

kaggle Search Competitions Datasets Kernels Discussion Learn ...

Profile Feature Engineered : 0.68310 Python notebook using data from WSDM - KKBox's Music Recommendation Challenge · 7,540 views ⌂ 64 Fork 171 ...

Version 5 5 commits

Notebook Data Log Comments

```
In [1]:  
# This Python 3 environment comes with many helpful analytics libraries installed  
# It is defined by the kaggle/python docker image: https://github.com/kaggle/docker-python  
# For example, here's several helpful packages to load in  
  
import numpy as np  
import pandas as pd  
import lightgbm as lgb  
import datetime  
import math  
import gc  
  
  
# Input data files are available in the "../input/" directory.  
# For example, running this (by clicking run or pressing Shift+Enter) will list the files in the  
input directory  
  
from subprocess import check_output  
print(check_output(["ls", "../input"]).decode("utf8"))  
  
# Any results you write to the current directory are saved as output.  
  
members.csv  
sample_submission.csv  
song_extra_info.csv  
songs.csv  
test.csv  
train.csv
```

```
In [2]:  
print('Loading data...')  
data_path = '../input/'  
train = pd.read_csv(data_path + 'train.csv', dtype={'msno' : 'category',  
                                                 'source_system_tab' : 'category',  
                                                 'source_screen_name' : 'category',  
                                                 'source_type' : 'category',  
                                                 'target' : np.uint8,  
                                                 'song_id' : 'category'})  
test = pd.read_csv(data_path + 'test.csv', dtype={'msno' : 'category',  
                                                 'source_system_tab' : 'category',  
                                                 'source_screen_name' : 'category',  
                                                 'source_type' : 'category',  
                                                 'song_id' : 'category'})  
songs = pd.read_csv(data_path + 'songs.csv', dtype={'genre_ids': 'category',  
                                                 'language' : 'category',  
                                                 'artist_name' : 'category',  
                                                 'composer' : 'category',  
                                                 'lyricist' : 'category',  
                                                 'song_id' : 'category'})  
members = pd.read_csv(data_path + 'members.csv', dtype={'city' : 'category',  
                                                 'bd' : np.uint8,  
                                                 'gender' : 'category',  
                                                 'registered_via' : 'category'},  
parse_dates=['registration_init_time', 'expiration_date'])  
songs_extra = pd.read_csv(data_path + 'song_extra_info.csv')  
print('Done loading...')  
  
Loading data...  
  
-----  
TypeError Traceback (most recent call last)  
<ipython-input-2-6e0df5b8adcf> in <module>()  
      6                                         'source_type' : 'category',  
      7                                         'target' : np.uint8,  
----> 8                                         'song_id' : 'category'})  
      9 test = pd.read_csv(data_path + 'test.csv', dtype={'msno' : 'category',  
     10                                         'source_system_tab' : 'category',  
  
/opt/conda/lib/python3.6/site-packages/pandas/io/parsers.py in parser_f(filepath_or_buffer, se  
p, delimiter, header, names, index_col, usecols, squeeze, prefix, mangle_dupe_cols, dtype, engi  
ne, converters, true_values, false_values, skipinitialspace, skiprows, nrows, na_values, keep_d
```

```
default_na, na_filter, verbose, skip_blank_lines, parse_dates, infer_datetime_format, keep_date_col, date_parser, dayfirst, iterator, chunksize, compression, thousands, decimal, lineterminator, quotechar, quoting, escapechar, comment, encoding, dialect, tupleize_cols, error_bad_lines, warn_bad_lines, skipfooter, skip_footer, doublequote, delim_whitespace, as_recarray, compact_ints, use_unsigned, low_memory, buffer_lines, memory_map, float_precision)
    703                 skip_blank_lines=skip_blank_lines)
    704
--> 705         return _read(filepath_or_buffer, kwds)
    706
    707     parser_f._name__ = name

/opt/conda/lib/python3.6/site-packages/pandas/io/parsers.py in _read(filepath_or_buffer, kwds)
    449
    450     try:
--> 451         data = parser.read(nrows)
    452     finally:
    453         parser.close()

/opt/conda/lib/python3.6/site-packages/pandas/io/parsers.py in read(self, nrows)
   1063             raise ValueError('skipfooter not supported for iteration')
   1064
-> 1065         ret = self._engine.read(nrows)
   1066
   1067         if self.options.get('as_recarray'):

/opt/conda/lib/python3.6/site-packages/pandas/io/parsers.py in read(self, nrows)
  1826     def read(self, nrows=None):
  1827         try:
-> 1828             data = self._reader.read(nrows)
  1829         except StopIteration:
  1830             if self._first_chunk:

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader.read()
pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._read_low_memory()
pandas/_libs/parsers.pyx in pandas._libs.parsers._concatenate_chunks()

/opt/conda/lib/python3.6/site-packages/numpy/core/numerictypes.py in find_common_type(array_types, scalar_types)
  1013
  1014 """
-> 1015     array_types = [dtype(x) for x in array_types]
  1016     scalar_types = [dtype(x) for x in scalar_types]
  1017
  1018     if self._first_chunk:
  1019
  1020         . . .
  1021
  1022     if self._first_chunk:
  1023
  1024         . . .
  1025
  1026     if self._first_chunk:
  1027
  1028         . . .
  1029
  1030         if self._first_chunk:
  1031
  1032         . . .
  1033
  1034         . . .
  1035     array_types = [dtype(x) for x in array_types]
  1036     scalar_types = [dtype(x) for x in scalar_types]
  1037
  1038
  1039     . . .
  1040
  1041     . . .
  1042
  1043     . . .
  1044
  1045     . . .
  1046     array_types = [dtype(x) for x in array_types]
  1047     scalar_types = [dtype(x) for x in scalar_types]
  1048
  1049
  1050     . . .
  1051
  1052     . . .
  1053
  1054     . . .
  1055     array_types = [dtype(x) for x in array_types]
  1056     scalar_types = [dtype(x) for x in scalar_types]
  1057
  1058
  1059     . . .
  1060
  1061     . . .
  1062
  1063     . . .
  1064
  1065     . . .
  1066     array_types = [dtype(x) for x in array_types]
  1067     scalar_types = [dtype(x) for x in scalar_types]
  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
```

```
In [3]: print('Data merging...')

train = train.merge(songs, on='song_id', how='left')
test = test.merge(songs, on='song_id', how='left')

members['membership_days'] = members['expiration_date'].subtract(members['registration_init_time']).dt.days.astype(int)

members['registration_year'] = members['registration_init_time'].dt.year
1867         if self.options.get('as_recarray'):

/opt/conda/lib/python3.6/site-packages/pandas/io/parsers.py in read(self, nrows)
1826     def read(self, nrows=None):
1827         try:
-> 1828             data = self._reader.read(nrows)
1829         except StopIteration:
1830             if self._first_chunk:

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader.read()

pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._read_low_memory()

pandas/_libs/parsers.pyx in pandas._libs.parsers._concatenate_chunks()

/opt/conda/lib/python3.6/site-packages/numpy/core/numerictypes.py in find_common_type(array_types, scalar_types)
1013
1014         """
-> 1015     array_types = [dtype(x) for x in array_types]
1016     scalar_types = [dtype(x) for x in scalar_types]
1017

/test/song_id/test/song_id.astype(<listcomp>[0])

# import gc
# del members, songs; gc.collect();

print('Done merging...')

Data merging...

-----
NameError                                 Traceback (most recent call last)
<ipython-input-3-1432efb892c8> in <module>()
      2
      3
-> 4     train = train.merge(songs, on='song_id', how='left')
      5     test = test.merge(songs, on='song_id', how='left')
      6

NameError: name 'train' is not defined
```

```
In [4]: print ("Adding new features")

return sum(map(x.count, ['|', '/', '\\', ';'])) + 1
return sum(map(x.count, ['|', '/', '\\', ';']))

train['lyricist'].fillna('no_lyricist',inplace=True)
test['lyricist'].fillna('no_lyricist',inplace=True)
train['lyricists_count'] = train['lyricist'].apply(lyricist_count).astype(np.int8)
test['lyricists_count'] = test['lyricist'].apply(lyricist_count).astype(np.int8)

def composer_count(x):
    if x == 'no_composer':
        return 0
    else:
        return sum(map(x.count, ['|', '/', '\\', ';'])) + 1

train['composer'].fillna('no_composer',inplace=True)
test['composer'].fillna('no_composer',inplace=True)
train['composer_count'] = train['composer'].apply(composer_count).astype(np.int8)
test['composer_count'] = test['composer'].apply(composer_count).astype(np.int8)

def is_featured(x):
    if 'feat' in str(x) :
        return 1
    return 0
```

```

test.song_id = test.song_id.astype('category')

# import gc
# del members, songs; gc.collect()

print('Done merging...')

Data merging...

-----
NameError                                 Traceback (most recent call last)
<ipython-input-3-1432efb892c8> in <module>()
      2
      3
----> 4 train = train.merge(songs, on='song_id', how='left')
      5 test = test.merge(songs, on= 'song_id', how='left')
      6

NameError: name 'train' is not defined

```

```

In [4]: print ("Adding new features")

TypeError                                 Traceback (most recent call last)
<ipython-input-2-6e0df5b8adc> in <module>()
      6
      7
----> 8     'source_type' : 'category',
      9     'target' : np.uint8,
     10     'song_id' : 'category'))
     11
     12     9 test = pd.read_csv(data_path + 'test.csv', dtype={'msno' : 'category',
     13                           'source_system_tab' : 'category',
     14                           'source_type' : 'category',
     15                           'target' : np.uint8,
     16                           'song_id' : 'category'})
     17
     18     10
     19     11
     20     12
     21     13
     22     14
     23     15
     24     16
     25     17
     26     18
     27     19
     28     20
     29     21
     30     22
     31     23
     32     24
     33     25
     34     26
     35     27
     36     28
     37     29
     38     30
     39     31
     40     32
     41     33
     42     34
     43     35
     44     36
     45     37
     46     38
     47     39
     48     39
     49     40
     50     41
     51     42
     52     43
     53     44
     54     45
     55     46
     56     47
     57     48
     58     49
     59     50
     60     51
     61     52
     62     53
     63     54
     64     55
     65     56
     66     57
     67     58
     68     59
     69     60
     70     61
     71     62
     72     63
     73     64
     74     65
     75     66
     76     67
     77     68
     78     69
     79     70
     80     71
     81     72
     82     73
     83     74
     84     75
     85     76
     86     77
     87     78
     88     79
     89     80
     90     81
     91     82
     92     83
     93     84
     94     85
     95     86
     96     87
     97     88
     98     89
     99     90
    100     91
    101     92
    102     93
    103     94
    104     95
    105     96
    106     97
    107     98
    108     99
    109     100
    110     101
    111     102
    112     103
    113     104
    114     105
    115     106
    116     107
    117     108
    118     109
    119     110
    120     111
    121     112
    122     113
    123     114
    124     115
    125     116
    126     117
    127     118
    128     119
    129     120
    130     121
    131     122
    132     123
    133     124
    134     125
    135     126
    136     127
    137     128
    138     129
    139     130
    140     131
    141     132
    142     133
    143     134
    144     135
    145     136
    146     137
    147     138
    148     139
    149     140
    150     141
    151     142
    152     143
    153     144
    154     145
    155     146
    156     147
    157     148
    158     149
    159     150
    160     151
    161     152
    162     153
    163     154
    164     155
    165     156
    166     157
    167     158
    168     159
    169     160
    170     161
    171     162
    172     163
    173     164
    174     165
    175     166
    176     167
    177     168
    178     169
    179     170
    180     171
    181     172
    182     173
    183     174
    184     175
    185     176
    186     177
    187     178
    188     179
    189     180
    190     181
    191     182
    192     183
    193     184
    194     185
    195     186
    196     187
    197     188
    198     189
    199     190
    200     191
    201     192
    202     193
    203     194
    204     195
    205     196
    206     197
    207     198
    208     199
    209     200
    210     201
    211     202
    212     203
    213     204
    214     205
    215     206
    216     207
    217     208
    218     209
    219     210
    220     211
    221     212
    222     213
    223     214
    224     215
    225     216
    226     217
    227     218
    228     219
    229     220
    230     221
    231     222
    232     223
    233     224
    234     225
    235     226
    236     227
    237     228
    238     229
    239     230
    240     231
    241     232
    242     233
    243     234
    244     235
    245     236
    246     237
    247     238
    248     239
    249     240
    250     241
    251     242
    252     243
    253     244
    254     245
    255     246
    256     247
    257     248
    258     249
    259     250
    260     251
    261     252
    262     253
    263     254
    264     255
    265     256
    266     257
    267     258
    268     259
    269     260
    270     261
    271     262
    272     263
    273     264
    274     265
    275     266
    276     267
    277     268
    278     269
    279     270
    280     271
    281     272
    282     273
    283     274
    284     275
    285     276
    286     277
    287     278
    288     279
    289     280
    290     281
    291     282
    292     283
    293     284
    294     285
    295     286
    296     287
    297     288
    298     289
    299     290
    300     291
    301     292
    302     293
    303     294
    304     295
    305     296
    306     297
    307     298
    308     299
    309     300
    310     301
    311     302
    312     303
    313     304
    314     305
    315     306
    316     307
    317     308
    318     309
    319     310
    320     311
    321     312
    322     313
    323     314
    324     315
    325     316
    326     317
    327     318
    328     319
    329     320
    330     321
    331     322
    332     323
    333     324
    334     325
    335     326
    336     327
    337     328
    338     329
    339     330
    340     331
    341     332
    342     333
    343     334
    344     335
    345     336
    346     337
    347     338
    348     339
    349     340
    350     341
    351     342
    352     343
    353     344
    354     345
    355     346
    356     347
    357     348
    358     349
    359     350
    360     351
    361     352
    362     353
    363     354
    364     355
    365     356
    366     357
    367     358
    368     359
    369     360
    370     361
    371     362
    372     363
    373     364
    374     365
    375     366
