PRODUCT DESIGN

BASIC USE AND INTERFACE

This product was designed to provide the user with a central dashboard that summarises their recent banking history. This central dashboard is also used for access to 3 different analytic dashboards: Wheat, Corn, and Market Sentiment (appendix fig. 1). The user will be able to look at recent history regarding the 2 commodity dashboards along with visuals of RNN generated prediction models. These dashboards also provide an indication of wheat / corn price based on recent weather conditions and has a redirect link to a commodities exchange if the user wishes to trade (appendix fig. 2). This product would be an online portal given by the company intended for small to medium agricultural clients whereby a username and password will be required to access their personal dashboard (appendix fig. 3).

INPUTS

From a programming perspective there are 4 main inputs needed for this product to function – wheat and corn commodity price history, commodity news articles in a json format, client cash account database of transactions to date and weather history data.

The user inputs will include a username and password that gives them access to their dashboard.

CALCULATIONS AND OUTPUTS

The dashboard uses the inputs to calculate different timeframes based on the current time of use. This is to achieve 3-month timeframes from the current date, creating personalised analytics for each user. The inputs would be processed within the backend algorithm. These algorithms include data cleansing functions, rolling averages and RNN modelling that outputs graphical data to present within our dashboard.

INPUT FEATURES

INPUT	*NECESSARY USE FEATURES*	*REQUIREMENTS*
Commodity Price History for Wheat & Corn	Date Settlement Price Open Interest Closing Volume Daily Open, High and Low Price	CSV or XLSX format Prices should be in numeric format At least 1 year of price history data Data must be in daily intervals
Commodity News	Date of news Headline / abstract	. json format
Weather Data	Date Maximum & Minimum Temperature Data at 9am & 3pm for: - Temperature - Humidity - Wind Speed	CSV or XLSX format Numeric data (except for date) Data must be in daily intervals
Client Cash Account Data	Client Transaction Date Type (Deposit or Withdrawal) Customer / Supplier Reference Description Flow Balance	CSV or XLSX format Flow and Balance must be numeric data

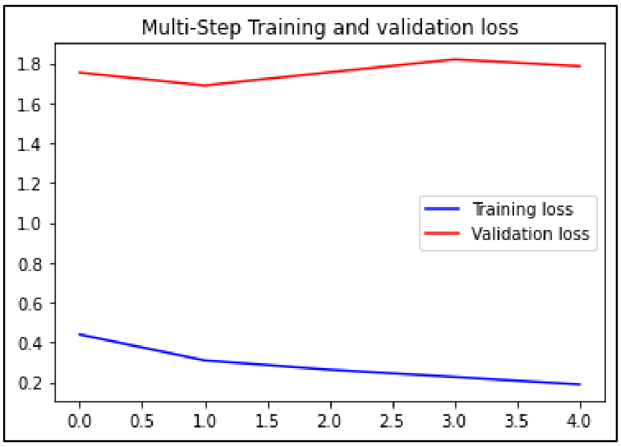
Station 2

Station 2 is needed to select the relevant features used in our model. This involves receiving ETL data listed in the table above and selecting and calculating the required data. The main objective completed in this station was the identification and calculation of weather data including average temperature, humidity, and wind speed. Additionally, station 2 is used to aggregate the relevant commodity and weather data into a singular data frame for use in modelling.

Model Design and Implementation

Design and Implementation

The model used in this product is a form of recurrent neural network (RNN) called Long-Short Term Memory (LTSM). The input variables pass through 3 "gates" that change the weighting of the input to create the output, i.e. each gate changes how the input is "remembered" from one point in time to another. The first gate is the 'input gate' which determines the input values to change the memory and assigns the value a weight. The second gate is the 'forget gate' which uses a sigmoid function that decides details to be removed from memory. The last gate is the 'output gate' which takes the original inputs and processed memory to determine the relevance for a given output. This process is repeated to minimise the loss value for the model given a training set of data. We can adjust and optimise the model (appendix fig. 4) based on the results of this loss function that is visualised within the algorithm:



In this example, training loss was decreased but validation loss stayed relatively constant, meaning further developments are needed to increase the model fit of the generated outputs to the validation data. For our product, the output was the price of the commodity (wheat or corn) with the following inputs:

- Average temperature
- Average humidity
- Rainfall (mm)
- Commodity Closing Volume
- Commodity Open Interest

The second model used is an NLP model to provide sentiment analytics for news relating to commodities. This model utilises the VADER and NLTK python libraries to parse through article headlines and determine the numerical significance of the sentiment and its influence on commodity price.

Assumptions and Limitations

The RNN model assumes a relationship exists between the inputs and outputs; its job is to find out the extent of this relationship. Hence, it's important to understand the business context and provide inputs with relevance to the output whilst remaining cautious for misleading correlations. The other limitation the model has is known as the "vanishing gradient". Essentially, the weight on each layer is a multiple of thea learning rate, the error term from the previous layer and the input to that layer. When these variables approach 0 the weight on these layers also approach 0 making it very difficult to train these layers as we pass our training timeline. A hardware restriction of this model includes the large computational power needed to provide accurate results for larger data sets.

The NLP model used has a similar context limitation wherein the words analysed in the VADER / NLTK library are broad and not business context specific. To develop sentiment accuracy a customised library would need to be developed and built within the model.

Expectation in Model Implementation

The expectation is to see 5 future value points generated by the model that is like the true results of the training data set. With this information we can predict whether the commodity price is going to go increase or decrease. We recommend the following steps taken to achieve this goal:

1. Decide input and output used within our model (Station 2)
2. Develop functions to be used in the model (Station 3)
3. Implement the model with station 2 and 3 to receive a graphical interpretation of predicted outputs

The sentiment indicator model also aims to analyse whether the news value is positive, neutral, or negative and the extent it affects the commodity price. Practically, the user will be able to visualise and determine whether recent news leans towards positive or negative and help inform a decision to trade the given commodity. The following steps are recommended to implement this model:

1. Set-up sentiment analyser from the VADER library
2. Run and merge sentiment scores to each headline
3. Group scores for the most recent month for utilisation in the dashboard in the form of a bar graph with the 3 categories – negative, neutral, and positive

Station 3

Station 3 is developing the model that we use to predict the future. We utilise the TensorFlow library and set the parameters to test our training data. We also decide whether the model is suitable to select single or multiple variables. Based on the business context a multi-variate approach was suitable in our model. We then run and optimize the model by tweaking the parameters (Epochs, evaluation interval, batch size and buffer size) to achieve an accurate prediction of commodity prices.

Station 4

Station 4 is the outputs that the user will use to inform their decisions. For the RNN model, station 4 involved running the functions from station 3 for the features considered in the commodity price (output). For the NLP model, station 4 is loading the pre-set libraries through the .json file of headlines. We then run the recent month headlines and visualise the general sentiment for the user in the dashboard.

Difficulties Faced

The main difficulty faced is understanding the RNN model process. Due to a lack of programming experience, it took time to understand the basic functionality of an RNN and was challenging to apply within the context of this project. Another difficulty was managing stakeholder expectations and futureproofing the product. The data factory framework helps the scalability of the model for backend programming use but was difficult to keep the user experience as the core objective. In other words, our objective is to provide a data driven analytics to the client and we use an RNN/NLP model to achieve this objective. However, as I was learning and optimising the model my solution became my own objective i.e. I just wanted to make the model work because I found it interesting. Ultimately, the user is not me or the programmers working on this product but the client and the customers. I had to remind myself of this to simplify the user experience and provide actionable data-driven insights as opposed to technically advanced data tools used in the backend.

Appendix

Product Design

Fig 1.

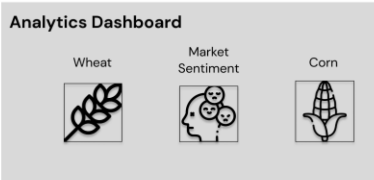
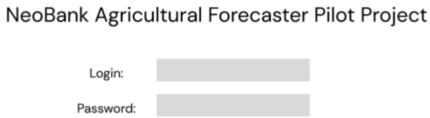
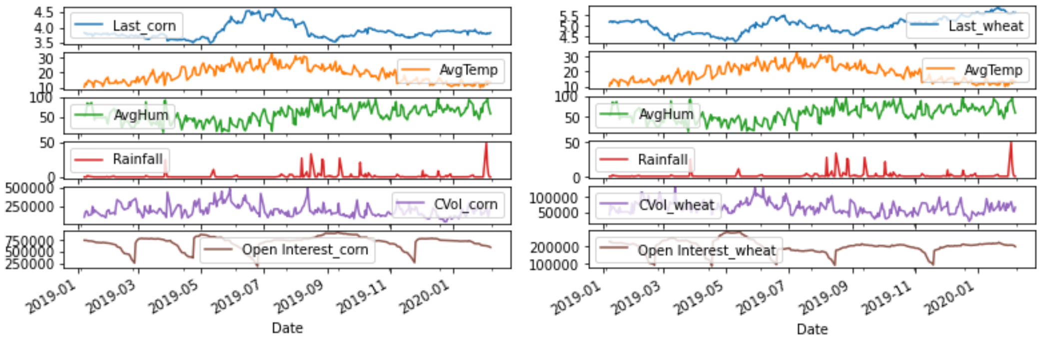


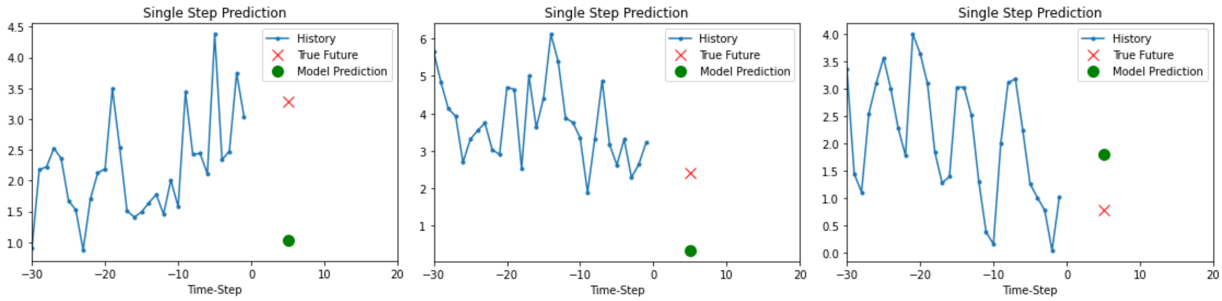
Fig 2.

Fig 3.



Graphs used for optimising model (Fig. 4):





Figma Wireframe Link:

https://www.figma.com/file/bf565ypvP6EQQsCxOwY079/NeoBank-Pilot-Project?node-id=12%3A37

Research References:

Essentials of Deep Learning: Introduction to Long Short-Term Memory -

https://www.analyticsvidhya.com/blog/2022/03/an-overview-on-long-short-term-memory-lstm/

An Overview on Long Short-Term Memory (LSTM) -

https://www.analyticsvidhya.com/blog/2022/03/an-overview-on-long-short-term-memory-lstm/

UX icons – Flaticon.com