AI in Finance Assignment 6

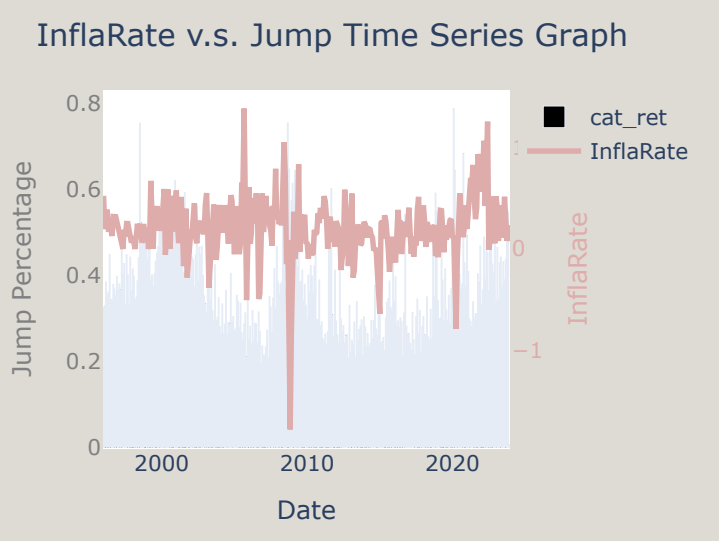
Yijia Zeng yzeng323@gatech.edu GTID: 903629003

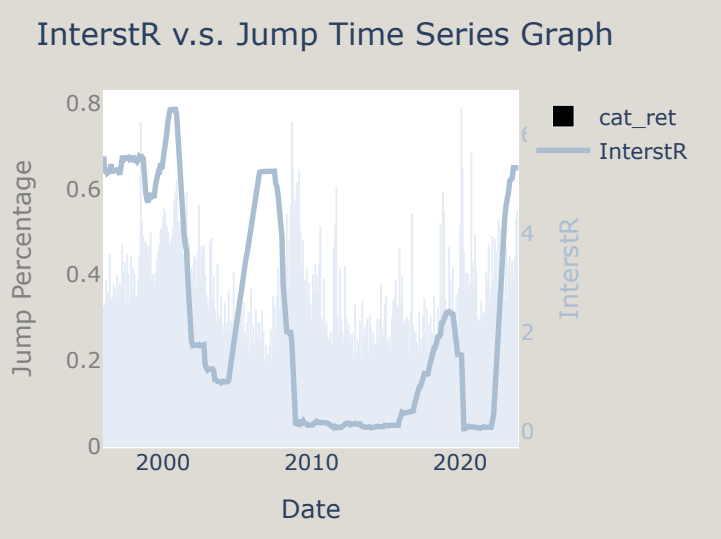
Data Loading

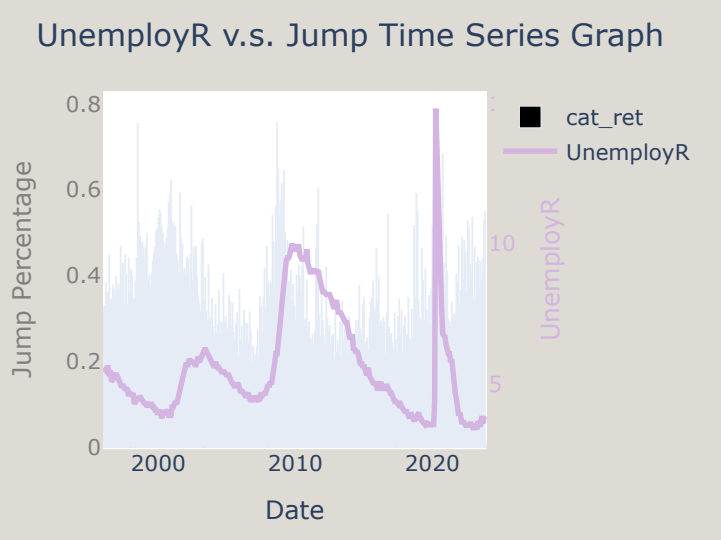
- msf data
- nber data
- SIFMA data
  - unemployment
  - interest rate
  - inflation
  - vix

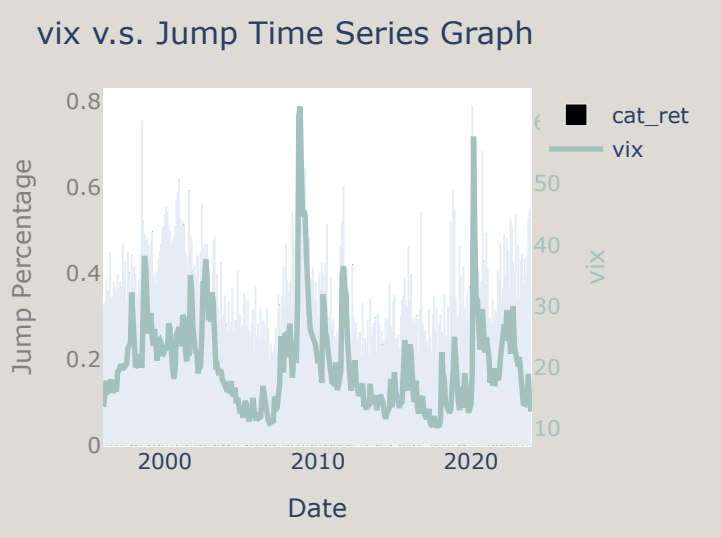
Categoricaln Outcome Var Construction

Percentage of Jumps Overtime









Discussion

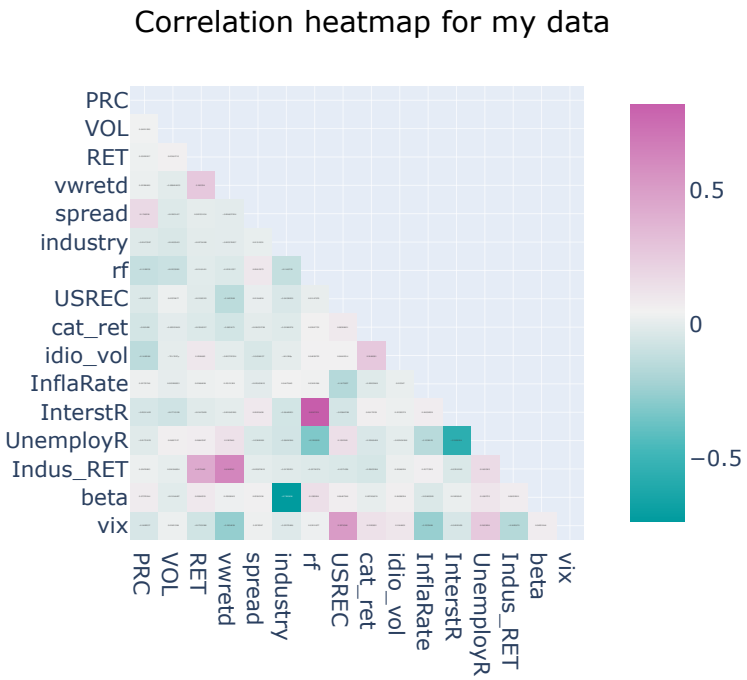
- According to the graph above, it seems that Interest rate follows the shape of the change of average jump for each month and its shape change a bit earlier than the shape of the average jump, serving as a precursor for upcoming higher likelihood of jumps. However, such pattern is not that clear for Unemployment rate and inflation rate. VIX seems to have a similar shape, but not necessarily a precursor. It seems that unemployment rate looks like a postcursor, and there is no obvious relationship between inflation rate and percentage of jump.

