IEEE Data Science Workshop

IEEE Investment Ranking Challenge solution

(4th place)

Kirill Romanov

Lausanne, 06.06.2018

Outline

1. My background
2. Solution overview
3. Key solution components:
    - Data preparation
    - Modeling and stacking
4. Interesting findings and conclusions

1. Author background

- Masters degree in Economics (specialization international finance)
- Working in management consulting (strategy, supply chain management)
- Write the trading strategies as a hobby, before data science became so popular
- Very excited to new wave of AI (deep learning, reinforcement learning, development of classical ML algorithms) and try to apply these technology to finance field

Solution overview (1/2)

1. Data analysis

- Features are anonymized and preprocessed - **no room for analytical insight**
- Target values are normally distributed – **perhaps good situation for linear models**
- Features seems to be continuous (perhaps due to preprocessing) - limited possibility for mean encoding and categorical features embedding – **challenge for tree based models?**
- Relatively small set of data. **Possible problem with cross-validation and challenge for NN models**?
- Dataset has outliers and missing values. Also contains features with different scale – **need preprocessing if we wand deals with linear models**
- For each predicting period we have only 60% of data. **It could be problem for subsequent periods because they miss 40%+ of history**

2. My approach

- Cannot use econometric models and market insight? Let’s concentrate on pure ML pipeline (data preprocessing – feature generation – model optimization – validation)
- For me simple linear models were the best from the point of view of speed and quality
- Small dataset? Good opportunity to test different combinations of features

3. Final architecture

- Four scenarios
- Main difference between scenarios - different features
- Model I used – Ridge regression
- Parameters for model optimization:
  - Feature selection (get rid off inefficient features)
  - Time window selection (how much past periods to use for prediction)
  - Regularization parameter (alpha)

Solution overview (2/2)

Detail solution pipeline is here:

Stage	Steps	Scenarios 0	 1	 2	 3	 4
I Data preparation	Exploratory data analysis	• Check target distribution • Check relationship between target and features and detect non informative columns	• No EDA here. Findings from scenario 0 was used as a basis for scenarios 1-4			
	A Data preprocessing	• **Nans:** fill by zeros • **Outliers:** don't remove • **Scaling**: no scale	• **Nans:** fill by zeros initial dataset. After creating technical indicators **ffill** and **bfill** was used • **Outliers**: Remove 0.0005 percentile from each side • **Scaling**: MinMax scaler	• **Nans:** fill by zeros initial dataset. After creating new features (technicals) **ffill** and **bfill** was used to replace nans • **Outliers**: don't remove • **Scaling**: no scale	• **Nans:** fill by zeros initial dataset. After creating new features (technicals) **ffill** and **bfill** was used to replace nans • **Outliers**: don't remove • **Scaling**: MinMax scaler	• **Nans:** fill by zeros initial dataset. **Outliers**: don't remove • **Scaling**: MinMax scaler
	B Feature engineering	• **Basic features:** use all basic features. Group them and calculate **mean** for 6-month period	• **Basic features:** remove non-informative features . Group them **(mean and std)** for 6-month period • **Technical indicators** • Synthetic features (pair interactions): **subtract** and **multiply**	• **Basic features:** use all basic features. Group them **(mean and std)** for 6-month period • **Technical indicators**	• **Basic features:** use only the most important in best predictors from scenario 2 . Group them **(mean and std)** for 6-month period • **Technical indicators** • Synthetic features (pair interactions): **add**, **subtract**, **multiply, divide**	• **Only features created by PCA method** (explaining 99% of variability**).** They were created from grouped basic features and synthetic features (add, subtract, multiply, divide)
II Modelling	Feature selection and time-window selection	• This scenario was used only for EDA and don't pass through this steps	**Find best feature for one combination: prediction period – time window :** use recursive feature elimination (RFE) from sckit-learn library with **step=100**	**Find best feature for one combination: prediction period – time window :** use recursive feature elimination (RFE) from sckit-learn library with **step=10**	**Find best feature for one combination: prediction period – time window :** use recursive feature elimination (RFE) from sckit-learn library with **step=100**	**Find best feature for one combination: prediction period – time window :** use recursive feature elimination (RFE) from sckit-learn library with **step=5**
			Find best time window option: use grid search with all possible combinations of time window for each prediction period			
	Hyperparameters optimization (best alpha)		Use grid search with the following combinations of alphas: [1e-15, 1e-10, 1e-8, 1e-4, 1e-3, 1e-2, 1, 5, 10, 20]			
III Stacking	Select best model for each period	• Estimate validation dataset for each model set. Select the model with the best validation score. For period 2017_1 select the best model for the period 2016_2				

