Stock Market Advanced Indicator Engineering And Learning

FINISHED

We're going to perform cleansing, profiling and exploratory data analysis on the real stock market data. We will be using the Huge Stock Market Data from NYSE, NASDAQ, and NYSE MKT (https://www.kaggle.com/datasets/borismarjanovic/price-volume-data-for-all-us-stocks-etfs?datasetId=4538&sortBy=voteCount).

Took 3 sec. Last updated by hd2225_nyu_edu at December 17 2022, 7:10:37 PM.

Reading Preprocessed Data From HDFS

FINISHED

In this step, we will read all of the preprocessedd data stored as parquet from the HDFS. Since the data is parquet, we will try to ensure that we infer the schema of the source data.

Took 0 sec. Last updated by hd2225_nyu_edu at December 17 2022, 7:10:43 PM.

Reading input data as DataFrame

SPARK JOB (http://nyu-dataproc-sw-zwj7.c.hpc-dataproc-19b8.internal:45961/jobs/job?id=62) FINISHED

```
val input_path = "loudacre/Project/hist_stock_datapre.parquet"
val input_data = spark.read.format("parquet").load(input_path)
```

Took 0 sec. Last updated by anonymous at December 11 2022, 3:08:49 PM.

```
z.show(input_data)
```

SPARK JOB (http://nyu-dataproc-sw-zwj7.c.hpc-dataproc-19b8.internal:45961/jobs/job?id=63) FINISHED

settings

Date	Open	High	Low	Close	Volume
1962-01-02	0.6277	0.6362	0.6201	0.6201	2575579
1962-01-03	0.6201	0.6201	0.6122	0.6201	1764749
1962-01-04	0.6201	0.6201	0.6037	0.6122	2194010
1962-01-05	0.6122	0.6122	0.5798	0.5957	3255244
1962-01-08	0.5957	0.5957	0.5716	0.5957	3696430
1962-01-09	0.5957	0.6037	0.5878	0.5957	2778285
1962-01-10	0.5957	0.6037	0.5957	0.5957	2337096
1962-01-11	0.5957	0.5957	0.5878	0.5957	1943605

Took 1 sec. Last updated by anonymous at December 11 2022, 3:08:50 PM. (outdated)

Finding Number of Columns in Data

FINISHED

```
input_data.columns.length
```

Took 0 sec. Last updated by anonymous at December 11 2022, 3:08:50 PM.

Exploring Data Schema

FINISHED

```
input_data.printSchema
```

Took 1 sec. Last updated by anonymous at December 11 2022, 3:08:51 PM.

Machine Learning

FINISHED

In this step, we will perform basic machine engineering to explore the potential models for label prediction.

Took 0 sec. Last updated by hd2225_nyu_edu at December 17 2022, 7:10:48 PM.

```
def charr = udf((st:String)=>{
    st.substring(0,1)
})
```

FINISHED

Took 0 sec. Last updated by anonymous at December 11 2022, 3:08:51 PM.

```
import org.apache.spark.sql.functions._
import org.apache.spark.sql.expressions.Window
var df = input_data
df = df.withColumn("rank", percent_rank().over(Window.partitionBy().orderBy("date")))
df = df.withColumn("label_new", charr(col("Label")))
```

FINISHED

Took 0 sec. Last updated by anonymous at December 11 2022, 3:08:51 PM.

```
val train_data = df.where("rank <= .8").drop("rank")
val test_data = df.where("rank > .8").drop("rank")
```

FINISHED

Took 1 sec. Last updated by anonymous at December 11 2022, 3:08:52 PM.

```
print("train size = ", train_data.count)
print("test size = ", test_data.count)
```

SPARK JOB FINISHED

Took 1 min 18 sec. Last updated by anonymous at December 11 2022, 3:10:10 PM.

```
z.show(train_data)
```

