1. This video is for COMP390. My name is Haoming Guo and the topic of my project is Deep learning for commodity price prediction.
2. Since this project is build in Google Colab without any UI interface this PowerPoint will contain all contents we need in the video. I will identify the project tasks First, and then show the implementation detail
3. Let’s move onto the dataset. The dataset is from Kaggle, which contains time-series price data for six commodities from 2000 to 2022. It is a .csv file with 6 columns like Symbol, Date, Open, high, Low and Close.
4. Before we begin to code, I fully considerate What I can do based on this dataset. However, to follow our project topic, the target is to find a best performance deep learning model to predicte exact price.
5. Well, since there are having so many models for the price prediction. It is not possible to implement and check them all. After reading some background material, I plan to use the Linear regression and LSTM model for this dataset. As for other models like Decision tree and random forest, since there are all numerical data in the dataset, it is not appliable too.
6. First thing of implementation is loading dataset. I use pandas library to store it in a dataframe. The figure besides is an overview of it, What’s more, dataframe.groupby() can help to divide the different commodity price separately. After that, I choose to use MinMax scaler to normalize in range 0-1 so that can accelerate the training process.
7. Next, I want to check the relativity between volume and price for further features selection. So I plot the close price and volume together and found that the relativity between them is not strong. Moreover, volumes even can be called noise. for example, the Nickel volume keeps near zero level until got a highest peak immediately. And as for US Wheat, there has a huge gap between two periods which is strange. So as result, I decide not to use volume as the features because of its random distribution.
8. Well, to try different models and features, I choose US Wheat as a sample. I use price data with lookbacks as features. And split the train and test dataset as ratio of 8:2 that because a too small train dataset may will cause the model overfitting and a too small test dataset may cause the lack of the generalization.
9. for linear regression, I import the sklearn library and define some supporting functions. The data\_split() can split the scaled data with lookback. And the function linearRegression\_lookback() will not only train a linear regression model. But also predict and return the mean square error between the prediction and ytest.

Here is the result of different model with different lookback in range from 10 to 100 with step long as 5.

Since the result from shuffled data is not monotonous, I also train the model for unshuffled data and monotonicity show up. I guess the reason behind this situation is the longer lookback model has, the more specific period hidden rule it will learn, which does not have generalization properties. It also can be called a kind of overfitting, I think.

1. To represent the performance more directly, I plot the ytest and the predition in same figure. As result, the model trained by close shuffled price with lookback window = 75 is the best linear regression model.
2. Here do the same things for every commodity in the dataset following the main steps above.
3. here are the parameter and feature of best performance model for every commodity.

As the reason why these commodities have different demand feature or lookback for training, I guess the longer lookback do not mean the better learn ability, since it will cause overfitting. As for features, I expected that all prices which have four kinds of different price, will train a better model than only use close price. However, the truth is not as simple as I expected. The answer about these questions may require a further study.

1. However, the main task is using a deep learning model for price prediction. So I build a RNN model, LSTM, which can handle the sequence data. It also can avoid the problem like Vanishing Gradients or Exploding Gradients or cannot catch the long period relationship. by adding three gates: input gates, forget gate and output gate.

Speaking of the implementation detail, first self-defined functions. is data\_split2(). it has slightly differences with the data\_split() in Linear regression. This is because lstm model need the data in shape of like (10,1) instead of in (10,)

1. Another function is called LSTM\_model(). this function can allow me to build different model structure. I can define the specific layers number, units number and the drop out rate. The structure will be start with several lstm and dropout layers and end with a dense layer to strict the output in one dimension.
2. However, the most difficult step is finding parameter combinations to get the best performance. The parameters what it has are shown below. It is not doubt that it has a quite heavy workload and will cost lots of time.
3. To explore the huge possible parameter space, I import the GridSearchCV from sklearn library. I create an object of it and run in different lookbacks to find the best model. Then predict and plot the mse between prediction and label. Based on that, I can directly find the best model details for this commodity price prediction. However, the epoch I define as ten by myself since I find the converge will be reach at 7 or 8 epoch, which represent as the mse will not reduce.
4. The shuffling of data will help to avoid overfitting. so I will prefer to use shuffled data when I train the model
5. This is using only close shuffled price data as feature. After grid Search, the parameter having best performance is shown below.
6. This is using all price data as feature. The details of the best parameter combination are shown below. This model has the lowest mse among the shuffled dataset.
7. Here are the two results for unshuffled data. Their performance are not as good as the best situation I just talk about. The main reason may possible be overfitting.
8. Above is just a case I find to build a lstm model for US Wheat. As for other commodity, I will do the same things again since I believe the different price movement or distribution may have the different paramter for prediction. And I will put their result in the dissertation instead of in this video.
9. In conclusion, I use shuffled data to avoid overfitting. Split dataset and Find a LSTM model structure having lowest mse during test with specific parameter. However, There still have many other questions about the topic. for example, I only consider the close price and all price as two possible features. And it still can produce more features like average price or max gap.
10. This is the end of this video. Thank u for your watching.