freestyle

June 27, 2024

1 Machine Learning for Finance Freestyle

In this lab you'll be given the opportunity to apply everything you have learned to build a trading strategy for SP500 stocks. First, let's introduce the dataset you'll be using.

1.1 The Data

Use BigQuery's magic function to pull data as follows:

```
Dataset Name: ml4f
Table Name: percent_change_sp500
```

The following query will pull 10 rows of data from the table:

[1]:
```
%%bigquery df
SELECT
    *
FROM
    `cloud-training-prod-bucket.ml4f.percent_change_sp500`
LIMIT
    10
```

```
Query is running:   0%|          |

Downloading:   0%|          |
```

[2]:
```
df.head()
```

[2]:
```
  symbol        Date   Open  Close  tomorrow_close  tomo_close_m_close  \
0      A  2003-12-17  27.51  27.18           27.82                0.64
1      A  2006-09-27  32.95  32.27           32.98                0.71
2      A  2006-04-18  37.08  37.99           38.61                0.62
3      A  2011-10-19  33.31  32.99           33.73                0.74
4      A  2002-01-22  28.40  27.16           28.00                0.84

   close_MIN_prior_5_days  close_MIN_prior_20_days  close_MIN_prior_260_days  \
0                0.976085                 0.976085                  0.421266
1                1.006508                 0.943291                  0.839170
2                0.956304                 0.954725                  0.536983
3                0.984238                 0.891179                  0.891179
```

```
4                1.048601                 1.019882                 0.694404

   close_MAX_prior_5_days  …  close_STDDEV_prior_20_days  \
0                1.010302  …                    0.018430
1                1.025101  …                    0.024930
2                0.973151  …                    0.016506
3                1.023947  …                    0.035434
4                1.121134  …                    0.063416

   close_STDDEV_prior_260_days  \
0                     0.168129
1                     0.089782
2                     0.142754
3                     0.184974
4                     0.365963

                             close_values_prior_260  days_on_market  \
0  [16.95, 16.79, 16.4, 17.16, 17.32, 17.18, 17.2…            1025
1  [33.03, 33.91, 33.65, 34.03, 33.64, 33.68, 33…            1724
2  [21.66, 21.58, 21.37, 21.14, 21.36, 20.81, 20…            1611
3  [33.8, 33.75, 33.68, 34.31, 34.23, 34.48, 34.6…            2999
4  [56.13, 58.25, 55.06, 53.25, 53.0, 55.31, 57.5…             544

   scaled_change  s_p_scaled_change  normalized_change  \
0       0.023547           0.011798           0.011749
1       0.022002           0.001713           0.020289
2       0.016320           0.002027           0.014293
3       0.022431           0.004554           0.017877
4       0.030928           0.007925           0.023003

                    company     industry  direction
0  Agilent Technologies Inc  Health Care         UP
1  Agilent Technologies Inc  Health Care         UP
2  Agilent Technologies Inc  Health Care         UP
3  Agilent Technologies Inc  Health Care         UP
4  Agilent Technologies Inc  Health Care         UP

[5 rows x 26 columns]
```

As you can see, the table contains daily open and close data for SP500 stocks. The table also contains some features that have been generated for you using navigation functions and analytic functions. Let's dig into the schema a bit more.

[10]:
```
%%bigquery
SELECT
    * EXCEPT(is_generated, generation_expression, is_stored, is_updatable)
FROM
```

```
    `cloud-training-prod-bucket.ml4f`.INFORMATION_SCHEMA.COLUMNS
WHERE
    table_name = "percent_change_sp500"
```

[10]:

```
                  table_catalog table_schema           table_name  \
0   cloud-training-prod-bucket         ml4f  percent_change_sp500
1   cloud-training-prod-bucket         ml4f  percent_change_sp500
2   cloud-training-prod-bucket         ml4f  percent_change_sp500
3   cloud-training-prod-bucket         ml4f  percent_change_sp500
4   cloud-training-prod-bucket         ml4f  percent_change_sp500
5   cloud-training-prod-bucket         ml4f  percent_change_sp500
6   cloud-training-prod-bucket         ml4f  percent_change_sp500
7   cloud-training-prod-bucket         ml4f  percent_change_sp500
8   cloud-training-prod-bucket         ml4f  percent_change_sp500
9   cloud-training-prod-bucket         ml4f  percent_change_sp500
10  cloud-training-prod-bucket         ml4f  percent_change_sp500
11  cloud-training-prod-bucket         ml4f  percent_change_sp500
12  cloud-training-prod-bucket         ml4f  percent_change_sp500
13  cloud-training-prod-bucket         ml4f  percent_change_sp500
14  cloud-training-prod-bucket         ml4f  percent_change_sp500
15  cloud-training-prod-bucket         ml4f  percent_change_sp500
16  cloud-training-prod-bucket         ml4f  percent_change_sp500
17  cloud-training-prod-bucket         ml4f  percent_change_sp500
18  cloud-training-prod-bucket         ml4f  percent_change_sp500
19  cloud-training-prod-bucket         ml4f  percent_change_sp500
20  cloud-training-prod-bucket         ml4f  percent_change_sp500
21  cloud-training-prod-bucket         ml4f  percent_change_sp500
22  cloud-training-prod-bucket         ml4f  percent_change_sp500
23  cloud-training-prod-bucket         ml4f  percent_change_sp500
24  cloud-training-prod-bucket         ml4f  percent_change_sp500
25  cloud-training-prod-bucket         ml4f  percent_change_sp500

                  column_name  ordinal_position is_nullable    data_type  \
0                      symbol                 1         YES       STRING
1                        Date                 2         YES         DATE
2                        Open                 3         YES      FLOAT64
3                       Close                 4         YES      FLOAT64
4              tomorrow_close                 5         YES      FLOAT64
5          tomo_close_m_close                 6         YES      FLOAT64
6        close_MIN_prior_5_days               7         YES      FLOAT64
7       close_MIN_prior_20_days               8         YES      FLOAT64
8      close_MIN_prior_260_days               9         YES      FLOAT64
9        close_MAX_prior_5_days              10         YES      FLOAT64
10      close_MAX_prior_20_days              11         YES      FLOAT64
11     close_MAX_prior_260_days              12         YES      FLOAT64
12       close_AVG_prior_5_days              13         YES      FLOAT64
13      close_AVG_prior_20_days              14         YES      FLOAT64
```

```
14      close_AVG_prior_260_days               15          YES          FLOAT64
15     close_STDDEV_prior_5_days               16          YES          FLOAT64
16    close_STDDEV_prior_20_days               17          YES          FLOAT64
17   close_STDDEV_prior_260_days               18          YES          FLOAT64
18        close_values_prior_260               19           NO  ARRAY<FLOAT64>
19                days_on_market               20          YES            INT64
20                 scaled_change               21          YES          FLOAT64
21             s_p_scaled_change               22          YES          FLOAT64
22             normalized_change               23          YES          FLOAT64
23                       company               24          YES           STRING
24                      industry               25          YES           STRING
25                     direction               26          YES           STRING

   is_hidden is_system_defined is_partitioning_column  \
0         NO                NO                     NO
1         NO                NO                     NO
2         NO                NO                     NO
3         NO                NO                     NO
4         NO                NO                     NO
5         NO                NO                     NO
6         NO                NO                     NO
7         NO                NO                     NO
8         NO                NO                     NO
9         NO                NO                     NO
10        NO                NO                     NO
11        NO                NO                     NO
12        NO                NO                     NO
13        NO                NO                     NO
14        NO                NO                     NO
15        NO                NO                     NO
16        NO                NO                     NO
17        NO                NO                     NO
18        NO                NO                     NO
19        NO                NO                     NO
20        NO                NO                     NO
21        NO                NO                     NO
22        NO                NO                     NO
23        NO                NO                     NO
24        NO                NO                     NO
25        NO                NO                     NO

   clustering_ordinal_position
0                         None
1                         None
2                         None
3                         None
4                         None
```

5	None
6	None
7	None
8	None
9	None
10	None
11	None
12	None
13	None
14	None
15	None
16	None
17	None
18	None
19	None
20	None
21	None
22	None
23	None
24	None
25	None

Most of the features, like `open` and `close` are pretty straightforward. The features generated using analytic functions, such as `close_MIN_prior_5_days` are best described using an example. Let's take the 6 most recent rows of data for IBM and reproduce the `close_MIN_prior_5_days` column.

[ ]:
```
%%bigquery
SELECT
    *
FROM
    `cloud-training-prod-bucket.ml4f.percent_change_sp500`
WHERE
    symbol = 'IBM'
ORDER BY
    Date DESC
LIMIT 6
```

For `Date = 2013-02-01` how did we arrive at `close_MIN_prior_5_days = 0.989716`? The minimum close over the past five days was `203.07`. This is normalized by the current day's close of `205.18` to get `close_MIN_prior_5_days = 203.07 / 205.18 = 0.989716`. The other features utilizing analytic functions were generated in a similar way. Here are explanations for some of the other features:

- **scaled_change**: `tomo_close_m_close / close`
- **s_p_scaled_change**: This value is calculated the same way as `scaled_change` but for the S&P 500 index.
- **normalized_change**: `scaled_change - s_p_scaled_change` The normalization using the S&P index fund helps ensure that the future price of a stock is not due to larger market effects.

Normalization helps us isolate the factors contributing to the performance of a stock_market.

- **direction**: This is the target variable we're trying to predict. The logic for this variable is as follows:

```
CASE
    WHEN normalized_change < -0.01 THEN 'DOWN'
    WHEN normalized_change > 0.01 THEN 'UP'
    ELSE 'STAY'
END AS direction
```

1.2 Create classification model for `direction`

In this example, your job is to create a classification model to predict the `direction` of each stock. Be creative! You can do this in any number of ways. For example, you can use BigQuery, Scikit-Learn, or AutoML. Feel free to add additional features, or use time series models.

1.2.1 Establish a Simple Benchmark

One way to assess the performance of a model is to compare it to a simple benchmark. We can do this by seeing what kind of accuracy we would get using the naive strategy of just predicting the majority class. Across the entire dataset, the majority class is 'STAY'. Using the following query we can see how this naive strategy would perform.

[9]:
```
%%bigquery
WITH subset as (
    SELECT
        Direction
    FROM
        `cloud-training-prod-bucket.ml4f.percent_change_sp500`
    WHERE
        tomorrow_close IS NOT NULL
)
SELECT
    Direction,
    100.0 * COUNT(*) / (SELECT COUNT(*) FROM subset) as percentage
FROM
    subset
GROUP BY
    Direction
```

[9]:

	Direction	percentage
0	STAY	53.766049
1	UP	23.240681
2	DOWN	22.993271

So, the naive strategy of just guessing the majority class would have accuracy of around 54% across the entire dataset. See if you can improve on this.