SPARK JOB FINISHED

settings

Date	Open	High	Low	Close	Volume
1962-01-02	0.6277	0.6362	0.6201	0.6201	2575579
1962-01-02	6.413	6.413	6.3378	6.3378	467056
1962-01-03	0.6201	0.6201	0.6122	0.6201	1764749
1962-01-03	6.3378	6.3963	6.3378	6.3963	350294
1962-01-04	0.6201	0.6201	0.6037	0.6122	2194010
1962-01-04	6.3963	6.3963	6.3295	6.3295	314365
1962-01-05	0.6122	0.6122	0.5798	0.5957	3255244
1962-01-05	6.3211	6.3211	6.1958	6.2041	440112

Took 57 sec. Last updated by anonymous at December 11 2022, 3:11:07 PM.

```
import org.apache.spark.ml.feature.VectorAssembler

val vecAssembler = new VectorAssembler()
  .setInputCols(Array("Open", "Volume"))
  .setOutputCol("features")

val vecTrainDF = vecAssembler.transform(train_data)
z.show(vecTrainDF.select("features", "Close"))
```

SPARK JOB FINISHED

settings

features	Close
[0.6277,2575579.0]	0.6201
[6.413,467056.0]	6.3378
[0.6201,1764749.0]	0.6201
[6.3378,350294.0]	6.3963
[0.6201,2194010.0]	0.6122
[6.3963,314365.0]	6.3295
[0.6122,3255244.0]	0.5957

[6.3211,440112.0]	6.2041

- Took 42 sec. Last updated by anonymous at December 11 2022, 4:10:31 PM.

```
import org.apache.spark.ml.regression.LinearRegression

val lr = new LinearRegression()
  .setFeaturesCol("features")
  .setLabelCol("Close")
```

FINISHED

Took 40 sec. Last updated by anonymous at December 11 2022, 4:10:31 PM.

```
val lrModel = lr.fit(vecTrainDF)
val m = lrModel.coefficients(0)
val b = lrModel.intercept
```

SPARK JOB FINISHED

Took 2 min 0 sec. Last updated by anonymous at December 11 2022, 4:12:31 PM.

```
import org.apache.spark.ml.Pipeline
val pipeline = new Pipeline().setStages(Array(vecAssembler, lr))
val pipelineModel = pipeline.fit(train_data)
```

SPARK JOB FINISHED

Took 4 min 14 sec. Last updated by anonymous at December 11 2022, 4:14:46 PM.

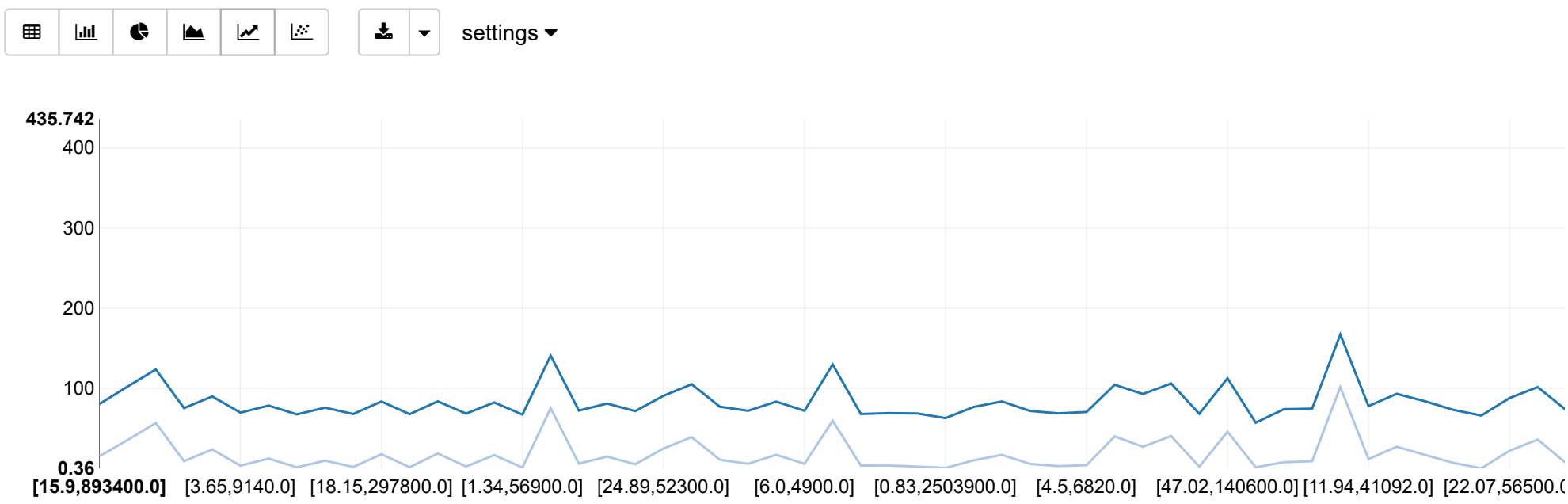
```
val predDF = pipelineModel.transform(test_data)
predDF.select("features", "Close", "prediction").show(10)
```

SPARK JOB FINISHED

Took 2 min 58 sec. Last updated by anonymous at December 11 2022, 4:15:30 PM.

```
z.show(predDF.limit(100).toDF())
```

SPARK JOB FINISHED



Took 1 min 39 sec. Last updated by anonymous at December 11 2022, 4:16:25 PM. (outdated)

```
import org.apache.spark.ml.evaluation.RegressionEvaluator
val regressionEvaluator = new RegressionEvaluator()
 .setPredictionCol("prediction")
 .setLabelCol("Close")
 .setMetricName("rmse")
val rmse = regressionEvaluator.evaluate(predDF)
println(f"RMSE is $rmse%.1f")

val r2 = regressionEvaluator.setMetricName("r2").evaluate(predDF)
println(s"R2 is $r2")
```

SPARK JOB FINISHED

Took 2 min 6 sec. Last updated by anonymous at December 11 2022, 3:20:12 PM.

```
import org.apache.spark.ml.feature.{OneHotEncoder, StringIndexer}
val categoricalCols = train_data.dtypes.filter{ case (field, dataType) =>
 field == "label_new"}.map(_._1)
val indexOutputCols = categoricalCols.map(_ + "_index")
val oheOutputCols = categoricalCols.map(_ + "_OHE")

val stringIndexer = new StringIndexer()
 .setInputCols(categoricalCols)
 .setOutputCols(indexOutputCols)
 .setHandleInvalid("skip")

val oheEncoder = new OneHotEncoder()
 .setInputCols(indexOutputCols)
 .setOutputCols(oheOutputCols)
```

FINISHED

```
val numericCols = train_data.dtypes.filter{ case (field, dataType) =>
 field == "Open" || field == "Volume"}.map(_._1)

val assemblerInputs = oheOutputCols ++ numericCols
val vecAssembler = new VectorAssembler()
 .setInputCols(assemblerInputs)
 .setOutputCol("features")
```

Took 0 sec. Last updated by anonymous at December 11 2022, 3:20:12 PM.

```
val lr = new LinearRegression()
 .setLabelCol("Close")
 .setFeaturesCol("features")
val pipeline = new Pipeline()
 .setStages(Array(stringIndexer, oheEncoder, vecAssembler, lr))
```

FINISHED

Took 1 sec. Last updated by anonymous at December 11 2022, 3:20:13 PM.

```
val pipelineModel = pipeline.fit(train_data)

val predDF = pipelineModel.transform(test_data)
predDF.select("features", "Close", "prediction").show(5)

val rmse = regressionEvaluator.evaluate(predDF)
println(f"RMSE is $rmse%.1f")

val r2 = regressionEvaluator.setMetricName("r2").evaluate(predDF)
println(s"R2 is $r2")
```