Data processing

I used the following types of features:

A. Grouped basic features

- In the initial dataset we have anonymized features for each month.
- As we have to predict 6-month return, these basic features were aggregated
- I use **mean** and **std** to aggregate six values of each feature for each period

C. Synthetic features

- As a basis I used grouped basic features (mean)
- Then, for each combination of feature 1, feature 2 from this subset I generate new features:
  - New Feature 1 = Feature 1 - Feature 2
  - New Feature 2 = Feature 1 + Feature 2
  - New Feature 3 = Feature 1 / (Feature 2 +0.01)
  - New Feature 4 = Feature 1 * Feature 2
- Depending on scenario, I generate different combination of basic features and different transformation types

B. Technical indicators

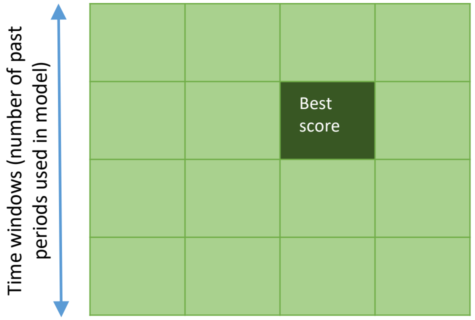
- **Main idea**: target value of current period could be basic feature for the next period (e.g predicted return for second half 2002 is the target for the first half of 2002 but feature for second half)
- However we couldn’t link it with the specific security. All we can is calculate mean value for the whole period. Additionally we can calculate aggregated technical indicators based on this value: Moving Average, Exponential Moving Average, Momentum, Rate of Change (see full list in paper)
- Finally, we can “encode” each row in the period by these aggregated values
- Unfortunately this type of features wasn’t efficient

D. PCA

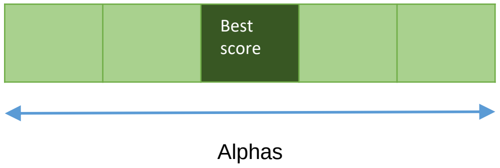
- At the first step I generated synthetic features
- Then, with python sklearn, I generate the components that explains at least 99% of variance and use them as a features

Modeling and stacking

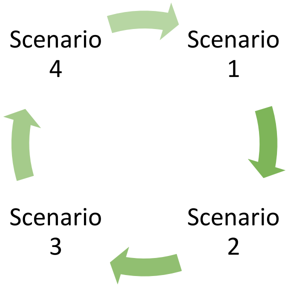
1. Train different combinations of models and check the results on validation dataset. For each prediction period. Then, select the model with the best Spearman score on validation dataset



2. Train with different regularization parameters (alphas) and select the models with the best score on validation dataset for each prediction period



3. Repeat for all scenarios



4. Find the best model from all scenarios for each prediction period based on best score on the validation dataset. For the period 2017_1 the best model for the period 2016_2 was used

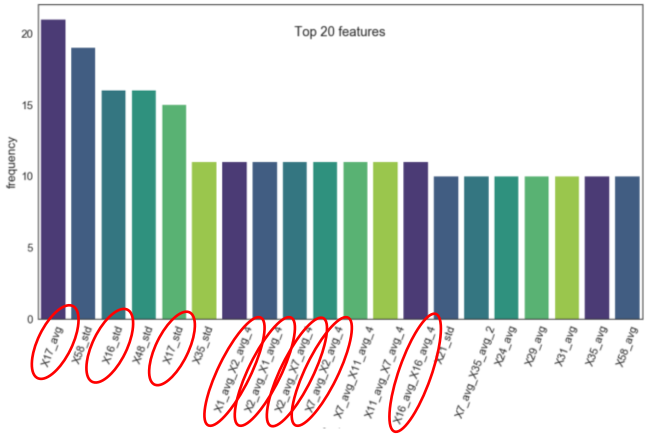
Period	Scenario 1	Scenario 2	Scenario 3	Scenario 4
2001_2	▨			
2002_1			▨	
2002_2				▨
……				
2017_!		▨		

Period	Best of breed model
2001_2	▨
2002_1	▨
2002_2	▨
……	▨
2017_!	▨

Interesting findings

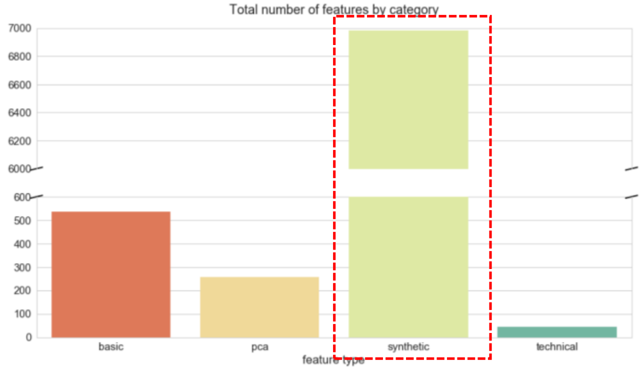
1. Top 20 features (best models).

X17, X16 and interactions of X2 with X1, X7, are top performers



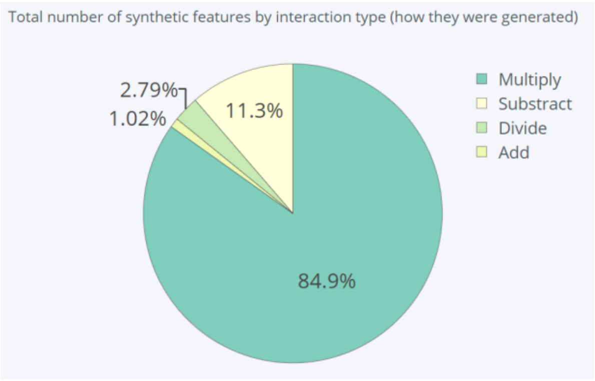
2. Total number of features by category.

- Basic features were the best.
- Synthetic features were quite useful as well
- PCA model moderately improved the score
- Generated technical indicators didn't work



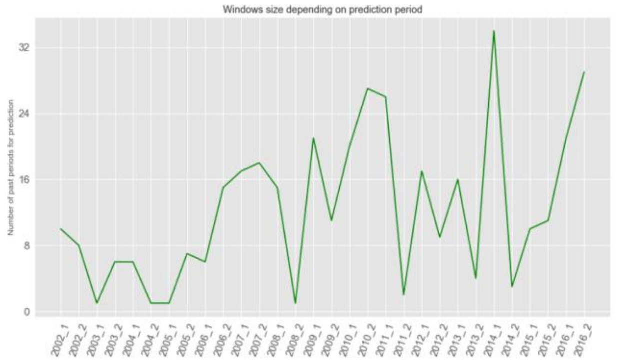
3. Total number of synthetic features by interaction type (how they were generated).

- Almost all good synthetic features were generated by multiply operation



4. Time window size depending on period

no clear dependency, however it seems like the models for the last periods try use more historical periods. Possible due to fact that we miss 40% of dataset for each predicting period



Conclusions

- For some reasons, the models developed as the part of this competition cannot be used in production
- However, they could be used as an additional insight for company analysts who know what each feature means and who can interpret these models
- Also this solution could be useful for data scientists, because some versions of this pipeline were used for other ML competitions and this approach were quite successful
- I'm going to upload full version of this solution with the presentation on GitHub https://github.com/kvr777/ieee-challenge
- In case of additional questions you can contact with me by email: kirill.v.romanov@gmail.com