(https://www.bigdatauniversity.com)

Project: Whether a loan is paid off

Deadline: 2020-08-16 18:00:00

Total marks: 6.0

Your information:

- Fullname:
- Date of birth:
- Place of birth:
- Email:
- Mobile phone:

In this notebook, we practice all the knowledge and skills that we learned in this course.

We apply the **Regression Algorithm** to predict: "Whether a loan is paid off on in collection" by accuracy evaluation methods.

Lets first load required libraries:

In [1]:
```
import itertools
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import NullFormatter
import pandas as pd
import numpy as np
import matplotlib.ticker as ticker
from sklearn import preprocessing
%matplotlib inline
```

About dataset

This dataset is about past loans. The **Loan_train.csv** data set includes details of 346 customers whose loan are already paid off or defaulted. It includes following fields:

Field	Description
Loan_status	Whether a loan is paid off on in collection
Principal	Basic principal loan amount at the
Terms	Origination terms which can be weekly (7 days), biweekly, and monthly payoff schedule
Effective_date	When the loan got originated and took effects
Due_date	Since it's one-time payoff schedule, each loan has one single due date
Age	Age of applicant
Education	Education of applicant
Gender	The gender of applicant

Data exploration

*** To predict "Whether a loan is paid off", we need some fields: 'Principal', 'Terms', 'Age', 'Gender', 'Effective_date'

The first things we need to do:

- Identify Variables
- Univariate Analysis
- Bi-variate Analysis
- Handle the Missing Values
- Handle Outlier Values

Tips: Step by step like Chapter2_Ex1_Housing prices

Load Data From CSV File

In [2]:
```
# Read CSV file: loan_train.csv
# code here
```

In [3]:
```
# Understanding to dataset
# shape
# info
# head(), tail()
# describe()
```

Convert 'due_date', 'effective_date' to date time object

In [4]:
```
# code here
```

Data visualization

How many sample of each class is in our data set?

In [5]:
```
# code
```

xxx people have paid off the loan on time while **xxx** have gone into collection

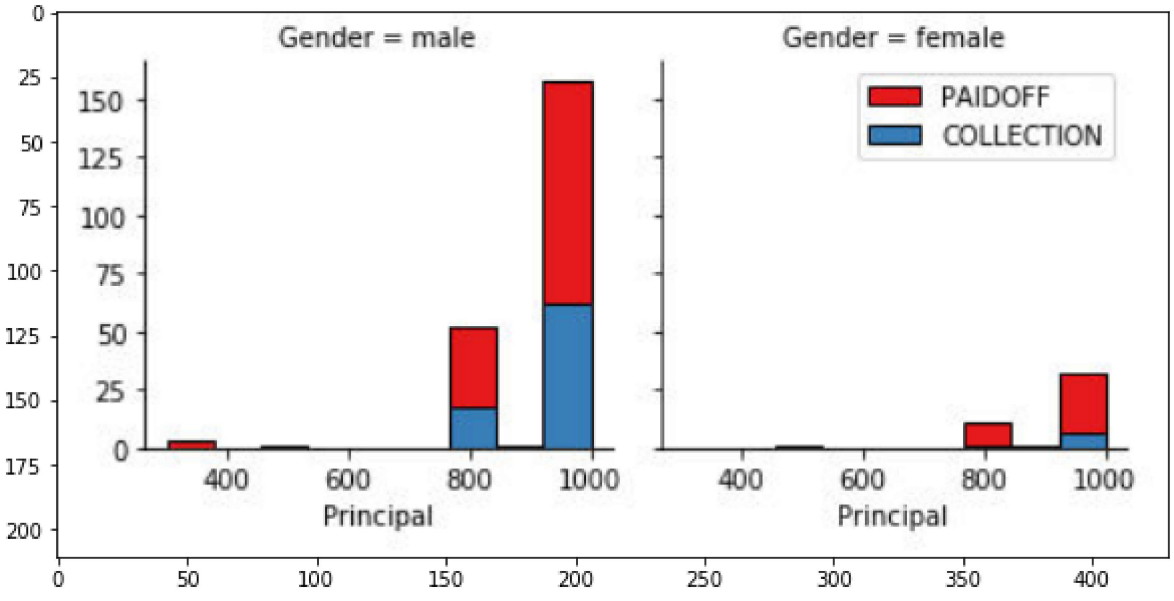
Lets plot some columns to underestand data better:

