Data sampling and Class balancing

In [1]:

```
import pandas as pd
import numpy as np
```

In [8]:

```
def sample_data(file_path, result_file, size):
    seed = 42
    balance_samples = pd.DataFrame()
    data = pd.read_csv(file_path, sep=' ', header=None)
    data.drop(data.columns[-1], axis=1, inplace=True)  # remove the last empty column
    data = data.applymap(lambda x: x.split(':')[1] if ':' in str(x) else x)
    data[0] = data[0].astype(int)
    data.iloc[:, 1:] = data.iloc[:, 1:].astype(float)

    query_ids = data[1]
    rng = np.random.RandomState(seed)
    unique_qid = np.unique(query_ids)
    if size < len(unique_qid):
        qid_mask = rng.permutation(len(unique_qid))[:size]
        subset_mask = np.in1d(query_ids, unique_qid[qid_mask])
        sample_data = data[subset_mask]
    else:
        sample_data = data

    # Filter balance lable. two document for each label for given query id
    df_resampled = pd.DataFrame()
    df_resampled_qid = pd.DataFrame()
    for qid in unique_qid:
        df_qid = sample_data[sample_data[1] == qid]

        for rel_score in range(5):
            df_rel_score = df_qid[df_qid[0] == rel_score]
            if len(df_rel_score) > 2:
                df_rel_score = df_rel_score.sample(n=2)
            df_resampled_qid = pd.concat([df_resampled_qid, df_rel_score])
        if len(df_resampled_qid) >= 10:
            df_resampled_qid = df_resampled_qid.sample(n=8)
        df_resampled = pd.concat([df_resampled, df_resampled_qid])

    df_resampled.to_csv(result_file, index=False)
```

In [9]:

```
sample_data = sample_data('train.txt', 'fold1_train_sample.csv',500)
```

```
/tmp/ipykernel_531531/1738193240.py:8: FutureWarning: In a future version, `df.iloc[:, i] = newvals` will attempt to set the values inplace instead of always setting a new array. To retain the old behavior, use either `df[df.colu
mns[i]] = newvals` or, if columns are non-unique, `df.isetitem(i, newvals)`
  data.iloc[:, 1:] = data.iloc[:, 1:].astype(float)
```

In [22]:

```
sample_data.head()
```

Out[22]:

	0	1	2	3	4	5	6	7	8	9	...	128	129	130	131	132	133	134	135	136	137
3308	0	445.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	...	27.0	0.0	2.0	124.0	55802.0	15.0	8.0	0.0	0.0	0.0
3309	0	445.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	38.0	0.0	0.0	266.0	1124.0	79.0	159.0	0.0	0.0	0.0
3310	0	445.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	...	36.0	0.0	1.0	153.0	437.0	1.0	45.0	0.0	0.0	0.0
3311	0	445.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	40.0	2.0	0.0	3031.0	1462.0	74.0	8.0	1.0	0.0	0.0
3312	0	445.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	22.0	19.0	0.0	16549.0	627.0	63.0	11.0	0.0	0.0	0.0

5 rows × 138 columns

In [41]:

```
sample_data[1] = sample_data[1].astype(int)
unique_qid = sample_data[1].unique()
```

```
100
```

In [42]:

```
unique_qid[:5]
```

Out[42]:

```
array([ 445,  850,  985, 1045, 1060])
```

In [46]:

```
df_resampled[df_resampled[1] == 1045]
```

Out[46]:

	0	1	2	3	4	5	6	7	8	9	...	128	129	130	131	132	133	134	135	136	137
7843	0	1045	4.0	0.0	1.0	1.0	4.0	1.00	0.0	0.25	...	53.0	9.0	49.0	13645.0	48184.0	29.0	40.0	0.0	0.0	0.000000
7841	0	1045	4.0	0.0	1.0	2.0	4.0	1.00	0.0	0.25	...	51.0	0.0	49.0	934.0	48184.0	10.0	62.0	0.0	0.0	0.000000
7815	1	1045	3.0	0.0	2.0	1.0	3.0	0.75	0.0	0.50	...	53.0	646.0	6.0	1018.0	54534.0	3.0	1.0	0.0	2.0	67.400000
7770	1	1045	4.0	0.0	2.0	1.0	4.0	1.00	0.0	0.50	...	67.0	1774.0	10.0	3146.0	57539.0	1.0	5.0	0.0	21.0	75.350000
7777	2	1045	4.0	0.0	2.0	1.0	4.0	1.00	0.0	0.50	...	65.0	559.0	10.0	3664.0	57919.0	4.0	5.0	0.0	30.0	82.998333
7830	2	1045	4.0	0.0	2.0	1.0	4.0	1.00	0.0	0.50	...	56.0	10154.0	9.0	2916.0	58181.0	4.0	5.0	0.0	35.0	48.554286

6 rows × 138 columns

In [9]:

```
sample_data('test.txt', 'fold1_test_sample.csv',100)
```

```
/tmp/ipykernel_561729/2608011632.py:7: FutureWarning: In a future version, `df.iloc[:, i] = newvals` will attempt to set the values inplace instead of always setting a new array. To retain the old behavior, use either `df[df.colu
mns[i]] = newvals` or, if columns are non-unique, `df.isetitem(i, newvals)`
  data.iloc[:, 1:] = data.iloc[:, 1:].astype(float)
```