Einleitung

Für das Projekt der JungBank, eine Aktienempfehlung auf Basis von Newsdaten, hat die Fintech Consulting ein Team aus sechs Experten zusammengestellt. Dieses besteht aus Andreas Edte, Paula Hölterhoff, Michael Reinko, Ann-Kathrin Kälberer, Moritz Müller und Yannik Hubrich. Die Aufgabe der JungBank sind wie folgt gestellt:

Wir wollen unsere Kunden besser beraten, in welche Aktien sie investieren sollen. Um Ihnen einen zusätzlichen Mehrwert zu bieten, sollen sie basierend auf Risiko-Vorliebe und Welt-Entwicklung einen Vorschlag mit Machine-Learning erhalten.

* Kunde muss in der Lage sein, Risiko Vorliebe einzustellen
* Ein Algorithmus soll Informationen von News-Seiten sammeln, diese sollen als Input verwendet werden, um eine ML-Prediction zu erzeugen
* Es sollen Kennzahlen und historische Daten erhoben werden
* Diese sollen zum Training des Algorithmus verwendet werden, um eine besseren Vorschlag zu erzeugen

Das Problem, welches sich aus der Aufgabenstellung ergibt hat die Fintech wie folgt interpretiert und so weit es der zeitliche Rahmen vorgab umgesetzt.

Mithilfe einer App, sollen den Kunden der JungBank Aktienempfehlungen und Aktieninformationen bereitsgestellt werden. Mithilfe eines Machine Learning Modells sollen Vorhersagen, basierend auf historischen Daten und Börsennews, gemacht werden.

Planung

Um das Projekt der JungBank uzustezten hat sich das Team der Fintech erst einmal das Vorgehen besprochen. Dabei wurde ein zeitlicher Plan erstellt, Teammitgliedern wurden Rollen und Verantwortungsberreiche zugeteilt, sowie eine Meetingstruktur bzw. eine Produktmanagmentmodell ausgewählt.

Das Produktmanagmentmodell für welches sich die Fintech Consulting entschlossen hat ist SCRUM. Es eignet sich nicht nur durch seine agilität sondern auch, dadurch das sich die Prinziepien und Werte von SCRUM mit denen der Fintech Consulting decken. SCRUM stellt eine agiles Framework für Softwareprojekte dar, die es erlaubt iterativ durch den Entwicklungsprozess zu manövrieren. Es besteht aus Rollen, Meetings und Werkzeugen. In diesem Fall sind Andreas Edte der Product Owner und Yannik Hubrich der SCRUM Master. Das restliche Team ist der SCRUM Rolle Team zugeteilt. Die Meetings die die Fintech Consulting abgehalten hat, waren ein Planning Meeting, jede Woche ein Sprint Meeting und ein Review Meeting. Da in den Sprint Meetings alle Tasks besprochen wurden und es zeitlich nihct möglich war sich jeden Tag zu treffen, hat die Fintech Consulting darauf verzichtet Daily Meetings abzuhalten.

Als Werkzeug wurden unter anderem Discord zur Kommunikation und für Meetings genutzt. Über Github wurde Code und Dokumente für alle Teammitglieder zur Verfügung gestellt und Clickup diente als Projektmanagmenttool in der alle Tasks und deren Inhaber dokumentiert und getrackt wurden.

Die Rollen wurden innerhalb des Teams nach Stärken der einzelnen Teammitglieder verteilt.

Andreas Edte ist durch sein umfassendes Wissen über den Börsenmarkt ein Experte auf diesem Gebiet und ist deshalb mit der Konzeption des Produktes betraut.

Yannik Hubrich war als SCRUM Master damit betraut die Organisation zu führen und dafür zu Sorgen das es für das Team keine Blocker gab. Außerdem war er für die Umsetzung des Machine Learning Modells und die Entwicklung einer geiegneten Datenbasis zuständig.

Moritz Müller war für die Entwicklung der App und des Frontends betraut. Er unterstütze außerdem Andreas Edte bei der Konzeptionierung des Modells.

Paula Hölterhoff war für die wirtschaftlichen Analysen und die Entwicklung einer Marketingstrategie für die JungBank zuständig.

Michael Reinko designte die Präsentation und die App nach deren Vorlage Moritz Müller sie umsetzen konnte.

Ann-Kathirn Kälberer war für die Dokumentaion verantwortlich.

Der zeitliche Plan sah folgendes vor.