Generate Independent Variables

- MSF dataset
  - RET - 24 months
  - spread - 12 months
  - VOL - 12 months
  - PRC - 6 months
  - vwretd - 24 months
- Self Constructed Var
  - Beta from NeuralBeta - 6 months
  - Idiosyncratic Volatility from Linear Regression - 6 months： Calculated from Linear Regression, using a lookback window of 24 months, early period will be filled with value afterwards
  - Industry Average ret - 12 months
- MacroEconomic Variable
  - Interest Rate - 6 months
  - Recession Indicator - 6 months
  - Risk-Free Return - 6 months
  - Inflation Rate - 6 months
  - unemployment rate - 6months
  - VIX - 6months

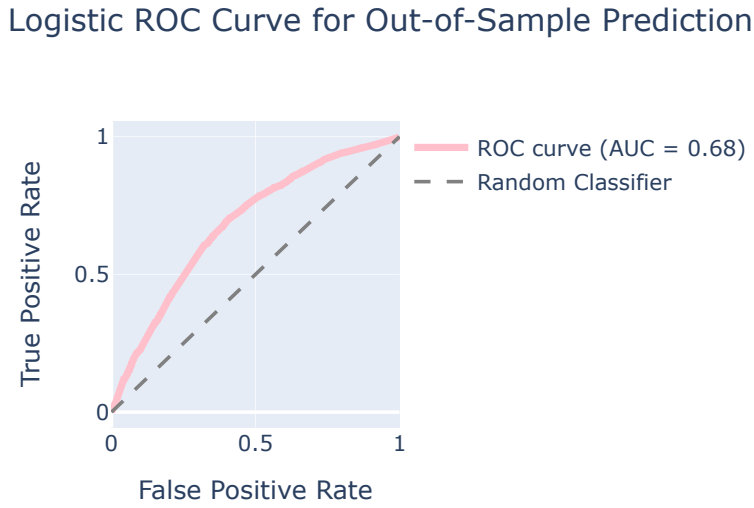
Logistic Regression

- Since multicollinearity in logistic regression can cause instability in coefficient estimates, we'll only consider variable themselves and exclude all lag variables.

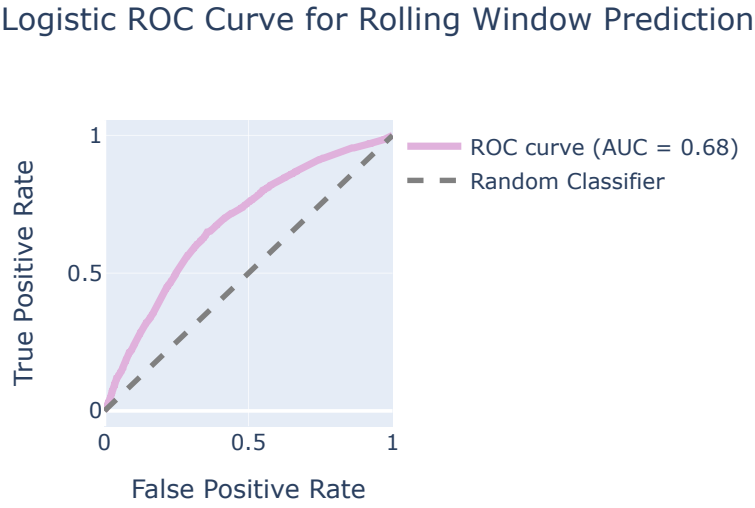


- Since Interest rate and risk free rate are highly correlated, we will only include interest rate
- Since Industry Return is highly correlated with vwretd and RET, we will drop Industry return
- SInce Beta and Industry seems to be highly correlated, we will drop Industry, which is a dummy variable

