

**Project Work
Anno 2023**

STOCK PRICE PREDICTION

*Master Data Science,
Big Data & Artificial Intelligence per la Finanza*



Indice

01

Creazione &
Presentazione
del TEAM



02

Scelta del tema
Project Work



03

Import & Analisi
Esplorativa dei DB
Stock e Tweet



04

Sentiment Analysis
dei Tweet e statistiche
a contorno



05

Modello Statistico
predittivo sui prezzi
finali dello Stock



Chi siamo:

01

Creazione &
Presentazione
del TEAM



Nicola Savastio



Chiara Boris



Michela Minniti



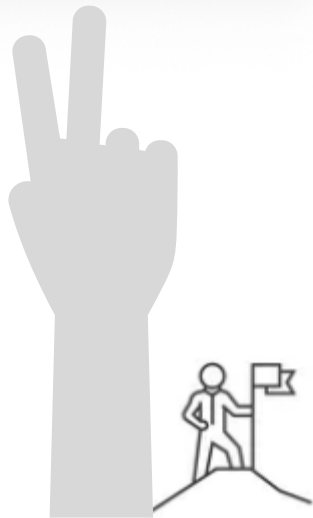
ChatGPT

The Automation

Stock price prediction & tweet

02

Scelta del tema
Project Work



L'analisi dei **Social Network** può concretamente fornire informazioni utili ed influenzare l'andamento dei mercati



La sfida del progetto è estrapolare diversi dataset dei titoli e combinarli con le informazioni dei Social Network per capire se c'è **correlazione** tra questi due argomenti e predire il prezzo finale del titolo



Testare gli algoritmi di **Machine Learning** trattati a lezione

Metodologia

02

Scelta del tema
Project Work



Step del progetto:

1. Preparazione dei dati
2. Analisi dei tweet giornalieri e creazione delle caratteristiche più rilevanti per la predizione
3. Divisione dei dati in due sottoinsiemi: uno per il training e uno per il test
4. Scelta dell'algoritmo di predizione: regressione lineare
5. Addestramento del modello con i dati di training, includendo le caratteristiche estratte dai tweet
6. Valutazione del modello calcolando l'errore medio assoluto
7. Rappresentazione grafica dei risultati confrontando i prezzi reali delle azioni con le predizioni del modello

Dataset Stock :

1. Crea una funzione che legge tutti i dataset nella cartella di rete
2. Crea un algoritmo che accoda tutti i CSV in unica tabella
3. Analisi esplorativa del dataset Stock

03

Import & Analisi
Esplorativa dei DB
Stock e Tweet


```
1 #importo le librerie utili
2
3 import pandas as pd
4 import os
5 import json
6 from google.colab import drive
7 from tqdm.auto import tqdm #serve per avere la barra di avanzamento nei loop
```



Osservazioni da Settembre 2012 a Settembre 2017

Row x Column (108592, 8) Stock_Name: 88

Out[4]:



	date	open	high	low	close	adj close	volume	stock_name
0	2012-09-04	95.108574	96.448570	94.928574	96.424286	87.121140	91973000.0	AAPL
1	2012-09-05	96.510002	96.621429	95.657143	95.747147	86.509338	84093800.0	AAPL
2	2012-09-06	96.167145	96.898575	95.828575	96.610001	87.288956	97799100.0	AAPL
3	2012-09-07	96.864288	97.497147	96.538574	97.205711	87.827171	82416600.0	AAPL
4	2012-09-10	97.207146	97.612854	94.585716	94.677139	85.542564	121999500.0	AAPL

trasformata la variabile date da testo a formato data



03

Import & Analisi
Esplorativa dei DB
Stock e Tweet



Dataset Tweet :

1. Crea una funzione che legge tutti i dataset nella cartella di rete
2. Crea un algoritmo che accoda tutti i CSV in unica tabella

```
16 # Define a function to get all the files in a folder and its subfolders
17 def get_files(path):
18     # Initialize an empty list to store the files
19     files = []
20     # Loop over all the files and folders in the given path
21     for file in os.listdir(path):
22         # If the file or folder is a file, add it to the list of files
23         if os.path.isfile(os.path.join(path, file)):
24             files.append(os.path.join(path, file))
25         # If the file or folder is a folder, call the function recursively on its path and add the results to the list of files
26         elif os.path.isdir(os.path.join(path, file)):
27             files.extend(get_files(os.path.join(path, file)))
28     # Return the list of files as output
29     return files
30
31
32 def read_json(file):
33     df = pd.read_json(
34         file, orient='records', lines=True
35     )["created_at", "user", "text"]
36     # Aggiungi due nuove colonne con il nome dello stock e la data, ottenuti dal percorso del file
37     stock_name = os.path.basename(os.path.dirname(file))
38     date = os.path.splitext(os.path.basename(file))[0]
39     df["stock_name"] = stock_name
40     df["date"] = date
41     df["user"] = df["user"].apply(lambda x: x["id"])
42     return df
43
44 def read_csv(file):
45     df = pd.read_csv(
46         file
47     )
48     stock_name = os.path.splitext(os.path.basename(file))[0]
49     df["stock_name"] = stock_name
```

Dataset Tweet :

3. Analisi Esplorativa dei Tweet

[285740 rows x 8 columns]

03

Import & Analisi
Esplorativa dei DB
Stock e Tweet



Osservazioni da Gennaio 2014 a Marzo 2016



	created_at	user	text
0	2015-06-27 19:41:36+00:00	2181326676	HYG iShares iBoxx High Yield Corporate Bond Fu...
1	2015-11-21 13:02:03+00:00	2881595931	\$C \$ODP \$BRCD:\n\nStocks With The Highest Dail...
2	2014-04-17 16:38:13+00:00	221896341	P\$C was the greatest rap group
3	2014-04-17 14:30:21+00:00	1839878976	\$C Barclays and Citigroup Funded Worst of U.S....
4	2014-04-17 03:13:30+00:00	300802438	RT @Fball_Is_MyDrug: "@OPM1995: im on this F\$C...

trasformata la variabile
created_at da testo a
formato data

La variabile text è stata ripulita con le seguenti funzioni:

- se presenti slang vengono ricondotti in parole
- se presenti emoji ed emoticon vengono ricondotti in parole
- rimossi eventuali spazi multipli o caratteri speciali usando le espressioni regolari
- rimossi le interruzioni di riga
- restituzione del testo normalizzato



Sentiment Analysis:

Utilizzata la funzione di `predict_sentiment`

che attribuisce per ogni riga del tweet il **valore di Logit più alto** e successivamente etichettato come **Negative, Neutral, Positive**

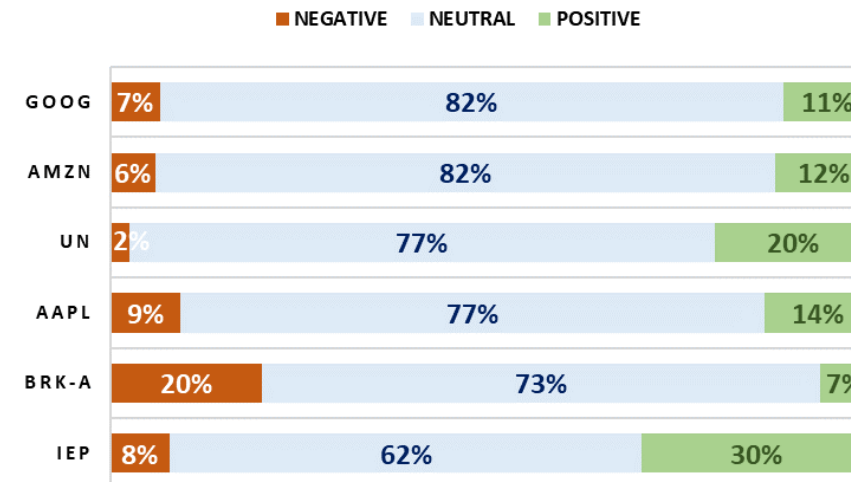
```
31 def predict_sentiment(text):
32     # Tokenizza il testo usando il tokenizer
33     encoded_input = tokenizer(text, return_tensors='pt')
34     encoded_input = encoded_input.to(device)
35     # Fai la predizione usando il modello
36     output = model(**encoded_input)
37     # Ottieni il valore della logit più alta
38     scores = output[0][0].detach()
39     scores = scores.cpu().numpy()
40     scores = np.exp(scores) / np.exp(scores).sum(-1)
41     # Mappa i valori delle logit alle etichette del sentimento
42     labels = ['NEGATIVE', 'NEUTRAL', 'POSITIVE', 'MIXED']
43     ranking = np.argsort(scores)
44     ranking = ranking[::-1]
45     # Restituisci l'etichetta e lo score del sentimento più probabile
46     return (labels[ranking[0]], scores[ranking[0]])
```

Analisi Esplorativa per alcuni Stock:

```
data.groupby("stock_name").date.agg(["min", "max", "median", "count"])
```

	min	max	median	count
stock_name				
AAPL	2014-01-01	2016-03-31	2015-03-09	20850
ABB	2014-02-11	2016-03-31	2015-09-04	101
ABBV	2014-01-03	2016-03-28	2015-03-05	889
AEP	2014-01-02	2016-03-31	2014-07-25	223
AGFS	2015-08-18	2016-03-30	2016-03-12	8
...
V	2014-01-01	2016-03-31	2015-05-08	1376
VZ	2014-01-03	2016-03-31	2015-04-25	1158
WFC	2014-01-02	2016-03-31	2015-04-09	1076
WMT	2014-01-02	2016-03-31	2015-06-11	1761
XOM	2014-01-02	2016-03-31	2015-05-14	1903

87 rows × 4 columns



Scelta del modello predittivo:

05

Modello Statistico
predittivo sui prezzi
finali dello Stock



Regressione Lineare

Per predire i prezzi di serie storiche finanziarie, un algoritmo comunemente utilizzato e adatto a questo tipo di problema è l'algoritmo di **regressione lineare**.

La regressione lineare è un algoritmo di apprendimento **supervisionato** che cerca di stabilire una relazione **lineare** tra le variabili di input e la variabile target

Motivo della scelta:

05

Modello Statistico
predittivo sui prezzi
finali dello Stock



➤ **Semplicità:** da implementare e interpretare, soprattutto nel caso di una variabile target continua come i prezzi delle azioni

➤ **Interpretabilità:** fornisce coefficienti che indicano la relazione e l'importanza relativa delle variabili di input

➤ **Adattabilità ai dati:** può essere applicata a una vasta gamma di dati finanziari

➤ **Performance:** nonostante la sua semplicità, può fornire risultati accurati, specialmente quando si considerano più variabili di input e vengono gestiti adeguatamente gli aspetti temporali

Cosa Abbiamo fatto:

Dopo le dovute ottimizzazioni e pulizie dei duplicati, **otteniamo un DataFrame degli Stock Price e della Sentiment Polarity dei Tweet raggruppati per Data, Stock Name**

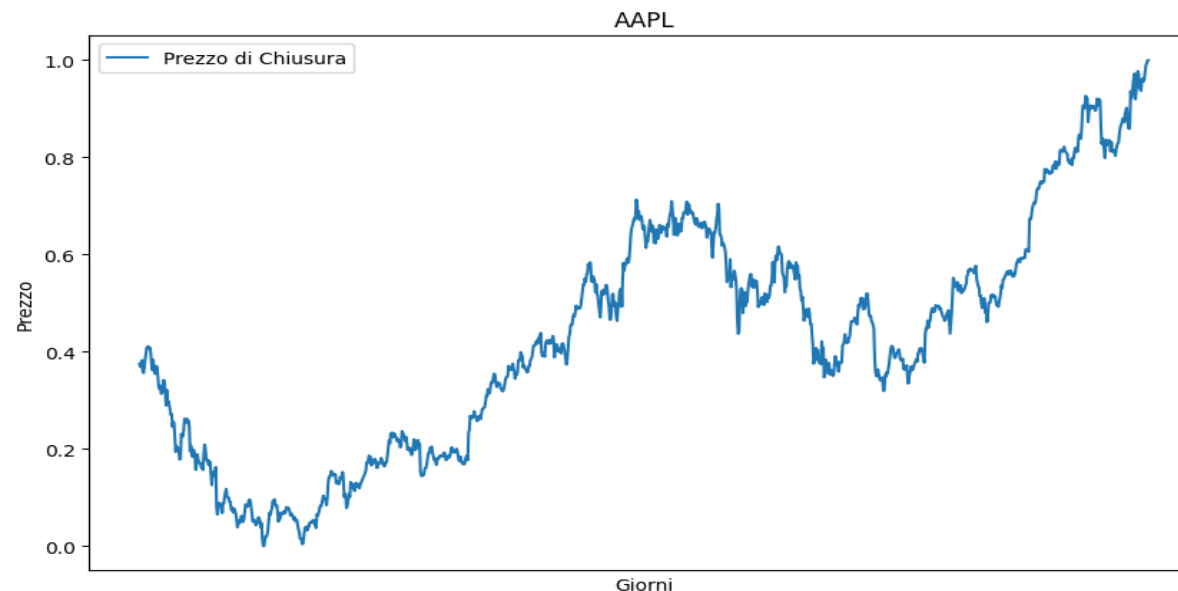
05

Modello Statistico
predittivo sui prezzi
finali dello Stock



Si è deciso di prendere come campione lo **stock AAPL**

```
1 #Guardo graficamente l'andamento dei prezzi del df
2
3 import matplotlib.pyplot as plt
4
5 # Crea un grafico della serie storica dei prezzi
6 plt.figure(figsize=(10, 6)) #creare figura dimensioni specificate. figsize dimensioni in pollici
7
8 #plot = disegnare punti in un diagramma collegati da una riga
9 plt.plot(df_stock['Date'], df_stock['Close_norm'], label='Prezzo di Chiusura')
10 plt.xlabel('Giorni')
11 plt.ylabel('Prezzo')
12 plt.title(desired_stock_name)
13 plt.legend() #aggiungere legenda
14 plt.xticks([]) #imposta etichette come vuoto, quindi non verranno visualizzate
15 plt.show()
```



Cosa Abbiamo fatto:

Successivamente, suddiviso
la nostra basedati in
Training e Test (20%)

05

Modello Statistico
predittivo sui prezzi
finali dello Stock



```
# Suddivido i dati in training e test (test = 20%)

from sklearn.model_selection import train_test_split
from nltk.sentiment import SentimentIntensityAnalyzer

df_train, df_test = train_test_split(df_merged, test_size=0.2, shuffle=False)

# Verifica che la data minima nell'insieme di test sia successiva alla data massima nell'insieme di training
if df_test['Date'].min() <= df_train['Date'].max():
    raise ValueError("La data minima nell'insieme di test non è successiva alla data massima nell'insieme di training")

print('duplicati: ' + str(df_merged['Date'].duplicated().sum())) # Verifica valori duplicati
```

Addestrato il modello e calcolato il **MSE**

```
[ ] #Utilizzo un modello di regressione lineare

import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
from nltk.sentiment import SentimentIntensityAnalyzer

# Esegui lo shift delle colonne Close
df_train['Close_norm'] = df_train['Close_norm'].shift(-1)
df_test['Close_norm'] = df_test['Close_norm'].shift(-1)

# Rimuovi l'ultima riga che contiene valori NaN
df_train.dropna(subset=['Close_norm'], inplace=True)
df_test.dropna(subset=['Close_norm'], inplace=True)

# Prepara variabili input e output
x_train = df_train[['Open', 'High', 'Low', 'Volume', 'Sentiment_Score']]
y_train = df_train['Close_norm']
x_test = df_test[['Open', 'High', 'Low', 'Volume', 'Sentiment_Score']]
y_test = df_test['Close_norm']

# Addestra il modello di regressione lineare
regression = LinearRegression()
regression.fit(x_train, y_train)

# Prevedi i valori di chiusura
y_pred = regression.predict(x_test)

# Calcola l'errore quadratico medio
mse = mean_squared_error(y_test, y_pred)

print("MSE:", mse)

MSE: 0.00022303625935109837
```

Utilizzata una **Left Join** dove considera
il perimetro temporale **Stock AAPL** ed
ipotizzando **che i valori mancanti dei
Tweet abbiamo uno score Neutral**

Calcolato il **MSE** anche per altri Stock

1 #Basic Materials	XOM	Exxon Mobil Corporation	MSE = 0.0005048741274048334
2 #Consumer Goods	AAPL	Apple Inc.	MSE = 0.00022303625935109837
3 #Healthcare	JNJ	Johnson & Johnson	MSE = 0.00020550425887674878
4 #Services	AMZN	Amazon.com Inc.	MSE = 0.00017356990206954763
5 #Utilities	NEE	NextEra Energy Inc.	MSE = 0.00022565583981202765
6 #Conglomerates	IEP	Icahn Enterprises L.P.	MSE = 8.291904066754802e-05
7 #Financial	BCH	Banco de Chile	MSE = 0.00037204290926620476
8 #Industrial Goods	GE	General Electric Company	MSE = 0.0004901901074632877
9 #Technology	GOOG	Alphabet Inc.	MSE = 0.00018952104312409387

Cosa Abbiamo fatto:

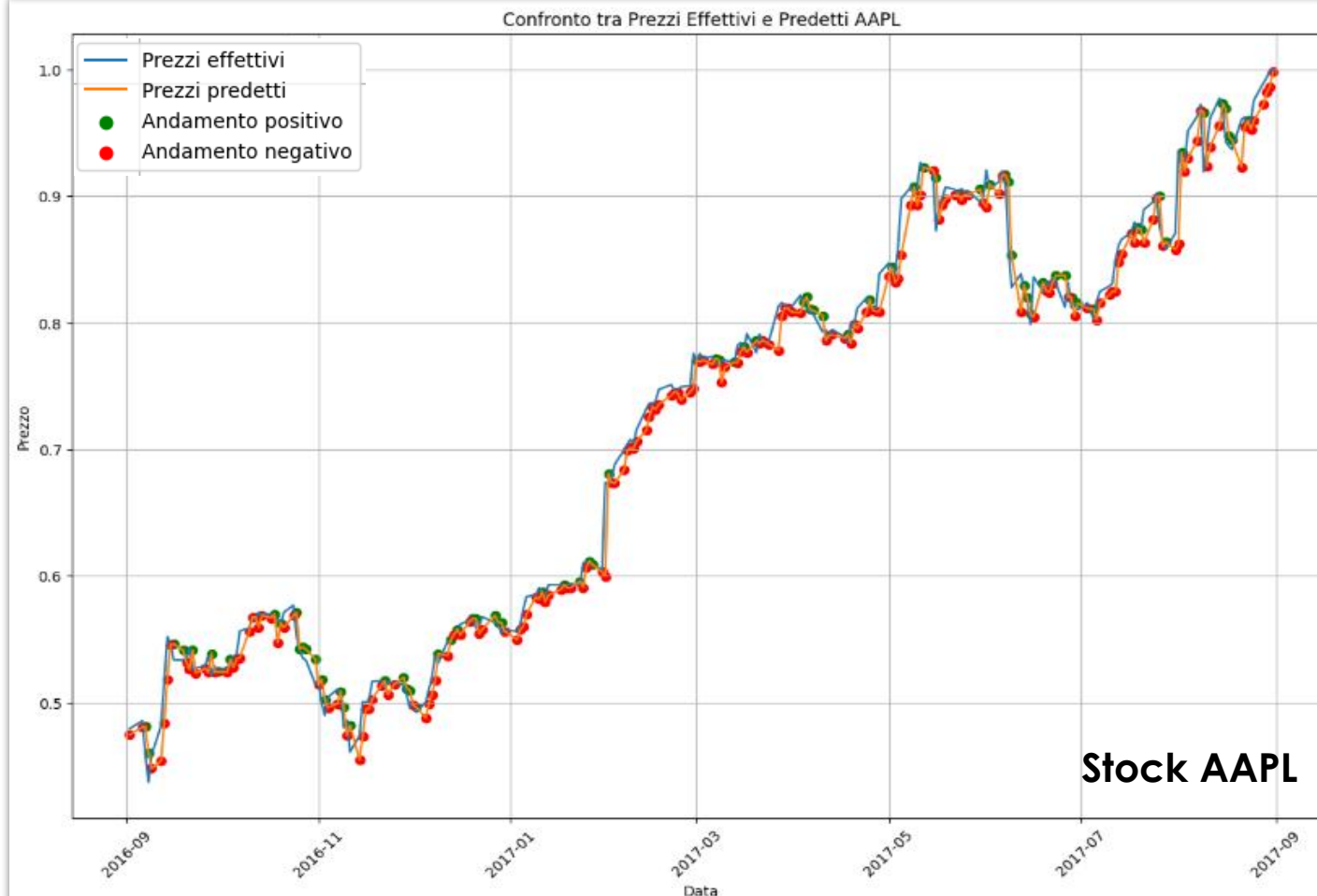
05

Modello Statistico
predittivo sui prezzi
finali dello Stock



```
# Aggiungi punti verdi per l'andamento positivo
plt.scatter(df_test['Date'][price_change > 0], y_pred[price_change > 0], color='green', label='Andamento positivo')

# Aggiungi punti rossi per l'andamento negativo
plt.scatter(df_test['Date'][price_change < 0], y_pred[price_change < 0], color='red', label='Andamento negativo')
```



Si osserva un
andamento
correlato tra
i **prezzi**
predetti e i
Sentiment
Positivi e
Negativi

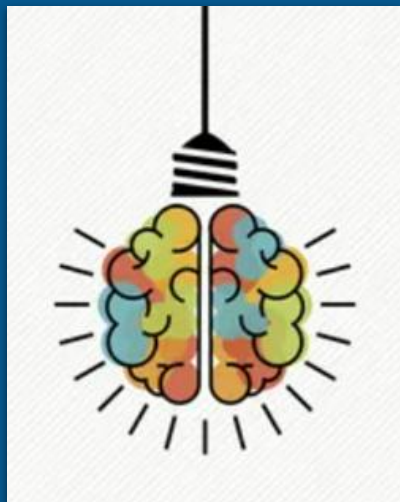
Conclusioni e Nuove Idee:



Il **MSE** ha un valore basso ciò significa che è presente una bassa volatilità nei dati ed il modello fitta bene



Questa tipologia di **modello**, dove si incrociano le previsioni dei prezzi con la Sentiment Polarity, **sono efficaci solamente se non è presente una spiccata volatilità nel trend dei prezzi**



New Idea

Replicare il modello per tutti gli Stock

Fare Challenging sul periodo osservato

(non solo left join ma magari filtri sulle date)

Utilizzo di altre metodologie per gestire gli NA dei Tweet mancanti

(imputare la media e ri-clusterizzare il Sentiment)

Utilizzare il modello XGBoost

(algoritmo di machine learning spesso utilizzato quando si utilizzano dati testuali, immagini e altri dati non strutturati)

Utilizzare una Rete Neurale

(complicato in quanto si dovrebbe capire quanti Neuroni inserire, quali funzioni di attivazione utilizzare, quanti Hidden Layer da inserire ed il conteggio delle epoche significativo per ottimizzare l'errore)