資料科學 - 作業三

tags: `Data Science`

系級：電機所碩一　　姓名：楊冠彥　　學號：R11921091

Problem 1. One-by-one Feature Selection (30%)

Do simple feature selection using naive one-by-one selection and answer the following questions in your report. (The formula in the lecture note is permitted.)

- ### Describe your feature selection method. (10%)

在特徵選擇部分，我使用的是 sklearn中的單變量選擇法做為naive one-by-one selection，單變量選擇法的主要想法是單獨檢查每個特徵以確定特徵與對應變量的關係強度。透過某些統計檢驗的方法分別對每個變量進行檢驗，得到一組分數、p-value值，之後在排序選擇分數最高(或p-value最小等)的那些特徵。而單變量選擇法是在特徵選擇的探索階段最適合使用的方法。

sklearn為各種統計檢驗方法和不同的排序選擇標準提供了多種工具，包括用於迴歸問題的 `f_regression`、`r_regression`，用於分類問題的`chi2`、`f_classif`；以及用於特徵排序和選擇的 `SelectKBest`、`SelectFpr`、`SelectFdr`等。

而本次使用SelectKBest來選擇前600個feature搭配Chi-squared，Chi-squared能夠分類任務的非負特徵，公式如下：

$$Chi_{square\_score}(f_i) = \sum_{j=1}^{r} \sum_{s=1}^{c} \frac{(n_{js} - \mu_{js})^2}{\mu_{js}}$$

where r = # of feature values, c = # of classes, n_{js} = # of instances with feature value j and class s, μ_{js} = expected # of instances with feature value j and class s.

- ### Show your result and code of the feature selection. (How many features are selected? Which features are selected? Performance changes for the classifier? Etc.) (20%)

Result

在feature selection階段選擇了做完卡方檢定後最好的前600個feature，同時我也將這些特徵輸出如下：

```
After selecting best 600 features: (62, 600)
Number of features: 600
Select top Feature:
```

```
['Hsa.3004', 'Hsa.13491', 'Hsa.13491', 'Hsa.37254', 'Hsa.541', 'Hsa.20836',
'Hsa.1977', 'Hsa.44472', 'Hsa.3087', 'Hsa.1447', 'Hsa.750', 'Hsa.45293',
'Hsa.2555', 'Hsa.467', 'Hsa.539', 'Hsa.2357', 'Hsa.474', 'Hsa.24464',
'Hsa.2597', 'Hsa.3835', 'Hsa.6080', 'Hsa.3002', 'Hsa.1119', 'Hsa.44472',
'Hsa.4689', 'Hsa.45293', 'Hsa.5710', 'Hsa.20836', 'Hsa.878', 'Hsa.1648',
'Hsa.1836', 'Hsa.2948', 'Hsa.7877', 'HSAC07', 'HSAC07', 'HSAC07', 'HSAC07',
'Hsa.8068', 'Hsa.1139', 'Hsa.3006', 'Hsa.5398', 'Hsa.1013', 'UMGAP', 'UMGAP',
'UMGAP', 'UMGAP', 'Hsa.2221', 'Hsa.98', 'Hsa.2019', 'Hsa.1732', 'Hsa.5363',
'Hsa.5444', 'Hsa.3094', 'Hsa.1949', 'Hsa.361', 'Hsa.8125', 'Hsa.2361',
'Hsa.255', 'Hsa.2699', 'Hsa.1013', 'Hsa.2542', 'Hsa.678', 'Hsa.36957',
'Hsa.44403', 'Hsa.2800', 'Hsa.9304', 'Hsa.36957', 'Hsa.33965', 'Hsa.2665',
'Hsa.954', 'Hsa.27685', 'Hsa.20', 'Hsa.489', 'Hsa.5292', 'Hsa.1985',
'Hsa.3566', 'Hsa.1978', 'Hsa.3409', 'Hsa.5346', 'Hsa.2582', 'Hsa.11712',
'Hsa.10755', 'Hsa.68', 'Hsa.4316', 'Hsa.914', 'Hsa.8126', 'Hsa.1737',
'Hsa.1732', 'Hsa.4252', 'Hsa.572', 'Hsa.73', 'Hsa.537', 'Hsa.36694',
'Hsa.957', 'Hsa.668', 'Hsa.451', 'Hsa.810', 'Hsa.2071', 'Hsa.558',
'Hsa.1222', 'Hsa.3017', 'Hsa.3409', 'Hsa.652', 'Hsa.63', 'Hsa.3197',
'Hsa.22614', 'Hsa.45604', 'Hsa.43252', 'Hsa.5821', 'Hsa.733', 'Hsa.31',
'Hsa.22614', 'Hsa.6146', 'Hsa.45604', 'Hsa.18664', 'Hsa.8831', 'Hsa.10510',
'Hsa.120', 'Hsa.1137', 'Hsa.2831', 'Hsa.2015', 'Hsa.832', 'Hsa.1116',
'Hsa.2773', 'Hsa.39141', 'Hsa.1822', 'Hsa.24948', 'Hsa.3003', 'Hsa.812',
'Hsa.8305', 'Hsa.1130', 'Hsa.7498', 'Hsa.338', 'Hsa.2846', 'Hsa.9817',
'Hsa.2710', 'Hsa.587', 'Hsa.3295', 'Hsa.778', 'Hsa.2014', 'Hsa.1190',
'Hsa.4992', 'Hsa.1006', 'Hsa.9277', 'Hsa.9817', 'Hsa.951', 'Hsa.692',
'Hsa.8147', 'Hsa.1532', 'Hsa.41315', 'Hsa.921', 'Hsa.823', 'i', 'i', 'i',
'i', 'Hsa.996', 'Hsa.9372', 'Hsa.33572', 'Hsa.692', 'Hsa.38375', 'Hsa.81',
'Hsa.2933', 'Hsa.539', 'Hsa.10358', 'Hsa.8658', 'Hsa.41126', 'Hsa.3001',
'Hsa.3031', 'Hsa.80', 'Hsa.29228', 'Hsa.20034', 'Hsa.43279', 'Hsa.1050',
'Hsa.9407', 'Hsa.2829', 'Hsa.2795', 'Hsa.994', 'Hsa.19784', 'Hsa.5544',
'Hsa.3180', 'Hsa.1534', 'Hsa.1166', 'Hsa.2779', 'Hsa.1722', 'Hsa.26914',
'Hsa.41279', 'Hsa.891', 'Hsa.3272', 'Hsa.256', 'Hsa.41347', 'Hsa.1044',
'Hsa.2917', 'Hsa.698', 'Hsa.103', 'Hsa.2950', 'Hsa.34874', 'Hsa.10176',
'Hsa.2126', 'Hsa.2809', 'Hsa.31', 'Hsa.153', 'Hsa.594', 'Hsa.562',
'Hsa.3910', 'Hsa.821', 'Hsa.2251', 'Hsa.1277', 'Hsa.36689', 'Hsa.7671',
'Hsa.41124', 'Hsa.538', 'Hsa.2440', 'Hsa.3152', 'Hsa.1278', 'Hsa.1121',
'Hsa.5971', 'Hsa.1504', 'Hsa.41282', 'Hsa.19731', 'Hsa.1221', 'Hsa.2086',
'Hsa.24506', 'Hsa.2249', 'Hsa.544', 'Hsa.3207', 'Hsa.714', 'Hsa.1473',
'Hsa.879', 'Hsa.929', 'Hsa.702', 'Hsa.3157', 'Hsa.2560', 'Hsa.35528',
'Hsa.1990', 'Hsa.8583', 'Hsa.2588', 'Hsa.36291', 'Hsa.2612', 'Hsa.1542',
'Hsa.592', 'Hsa.59', 'Hsa.3088', 'Hsa.41875', 'Hsa.37937', 'Hsa.27686',
'Hsa.17101', 'Hsa.5464', 'Hsa.561', 'Hsa.1867', 'Hsa.19232', 'Hsa.3185',
'Hsa.831', 'Hsa.6472', 'Hsa.949', 'Hsa.3454', 'Hsa.2379', 'Hsa.3007',
'Hsa.36657', 'Hsa.36665', 'Hsa.286', 'Hsa.5633', 'Hsa.4996', 'Hsa.12209',
'Hsa.561', 'Hsa.26083', 'Hsa.2191', 'Hsa.1591', 'Hsa.726', 'Hsa.1098',
'Hsa.8219', 'Hsa.470', 'Hsa.41164', 'Hsa.1165', 'Hsa.2780', 'Hsa.3209',
'Hsa.853', 'Hsa.2487', 'Hsa.1244', 'Hsa.1207', 'Hsa.8010', 'Hsa.17822',
'Hsa.6422', 'Hsa.950', 'Hsa.1773', 'Hsa.847', 'Hsa.1132', 'Hsa.1106',
'Hsa.9251', 'Hsa.39288', 'Hsa.42949', 'Hsa.6458', 'Hsa.305', 'Hsa.3306',
'Hsa.28914', 'Hsa.3194', 'Hsa.9691', 'Hsa.45499', 'Hsa.465', 'Hsa.1515',
'Hsa.22762', 'Hsa.318', 'Hsa.168', 'Hsa.7395', 'Hsa.1485', 'Hsa.285',
'Hsa.2646', 'Hsa.3517', 'Hsa.2966', 'Hsa.10047', 'Hsa.1036', 'Hsa.1715',
'Hsa.852', 'Hsa.9972', 'Hsa.38375', 'Hsa.109', 'Hsa.465', 'Hsa.2179',
'Hsa.733', 'Hsa.612', 'Hsa.1284', 'Hsa.154', 'Hsa.61', 'Hsa.2856',
'Hsa.1806', 'Hsa.3305', 'Hsa.2372', 'Hsa.4251', 'Hsa.3239', 'Hsa.1567',
'Hsa.3067', 'Hsa.692', 'Hsa.465', 'Hsa.2985', 'Hsa.1726', 'Hsa.773',
'Hsa.7852', 'Hsa.2513', 'Hsa.1854', 'Hsa.1516', 'Hsa.1272', 'Hsa.18401',
'Hsa.1433', 'Hsa.695', 'Hsa.680', 'Hsa.2863', 'Hsa.3349', 'Hsa.316',
'Hsa.41280', 'Hsa.287', 'Hsa.1131', 'Hsa.41280', 'Hsa.14842', 'Hsa.3803',
```

```
'Hsa.36685', 'Hsa.8374', 'Hsa.2778', 'Hsa.884', 'Hsa.34914', 'Hsa.24279',
'Hsa.140', 'Hsa.26528', 'Hsa.21379', 'Hsa.3212', 'Hsa.2747', 'Hsa.895',
'Hsa.36952', 'Hsa.742', 'Hsa.3068', 'Hsa.2725', 'Hsa.1610', 'Hsa.57',
'Hsa.3278', 'Hsa.21847', 'Hsa.10706', 'Hsa.3876', 'Hsa.9816', 'Hsa.6617',
'Hsa.2586', 'Hsa.3331', 'Hsa.5756', 'Hsa.2060', 'Hsa.37541', 'Hsa.3115',
'Hsa.1127', 'Hsa.2316', 'Hsa.1073', 'Hsa.3348', 'Hsa.9218', 'Hsa.1205',
'Hsa.146', 'Hsa.4937', 'Hsa.31801', 'Hsa.44595', 'Hsa.2092', 'Hsa.2157',
'Hsa.23249', 'Hsa.717', 'Hsa.549', 'Hsa.1714', 'Hsa.602', 'Hsa.1588',
'Hsa.2051', 'Hsa.579', 'Hsa.9856', 'Hsa.462', 'Hsa.2503', 'Hsa.1088',
'Hsa.490', 'Hsa.1347', 'Hsa.40063', 'Hsa.1043', 'Hsa.2152', 'Hsa.1331',
'Hsa.94', 'Hsa.41338', 'Hsa.25481', 'Hsa.39753', 'Hsa.1030', 'Hsa.2152',
'Hsa.3648', 'Hsa.612', 'Hsa.7048', 'Hsa.1047', 'Hsa.229', 'Hsa.11572',
'Hsa.2436', 'Hsa.37553', 'Hsa.816', 'Hsa.60', 'Hsa.6910', 'Hsa.103',
'Hsa.1171', 'Hsa.23124', 'Hsa.42625', 'Hsa.6030', 'Hsa.7203', 'Hsa.2962',
'Hsa.1994', 'Hsa.8214', 'Hsa.3010', 'Hsa.2997', 'Hsa.1670', 'Hsa.8040',
'Hsa.2591', 'Hsa.2553', 'Hsa.1617', 'Hsa.2451', 'Hsa.5211', 'Hsa.36655',
'Hsa.8781', 'Hsa.2774', 'Hsa.35471', 'Hsa.2549', 'Hsa.1682', 'Hsa.1280',
'Hsa.2793', 'Hsa.2847', 'Hsa.421', 'Hsa.11673', 'Hsa.41187', 'Hsa.43894',
'Hsa.975', 'Hsa.367', 'Hsa.2483', 'Hsa.2471', 'Hsa.2996', 'Hsa.3016',
'Hsa.902', 'Hsa.7', 'Hsa.7462', 'Hsa.17564', 'Hsa.5392', 'Hsa.41247',
'Hsa.341', 'Hsa.2827', 'Hsa.2967', 'Hsa.19843', 'Hsa.2789', 'Hsa.1902',
'Hsa.25522', 'Hsa.663', 'Hsa.3263', 'Hsa.2715', 'Hsa.2867', 'Hsa.31630',
'Hsa.477', 'Hsa.1477', 'Hsa.27537', 'Hsa.1832', 'Hsa.1464', 'Hsa.612',
'Hsa.6288', 'Hsa.329', 'Hsa.29817', 'Hsa.1254', 'Hsa.56', 'Hsa.6619',
'Hsa.654', 'Hsa.3252', 'Hsa.1410', 'Hsa.25536', 'Hsa.2551', 'Hsa.108',
'Hsa.491', 'Hsa.2344', 'Hsa.2808', 'Hsa.2818', 'Hsa.6104', 'Hsa.1045',
'Hsa.2959', 'Hsa.2964', 'Hsa.1140', 'Hsa.9353', 'Hsa.24582', 'Hsa.2210',
'Hsa.2280', 'Hsa.1731', 'Hsa.21562', 'Hsa.121', 'Hsa.3141', 'Hsa.1380',
'Hsa.28939', 'Hsa.2928', 'Hsa.3093', 'Hsa.2091', 'Hsa.9994', 'Hsa.18790',
'Hsa.10664', 'Hsa.2097', 'Hsa.9744', 'Hsa.33268', 'Hsa.662', 'Hsa.1240',
'Hsa.20376', 'Hsa.3230', 'Hsa.305', 'Hsa.45446', 'Hsa.2458', 'Hsa.1592',
'Hsa.687', 'Hsa.41346', 'Hsa.627', 'Hsa.1579', 'Hsa.14069', 'Hsa.3083',
'Hsa.2821', 'Hsa.1380', 'Hsa.38171', 'Hsa.3316', 'Hsa.960', 'Hsa.2688',
'Hsa.2645', 'Hsa.584', 'Hsa.3254', 'Hsa.23124', 'Hsa.8223', 'Hsa.2250',
'Hsa.601', 'Hsa.6814', 'Hsa.16296', 'Hsa.2337', 'Hsa.2794', 'Hsa.2753',
'Hsa.23824', 'Hsa.2199', 'Hsa.41283', 'Hsa.1039', 'Hsa.24490', 'Hsa.1387',
'Hsa.3296', 'Hsa.2291', 'Hsa.17514', 'Hsa.41323', 'Hsa.1660', 'Hsa.404',
'Hsa.25322', 'Hsa.2196', 'Hsa.2456', 'Hsa.2939', 'Hsa.8192', 'Hsa.466',
'Hsa.2705', 'Hsa.2644', 'Hsa.7652', 'Hsa.1763', 'Hsa.1423', 'Hsa.9174',
'Hsa.5908', 'Hsa.72', 'Hsa.3250', 'Hsa.1143', 'Hsa.36696', 'Hsa.31500',
'Hsa.11616', 'Hsa.1435', 'Hsa.33', 'Hsa.58', 'Hsa.1198', 'Hsa.41260']
```

做完SelectKBest後因為原本sample code就有feature evaluation部分，所以就將feature做個排序再把ranking_idx全部餵進去就可以了，至於feature evaluation的Classifier我測試了Decision Tree Classifier。

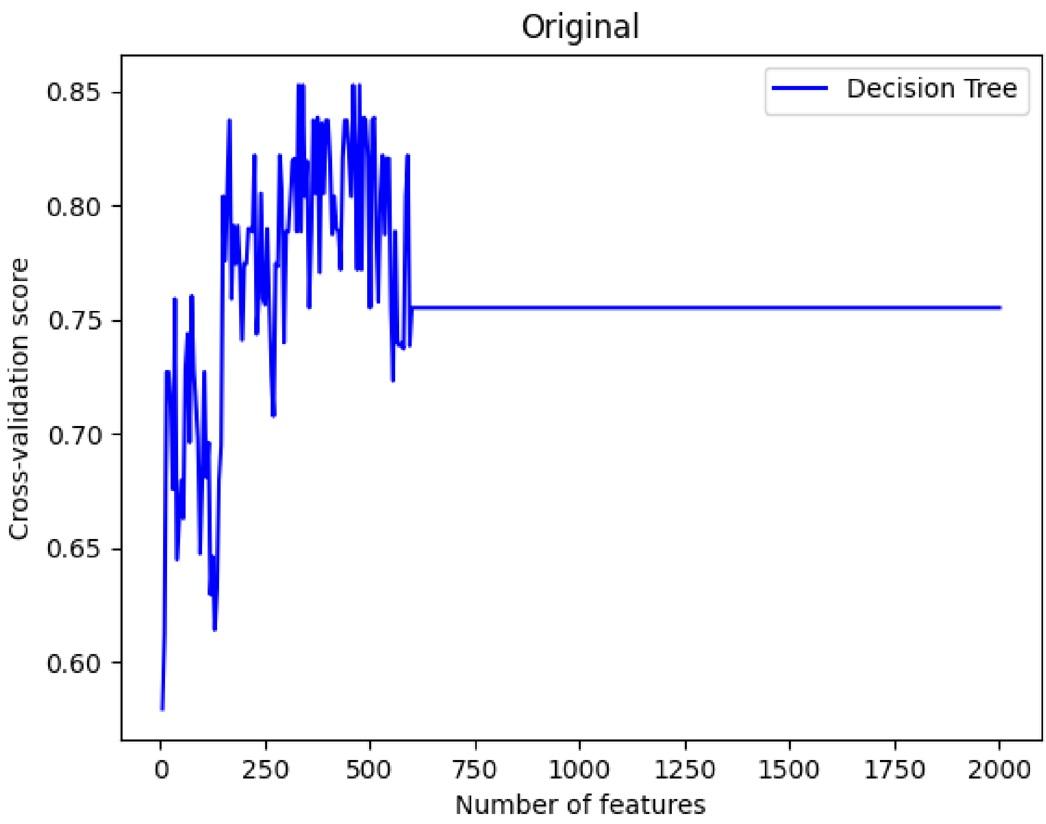
feature evaluation使用DecisionTreeClassifier跑到第330個特徵時找到最好的結果，輸出結果如下：

```
Max of Decision Tree: 0.8525641025641025
Number of features: 330
```

並且我將這些特徵全部輸出出來，如下：

```
Feature:
['Hsa.3004', 'Hsa.13491', 'Hsa.13491', 'Hsa.37254', 'Hsa.541', 'Hsa.20836',
'Hsa.1977', 'Hsa.44472', 'Hsa.3087', 'Hsa.1447', 'Hsa.750', 'Hsa.45293',
'Hsa.2555', 'Hsa.467', 'Hsa.539', 'Hsa.2357', 'Hsa.474', 'Hsa.24464',
'Hsa.2597', 'Hsa.3835', 'Hsa.6080', 'Hsa.3002', 'Hsa.1119', 'Hsa.44472',
'Hsa.4689', 'Hsa.45293', 'Hsa.5710', 'Hsa.20836', 'Hsa.878', 'Hsa.1648',
'Hsa.1836', 'Hsa.2948', 'Hsa.7877', 'HSAC07', 'HSAC07', 'HSAC07', 'HSAC07',
'Hsa.8068', 'Hsa.1139', 'Hsa.3006', 'Hsa.5398', 'Hsa.1013', 'UMGAP', 'UMGAP',
'UMGAP', 'UMGAP', 'Hsa.2221', 'Hsa.98', 'Hsa.2019', 'Hsa.1732', 'Hsa.5363',
'Hsa.5444', 'Hsa.3094', 'Hsa.1949', 'Hsa.361', 'Hsa.8125', 'Hsa.2361',
'Hsa.255', 'Hsa.2699', 'Hsa.1013', 'Hsa.2542', 'Hsa.678', 'Hsa.36957',
'Hsa.44403', 'Hsa.2800', 'Hsa.9304', 'Hsa.36957', 'Hsa.33965', 'Hsa.2665',
'Hsa.954', 'Hsa.27685', 'Hsa.20', 'Hsa.489', 'Hsa.5292', 'Hsa.1985',
'Hsa.3566', 'Hsa.1978', 'Hsa.3409', 'Hsa.5346', 'Hsa.2582', 'Hsa.11712',
'Hsa.10755', 'Hsa.68', 'Hsa.4316', 'Hsa.914', 'Hsa.8126', 'Hsa.1737',
'Hsa.1732', 'Hsa.4252', 'Hsa.572', 'Hsa.73', 'Hsa.537', 'Hsa.36694',
'Hsa.957', 'Hsa.668', 'Hsa.451', 'Hsa.810', 'Hsa.2071', 'Hsa.558',
'Hsa.1222', 'Hsa.3017', 'Hsa.3409', 'Hsa.652', 'Hsa.63', 'Hsa.3197',
'Hsa.22614', 'Hsa.45604', 'Hsa.43252', 'Hsa.5821', 'Hsa.733', 'Hsa.31',
'Hsa.22614', 'Hsa.6146', 'Hsa.45604', 'Hsa.18664', 'Hsa.8831', 'Hsa.10510',
'Hsa.120', 'Hsa.1137', 'Hsa.2831', 'Hsa.2015', 'Hsa.832', 'Hsa.1116',
'Hsa.2773', 'Hsa.39141', 'Hsa.1822', 'Hsa.24948', 'Hsa.3003', 'Hsa.812',
'Hsa.8305', 'Hsa.1130', 'Hsa.7498', 'Hsa.338', 'Hsa.2846', 'Hsa.9817',
'Hsa.2710', 'Hsa.587', 'Hsa.3295', 'Hsa.778', 'Hsa.2014', 'Hsa.1190',
'Hsa.4992', 'Hsa.1006', 'Hsa.9277', 'Hsa.9817', 'Hsa.951', 'Hsa.692',
'Hsa.8147', 'Hsa.1532', 'Hsa.41315', 'Hsa.921', 'Hsa.823', 'i', 'i', 'i',
'i', 'Hsa.996', 'Hsa.9372', 'Hsa.33572', 'Hsa.692', 'Hsa.38375', 'Hsa.81',
'Hsa.2933', 'Hsa.539', 'Hsa.10358', 'Hsa.8658', 'Hsa.41126', 'Hsa.3001',
'Hsa.3031', 'Hsa.80', 'Hsa.29228', 'Hsa.20034', 'Hsa.43279', 'Hsa.1050',
'Hsa.9407', 'Hsa.2829', 'Hsa.2795', 'Hsa.994', 'Hsa.19784', 'Hsa.5544',
'Hsa.3180', 'Hsa.1534', 'Hsa.1166', 'Hsa.2779', 'Hsa.1722', 'Hsa.26914',
'Hsa.41279', 'Hsa.891', 'Hsa.3272', 'Hsa.256', 'Hsa.41347', 'Hsa.1044',
'Hsa.2917', 'Hsa.698', 'Hsa.103', 'Hsa.2950', 'Hsa.34874', 'Hsa.10176',
'Hsa.2126', 'Hsa.2809', 'Hsa.31', 'Hsa.153', 'Hsa.594', 'Hsa.562',
'Hsa.3910', 'Hsa.821', 'Hsa.2251', 'Hsa.1277', 'Hsa.36689', 'Hsa.7671',
'Hsa.41124', 'Hsa.538', 'Hsa.2440', 'Hsa.3152', 'Hsa.1278', 'Hsa.1121',
'Hsa.5971', 'Hsa.1504', 'Hsa.41282', 'Hsa.19731', 'Hsa.1221', 'Hsa.2086',
'Hsa.24506', 'Hsa.2249', 'Hsa.544', 'Hsa.3207', 'Hsa.714', 'Hsa.1473',
'Hsa.879', 'Hsa.929', 'Hsa.702', 'Hsa.3157', 'Hsa.2560', 'Hsa.35528',
'Hsa.1990', 'Hsa.8583', 'Hsa.2588', 'Hsa.36291', 'Hsa.2612', 'Hsa.1542',
'Hsa.592', 'Hsa.59', 'Hsa.3088', 'Hsa.41875', 'Hsa.37937', 'Hsa.27686',
'Hsa.17101', 'Hsa.5464', 'Hsa.561', 'Hsa.1867', 'Hsa.19232', 'Hsa.3185',
'Hsa.831', 'Hsa.6472', 'Hsa.949', 'Hsa.3454', 'Hsa.2379', 'Hsa.3007',
'Hsa.36657', 'Hsa.36665', 'Hsa.286', 'Hsa.5633', 'Hsa.4996', 'Hsa.12209',
'Hsa.561', 'Hsa.26083', 'Hsa.2191', 'Hsa.1591', 'Hsa.726', 'Hsa.1098',
'Hsa.8219', 'Hsa.470', 'Hsa.41164', 'Hsa.1165', 'Hsa.2780', 'Hsa.3209',
'Hsa.853', 'Hsa.2487', 'Hsa.1244', 'Hsa.1207', 'Hsa.8010', 'Hsa.17822',
'Hsa.6422', 'Hsa.950', 'Hsa.1773', 'Hsa.847', 'Hsa.1132', 'Hsa.1106',
'Hsa.9251', 'Hsa.39288', 'Hsa.42949', 'Hsa.6458', 'Hsa.305', 'Hsa.3306',
'Hsa.28914', 'Hsa.3194', 'Hsa.9691', 'Hsa.45499', 'Hsa.465', 'Hsa.1515',
'Hsa.22762', 'Hsa.318', 'Hsa.168', 'Hsa.7395', 'Hsa.1485', 'Hsa.285',
'Hsa.2646', 'Hsa.3517', 'Hsa.2966', 'Hsa.10047', 'Hsa.1036', 'Hsa.1715',
'Hsa.852', 'Hsa.9972', 'Hsa.38375', 'Hsa.109', 'Hsa.465', 'Hsa.2179',
'Hsa.733', 'Hsa.612', 'Hsa.1284', 'Hsa.154', 'Hsa.61', 'Hsa.2856',
'Hsa.1806', 'Hsa.3305', 'Hsa.2372', 'Hsa.4251', 'Hsa.3239', 'Hsa.1567']
```

分類器的效能變化則用sample code視覺化的程式碼將圖表呈現，如下：



Source Code

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score
import sklearn
from sklearn.feature_selection import chi2, SelectKBest
from sklearn.ensemble import RandomForestClassifier

# ===========================================================
#                           Load Data
# ===========================================================
indexes = pd.read_csv('hw3_Data1/index.txt', delimiter = '\t', header =
None).T
x = pd.read_csv('hw3_Data1/gene.txt', delimiter = ' ', header =
None).to_numpy().T
y = pd.read_csv('hw3_Data1/label.txt', header = None).to_numpy()
y = (y>0).astype(int).reshape(y.shape[0])
index = indexes.iloc[0,].str.strip()
# print(y)
# print(np.shape(x))
# print(index)

# ===========================================================
#                           Load Data
```

```python
# ============================================================


# ============================================================
#                        Feature ranking
# ============================================================
# TODO: Design your score function for feature selection
# TODO: To use the provided evaluation sample code, you need to generate ranking_idx, which is the sorted index of feature
def get_feature(score_func, x, y):
    select = SelectKBest(score_func=score_func, k=600)
    z = select.fit_transform(x,y)
    # print(z)
    print("After selecting best 600 features:", np.shape(z))
    filter = select.get_support()
    # print(filter)
    return index[filter], filter

def get_ranking_idx(features):
    ranking_idx = []
    for i in range(2000):
        if features[i] == True:
            ranking_idx.append(i)
    return ranking_idx
f, filter = get_feature(chi2, x, y)
ranking_idx = get_ranking_idx(filter)
fo = index[ranking_idx]
fo1 = fo.tolist()
print(f"Number of features: {len(ranking_idx)}")
print("Select top 600 Feature:")
print(fo1)

# ============================================================
#                        Feature ranking
# ============================================================


# ============================================================
#                        Feature evaluation
# ============================================================
# Use a simple dicision tree with 5-fold validation to evaluate the feature selection result.
# You can try other classifier and hyperparameter.
score_history = []
for m in range(5, 2001, 5):
    # Select Top m feature
    x_subset = x[:, ranking_idx[:m]]

    # Build random forest
    clf = DecisionTreeClassifier(random_state=0)
    # clf = SVC(kernel='rbf', random_state=0) #build SVM
    # clf = RandomForestClassifier(random_state=0) #build RandomForest

    # Calculate validation score
    scores = cross_val_score(clf, x_subset, y, cv=5)

    # Save the score calculated with m feature
    score_history.append(scores.mean())
```

```python

# Report best accuracy.
num_feature = np.argmax(score_history)*5+5
f_new = index[ranking_idx[:num_feature]]
f_new_l = f_new.tolist()

# print("ranking_idx:")
# print(ranking_idx.tolist())
print(f"Max of Decision Tree: {max(score_history)}")
# print(f"Max of SVM: {max(score_history)}")
# print(f"Max of Radom Forest: {max(score_history)}")
print(f"Number of features: {num_feature}")
# print("Feature:")
# print(f_new_l)

# ==========================================================
#                         Feature evaluation
# ==========================================================


# ==========================================================
#                         Visualization
# ==========================================================
plt.plot(range(5, 2001, 5), score_history, c='blue')
plt.title('Original')
plt.xlabel('Number of features')
plt.ylabel('Cross-validation score')
plt.legend(['Decision Tree'])
# plt.legend(['SVM'])
# plt.legend(['Ramdom Forest'])
plt.savefig('3-1_result.png')

# ==========================================================
#                         Visualization
# ==========================================================
```

Problem 2. Subset-Based Feature Selection (40%)

Implement a subset-based feature selection using PSO (particle swarm optimization), SA (simulated annealing) or GA (genetic algorithm) heuristics and answer the following questions in your report.

- **Describe your algorithm, including: the metaheuristic you choose (SA, PSO or GA), your objective function, the tunable parameters and tunable algorithm components (besides the objective function/cost function module) in your metaheuristic. What are the specific values/methods you use for your tunable parameters and algorithm component(s), if any. (20%)**

我選擇 genetic algorithm作為metaheuristic。此外，我一律使用 DecisionTreeClassifier 作為我的 objective function。

我使用 sklearn-genetic-opt 這項套件中的GAFeatureSelectionCV，GAFeatureSelectionCV這個 funtion會根據我們給定的parameter在範圍內找出特徵選擇的演化最佳化結果。

GAFeatureSelectionCV在進行如下過程：
1) 隨機初始化種群 p
2) 確定種群的適應度
3) 直到收斂重複：
  a) 從母體中選擇父母，公式如下：

$$P(h_i) = \frac{F(h_i)}{\sum_{i=1}^{p} F(h_i)}$$

  b) 交叉產生新種群

  c) 對新種群進行變異

  d) 計算新種群的適應度

other parameters：

```
cv: 決定交叉驗證拆分策略。cv:5則拆成每5份去做驗證查看準確度。
scoring: 評估交叉驗證模型在測試集上效能的策略。
population_size: 對隨機產生的個體進行抽樣的初始種群規模。
generations: 運行進化算法的世代數或迭代數。
n_jobs: 要並行處理的作業數。
keep_top_k: 保留在 hof(hall of fame) 中的最佳解決方案的數量。
crossover_probability: 兩個個體之間交叉操作的機率。
mutation_probability: 發生子突變的機率。
tournament_size:進行tournament選擇的人數。
elitism: 如果 True 則將 tournament_size 最佳解決方案帶到下一代。
```

在執行了GASearchCV：

`code`

```
clf = DecisionTreeClassifier(random_state=0)

evolved_selector = GAFeatureSelectionCV(
    estimator=clf,
    cv=5,
    scoring="accuracy",
    population_size=10,
    generations=5,
    n_jobs=-1,
    verbose=True,
    keep_top_k=2,
```

```
    elitism=True,
    crossover_probability=0.2,
    mutation_probability=0.8,
    tournament_size=3,
)

# Train and select the features
evolved_selector.fit(x, y)
```

得到如下結果：

```
gen nevals  fitness     fitness_std fitness_max fitness_min
0   5       0.741026    0.042296    0.817949    0.703846
1   10      0.767949    0.0210974   0.803846    0.741026
2   10      0.786667    0.0248281   0.834615    0.770513
3   10      0.792821    0.0346638   0.834615    0.753846
4   10      0.790256    0.0366909   0.819231    0.720513
5   10      0.781538    0.0285595   0.837179    0.755128
6   10      0.807436    0.0241135   0.837179    0.771795
7   10      0.823846    0.00666667  0.837179    0.820513
8   10      0.827179    0.00816497  0.837179    0.820513
9   10      0.823333    0.016958    0.837179    0.802564
10  10      0.81641     0.016958    0.837179    0.802564
```

經過 genetic algorithm 找出好的基因後，使用套件的method，如下：

```
# Get best feature
features = evolved_selector.best_features_
```

再使用原本sample code就有的feature evaluation部分，先做個索引排序再把ranking_idx全部餵進去就可以了，至於feature evaluation我測試了Decision Tree Classifier、SVM 和 Random Forest。

- **Show your result and code of the feature selection. (How many features are selected? Which features are selected? Performance changes for the classifier? Etc.) (20%)**

Result

在feature selection階段Genetic algorithm選擇了最好的962個feature，同時我也將這些特徵輸出如下：

```
After GA:
Number of features: 962
Feature selection after GA:
```

```
['Hsa.13491', 'Hsa.37254', 'Hsa.474', 'Hsa.6080', 'Hsa.45293', 'Hsa.5710',
'Hsa.909', 'Hsa.1836', 'Hsa.1273', 'Hsa.19249', 'HSAC07', 'HSAC07', 'HSAC07',
'HSAC07', 'Hsa.1139', 'Hsa.2299', 'UMGAP', 'Hsa.1238', 'Hsa.6039', 'Hsa.98',
'Hsa.5363', 'Hsa.8093', 'Hsa.5444', 'Hsa.3094', 'Hsa.8125', 'Hsa.1013',
'Hsa.2542', 'Hsa.678', 'Hsa.36957', 'Hsa.6977', 'Hsa.24121', 'Hsa.36957',
'Hsa.938', 'Hsa.33965', 'Hsa.2665', 'Hsa.20', 'Hsa.12257', 'Hsa.2026',
'Hsa.5292', 'Hsa.32730', 'Hsa.1985', 'Hsa.10363', 'Hsa.624', 'Hsa.2582',
'Hsa.11712', 'Hsa.10755', 'Hsa.2600', 'Hsa.68', 'Hsa.2918', 'Hsa.7280',
'Hsa.2929', 'Hsa.21232', 'Hsa.1961', 'Hsa.916', 'Hsa.1732', 'Hsa.13183',
'Hsa.572', 'Hsa.45260', 'Hsa.73', 'Hsa.3129', 'Hsa.34312', 'Hsa.537',
'Hsa.36694', 'Hsa.957', 'Hsa.668', 'Hsa.10975', 'Hsa.2071', 'Hsa.91',
'Hsa.558', 'Hsa.1222', 'Hsa.3017', 'Hsa.1829', 'Hsa.1367', 'Hsa.3197',
'Hsa.22614', 'Hsa.11096', 'Hsa.10306', 'Hsa.43252', 'Hsa.31', 'Hsa.6146',
'Hsa.21418', 'Hsa.18664', 'Hsa.25451', 'Hsa.8831', 'Hsa.10510', 'Hsa.2831',
'Hsa.1116', 'Hsa.39141', 'Hsa.6376', 'Hsa.580', 'Hsa.24948', 'Hsa.24652',
'Hsa.3003', 'Hsa.812', 'Hsa.8305', 'Hsa.15115', 'Hsa.3922', 'Hsa.41218',
'Hsa.2710', 'Hsa.307', 'Hsa.587', 'Hsa.3547', 'Hsa.3295', 'Hsa.4954',
'Hsa.1939', 'Hsa.568', 'Hsa.4992', 'Hsa.1006', 'Hsa.9817', 'Hsa.21339',
'Hsa.100', 'Hsa.8656', 'Hsa.951', 'Hsa.8606', 'Hsa.13102', 'Hsa.8177',
'Hsa.8147', 'Hsa.41315', 'Hsa.921', 'Hsa.628', 'Hsa.2714', 'Hsa.2613',
'Hsa.823', 'i', 'i', 'Hsa.996', 'Hsa.33572', 'Hsa.2806', 'Hsa.38375',
'Hsa.5122', 'Hsa.10358', 'Hsa.854', 'Hsa.80', 'Hsa.29228', 'Hsa.20034',
'Hsa.1050', 'Hsa.39432', 'Hsa.2829', 'Hsa.2795', 'Hsa.6165', 'Hsa.19784',
'Hsa.1178', 'Hsa.26767', 'Hsa.3180', 'Hsa.2902', 'Hsa.3307', 'Hsa.19003',
'Hsa.41347', 'Hsa.1044', 'Hsa.13598', 'Hsa.1738', 'Hsa.17649', 'Hsa.2950',
'Hsa.34874', 'Hsa.2598', 'Hsa.1276', 'Hsa.1450', 'Hsa.1096', 'Hsa.11096',
'Hsa.2809', 'Hsa.1466', 'Hsa.35496', 'Hsa.31', 'Hsa.153', 'Hsa.3910',
'Hsa.194', 'Hsa.995', 'Hsa.821', 'Hsa.848', 'Hsa.9631', 'Hsa.2600',
'Hsa.1277', 'Hsa.13702', 'Hsa.26747', 'Hsa.2505', 'Hsa.2440', 'Hsa.3152',
'Hsa.5548', 'Hsa.27832', 'Hsa.1008', 'Hsa.1670', 'Hsa.28145', 'Hsa.16100',
'Hsa.454', 'Hsa.9667', 'Hsa.2995', 'Hsa.11780', 'Hsa.19731', 'Hsa.244',
'Hsa.1221', 'Hsa.34351', 'Hsa.1033', 'Hsa.30128', 'Hsa.5143', 'Hsa.8736',
'Hsa.544', 'Hsa.714', 'Hsa.1108', 'Hsa.879', 'Hsa.929', 'Hsa.13624',
'Hsa.2592', 'Hsa.2232', 'Hsa.41083', 'Hsa.3299', 'Hsa.4234', 'Hsa.2560',
'Hsa.3328', 'Hsa.551', 'Hsa.35528', 'Hsa.3852', 'Hsa.1990', 'Hsa.8583',
'Hsa.27721', 'Hsa.6581', 'Hsa.1896', 'Hsa.22251', 'Hsa.591', 'Hsa.10067',
'Hsa.36291', 'Hsa.479', 'Hsa.2163', 'Hsa.592', 'Hsa.40177', 'Hsa.59',
'Hsa.13670', 'Hsa.2268', 'Hsa.3088', 'Hsa.41875', 'Hsa.2311', 'Hsa.27686',
'Hsa.2119', 'Hsa.2529', 'Hsa.39', 'Hsa.561', 'Hsa.3909', 'Hsa.39809',
'Hsa.3185', 'Hsa.50', 'Hsa.831', 'Hsa.9285', 'Hsa.6472', 'Hsa.3454',
'Hsa.2379', 'Hsa.2951', 'Hsa.16793', 'Hsa.934', 'Hsa.36657', 'a.1000',
'Hsa.286', 'Hsa.215', 'Hsa.2132', 'Hsa.24206', 'Hsa.1948', 'Hsa.26083',
'Hsa.2191', 'Hsa.726', 'Hsa.962', 'Hsa.1793', 'Hsa.37169', 'Hsa.1165',
'Hsa.2094', 'Hsa.2780', 'Hsa.23285', 'Hsa.6376', 'Hsa.3209', 'Hsa.853',
'Hsa.1133', 'Hsa.2141', 'Hsa.2487', 'Hsa.1244', 'Hsa.1207', 'Hsa.834',
'Hsa.1768', 'Hsa.1640', 'Hsa.3150', 'Hsa.36019', 'Hsa.2062', 'Hsa.17822',
'Hsa.6422', 'Hsa.20028', 'Hsa.847', 'Hsa.1132', 'Hsa.1106', 'Hsa.2478',
'Hsa.1630', 'Hsa.6376', 'Hsa.2416', 'Hsa.897', 'Hsa.1739', 'Hsa.22167',
'Hsa.39288', 'Hsa.3022', 'Hsa.2304', 'Hsa.6458', 'Hsa.3306', 'Hsa.7324',
'Hsa.36010', 'Hsa.9691', 'Hsa.3135', 'Hsa.45499', 'Hsa.36666', 'Hsa.2298',
'Hsa.318', 'Hsa.3098', 'Hsa.5142', 'Hsa.27592', 'Hsa.2007', 'Hsa.7395',
'Hsa.1485', 'Hsa.400', 'Hsa.845', 'Hsa.1668', 'Hsa.347', 'Hsa.285',
'Hsa.2646', 'Hsa.28193', 'Hsa.3517', 'Hsa.2156', 'Hsa.3159', 'Hsa.3820',
'Hsa.1486', 'Hsa.37058', 'Hsa.2830', 'Hsa.2966', 'Hsa.1036', 'Hsa.1715',
'Hsa.12892', 'Hsa.2568', 'Hsa.2840', 'Hsa.14896', 'Hsa.978', 'Hsa.41163',
'Hsa.9972', 'Hsa.3260', 'Hsa.896', 'Hsa.12754', 'Hsa.43155', 'Hsa.3136',
'Hsa.321', 'Hsa.34384', 'Hsa.2567', 'Hsa.37038', 'Hsa.31933', 'Hsa.61',
'Hsa.5120', 'Hsa.893', 'Hsa.3305', 'Hsa.10171', 'Hsa.11340', 'Hsa.2372',
```

```
'Hsa.3164', 'Hsa.21369', 'Hsa.741', 'Hsa.14360', 'Hsa.2317', 'Hsa.2634',
'Hsa.4251', 'Hsa.3239', 'Hsa.1537', 'Hsa.25745', 'Hsa.3067', 'Hsa.692',
'Hsa.2884', 'Hsa.2135', 'Hsa.6048', 'Hsa.2985', 'Hsa.1726', 'Hsa.7852',
'Hsa.3926', 'Hsa.2513', 'Hsa.1854', 'Hsa.3353', 'Hsa.1259', 'Hsa.1272',
'Hsa.3040', 'Hsa.9285', 'Hsa.6596', 'Hsa.3349', 'Hsa.3145', 'Hsa.41280',
'Hsa.2650', 'Hsa.2801', 'Hsa.18513', 'Hsa.287', 'Hsa.8299', 'Hsa.1131',
'Hsa.3099', 'Hsa.41280', 'Hsa.487', 'Hsa.3803', 'Hsa.1385', 'Hsa.646',
'Hsa.459', 'Hsa.8374', 'Hsa.2778', 'Hsa.2469', 'Hsa.2955', 'Hsa.1694',
'Hsa.10516', 'Hsa.12893', 'Hsa.42738', 'Hsa.17030', 'Hsa.2769', 'Hsa.24048',
'Hsa.27738', 'Hsa.14453', 'Hsa.9153', 'Hsa.2238', 'Hsa.41169', 'Hsa.11932',
'Hsa.44397', 'Hsa.9025', 'Hsa.45192', 'Hsa.140', 'Hsa.3138', 'Hsa.26528',
'Hsa.1840', 'Hsa.38203', 'Hsa.946', 'Hsa.26719', 'Hsa.3212', 'Hsa.2654',
'Hsa.13023', 'Hsa.8983', 'Hsa.1015', 'Hsa.2089', 'Hsa.44036', 'Hsa.1317',
'Hsa.742', 'Hsa.8096', 'Hsa.2725', 'Hsa.1089', 'Hsa.2409', 'Hsa.1419',
'Hsa.364', 'Hsa.116', 'Hsa.1309', 'Hsa.2565', 'Hsa.1598', 'Hsa.35955',
'Hsa.2348', 'Hsa.57', 'Hsa.24829', 'Hsa.4919', 'Hsa.3278', 'Hsa.1177',
'Hsa.471', 'Hsa.27648', 'Hsa.25142', 'Hsa.21847', 'Hsa.2881', 'Hsa.959',
'Hsa.3148', 'Hsa.11675', 'Hsa.2475', 'Hsa.6617', 'Hsa.36683', 'Hsa.8194',
'Hsa.3331', 'Hsa.7554', 'Hsa.11839', 'Hsa.447', 'Hsa.3115', 'Hsa.428',
'Hsa.20976', 'Hsa.1260', 'Hsa.1127', 'Hsa.1522', 'Hsa.9218', 'Hsa.2841',
'Hsa.1089', 'Hsa.34968', 'Hsa.22909', 'Hsa.1138', 'Hsa.25748', 'Hsa.1288',
'Hsa.84', 'Hsa.2386', 'Hsa.275', 'Hsa.33368', 'Hsa.4937', 'Hsa.1470',
'Hsa.1453', 'Hsa.7802', 'Hsa.1802', 'Hsa.3215', 'Hsa.3198', 'Hsa.1359',
'Hsa.2274', 'Hsa.44595', 'Hsa.2157', 'Hsa.925', 'Hsa.26628', 'Hsa.23249',
'Hsa.41495', 'Hsa.2986', 'Hsa.717', 'Hsa.1714', 'Hsa.13610', 'Hsa.1973',
'Hsa.41314', 'Hsa.2503', 'Hsa.1088', 'Hsa.37192', 'Hsa.1347', 'Hsa.28800',
'Hsa.40063', 'Hsa.7784', 'Hsa.1043', 'Hsa.3969', 'Hsa.36599', 'Hsa.3203',
'Hsa.2974', 'Hsa.1672', 'Hsa.1860', 'Hsa.3888', 'Hsa.24877', 'Hsa.16501',
'Hsa.3166', 'Hsa.2558', 'Hsa.6546', 'Hsa.2777', 'Hsa.1331', 'Hsa.1618',
'Hsa.23677', 'Hsa.27958', 'Hsa.25481', 'Hsa.1659', 'Hsa.39753', 'Hsa.8532',
'Hsa.2610', 'Hsa.1030', 'Hsa.1283', 'Hsa.1467', 'Hsa.809', 'Hsa.1893',
'Hsa.11885', 'Hsa.3410', 'Hsa.376', 'Hsa.432', 'Hsa.3648', 'Hsa.3967',
'Hsa.2765', 'Hsa.2101', 'Hsa.32865', 'Hsa.326', 'Hsa.23565', 'Hsa.10169',
'Hsa.2609', 'Hsa.3102', 'Hsa.8833', 'Hsa.14595', 'Hsa.17091', 'Hsa.2547',
'Hsa.904', 'Hsa.42442', 'Hsa.7348', 'Hsa.1211', 'Hsa.229', 'Hsa.43540',
'Hsa.2756', 'Hsa.1787', 'Hsa.1770', 'Hsa.3118', 'Hsa.2146', 'Hsa.7705',
'Hsa.1675', 'Hsa.37553', 'Hsa.1957', 'Hsa.32319', 'Hsa.60', 'Hsa.6910',
'Hsa.22461', 'Hsa.2530', 'Hsa.943', 'Hsa.360', 'Hsa.974', 'Hsa.6438',
'Hsa.8551', 'Hsa.16443', 'Hsa.2183', 'Hsa.27341', 'Hsa.1670', 'Hsa.36649',
'Hsa.2339', 'Hsa.2922', 'Hsa.399', 'Hsa.290', 'Hsa.19945', 'Hsa.13628',
'Hsa.2530', 'Hsa.2591', 'Hsa.2585', 'Hsa.1617', 'Hsa.41281', 'Hsa.2451',
'Hsa.718', 'Hsa.22242', 'Hsa.4153', 'Hsa.746', 'Hsa.35471', 'Hsa.981',
'Hsa.20609', 'Hsa.5130', 'Hsa.5141', 'Hsa.2793', 'Hsa.1479', 'Hsa.1875',
'Hsa.11673', 'Hsa.41187', 'Hsa.2509', 'Hsa.15598', 'Hsa.1885', 'Hsa.1517',
'Hsa.367', 'Hsa.37983', 'Hsa.1701', 'Hsa.1509', 'Hsa.40957', 'Hsa.2837',
'Hsa.2706', 'Hsa.1460', 'Hsa.7336', 'Hsa.2996', 'Hsa.1154', 'Hsa.667',
'Hsa.2901', 'Hsa.7', 'Hsa.2410', 'Hsa.7462', 'Hsa.17564', 'Hsa.814',
'Hsa.2614', 'Hsa.1719', 'Hsa.2340', 'Hsa.2354', 'Hsa.1320', 'Hsa.41247',
'Hsa.2957', 'Hsa.3238', 'Hsa.34431', 'Hsa.1062', 'Hsa.36671', 'Hsa.341',
'Hsa.2827', 'Hsa.7648', 'Hsa.28895', 'Hsa.35496', 'Hsa.28468', 'Hsa.36943',
'Hsa.6317', 'Hsa.19143', 'Hsa.44067', 'Hsa.1620', 'Hsa.10314', 'Hsa.14763',
'Hsa.34510', 'Hsa.27300', 'Hsa.25522', 'Hsa.2678', 'Hsa.2904', 'Hsa.1066',
'Hsa.3208', 'Hsa.32535', 'Hsa.23150', 'Hsa.27488', 'Hsa.477', 'Hsa.1477',
'Hsa.990', 'Hsa.17210', 'Hsa.27537', 'Hsa.26969', 'Hsa.3154', 'Hsa.660',
'Hsa.983', 'Hsa.2897', 'Hsa.2499', 'Hsa.1939', 'Hsa.36528', 'Hsa.18693',
'Hsa.3072', 'Hsa.612', 'Hsa.9711', 'Hsa.2498', 'Hsa.1776', 'Hsa.6288',
'Hsa.329', 'Hsa.1574', 'Hsa.33867', 'Hsa.8856', 'Hsa.27827', 'Hsa.5164',
'Hsa.3813', 'Hsa.56', 'Hsa.3225', 'Hsa.3182', 'Hsa.6619', 'Hsa.654',
```

```
'Hsa.32', 'Hsa.1410', 'Hsa.9641', 'Hsa.2240', 'Hsa.25536', 'Hsa.2085',
'Hsa.12241', 'Hsa.108', 'Hsa.2330', 'Hsa.1786', 'Hsa.491', 'Hsa.41152',
'Hsa.28641', 'Hsa.26945', 'Hsa.9574', 'Hsa.27481', 'Hsa.44244', 'Hsa.862',
'Hsa.36678', 'Hsa.2818', 'Hsa.6555', 'Hsa.21957', 'Hsa.25813', 'Hsa.1045',
'Hsa.2154', 'Hsa.2998', 'Hsa.42443', 'Hsa.9103', 'Hsa.2964', 'Hsa.25762',
'Hsa.44229', 'Hsa.297', 'Hsa.1140', 'Hsa.2343', 'Hsa.2570', 'Hsa.14914',
'Hsa.11582', 'Hsa.9353', 'Hsa.24582', 'Hsa.2210', 'Hsa.952', 'Hsa.10437',
'Hsa.3715', 'Hsa.944', 'Hsa.2280', 'Hsa.21195', 'Hsa.1373', 'Hsa.21562',
'Hsa.121', 'Hsa.41231', 'Hsa.716', 'Hsa.3141', 'Hsa.15776', 'Hsa.28939',
'Hsa.1788', 'Hsa.542', 'Hsa.32411', 'Hsa.8164', 'Hsa.9246', 'Hsa.1691',
'Hsa.2300', 'Hsa.7781', 'Hsa.492', 'Hsa.1877', 'Hsa.1774', 'Hsa.485',
'Hsa.2115', 'Hsa.33144', 'Hsa.21104', 'Hsa.18790', 'Hsa.1274', 'Hsa.18589',
'Hsa.2097', 'Hsa.345', 'Hsa.2192', 'Hsa.145', 'Hsa.41159', 'Hsa.12465',
'Hsa.1402', 'Hsa.33268', 'Hsa.27808', 'Hsa.1240', 'Hsa.20376', 'Hsa.3230',
'Hsa.2692', 'Hsa.40211', 'Hsa.25371', 'Hsa.34776', 'Hsa.2458', 'Hsa.15734',
'Hsa.2926', 'Hsa.1592', 'Hsa.687', 'Hsa.41346', 'Hsa.10908', 'Hsa.14069',
'Hsa.192', 'Hsa.9295', 'Hsa.4296', 'Hsa.2821', 'Hsa.335', 'Hsa.6416',
'Hsa.2168', 'Hsa.41368', 'Hsa.18469', 'Hsa.3933', 'Hsa.37654', 'Hsa.2206',
'Hsa.2347', 'Hsa.2528', 'Hsa.39731', 'Hsa.38171', 'Hsa.26979', 'Hsa.3280',
'Hsa.3316', 'Hsa.358', 'Hsa.41002', 'Hsa.2683', 'Hsa.1539', 'Hsa.21736',
'Hsa.2407', 'Hsa.2517', 'Hsa.8085', 'Hsa.176', 'Hsa.26971', 'Hsa.2687',
'Hsa.1468', 'Hsa.4286', 'Hsa.2688', 'Hsa.12260', 'Hsa.18833', 'Hsa.584',
'Hsa.2749', 'Hsa.25294', 'Hsa.42382', 'Hsa.14154', 'Hsa.2945', 'Hsa.67',
'Hsa.21757', 'Hsa.1550', 'Hsa.230', 'Hsa.33277', 'Hsa.23124', 'Hsa.39809',
'Hsa.3963', 'Hsa.34575', 'Hsa.35741', 'Hsa.1497', 'Hsa.21868', 'Hsa.33576',
'Hsa.2250', 'Hsa.15716', 'Hsa.888', 'Hsa.2337', 'Hsa.1811', 'Hsa.1145',
'Hsa.913', 'Hsa.42826', 'Hsa.41239', 'Hsa.1057', 'Hsa.2753', 'Hsa.2137',
'Hsa.2058', 'Hsa.8863', 'Hsa.9671', 'Hsa.3223', 'Hsa.33637', 'Hsa.17766',
'Hsa.504', 'Hsa.10746', 'Hsa.37609', 'Hsa.37528', 'Hsa.29913', 'Hsa.2626',
'Hsa.565', 'Hsa.10700', 'Hsa.25830', 'Hsa.920', 'Hsa.7870', 'Hsa.3173',
'Hsa.3187', 'Hsa.26430', 'Hsa.41108', 'Hsa.3024', 'Hsa.2389', 'Hsa.43978',
'Hsa.31395', 'Hsa.634', 'Hsa.2573', 'Hsa.41323', 'Hsa.1660', 'Hsa.404',
'Hsa.134', 'Hsa.40449', 'Hsa.1373', 'Hsa.36710', 'Hsa.45658', 'Hsa.2196',
'Hsa.33695', 'Hsa.35367', 'Hsa.1765', 'Hsa.36161', 'Hsa.2939', 'Hsa.8192',
'Hsa.41299', 'Hsa.2705', 'Hsa.1614', 'Hsa.2644', 'Hsa.1607', 'Hsa.1185',
'Hsa.28784', 'Hsa.7652', 'Hsa.3201', 'Hsa.45458', 'Hsa.1209', 'Hsa.1763',
'Hsa.7728', 'Hsa.1439', 'Hsa.1423', 'Hsa.1297', 'Hsa.8503', 'Hsa.33095',
'Hsa.664', 'Hsa.9174', 'Hsa.1758', 'Hsa.2243', 'Hsa.15198', 'Hsa.6782',
'Hsa.3', 'Hsa.72', 'Hsa.3186', 'Hsa.3250', 'Hsa.2237', 'Hsa.41165',
'Hsa.1878', 'Hsa.1491', 'Hsa.17213', 'Hsa.2918', 'Hsa.1143', 'Hsa.44116',
'Hsa.2891', 'Hsa.36705', 'Hsa.11616', 'Hsa.3105', 'Hsa.17130', 'Hsa.826',
'Hsa.2134', 'Hsa.41098', 'Hsa.9590', 'Hsa.1415', 'Hsa.3082', 'Hsa.15007',
'Hsa.17901', 'Hsa.475', 'Hsa.2084', 'Hsa.2208', 'Hsa.27285', 'Hsa.41260',
'Hsa.14822', 'Hsa.336', 'Hsa.984', 'Hsa.3952', 'Hsa.9683']
```

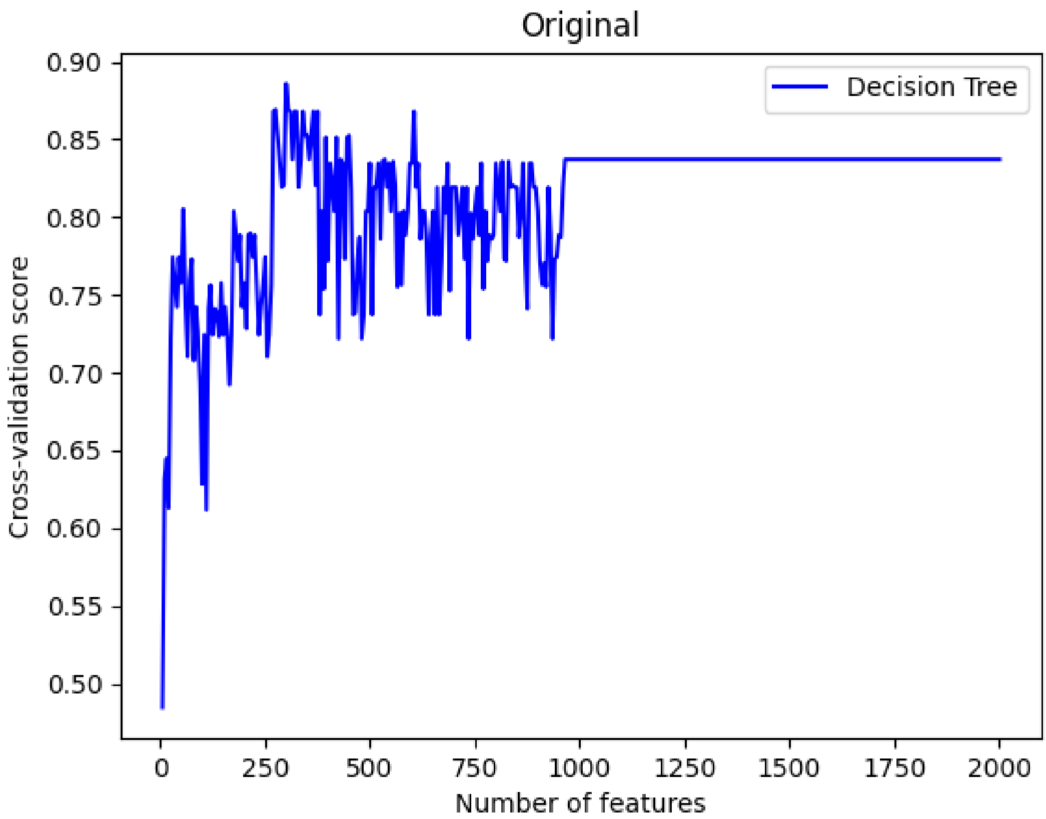
feature evaluation使用DecisionTreeClassifier的方法在第300個特徵時找到最好的結果，輸出結果如下：

```
Max of Decision Tree: 0.8858974358974357
Number of features: 300
```

並且我將這些特徵全部輸出出來，如下：

```
Feature chosen in feature evaluation:
['Hsa.13491', 'Hsa.37254', 'Hsa.474', 'Hsa.6080', 'Hsa.45293', 'Hsa.5710',
'Hsa.909', 'Hsa.1836', 'Hsa.1273', 'Hsa.19249', 'HSAC07', 'HSAC07', 'HSAC07',
'HSAC07', 'Hsa.1139', 'Hsa.2299', 'UMGAP', 'Hsa.1238', 'Hsa.6039', 'Hsa.98',
'Hsa.5363', 'Hsa.8093', 'Hsa.5444', 'Hsa.3094', 'Hsa.8125', 'Hsa.1013',
'Hsa.2542', 'Hsa.678', 'Hsa.36957', 'Hsa.6977', 'Hsa.24121', 'Hsa.36957',
'Hsa.938', 'Hsa.33965', 'Hsa.2665', 'Hsa.20', 'Hsa.12257', 'Hsa.2026',
'Hsa.5292', 'Hsa.32730', 'Hsa.1985', 'Hsa.10363', 'Hsa.624', 'Hsa.2582',
'Hsa.11712', 'Hsa.10755', 'Hsa.2600', 'Hsa.68', 'Hsa.2918', 'Hsa.7280',
'Hsa.2929', 'Hsa.21232', 'Hsa.1961', 'Hsa.916', 'Hsa.1732', 'Hsa.13183',
'Hsa.572', 'Hsa.45260', 'Hsa.73', 'Hsa.3129', 'Hsa.34312', 'Hsa.537',
'Hsa.36694', 'Hsa.957', 'Hsa.668', 'Hsa.10975', 'Hsa.2071', 'Hsa.91',
'Hsa.558', 'Hsa.1222', 'Hsa.3017', 'Hsa.1829', 'Hsa.1367', 'Hsa.3197',
'Hsa.22614', 'Hsa.11096', 'Hsa.10306', 'Hsa.43252', 'Hsa.31', 'Hsa.6146',
'Hsa.21418', 'Hsa.18664', 'Hsa.25451', 'Hsa.8831', 'Hsa.10510', 'Hsa.2831',
'Hsa.1116', 'Hsa.39141', 'Hsa.6376', 'Hsa.580', 'Hsa.24948', 'Hsa.24652',
'Hsa.3003', 'Hsa.812', 'Hsa.8305', 'Hsa.15115', 'Hsa.3922', 'Hsa.41218',
'Hsa.2710', 'Hsa.307', 'Hsa.587', 'Hsa.3547', 'Hsa.3295', 'Hsa.4954',
'Hsa.1939', 'Hsa.568', 'Hsa.4992', 'Hsa.1006', 'Hsa.9817', 'Hsa.21339',
'Hsa.100', 'Hsa.8656', 'Hsa.951', 'Hsa.8606', 'Hsa.13102', 'Hsa.8177',
'Hsa.8147', 'Hsa.41315', 'Hsa.921', 'Hsa.628', 'Hsa.2714', 'Hsa.2613',
'Hsa.823', 'i', 'i', 'Hsa.996', 'Hsa.33572', 'Hsa.2806', 'Hsa.38375',
'Hsa.5122', 'Hsa.10358', 'Hsa.854', 'Hsa.80', 'Hsa.29228', 'Hsa.20034',
'Hsa.1050', 'Hsa.39432', 'Hsa.2829', 'Hsa.2795', 'Hsa.6165', 'Hsa.19784',
'Hsa.1178', 'Hsa.26767', 'Hsa.3180', 'Hsa.2902', 'Hsa.3307', 'Hsa.19003',
'Hsa.41347', 'Hsa.1044', 'Hsa.13598', 'Hsa.1738', 'Hsa.17649', 'Hsa.2950',
'Hsa.34874', 'Hsa.2598', 'Hsa.1276', 'Hsa.1450', 'Hsa.1096', 'Hsa.11096',
'Hsa.2809', 'Hsa.1466', 'Hsa.35496', 'Hsa.31', 'Hsa.153', 'Hsa.3910',
'Hsa.194', 'Hsa.995', 'Hsa.821', 'Hsa.848', 'Hsa.9631', 'Hsa.2600',
'Hsa.1277', 'Hsa.13702', 'Hsa.26747', 'Hsa.2505', 'Hsa.2440', 'Hsa.3152',
'Hsa.5548', 'Hsa.27832', 'Hsa.1008', 'Hsa.1670', 'Hsa.28145', 'Hsa.16100',
'Hsa.454', 'Hsa.9667', 'Hsa.2995', 'Hsa.11780', 'Hsa.19731', 'Hsa.244',
'Hsa.1221', 'Hsa.34351', 'Hsa.1033', 'Hsa.30128', 'Hsa.5143', 'Hsa.8736',
'Hsa.544', 'Hsa.714', 'Hsa.1108', 'Hsa.879', 'Hsa.929', 'Hsa.13624',
'Hsa.2592', 'Hsa.2232', 'Hsa.41083', 'Hsa.3299', 'Hsa.4234', 'Hsa.2560',
'Hsa.3328', 'Hsa.551', 'Hsa.35528', 'Hsa.3852', 'Hsa.1990', 'Hsa.8583',
'Hsa.27721', 'Hsa.6581', 'Hsa.1896', 'Hsa.22251', 'Hsa.591', 'Hsa.10067',
'Hsa.36291', 'Hsa.479', 'Hsa.2163', 'Hsa.592', 'Hsa.40177', 'Hsa.59',
'Hsa.13670', 'Hsa.2268', 'Hsa.3088', 'Hsa.41875', 'Hsa.2311', 'Hsa.27686',
'Hsa.2119', 'Hsa.2529', 'Hsa.39', 'Hsa.561', 'Hsa.3909', 'Hsa.39809',
'Hsa.3185', 'Hsa.50', 'Hsa.831', 'Hsa.9285', 'Hsa.6472', 'Hsa.3454',
'Hsa.2379', 'Hsa.2951', 'Hsa.16793', 'Hsa.934', 'Hsa.36657', 'a.1000',
'Hsa.286', 'Hsa.215', 'Hsa.2132', 'Hsa.24206', 'Hsa.1948', 'Hsa.26083',
'Hsa.2191', 'Hsa.726', 'Hsa.962', 'Hsa.1793', 'Hsa.37169', 'Hsa.1165',
'Hsa.2094', 'Hsa.2780', 'Hsa.23285', 'Hsa.6376', 'Hsa.3209', 'Hsa.853',
'Hsa.1133', 'Hsa.2141', 'Hsa.2487', 'Hsa.1244', 'Hsa.1207', 'Hsa.834',
'Hsa.1768', 'Hsa.1640', 'Hsa.3150', 'Hsa.36019', 'Hsa.2062', 'Hsa.17822',
'Hsa.6422', 'Hsa.20028', 'Hsa.847', 'Hsa.1132', 'Hsa.1106', 'Hsa.2478',
'Hsa.1630', 'Hsa.6376', 'Hsa.2416', 'Hsa.897', 'Hsa.1739', 'Hsa.22167',
'Hsa.39288', 'Hsa.3022', 'Hsa.2304', 'Hsa.6458', 'Hsa.3306', 'Hsa.7324',
'Hsa.36010', 'Hsa.9691', 'Hsa.3135']
```

分類器的效能變化則用sample code視覺化的程式碼將圖表呈現，如下：



Source Code

```python
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score

from sklearn_genetic import GAFeatureSelectionCV

# ============================================================
#                           Load Data
# ============================================================
# TODO: Load data here.
indexes = pd.read_csv('hw3_Data1/index.txt', delimiter = '\t', header =
None).T
x = pd.read_csv('hw3_Data1/gene.txt', delimiter = ' ', header =
None).to_numpy().T
y = pd.read_csv('hw3_Data1/label.txt', header = None).to_numpy()
index = indexes.iloc[0,].str.strip()
y = (y>0).astype(int).reshape(y.shape[0])

# ============================================================
#                           Load Data
# ============================================================


# ============================================================
#                        Feature ranking
```

```
# ============================================================
# TODO: Design your score function for feature selection
# TODO: To use the provided evaluation sample code, you need to generate ranking_idx, which is the sorted index of feature
clf = DecisionTreeClassifier(random_state=0)

evolved_selector = GAFeatureSelectionCV(
    estimator=clf,
    cv=5,
    scoring="accuracy",
    population_size=5,
    generations=10,
    n_jobs=-1,
    verbose=True,
    keep_top_k=2,
    elitism=True,
    crossover_probability=0.2,
    mutation_probability=0.8,
    tournament_size=3,
)

# Train and select the features
evolved_selector.fit(x, y)

# Get best feature
features = evolved_selector.best_features_

# print(np.shape(features))

def get_ranking_idx(features):
    ranking_idx = []
    for i in range(2000):
        if features[i] == True:
            ranking_idx.append(i)
    return ranking_idx

ranking_idx = get_ranking_idx(features)
fo = index[ranking_idx]
fo1 = fo.tolist()
print("After GA:")
print(f"Number of features: {len(ranking_idx)}")
print("Feature chosen after GA:")
print(fo1)

# ============================================================
#                        Feature ranking
# ============================================================


# ============================================================
#                        Feature evaluation
# ============================================================
# Use a simple dicision tree with 5-fold validation to evaluate the feature selection result.
# You can try other classifier and hyperparameter.
score_history = []
for m in range(5, 2001, 5):
    # Select Top m feature
```

```python
    x_subset = x[:, ranking_idx[:m]]

    # Build random forest
    clf = DecisionTreeClassifier(random_state=0)
    #clf = SVC(kernel='rbf', random_state=0) #build SVM

    # Calculate validation score
    scores = cross_val_score(clf, x_subset, y, cv=5)

    # Save the score calculated with m feature
    score_history.append(scores.mean())

# Report best accuracy.
num_feature = np.argmax(score_history)*5+5
f_new = index[ranking_idx[:num_feature]]
f_new_l = f_new.tolist()
r_idx = list(ranking_idx)
# print("ranking_idx:")
# print(r_idx)
# print(len(f_new_l))
print(f"Max of Decision Tree: {max(score_history)}")
# print(f"Max of SVM: {max(score_history)}")
# print(f"Max of Radom Forest: {max(score_history)}")
print(f"Number of features: {num_feature}")
# print("Feature chosen in feature evaluation:")
# print(f_new_l)

# ===========================================================
#                        Feature evaluation
# ===========================================================


# ===========================================================
#                        Visualization
# ===========================================================
plt.plot(range(5, 2001, 5), score_history, c='blue')
plt.title('Original')
plt.xlabel('Number of features')
plt.ylabel('Cross-validation score')
plt.legend(['Decision Tree'])
# plt.legend(['SVM'])
# plt.legend(['Ramdom Forest'])
plt.savefig('3-2_result.png')
# ===========================================================
#                        Visualization
# ===========================================================
```

Problem 3. ARIMA Forecast (30%)

Implement a Python program to perform an ARIMA analysis on Taiwan's Stock Exchange Index. Build an ARMIA model based on the "close" value data from 11/03/2021 up to 10/03/2022. And then forecast the "close" values for 10/04/2022 up to 11/03/2022.

- **What are the ARIMA parameters (p, d, q, P, D, Q, s) that you use? And what is the mean square error (MSE) of your forecast? (15%)**

ARIMA(p, d, q, P, D, Q, s)

我使用pmdarima中的auto_arima module，並手動測試 s 的部分，找出最好的 p, d, q, P, D, Q 這六項參數，s部分我分別逐一帶入7~60，得到AIC最小的Best Model是: **'ARIMA(0,1,0)(0,0,1)[25]'**，即 ARIMA(p, d, q, P, D, Q, s) = ARIMA (0, 1, 0, 0, 0, 1, 25)

`auto-ARIMA code`

```
# d-val
d_val = ndiffs(train_data['Close'], test='adf')
print("d-val: ")
print(d_val)

# fit stepwise auto-ARIMA
#Define auto-arima to find best model
model = pm.auto_arima(train['Close'],
                      d = d_val,
                      start_p = 0,
                      max_p = 5,
                      start_q = 0,
                      max_q = 5,
                      D=None,
                      m=25,
                      # don't want to know if an order does not work
                      suppress_warnings=True,
                      trace=True,
                      # don't want convergence warnings
                      stepwise=True)  # set to stepwise

```

`search stepwise to minimize AIC`

```
Performing stepwise search to minimize aic
 ARIMA(0,1,0)(1,0,1)[25] intercept   : AIC=2973.335, Time=0.56 sec
 ARIMA(0,1,0)(0,0,0)[25] intercept   : AIC=2973.657, Time=0.01 sec
 ARIMA(1,1,0)(1,0,0)[25] intercept   : AIC=2973.799, Time=0.39 sec
 ARIMA(0,1,1)(0,0,1)[25] intercept   : AIC=2973.375, Time=0.30 sec
 ARIMA(0,1,0)(0,0,0)[25]             : AIC=2973.595, Time=0.01 sec
 ARIMA(0,1,0)(0,0,1)[25] intercept   : AIC=2971.964, Time=0.25 sec
 ARIMA(0,1,0)(0,0,2)[25] intercept   : AIC=2973.473, Time=1.47 sec
 ARIMA(0,1,0)(1,0,0)[25] intercept   : AIC=2972.353, Time=0.25 sec
 ARIMA(0,1,0)(1,0,2)[25] intercept   : AIC=inf, Time=3.83 sec
 ARIMA(1,1,0)(0,0,1)[25] intercept   : AIC=2973.431, Time=0.28 sec
 ARIMA(1,1,1)(0,0,1)[25] intercept   : AIC=2975.191, Time=0.46 sec
```

```
 ARIMA(0,1,0)(0,0,1)[25]             : AIC=2972.372, Time=0.19 sec

Best model:  ARIMA(0,1,0)(0,0,1)[25] intercept
Total fit time: 7.986 seconds
```

auto_arima的parameter，還有model輸出的summary可見最下面的 **Appendix.**

接著由於作業有提及要用繳交的code不行有auto_arima，所以就寫個ARIMA的版本，只是參數部分使用auto_arima找到的。

`code`

```python
#splitting the data to train and test sets based on Ntest value
from pmdarima.arima import ARIMA
#from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
Ntest = len(test_df['Close'])
train = all_df.iloc[:-Ntest]
test = all_df.iloc[-Ntest:]

y= df['Close'].to_numpy()
model1 = ARIMA(order=(0,1,0), seasonal_order = (0, 0, 1, 25))
model1.fit(y)
```

`ARIMA get_params()`

```
{
    'maxiter': 50,
    'method': 'lbfgs',
    'order': (0, 1, 0),
    'out_of_sample_size': 0,
    'scoring': 'mse',
    'scoring_args': None,
    'seasonal_order': (0, 0, 1, 25),
    'start_params': None,
    'suppress_warnings': False,
    'trend': None,
    'with_intercept': True
}
```

`ARIMA summary()`

```
                                      SARIMAX Results

==============================================================================
================
Dep. Variable:                                    y   No. Observations:
                225
Model:               SARIMAX(0, 1, 0)x(0, 0, [1], 25)   Log Likelihood
        -1482.982
Date:                              Fri, 18 Nov 2022   AIC
          2971.964
Time:                                      23:25:58   BIC
          2982.199
```

```
Sample:                                           0   HQIC                          2976.096
                                                - 225                                         
Covariance Type:                                opg                                         
==================================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------
intercept    -16.7867     10.795     -1.555      0.120     -37.945       4.371
ma.S.L25      -0.1357      0.068     -1.996      0.046      -0.269      -0.002
sigma2       3.28e+04   2995.013     10.950      0.000    2.69e+04    3.87e+04
=======================================================================================
Ljung-Box (L1) (Q):                   0.64   Jarque-Bera (JB):                  3.84
Prob(Q):                              0.42   Prob(JB):                          0.15
Heteroskedasticity (H):               1.78   Skew:                             -0.29
Prob(H) (two-sided):                  0.01   Kurtosis:                          3.28
=======================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

Mean Squared Error

```
Mean Squared Error:  91327.32538343023
Mean Absolute Error:  259.8009363027745
```

- **Plot the whole stock (11/03/2021-11/03/2022) and your forecast data (10/04/2022~/11/03/2022) on the same figure. (5%) and submit you code (10%)**

`code`

```
def plot_result(model, data, col_name, Ntest):
    params = model.get_params()
    d = params['order'][1]

    train_pred = model.predict_in_sample(start=d, end=-1)

    test_pred, conf = model.predict(n_periods=Ntest,
                                    return_conf_int=True)

    plt.plot(data[col_name].index, data[col_name],
```

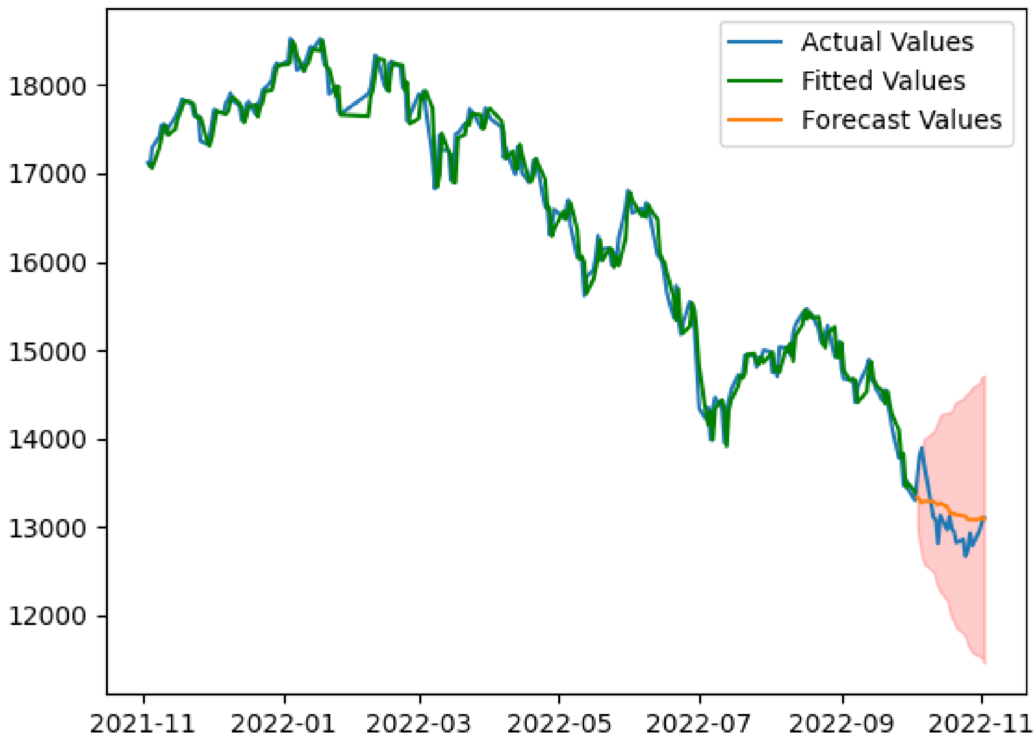
```python
                  color='tab:blue', label='Actual Values')
        plt.plot(train.index[d:], train_pred,
                  color='tab:green', label='Fitted Values')
        plt.plot(test.index, test_pred,
                  color='tab:red', label='Forecast Values')
        plt.legend()
        plt.savefig('3-3_result.png')


        #evaluating the model using MSE and MAE metrics
        y_true = test[col_name].values
        rmse = mean_squared_error(y_true,test_pred)
        mae = mean_absolute_error(y_true,test_pred)
        return rmse, mae

    rmse , mae = plot_result(model, all, 'Close',
                              Ntest=len(test['Close']))
    print('Mean Squared Error: ', rmse)
    print('Mean Absolute Error: ', mae)
```

- **Plot the whole stock (11/03/2021-11/03/2022) and your forecast data (10/04/2022~/11/03/2022) on the same figure. (5%) and submit you code (10%)**

Plot Result



Source Code

```
import pandas as pd
from matplotlib import pyplot as plt
import pmdarima as pm
from pmdarima.arima.utils import ndiffs
from pmdarima.arima import ARIMA
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error

# ============================================================
#                            Load Data
# ============================================================
# TODO: Load data here.
train_df = pd.read_csv('hw3_Data2/train.csv',
                         index_col="Date", parse_dates=True)
test_df = pd.read_csv('hw3_Data2/test.csv',
                        index_col="Date", parse_dates=True)
all_df = pd.concat([train_df,test_df])

# print("train: ")
# print(train_df)
# print("test: ")
# print(test_df)

# ============================================================
#                            Load Data
# ============================================================


# ============================================================
#                             ARIMA
# ============================================================

# d-val
d_val = ndiffs(train_df['Close'], test='adf')
print("d-val: ")
print(d_val)

## fit stepwise auto-ARIMA
#splitting the data to train and test sets based on Ntest value
#last  days


#Define auto-arima to find best model
# model = pm.auto_arima(train_df['Close'],
#                         d = d_val,
#                         start_p = 0,
#                         max_p = 5,
#                         start_q = 0,
#                         max_q = 5,
#                         D=None,
#                         m=25,
#                         stepwise=True,
#                         trace=True)
# print(model.get_params())
# print(model.summary())
```

```python

y = train_df['Close'].to_numpy()
model1 = ARIMA(order=(0,1,0), seasonal_order = (0, 0, 1, 25))
model1.fit(y)
print(model1.get_params())
print(model1.summary())

# ===========================================================
#                           ARIMA
# ===========================================================


# ===========================================================
#                       Visualization
# ===========================================================
def plot_result(model, data, train, test, col_name, Ntest):

    params = model.get_params()
    d = params['order'][1]

    #In sample data prediction
    train_pred = model.predict_in_sample(start=d, end=-1)

    test_pred, conf = model.predict(n_periods=Ntest, return_conf_int=True)
    #print(len(test_pred))

    #plotting real values, fitted values and prediction values
    plt.plot(data[col_name].index, data[col_name], label='Actual Values')
    plt.plot(train.index[d:], train_pred, color='green', label='Fitted Values')
    plt.plot(test.index, test_pred, label='Forecast Values')
    #print(test.index)
    plt.fill_between(test.index, conf[:,0], conf[:,1], color='red', alpha=0.2)
    plt.legend()
    plt.savefig('3-3_result.png')
    #evaluating the model using RMSE and MAE metrics
    y_true = test_df[col_name].values
    mse = mean_squared_error(y_true,test_pred)
    mae = mean_absolute_error(y_true,test_pred)
    return mse, mae

mse , mae = plot_result(model1, all_df, train_df, test_df, 'Close', Ntest=len(test_df['Close']))
print('Mean Squared Error: ', mse)
print('Mean Absolute Error: ', mae)
# ===========================================================
#                       Visualization
# ===========================================================
```

Appendix.

`auto-ARIMA get_params()`

```
{
    'maxiter': 50,
    'method': 'lbfgs',
    'order': (0, 1, 0),
    'out_of_sample_size': 0,
    'scoring': 'mse',
    'scoring_args': {},
    'seasonal_order': (0, 0, 1, 25),
    'start_params': None,
    'suppress_warnings': True,
    'trend': None,
    'with_intercept': True
}
```

`auto-ARIMA summary()`

```
                                      SARIMAX Results
==========================================================================================
Dep. Variable:                                    y   No. Observations:             225
Model:             SARIMAX(0, 1, 0)x(0, 0, [1], 25)   Log Likelihood          -1482.982
Date:                              Fri, 18 Nov 2022   AIC                      2971.964
Time:                                      23:39:35   BIC                      2982.199
Sample:                                           0   HQIC                     2976.096
                                              - 225
Covariance Type:                                opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept    -16.7867     10.795     -1.555      0.120     -37.945       4.371
ma.S.L25      -0.1357      0.068     -1.996      0.046      -0.269      -0.002
sigma2       3.28e+04   2995.013     10.950      0.000    2.69e+04    3.87e+04
===================================================================================
Ljung-Box (L1) (Q):                   0.64   Jarque-Bera (JB):                 3.84
Prob(Q):                              0.42   Prob(JB):                         0.15
Heteroskedasticity (H):               1.78   Skew:                            -0.29
Prob(H) (two-sided):                  0.01   Kurtosis:                         3.28
```

```
====================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```