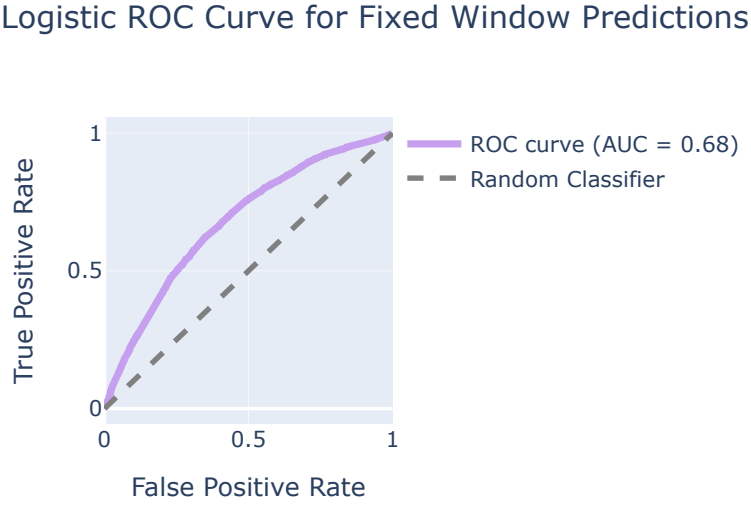
```
AUC: 0.68
KS Statistic (Scipy): 0.29
Misclassification Rate (Ridge): 0.37
```




```
AUC: 0.68
KS Statistic (Scipy): 0.29
Misclassification Rate (Ridge): 0.36
```




```
AUC: 0.68
KS Statistic (Scipy): 0.27
Misclassification Rate (Ridge): 0.36
```



Discussion

- According to the graphs and printed data, the Rolling window seems to have the best result, which has an AUC =0.68, and a KS = 0.60
- AUC = 0.68: This model will correctly rank a randomly selected positive instance (a jump) higher than a randomly selected negative instance (no jump). This is better than random guess but not a highly accurate model.
- KS = 0.29: There is a 29% difference between the CDF of the true positive rate and the false positive rate at some threshold.
- Feature used (t-1 data for each variable) : 'date',"PRC","VOL","RET","vwretd","spread","USREC","cat_ret","idio_vol", "InflaRate","InterstR","UnemployR","beta", 'rf', 'industry'

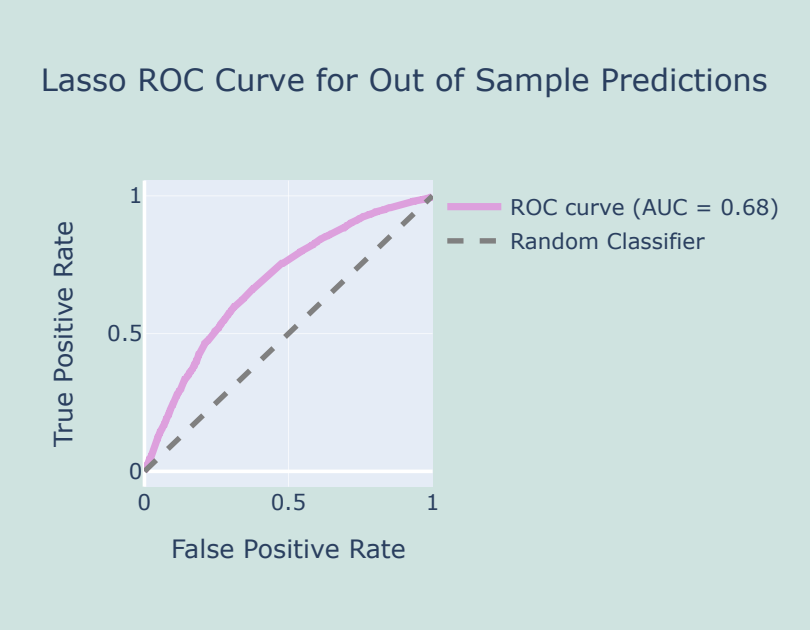
LASSO Logistic regression and Ridge Logistic Regression

