# PA2 - ID3 Decision Tree

Zhanchong Deng

A15491777

### Necessary Imports

In [1]:

```
%load_ext autoreload
%autoreload 2
import numpy as np
import pandas as pd
import ID3 as pa2
from scipy.stats import entropy
from scipy import stats
import matplotlib.pyplot as plt
```

### Import Data

In [2]:

```
training = pa2.loadData('pa2train.txt')
validation = pa2.loadData('pa2validation.txt')
test = pa2.loadData('pa2test.txt')
```

### Training, without Pruning

In [3]:

```
tree = pa2.id3()
```

```
Created a new ID3 tree
```

In [4]:

```
%time tree.fit(training)
```

```
Wall time: 15.9 s
```

### 1) Visualizing resulted Tree

Below is my representation of the tree

- Number of **tabs** indicates which level the node is at.
- For non leaves: (label)[rule](# of data in this node)
- For leaves: (label)[predicted label](# of data in this node)

In [5]:

```
# If root does not count as a level
print(tree.printTreeAt(3))
```

```
(root)[is feature at 5 < 0.5?](num_data:2000)
        (yes)[is feature at 1 < 415000.0?](num_data:1319)
                (yes)[is feature at 17 < 2506.5?](num_data:1284)
                (no)[is feature at 21 < 208.0?](num_data:35)
        (no)[is feature at 5 < 1.5?](num_data:681)
                (yes)[is feature at 20 < 584.5?](num_data:292)
                (no)[is feature at 21 < 2006.0?](num_data:389)
```

### 2) Validation/Test Errors

Validation Error:

In [6]:

```
tree.error(validation)
```

Out[6]:

```
0.179
```

Test Error:

In [7]:

```
tree.error(test)
```

Out[7]:

```
0.173
```

### 3) Pruning decision tree with Greedy apprach in BFS order

Prune 1 and 2 nodes with validation/test errors:

In [8]:

```
tree.pruneTree(validation, test, 2)
```

```
Pruned 1 time(s) with error:
        Validation error: 0.122
        Test error: 0.117
Pruned 2 time(s) with error:
        Validation error: 0.107
        Test error: 0.103
```

Tree after two pruned nodes:

In [9]:

```
print(tree.printTree())
```

```
(root)[is feature at 5 < 0.5?](num_data:2000)
        (yes)[0.0](num_data:1319)
        (no)[1.0](num_data:681)
```

### 4) Most prominent feature:

The most prominent feature must be the feature selected as threshold at root.

In [10]:

```
features_name = open('pa2features.txt')
features_name.seek(0)
columns = features_name.read().split('\n')[:-1]
```

In [11]:

```
columns[tree.root.feature]
```

Out[11]:

```
'PAYMENT_DELAY_SEPTEMBER'
```

Done.