TO-DO-LIST PROGETTO
– FASE 1 –

1. **Documentazione:** Solitamente, chi produce un componente produce pure la documentazione correlata a quel componente.

2. **Estrazione dei Dati:**

   - Creare una Classe **“JSON_Saver”** che avrà un metodo **“save”** che estrapola determinati tipi di candele (volendo anche un solo tipo) di un determinato market (servendosi della classe Extractor) in un determinato intervallo di tempo (aiutarsi con la classe Time_Stamp_Converter per il timestamp) e le salva in una serie di file .json diversificati per tipo di candela.

3. **Refining dei Dati:**

   - Creare una classe **“Dataset_Maker”** che avrà un metodo **extract** che prende in input il path di un file json ,che conterrà candele dello stesso tipo, estratte sfruttando la classe **“JSON_Saver”**.

     A questo punto, il metodo (partendo dal json, che poi sarà un dizionario) dovrà creare esattamente 1 matrice e 1 Vettore:

     ↔ La Matrice è quella delle Osservazioni (contenente tutti i valori tranne quelli che appartengono al Campo ClosePrice) a cui è stato sottratto l’ultimo campo.
     (in sostanza, nella Matrice restano i seguenti campi:
     CloseTime, OpenPrice, HighPrice, LowPrice, Volume).

     ↔ Il Vettore è il Vettore (Colonna) delle Etichette (Contenente esclusivamente tutti i valori del Campo ClosePrice).

     A questo punto, il compito del metodo **extract** finisce scrivendo in un file .pkl (sfruttando il modulo Python “cPickle”) la Matrice delle Osservazioni e il Vettore delle Etichette.

     Nella buona sostanza, con **extract** abbiamo creato un Dataset.

4. **Cleaning dei Dati:**

   - Creare una classe **“Cleaner”** che ha due metodi: features_scaling (si commenta da solo) e mean_normalization (si commenta da solo).

5. Learnig:

- Creare una classe **“Dataset_Splitter”** che ha un metodo **“split”** che (a partire da un file .pkl di un tipo di candele creato con la classe **“Datase_Maker”** e da un numero chimato k) si occupa ,in prima battuta, di suddividere il Dataset in Training-Set (70 %) e Test-Set (30 %).

  In seconda battuta, questo metodo dovrà suddividere il Training-Set in k Porzioni di uguale dimensione.

  Infine, questo metodo dovrà scrivere in 2 file .pkl rispettivamente: le k porzioni del Training-Set e il Test-Set.

  Inoltre, questa classe dovrà avere un metodo **“merge”** per fondere una serie di porzioni (escludendone una) di un Dataset in un’unica porzione; inoltre questo metodo restituisce 2 insiemi che contengono rispettivamente tutti gli elementi uniti (più precisamente, è l’unione di k-1 porzioni) e l’unica porzione esclusa.

- Creare una classe **Curve** che ha metodi che disegnano rispettivamente i seguenti Grafici:

  1. Grafico Ipotesi (2D)
  2. Grafico Ipotesi (3D)
  3. Grafico Grado del Polinomio VS Errore (2D)
  4. Grafico Lambda VS Error (2D) **(IN DUBBIO)**
  5. Grafico Training-Set-Size VS Error (2D) **(IN DUBBIO)**
  6. REC Curve **(IN DUBBIO)**

- Creare una classe **“Normal Equations”** che ha un metodo **learn (X_Training,Y_Training)**, che allena un modello usando le Equazioni Normali come Algoritmo di Learning. Il metodo restituisce un oggetto della classe LinearRegression.

- Creare due classi:

  1. **Recorder_Degree,**
     E’ una classe che serve per tracciare il grafico “Grado del Polinomio VS Errore”.
     Questa classe conterrà tre liste: la lista dei gradi, la lista degli errori di training e la lista degli errori del convalidation-test.
     Ogni volta che si completa una serie di allenamenti dello stesso modello in cui il grado del modello aumenta sempre di più fare le seguenti cose:
     alla fine di ogni allenamento si inserisce il grado del Modello e i valori di J-train e J-cv nelle tre liste (Lista dei Gradi,Lista degli Errori di Training, Lista degli Errori di Convalidation-Test).
     Ovviamente questa classe avrà un metodo per restituire queste 3 liste.

  2. **Recorder_Lambda (IN DUBBIO),**
     E’ una classe che serve per tracciare il grafico “Lambda VS Errore”.
     Questa classe conterrà tre liste: la lista dei Lambda, la lista degli errori di training e la lista degli errori di convalidation-test.
     Ogni volta che si completa una serie di allenamenti dello stesso modello in cui il parametro lambda viene sempre cambiato fare le seguenti cose:

per ogni allenamento completato si inserisce il valore di lambda di quell'allenamento e i valori di J-train e J-cv nelle tre liste (Lista dei Lambda,Lista degli Errori di Training, Lista degli Errori di Convalidation-Test).
Ovviamente questa classe avrà un metodo per restituire queste 3 liste.

6.Valutazione dei Risultati:

- Costruire una classe **"MSE"**, che ha un metodo che calcola l'MSE sul Dataset in input sfruttando un Modello, anch'esso passato in input.

7.Inferenza:

- Creare un piccolo software ,anche a riga di comando, che prende in input una candela di 12 h non ancora chiusa, e che dia in output la predizione del prezzo di chiusura della Candela, sfruttando la classe Model (opportunamente settata) nel caso della Discesa del Gradiente oppure la classe LinearRegression (opportunamente settata) nel caso di Equazioni Normali.