

# 주식 트레이딩 강화학습

인공지능금융투자  
2022-2학기 기말 프로젝트

20191026 윤서영  
20201050 이연주 | 팀장  
20201064 허지원

# CONTENTS

---

## 01

최초 계획 및 최종결과

처음 계획과  
최종 진행 내용

## 02

개발한 피쳐

수집기  
분석기  
판단로직

## 03

실전 자동 매매

매매 결과

## 04

협업 내용

역할 분담  
github 리포지터리 소개  
개발 경험

01

계획 및 결과

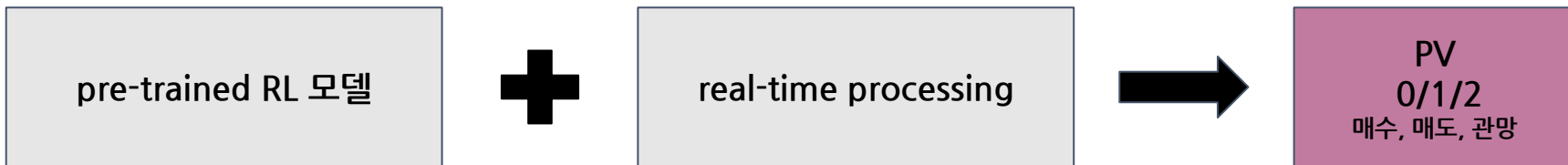
# 01. 최초 계획

## 목표

- 자동화 프로그램 구축

실시간 tr 처리 - 데이터베이스 저장

- 실시간 예측 모델 생성



# 01. 최종 결과

## - 실시간 프로그램 구축

- 1) 실시간 tr 처리 - 데이터베이스 저장
- 2) 실시간 구독 및 매도 매수 프로그램 구축

## - 강화학습 모델

- 1) Policy Gradient Learner + LSTM

▼ models
> 20221213003241_pg_lstm
> 20221213005200_pg_lstm
> 20221213010018_pg_lstm
> 20221213010808_pg_lstm
> 20221213010834_pg_lstm
> 20221213014235_pg_lstm

▼ output
> 1봉결과
> 60봉결과
▼ 069500
> predict_20221213005200_pg_lstm
> test_20221213005200_pg_lstm
> train_20221213005200_pg_lstm

# 02

개발 피쳐

# 00. 프로젝트 파일 소개

```
▼ PROJECT
> __pycache__
> analyzer
> dynamic_TR
> rltrader
> static_TR
> tool
≡ command_venv_32BIT.txt
≡ command_venv_64BIT.txt
≡ kiwoom_tick.db
≡ kiwoom.db
🔗 model_predict.py
🔗 model_test.py
🔗 model_train.py
🔗 retrieve_dynamic_TR.py
🔗 retrieve_static_TR.py
```

```
▼ analyzer
🔗 fundamentalAnalysis.ipynb
🔗 fundamentalAnalysis.py
🔗 technicalAnalysis.ipynb
🔗 technicalAnalysis.py
▼ dynamic_TR
🔗 const.py
🔗 pytrader.py
🔗 real_kiwoom.py
🔗 realTime_TR.py
▼ rltrader
> networks
🔗 agent.py
🔗 data_manager.py
🔗 environment.py
🔗 learners.py
🔗 settings.py
🔗 utils.py
🔗 visualizer.py
▼ static_TR
🔗 01_retrieveTR_day.py
🔗 02_retrieveTR_min.py
🔗 03_retrieveTR_min_60.py
🔗 retrieveTR.py
🔗 static_kiwoom.py
▼ tool
🔗 manageDB.py
```

## 계층 구조 이용

### static\_TR

- opt10080 분봉 TR -> SQL에 저장  
\*정기적으로 실행하면 됨
- model\_train/test의 데이터

### dynamic\_TR

- 실시간 TR 구독
- model\_predict의 데이터

### rltrader

- LSTM model
- Agent
- Environment

## main 실행 파일

mode\_train

model\_test

model\_predict

retrieve\_static\_TR

## python 파일 (클래스, 함수) 연결

```
from static TR import static_kiwoom
```

# 01. 수집기

## SQL 구조, SQL 초기화

- 테이블 (4)
  - 069500
  - 114800
  - 226490
  - company\_info

opt10080

- 분봉 TR, SQL에 저장될 데이터
- 최대 약 1년치 데이터를 받을 수 있음

SQL 주요 action

- 해당 종목코드 테이블이 없을 시, 빈 테이블 생성
- 해당 종목코드 테이블이 있다면, last\_update 이후의 TR정보를 저장

테이블 (4)

- company\_info: 종목코드 테이블에 대한 정보

SQL 업데이트 시 last\_update 날짜 저장, 중복 허용x (\*st\_code (PK))

- [종목코드] (069500, 114800, 226490): TR 데이터 (ohlcv)

중복 허용 x (lower\_bound=last\_update, \*date (PK))

실행 기준 전날까지의 데이터를 저장 (upper\_bound = yesterday)

테이블 (4)

이름	타입	스키마
069500		CREATE TABLE "069500" ( "date" TEXT, "open" INTEGER, "high" INTEGER, "low" INTEGER, "close" INTEGER, "volume" INTEGER )
date	TEXT	"date" TEXT
open	INTEGER	"open" INTEGER
high	INTEGER	"high" INTEGER
low	INTEGER	"low" INTEGER
close	INTEGER	"close" INTEGER
volume	INTEGER	"volume" INTEGER
114800		CREATE TABLE "114800" ( "date" TEXT, "open" INTEGER, "high" INTEGER, "low" INTEGER, "close" INTEGER, "volume" INTEGER )
date	TEXT	"date" TEXT
open	INTEGER	"open" INTEGER
high	INTEGER	"high" INTEGER
low	INTEGER	"low" INTEGER
close	INTEGER	"close" INTEGER
volume	INTEGER	"volume" INTEGER
226490		CREATE TABLE "226490" ( "date" TEXT, "open" INTEGER, "high" INTEGER, "low" INTEGER, "close" INTEGER, "volume" INTEGER )
date	TEXT	"date" TEXT
open	INTEGER	"open" INTEGER
high	INTEGER	"high" INTEGER
low	INTEGER	"low" INTEGER
close	INTEGER	"close" INTEGER
volume	INTEGER	"volume" INTEGER
company_info		CREATE TABLE company_info ( st_code VARCHAR(20) not NULL, nm_code VARCHAR(40), last_update TEXT not NULL, PRIMARY KEY (st_code) )
st_code	VARCHAR(20)	"st_code" VARCHAR(20) NOT NULL
nm_code	VARCHAR(40)	"nm_code" VARCHAR(40)
last_update	TEXT	"last_update" TEXT NOT NULL

테이블(T): company\_info

st_code	nm_code	last_update
필터	필터	필터

테이블(T): 069500

date	open	high	low	close	volume
필터	필터	필터	필터	필터	필터

테이블(T): 114800

date	open	high	low	close	volume
필터	필터	필터	필터	필터	필터

테이블(T): 226490

date	open	high	low	close	volume
필터	필터	필터	필터	필터	필터



# 01. 수집기

## SQL class 구현 (tool/manageDB.py)

### setQuery

- sql 문법 모음
  - \* 숫자(int)를 테이블의 이름을 넣기 위해서 []로 감싸줘야함
  - \* 중복에 벗어나기 위한 안정 장치: Primary Key - 무결성 제약조건 에러가 발생

### sqlWriter

#### [1. ohcv 데이터 저장 준비]

- \_\_init\_\_(codes): 초기화 함수, DB 연결 및 '없는' 테이블 생성
- get\_last\_update(): DB에 저장된 last\_update 날짜를 읽기
  - \* Opt1. 해당 종목코드를 처음 접속: return '0' / Opt2. 해당 종목코드에 접속한 적 있음: return 'last\_update'

#### [2. ohcv 데이터 저장 & company\_info 업데이트]

- update\_comany\_info(info): 종목코드 저장 완료 후 실행 (종목코드 행이 있다면, update, 없다면, insert)
- save\_query(st\_code, nm\_code, df): df를 st\_code 테이블에 저장, df.to\_sql('append') (종목코드 테이블이 있다면, update, 없다면, insert)

### sqlReader

- read\_query(st\_code): 해당 [종목코드]'s ohlc 데이터 반환 (type=dataframe)

```
table_create_query = """
CREATE TABLE IF NOT EXISTS [{TABLE_NAME}] (
    date TEXT not NULL,
    open INTEGER,
    high INTEGER,
    low INTEGER,
    close INTEGER,
    volume INTEGER,
    PRIMARY KEY (date)
)
```

setQuery 예시

# 01. 수집기

## 키움 API 통신

키움 API 통신할 때,  
현재 가지고 있는 데이터 제외하고,  
opt10080 TR 요청 (한번에 900개)  
통신 한 번의 TR 개수: 조절 불가능

```
01_retrieveTR_min.py X 02_retrieveTR_min.py X
1 import sys
2 from static.kiwoom import *
3 from tool import retrieveTR
4 import pandas as pd
5
6 """
7 실시간 TR
8 https://hminito.tistory.com/35
9 """
10 # TR_REQ_TIME_INTERVAL = 0.2
11 if __name__ == "__main__":
12     app = Application(sys.argv)
13     kiwoom = Kiwoom()
14     kiwoom.connect()
15
16     item_code = "000000"
17     kiwoom.yyyyymmddhms = "20220118153000"
18     df = retrieveTR_TR_min(kiwoom, item_code, 1)
19     print(df)
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
263
```

# 01. 수집기

## 키움 API 통신 관련함수

static\_kiwoom VS retrieveTR VS retrieve\_static\_TR

static\_kiwoom

- 키움 통신의 직접적으로 관여

retrieveTR

- static\_kiwoom에서 정의한 함수 이용해서 일봉 데이터 return

retrieve\_static\_TR

- main 실행함수

```
fundamentalAnalysis.pyb • retrieve_static_TR.py • technicalAnalysis.pyb
retrieve_static_TR.py
1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.
12.
13.
14.
15.
16.
17.
18.
19.
20.
21.
22.
23.
24.
25.
26.
27.
28.
29.
30.
31.
32.
33.
34.
35.
36.
37.
38.
39.
40.
41.
42.
43.
44.
45.
46.
47.
48.
49.
50.
51.
52.
53.
54.
55.
56.
57.
58.
59.
60.
61.
62.
63.
64.
65.
66.
67.
68.
69.
70.
71.
72.
73.
74.
75.
76.
77.
78.
79.
80.
81.
82.
83.
84.
85.
86.
87.
88.
89.
90.
91.
92.
93.
94.
95.
96.
97.
98.
99.
100.
101.
102.
103.
104.
105.
106.
107.
108.
109.
110.
111.
112.
113.
114.
115.
116.
117.
118.
119.
120.
121.
122.
123.
124.
125.
126.
127.
128.
129.
130.
131.
132.
133.
134.
135.
136.
137.
138.
139.
140.
141.
142.
143.
144.
145.
146.
147.
148.
149.
150.
151.
152.
153.
154.
155.
156.
157.
158.
159.
160.
161.
162.
163.
164.
165.
166.
167.
168.
169.
170.
171.
172.
173.
174.
175.
176.
177.
178.
179.
180.
181.
182.
183.
184.
185.
186.
187.
188.
189.
190.
191.
192.
193.
194.
195.
196.
197.
198.
199.
200.
201.
202.
203.
204.
205.
206.
207.
208.
209.
210.
211.
212.
213.
214.
215.
216.
217.
218.
219.
220.
221.
222.
223.
224.
225.
226.
227.
228.
229.
230.
231.
232.
233.
234.
235.
236.
237.
238.
239.
240.
241.
242.
243.
244.
245.
246.
247.
248.
249.
250.
251.
252.
253.
254.
255.
256.
257.
258.
259.
260.
261.
262.
263.
264.
265.
266.
267.
268.
269.
270.
271.
272.
273.
274.
275.
276.
277.
278.
279.
280.
281.
282.
283.
284.
285.
286.
287.
288.
289.
290.
291.
292.
293.
294.
295.
296.
297.
298.
299.
300.
301.
302.
303.
304.
305.
306.
307.
308.
309.
310.
311.
312.
313.
314.
315.
316.
317.
318.
319.
320.
321.
322.
323.
324.
325.
326.
327.
328.
329.
330.
331.
332.
333.
334.
335.
336.
337.
338.
339.
340.
341.
342.
343.
344.
345.
346.
347.
348.
349.
350.
351.
352.
353.
354.
355.
356.
357.
358.
359.
360.
361.
362.
363.
364.
365.
366.
367.
368.
369.
370.
371.
372.
373.
374.
375.
376.
377.
378.
379.
380.
381.
382.
383.
384.
385.
386.
387.
388.
389.
390.
391.
392.
393.
394.
395.
396.
397.
398.
399.
400.
401.
402.
403.
404.
405.
406.
407.
408.
409.
410.
411.
412.
413.
414.
415.
416.
417.
418.
419.
420.
421.
422.
423.
424.
425.
426.
427.
428.
429.
430.
431.
432.
433.
434.
435.
436.
437.
438.
439.
440.
441.
442.
443.
444.
445.
446.
447.
448.
449.
450.
451.
452.
453.
454.
455.
456.
457.
458.
459.
460.
461.
462.
463.
464.
465.
466.
467.
468.
469.
470.
471.
472.
473.
474.
475.
476.
477.
478.
479.
480.
481.
482.
483.
484.
485.
486.
487.
488.
489.
490.
491.
492.
493.
494.
495.
496.
497.
498.
499.
500.
501.
502.
503.
504.
505.
506.
507.
508.
509.
510.
511.
512.
513.
514.
515.
516.
517.
518.
519.
520.
521.
522.
523.
524.
525.
526.
527.
528.
529.
530.
531.
532.
533.
534.
535.
536.
537.
538.
539.
540.
541.
542.
543.
544.
545.
546.
547.
548.
549.
550.
551.
552.
553.
554.
555.
556.
557.
558.
559.
560.
561.
562.
563.
564.
565.
566.
567.
568.
569.
570.
571.
572.
573.
574.
575.
576.
577.
578.
579.
580.
581.
582.
583.
584.
585.
586.
587.
588.
589.
590.
591.
592.
593.
594.
595.
596.
597.
598.
599.
600.
601.
602.
603.
604.
605.
606.
607.
608.
609.
610.
611.
612.
613.
614.
615.
616.
617.
618.
619.
620.
621.
622.
623.
624.
625.
626.
627.
628.
629.
630.
631.
632.
633.
634.
635.
636.
637.
638.
639.
640.
641.
642.
643.
644.
645.
646.
647.
648.
649.
650.
651.
652.
653.
654.
655.
656.
657.
658.
659.
660.
661.
662.
663.
664.
665.
666.
667.
668.
669.
670.
671.
672.
673.
674.
675.
676.
677.
678.
679.
680.
681.
682.
683.
684.
685.
686.
687.
688.
689.
690.
691.
692.
693.
694.
695.
696.
697.
698.
699.
700.
701.
702.
703.
704.
705.
706.
707.
708.
709.
710.
711.
712.
713.
714.
715.
716.
717.
718.
719.
720.
721.
722.
723.
724.
725.
726.
727.
728.
729.
730.
731.
732.
733.
734.
735.
736.
737.
738.
739.
740.
741.
742.
743.
744.
745.
746.
747.
748.
749.
750.
751.
752.
753.
754.
755.
756.
757.
758.
759.
760.
761.
762.
763.
764.
765.
766.
767.
768.
769.
770.
771.
772.
773.
774.
775.
776.
777.
778.
779.
780.
781.
782.
783.
784.
785.
786.
787.
788.
789.
790.
791.
792.
793.
794.
795.
796.
797.
798.
799.
800.
801.
802.
803.
804.
805.
806.
807.
808.
809.
810.
811.
812.
813.
814.
815.
816.
817.
818.
819.
820.
821.
822.
823.
824.
825.
826.
827.
828.
829.
830.
831.
832.
833.
834.
835.
836.
837.
838.
839.
840.
841.
842.
843.
844.
845.
846.
847.
848.
849.
850.
851.
852.
853.
854.
855.
856.
857.
858.
859.
860.
861.
862.
863.
864.
865.
866.
867.
868.
869.
870.
871.
872.
873.
874.
875.
876.
877.
878.
879.
880.
881.
882.
883.
884.
885.
886.
887.
888.
889.
890.
891.
892.
893.
894.
895.
896.
897.
898.
899.
900.
901.
902.
903.
904.
905.
906.
907.
908.
909.
910.
911.
912.
913.
914.
915.
916.
917.
918.
919.
920.
921.
922.
923.
924.
925.
926.
927.
928.
929.
930.
931.
932.
933.
934.
935.
936.
937.
938.
939.
940.
941.
942.
943.
944.
945.
946.
947.
948.
949.
950.
951.
952.
953.
954.
955.
956.
957.
958.
959.
960.
961.
962.
963.
964.
965.
966.
967.
968.
969.
970.
971.
972.
973.
974.
975.
976.
977.
978.
979.
980.
981.
982.
983.
984.
985.
986.
987.
988.
989.
990.
991.
992.
993.
994.
995.
996.
997.
998.
999.
1000.
1001.
1002.
1003.
1004.
1005.
1006.
1007.
1008.
1009.
1010.
1011.
1012.
1013.
1014.
1015.
1016.
1017.
1018.
1019.
1020.
1021.
1022.
1023.
1024.
1025.
1026.
1027.
1028.
1029.
1030.
1031.
1032.
1033.
1034.
1035.
1036.
1037.
1038.
1039.
1040.
1041.
1042.
1043.
1044.
1045.
1046.
1047.
1048.
1049.
1050.
1051.
1052.
1053.
1054.
1055.
1056.
1057.
1058.
1059.
1060.
1061.
1062.
1063.
1064.
1065.
1066.
1067.
1068.
1069.
1070.
1071.
1072.
1073.
1074.
1075.
1076.
1077.
1078.
1079.
1080.
1081.
1082.
1083.
1084.
1085.
1086.
1087.
1088.
1089.
1090.
1091.
1092.
1093.
1094.
1095.
1096.
1097.
1098.
1099.
1100.
1101.
1102.
1103.
1104.
1105.
1106.
1107.
1108.
1109.
1110.
1111.
1112.
1113.
1114.
1115.
1116.
1117.
1118.
1119.
1120.
1121.
1122.
1123.
1124.
1125.
1126.
1127.
1128.
1129.
1130.
1131.
1132.
1133.
1134.
1135.
1136.
1137.
1138.
1139.
1140.
1141.
1142.
1143.
1144.
1145.
1146.
1147.
1148.
1149.
1150.
1151.
1152.
1153.
1154.
1155.
1156.
1157.
1158.
1159.
1160.
1161.
1162.
1163.
1164.
1165.
1166.
1167.
1168.
1169.
1170.
1171.
1172.
1173.
1174.
1175.
1176.
1177.
1178.
1179.
1180.
1181.
1182.
1183.
1184.
1185.
1186.
1187.
1188.
1189.
1190.
1191.
1192.
1193.
1194.
1195.
1196.
1197.
1198.
1199.
1200.
1201.
1202.
1203.
1204.
1205.
1206.
1207.
1208.
1209.
1210.
1211.
1212.
1213.
1214.
1215.
1216.
1217.
1218.
1219.
1220.
1221.
1222.
1223.
1224.
1225.
1226.
1227.
1228.
1229.
1230.
1231.
1232.
1233.
1234.
1235.
1236.
1237.
1238.
1239.
1240.
1241.
1242.
1243.
1244.
1245.
1246.
1247.
1248.
1249.
1250.
1251.
1252.
1253.
1254.
1255.
1256.
1257.
1258.
1259.
1260.
1261.
1262.
1263.
1264.
1265.
1266.
1267.
1268.
1269.
1270.
1271.
1272.
1273.
1274.
1275.
1276.
1277.
1278.
1279.
1280.
1281.
1282.
1283.
1284.
1285.
1286.
1287.
1288.
1289.
1290.
1291.
1292.
1293.
1294.
1295.
1296.
1297.
1298.
1299.
1300.
1301.
1302.
1303.
1304.
1305.
1306.
1307.
1308.
1309.
1310.
1311.
1312.
1313.
1314.
1315.
1316.
1317.
1318.
1319.
1320.
1321.
1322.
1323.
1324.
1325.
1326.
1327.
1328.
1329.
1330.
1331.
1332.
1333.
1334.
1335.
1336.
1337.
1338.
1339.
1340.
1341.
1342.
1343.
1344.
1345.
1346.
1347.
1348.
1349.
1350.
1351.
1352.
1353.
1354.
1355.
1356.
1357.
1358.
1359.
1360.
1361.
1362.
1363.
1364.
1365.
1366.
1367.
1368.
1369.
1370.
1371.
1372.
1373.
1374.
1375.
1376.
1377.
1378.
1379.
1380.
1381.
1382.
1383.
1384.
1385.
1386.
1387.
1388.
1389.
1390.
1391.
1392.
1393.
1394.
1395.
1396.
1397.
1398.
1399.
1400.
1401.
1402.
1403.
1404.
1405.
1406.
1407.
1408.
1409.
1410.
1411.
1412.
1413.
1414.
1415.
1416.
1417.
1418.
1419.
1420.
1421.
1422.
1423.
1424.
1425.
1426.
1427.
1428.
1429.
1430.
1431.
1432.
1433.
1434.
1435.
1436.
1437.
1438.
1439.
1440.
1441.
1442.
1443.
1444.
1445.
1446.
1447.
1448.
1449.
1450.
1451.
1452.
1453.
1454.
1455.
1456.
1457.
1458.
1459.
1460.
1461.
1462.
1463.
1464.
1465.
1466.
1467.
1468.
1469.
1470.
1471.
1472.
1473.
1474.
1475.
1476.
1477.
1478.
1479.
1480.
1481.
1482.
1483.
1484.
1485.
1486.
1487.
1488.
1489.
1490.
1491.
1492.
1493.
1494.
1495.
1496.
1497.
1498.
1499.
1500.
1501.
1502.
1503.
1504.
1505.
1506.
1507.
1508.
1509.
1510.
1511.
1512.
1513.
1514.
1515.
1516.
1517.
1518.
1519.
1520.
1521.
1522.
1523.
1524.
1525.
1526.
1527.
1528.
1529.
1530.
1531.
1532.
1533.
1534.
1535.
1536.
1537.
1538.
1539.
1540.
1541.
1542.
1543.
1544.
1545.
1546.
1547.
1548.
1549.
1550.
1551.
1552.
1553.
1554.
1555.
1556.
1557.
1558.
1559.
1560.
1561.
1562.
1563.
1564.
1565.
1566.
1567.
1568.
1569.
1570.
1571.
1572.
1573.
1574.
1575.
1576.
1577.
1578.
1579.
1580.
1581.
1582.
1583.
1584.
1585.
1586.
1587.
1588.
1589.
1590.
1591.
1592.
1593.
1594.
1595.
1596.
1597.
1598.
1599.
1600.
1601.
1602.
1603.
1604.
1605.
1606.
1607.
1608.
1609.
1610.
1611.
1612.
1613.
1614.
1615.
1616.
1617.
1618.
1619.
1620.
1621.
1622.
1623.
1624.
1625.
1626.
1627.
1628.
1629.
1630.
1631.
1632.
1633.
1634.
1635.
1636.
1637.
1638.
1639.
1640.
1641.
1642.
1643.
1644.
1645.
1646.
1647.
1648.
1649.
1650.
1651.
1652.
1653.
1654.
1655.
1656.
1657.
1658.
1659.
1660.
1661.
1662.
1663.
1664.
1665.
1666.
1667.
1668.
1669.
1670.
1671.
1672.
1673.
1674.
1675.
1676.
1677.
1678.
1679.
1680.
1681.
1682.
1683.
1684.
1685.
1686.
1687.
1688.
1689.
1690.
1691.
1692.
1693.
1694.
1695.
1696.
1697.
1698.
1699.
1700.
1701.
1702.
1703.
1704.
1705.
1706.
1707.
1708.
1709.
1710.
1711.
1712.
1713.
1714.
1715.
1716.
1717.
1718.
1719.
1720.
1721.
1722.
1723.
1724.
1725.
1726.
1727.
1728.
1729.
1730.
1731.
1732.
1733.
1734.
1735.
1736.
1737.
1738.
1739.
1740.
1741.
1742.
1743.
1744.
1745.
1746.
1747.
1748.
1749.
1750.
1751.
1752.
1753.
1754.
1755.
1756.
1757.
1758.
1759.
1760.
1761.
1762.
1763.
1764.
1765.
1766.
1767.
1768.
1769.
1770.
1771.
1772.
1773.
1774.
1775.
1776.
1777.
1778.
1779.
1780.
1781.
1782.
1783.
1784.
1785.
1786.
1787.
1788.
1789.
1790.
1791.
1792.
1793.
1794.
1795.
1796.
1797.
1798.
1799.
1800.
1801.
1802.
1803.
1804.
1805.
1806.
1807.
1808.
1809.
1810.
1811.
1812.
1813.
1814.
1815.
1816.
1817.
1818.
1819.
1820.
1821.
1822.
1823.
1824.
1825.
1826.
1827.
1828.
1829.
1830.
1831.
1832.
1833.
1834.
1835.
1836.
1837.
1838.
1839.
1840.
1841.
1842.
1843.
1844.
1845.
1846.
1847.
1848.
1849.
1850.
1851.
1852.
1853.
1854.
1855.
1856.
1857.
1858.
1859.
1860.
1861.
1862.
1863.
1864.
1865.
1866.
1867.
1868.
1869.
1870.
1871.
1872.
1873.
1874.
1875.
1876.
1877.
1878.
1879.
1880.
1881.
1882.
1883.
1884.
1885.
1886.
1887.
1888.
1889.
1890.
1891.
1892.
1893.
1894.
1895.
1896.
1897.
1898.
1899.
1900.
1901.
1902.
1903.
1904.
1905.
1906.
1907.
1908.
1909.
1910.
1911.
1912.
1913.
1914.
1915.
1916.
1917.
1918.
1919.
1920.
1921.
1922.
1923.
1924.
1925.
1926.
1927.
1928.
1929.
1930.
1931.
1932.
1933.
1934.
1935.
1936.
1937.
1938.
1939.
1940.
1941.
1942.
1943.
1944.
1945.
1946.
1947.
1948.
1949.
1950.
1951.
1952.
1953.
1954.
1955.
1956.
1957.
1958.
1959.
1960.
1961.
1962.
1963.
1964.
1965.
1966.
1967.
1968.
1969.
1970.
1971.
1972.
1973.
1974.
1975.
1976.
1977.
1978.
1979.
1980.
1981.
1982.
1983.
1984.
1985.
1986.
1987.
1988.
1989.
1990.
1991.
1992.
1993.
1994.
1995.
1996.
1997.
1998.
1999.
2000.
2001.
2002.
2003.
2004.
2005.
2006.
2007.
2008.
2009.
2010.
2011.
2012.
2013.
2014.
2015.
2016.
2017.
2018.
2019.
2020.
2021.
2022.
2023.
2024.
2025.
2026.
2027.
2028.
2029.
2030.
2031.
2032.
2033.
2034.
2035.
2036.
2037.
2038.
2039.
2040.
2041.
2042.
2043.
2044.
2045.
2046.
2047.
2048.
2049.
2050.
2051.
2052.
2053.
2054.
2055.
2056.
2057.
2058.
2059.
2060.
2061.
2062.
2063.
2064.
2065.
2066.
2067.
2068.
2069.
2070.
2071.
2072.
2073.
2074.
2075.
2076.
2077.
2078.
2079.
2080.
2081.
2082.
2083.
2084.
2085.
2086.
2087.
2088.
2089.
2090.
2091.
2092.
2093.
2094.
2095.
2096.
2097.
2098.
2099.
2100.
2101.
2102.
2103.
2104.
2105.
2106.
2107.
2108.
2109.
2110.
2111.
2112.
2113.
2114.
2115.
2116.
2117.
2118.
2119.
2120.
2121.
2122.
2123.
2124.
2125.
2126.
2127.
2128.
2129.
2130.
2131.
2132.
2133.
2134.
2135.
2136.
2137.
2138.
2139.
2140.
2141.
2142.
2143.
2144.
2145.
2146.
2147.
2148.
2149.
2150.
2151.
2152.
2153.
2154.
2155.
2156.
2157.
2158.
2159.
2160.
2161.
2162.
2163.
2164.
2165.
2166.
2167.
2168.
2169.
2170.
2171.
2172.
2173.
2174.
2175.
2176.
2177.
21
```