Lasso

```
Optimal lambda (alpha): 0.003295496694633643
Selected features by LASSO: Index(['RET', 'vwretd', 'industry', 'USREC', 'idio_vol', 'InterstR', 'vi
x',
       'Indus_RET', 'RET_lag_1', 'RET_lag_2', 'RET_lag_3', 'RET_lag_4',
       'RET_lag_5', 'RET_lag_6', 'RET_lag_7', 'RET_lag_8', 'RET_lag_9',
       'RET_lag_10', 'RET_lag_11', 'RET_lag_12', 'RET_lag_23', 'vwretd_lag_2',
       'vwretd_lag_3', 'vwretd_lag_5', 'vwretd_lag_6', 'vwretd_lag_7',
       'vwretd_lag_9', 'vwretd_lag_12', 'vwretd_lag_17', 'vwretd_lag_22',
       'vwretd_lag_23', 'VOL_lag_10', 'spread_lag_2_x', 'spread_lag_6',
       'spread_lag_7', 'spread_lag_9', 'spread_lag_2_y', 'UnemployR_lag_5',
       'USREC_lag_4', 'Indus_RET_lag_5', 'vix_lag_3'],
      dtype='object')
```

- Use the 2018-2023 as the out of sample period to predict, since it performs similar with the remaining two methods

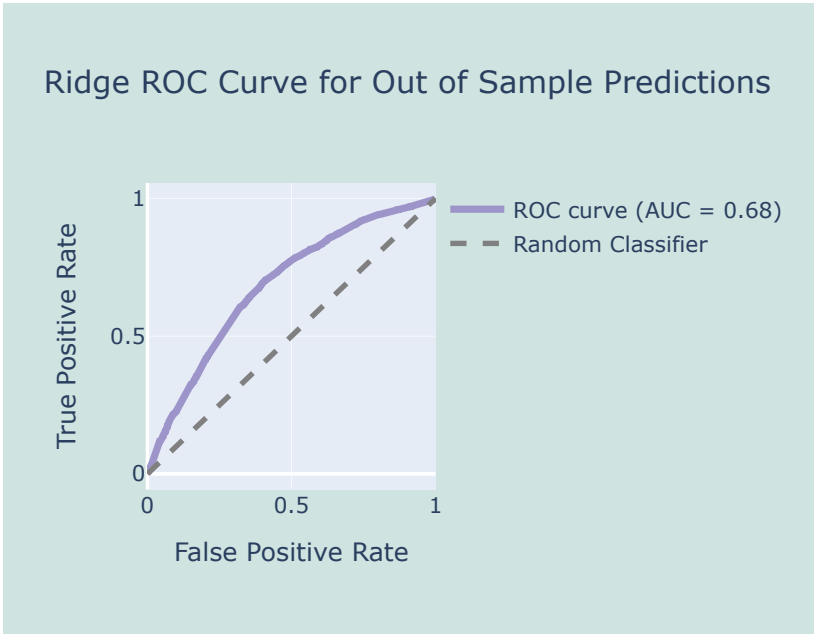
```
KS Statistic: 0.2904206277758545
Misclassification Rate (Ridge): 0.36
```



Ridge

- since multicollnearity is a concerns for ridge regression, we will only include same features as in logistic regression

```
Optimal lambda (alpha): 1000.0
KS Statistic: 0.2949282323330216
Misclassification Rate (Ridge): 0.37
```



Model KNN

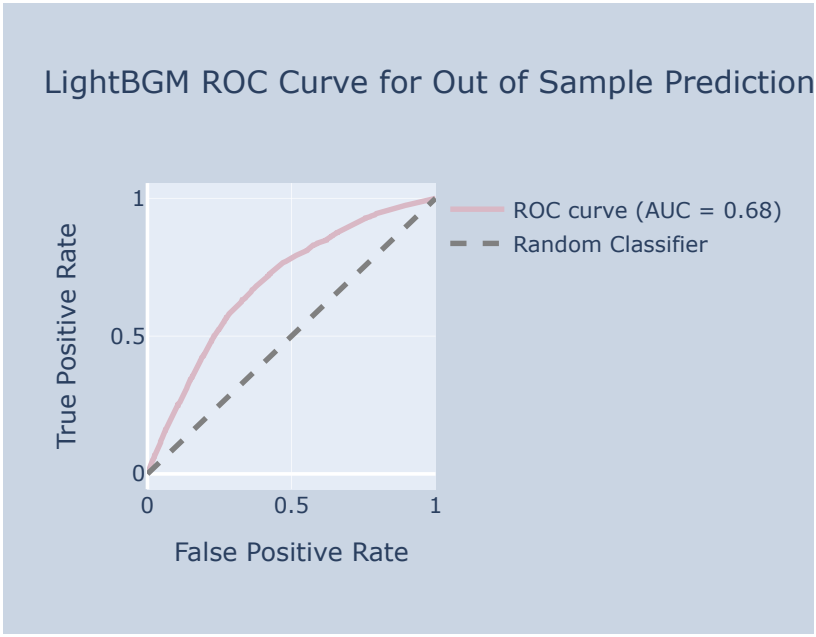
```
Misclassification rate: 0.40
Confusion Matrix:
 [[4086    5]
 [2746    7]]

Optimal K value: 28
Misclassification rate (KNN, Optimal K=28): 0.40
Confusion Matrix (KNN, Optimal K):
 [[4091    0]
 [2753    0]]
```

Discussion & Comparison

XGBoost Model

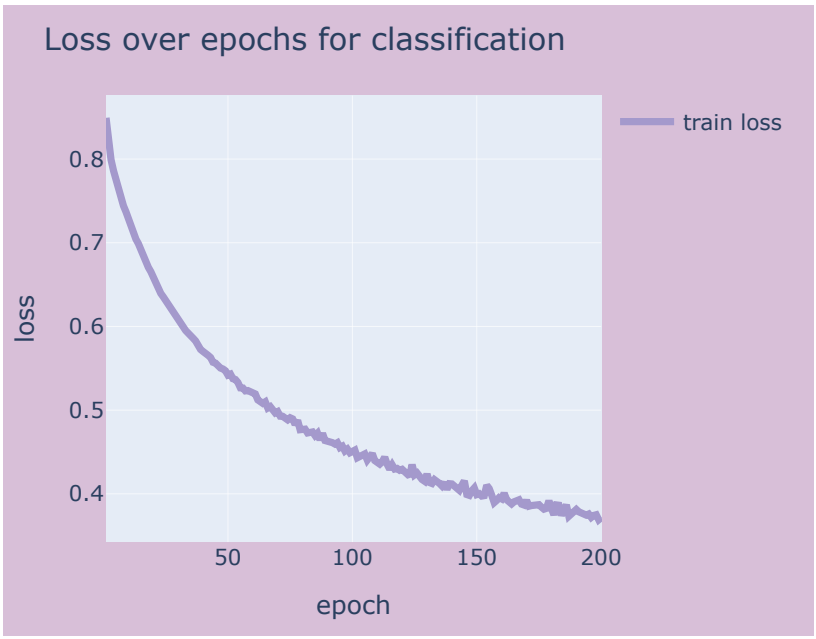
LightBGM Model

```
Best parameters for LightGBM: {'learning_rate': 0.01, 'max_depth': 3, 'n_estimators': 100}
KS Statistic: 0.30548439279546863
Misclassification Rate (LGBM): 0.36
```

