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

```
data = pd.read_csv("age_of_marriage_data.csv")
data.head()
```

	id	gender	height	religion	caste	mother_tongue	professi
0	1	female	5'4"	NaN	others	Telugu	N
1	2	male	5'7"	Jain	Shwetamber	Gujarati	Doct Healthc Professio
2	3	male	5'7"	Hindu	Brahmin	Hindi	Entreprene / Busin
3	4	female	5'0"	Hindu	Thakur	Hindi	Archit
4	5	male	5'5"	Christian	Born Again	Malayalam	Sa Profession Market

```
data.isnull().sum()
```

```
id                  0
gender             29
height            118
religion          635
caste             142
mother_tongue     164
profession        330
location          155
country            16
age_of_marriage    19
dtype: int64
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2567 entries, 0 to 2566
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   id               2567 non-null   int64  
 1   gender           2538 non-null   object 
 2   height           2449 non-null   object 
 3   religion         1932 non-null   object 
 4   caste            2425 non-null   object 
 5   mother_tongue    2403 non-null   object 
 6   profession       2237 non-null   object 
 7   location         2412 non-null   object 
 8   country          2551 non-null   object 
 9   age_of_marriage  2548 non-null   float64
dtypes: float64(1), int64(1), object(8)
```

```
(data.shape[0] - data.dropna().shape[0])/data.shape[0]
```

```
0.24737047136735488
```

```
data.dropna(inplace=True)
```

```
data.shape
```

```
(1932, 10)
```

```
data.head(10)
```

	id	gender	height	religion	caste	mother_tongue	profess
1	2	male	5'7"	Jain	Shwetamber	Gujarati	Doc Health Professi
2	3	male	5'7"	Hindu	Brahmin	Hindi	Entrepren / Busi
3	4	female	5'0"	Hindu	Thakur	Hindi	Arch
4	5	male	5'5"	Christian	Born Again	Malayalam	S Professio Marke
5	6	male	5'5"	Hindu	Valmiki	Hindi	Sports
6	7	female	5'2"	Hindu	Rajput - Lodhi	Hindi	Ban Professi

```
from sklearn.preprocessing import LabelEncoder
```

/ Busi

```
real_x = data.loc[:,['gender','height','religion','caste','mother_tongue','
real_y = data.age_of_marriage
```

```
encoder = LabelEncoder()
real_x.loc[: , ['gender', 'religion', 'caste',  'mother_tongue' ,'country']]
real_x.loc[: , ['gender',   'religion', 'caste',  'mother_tongue' ,'country
```

```
real_x.head()
```

	gender	height	religion	caste	mother_tongue	country
1	1	5'7"	2	34	6	19
2	1	5'7"	1	14	8	5
3	0	5'0"	1	36	8	5
4	1	5'5"	0	13	13	5
5	1	5'5"	1	38	8	5

```
int(real_x.loc[1,'height'].split('\'')[0])*30.48
```

```
152.4
```

```
int(real_x.loc[1,'height'].split('\'')[1].replace('"',''))*2.54
```

```
17.78
```

```
def h_cms(h):
    return int(h.split('\'')[0])*30.48+\
    int(h.split('\'')[1].replace('"',''))*2.54
```

```
real_x['height_cms'] = real_x.height.apply(h_cms)
```

```
real_x.head()
```

```
   gender  height  religion  caste  mother_tongue  country  height_cm
1       1    5'7"         2     34              6       19      170.1
2       1    5'7"         1     14              8        5      170.1
3       0    5'0"         1     36              8        5      152.4
4       1    5'5"         0     13             13        5      165.1
5       1    5'5"         1     38              8        5      165.1
```

```
real_x.drop('height',inplace=True,axis=1)
```

```
real_x.head()
```

	gender	religion	caste	mother_tongue	country	height_cms
1	1	2	34	6	19	170.18
3	0	1	36	8	5	152.40

```
rom sklearn.model_selection import train_test_split
_train,X_test,y_train,y_test = train_test_split(real_x,real_y,test_size=0.2
```

```
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor(n_estimators=80,max_depth=11)
model.fit(X_train,y_train)
y_predict = model.predict(X_test)
```

```
y_predict
```

```
array([28.64298148, 33.82640557, 29.65686506, 28.57483218, 30.17390703
       27.20927164, 33.36355114, 27.15474039, 29.34457533, 31.95865217
       31.18938527, 31.39499836, 31.31563331, 31.31563331, 28.96133658
       31.40536052, 32.36497424, 28.63638393, 29.8530451 , 25.95684028
       28.40562018, 34.35435075, 31.73243386, 28.73666868, 30.77792601
       30.58139559, 31.23672916, 31.54782723, 28.91652505, 31.68956206
       33.73682234, 31.31563331, 30.82956964, 29.60693766, 27.13296542
       28.86963151, 30.02406642, 33.79127576, 27.12126488, 26.97934524
       28.79798309, 28.83028741, 29.61853532, 29.41295421, 28.43936693
       28.67969542, 27.82341317, 27.14427083, 34.63985936, 33.21049624
       28.89409145, 27.28605242, 29.46936462, 31.2574392 , 31.31191609
       30.00711938, 30.11458748, 29.9780508 , 29.09399583, 30.80901975
       28.61616028, 29.52370771, 34.21336411, 28.29548389, 28.34581288
       32.00967045, 28.36024087, 28.6483896 , 31.79555396, 29.56037156
       28.46532085, 28.44787547, 31.93393258, 27.62380763, 28.7927714 
       30.83235916, 29.58280019, 31.30387617, 31.35636587, 28.19123425
       29.86162362, 28.54473463, 29.73003551, 28.81741926, 28.42583661
       27.05755823, 30.03600973, 29.64529505, 28.36028255, 30.90510959
       33.07824316, 29.79277274, 29.42437721, 32.7681038 , 30.39958396
       25.10416667, 31.17351869, 28.62660939, 28.00042194, 31.42698999
       31.18139253, 28.14879008, 31.04785198, 29.15615651, 29.69978254
       30.54001004, 30.80663532, 31.6296733 , 28.99451861, 30.67835428
       31.44557659, 28.44787547, 31.38825867, 28.85      , 28.22014277
       29.78055812, 29.33156732, 32.76232091, 32.58287829, 29.46362111
       28.37025851, 30.4692513 , 31.7768755 , 33.16424612, 26.88897772
       31.30387617, 26.2241369 , 27.94284102, 31.28790392, 29.01360211
       26.81383433, 28.6195947 , 31.80814577, 26.66553571, 27.33949275
       28.51543809, 28.76053632, 31.23000349, 27.64366611, 31.32879282
       27.73463158, 28.81365277, 30.38413525, 28.61290641, 27.4259383 
       31.45421341, 31.13853991, 29.59430705, 26.26621528, 30.09887904
       29.52678338, 29.31590909, 31.36199112, 33.38771024, 31.5644105 
       29.42629869, 27.15016761, 28.40383064, 31.20106499, 26.25732143
       30.15224036, 30.48249456, 28.70776906, 27.51885508, 28.8671754 
       26.63375   , 33.2197621 , 28.51543809, 32.91892829, 34.15354727
       27.16813854, 30.78095068, 28.76269753, 30.38853819, 31.01545068
       30.21352634, 31.13853991, 29.56977305, 29.26125681, 29.82778095
       29.09399583, 28.55633541, 31.12952236, 29.22878562, 35.02952381
       28.61608587, 31.36169661, 30.39958396, 28.52951389, 27.20927164
       33.90348214, 28.79820916, 30.2397825 , 29.61749824, 31.79755012
       29.74020833, 28.14007047, 31.5644105 , 29.32709715, 32.5823169 
       32.25009316, 31.19592735, 30.82688935, 29.22457817, 27.87529651
       30.13594456, 29.58280019, 27.19427083, 27.82771724, 29.58289675
       28.80717463, 31.28535901, 30.84153352, 27.19427083, 35.29541667
       29.58280019, 31.57360104, 32.66179383, 29.07800374, 33.74692756
       30.7391969 , 34.61506543, 29.55445835, 30.65569143, 29.55522044
       29.09438395, 30.58070989, 30.22370997, 29.01800593, 30.25522246
       29.6638501 , 30.53769415, 32.77690024, 32.91892829, 28.6191109 
       29.6658901 , 30.55769425, 31.77696624, 31.91891819, 28.6191169
```

```
y_test
```

```
1277    29.0
1502    33.0
645     30.0
1489    28.0
2239    30.0
        ...
351     28.0
274     32.0
510     29.0
601     31.0
1786    32.0
Name: age_of_marriage, Length: 387, dtype: float64
       27.92708333, 27.51885508, 30.11458748, 28.83028741, 31.9019721
```

```
from sklearn.metrics import mean_absolute_error, r2_score
print("MAE : ", mean_absolute_error(y_test,y_predict))
r2_score(y_test,y_predict)
```

```
MAE :  1.0391761467360092
0.6962824751634216
       30.15801125, 28.1440558 , 32.66715592, 29.52765715, 29.5697750
```

```
from sklearn.externals import joblib
joblib.dump(model,'marriage_age_predict_model.pkl')
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/externals/joblib/__init
  warnings.warn(msg, category=FutureWarning)
['marriage_age_predict_model.pkl']
```