Mori Levinzon 308328467

Nadav Rubinstein 208686659

Mandatory Part – Modeling:

Loaded the ElectionsDava.csv and we applied the manipulations to the pre-processed data as we did in the previous exercise. These manipulations include:

1. Split the total sample set (Using Stratified Shuffle Split) to 3 sets:

Data Set	Percentage from Original Data Set
Train set	65%
Validating set	10%
Test set	25%

2. Outliers values disposal such as negative values
3. Complete missing values using the usual methods: closest fit, feature correlation, mean and majority.
4. Selecting the best set of features as chosen in this exercise
5. Normalize the categorial values and Z-scale the nominal values
6. Extract the data to 3*2 csv files (before + after the change)

Afterwards, we approached the prediction assignment, when in the mandatory part we were required to predict:

- Who is the winning party?
- Distribution of votes among voters
- Supply shuttle services to voters of each party

In order to deal with these missions we would like to find the best classifier among a group of classifiers, So we performed the following process:

1. Choosing the types of classifiers which we want to make the predictions:
    a. Random Forest Classifier – A committee algorithm. The classification is determined by the majority votes of the decision tree participating in the built forest. We choose it since it is known as a strong algorithm , which includes the decision tree principle and because decision trees are a family of algorithms learned in the course.

b. Stochastic Gradient Descent Classifier – We choose this model because we knew it's operate method and because we wanted to know if there is a linear separation to the problem.
c. K- Nearest Neighbors – We chose this model because we knew it and it method. In addition, assuming that people with the same characteristics tend to choose the same, it is worth checking the performance of the information.
d. Decision Tree- Regular decision tree as taught in the lecture. We chose this model even though we used trees forest because we wanted to see the performance difference between the Random Forest Classifier and this classifier and we also managed to draw the resulting decision tree and understand the splits from him (A file named decision_tree.dot is added to the submission).

2. Choosing the best parameters for each selected classifier type. For each of the types of classifiers mentioned above we built a few examples of it with sets of different hyper-parameters and evaluated its performance by the accuracy measure:

a.
```
random_forest_tuple = (

    RandomForestClassifier(random_state=0, criterion='entropy', min_samples_split=5,
    min_samples_leaf=3, n_estimators=50),

    RandomForestClassifier(random_state=0, criterion='entropy', min_samples_split=3,
    min_samples_leaf=1, n_estimators=500),

    RandomForestClassifier(random_state=0, criterion='gini', min_samples_split=3,
    min_samples_leaf=1, n_estimators=500)

)
```

b.
```
sgd_tuple = (

SGDClassifier(random_state=0, max_iter=1000, tol=1e-3),

SGDClassifier(random_state=0, max_iter=1000, tol=1e-2),

SGDClassifier(random_state=0, max_iter=1500, tol=1e-4),
)
```

c.
```
knn_tuple = (

    KNeighborsClassifier(n_neighbors=3, algorithm='auto'),

    KNeighborsClassifier(n_neighbors=5, algorithm='auto'),
)
```

d.
```
tree_tuple = (
DecisionTreeClassifier(random_state=0, criterion='gini', min_samples_split=5,
   min_samples_leaf=3),

DecisionTreeClassifier(random_state=0, criterion='entropy', min_samples_split=3,
   min_samples_leaf=1)
)
```

Each model was evaluated using the k-fold cross validation method with the training set (set k = 5).

3. After selecting the best parameter set for each classifier, the best classifier type was selected using the classifier performance test over the validation set. We will emphasize that for the classifier performance is the percentage of accuracy across the test set. We chose precision as a benchmark for performance evaluation because in order to predict the winning party and the distribution of votes between the parties, we need high accuracy to provide consistent predictions with the set of examples.

   The results we received:

   Random Forest Classifier accuracy score on validation set is: 91.6 %

   SGD Classifier accuracy score on validation set is: 69.7 %

   KNN Classifier accuracy score on validation set is: 82.4 %

   Decision Tree Classifier accuracy score on validation set is: 87.1 %

So the selected classifier is Random Forest Classifier, with the following set of parameters:

```
andomForestClassifier(random_state=0, criterion='gini;, n_samples_split=3,
min_samples_leaf=1, n_estimators=500)
```

4. The predictions were made by the selected classifier when they were re-trained on the training set and the validation set together. The reason to do so is because we have already used the validation set to choose which classifier to use and we do not intend to use it again and of course the validation set contains samples that we would like to run trough our classifier. In general the larger the training set the better the classifier.

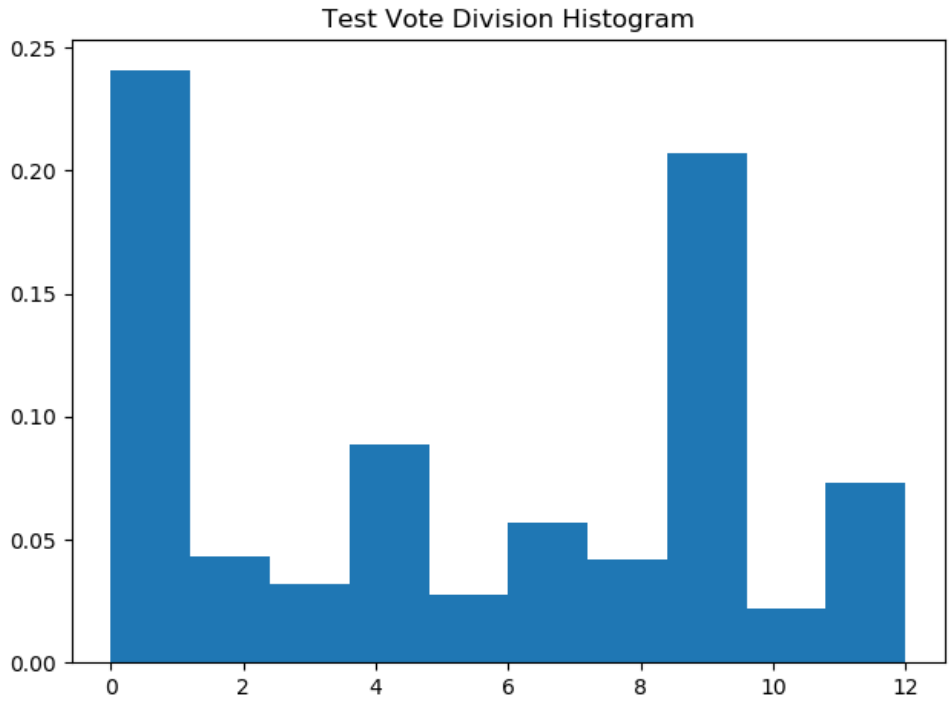
The following classifier performance on the test set received an accuracy of 91.88% and an error of 8.12%.

As it may be noticed, we got higher accuracy on the test set than the k-fold cross validation performance test and also more than the performance test on the validation set, which indicates that the classifier does not suffer from overfitting on its training set.

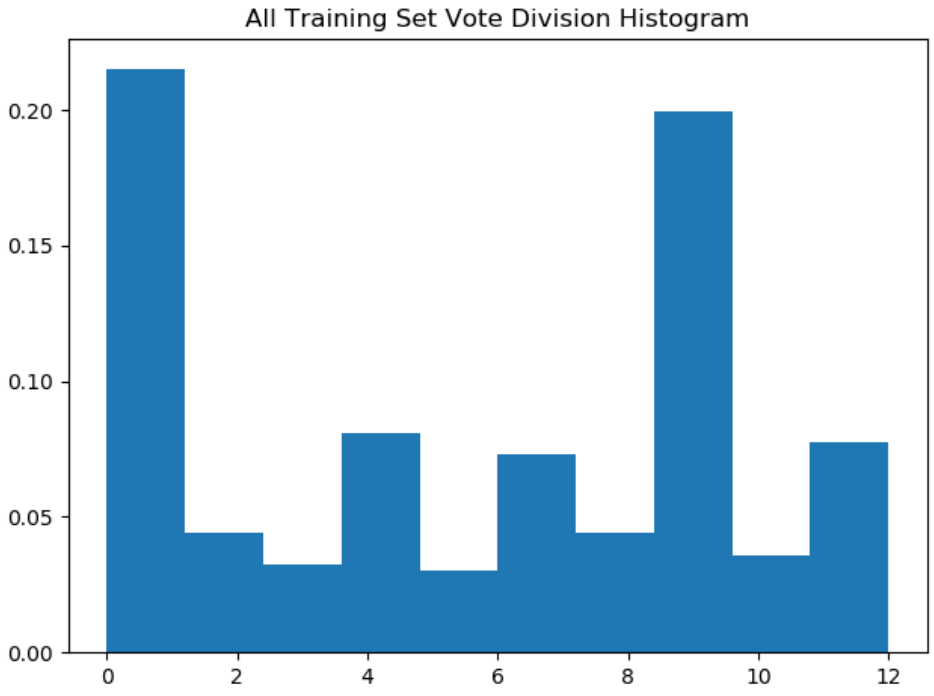
5. We'll explain how we worked for each of the predictions we were asked to make:
    a. For the winning party prediction, we made a prediction on the test set and chose the party that received the most votes, and according to our classifier the winning party is: Turquoises.
    b. For the distribution of votes between the parties, we predicted the test set and built a histogram according to the parties:

Color	Vote Percentage
Blues	5.68%
Browns	**23.24%**
Greens	5.2%
Greys	3.84%
Khakis	10.64%
Oranges	3.28%
Pinks	2.80%
Purples	4.04%
Reds	5.04%
Turquoises	**24.84%**
Violets	2.64%
Whites	4.04%
Yellows	4.72%

Distribution of votes as predicted by the classified:



Distribution of votes voting by training set:

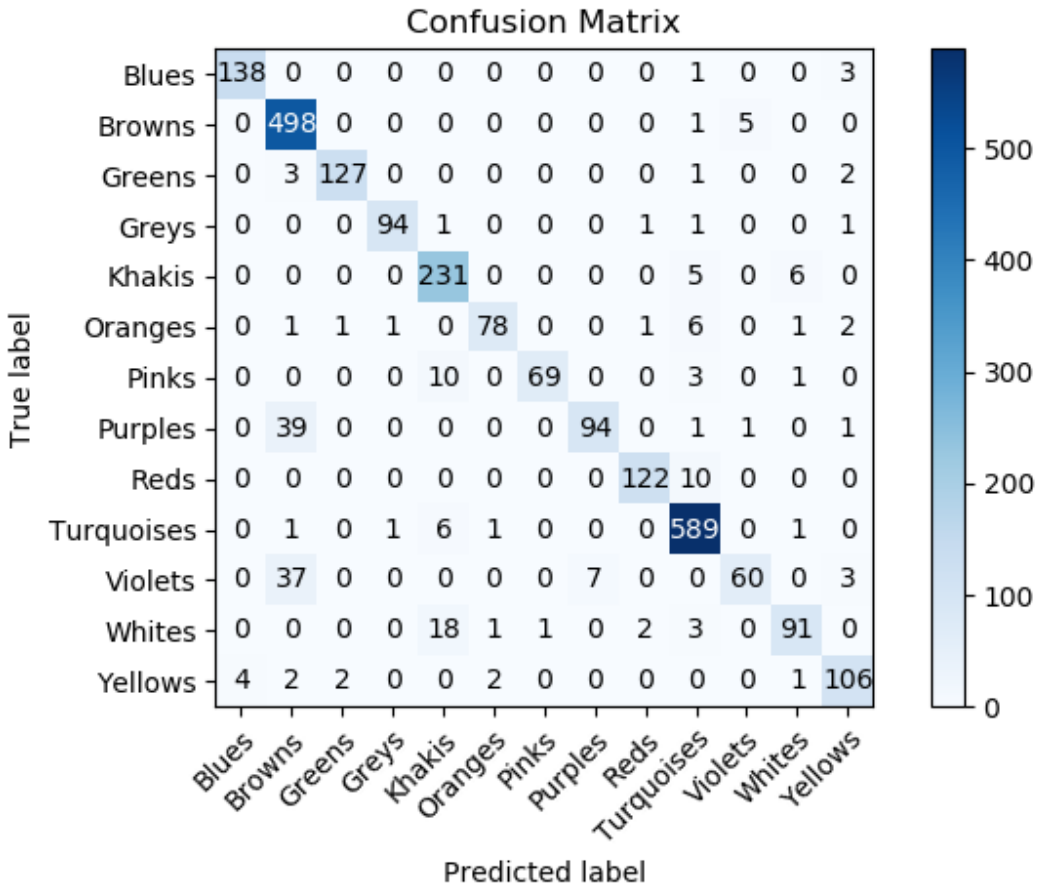


c. For the voting shuttle service, we predicted on the test set when instead of predicting a single classification for each voter, we predicted what the probability would be for each party. Then we set a threshold for the shuttle service which states that given a voter and his voting chance for a particular party whether the party should provide him with a shuttle service. We set the threshold for 60%. Here is the shuttle service for each party:

- **Browns**: [1, 3, 10, 21, 22, 32, 35, 36, 41, 43, 49, 51, 53, 56, 62, 80, 83, 95, 98, 104, 106, 120, 122, 124, 127, 129, 133, 134, 154, 170, 176, 188, 200, 203, 205, 215, 217, 218, 222, 224, 225, 231, 235, 240, 243, 244, 246, 249, 251, 255, 257, 267, 269, 270, 275, 279, 281, 289, 290, 307, 322, 329, 341, 344, 345, 352, 355, 367, 370, 374, 397, 420, 425, 430, 433, 441, 447, 452, 461, 471, 474, 489, 495, 497, 499, 501, 506, 510, 513, 516, 519, 528, 545, 553, 555, 559, 560, 567, 570, 576, 579, 585, 586, 587, 589, 590, 593, 605, 606, 621, 623, 630, 632, 633, 637, 648, 651, 652, 659, 666, 677, 682, 683, 696, 702, 704, 719, 723, 735, 750, 763, 783, 784, 787, 788, 792, 793, 800, 801, 817, 824, 838, 845, 856, 869, 871, 875, 893, 896, 903, 908, 922, 923, 930, 932, 937, 940, 942, 944, 950, 963, 965, 968, 971, 972, 984, 988, 992, 995, 1016, 1021, 1024, 1026, 1032, 1034, 1042, 1051, 1055, 1058, 1064, 1075, 1087, 1089, 1097, 1100, 1112, 1114, 1117, 1118, 1121, 1123, 1126, 1127, 1129, 1138, 1139, 1154, 1155, 1156, 1157, 1162, 1164, 1175, 1178, 1185, 1198, 1201, 1202, 1204, 1206, 1207, 1209, 1216, 1219, 1221, 1230, 1231, 1233, 1234, 1241, 1243, 1244, 1247, 1266, 1269, 1270, 1276, 1284, 1292, 1300, 1304, 1309, 1323, 1324, 1327, 1337, 1355, 1357, 1358, 1367, 1368, 1369, 1370, 1379, 1384, 1389, 1390, 1405, 1406, 1408, 1410, 1411, 1413, 1414, 1417, 1422, 1429, 1459, 1467, 1470, 1479, 1480, 1484, 1497, 1510, 1511, 1514, 1521, 1532, 1533, 1540, 1545, 1547, 1576, 1577, 1579, 1585, 1588, 1592, 1610, 1614, 1615, 1637, 1643, 1658, 1673, 1675, 1677, 1685, 1686, 1687, 1692, 1722, 1723, 1729, 1736, 1740, 1744, 1745, 1748, 1751, 1752, 1771, 1785, 1788, 1795, 1796, 1805, 1807, 1808, 1817, 1818, 1819, 1830, 1834, 1839, 1843, 1845, 1850, 1852, 1854, 1858, 1860, 1862, 1865, 1867, 1876, 1883, 1886, 1891, 1893, 1894, 1898, 1899, 1901, 1906, 1917, 1932, 1938, 1943, 1947, 1971, 1977, 1984, 1987, 1992, 1993, 1995, 2007, 2010, 2012, 2031, 2040, 2043, 2047, 2065, 2076, 2077, 2082, 2083, 2084, 2091, 2100, 2104, 2115, 2116, 2121, 2130, 2132, 2139, 2152, 2157, 2158, 2159, 2165, 2170, 2173, 2178, 2179, 2190, 2191, 2196, 2199, 2218, 2230, 2235, 2236, 2243, 2251, 2253, 2255, 2259, 2261, 2266, 2279, 2282, 2290, 2291, 2292, 2294, 2311, 2316, 2318, 2324, 2328, 2330, 2331, 2336, 2345, 2348, 2350, 2357, 2360, 2361, 2365, 2378, 2379, 2380, 2386, 2393, 2396, 2405, 2407, 2412, 2418, 2419, 2425, 2428, 2434, 2438, 2453, 2454, 2459, 2462, 2469, 2475, 2479, 2481, 2489, 2498]
- **Turquoises:** [2, 4, 8, 12, 16, 18, 20, 27, 28, 30, 42, 44, 54, 57, 61, 67, 72, 73, 85, 91, 113, 115, 116, 117, 119, 138, 139, 143, 151, 153, 155, 156, 166, 167, 168, 173, 174, 179, 184, 186, 187, 189, 193, 207, 210, 226, 229, 230, 232, 233, 259, 260, 261, 265, 276, 278, 282, 284, 295, 298, 300, 302, 308, 310, 312, 314, 317, 324, 339, 340, 342, 343, 346, 347, 349, 359, 360, 362, 369, 371, 377, 379, 381, 393, 401, 402, 404, 406, 410, 412, 413, 418, 419, 421, 423, 434, 440, 443, 444, 451, 453, 454, 460, 464, 469, 472, 478, 481, 482, 487, 504, 511, 512, 515, 518, 523, 525, 529, 535, 536, 537, 539, 541, 542, 546, 548, 551, 552, 557, 564, 569, 573, 580, 583, 588, 594, 596, 598, 599, 601, 602, 603, 604, 609, 626, 641, 642, 657, 660, 664, 670, 671, 675, 676, 681, 684, 685, 688, 691, 697, 699, 700, 703, 710, 712, 720, 727, 734, 739, 743, 744, 746, 747, 752, 762, 764, 771, 774, 779, 794, 798, 803, 805, 812, 815, 821, 822, 828, 834, 839, 840, 842, 847, 850, 855, 862, 864, 867, 868, 876, 879, 880, 898, 902, 906, 907, 911, 912, 921, 926, 927, 928, 933, 936, 955, 960, 976, 977, 979, 982, 985, 986, 991, 997, 999, 1002, 1009, 1018, 1025, 1028, 1033, 1038, 1045, 1049, 1059, 1070, 1071, 1076, 1077, 1085, 1099, 1104, 1109, 1110, 1115, 1119, 1120, 1124, 1125, 1131, 1132, 1133, 1143, 1146, 1147, 1150, 1158, 1160, 1163, 1165, 1167, 1169, 1171, 1172, 1173, 1187, 1189, 1190, 1194, 1199, 1225, 1229, 1235, 1238, 1239, 1248, 1256, 1259, 1263, 1265, 1272, 1275, 1277, 1282, 1286, 1288, 1294, 1295, 1296, 1297, 1306, 1308, 1310, 1312, 1316, 1321, 1336, 1338, 1340, 1341, 1344, 1345, 1348, 1352, 1353, 1356, 1363, 1372, 1381, 1386, 1391, 1397, 1415, 1419, 1420, 1423, 1425, 1426, 1435, 1439, 1441, 1442, 1445, 1451, 1455, 1464, 1466, 1471, 1475, 1483, 1485, 1489, 1492, 1496, 1498, 1500, 1504, 1507, 1513, 1516, 1517, 1519, 1522, 1524, 1527, 1528, 1534, 1537, 1543, 1544, 1549, 1551, 1554, 1559, 1560, 1561, 1566, 1567, 1571, 1572, 1581, 1583, 1587, 1595, 1597, 1598, 1599, 1601, 1604, 1613, 1631, 1635, 1638, 1652, 1656, 1660, 1669, 1670, 1674, 1680, 1683, 1684, 1689, 1690, 1699, 1700, 1702, 1707, 1712, 1714, 1719, 1720, 1727, 1728, 1730, 1731, 1737, 1738, 1742, 1750, 1754, 1760, 1761, 1766, 1767, 1769, 1770, 1772, 1779, 1786, 1791, 1811, 1816, 1822, 1827, 1831, 1832, 1835, 1846, 1847, 1853, 1855, 1857, 1863, 1868, 1874, 1877, 1888, 1897, 1900, 1905, 1910, 1915, 1918, 1919, 1922, 1926, 1927, 1928, 1931, 1933, 1937, 1948, 1955, 1956, 1958, 1959, 1965, 1967, 1970, 1976, 1980, 1997, 1999, 2000, 2003, 2009, 2014, 2015, 2017, 2018, 2019, 2023, 2028, 2035, 2036, 2038, 2044, 2046, 2051, 2058, 2060, 2070, 2071, 2073, 2074, 2081, 2085, 2086, 2087, 2099, 2103, 2112, 2114, 2119, 2120, 2126, 2128, 2129, 2131, 2155, 2168, 2181, 2182, 2187, 2188, 2192, 2200, 2201, 2209, 2210, 2213, 2215, 2219, 2224, 2225, 2226, 2241, 2248, 2256, 2258, 2276, 2278, 2280, 2283, 2296, 2300, 2306, 2308, 2309, 2313, 2320, 2321, 2325, 2337, 2343, 2344, 2346, 2352, 2363, 2364, 2369, 2373, 2374, 2376, 2389, 2391, 2400, 2401, 2402, 2403, 2411, 2416, 2430, 2431, 2432, 2445, 2448, 2451, 2452, 2458, 2460, 2461, 2465, 2466, 2470, 2471, 2473, 2474, 2480, 2483, 2485, 2490, 2492]
- **Oranges:** [6, 75, 99, 100, 147, 351, 372, 380, 492, 611, 714, 806, 820, 841, 843, 910, 939, 1006, 1007, 1183, 1228, 1257, 1354, 1359, 1440, 1495, 1573, 1591, 1679, 1682, 1691, 1734, 1781, 1913, 2016, 2037, 2105, 2137, 2153, 2166, 2211, 2212, 2232, 2242, 2273, 2302, 2429, 2496]
- **Yellows:** [13, 89, 195, 202, 206, 266, 315, 334, 396, 473, 484, 600, 607, 645, 690, 693, 729, 732, 749, 873, 1083, 1084, 1091, 1181, 1213, 1283, 1320, 1342, 1351, 1374, 1383, 1396, 1400, 1535, 1546, 1555, 1630, 1645, 1651, 1664, 1724, 1757, 1885, 2027, 2092, 2149, 2186, 2244, 2317, 2338, 2375, 2387, 2424, 2436]
- **Khakis:** [14, 15, 34, 38, 66, 74, 131, 145, 152, 159, 180, 185, 219, 236, 258, 268, 273, 277, 301, 330, 333, 337, 358, 386, 392, 395, 409, 411, 417, 437, 455, 459, 465, 466, 468, 477, 517, 527, 534, 538, 549, 554, 562, 584, 595, 618, 631, 635, 646, 653, 661, 686, 737, 738, 755, 777, 781, 804, 808, 819, 823, 832, 852, 863, 865, 877, 878, 894, 920, 931, 948, 964, 1012, 1019, 1027, 1039, 1048, 1056, 1057, 1060, 1069, 1072, 1073, 1090, 1093, 1096, 1098, 1102, 1106, 1107, 1136, 1142, 1182, 1208, 1211, 1218, 1245, 1246, 1252, 1254, 1268, 1298, 1302, 1305, 1319, 1326, 1343, 1388, 1402, 1424, 1428, 1438, 1452, 1477, 1481, 1523, 1550, 1584, 1589, 1607, 1621, 1622, 1648, 1655, 1663, 1667, 1671, 1678, 1710, 1713, 1718, 1741, 1759, 1774, 1777, 1778, 1789, 1793, 1794, 1797, 1813, 1820, 1824, 1825, 1842, 1849, 1859, 1864, 1872, 1873, 1884, 1909, 1914, 1921, 1941, 1945, 1951, 1981, 1996, 2001, 2011, 2026, 2041, 2042, 2053, 2101, 2118, 2124, 2136, 2142, 2164, 2223, 2247, 2252, 2263, 2264, 2268, 2269, 2271, 2274, 2295, 2299, 2310, 2314, 2326, 2334, 2356, 2366, 2404, 2415, 2422, 2433, 2443, 2446, 2447, 2477, 2478, 2482, 2487, 2494]
- **Purples:** [17, 78, 169, 223, 306, 309, 385, 488, 565, 636, 758, 846, 899, 913, 925, 1050, 1116, 1141, 1145, 1166, 1273, 1432, 1456, 1542, 1557, 1654, 1783, 1889, 1930, 1952, 1975, 2025, 2068, 2088, 2160, 2177, 2203, 2204, 2260, 2272, 2362, 2382, 2414, 2467]
- **Greens:** [19, 26, 45, 68, 88, 163, 178, 192, 208, 242, 263, 274, 288, 296, 327, 338, 353, 363, 387, 394, 399, 400, 445, 456, 532, 643, 655, 668, 669, 694, 695, 711, 760, 775, 814, 816, 826, 829, 833, 835, 870, 886, 889, 901, 915, 919, 943, 959, 969, 993, 994, 1000, 1031, 1043, 1052, 1079, 1081, 1095, 1113, 1237, 1240, 1255, 1261, 1281, 1285, 1289, 1392, 1398, 1407, 1409, 1434, 1472, 1491, 1493, 1538, 1553, 1570, 1606, 1644, 1672, 1717, 1725, 1790, 1801, 1812, 1826, 1934, 1986, 1994, 2049, 2093, 2140, 2148, 2150, 2151, 2156, 2171, 2175, 2183, 2237, 2254, 2257, 2262, 2281, 2285, 2301, 2333, 2339, 2347, 2351, 2358, 2367, 2390, 2397, 2399, 2437, 2440, 2441, 2442, 2450, 2486, 2491]
- **Greys:** [24, 31, 46, 81, 141, 144, 148, 149, 194, 197, 253, 331, 350, 357, 365, 435, 446, 449, 479, 486, 491, 524, 526, 533, 578, 581, 622, 730, 731, 782, 797, 810, 853, 859, 881, 946, 951, 954, 1017, 1023, 1200, 1330, 1350, 1385, 1412, 1453, 1508, 1515, 1639, 1693, 1755, 1763, 1764, 1782, 1809, 1814, 1815, 1939, 1946, 1960, 1969, 1982, 1991, 2022, 2050, 2057, 2097, 2176, 2195, 2245, 2275, 2303, 2354, 2392, 2406, 2449, 2456]
- **Reds:** [37, 128, 146, 213, 214, 220, 227, 248, 250, 256, 262, 321, 323, 335, 354, 415, 416, 457, 480, 508, 514, 520, 540, 547, 550, 572, 610, 625, 634, 692, 713, 716, 757, 857, 892, 941, 947, 956, 973, 1003, 1020, 1022, 1029, 1035, 1065, 1074, 1086, 1174, 1180, 1212, 1224, 1226, 1251, 1258, 1318, 1404, 1418, 1446, 1454, 1503, 1520, 1558, 1582, 1602, 1620, 1623, 1624, 1628, 1659, 1681, 1688, 1705, 1721, 1746, 1804, 1828, 1833, 1836, 1882, 1907, 1923, 1953, 1963, 2021, 2032, 2059, 2061, 2090, 2109, 2123, 2146, 2154, 2172, 2197, 2208, 2249, 2270, 2293, 2353, 2355, 2371, 2383, 2408, 2435]

- **Blues:** [47, 50, 69, 86, 87, 93, 97, 101, 150, 158, 162, 175, 221, 228, 241, 245, 280, 294, 299, 348, 375, 376, 378, 431, 436, 438, 490, 500, 522, 543, 575, 629, 656, 680, 698, 725, 733, 773, 795, 799, 818, 825, 827, 860, 874, 884, 924, 929, 934, 938, 953, 974, 975, 987, 989, 990, 996, 1014, 1044, 1047, 1053, 1080, 1082, 1092, 1108, 1122, 1152, 1184, 1186, 1197, 1249, 1291, 1301, 1307, 1317, 1334, 1335, 1339, 1366, 1375, 1433, 1531, 1536, 1565, 1568, 1596, 1603, 1708, 1756, 1758, 1768, 1800, 1829, 1841, 1856, 1866, 1881, 1895, 1902, 1911, 1912, 1925, 1940, 1961, 1989, 2004, 2034, 2067, 2069, 2141, 2147, 2162, 2163, 2207, 2214, 2216, 2227, 2240, 2289, 2368, 2370, 2484, 2495, 2497]
- **Pinks:** [48, 84, 112, 364, 382, 458, 476, 483, 485, 591, 597, 678, 754, 786, 802, 882, 887, 895, 983, 1144, 1179, 1250, 1271, 1274, 1278, 1427, 1450, 1465, 1469, 1487, 1574, 1616, 1619, 1668, 1696, 1697, 1698, 1704, 1709, 1716, 1765, 1798, 1799, 1880, 1974, 2079, 2080, 2089, 2135, 2174, 2297]
- **Whites:** [52, 123, 171, 212, 283, 493, 503, 509, 521, 531, 558, 561, 566, 613, 644, 654, 667, 701, 706, 708, 742, 772, 837, 935, 952, 978, 980, 1011, 1036, 1088, 1134, 1191, 1210, 1227, 1232, 1253, 1331, 1364, 1444, 1494, 1552, 1642, 1649, 1715, 1732, 1747, 1840, 1870, 1950, 1978, 1990, 2024, 2110, 2111, 2205, 2335, 2359, 2385, 2398, 2410],
- **Violets:** [102, 110, 286, 292, 432, 662, 705, 765, 851, 1015, 1193, 1457, 1476, 1482, 1506, 1594, 1618, 1753, 1936, 1962, 2030, 2048, 2108, 2189]

6. Confusion Matrix:



We can observe Critical trends of the classifier, For example the classification of large quantities of Browns color instead of Purples and Violets colors. Allot of votes was given to the White instead of the Khakis and vice versa, which means the classifier tends to get confused between White and Khakis classification.

First Bonus :

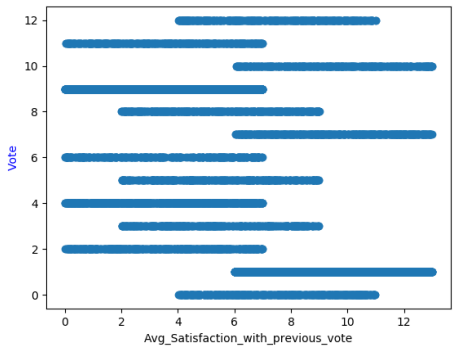
1. Automation of choosing the best model:
   First, we emphasize that from the beginning we made an automatic selection of the model chosen by maximizing the accuracy measure over the validation set. We chose this accuracy measure because we understand that in order to provide the predictions of a winning party and the distribution of votes, accuracy is the most important measure and therefore we examined a classifier that maximizes this measure.
   After selecting the parameters for each classifier, we automatically went over each classifier and selected the classifier with the maximum accuracy percentage over the validation set, as mentioned before the selected classifier is: Random Forest Classifier.
2. In the mandatory sections, we chose the accuracy measure to select the best classifier. For each of the prediction tasks, we now select a measure that we think is best for performing the task and by which we select the classifier:
   a. According to the training set it can be seen that the winning party is the Turquoises, so the measure we chose to perform the task is a binary index, whether the classifier was able to guess what the winning party, tested the same types of models as before and we saw that all models were able to predict who the winning party is and so it is preferable to choose The simplest classifier like KNN which is easy to understand and does not require long learning.
   b. For the population vote distribution task, we chose the precision measure with the understanding that the more accurate is classified the more accurate the vote distribution would be and similar to the training set, thus we chose the Random Forest Classifier class, as before. We will elaborate that because the classification probability can be obtained directly from decision tree algorithms, these algorithms will be more robust when asked about probabilistic metrics.
   c. For the shuttle assignment there are some considerations that we did not consider in the previous prediction, for example: Quality of Service Consideration: We want to send shuttle services to anyone who intends to vote for the party and therefore want to maximize the True Positive index and minimize the False Negative for a particular party which means maximizing the recall.

For financial reasons, we do not want to send shuttle services for a voter who will not vote for the party at the end, which means that we want that our classifier will minimize the False Positive index and therefore want to maximize the precision index.
We can now provide each party with the best classification for it based on its considerations, for example for a party that considers both considerations we want to maximize F1.
(implemented this in the attached script)

After the script is run, its results can be examined and noticed, for example, that for the Blues Party KNN achieves a high precision for it, as opposed to the previous test that chose Random Forest.

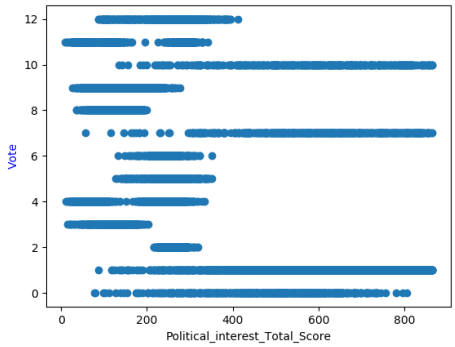
3. For the fourth prediction task, identifying voter characteristics which changes can lead to another winner in the election we performed the following process:
    a. Understanding the characteristics that can change the conductor in elections, we note that the winning color is Turquoises and immediately followed by Brown, so we want to look for manipulations that will cause our classifier to transfer Turquoises color to other colors and especially to Brown.
    To understand who the traits are, we looked again at the relationship between the traits and the classification we performed in the previous exercise and came to these conclusions:

    Trait: Avg_Satisfaction_with_previous_vote



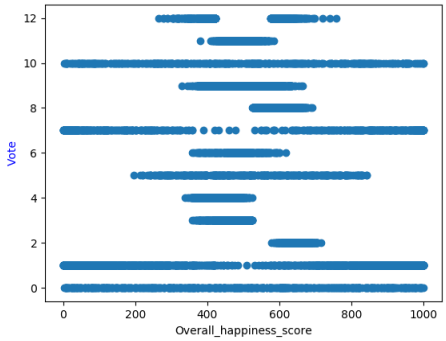
The Turquoises color (9) mainly indicates small values and the Brown color (1) mainly large values, so possible manipulation will be to increase the values in this property.

Trait: Political_interest_Total_Score



The Turquoises color (9) mainly indicates small values and the Brown color (1) mainly avg and large values, so possible manipulation will be to increase the values in this property.

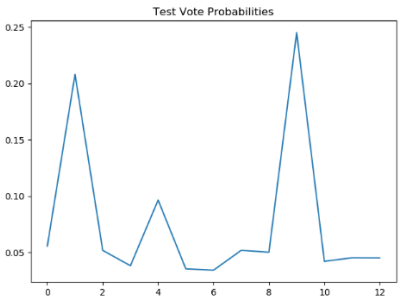
Trait: Overall_happiness_score



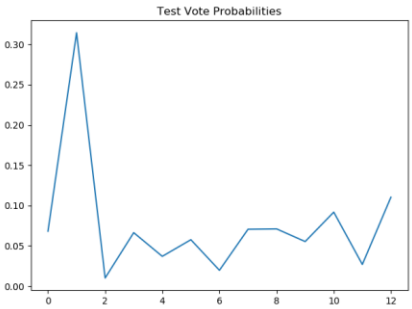
The Turquoises color (9) mainly indicates values in the middle of the scale and the Brown color (1) mainly small and large values, so possible manipulation will be to decrease the values in this property.

b. We performed the manipulations on the test set when the training set was left unchanged.
c. Compare The results of the manipulations to the original results:

The original results:

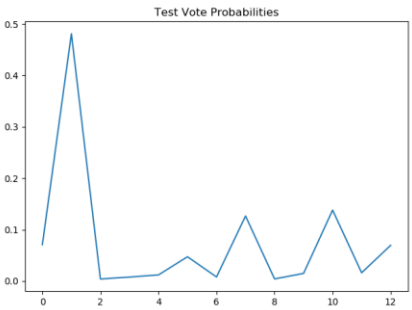


increasing Avg_Satisfaction_with_previous_vote Manipulation:



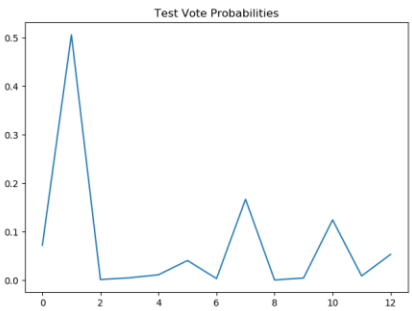
The browns wins against the Turquoises who have lost allot of votes

increasing Political_interest_Total_Score Manipulation:



The browns wins against the Turquoises who have lost allot of votes

increasing Overall_happiness_score Manipulation:



The browns wins against the Turquoises who even failed to reach the second place.

Second Bonus LMS Vs. Perceptron:

1. Comparing the algorithms given the following hyperparameters:

```
Perceptron(alpha=0.0001, max_iter=300, verbose=True, tol=1e-3)
AdalineSGD(eta=0.001, epochs=300)
```

we trained on 70% of the data set and tested on 30 %

a. Iris Sample set

1 Vs. All	Perceptron		Adaline, LMS Widrow-Hoff	
Class	Accuracy	Convergent [epochs]	Accuracy	Convergent [epochs]
0	Train:100% Test: 100%	7	Train:100% Test: 100%	1
1	Train: 77.14 % Test:71.11%	23	Train:74.29% Test:75.56%	169
2	Train: 97.14% Test:97.78%	8	Train: 93.33% Test:93.33%	198

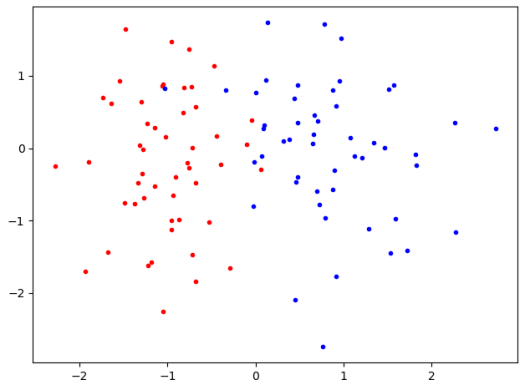
b. Digits sample set

1 Vs. All	Perceptron		Adaline, LMS, Widrow-Hoff	
Class	Accuracy	Convergent [epochs]	Accuracy	Convergent [epochs]
0	Train: 100.00% Test: 99.81%	11	Train: 99.76% Test: 99.44%	3
1	Train: 98.97% Test: 95.56%	12	Train: 98.01% Test: 97.41%	89
2	Train: 99.44% Test: 98.89%	9	Train: 98.65% Test: 99.81%	7
3	Train: 98.25% Test: 96.48%	37	Train: 98.01% Test: 97.96%	1
4	Train: 100.00% Test: 98.89%	17	Train: 99.44% Test: 99.26%	8
5	Train: 100.00% Test: 98.89%	30	Train: 99.12% Test: 98.70%	2
6	Train: 99.68% Test: 99.26%	18	Train: 99.44% Test: 98.89%	77
7	Train: 99.20%	14	Train: 99.05%	2

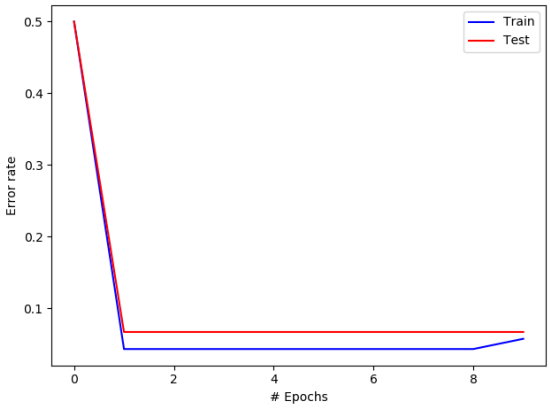
	Test: 99.07%		Test: 99.63%	
8	Train: 95.47% Test: 94.26%	11	Train: 96.02% Test: 96.11%	83
9	Train: 98.33% Test: 97.41%	17	Train: 97.30% Test: 96.85%	11

We noticed that the Adaline algorithm converges faster on the digits data set and for most classifications, achieves better accuracy than perceptron.

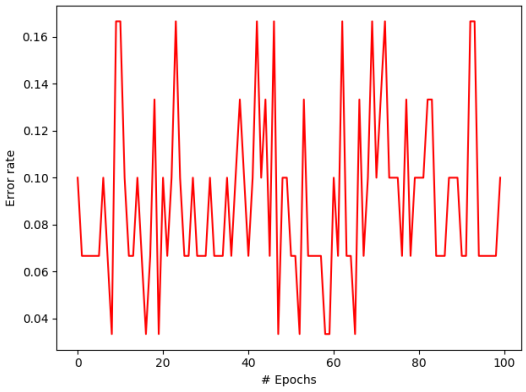
2. In order to create an data set that Perceptron converges faster we will create a set of information that is very easy to separate linearly and that data set which cannot be separated linearly then Perceptron will not converge at all and the Adaline that tries to minimize the cost function which is MSE .
   The first set of information contains 2 non-separable features:



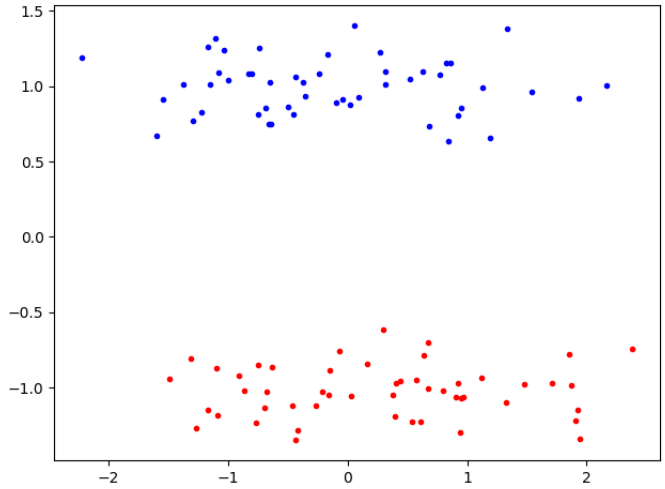
The Adaline Converge in the First Epoch:



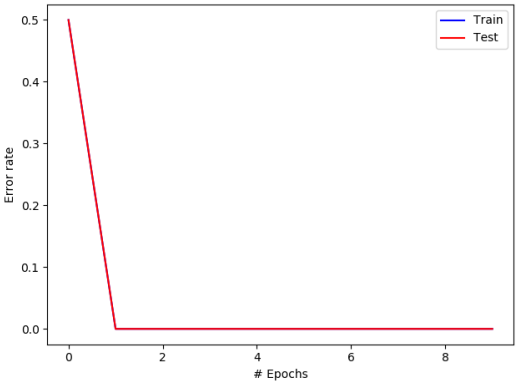
It can be seen that the Perceptron does not have a downward trend on the training set even when increasing the amount of information transferred :



The second set of information contains 2 separable features



The Adaline Converge in the First Epoch:



We can see that the Perceptron converges right at the start with 100% accuracy and additional transitions do not change anything