# 01. 수집기

## 예시

case1)

company\_info: 종목 코드 테이블의 정보 조회

- 069500: 2022년 11월 18일 06시
- 114800: 2022년 11월 18일 11시
- 226490: 조회 x -> remained\_data=True일 때까지 정보 update

sqlWriter()

- 069500: 2022.11.18 09시 00분 ~
  - 114800: 2022.11.18 11시 49분 ~
  - 226490: 2021.12.06 09시 00분 ~
- \*조회 종료 시점: 조회 시점.today()의 전날 15시 30분

update\_comany\_info()

- dataframe의 가장 최근 날짜를 업데이트 (=조회 종료 시점)

	st_code	nm_code	last_update
	필터	필터	필터
1	069500	kodex_200	20221118069500
2	114800	kodex_inverse	20221118114800

20221118090400	31785	31805	31785	31790	5343
20221118090300	31830	31845	31790	31790	22231
20221118090200	31795	31840	31795	31830	28964
20221118090100	31755	31810	31750	31800	7647
20221118090000	31795	31795	31745	31755	26089
[4656 rows x 5 columns]					
	open	high	low	close	volume
20221206101700	4910	4915	4905	4910	15487
20221206101600	4910	4915	4910	4910	20526
20221206101500	4910	4915	4910	4910	538
20221206101400	4915	4915	4910	4910	10104
20221206101300	4910	4915	4910	4915	2969
...	...	...	...	...	...
20221118115300	4795	4795	4795	4795	58285
20221118115200	4795	4795	4795	4795	16
20221118115100	4795	4800	4790	4795	97513
20221118115000	4800	4800	4795	4795	53204
20221118114900	4795	4800	4795	4795	6351
[4480 rows x 5 columns]					
	open	high	low	close	volume
20221206102100	24465	24465	24450	24450	1500
20221206101500	24465	24465	24465	24465	14
20221206101400	24470	24470	24470	24470	2
20221206101300	24475	24475	24470	24470	122
20221206101200	24490	24490	24485	24485	82
...	...	...	...	...	...
20211206090400	29220	29220	29210	29220	346
20211206090300	29299	29299	29235	29235	5
20211206090200	29245	29245	29225	29225	77
20211206090100	29340	29343	29330	29330	131
20211206090000	29323	29323	29230	29230	743
[77581 rows x 5 columns]					

# 01. 수집기

static\_TR 완료 후 DB 모습

	st_code	nm_code	last_update
	필터	필터	필터
1	069500	KODEX 200	20211201091300
2	114800	KODEX 인버스	20221209153000
3	226490	KODEX 코스피	20221209153000

테이블(T): 069500

	date	open	high	low	close	volume
	필터	필터	필터	필터	필터	필터
1	20211201090000	36742	36932	36742	36825	81467
2	20211201090100	36830	36957	36830	36937	97355
3	20211201090200	36927	36991	36927	36932	57278
4	20211201090300	36942	37001	36942	36991	19861
5	20211201090400	36991	37011	36962	36972	13302
6	20211201090500	36962	36962	36879	36879	20260
7	20211201090600	36874	36903	36805	36805	38408
8	20211201090700	36800	36825	36771	36781	42830
9	20211201090800	36776	36790	36742	36746	11226
10	20211201090900	36746	36869	36746	36849	51686
11	20211201091000	36854	36918	36835	36918	19066
12	20211201091100	36918	36927	36883	36908	8039
13	20211201091200	36898	36942	36898	36903	21819
14	20211201091300	36898	36913	36844	36854	19864



테이블(T): company\_info

	st_code	nm_code	last_update
	필터	필터	필터
1	069500	KODEX 200	20221209153000
2	114800	KODEX 인버스	20221209153000
3	226490	KODEX 코스피	20221209153000

테이블(T): 069500

	date	open	high	low	close	volume
	필터	필터	필터	필터	필터	필터
1	20211201090000	36742	36932	36742	36825	81467
2	20211201090100	36830	36957	36830	36937	97355
3	20211201090200	36927	36991	36927	36932	57278
4	20211201090300	36942	37001	36942	36991	19861
5	20211201090400	36991	37011	36962	36972	13302
6	20211201090500	36962	36962	36879	36879	20260
7	20211201090600	36874	36903	36805	36805	38408
8	20211201090700	36800	36825	36771	36781	42830
9	20211201090800	36776	36790	36742	36746	11226
10	20211201090900	36746	36869	36746	36849	51686
11	20211201091000	36854	36918	36835	36918	19066
12	20211201091100	36918	36927	36883	36908	8039
13	20211201091200	36898	36942	36898	36903	21819
14	20211201091300	36898	36913	36844	36854	19864

96965	20221209151900	31210	31225	31210	31220	1982
96966	20221209153000	31230	31230	31230	31230	55738

96951 - 96966 of 96966

특정 레코드 행으로 가기:

# 01. 수집기

## 실시간 TR 메소드

### SetRealReg()

- 특정 종목들에 대한 실시간 타입을 구독 신청

### OnReceiveRealData()

- 구독 해지를 하기 전까지 서버에서 실시간 타입에 대한 이벤트가 발생할 때 마다 프로그램에 이벤트 발생시킴

### on\_receive\_real\_data()

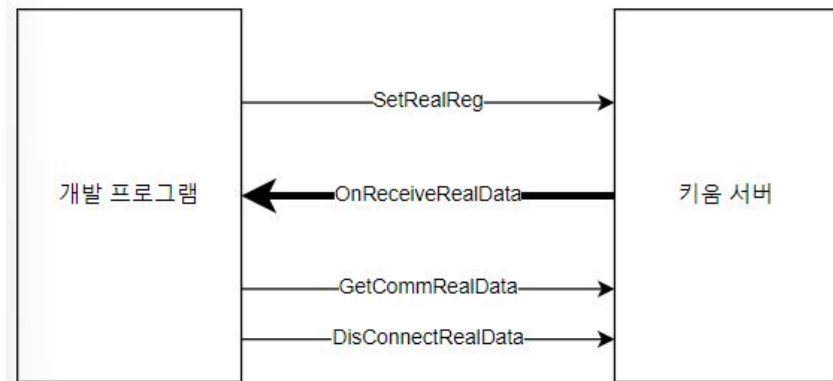
- OnReceiveRealData 이벤트를 처리하는 핸들러

### GetCommRealData()

- 특정 FID 값에 해당하는 데이터를 가져옴

### DisconnectRealData()

- 구독 해지



# 01. 수집기

## 실시간 TR받아오기

```
SetRealReg( self,  
screen_no,    // 화면 번호  
code_list,    // 종목 코드 리스트  
fid_list,     // 실시간 FID 리스트  
real_tpye    // 실시간 등록 타입  
)
```

```
def SetRealReg(self, screen_no, code_list, fid_list, real_type):  
    self.ocx.dynamicCall("SetRealReg(QString, QString, QString, QString)",  
                           screen_no, code_list, fid_list, real_type)
```

ex)

```
self.SetRealReg("2000", "005930", "215;20;214", 0)
```

- 종목 코드와 FID 리스트를 이용해서 실시간 시세를 등록하는 함수
- 한 번에 등록 가능한 종목과 FID 개수는 100종목
- 실시간 등록 타입을 0으로 설정 시 기존 종목들은 해지되고 등록된 종목의 실시간 시세가 등록됨
- 실시간 등록 타입을 1로 설정 시 먼저 등록한 종목들과 함께 실시간 시세가 등록됨

# 01. 수집기

실시간 TR받아오기

```
_on_receive_real_data()  
def _on_receive_real_data(self, code, real_type, data):  
    if real_type == "장시작시간":  
        gubun = self.GetCommRealData(code, 215)  
        remained_time = self.GetCommRealData(code, 214)  
        print(gubun, remained_time)  
        if gubun == 4:  
            self.DisConnectRealData("1000")
```

- 장 종과 장시작 전& 장 시작 후 로 구분하여 실시간 데이터를 받아옴
- 장시작 전 후에는 10분 단위로 이벤트 발생
- 1분을 남기고는 10초 단위로 이벤트 발생



# 01. 수집기

## 실시간 TR받아오기

\_on\_receive\_real\_data()

장 중에는 주문체결 시간과 ohlcv데이터를 받아옴 -> 딕셔너리에 저장

```
elif real type == "주식체결":  
    signed at = self.ocx.dynamicCall("GetCommRealData(QString, int)", code, 20)  
    close = self.ocx.dynamicCall("GetCommRealData(QString, int)", code, 10)  
    close = abs(int(close))  
    high = self.ocx.dynamicCall("GetCommRealData(QString, int)", code, 17)  
    high = abs(int(high))  
    open = self.ocx.dynamicCall("GetCommRealData(QString, int)", code, 16)  
    open = abs(int(open))  
    low = self.ocx.dynamicCall("GetCommRealData(QString, int)", code, 18)  
    low = abs(int(low))  
    accum volume = self.ocx.dynamicCall("GetCommRealData(QString, int)", code, 13)  
    accum volume = abs(int(accum volume))
```

# 01. 수집기

## 실시간 TR받아오기

```
Crystal012 > Research > real-time_TR > realTime_TR.py > MyWindow > _on_receive_real_data
64
65 def _on_receive_real_data(self, code, real_type, data):
66     if real_type == "장시작시간":
67         gubun = self.GetCommRealData(code, 215)
68         remained_time = self.GetCommRealData(code, 214)
69         print(gubun, remained_time)
70         if gubun == 4:
71             self.DisconnectRealData("1000")
72
73     elif real_type == "주식체결":
74
```

문제 출력 디버그 콘솔 터미널

```
111124 59500 60000 59900 59400 4726261
111125 59600 60000 59900 59400 4726407
111125 59600 60000 59900 59400 4726409
111125 59500 60000 59900 59400 4726426
111125 59500 60000 59900 59400 4726446
111126 59600 60000 59900 59400 4726569
111126 59500 60000 59900 59400 4726577
111126 59600 60000 59900 59400 4726587
111126 59500 60000 59900 59400 4726588
111129 59600 60000 59900 59400 4726589
111129 59600 60000 59900 59400 4726627
111130 59500 60000 59900 59400 4726680
111130 59600 60000 59900 59400 4726682
111131 59500 60000 59900 59400 4726696
111131 59500 60000 59900 59400 4726713
111131 59600 60000 59900 59400 4726714
111131 59600 60000 59900 59400 4726764
111132 59600 60000 59900 59400 4726782
111133 59600 60000 59900 59400 4726792
111133 59600 60000 59900 59400 4726817
111133 59500 60000 59900 59400 4726819
111133 59600 60000 59900 59400 4726820
111133 59600 60000 59900 59400 4726847
111134 59500 60000 59900 59400 4726849
111134 59500 60000 59900 59400 4726850
111134 59500 60000 59900 59400 4726851
111134 59500 60000 59900 59400 4726853
111134 59600 60000 59900 59400 4726860
```

069500 151958 31070 31100 31010 30930 2367228

069500 151959 31070 31100 31010 30930 2367728

2 001000

2 000900

2 000800

2 000700

2 000600

2 000500

2 000400

2 000300

2 000200

2 000100

2 000050

2 000040

2 000030

2 000020

2 000010

4 000000

069500 153017 31035 31100 31010 30930 2417378

8 000000

주식코드 / 체결시간 / 시가 / 고가 / 저가 / 종가 / 누적 거래량

장운영구분

0 - 장시작 전

3 - 장 중

2 - 장 종료 10분전 동시호가

4 - 장 종료

## 02. 분석기(1)

데이터 전처리(rltrainer/data\_manger.py)

차트 데이터는 날짜, 시가, 고가, 저가, 종가, 거래량으로 구성

현재 종가와 전일 종가의 비율, 이동평균 종가의 비율, 현재 거래량과 전일 거래량의 비율, 이동평균 거래량의 비율을 학습에 사용(총 31개의 열)

- 각 이동평균에 대한 이동평균 종가 비율, 이동평균 거래량 비율: `close_ma{이동평균크기}_ratio`,

`volume_ma{이동평균크기}_ratio` -> (`rolling().mean()` 함수, 5 10 20 60 120)

- 시가/전일종가 비율: `open_lastclose_ratio`

- 고가/종가 비율: `high_close_ratio`

- 저가/종가 비율: `low_close_ratio`

- 종가/전일종가 비율: `close_lastclose_ratio`

- 거래량/전일거래량 비율: `volume_lastvolume_ratio`

## 02. 분석기(2)

가시화 모듈(visualizer.py)

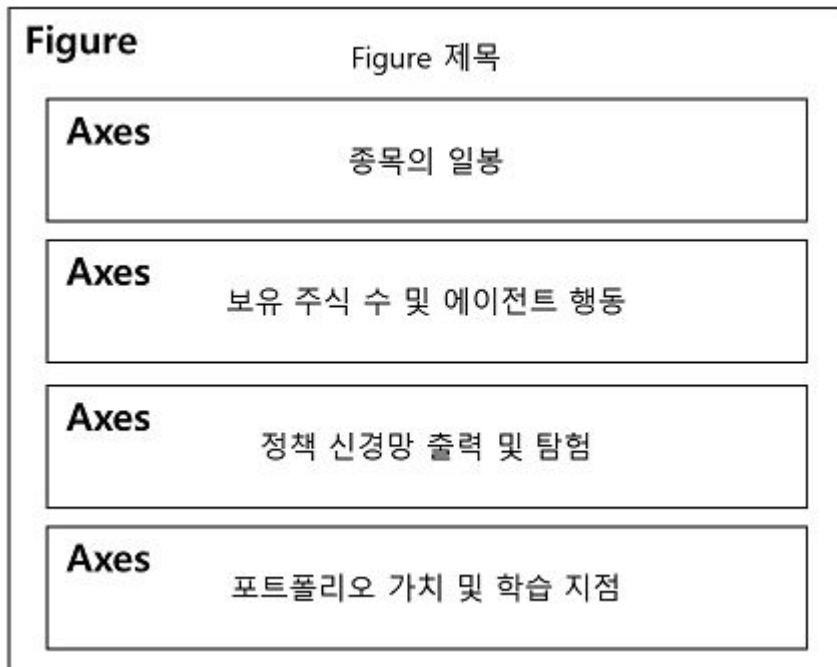
Figure 제목: 파라미터, 에포크 및 탐험률

Axes 1: 종목의 일봉 차트

Axes 2: 보유 주식 수 및 에이전트 행동 차트

Axes 3: 정책 신경망 출력 및 탐험 차트

Axes 4: 포트폴리오 가치 및 학습 지점 차트



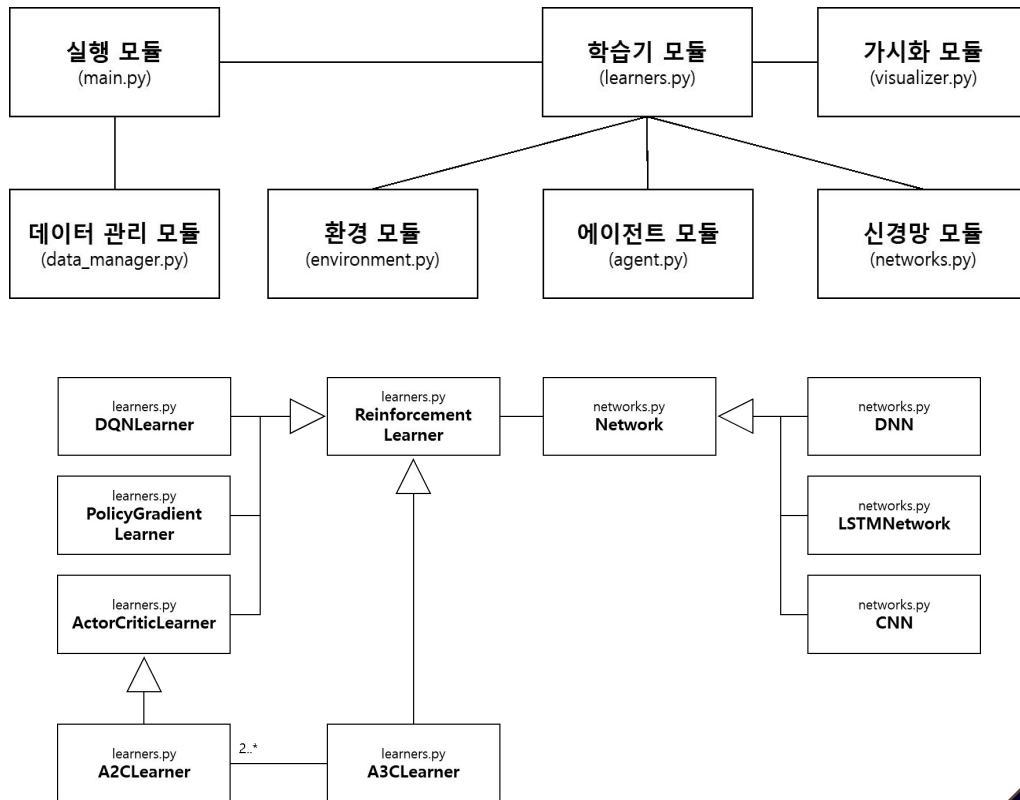
### 03. 판단로직(1)

#### 강화학습 모듈

- 실행 모듈(main.py)
- 학습기 모듈(learner.py)
- 가시화 모듈(visualizer.py)
- 데이터 관리 모듈(data\_manager.py)
- 환경 모듈(environment.py)
- 에이전트 모듈(agent.py)
- 신경망 모듈(networks.py)

#### 클래스 다이어그램

- Policy Gradient Learner
- Reinforcement Learner
- LSTM Network



## 03. 판단로직(2)

### 학습기 모듈(Learner.py)

#### Policy Gradient

- x: 일련의 학습 데이터 및 에이전트 상태로 구성된 샘플 배열
- y\_policy: 정책 신경망의 출력
- simoid(보상 값): 정책신경망 학습 레이블

#### LSTM

- get\_network\_head(): LSTM 구조 설정
- train\_on\_batch(): 학습 데이터와 레이블 x, y를 입력으로 받아서 모델을 학습
- predict(): 샘플에 대한 행동의 가치 또는 확률 예측

```
class PolicyGradientLearner(ReinforcementLearner):
    def __init__(self, *args, policy_network_path=None, **kwargs):
        super().__init__(*args, **kwargs)
        self.policy_network_path = policy_network_path
        self.init_policy_network()

    def get_batch(self):
        memory = zip(
            reversed(self.memory_sample),
            reversed(self.memory_action),
            reversed(self.memory_policy),
            reversed(self.memory_reward),
        )
        x = np.zeros((len(self.memory_sample), self.num_steps, self.num_features))
        y_policy = np.zeros((len(self.memory_sample), self.agent.NUM_ACTIONS))
        for i, (sample, action, policy, reward) in enumerate(memory):
            x[i] = sample
            r = self.memory_reward[-1] - reward
            y_policy[i, :] = policy
            y_policy[i, action] = utils.sigmoid(r)

        return x, None, y_policy
```

### 03. 판단로직(3)

#### 학습기 모듈(Learner.py) 실행 결과

predict\_20221212022551\_pg\_lstm

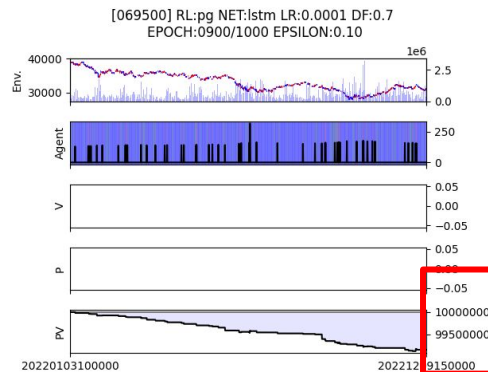
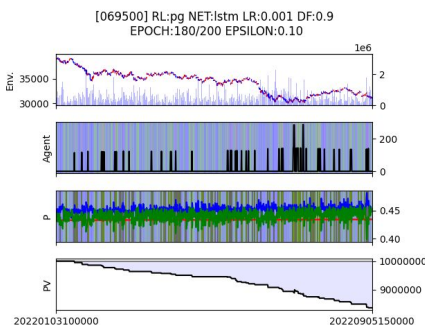
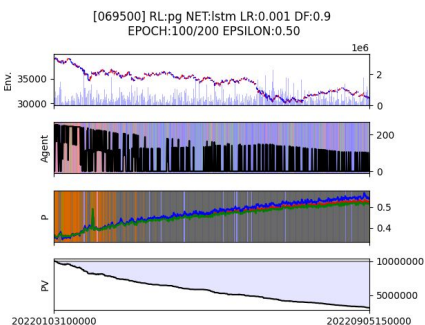
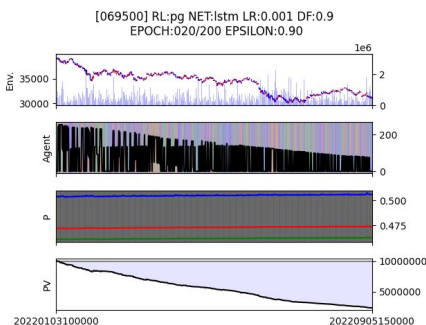
test\_20221212022551\_pg\_lstm

train\_20221212022551\_pg\_lstm

#### train, test, predict 진행

- train data = [20220101090000 ~ 20220906000000]
- test data = [20220906090000 ~ 20221206000000]
- predict data = [2022120609000 ~ 20221207153500]

train, test, predict 외에도 update 있음 (모델 추가 학습)



# 03

실전 자동 매매



## 03. 실전 자동 매매

### pytrader 함수 (1)

- stock\_account, order\_type, code, order\_quantity, order\_price, order\_classification을 입력받아 order\_type에 따라 매수, 매도 실행
- KOA 내장 함수인 SendOrder()를 통해 매수/매도 주문

```
def pyTrader(stock_account, order_type, code, order_quantity, order_price, order_classification, origin_order_number):  
    if order_type == 0:  
        if order_classification == "03":  
            kiwoom.SendOrder("시장가매수", "0101", stock_account, 1, code, order_quantity, 0, "03", "")  
        elif order_classification == "00":  
            kiwoom.SendOrder("지정가매수", "0101", stock_account, 1, code, order_quantity, order_price, "00", "")  
  
    elif order_type == 1:  
        if order_classification == "03":  
            kiwoom.SendOrder("시장가매도", "0101", stock_account, 2, code, order_quantity, 0, "03", "")  
        elif order_classification == "00":  
            kiwoom.SendOrder("지정가매도", "0101", stock_account, 2, code, order_quantity, order_price, "00", "")  
  
    elif order_type == 2:  
        pass
```

## 03. 실전 자동 매매

### pytrader 함수 (2)

```
SendOrder(  
    BSTR sRQName,      // 사용자 >구분명  
    BSTR sScreenNo,    // 화면 번호  
    BSTR sAccNo,       // 계좌번호 10자리  
    LONG nOrderType,   // 주문 유형(1: 신규 매수, 2: 신규 매도 3: 매수 취소, 4: 매도 취소, 5:  
매수 정정, 6: 매도 정정)  
    BSTR sCode,        // 종목 코드(6자리)  
    LONG nQty,         // 주문 수량  
    LONG nPrice,       // 주문 가격  
    BSTR sHogaGb,      // 거래 구분(혹은 호가 구분)은 아래 참고  
    BSTR sOrgOrderNo   // 원주문 번호. 신규 주문에는 공백 입력, 정정/취소 시 입력합니다.  
)
```

### 03. 실전 자동 매매

#### pytrader 함수 (3)

```
pyTrader(stock_account, 1, "069500", "100", 0, "03", "")
```

: 계좌에서 “069500” 종목 100주를 현재 시장가로 매수

```
pyTrader(stock_account, 0, "069500", "100", 0, "03", "")
```

: 계좌에서 “069500” 종목 100주를 현재 시장가로 매도



체결확인 대주주문 래속주문 단일가주 ⋮

8035-0376 [위탁] ●●●● 조회

전체 삼성전자 005930 🔍

종목명	주문구분	주문수량	체결수량 ▶
KODEX 200	현금매도	100	100
삼성전자	현금매수	10	10
KODEX 200	현금매수	100	100
삼성전자	현금매수	10	10

# 04

협업 내용

# 00. 협업내용

- 주 2회 오프라인 정기회의 (총 12회)
- 회의록 작성: [AIFT\(Artificail Intelligence in Financial Trading\) 2022](#)
- github pull request (총 31회)

< 오늘 >				
화	수	목	금	토
11월 1일	2	3	4	5
	1차 회의 팀프로젝트회의	2차 회의 팀프로젝트회의		
8	9	10	11	12
		3차 회의 팀프로젝트회의		진행사항 보고
15	16	17	18	19
		4차 회의 팀프로젝트회의		
22	23	24	25	26
5차 회의 팀프로젝트회의		6차 회의 팀프로젝트회의		
29	30	12월 1일	2	3
7차 회의		8차 회의 팀프로젝트회의		

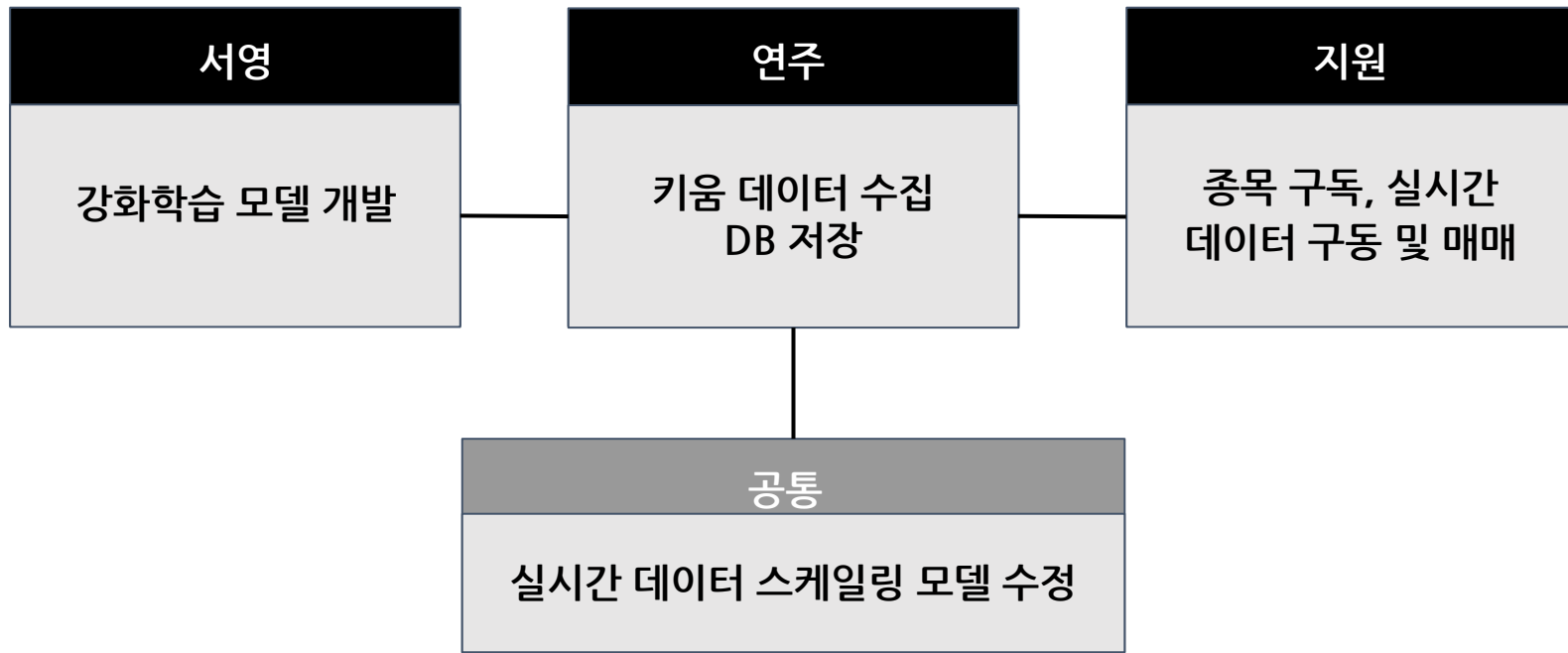
<input type="checkbox"/>	0 Open ✓ 28 Closed
<input type="checkbox"/>	model_prdict 기본틀 #28 by JiwonHeo01 was merged yesterday
<input type="checkbox"/>	Update .gitignore #27 by yeonju52 was merged yesterday
<input type="checkbox"/>	add retrieveTR_ver2 & sql->RL connect #26 by yeonju52 was merged yesterday
<input type="checkbox"/>	Feature/data collector #25 by yeonju52 was merged yesterday

## 요약

- 실시간 데이터 수집 or (실시간 체결 데이터 받아오기 - 자동 매매 서버 구축하기  
ex) 9시에 tr데이터 받기 시작)
  - 서열 : 판단 로직 - 강화학습(어떤 종목 살지 결정)
  - 지원아: 실시간 체결하기 or (실시간 데이터 받아오기)
- AWS



## 01. 역할분담



## 02. github 리포지토리

FALL2022\_AIFT: <https://github.com/FALL2022-AIFT/Crystal012>

### Folder List

- **Assignments**: 회의록 및 진행상황 보고 (.pptx or .pdf)
- **Research**: 개인 조사 및 공부, 연습 코드 업로드
- **Project**: 최종 모델 디벨롭 (.py or .ipynb)

### 브랜치별 개발

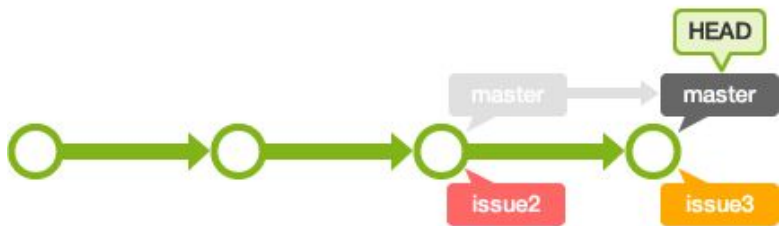
- feature: 최종 모델 배포
- research: 연습

- dataCollector: static 데이터 수집기
- rl\_model\_develop: 강화학습 모델 개발
- realtime\_TR: 실시간 데이터 수집기
- receiveTR: 1분봉 데이터
- ~~create\_mariadb: MariaDB 사용환경 구축~~  
-> sqlite3로 변경

Branches	Tags
✓ main	default
feature/dataCollector	
feature/rl_model_develop	
research/analyzer	
research/create_mariadb	
research/realtime_TR	
research/receiveTR	

### 03. 개발 경험

- 일봉이 아닌 분봉 데이터를 이용해 딥러닝 모델을 만들고 적용하는게 어려웠음
- 팀원 별로 작업 브랜치가 달라 브랜치를 작업하는 동안 main브랜치가 변경됐을 때 머지 과정에서 충돌이 일어남
  - > 내 브랜치에서 main브랜치를 rebase하고 main으로 merge하여 해결
- 파이썬 프로그램 계층구조: import 형식이 다름 (Class vs Def)
- 생각보다 학습 성능이 떨어짐
- SQL과 파이썬 연동이 어려웠음
- 개발 환경 구축이 어려웠음 (32bit vs 64bit)







# Thank you

크리스탈012

