CoxAssignment03

Task: This assignment will help you understand evaluation metrics. You will also be required to implement these evaluation metrics.

Dataset: Please use the fictituous data for this exercise.

In [2]:
```
%matplotlib inline
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import accuracy_score, roc_curve, auc
import matplotlib.pylab as plt
from dmba import regressionSummary, classificationSummary, liftChart, gainsChart
```

Confusion Matrix

A data mining routine has been applied to a transaction dataset and has classified 88 records as fraudulent (30 correctly so) and 952 as non-fraudulent (920 correctly so). Construct the confusion matrix and calculate the overall error rate.

Construct the confusion matrix.

Answer:

```
classification confusion matrix
|-------------------------------------------------------------------
--|
|                   |                Predicted Class
|
|-------------------------------------------------------------------
--|
| Actual Class      |         Fraudulant(1)    |   Non-fraudulant
(0) |
|-------------------------------------------------------------------
--|
| Fraudulant (1)    |             30           |         3
|
|-------------------------------------------------------------------
--|
| Non-fraudulant (0) |             1           |        920
|
|-------------------------------------------------------------------
```

--|

Calculate the Error Rate for the confusion matrix shown above.

In [3]:
```
((88 - 30) + (952 - 920)) / (88 + 952)
```

Out[3]:
```
0.08653846153846154
```

Error Rate, Specificity, and Sensitivity

In [24]:
```
# create a data frame of data

data = {'Propensity': [0.03, 0.52, 0.38, 0.82, 0.33, 0.42, 0.55, 0.59, 0.09, 0.21, 0.4
                       0.29, 0.08, 0.02],
        'Actual': [0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]}

# convert to data frame
df = pd.DataFrame(data)
```

Create a confusion matrix where values are 1 if they are greater than .25, and 0 for anything else.

In [38]:
```
class_names = [1, 0]
```

In [39]:
```
predicted = [1 if p > .25 else 0 for p in df.Propensity]
classificationSummary(df.Actual, predicted, class_names=class_names)
```

```
Confusion Matrix (Accuracy 0.6000)

       Prediction
Actual 1 0
     1 9 8
     0 0 3
```

Calculate error rate, sensitivity, and specificity for the confusion matrix.

In [43]:
```
8/20
```

Out[43]:
```
0.4
```

In [42]:
```
3/3
```

Out[42]:
```
1.0
```

In [41]:
```
9/17
```

Out[41]:
```
0.5294117647058824
```

In [5]:
```

```

```
Error Rate =  0.4

Sensitivity= 1.0

Specificity =  0.5294117647058824
```

Create a confusion matrix where values are 1 if they are greater than .5, and 0 for anything else.

In [30]:
```
# cutoff = 0.5
predicted = [1 if p > .5 else 0 for p in df.Propensity]
classificationSummary(df.Actual, predicted, class_names=class_names)
```

```
Confusion Matrix (Accuracy 0.9000)

       Prediction
Actual  1  0
     1 15  2
     0  0  3
```

Calculate error rate, sensitivity, and specificity for the confusion matrix.

In [44]:
```
2/20
```

Out[44]:
```
0.1
```

In [45]:
```
3/3
```

Out[45]:
```
1.0
```

In [46]:
```
15/17
```

Out[46]:
```
0.8823529411764706
```

In [7]:
```

```

```
Error Rate =  0.1

Sensitivity= 1.0

Specificity =  0.8823529411764706
```

Create a confusion matrix where values are 1 if they are greater than .75, and 0 for anything else.

In [31]:
```
# cutoff = 0.75
predicted = [1 if p > .75 else 0 for p in df.Propensity]
classificationSummary(df.Actual, predicted, class_names=class_names)
```

```
Confusion Matrix (Accuracy 0.9500)

       Prediction
Actual  1  0
     1 17  0
     0  1  2
```

Calculate error rate, sensitivity, and specificity for the confusion matrix.

In [47]:
```
1/20
```

Out[47]:
```
0.05
```

In [48]:
```
2/3
```

Out[48]:
```
0.6666666666666666
```

In [49]:
```
17/17
```

Out[49]:
```
1.0
```

In [9]:
```

```

```
Error Rate =  0.05

Sensitivity= 0.6666666666666666

Specificity =  1.0
```

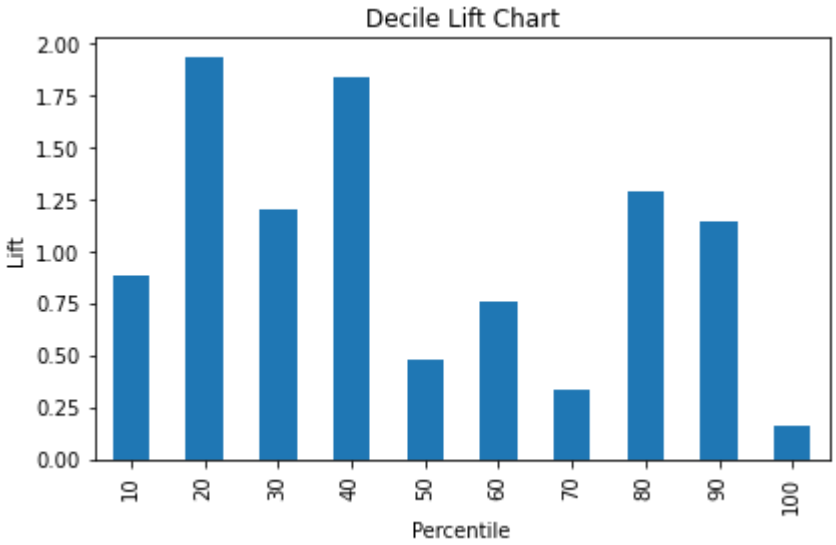
Please answer below, which cutoff performed the best based on the evaluation metrics? Please interpret the evaluation metrics.

The evaluation matrix that performed the best was the one where the cut off was .75. It showed accurary at 95% while having the lowest error rate of the 3 options at 5%. The higher the sensitivety the greater the model is. Having a high spcificity for this data shows that our model was to very accuratley classify data into the appropriate designations.

Lift Chart

Please create a lift chart for the dataframe. Interpret the lift chart.

In [57]:
```
# decile lift chart
liftChart(df.Propensity, labelBars=False)
plt.tight_layout()
plt.show()
```



In the 20th and 40th percentilce saw greater lift then in the 80th and 90th percentile that still show high lift compared to the middle and end of the chart. This means that number in the 20th and 40th percentile have a higher average use compared to other percentiles.

In [ ]: