



Ubiquant Market Prediction



Kaggle Team:
CS554-GroupDSS

Problem Definition

- ❑ Forecast the daily investment's return rate.
- ❑ The aim of this project is to predict obfuscated (unclear) metric which is relevant for making trading decisions from the features derived from real historic data from thousands of investments.



Criteria of Success

- The model is being scored based on the mean of Pearson correlation coefficient (also known as Pearson's r) for each time ID.

$$r = \frac{\sum (x_i - \bar{x}) (y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

Data Description

Training Data:

- **Rows:** 3,141,410
- **Columns:** 304
- **Input Attributes:** row_id, time_id, investment_id, market features[f_0,..., f_299]
- **Target Attribute:** target
- **Data Types:** time_id (int64), investment_id (int64), features (f_0, f_1, ...f_299) (float64)
- **Scale:** All the features (f_0, f_1, ... , f_299) follow the scale between -25 and 25.

Test Data:

- Unseen data with over a million rows

Data Preparation

- ▶ Missing or incomplete records
- ▶ Outliers or invalid data
- ▶ Inconsistent Data types



Missing or incomplete records

- ❖ We initially tried to check each and every cell to check whether there are any null values in the dataset.
- ❖ Using **isnull()** of pandas, we tested each and every cell for null values.
- ❖ Upon checking we did not find any null values and the data seems to be cleaned.

```
print(df.isnull().any().any()) #No Null Values
```

False

Outliers or invalid data

- ❖ In time series data, outliers should be further investigated to ensure they are not part of a seasonal (or other cyclical) trend that pops up every so often and may appear as an aberrant value.
- ❖ Since, the features here in the competition are hidden, removing the outliers is not recommended.
- ❖ However, we did the outlier test on all the features to check if there are any outliers, but there were no such aberrant values.

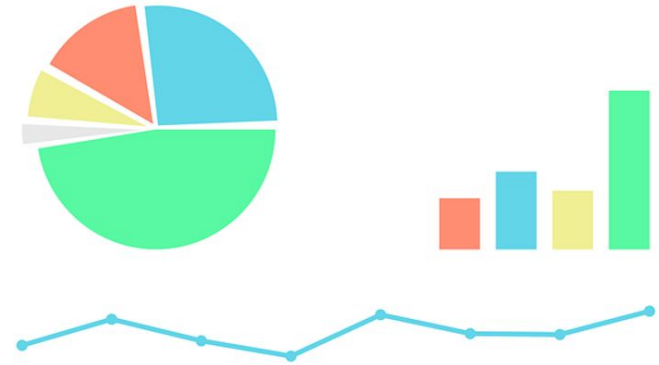
Inconsistent Datatype

- ▶ Testing all the columns of the dataset we tested that if there were any multiple datatype in a single column which might require standardization.
- ▶ Using dtypes() function of pandas to check if there were any different datatypes inside a particular column.
- ▶ Upon checking, all the columns were standardized either into float64 or int64.

```
time_id           int64
investment_id      int64
target            float64
f_0               float64
f_1               float64
...
f_295             float64
f_296             float64
f_297             float64
f_298             float64
f_299             float64
Length: 303, dtype: object
```

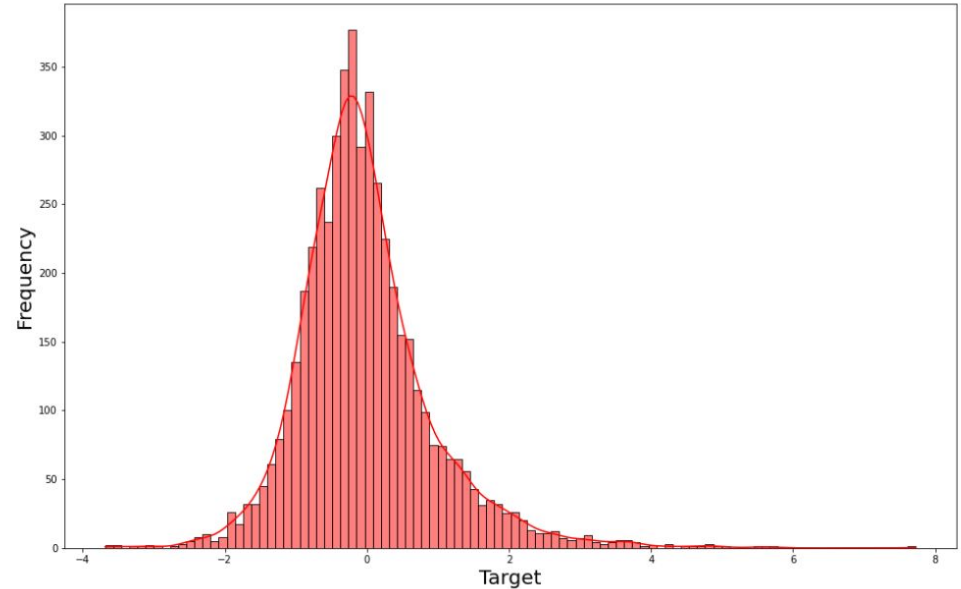
Data Exploration(Univariate)

- ▶ Objective: The main objective of univariate analysis is to explore all the variables in the data frame using statistical and visualization techniques.
- ▶ Visualization: We are using histograms to visualize the attributes.
- ▶ Descriptive Statistics: Frequency of observations, mode, variance, standard deviation, skewness, kurtosis.