- Use seaborn or matplotlib to draw some plots like that:

In [6]:
```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
```

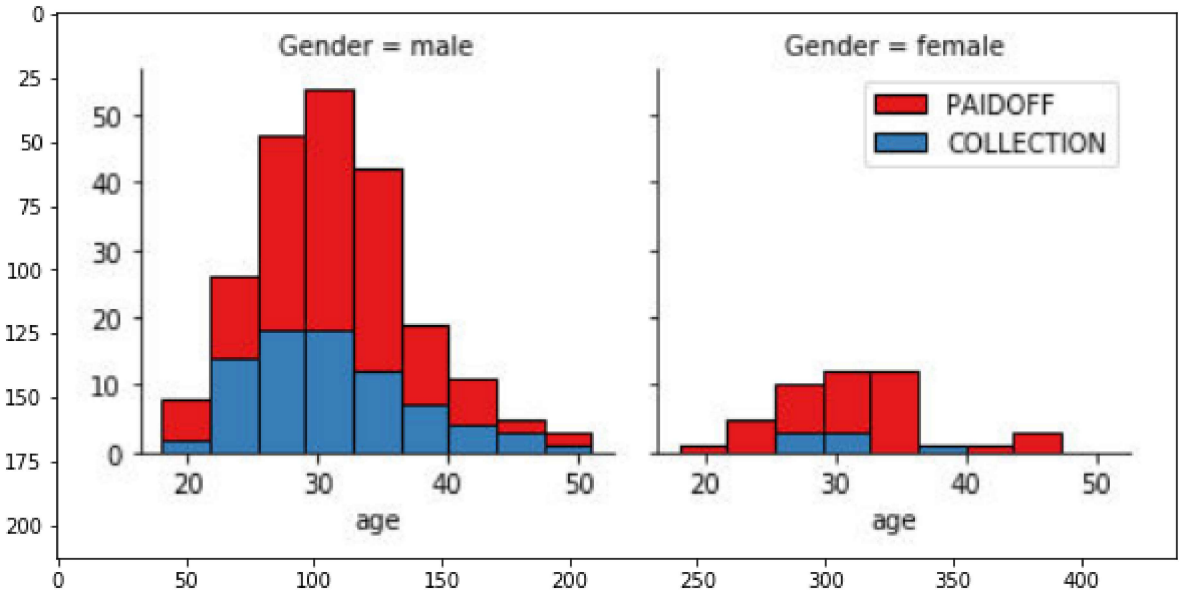
In [7]:
```
img1 = np.array(Image.open('Principal_Male_Female.jpg'))
```

In [8]:
```
plt.figure(figsize=(10,5))
plt.imshow(img1, interpolation='bilinear')
plt.show()
```



In [9]:
```
img2 = np.array(Image.open('Age_Male_Female.jpg'))
```

In [10]:
```
plt.figure(figsize=(10,5))
plt.imshow(img2, interpolation='bilinear')
plt.show()
```



In [11]:
```
# code here
```

Pre-processing: Feature selection/extraction

Lets look at the day of the week people get the loan

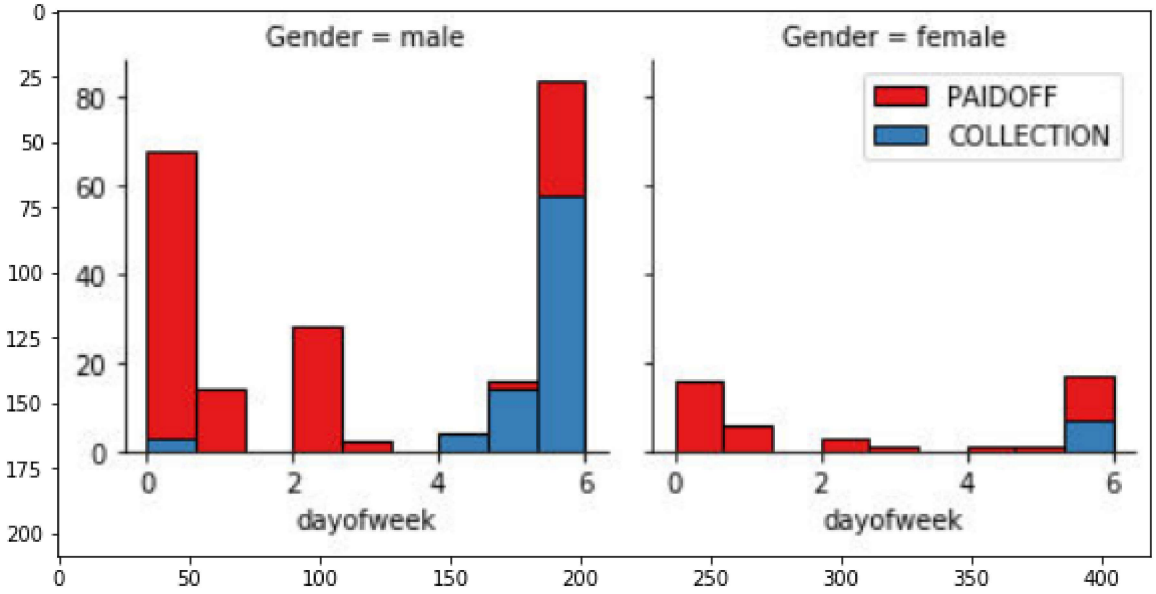
- Make new column 'dayofweek' from 'effective_date'
  - Example: 2016-09-08 => dayofweek is 3 (The day of the week with Monday=0, Sunday=6)
  - Link: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DatetimeIndex.dayofweek.html (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DatetimeIndex.dayofweek.html)

In [12]:
```
# code here
```

Lets plot some columns to underestand data better:

In [13]:
```
img3 = np.array(Image.open('day_of_week.jpg'))
```

In [14]:
```
plt.figure(figsize=(10,5))
plt.imshow(img3, interpolation='bilinear')
plt.show()
```



In [15]:
```
# code here
```

We see that people who get the loan at the end of the week don't pay it off, so lets use Feature binarization to set a threshold values less then day 4

- Make new column 'weekend': =1 if 'dayofweek'>3, else =0

In [16]:
```
# code here
```

Convert Categorical features to numerical values

- groupby 'Gender' and count by 'loan_status'

In [17]:
```
# code here
```

xxx % of female pay there loans while only **xxx** % of males pay there loan

Lets convert male to 0 and female to 1:

In [18]:
```
# code here
```

One Hot Encoding

How about education?

- groupby 'education' and count by 'loan_status'

```
In [19]: # code here
```

Feature befor One Hot Encoding

- Print head() data with 5 columns: 'Principal','terms','age','Gender','education'

```
In [20]: # code here
```

Use one hot encoding technique to convert categorical varables to binary variables and append them to the feature Data Frame

- Make new dataframe **Feature** has: 'Principal','terms','age','Gender','weekend', 'education'
- In **Feature**: Use one hot encoding technique to convert 'education' to binary variable, then drop column 'Master or Above'

```
In [21]: # code here
```

Feature selection

Lets defind feature sets, X:

- X is input, X = Feature

```
In [22]: # code here
```

What are our lables?

- y is output, y = 'loan_status' column

```
In [23]: # code here
```

Normalize Data

Data Standardization give data zero mean and unit variance (technically should be done after train test split)

- Find the suitable Scaler to scale data of X (if we need to do to have a better prediction)

In [24]:
```
# code here
```

In [ ]:
```

```