    376     367
    377     368
    378     369
    379     370
    380     371
    381     372
    382     373
    383     374
    384     375
    385     376
    386     377
    387     378
    388     379
    389     380
    390     381
    391     382
    392     383
    393     384
    394     385
    395     386
    396     387
    397     388
    398     389
    399     390
    400     391
    401     392
    402     393
    403     394
    404     395
    405     396
    406     397
    407     398
    408     399
    409     400
    410     401
    411     402
    412     403
    413     404
    414     405
    415     406
    416     407
    417     408
    418     409
    419     410
    420     411
    421     412
    422     413
    423     414
    424     415
    425     416
    426     417
    427     418
    428     419
    429     420
    430     421
    431     422
    432     423
    433     424
    434     425
    435     426
    436     427
    437     428
    438     429
    439     430
    440     431
    441     432
    442     433
    443     434
    444     435
    445     436
    446     437
    447     438
    448     439
    449     440
    450     441
    451     442
    452     443
    453     444
    454     445
    455     446
    456     447
    457     448
    458     449
    459     450
    460     451
    461     452
    462     453
    463     454
    464     455
    465     456
    466     457
    467     458
    468     459
    469     460
    470     461
    471     462
    472     463
    473     464
    474     465
    475     466
    476     467
    477     468
    478     469
    479     470
    480     471
    481     472
    482     473
    483     474
    484     475
    485     476
    486     477
    487     478
    488     479
    489     480
    490     481
    491     482
    492     483
    493     484
    494     485
    495     486
    496     487
    497     488
    498     489
    499     490
    500     491
    501     492
    502     493
    503     494
    504     495
    505     496
    506     497
    507     498
    508     499
    509     500
    510     501
    511     502
    512     503
    513     504
    514     505
    515     506
    516     507
    517     508
    518     509
    519     510
    520     511
    521     512
    522     513
    523     514
    524     515
    525     516
    526     517
    527     518
    528     519
    529     520
    530     521
    531     522
    532     523
    533     524
    534     525
    535     526
    536     527
    537     528
    538     529
    539     530
    540     531
    541     532
    542     533
    543     534
    544     535
    545     536
    546     537
    547     538
    548     539
    549     540
    550     541
    551     542
    552     543
    553     544
    554     545
    555     546
    556     547
    557     548
    558     549
    559     550
    560     551
    561     552
    562     553
    563     554
    564     555
    565     556
    566     557
    567     558
    568     559
    569     560
    570     561
    571     562
    572     563
    573     564
    574     565
    575     566
    576     567
    577     568
    578     569
    579     570
    580     571
    581     572
    582     573
    583     574
    584     575
    585     576
    586     577
    587     578
    588     579
    589     580
    590     581
    591     582
    592     583
    593     584
    594     585
    595     586
    596     587
    597     588
    598     589
    599     590
    600     591
    601     592
    602     593
    603     594
    604     595
    605     596
    606     597
    607     598
    608     599
    609     600
    610     601
    611     602
    612     603
    613     604
    614     605
    615     606
    616     607
    617     608
    618     609
    619     610
    620     611
    621     612
    622     613
    623     614
    624     615
    625     616
    626     617
    627     618
    628     619
    629     620
    630     621
    631     622
    632     623
    633     624
    634     625
    635     626
    636     627
    637     628
    638     629
    639     630
    640     631
    641     632
    642     633
    643     634
    644     635
    645     636
    646     637
    647     638
    648     639
    649     640
    650     641
    651     642
    652     643
    653     644
    654     645
    655     646
    656     647
    657     648
    658     649
    659     650
    660     651
    661     652
    662     653
    663     654
    664     655
    665     656
    666     657
    667     658
    668     659
    669     660
    670     661
    671     662
    672     663
    673     664
    674     665
    675     666
    676     667
    677     668
    678     669
    679     670
    680     671
    681     672
    682     673
    683     674
    684     675
    685     676
    686     677
    687     678
    688     679
    689     680
    690     681
    691     682
    692     683
    693     684
    694     685
    695     686
    696     687
    697     688
    698     689
    699     690
    700     691
    701     692
    702     693
    703     694
    704     695
    705     696
    706     697
    707     698
    708     699
    709     700
    710     701
    711     702
    712     703
    713     704
    714     705
    715     706
    716     707
    717     708
    718     709
    719     710
    720     711
    721     712
    722     713
    723     714
    724     715
    725     716
    726     717
    727     718
    728     719
    729     720
    730     721
    731     722
    732     723
    733     724
    734     725
    735     726
    736     727
    737     728
    738     729
    739     730
    740     731
    741     732
    742     733
    743     734
    744     735
    745     736
    746     737
    747     738
    748     739
    749     740
    750     741
    751     742
    752     743
    753     744
    754     745
    755     746
    756     747
    757     748
    758     749
    759     750
    760     751
    761     752
    762     753
    763     754
    764     755
    765     756
    766     757
    767     758
    768     759
    769     760
    770     761
    771     762
    772     763
    773     764
    774     765
    775     766
    776     767
    777     768
    778     769
    779     770
    780     771
    781     772
    782     773
    783     774
    784     775
    785     776
    786     777
    787     778
    788     779
    789     780
    790     781
    791     782
    792     783
    793     784
    794     785
    795     786
    796     787
    797     788
    798     789
    799     790
    800     791
    801     792
    802     793
    803     794
    804     795
    805     796
    806     797
    807     798
    808     799
    809     800
    810     801
    811     802
    812     803
    813     804
    814     805
    815     806
    816     807
    817     808
    818     809
    819     810
    820     811
    821     812
    822     813
    823     814
    824     815
    825     816
    826     817
    827     818
    828     819
    829     820
    830     821
    831     822
    832     823
    833     824
    834     825
    835     826
    836     827
    837     828
    838     829
    839     830
    840     831
    841     832
    842     833
    843     834
    844     835
    845     836
    846     837
    847     838
    848     839
    849     840
    850     841
    851     842
    852     843
    853     844
    854     845
    855     846
    856     847
    857     848
    858     849
    859     850
    860     851
    861     852
    862     853
    863     854
    864     855
    865     856
    866     857
    867     858
    868     859
    869     860
    870     861
    871     862
    872     863
    873     864
    874     865
    875     866
    876     867
    877     868
    878     869
    879     870
    880     871
    881     872
    882     873
    883     874
    884     875
    885     876
    886     877
    887     878
    888     879
    889     880
    890     881
    891     882
    892     883
    893     884
    894     885
    895     886
    896     887
    897     888
    898     889
    899     890
    900     891
    901     892
    902     893
    903     894
    904     895
    905     896
    906     897
    907     898
    908     899
    909     900
    910     901
    911     902
    912     903
    913     904
    914     905
    915     906
    916     907
    917     908
    918     909
    919     910
    920     911
    921     912
    922     913
    923     914
    924     915
    925     916
    926     917
    927     918
    928     919
    929     920
    930     921
    931     922
    932     923
    933     924
    934     925
    935     926
    936     927
    937     928
    938     929
    939     930
    940     931
    941     932
    942     933
    943     934
    944     935
    945     936
    946     937
    947     938
    948     939
    949     940
    950     941
    951     942
    952     943
    953     944
    954     945
    955     946
    956     947
    957     948
    958     949
    959     950
    960     951
    961     952
    962     953
    963     954
    964     955
    965     956
    966     957
    967     958
    968     959
    969     960
    970     961
    971     962
    972     963
    973     964
    974     965
    975     966
    976     967
    977     968
    978     969
    979     970
    980     971
    981     972
    982     973
    983     974
    984     975
    985     976
    986     977
    987     978
    988     979
    989     980
    990     981
    991     982
    992     983
    993     984
    994     985
    995     986
    996     987
    997     988
    998     989
    999     990
    1000     991
    1001     992
    1002     993
    1003     994
    1004     995
    1005     996
    1006     997
    1007     998
    1008     999
    1009     1000
    1010     1001
    1011     1002
    1012     1003
    1013     1004
    1014     1005
    1015     1006
    1016     1007
    1017     1008
    1018     1009
    1019     1010
    1020     1011
    1021     1012
    1022     1013
    1023     1014
    1024     1015
    1025     1016
    1026     1017
    1027     1018
    1028     1019
    1029     10
```

```
In [5]:
print ("Train test and validation sets")
for col in train.columns:
    if train[col].dtype == object:
        train[col] = train[col].astype('category')
        test[col] = test[col].astype('category')

X_train = train.drop(['target'], axis=1)
y_train = train['target'].values

X_test = test.drop(['id'], axis=1)
ids = test['id'].values

# del train, test; gc.collect();

d_train_final = lgb.Dataset(X_train, y_train)
watchlist_final = lgb.Dataset(X_train, y_train)
print('Processed data...')

Train test and validation sets

-----
NameError                                 Traceback (most recent call last)
<ipython-input-5-6f773e45101b> in <module>()
      1 print ("Train test and validation sets")
----> 2 for col in train.columns:
      3     if train[col].dtype == object:
      4         train[col] = train[col].astype('category')
      5         test[col] = test[col].astype('category')

NameError: name 'train' is not defined
```

```
In [6]:
params = {
    'objective': 'binary',
    'metric': 'binary_logloss',
    'boosting': 'gbdt',
    'learning_rate': 0.3 ,
    'verbose': 0,
    'num_leaves': 108,
    'bagging_fraction': 0.95,
    'bagging_freq': 1,
    'bagging_seed': 1,
    'feature_fraction': 0.9,
    'feature_fraction_seed': 1,
    'max_bin': 256,
    'max_depth': 10,
    'num_rounds': 200,
    'metric' : 'auc'
}

%time model_f1 = lgb.train(params, train_set=d_train_final, valid_sets=watchlist_final, verbose_eval=5)

-----
NameError                                 Traceback (most recent call last)
<ipython-input-6-f357cb29e712> in <module>()
    17
    18
--> 19 get_ipython().magic('time model_f1 = lgb.train(params, train_set=d_train_final, valid_
sets=watchlist_final, verbose_eval=5)'

/opt/conda/lib/python3.6/site-packages/IPython/core/interactiveshell.py in magic(self, args)
2156         magic_name, _, magic_arg_s = arg_s.partition(' ')
2157         magic_name = magic_name.lstrip(prefilter.ESC_MAGIC)
-> 2158         return self.run_line_magic(magic_name, magic_arg_s)
2159
2160 #-----
```

/opt/conda/lib/python3.6/site-packages/IPython/core/interactiveshell.py in run_line_magic(self, magic_name, line)
2077 kwargs['local_ns'] = sys._getframe(stack_depth).f_locals
2078 with self.builtin_trap:
-> 2079 result = fn(*args,**kwargs)
2080
2081 return result

```

<decorator-gen-59> in time(self, line, cell, local_ns)

/opt/conda/lib/python3.6/site-packages/IPython/core/magic.py in <lambda>(f, *a, **k)
    186     # but it's overkill for just that one bit of state.
    187     def magic_deco(arg):
--> 188         call = lambda f, *a, **k: f(*a, **k)
    189
    190     if callable(arg):

/opt/conda/lib/python3.6/site-packages/IPython/core/magics/execution.py in time(self, line, cel
1, local_ns)
    1183         else:
    1184             st = clock2()
-> 1185             exec(code, glob, local_ns)
    1186             end = clock2()
    1187             out = None

<timed exec> in <module>()

NameError: name 'd_train_final' is not defined

```

In [7]:

```

params = {
    'objective': 'binary',
    'metric': 'binary_logloss',
    'boosting': 'dart',
    'learning_rate': 0.3 ,
    'verbose': 0,
    'num_leaves': 108,
    'bagging_fraction': 0.95,
    'bagging_freq': 1,
    'bagging_seed': 1,
    'feature_fraction': 0.9,
    'feature_fraction_seed': 1,
    'max_bin': 256,
    'max_depth': 10,
    'num_rounds': 200,
    'metric' : 'auc'
}

%time model_f2 = lgb.train(params, train_set=d_train_final, valid_sets=watchlist_final, verbos
e_eval=5)

```

```

-----
NameError                                 Traceback (most recent call last)
<ipython-input-7-063832c731db> in <module>()
    17
    18
--> 19 get_ipython().magic('time model_f2 = lgb.train(params, train_set=d_train_final, valid_
sets=watchlist_final, verbose_eval=5')

/opt/conda/lib/python3.6/site-packages/IPython/core/interactiveshell.py in magic(self, arg_s)
    2156         magic_name, _, magic_arg_s = arg_s.partition(' ')
    2157         magic_name = magic_name.lstrip(prefilter.ESC_MAGIC)
--> 2158         return self.run_line_magic(magic_name, magic_arg_s)
    2159
    2160     #-----

/opt/conda/lib/python3.6/site-packages/IPython/core/interactiveshell.py in run_line_magic(self,
magic_name, line)
    2077             kwargs['local_ns'] = sys._getframe(stack_depth).f_locals
    2078             with self.builtin_trap:
--> 2079                 result = fn(*args,**kwargs)
    2080             return result
    2081

<decorator-gen-59> in time(self, line, cell, local_ns)

/opt/conda/lib/python3.6/site-packages/IPython/core/magic.py in <lambda>(f, *a, **k)
    186     # but it's overkill for just that one bit of state.
    187     def magic_deco(arg):
--> 188         call = lambda f, *a, **k: f(*a, **k)
    189
    190     if callable(arg):

/opt/conda/lib/python3.6/site-packages/IPython/core/magics/execution.py in time(self, line, cel
1, local_ns)
    1183         else:
    1184             st = clock2()
-> 1185             exec(code, glob, local_ns)

```

```

1186         end = clock2()
1187         out = None

<timed exec> in <module>()

NameError: name 'd_train_final' is not defined

```

```

In [8]:
print('Making predictions')
p_test_1 = model_f1.predict(X_test)
p_test_2 = model_f2.predict(X_test)
p_test_avg = np.mean([p_test_1, p_test_2], axis = 0)

```

```
print('Done making predictions')
```

```
Making predictions
```

```

-----  

NameError                                                 Traceback (most recent call last)  

<ipython-input-8-dc6069237382> in <module>()  

    1 print('Making predictions')  

----> 2 p_test_1 = model_f1.predict(X_test)  

    3 p_test_2 = model_f2.predict(X_test)  

    4 p_test_avg = np.mean([p_test_1, p_test_2], axis = 0)  

    5  

NameError: name 'model_f1' is not defined

```

```
In [9]:
print ('Saving predictions Model model of gbdt')
```

```

subm = pd.DataFrame()
subm['id'] = ids
subm['target'] = p_test_avg
subm.to_csv(data_path + 'submission_lgbm_avg.csv.gz', compression = 'gzip', index=False, float_
format = '%.5f')

print('Done!')

```

```
Saving predictions Model model of gbdt
```

```

-----  

NameError                                                 Traceback (most recent call last)  

<ipython-input-9-2366982ed7da> in <module>()  

    2  

    3 subm = pd.DataFrame()  

----> 4 subm['id'] = ids  

    5 subm['target'] = p_test_avg  

    6 subm.to_csv(data_path + 'submission_lgbm_avg.csv.gz', compression = 'gzip', index=False
, float_format = '%.5f')

NameError: name 'ids' is not defined

```

Data

Data Sources

- WSDM - KKBox's Music Recommendation Challenge
 - members.csv
 - sample_submission.csv
 - song_extra_info.csv
 - songs.csv
 - test.csv
 - train.csv

WSDM - KKBox's Music Recommendation Challenge



Can you build the best music recommendation system?
Last Updated: a year ago

About this Competition

In this task, you will be asked to predict the chances of a user listening to a song repetitively after the first observable listening event within a time window was triggered. If there are recurring listening event(s) triggered within a month after the user's very first observable listening event, its target is marked 1, and 0 otherwise in the training set. The same rule applies to the testing set.

KKBOX provides a training data set consists of information of the first observable listening event for each unique user-song pair within a specific time duration. Metadata of each unique user and song pair is also provided. The use of public data to increase the level of accuracy of your prediction is encouraged.

The train and the test data are selected from users listening history in a given time period. Note that this time period is chosen to be before the [WSDM-KKBox Churn Prediction](#) time period. The train and test sets are split based on time, and the split of public/private are based on unique user/song pairs.

Tables

train.csv

- msno: user id
- song_id: song id
- source_system_tab: the name of the tab where the event was triggered. System tabs are used to categorize KKBOX mobile apps functions. For example, tab `my library` contains functions to manipulate the local storage, and tab `search` contains functions relating to search.
- source_screen_name: name of the layout a user sees.
- source_type: an entry point a user first plays music on mobile apps. An entry point could be `album`, `online-playlist`, `song` .. etc.
- target: this is the target variable. `target=1` means there are recurring listening.

Run Info

Succeeded	True	Run Time	3614.1 seconds
Exit Code	0	Queue Time	0 seconds
Docker Image Name	kaggle/python(Dockerfile)	Output Size	0
Timeout Exceeded	False	Used All Space	False
Failure Message			

Log

[Download Log](#)

```
Time Line # Log Message
1 [
2   "data": "[NbConvertApp] Converting notebook __temp_notebook_source__.ipynb to html\n",
3   "stream_name": "stderr",
4   "time": 1.5475757209933363
5 ],
6   "data": "[NbConvertApp] Writing 308692 bytes to __results__.html\n",
7   "stream_name": "stderr",
8   "time": 1.7093351599760354
9 ],
10  "data": "[NbConvertApp] Converting notebook __temp_notebook_source__.ipynb to notebook\n",
11  "stream_name": "stderr",
12  "time": 1.54721229977347
13 ],
14  "data": "[NbConvertApp] Executing notebook with kernel: python3\n",
15  "stream_name": "stderr",
16  "time": 1.5799874399672262
17 ],
18  "data": "Fontconfig warning: ignoring C.UTF-8: not a valid language tag\n",
19  "stream_name": "stderr",
20  "time": 2.548812124005053
21 ],
22  "data": "[NbConvertApp] Writing 45552 bytes to __notebook__.ipynb\n",
23  "stream_name": "stderr",
24  "time": 28.307915177952964
25 ],
26  "data": "[NbConvertApp] Converting notebook __notebook__.ipynb to html\n",
27  "stream_name": "stderr",
28  "time": 1.64721109397942
29 ],
30  "data": "[NbConvertApp] Writing 341770 bytes to __results__.html\n",
31  "stream_name": "stderr",
32  "time": 1.845204734010622
33 }
34
35 Complete. Exited with code 0.
```

Comments (49)

Sort by
All Comments ▾ Hotness ▾



Click here to enter a comment...



Ferz • Posted on Latest Version • a year ago • Options • Reply

^ 2 ▾

And one more question. Why did you use in the first lgbm 'gbdt' and in the second one 'dart'?



kmaster • Posted on Latest Version • a year ago • Options • Reply

^ 2 ▾

Thank you for sharing.
When I trv to run:

```
train['artistcomposer'] = (train['artistname'] == train['composer']).astype(np.int8)
```

I got this error:

TypeError: Categoricals can only be compared if 'categories' are the same. Categories are different lengths

However, len(train['artist_name']) == len(train['composer']) is true.

Can you please show me some directions for this?

Jingxi Xu • Posted on Latest Version • a year ago • Options • Reply

^ 1 ▼

Why is everyone saying that it is beautiful while it actually does not work.

蜗牛爱上星星 • Posted on Latest Version • a year ago • Options • Reply

^ 1 ▼

It works.

蜗牛爱上星星 • Posted on Latest Version • a year ago • Options • Reply

^ 0 ▼

python 2.7, pandas 0.19

Randolph • Posted on Latest Version • a year ago • Options • Reply

^ 0 ▼

Thank you. Is there any other EDA work can improve the model and make the prediction result better? I mean a real EDA work because some Feature work we do is not helpful.

Since i analysis the data and make some feature work which i think maybe helpful but it harms the model performance in actual. So anyone can share the useful EDA work?

Alexander Kireev • Posted on Version 2 • a year ago • Options • Reply

^ 1 ▼

Thank you. Beautiful work.

Max Yue • Posted on Version 2 • a year ago • Options • Reply

^ 0 ▼

Beautiful!!

Max Yue • Posted on Version 2 • a year ago • Options • Reply

^ 0 ▼

Great feature extraction! When someone doing feature extraction, feature analysis will normally follow, so we can see which features contribute the model.

asmita vikas Kernel Author • Posted on Version 2 • a year ago • Options • Reply

^ 1 ▼

noted! I'll add that next, thank you

Maple • Posted on Version 2 • a year ago • Options • Reply

^ 0 ▼

great work.

Maple • Posted on Version 2 • a year ago • Options • Reply

^ 0 ▼

why train['countsongplayed'] and train['countsongplayed'] have some negative values.

asmita vikas Kernel Author • Posted on Version 2 • a year ago • Options • Reply

^ 1 ▼

oops! that's coz I was making the column as np.int8 but the values were of larger.
Fixed it to int64

Maple • Posted on Latest Version • a year ago • Options • Reply

^ 0 ▼

I want to know why song language 17 or 45 are set to 1 and the others are set to 0.

SuperFrankie • Posted on Version 2 • a year ago • Options • Reply

^ 0 ▼

Great job.

 zhihuidayong · Posted on Version 2 · a year ago · Options · Reply

^ 0 ^

I only got 0.64 score with the same code



asmita vikas **Kernel Author** · Posted on Version 4 · a year ago · Options · Reply

^ 2 ^



hmm, that sounds weird. I did make some changes now and got 0.68
Maybe you could try again?



lystdo · Posted on Version 4 · a year ago · Options · Reply

^ 2 ^



You may check the version of python.



zhihuidayong · Posted on Latest Version · a year ago · Options · Reply

^ 0 ^



Yeah, I use version 2.6, is there something with that?



zhihuidayong · Posted on Latest Version · a year ago · Options · Reply

^ 0 ^



I got 0.66 score this time, why are we different so much



lystdo · Posted on Latest Version · a year ago · Options · Reply

^ 0 ^



Shubham Gupta · Posted on Latest Version · a year ago · Options · Reply

^ 0 ^



I am getting a low score of 0.62, although I am using pandas=0.19 and python 2.7, and secondly, what about the remaining null values in other features, do we replace them with a constant value or leave them as it is?? can you please share the version of lightgbm and sklearn please.



DawuChen · Posted on Latest Version · a year ago · Options · Reply

^ 0 ^

Hey! Thx for your nice work. But can I ask how you solve the problem of "TypeError: data type not understood" when read_csv use category dtype? I also encountered the same problem. Please help.



asmita vikas **Kernel Author** · Posted on Latest Version · a year ago · Options · Reply

^ 1 ^



Well, I had written the script for Python2.7 and it ran fine.
I'm guessing its a problem with Python3. I'll try to fix it asap. Meanwhile could you try it on another version?



DawuChen · Posted on Latest Version · a year ago · Options · Reply

^ 0 ^



I am also using python 2.7 but got the same error. I had tried update pandas to the newest version but it still can't work. By the way, I found sometimes dtype='category' can work on numeric type but always not on object type, it's weird...



DawuChen · Posted on Latest Version · a year ago · Options · Reply

^ 1 ^



Got the reason, it is because the version of pandas is too new, there is bug in version 0.21.0
see:
[\[https://www.kaggle.com/c/kkbox-music-recommendation-challenge/discussion/43352\]](https://www.kaggle.com/c/kkbox-music-recommendation-challenge/discussion/43352)[1]



Fuzzylogic · Posted on Latest Version · a year ago · Options · Reply

^ 0 ^



Same for me.
Any workaround ..? I am also looking



DawuChen · Posted on Latest Version · a year ago · Options · Reply

^ 1 ^



As far, there is no workaround to use 'category' when read csv. But you can use 'astype' function to change dtype after you read data into memory.



Vinay Patolla · Posted on Latest Version · a year ago · Options · Reply

^ 0 ^

downgrade pandas to 0.19

Fuzzylogic • Posted on Latest Version • a year ago • Options • Reply

great work..thanks a ton for sharing !

Jingxi Xu • Posted on Latest Version • a year ago • Options • Reply

train['artistcomposer'] = (train['artistname'] == train['composer']).astype(np.int8) does not work with the error that "Categoricals can only be compared if 'categories' are the same. Categories are different lengths". Can I have any hints on that?

Layla • Posted on Latest Version • a year ago • Options • Reply

I got the same error. Did you figure it out?

asmita vikas Kernel Author • Posted on Latest Version • a year ago • Options • Reply

it could be an issue with pandas version. Can you try downgrading pandas to 0.19?

BrianHur • Posted on Latest Version • a year ago • Options • Reply

Thanks! I had the same error, but downgrading to pandas 0.19 fixed it for me

WendyCui • Posted on Latest Version • a year ago • Options • Reply

Hi, thank you for sharing!! But there's a part of your code confusing me. Why you just considered two values(17, 45) of 'language' attribute in songs.csv? Could you please give me some hints on this?

asmita vikas Kernel Author • Posted on Latest Version • a year ago • Options • Reply

Well, if you look at the distributions, song_lang with values 17 and 45 have the highest percentage of 1s in response variable. So I took them as high priority

a year ago

This Comment was deleted.

asmita vikas Kernel Author • Posted on Latest Version • a year ago • Options • Reply

Response variable is the variable we try to predict, in this case "target". It is also called the dependent variable.

a year ago

This Comment was deleted.

a year ago

This Comment was deleted.

Felicia • Posted on Latest Version • a year ago • Options • Reply

TypeError: data type not understood

I got the same error.

How did you fix that? Category dtype is not recognized in read_csv

deathstar • Posted on Latest Version • a year ago • Options • Reply

Great work on feature engineering.

++

cowarder • Posted on Latest Version • a year ago • Options • Reply

Thank you for your great work! But I got some trouble while running the code "train['genreids'].fillna('nogenre_id',inplace=True)". The error message is "ValueError: fill value must be in categories", how can I solve this?



Yihong Wang • Posted on Latest Version • a year ago • Options • Reply

^ 1 ▼

```
train['genreids'].cat.addcategories("nogenreid").fillna("nogenreid",inplace=True)
```



Ferz • Posted on Latest Version • a year ago • Options • Reply

^ 0 ▼

Hi, thank you for sharing. Nice work. I am a beginner. Can you tell me why did you use 2 different lgbm? Did you find out that their average is better than single one?



LongYin • Posted on Latest Version • 4 months ago • Options • Reply

^ 0 ▼

This is just for ensemble.



zouyu • Posted on Latest Version • a year ago • Options • Reply

^ 0 ▼

good jobs, i will use some time to research your code !thank you very much!



KimchiSoup • Posted on Latest Version • a year ago • Options • Reply

^ 0 ▼

it was helpful to me. +_+d



rbaral • Posted on Latest Version • 9 months ago • Options • Reply

^ 0 ▼

How did you come with the parameters for the LGB models?