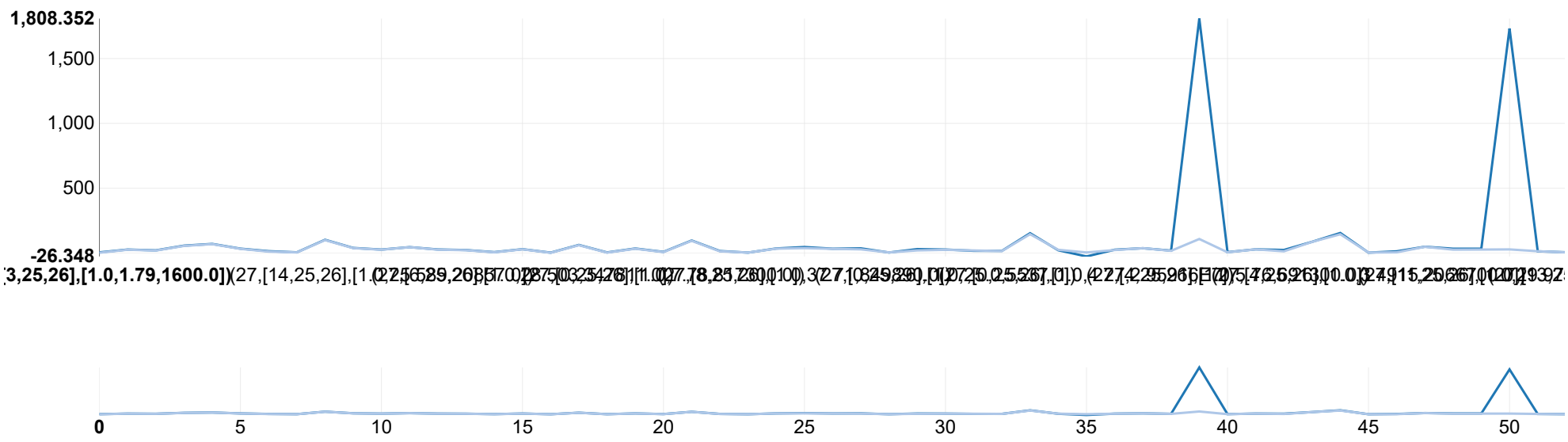
SPARK JOB FINISHED

Took 8 min 36 sec. Last updated by anonymous at December 11 2022, 3:28:49 PM.

```
z.show(predDF.limit(100).toDF())
```

SPARK JOB FINISHED

settings



Took 1 min 4 sec. Last updated by anonymous at December 11 2022, 4:04:00 PM. (outdated)

Advanced Technical Indicator Engineering

FINISHED

In this step, we will perform advanced Technical Indicator engineering to explore the potential Technical Indicator to show on dashboard.

Took 0 sec. Last updated by hd2225_nyu_edu at December 17 2022, 7:10:56 PM.

```
val exp_data = input_data.select("Date", "Close")
val fil = exp_data.where($"Label" === "bgr")
z.show(fil)
```

SPARK JOB FINISHED

settings

Date	Close
2005-02-25	12.115
2005-02-28	12.158
2005-03-01	12.141
2005-03-02	12.17
2005-03-03	12.246
2005-03-04	12.294

Took 2 sec. Last updated by anonymous at December 11 2022, 3:28:51 PM.

Took 0 sec. Last updated by anonymous at December 11 2022, 3:28:51 PM. FINISHED

```
import org.apache.spark.sql.DataFrame
import org.apache.spark.sql.functions._
import org.apache.spark.sql.types._
import scala.collection.mutable.ArrayBuffer
```

FINISHED

Took 0 sec. Last updated by anonymous at December 11 2022, 3:28:51 PM.

```
val w = org.apache.spark.sql.expressions.Window.orderBy("D
import org.apache.spark.sql.functions.lag
val xy = input_data.where($"Label" === "odp").select("Date", "Close")
xy.show()
```

SPARK JOB (http://nyu-dataproc-sw-zwj7.c.hpc-dataproc-19b8.internal:45961/jobs/job?id=103) FINISHED

Took 1 sec. Last updated by anonymous at December 11 2022, 3:28:52 PM.

```
val mm = xy.withColumn("Close_new", lag("Close", 1, 0).over(w))
mm.show()
```

SPARK JOB FINISHED

Took 3 sec. Last updated by anonymous at December 11 2022, 3:28:55 PM.

```
val momentum = mm.withColumn("momentum_1d",
    (col("Close") - col("Close_new"))
    )
momentum.show()
```

SPARK JOB FINISHED

Took 3 sec. Last updated by anonymous at December 11 2022, 3:28:58 PM.

```
import scala.collection.mutable.WrappedArray
val rsi = udf((values: WrappedArray[Double])=> {
    val up_temp = values.filter(_ > 0)
    val down_temp = values.filter(_ < 0)
    val up: Double = up_temp.sum/up_temp.length
    var down: Double = -1* (down_temp.sum/down_temp.length)
    if(down_temp.length == 0){
       down = 0.0
    }
    // print("up = ", up)
    // print("down = ", down)
    100 * up/(up+down)
})
```

FINISHED

Took 0 sec. Last updated by anonymous at December 11 2022, 3:28:58 PM.

```
import org.apache.spark.sql.expressions.WindowSpec
val df_rsi_temp = momentum.withColumn("RSI_IM", ((collect_list("momentum_1d"))
            .over(org.apache.spark.sql.expressions.Window.orderBy(col("Date")).rowsBetween(-14,0))))
z.show(df_rsi_temp)
```

SPARK JOB FINISHED

settings

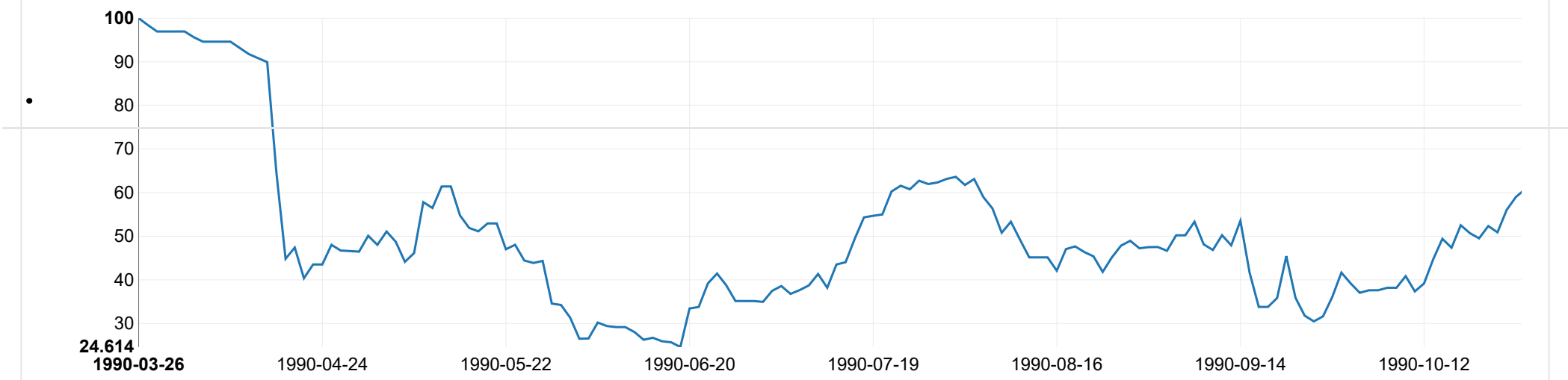
Date	Close	Close_new
1990-03-26	2.4435	0.0
1990-03-27	2.4046	2.4435
1990-03-28	2.4435	2.4046
1990-03-29	2.4046	2.4435
1990-03-30	2.4046	2.4046
1990-04-02	2.3659	2.4046

Took 3 sec. Last updated by anonymous at December 11 2022, 3:29:01 PM. (outdated)

```
val df_rsi = df_rsi_temp.withColumn("RSI",
               rsi(col("RSI_IM")))
z.show(df_rsi)
```

SPARK JOB FINISHED

settings



Took 3 sec. Last updated by anonymous at December 11 2022, 3:29:04 PM. (outdated)

```
df_rsi.limit(100).toDF().select("Close", "Close_new", "momentum_1d", "RSI").show()
```

SPARK JOB FINISHED

Took 2 sec. Last updated by anonymous at December 11 2022, 10:09:13 PM.

```
def bbands()={
    val df_bb_temp = df_rsi.withColumn("BB_IM", ((collect_list("momentum_1d"))
            .over(org.apache.spark.sql.expressions.Window.orderBy(col("Date")).rowsBetween(-20,0))))
    df_bb_temp
}
```

FINISHED

Took 1 sec. Last updated by anonymous at December 11 2022, 3:29:05 PM.

```
val bb_df = bbands()
z.show(bb_df)
```

SPARK JOB FINISHED

settings

Date	Close	Close_new	momentum_1d
1990-03-26	2.4435	0.0	2.4435
1990-03-27	2.4046	2.4435	-0.038899999999999935
1990-03-28	2.4435	2.4046	0.038899999999999935
1990-03-29	2.4046	2.4435	-0.038899999999999935

Took 3 sec. Last updated by anonymous at December 11 2022, 3:29:08 PM.

```
import scala.collection.mutable.WrappedArray
val bband_avg = udf((values: WrappedArray[Double])=> {
    values.sum/values.length
})
val bband_std = udf((a: WrappedArray[Double]) => {
    val mean = a.sum / a.length
    val squareErrors = a.map(x => x - mean).map(x => x * x)
    math.sqrt(squareErrors.sum / a.length)
})
```

FINISHED

Took 0 sec. Last updated by anonymous at December 11 2022, 3:29:08 PM.

```
val df_bb_avg = bb_df.withColumn("BB_AVG",
                bband_avg(col("BB_IM")))
val df_bb_std = df_bb_avg.withColumn("BB_STD",
                bband_std(col("BB_IM")))
z.show(df_bb_std)
```

SPARK JOB FINISHED

settings

Date	Close	Close_new	momentum_1d	RSI_IM
1990-03-26	2.4435	0.0	2.4435	WrappedArray(2.4435)
1990-03-27	2.4046	2.4435	-0.038899999999999935	WrappedArray(2.4435,

				-0.038899999999999935)
1990-03-28	2.4435	2.4046	0.038899999999999935	WrappedArray(2.4435, -0.038899999999999935, 0.038899999999999935)
1990-03-29	2.4046	2.4435	-0.038899999999999935	WrappedArray(2.4435, -0.038899999999999935, 0.038899999999999935, -0.038899999999999935)

Took 3 sec. Last updated by anonymous at December 11 2022, 3:29:11 PM.

```
val df_bb_lower = df_bb_std.withColumn("BB_Lower_Band",
                (col("BB_AVG") - (col("BB_STD")*20))).drop("RSI_IM").drop("BB_IM")
z.show(df_bb_lower)
```

SPARK JOB FINISHED

settings

Date	Close	Close_new	momentum_1d	RSI
1990-03-26	2.4435	0.0	2.4435	100.0
1990-03-27	2.4046	2.4435	-0.038899999999999935	98.432968
1990-03-28	2.4435	2.4046	0.038899999999999935	96.961174
1990-03-29	2.4046	2.4435	-0.038899999999999935	96.961174
1990-03-30	2.4046	2.4046	0.0	96.961174
1990-04-02	2.3659	2.4046	-0.03869999999999996	96.966224
1990-04-03	2.4435	2.3659	0.07759999999999989	95.647300
1990-04-04	2.3659	2.4435	-0.07759999999999989	94.619443

Took 3 sec. Last updated by anonymous at December 11 2022, 3:29:14 PM. (outdated)

```
val df_bb_upper = df_bb_lower.withColumn("BB_Upper_Band",
                (col("BB_AVG") + (col("BB_STD")*20))).drop("RSI_IM").drop("BB_IM")
z.show(df_bb_upper)
```

SPARK JOB FINISHED

settings

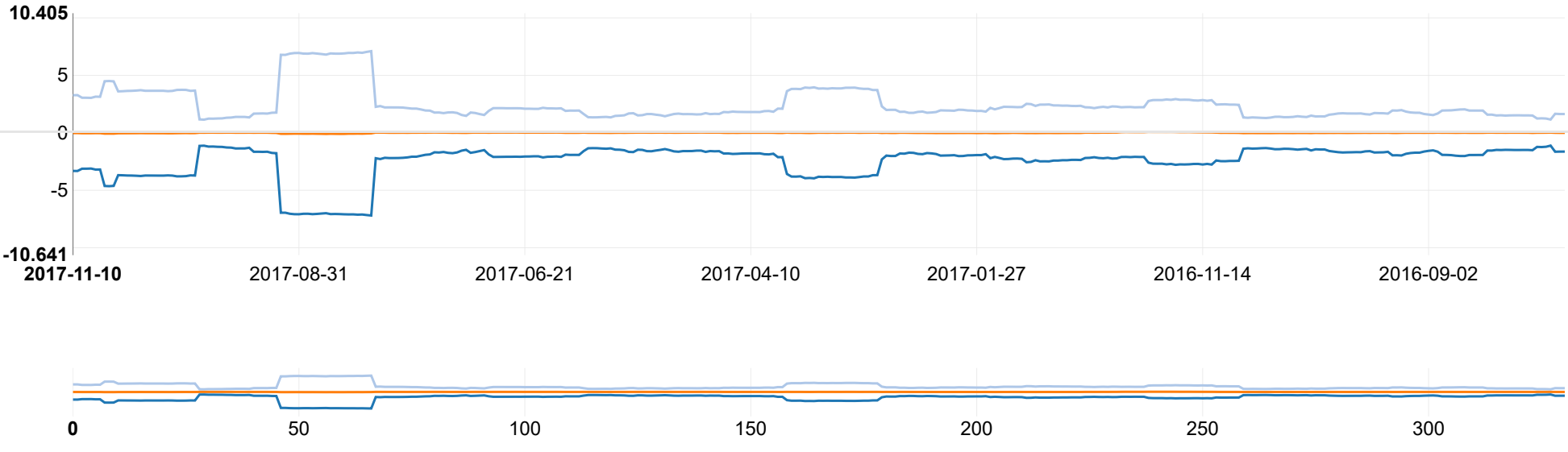
Date	Close	Close_new	momentum_1d	RSI
1990-03-26	2.4435	0.0	2.4435	100.0
1990-03-27	2.4046	2.4435	-0.038899999999999935	98.43296809539156
1990-03-28	2.4435	2.4046	0.038899999999999935	96.96117490821031
1990-03-29	2.4046	2.4435	-0.038899999999999935	96.96117490821031
1990-03-30	2.4046	2.4046	0.0	96.96117490821031
1990-04-02	2.3659	2.4046	-0.03869999999999996	96.96622483789486
1990-04-03	2.4435	2.3659	0.07759999999999989	95.64730057911451
1990-04-04	2.3659	2.4435	-0.07759999999999989	94.61944318675329

Took 3 sec. Last updated by anonymous at December 11 2022, 3:29:18 PM. (outdated)

```
val df_bb = df_bb_upper.withColumn("BB_Middle_Band",
                (col("BB_AVG"))).orderBy(desc("Date"))
z.show(df_bb)
```

SPARK JOB FINISHED

settings



Took 3 sec. Last updated by anonymous at December 11 2022, 3:29:21 PM. (outdated)