# Final Project

MTH 3270

Tobias Boggess, Carl Perry

5/12/2022

### Data Wrangling

The task of data wrangling for this project was fairly straightforward. After converting the 4 year data into a data frame, it was then filtered to the academic year 2017. Then the supplemental data was combined through the left join by unit ID, institution name, and year. The data was then split into a test and training sets and mutate was used to add the necessary prediction columns for the first task. The second task used select to isolate the columns of supplemental data for k cluster analysis, and rescale to standardize the explanatory variables.

### Task 1

### Decision Tree

A decision tree was used as the classification procedure and the explanatory variables selected were Undergrad enrollment, tuition and fees, and total enrollment. These variables were used to train and test the model with a minimum split value of 10, 100, and 300 for the tuning parameter.

With a minimum split of 10 the model reported an accuracy of .562, which was the highest correct classification rate of the three tested. The other tested values both reported an accuracy of .560.

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy multiclass     0.562
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy multiclass     0.560
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy multiclass     0.560
```

**K Nearest Neighbor**
| The other classification procedure used was k nearest neighbor, and the explanatory variables selected were Undergrad enrollment, tuition and fees, and total enrollment. These variables were used to train and test the model with k values of 15, 60, and 100 for the tuning parameter.

With a k value of 15 the model had an accuracy of .718, which was the highest reported correct classification rate. With a k value of 60 the model had an accuracy of .658, and at 100 the model had an accuracy of .648.

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy multiclass     0.718
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy multiclass     0.658
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy multiclass     0.648
```

**Task 2**

**k Means**
| The cluster analysis used was k means, and the explanatory variables used were undergraduate enrollment, student to faculty ratio, total enrollment, graduation rate, and reported percentage of white students. The analysis was run using a total of five K groups. The clusters included 462, 264, 253, 527 and 136 institutions respectively.

The sum of squares by cluster was 63.5%. This would indicate that the majority of clusters correspond to the four-year institutional categories.

```
## K-means clustering with 5 clusters of sizes 462, 264, 253, 527, 136
##
## Cluster means:
##   DRVEF2017_RV.Undergraduate.enrollment EF2017D_RV.Student.to.faculty.ratio
## 1                            0.04893451                           0.2915020
## 2                            0.04731893                           0.2119565
## 3                            0.04745887                           0.3270321
## 4                            0.04096157                           0.2595495
## 5                            0.36688230                           0.3906650
##   total_enrollment DRVGR2017_RV.Graduation.rate..total.cohort col_white
## 1       0.04585127                                  0.3691126 0.6431043
## 2       0.04764690                                  0.7357955 0.4638269
## 3       0.04743947                                  0.3250198 0.1552305
## 4       0.04015783                                  0.6395825 0.7618411
## 5       0.36710828                                  0.6597059 0.5550567
##
## Clustering vector:
##    1    2    3    4    5    7    8   10   11   12   13   14   15   16   17   18
##    3    1    1    3    5    1    5    3    1    1    1    1    1    1    3    1
```

```
##  19  20  21  22  23  24  25  26  27  28  30  31  32  33  34  36
##   1   1   3   4   1   4   3   3   1   3   3   1   1   1   2   5
##  37  38  39  40  42  43  45  46  47  50  51  52  53  55  57  58
##   2   4   5   5   1   1   1   2   2   1   2   2   1   1   4   5
##  59  61  62  63  64  65  66  67  68  69  70  72  73  74  76  77
##   3   1   1   1   4   1   4   4   4   1   3   1   1   1   3   2
##  78  79  80  82  83  84  85  86  88  89  90  91  92  93  94  95
##   2   3   2   2   2   2   2   5   3   3   3   5   5   3   5   5
##  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111
##   3   5   3   5   5   5   5   5   5   5   5   5   5   2   2   2
## 112 113 114 115 117 118 119 120 122 123 124 125 127 130 131 132
##   3   2   2   2   1   1   2   2   2   3   1   3   3   2   2   2
## 134 136 138 139 140 141 142 143 144 145 146 147 150 151 152 153
##   3   4   2   3   2   3   3   3   3   2   2   2   3   2   2   2
## 154 155 156 159 160 161 162 163 164 165 166 167 168 169 170 171
##   2   2   2   5   2   1   2   5   2   4   5   2   2   1   2   2
## 172 174 175 176 177 178 179 180 181 182 183 187 190 191 192 194
##   2   2   2   2   4   2   2   2   1   5   2   3   2   3   3   2
## 196 198 199 200 202 203 204 208 209 210 211 212 215 216 217 218
##   3   3   3   3   2   2   3   2   3   3   2   2   3   1   1   1
## 219 220 221 223 224 225 226 227 228 229 230 231 232 233 234 237
##   5   1   2   4   5   5   4   1   1   1   1   4   1   2   1   4
## 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255
##   1   2   3   1   4   5   4   4   2   2   1   4   4   3   4   4
## 256 257 258 259 260 261 262 263 264 265 266 267 268 269 271 273
##   2   1   2   2   1   2   4   4   2   3   5   1   3   3   2   3
## 274 275 276 277 278 279 280 282 283 284 285 287 288 289 290 291
##   2   2   2   3   3   3   3   4   3   3   1   1   5   3   4   3
## 292 293 294 297 298 299 300 301 302 303 304 305 306 307 308 309
##   2   3   5   2   5   3   4   5   5   3   1   3   1   3   2   5
## 310 311 312 313 314 319 320 321 322 323 324 325 326 327 329 330
##   3   1   2   2   3   4   1   4   4   1   1   1   3   1   4   1
## 331 332 333 335 336 338 339 340 341 344 345 346 347 348 349 351
##   4   1   3   1   2   4   1   3   3   3   3   3   3   1   2   3
## 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367
##   1   3   4   3   1   3   3   1   4   1   1   1   2   3   5   1
## 368 369 370 371 372 373 374 375 376 377 378 380 381 382 383 384
##   4   5   5   5   1   1   3   2   3   3   3   1   2   3   1   2
## 385 386 387 388 389 390 391 393 394 396 398 399 400 401 402 403
##   3   4   1   1   2   1   1   3   3   3   1   1   3   1   3   3
## 404 405 406 407 409 410 411 412 413 414 415 416 417 418 419 420
##   3   3   3   2   3   2   4   5   1   4   4   1   4   5   2   2
## 421 422 423 425 426 427 428 429 430 431 432 434 436 437 439 441
##   4   1   4   4   3   2   1   1   5   3   4   1   1   1   2   4
## 442 443 445 446 448 449 451 452 453 454 455 456 458 459 460 461
##   4   2   5   2   2   2   2   4   2   1   1   1   4   4   4   3
## 463 464 465 466 467 468 469 471 472 473 474 477 479 481 482 483
##   4   2   1   2   3   4   1   3   1   1   2   4   2   4   1   1
## 484 485 486 487 489 490 493 495 496 497 498 500 501 502 503 504
##   1   4   1   2   1   4   3   5   1   4   4   4   1   4   2   4
## 505 506 507 508 509 510 512 513 514 515 516 517 518 519 520 521
##   4   2   4   4   1   4   5   4   3   1   1   1   1   5   1   1
## 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537
##   1   4   4   4   3   2   4   4   4   4   4   4   4   4   4   5
```

```
##  538  542  543  544  545  546  547  548  549  550  552  553  554  555  556  557 
##    1    3    4    1    4    4    4    4    4    2    4    1    4    1    1    2 
##  559  560  561  562  563  565  566  567  568  569  570  571  572  573  574  575 
##    5    1    5    4    4    4    4    4    4    4    4    4    1    1    4    1 
##  577  578  579  580  581  582  583  584  585  586  587  588  589  590  591  592 
##    4    4    1    1    1    1    1    4    1    3    5    1    5    1    1    1 
##  593  594  596  597  598  599  600  601  602  603  604  605  606  607  608  609 
##    1    1    1    1    1    1    1    1    1    2    4    4    4    2    1    1 
##  610  611  612  613  614  615  616  617  618  619  620  621  622  623  624  625 
##    4    4    1    1    4    4    3    1    5    1    1    4    4    1    4    1 
##  626  627  628  630  631  632  633  634  635  637  638  639  640  641  642  643 
##    1    1    1    4    1    1    1    3    3    1    5    1    1    1    2    1 
##  644  645  646  647  648  649  650  651  652  653  654  655  657  658  659  660 
##    1    1    1    1    2    1    1    3    3    1    4    3    4    4    2    2 
##  661  662  663  664  665  666  667  668  669  670  671  672  673  674  675  676 
##    1    4    1    1    4    4    1    4    4    4    1    1    4    3    3    2 
##  677  678  679  680  682  683  684  687  688  689  690  691  692  693  694  695 
##    3    3    1    2    2    2    4    2    5    2    3    3    4    4    1    4 
##  697  698  699  700  701  703  705  706  707  708  710  711  712  713  714  715 
##    4    5    1    4    4    3    1    3    2    1    2    4    4    1    1    2 
##  716  717  718  719  720  721  722  723  724  725  726  727  728  729  730  731 
##    2    3    2    2    2    4    3    2    4    1    4    4    4    4    4    4 
##  732  733  734  735  737  738  739  740  741  742  743  744  745  747  748  750 
##    4    2    4    4    4    5    5    3    2    2    2    4    4    2    2    2 
##  752  753  754  755  756  757  758  760  761  762  763  764  765  766  767  768 
##    4    4    2    2    3    1    1    2    4    1    4    2    2    2    4    4 
##  769  770  772  773  774  775  776  777  778  779  780  781  783  785  786  787 
##    4    4    2    2    4    4    2    4    4    4    2    4    5    1    2    1 
##  788  789  790  791  792  793  794  795  796  797  798  799  802  803  804  805 
##    4    1    1    4    4    5    1    4    2    1    4    4    5    5    4    4 
##  806  807  808  809  810  811  812  813  814  815  816  818  819  820  823  824 
##    1    4    1    1    1    4    2    1    1    4    1    5    5    1    2    4 
##  825  826  827  828  829  830  831  832  833  834  835  836  837  838  839  840 
##    4    4    2    4    4    4    4    4    2    1    1    5    4    4    4    2 
##  841  842  843  844  845  846  847  848  849  850  851  852  853  854  855  856 
##    1    1    4    2    4    4    1    4    4    4    4    4    4    1    1    4 
##  858  859  860  861  862  863  864  865  866  867  868  869  870  871  872  873 
##    4    3    2    4    1    3    4    5    1    3    4    5    3    1    3    4 
##  875  876  877  878  879  880  881  882  883  884  885  886  887  888  889  891 
##    1    1    1    4    4    4    1    1    4    1    4    4    4    4    3    4 
##  892  893  894  896  897  898  899  900  901  902  903  904  905  906  907  908 
##    4    3    1    4    1    1    1    1    1    5    1    4    4    4    4    4 
##  909  910  911  913  914  915  916  918  919  920  921  922  923  924  925  926 
##    4    1    3    4    4    1    4    4    4    5    2    4    4    4    4    1 
##  927  928  930  932  933  934  935  936  937  938  939  940  941  942  945  946 
##    1    4    1    4    1    1    4    1    1    4    4    1    4    4    4    4 
##  948  949  950  951  954  955  956  957  958  959  960  963  964  966  968  969 
##    4    2    1    1    4    5    1    1    2    4    1    5    5    3    3    4 
##  970  971  973  974  975  976  977  978  979  980  981  982  985  986  987  988 
##    2    4    1    5    4    4    1    4    4    4    1    4    3    2    2    2 
##  989  990  991  992  993  994  995  996  997  998  999 1000 1001 1002 1003 1004 
##    3    3    1    1    5    3    3    4    5    2    2    1    4    2    2    5 
## 1005 1006 1007 1009 1010 1011 1013 1014 1015 1016 1017 1018 1020 1022 1023 1024 
##    2    3    2    4    4    1    2    2    1    3    3    1    1    1    3    1 
```

```
## 1025 1026 1027 1028 1029 1031 1032 1033 1035 1036 1037 1038 1039 1040 1041 1042
##    3    3    1    5    3    3    3    2    2    3    3    2    2    4    2    2
## 1043 1044 1045 1047 1048 1049 1050 1051 1052 1053 1055 1056 1057 1058 1059 1060
##    3    1    3    4    4    1    4    2    3    2    2    2    2    2    1    3
## 1062 1063 1064 1065 1066 1067 1068 1069 1070 1071 1072 1073 1074 1075 1076 1077
##    3    3    3    3    3    4    4    4    3    4    1    2    2    4    4    4
## 1078 1079 1080 1081 1082 1083 1084 1085 1086 1087 1090 1091 1092 1093 1094 1095
##    2    4    2    4    4    2    4    4    1    4    1    2    2    2    4    1
## 1096 1097 1098 1099 1100 1101 1102 1103 1104 1105 1107 1108 1109 1110 1111 1112
##    1    3    4    1    1    2    3    1    3    4    2    5    4    3    2    3
## 1114 1115 1116 1117 1118 1119 1120 1121 1122 1123 1125 1126 1127 1128 1129 1130
##    4    4    2    1    4    1    2    4    4    2    4    1    4    4    2    2
## 1131 1132 1133 1134 1135 1137 1138 1139 1140 1141 1142 1143 1144 1146 1147 1148
##    1    4    2    2    4    1    1    5    5    5    5    4    4    1    4    4
## 1149 1150 1151 1152 1153 1154 1155 1156 1158 1159 1161 1162 1163 1164 1165 1166
##    4    4    4    4    1    2    3    4    1    4    5    1    4    4    4    4
## 1167 1168 1169 1170 1171 1172 1173 1174 1175 1176 1177 1178 1179 1180 1181 1182
##    1    2    2    4    4    1    4    1    1    1    4    4    4    4    3    4
## 1183 1184 1185 1186 1187 1188 1189 1190 1191 1192 1193 1194 1195 1196 1197 1198
##    1    4    4    1    1    4    3    4    1    1    3    1    1    1    3    2
## 1199 1200 1201 1202 1203 1204 1205 1206 1207 1208 1210 1211 1212 1213 1214 1215
##    2    5    3    4    3    1    1    1    1    4    3    1    1    3    1    4
## 1216 1217 1218 1219 1220 1221 1222 1223 1224 1225 1226 1227 1228 1229 1230 1231
##    1    1    1    3    4    5    5    5    3    4    5    3    4    1    3    1
## 1232 1233 1234 1235 1236 1237 1238 1239 1240 1241 1242 1243 1244 1246 1248 1250
##    1    1    3    2    3    1    1    4    4    1    3    3    4    3    3    3
## 1251 1252 1253 1254 1255 1256 1257 1258 1259 1260 1261 1262 1263 1264 1265 1266
##    1    4    4    1    1    4    4    1    1    1    1    4    4    4    4    4
## 1267 1268 1269 1270 1272 1273 1274 1275 1276 1277 1278 1279 1280 1281 1282 1283
##    4    2    4    3    5    1    4    2    1    4    4    1    2    4    1    1
## 1284 1285 1286 1287 1288 1289 1290 1291 1292 1293 1294 1295 1296 1297 1298 1299
##    4    4    4    5    4    1    1    4    4    1    1    5    4    4    4    4
## 1300 1301 1302 1303 1304 1305 1306 1307 1308 1309 1310 1311 1312 1313 1314 1315
##    4    1    2    1    4    5    5    4    4    4    4    1    4    3    1    3
## 1317 1318 1319 1320 1321 1322 1323 1324 1325 1326 1328 1330 1331 1332 1333 1334
##    4    4    3    4    4    4    1    4    4    1    1    1    3    1    1    1
## 1335 1336 1337 1338 1339 1340 1341 1343 1344 1345 1346 1347 1348 1349 1350 1351
##    1    1    1    3    1    1    4    1    5    4    4    5    2    1    1    1
## 1352 1353 1354 1355 1357 1358 1359 1360 1361 1364 1365 1368 1370 1371 1372 1373
##    1    1    2    1    1    1    1    4    2    3    1    1    5    5    1    2
## 1374 1375 1376 1377 1379 1380 1381 1385 1386 1387 1388 1389 1392 1393 1394 1395
##    5    2    2    1    4    2    1    4    1    4    4    4    4    4    4    2
## 1396 1397 1398 1399 1400 1401 1402 1403 1404 1405 1406 1407 1408 1409 1410 1411
##    4    2    4    4    2    1    1    4    2    3    4    2    4    2    2    4
## 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1424 1425 1426 1427 1428
##    1    2    1    4    2    4    4    4    1    2    4    4    4    4    1    4
## 1429 1431 1432 1433 1434 1435 1436 1437 1438 1439 1440 1441 1443 1444 1445 1446
##    4    2    2    4    4    2    3    4    4    4    4    4    4    4    2    4
## 1447 1448 1449 1450 1451 1452 1454 1455 1456 1457 1458 1459 1460 1461 1462 1463
##    1    4    2    4    4    4    4    4    1    4    2    1    4    5    1    1
## 1464 1465 1466 1467 1468 1469 1471 1472 1473 1474 1475 1476 1477 1478 1479 1480
##    1    4    3    4    1    2    2    4    2    1    1    4    5    4    4    2
## 1481 1482 1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1495 1496 1497
##    4    4    4    4    4    4    4    2    4    2    4    5    1    4    1    4
```

```
## 1498 1499 1500 1501 1502 1503 1504 1505 1506 1507 1508 1509 1510 1511 1512 1513
##    4    4    4    4    4    4    4    1    4    3    3    3    2    4    2    4
## 1514 1515 1516 1517 1518 1519 1520 1521 1522 1523 1524 1525 1526 1527 1528 1530
##    1    4    2    4    4    3    4    1    3    4    1    4    4    3    5    4
## 1531 1532 1533 1534 1536 1539 1540 1541 1542 1545 1546 1547 1548 1549 1550 1551
##    1    4    2    1    1    3    1    4    4    5    1    3    1    3    4    4
## 1552 1555 1556 1557 1558 1559 1561 1562 1563 1564 1565 1566 1567 1568 1569 1570
##    3    3    4    1    1    4    1    1    1    4    4    4    4    1    3    1
## 1572 1573 1574 1575 1576 1577 1578 1579 1580 1581 1582 1583 1584 1585 1586 1587
##    4    1    4    1    1    1    4    1    3    4    4    4    4    1    3    3
## 1588 1589 1590 1591 1592 1593 1594 1596 1598 1599 1600 1601 1602 1603 1604 1605
##    4    4    1    4    1    5    5    4    2    1    1    5    4    3    4    1
## 1607 1608 1609 1610 1611 1612 1614 1616 1617 1618 1619 1620 1621 1622 1623 1624
##    4    2    4    1    4    3    1    1    4    1    1    3    2    1    5    1
## 1625 1626 1627 1628 1629 1630 1631 1633 1634 1636 1637 1638 1639 1640 1641 1642
##    3    4    1    4    1    1    1    1    3    3    3    5    1    3    3    3
## 1643 1644 1645 1646 1647 1648 1649 1650 1651 1652 1653 1654 1655 1656 1657 1658
##    1    3    4    1    1    1    1    5    3    3    3    3    2    2    3    5
## 1659 1660 1661 1662 1663 1664 1665 1666 1667 1668 1671 1672 1673 1674 1675 1676
##    1    3    2    1    2    1    5    3    3    1    3    5    5    5    5    3
## 1677 1678 1679 1680 1682 1683 1684 1685 1686 1687 1690 1691 1692 1694 1696 1697
##    1    4    3    1    5    3    5    3    3    2    3    1    3    3    1    1
## 1699 1700 1701 1702 1703 1704 1705 1706 1709 1710 1711 1712 1713 1714 1715 1716
##    3    3    3    3    3    3    2    1    3    5    1    1    5    5    5    5
## 1717 1718 1719 1720 1723 1724 1725 1726 1730 1731 1732 1733 1736 1737 1738 1739
##    4    1    1    4    2    4    4    1    4    2    4    4    1    4    1    1
## 1740 1742 1743 1745 1746 1747 1748 1749 1750 1751 1752 1753 1754 1755 1756 1757
##    4    1    4    4    4    1    5    4    3    4    5    5    4    4    1    4
## 1758 1760 1761 1762 1763 1764 1765 1766 1767 1768 1769 1770 1771 1772 1773 1774
##    2    3    5    4    4    4    2    2    4    2    1    3    4    1    5    5
## 1775 1776 1777 1778 1779 1780 1782 1783 1785 1788 1790 1792 1793 1794 1795 1796
##    5    4    3    3    2    4    3    3    3    3    3    1    2    4    1    4
## 1797 1798 1800 1801 1802 1803 1804 1805 1806 1807 1808 1809 1810 1811 1812 1813
##    4    3    4    4    4    2    2    2    4    5    5    5    4    4    1    4
## 1814 1815 1816 1818 1819 1820 1821 1822 1823 1824 1825 1826 1827 1828 1829 1831
##    2    2    3    1    1    4    1    1    1    1    1    1    1    4    1    1
## 1832 1833 1834 1835 1836 1839 1840 1841 1842 1843 1844 1846 1847 1848 1849 1850
##    1    1    1    4    5    1    4    2    1    4    4    4    4    4    2    4
## 1851 1852 1853 1854 1855 1856 1857 1858 1860 1861 1862 1863 1864 1865 1866 1867
##    4    4    4    4    1    4    4    4    4    4    4    4    4    4    4    1
## 1868 1869 1870 1871 1872 1873 1874 1876 1877 1878
##    4    1    5    5    4    4    4    1    1    4
##
## Within cluster sum of squares by cluster:
## [1] 22.48502 12.16725 16.70263 15.46362 13.95605
##  (between_SS / total_SS =  63.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Code for Final Project

```
# Authors: TObias Boggess, Carl Perry
# Date: May 04, 2022
# Description: Final Project

################################################################################
#                                 Saving Data                                  #
################################################################################
library(dplyr)
library(tidyr)
library(ggplot2)
library(randomForest)
library(rpart)
library(yardstick)
library(kknn)
library(scales)
library(mclust)
# Load in the Racial and Ethnic Representativeness of US Postsecondary
# Education Institutions data set.
my.file <- file.choose()
fryr.cllg <-
  read.csv(file = my.file,
           header = TRUE,
           sep = ",",
           stringsAsFactors = FALSE)

# Filter out all other years besides 2017
fryr.cllg <- filter(fryr.cllg, year == 2017)

# Load in supplemental 2017 data set
my.file1 <- file.choose()
sup.2017 <-
  read.csv(
    file = my.file1,
    header = TRUE,
    sep = ",",
    stringsAsFactors = FALSE
  )

# Joint data sets of fryr.cllg and sup.2017
comb.fryr <- left_join(
  x = fryr.cllg,
  y = sup.2017,
  by = c("unitid", "inst_name" = "institution.name", "year")
)
# Splitting data set into train and test data sets
set.seed(54)
temp <- sort(sample(nrow(comb.fryr), nrow(comb.fryr)*.75))
comb.fryr.train <- comb.fryr[temp,]
comb.fryr.test <- comb.fryr[-temp,]

