USD INDEX TRADE RECOMMENDER SYSTEM

Kevin Glynn (mentor)

Srinivas Krishnaswamy (Student)

WHAT IS THE DOLLAR INDEX ?

It is a weighted geometric mean of the United States dollar (USD)'s value relative to following select currencies

- Euro (EUR), 57.6%
- Japanese yen (JPY), 13.6%
- Pound sterling (GBP),11.9%
- Canadian dollar (CAD), 9.1%
- Swedish krona (SEK), 4.2%
- Swiss franc (CHF) 3.6%

Established in 1973 with a value of 100. If it is above 100, USD is stronger than 1973, if below 100 then weaker than 1973 !

Factors Influencing the U.S. Dollar

1. Daily prices of the U.S.Dollar Index
2. Federal Reserve Fund Rate
3. Oil prices
4. Personal Consumption
5. Consumer Prices
6. Home prices
7. Inflation
8. Stock Market Prices
9. Social Security
10. Employment Numbers
11. Payroll Data
12. Employment growth
13. Federal retirement benefits
14. Saving rate of the residents of the U.S.
15. Home construction
16. Return on Investment (ROI) on the SnP

And many, many more not Listed here .....

Who cares and why ?

Traders who trade in bond, currency and gold markets within the world's capital market financial system.

The US Dollar is the world's reserve currency and as such any change in it's value can affect humans worldwide.

How can machine learning be used ?

If we can build a system to predict the next day's closing price correctly, we can make a trade - I.e. buy / sell the dollar index if prediction says it is going up (sell / buy if the dollar index goes down)

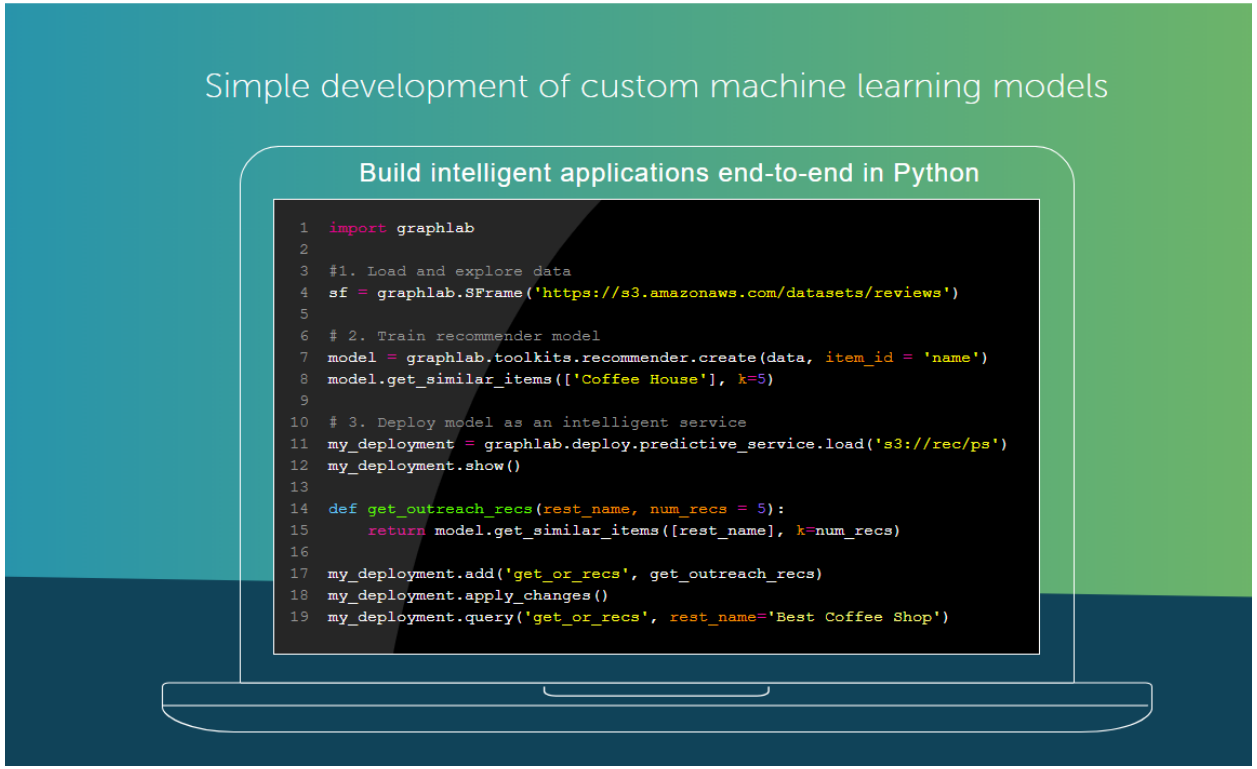
So let's build one …

Here are the steps –

1. Acquire data from websites that provide financial data – Quandl and investing.com
2. Split the data into train/test
3. Train model on the training dataset
4. Test the model on the testing dataset
5. Evaluate the test dataset
6. Operationalize the ML pipeline, either on a local computer or on the cloud
7. Usage -
   - Passing in the input dataset (features) should return the output price (label). The label in our case is the percentage gain or loss
   - If the percentage gain is within 1.5% +/-, then the user places a trade for the next day.
   - Close out the position by the close of business the next day (since we are using daily close prices as the label)

LOCAL OPERATIONALIZATION

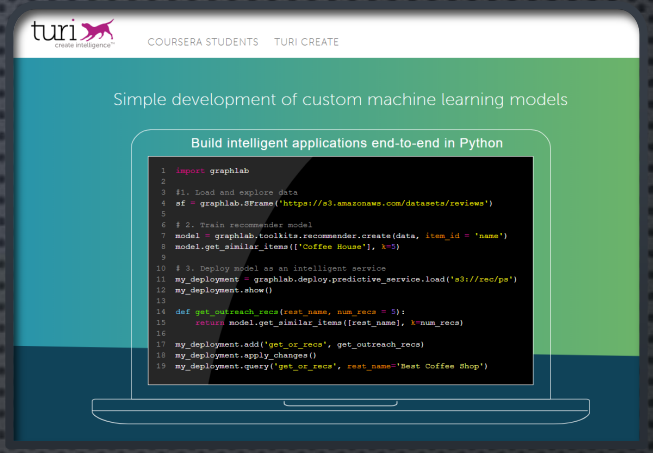
TURI CREATE



CLOUD OPERATIONALIZATION

MICROSOFT AZURE ML

WHICH ML MODEL SHOULD WE PICK ?



TURI CREATE ANALYZES THE DATASET AND PICKS THE BEST MODEL – IN THIS CASE GRADIENT BOOSTED DECISION TREES (GBMS).

IN AZURE ML 3 MODELS WERE USED, BOOSTED TREES, NEURAL NETS AND LINEAR REGRESSION. THE RESULTS WERE THEN COMPARED.

Building a Machine Learning Pipeline in GraphLab

In [41]:
```
def trade_workflow(path):

    # Get Data
    data = create_trade_sFrame(path)

    # Make a train-test split
    train_data, test_data = data.random_split(0.8)

    # Train model.
    model = train_trade_model(train_data)

    # Make recommendations
    results, sfTrade = recommend_trades(model, test_data)

    print(results)

    # Return the SFrame of recommendations.

    return sfTrade
```

Creating and starting a job in GraphLab (on the Local machine)

In [42]:
```
import graphlab as gl

# Deploy the job locally
job_local = gl.deploy.job.create(trade_workflow, path = 'D:/SpringBoard/Capstone2/Data/US Dollar Index Historical Data_17Yr.csv')
```

```
[INFO] graphlab.deploy.job: Validating job.
[INFO] graphlab.deploy.job: Creating a LocalAsync environment called 'async'.
[INFO] graphlab.deploy.job: Validation complete. Job: 'trade_workflow-Sep-16-2018-11-39-07' ready for execution.
[INFO] graphlab.deploy.job: Job: 'trade_workflow-Sep-16-2018-11-39-07' scheduled.
```

Getting the status of a job in GraphLab (on the Local machine)

In [46]:
```
# get status immediately after creating this job.
job_local.get_status()
```

Out[46]:
```
u'Completed'
```

LOCAL OPERATIONALIZATION

TURI CREATE (GRAPH LAB)

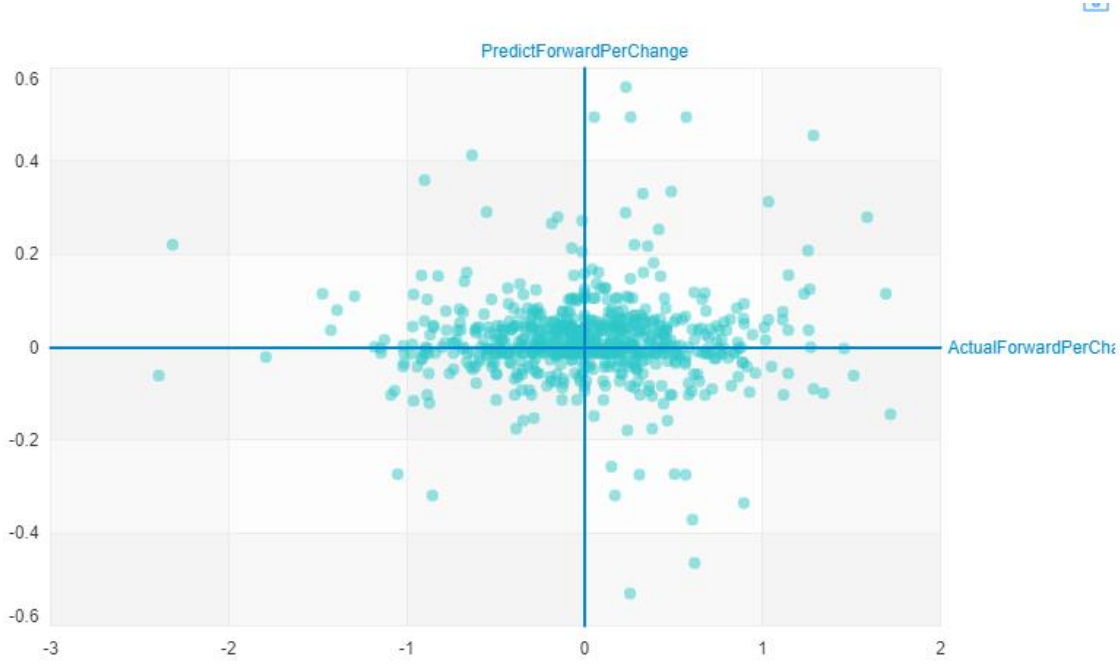
In [29]:
```
model
```

Out[29]:
```
Class                          : BoostedTreesRegression

Schema
------
Number of examples             : 2585
Number of feature columns      : 55
Number of unpacked features    : 55

Settings
--------
Number of trees                : 10
Max tree depth                 : 6
Training time (sec)            : 0.0469
Training rmse                  : 0.5994
Validation rmse                : 0.4986
Training max_error             : 19.7859
Validation max_error           : 2.2403
```

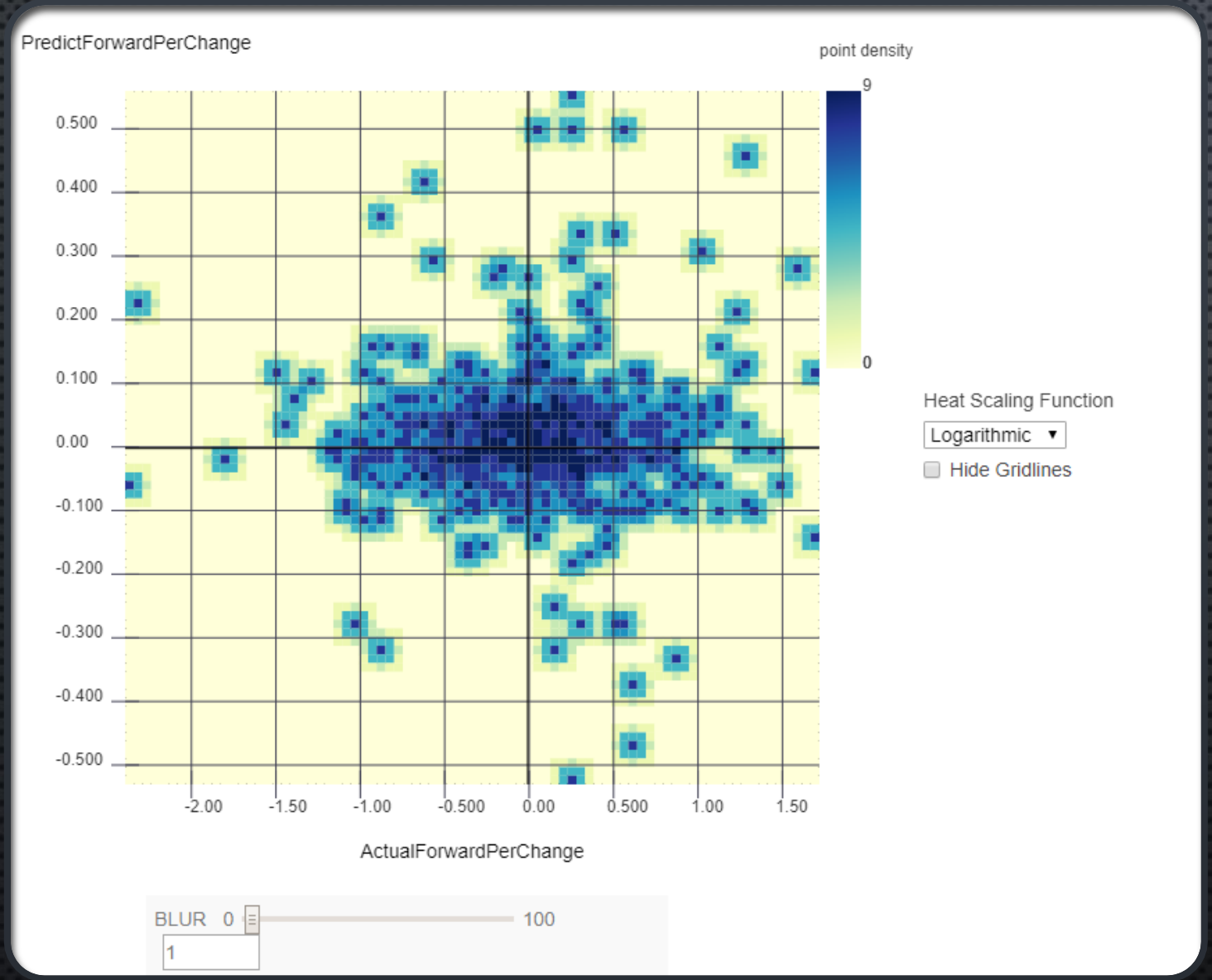
Make predictions and evaluate the model ...

In [30]:
```
#Make predictions and evaluate results.
predictions = model.predict(test_data)
results     = model.evaluate(test_data)
print(results)
```

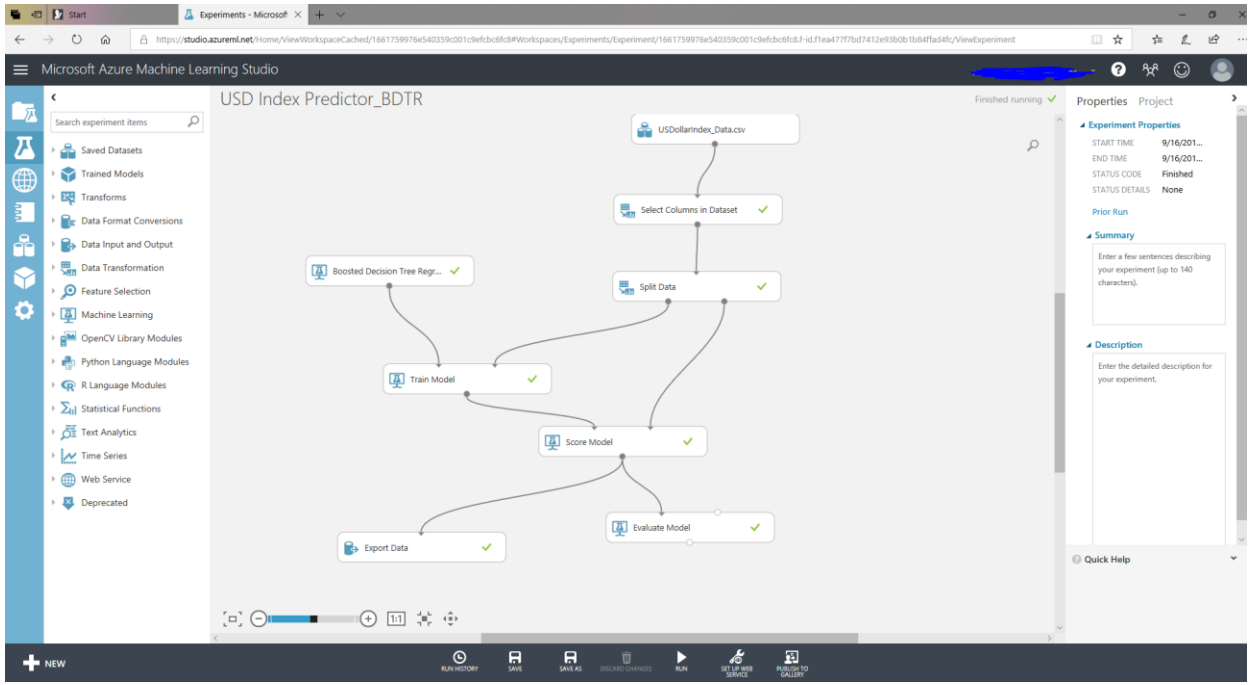
```
{'max_error': 2.534599542617798, 'rmse': 0.5210959975858873}
```


PredictForwardPerChange
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-3
-2
-1
0
1
2
ActualForwardPerCha

TURI CREATE


PredictForwardPerChange
point density
9
0
0.500
0.400
0.300
0.200
0.100
0.00
-0.100
-0.200
-0.300
-0.400
-0.500
-2.00
-1.50
-1.00
-0.500
0.00
0.500
1.00
1.50
ActualForwardPerChange
Heat Scaling Function
Logarithmic
Hide Gridlines
BLUR 0
100
1

TURI CREATE



CLOUD OPERATIONALIZATION

AZURE ML – BOOSTED DECISION TREES REGRESSION

Microsoft Azure Machine Learning Studio

USD Index Predictor_BDTR

Finished running

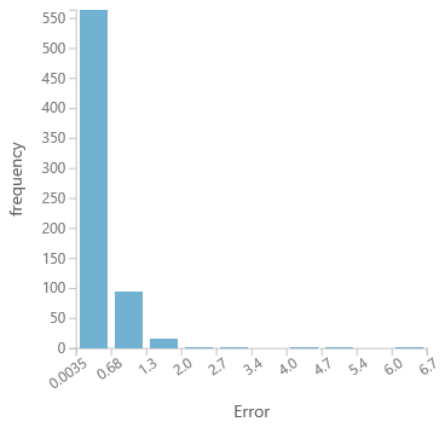
Properties Project

USD Index Predictor_BDTR › Evaluate Model › Evaluation results

Metrics

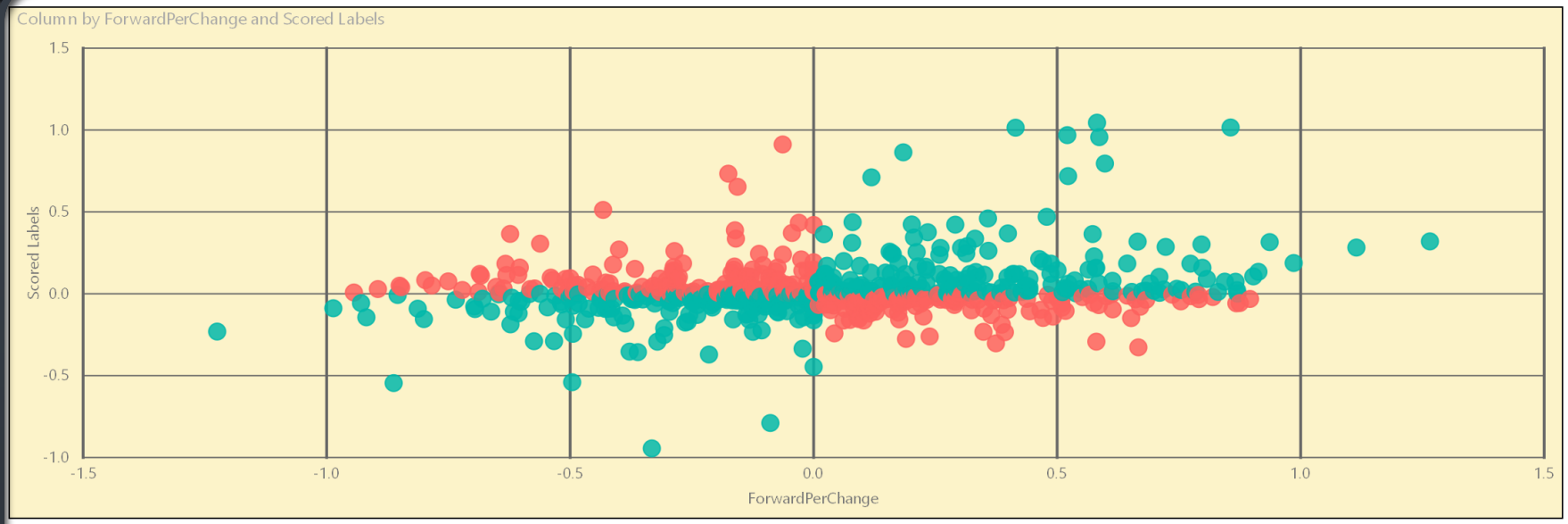
Mean Absolute Error	0.422229
Root Mean Squared Error	0.677442
Relative Absolute Error	1.164972
Relative Squared Error	1.949284
Coefficient of Determination	-0.949284

Error Histogram



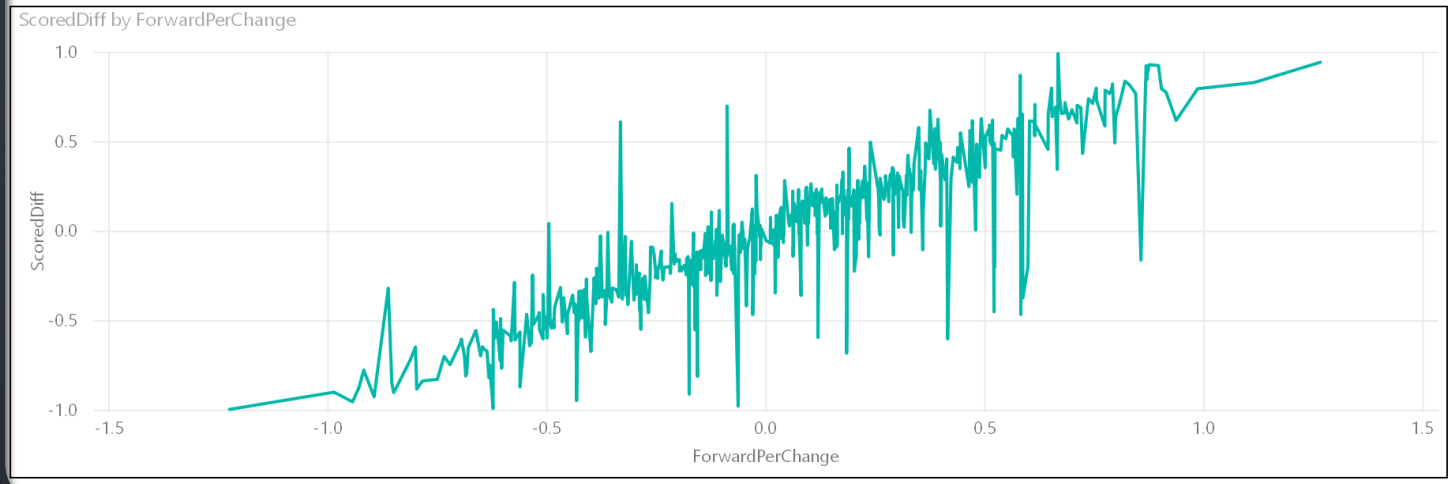
NEW

RUN HISTORY SAVE SAVE AS DISCARD CHANGES RUN SET UP WEB SERVICE PUBLISH TO GALLERY



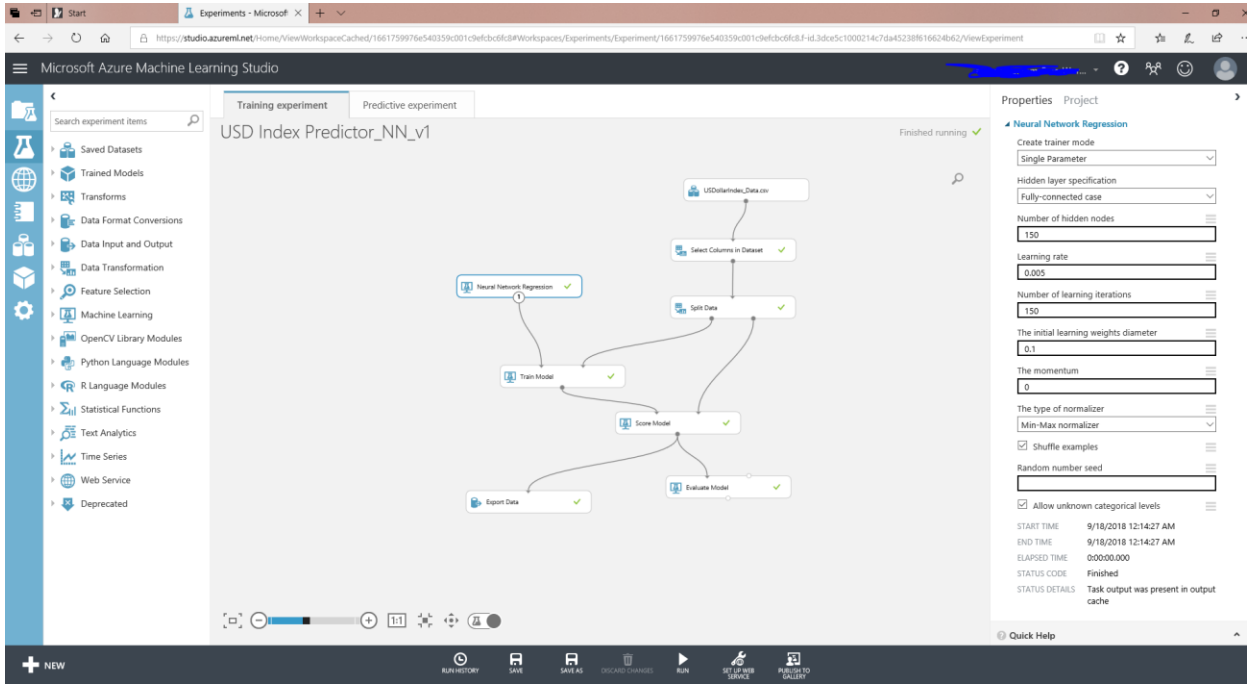
348
Profitable Trades

629
Number of Days



Index	ForwardPerChange	Scored Labels	ScoredDiff
623	0.67	-0.33	0.99
541	1.27	0.32	0.95
355	0.87	-0.06	0.93
383	0.90	-0.03	0.93
145	0.87	-0.06	0.93
14	0.58	-0.29	0.87
430	0.87	0.00	0.87
96	0.87	0.02	0.85
324	0.82	-0.02	0.84
212	1.11	0.28	0.83
153	0.79	-0.03	0.82
622	0.83	0.01	0.82

Boosted Decision Trees Regression



CLOUD OPERATIONALIZATION

AZURE ML – NEURAL NETWORK REGRESSION

Microsoft Azure Machine Learning Studio

Training experiment | Predictive experiment

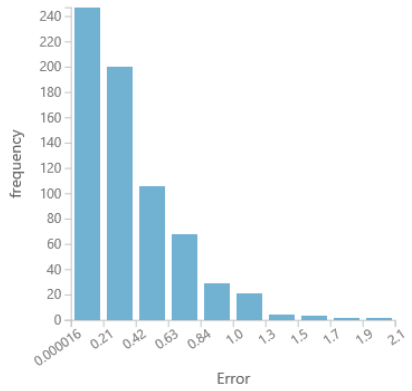
Properties | Project

USD Index Predictor_NN_v1 › Evaluate Model › Evaluation results

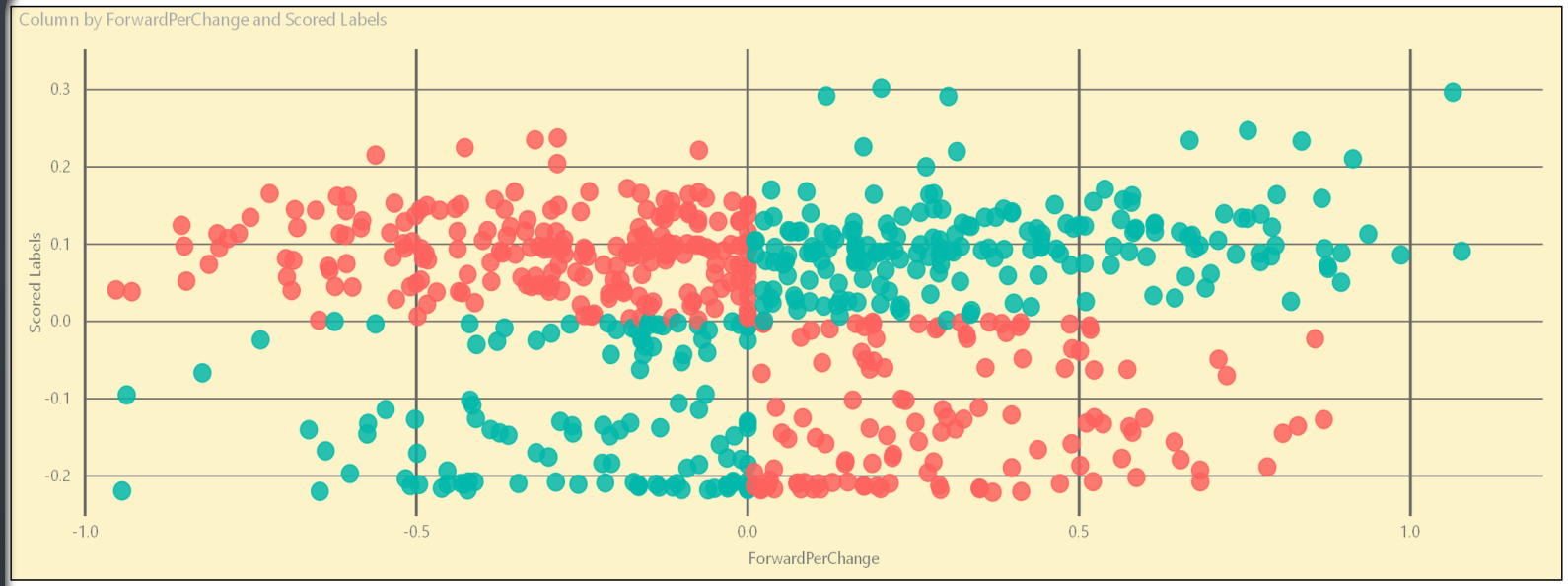
Metrics

Mean Absolute Error	0.381917
Root Mean Squared Error	0.502182
Relative Absolute Error	1.053748
Relative Squared Error	1.071158
Coefficient of Determination	-0.071158

Error Histogram

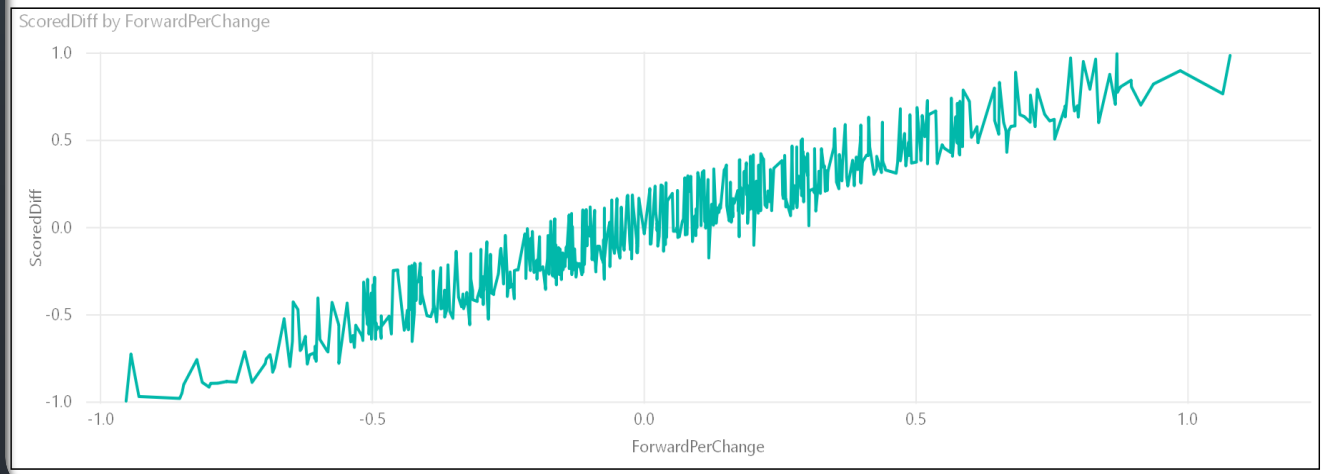


NEW | RUN HISTORY | SAVE | SAVE AS | DISCARD CHANGES | RUN | SET UP WEB SERVICE | PUBLISH TO GALLERY



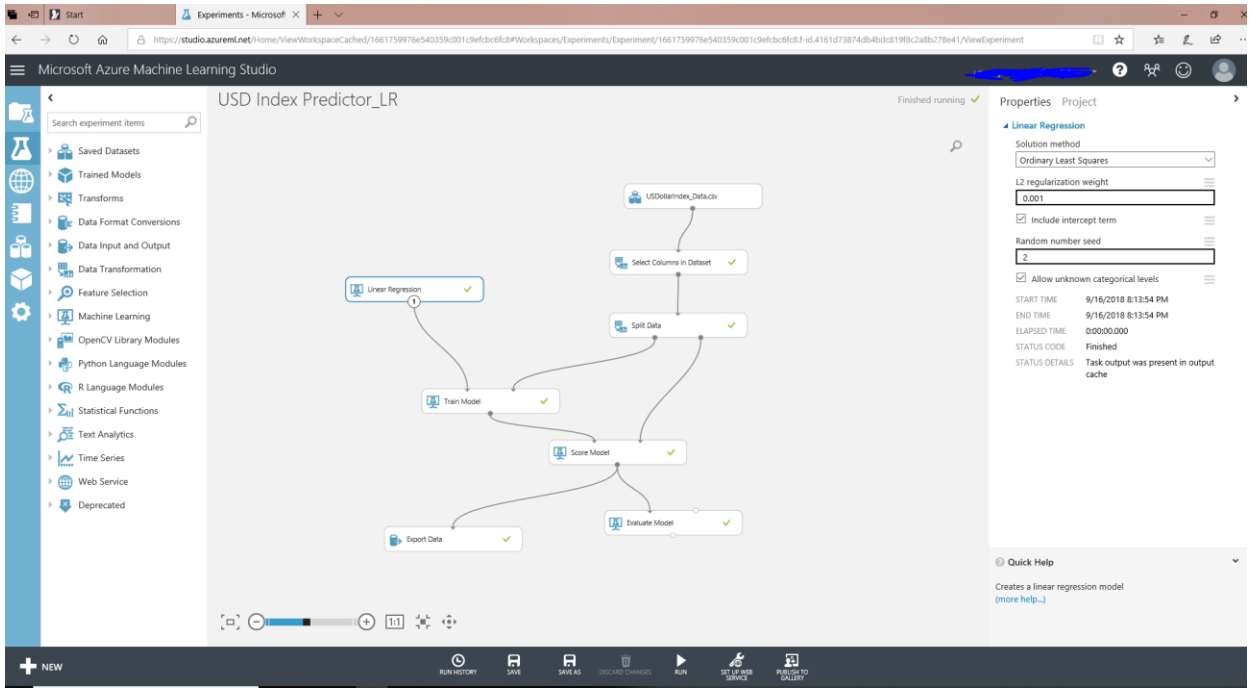
307
Profitable Trades

644
Number of Days



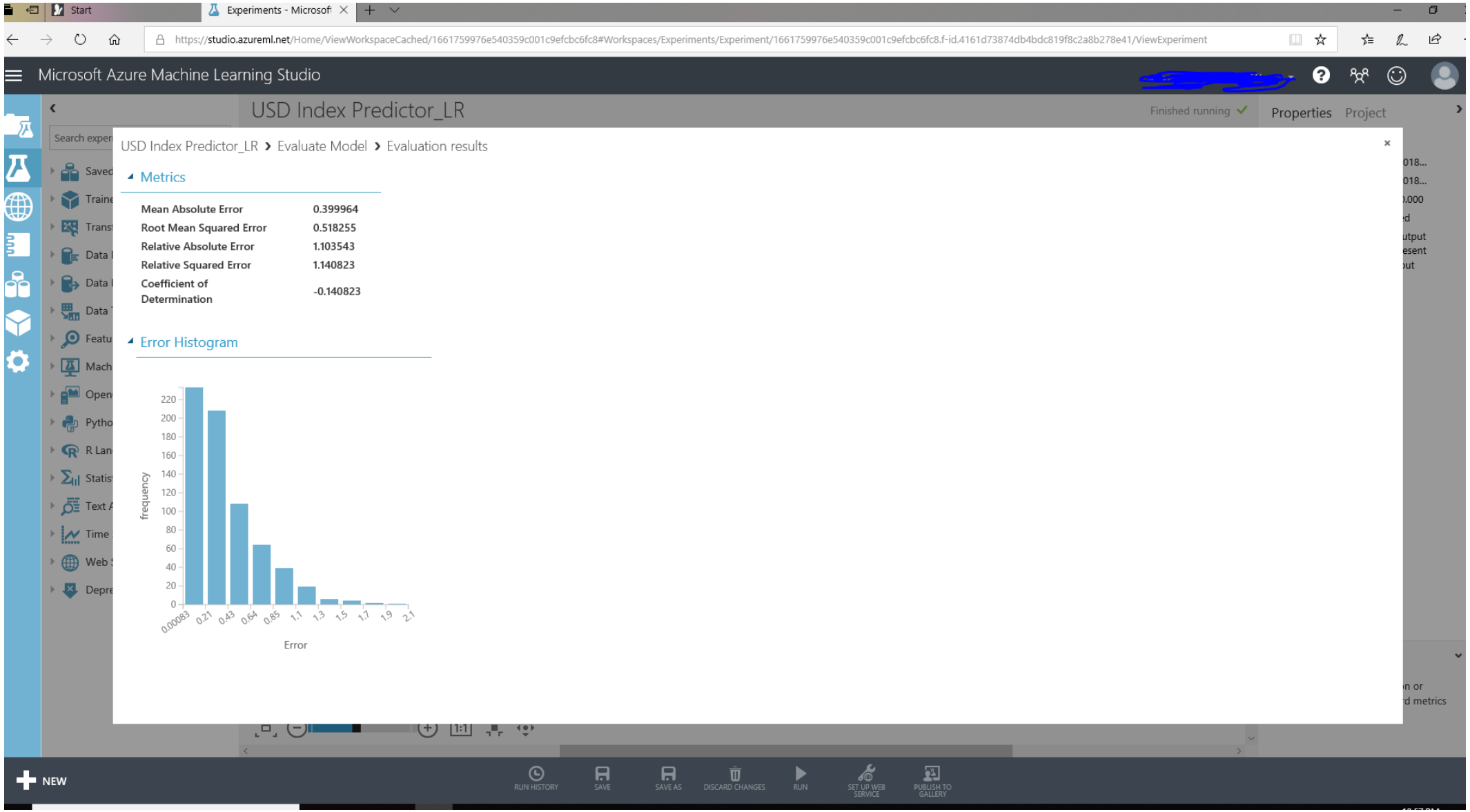
Index	ForwardPerChange	Scored Labels	ScoredDiff
1	0.00	0.01	-0.01
2	0.20	0.30	-0.10
4	-0.42	0.00	-0.42
5	-0.28	-0.13	-0.15
6	0.26	0.10	0.16
7	0.77	0.14	0.64
8	0.02	-0.22	0.24
9	0.13	0.11	0.03
10	0.28	0.11	0.17
11	0.18	-0.05	0.23
12	-0.60	0.04	-0.64
13	0.19	-0.05	0.24

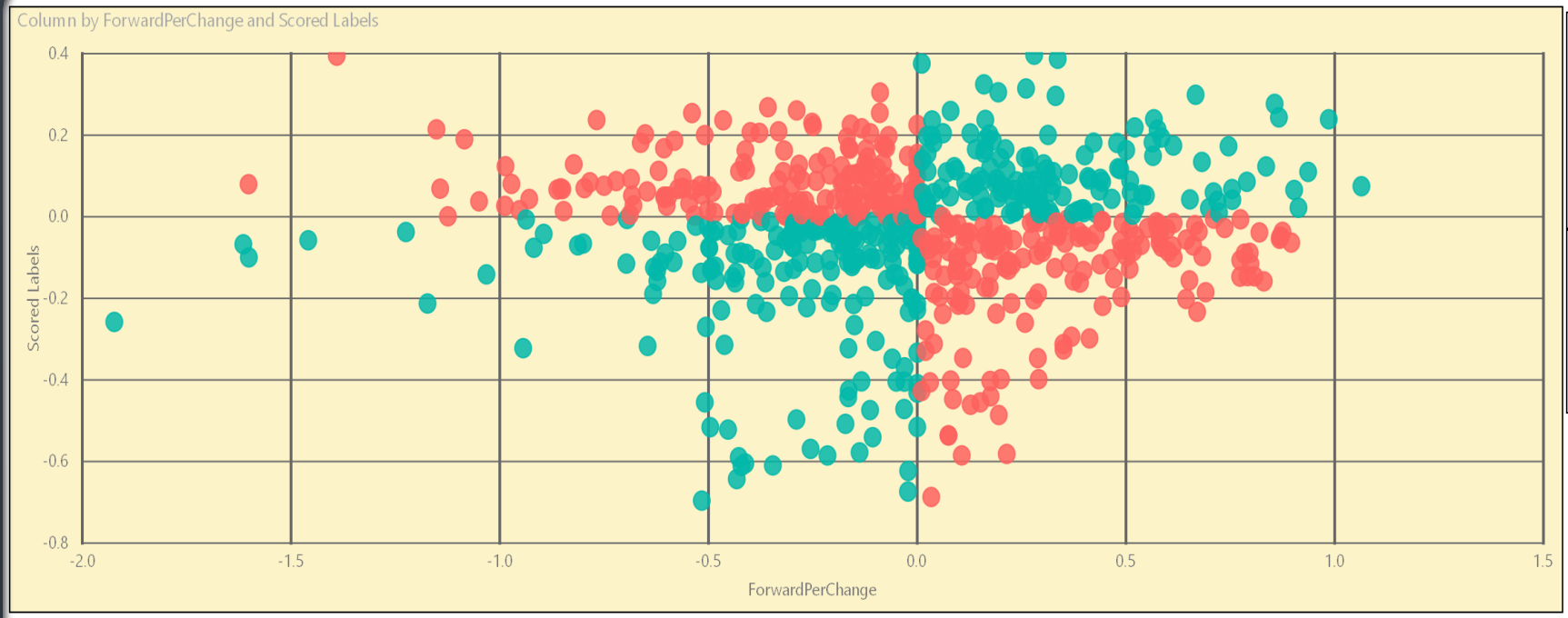
Neural Nets Regression



CLOUD OPERATIONALIZATION

AZURE ML – LINEAR REGRESSION


Start
Experiments - Microsoft
https://studio.azureml.net/Home/ViewWorkspaceCached/1661759976e540359c001c9efcbc6fc8#Workspaces/Experiments/Experiment/1661759976e540359c001c9efcbc6fc8.f-id.4161d73874db4bdc819f8c2a8b278e41/ViewExperiment
Microsoft Azure Machine Learning Studio
USD Index Predictor_LR
Finished running
Properties
Project
USD Index Predictor_LR › Evaluate Model › Evaluation results
Metrics
Mean Absolute Error 0.399964
Root Mean Squared Error 0.518255
Relative Absolute Error 1.103543
Relative Squared Error 1.140823
Coefficient of Determination -0.140823
Error Histogram
frequency
Error
NEW
RUN HISTORY
SAVE
SAVE AS
DISCARD CHANGES
RUN
SET UP WEB SERVICE
PUBLISH TO GALLERY

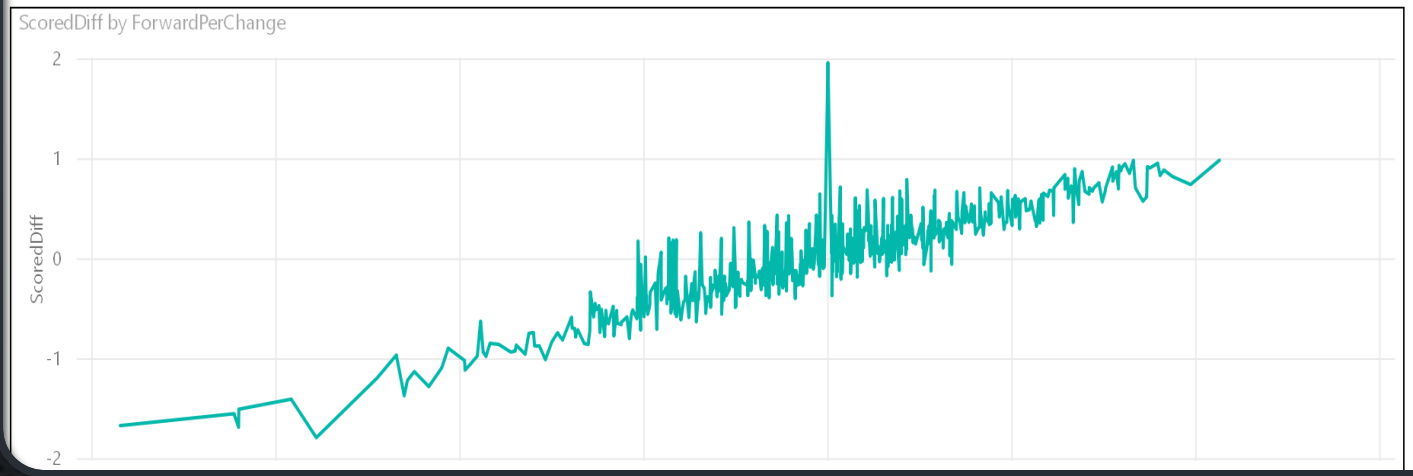


318

Profitable Trades

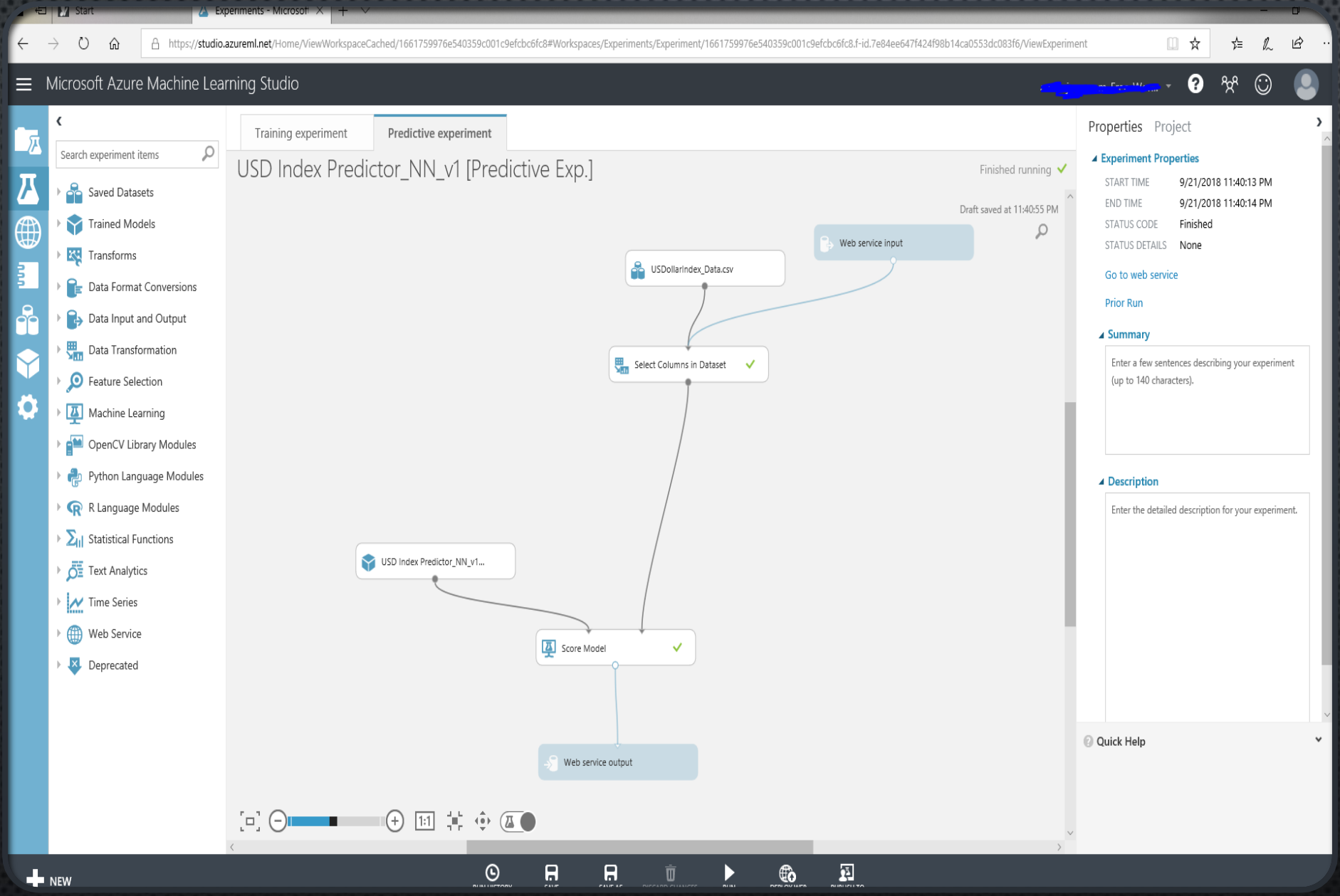
659

Number of Days

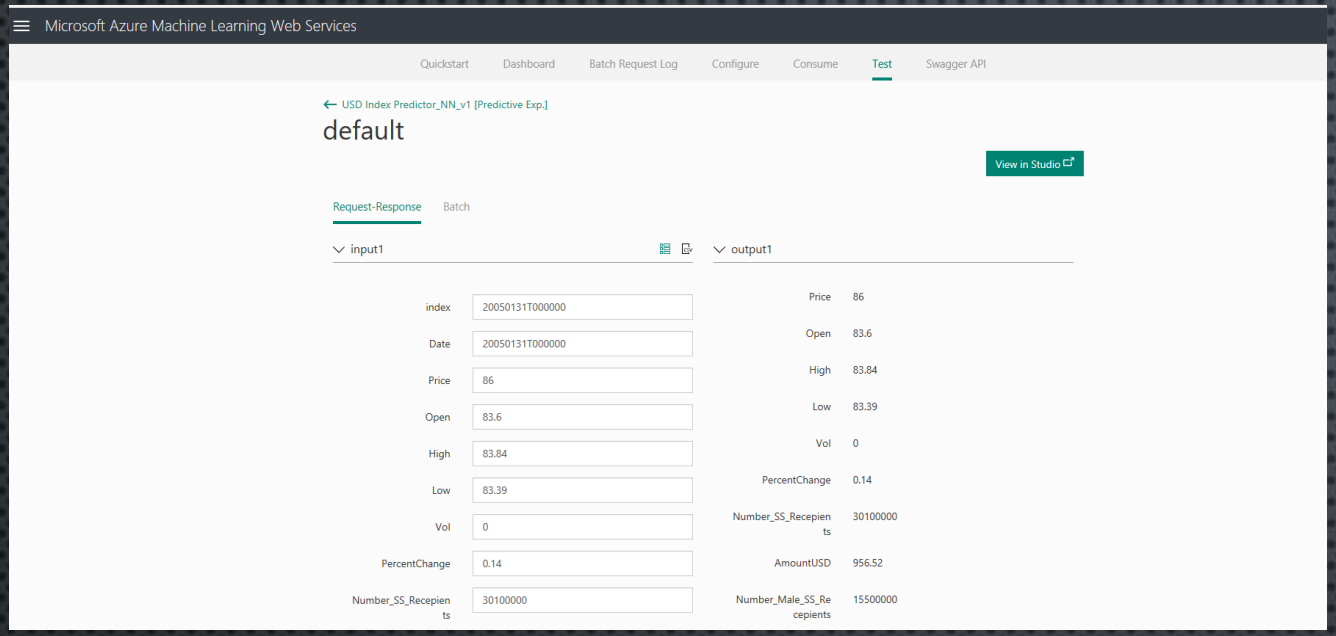


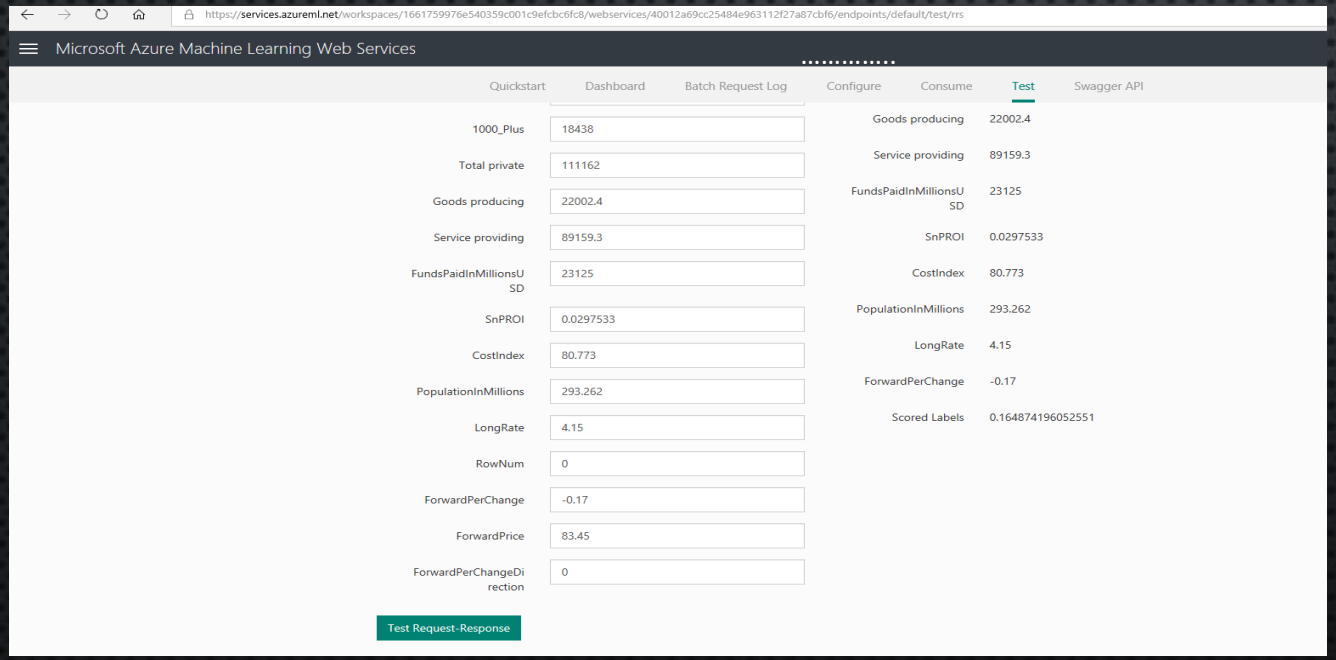
Index	ForwardPerChange	Scored Labels	ScoredDiff
1	0.00	-0.21	0.21
2	0.20	0.06	0.14
4	-0.42	0.01	-0.43
5	-0.28	0.13	-0.41
6	0.26	0.31	-0.05
7	0.77	-0.01	0.78
8	0.02	-0.33	0.35
9	0.13	-0.15	0.28
10	0.28	-0.20	0.48
11	0.18	-0.06	0.24

Linear Regression



Using the trained model to create and deploy an Azure ML web service





Testing the web service manually.

Conclusions

- Both turi – create (GraphLab) and Azure ML gave the best results for boosted Decision Tree Models
- If trades were made based on this system, it would be make a profit slightly more than 50% of the time (Profitable trades are defined as those trades in which the sign +/- of the actual label is the same as the predicted label). This means that the system is only slightly better than a coin toss !

Links

Turi create (GraphLab) Jupyter Notebook and source data

https://github.com/srinikris/SpringBoard_Capstone_2

Azure ML Link

https://gallery.cortanaintelligence.com/Experiment/USD-Index-Predictor-NN-v1-Predictive-Exp