In [1]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

In [2]:
```
import pandas as pd
import keras.utils as keras
from sklearn import preprocessing
import numpy as np
```

```
Using TensorFlow backend.
```

In [3]:
```
## Read data from csv file 'student-mat.csv'
math_data = pd.read_csv('student-mat.csv', sep=';')

## Read data from csv file 'student-por.csv'
por_data = pd.read_csv('student-por.csv', sep=';')

## Two datasets are similar except for the final 4 columns.
math_data.head()
por_data.head()

## shape
math_data.shape
por_data.shape
```

Out[3]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

Out[3]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

Out[3]: (395, 33)

Out[3]: (649, 33)

In [4]:
```
## Rename columns.
math_data = math_data.rename(columns= {'absences': 'm_absences',
                                       'G1': 'M1',
                                       'G2': 'M2',
                                       'G3': 'M3'})


por_data = por_data.rename(columns= {'absences': 'p_absences',
                                       'G1': 'P1',
                                       'G2': 'P2',
                                       'G3': 'P3'})
```

In [5]:
```
math_data.head()
por_data.head()
```

Out[5]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

Out[5]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

In [6]:
```
math_data['school'].value_counts()
por_data['school'].value_counts()
```

Out[6]:
```
GP    349
MS     46
Name: school, dtype: int64
```

Out[6]:
```
GP    423
MS    226
Name: school, dtype: int64
```

In [22]:
```
math_data['Mjob'].value_counts()
math_data['Fjob'].value_counts()
por_data['Mjob'].value_counts()
por_data['Fjob'].value_counts()
```

Out[22]:
```
other       141
services    103
at_home      59
teacher      58
health       34
Name: Mjob, dtype: int64
```

Out[22]:
```
other       217
services    111
teacher      29
at_home      20
health       18
Name: Fjob, dtype: int64
```

Out[22]:
```
other       258
services    136
at_home     135
teacher      72
health       48
Name: Mjob, dtype: int64
```

Out[22]:
```
other       367
services    181
at_home      42
teacher      36
health       23
Name: Fjob, dtype: int64
```

In [8]:
```
## Missing data check.
math_missing_data = math_data.isnull().sum()
print(math_missing_data)
## No missing data in math_data.

print()

por_missing_data = por_data.isnull().sum()
print(por_missing_data)
## No missing data in por_data
```

```
school        0
sex           0
age           0
address       0
famsize       0
Pstatus       0
Medu          0
Fedu          0
Mjob          0
Fjob          0
reason        0
guardian      0
traveltime    0
studytime     0
failures      0
schoolsup     0
famsup        0
paid          0
activities    0
nursery       0
higher        0
internet      0
romantic      0
famrel        0
freetime      0
goout         0
Dalc          0
Walc          0
health        0
m_absences    0
M1            0
M2            0
M3            0
dtype: int64
```

```
school        0
sex           0
age           0
address       0
famsize       0
Pstatus       0
Medu          0
Fedu          0
Mjob          0
Fjob          0
reason        0
guardian      0
traveltime    0
studytime     0
failures      0
schoolsup     0
famsup        0
paid          0
activities    0
nursery       0
higher        0
internet      0
```

```
romantic      0
famrel        0
freetime      0
goout         0
Dalc          0
Walc          0
health        0
p_absences    0
P1            0
P2            0
P3            0
dtype: int64
```

In [9]:
```
## Label Encode Sex. 0 = female, 1 = male
le_math = preprocessing.LabelEncoder()
le_math.fit(math_data['sex'])
math_sex_array = le_math.transform(math_data['sex'])
math_data['sex'] = math_sex_array
math_data.head()

le_por = preprocessing.LabelEncoder()
le_por.fit(por_data['sex'])
por_sex_array = le_por.transform(por_data['sex'])
por_data['sex'] = por_sex_array
por_data.head()
```

Out[9]:
```
LabelEncoder()
```

Out[9]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	free
0	GP	0	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	0	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	0	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	0	15	U	GT3	T	4	2	health	services	...	3	
4	GP	0	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

Out[9]:
```
LabelEncoder()
```

Out[9]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	free
0	GP	0	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	0	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	0	15	U	LE3	T	1	1	at_home	other	...	4	
3	GP	0	15	U	GT3	T	4	2	health	services	...	3	
4	GP	0	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

In [10]:
```
## Label Encode Parental Status. 0 = apart 1 = together
le_math_Pstatus = preprocessing.LabelEncoder()
le_math_Pstatus.fit(math_data['Pstatus'])
math_Pstatus_array = le_math_Pstatus.transform(math_data['Pstatus'])
math_data['Pstatus'] = math_Pstatus_array
math_data.head()

le_por_Pstatus = preprocessing.LabelEncoder()
le_por_Pstatus.fit(por_data['Pstatus'])
por_Pstatus_array = le_por_Pstatus.transform(por_data['Pstatus'])
por_data['Pstatus'] = por_Pstatus_array
por_data.head()
```

Out[10]: LabelEncoder()

Out[10]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	0	18	U	GT3	0	4	4	at_home	teacher	...	4	
1	GP	0	17	U	GT3	1	1	1	at_home	other	...	5	
2	GP	0	15	U	LE3	1	1	1	at_home	other	...	4	
3	GP	0	15	U	GT3	1	4	2	health	services	...	3	
4	GP	0	16	U	GT3	1	3	3	other	other	...	4	

5 rows × 33 columns

Out[10]: LabelEncoder()

Out[10]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	0	18	U	GT3	0	4	4	at_home	teacher	...	4	
1	GP	0	17	U	GT3	1	1	1	at_home	other	...	5	
2	GP	0	15	U	LE3	1	1	1	at_home	other	...	4	
3	GP	0	15	U	GT3	1	4	2	health	services	...	3	
4	GP	0	16	U	GT3	1	3	3	other	other	...	4	

5 rows × 33 columns

In [11]:
```
## Label Encode Address (Urban or Rural). 1 = Urban 0 = Rural
le_math_address = preprocessing.LabelEncoder()
le_math_address.fit(math_data['address'])
math_address_array = le_math_address.transform(math_data['address'])
math_data['address'] = math_address_array
math_data.head()

le_por_address = preprocessing.LabelEncoder()
le_por_address.fit(por_data['address'])
por_address_array = le_por_address.transform(por_data['address'])
por_data['address'] = por_address_array
por_data.head()
```

Out[11]: LabelEncoder()

Out[11]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	0	18	1	GT3	0	4	4	at_home	teacher	...	4	
1	GP	0	17	1	GT3	1	1	1	at_home	other	...	5	
2	GP	0	15	1	LE3	1	1	1	at_home	other	...	4	
3	GP	0	15	1	GT3	1	4	2	health	services	...	3	
4	GP	0	16	1	GT3	1	3	3	other	other	...	4	

5 rows × 33 columns

Out[11]: LabelEncoder()

Out[11]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	0	18	1	GT3	0	4	4	at_home	teacher	...	4	
1	GP	0	17	1	GT3	1	1	1	at_home	other	...	5	
2	GP	0	15	1	LE3	1	1	1	at_home	other	...	4	
3	GP	0	15	1	GT3	1	4	2	health	services	...	3	
4	GP	0	16	1	GT3	1	3	3	other	other	...	4	

5 rows × 33 columns

In [12]:
```
## Label Encode schoolsup (extra educational support) 1 = yes, 0 = no.
le_math_schoolsup = preprocessing.LabelEncoder()
le_math_schoolsup.fit(math_data['schoolsup'])
math_schoolsup_array = le_math_schoolsup.transform(math_data['schoolsup'])
math_data['schoolsup'] = math_schoolsup_array
math_data['schoolsup'].head()

le_por_schoolsup = preprocessing.LabelEncoder()
le_por_schoolsup.fit(por_data['schoolsup'])
por_schoolsup_array = le_por_schoolsup.transform(por_data['schoolsup'])
por_data['schoolsup'] = por_schoolsup_array
por_data['schoolsup'].head()
```

Out[12]:
```
LabelEncoder()
```

Out[12]:
```
0    1
1    0
2    1
3    0
4    0
Name: schoolsup, dtype: int64
```

Out[12]:
```
LabelEncoder()
```

Out[12]:
```
0    1
1    0
2    1
3    0
4    0
Name: schoolsup, dtype: int64
```

In [13]:
```
## Label Encode famsup (Family Support) 1 = yes, 0 = no
le_math_famsup = preprocessing.LabelEncoder()
le_math_famsup.fit(math_data['famsup'])
math_famsup_array = le_math_famsup.transform(math_data['famsup'])
math_data['famsup'] = math_famsup_array
math_data['famsup'].head()

le_por_famsup = preprocessing.LabelEncoder()
le_por_famsup.fit(por_data['famsup'])
por_famsup_array = le_por_famsup.transform(por_data['famsup'])
por_data['famsup'] = por_famsup_array
por_data['famsup'].head()
```

Out[13]:
```
LabelEncoder()
```

Out[13]:
```
0    0
1    1
2    0
3    1
4    1
Name: famsup, dtype: int64
```

Out[13]:
```
LabelEncoder()
```

Out[13]:
```
0    0
1    1
2    0
3    1
4    1
Name: famsup, dtype: int64
```

In [14]:
```
## Label Encode paid (extra paid classes within the course subject (Math or Po
rtuguese)) 1 = yes, 0 = no
le_math_paid = preprocessing.LabelEncoder()
le_math_paid.fit(math_data['paid'])
math_paid_array = le_math_paid.transform(math_data['paid'])
math_data['paid'] = math_paid_array
math_data['paid'].head()

le_por_paid= preprocessing.LabelEncoder()
le_por_paid.fit(por_data['paid'])
por_paid_array = le_por_paid.transform(por_data['paid'])
por_data['paid'] = por_paid_array
por_data['paid'].head()
```

Out[14]:
```
LabelEncoder()
```

Out[14]:
```
0    0
1    0
2    1
3    1
4    1
Name: paid, dtype: int64
```

Out[14]:
```
LabelEncoder()
```

Out[14]:
```
0    0
1    0
2    0
3    0
4    0
Name: paid, dtype: int64
```

In [15]:
```
## Label Encode activities (extra-curricular activities) 1 = yes, 0 = no
le_math_activities = preprocessing.LabelEncoder()
le_math_activities.fit(math_data['activities'])
math_activities_array = le_math_activities.transform(math_data['activities'])
math_data['activities'] = math_activities_array
math_data['activities'].head()

le_por_activities= preprocessing.LabelEncoder()
le_por_activities.fit(por_data['activities'])
por_activities_array = le_por_activities.transform(por_data['activities'])
por_data['activities'] = por_activities_array
por_data['activities'].head()
```

Out[15]:
```
LabelEncoder()
```

Out[15]:
```
0    0
1    0
2    0
3    1
4    0
Name: activities, dtype: int64
```

Out[15]:
```
LabelEncoder()
```

Out[15]:
```
0    0
1    0
2    0
3    1
4    0
Name: activities, dtype: int64
```

In [16]:
```
## Label Encode nursery (attended nursery school) 1 = yes, 0 = no
le_math_nursery = preprocessing.LabelEncoder()
le_math_nursery.fit(math_data['nursery'])
math_nursery_array = le_math_nursery.transform(math_data['nursery'])
math_data['nursery'] = math_nursery_array
math_data['nursery'].head()

le_por_nursery= preprocessing.LabelEncoder()
le_por_nursery.fit(por_data['nursery'])
por_nursery_array = le_por_nursery.transform(por_data['nursery'])
por_data['nursery'] = por_nursery_array
por_data['nursery'].head()
```

Out[16]:
```
LabelEncoder()
```

Out[16]:
```
0    1
1    0
2    1
3    1
4    1
Name: nursery, dtype: int64
```

Out[16]:
```
LabelEncoder()
```

Out[16]:
```
0    1
1    0
2    1
3    1
4    1
Name: nursery, dtype: int64
```

In [17]:
```
## Label Encode higher (wants to take higher education) 1 = yes, 0 = no
le_math_higher = preprocessing.LabelEncoder()
le_math_higher.fit(math_data['higher'])
math_higher_array = le_math_higher.transform(math_data['higher'])
math_data['higher'] = math_higher_array
math_data['higher'].head()

le_por_higher= preprocessing.LabelEncoder()
le_por_higher.fit(por_data['higher'])
por_higher_array = le_por_higher.transform(por_data['higher'])
por_data['higher'] = por_higher_array
por_data['higher'].head()
```

Out[17]:
```
LabelEncoder()
```

Out[17]:
```
0    1
1    1
2    1
3    1
4    1
Name: higher, dtype: int64
```

Out[17]:
```
LabelEncoder()
```

Out[17]:
```
0    1
1    1
2    1
3    1
4    1
Name: higher, dtype: int64
```

In [18]:
```
## Label Encode internet (Internet access at home) 1 = yes, 2 = no
le_math_internet = preprocessing.LabelEncoder()
le_math_internet.fit(math_data['internet'])
math_internet_array = le_math_internet.transform(math_data['internet'])
math_data['internet'] = math_internet_array
math_data['internet'].head()

le_por_internet= preprocessing.LabelEncoder()
le_por_internet.fit(por_data['internet'])
por_internet_array = le_por_internet.transform(por_data['internet'])
por_data['internet'] = por_internet_array
por_data['internet'].head()
```

Out[18]:
```
LabelEncoder()
```

Out[18]:
```
0    0
1    1
2    1
3    1
4    0
Name: internet, dtype: int64
```

Out[18]:
```
LabelEncoder()
```

Out[18]:
```
0    0
1    1
2    1
3    1
4    0
Name: internet, dtype: int64
```

In [19]:
```
## Label Encode romantic (in a romantic relationship) 1 = yes, 2 = no
le_math_romantic = preprocessing.LabelEncoder()
le_math_romantic.fit(math_data['romantic'])
math_romantic_array = le_math_romantic.transform(math_data['romantic'])
math_data['romantic'] = math_romantic_array
math_data['romantic'].head()

le_por_romantic= preprocessing.LabelEncoder()
le_por_romantic.fit(por_data['romantic'])
por_romantic_array = le_por_romantic.transform(por_data['romantic'])
por_data['romantic'] = por_romantic_array
por_data['romantic'].head()
```

Out[19]:
```
LabelEncoder()
```

Out[19]:
```
0    0
1    0
2    0
3    1
4    0
Name: romantic, dtype: int64
```

Out[19]:
```
LabelEncoder()
```

Out[19]:
```
0    0
1    0
2    0
3    1
4    0
Name: romantic, dtype: int64
```

In [21]:
```
## Saving the new encoded datset as a CSV
export_math = math_data.to_csv('encoded_math_data.csv')
export_por = por_data.to_csv('encoded_por_data.csv')
```

In [ ]:
```

```