In [25]:

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn import preprocessing
```

In [27]:

```
creditData = pd.read_csv("credit_data.csv")
```

In [28]:

```
creditData.head()
```

Out[28]:

	clientid	income	age	loan	LTI	default
0	1	66155.925095	59.017015	8106.532131	0.122537	0
1	2	34415.153966	48.117153	6564.745018	0.190752	0
2	3	57317.170063	63.108049	8020.953296	0.139940	0
3	4	42709.534201	45.751972	6103.642260	0.142911	0
4	5	66952.688845	18.584336	8770.099235	0.130989	1

In [29]:

```
creditData.describe()
```

Out[29]:

	clientid	income	age	loan	LTI	default
count	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
mean	1000.500000	45331.600018	40.927143	4444.369695	0.098403	0.141500
std	577.494589	14326.327119	13.262450	3045.410024	0.057620	0.348624
min	1.000000	20014.489470	18.055189	1.377630	0.000049	0.000000
25%	500.750000	32796.459717	29.062492	1939.708847	0.047903	0.000000
50%	1000.500000	45789.117313	41.382673	3974.719419	0.099437	0.000000
75%	1500.250000	57791.281668	52.596993	6432.410625	0.147585	0.000000
max	2000.000000	69995.685578	63.971796	13766.051239	0.199938	1.000000

In [30]:

```
print(creditData.corr())
```

```
          clientid    income       age      loan       LTI   default
clientid  1.000000  0.039280 -0.030341  0.018931  0.002538 -0.020145
income    0.039280  1.000000 -0.034984  0.441117 -0.019862  0.002284
age      -0.030341 -0.034984  1.000000  0.006561  0.021588 -0.444765
loan      0.018931  0.441117  0.006561  1.000000  0.847495  0.377160
LTI       0.002538 -0.019862  0.021588  0.847495  1.000000  0.433261
default  -0.020145  0.002284 -0.444765  0.377160  0.433261  1.000000
```

In [31]:

```
features = creditData[['income','age','loan']]
targetVariables = creditData.default
```

In [33]:

```
features = preprocessing.MinMaxScaler().fit_transform(features) # there is a huge difference
```

In [34]:

```
features
```

Out[34]:

```
array([[0.9231759 , 0.89209175, 0.58883739],
       [0.28812165, 0.65470788, 0.47682695],
       [0.74633429, 0.9811888 , 0.58262011],
       ...,
       [0.48612202, 0.21695807, 0.40112895],
       [0.47500998, 1.        , 0.1177903 ],
       [0.98881367, 0.82970913, 0.53597028]])
```

In [31]:

```
featureTrain, featureTest, targetTrain, targetTest = train_test_split(features, targetVariables, test_size =
0.3)
```

In [32]:

```
model = KNeighborsClassifier(n_neighbors=4) # 4 is the k value
fittedModel = model.fit(featureTrain, targetTrain)
predictions = fittedModel.predict(featureTest)
```

In [33]:

```
cross_valid_scores = []
```

In [34]:

```
for k in range(1, 100):
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, features, targetVariables, cv=10, scoring='accuracy')
    cross_valid_scores.append(scores.mean())
```

In [35]:

```
print("Optimal k with cross_valiation: ", np.argmax(cross_valid_scores))
```

```
Optimal k with cross_valiation:  28
```

In [37]:

```
print(confusion_matrix(targetTest, predictions))
print(accuracy_score(targetTest, predictions))
```

```
[[522   1]
 [  8  69]]
0.985
```