UNIVERSITÀ DEGLI STUDI DI SALERNO

Dipartimento di Scienze Economiche e Statistiche

Corso di Laurea Magistrale in Scienze Statistiche per la Finanza

Tesi in

Statistics for Finance and Insurance

LA SOCIAL SENTIMENT ANALYSIS PER LA PREVISIONE DEGLI ANDAMENTI DI MERCATO

Relatore:

Ch.ma Prof.ssa Alessandra Amendola

Candidato:

Gabriele Sollai

matr. 0222400653

ANNO ACCADEMICO 2021/2022

INDICE

INTRODUZIONE

Nella storia recente, l'importanza dei social media è aumentata a dismisura. Si stima che circa il 59.3% della popolazione mondiale, 4.74 miliardi di persone, utilizzano i social. I numeri parlano di un aumento di circa 190 milioni di persone iscritte solo nell'ultimo anno. Gli utenti passano in media 2 ore e 28 minuti ogni giorno sui social più in voga, in crescita, anche questo dato, di un minuto rispetto all'Ottobre 2021[1]. I social, dunque, sono diventati luoghi virtuali fondamentali nella vita di quasi l'intera popolazione munita di un accesso a internet. In essi si possono ritrovare amici di vecchia data, divertirsi, parlare con persone distanti geograficamente. Una delle evoluzioni più grandi, inteso sia in modo positivo che in modo negativo, è sul lato dell'informazione: il 34% degli utenti utilizza i social per informarsi rispetto a ciò che accade nel mondo, tralasciando quindi i mezzi di informazione ordinaria come televisione o giornali cartacei, preferendo a questi ultimi le loro versioni online, più veloci e facili da reperire, più comodi nella loro fruizione, portando, però, vari lati negativi quali la diffusione di fake news che, specialmente durante il periodo pandemico hanno avuto particolare allargamento tra la massa di persone utilizzanti i maggior social.

Il dato più attinente e più indicativo rispetto a questo lavoro di Tesi, però è la percentuale di persone che utilizzano Twitter come social media preferito per il reperimento di news in tutti gli ambiti della nostra vita: dalla finanza alle notizie di cronaca fino ad arrivare a qualunque curiosità che un utente possa avere.

La finanza, in particolare, è un topic molto presente su questo social e, di conseguenza, ha senso parlare di influenza sul pubblico interessato a quest'ultima rispetto a scelte di investimento future o rispetto ad aggiornamenti delle proprie posizioni. Scopo di questo lavoro sarà proprio quello di andare a verificare se è possibile ricavare delle features tali da poter permettere, attraverso la sentiment analysis dei tweets e attraverso l'analisi delle serie storiche dei titoli, di prevedere la crescita o la decrescita di un data azione, indice o obbligazione.

Verranno utilizzate, all'interno del presente lavoro, a questo scopo, tecniche di web scraping per la raccolta dei tweets riguardanti un determinato argomento, in particolar modo il topic principale della tesi, l'indice S&P 500, tecniche di data visualization per cogliere al meglio l'andamento sia del sentiment sia dell'indice prescelto e, infine, modelli di machine learning per costruire e adattare al meglio ai nostri dati un modello che ci permetta di andare a ricavare informazioni preziose per effettuare delle scelte di investimento, per coloro che

[1] www.datareportal.com/social-media-users

hanno in programma di immettere capitali nel mercato finanziario, o delle scelte di aggiornamento delle proprie posizioni finanziarie, per coloro che hanno già capitali all'interno del mercato ma vogliono ottimizzare il proprio portafoglio. Verranno poi effettuati paragoni tra l'implementazione dei modelli con e senza il sentiment stimato per quantificare l'impatto sulle previsioni della variabile aggiunta. Il tutto verrà implementato utilizzando il linguaggio di programmazione "Python", uno dei linguaggi più versatili e utilizzati nell'informatica odierna.

Il lavoro è suddiviso in quattro capitoli. Nel primo si farà una panoramica generale riguardante la storia e gli sviluppi del Natural Language Processing, della sentiment analysis e del machine learning. Nel secondo si daranno accenni teorici necessari per lo sviluppo del lavoro, analizzando da vicino le tecniche utilizzate successivamente. Nel terzo capitolo sarà esposto il case study e i risultati ottenuto mentre nel quarto ed ultimo capitolo vi saranno le conclusioni e i possibili sviluppi futuri del lavoro.

CAPITOLO UNO

ACCENNI STORICI

SOMMARIO: 1.1 L'indice S&P500: tra storia e dati; 1.2 La storia del Natural Language Processing; 1.3 La Sentiment Analysis; 1.4 Il Machine Learning

1.1 L'indice S&P500: tra storia e dati

Prima di entrare nel vivo dell'analisi, è opportuno descrivere la storia dell'indice scelto per il lavoro: lo Standard & Poor's 500 (di seguito S&P500).

Gli indici finanziari non sono altro che panieri di azioni in cui il prezzo dell'indice dipende dai titoli contenuti dallo stesso. In particolare, l'indice preso in considerazione è uno dei più importanti indici azionari nordamericani, calcolato a partire dal 1957, anno dal quale ha preso sempre più peso e rilevanza all'interno del mercato, divenendo principale benchmark dei titoli azionari quotati a Wall Street ed ha, come vedremo anche nell'analisi dei tweets, un quantitativo elevatissimo di opzioni costruite su di esso. L'indice è stato costituito da Standard & Poor's, l'agenzia di rating più rilevante, insieme a Moody's, degli Stati Uniti e del mondo intero. Al suo interno vi sono 505 titoli riferenti a 500 società quotate sul NYSE e sul Nasdaq che arrivano ad avere una rappresentatività pari circa all'80% della capitalizzazione dell'intero mercato azionario statunitense. L'indice è in costante evoluzione e, proprio per questo, molte società sono spesso analizzate per rientrare o uscire dal calcolo dello stesso.

I requisiti per entrarne a far parte sono precisi e stringenti, i titoli devono infatti:

- Avere una capitalizzazione superiore a 6.1 miliardi di dollari;
- Avere un flottante almeno del 50%
- Avere un volume di scambio registrato negli ultimi 6 mesi non inferiore a 250'000 azioni;
- Avere un valore medio annuo superiore ad un dollaro

Le società, infine, devono presentare utili di bilancio nei quattro precedenti periodi trimestrali.

Per entrare a far parte dell'indice, i titoli vengono sottoposti ad un'analisi seguendo il metodo della capitalizzazione flottante: invece che andare a moltiplicare il totale di azioni immesse sul mercato dell'azienda per il prezzo delle stesse, viene moltiplicato il prezzo per il numero di azioni prontamente disponibili alla vendita, escludendo, di conseguenza, quelle possedute da addetti ai lavori, promotori o governi. Il calcolo, dunque, non verrà influenzato da tutte quelle azioni non disponibili per l'immediata vendita, rendendo quindi più restrittivo l'ingresso nel calcolo dell'indice per i titoli in cui la maggior parte delle azioni viene posseduta da governi o bloccate per qualsiasi motivo[2].

Al momento della scrittura del lavoro il peso rilevato maggiore di un titolo all'interno

[2] www.borsaitaliana.it/notizie/sotto-la-lente/sp500.htm

dell'indice è quello relativo ad Apple Inc. che può vantare il 7.16% della quota totale delle 500 imprese, seguita dal 6,02% di Microsoft Corporation e da, rispettivamente, 3.38% e 2.14% di Amazon e Tesla. I primi 5 titoli hanno un peso specifico cumulato addirittura del 22.5%[3].

L'andamento dell'indice, inoltre, è spesso associato all'andamento dell'intera economia americana ed occidentale, tanto che viene preso come benchmark per investimenti rischiosi per verificarne l'andamento e il costo-opportunità.

Qui di seguito viene rappresentato l'andamento dell'indice negli ultimi 5 anni. ciò che viene visualizzato è, in realtà, l'andamento del prezzo di chiusura aggiustato del titolo. Quest'ultimo è definito in finanza come il prezzo corretto ed aggiustato calcolato dopo le "corporate actions" che possono essere rappresentate dalla distribuzione dei dividendi o dalla divisione delle azioni (stock split) in modo tale da poter raggiungere una più ampia fetta di pubblico. In questo caso, essendo un indice, però, il prezzo di chiusura aggiustato sarà identico al prezzo di chiusura registrato senza aggiustamenti.

Figura 1: Serie storica del S&P500

È possibile notare come il prezzo sia passato da meno di 2500$ a più di 4500$ nel periodo preso in considerazione e questo è accaduto nonostante una brusca frenata riportata all'inizio del 2020 dovuta allo scoppio della pandemia da Covid-19 e alla conseguente paura generata e diffusa nel mercato per l'estrema incertezza che governava quei momenti. Trend di questo tipo sono, nel lungo periodo, una fonte di attrazione per gli investitori che, spesso,

[3] www.marketscreener.com/quotazioni/indice/S-P-500-4985/composizione

si rivolgono a fondi a gestione passiva (ETF), nel tentativo di replicare l'andamento di questo indice in modo da riuscire a realizzare rendimenti che corrispondono al rendimento dell'intero mercato.

Nella figura sottostante, invece, viene riportato l'andamento storico degli ultimi 5 anni dei rendimenti ottenuti se gli investitori fossero stati in grado di replicare l'indice. È possibile osservare come la parte in alto, rappresentante i rendimenti positivi, è sovrastante rispetto a quella in rossa, rappresentante i negativi. Il picco, sia in negativo che in positivo si attesta intorno al 10% di risaltita e di discesa intorno al periodo dello scoppio della pandemia dovuta al Coronavirus agli inizi del 2020.

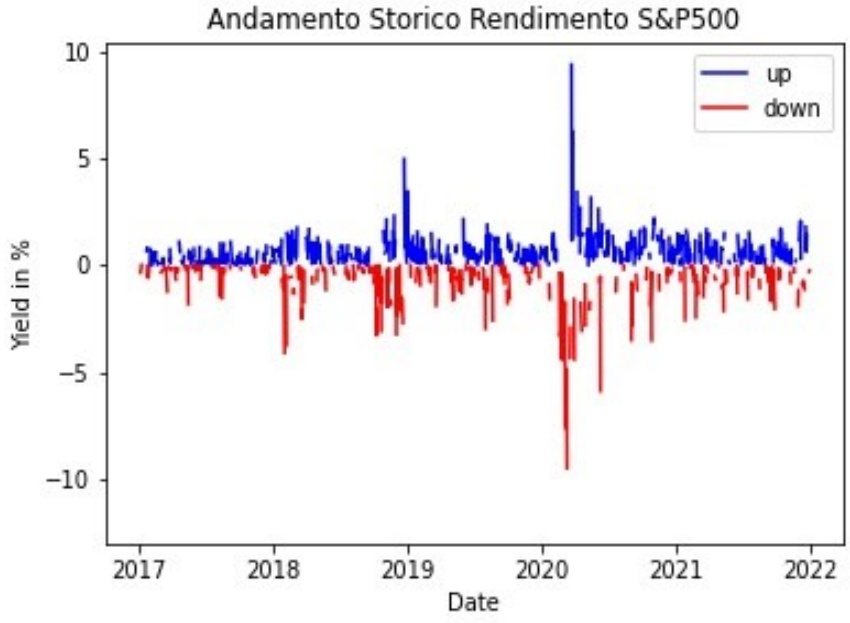


Figura 2: Serie storica del rendimento S&P500

1.2 La storia del Natural Language Processing

Il Natural Language Processing (NLP) è una branca dell'intelligenza artificiale che ha come obiettivo la comprensione, da parte delle macchine moderne, del linguaggio umano. L'elaborazione del linguaggio naturale, posto così come è, non è assolutamente comprensibile alle macchine, programmate per comprendere codice binario composto da 0 e 1, scopo della materia, di conseguenza particolarmente attinente con il lavoro sottoposto, è quindi proprio quello di ricercare un metodo efficace ed efficiente per la traduzione del

linguaggio umano ad un linguaggio di programmazione per poi tradurlo ancora una volta in codice binario in modo che la macchina riesca a comprenderlo e ad effettuare i calcoli richiesti.

La storia dell'NLP è particolarmente recente ma anche in una fase di crescita esponenziale causata dalla miriade di applicazioni in cui esso potrebbe apportare degli sviluppi fondamentali: un esempio su tutti è l'ambito pubblicitario online con cui le aziende riescono a recepire le recensioni e le preferenze dei clienti in modo automatizzato, portando degli efficientamenti sia dal punto di vista della qualità della pubblicità sia dal punto di vista della quantità di vendita, riuscendo ad intercettare una fetta di pubblico più interessata al proprio prodotto (targeting dei clienti) e, infine, a migliorare il proprio prodotto anche seguendo le preferenze dei suddetti clienti.

L'NLP ha origine intorno al 1950 quando, all'IBM-Georgetown Demonstration, fu mostrata per la prima volta una macchina rudimentale per la traduzione automatica dal Russo all'Inglese e, in seguito a questo evento, iniziarono anche le prime ricerche nell'ambito pubblicate per la maggior parte sul giornale Mechanical Translation. Già nel 1961, però, molti avevano colto i possibili sviluppi che potevano venir fuori dallo studio della materia erano innumerevoli. Alla "Teddington International Conference on Machine Translation of Languages and Applied Language Analysis", infatti, fu mostrato che in poco più di 5 anni erano già stati portati avanti moltissime ricerche e lavori sulla morfologia, sulla sintassi, sulla semantica e, ultimo ma non per importanza, sul trasferimento dalla teoria agli hardware, portando a fare passi enormi anche sul lato informatico.

Il primo periodo fu particolarmente florido per la ricerca anche se complicato: si stava, per la prima volta, cercando di far interpretare dati non strutturati a macchine che, in quel periodo storico, erano particolarmente limitate dal punto di vista della potenza di calcolo, si capisce dunque quanto lavoro ci dovesse essere sia dal punto di vista teorico ma anche, e soprattutto, dal punto di vista dell'ottimizzazione del lavoro della macchina.

La ricerca si allargò fin dai primi anni '60 a tutto il mondo, dall'Unione Sovietica agli USA, passando per Giappone e Europa. La maggior parte della ricerca aveva come topic principale la sintassi che doveva essere, secondo la maggior parte dei ricercatori, il driver principale degli studi e degli sviluppi. Una delle prime ricerche, influenzate dall'intelligenza artificiale, fu pubblicata nel 1961 in cui gli autori, Green e Wolf, riuscirono a programmare un sistema di domande e risposte sul baseball sfruttando i dati immagazzinati in locale. Furono pubblicati seguentemente dei successori al programma di domande e risposte, tra cui LUNAR, di Woods et al., o SHRLDU, di Winograd (1973).

Nel corso dell'ARPA Speech Understanding Research, nel 1980, fu consolidato l'interesse verso questo campo. Si iniziarono ad intravedere i primi grandi ed autonomi databases che riuscivano a dare un apporto particolarmente ampio alla ricerca e allo sviluppo di nuovi programmi informatici sull'NLP.
Negli anni '80, invece, ci si approcciò ad un nuovo metodo di studio, quello logico-grammatica dei testi, incoraggiato anche dal fatto che gli studiosi nel campo stavano diventando sempre di più e c'erano molte più ricerche pratiche basate anche sulle query ai database[4].
Al giorno d'oggi il Natural Language Processing è un ambito particolarmente interessante grazie a moltissimi lavori di ricerca che hanno portato sviluppi incredibili per la comprensione, da parte delle macchine, del linguaggio umano, e grazie anche allo sviluppo impressionante che si è ottenuto negli ultimi 30 anni nell'ambito dell'Information Technology, permettendo di sfociare anche, ma non solo, nella Sentiment Analysis, ambito in cui questo lavoro si sviluppa.

1.3 La Sentiment Analysis

La Sentiment Analysis è una delle aree di studio e ricerca più in crescita del nostro periodo storico. Nata praticamente nello stesso periodo, viene affiancata spesso al Natural Language Processing in quanto in quest'ultima si riesce a passare da dati non strutturati a dati strutturati e comprensibili per le macchine e, nella Sentiment Analysis, in seguito, vengono utilizzate tecniche di Machine Learning e di Deep Learning per l'analisi del linguaggio volte a comprendere il sentiment di quella data frase, di quel dato articolo o di quel testo, più in generale.
Nella seguente figura sarà possibile visualizzare l'andamento della serie storica delle ricerche effettuate su Google.com riguardanti la Sentiment Analysis negli ultimi 18 anni circa. È possibile notare come la tendenza delle ricerche è in crescita non lineare ma esponenziale, sintomo di una probabile crescita dell'interesse anche nel futuro prossimo in cui la ricerca potrebbe condurre ad altri metodi di analisi del sentiment, capaci, ad esempio, di ottimizzare ancor più le analisi odierne che riescono a prevedere, in media, tra l'80% e il 90% dei sentimenti correttamente. Per leggere al meglio il grafico basti sapere che il valore sulle ordinate pari a 100 indica il picco di interesse riguardo ad un dato argomento con una maggior frequenza di ricerca fino a scendere a 0 in cui non vi sono ricerche o non vi sono

[4] Karen Spärck Jones, 2001, Natural Language Processing: A Historical Review

dati sufficienti per una stima del parametro.

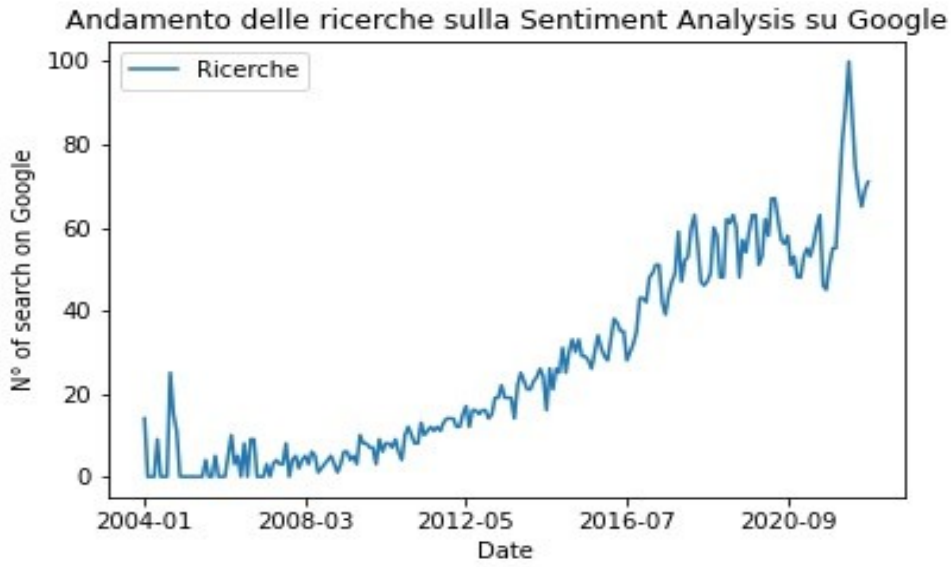


Figura 3: Serie storica delle ricerche su Google riguardanti la Sentiment Analysis

A riprova di ciò si può anche osservare la Figura 4 che ci mostra la serie storica del numero di pubblicazioni effettuate nello stesso periodo.

Figura 4: Serie storica pubblicazioni

È possibile notare come anche le ricerche svolte sul tema dell'analisi dei sentimenti siano

notevolmente cresciute in quantità, passando dall'essere poco meno di 20'000 nel 2004 alle circa 100'000 nel 2019[5].

Come detto precedentemente, la Sentiment Analysis si pone l'obiettivo di riuscire ad etichettare come Positivo, Negativo o Neutrale un determinato contenuto testologico. Dal momento che, però, l'analisi svolta in questo lavoro viene effettuata su dati rilevati dal Social Network "Twitter", viene aggiunto un grado di difficoltà ulteriore in quanto il linguaggio utilizzato non è classico e formale ma molto più informale e ricco di rumore, termini colloquiali e slang di settore che dovranno essere interpretati nel migliore dei modi dal modello prescelto.

Il primo obiettivo sarà dunque quello di andare a suddividere i tweets in Oggettivi e Soggettivi, per poi prevedere, nel caso di questi ultimi, la corretta etichetta da assegnarvi. Il modello utilizzato per l'analisi del sentiment, chiamato TextBlob, avrà come output proprio questi due elementi: Polarity del contenuto, compresa tra -1 e 1, che indicherà quanto quel tweet è positivo (+1), negativo (-1) o neutrale (0), per poi assegnare un secondo score che sarà chiamato "Subjectivity", compreso tra 0 e 1, che indicherà se quel contenuto è particolarmente soggettivo (0) o particolarmente basato su fatti e di conseguenza oggettivo (1).

Un'ulteriore difficoltà ritrovabile nell'analisi è quella che alla base dell'analisi del sentiment vi è un'assunzione molto forte: viene assunto, cioè, che i testi forniti al modello, siano indipendenti ed identicamente distribuiti e che quindi la rete di amicizie collegate all'interno del contesto social non abbiano alcun impatto sui testi pubblicati di un dato autore. Questa assunzione riduce la complessità delle analisi ma si allontana da ciò che è la realtà. Sono stati fatti però passi da gigante anche su questo fronte nella ricerca che sta cercando di portare avanti modelli di Reti Neurali in grado di gestire anche le possibili correlazioni che intercorrono tra gli utenti e le proprie opinioni (e.g. RoBERTa sviluppato da Meta).

Come accennato nell'introduzione, le applicazioni della Sentiment Analysis spaziano in innumerevoli campi ma, allo stato dell'arte, l'applicazione migliore la si trova nel campo del marketing e del miglioramento dei processi produttivi aziendali. Tramite le opinioni rilevate, infatti, le aziende riescono a comprendere al meglio i motivi che spingono gli utenti ad acquistare o a non acquistare i propri prodotti con la conseguenza di riuscire ad intercettare una fetta maggiore di pubblico nel momento in cui si riescono a migliorare i difetti e ad esaltare i pregi dei servizi offerti. Altre applicazioni, per ora di minor rilievo ma

[5] www.trends.google.it

comunque in crescita, sono ad esempio presenti nell'ambito politico, in cui la comunicazione riesce sempre più a capire quali sono le questioni che importano alla maggioranza degli elettori per essere poi pronti, in campagna elettorale, a sfruttare quella conoscenza per il convincimento degli stessi, sono presenti applicazioni anche in ambiti di sistemi di raccomandazione di contenuti, basti pensare alle piattaforme di streaming online che, grazie ai dati acquistati o rilevati dai social, riescono a generare un profilo ad hoc per il consumatore che si vedrà consigliare contenuti più adatti ai suoi gusti personali.

1.4 Il Machine Learning

La storia del Machine Learning è altrettanto importante ed entusiasmante in quanto nacque con l'intento di imitare i processi cognitivi degli esseri umani. Furono il Logico Walter Pitts e il neuro-scienziato Warren McCulloch a pubblicare i primi studi che miravano a imitare la mente umana attraverso il primo modello matematico di una rete neurale che viene posta alla base della capacità umana di imparare dai propri errori, di prendere decisioni basate sulle informazioni possedute al momento della scelta e di prevedere le possibili conseguenze, e le entità delle stesse, che qualunque scelta possibile potrebbero avere su una data situazione. Un secondo momento importantissimo per la storia del machine learning fu generato da Alan Turing, matematico divenuto famoso per la decifrazione, durante il periodo della Seconda Guerra Mondiale, della macchina tedesca "Enigma" e dei messaggi crittografati inviati tramite la stessa. Turing, dopo la guerra, regalò però quella che potrebbe essere considerata la prima definizione di Intelligenza Artificiale: il Turing Test[6]. Quest'ultimo affermava che una macchina, per poter essere considerata intelligente, doveva riuscire ad ingannare un umano in una conversazione anonima di essere un umano a sua volta, solo allora ci si poteva arrogare il diritto di poter definire suddetta macchina "intelligente". Fu però nel 1952, con Arthur Samuel, che si coniò per la prima volta il termine "Machine Learning", quando il computer scientist riuscì a inventare un gioco di dama per computer che aveva però una particolarità: più il programma giocava più era difficile contro di essa grazie a un algoritmo minimax per lo studio delle mosse. Pietre Miliari da qui in avanti sono praticamente infinite ma le più simboliche avvennero poi nel 1997, quando un "super computer" della multinazionale IBM riuscì a battere a scacchi il campione Garry Kasparov, e nel 2014, quando il chatbot russo "Eugene Goostman" riuscì a convincere il 33% di una giuria di essere umano, passando così il Turing Test, avverando

[6] Koch, History of Machine Learning – A Journey through the Timeline

così la previsione del matematico che ipotizzò che una macchina avrebbe superato l'omonimo test agli inizi degli anni 2000.

All'interno di questo lavoro verranno utilizzati vari modelli statistici che rientrano sotto la branca del supervised Machine Learning, tra questi ritroveremo la Random Forest, il Naive Bayes, la Regressione Logistica e l'Analisi Discriminante Lineare (LDA) che verranno approfonditi nel prossimo capitolo.

Anche in questo caso, come precedentemente svolto per gli altri paragrafi, risulta utile riportare per completezza un ulteriore visualizzazione grafica generata, grazie al linguaggio di programmazione Python, dai dati ricavati da Google in merito alle ricerche svolte dagli utenti nell'arco temporale che va dal 2004 ad oggi.

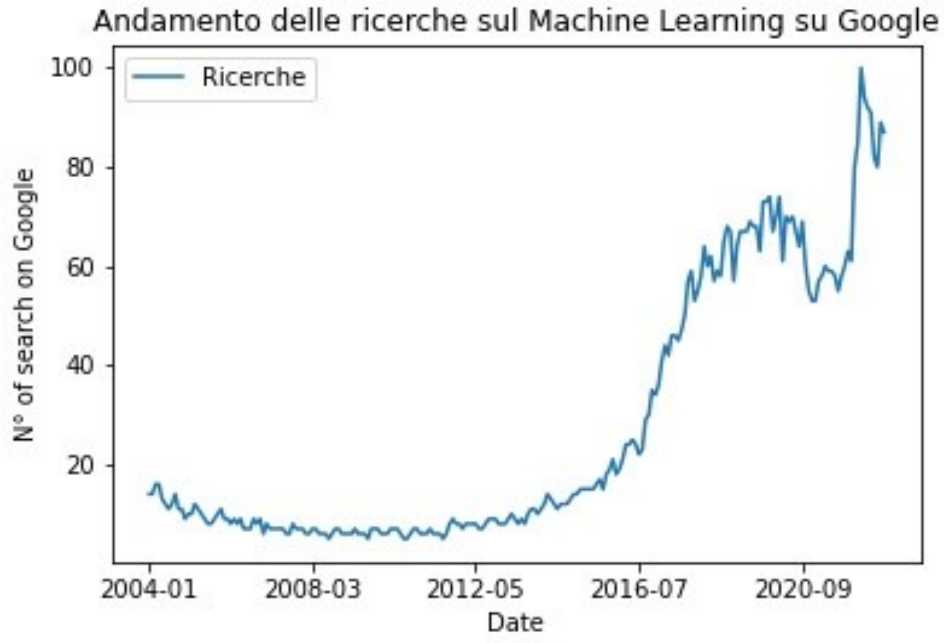


Figura 5: Serie storica delle ricerche sul Machine Learning

Anche questa volta, come nelle precedenti, è possibile osservare quanto il motore di ricerca più famoso ed utilizzato al mondo abbia registrato una crescita esponenziale nell'interesse degli utenti, stimando un picco dell'interesse intorno agli inizi del 2021, spinto probabilmente anche dalla pandemia da Coronavirus che ha accelerato tutti i processi informatici in atto in quel periodo.

CAPITOLO DUE

STRUMENTI E TECNICHE PER L'ANALISI DEI DATI

SOMMARIO: 2.1 Il trattamento dei dati non strutturati; 2.2 TextBlob; 2.3 I principali modelli di Machine Learning; 2.3.1 La Regressione Logistica; 2.3.2 Il Decision Tree e la Random Forest; 2.3.3 Il Naive Bayes; 2.3.4 Support Vector Machine; 2.3.5 La Linear Discriminant Analysis (LDA); 2.4 Metodi per la misurazione dell'abilità predittiva; 2.4.1 La Confusion Matrix; 2.4.2 Sensitività e Specificità; 2.4.3 2.4.3 La ROC Curve e l'AUC.

2.1 Il trattamento dei dati non strutturati

Il processo analitico portato avanti in questo lavoro parte dal trattamento dei dati testuali, che rappresentano uno dei casi principali di dati non strutturati. Questa tipologia di dati sono informazioni che sono generate in modo tale da non poter essere immagazzinate nei classici database relazionali e che non seguono le regole dei modelli di dati tradizionali. Altri esempi di unstructured data possono essere le immagini o i video che, come si può immaginare, sono completamente diversi dai database composti da set di dati quantitativi o qualitativi immagazzinati in tabelle con un immediato utilizzo analitico qualora fosse necessario. Con lo sviluppo del Machine Learning, dell'Intelligenza Artificiale e, in questo caso, del Natural Language Processing, vi sono però svariate tecniche per rendere usufruibili i dati non strutturati come possono essere i tweets raccolti per questo lavoro.

Il text pre-processing si pone l'obiettivo di andare a convertire un corpus di testo grezzo in una sequenza ben definita di unità pronte all'uso. Bisogna quindi identificare, all'interno del testo, quelle che sono le lettere, il livello più basico dei linguaggi scritti, le parole, formate da un insieme di lettere, e le frasi, insiemi di parole che andranno a formare il significato intrinseco.

La prima parte del processo testuale parte con lo studio del documento che va a convertire un documento digitale in qualcosa che possa essere interpretato dalla macchina utilizzata per la sua lettura, questo processo si sviluppa attraverso la codifica dei caratteri, che rende leggibile il testo alla macchina, l'identificazione del linguaggio naturale utilizzato e il sezionamento del testo che va a dividere il testo, ad esempio, in titoli, links o tabelle.

Si passa poi alla fase della segmentazione del testo in cui il testo viene suddiviso in tutti gli elementi che lo compongono fino ad arrivare alle parole utilizzate che verranno a loro volta scomposte, identificando i loro confini, in una sequenza di caratteri.

Nell'ambito del NLP, ogni parola sarà un token e la scomposizione in tokens del testo sarà una delle parti principali di ogni lavoro chiamato tokenization. Questa parte di lavoro sembrerebbe particolarmente semplice e diretta in quanto basterebbe identificare lo spazio tra una parola e un'altra ma i linguaggi naturali sono completamente diversi gli uni dagli altri, specialmente quando si passa da un continente all'altro. Nel caso delle lingue europee, ad esempio, per la maggior parte dei casi sarà sufficiente andare a ricercare lo spazio vuoto tra una parola ed un'altra ma, ad esempio, ciò non può accadere nei linguaggi non segmentati in cui non vi sono confini ben delimitati, un esempio diffusissimo è la lingua cinese in cui si vanno a formare frasi intere senza andare a delimitare i tokens. In questo lavoro si farà uso delle tecniche di tokenization per le lingue europee in quanto i tweets sono

stati raccolti in lingua inglese che, quindi, hanno al loro interno parole ben delimitate ma che portano con esse difficoltà legate al significato delle stesse in quanto, prendendo ad esempio proprio l'inglese, ogni parola può assumere diversi significati che possono variare a seconda del contesto utilizzato o del posto che occupa all'interno della frase. Bisogna, inoltre, far attenzione all'uso della punteggiatura e delle maiuscole: un esempio del primo possono essere le parole che nell'uso quotidiano contengono delle punteggiature che ne formano il significato, basti pensare ai nomi delle strade, delle piazze, delle città o anche dei fiumi, spesso indicate con punteggiature al loro interno come "Via S.Francesco" o, per riportarne un caso in inglese, "St. Johns river", nel secondo caso, invece, le maiuscole potrebbero portare alla formazione, nel nostro dataset, di due tokens distinti quando invece possono essere raccolti sotto lo stesso, un esempio può essere un nome proprio di persona che, soprattutto sui social media in cui si scrive in modo formale, rapido e talvolta con errori grammaticali, potrebbero essere ritrovati sia con l'iniziale maiuscola sia minuscola.

Vi è un'ulteriore fase in cui bisogna effettuare una segmentazione delle frasi, andando a riconoscere quando una frase termina per dividerla rispetto alle altre. In questo lavoro questa fase è ridotta ai minimi termini in quanto, per la maggior parte dei casi, i tweets sono segmenti brevi e formati da una sola frase in cui viene espresso tutto il significato dato anche il limite numerico di caratteri imposto dal social.

Ogni parola però, può avere significati completamente diversi l'uno dall'altro, spesso questo può dipendere dal contesto in cui quella data parola viene utilizzata, basti pensare, in lingua inglese alla parola "left" che può assumere sia la traduzione di "lasciato" sia quella di "sinistra", si riportano di seguito due esempi immediati: "I left my apartment"; "I went to the left". Gli sviluppi delle Neural Networks hanno portato a dei miglioramenti impressionanti nell'ambito del riconoscimento del contesto, specialmente con l'avvento delle reti neurali BERT, appartenenti alla categoria dei transformers, che riescono a riconoscere, nella maggior parte dei casi, anche il contesto in cui si trova ogni singola parola dato che il passaggio nella rete neurale avviene simultaneamente per ogni token all'interno della frase e non più uno per volta come avveniva in passato coi modelli LSTM (Long Short-Term Memory). (Handbook of NLP)

Un ulteriore passaggio da effettuare, oltre alla rimozione della punteggiatura e delle maiuscole, è la text normalization. In questa fase del lavoro vengono viene data importanza alla potenza di calcolo delle macchine utilizzate secondo il principio della parsimonia che renderà l'analisi più efficiente sia dal punto di vista della potenza utilizzata sia dal punto di vista della velocità con cui si riuscirà ad ottenere il risultato desiderato. Nella text

normalization, come dice il termine stesso, vengono applicate delle tecniche per andare a standardizzare il testo in modo tale da andare a ridurre il numero di tokens che verranno processati successivamente[7].

Una prima fase della text normalization è la rimozione delle c.d. stopwords, queste sono delle parole che non aumentano il significato di una frase ai fini dell'analisi ma sono presenti in maniera massiccia in quanto grammaticamente consentono di andare a formare quella che poi è la frase corretta: alcune delle stopwords possono essere identificate negli articoli come "the", nelle congiunzioni come "and".

Si passa poi alla Part-of-Speech (POS) tagging fase, in cui ogni parola viene categorizzata, a seconda del linguaggio utilizzato, in, ad esempio, aggettivi, avverbi, nomi, numeri e così via. Questa fase va a imporre quello che è il giusto ordine delle parole all'interno delle parole andando quindi a implementare una piccola parte, ma pur sempre rilevante, del contesto che la frase originale aveva.

Seguendo sempre il principio della parsimonia nominato precedentemente, si passa alle tecniche di Stemming e Lemmatization. Con queste tecniche si va a ridurre la dimensione del vocabolario utilizzato per la comprensione del testo. Molte parole del nostro linguaggio hanno declinazioni diverse a seconda del caso in cui ci troviamo, nella lingua inglese, ad esempio, basti pensare alla parola depend: questa può avere più declinazioni come depends, depending o dependent che, nell'analisi risulterebbero in quattro tokens separati tra loro e trattati come non appartenenti alla stessa famiglia. Attraverso lo Stemming e la Lemmatization, tutte le parole vengono processate e portate, dove possibile, a formare un unico token che comprende al suo interno tutte quelle che possono essere le declinazioni di quella particolare parola. Più in particolare, lo Stemming, tecnica meno sofisticata e di conseguenza più veloce ma meno efficiente, va a troncare le parole di netto andando ad eliminare la desinenza delle parole lasciando solo le radici delle stesse in un modo più rudimentale della Lemmatization e, cioè, eliminando ciò che considera essere le parti finali di una parola, utilizzando, spesso, quello che è l'algoritmo più utilizzato per questa tecnica: il Porter Stemmer ideato dall'omonimo autore nel 1980, che segue delle regole basilari come, ad esempio:

- SSES → SS; Es.: Chesses → Chess
- IES → S; Es.: Allergies → Allergi
- S → /; Es.: Dogs → Dog

[7] Jurafsky, Martin, Speech and Language Processing

Una tecnica più sofisticata e precisa ma anche più lenta è quella della Lemmatization. Le due tecniche hanno la stessa finalità ma, a differenza dello Stemming, la tecnica in questione va ad identificare con precisione il lemma di ogni parola effettuando una vera e propria analisi morfologica di ogni singolo token componente la frase utilizzando di volta in volta dizionari incorporati all'interno delle librerie che consentono la Lemmatization.
Viene riportato di seguito un esempio lampante eseguito attraverso il linguaggio di programmazione Python che utilizza la libreria "nltk" per eseguire entrambe le tecnche spiegate precedentemente:

```
import nltk
from nltk.stem import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer
```

```
text = ['cries', 'cry', 'crying']
```

```
stemmer = PorterStemmer()
text_stemmed = [stemmer.stem(y) for y in text]
print(text_stemmed)
```

```
['cri', 'cri', 'cri']
```

```
lem = WordNetLemmatizer()
text_lem = [lem.lemmatize(y) for y in text]
print(text_lem)
```

```
['cry', 'cry', 'cry']
```

È possibile notare da questo esempio come le due tecniche raggiungono lo stesso scopo, quello di portare più parole ad un solo token ma, nel caso dello stemming non viene correttamente individuato il verbo "to cry" indicandolo con la lettera i finale, mentre, nel caso della Lemmatization, grazie ai dizionari integrati, viene riconosciuto correttamente la forma verbale riportata nell'output.
Per l'esecuzione di ogni passaggio spiegato precedentemente è stata creata un'apposita funzione, "clean_string", che riceve in input il testo grezzo e la tecnica preferita per il "cut" della parola, e restituisce come output il testo pulito e pronto per le analisi, questa funzione verrà poi applicata successivamente ad ogni tweet in modo da andare a formare una nuova colonna del dataset che sarà quella sottoposta all'analisi del sentiment:

```
nlp = spacy.load('en_core_web_sm')

def clean_string(text, stem="None"):
    final_string = ""
    text = text.lower() #Riportare tutto il testo con lettere minuscole
    text = re.sub(r'\n', '', text) #Rimuovere Escape Sequences
    text = text.split()
    stop = nltk.corpus.stopwords.words("english")
    stop = useless_words + ['hi', 'im']
    text_filtered = [word for word in text if not word in stop] #Rimuovere stopwords
    text_filtered = [re.sub(r'\w*\d\w*', '', w) for w in text_filtered]

    if stem == 'Stem': #Stemming
        stemmer = PorterStemmer()
        text_stemmed = [stemmer.stem(y) for y in text_filtered]
    elif stem == 'Lem': #Lemmatizing
        lem = WordNetLemmatizer()
        text_stemmed = [lem.lemmatize(y) for y in text_filtered]
    else:
        text_stemmed = text_filtered

    final_string = ' '.join(text_stemmed)

    return final_string
```

2.2 TextBlob

TextBlob è la libreria che verrà utilizzata successivamente in Python per l'analisi del sentiment. Questa può, in realtà, essere utilizzata però anche per altri scopi come, ad esempio, la Part-of-Speech Tagging, la classificazione o la traduzione.

Il metodo "sentiment" degli oggetti creati attraverso la funzione omonima della libreria, restituirà come output due tuple che saranno, rispettivamente, un float compreso tra -1 e 1, la polarity, ed un float compreso tra 0 e 1, la subjectivity. L'algoritmo decisionale implementato per arrivare a questi due valori è detto lexicon-based, andando ad attribuire ad ogni parola un valore di sentiment e uno di subjectivity per poi arrivare ad un punteggio totale per ogni frase attraverso la media dei punteggi ottenuti. L'algoritmo riesce a riconoscere anche le negazioni andando, quando rilevate, ad applicare un coefficiente di -0.5 allo score risultante dall'analisi. L'algoritmo è di tipo pre-trained, ciò in quanto è stato addestrato su un database di recensioni in modo tale da riuscire a riconoscere ad ogni parola uno score, ha anche un dizionario al suo interno e, infatti, viene categorizzato come Lexicon-Based Model in quanto sfrutta degli score prestabiliti per la costruzione di un punteggio derivante da quelli singoli. TextBlob è uno degli strumenti più semplici da utilizzare riscontrando ottimi risultati anche dal punto di vista della velocità di calcolo ma viene superato quando si va a trattare Reti Neurali con più layers che si traducono in una maggior precisione dal punto di vista del sentiment ma che, di contro, hanno lo svantaggio

di essere particolarmente computer-intensive e lenti. In questo lavoro viene utilizzato TextBlob proprio per la sua semplicità di applicazione e la richiesta inferiore di risorse necessarie allo svolgimento dei calcoli, si parlerà della scelta dell'algoritmo per l'analisi del sentiment più approfonditamente nel capitolo riservato alle conclusioni e agli sviluppi futuri del lavoro[8].

2.3 I principali modelli di Machine Learning

In questo paragrafo sarà effettuata una panoramica dei principali modelli di Machine Learning con binary outcome utilizzati per lo svolgimento dell'analisi successiva in modo tale da renderne più immediato l'utilizzo e la comprensione.

2.3.1 La Regressione Logistica

Quando si parla di binary outcome si ha a che fare con modelli che invece di restituire outcome continui, come ad esempio la regressione lineare semplice, restituiscono output che hanno solo due valori possibili: 0 e 1[9].
Il più semplice modello con questa caratteristica è la Regressione Logistica. La differenza sostanziale è che qui si farà riferimento alle probabilità:

$$P\left(y_i = 1 \mid x_i^T\right) = F(x, \beta)$$

L'obiettivo, quindi, è quello di andare a scegliere una funzione F tale da derivare la probabilità che y sia uguale a 1 dati i predittori importati come input. Nella regressione logistica, la F() sarà la standard logical distribution (LOGIT) in cui:

$$F(w) = L(w) = \frac{e^w}{1 + e^w}$$

Si va quindi ad osservare una risposta y^* non osservata, latente, che riesce a sfuggire ai limiti imposti dalla variabile y osservata che può risultare solo in uno 0 o in un 1. L'equazione della regressione logistica sarà data da:

$$y_i^* = x_i^T \beta + \varepsilon_i$$

E l'output prodotto sarà sottoposto al paragone con un treshold che potrà essere, ad esempio lo 0, in modo tale da assegnare un valore di 0 o 1 alla y osservata a seconda di dove si trova la nostra variabile latente rispetto al treshold scelto quindi:

[8] www.github.com/sloria/TextBlob
[9] Greene, 2012, Econometric Analysis.

$$P(y_i = 1) = P(y_i^* > 0) = P\left(x_i^{\top}\beta + \varepsilon_i > 0\right) = P(-\varepsilon_i < 0) = F(x, \beta)$$

Per applicare il modello ai nostri dati, quindi, si deve arrivare ad una forma di questo tipo:

$$p_i = x_i^T \beta$$

Il problema principale in questa equazione è che la parte a sinistra, la probabilità, potrà assumere solo valori compresi tra 0 e 1 mentre quella a destra potrà assumere qualunque valore. Per rimuovere il vincolo sulla probabilità sarà dunque necessario applicare una trasformazione che avviene in due fasi:

- Si trasforma la probabilità in odds che è il rapporto tra la probabilità che l'evento accada rispetto alla likelihood che lo stesso non si verifichi:

$$odds = \frac{p_i}{1 - p_i} = \frac{P(y = 1)}{1 - P(y = 1)} = \frac{P(y = 1)}{P(y = 0)}$$

- Applicando poi l'operatore logaritmo avremo rimosso il vincolo in quanto il logit potrà assumere qualunque valore reale:

$$logit = log\frac{p_i}{1 - p_i}$$

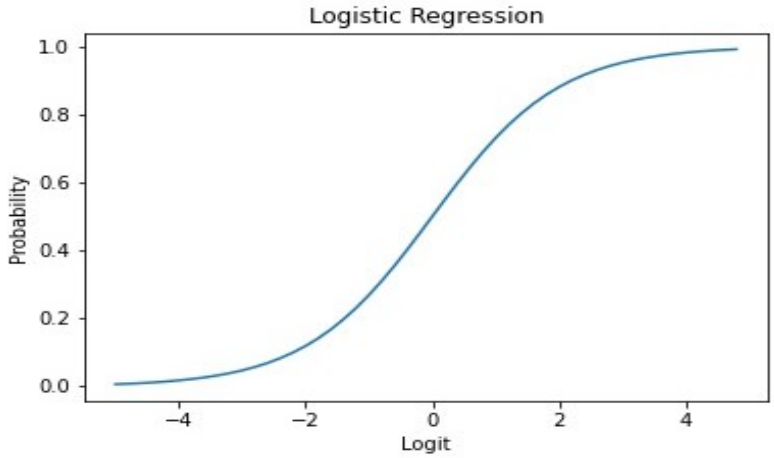


Figura 6: Rappresentazione Logit vs Probability

Il modello logistico, infine, avrà dunque la seguente equazione:

$$log\frac{p_i}{1 - p_i} = x_i^T \beta$$

Utilizzando il metodo della Massima Verosimiglianza si stimeranno i parametri e, una volta dato beta, si riuscirà a calcolare la probabilità dell'evento successo che, per questo modello, dopo una serie di passaggi algebrici, sarà:

$$p_i = \frac{e^{x_i^T \beta}}{1 - e^{x_i^T \beta}}$$

2.3.2 Il Decision Tree e la Random Forest

Il secondo modello preso in considerazione per l'analisi è la Random Forest. Risulta necessaria però una introduzione dell'albero di decisione in quanto parte fondamentale del modello basasto sulla Random Forest.

L'albero di decisione è un modello di classificazione, quindi, come nella regressione logistica, si effettua una suddivisione dei dati in categorie non continue, in questo caso di studio con output binario. Per la costruzione di un albero decisionale, viene suddiviso il dataset in sottoinsiemi omogenei rispetto all'output. La divisione, effettuata seguendo l'algoritmo ID3, avviene calcolando ad ogni passaggio l'entropia e il guadagno d'informazione per poi scegliere l'attributo che ha una entropia più bassa o il più alto guadagno d'informazione. L'entropia viene definita come la misura di casualità contenuta nell'informazione processata mentre il guadagno d'informazione è una proprietà statistica che misura quanto l'attributo selezionato riesca a separare in modo ottimale il subset. Viene a crearsi poi un albero in cui i nodi hanno delle determinate condizioni che decideranno dove trasportare, attraverso i rami, l'input per assegnare, infine, attraverso le foglie finali, il valore stimato che potrà essere 0 o 1.

Nella figura seguente, attraverso l'utilizzo del dataset "Iris", è stato creato un esempio di albero decisionale grezzo in modo da riportare il modello con al suo interno i nodi di decisione stimati e le foglie finali a cui poi spetta la scelta dell'output da assegnare ad ogni elemento del test set o degli elementi di cui si vuole stimare la variabile output che, nel caso del dataset didattico preso ad esempio, sarà la categoria di fiore a cui appartiene l'input immesso.

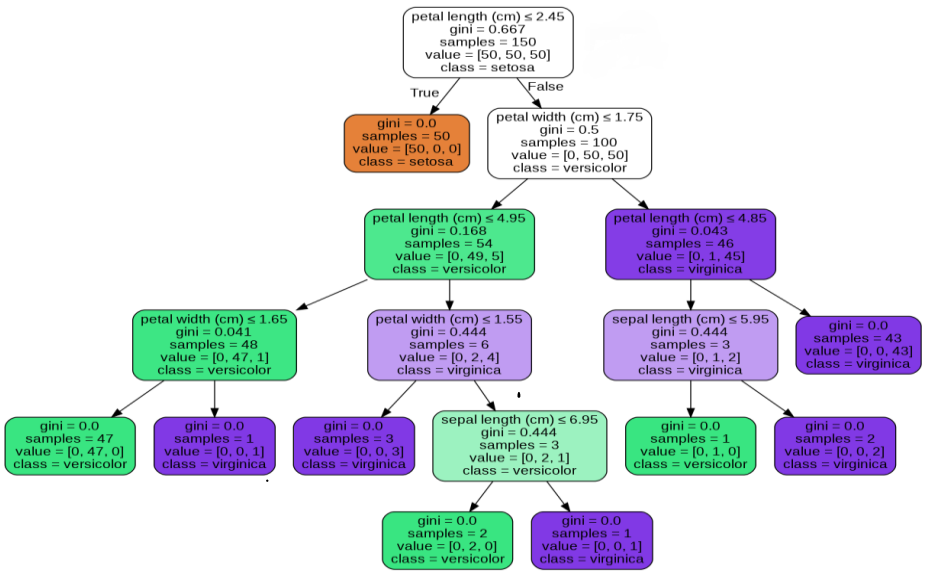


Figura 7: Esempio di Decision Tree per IRIS

Viene riportato, nel grafico, anche l'indice di Gini corrispondente alla probabilità che un elemento scelto casualmente sarebbe incorrettamente etichettato se la scelta della label fosse stata randomica e raggiunge lo 0 quando il modello assegna all'elemento la categoria stimata che, in questo caso, rappresenta la classe di pianta a cui appartiene.
Il modello Random Forest è un metodo "ensemble" in quanto va a effettuare le classificazioni attraverso un meccanismo di voto dato da più modelli di Decision Trees. Il concetto che regola l'output è piuttosto semplice: si va a suddividere il dataset originale in più subset selezionati casualmente, in modo da diminuire la correlazione tra i modelli stimati successivamente; per ogni subset viene poi stimato un albero decisionale attraverso cui poi viene stimata la classe di appartenenza dell'elemento preso in considerazione. Una volta stimati i modelli per ogni subset, il modello sceglie la classe di appartenenza seguendo un meccanismo di voto, la classe di appartenenza che è stata scelta più volte dai singoli modelli risulterà essere la categoria stimata per l'elemento immesso in input. L'albero decisionale era stato superato da modelli più sofisticati e precisi ma, in seguito allo sviluppo della Random Forest, nel 1995, è tornato ad essere uno dei modelli di Supervised Machine Learning più utilizzati nell'ambito della classificazione[10].

2.3.3 Il Naive Bayes

Il Naive Bayes è uno dei modelli più utilizzati quando si tratta di analisi del testo come, ad esempio, nella classificazione delle mail in Spam o Non Spam. Alla base del modello vi è uno dei più teoremi cardine della Statistica Bayesiana: il teorema di Bayes. Questo afferma che la probabilità che l'eventa A accada, dato l'evento B, è uguale al prodotto tra la probabilità che l'evento B accada, dato l'evento A, e la probabilità non condizionata dell'evento A, il tutto rapportato alla probabilità non condizionata dell'evento B, analiticamente:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Basandosi su questo approccio si arriva di conseguenza alla formula analitica del modello:

$$P(y|x_1, x_{2,\dots}x_n) = \frac{P(x_1, x_2, \dots, x_n|y)P(y)}{P(x_1, x_2, \dots, x_n)}$$

Dove il termine a sinistra, è la probabilità che un'osservazione appartenga alla classe y, date

[10] Tin Kam Ho, 1995, Random Decision Forest

tutte le informazioni *x*, chiamata probabilità posteriore; il primo termine del numeratore a destra, invece, rappresenta la probabilità che le osservazioni abbiano tali valori, dato il fatto che appartengono alla classe *y*, chiamata likelihood; mentre il secondo termine al numeratore e quello al denominatore sono rispettivamente la probabilità a priori e la probabilità marginale.

Attraverso questo algoritmo, si va a paragonare ogni singolo risultato per ogni classe possibile in modo di verificare quale di questi abbia la probabilità più elevata di verificarsi, andando quindi ad assegnare la rispettiva classe prevista dalla stima.

L'ostacolo principale è l'assunzione di una distribuzione di probabilità all'elemento likelihood, la più comune, assunta anche successivamente nel corso del lavoro di Tesi, è la distribuzione Normale avendo quindi:

$$P(x_j|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} e^{-\frac{(x_j-\mu)^2}{2\sigma_y^2}}$$

Una seconda assunzione che viene fatta è che le osservazioni siano indipendenti tra loro. Proprio da quest'ultima deriva però il nominativo di Naive (Ingenuo) in quanto essa risulta spesso violata nella realtà dei dati in esame ma viene data come assolta a prescindere dai risultati dei test d'Indipendenza[11].

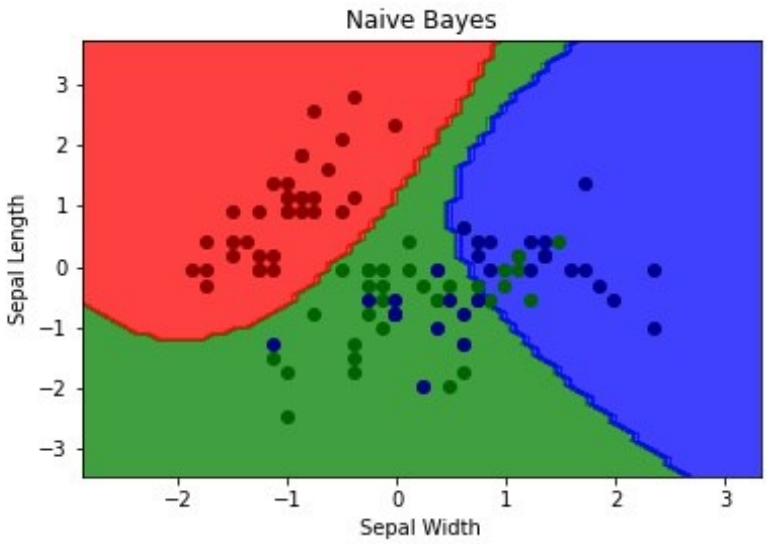


Figura 8: Naive Bayes applicato ad IRIS

[11] Albon, 2018, Machine Learning with Python Cookbook

2.3.4 Support Vector Machine

Le macchine a vettori di supporto sono modelli di classificazione il cui scopo è ricercare la separazione migliore tra le classi e, cioè, di massimizzare il margine tra queste in modo tale che la distanza tra i punti più vicini alla retta, appartenenti a due classi diverse, sia la più ampia possibile. I support vectors sono i valori di ogni categoria che risiedono nei punti più vicini alla retta di separazione. Ciò risulta essere un punto a vantaggio del modello rispetto a quelli studiati finora in quanto, in questo caso, si va a cercare di separare le due classi utilizzando i punti più simili tra loro ma appartenenti a diverse categorie, rendendo dunque più complicato il compito del classificatore ma anche potenzialmente più preciso il risultato della divisione. Nella figura che segue è possibile verificare graficamente ciò che è stato spiegato precedentemente, seguendo una classificazione binaria, rappresentata mediante lo scatter plot delle osservazioni rispetto alle variabili x1 e x2. Le linee tratteggiate sono i vettori di supporto, tracciati in corrispondenza dei punti più vicini alla retta di separazione con equazione $wx - b = 0$

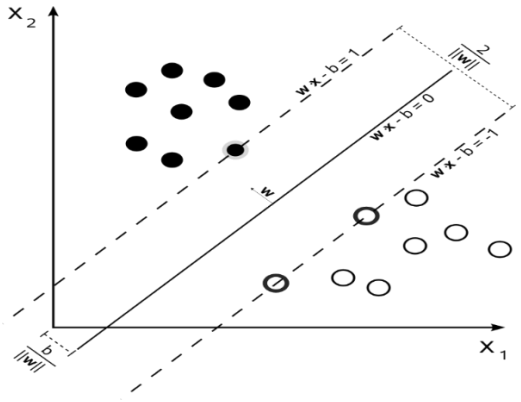


Figura 9: Rappresentazione teorica SVM

Per rendere, anche in questo modello, il tutto più pratico, ho applicato l'algoritmo al dataset didattico "Iris" in modo da poter verificare, nella figura seguente, anche se con due soli parametri per rendere il grafico visualizzabile, l'efficacia prodotta dal SVM.

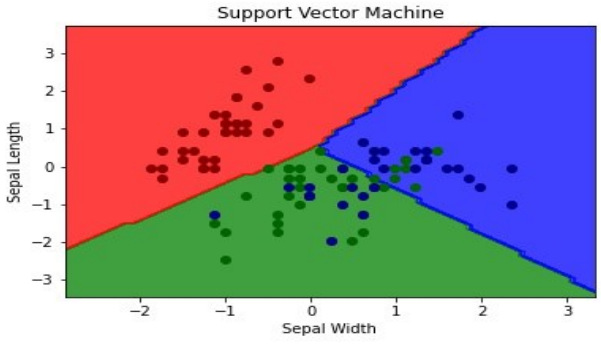


Figura 10: Rappresentazione SVM applicato ad IRIS

È possibile notare come in questo caso, con sole due variabili, si riesca a rappresentare una classificazione senza errori per i punti di colore rosso mentre, per le altre due classi, sono stati commessi alcuni errori che sarebbero minimizzati se si fossero prese tutte le variabili contenute all'interno del dataset.

2.3.5 La Linear Discriminant Analysis (LDA)

La Regressione Logistica ha dei limiti come, ad esempio, l'instabilità delle previsioni quando le classi sono ben separate tra loro o quando ci sono poche osservazioni per la stima dei parametri. La Linear Discriminant Analysis mira a risolvere questi limiti. Lo scopo del modello è quello di andare a massimizzare la distanza tra due o più categorie di dati etichettati attraverso il vettore delle medie e la matrice varianza/covarianza degli stessi. Una volta stimati questi valori vengono poi immessi come input all'interno del modello che va a stimare una classe di appartenenza per i nuovi dati. Le medie delle classi e le proprie varianze vengono calcolate normalmente ma assumendo, per semplificare il modello, che le varianze siano uguali per ogni classe, seguendo quindi le formule:

$$\mu_k = \frac{1}{n_k}\sum_{i=1}^{n_k} x_i \qquad \sigma^2 = \frac{1}{n-k}\sum_{i\equiv 1}^{n}(x_i - \mu)^2$$

L'algoritmo quindi stima, per la previsione della classe del nuovo set di dati immessi, la probabilità di appartenenza rispetto a ciascuna categoria all'interno del dataset, andando poi ad assegnare questo nuovo set alla categoria che risulta avere la probabilità più alta secondo le stime. Questa probabilità avviene, come nel Naive Bayes, attraverso l'applicazione del Teorema di Bayes assumendo la normalità dei dati[12]. Grazie a ciò, partendo dalla funzione di densità della variabile casuale normale, è possibile arrivare a quella che viene chiamata linear discriminant function $\delta_k$che analiticamente sarà data da:

$$\delta_k(x) = \log \pi_k - \frac{1}{2}\mu_k^T \Sigma^{-1}\mu_k + x^T \Sigma^{-1}\mu_k$$

Applicando questa funzione a qualunque nuovo set di dati, che siano dati dal test set o dati out of sample, si potrà verificare quale classe *k* ha un valore maggiore, quest'ultima sarà di conseguenza la categoria assegnata alla nuova osservazione.

Anche per questo algoritmo viene riportata la sua applicazione e rappresentazione grafica sul dataset "Iris" dove può emergere la linearità della divisione effettuata e anche la differenza con i modelli analizzati precedentemente rispetto alle stime dei separatori che li

[12] www.machinelearningmastery.com/linear-discriminant-analysis-for-machine-learning/

contraddistinguono.

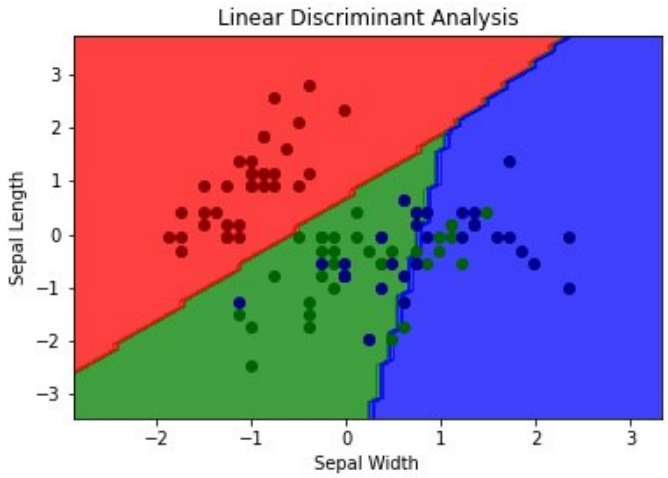


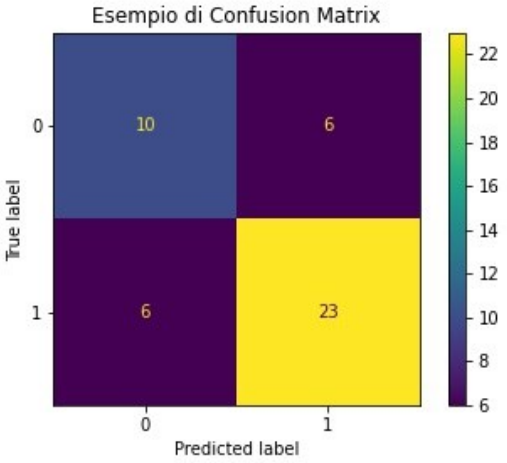
Figura 11: Linear Discriminant Analysis applicata ad IRIS

2.4 Metodi per la misurazione dell'abilità predittiva

La scelta del modello ottimale con cui portare a termine l'analisi è una delle fasi principali di ogni lavoro che abbia a che fare con le previsioni di un dato fenomeno. Per effettuare tale scelta è necessario mettere in pratica delle tecniche oggettive e quantificabili per ottenere la miglior goodness-of-fit misurata. Quest'ultima può essere misurata sia basandosi su alcune metriche appartenenti al modello internamente come, ad esempio, lo Pseudo-Rsquared o l'indice di McFadden; in questo lavoro, è stato scelto però un approccio dal lato della bontà previsionale dei modelli. Il miglior algoritmo, quindi, sarà quello che riuscirà ad ottenere un punteggio migliore nell'ambito della precisione delle previsioni stimate, che saranno valutate con alcune metriche di seguito descritte.

2.4.1 La Confusion Matrix

Il punto di partenza della valutazione dei risultati previsionali di ogni modello è la Matrice di Confusione. Nell'ambito della classificazione, questa matrice riesce ad individuare le prime metriche necessarie per la costruzione di quelle successive. Ciascun elemento della matrice sarà fondamentale in quanto va a rappresentare in formato tabulare quelli che sono: i True Positive (TP), i False Positive (FP), i False negative (FN) i True negative.
Nella figura seguente viene riportato un esempio chiarificatore per l'esplicazione dei segmenti che compongono la matrice:



- Nel primo quadrante, in alto a sinistra, vengono visualizzati i True Negative, in questo caso, dunque, il modello stimato ha previsto correttamente che 10 osservazioni andavano etichettate con uno 0 (Negativo).
- Nel secondo quadrante, in alto a destra, vengono visualizzati i False Positive, indicando che 6 osservazioni avessero come categoria di appartenenza un 1 quando invece nella realtà erano appartenenti ai negativi.
- Nel terzo quadrante, in basso a sinistra, è possibile osservare i False Negative, osservazioni appartenenti alla categoria 1 ma stimati essere appartenenti alla categoria 0.
- Nel quarto ed ultimo quadrante vengono visualizzati i True Positive: il modello qui ha previsto correttamente l'appartenenza alla categoria dei positivi di 23 osservazioni del test set.

2.4.2 Sensitività e Specificità

Le prime metriche calcolabili a partire dai risultati visualizzati nella Confusion Matrix precedente sono la Sensitività e la Specificità. La prima di queste, detta anche True Positive Rate, risulta essere la proporzione tra i positivi correttamente identificati e il numero totale di positivi. La seconda, invece, detta True Negative Rate, sarà la proporzione tra il numero dei True Negative rispetto al totale dei Negativi.

$$Sensitività: TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$$

$$Specificità: TNR = \frac{TN}{N} = \frac{TN}{TN + FP}$$

Nel caso della matrice di confusione riportata precedentemente, avremo una sensitività di: $\frac{23}{23+6} = \frac{23}{29} = 0{,}79$. Per la specificità, invece: $\frac{10}{10+6} = \frac{10}{16} = 0{,}62$.

La funzione utilizzata in Python per la creazione del report di classificazione seguentemente al fit del modello, si chiama "classification_report", l'output di quest'ultima, tuttavia, mostrerà in maniera diversa le stesse misure. La tabella, infatti avrà una riga dedicata alle caratteristiche degli 0 e una rispetto alle caratteristiche statistiche degli 1. Nella figura successiva viene riportato un esempio.

```
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.62	0.62	0.62	16
1	0.79	0.79	0.79	29
accuracy			0.73	45
macro avg	0.71	0.71	0.71	45
weighted avg	0.73	0.73	0.73	45

È possibile notare come le misure siano speculari ma semplicemente in formato diverso. In particolare, il recall rispetto allo 0 risulterà essere la specificità descritta precedentemente, lo stesso, rispetto questa volta all'uno, sarà invece la misura della sensitività. La precisione è una nuova misura introdotta che rappresenta, nel caso in corrispondenza allo 0, il rapporto tra il numero di volte in cui il modello ha predetto correttamente uno 0 rispetto al numero totale di volte in cui è stato assegnato uno 0. Identicamente in corrispondenza dell'1, il rapporto sarà tra numero di 1 corretti e numero totale di 1 previsti. Sulla sinistra possiamo trovare lo score f1 che può essere interpretato come una media armonica tra precision e recall, la sua formulazione analitica sarà infatti:$F1 = 2\frac{PRECISION*RECALL}{PRECISION+RECALL}$. Il supporto sarà invece il numero totale delle previsioni.

Allo stesso modo, in basso troviamo una tabella riservata all'accuratezza. In questo caso la macro avg e la weighted avg saranno, rispettivamente, la media aritmetica degli score risultanti dalla colonna di appartenenza e la media pesata degli stessi, dove i pesi sono i supporti relativi alle due classi.

2.4.3 La ROC Curve e l'AUC

Gli altri due utili strumenti ricavati dalla costruzione della Confusion Matrix e dal calcolo degli indicatori precedentemente descritti sono la Receiver Operating Characteristics Curve e l'Area Under the Curve. Entrambi sono valori che ci aiutano a comprendere l'abilità effettiva del modello di classificare le osservazioni.

La ROC Curve è generata a partire dal plot del True Positive Rate rispetto al False Positve Rate, entrambi calcolati a più treshold, congiungendo i vari punti ricavati si otterrà la

suddetta curva. L'Area Under the Curve, come richiamato dal nome dell'indicatore stesso, risulterà invece essere l'integrale definito della curva, che avrà un valore compreso tra 0.5 e 1 e rappresenterà la probabilità che il modello rappresenti correttamente un esempio positivo scelto casualmente in maniera migliore rispetto ad un esempio negativo. Seguendo i parametri di Swets avremo che:

- AUC = 0.5 → Il test non è informativo;
- AUC compreso tra 0.5 e 0.7 → Il test è poco informativo;
- AUC compreso tra 0.7 e 0.9 → Il test è mediamente informativo;
- AUC compreso tra 0.9 e 0.99 → Il test è altamente informativo;
- AUC = 1 → Il test risulta essere perfetto

È auspicabile, d'altronde, per evitare il fenomeno dell'overfitting, che l'AUC sia inferiore ad 1.

Nella seguente figura è possibile osservare la ROC Curve relativa ai dati utilizzati precedentemente a titolo esemplificativo:

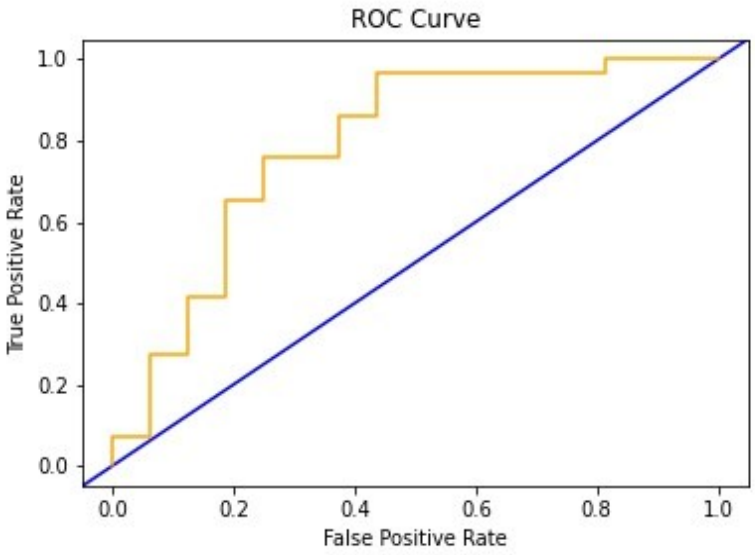


Figura 12 Esempio grafico ROC Curve

L'AUC calcolata, invece, è corrispondente a: 0.7866.

È possibile notare come la curva salga ma che abbia una AUC rientrante nella fascia dei test "mediamente informativi" secondo i criteri Swets riportati precedentemente, indicando che il modello potrebbe non essere la scelta migliore per la previsione out of sample[13].

[13] Bradley, 1997, The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms

CAPITOLO TRE

IL CASE STUDY

SOMMARIO: 3.1 L’estrazione e la pulizia dei dati; 3.2 Data Visualization; 3.3 L’applicazione dei modelli di Machine Learning; 3.3.1 La Logistic Regression; 3.3.2 La Random Forest; 3.3.3 Il Naive Bayes; 3.3.4 Support Vector Machine; 3.3.5 La Linear Discriminant Analysis

3.1 L'estrazione e la pulizia dei dati

Scopo di questo lavoro è la ricerca di un nesso causale tra il sentiment dei testi pubblicati dalla platea risiedente sui social e l'andamento del mercato. Una volta che tutti i dati saranno stati raccolti e puliti, quindi, il lavoro verterà sul verificare se la sentiment analysis possa dare dei miglioramenti, in termini di precisione, nelle previsioni dei movimenti dei titoli quotati rispetto al caso in cui questo predittore non fosse presente nel dataset preso in considerazione.

Per lo svolgimento del lavoro di Tesi è stata necessaria una prima fase di raccolta dei dati, sia per quanto riguarda i tweets sia per i dati relativi all'indice S&P500.

I tweets sono stati raccolti utilizzando una libreria chiamata "snscrape"[14], quest'ultima lavora principalmente per lo scraping dei Social Network come Facebook, Reddit o Instagram, cosa che lo rende molto appetibile anche per lo sviluppo di lavori futuri, magari implementando più fonti per l'opinion mining. Grazie ad essa è possibile generare una query in Python immettendo dei filtri per la richiesta che dovrà effettuare al Social in questione. La query immessa per questo lavoro andava a richiedere tutti i tweets presenti su Twitter che contenessero, però, le parole "SP500" (metodo colloquiale di scrittura per l'indice) o "S&P500". A questo però sono stati aggiunti altri filtri come, ad esempio, l'esclusione dall'analisi delle risposte ai tweets presi in considerazione, ciò è stato fatto per due motivi: il primo è che il dataset con le restrizioni apportate ha un quantitativo già elevato di tweets che sarebbe diventato ingestibile nel caso in cui fossero state inserite anche le risposte agli stessi e, secondariamente, per evitare che la presenza di molti bot, account creati per scopi pubblicitari o di truffa, potessero impattare negativamente sulle analisi svolte, influendo sul sentiment generato dagli algoritmi e di conseguenza anche sulle previsioni dei movimenti di mercato che successivamente verranno esposte. Le date prese in considerazione partono da Gennaio 2021 e arrivano fino a Dicembre 2021 e la lingua impostata come preferita nella raccolta è quella inglese in quanto la maggioranza degli algoritmi sono addestrati per il rilevamento del Sentiment in maniera più precisa con quest'ultima, grazie al suo utilizzo mondiale.

Il dataset è formato da 65'329 tweets ed ha due variabili: una, "Date", riservata al timestamp rilevato alla pubblicazione del tweet e un'altra, "Content", che immagazzina il contenuto del tweet.

Il principale lavoro che viene effettuato sui dati grezzi è quello descritto nel secondo

[14] www.github.com/JustAnotherArchivist/snscrape

capitolo di questo lavoro e riguarda la pulizia dei tweets seguendo le regole del Natural Language Processing. Il primo task è stato quello di andare a rimuovere tutti i nomi di utenti taggati, sostituendoli con "@user", dall'autore del singolo tweet sia per motivi legati alla privacy sia per motivi legati alla buona riuscita dell'analisi in quanto quel tag avrebbe potuto, in seguito, essere considerato dall'algoritmo, erroneamente, come una parola del vocabolario che quindi potrebbe influire sulle analisi. Sono stati rimossi poi da tutti i contenuti raccolti i link riportanti a siti esterni, sostituendoli con "http".

Partendo poi con il lavoro canonico di pulizia dei dati testuali, è stata applicata la funzione mostrata nel capitolo precedente, dividendo ogni tweet in tokens che vengono poi singolarmente per la pulizia, andando quindi a riportare tutto il testo in lettere minuscole, a rimuovere tutte le Escape Sequence come "\n", ad individuare e rimuovere le stopwords e, infine, ad applicare la tecnica "Lemmatize" per ricercare, per ogni singola parola, il lemma d'origine e raggruppare più parole sotto lo stesso lemma, alla fine di ogni iterazione vengono poi ricongiunte tutte le parole nello stesso ordine in cui erano precedentemente e inserite nella riga corrispondente al contenuto originale ma, questa volta, sotto la variabile "Cleaned", che sarà poi soggetta all'analisi del Sentiment.

Si riporta di seguito un esempio ricavato dal dataset con tutti i passaggi effettuati per la pulizia di un singolo tweet contenuto, ciò fa capire quanto le librerie siano efficienti e quanto il Natural Language Processing si sia evoluto anche dal punto di vista computazionale, fino ad arrivare ad una pulizia completa di quasi 66'000 tweets impiegando circa 20 secondi.

Tweet grezzo corrispondente all'indice 18700: "`Google, Amazon, Microsoft and Apple individually have market caps larger than the entire S&pP500, Real Estate &amp, Utilties Sectors;`"

Tweet pulito dalla funzione creata applicando come la Lemmatization: "`google, amazon, microsoft apple individually market cap larger entire s&p, real estate &amp, utilties sector`"

Per quanto riguarda, invece, l'estrazione dei dati riguardanti l'indice preso in considerazione, è stata utilizzata la libreria "yfinance" che richiede al sito finance.yahoo.com i dati relativi al ticker immesso come parametro della funzione "download". Il dataset così formato avrà 7 variabili e sarà formato da circa 260 osservazioni coincidenti con i giorni di apertura del mercato. Le 7 variabili saranno: la data relativa ai dati giornalieri, il prezzo di apertura corrispondente, il massimo e il minimo raggiunti nel corso della giornata, il prezzo di chiusura, il volume degli scambi e il prezzo di chiusura aggiustato che, in questo caso specifico, trattandosi di un indice e non di un titolo corporate,

non avrà differenze rispetto al prezzo di chiusura standard in quanto l'aggiustamento tiene conto della distribuzione dei dividendi e degli stock split che, nel caso dell'S&P500, non sono presenti.

3.2 Data Visualization

Nel corso di questo paragrafo verranno riportati molteplici plot per la comprensione dei due dataset estrapolati. Una volta applicato l'algoritmo di TextBlob a tutti i tweets precedentemente menzionati, è possibile ricavare, grazie agli strumenti di Data Visualization a disposizione, varie insights in essi contenuti.
Una prima immagine a cui vale la pena dedicare attenzione è l'andamento del numero dei tweets per giorno di pubblicazione:

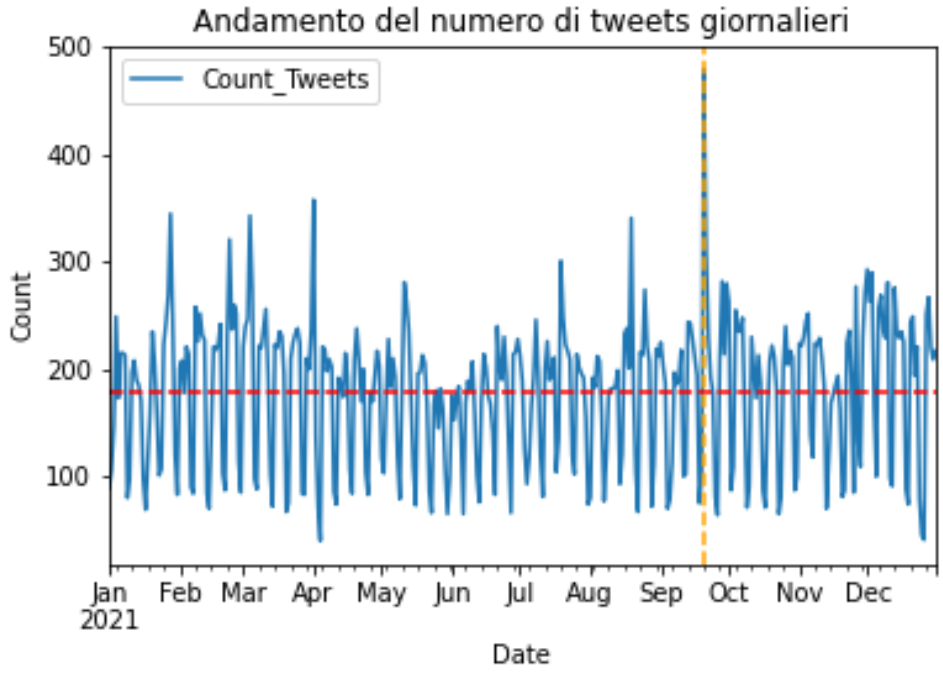


Figura 13: Serie Storica numero dei tweets

Viene mostrato come la pubblicazione dei tweets relativi all'indice siano stati per lo più costanti in media nel tempo, senza particolari trend da riportare, uno spunto interessante però può essere dato dal picco, segnalato dalla linea verticale in giallo, avvenuto il 20 Settembre 2021. Filtrando le notizie circolanti durante quella giornata è possibile capire fin da subito come il picco dei tweets pubblicati coincidano con uno dei cali più consistenti da Maggio 2021 ad allora per l'indice, come riportato dalla CNBC all'interno del suo articolo[15]

[15]www.cnbc.com/2021/09/20/stocks-see-worst-day-in-months-four-experts-share-their-strategies.html

datato proprio 20 Settembre. Molti insider, tra cui Rick Rieder, CIO del colosso BlackRock, esclamarono che i motivi erano sostanzialmente in tutti i campi sociali: il Covid stava riprendendo piede, la Cina si mostrava sempre più forte e la crescita economica stava rallentando. Proprio il CIO affermò però che probabilmente, data la pressione del mercato in quei giorni, ci sarebbero stati acquisti a basso prezzo di azioni in disceso da parte di molti investitori, compresa l'enorme Società d'Investimenti. Già dalla prima figura, quindi, è possibile osservare quanto la Social Media Sentiment Analysis in ambito finanziario possa essere importante in quanto potrebbe portare ad avere spunti aggiuntivi rispetto ai tradizionali indicatori per la scelta degli investimenti.

Viene riportata anche la tabella risultante dall'output di summary con le principali statistiche del conteggio dei tweets giornalieri.

index	count	mean	Std	min	0.25	0.5	0.75	max
Summary	365	178.9671233	66.88085202	40	115	194	223	480

La media, ricalcata con una linea tratteggiata orizzontale rossa nel grafico precedente, è di circa 179 tweets al giorno, con un minimo raggiunto di 40, una mediana di 194 e una deviazione standard di circa 67.

Un altro aspetto da portare all'attenzione è la distribuzione del sentiment stimato dei tweets estrapolati in precedenza, in modo da poter fin da subito notare come, in generale, il sentiment dei tweets sia stato durante l'arco temporale preso in considerazione.

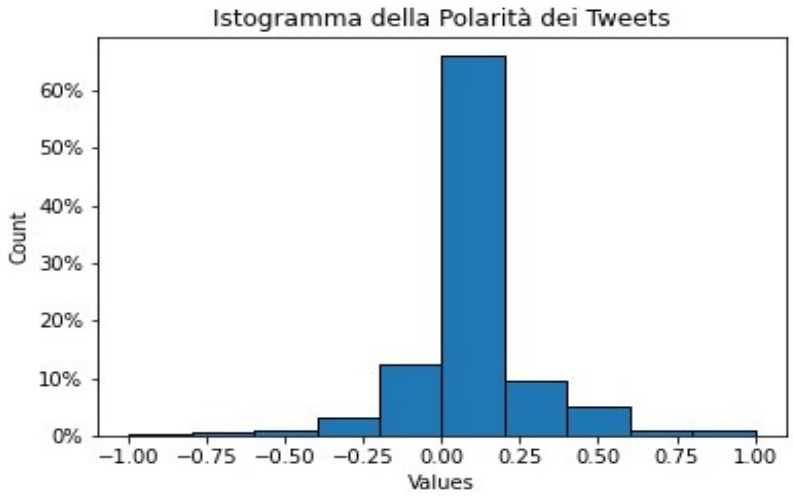


Figura 14: Istogramma della polarità dei tweets

È possibile notare come la maggior parte dei tweets, più del 60% del totale, è stata classificata tra lo 0 e lo 0.25 nella scala tra -1 e +1, ciò può accadere quando si lavora con la Social Media Sentiment Analysis dove l'argomento cardine comporta la pubblicazione

di notizie senza particolari opinioni come ad esempio:

```
print(data[['Content','Sentiment']].iloc[235, :])
Content      Current S&P500 #SP500 Price: $4,778.73 📉
Sentiment                                                0.0
Name: 235, dtype: object
```

In questo caso è possibile notare come il solo scopo della pubblicazione del tweet fosse quello di informare il pubblico di un prezzo risultante in calo (intuibile attraverso l'emoji utilizzata rappresentante un grafico in discesa). Lo score del sentiment è di 0, indicato quindi come neutrale, ciò viene considerato corretto in quanto non si rilevano delle opinioni ma solo una mera trascrizione di fatti osservabili.

Si può notare come, al di là del treshold di -0.25 e +0.25, risultino in maggioranza i tweets identificati con uno score positivo, sintomo del fatto che, esclusi i report di notizie quotidiane, la maggior parte di coloro che pubblicavano tweets riguardanti l'indice preso in analisi avessero un sentiment positivo nei confronti dello stesso. Viene riportato per contezza il conteggio accurato dell'osservazione espressa pocanzi:

```
print(data['Sentiment'][data['Sentiment']>0.25].count())
print(data['Sentiment'][data['Sentiment']<-0.25].count())
8002
2249
```

Vi sono circa 10'250 tweets al di fuori della soglia di neutralità e, come è possibile osservare dai dati riportati, i tweets positivi siano circa il 78% del totale mentre i negativi solo il 22%. Da ciò è quindi ricavabile che, durante il periodo, gli utenti che volevano esprimere un'opinione, abbiano avuto un Sentiment positivo. Da notare che dal conteggio sono stati esclusi i tweet con un Sentiment solo leggermente positivo in quanto considerati appartenenti alla categoria dei Neutrali, se fosse stato preso come treshold la sola positività, la percentuale sarebbe passata a più del 90% dei tweets etichettati come positivi.

Solitamente, negli studi riguardanti il Natural Language Processing, risulta interessante anche guardare alle differenze che contraddistinguono le due labels:

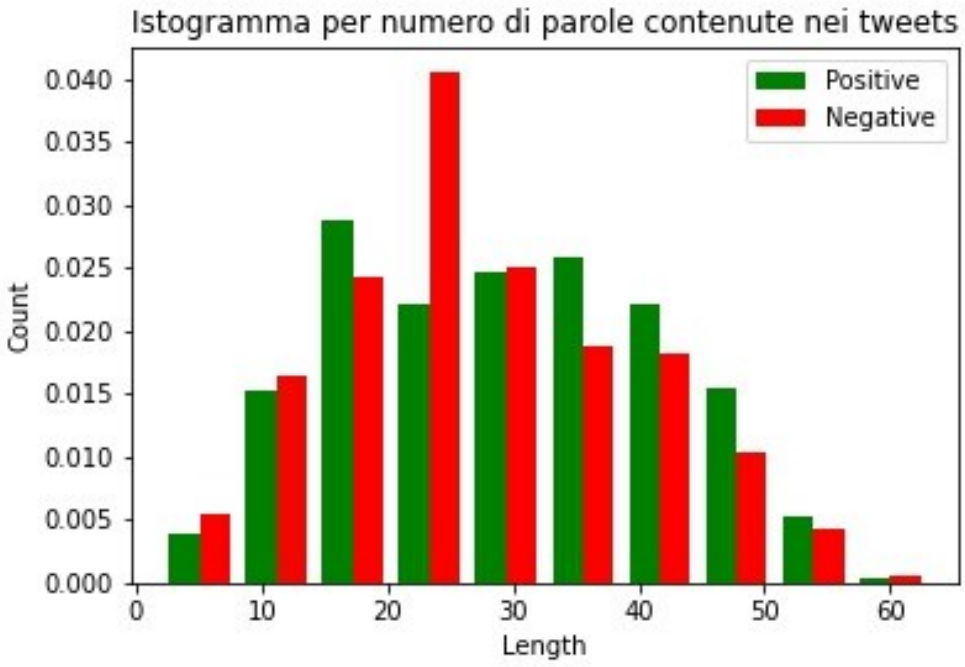


Figura 15: Istogramma lunghezza dei tweets

È riportato quindi un istogramma in cui viene mostrato un confronto tra la distribuzione del numero di parole nei tweets contrassegnati con un punteggio superiore a +0.25 come positivi e quelli con un punteggio inferiore a -0.25 come negativi. Si può notare come la differenza sia principalmente nel fatto che i tweets positivi abbiano un picco al 3% in corrispondenza del numero di parole al loro interno comprese tra le 10 e le 20 mentre quelli negativi lo abbiano al 4% quando la lunghezza è compresa tra le 20 e le 30 parole.

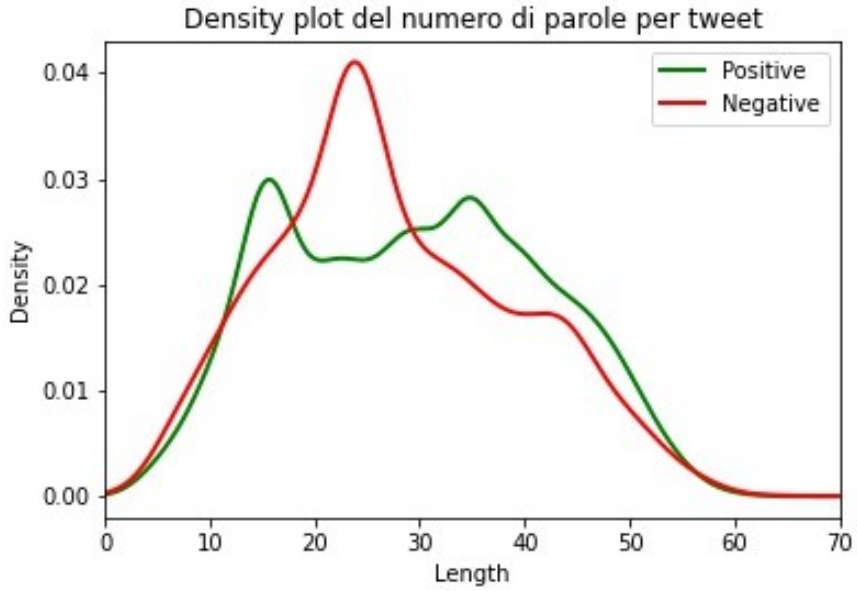


Figura 16: Densità del numero di parole nei tweets.

Per una maggiore leggibilità è stato riportato anche il Density plot nello stesso ordine di grandezza. Risulta qui più evidente, invece, che vi è una maggiore densità di tweets con un numero più alto di parole contenute al loro interno che sono state etichettate come positive, sintomo, probabilmente, che coloro che risultavano essere più ottimisti rispetto all'indice, avessero più pensieri da esprimere rispetto a chi era pessimista, il quale, questi ultimi, si limitavano ad esprimere un pensiero che, con buona probabilità, era intorno alle 20 o 30 parole.

Altro strumento fondamentale per la Data Visualization in ambito NLP è la Word Cloud. Essa rappresenta una nuova forma di visualizzazione dei dati testuali, utilizzata anche spesso per la comunicazione e il marketing. Questa è costruita a partire dalla lista delle parole contenute all'interno del dataset, schematizzate sia in dimensione che in colore in base al numero di volte che quella determinata parola è stata utilizzata.

Ho provveduto alla divisione del dataset originale in due nuovi e distinti in base alla positività o negatività del sentiment. Vengono riportate qui di seguito, dunque, tre Word Cloud separate: una costruita a partire dal dataset originale, una con i soli elementi contrassegnati come positivi e un'ultima con gli elementi negativi. Viene poi riportato, per ogni set di dati, un barplot con i dati esatti dei conteggi delle parole all'interno di ciascuno di essi.

Partendo con la prima, si può visualizzare la nuvola di parole corrispondente a tutto il dataset originale quindi sia con i tweets positivi che quelli negativi.

Figura 17: WordsCloud generale

Per la costruzione dell'immagine in questione è stato usato la library dedicata di Python "wordcloud", utilizzando come contorno il logo di Twitter. Si nota come le parole più utilizzate all'interno del dataset siano tutte attinenti alla finanza, la più utilizzata risulta essere 'stock', azione, richiamo quindi al probabile utilizzo per inquadrare l'azione in oggetto in quello specifico tweet. Spicca, inoltre, un altro indice molto conosciuto, il nasdaq, sinonimo quindi di attenzione degli utenti non solo al singolo indice ma anche alla totalità. Si parla poi di futures, che riportano l'attenzione sull'indice in quanto, attraverso gli Exchange Traded Fund è possibile andare ad acquistare un titolo che replichi fedelmente l'andamento del sottostante e, con la parola future, ci si potrebbe riferire ad un acquisto future che abbia come sottostante il titolo scambiato per ETF. Interessante è anche la presenza della cryptovaluta "Bitcoin" che spesso viene richiamata soprattutto per mettere in paragonare i rendimenti dei due ticker. La parola "SPY" invece, molto utilizzata nei tweets, rappresenta uno degli ETF più famosi che vanno a replicare l'indice S&P500. Successivamente viene riportata la seconda WordCloud che invece sarà specifica per i tweets positivi.

Figura 18: WordsCloud tweets positivi

Essendo i tweets positivi numericamente in maggioranza, è normale ritrovare molte parole contenute nel precedente grafico ma, a differenza del precedente, spiccano le presenze delle parole etf, che può essere richiamata in quanto acquisto in voga negli ultimi anni da parte degli investitori, tsla, ticker della Tesla di Elon Musk che è entrata a far parte dell'indice nel dicembre del 2020, poco prima del periodo selezionato nell'analisi, aapl, ticker di una delle società più capitalizzate al mondo che richiama l'attenzione di moltissimi investitori

in tutto il mondo.

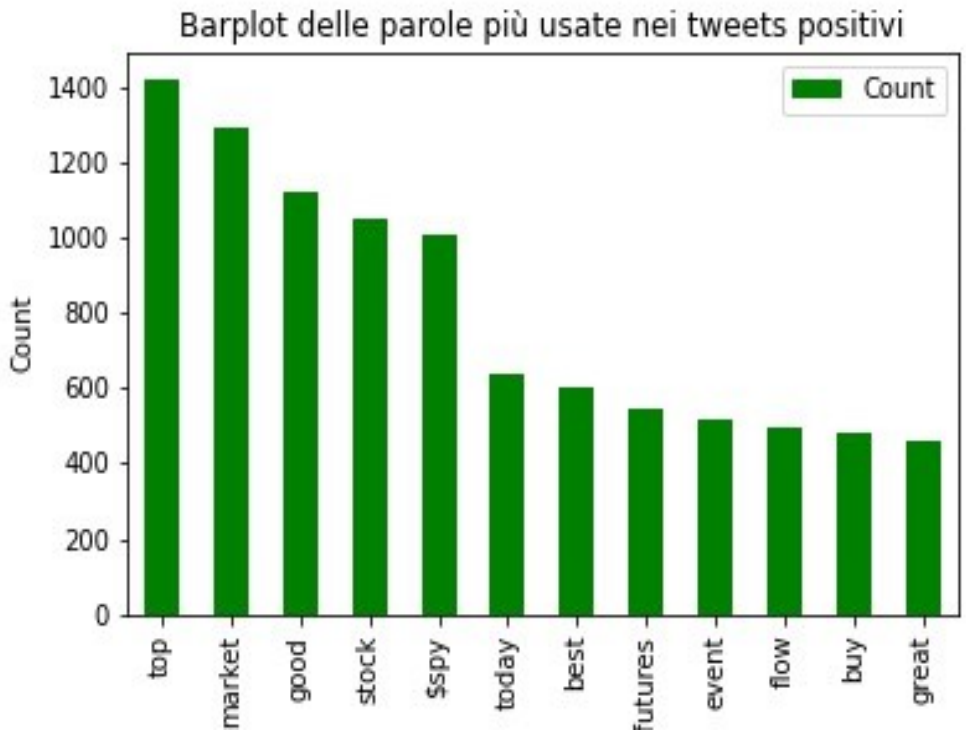


Figura 19: Parole più usate nei tweets positivi.

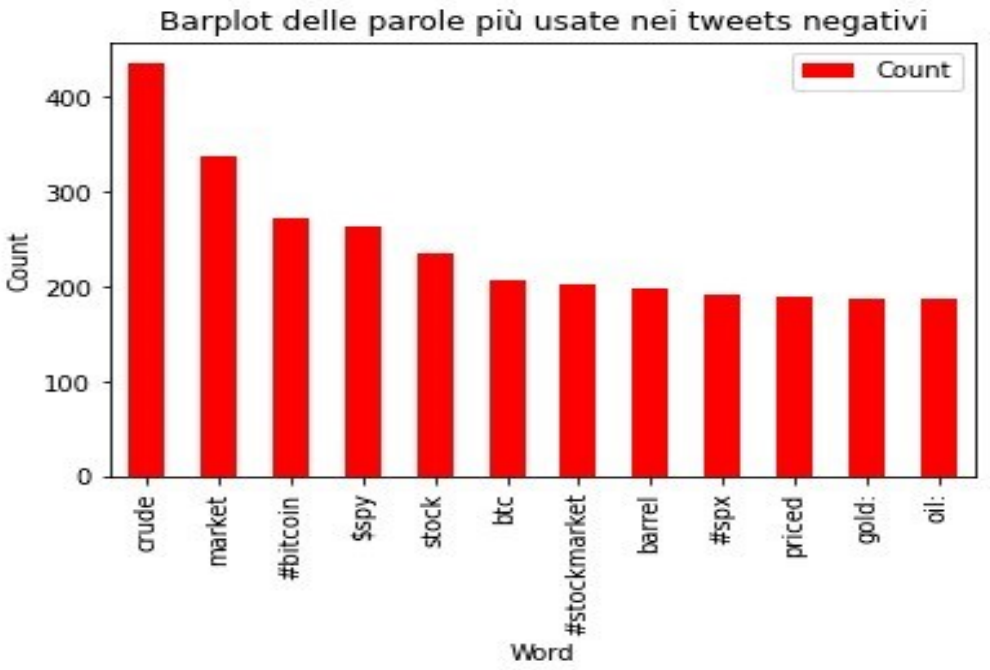
Attraverso l'istogramma riportato appena sopra è possibile notare come le parole utilizzate maggiormente nei tweets positivi siano particolarmente rilevanti per la Sentiment Analysis, sono da osservare le parole come "top" riportata più di 1400 volte, "good", più di 1000, "best", rilevato circa 600 volte, o "great", circa 400 volte. La più interessante, tuttavia, potrebbe essere la parola "buy" che è la rappresentazione migliore dell'ottimismo verso un titolo in ambito finanziario. Osservare le parole più utilizzate, nel Natural Language Processing, ci fa sia capire come l'algoritmo funzioni, in questo caso ad esempio andando ad assegnare score elevati alle parole positive come quelle menzionate, e ci fanno anche riflettere sulla bontà del modello utilizzato, se avessimo trovato parole come "worst" o "sell" ci saremmo ad esempio interrogati sul motivo per il quale ci fossero e, di conseguenza, avremmo riadattato o cambiato totalmente metodo di rilevamento del sentiment.

L'ultima nuvola di parole rappresenta invece le parole più frequenti contenute nei tweets contrassegnati da TextBlob come Negativi.

Figura 20: WordsCloud tweets negativi.

Vediamo qui una differenza fondamentale rispetto ai due grafici osservati precedentemente, spicca, di gran lunga, l'utilizzo della parola "oil" e della parola "crude".

Il loro utilizzo, all'interno dei tweets, avrebbe una probabile doppia valenza: sarebbe anzitutto un segnale che il prezzo del 'Crude Oil' durante il periodo considerato potrebbe aver subito una variazione sostanziale e, se così fosse, sarebbe anche un modo indiretto per confermare che la Sentiment Analysis ha dato risultati attendibili per i tweets negativi.



L'istogramma conferma quanto il crude oil sia stato sotto i riflettori nell'anno 2021 e tutt'oggi dato l'incremento delle materie prime dovuto al conflitto tra Russia e Ucraina scoppiato a Febbraio 2022. Per lo studio di questo fenomeno e la conferma dello stesso è stata effettuata un'analisi grafica dell'andamento del prezzo del Crude Oil nel periodo d'interesse.

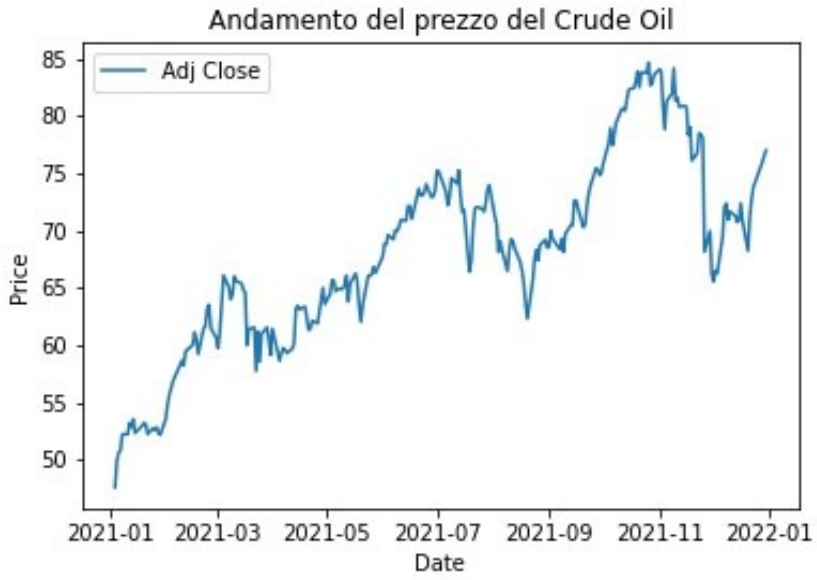


Figura 21: Serie storica Crude Oil

Si nota subito come il prezzo, nello specifico del prezzo di chiusura, sia stato soggetto ad un rialzo vertiginoso, passando dai 45$ di inizio anno ad un picco di 85$ ad inizio novembre 2021. Questa risalita è stata quindi portatrice di sentiment negativo per gli utenti del social media che hanno pubblicato tweets a riguardo. Ancora una volta si conferma che i tweets e il relativo sentiment possono essere degli indicatori di eventi finanziari e proprio per questo la ricerca sia pubblica che privata sta ampliando le risorse messe a disposizione su questo tema.

Si passerà ora alla visualizzazione dei dati finanziari estrapolati che saranno poi oggetto delle stime in analisi insieme al Sentiment. Come anticipato i dati in questione saranno presi dal sito finance.yahoo.com, attraverso Python, che fornisce tutti i dati necessari per lo studio. Si riporta di seguito la tabella con i primi dati a disposizione.

Date	Open	High	Low	Close	Adj Close	Volume
12/30/2021	$ 4,794.23	$ 4,808.93	$ 4,775.33	$ 4,778.73	$ 4,778.73	$ 2,390,990,000.00
12/29/2021	$ 4,788.64	$ 4,804.06	$ 4,778.08	$ 4,793.06	$ 4,793.06	$ 2,369,370,000.00
12/28/2021	$ 4,795.49	$ 4,807.02	$ 4,780.04	$ 4,786.35	$ 4,786.35	$ 2,217,050,000.00
12/27/2021	$ 4,733.99	$ 4,791.49	$ 4,733.99	$ 4,791.19	$ 4,791.19	$ 2,264,120,000.00
12/23/2021	$ 4,703.96	$ 4,740.74	$ 4,703.96	$ 4,725.79	$ 4,725.79	$ 2,194,630,000.00

Le colonne saranno poi selezionate per la stima successiva delle previsioni. I giorni presi in considerazione saranno corrispondenti alle date di apertura della borsa e, di conseguenza, saranno circa 240. Come si è visto nei grafici riportati al Capitolo Uno di questo studio, l'indice fin dal 2004 ha avuto una crescita incessante e ciò, nonostante tutti i fronti di crisi riportati precedentemente dal CIO di BlackRock, è stato rilevato anche nel corso dell'anno 2021.

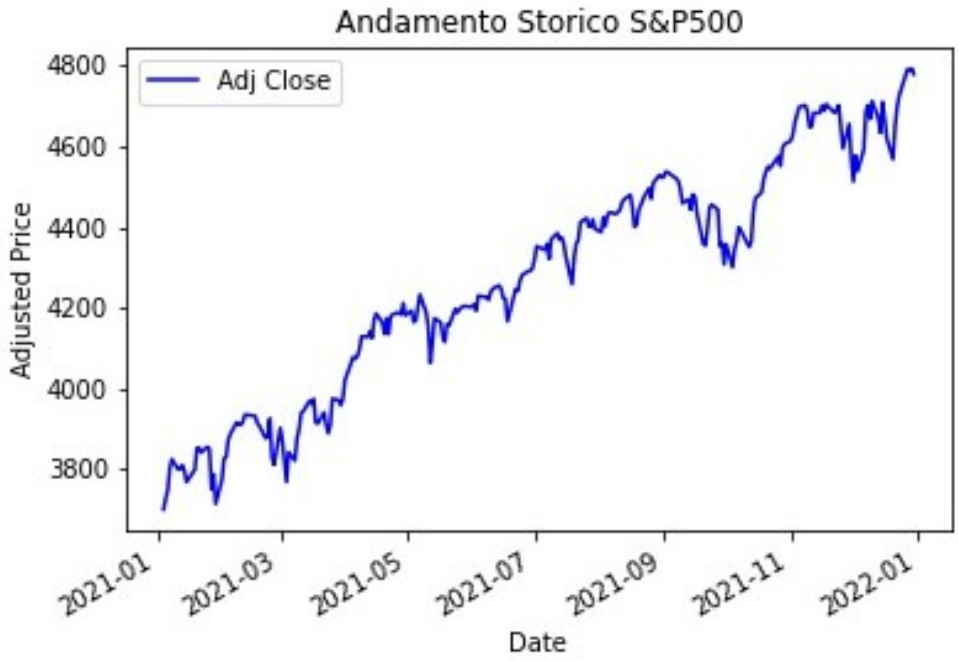


Figura 22: Andamento S&P500 nel 2021

Fin dall'inizio del 2021, anno immediatamente successivo a quello in cui il mondo intero viveva l'inizio della tragedia scatenata dalla pandemia e dai conseguenti crolli sui mercati finanziari mondiali, l'S&P500 ha subito una crescita vertiginosa passando dai circa 3900$ ad inizio anno ai circa 4794$ registrati a dicembre. Ancora una volta è possibile notare ciò di cui si parlava all'inizio di questo paragrafo, la caduta che ha avuto il prezzo dell'S&P in corrispondenza dei mesi di settembre e ottobre ha fatto sì che ci fosse un balzo nell'attenzione verso lo stesso da parte degli utenti del Social Media che hanno fatto sentire la propria voce. Per lo studio in questione è stata creata una variabile binaria che indica se, tra un giorno di apertura del mercato e l'altro, vi sia stata una crescita del prezzo o meno. Quest'ultima sarà la variabile risposta all'interno dei modelli che si andranno ad implementare e, quindi, si verificherà se è possibile prevedere una crescita grazie all'utilizzo del Sentiment, in maniera più accurata rispetto a quando lo stesso non viene implementato nel modello. Una prima grafico statistico d'interesse di questa nuova variabile è sicuramente

l'istogramma, con esso si potrà subito osservare come sia stato effettivamente l'andamento del mercato in termini di giorni in cui vi è stata una crescita del prezzo, contrassegnato con un 1, e quando invece il prezzo è rimasto costante o è diminuito, con uno 0.

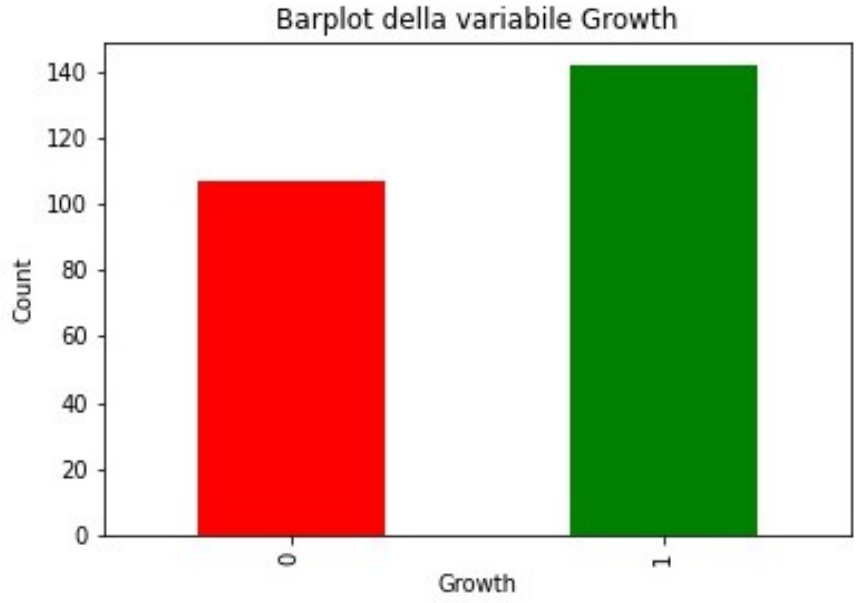
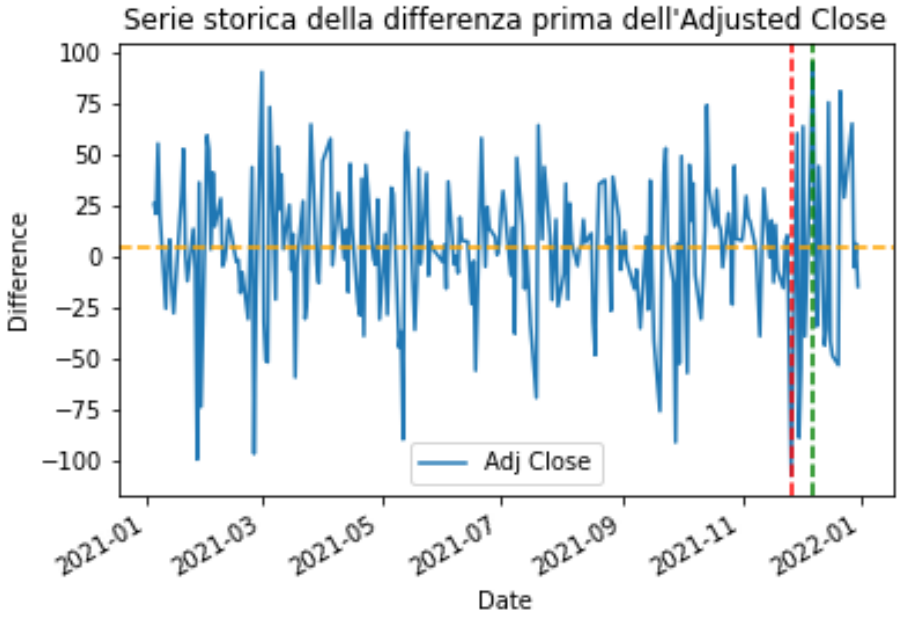


Figura 23: Barplot della variabile Growth

Il numero dei giorni in cui vi è stata una crescita rispetto al giorno precedente è stato di 147 mentre il numero di decrescite o di costanza ammonta a 107, ciò ci conferma che l'andamento totale è stato positivo data anche la costanza delle crescite del prezzo da un giorno all'altro che è possibile visualizzare grazie ad una serie storica alle differenze prime generata sulla variabile 'Adj Close'.



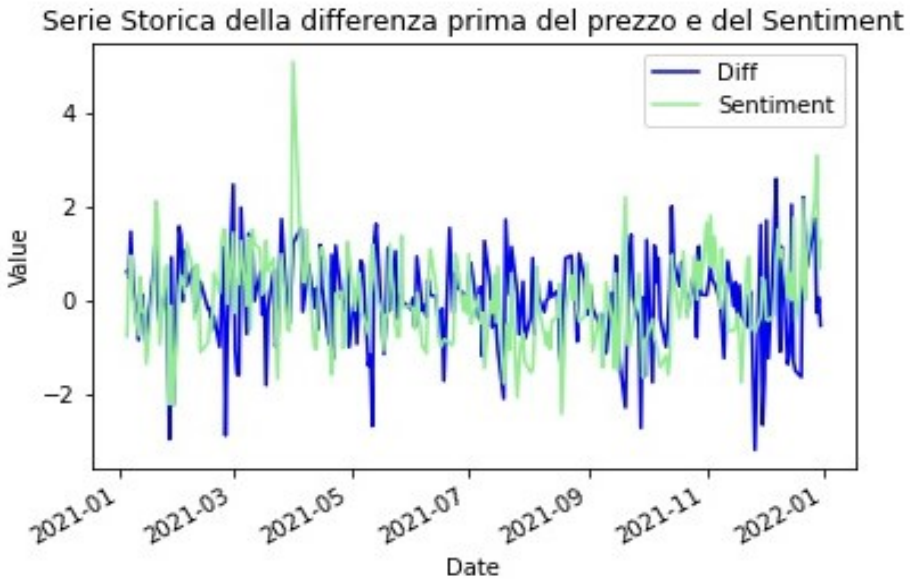
La linea arancione visualizzabile rappresenta la media del periodo e, come è possibile notare, è di poco sopra lo 0 ma pur sempre positiva. L'osservazione più interessante di questo grafico, però, risiede nell'entità delle cadute del prezzo dell'indice. Per quanto le crescite siano state costanti, con due picchi in corrispondenza delle più elevate, il grafico mostra come vi siano state diminuzioni più accentuate. Con la linea verde è rappresentato il massimo valore della crescita giornaliera tra un giorno e l'altro mentre con quella di colore rosso è evidenziato il picco negativo avvenuto nel periodo. Per quantificare l'entità si riporta di seguito l'output delle due funzioni.

```
print(Historic_interest['Adj Close'][Historic_interest['Adj Clo
se'].idxmax()])
print(Historic_interest['Adj Close'][Historic_interest['Adj Clo
se'].idxmin()])
95.080078125
-106.83984375
```

Risulta chiaro come la caduta più elevata supera di gran lunga, circa di 11$, la più elevata risalita. Ciò potrebbe essere l'indicatore dell'avversione al rischio degli investitori che, nel caso di notizie negative o possibili tali, effettuano vendite in massa dei titoli contenuti nell'indice soggetti a tali avvenimenti.

Ultima possibile visualizzazione può essere un paragone grafico tra la differenza prima dei prezzi di chiusura e l'andamento del Sentiment stimato. Queste misure hanno diverse scale e, quindi, è stata effettuata una standardizzazione delle due variabili in modo da renderle immuni all'unità di misura utilizzata. La standardizzazione segue la formula analitica:

$$z_i = \frac{x_i - \bar{x}}{\sigma}$$



Nei periodi di tranquillità è possibile notare come le due serie abbiano un andamento molto simile che si va a discostare soprattutto quando vi sono picchi negativi nelle differenze prime dei prezzi. Probabile che vi sia, tra gli utenti, un cauto ottimismo verso l'indice anche nei momenti di decrescita, anche dovuto alle performance storiche dell'indice che vede un rendimento medio annuo dal 1926 al 2016 del 6.70%, percentuale che ha contribuito a far crescere la fiducia verso il paniere di titoli anche in momenti bui dello stesso, corrispondenti a quelli dell'intera economia americana.

3.3 L'applicazione dei modelli di Machine Learning

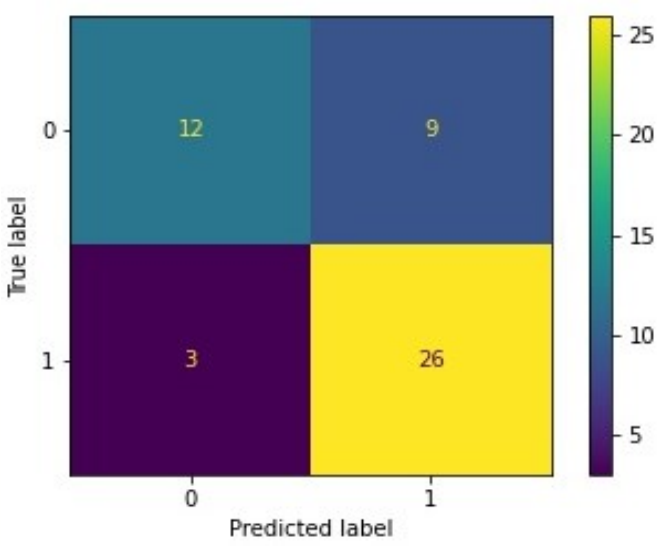
Una volta creato e descritto il dataset, costruito a partire dai due set originali, i tweets e i prezzi, si passa dunque alla fase finale del lavoro di Tesi che consiste nell'applicazione e, se possibile, nella scelta del modello migliore applicabile a tale scopo. I modelli che verranno usati di seguito sono gli stessi descritti in precedenza nel Capitolo Due. Gli stessi verranno applicati due volte, una prima volta sul dataset contenente la variabile riguardante il Sentiment, una seconda volta senza la stessa. Ciò per mettere a confronto i risultati ottenuti e osservare se quest'ultima apporti effettivamente un contributo nella stima delle previsioni. Per l'applicazione dei modelli e la valutazione dei risultati è stato ancora una volta diviso il dataset in due: una parte, il training set, formato dall'80% delle osservazioni, verrà usato per l'addestramento dei modelli, l'altra, il test set, verrà utilizzato per la verifica delle previsioni e per la creazione dei report necessari alla valutazione della loro bontà di adattamento ai dati.

3.3.1 La Logistic Regression

Il primo modello che viene preso in considerazione è la Logistic Regression. Si riportano di seguito i risultati ottenuti dall'applicazione dello stesso.

```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
model = classifier.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred, labels=model.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
                              display_labels=model.classes_)
disp.plot()
plt.show()
```

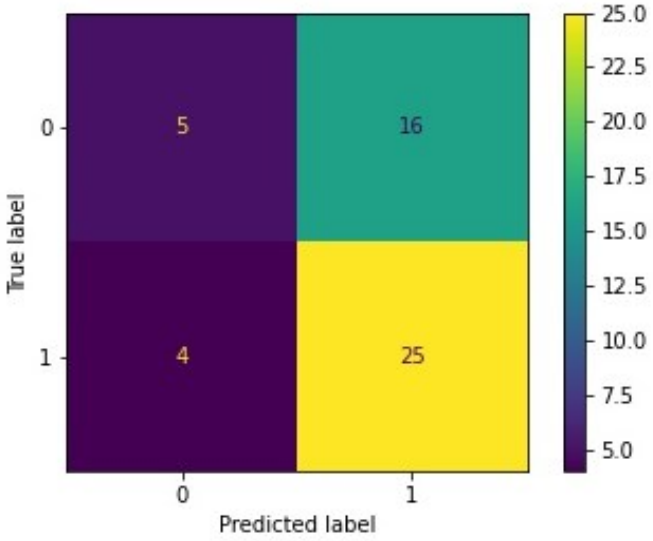
	precision	recall	f1-score	support
0	0.80	0.57	0.67	21
1	0.74	0.90	0.81	29
accuracy			0.76	50
macro avg	0.77	0.73	0.74	50
weighted avg	0.77	0.76	0.75	50



È subito chiaro come la regressione abbia avuto dei buoni risultati sul dataset con all'interno la variabile Sentiment. Il test set contiene 50 osservazioni, 21 che rappresentano una decrescita dell'indice e 29 che ne rappresentano una crescita. Il modello riesce a prevedere correttamente 12 volte lo 0 e 26 volte l'1. Sbaglia però, generando False Negative, in 3 occasioni e 9 volte generando False Positive. Il modello, dunque, riesce a prevedere molto bene le occasioni in cui il prezzo dell'indice cresce ma ne sovrastima l'impatto, producendo un errore grave in quanto il modello suggerisce di investire per 9 volte in quei determinati giorni mentre in quelle occasioni vi sarebbe stata una perdita. Per riportare il tutto alle formule riportate nel capitolo precedente, si stima che il True Positive Rate è di 0.8965 mentre il True Negative Rate è di 0.5714.

Per un confronto diretto vengono riportati i risultati dello stesso modello addestrato questa volta sul dataset senza la variabile Sentiment. E' stato impostato un random state al momento della creazione del train e test set in modo da rendere paragonabili i due risultati, addestrando dunque questo modello e i successivi su dataset uguali a meno della variabile sentiment. Il codice utilizzato per la generazione dei risultati è lo stesso del precedente, quindi, viene omesso per evitarne la ridondanza.

```
              precision    recall  f1-score   support

           0       0.56      0.24      0.33        21
           1       0.61      0.86      0.71        29

    accuracy                           0.60        50
   macro avg       0.58      0.55      0.52        50
weighted avg       0.59      0.60      0.55        50
```



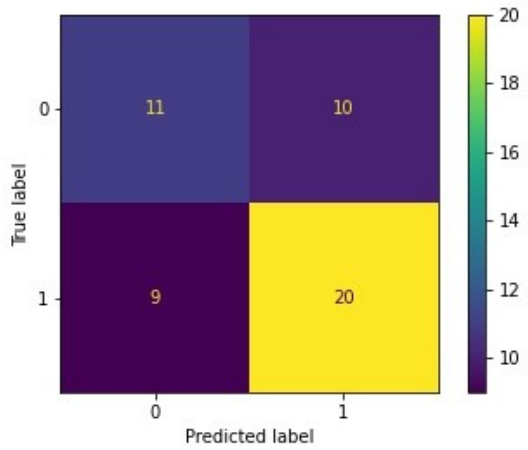
Si nota subito come l'omissione della sola variabile Sentiment abbia avuto un impatto significativo sulle previsioni, si è passato da un 80% di precisione sullo 0 e un 74% sull'1 a, rispettivamente, 56 e 61% e ad un TPR di 0.86 e TPN di 0.24, notevolmente più bassi dei precedenti risultati.

3.3.2 La Random Forest

Il secondo modello applicato ai dataset è la Random Forest, metodo di ensemble in cui i singoli alberi, attraverso un sistema di votazione, stabiliscono se la stima debba essere 0 o 1. I risultati sul dataset con la variabile Sentiment sono riportati di seguito.

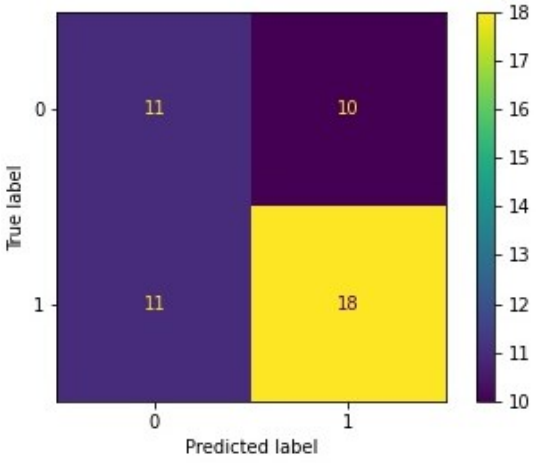
```
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier()
model = classifier.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred, labels=model.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
                              display_labels=model.classes_)
disp.plot()
plt.show()
```

	precision	recall	f1-score	support
0	0.58	0.52	0.55	21
1	0.68	0.72	0.70	29
accuracy			0.64	50
macro avg	0.63	0.62	0.62	50
weighted avg	0.64	0.64	0.64	50



In questo caso il modello non riesce ad ottenere risultati ottimali confrontati con il precedente, otteniamo una precisione del 58% sulla previsione degli eventi negativi e 68% sui positivi. Il TPR è pari a 0.72 mentre il TNR è di 0.52, indicando che la Random Forest non riesce ad adattarsi al meglio ai dati. Si riportano per completezza i risultati ottenuti sul dataset senza la variabile Sentiment.

	precision	recall	f1-score	support
0	0.50	0.52	0.51	21
1	0.64	0.62	0.63	29
accuracy			0.58	50
macro avg	0.57	0.57	0.57	50
weighted avg	0.58	0.58	0.58	50



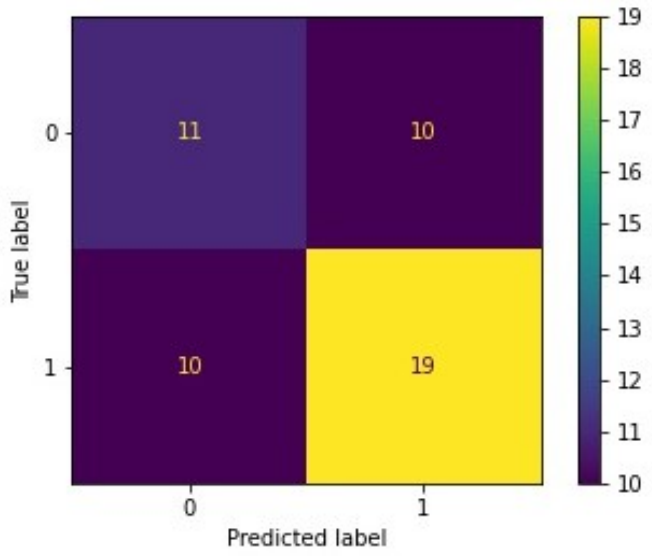
Le differenze tra i due modelli, in questo caso, risultano meno marcate rispetto al confronto precedente. Risultano esserci, in termini assoluti, solo due False Negative in più rispetto al precedente. La precisione media rimane però più alta nel modello con il Sentiment essendo il TPR e il TNR rispettivamente di 0.62 e 0.52.

3.3.3 Il Naive Bayes

Il terzo modello di cui si riportano gli output ottenuti è il Naive Bayes. Quest'ultimo risulta essere uno dei più utilizzati nel Natural Language Processing come proprio nell'algoritmo di TextBlob o nella Spam Detection.

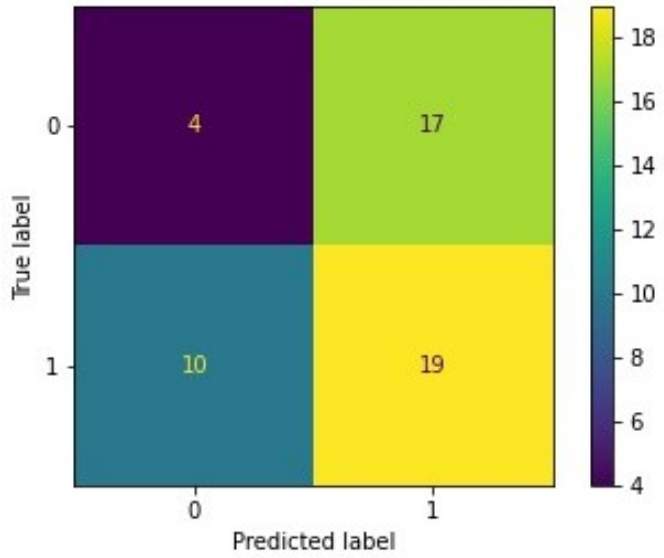
```
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
model = classifier.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred, labels=model.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
                              display_labels=model.classes_)
disp.plot()
plt.show()
```

```
              precision    recall  f1-score   support

           0       0.52      0.52      0.52        21
           1       0.66      0.66      0.66        29

    accuracy                           0.60        50
   macro avg       0.59      0.59      0.59        50
weighted avg       0.60      0.60      0.60        50
```



Anche in questo caso i risultati non sono paragonabili a quelli ottenuti dalla Regressione Logistica nel primo paragrafo di questo capitolo, risulta esservi una situazione con un TPR e un TFR di 0.66 e di 0.52 con una precisione media totale intorno al 60%. Il modello, pur essendo adattabile all'NLP, non può essere adattato ai dati in oggetto composti sia da osservazione finanziarie che da quelle di tipologia appartenente al NLP. Per puro scopo comparativo vengono riportati gli output riguardanti il modello addestrato sul dataset senza la variabile Sentiment.

	precision	recall	f1-score	support
0	0.29	0.19	0.23	21
1	0.53	0.66	0.58	29
accuracy			0.46	50
macro avg	0.41	0.42	0.41	50
weighted avg	0.43	0.46	0.44	50



Otteniamo in questo caso dei risultati pessimi in seguito all'applicazione del Naive Bayes sul dataset senza la variabile Sentiment che, in questo caso, ha avuto un'importanza fondamentale, soprattutto nella valutazione dei casi Negativi. Otteniamo un TPR del 66%, uguale al precedente, ma un TNR solo del 19%. La scelta di questo modello e di questo particolare dataset senza la colonna del Sentiment avrebbe scaturito una perdita ingente di capitali, ad esempio, nella società di investimento dove essi sarebbero stati applicati come riferimenti per la scelta di investimenti.

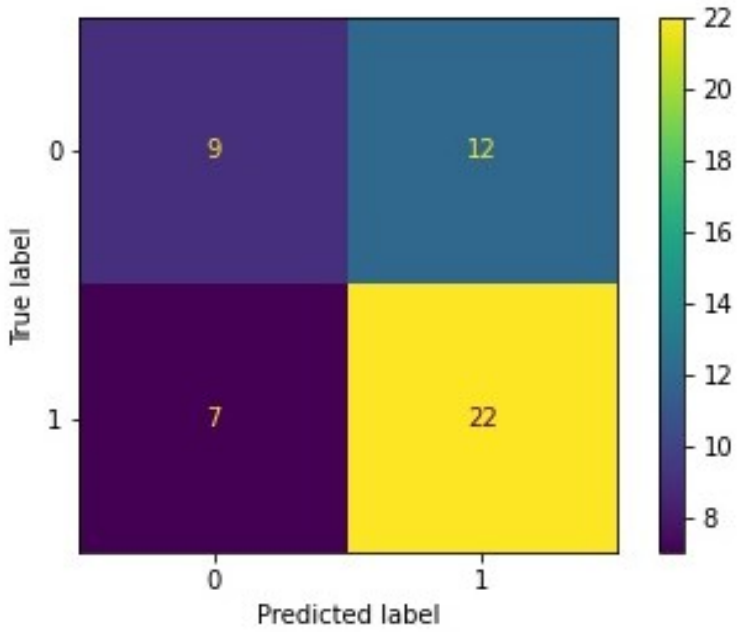
3.3.4 Support Vector Machine

Con il quarto modello si utilizza, anche questa volta, un modello molto usato nello sviluppo di algoritmi per la Supervised Sentiment Detection, il Support Vector Machine. Vengono

riportati gli output, come precedentemente fatto, sia per il dataset con il Sentiment sia per quello privo della stessa.

```
from sklearn.svm import SVC
classifier = SVC()
model = classifier.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred, labels=model.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
                              display_labels=model.classes_)
disp.plot()
plt.show()
```

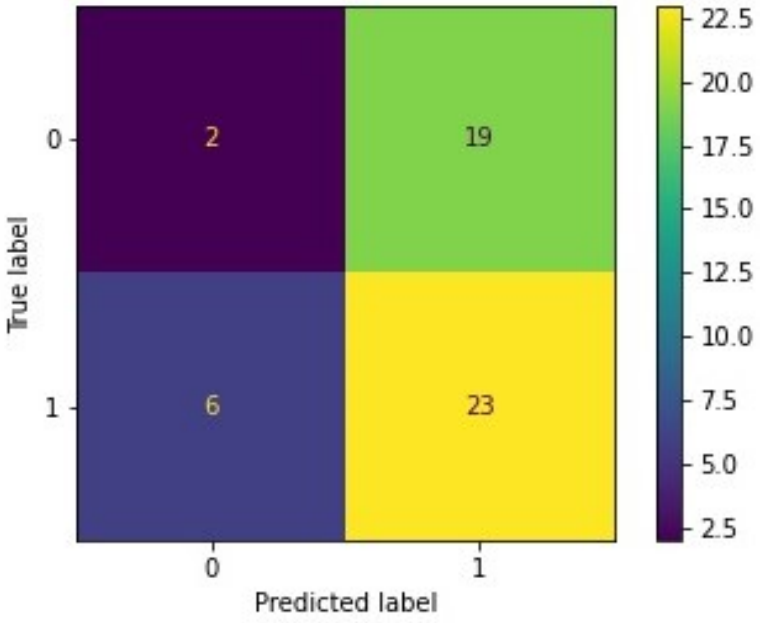
```
              precision    recall  f1-score   support

           0       0.56      0.43      0.49        21
           1       0.65      0.76      0.70        29

    accuracy                           0.62        50
   macro avg       0.60      0.59      0.59        50
weighted avg       0.61      0.62      0.61        50
```



Sono stati ottenuti risultati in linea con i modelli precedenti ma non paragonabili, ancora una volta con la Regressione Logistica. Un TPR del 76% e un TFR del 43% non sono vicini a quelli ottenuti precedentemente rendendo quindi il modello non utilizzabile.

Per quanto riguarda invece il confronto con lo stesso modello utilizzato però in assenza del valore della Sentiment:

```
              precision    recall  f1-score   support

           0       0.25      0.10      0.14        21
           1       0.55      0.79      0.65        29

    accuracy                           0.50        50
   macro avg       0.40      0.44      0.39        50
weighted avg       0.42      0.50      0.43        50
```



Nuovamente il modello senza il Sentiment sovrastima pesantemente la stima dei caratteri positivi incidendo sul True Negative Rate portandolo solo al 10%, su 50 osservazioni contenute nel test set sono stati stimati 42 esiti positivi, risultando quindi in un pessimo modello per le scelte d'investimento.

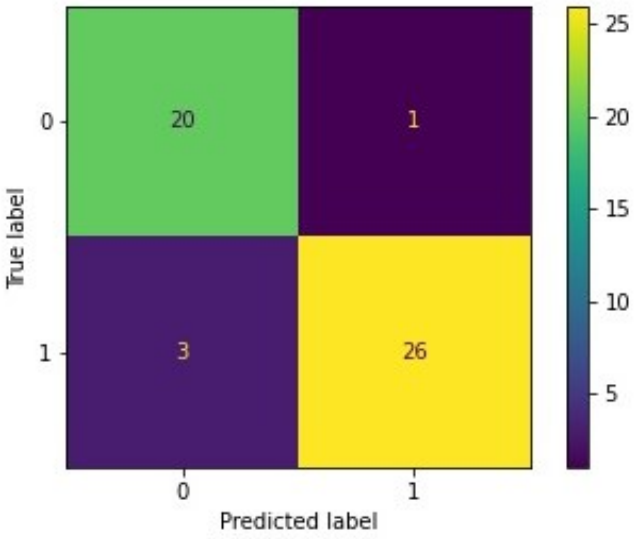
3.3.5 La Linear Discriminant Analysis

L'ultimo modello in analisi, l'LDA, ha riportato quelli che sono gli esiti principali per lo studio del caso in questione. Molto simile alla Principal Component Analysis svolge quindi doppio compito dove si va sia a ridurre la dimensionalità del dataset sia a prevedere le categorie a cui appartengono le osservazioni.

```
from sklearn.discriminant_analysis import LinearDiscriminantAna
lysis
classifier = LinearDiscriminantAnalysis()
model = classifier.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
cm = confusion_matrix(y_test, y_pred, labels=model.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
                              display_labels=model.classes_)
disp.plot()
plt.show()
```

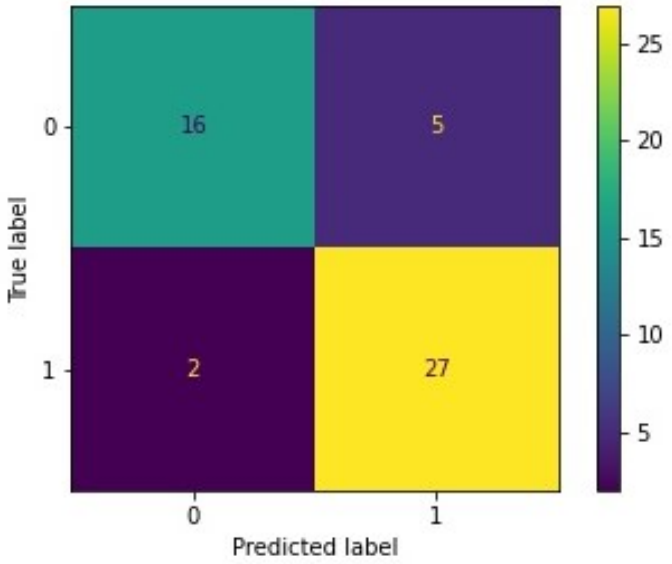
	precision	recall	f1-score	support
0	0.87	0.95	0.91	21
1	0.96	0.90	0.93	29
accuracy			0.92	50
macro avg	0.92	0.92	0.92	50
weighted avg	0.92	0.92	0.92	50



Otteniamo questa volta dei risultati prossimi alla perfezione. Il modello riesce a categorizzare benissimo le osservazioni, commettendo solo in un'occasione un errore grave per le finanze di chi lo utilizza e 3 volte un errore meno grave. Stima invece correttamente la categoria di appartenenza di 20 osservazioni negative su 21 totali e di 26 osservazioni positive su 29 totali. Porta quindi il TPR al 90% e il TNR ad un risultato pari al 95%.

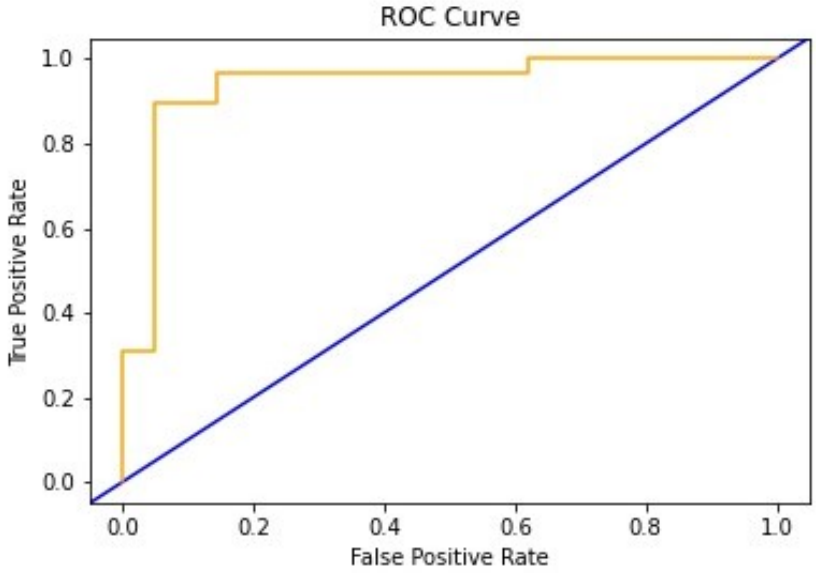
Osserviamo ora cosa succederebbe se applicassimo questo stesso modello ad un dataset privo del Sentiment degli utenti.

	precision	recall	f1-score	support
0	0.89	0.76	0.82	21
1	0.84	0.93	0.89	29
accuracy			0.86	50
macro avg	0.87	0.85	0.85	50
weighted avg	0.86	0.86	0.86	50



Anche nel caso di assenza del Sentiment notiamo come i risultati ottenuti siano ottimi ma non sufficienti a superare quelli ottenuti con la stessa variabile. Risultano esservi meno errori di primo tipo ma più errori gravi, del secondo tipo, prevedendo un esito positivo quando in realtà sarebbe stato negativo ben cinque volte. Il True Positive Rate sale dunque leggermente fino ad arrivare al 93% ma scende drasticamente il True Negative Rate fino al 76% rispetto al 95 ottenuto in presenza del Sentiment, facendo dunque notare come l'impatto della Sentiment Analysis in questo caso studio sia particolarmente rilevante in tutti i modelli presi in considerazione dato che vi è stato in ogni caso un miglioramento delle prestazioni passando da un modello addestrato senza la variabile ad uno addestrato su un dateset con la stessa. Viene riportata, infine, la Receiver Operating Charateristic Curve stimata sul primo dataset.

```
y_pred_proba = model.predict_proba(X_test)[::,1]
fpr, tpr, _ = roc_curve(y_test,  y_pred_proba)
fig, ax = plt.subplots()
line = mlines.Line2D([0, 1], [0, 1], color='blue')
transform = ax.transAxes
line.set_transform(transform)
ax.add_line(line)
plt.plot(fpr,tpr, color = 'orange')
plt.title('ROC Curve')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



Come anticipato nel Capitolo Due del lavoro, la ROC Curve è tanto migliore quanto più il vertice della linea rappresentante il rapporto tra TPR e FPR tende al punto di coordinate (0,1). In questo caso la curva è particolarmente vicina a quel punto riuscendo quindi a farci comprendere ulteriormente quanto il modello sia accurato. L'AUC, Area Under the Curve, risulta essere pari a 0.94088 che, secondo i criteri di Swets, rende il modello scelto categorizzabile con "Altamente informativo".

CAPITOLO QUATTRO

CONCLUSIONI

SOMMARIO: 4.1 Conclusioni; 4.2 Sviluppi e lavori futuri

4.1 Conclusioni

Lo scopo che questo lavoro si prefiggeva era quello di andare a valutare l'incidenza della Sentiment Analysis nel momento in cui si andava a rapportare con un dataset di tipo finanziario. I risultati ottenuti sono particolarmente buoni e fanno ben sperare per quanto riguarda l'applicazione futura di questa sottocategoria del Natural Language Processing a più ambiti della vita quotidiana. Si riportano di seguito due tabelle con i risultati dei 5 modelli utilizzati in termini di precisione media e precisione pesata per quanto riguarda lo score F1, rispettivamente applicati sul dataset contenente la variabile Sentiment e sul dataset privo di quest'ultima.

TABELLA RISULTATI DATASET CON SENTIMENT

Modello	Accuratezza Media	Accuratezza Pesata
Logistic Regression	74%	75%
Random Forest	62%	62%
Naive Bayes	59%	60%
SVM	59%	61%
LDA	92%	92%

TABELLA RISULTATI DATASET SENZA SENTIMENT

Modello	Accuratezza Media	Accuratezza Pesata
Logistic Regression	52%	55%
Random Forest	63%	64%
Naive Bayes	41%	44%
SVM	39%	43%
LDA	85%	86%

È chiaro come l'aggiunta della variabile stimata con TextBlob sia stata fondamentale per il lavoro e per le previsioni che ne sono scaturite, è osservabile come ogni modello, con l'aggiunta della stessa, sia riuscita a stimare meglio la categoria di appartenenza delle osservazioni. L'unico caso in cui ciò non è avvenuto è stato il modello Random Forest che ha riportato, nello score F1, un'accuratezza migliore rispetto al dataset col Sentiment ma che comunque ha prodotto previsioni non adeguatamente accurate.

4.2 Sviluppi e lavori futuri

Nell'ambito del Natural Language Processing e della Finanza vi saranno sicuramente degli sviluppi che verteranno a produrre stime e previsioni sempre più precise di più titoli

compreso quello studiato all'interno di questo lavoro. Grazie allo sviluppo del Deep Learning vi sono già a disposizione modelli più accurati per la stima del sentiment dei tweets e dei testi più in generale, un esempio su tutti è la Rete Neurale RoBERTa, ideata dalla società Meta, la quale ha raggiunto risultati ottimi sulla Sentiment Analysis oppure, nell'ambito finanziario, vi è il FinBERT, meno giovane ma comunque efficiente. Entrambi si basano sulla Rete Neurale BERT concepita da Google nel 2016 e potrebbero portare dei miglioramenti nell'ambito di ricerca di questa Tesi. Questi modelli, utilizzando Reti Neurali particolarmente complesse e cpu-intensive, non hanno trovato spazio in questa Tesi a causa della dimensione del dataset di tweets, che contava circa 65'000 osservazioni, e della loro stessa complessità ma potrebbero trovar spazio in lavori futuri, prendendo in considerazione altri titoli o intervalli temporali più ampi, in modo tale da ricercare quale sia il miglior ambito di applicazione del Natural Language Processing e, più nello specifico, della Sentiment Analysis per i dati finanziari.

BIBLIOGRAFIA E SITOGRAFIA

1. Karen Spärck Jones, 2001, Natural Language Processing: A Historical Review.
2. Koch, History of Machine Learning – A Journey through the Timeline.
3. Jurafsky, Martin, 2020, Speech and Language Processing.
4. Greene, 2012, Econometric Analysis.
5. Tin Kam Ho, 1995, Random Decision Forest
6. Albon, 2018, Machine Learning with Python Cookbook.
7. Bradley, 1997, The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms.
8. Mäntylä,Graziotin, Kuutila, 2016, The Evolution of Sentiment Analysis - A Review of Research Topics, Venues, and Top Cited Papers.
9. Mylavarapu, 2021, News based prediction of Stock price
10. Goel, Mittal, 2011 ,Stock Prediction Using Twitter Sentiment Analysis
11. www.datareportal.com/social-media-users
12. www.borsaitaliana.it/notizie/sotto-la-lente/sp500.htm
13. www.marketscreener.com/quotazioni/indice/S-P-500-4985/composizione
14. www.trends.google.it
15. www.github.com/sloria/TextBlob
16. www.machinelearningmastery.com/linear-discriminant-analysis-for-machine-learning
17. www.github.com/JustAnotherArchivist/snscrape
18. www.wikipedia.com
19. www.scikit-learn.org
20. https://www.cnbc.com/2021/09/20/stocks-see-worst-day-in-months-four-experts-share-their-strategies.html

RINGRAZIAMENTI

Al termine del mio percorso di studi mi sembra doveroso citare alcune delle persone che più mi hanno aiutato nel corso di questi anni. Parto innanzitutto col ringraziare la Professoressa Alessandra Amendola, relatrice di questo lavoro e della mia Tesi di Laurea Triennale, per essere stata sempre disponibile e paziente e per avermi fatto scoprire la mia passione per la Statistica negli anni passati durante i tre Esami sostenuti con lei.

Ringrazio mia mamma e mio padre, sempre presenti anche nelle situazioni più complicate e sempre pronti a darmi un consiglio quando necessario.

Ringrazio i miei fratelli, Serena e Agostino, anche se distanti fisicamente sempre pronti a darmi una mano sia materialmente che moralmente.

Ringrazio Concetta per aver supportato tutte le scelte fatte negli anni e per aver avuto sempre un consiglio o una parola di conforto, anche nei momenti più bui.

Ringrazio Martina, che ha dovuto subire tutte le mie preoccupazioni e paranoie per più di tre anni, che mi ha supportato e sopportato dandomi sempre la forza di continuare a credere in me e nelle mie potenzialità.

Ringrazio Orlando, Elia e Domenico, amici di una vita, per essermi stato vicino nei momenti in cui era più difficile farlo e per sopportare il mio caratteraccio.

Ringrazio Dalila, una sorella acquisita, per i consigli, gli aiuti e perché è anche grazie a lei che ho imparato i miei metodi di studio e, riprendo dai ringraziamenti della Tesi di Laurea Triennale, per sopportare Agostino.

Ringrazio Giovanni, un fratello acquisito, per esser stato vicino a me e alla mia famiglia negli anni più duri e, anche qui riprendo dai ringraziamenti della Laurea Triennale, per sopportare mia sorella.

Ringrazio Giorgio, amico trovato grazie all'inizio del mio percorso di studi, per aver condiviso con me questo cammino lungo il quale abbiamo avuto modo di legare sempre di più.

Ringrazio Giovanni, amico con tante passioni in comune, per credere sempre in me e soprattutto perché, prima o poi, avremo quella villa tanto sognata.

Ringrazio Chiara, che anche con i nostri diverbi, a modo suo, ha sempre fatto in modo di esserci nei momenti di bisogno.

Ringrazio Peppe, Gerardo, Serena, Carmine, Simone e tutti quelli che sono stati presenti in questi anni, è anche grazie a voi che sono qui dove sono.