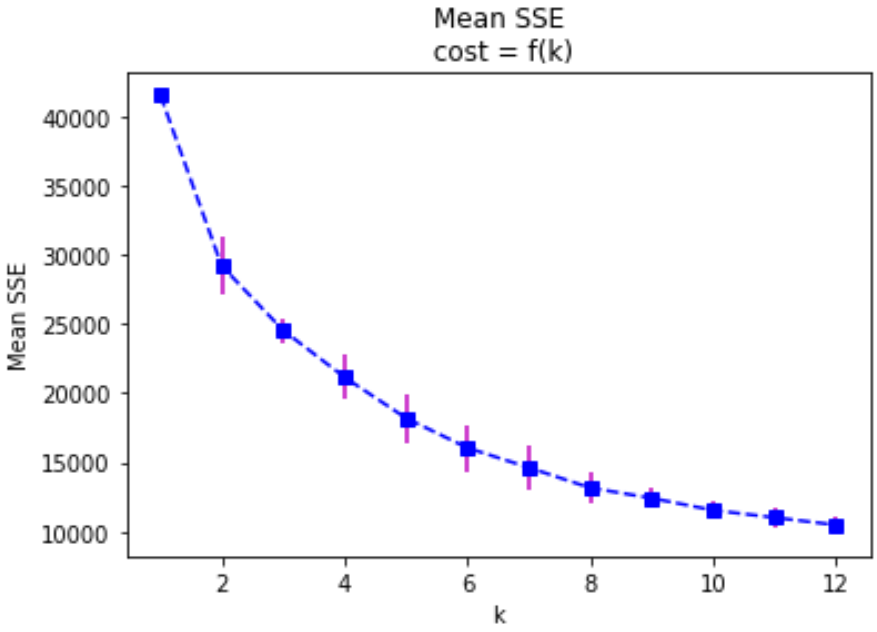
Angela Krontiris
CISC 6930: Data Mining
Assignment 4
Due: November 21

Question 1: K Means Clustering Python Results (runtime: 13 mins)

a) For each k = 1,2,...,12 compute the mean SSE, which we denote μk and the sample standard deviation of SSE, which we denote σk, over all 25 clustering runs for that value of k. Generate a line plot of the mean SSE (μk) as a function of k. Include error bars that indicate the 95% confidence interval: (μk − 2σk to μk +2σk).



b) Produce a table containing the 4 columns: k, μk, μk − 2σk and μk + 2σk for each of the values of k = 1,2,...,12.

	k	μk	σk	μ + 2σ	μ - 2σ
1	1	41562.000000	1.485199e-11	41562.000000	41562.000000
2	2	29205.015145	2.112634e+03	33430.283727	24979.746563
3	3	24551.016313	9.078673e+02	26366.750918	22735.281709
4	4	21157.861152	1.598800e+03	24355.461212	17960.261092
5	5	18205.595280	1.778095e+03	21761.785413	14649.405147
6	6	16052.341778	1.688840e+03	19430.021250	12674.662306
7	7	14626.526974	1.579643e+03	17785.813914	11467.240035
8	8	13161.110598	1.180318e+03	15521.745824	10800.475372
9	9	12416.079667	7.236662e+02	13863.412130	10968.747203
10	10	11542.334411	6.540337e+02	12850.401836	10234.266986
11	11	11013.098168	7.817020e+02	12576.502095	9449.694241
12	12	10481.910929	6.691490e+02	11820.208873	9143.612984

c) As k increases and approaches the total number of examples N, what value does the SSE approach? What problems does this cause in terms of using SSE to choose an optimal k?

When k(# of clusters) equals the number of samples in a dataset, every point in the data corresponds to its own cluster, total distortion is zero. In this case, the clusters will not be very meaningful.

d) Can you suggest another measure of cluster compactness and separation that might be more useful than SSE?

Silhouette analysis can be used as a graphical tool to plot a measure of how tightly grouped the samples in the clusters are. Applying the following three steps:

1. Calculate the **cluster cohesion,** ai, as the average distance between a sample, xi, and all the other points in the same cluster.
2. Calculate the **cluster separation**, bi, from the next closest cluster as the average distance between the sample, xi, and all the samples in the nearest cluster.
3. Calculate the **silhouette,** si, as the difference between cluster cohesion and separation divided by the greater of the two: si = bi – ai / max{bi, ai}

2_Dendrogram_Agglomerative_Clustering

November 20, 2018

0.0.1 Calculating pairwise distance matrix

In [5]:
```
from matplotlib import pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
import numpy as np

X = [[i] for i in [0,4,5,20,25,39,43,44]]
X_arr = np.array(X)
print(X_arr)
```

```
[[ 0]
 [ 4]
 [ 5]
 [20]
 [25]
 [39]
 [43]
 [44]]
```

In [6]:
```
n = X_arr.shape[0]
p = X_arr.shape[1]
dist = np.zeros((n, n))
for i in range(n):
    for j in range(n):
        s = 0
        for k in range(p):
            s += (X_arr[i, k] - X_arr[j, k])**2
        dist[i, j] = np.sqrt(s)
dist
```

Out[6]:
```
array([[  0.,   4.,   5.,  20.,  25.,  39.,  43.,  44.],
       [  4.,   0.,   1.,  16.,  21.,  35.,  39.,  40.],
       [  5.,   1.,   0.,  15.,  20.,  34.,  38.,  39.],
       [ 20.,  16.,  15.,   0.,   5.,  19.,  23.,  24.],
       [ 25.,  21.,  20.,   5.,   0.,  14.,  18.,  19.],
       [ 39.,  35.,  34.,  19.,  14.,   0.,   4.,   5.],
       [ 43.,  39.,  38.,  23.,  18.,   4.,   0.,   1.],
       [ 44.,  40.,  39.,  24.,  19.,   5.,   1.,   0.]])
```

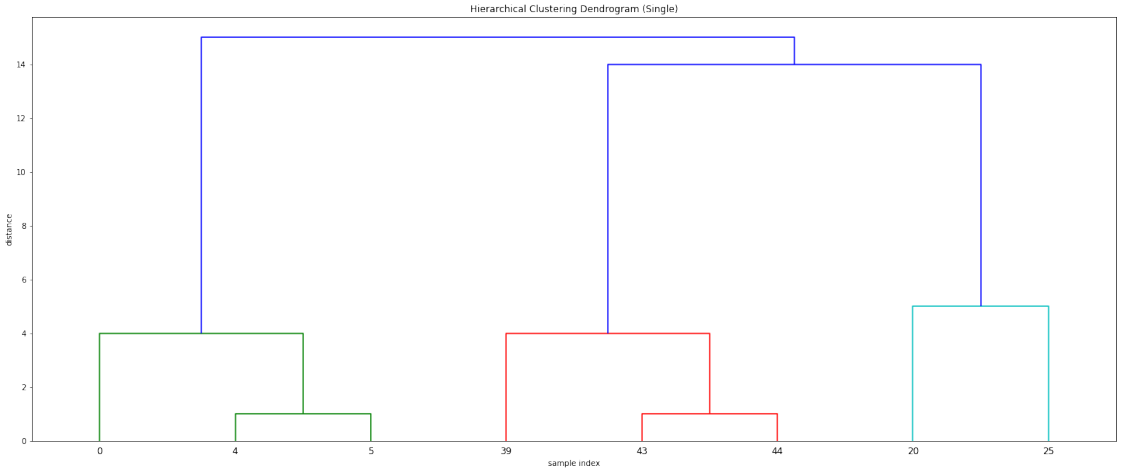
0.0.2 Plotting the dendrogram

A dendrogram is a visualization in form of a tree showing the order and distances of merges during the hierarchical clustering.

a) Build a dendrogram using the single-link bottom-up approach

```
In [7]: # Generate the linkage Matrix using single link method
        Z = linkage(X, 'single','euclidean')
        fig = plt.figure(figsize=(25,10))

        #create dendrogram
        labels = ['0','4','5','20','25','39','43','44']
        dendrogram = dendrogram(Z,labels=labels)
        plt.title('Hierarchical Clustering Dendrogram (Single)')
        plt.xlabel('sample index')
        plt.ylabel('distance')
        plt.show()
```



b) List the data points in the top two level clusters

- Top level cluster 1: [0,4,5]
- Top level cluster 2: [20,25,39,43,44]

$C_1 = \{(1,1), (2,2), (3,3)\}$

$C_2 = \{(5,2), (6,2), (7,2), (8,2), (9,2)\}$

(a) Calculate the mean of m_1 and m_2:

$$m_1 = \frac{\begin{pmatrix}1\\1\end{pmatrix} + \begin{pmatrix}2\\2\end{pmatrix} + \begin{pmatrix}3\\3\end{pmatrix}}{3} = \frac{\begin{pmatrix}6\\6\end{pmatrix}}{3} = \begin{pmatrix}2\\2\end{pmatrix}$$

$$m_2 = \frac{\begin{pmatrix}5\\2\end{pmatrix} + \begin{pmatrix}6\\2\end{pmatrix} + \begin{pmatrix}7\\2\end{pmatrix} + \begin{pmatrix}8\\2\end{pmatrix} + \begin{pmatrix}9\\2\end{pmatrix}}{5} = \frac{\begin{pmatrix}35\\10\end{pmatrix}}{5} = \begin{pmatrix}7\\2\end{pmatrix}$$

(b) Total mean vector m

$$m = \frac{\begin{pmatrix}1\\1\end{pmatrix} + \begin{pmatrix}2\\2\end{pmatrix} + \begin{pmatrix}3\\3\end{pmatrix} + \begin{pmatrix}5\\2\end{pmatrix} + \begin{pmatrix}6\\2\end{pmatrix} + \begin{pmatrix}7\\2\end{pmatrix} + \begin{pmatrix}8\\2\end{pmatrix} + \begin{pmatrix}9\\2\end{pmatrix}}{8} = \frac{\begin{pmatrix}41\\16\end{pmatrix}}{8} = \begin{pmatrix}5.125\\2\end{pmatrix}$$

(c) Scatter matrices, S_1 and S_2

$$= \left[\begin{pmatrix}1\\1\end{pmatrix} - \begin{pmatrix}2\\2\end{pmatrix}\right]\left[\begin{pmatrix}1\\1\end{pmatrix} - \begin{pmatrix}2\\2\end{pmatrix}\right]^T = \begin{pmatrix}-1\\-1\end{pmatrix}\begin{pmatrix}-1 & -1\end{pmatrix} = \begin{bmatrix}1 & 1\\1 & 1\end{bmatrix}$$

$$= \left[\begin{pmatrix}2\\2\end{pmatrix} - \begin{pmatrix}2\\2\end{pmatrix}\right]\left[\begin{pmatrix}2\\2\end{pmatrix} - \begin{pmatrix}2\\2\end{pmatrix}\right]^T = \begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$$

$$= \left[\begin{pmatrix}3\\3\end{pmatrix} - \begin{pmatrix}2\\2\end{pmatrix}\right]\left[\begin{pmatrix}3\\3\end{pmatrix} - \begin{pmatrix}2\\2\end{pmatrix}\right]^T = \begin{bmatrix}1 & 1\\1 & 1\end{bmatrix}$$

$$S_1 = \begin{bmatrix}1 & 1\\1 & 1\end{bmatrix} + \begin{bmatrix}0 & 0\\0 & 0\end{bmatrix} + \begin{bmatrix}1 & 1\\1 & 1\end{bmatrix} = \begin{bmatrix}2 & 2\\2 & 2\end{bmatrix}$$

$$= \left[\begin{pmatrix}5\\2\end{pmatrix} - \begin{pmatrix}7\\2\end{pmatrix}\right]\left[\begin{pmatrix}5\\2\end{pmatrix} - \begin{pmatrix}7\\2\end{pmatrix}\right]^T = \begin{pmatrix}-2\\0\end{pmatrix}\begin{pmatrix}-2 & 0\end{pmatrix} = \begin{bmatrix}4 & 0\\0 & 0\end{bmatrix}$$

$$= \left[\begin{pmatrix}6\\2\end{pmatrix} - \begin{pmatrix}7\\2\end{pmatrix}\right]\left[\begin{pmatrix}6\\2\end{pmatrix} - \begin{pmatrix}7\\2\end{pmatrix}\right]^T = \begin{pmatrix}-1\\0\end{pmatrix}\begin{pmatrix}-1 & 0\end{pmatrix} = \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix}$$

$$= \left[\begin{pmatrix}7\\2\end{pmatrix} - \begin{pmatrix}7\\2\end{pmatrix}\right]\left[\begin{pmatrix}7\\2\end{pmatrix} - \begin{pmatrix}7\\2\end{pmatrix}\right]^T = \begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$$

$$= \left[\begin{pmatrix}8\\2\end{pmatrix} - \begin{pmatrix}7\\2\end{pmatrix}\right]\left[\begin{pmatrix}8\\2\end{pmatrix} - \begin{pmatrix}7\\2\end{pmatrix}\right]^T = \begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}1 & 0\end{pmatrix} = \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix}$$

$$= \left[\begin{pmatrix}9\\2\end{pmatrix} - \begin{pmatrix}7\\2\end{pmatrix}\right]\left[\begin{pmatrix}9\\2\end{pmatrix} - \begin{pmatrix}7\\2\end{pmatrix}\right]^T = \begin{pmatrix}2\\0\end{pmatrix}\begin{pmatrix}2 & 0\end{pmatrix} = \begin{bmatrix}4 & 0\\0 & 0\end{bmatrix}$$

$$S_2 = \begin{bmatrix}4 & 0\\0 & 0\end{bmatrix} + \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix} + \begin{bmatrix}0 & 0\\0 & 0\end{bmatrix} + \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix} + \begin{bmatrix}4 & 0\\0 & 0\end{bmatrix} = \begin{bmatrix}10 & 0\\0 & 0\end{bmatrix}$$

d) Within cluster scatter Matrix, S_W

$$S_W = S_1 + S_2$$

$$S_W = \begin{bmatrix}2 & 2\\2 & 2\end{bmatrix} + \begin{bmatrix}10 & 0\\0 & 0\end{bmatrix} = \begin{bmatrix}12 & 2\\2 & 2\end{bmatrix}$$

e) Between scatter matrix, S_B

$$S_B = \sum_{i=1}^{N} N_i(\mu_i - M)(\mu_i - M)^T$$

$$S_1 = 3\left[\begin{pmatrix}2\\2\end{pmatrix} - \begin{pmatrix}5.125\\2\end{pmatrix}\right]\left[\begin{pmatrix}2\\2\end{pmatrix} - \begin{pmatrix}5.125\\2\end{pmatrix}\right]^T$$

$$= 3\begin{pmatrix}-3.125\\0\end{pmatrix}\begin{pmatrix}-3.125 & 0\end{pmatrix} = 3\begin{bmatrix}9.766 & 0\\0 & 0\end{bmatrix} = \begin{bmatrix}29.297 & 0\\0 & 0\end{bmatrix}$$

$$S_2 = 5\left[\begin{pmatrix}7\\2\end{pmatrix} - \begin{pmatrix}5.125\\2\end{pmatrix}\right]\left[\begin{pmatrix}7\\2\end{pmatrix} - \begin{pmatrix}5.125\\2\end{pmatrix}\right]^T$$

$$= 5\begin{pmatrix}1.875\\0\end{pmatrix}\begin{pmatrix}1.875 & 0\end{pmatrix} = 5\begin{bmatrix}3.516 & 0\\0 & 0\end{bmatrix} = \begin{bmatrix}17.578 & 0\\0 & 0\end{bmatrix}$$

$$S_B = S_1 + S_2 = \begin{bmatrix} 29.297 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 17.578 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 46.875 & 0 \\ 0 & 0 \end{bmatrix}$$

f) Scatter criterion, $\frac{tr(S_B)}{tr(S_W)}$

* Measures how good your clustering is. The higher the better.

Trace: sum along diagonals of the array

$$\text{Scatter criterion} = \frac{tr(S_B)}{tr(S_W)} = \frac{46.875}{14} = 3.348$$

4_DBSCAN

November 20, 2018

0.0.1 DBSCAN Algorithm

Unlike K-Means, DBSCAN does not require the number of clusters as a parameter. Rather it infers the number of clusters based on the data, and it can discover clusters of arbitrary shape (for comparison, K-Means usually discovers spherical clusters).

Two important parameters are required for DBSCAN: - Radius (): How far apart should observations be, to be considered in the same cluster? - Minimum # of Samples: How many observations should be in the radius of a data point?

Using these two parameters, DBSCAN categories the data points into three categories:

- Core Points: A data point p is a core point if dist(p,) [-neighborhood of p] contains at least minPts ; |dist(p,)| >= minPts.
- Border Points: A data point *q is a border point if dist(q,) contains less than minPts data points, but q is reachable from some core point p.
- Outlier: A data point o is an outlier if it is neither a core point nor a border point. Essentially, this is the "other" class.

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# DBSCAN with parameters
eps = (2)**0.5
min_samples = 3
pts = [(0, 0),(1, 2),(1, 6),(2, 3),(3, 4),(5, 1),(4, 2),(5, 3),(6, 2),(7, 4)]

# X= np.array(pts)
# print("Array of points:\n",X)

# convert pts to a dataframe
df = pd.DataFrame(pts,columns=['x','y'])
df
```

Out[2]:

	x	y
0	0	0
1	1	2
2	1	6
3	2	3

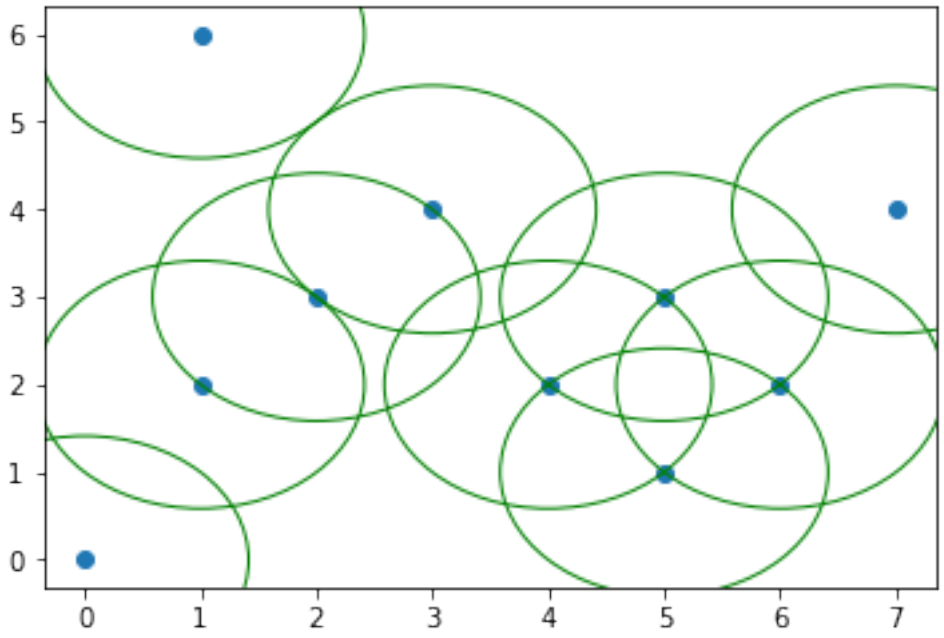
```
4  3  4
5  5  1
6  4  2
7  5  3
8  6  2
9  7  4
```

```
In [3]: # Plot points
        fig, ax = plt.subplots()

        plt.scatter(df['x'],df['y'])

        X = df['x']
        Y = df['y']

        for index in range(len(X)):
            neighbors = plt.Circle((X[index], Y[index]), eps, facecolor='none', edgecolor='gre
            ax.add_artist(neighbors)
        plt.show()
```



a) List the clusters in terms of their points

- Core: Data points lying within the cluster itself: data points which satisfy the minimum samples requirement
- Edge: Data points lying outside the cluster: data points that are within the radius of a core point yet do not satisfy the minimum samples requirement.

- Noise: Data points that are bad training observations: data points that do not contain the minimum number of samples nor are within the radius of a core point.

cluster 1: [(1,2), (2,3), (3,4)] Cluster 1 has one core point, (2,3)

cluster 2: [(4,2), (5,1), (5,3), (6,2)] Each point in cluster 2 is a core point. If more than one core point fall within the same circle, then it becomes one big cluster.

b) What are the density-connected points? The density connected points are the core points within cluster 2. It is also a symmetric relationship.

density connected points: [(4,2), (5,1), (5,3), (6,2)]

c) What points (if any) does DBSCAN consider as noise? Noise points are neither a core point nor a border point.

noise points : [(0,0), (1,6), (7,4)]

5_NaiveBayes_Classifier_Performance

November 20, 2018

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Create dictionary
data = {'Instance':[1,2,3,4,5,6,7,8,9,10],
        'True_Class_Label': ['P','N','P','P','N','P','N','N','N','P'],
        'Predicted_Probability_Positive_Class': [0.95,0.85,0.78,0.66,0.60,0.55,0.43,0.4
       }

# Convert data dictionary to dataframe
df = pd.DataFrame(data)
# df
```

In [3]:
```
# Create a predicted class label (positive or negative class of the predicted probabil

df['Pred_Class_Label'] = np.where(df['Predicted_Probability_Positive_Class'] >= 0.5, 'P
df
```

Out[3]:

	Instance	Predicted_Probability_Positive_Class	True_Class_Label	Pred_Class_Label
0	1	0.95	P	P
1	2	0.85	N	P
2	3	0.78	P	P
3	4	0.66	P	P
4	5	0.60	N	P
5	6	0.55	P	P
6	7	0.43	N	N
7	8	0.42	N	N
8	9	0.41	N	N
9	10	0.40	P	N

```
5                P
6                N
7                N
8                N
9                N
```

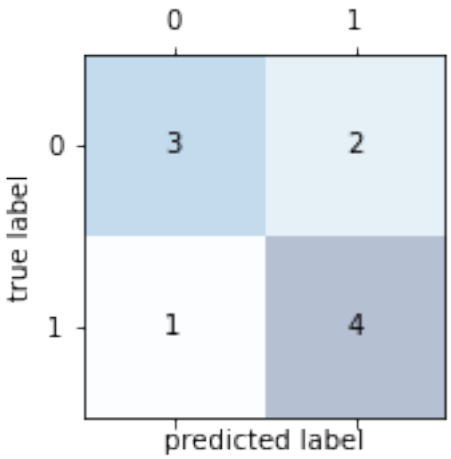
a) Confusion Matrix

```
In [4]: from sklearn.metrics import confusion_matrix
        y_test = df['True_Class_Label']
        y_pred = df['Pred_Class_Label']
        confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)
        print(confmat)
```

```
[[3 2]
 [1 4]]
```

```
In [5]: #Plotting confusion matrix using Matplotlib's matshow function

        fig, ax = plt.subplots(figsize=(2.5,2.5))
        ax.matshow(confmat, cmap=plt.cm.Blues, alpha=0.3)
        for i in range(confmat.shape[0]):
            for j in range(confmat.shape[1]):
                ax.text(x=j, y=i, s=confmat[i,j], va='center',ha='center')
        plt.xlabel('predicted label')
        plt.ylabel('true label')
        plt.show()
```



b) Prediction Accuracy ACC = (TP + TN)/(FP + FN + TP + TN)

```
In [6]: tp,tn,fn,fp = 4, 3, 1, 2

        ACC = (tp + tn)/(fp + fn + tp + tn)
        print("Accuracy: %.2f" % ACC)
```

```
Accuracy: 0.70
```

c) Precision (PRE) and Recall (REC) and F1 Score

- PRE = TP/(TP + FP)
- REC = TP/(FN + TP)
- F1 = 2 * ((PRE * REC)/(PRE + REC))

```
In [11]: PRE = tp/(tp + fp)
         print("Precision: %.2f" % PRE)

         REC = tp/(fn + tp)
         print("Recall: %.2f" % REC)

         F1 = 2 * ((PRE * REC)/(PRE + REC))
         print("F1-score: %.2f" % F1)
```

```
Precision: 0.67
Recall: 0.80
F1-score: 0.73
```

d) Specificity

- SPE = TN/(TN+FP)
- Specificity: = Correctly Rejected/ Correctly Rejected + Mistakenly Selected = Correctly Rejected/ Total poor candidates who actually deserved Rejection

```
In [9]: SPE = tn/(tn + fp)
        print("Specificity: %.2f" % SPE)
```

```
Specificity: 0.60
```