1. Innerhalb der ersten Woche setzt sich das Team intensiv mit den Anforderungen für das Produkt ausseinander. Durch Recherche und Brainstorming ergibt sich eine Konzeption und die einzelnen Komponenten des Produktes. Außerdem wird die Entwicklungsumgebung aufgesetzt und strukturiert.
2. Die nächste Woche wird dann für wirtschaftliche Analysen genutzt.
3. Dann wird eine Woche lang die Datengrundlage für das Modell entwickelt und extrahiert.
4. In der vierten Woche wird die Präsentation für die Zwischenbesprechung mit der JungBank vorbereitet.
5. In den letzten Wochen wird dann der Code für das Modell geschrieben und die Daten bereinigt.

Das Hauptrpoblem das wir bei der Planung hatten, ist das wir den Aufwand zeitlich überschätzt haben und statt mit dem kleinstmöglichen funktionierenden Prototypen zu planen auch schon fortgeschrittene und erst einmal unnötige Bestandteile geplant haben und umsetzen wollten. Hierzu gehört zum Beispiel eine Entwicklung und Umsetzung einer Datenbankstruktur und einer Pipeline um mit dieser zu Kommunizieren. Da wir dies zu spät erkannt haben, konnten wir den Filter Algorithmus nicht mehr umsetzen. Wir haben uns dazu entschlossen diesen zu vernachlässigen, da er am einfachsten Umzusetzen ist und nicht unbedingt notwendig ist, da man die Aktien auch erst mal nur manuell filtern kann.

Konzeption des Produktes

In der Konzeption wurde früh erkennbar, dass sich historische Börsendaten nicht für einen Machine Learning Algorithmus eignen, da sie sich zufällig verteilen.

Die Fintech entschloss sich deswegen einen Algorithmus zu bauen, der anhand von News, deren Reichweite und Sentiment eine Vorhersage auf einen fallenden oder steigenden Kurs, n Tage nach deren Veröffentlichung, zu treffen. Diese Vorhersage wird pro Aktie getroffen. Die Annahme ist hierbei das News die eine große Reichweite haben, je nach ihrem Sentiment (positiv oder negativ) die Börse n Tage nach ihrere Veröffentlichung beeinflussen.

Eine Filterung der Aktien findet durch einen Auswahlagorithmus statt, der als Eingabe die Risikobereitschaft, den Anlagezeitraum und die Branchenpriorität des Users nutzt um die Aktienempfehlung einzugrenzen.

Im Frontend werden die Eingaben vom User getätigt und an das Backend übergeben, wo sie dann zu Empfehlungen verarbeitet und ausgegeben werden.

Um diese Vorrausetzungen zu erfüllen sind folgende Komponenten notwendig

* Geeignete Datenquellen aus der man News entweder über eine iterativ API oder in einem Batch abrufen kann und in einer Datenbank speichert. Das selbe gillt für eine Datenquellen für historsiche Börsendaten.
* Eine Pipeline, die die Daten bereinigt, unnötige Datensätze löscht und zu einer geeigneten Datengrundlage transformiert.
* Eine Sentiment Analyse
* Ein Machine Learning Modell, welches die Empfehlungen anhand der Datenquelle generieren kann
* Eine App oder Frontend
* Tabellen (Datenbasis oder Datenbank), die Newsdaten, Börsendaten und Kundendaten enthalten

Die Datenquellen

Relativ einfach war es eine geeigenete Datenquelle zu historsichen Datenquellen zu finden, hierfür wurde die Yahoo Finance API genutzt, da sie ausreichend Kennzahlen sowohl zu einzelnen Aktien als auch zu historischen Kurswerten enthält. Die API kann über mehrere Python Bibliotheken angesprochen werden, die Fintech nutzt dafür die yfinance Bibliothek.

Schwieriger gestalltete sich die Suche nach einer geeigeneten API oder Möglichkeit Newsdaten zu sammelen. Das Hauptproblem dabei war das der Text des Artikels auf max 200 Zeichen begrenzt war, sodass meistens keine Zuordnung zu einer oder mehreren erwähnten Aktien möglich war. Die APIs, die den gesamten Artikel abrufen konnten, waren leider sehr kostenintensiv.

Durch dieses Problem konnte die FIntech keine geeigente Datengrundlage für News finden und nahm als Alternative Social Media Posts. Auch hier war das Problem das Aktien nicht namentlich erwähnt wurden, was eine Zuordnung unmöglich machte. Allerdings konnte die Fintech Twitter Accounts identifizieren, die sehr häufig Tickersymbole in ihren Tweets nutzten, was eine Zuordnung möglich machte. Über die Twitter API und die Python Bibliothek Tweepy konnten wir so bis zu 500000 Tweets pro Monat abrufen. So konnte eine mehr oder weniger geeignete Datenquelle erstellt werden.

Die Pipeline

Die meisten Probleme die mit dem Bereinigen und Transformiern der Twitter und Börsendaten auftraten, wurden erst während der Umsetzung erkennbar, konnten aber alle gelöst werden. Ein Problem bestand darin, dass der ursprüngliche Plan, nämlich eine Pipleine von der API in eine Datenbank zu bauen, aufgrund von mangelnder Zeit nicht umgesetzt werden konnte, weswegen zu Testzwecken erst mal die Daten nach der Bereinigung und Transformation in einer CSV abgespeichert werden.

Die Sentiment Analyse

Aufgrund der sehr unaussagekräftigen Datenbasis, war es schwierig eine wirklich gute Sentiment Anaylse zu erhalten, da in einem Twitter Tweet häufig mehrere Aktien genannt werden, wobei eine postiv genannt wird und die andere durch den dadruch entstehenden Kontext als negativ betrachtet werden muss. Die von uns genutzte Sentiment Analyse kann jedoch nur die Stimmung des gesamten Tweets beurteilen, wodurch alle genannten Aktien entweder positiv oder negativ betrachtet werden. Dieses Problem ist uns bewusst, allerdings konnten wir es in der gegebenen Zeit nicht lösen. Die Sentiment Analyse die wir nutzen ist Textblob eine Python Bibliothek.

Das Machine Learning Modell

Als Modell wurde für dieses Projekt ein Decisoion Tree und ein Random Forest trainiert, allerdings konnte nur für 4 verschiedene Aktien, nämlich Amazon, Apple, Microsoft und Tesla eine Modell erstellt werden. Durch das bereinigen und transformieren der Daten wird nämlich nur für diese vier Aktien eine ausreichend große (ca. 100 Zeilen) Datenbasis gebaut. Auch durch das die mangelnde Qualität der Datenbasis ist das Modell leider nicht sonderlich gut und kommt nur auf etwa 50% Genauigkeit bei den Testdaten. Dieses Problem könnte durch eine bessere Datenbasis behoben werden. Aufgrund der Twitterdaten ist die Datenbasis jedoch leider nihct sehr aussagekräftig. Durch weiteres transformieren und filtern der Daten könnte ggf. ebenfalls eine bessere Datenbasis geschaffen werden.

Die Tabellen

Die Tabellen konnten alle so umgestzt werden, wie erhofft, jedoch anstatt in einer Datenabank für diesen ersten Prototypen in einer CSV. Die Tabellen für die Kundendaten wurde aufgrund von Zeitmangel noch nicht angelegt. Die Tabelle für die einzelnen Aktien wurde zwar schon erstellt, jedoch noch nicht mit Daten zu Risiko, Anlagezeitraum und Branche erweitert. Bisher betseht diese Tabelle nur aus den Tickersymbolen. Die Newstabelle ist nach der Bereinigung etwa 3000 Zeilen groß und enhält neben dem dem Datum der einzelnen Tweets auch Likes, Retweets, Polarity und Subjectivity (Sentimnetdaten) und eine Liste der erwähnten Aktien, sowie eine Liste mit deren durchschnittlichen Börsenerfolg nach einem Tag.

Umsetzung Datenbasis und Modell

Bei der Umsetzung wurde neben Standardbibliotheken mit Textblob, Pandas und Scikit-Learn gearbeitet. Textblob ist eine Bibliothek für Sentimentanalysen die auch mit wenig Zeichen im Text umgehen und Ergebnisse liefern kann.

Pandas ist eine Bibliothek die Daten in einer Variable als Tabellenform (Dataframe) speichern kann und somit ein Arbeiten mit Daten sehr vereinfacht. Man kann Daten so gut löschen, bereinigen, kombinieren und transformieren.

Scikit-Learn ist eine Bibliothek die diverse Machine Learning Algorithmen und Evalutaionsmodelle beinhaltet, sodass man mit einem geeigneten Datensatz unkompliziert ML Modelle erstellen kann.

Im ersten Schritt werden im Script *‘twitter.py’* mithilfe der Tweepy Bibliothek, aus drei verschiedenen Twitter Accounts jeweils die letzten 10000 Tweets abgerufen und in der CSV *‘twitter\_data.csv’* gespeichert. Als Spalten werden gespeichert, *['username', 'id', 'date', 'retweet', 'likes', 'text'].*

Im zweiten Schritt werden im Skript *‘clean\_twitter\_data.py’* mizhilfe von der Pandas Bibliothek eine Dataframe aus der zuvor gespeicherten CSV erstellt und dann Zeilen gelöscht, deren Tweettext kein Tickersymbol enthält. Alle Zeilen deren Tweettext eine oder mehrere Tickersymbole enthält, werden um eine neue Spalte *['Symbol']* ergänzt. Die Werte dieser Spalte sind Listen mit den im Tweettext gefundenen Tickersymbolen. Dieser Datafram wird dann als CSV *‘twitter\_data\_clean.csv’* gespeichert. Die Spalten lauten nun *['username', 'id', 'date', 'retweet', 'likes', 'text', ‘Symbol’].*

Im nächsten Schritt wird dann die Sentimentanalyse im Script *‘sentiment\_analysis.py’* durchgeführt. Dafür wird wieder Pandas genutzt um einen Dataframe aus der zuvor gespeicherten CSV *‘twitter\_data\_clean.csv’* zu generieren, der dann mithilfe von der Bibliothek Textblob mit Sentiment Daten erweitert wird. Dafür werden die Tweettexte erst von URLS, Retweets, Hashtags (bspw. #aktien) und Erwähnungen (bspw. @NYTimes) bereinigt, da sie das Ergebnis der Sentimentanalyse verfälschen könnten. Dann wird mithilfe von Textblob die *Polarity* in negative oder posititve Richtung und *Subjectivity* berechnet. Die *Polarity* gibt dabei an wie polarisierend der Text in entweder positive oder negative Richtung ist. Aus der *Polarity*  wird dann noch ein Score berechnet, der angibt ob die Polarity positiv (>0), negativ (<0) oder neutral (=0) ist. Das Ergebnis wird dann in der Datei *‘twitter\_data\_with\_sentiment.csv’*  mit den Spalten *['username', 'id', 'date', 'retweet', 'likes', 'text', ‘Symbol’, ‘Subjectivity’, ‘Polarity’, ‘Score’]* gespeichert.

Anschließend wird im Script *‘get\_stockprice\_after\_x\_days.py’* die CSV *‘twitter\_data\_with\_sentiment.csv’*  mit dem durchschnittlichen Börsenerfolg einen Tag danach erweitert. Dafür wird die CSV wieder in einen Dataframe umgewandelt und dann für jedes Element jeder Liste in der Spalte *['Symbol']*  die *yahoofinance* Bibliothek genutzt um diese Daten von der Yahoo Finance API zu erhalten und zu berechnen. In der Spalte *['Growth\_one\_day']* werden die Ergebnise als Liste gespeichert. Das Ergbnis wird dann in der CSV *‘twitter\_and stock\_growth\_data\_final.csv’* mit den Spalten *['username', 'id', 'date', 'retweet', 'likes', 'text', ‘Symbol’, ‘Subjectivity’, ‘Polarity’, ‘Score’, ‘Growth\_one\_day’]* gespeichert.

Im letzten Schritt der Bildung der Datenbasis wird im Script *‘Final\_cleaning.py’* alle Zeilen der CSV *‘twitter\_and stock\_growth\_data\_final.csv’* gelöscht, für die keine Börsendaten gefunden werden konnten. Dies ist zum Beispiel dann der Fall, wenn der Tweet an einem Freitag oder Samstag veröffentlicht wurde, da Samstag und Sonntags die Börsen geschlossen sind. Diese endgültige und final bereinigte Datenbasis wird dann in *‘twitter\_and\_stockdata\_cleaned\_final.csv’ mit den Spalten ['username', 'id', 'date', 'retweet', 'likes', 'text', ‘Symbol’, ‘Subjectivity’, ‘Polarity’, ‘Score’, ‘Growth\_one\_day’]* gespeichert.

Als letztes wird im Script *‘Machine\_Learning\_Modell.py’* die CSV *‘twitter\_and\_stockdata\_cleaned\_final.csv’*  importiert und es wird für jede Aktie, die über genug Zeilen verfügt ein Entscheidungsbaum und ein Random Forest erstellt. Daür wird die Bibliothek Skit-Learn verwendet. Als erstes wird aus der CSV ein Pandas Dataframe, bei dem nur die Spalten *[‘date’, 'retweet', 'likes', 'Subjectivity', 'Polarity', ‘Growth\_one\_day’]* relavnat sind, pro Aktie erstellt und auf Datum aggregiert. Falls dieser groß genug ist, wird ein Modell Entscheidungsbaum und ein Random Forest erstellt und deren Genaugigkeit ausgegeben.