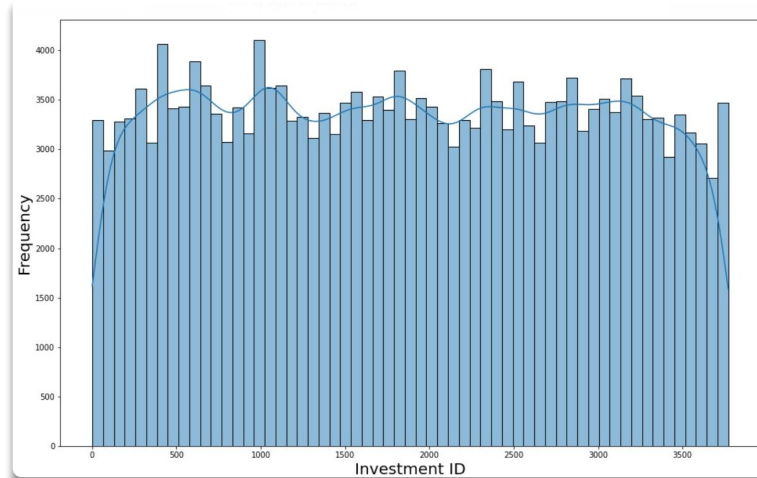
Frequency Count of 'target'

- ▶ The target variable is normally distributed with handful outliers, we cannot eliminate the outliers as those are price-based data and in real markets, prices are bound to fluctuate.



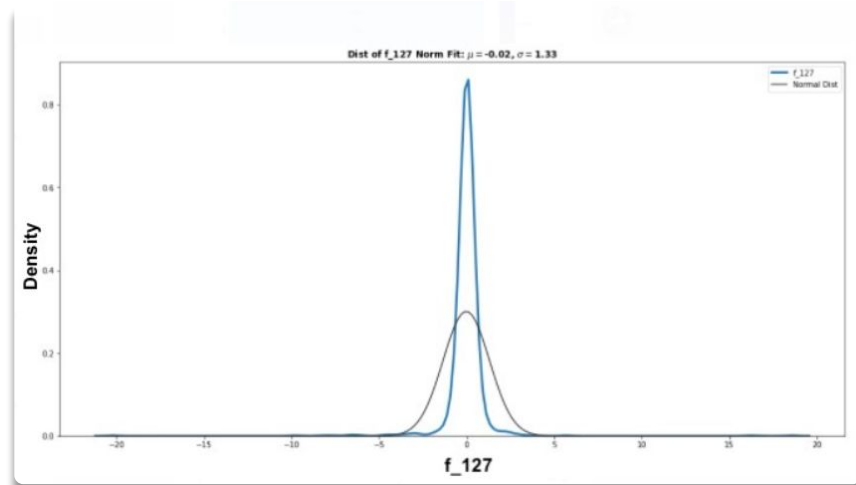
Frequency count of 'investment_id'

- ▶ As we can see from the histogram of Investment ID, it highlights the distribution of investment randomly and equally across different portfolio class.



Top features not following Gaussian distribution

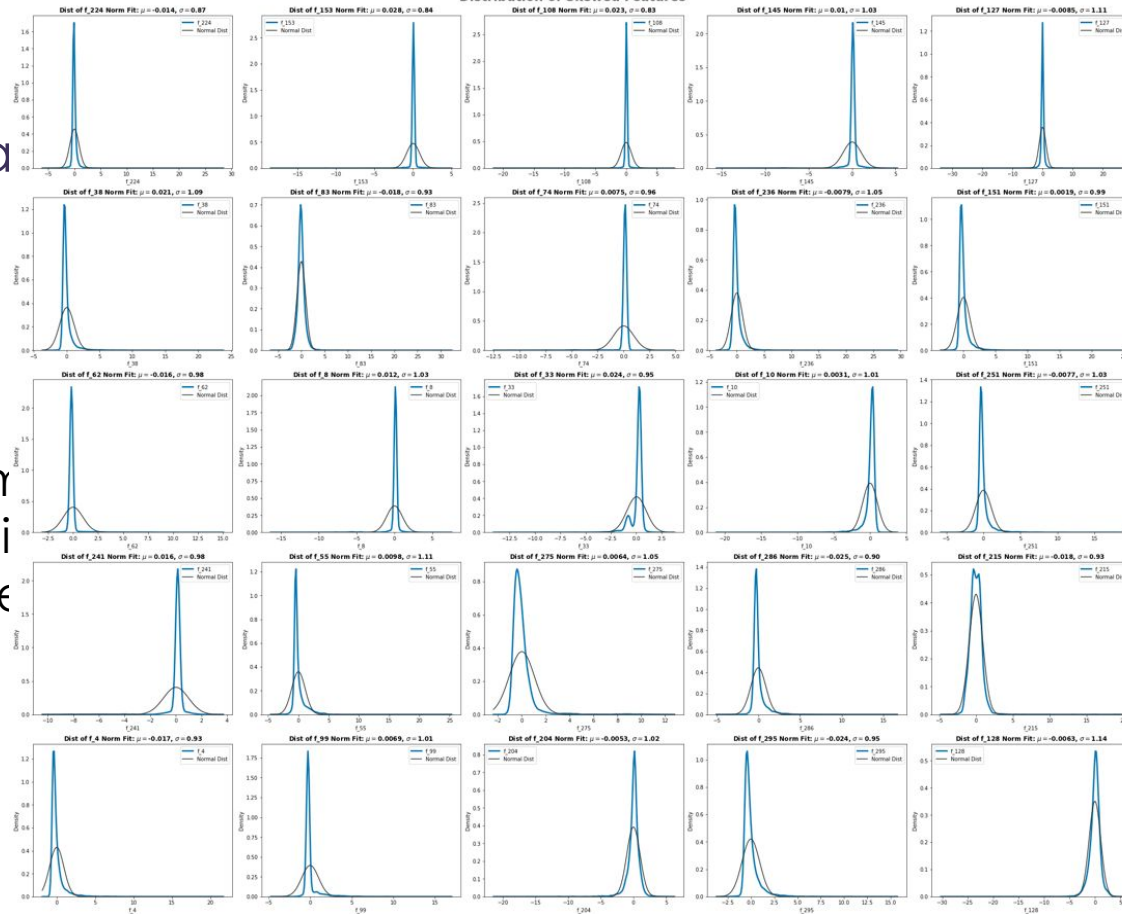
- From the features (f_0, f_1, \dots, f_{299}), the features not following gaussian distribution are observed and plotted.



Top fea

► From
havi
obse

Distribution of Skewed Features



Data Exploration(Bivariate)



OBJECTIVE: THE MAIN OBJECTIVE OF BIVARIATE ANALYSIS IS TO ANALYZE TWO VARIABLES IN ORDER TO UNDERSTAND THE RELATIONSHIP BETWEEN THEM.



VISUALIZATION: WE ARE USING SCATTERPLOT-MATRIX, HEATMAP, HISTOGRAMS AND JOINT-PLOTS TO VISUALIZE THE RELATIONSHIP BETWEEN THE ATTRIBUTES.

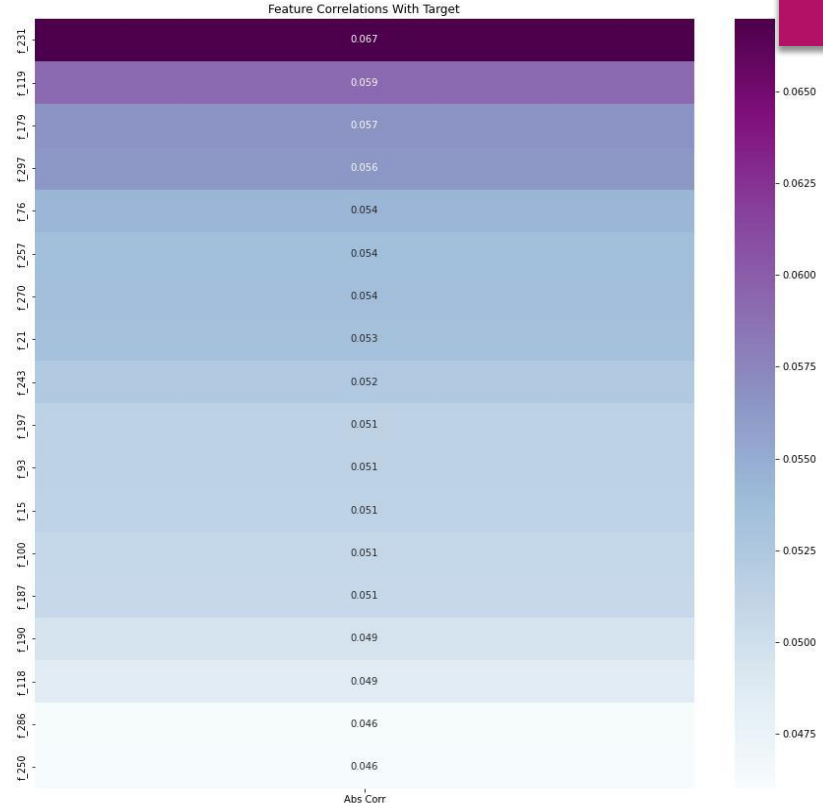


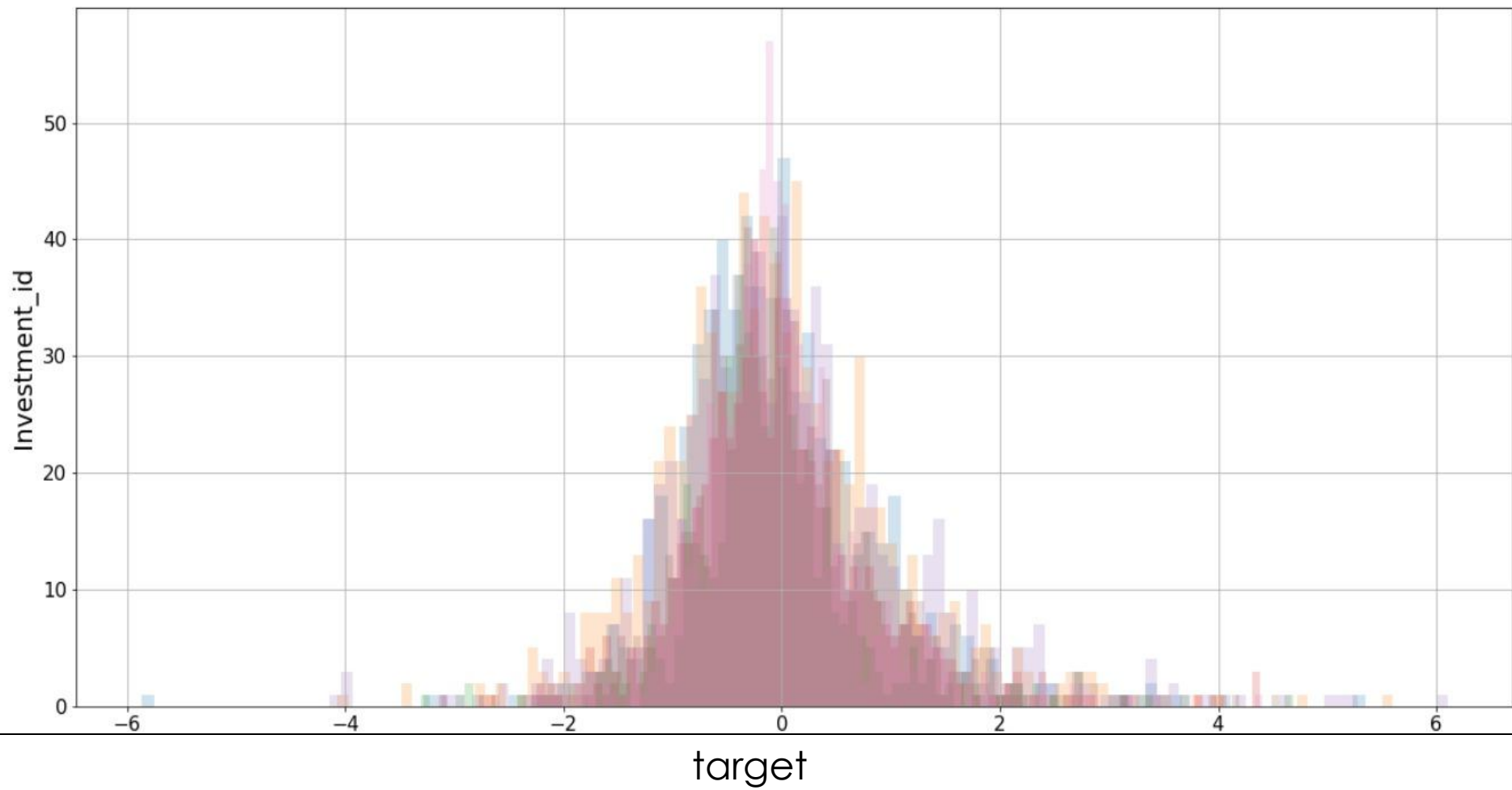
DESCRIPTIVE STATISTICS: LINEAR CORRELATION DETERMINES THE STRENGTH OF LINEAR RELATIONSHIP BETWEEN TWO VARIABLES.



Correlation of features with target

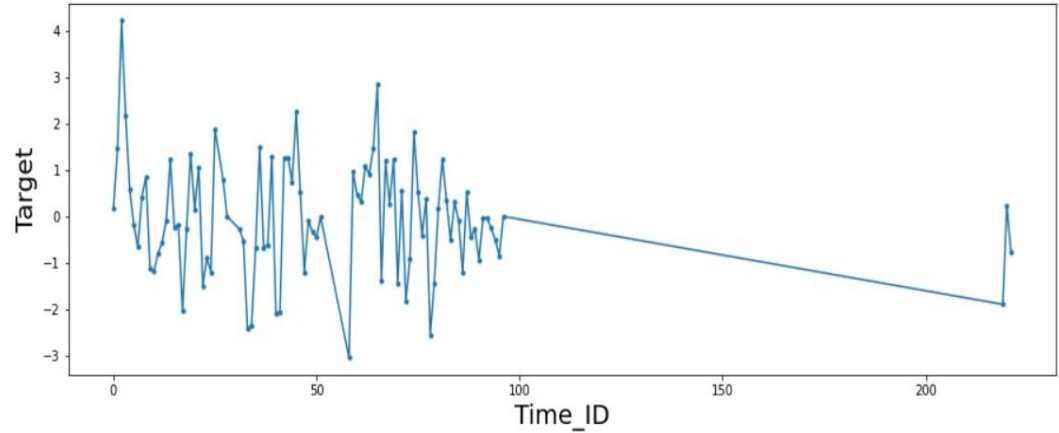
- ▶ Top features having maximum correlation with the target attribute whose correlation is greater than $\text{abs}(0.04)$.
- ▶ There is no perfect linear correlation between features and target.
- ▶ The main objective is to test the multicollinearity between the features and this could be the baseline step for performing feature engineering for our model.





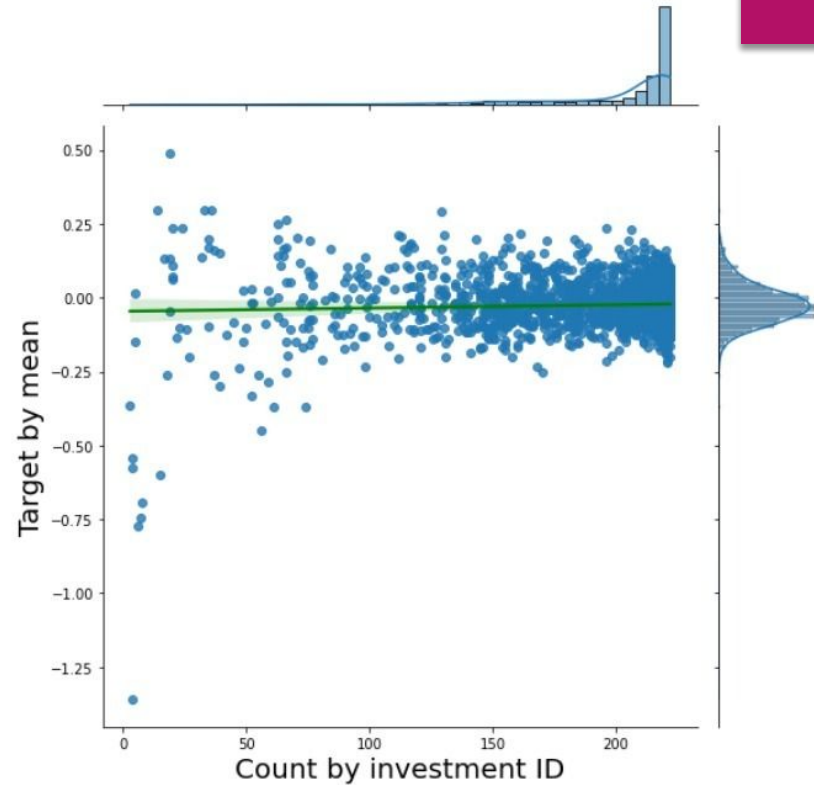
Time id vs target

- ▶ The below plot displays changes in target with respect to time_id for key investments.
- ▶ Plots for different set of time_id's also does not follow any pattern.



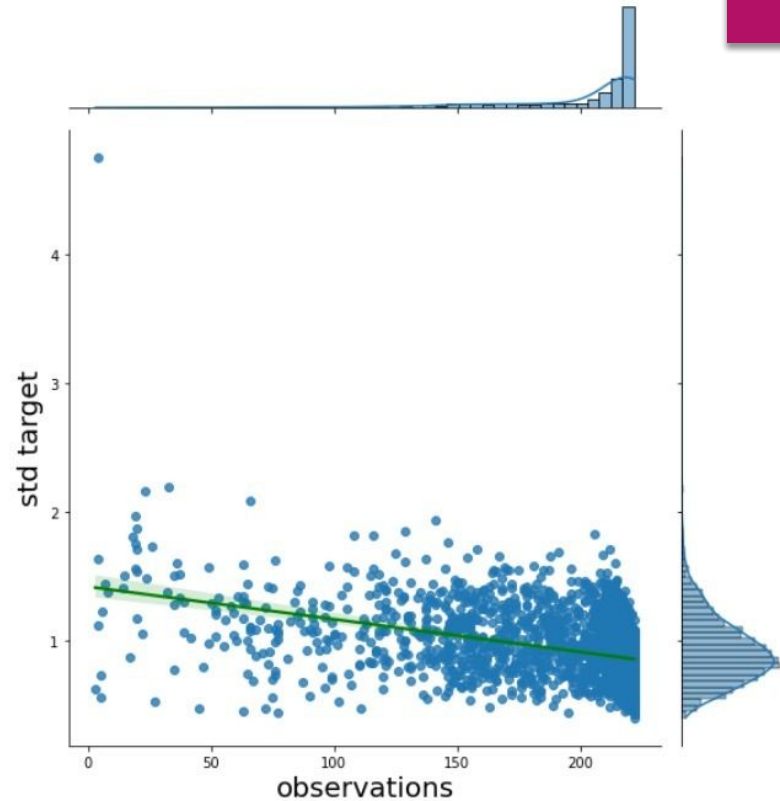
Investment count vs target dispersion

- ▶ This plot displays how the mean of target attribute converges with the increase in count of investment_id.
- ▶ As the number of investment_id increases the dispersion of target is reduced and it is more focused around the mean.



Investment count vs target dispersion using std.deviation

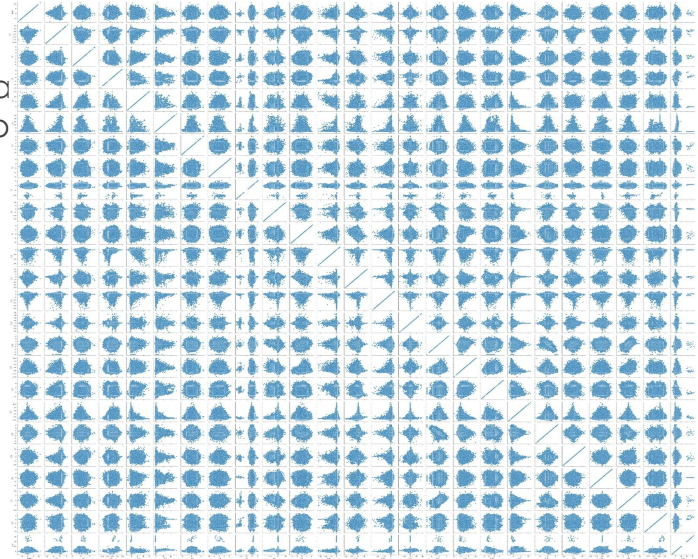
- ▶ This plot displays how the standard deviation of target attribute converges with the increase in count of investment_id
- ▶ As the number of investment_id increases the dispersion of target is reduced and it is more focused around the standard deviation.



