**Part I: Anomaly Detection**

D = 0.034; k = 20; p = 10; n = 28;

Approach A:

Accuracy: 95.99%

Confusion Matrix:

Predict

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 | 1 | |
| 0 | 1535 | | 36 |
| 1 | 28 | | 0 |

Actual

Approach B:

Accuracy: 96.49%

Confusion Matrix:

Predict

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 | 1 | |
| 0 | 1543 | | 28 |
| 1 | 28 | | 0 |

Actual

Approach C:

Accuracy: 96.49%

Confusion Matrix:

Predict

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 | 1 | |
| 0 | 1543 | | 28 |
| 1 | 28 | | 0 |

Actual

Since differences between each column are a lot, I normalize each column at first. Then I used four setups to find proper D, k, n, p. I calculated the mean of distances to determine a range of distances. Also, the number of wines whose qualities are larger than 8 is 18, and the number of wines whose qualities are less than 3 is 10. Thus, I will find a number around 20 to be k.

Approach A:

1. Fix distance=0.02. Then make x-axis be k values, 1~20, and y-axis be p values 1 ~10. The accuracy result table shows that K doesn’t affect accuracy and larger p, lower accuracy.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 | 17 | 19 |
| 1 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 |
| 2 | 0.018 | 0.941 | 0.941 | 0.941 | 0.941 | 0.941 | 0.941 | 0.941 | 0.941 | 0.941 |
| 3 | 0.018 | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 |
| 4 | 0.018 | 0.018 | 0.921 | 0.921 | 0.921 | 0.921 | 0.921 | 0.921 | 0.921 | 0.921 |
| 5 | 0.018 | 0.018 | 0.909 | 0.909 | 0.909 | 0.909 | 0.909 | 0.909 | 0.909 | 0.909 |
| 6 | 0.018 | 0.018 | 0.018 | 0.901 | 0.901 | 0.901 | 0.901 | 0.901 | 0.901 | 0.901 |
| 7 | 0.018 | 0.018 | 0.018 | 0.889 | 0.889 | 0.889 | 0.889 | 0.889 | 0.889 | 0.889 |
| 8 | 0.018 | 0.018 | 0.018 | 0.018 | 0.884 | 0.884 | 0.884 | 0.884 | 0.884 | 0.884 |
| 9 | 0.018 | 0.018 | 0.018 | 0.018 | 0.882 | 0.882 | 0.882 | 0.882 | 0.882 | 0.882 |
| 10 | 0.018 | 0.018 | 0.018 | 0.018 | 0.018 | 0.874 | 0.874 | 0.874 | 0.874 | 0.874 |

1. Fix the k = 20. Then make x-axis be p values, 4~24, and y-axis be D values 0.025~0.035.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 | 20 | 22 |
| 0.025 | 0.952 | 0.948 | 0.939 | 0.929 | 0.917 | 0.912 | 0.907 | 0.903 | 0.9 | 0.018 |
| 0.0255 | 0.952 | 0.949 | 0.942 | 0.935 | 0.922 | 0.916 | 0.912 | 0.907 | 0.905 | 0.018 |
| 0.026 | 0.952 | 0.95 | 0.944 | 0.937 | 0.929 | 0.924 | 0.914 | 0.911 | 0.907 | 0.018 |
| 0.0265 | 0.955 | 0.95 | 0.944 | 0.941 | 0.934 | 0.926 | 0.92 | 0.914 | 0.911 | 0.018 |
| 0.027 | 0.957 | 0.952 | 0.946 | 0.942 | 0.935 | 0.93 | 0.924 | 0.919 | 0.914 | 0.018 |
| 0.0275 | 0.957 | 0.952 | 0.948 | 0.943 | 0.938 | 0.932 | 0.927 | 0.924 | 0.919 | 0.018 |
| 0.028 | 0.96 | 0.953 | 0.949 | 0.943 | 0.939 | 0.935 | 0.931 | 0.927 | 0.922 | 0.018 |
| 0.0285 | 0.961 | 0.956 | 0.952 | 0.944 | 0.941 | 0.937 | 0.934 | 0.931 | 0.926 | 0.018 |
| 0.029 | 0.961 | 0.958 | 0.954 | 0.947 | 0.942 | 0.937 | 0.934 | 0.932 | 0.931 | 0.018 |
| 0.0295 | 0.962 | 0.959 | 0.957 | 0.951 | 0.946 | 0.942 | 0.934 | 0.933 | 0.932 | 0.018 |

Approach B:

1. Make x-axis be k values, 10~28, and y-axis be n values 10~46.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 10 | 12 | 14 | 16 | 18 | 20 | 22 | 24 | 26 | 28 |
| 10 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 |
| 14 | 0.974 | 0.974 | 0.974 | 0.974 | 0.974 | 0.974 | 0.974 | 0.974 | 0.974 | 0.974 |
| 18 | 0.971 | 0.971 | 0.971 | 0.971 | 0.971 | 0.971 | 0.971 | 0.971 | 0.971 | 0.971 |
| 22 | 0.969 | 0.969 | 0.969 | 0.969 | 0.969 | 0.969 | 0.969 | 0.969 | 0.969 | 0.969 |
| 26 | 0.966 | 0.966 | 0.966 | 0.966 | 0.966 | 0.966 | 0.966 | 0.966 | 0.966 | 0.966 |
| 30 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 |
| 34 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 |
| 38 | 0.959 | 0.959 | 0.959 | 0.959 | 0.959 | 0.959 | 0.959 | 0.959 | 0.959 | 0.959 |
| 42 | 0.956 | 0.957 | 0.957 | 0.957 | 0.956 | 0.956 | 0.956 | 0.956 | 0.956 | 0.956 |
| 46 | 0.956 | 0.955 | 0.955 | 0.955 | 0.955 | 0.955 | 0.955 | 0.955 | 0.955 | 0.956 |

Approach C:

1. Make x-axis be k values10~28, and y-axis be n values 10~46.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 10 | 12 | 14 | 16 | 18 | 20 | 22 | 24 | 26 | 28 |
| 10 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 | 0.976 |
| 15 | 0.973 | 0.973 | 0.973 | 0.973 | 0.973 | 0.973 | 0.973 | 0.973 | 0.973 | 0.973 |
| 20 | 0.97 | 0.97 | 0.97 | 0.97 | 0.97 | 0.97 | 0.97 | 0.97 | 0.97 | 0.97 |
| 25 | 0.967 | 0.967 | 0.967 | 0.967 | 0.967 | 0.967 | 0.967 | 0.967 | 0.967 | 0.967 |
| 30 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 | 0.964 |
| 35 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 | 0.961 |
| 40 | 0.957 | 0.959 | 0.959 | 0.959 | 0.959 | 0.959 | 0.959 | 0.957 | 0.957 | 0.957 |
| 45 | 0.956 | 0.957 | 0.957 | 0.957 | 0.956 | 0.956 | 0.956 | 0.956 | 0.956 | 0.956 |
| 50 | 0.954 | 0.954 | 0.954 | 0.954 | 0.954 | 0.954 | 0.952 | 0.952 | 0.952 | 0.952 |
| 55 | 0.951 | 0.951 | 0.951 | 0.951 | 0.951 | 0.951 | 0.951 | 0.949 | 0.949 | 0.949 |

**Part II: Link Analysis**

Main codes:

Normalization:

matrix\_A = zeros(100,100);

sum\_col = sum(A')';

row\_ind\_list = find(sum\_col ~=0)';

for ind = row\_ind\_list

column\_ind = find(A(ind,:) ~=0);

matrix\_A(ind,column\_ind) = 1/sum\_col(ind,1);

end

normal\_A = sparse(matrix\_A);

A=normal\_A;

Compute PR:

pr=ones(length(U),1); % initial ranks

d =0.85;

constant\_v = ones(length(U),1)\*(1-d)';

for iter=1:200

pr = constant\_v + (A')\*pr.\*d;

end

Figure: