CS540 Programming Homework 3 Solution

In [1]:
```
from collections import Counter, OrderedDict
import pandas as pd
import numpy as np
import itertools
import math
import ast
import sys


data = pd.read_csv('tmdb_5000_movies.csv')
print(data.shape)
```

```
(4803, 20)
```

In [2]:
```
# Extract the columns for budget, genres, runtime, vote_average, vote_count, revenue
data = data[['budget', 'genres', 'runtime', 'vote_average', 'vote_count', 'revenue']]
print(data.shape)
```

```
(4803, 6)
```

In [3]:
```
# Remove all data points with missing features
data = data.dropna(axis=0, how='any')
print(data.shape)
```

```
(4801, 6)
```

In [4]:
```
data = data.loc[(data != 0).all(axis=1)]
print(data.shape)
```

```
(3227, 6)
```

In [5]:
```
data = data[data.astype(str)['genres'] != '[]']
data['genres'] = data['genres'].map(lambda x: [d['name']
                                               for d in ast.literal_eval(x)])
print(data.shape)
```

```
(3226, 6)
```

In [6]:
```
# convert revenue column into binary variable
thresh = 10000000
y = data['revenue'].map(lambda x: x > thresh).astype(int).values
data = data.drop(['revenue'], axis=1)
print(data.shape)
```

```
(3226, 5)
```

In [7]:
```
# convert budget, genres, runtime, vote_average, vote_count columns
# into categorical data
epsilon = 0.01

# get a random k, where 2 <= k <= 10
k_list = np.random.randint(low=2, high=11, size=(5,), dtype=np.int8)
min_list = data.min(axis=0)
max_list = data.max(axis=0)

# get a random k, where 2 <= k <= 9 and add 'other' as
# additional category with most common k genres
genres_list = [name for name, count in Counter(data['genres'].sum()).most_common(
    np.random.randint(low=2, high=10))] + ['Other']
k_list[1] = len(genres_list)
```

In [8]:
```
h_y = -sum((val/len(y)*math.log2(val/len(y))
            for _, val in dict(Counter(y)).items()))    # binary entropy

output = pd.DataFrame((np.zeros((5, 3), dtype=object)),
                      columns=['k', 'info_gain', 'num_pos/neg'])
output.index = ['budget', 'genres', 'runtime', 'vote_average', 'vote_count']
output['k'] = k_list

for ind, col in enumerate(data):
    # convert features into categorical data
    if col == 'genres':
        # map to 1 - K + 1 categories, where K is a random number between 2 and 9
        data[col] = data[col].map(lambda x: genres_list.index(x[0])
                                  if x[0] in genres_list else k_list[ind]-1)
    else:
        # map to 1 - K categories, where K is a random number between 2 and 10
        data[col] = data[col].map(lambda x: math.floor((x - min_list[ind])/(
                (max_list[ind] + epsilon - min_list[ind]) / k_list[ind])))

    # count number of positive and negative examples for each feature
    d = OrderedDict.fromkeys(list(itertools.product(
    range(0, 2), range(0, k_list[ind]))), 0)
```

```
            d.update(dict(Counter(list(zip(y, data[col])))))

            for key in list(itertools.product([0], range(k_list[ind]))):
                d.move_to_end(key)

            d = OrderedDict(sorted(d.items(), key=lambda x: x[0][1]))
            output.at[col, 'num_pos/neg'] = list(d.values())

            # compute the information gain
            count_x = dict(Counter(data[col]))
            h_yx = - sum(val/len(y)*math.log2(val/count_x[key[1]])
                         for key, val in d.items() if val != 0)  # conditional entropy
            output.at[col, 'info_gain'] = format(h_y - h_yx, '.8f')
```

In [9]:
```
output
```

Out[9]:

	k	info_gain	num_pos/neg
budget	7	0.06780851	[1851, 549, 554, 8, 172, 0, 70, 0, 19, 0, 2, 0...
genres	4	0.01141179	[556, 189, 518, 116, 97, 21, 1498, 231]
runtime	3	0.00245520	[2451, 531, 217, 25, 1, 1]
vote_average	7	0.00474029	[3, 5, 13, 9, 122, 38, 592, 110, 1107, 219, 70...
vote_count	9	0.04289481	[2084, 549, 347, 7, 131, 1, 56, 0, 29, 0, 11, ...

In [10]:
```
output['result'] = output['num_pos/neg'].map(lambda x: ','.join(map(str, x)))
output[['k', 'info_gain', 'result']].apply(
    lambda x: ','.join(x.fillna('').map(str)), axis=1).to_csv(
    'output.txt', header=None, index=None, sep=' ')
```