################################################################################
#                                   Task One                                   #
```

```r
####################################################################################


################################### Decision Trees ################################
# Decision tree with minsplit of 10
set.seed(34)
tree.comb <- rpart(fourcat ~ DRVEF2017_RV.Undergraduate.enrollment +
                             DRVIC2017.Tuition.and.fees..2016.17 +
                             total_enrollment,
                             data = comb.fryr.train,
                             control = rpart.control(minsplit = 10))

# Creates the predictions based on the decision tree above
comb.preds <- predict(tree.comb, newdata = comb.fryr.test, type = "class")
# comb.preds

# Adding predictions to data frame comb.fryr.test
comb.fryr.test1 <- mutate(comb.fryr.test, predType = comb.preds)

# Determines the accuracy of the decision tree above
accuracy(
  data = comb.fryr.test1,
  truth = as.factor(fourcat),
  estimate = as.factor(predType)
)


# Decision tree with minsplit of 100
set.seed(12)
tree.comb1 <- rpart(fourcat ~ DRVEF2017_RV.Undergraduate.enrollment +
                              DRVIC2017.Tuition.and.fees..2016.17 +
                              total_enrollment,
                              data = comb.fryr.train,
                              control = rpart.control(minsplit = 100))

# Creates the predictions based on the decision tree above
comb.preds1 <- predict(tree.comb1, newdata = comb.fryr.test, type = "class")
# comb.preds1

# Adding predictions to data frame comb.fryr.test
comb.fryr.test2 <- mutate(comb.fryr.test, predType = comb.preds1)

# Determines the accuracy of the decision tree above
accuracy(
  data = comb.fryr.test2,
  truth = as.factor(fourcat),
  estimate = as.factor(predType)
)


# Decision tree with minsplit of 300
set.seed(98)
tree.comb2 <- rpart(fourcat ~ DRVEF2017_RV.Undergraduate.enrollment +
```

```
                      DRVIC2017.Tuition.and.fees..2016.17 +
                      total_enrollment,
                    data = comb.fryr.train,
                    control = rpart.control(minsplit = 300))

# Creates the predictions based on the decision tree above
comb.preds2 <- predict(tree.comb2, newdata = comb.fryr.test, type = "class")
# comb.preds1

# Adding predictions to data frame comb.fryr.test
comb.fryr.test3 <- mutate(comb.fryr.test, predType = comb.preds2)

# Determines the accuracy of the decision tree above
accuracy(
  data = comb.fryr.test3,
  truth = as.factor(fourcat),
  estimate = as.factor(predType)
)


################################ K Nearest Neighbors ###########################
comb.fryr.knn.train <-
  filter(comb.fryr.train,
         !is.na(comb.fryr.train$DRVEF2017_RV.Undergraduate.enrollment) &
         !is.na(comb.fryr.train$DRVIC2017.Tuition.and.fees..2016.17))

comb.fryr.knn.test <-
  filter(comb.fryr.train,
         !is.na(comb.fryr.train$DRVEF2017_RV.Undergraduate.enrollment) &
         !is.na(comb.fryr.train$DRVIC2017.Tuition.and.fees..2016.17))

# K nearest neighbor with tuning parameter k = 15
set.seed(102)
 knn.comb <- kknn(as.factor(fourcat) ~ DRVEF2017_RV.Undergraduate.enrollment +
                                       DRVIC2017.Tuition.and.fees..2016.17 +
                                       total_enrollment,
                  train = comb.fryr.knn.train,
                  test = comb.fryr.knn.test,
                  k = 15)

# Predictions to be added to data frame in test data set
knn.comb.preds <- fitted(knn.comb)
comb.fryr.test4 <- mutate(comb.fryr.knn.test, predFourcat = knn.comb.preds)

# Accuracy of k nearest neighbor model above
accuracy(comb.fryr.test4,
         truth = as.factor(fourcat),
         estimate = as.factor(predFourcat))


