Brian, Damian, Edwin

Testrapport

IPBDAM5

# Code overzicht

De code bestaat uit 5 python bestanden. De eerste die gebruikt wordt is de HistorischeCandlesticksophalen.py. In deze code wordt de data van binance gehaald waarvan tot aan juli 2019 tot nu data over Dogecoin is opgehaald. Dit zijn de candlesticks en die worden vervolgens opgeslagen in de staging area van de MYSQL database op de VM van school. Hier gebeurt alleen al een volledig ETL proces waarin data wordt opgehaald, getransformeerd zodat het in de database gestopt kan worden en uiteindelijk Load waarbij de data wordt opgeslagen in de staging area.

Vervolgens wordt de HistorischeTradesUitvoeren.py gebruikt om de trades uit te voeren door de data te gebruiken die opgehaald is uit het eerste py bestand: HistorischeCandlesticksophalen.py. Hier wordt dus de wallet bijgehouden en alle informatie die bij de trades horen. Beginnend bij de primaire trade wordt er doormiddel van een patroon een signaal bepaald waardoor er gekocht wordt of verkocht. Vervolgens een uur later gebeurt er een secundaire trade die de 100dollar aan Dogecoin terugkoopt als er verkocht was of verkoopt als er gekocht was. Bij elke trade wordt de tijd bijgehouden en kun je zien wat de wallet prijs was op het moment van die specifieke trade.

Om de winst per patroon te bekijken hebben wij een winstperpatroon.py gemaakt. Hiermee kun je zien hoeveel dollar winst of verlies er gemaakt is bij de verschillende patronen. Zo kun je dus zien welk patroon wel geld opbrengt en welke misschien wel verlies draait en die zou je er dan eventueel uit kunnen halen.

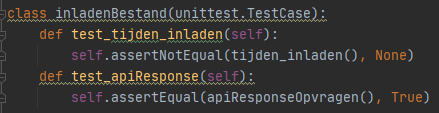
Voor de sentimentenanalyse hebben wij het bestand RedditSentimentAnalyse.py gemaakt. Dit bestand haalt de hot reddit pagina van de Dogecoin reddit op met de 100 laatste posts. Dit gebeurt elk uur waarbij er bij de posts gekeken wordt naar het aantal comments en de likes - de dislikes. Hier komt uiteindelijk een score uit en zo kun je per uur zien wat de score is. Hiermee kun je dan nagaan of er hierbij een verband is met de dogecoin koers en of je het wilt laten meebepalen bij het kopen en verkopen.

Tot slot is het bestand HistorischeDataTimestampsOmzetten.py ervoor om timestamps om te zetten naar datetime. Met timestamps kun je zelf lastig zien wat de datum is en door dit om te zetten maakt het een stuk overzichtelijker om te zien welke datum bij welk record hoort.

# Unittests

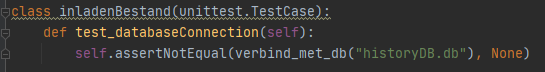
We hebben unit testen uitgevoerd voor de code die we gemaakt hebben. De belangrijkste processen zijn verdeeld in 3 python bestanden, namelijk het data ophalen, de geschiedenis trades uitvoeren en het bestand waar die van alle trades berekend hoeveel winst we per verschillend patroon hebben verdiend of hoeveel verlies we hebben per patroon.

Om te beginnen kijken we naar het data ophalen, in dit python bestand worden alle candlesticks van de afgelopen 3 jaar opgehaald. Hierin gebeuren een aantal stappen en 2 daarvan hebben we getest zoals hieronder te zien:

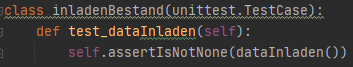


Hier testen we de funtie tijden\_inladen(). In deze functie laadt die het tijden bestand op van alle timestamps die we willen ophalen en die splitst die op in een list en geeft die dan weer terug als variabel. Als het inladen van het bestand niet lukt dan geeft die niks terug of een error, in de meeste gevallen geeft die een None bestand terug. Waar wij op testen is dat die geen None waarde teruggeeft. Als die dus iets anders dan een None waarde teruggeeft is de test gelukt, in dit geval zou die alleen een list kunnen teruggeven. Als die None teruggeeft is de test gefaald. Dit hebben we getest door het bronbestand weg te halen waardoor die het bestand dus ook niet kan laden. In dit geval zal die dus een None type variabel maken en zal de test falen.

Het is ook belangrijk dat we de verbinding met de API van Binance testen voordat we de vervolgstappen uitvoeren. Met deze unittest kijkt die of we een juiste API response krijgen in de vorm van een HTTP code. Deze code wordt omgezet naar een True of False, het is True als de response goed is en False als de response slecht is, in de unittest kijkt die dus naar de response en of de response True is. Als die True is dan is de unittest gelukt, anders is die gefaald.



In de historyTrades is het belangrijkste dat we de connectie met de database testen. In deze unittest roepen we de verbinden met database functie aan met de sleutel naar de database. Deze functie geeft de verbinding terug die we kunnen gebruiken om SQL query’s uit te voeren. Het moet dus niet een None variabel zijn. Hierop testen we en als die dus geen None variabel teruggeeft is de unittest gelukt, anders is die gefaald.



In het bestand winstPerPatroon wordt bij alle trades gekeken hoeveel winst we per patroon hebben gemaakt in totaal. Hierbij hebben we een functie waar alle data ingeladen wordt en gecheckt wordt en dit wordt teruggegeven voor de volgende functie die de data weer gebruikt om berekeningen mee te doen. Het is dus de bedoeling dat de variabel niet leeg is omdat we hier niks mee kunnen. In deze unittest wordt dus gekeken of er wel een variabel wordt teruggeven waar echt iets in staat en dus niet van het type None is. Als het een None is zal de unittest falen, als het geen None type variabel is zal de test slagen en heeft hij de data dus goed kunnen inladen.