**Lab 1: Missing Data Imputation**

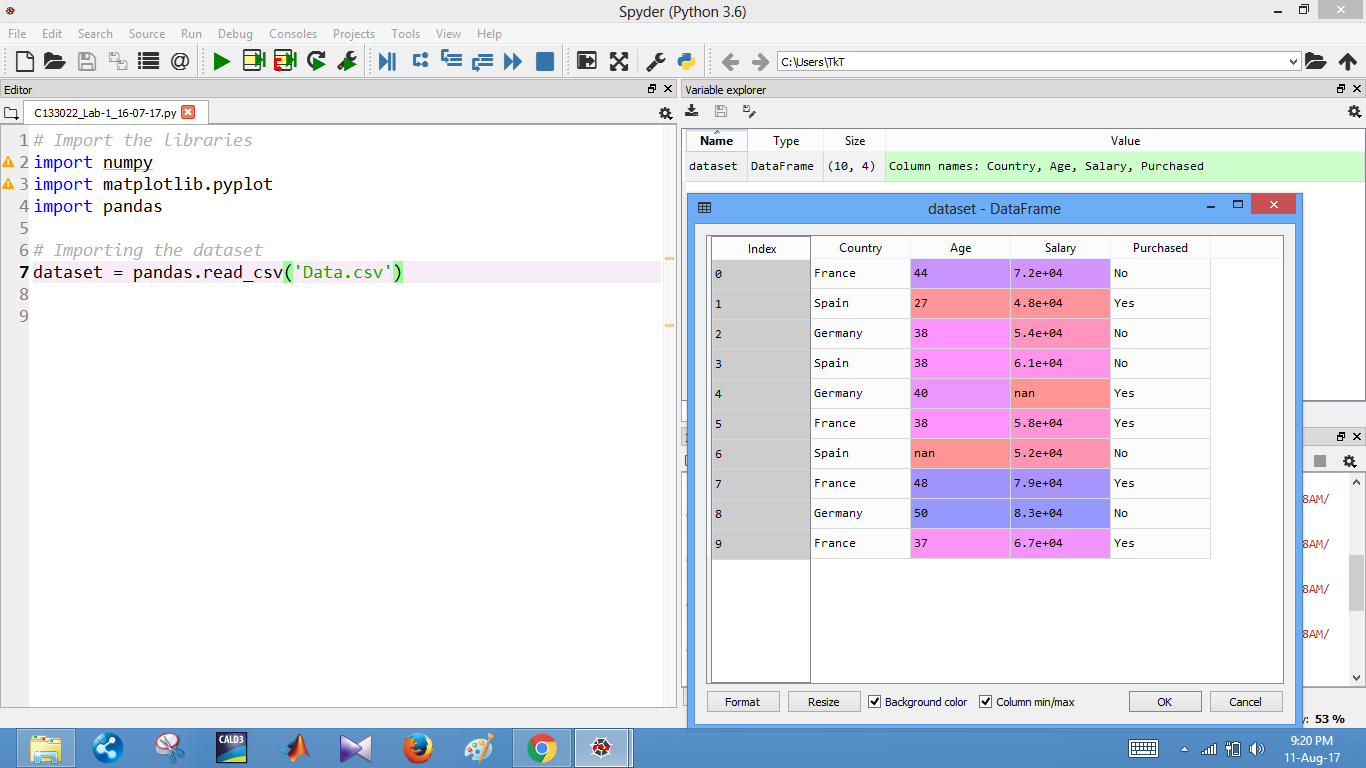
**Objectives:** To impute missing data using different stategies.

**Description:**

* At first **Anaconda 4.4.0** is downloaded from <https://www.continuum.io/downloads> and is installed.
* After that an application named  **spyder** is openned which is included in the anaconda.
* A sample data file is made from Microsoft Excel and is saved as **Data.csv** file.
* Then we took the **data.csv** file as input in a variable named **dataset**.
* For this ***pandas.read\_csv()*** is used that read **CSV** file into DataFrame (Fig.01-Left).

[CSV is stand for Comma Separated Values.]

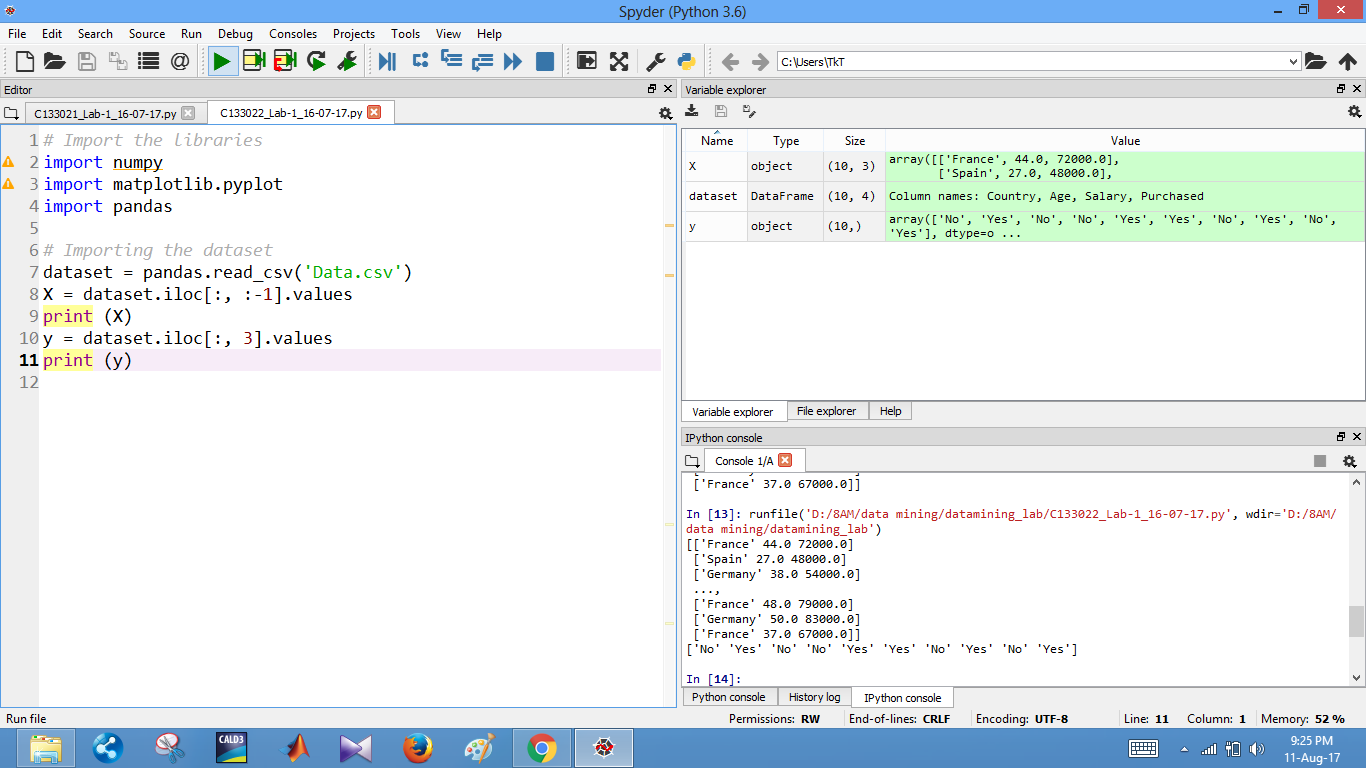
* Double-Clicking on the button ‘**dataset**’ shows the **‘data.csv’** file. (Fig.01- Right).



(Left) (Right)

Fig.01

* First three columns are assigned in variable **X**.
* And the last column is assigned in variable **Y**. (Fig.02-Left)



(Left) (Right)

Fig.02

* **sklearn.preprocessing** module includes scaling, centering, normalization, binarization and imputation methods.
* **Imputer** class is imported from **sklearn.preprocessing** package for completing missing values.
* To see the definition of the class we can right click on the class name and then select **go to definition**(Fig-03).

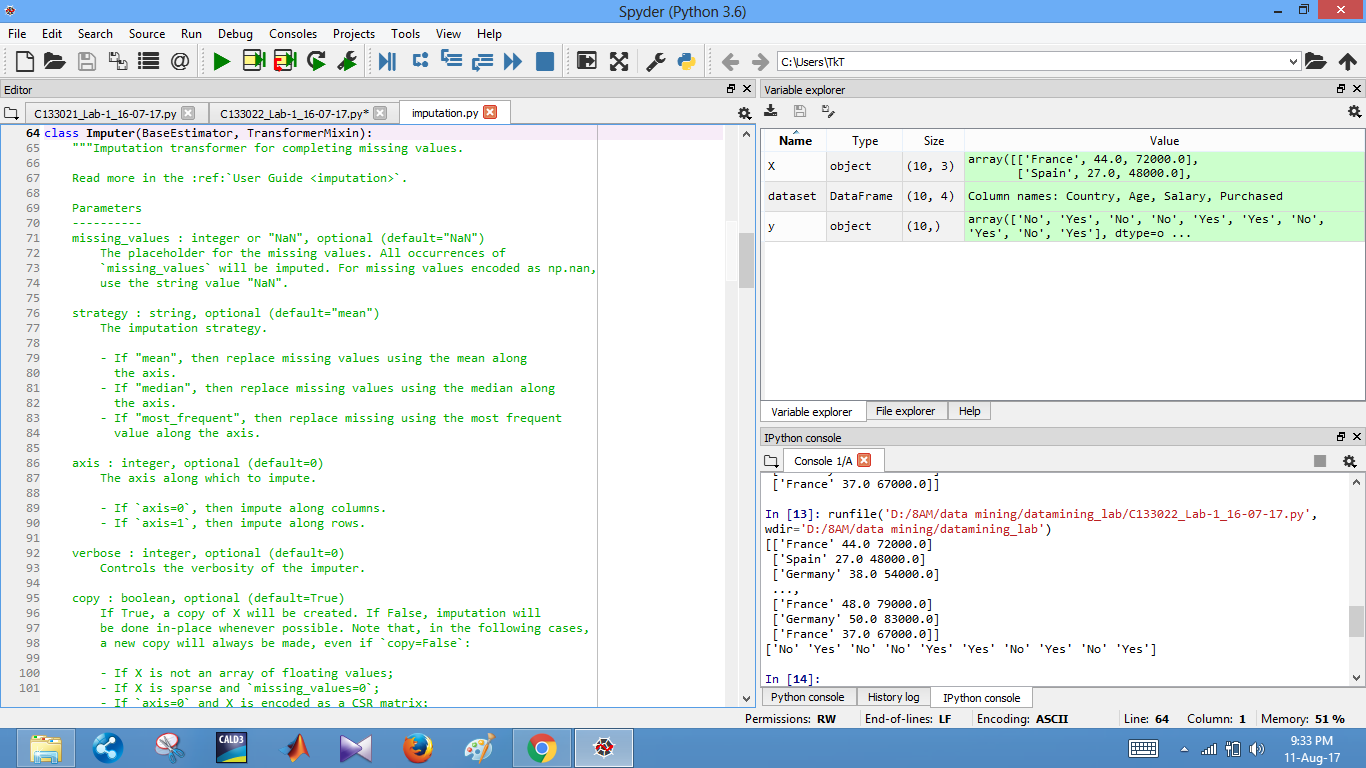
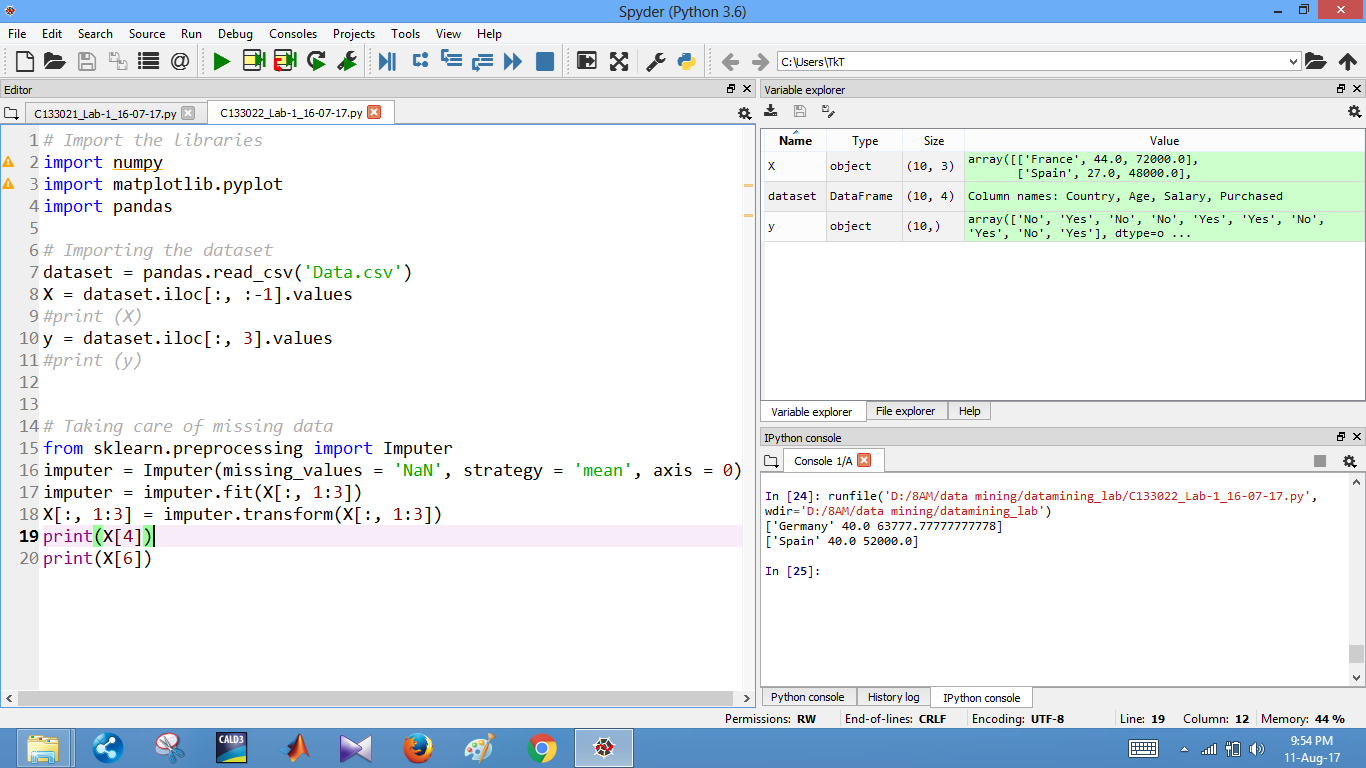


Fig.03

* Missing values in the sample data are imputed (Fig.04-Right) and are seen in the console.
* **fit(X[,y])** fit the imputer on **X**. (Fig.04-Left)
* **transform(X)** Impute all missing values in **X**. (Fig.04-Left)
* 

(Left) (Right)

Fig.04

**Results/Conclusions:**

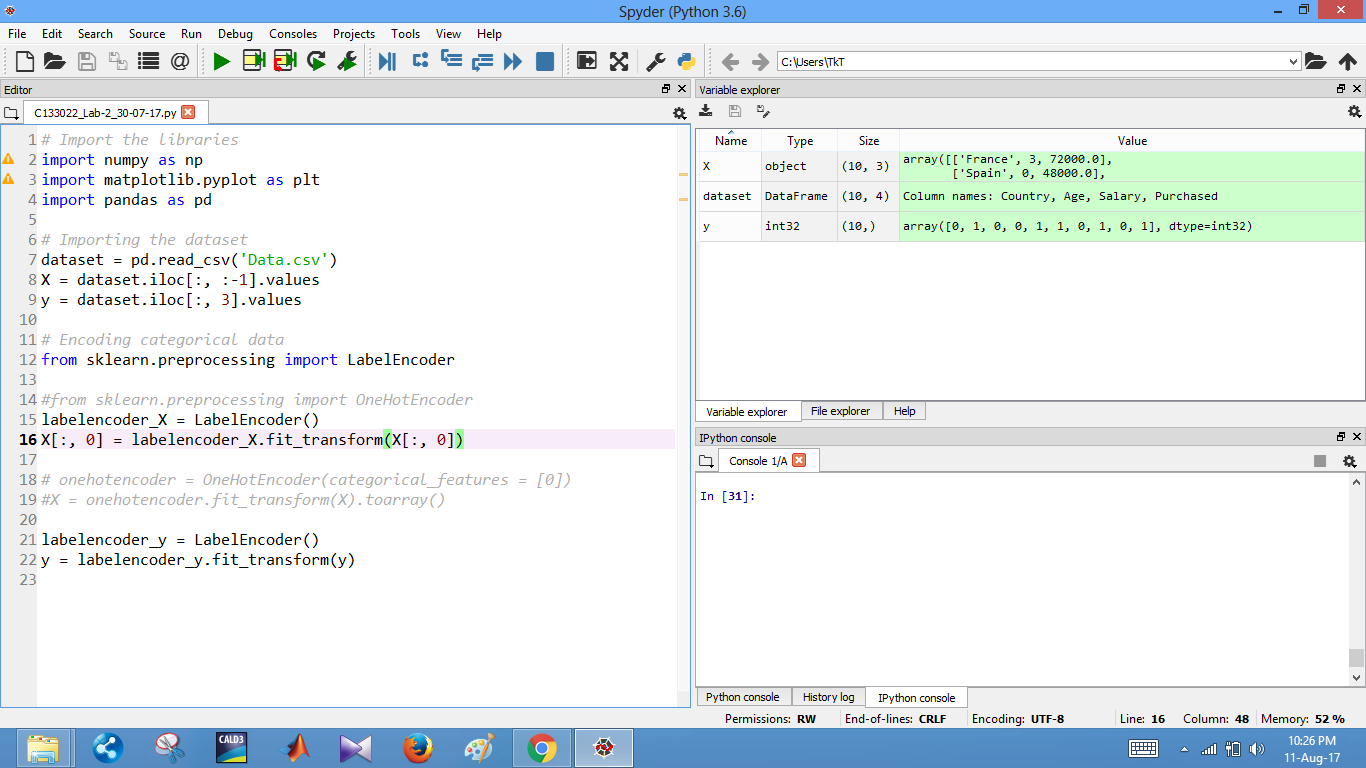
* The missing values were successfully imputed.

**Lab 2; Encoding Categorial Data**

**Objectives:** To encode the categorial data using LabelEncoder class.

**Description:**

* At first sample **Data.csv** file is taken as input .
* **LabelEncoder** class is imported from **sklearn.preprocessing**. (Fig.05-Right)
* **LabelEncoder**  encode labels with value between 0 and n\_classes-1.
* [**fit\_transform**](http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html#sklearn.preprocessing.LabelEncoder.fit_transform)(y) fit label encoder and return encoded labels (Fig.05-Right)
* We can see country names are labeled using **0,1,2**,… in fig.05 on left side in **y** array.



(Left) (Right)

Fig.05

**Results/Conclusions:**

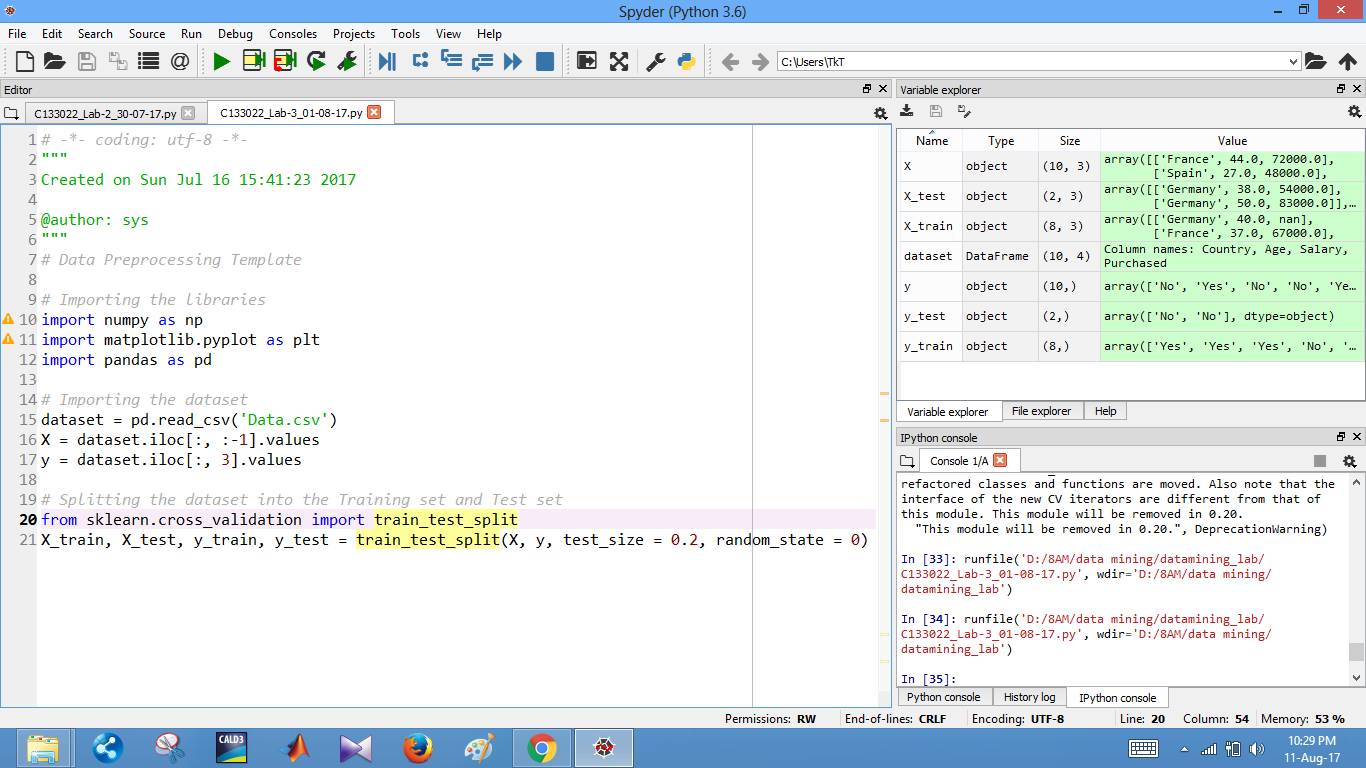
* The **country names** were labeled successfully using **LabelEncoder**.
* But there was a problem faced when we try to use **OneHotEncoder class**.

**Lab 3: Splitting the data set into Trainning set and Testing set**

**Objectives:** To Split the data set into Trainning set and Testing set using train\_test\_split class.

**Description:**

* At first sample **Data.csv** file is taken as input .
* **train\_test\_split** class is imported from **sklearn.preprocessing**. (Fig.06-Right)
* **train\_test\_split()** Split arrays or matrices into random train and test subsets.
* We can see input data is splitted into train set and test set in Fig.06 on left side.



(Left) (Right)

Fig.06

**Results/Conclusions:**

* The sample data were successfully splitted into train set and test set.