PROGETTO MACHINE LEARNING

Analisi e predizione

Manuel Tanzi - 307720

E-mail: manuel.tanzi1@studenti.unipr.it

Progetto GitLab:
https://gitlab.com/tanzone2000/machine-learning.git
Dataset Riferimento:
https://www.kaggle.com/mattiuzc/stock-exchange-data

UNIVERSITÀ
DI PARMA

Introduzione

Argomento

Si utilizza un Dataset composto dai dati generati dalle API di yahoo-finance che fornisce notizie finanziarie, dati e commenti, tra cui le quotazioni in borsa.
I dati forniti interessano il rendimento di diversi anni di andamento di alcuni titoli e azioni, in particolare del valore delle monete di vari Paesi divisa per simbolo di appartenenza.
Si vuole quindi avere una rappresentazione chiara dell'andamento, accompagnata da alcune informazioni necessarie e utili per l'analisi delle valute.
Inoltre, si cerca di eseguire un'analisi predittiva tramite alcuni degli algoritmi di machine learning visti a lezione implementate attraverso diverse tecniche e strategie.

Procedimento

Si utilizza una scaletta prefissata per la realizzazione del progetto:

1. Raccolta dei dati
2. Esplorazioni dati
3. Data cleaning
4. Feature selection
5. Feature engineering
6. Feature scaling
7. Costruzione modelli
8. Confronto modelli
9. Analisi prestazioni

Si è deciso di non utilizzare algoritmi come T-sne e PCA poiché per questo tipo di problema risulta utile semplicemente una visualizzazione a grafico XY senza preoccuparsi di Curse of Dimensionality.

Sviluppo

Durante lo sviluppo vengono eseguiti molteplici casi, dai più irreali ad una possibile condizione realistica in cui si prova a predire un certo numero di giorni futuri stabiliti da una costante presente nel file apposito (constants.py).
La maggior parte delle funzioni sono generalizzate e parametrizzabili in maniera tale da avere una maggior libertà per simulare i vari modelli attraverso più combinazioni possibili e sperimentare.

Considerazioni

Il sistema da analizzare è un processo aleatorio non markoviano ovvero che l'istante al tempo t non dipende dalla storia passata. Questo fa si che i modelli di Machine Learning non riconoscano un pattern ricorrente e quindi costruirne una funziona adeguata solamente utilizzando i dati forniti.

Ipotesi

Matematicamente, con un problema di regressione, si sta cercando di trovare l'approssimazione della funzione con la deviazione minima dell'errore.
L'analisi di regressione è il modello statistico che viene utilizzato per prevedere i dati numerici anziché le etichette ma può anche identificare il movimento di distribuzione in base ai dati disponibili o ai dati storici.

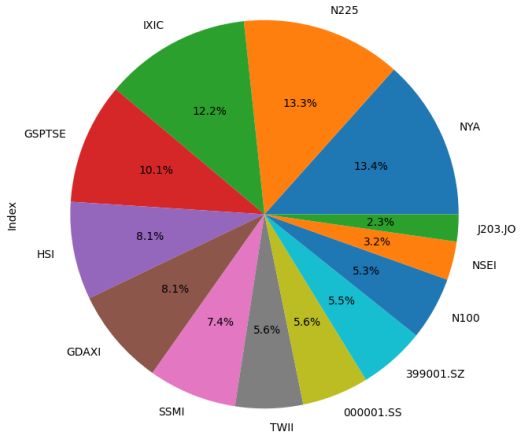
Si ipotizzi invece di aggiungere due features (BUY-TIME, SELL-TIME) caratterizzate da un valore identificativo [0, 1], calcolati secondo alcuni algoritmi che studiano l'andamento medio come SMA, EMA, STD, MEAN e CHANGE.
Si identifica quindi qual è il miglior momento per effettuare quella determinata azione di compra-vendita a seconda delle analisi ottenute con gli algoritmi definiti sopra.
A questo punto si potrebbe eseguire un task di classificazione per prevedere se nell'immediato futuro sia meglio eseguire un'azione di BUY o SELL o di STALLO.

Esplorazione Dati

Dataset

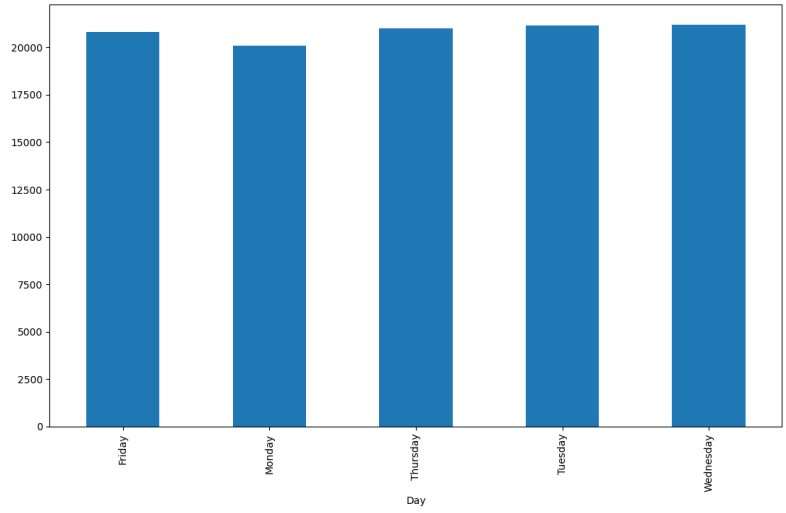
Si nota come all'interno del dataset siano presenti dei valori nulli dati dal fatto che in alcune giornate per alcuni simboli non sono stati registrati i dati azionari. Questo poteva essere risolto sostituendo i valori con mancanti con il valore media calcato tra il giorno precedente e il successivo.
Si è preferito invece cancellare completamente la riga.

Index	0
Date	0
Open	2204
High	2204
Low	2204
Close	2204
Adj Close	2204
Volume	2204



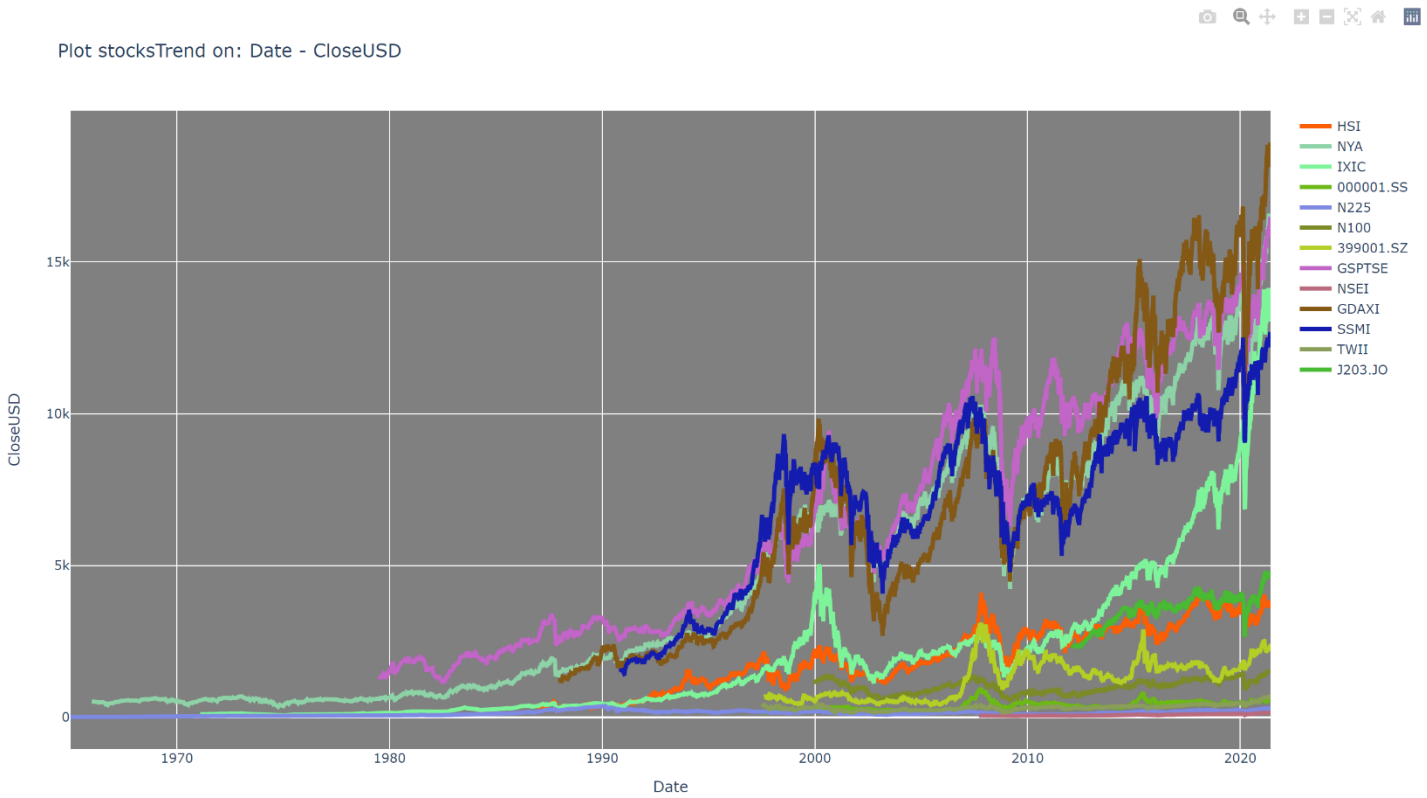
Attraverso il grafico a torta notiamo invece che nel dataset fornito si hanno i dati relativi a differenti simboli in quantità differenti si procede quindi ad una scrematura del dataset suddividendo in sotto-dataset ognuno col proprio file abbinato così da rendere le operazioni più semplici e leggibili.

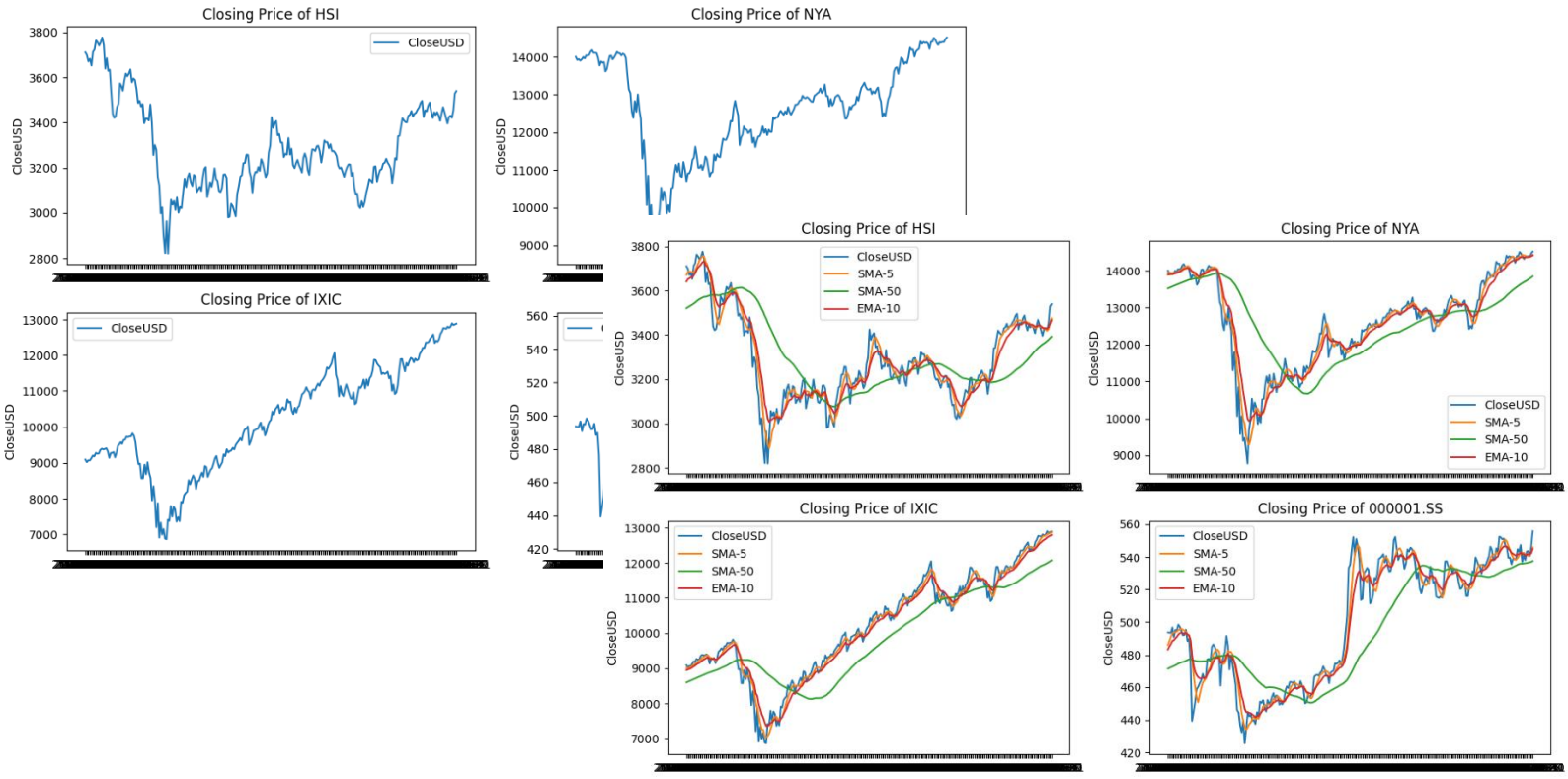
Il grafico a fianco indica invece i giorni della settimana in cui compare l'annotazione dei dati e si nota la mancanza del sabato e domenica poiché infatti il mercato azionario rimane chiuso e non accessibile.



Data visualization

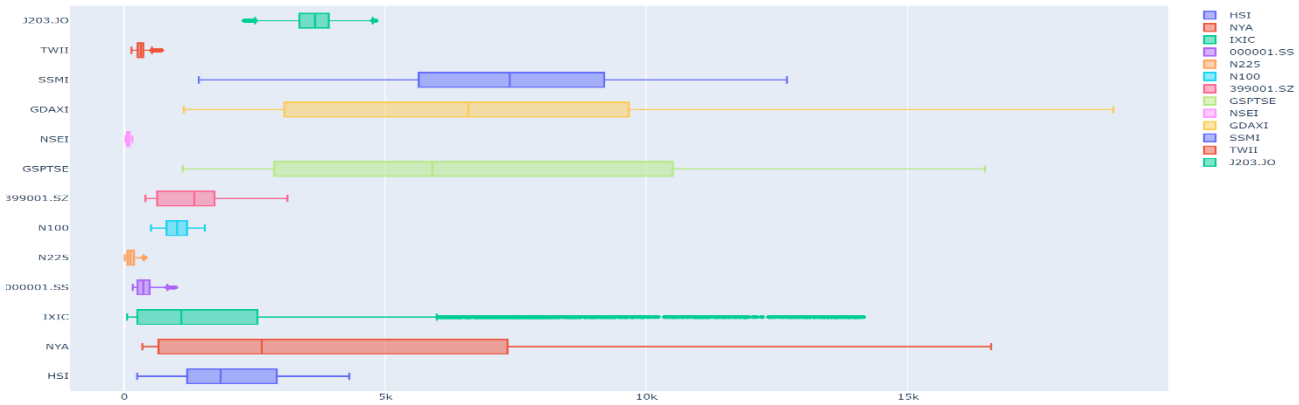
I grafici sottostanti mostrano alcuni dei possibili plot creabili nel progetto.
È stata utilizzata una libreria di supporto (Plotly) che è in grado di creare grafici sul web dinamici e personalizzabili in tempo reale come mostrato nel primo grafico sottostante che mette in evidenza tutti i trend dei simboli presenti all'interno del dataset.
Viene utilizzata poi la libreria MatPlotLib per la stampa di alcuni simboli affiancate anche ad alcune features extra calcolate separatamente.



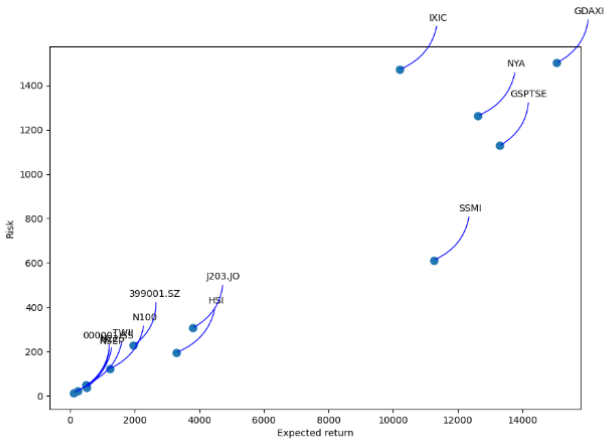


Data Analysis

Come prima operazione si procede per la visualizzazione dei valori outliers, che fanno quindi eccezione e creano particolari problemi di overfitting poi durante lo studio del modello.



Tra le varie analisi eseguite si effettua anche un'analisi del rischio per ogni simbolo tramite la loro volatilità e si ottiene il seguente grafico dalla quale possiamo intuirne anche il loro possibile andamento nel tempo o comunque avere una idea generale di quanto possa variare nel tempo una determinata azione.



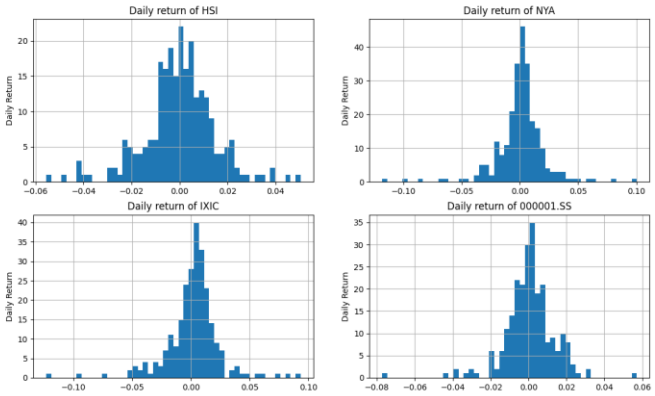

```
Augmented Dickey-Fuller Test:
ADF test statistic      -1.424204
p-value                  0.570619
# lags used             1.000000
# observations         412.000000
critical value (1%)     -3.446322
critical value (5%)     -2.868581
critical value (10%)    -2.570521
Weak evidence against the null hypothesis
Fail to reject the null hypothesis
Data has a unit root and is non-stationary
```

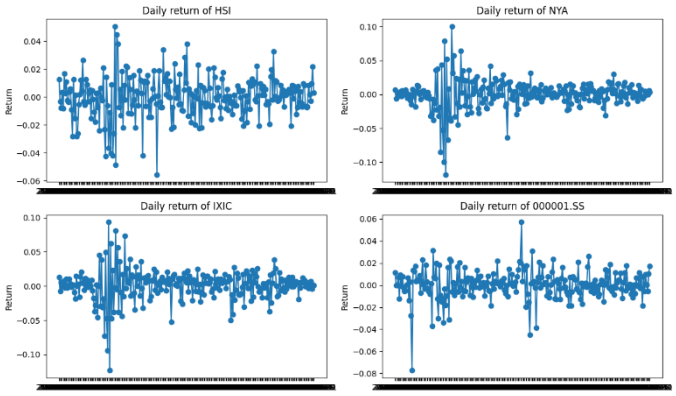
Si eseguono inoltre dei test di Augmented Dickey-Fuller per determinare se data una serie temporale è stazionaria ovvero una serie le cui proprietà non dipendono dal momento in cui la serie viene osservata.
Pertanto, le serie temporali con tendenza, o con stagionalità, non sono stazionarie
Una serie storica con comportamento ciclico (ma senza trend o stagionalità) è stazionaria.
Riporto a fianco il caso studiato per la stock "HSI" e si ottiene il seguente risultato (medesimo per tutti gli altri simboli presenti)

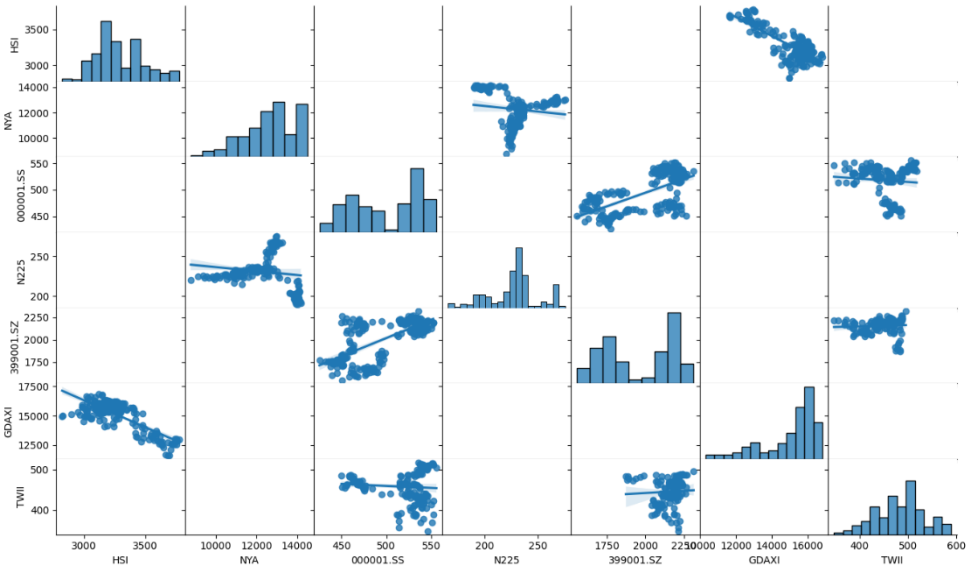
Infine, si nota che i valori monetari sono convertiti come valore della propria moneta, quindi, viene realizzata una funzione in grado di effettuare la conversione in qualsiasi valore monetario desiderato.
Si decide quindi di standardizzare le unità con la valuta € euro per le future analisi.

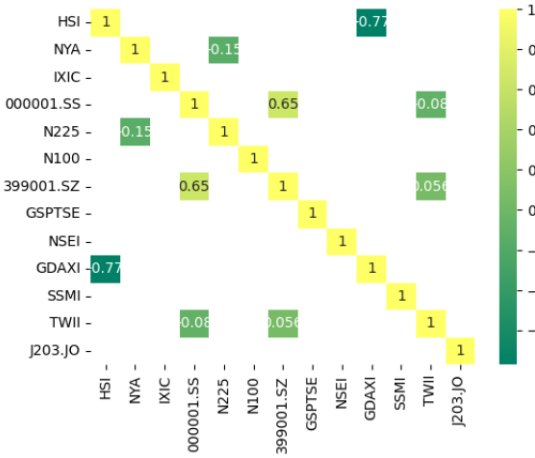
Extra data

Questi grafici sottostanti rappresentano il ritorno finanziario giornaliero, nei suoi termini più semplici, è il denaro guadagnato o perso su un investimento in un certo periodo di tempo.
Un rendimento può essere espresso nominalmente come la variazione del valore in dollari di un investimento nel tempo. Un rendimento può anche essere espresso come percentuale derivata dal rapporto tra profitto e investimento.









Pre processing

Feature engineering

Attraverso l'analisi svolta precedentemente ci si è accorti di alcuni dati mancanti che quindi sono stati presi in considerazione e sistemati cancellandoli dal dataset.

Viene inoltre cambiata la valuta del dataset che si prende in considerazione per l'analisi in € euro così da poter avere un'idea più chiara e rappresentativa.

Gli outliers decido invece di trascurarli in quanto il task non è particolarmente influenzato da quest'ultimi.

Feature Selection

Questa operazione per la tipologia di task svolto risulta pleonastica in quanto tra le mie feature possiedo solamente ["Date"]. Viene comunque eseguito poiché effettuo dei test di prova consapevole della incorrettezza, utilizzando diverse features che possono essere quelle presenti nel dataset:

["Low", "High" ...] - ["SMA-n", "EMA-n", "CHANGE-n" …].

Split Train e Test

Per simulare al meglio una situazione realistica decido di eseguire uno split del dataset di un singolo stock, in particolare "HSI".

Lo split viene eseguito ordinando il set per data e prendendo come

Train-set: [(l'intero dataset) – (n giorni futuri da predire)]

Test-set: [(n giorni futuri da predire)]

oppure

Train-set: [(l'intero dataset) – (% giorni futuri da predire)]

Test-set: [(% giorni da predire)]

oppure la funzione di split predisposta dalla libreria di sklearn.

Il primo split proposto è quello che si immedesima di più realisticamente.

Feature scaling

Si decide di strutturare il codice in maniera tale che si possano eseguire differenti tipi di scaling.

Prendo in considerazione:

- MinMaxScaler()
- MaxAbsScaler()
- StandardScaler()

Sono state eseguite quindi differenti approcci per trovare quello migliore e l'utilizzo mi permette di normalizzare il range di variazione delle caratteristiche (feature) del dataset.

In questo modo miglioro la qualità dei risultati finali.

```python
# Prepocessing di normalizzazione e standizzazione
def _preProcessing(X_train, X_test, preType: str = PRETYPE_MINMAX):
    if preType == PRETYPE_FALSE:
        return X_train, X_test, None

    scaler = MinMaxScaler(feature_range=(-1, 1))
    if preType == PRETYPE_MINMAX:
        scaler = MinMaxScaler(feature_range=(0, 1))

    if preType == PRETYPE_MAXABS:
        scaler = MaxAbsScaler()

    if preType == PRETYPE_SCALER:
        scaler = StandardScaler()

    xCols = X_train.columns
    X_train = pd.DataFrame(scaler.fit_transform(X_train), columns=xCols)
    X_test = pd.DataFrame(scaler.transform(X_test), columns=xCols)

    return X_train, X_test, scaler


# Postpocessing di normalizzazione e standizzazione
def _postProcessing(X_train, X_test, scaler, xCols, preType: str = PRETYPE_MINMAX):
    if preType == PRETYPE_FALSE:
        return X_train, X_test

    X_train = pd.DataFrame(scaler.inverse_transform(X_train), columns=xCols)
    X_test = pd.DataFrame(scaler.inverse_transform(X_test), columns=xCols)

    return X_train, X_test
```

Il migliore ottenuto è MinMaxScaler con range [-1, 1].

Costruzione modelli

Tipi di modello

Sono stati presi in considerazione differenti modelli di regressione al fine di calcolare una sequenza numerica che rispecchiasse l'andamento del trend di una stock specifica (come riferimento viene usata "HSI").

Nello specifico vengono utilizzati:

- LinearRegression
- RandomForestRegressor
- AdaBoostRegressor
- Rete neurale
- PolynomialFeatures
- LogisticRegression
- GradientBoostingRegressor
- **LSTM**

In particolare, il modello **LSTM** che non è stato introdotto a lezione, ma visto sul web e consigliato da un collega consente di elaborare intere sequenze di dati senza trattare ogni punto della sequenza in modo indipendente, ma piuttosto conservando informazioni utili sui dati precedenti nella sequenza per aiutare con l'elaborazione di sequenze di dati.

In ogni caso secondo il mio studio RidgeRegression risulta essere il migliore e riporto i dati ottenuti di alcuni dei modelli sottoposti a sperimentazioni.

Risultati ottenuti:

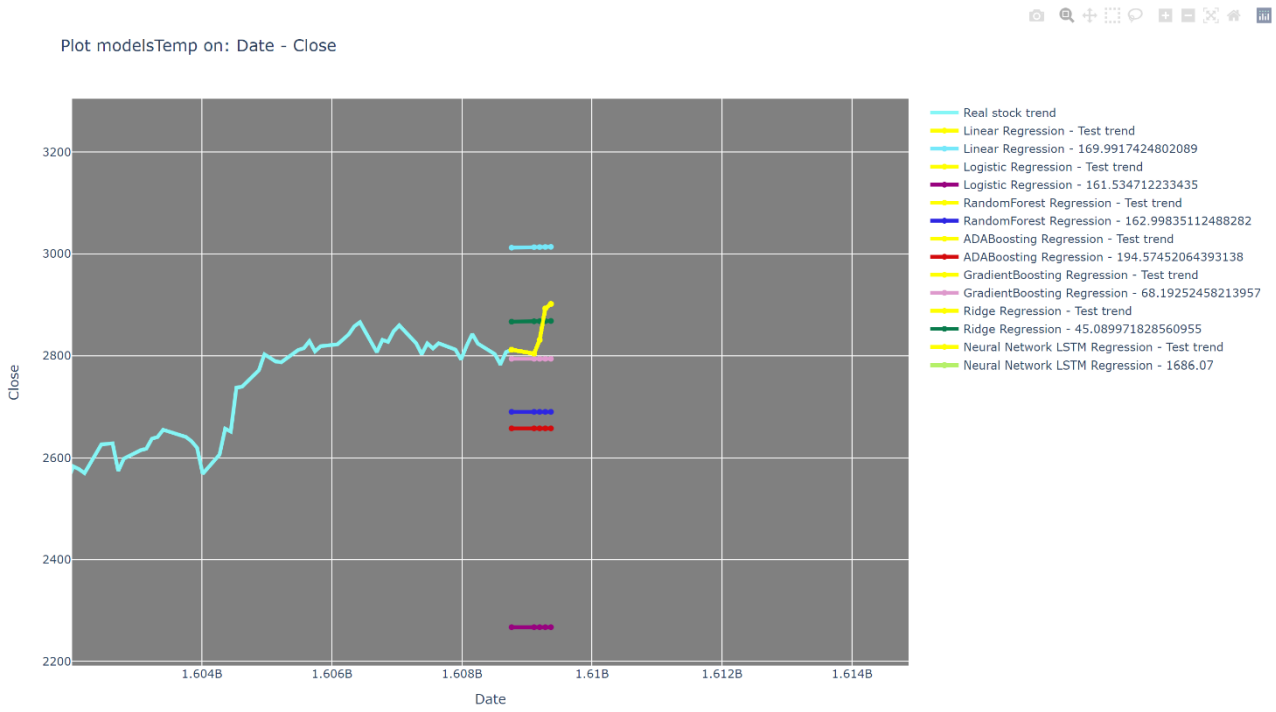
```
Type of Regression:      Linear Regression
MRSE:                    169.9917424802089
R2 Score:                 -16.15704833073024
CV Mean:                  -1.069616751932888
STD:                      1.1463860282045737
-----------------------------------------
Type of Regression:      Logistic Regression
MRSE:                    161.534712233435
R2 Score:                 -14.492398094646367
-----------------------------------------
Type of Regression:      RandomForest Regression
MRSE:                    162.99835112488282
R2 Score:                 -14.774418033186967
CV Mean:                  -2.5264590819787647
STD:                      2.960772802486996
-----------------------------------------
```

```
Type of Regression:      ADABoosting Regression
MRSE:                    194.57452064393138
R2 Score:                 -21.478061848167172
CV Mean:                  -3.5150970004322386
STD:                      3.0836997433283746
-----------------------------------------
Type of Regression:      GradientBoosting Regression
MRSE:                    68.19252458213957
R2 Score:                 -1.7609622032370815
CV Mean:                  -2.1222735144902516
STD:                      2.5889357050369686
-----------------------------------------
Type of Regression:      Ridge Regression
MRSE:                    45.089971828560955
R2 Score:                 -0.2071100101504464
CV Mean:                  -2.3806828574586967
STD:                      2.639675907758112
-----------------------------------------
```

Grafico delle predizioni

Riporto il grafico colle tendenze di alcuni dei modelli eseguiti, che predicono l'andamento nei futuri 5 giorni.

La linea gialla rappresenta il test-set e Ridge regression è quello che ne valuta la miglior prestazione.



Esecuzione dei modelli

```
# Split
X_train, y_train, X_test, y_test = _split(df[:], splitType, size)
xCols = X_train.columns
# BestFeatures
_bestFeatures(X_train, y_train.values.reshape(-1, ), bestType)
# PreProcessing
X_train, X_test, scaler = _preProcessing(X_train[:], X_test[:], preType)
# Cross Validation
_crossValidation(Ridge(), X_train, y_train, name, KFOLD_NUM, crossType)
# Random e Grid Search
best = BEST_RIDGE_REGRESSION
best = _randSearch(Ridge(), X_train, y_train.values.reshape(-1, ), RIDGE_REGRESSION_SPACE, randType, best)
best = _gridSearch(Ridge(), X_train, y_train.values.reshape(-1, ), best, gridType)
# Cross Validation
_crossValidation(Ridge(**best), X_train, y_train, name, KFOLD_NUM, crossType)
# Learning
model = Ridge(**best)
model.fit(X_train, y_train.values.reshape(-1,))
y_pred = model.predict(X_test)
rmse = _paramsErrors(model, X_train, y_train.values.reshape(-1, ), y_test, y_pred, name)
# PostProcessing
X_train, X_test = _postProcessing(X_train[:], X_test[:], scaler, xCols, preType)
```

Si prende come esempio la costruzione della funzione Ridge ma il procedimento è più o meno analogo anche per il resto dei modelli citati precedentemente.

1. *Split parametrizzato*
2. *Feature migliori*
3. *Pre processing*
4. *Cross validation*
5. *Fine tuning*
6. *Cross validation*
7. *Learning e predizione*
8. *Post processing*
9. *Stampa*

Fine Tuning

È tra le operazioni più onerose in quantità di tempo richiesto ma necessaria per avere una maggior precisione e risoluzione dell'algoritmo studiato andando a modificarne i parametri di addestramento.

Questa operazione è stata eseguita almeno una volta per modello creando così un set di parametri ottimali per l'esecuzione senza necessariamente ripete l'operazione ogni volta.

I parametri configurabili sono reperibili nel file **constants.py**

Configurazione

All'interno del file **models.py** è possibile trovare da *riga 478* una serie di parametri configurabili a proprio piacimento per giocare e provare i diversi modelli in fare di esecuzione ponendo l'attributo "**active**" [True, False].

Compaiono altri parametri per consento l'avvio o meno di funzionalità extra all'interno della funzione predisposta per la creazione del modello.

All'interno del file **constants.py** invece si trovano i vari parametri anch'esso configurabili a piacimento per provare differenti combinazioni al fine di sperimentare sul dataset.

I file di "**Main**" sono due e questi ne identificano la parte di analisi e la parte inerente all'apprendimento dei diversi modelli secondo vari casi già predisposti nella funzione di main nel quale basta porre a True il blocco funzione per farlo avviare. Invece nel main di analisi il codice compare totalmente commentato per cui per avviare ciò che si desidera è necessario togliere il commento dal blocco desiderato.

Conclusioni

Come detto precedentemente, risulta impossibile predire con estrema precisione un andamento corretto in questo tipo di task basandosi solamente sull'analisi dell'evoluzione passata. È risultato comunque interessante approcciarsi a questa tipologia di problema soprattutto sulla parte riguardante l'analisi dei dati in quanto è possibile ottenere differenti tipologie di visualizzazione e tra quelle non citate sopra ma presenti nel progetto ci sono:

- Grafici **OHLC** (Open, High, Low, Close)
- Grafici **Candlestick**

E vari plot di correlazioni attraverso le funzioni di:

- **HeatMap**
- **Joint**
- **Pair**
- **Details**

Extra

Create funzioni di extra di supporto, presenti all'interno del file utility.py

Si possono trovare funzioni come *SMA, EMA, CHANGE, EXPANDING MEAN, EXPANDING STD, PROFITTO e BACKTEST.*

Sono presenti anche le funzioni che creano le feature di:

BUY-TIME e SELL-TIME

Ovvero, attraverso lo studio del grafico con le funzioni di supporto citate sopra, stabilire il momento in cui era più conveniente effettuare un'operazione di BUY o di SELL.

Per mia scarsa conoscenza in questo campo nonostante le varie letture di "*investopedia*.com" non sono riuscito ad ottenere dei buoni risultati per andare a provare e sperimentare un task di classificazione utilizzando questa metodologia.