Preprocessing

July 29, 2024

[1]:
```
import numpy as np
import pandas as pd
from sklearn import preprocessing
```

[2]:
```
raw_data = pd.read_csv('Audiobooks_data.csv', header=None)
raw_data
```

[2]:

	0	1	2	3	4	5	6	7	8	9	10	11
0	994	1620.0	1620	19.73	19.73	1	10.00	0.99	1603.8	5	92	0
1	1143	2160.0	2160	5.33	5.33	0	8.91	0.00	0.0	0	0	0
2	2059	2160.0	2160	5.33	5.33	0	8.91	0.00	0.0	0	388	0
3	2882	1620.0	1620	5.96	5.96	0	8.91	0.42	680.4	1	129	0
4	3342	2160.0	2160	5.33	5.33	0	8.91	0.22	475.2	0	361	0
…	…	…	…	…	…	..	…	…	…	..	…	..
14079	28220	1620.0	1620	5.33	5.33	1	9.00	0.61	988.2	0	4	0
14080	28671	1080.0	1080	6.55	6.55	1	6.00	0.29	313.2	0	29	0
14081	31134	2160.0	2160	6.14	6.14	0	8.91	0.00	0.0	0	0	0
14082	32832	1620.0	1620	5.33	5.33	1	8.00	0.38	615.6	0	90	0
14083	251	1674.0	3348	5.33	10.67	0	8.91	0.00	0.0	0	0	1

[14084 rows x 12 columns]

[3]:
```
unscaled_inputs = raw_data.drop(columns=[0, raw_data.columns[-1]])
unscaled_inputs
```

[3]:

	1	2	3	4	5	6	7	8	9	10
0	1620.0	1620	19.73	19.73	1	10.00	0.99	1603.8	5	92
1	2160.0	2160	5.33	5.33	0	8.91	0.00	0.0	0	0
2	2160.0	2160	5.33	5.33	0	8.91	0.00	0.0	0	388
3	1620.0	1620	5.96	5.96	0	8.91	0.42	680.4	1	129
4	2160.0	2160	5.33	5.33	0	8.91	0.22	475.2	0	361
…	…	…	…	…	..	…	…	…	..	…
14079	1620.0	1620	5.33	5.33	1	9.00	0.61	988.2	0	4
14080	1080.0	1080	6.55	6.55	1	6.00	0.29	313.2	0	29
14081	2160.0	2160	6.14	6.14	0	8.91	0.00	0.0	0	0
14082	1620.0	1620	5.33	5.33	1	8.00	0.38	615.6	0	90
14083	1674.0	3348	5.33	10.67	0	8.91	0.00	0.0	0	0

```
[14084 rows x 10 columns]
```

[4]:
```
targets = raw_data.iloc[:,-1]
targets
```

[4]:
```
0        0
1        0
2        0
3        0
4        0
        ..
14079    0
14080    0
14081    0
14082    0
14083    1
Name: 11, Length: 14084, dtype: int64
```

0.0.1 Balance data

[5]:
```
targets_positive = int(np.sum(targets))
targets_zero = 0
i_to_remove = []
for i in range(targets.shape[0]):
    if targets[i] == 0:
        targets_zero +=1
        if targets_zero > targets_positive:
            i_to_remove.append(i)

unscaled_inputs_equal_priors = np.delete(unscaled_inputs, i_to_remove, axis=0)
targets_equal_priors = np.delete(targets, i_to_remove, axis=0)
```

0.0.2 Standardize inputs

[6]:
```
scaled_inputs = preprocessing.scale(unscaled_inputs_equal_priors)
scaled_inputs
```

[6]:
```
array([[ 0.21053387, -0.18888517,  1.97823887, …,  4.80955413,
         11.83828419,  0.09415043],
       [ 1.27894497,  0.41646744, -0.39082475, …, -0.41569922,
        -0.20183481, -0.80255852],
       [ 1.27894497,  0.41646744, -0.39082475, …, -0.41569922,
        -0.20183481,  2.979214  ],
       …,
       [ 1.27894497,  0.41646744, -0.39082475, …, -0.41569922,
        -0.20183481, -0.7440775 ],
       [ 0.31737498,  1.7482432 ,  0.04679395, …, -0.41569922,
```

```
 -0.20183481, -0.80255852],
[ 0.31737498,  1.7482432 , -0.39082475, …, -0.41569922,
 -0.20183481, -0.80255852]])
```

0.0.3 Shuffle data

[7]:
```
shuffled_indices = np.arange(scaled_inputs.shape[0])
np.random.shuffle(shuffled_indices)
shuffled_inputs = scaled_inputs[shuffled_indices]
shuffled_targets = targets_equal_priors[shuffled_indices]
shuffled_targets
```

[7]:
```
array([1, 0, 1, …, 1, 1, 0], dtype=int64)
```

0.0.4 Split data

[8]:
```
samples_count = shuffled_inputs.shape[0]

train_samples_count = int(0.8*samples_count)
validation_samples_count = int(0.1*samples_count)
test_samples_count = samples_count - train_samples_count -␣
 ↪validation_samples_count

train_inputs = shuffled_inputs[:train_samples_count]
train_targets = shuffled_targets[:train_samples_count]

validation_inputs = shuffled_inputs[train_samples_count:␣
 ↪train_samples_count+validation_samples_count]
validation_targets = shuffled_targets[train_samples_count:␣
 ↪train_samples_count+validation_samples_count]

test_inputs = shuffled_inputs[train_samples_count+validation_samples_count:]
test_targets = shuffled_targets[train_samples_count+validation_samples_count:]

print(np.sum(train_targets), train_samples_count, np.sum(train_targets)/
 ↪train_samples_count)
print(np.sum(validation_targets), validation_samples_count, np.
 ↪sum(validation_targets)/validation_samples_count)
print(np.sum(test_targets), test_samples_count, np.sum(test_targets)/
 ↪test_samples_count)
```

```
1789 3579 0.49986029617211514
232 447 0.5190156599552572
216 448 0.48214285714285715
```

[9]:
```
np.savez('Audiobooks_data_train', inputs=train_inputs, targets=train_targets)
```

```
np.savez('Audiobooks_data_validation', inputs=validation_inputs,␣
  ↪targets=validation_targets)
np.savez('Audiobooks_data_test', inputs=test_inputs, targets=test_targets)
```

[ ]: