanti nei dati, migliorando la sua capacità di generalizzazione. Tale transizione è evidenziata da un lieve calo del Training Score, fenomeno non negativo in questo contesto, poiché riflette un bilanciamento tra adattamento ai dati e robustezza predittiva.
- Il Testing Score segue un andamento crescente, indicando un miglioramento nella capacità del modello di generalizzare su dati mai visti durante l'addestramento. L'aumento del Testing Score, combinato con la convergenza verso il Training Score, suggerisce che il modello sta beneficiando dell'aumento della quantità di dati.
- La distanza tra le due curve nella fase iniziale è ampia: il modello si adatta bene ai dati di addestramento (Training Score alto) ma non generalizza (Cross-Validation Score basso), segnalando overfitting e alta varianza. Con l'aumentare dei dati, il Training Score diminuisce, mentre il Cross-Validation Score cresce, riducendo il divario.
- Questo indica un miglioramento della generalizzazione e una riduzione della varianza. Nella fase finale, le curve convergono, segnalando un buon bilanciamento tra bias e varianza. Nonostante ciò il punteggio rimane relativamente basso, ciò può presagire che il modello è in underfitting

BERT - Matrice di confusione



- Classe positiva : Il modello raggiunge buone performance per la classe positiva. Su un totale di 49.955 esempi positivi, ne classifica correttamente 44.810, con un'accuratezza di circa l'89,7%. Gli errori includono 5.145 esempi positivi classificati erroneamente come negativi o neutrali, mentre 7822 neutri e negativi classificati come positivi.
- Classe neutra : Il modello ha difficoltà significative nella classificazione della classe neutra. Solo 21.386 esempi neutri su 35.360 sono classificati correttamente, con un'accuratezza del 60,5%. Il 25% degli esempi neutri viene erroneamente etichettato come negativo, e circa il 14% viene predetto come positivo.
- Classe negativa : Il modello performa meglio per questa classe. Su 45.725 esempi negativi, 35.565 vengono classificati correttamente, con un'accuratezza del 77,7%. Tuttavia, circa il 22% dei negativi viene confuso con la classe neutra o positiva. Inoltre, alcuni esempi di altre classi vengono erroneamente etichettatati come negativi.

BERT - ROC Curve



Il modello si comporta in modo eccellente con le classi positiva e negativa , mostrando una curva ben bilanciata. Si riesce infatti a ottenere un True Positive Rate (TPR) elevato senza incrementare in modo significativo il False Positive Rate (FPR) , evidenziando una buona capacità discriminativa per queste due classi.

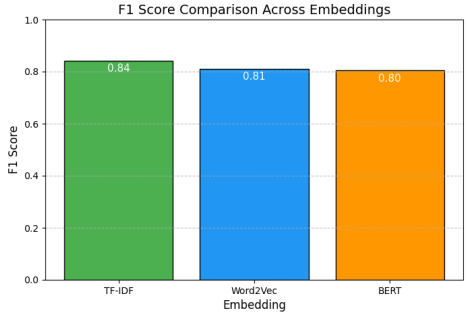
Tuttavia, la classe neutra continua a rappresentare una sfida. Nonostante la curva sia complessivamente buona, risulta meno performante rispetto alle altre. Il modello, per migliorare il TPR della classe neutra, tende ad aumentare il numero di False Positive , confermando la difficoltà di separare questa classe dalle altre.

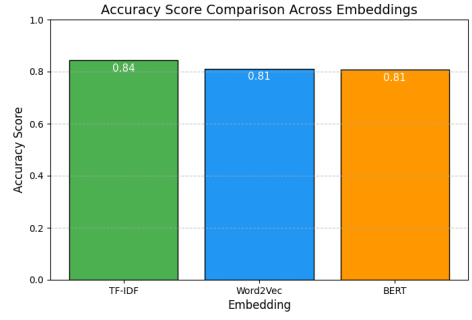
BERT - Conclusioni

Le analisi condotte portano a concludere che il modello di Regressione Logistica con embedding basato su BERT soffre di underfitting, in particolare per la classe neutra. Sebbene il modello riesca a distinguere efficacemente le classi negative e positive, evidenziando buone prestazioni su queste categorie, la sua capacità di identificare correttamente i sentimenti neutrali risulta limitata.

Questo comportamento è attribuibile principalmente alla standardizzazione dei vettori densi e continui generati da BERT. Tale processo, necessario per adattare i vettori alla Regressione Logistica, comporta una perdita di informazioni rilevanti, che penalizza il modello nella comprensione delle sfumature caratteristiche della classe neutra. Questa difficoltà era già emersa con gli embedding Word2Vec, confermando che la trasformazione dei vettori in rappresentazioni standardizzate riduce la ricchezza semantica disponibile per il modello.

3.5.4 - Prestazioni della Nete Neurale Discutiamo adesso delle prestazioni registrate:

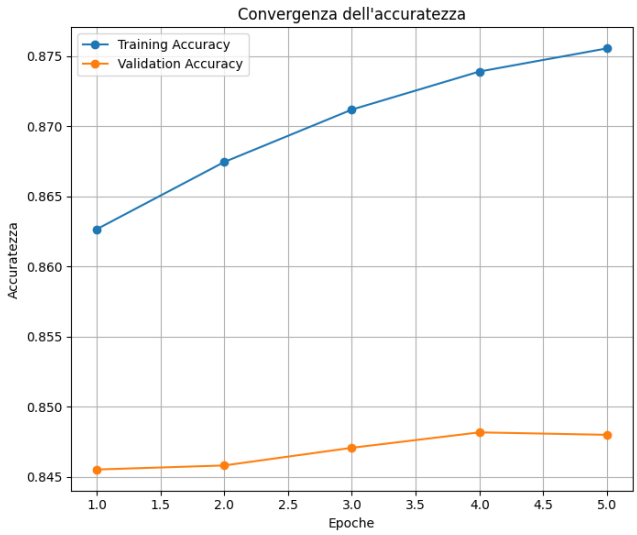




Dall'analisi di entrambi i grafici emerge chiaramente che il modello raggiunge performance superiori utilizzando l'embedding TF-IDF. Nei paragrafi seguenti, esamineremo il comportamento del modello con ciascun embedding e cercheremo di comprendere le ragioni per cui TF-IDF si dimostra la scelta più efficace per questa architettura.

TF-IDF - Learning Curve

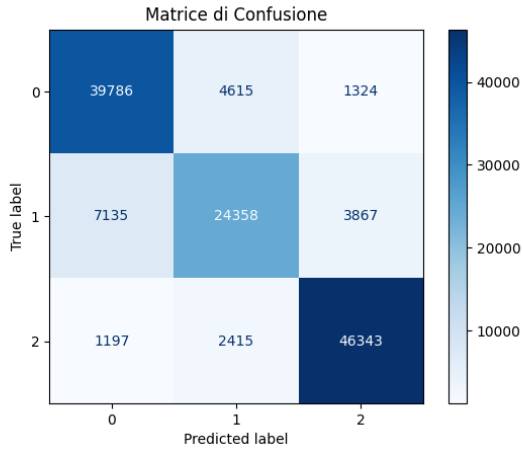
Dalla learning curve si evince che il modello progredisce in entrambe le fasi senza evidenti problemi.



La curva di Training mostra come il modello, con l'avanzamento delle epoche, riesca ad apprendere i pattern dai dati, migliorando progressivamente le proprie performance. Un trend simile è osservabile nella curva di Testing , dove, con l'avanzamento delle epoche, il modello migliora il proprio score.

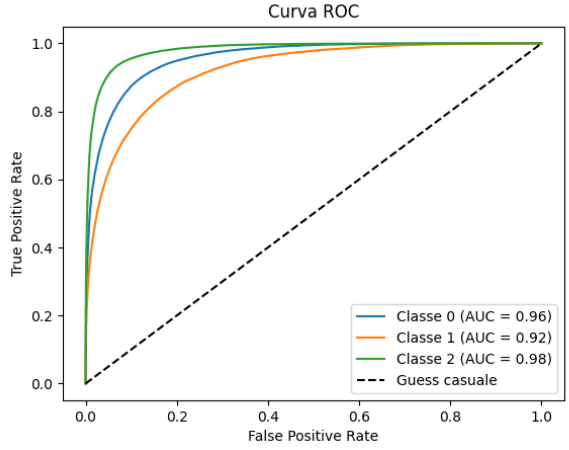
La distanza tra le due curve è relativamente bassa, circa 0,3, il che, sommato allo score elevato ottenuto in entrambi i contesti, suggerisce che il modello riesce a bilanciare in modo ottimale il bias e la varianza.

TF-IDF - Matrice di confusione



- Classe positiva : I risultati per la classe positiva sono ottimi, poiché il 92% dei dati effettivamente positivi vengono correttamente classificati. Il numero di falsi positivi è contenuto, con solo il 9% delle predizioni errate, la maggior parte delle quali riguarda dati effettivamente neutri.
- Classe neutra : Anche per la classe neutra i risultati sono buoni, con il 68% dei dati neutri correttamente classificati. Il restante 32% delle predizioni sono perlopiù errori di classificazione come classe negativa. Per quanto riguarda i falsi positivi, rappresentano il 22% delle predizioni totali, il che indica che il modello performa bene nella classificazione, evitando un eccessivo numero di falsi positivi.
- Classe negativa : I risultati per la classe negativa sono simili a quelli ottenuti per la classe positiva. In particolare, il modello è in grado di riconoscere correttamente l'87% dei dati effettivamente negativi. Il restante 13% consiste principalmente in predizioni errate di classe neutra. Parlando di errori, anche i falsi positivi sono relativamente bassi, rappresentando solo il 17% delle predizioni negative totali.

TF-IDF - ROC Curve



La ROC Curve conferma le analisi precedenti, poiché si può osservare chiaramente come il modello riesca a ottenere un ottimo TPR (True Positive Rate) mantenendo un FPR (False Positive Rate) molto basso. Ciò indica che il modello è in grado di identificare correttamente la maggior parte dei veri positivi senza commettere troppi errori, come falsi positivi.

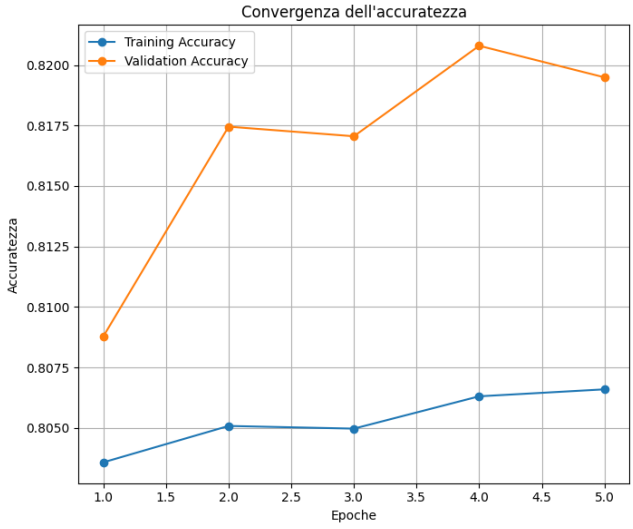
TF-IDF - Conclusioni

La combinazione di TF-IDF e Feedforward Neural Networks (FNN) consente di ottenere un modello semplice, rapido ed efficace, capace di raggiungere buone performance nell'analisi del sentiment. Questo approccio si distingue per la sua efficienza computazionale, in quanto non richiede la gestione di strutture dati complesse, come nel caso delle RNN, ma è comunque in grado di produrre risultati accurati. Ciò è dovuto alla capacità di TF-IDF di estrarre le parole significative dai dati, consentendo al modello di concentrarsi su termini davvero rilevanti per la classificazione del sentiment.

Inoltre, la combinazione di TF-IDF e FNN si traduce in un modello meno suscettibile al rischio di overfitting, rispetto ad altri modelli complessi come le RNN o le LSTM, specialmente quando il dataset non è particolarmente grande.

La capacità del FNN di lavorare con vettori sparsi e ben strutturati (come quelli generati da TF-IDF) consente al modello di generalizzare efficacemente, riducendo al minimo la necessità di modelli pesanti e computazionalmente costosi, e migliorando così l'efficienza generale.

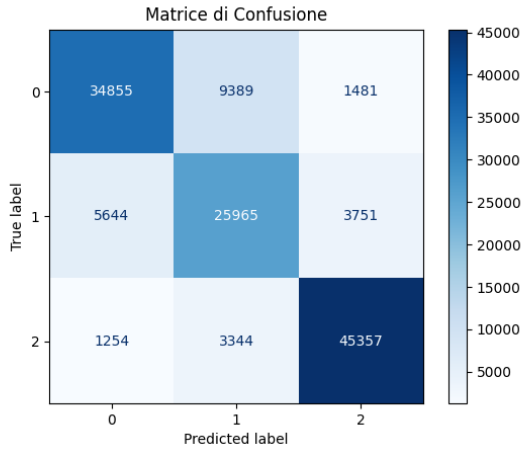
Word2Vec - Learning Curve



- Training score : L'accuratezza migliora di poco più di un decimo con il progredire delle epoche, un progresso modesto che suggerisce una capacità di apprendimento limitata . Il modello sembra avere difficoltà nell'estrarre pattern significativi dai dati, pur mantenendo una stabilità nelle sue performance senza peggioramenti.
- Testing score : L'aumento dell'accuratezza è ancora più contenuto, con incrementi di soli pochi centesimi. Questo indica che il modello non generalizza adeguatamente sui dati non visti, con variazioni minime tra le epoche che evidenziano la difficoltà nel trarre pieno vantaggio dalle informazioni presenti nelle feature di Word2Vec.
- La distanza tra le due curve, che inizia con pochi centesimi e si stabilizza su circa un decimo e mezzo alla fine, suggerisce che il modello potrebbe trovarsi in una fase di underfitting . Questo comportamento denota una difficoltà del modello nel migliorare significativamente sia sui dati di addestramento che su quelli di test.

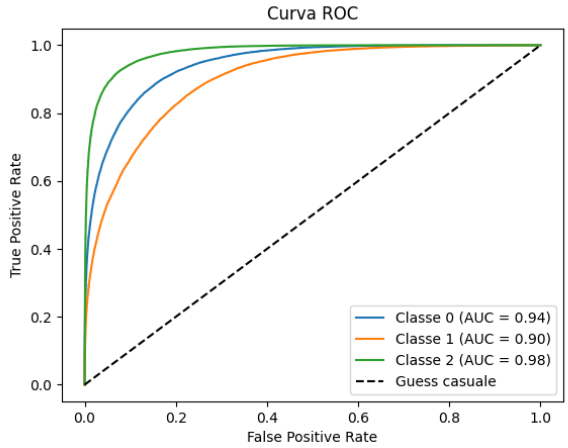
Word2Vec - Learning Curve

Performance classi :



- Classe Positiva : Il modello si dimostra particolarmente efficace nella classificazione della classe positiva, riuscendo a identificare correttamente il 91% dei dati effettivamente positivi. Tra i dati rimanenti, la maggior parte è stata classificata come neutra. Gli errori sono contenuti, poiché i falsi positivi rappresentano solo il 10% delle predizioni positive, anche in questo caso, la maggior parte degli errori riguarda dati neutri classificati come positivi.
- Classe Neutra : Anche in questo caso, la rete ha rilevato una buona percentuale di dati neutri, riuscendo a identificare il 73% dei dati neutri totali. Tra i dati rimanenti, il 15% è stato classificato come negativi e il 12% come positivi. Per quanto riguarda i falsi positivi, ovvero i commenti non neutri classificati erroneamente come neutri, si osserva un numero significativo, in particolare tra i dati negativi. Infatti, il 24% dei neutrali predetti riguarda dati negativi.
- Classe Negativa : Vengono riconosciuti il 78% dei dati effettivamente negativi, il restante 23% sono perlopiù dati negativi classificati come neutri. I falsi positivi ammontano a 6.898 ovvero il 16% delle predizioni totali negative.

Word2Vec - Roc Curvee



La curva ROC conferma quanto emerso dall'analisi della matrice di confusione, in quanto la curva della classe positiva (2) e quella della classe negativa (0) si avvicinano maggiormente alla curva ideale, che passa per i punti (0,1) e (1,1). Al contrario, la curva relativa alla classe neutra (1) mostra un'area sotto la curva (AUC) inferiore, indicando che il modello ha maggiori difficoltà nella classificazione corretta di questa classe.

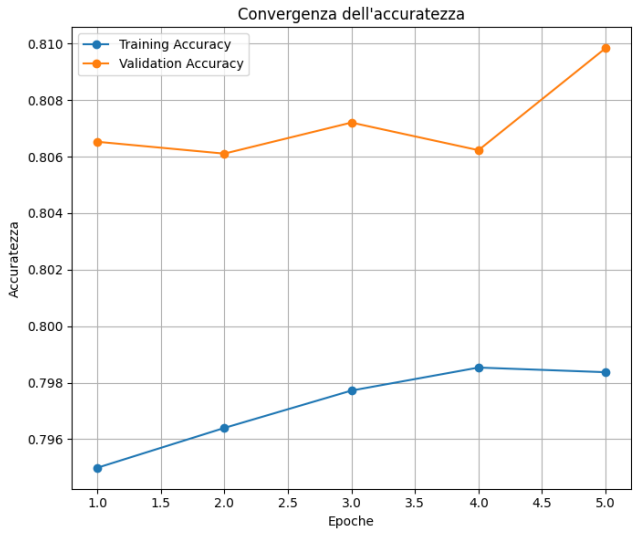
Word2Vec - Conclusioni

Il modello basato su Word2Vec embedding ottiene generalmente buoni risultati. Tuttavia, dalle analisi emerge che il modello presenta alcune difficoltà: sembra infatti non adattarsi completamente alla codifica, portando a una situazione di underfitting . Nonostante l'esposizione a nuovi dati, il modello non riesce a migliorarsi.

Questo comportamento può essere attribuito alla compatibilità limitata di Word2Vec con la rete neurale, in quanto il modello non riesce a catturare adeguatamente le relazioni semantiche più complesse tra le parole, riducendo la sua capacità di generalizzazione e apprendimento efficace.

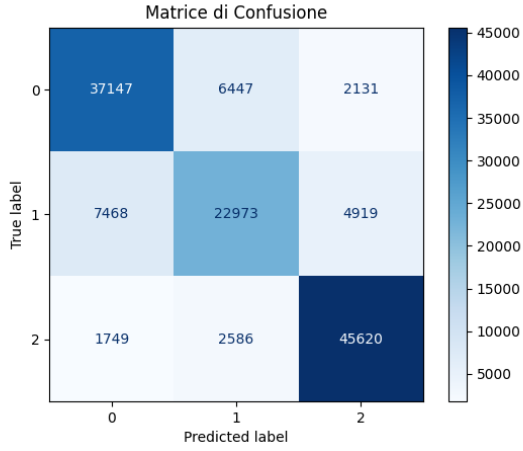
BERT - Learning Curve

La curva di apprendimento presenta caratteristiche simili a quella osservata con l'embedding Word2Vec.



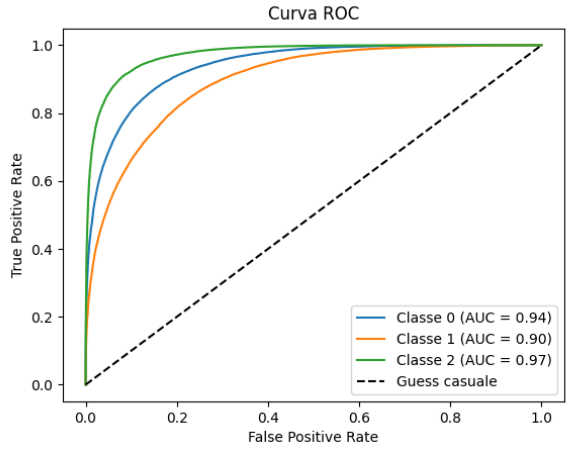
In questo caso, il Training Score migliora solo di millesimi durante le epoche, con un andamento analogo riscontrato anche nel Testing Score. Inoltre, la distanza tra le due curve rimane estremamente ridotta. Questi aspetti suggeriscono la stessa conclusione espressa in precedenza: il modello sembra soffrire di underfitting , incapace di apprendere in modo adeguato dai dati e di generalizzare efficacemente su nuovi esempi.

BERT - Matrice di confusione



- Classe Positiva : I risultati per questa classe sono molto soddisfacenti, con il modello che classifica correttamente il 91% dei dati effettivamente positivi. Il restante 9% riguarda principalmente errori di classificazione come classe neutra. Anche il numero di falsi positivi è contenuto: solo il 13% delle predizioni totali di classe positiva.
- Classe Neutra : Questa classe si rivela particolarmente problematica per il modello, che riesce a classificare correttamente solo il 65% dei dati effettivamente neutri. Tra i dati rimanenti, il 21% viene erroneamente classificato come negativo e il 14% come positivo. Inoltre, i falsi positivi, rappresentano circa il 28% delle predizioni di classe neutra, con la maggior parte di questi errori riguardanti dati effettivamente negativi.
- Classe Negativa : Il modello dimostra una buona capacità di classificazione per questa classe, identificando correttamente l'81% dei dati effettivamente negativi. Gli errori di classificazione riguardano principalmente dati negativi etichettati erroneamente come neutri. Tuttavia, i falsi positivi costituiscono circa il 19% delle predizioni totali di classe negativa, e tra questi, l'81% dei dati classificati erroneamente come negativi è effettivamente neutro.

BERT - ROC Curve



L'analisi della curva ROC rispecchia quanto osservato nella matrice di confusione. Le curve relative alla classe positiva (2) e alla classe negativa (0) si avvicinano sensibilmente alla curva ideale. Al contrario, la curva associata alla classe neutra (1) presenta un'area sotto la curva (AUC) significativamente più bassa, evidenziando le difficoltà del modello nel distinguere correttamente questa categoria rispetto alle altre.

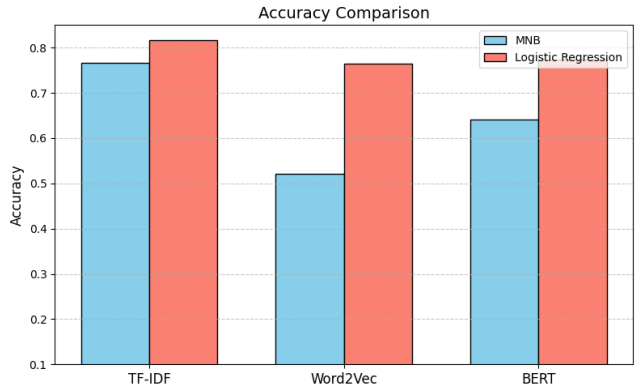
BERT - Conclusioni

Dall'analisi dei grafici precedenti si osserva una marcata somiglianza nei risultati ottenuti con l'embedding basato su BERT e quello basato su Word2Vec. Questa convergenza può essere spiegata dal fatto che, nonostante BERT sia progettato per catturare relazioni contestuali complesse e sequenziali tra le parole, la rete FNN utilizzata non è in grado di elaborare dati sequenziali. Essendo priva di una struttura adatta a sfruttare la natura sequenziale e contestuale di BERT, la FNN tratta le informazioni elaborate da BERT in modo simile a quelle di Word2Vec, che non conserva direttamente la sequenzialità. Questo limita i vantaggi di un embedding avanzato come BERT, che potrebbe esprimere al meglio il suo potenziale in un'architettura in grado di gestire sequenze, come una rete ricorrente (RNN) o un Transformer.

3.5.5 - Confronto tra modelli Il confronto tra i modelli seguirà un approccio semplice e diretto. Per ogni confronto, analizzeremo le prestazioni dei modelli in base alle tre metriche utilizzate precedentemente nell'analisi dei singoli modelli, con l'unica differenza che, al posto della Learning Curve, considereremo esclusivamente la metrica di accuratezza, che rappresenta il fondamento su cui la Learning Curve si basa.

3.5.6 - Confronto MNB - LR La scelta di confrontare Multinomial Naive Bayes (MNB) e Regressione Logistica è motivata dalla loro semplicità di implementazione e dalla velocità di addestramento , caratteristiche che li rendono ideali per analisi efficienti, soprattutto in contesti con risorse limitate o tempistiche ristrette

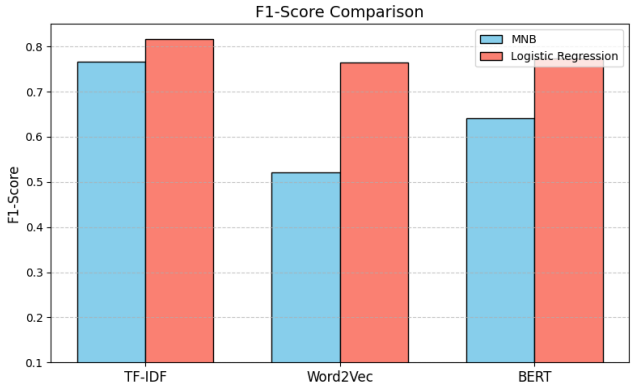
Accuracy



L'analisi dell'accuratezza evidenzia una netta superiorità della Regressione Logistica rispetto al Multinomial Naive Bayes (MNB). La Regressione Logistica dimostra una performance più consistente e superiore in tutte le rappresentazioni del testo analizzate, confermando la sua maggiore robustezza e versatilità. In particolare, i risultati ottenuti con embedding complessi, come Word2Vec, sottolineano la capacità del modello di adattarsi efficacemente a codifiche sofisticate, mantenendo prestazioni elevate.

Un ulteriore indicatore dell'efficienza della Regressione Logistica è dato dal confronto diretto dei punteggi: il miglior risultato ottenuto dal MNB, utilizzando l'embedding più adatto a tale modello, risulta essere approssimativamente pari al peggior punteggio della Regressione Logistica con Word2Vec. Questo divario evidenzia in modo inequivocabile la superiorità della Regressione Logistica in termini di adattabilità e prestazioni.

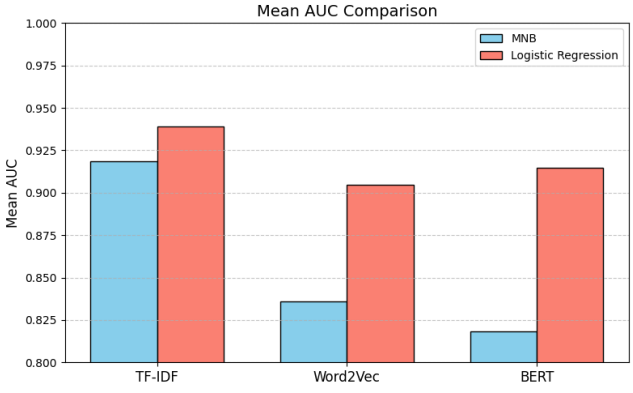
F1 Score



L'analisi dei punteggi F1 conferma che la Regressione Logistica supera costantemente il Multinomial Naive Bayes (MNB) in tutte le rappresentazioni testuali utilizzate. Questo risultato evidenzia la capacità della Regressione Logistica di bilanciare efficacemente precision e recall, garantendo una performance complessiva superiore rispetto a MNB.

Anche dall'osservazione del grafico emerge un punto già evidenziato nel confronto dell'accuratezza: la peggior performance registrata dalla Regressione Logistica corrisponde al miglior punteggio ottenuto dal Multinomial Naive Bayes.

Mean AUC



L' AUC (Area Under the Curve) è una metrica che valuta la capacità del modello di distinguere correttamente tra le classi, rappresentando l'area sotto la curva ROC. Un valore più alto di AUC indica una migliore separabilità tra le classi, e quindi una maggiore capacità del modello di fare previsioni corrette.

In questo confronto, abbiamo utilizzato la media dei tre score AUC del modello per ciascuna classe come metrica di confronto, al fine di ottenere una visione complessiva della performance del modello in termini di capacità di discriminazione tra le diverse classi.

L'analisi dei valori di AUC conferma che la Regressione Logistica supera sistematicamente il Multinomial Naive Bayes (MNB) in tutte le rappresentazioni testuali considerate, evidenziando una migliore capacità discriminativa tra le classi.

In particolare, la Regressione Logistica ottiene AUC medi significativamente più alti rispetto a MNB per ciascuna classe, indipendentemente dall'embedding utilizzato (TF-IDF, Word2Vec, BERT). Questi risultati sottolineano la maggiore efficacia della Regressione Logistica nell'interpretare sia rappresentazioni sparse sia dense, sfruttando al meglio le informazioni fornite dagli embeddings e garantendo una performance superiore in termini di capacità discriminatorie.

Conclusioni

Dai confronti effettuati, emerge in modo evidente che la Regressione Logistica è il modello vincente. Pur essendo un algoritmo semplice e veloce, riesce a ottenere performance altamente competitive, dimostrando una chiara superiorità rispetto a modelli simili, come il Multinomial Naive Bayes (MNB), che non sono in grado di gestire i dati testuali con la stessa efficacia.

La ragione di questa differenza risiede nella struttura stessa dell'algoritmo: la Regressione Logistica, grazie alla sua capacità di ottimizzare decision boundaries lineari, riesce a sfruttare al meglio le rappresentazioni testuali, soprattutto quando sono trasformate in embeddings più complessi e ricchi di informazioni come TF-IDF, Word2Vec o BERT. Al contrario, MNB, basandosi su assunzioni più rigide (come l'indipendenza tra le caratteristiche), fatica a catturare le relazioni più profonde nei dati.

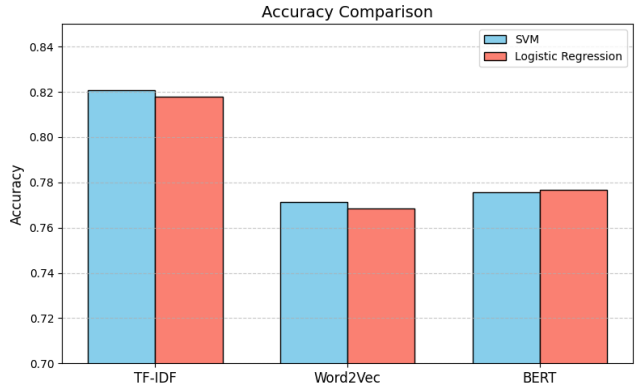
Il confronto a favore della Regressione Logistica si evidenzia ulteriormente nella classificazione del sentiment neutro, notoriamente più complessa. Mentre il Multinomial Naive Bayes, anche con la migliore rappresentazione testuale, riesce a classificare correttamente solo circa il 48% dei dati etichettati come neutri, la Regressione Logistica, utilizzando la stessa codifica, raggiunge una percentuale di classificazione corretta intorno al 68%.

Pur non essendo un risultato ottimale, rappresenta comunque un miglioramento, dimostrando una maggiore capacità della Regressione Logistica nel gestire la complessità e l'ambiguità tipiche delle classi neutre rispetto a MNB. Questo risultato consolida ulteriormente la superiorità della Regressione Logistica in compiti di sentiment analysis.

3.5.7 - Confronto SVM - LR La scelta di confrontare i modelli di Regressione Logistica (LR) e Support Vector Machine (SVM) è motivata dalla loro natura concettualmente affine, pur differendo negli obiettivi ottimizzati. Entrambi i modelli operano in uno spazio lineare, cercando di identificare una superficie di separazione ottimale tra le classi. Tuttavia:

- La Regressione Logistica adotta un approccio probabilistico, massimizzando la verosimiglianza dei dati attraverso una funzione logistica. La "retta" ottenuta minimizza la distanza complessiva dai punti, privilegiando la calibrazione delle probabilità di appartenenza alle classi.
- Le SVM , invece, perseguono un obiettivo geometrico: individuano l' iperpiano a margine massimo che separa le classi, massimizzando la distanza dai punti più vicini (vettori di supporto). Questo le rende particolarmente robuste a situazioni di margine ridotto o rumore nei dati.

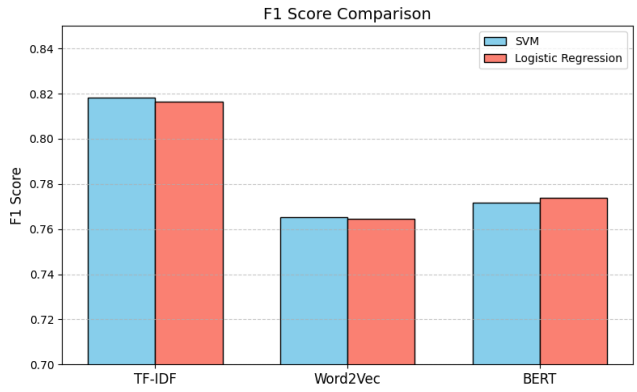
Accuracy



Il grafico dimostra che SVM e Regressione Logistica (LR) presentano prestazioni quasi sovrapponibili su tutte le codifiche testuali (TF-IDF, Word2Vec, BERT), confermando che i modelli lineari condividono limiti intrinseci nel gestire relazioni complesse o non lineari.

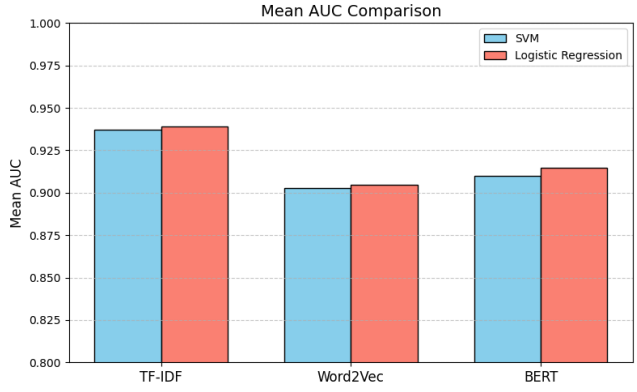
Le differenze osservate sono minime: SVM mostra un vantaggio trascurabile (<2%) con TF-IDF e Word2Vec, probabilmente per la sua capacità di ottimizzare il margine decisionale in spazi ad alta dimensionalità, mentre LR prevale lievemente con BERT, dove la sua natura probabilistica e la capacità di calibrare le feature gli consentono di sfruttare meglio le informazioni contestuali degli embedding avanzati. Queste variazioni, tuttavia, non sono statisticamente significative e non indicano una superiorità netta di un modello sull'altro.

F1-Score



Per quanto riguarda l'F1-Score, il discorso è sostanzialmente lo stesso: come si può osservare, non ci sono differenze significative tra i modelli. Le prestazioni risultano essere molto simili, suggerendo che entrambi i modelli gestiscono in modo comparabile il trade-off tra precision e recall.

Mean AUC



Nonostante il grafico relativo all'Area Under the Curve (AUC) possa sembrare non allineato con i confronti precedenti in termini di F1-Score e accuracy, in realtà le metriche analizzate offrono diverse prospettive sulla performance dei modelli. Mentre F1-Score e accuracy misurano direttamente la capacità di classificare correttamente le osservazioni, AUC si concentra sulla capacità di un modello di separare correttamente le classi a diverse soglie di decisione.

Nel confronto tra Support Vector Machine (SVM) e Regressione Logistica (LR) utilizzando la media dell'AUC sulle tre classi, si osserva che la Regressione Logistica ottiene risultati leggermente superiori rispetto a SVM, specialmente quando si utilizzano rappresentazioni testuali avanzate come BERT.

Nonostante entrambi i modelli mostrino valori di AUC elevati, indicando una solida capacità discriminativa, la Regressione Logistica si distingue per una performance media migliore, probabilmente grazie alla sua efficienza nel gestire features ad alta dimensionalità e alla robustezza in scenari con dati ben bilanciati.

Conclusioni

Date le seguenti analisi, non è possibile decretare un effettivo vincitore tra i modelli, in quanto entrambi ottengono performance con differenze minime. Questo ci porta quindi alla necessità di scegliere il modello migliore in base ai requisiti non funzionali del nostro progetto.

In particolare, se consideriamo come metrica il trade-off Velocità/Precisione, la Regressione Logistica emerge come la scelta preferibile. Nonostante le differenze nei punteggi di performance essere trascurabili, la Regressione Logistica riesce a ottenere risultati quasi altrettanto precisi come SVM, ma con un minor costo computazionale sia in fase di training che di predizione. Questo la rende una scelta vantaggiosa in contesti che richiedono efficienza e velocità, senza compromettere significativamente la qualità dei risultati.

3.6 | Deployment

Per il deployment del sistema di sentiment analysis, la scelta del modello è ricaduta sulla rete neurale feedforward in combinazione con la codifica TF-IDF. Questa decisione è stata guidata da un'attenta analisi delle prestazioni di diversi modelli, tra cui il Multinomial Naive Bayes, la regressione logistica e il Support Vector Machine, oltre a rappresentazioni testuali più avanzate come Word2Vec e BERT.

La combinazione FNN con TF-IDF si è dimostrata una soluzione ottimale in termini di accuratezza, F1-score e capacità discriminativa, pur bilanciando la complessità computazionale. Le principali motivazioni alla base della scelta della rete neurale feedforward con TF-IDF sono le seguenti:

1. **Compatibilità tra FNN e TF-IDF**: La rappresentazione TF-IDF genera feature sparse che enfatizzano l'importanza relativa dei termini nel testo rispetto al corpus, riducendo il rumore nei dati. La FNN, configurata con un numero adeguato di neuroni e funzioni di attivazione non lineari, riesce a sfruttare queste rappresentazioni per apprendere relazioni complesse, offrendo previsioni precise. Questa combinazione ha superato approcci lineari come LR e MNB in scenari con dati a rappresentazione sparsa.
2. **Prestazioni sulla Classe Neutra**: La classificazione della classe neutra si è rivelata particolarmente complessa, come evidenziato dai risultati ottenuti. Nonostante il costo computazionale maggiore rispetto ai modelli lineari, la FNN si è distinta per un F1-score superiore su questa classe, riducendo il numero di errori rispetto a SVM e LR.
3. **Confronto con Modelli Alternativi**: Pur essendo modelli computazionalmente leggeri, MNB e LR hanno mostrato limiti nella capacità di cogliere relazioni complesse nei dati. Anche l'uso di rappresentazioni dense come Word2Vec e BERT non ha garantito prestazioni ottimali con LR e SVM, evidenziando problemi di underfitting e confusione nelle classi neutrali. La combinazione FNN + TF-IDF ha bilanciato efficacemente precision e recall, mantenendo buone performance su tutte le classi.
4. **Adattabilità e Scalabilità**: La rete neurale offre la possibilità di essere facilmente scalata o ottimizzata aggiungendo strati o neuroni. Questo consente al sistema di adattarsi a rappresentazioni testuali più complesse o a dataset più grandi in futuro, garantendo flessibilità nell'evoluzione del modello.

5. **Efficienza Computazionale**: Sebbene le FNN siano generalmente più dispendiose rispetto ai modelli lineari, la scelta di una rete semplice, combinata con la natura sparsa delle feature TF-IDF, ha permesso di contenere i tempi di training e inferenza entro limiti accettabili. In scenari operativi, questa configurazione risulta gestibile anche in termini di risorse hardware.

In conclusione, la scelta di una rete neurale feedforward con TF-IDF rappresenta un compromesso ideale tra precisione, flessibilità e scalabilità. Nonostante il costo computazionale leggermente superiore rispetto ai modelli lineari, la FNN ha dimostrato di essere particolarmente efficace nel bilanciare le metriche chiave e nell'affrontare la complessità della classe neutra, offrendo un modello performante e pronto per l'implementazione.

4 | IMPLEMENTED APPLICATION

L'applicazione che utilizzerà il modello di sentiment analysis sarà strutturata principalmente in due sezioni: una dedicata agli utenti (studenti) e l'altra riservata alle istituzioni o ai singoli docenti. Gli studenti, utilizzando un codice fornito loro dai docenti, potranno accedere a semplici form in cui sarà possibile inserire commenti su un tema stabilito dall'insegnante. Una volta aggiunti i commenti desiderati, potranno tornare alla homepage ed eventualmente accedere ad altri form disponibili. I docenti, invece, avranno la possibilità di visualizzare informazioni generali relative ai form creati e ai commenti ricevuti, creare nuovi form specificando un titolo e una descrizione/tema, e infine accedere a una visualizzazione dettagliata delle informazioni relative a ciascun form.

Tecnologie utilizzate Per la realizzazione dell'applicazione, abbiamo deciso di utilizzare tecnologie moderne che garantissero un elevato riuso e personalizzazione dei componenti, oltre a consentire una semplice integrazione con quelli già esistenti. Segue un riassunto delle tenologie usate per frontend e backend

- Frontend: È stata adottata una combinazione di Vue, TypeScript e TailwindCSS, poiché queste tecnologie risultano già familiari al nostro team e rispondono pienamente alle esigenze progettuali. I vari moduli frontend interagiscono con il backend tramite chiamate fetch che sfruttano il protocollo HTTP, mentre per la gestione della navigazione interna è stato utilizzato Vue Router.

- Backend: Per il backend, abbiamo ritenuto che l'implementazione di un semplice modulo FastAPI fosse sufficiente a soddisfare le funzionalità richieste dal frontend. Non essendo necessario gestire dati particolarmente variegati o complessi, abbiamo deciso di non adottare un servizio di database dedicato. Al suo posto, i dati vengono memorizzati e gestiti utilizzando file JSON, replicando la logica dei database non relazionali.

Data la configurazione altamente flessibile, ma poco robusta, non ci siamo concentrati su aspetti come il multithreading per gestire accessi simultanei o sull'integrazione di sistemi di autenticazione e registrazione. Questi elementi, tuttavia, rappresentano spunti per possibili miglioramenti futuri. Inoltre, è stata predisposta un'istanza specifica del modello di machine learning (ML) selezionato in precedenza, salvando i binari e le informazioni necessarie per il suo funzionamento.

Flusso operativo Il flusso delle azioni più comuni che un utente può compiere sulla piattaforma è il seguente:

1. Creazione del form: Il docente crea un nuovo form, inserendo un titolo e una descrizione/tema. Una volta creato, viene generato un codice univoco che il docente condivide con gli studenti.
2. Accesso degli studenti: Gli studenti accedono al form utilizzando il codice fornito dal docente e inviano uno o più commenti sul tema specificato e poi tornano alla homepage.
3. Elaborazione dei commenti: I commenti vengono elaborati dal modello di sentiment analysis, memorizzati nel

sistema ed inviate alle varie interfaccie frontend dei docenti.

4. Aggiornamento delle informazioni: Le informazioni relative al singolo form e quelle generali vengono automaticamente aggiornate e rese disponibili al docente.
5. Feedback ai studenti: In base ai risultati dell'analisi dei sentimenti estratti dai commenti, i docenti possono fornire un feedback agli studenti, completando così il ciclo di interazione.

5 | FINAL CONSIDERATIONS

5.1 | General note

Nonostante il suo potenziale, la sentiment analysis deve affrontare diverse sfide. Una delle più rilevanti è rappresentata dall'innata complessità del linguaggio umano. Elementi come il sarcasmo, l'ironia e le sfumature culturali sfuggono spesso a una classificazione diretta, creando difficoltà anche per gli algoritmi più avanzati. Inoltre, l'ambiguità di alcune parole e la natura in continua evoluzione del linguaggio richiedono aggiornamenti e perfezionamenti costanti nei modelli di sentiment analysis. Un'ulteriore sfida è data dalla necessità di elaborare dati multilingue, il che implica l'uso di strumenti in grado di operare efficacemente in contesti linguistici diversi.

Negli ultimi anni, i progressi nel machine learning e nel deep learning hanno significativamente migliorato le capacità della sentiment analysis. Le reti neurali, in particolare le recurrent neural networks (RNNs) e i transformers, hanno permesso ai modelli di catturare schemi complessi nel linguaggio e di aumentare la precisione nella classificazione dei sentiment. Inoltre, i modelli linguistici pre-addestrati, come BERT (Bidirectional Encoder Representations from Transformers), hanno rivoluzionato il settore fornendo rappresentazioni testuali robuste che possono essere ottimizzate per compiti specifici di sentiment analysis.

5.2 | Model improvements

Per quanto riguarda i modelli da noi selezionati, addestrati e valutati, esistono diversi elementi che ampliano i margini di miglioramento. In primo luogo, la raccolta e l'utilizzo di dataset più ampi, accuratamente processati e annotati, potrebbe incrementare il livello di generalizzazione e ridurre il bias presente nei modelli. Questo approccio contribuirebbe anche a mitigare l'errore introdotto dalle scelte progettuali adottate, come l'utilizzo dello stesso modello per classificare i dati utilizzati nell'addestramento, una pratica che può compromettere l'accuratezza delle valutazioni.

Un altro aspetto da considerare riguarda i modelli stessi. Sarebbe possibile addestrare e testare reti neurali più complesse rispetto a quelle utilizzate, oppure esplorare tecniche avanzate come il fine-tuning di modelli pre-addestrati (foundation models) per adattarli alla sentiment analysis. Inoltre, l'adozione di approcci ensemble, che combinano metodologie rule-based con modelli di ML e DL, potrebbe migliorare ulteriormente la precisione.

Approcci che includano livelli di granularità più elevati rappresentano un altro potenziale vantaggio, nonostante le complessità che comportano. Lo sviluppo di modelli specifici per il rilevamento del sarcasmo o dell'ironia — contesti in cui anche i modelli più avanzati spesso faticano — risulterebbe particolarmente utile. Non meno importante sarebbe la creazione di modelli multilingua, capaci di operare in contesti linguistici diversi, o modelli multimodali, che integrino informazioni provenienti da diverse fonti, come il tono vocale o le espressioni facciali, per aumentare l'efficacia della sentiment analysis nel mondo reale.

5.3 | Application improvements

Anche l'applicazione web sviluppata presenta margini di miglioramento significativi. Ad esempio, l'esperienza utente potrebbe essere ampliata, includendo sezioni interattive che spieghino il funzionamento dei modelli di sentiment analysis, oppure offrendo strumenti ludici che mostrino il loro impatto nel mondo reale. Per la sezione dedicata alle istituzioni, si potrebbero integrare statistiche più dettagliate, infografiche chiare e la possibilità di creare form più articolati, rendendo l'interfaccia più informativa e intuitiva.

6 | CONCLUSION

In conclusione, la sentiment analysis è uno strumento potente per interpretare le dimensioni emotive e soggettive del testo. Sfruttando tecniche computazionali per analizzare il linguaggio, offre preziose intuizioni sugli atteggiamenti e le opinioni umane, diventando indispensabile in numerosi ambiti, dal business alla politica, fino all'accademia e oltre. Con l'evolversi del settore, la sentiment analysis promette di approfondire la nostra comprensione della comunicazione umana e di migliorare la nostra capacità di prendere decisioni basate sui dati in un mondo sempre più complesso.