Scatterplot matrix of features

	feature_a	feature_b	correlation
300	f_262	f_228	0.923792
302	f_205	f_148	0.919552
304	f_4	f_262	0.894998
306	f_4	f_228	0.886752
308	f_168	f_41	0.858884
310	f_211	f_254	0.846825
312	f_148	f_49	0.843790
314	f_164	f_61	0.843752
316	f_161	f_28	0.836947
318	f_49	f_205	0.832209

strongly correlated
following order of co



Modeling

- ▶ Econometric Model (VAR, VARMAX)
 - ▶ Time Series Model using Deep Learning (RNN, LSTM)
 - ▶ K-fold Combinatorics Cross Validation with Deep Learning
- ❖ We have analyzed the strengths of the above models and chosen K-fold Combinatorics Cross Validation with Deep Learning which is ideally suitable for our dataset.



Let's choose our model

K-Fold Combinatorics Cross Validation with Deep Learning	Econometric Models & Time Series Models using Deep Learning
1. Works better on time-series/non-time series data	1. Works well with only time series data
2. There is no restrictions on data.	2. Data is required to be free from trend/seasonality/volatility
3. Computationally inexpensive	3. Computationally expensive

Epoch 000,000 Learning rate 0.01 Activation Sigmoid Regularization None Regularization rate 0 Problem type Regression

FEATURES

Which properties do you want to feed in?

+ - 6 HIDDEN LAYERS

OUTPUT

Test loss 0.133
Training loss 0.126

Investment ID

Features



+ -

2 neurons

+ -

2 neurons

+ -

2 neurons

+ -

1 neuron

+ -

1 neuron

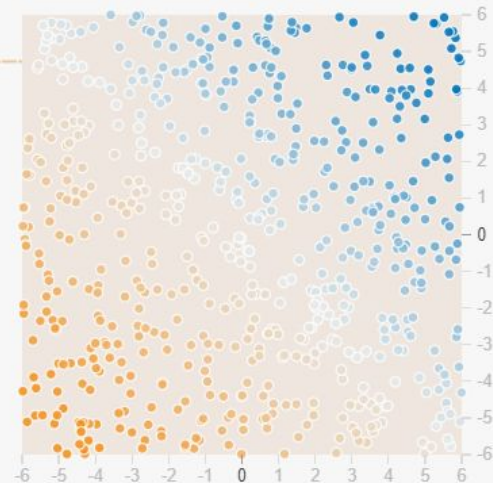
+ -

1 neuron



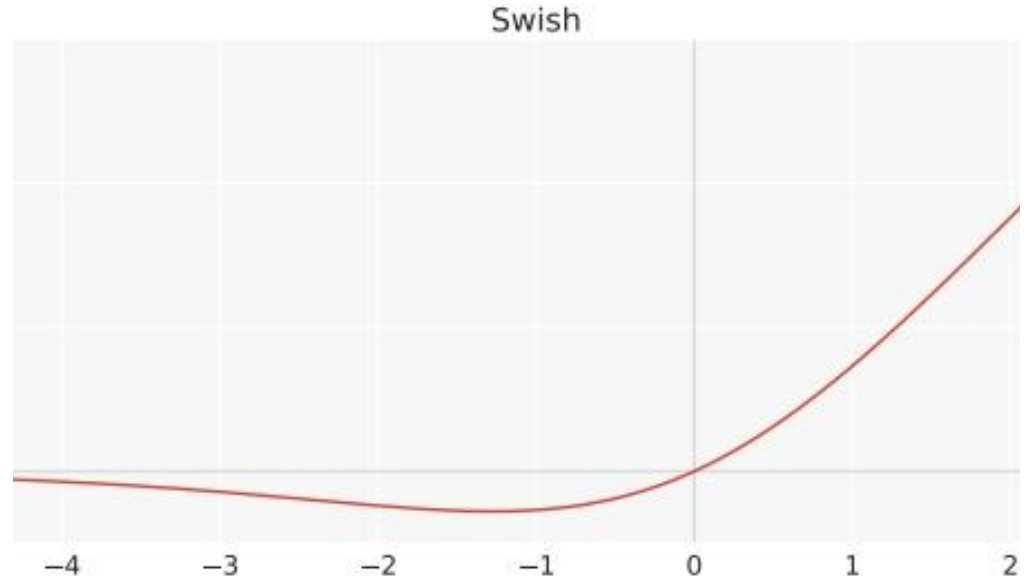
This is the output from one **neuron**. Hover to see it larger.

The outputs are mixed with varying **weights**, shown by the thickness of the lines.

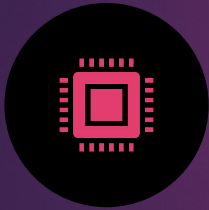


Deep Learning Parameters used for our models

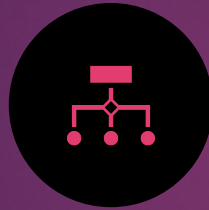
- ▶ Activation Function: Swish
- ▶ Model Regularization: L2
- ▶ Dense Layer: 256
- ▶ Dropout: 0.1
- ▶ Accuracy Metrics: RMSE
- ▶ Optimizer: adam



K-fold Combinatorics Deep Neural Network



WORKS WELL ON
TIME-SERIES/NON-TIME
SERIES DATA.



ENSURES THAT THERE IS
NO DATA LEAK
BETWEEN TRAINING
AND TEST SET, I.E.
SAMPLES IN BOTH THE
SETS ARE INDEPENDENT
OF EACH OTHER.



