Data Preprocessing with Pandas

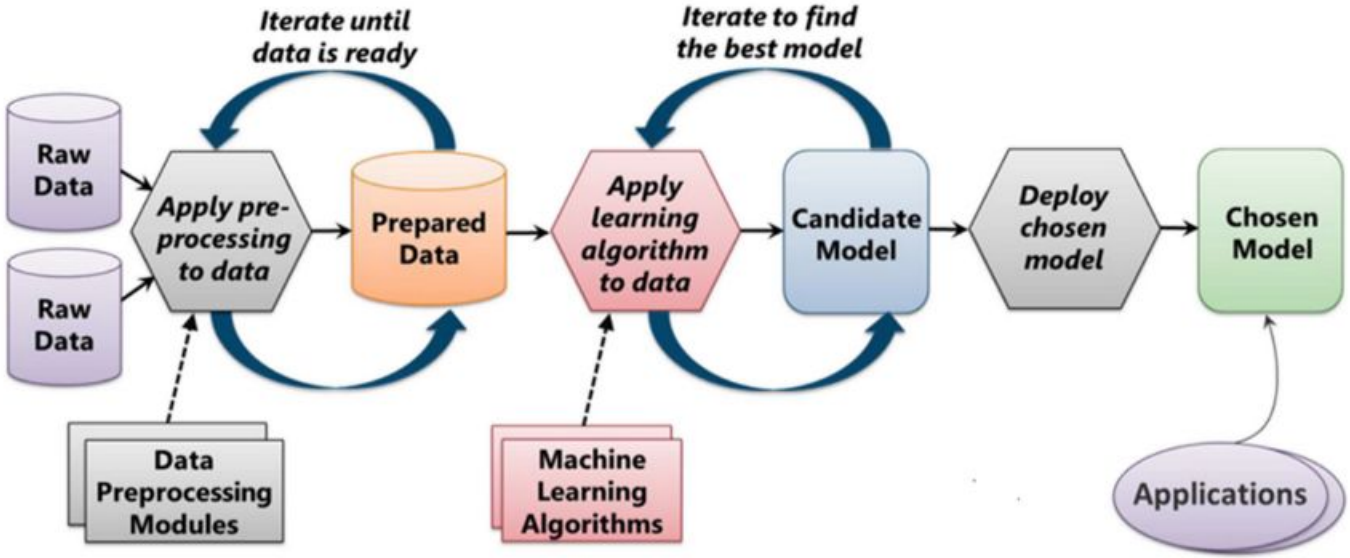
About Pandas

- High-level data manipulation tool developed by Wes McKinney.
- Built on the Numpy package
- Key data structure is called the DataFrame

DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables. #like a table

Series : A one-dimensional labeled array a capable of holding any data type

The Machine Learning Process



From "Introduction to Microsoft Azure" by David Chappell

Steps in Data Preprocessing

Step 1 : Import the libraries

Step 2 : Import the data-set

Step 3 : Check out the missing values

We'll be covering these steps today using pandas

Step 4 : See the Categorical Values

Step 5 : Splitting the data-set into Training and Test Set

Step 6 : Feature Scaling

1. Import the libraries

let's get the playground ready

#fire up the jupyter notebook and lets create a new notebook

#import pandas library import pandas as pd

2. Import the data-set

Before working on actual data, let's learn about Dataframes and Series by creating some.

```
pd.DataFrame({'Yes': [50, 21], 'No': [131, 2]})
```

```
pd.DataFrame({'Bob': ['I liked it.', 'It was awful.'],
              'Sue': ['Pretty good.', 'Bland.']},
             index=['Product A', 'Product B'])
```

```
pd.Series([1, 2, 3, 4, 5])
```

Can you create this dataset?

Don't forget the indexes!

	Student A	Student B	Student C
English	98	75	90
Maths	45	99	67

2.Importing Dataset

CSV file is a table of values separated by commas. Hence the name: "comma-seperated values", or CSV.

```
df=pd.read_csv("titanic.csv")        #open the csv file , it has more than 30 parameters
df.to_csv('myCSV.csv')               #write the dataframe to a csv file
df.shape                             #how large the dataset is (rows,cols)
df.head()                            #grabs first five rows
df.columns                           #displays the columns in the dataset
df['name']                           #displays column name
```

Select name, age, ticket columns ?

change the column names?

2.Importing Dataset

```
anotherDf=df[df['sex']=='female']

#what will this statement do?

Try it!


 #Selecting a subset of existing dataset based on a condition
```

2.Importing Dataset

Indexing and Slicing

Naive accessors

Index-based selection

```
df.iloc[0]          #accesses the first row

df.iloc[:,0]        #access the first column

df.iloc[1:3,]       #row 1 to 3

df.iloc[[0,3,4],0]  #selects 0th column of 0, 3 ,4th row : a list can be used to specify rows or columns

df.iloc[-5,0]       #counts toward the end; selects 0th column of 5th row from last
```

2.Importing Dataset

Label-based selection

```
df.loc[0,'name']                #selects column ‘name’ of 0th row

df.loc[0,[‘name’,’age’]]
```

Conditional selection

```
df.sex==’female’

df.loc[(df.sex==’female’) & (df.age>20)]

df.loc[df.body.notnull()]                    #selects values which are not empty
```

Iterate through Dataframe

```
count=0
for index,row in df.iterrows():
    if row['age']>20:
        count+=1
print(count)

#check out stack overflow for other ways of iterating a dataframe
```

3. Check out missing values

fillna() can “fill in” NA values with non-NA data

Rest Try to explore :

https://pandas.pydata.org/pandas-docs/stable/missing_data.html