5. df\_SPY = df\_SPY.withColumn("Return", (df\_SPY["Adj Close"] - df\_SPY["Open"]) / df\_SPY["Open"] \* 100);

7. from pyspark.sql.window import Window

8. from pyspark.sql import functions as F

9. window21 = Window.orderBy("Date").rowsBetween(-21,0);

10. df\_SPY = df\_SPY.withColumn("21D\_CumReturn", F.sum(F.col("Return")/100).over(window21));

11. window252 = Window.orderBy("Date").rowsBetween(-252,0);

12. df\_SPY = df\_SPY.withColumn("252D\_CumReturn", F.sum(F.col("Return")/100).over(window252));

13. df\_SPY = df\_SPY.withColumn("21DAverage", F.mean(F.col("Return")).over(window21));

14. df\_SPY = df\_SPY.withColumn("21DAverageMax", F.max(F.col("21DAverage")).over(window21));

15. df\_SPY = df\_SPY.withColumn("21DLiquidity",F.sum(F.col("volume")).over(window21));

16. df\_SPY = df\_SPY.withColumn("21DAvgLiquidity",F.mean(F.col("21DLiquidity")).over(window21));

17. df\_SPY = df\_SPY.withColumn("252DLiquidity",F.sum(F.col("volume")).over(window252));

18. df\_SPY = df\_SPY.withColumn("252DAvgLiquidity",F.mean(F.col("252DLiquidity")).over(window252));

21. df\_SPY = df\_SPY.withColumn("ABTO\_Reverse", F.col('252DAvgLiquidity')/F.col('21DAvgLiquidity'));

28. for i in df\_SPY.columns:print( "Correlation to Return for ", i, df\_SPY.stat.corr('Return',i))

29. from pyspark.ml.feature import VectorAssembler

30. from pyspark.ml.regression import LinearRegression

31. vectorAssembler = VectorAssembler(inputCols = ['21D\_CumReturn\_STR', '252D\_CumReturn\_MOM', '21DAverageMAX\_MAX', '252DAvgLiquidityto21DAvgLiquidity\_ABTO\_Reverse'],outputCol = 'features')

32. m\_SPY = vectorAssembler.transform(df\_SPY)

33. from pyspark.sql.functions import col

34. m\_SPY.select(col("features"),col("Return")).show()

35. splits = m\_SPY.randomSplit([0.7, 0.3])

36. training = splits[0]

37. testing = splits[1]

38. from pyspark.ml.regression import LinearRegression

39. lr = LinearRegression(featuresCol = 'features', labelCol='Return', maxIter=10, regParam=0.3, elasticNetParam=0.8)

40. lr\_model = lr.fit(training)

41. print("Coefficients: " + str(lr\_model.coefficients))

42. print("Intercept: " + str(lr\_model.intercept))

43. trainingSummary = lr\_model.summary

44. print("r2: %f" % trainingSummary.r2)

45. lr\_predictions = lr\_model.transform(testing)

46.lr\_predictions.select(col("prediction"),col("Return"),col("features"))

47. from pyspark.ml.evaluation import RegressionEvaluator

48. lr\_evaluator = RegressionEvaluator(predictionCol="prediction", \

... labelCol="Return",metricName="r2")

49. print("R2 on the test data = %g" % lr\_evaluator.evaluate(lr\_predictions))