# K nearest neighbor with tuning parameter k = 60
set.seed(103)
knn.comb1 <- kknn(as.factor(fourcat) ~ DRVEF2017_RV.Undergraduate.enrollment +
```

```
                   DRVIC2017.Tuition.and.fees..2016.17 +
                   total_enrollment,
                 train = comb.fryr.knn.train,
                 test = comb.fryr.knn.test,
                 k = 60)

# Predictions to be added to data frame in test data set
knn.comb.preds <- fitted(knn.comb1)
comb.fryr.test5 <- mutate(comb.fryr.knn.test, predFourcat = knn.comb.preds)

# Accuracy of k nearest neighbor model above
accuracy(comb.fryr.test5,
         truth = as.factor(fourcat),
         estimate = as.factor(predFourcat))


# K nearest neighbor with tuning parameter k = 100
set.seed(104)
knn.comb2 <- kknn(as.factor(fourcat) ~ DRVEF2017_RV.Undergraduate.enrollment +
                   DRVIC2017.Tuition.and.fees..2016.17 +
                   total_enrollment,
                 train = comb.fryr.knn.train,
                 test = comb.fryr.knn.test,
                 k = 100)

# Predictions to be added to data frame in test data set
knn.comb.preds <- fitted(knn.comb2)
comb.fryr.test6 <- mutate(comb.fryr.knn.test, predFourcat = knn.comb.preds)

# Accuracy of k nearest neighbor model above
accuracy(comb.fryr.test6,
         truth = as.factor(fourcat),
         estimate = as.factor(predFourcat))


################################################################################
#                                    Task Two                                  #
################################################################################

# K cluster using DRVEF2017_RV.Undergraduate.enrollment,
# EF2017D_RV.Student.to.faculty.ratio, total_enrollment,
# DRVGR2017_RV.Graduation.rate.total.cohort, and
# col_white

# Selecting columns to use in cluster
fryr.clust <-
  select(comb.fryr,
         DRVEF2017_RV.Undergraduate.enrollment,
         EF2017D_RV.Student.to.faculty.ratio,
         total_enrollment,
         DRVGR2017_RV.Graduation.rate..total.cohort,
         col_white)
```

```r
# Rescaling each column to fit on same scale
fryr.clust$DRVEF2017_RV.Undergraduate.enrollment <-
  rescale(x = fryr.clust$DRVEF2017_RV.Undergraduate.enrollment,
          to = c(0, 1),
          from = range(fryr.clust$DRVEF2017_RV.Undergraduate.enrollment,
                       na.rm = TRUE, finite = TRUE))

fryr.clust$EF2017D_RV.Student.to.faculty.ratio <-
  rescale(x = fryr.clust$EF2017D_RV.Student.to.faculty.ratio,
          to = c(0, 1),
          from = range(fryr.clust$EF2017D_RV.Student.to.faculty.ratio,
                       na.rm = TRUE, finite = TRUE))

fryr.clust$total_enrollment <-
  rescale(x = fryr.clust$total_enrollment,
          to = c(0, 1),
          from = range(fryr.clust$total_enrollment,
                       na.rm = TRUE, finite = TRUE))

fryr.clust$DRVGR2017_RV.Graduation.rate..total.cohort <-
  rescale(x = fryr.clust$DRVGR2017_RV.Graduation.rate..total.cohort,
          to = c(0, 1),
          from = range(fryr.clust$DRVGR2017_RV.Graduation.rate..total.cohort,
                       na.rm = TRUE, finite = TRUE))

fryr.clust$col_white <-
  rescale(x = fryr.clust$col_white,
          to = c(0, 1),
          from = range(fryr.clust$col_white,
                       na.rm = TRUE, finite = TRUE))


# Clustering variables to try to fit fourcat in normal data set --> comb.fryr
# Using k = 5
set.seed(675)
fryr_kmclust <- kmeans(na.omit(fryr.clust), centers = 5)
fryr_kmclust
```