Dimensionality Reduction and Visualization

Exercise Set 2 Solutions

Md. Abdullah-Al Mamun

Part A: Forward Selection and Variable Rankin

Problem A1: Forward Selection

a. By Implementing the nearest neighbour predictor and computing the leave-one-out error of nearest neighbour prediction for the Sculpt Faces data set when using all the data features. This number is the baseline performance that we must improve, in order to have any benefit from feature selection. We have got total error 5.05e+05.

b. The leave one out procedure was adjusted into a function in a separate file. The forward selection process was implemented in a while loop until best criterion is reached. The criterion here is when the error produced by a subset of a features is larger than the previous subset in the iteration.

```
>> best_features_errors

best_features_errors =

   1.0e+05 *

    3.1012    1.8679    1.2569    1.1821    1.0877
```

Where stopped:

```
>> sub_total_error

sub_total_error =

   1.1322e+05
```

```
>> best_features

best_features =

   190    54   138   225    33
```

c. Similar strategy has been used like before, the stopping criterion was to break the while loop when the length of best features reached is the length of the original features.

for visualizing the results, the vector of features taken in every step was reshaped. First column represents step 1 to 16, second column 17 to 33 and so on:

```
>> T=table(reshape(best_features,[16 16]));
>> T.Var1

ans =

   190   201   175   166   113    88   202    69   111    21     4   159   256   236   179   204
    54   181    13    18    72   172   200   241    42   248   212    24    70   184   109   131
   138    27   219   167   183    71   182    68   196   255   203    30    32   233    20   240
   225   168    47    74   223   189   116    59     2    41   156   145   237    35    91   124
    33   117   154    97   102   245   134    80   141    86   142   176    48   232   228   194
    25   246   114    19    63   199    34    52   234   177   178    66   150     5   120   174
   148     9    57   247   100    50   221    94    79   144    43   222    93    75   244    26
   103   133    10   254   185   216   252    39   211   151   153   130   218   140    29   197
    14   206   125   143   171    36   160   213   242    46   119   118   158    12    83    11
   193   106   209   115   220    16   239    98   173   165    78    84    64   128   161   169
   230    81   195   147   180     1   231   186   146     6    92   235    76   224    31   157
   139    55    60    82   129    53   188    40   164    23   205   132     8   243   122   110
    85   136   229   191    38   152   127   187   112    65    77   250    87    22    73   217
   208   126   105   198    58     3   107   253    28   249    95   137   238   226    67    89
   207   121    99   215   227     7   170    49    17    61    51    45   104    56    44   162
   108   251    90   123   163   101   149   192   135   214    62   155    96    37    15   210
```

Corresponding errors:

```
>> T=table(reshape(best_features_errors,[16 16]));
>> T.Var1

ans =

   1.0e+05 *

    3.1012    0.5026    0.4389    0.4183    0.4347    0.4888    0.4929    0.5470    0.6756    0.6871    0.7711    1.0680    1.2897    1.6466    1.9637    2.5876
    1.8679    0.4870    0.4205    0.4064    0.4383    0.5038    0.4901    0.5608    0.6746    0.6596    0.8434    1.1108    1.2769    1.8112    1.9856    2.6235
    1.2569    0.4816    0.4148    0.4172    0.4392    0.5058    0.5129    0.5419    0.6620    0.6691    0.8843    1.1445    1.2441    1.7709    2.0817    2.8732
    1.1821    0.4693    0.4084    0.4162    0.4749    0.5066    0.5209    0.5276    0.6834    0.7193    0.9418    1.1312    1.2522    1.7508    2.0980    2.9628
    1.0877    0.4625    0.4331    0.3927    0.4819    0.5025    0.5431    0.4882    0.6828    0.7625    0.8991    1.1685    1.3420    1.6795    2.0444    2.9417
    1.1322    0.4813    0.4176    0.3916    0.4732    0.4882    0.5178    0.5296    0.6345    0.7497    0.9061    1.1754    1.2461    1.6768    2.2704    3.1281
    0.9218    0.4819    0.4203    0.4069    0.4752    0.4819    0.5349    0.5622    0.6878    0.7572    0.9457    1.2474    1.2473    1.7478    2.3682    3.0557
    0.8315    0.5373    0.4357    0.4134    0.4941    0.4879    0.5088    0.5889    0.6753    0.8615    0.9184    1.2163    1.2524    1.7961    2.4621    3.4983
    0.7655    0.5089    0.4355    0.4042    0.4859    0.4875    0.5282    0.6633    0.6469    0.8259    0.9556    1.2657    1.3957    1.8044    2.3528    3.6158
    0.6856    0.5105    0.4430    0.4077    0.4810    0.4896    0.5194    0.6813    0.6567    0.8781    0.9508    1.3182    1.4495    1.8489    2.2401    3.4415
    0.6725    0.4758    0.4260    0.4104    0.4723    0.4973    0.5458    0.6511    0.6687    0.9251    0.9766    1.3747    1.4697    1.8755    2.2457    3.9892
    0.5809    0.4552    0.4233    0.4275    0.4723    0.5186    0.5615    0.6380    0.6420    0.9655    0.9969    1.3383    1.4356    2.0475    2.3877    4.4196
    0.5320    0.4480    0.4391    0.4227    0.4960    0.5089    0.5791    0.6559    0.6872    0.9277    1.0531    1.2890    1.6296    2.0336    2.4895    4.4086
    0.4999    0.4508    0.4081    0.4209    0.5119    0.4935    0.5535    0.6984    0.7068    0.9849    1.1022    1.2816    1.5684    1.9034    2.4707    4.5161
    0.5145    0.4319    0.4084    0.4283    0.4885    0.5027    0.5416    0.6960    0.6831    0.9369    1.1077    1.3567    1.6862    2.0176    2.4193    4.4202
    0.4875    0.4403    0.4059    0.4186    0.4860    0.5148    0.5354    0.6748    0.7131    0.8262    1.0798    1.2836    1.8197    1.9836    2.5231    5.0713
```

d. Yes, a better solution has been achieved.

From section b, the amount of error reduced from the original all features included was 78.55%:

```
>> (1-best_features_errors(end)/Total_Error)*100

ans =

   78.5510
```

From section c, it was a 92,3% reduction for 54 features:

```
>> [Val index]=min(best_features_errors)

Val =

   3.9161e+04


index =

    54
```

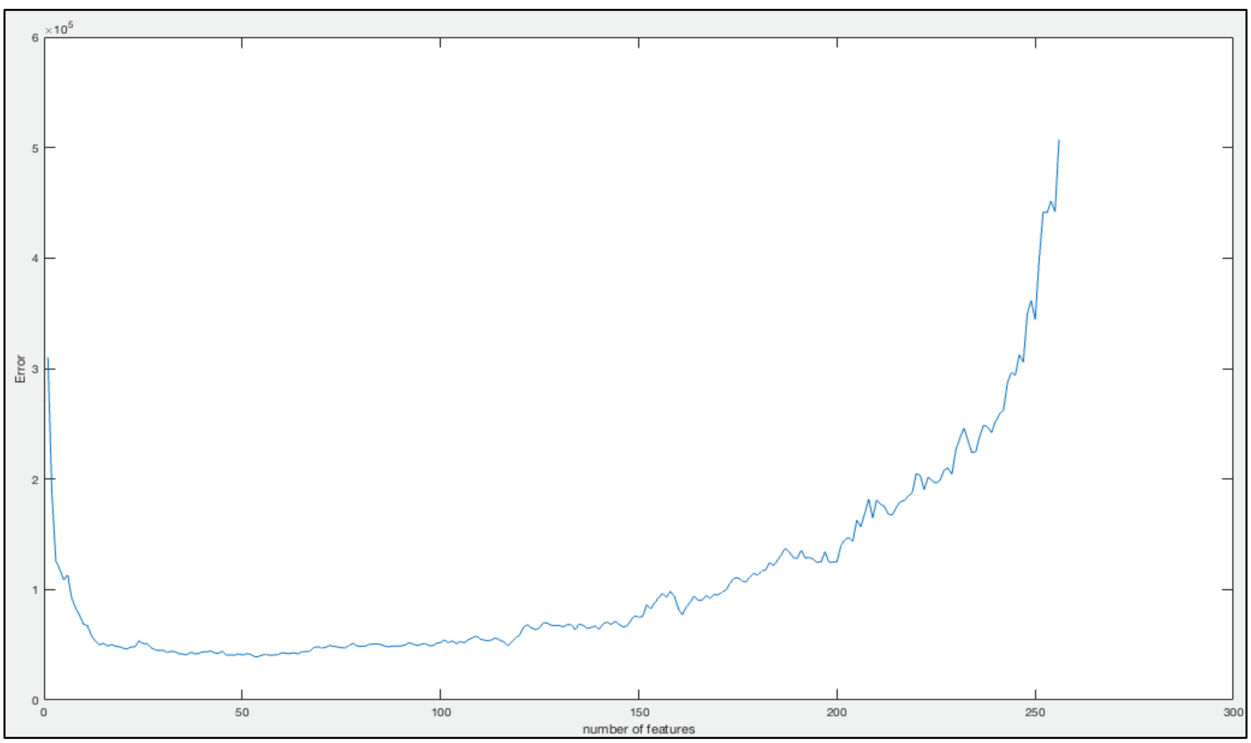
```
>> (1-Val/Total_Error)*100

ans =

   92.2779
```

The result could be explained with the fact that the algorithm used in section b does not encounter the possibility of a future combination of feature resulting in lower error.

The corresponding error plot for the number of features used:



Problem A2:

1) Using Ranking method, we can improve variable selection methods. Variable selection process can be improving by knowing the details about every variable.

For example, if the Pearson correlation is to be used on the previous problem. Such as:

```
corrs=corr(X,Y);
corrs_mean=mean(abs(corrs),2);
features_num=sum(corrs_mean>mean(corrs_mean));
```

```
>> features_num

features_num =

   101
```

Consequently, as part of the stopping criterion in the forward selection, the maximum number of features would be 101. 54 variables optimum revealed earlier and reducing computation cost from 256 to 101 variables.

2) Using the squared error of the leave one out method, the following order (rank) of variable was obtained:

```
>> T=table(reshape(indices,[16,16]));
>> T.Var1

ans =

   190    35   147   221    39    78    87   240    74   113   213    61   137   193    80   121
    25   175   191   192   231   199    22   236   243   145   226   131   176    30   157   253
    33   206   201    92   128   212   200   161   170    93   217    67   152    37   117   126
    60    36    95   239    82    64     1   163    24    16    15    68   102   109   144   122
    19   237   177   211   168   209    49    62    52   218   154    32    31   133   186   189
    54   116    88   256   185   148     3   105    75   171    81    98   184    69   178   159
    50    18    21    20    48   139   248   179   242   251   187     6   255   136   234   197
   207    57   204     4   100    43   183    27   232   155   214   127   247   203   202    13
   229   245   153    38    96   196   195    47   142     2    29   233   166    65   164   124
   198   215    71   158   141   180    63   146    10    28   194    76    94   125   129   156
   220   181    53    41    59   114    91    51   132   238   149     8   244   160   250    11
   219    34   150   208   151     7   119    66    97   115    84    86    58   210    89   107
   225   169    46   235   205   174    77    79   110   138    73   112    99   123   143   254
   230    56   216   130   223   111   135   118   165   252    70   134   182   241   104    12
   227    26     5    45    83    23    44   173   162   108   140     9   224    42    90   167
   246    55    40   222   103   172    85   106   120    17   249    14   101   188    72   228
```

The order was completely different than in the previous example, the difference was about 99.61%:

```
>> 1-sum(sum(best_features==indices'))/length(indices)

ans =

    0.9961
```

For the 54 optimum features was 98.15 difference:

```
>> 1-sum(sum(best_features(1:54)==indices(1:54)'))/length(indices(1:54)')

ans =

    0.9815
```

Yet only about 35% of the features were similar in the previous 54 optimum features obtained in problem 1 c:

```
same_features_per =

   35.1852
```