**AKTIENPREIS VORHERSAGE**

**Einleitung**

Dieser Code wurde im Rahmen eines IC Kurses an der Universität St. Gallen geschrieben und ist somit nicht als für echte Investitionen an der Börse zu verwenden, sondern dient der Demonstration was mit einem simplen Code alles machbar ist.

Der Code besteht aus den folgenden Phasen:

1. Auslesen der S&P 500 Ticker von Wikipedia
2. Auslesen der historischen Daten für alle S&P 500 Firmen mit QUANDL
3. Vorhersage der zukünftigen Bewegungen mit statistischen analysen (LSTM model)
4. Weitere Datensammlung und -analyse
   1. Auslesen relevanter zusätzliche Daten z.B. Gold-, Dollar- und Ölpreise.
   2. Sentiment-Analyse und Summarizer von Zeitungsartikeln

Das Ziel dieses Projektes ist es relevante Börsedaten zu sammeln und analysieren um ein Dashboard zusammenzustellen welche private Investoren unterstützen soll, für Kurzfristige Investitionen.

**Anleitung**

Um dieselben Resultate zu erreichen müssen folgende Schritte verfolgt werden.

1. Installieren sämtlicher Python Packages welche unter dem letzten Punkt «Requirements» dieses Dokuments stehen
2. GitHub Repository mit GitKraken öffnen und somit eine **lokale Kopie** des GitHubs erstellen.
3. Dateipfad der lokalen Github Kopie herauslesen bis und mit «…\Short»
4. Sämtliche Python Dokumente («Additional Data extraction.py», «Prediction for all 500 tickers as loop.py”, “Prediction single ticker.py”, “S&P500 historische Daten abrufen.py” & “Sentiment Analysis and Summarizer.py”) öffnen und im Code die Variable «DATEIPFAD», welche jeweils am Anfang des Dokuments steht mit dem lokalen GitHub Pfad ersetzen.
5. «S&P500 historische Daten abrufen.py» laufen lassen.
6. «Prediction for all 500 ticker as loop.py” laufen lassen wenn alle Resultate gebraucht werden. Dauert mehrere Stunden zum alle Ticker zu durchlaufen.
   1. Alternativ «Predcition single ticker.py” im Code die Variable «COLUMN” (am Anfang des Codes zu finden) ändern um den gewünschten S&P 500 Ticker einzugeben.
7. «Additional Data extraction.py» laufen lassen um weitere Informationen zu ziehen. In der Liste «lst» können hier weitere QUANDL Ticker hinzugefügt werden je nach Bedarf.
8. «Sentiment Analysis and Summarizer» laufen lassen um vorher manuell erstellte CSV mit Zeitungsartikeln zu auf Sentiment zu analysieren und Zusammenzufassen. CSV Dokument muss im Code ausgewählt werden. Drei Beispiele sind vorhanden, wobei immer zwei als Kommentar markiert werden müssen.
   1. Beispieldokumente unter «…\Short\Data\Manual News Data\»

**Auslesen der S&P500 Ticker**

Als erstes wird als Teil des Pythondokuments S&P historische Daten abrufen.py die S&P500 Liste aus Wikipedia gelesen. Hierfür wird der Webscraper BeautifulSoup verwendet. Alternativ wurde auch ein code mit den Pandas und Read\_HTML Packages geschrieben. Mit diesen beiden Anwendungen werden die Ticker der 500 Firmen ausgelesen und in ein CSV Dokument gespeichert mit dem Namen SP500.csv, welches in den GitHub Ordner gespeichert wird. (unter «…\Short\Data\S&P 500 lists\»)

**Auslesen der historischen Daten**

Die zweite Phase ist erneut im Pythondokument S&P historische Daten abrufen.py enthalten. In dieser Phase wird die eigentliche Datenbank erstellt mit welcher später gearbeitet wird. Mit der Quandl API werden für sämtliche Ticker die Daten herausgelesen (Funktion: «get\_data\_from\_quandl()») und in einzelne CSV Dokumente gespeichert (unter «…\Short\Data\Historical data\by\_ticker\»). Diese einzelnen CSV Dokumente werden in einem zweiten Schritt zusammengefügt mit den Informationen Adjusted Close, Date & Ticker. (Funktion: «compile\_data()») Es wurde Adjusted Close gewählt (anstatt Close, Open, Volume, etc.), da der Adjusted Close bzgl Aktiensplits angepasst wurde und somit über die Jahre hinweg vergleichbar ist. Dieser zweite Schritt speichert ein neues CSV Dokument sp500\_joined\_closes.csv als Matrix, mit dem Datum an der y-Achse und den Ticker an der x-Achse (unter «…\Short\Data\Historical data\»).

Zusätzlich wurde die function «visualize\_data()» geschrieben, welche die Korrelation der Daten visuell darstellt, damit man schnell sehen kann dass z.B. bei einzelnen Firmen Daten fehlen. Dies ist lediglich zur Kontrolle und beeinflusst das Endresultat nicht.

Die Funktionen “extract\_featuresets(ticker)» und «do\_ml(ticker)» wurde als Kommentare gelassen, da diese eine Machine Learning Methode wären um die zukünftigen Preise zu vorhersagen. Jedoch waren die Resultate nicht ausreichend, welches es von uns für das Endprodukt nicht weiter verwendet wurde.

**Vorhersage der zukunftigen Preise**

Für die Vorhersage wurden zwei Pythondokumente erstellt: «Prediction for all 500 tickers as loop.py» & «Prediction single ticker.py». Wobei das erste sämtliche S&P 500 firmen durchgeht und für alle die Vorhersage macht und das zweite Dokument ist für wenn man nur für eine Firma den Code laufen lassen will, wobei man den Ticker manuell eingeben muss. Da beide Dokumente im Aufbau sonst sehr ähnlich sind, gilt folgende Beschreibung für beide.

Der Code beginnt damit, dass das CSV Dokument sp500\_joined\_closes.csv geladen wird. Anschliessen warden die Daten für den gewählten Ticker ausgelesen. Anschliessend müssen die relevanten Daten für die weiteren Schritte vorbereitet werden. Hierzu werden zum einen mit dem «MinMaxScaler()» die Daten «normalized». Und zum anderen werden die Daten aufgeteilt in 70% Trainingsdaten und 30% Testdaten. Also nächstes wird das eigentliche Model vorbereitet. Für dieses Projekt wird ein Deep Learning Ansatz von Keras verwendet. Das sequentielle Model besteht aus einer LSTM Layer und zwei Dense Layern. Die resultierenden Vorhersagen werden anschliessend als CSV pro Ticker gespeichert (unter «…\Short\Data\Prediction data\by\_ticker\» für loop und unter «…\Short\Data\Prediction data\test\_single prediction\» für einzelne Ticker).

Die resultierenden CSV Dokumente sind nicht formatiert und benötigen eine manuelle Formatierung bevor sie für weitere Schritte verwendet werden können. Damit man die Qualität der Resultate einfach einsehen kann wird zudem, wenn alle Ticker durchlaufen werden noch ein zusätzliches Dokument erstellt (unter «…\Short\Data\Prediction data\mean\_sqrd\_error.csv»), welches den Mean Squared Error für jeden Ticker auflistet. Je tiefer dieser Wert, desto höher ist die Genauigkeit der Vorhersagen im Vergleich zu den historischen Daten.

**Weitere Datensammlung und Analyse**

Um das Ziel eines umfangreichen und übersichtlichen Dashboards zu erreichen benötigen wir weitere Daten. Deshalb, wurde der Code “Additional Data extraction.py” aufgestellt. Dieser Code verwendet einen ähnlichen Aufbau wie bereits gesehen. Daten werden mittels Quandl herausgelesen und ein CSV Dokument wird erstellt (unter “…\Short\Data\Historical data\additional data\”).

Zudem wurden Zeitungsartikel manuell herausgesucht welche mit dem Code im Pythondokument «Sentiment Analysis und Summarizer.py» analysiert wird. Mit dem VaderSentiment Packet werden das Stimmungsniveau (=Sentiment) der Zeitungsartikel analysiert. Mit dem Gensim Summarizer Packet werden Zusammenfassungen der Artikel kreiert. Die Resultate dieser analyse werden wiederum als CSV gespeichert (unter «…\Short\Data\Manual News Data\Sentiment scores\»). Die Sentiment Resultate werden verwendet um zusätzliche Daten aus den Nachrichten zu ziehen welche auf einen steigenden oder sinkenden Kurs deuten. Und die Zusammenfassung wäre ein möglicher Weg die Artikel in Kurzform für die Nutzer anzubieten.

**REQUIREMENTS (Python packages)**

* gensim
* vaderSentiment
* pandas
* numpy
* sklearn
* matplotlib
* bs4
* quandl
* pandas\_datareader
* keras
* tensorflow
* importlib