THIS METHOD GIVES US
EXACT NUMBER OF
COMBINATIONS OF
TRAINING/TESTING SETS
REQUIRED TO
CONSTRUCT A SET OF
BACKTESTING PATHS.



THE BELOW
COMBINATORIAL
EQUATION IS USED TO
CALCULATE THE
NUMBER OF PATHS.

5 fold cross validation: How it improved our score

Pearson Score: 0.09931402783912877

Pearson Score: 0.1233757040261945

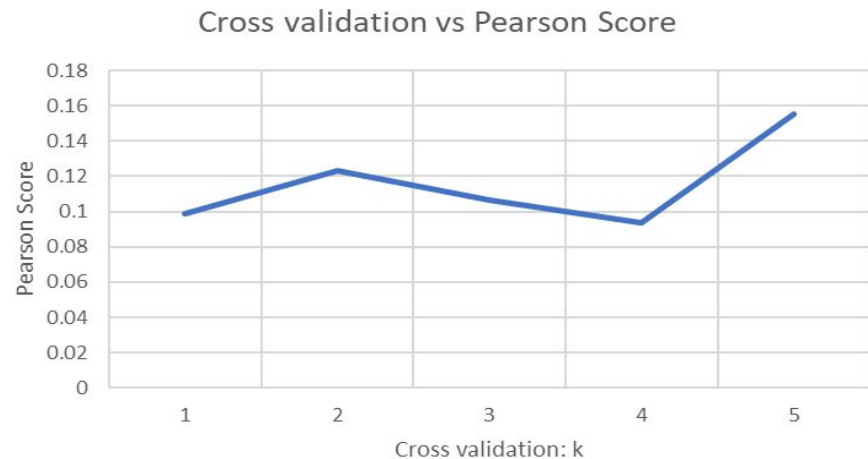
Pearson Score: 0.10629531770354542

Pearson Score: 0.09360857484560811

Pearson Score: 0.13427723084997586

CPU times: user 2min 33s, sys: 1min 14s, total: 3min 47s

Wall time: 2min 34s



Our best submission using this model

306

CS554-GroupDSS



0.1552

9

38m



Your Best Entry!

Your submission scored , which is not an improvement of your previous score. Keep trying!



We have implemented the model as mentioned in the previous slide.



We are still improving our model by integrating our model with other already posted models.



We will use ensembling techniques to merge our model



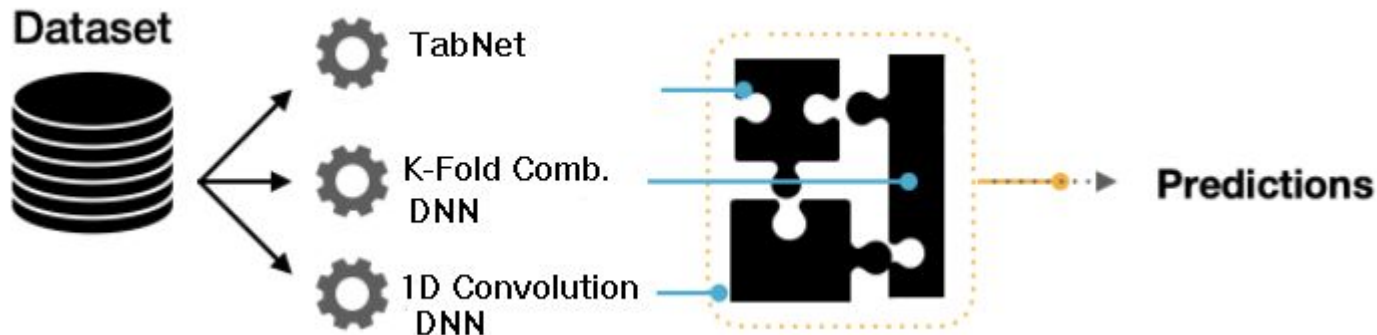
This will improve our rank significantly.

Ensemble Models

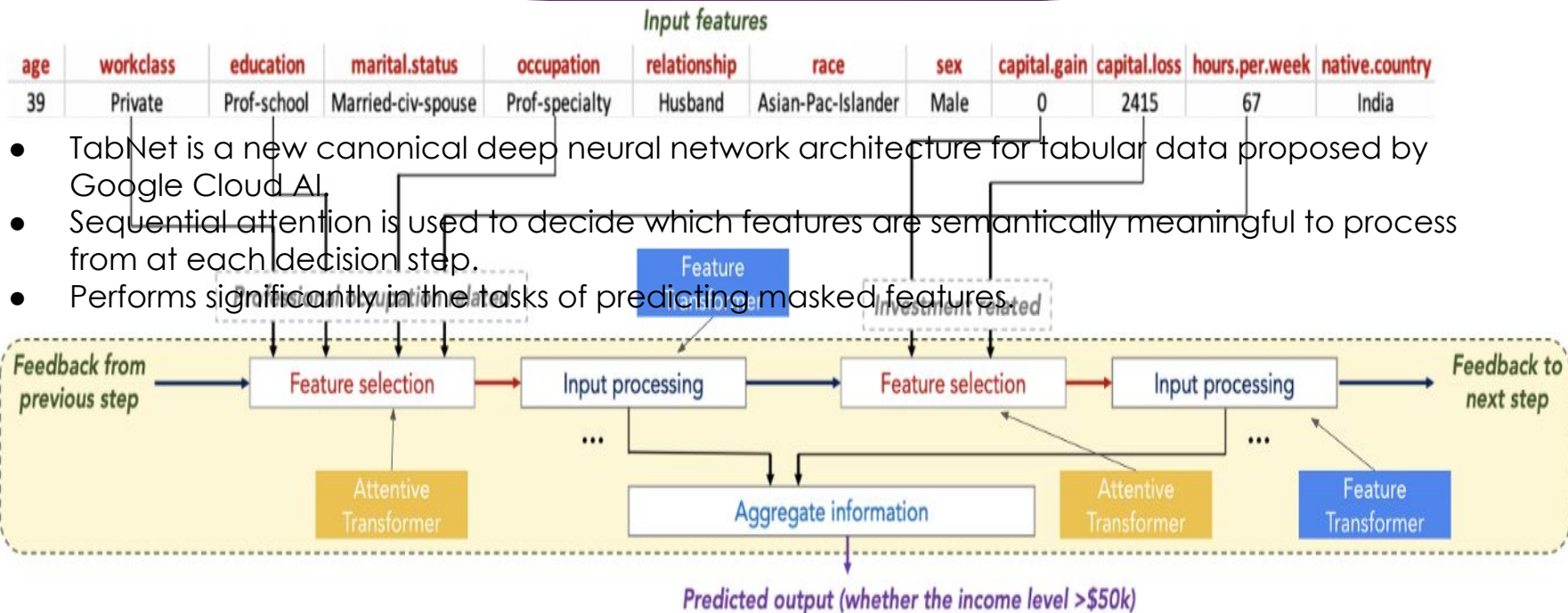
1. Tabnet
(pytorch)

2. K-fold
Combinatorics
Deep Neural
Network

3. 1D
Convolutional
Deep Neural
Network



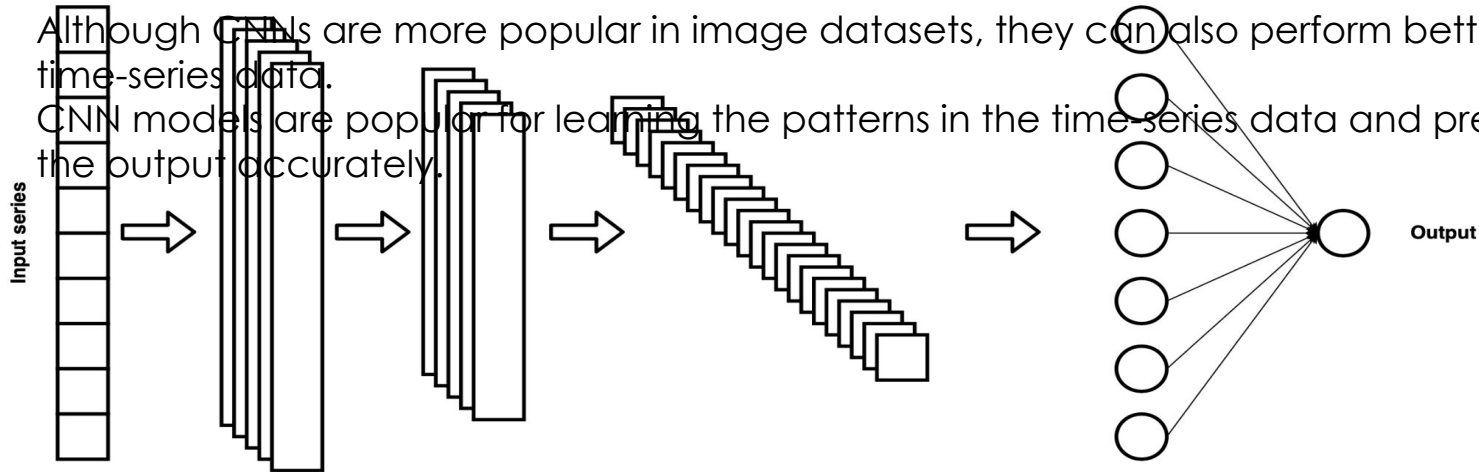
TabNet



1D Deep Convolutional Neural Network



- Although CNNs are more popular in image datasets, they can also perform better on time-series data.
- CNN models are popular for learning the patterns in the time-series data and predicting the output accurately.





Our best submission using ensembling

- The ensemble technique has helped in improving the average prediction performance over any contributing member in ensemble by reducing the variance component of prediction errors made by contributing models.

201

CS554-GroupDSS






0.1556

11

20m



• This technique of merging our models has helped us to achieve better pearson correlation coefficient than our last submission.

Your Best Entry!

Your most recent submission scored 0.1556, which is an improvement of your previous score of 0.0940.

• This score can be improved further by allocating the weights in the best possible combination.

[Tweet this](#)

Model Evaluation

- Our model was evaluated by taking the average of *pearson correlation coefficient* between forecasted value and the target, for each time time ID.
- The pearson correlation coefficient value can range from -1 to +1.
- A higher pearson correlation coefficient value indicates that the model has a better ability to forecast an investment's return.

Model	Pearson Correlation Coefficient
K-fold combinatorics deep neural network	0.1552
Ensemble Model	0.1556

Model deployment

The model deployment for Ubiquant Market Prediction consists of the following steps:

- **Step-1:** Import the ubiquitous python time series API
- **Step-2:** Initialize the ubiquitous environment
- **Step-3:** Run an iterator which loops over the test set (unseen market data) and sample submissions to make the prediction
- **Step-4:** Register the prediction



Thank
You