|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test\_size | Feature | AUPRC\_test | AUPRC\_train |
| Logistic Regression | 0.2 | Past 10 days’ corn price + average sentiment score of the current day | 0.5076 | 0.5013 |
|  | 0.2 | Past 90 days’ corn price + past 10 days’ average sentiment score | 0.5162 | 0.5489 |
| Linear-SVM | 0.2 | Past 60 days’ corn price + average sentiment score of the current day | 0.4949 | 0.5492 |
|  | 0.2 | Past 60 days’ corn price + past 10 day’ average sentiment score | 0.4678 | 0.5423 |
| Kernel-SVM | 0.2 | Past 60 days’ corn price + average sentiment score of the current day | 0.5833 | 0.9691 |
| Multi-Layer Perceptron | 0.2 | Past 150 days’ corn price + average sentiment score of the current day | 0.4303 | 0.4591 |
|  | 0.2 | Past 90 days’ corn price + past 10 days’ average sentiment score | 0.4667 | 0.4649 |

TABLE1: Multi-class classification – Up/Down/Neutral

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Test\_size | Return\_Train | Return\_test | Feature | AUPRC\_test | AUPRC\_train |
| Logistic Regression | 0.2 | 13342 | 11155 | Past 120 days’ corn price + past 10 days’ average sentiment score | 0.6596 | 0.7005 |
| Linear-SVM | 0.2 | 14685 | 12296 | Past 120 days’ corn price + average sentiment score of the current day | 0.6874 | 0.7140 |
| Kernel-SVM | 0.2 | 20456 | 12389 | Past 120 days’ corn price + average sentiment score of the current day | 0.6664 | 0.8116 |
| Multi-Layer Perceptron | 0.2 | 13263 | 8752 | Past 150 days’ corn price + average sentiment score of the current day | 0.4368 | 0.6233 |

TABLE2: Binary classification – Up/Down

**Classification problem**. In this section, we study the problem of predicting corn price movement. Here we apply four different classification models – logistic regression, linear SVM, kernel SVM and multi-layer perceptron, and compare their performance in both training dateset and test dateset. All algorithms were implemented in Python.

**Experiment setup.** First, we derived corn price movement signals (Up/Down/Neutral) from daily corn price. The goal is to predict the movement of corn price for the next day. Let’s denote the corn price at current day as , and the mean value of the next five days’ price as . Then if , we label the movement as Neutral. Otherwise, if , we label the movement as Down, and if , we label the movement as Up. In this way, for each day, we have the movement for the next day as our ground truth. Second, we use tweets from 2008-2017 as our additional feature. We first clean up the data by dropping the tweets from 2008-2009 where the tweets are sparse. Next, we clean the tweet text by removing special characters and URLs. Finally, we use Stanford CoreNLP to calculate the sentiment score for each tweet (1 for negative, 2 for neutral, and 3 for positive).

For model training and test, we use first 80% of the data as training set, and the rest of the data as test set. For different models, we’ve tried different combinations of features, and only report the best performances. TABLE1 and TABLE2 together summarize our results.

**Multi-class classification and neural networks**. The corn price movement classification problem is a multi-class classifications, since we have three movement classes(Up/Down/Neutral). Hence, we use the ***One-vs-Rest*** strategy to solve it as a binary classification problem. More specifically, for each class, we will train a model that could distinguish this class from the others. The AUPRC score is then calculated based on the average AUPRC score for each class. For multi-layer perceptron model, we use the **KERAS** framework, which is a high-level neural networks API, also written in Python and capable of running on top of most commonly used deep learning framework – TensorFlow, CNTK or Theano. In our experiments, we use TensorFlow as our backend for training and evaluation.

**Discussions.** Let’s first focus on the AUPRC score on test set. Clearly, all of models could beat the naïve random model (1/3 = 0.33333), but still, the highest accuracy (0.58 by Kernel SVM) we could achieve is less than 0.6. Moreover, we can also see from the chart that using more features could improve the performance a little bit. So generally speaking, treating financial time series as a classification problem is harder than model it as regression problem. To have a better classifier, we think using more different features, (and of course, more data), could be a promising direction.

Next, we compare the performance among the four models. Logistic regression and linear SVM have roughly the same performance, both in training set and test set. Since these two models are both used to classify linearly-separable data (but with different object function), it’s not surprising to see this result. Furthermore, the results from there two linear model also conveys the message that our data is not linearly-separable, thus, non-linearly model should do a better work here. As we expected, the (Gaussian/RBF) kernel-SVM indeed further increase the AUPRC score on both training and test set to around 0.6, which is the best we could achieve. However, the AUPRC score on training set for kernel-SVM is a little bit too high, over 0.95, which might have the overfitting problem. For multi-layer perceptron model, the result is a bit unexpected, which is the worst – only has 0.4667 and 0.4649 on training set and test set respectively. We think the problem might be the limit size of our training data. Neural networks indeed have much stronger learning abilities for more complex problem, but without enough data, some simple models (like logistic and SVM) could also beat it.

**From multi-class classification to binary classification.** Based on all our experiment results, we can find that multi-class classification problem on corn price movement is difficult. So we consider solve a simpler version of this problem, transforming the multi-class classification to binary classification. The transformation is simple – we drop the Neutral signal. The corn price would either go Up or go Down for the next day. In this way, we have a binary classification problem. We repeat the performance evaluation on all models and summarize the results in TABLE2. The conclusion is similar to multi-class case, but the overall performance for all models has about 10% increase.

**Conclusion.** In this section, we conclude all our experiments and findings, and provide a final recommendation on our prediction models. For multi-class classification problem (Up/Down/Neural), our recommended model is (Gaussian/RBF) Kernel-SVM. The feature vector (61 dimensions) uses past 60 days’ corn price and the average sentiment score of tweets from the current day. For training and testing, the feature vector is scaled by using Python scikit-learn preprocessing package. The reasons are the following: 1. We find that our data is non-linearly separable. Hence, a non-linearly classification model should be a better choice. 2. Kernel SVM is a simpler model compared to neural networks, both in coding and deployment. Moreover, multi-layer perceptron networks are hard to explain, while Kernel SVM 3. Given the limit size of data, neural networks may not be able to show its full strength. In sum, (Gaussian/RBF) Kernel-SVM is our best choice for classification problem.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | P1 | Annualized return in training | Annualized return in testing |
| Logistic |  | 7.72% | 11.93% |
| Linear SVM |  | 10.42% | 23.75% |
| Kernel SVM |  | 20.27% | 24.71% |
| Multi-layer Perceptron |  | 7.56% | -12.84% |