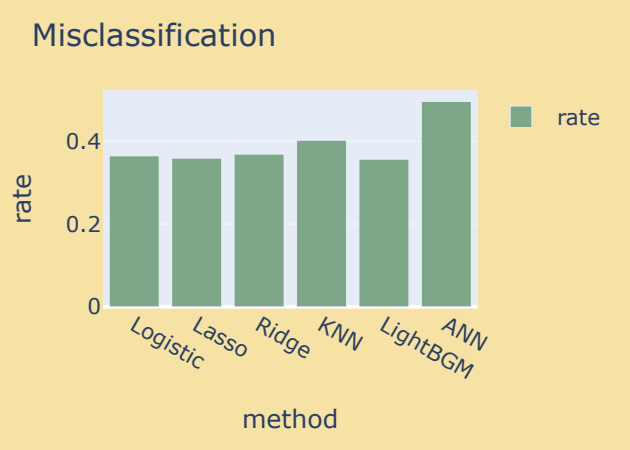


ANN Model

```
Epoch [25/200], Loss: 0.7834
Epoch [50/200], Loss: 0.4377
Epoch [75/200], Loss: 0.3585
Epoch [100/200], Loss: 0.6226
Epoch [125/200], Loss: 0.4810
Epoch [150/200], Loss: 0.3845
Epoch [175/200], Loss: 0.3021
Epoch [200/200], Loss: 0.4157
```



```
Test Accuracy: 0.5039
Misclassification rate: 0.4961

Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.70      0.67      4091
           1       0.23      0.19      0.21      1324
           2       0.26      0.24      0.25      1429

    accuracy                           0.50      6844
   macro avg       0.38      0.38      0.38      6844
weighted avg       0.48      0.50      0.49      6844


Confusion Matrix:
[[2855  560  676]
 [ 775  248  301]
 [ 819  264  346]]
```



Discussion & Comparison

- **Lasso v.s. Logistic**: According to the bar chart above, Lasso Regression has a similar performance as the Logistic Regression with similar misclassification rate of ~0.36, a similar KS statistics ~0.29, and a similar AUC. Lasso seems to have slightly lower misclassification rate. Since I performed feature selections to avoid multicollnearity variables for Logistic regression, Lasso also will perform an automatic feature selection, and also both models assume a linear relationship between the features and the log-odds of the outcome, they have similar behaviors.
- **Ridge v.s. Previous**: Ridge Regression also have a similar performance as the previous two models, with a slightly higher misclassification rate. Ridge Regression shrinks all coefficients towards zero but doesn't set any to exactly zero. This means it keeps all features in the model, potentially including some that add noise rather than signal. I used the same dataset for Logistic in Ridge regression, which suppose to enhance its performance. However, given Ridge is particularly good at handling multicollinearity and my data don't have significant nulticollinearity issue, this advantage is not relevant.
- **KNN v.s.Previous**: KNN has a higher misclassification rate compared with previous models. KNN is a non-parametric model, the corresponding flexibility of this model can lead to overfitting, especially with high-dimensional data. Moreover, stock market data often involves many features (dimensionality). KNN can struggle with high-dimensional spaces because as the number of dimensions increases, the concept of "nearest" becomes less meaningful. Also, Data points become sparse in high-dimensional spaces, making it harder for KNN to find truly similar instances.
- **LightBGM v.s. Previous**: LightBGM model performs similar with Lasso, Logistic, and Ridge with a slightly higher KS statistics, indicating it might have a better ability to distinguish between two classes. LightBGM can model non-linear relationships, which might account for the slight improvement in the KS statistic. Moreover, LightGBM performs feature selection inherently, similar to Lasso, which might explain why it doesn't underperform compared

to linear models. However, my current data might not include complex interactions or non-linear transformations that LightGBM could exploit.

- **ANN v.s. Previous**: The artificial Neural Network Method performed the worst among all models, with a misclassification of about 0.5, similar to random guess. ANNs are powerful but complex. If the underlying relationship in your data is relatively simple, as suggested by the better performance of linear models, an ANN might be overkill and struggle to find the signal. Moreover,ANNs can easily overfit, especially with limited or noisy data, leading to poor generalization. It is also possible for it to stuck in a local minimum.