1. Project Desciption:

The project have 3 instances Development , Training & testing and submission of data. This also involves installation and configuration of python , pycharm IDE, pip and pandas packages in windows environment. Here is break up of project components.

1.1 Data source files:

Select the project Data Science on Big Data Mart project and down load the 4 four files from resource folder in Dyzer.com web site. Place the files in the following folders in your local desktop.

#/users/Krishna/Python37-32/Scripts/SampleSubmission\_TmnO39y.csv  
##/root/hackerday/15\_predict\_big\_mart\_sales/Test\_u94Q5KV.csv  
##/root/hackerday/15\_predict\_big\_mart\_sales/Train\_UWu5bXk.csv  
#/users/Krishna/Python37-32/Scripts/Train\_UWu5bXk.csv

**C:/users/Krishna/Python37-32/Scripts/SampleSubmission\_TmnO39y.csv**

**C:/USERS/Krishna/** Python37-32/Scripts **/15\_predict\_big\_mart\_sales/Test\_u94Q5KV.csv  
C:/USERS/Krishna/ Python37-32/Scripts /Train\_UWu5bXk.csv**

C:/Users/krishna/Python37-32/Scripts/Train\_UWu5bXk.csv

1. Software install

The following versions of software installed.

Python version 3.7

Pycharm 20.1.1. down load the software from <https://www.jetbrains.com/pycharm/download/#section=windows>

Install community version of Picharm IDE. Python editior to be configured configured. Please follow the you tube link on picharm editior <https://www.youtube.com/watch?v=qSJvVAC8uc8>

Pip 20.1.1

Check pip version using the following command. If the version is lower, upgrade the pip version.

pip -v --version

Pandas 2.8.1 .

Note: Pandas3-0.0.1.tar is latest version. This is not compatible with the existing environment

Please use command prompt and pip to install pandas package. Please refer the you tube link for pandas installation. you can also use pycharm editor to select the pandas packages and pip packages required.

Use python -m pip install pandas command to install the pandas version required.

<https://www.youtube.com/watch?v=8Sipkd9vNKk&t=19s>

3. Lenear Regression

3.1 Decision trees

Use decision tree. The package name used is tree.

From sklearn import tree. There is no difference between decision tree and linear model. Regression model to be developed and loaded into Machine learning. Please refer the following link for further inputs in you tube (Gradient boosting models part1- Regression main ideas)

<https://www.youtube.com/watch?v=3CC4N4z3GJc>

3.2 Data load and process

Data loaded using pandas packages using local data process (PYTHON) . Export data into Excel sheets after processing (calculations) into text files or excel data sheets. Prepare data model using regression data model sns.Lmplot.

3.3 Graphical charts

prepare graphical charts using sns.Boxplot Detailed steps are mentions in the following pages of this document.

4 . Data load pandas package code

Source files:

##%%  
#"/users/Krishna/Python37-32/Scripts/SampleSubmission\_TmnO39y.csv"  
##/root/hackerday/15\_predict\_big\_mart\_sales/Test\_u94Q5KV.csv  
##/root/hackerday/15\_predict\_big\_mart\_sales/Train\_UWu5bXk.csv  
#"/users/Krishna/Python37-32/Scripts/Train\_UWu5bXk.csv"  
##%%

## list of packages ##  
import pandas as pd  
  
import pandas as pd  
train=pd.read\_csv("C:/Users/krishna/Python37-32/Scripts\Train\_UWu5bXk.csv")  
train.head()  
  
##%%  
train.describe()  
  
  
  
  
train.head()  
  
sns=train  
import seaborn as sns ## visulazation package  
import matplotlib.pyplot as plt ## visulazation package  
%matplotlib inline ## to get plot on python notebook  
#sns.lmplot('Item\_Weight','Item\_Outlet\_Sales',data=train)  
  
##%%  
  
sns.lmplot("Item\_Weight", "Item\_Outlet\_Sales", data=train) ## scatter plot between sales and weight  
plt.show()  
  
##%%  
  
sns.lmplot("Item\_Visibility", "Item\_Outlet\_Sales", data=train) ## scatter plot between sales and weight  
plt.show()  
  
##%%  
  
sns.lmplot("Item\_MRP", "Item\_Outlet\_Sales", data=train) ## scatter plot between sales and weight  
plt.show()  
  
##%%  
  
sns.boxplot(x="Item\_Fat\_Content",y="Item\_Outlet\_Sales",data=train)  
  
##%%  
  
sns.boxplot(x="Outlet\_Size",y="Item\_Outlet\_Sales",data=train)  
  
##%%  
  
sns.boxplot(x="Outlet\_Size",y="Item\_Outlet\_Sales",data=train)  
  
##%%  
  
## Data preparation..  
train.describe()  
  
##%%  
  
#train.Item\_Weight##  
sum(train['Item\_Weight'].isnull()) ## to check total missing values  
sum(train['Item\_Visibility'].isnull()) ## to check total missing values  
  
##%%  
  
train.apply(lambda x :sum(x.isnull())) ## ( 1st - name of df then apply inside it function)  
  
##%%  
  
## Impute the missing values ## function to be used is fillna()  
train['Item\_Weight'].fillna(12.6,inplace=True) ## missing is replaced by 12.6  
train['Outlet\_Size'].value\_counts()## value\_counts to get the freq  
train['Outlet\_Size'].fillna('Medium',inplace=True)  
train.apply(lambda x :sum(x.isnull())) ## ( 1st - name of df then apply inside it function)  
  
##%%  
  
from sklearn import preprocessing ## sklern is ML library-- preprocessing   
encoding=preprocessing.LabelEncoder() ## create label encoding object  
train['Item\_Fat\_Content'] = encoding.fit\_transform(train['Item\_Fat\_Content'])  
train['Item\_Type'] = encoding.fit\_transform(train['Item\_Type'])  
train['Outlet\_Identifier'] = encoding.fit\_transform(train['Outlet\_Identifier'])  
train['Outlet\_Size'] = encoding.fit\_transform(train['Outlet\_Size'])  
train['Outlet\_Location\_Type'] = encoding.fit\_transform(train['Outlet\_Location\_Type'])  
train['Outlet\_Type'] = encoding.fit\_transform(train['Outlet\_Type'])  
  
##%%  
  
train.head(3)  
  
  
##%%  
  
## Building a Linear Model  
#X\_train=train[features] ##   
#y\_train=train['Item\_Outlet\_Sales'] ##  
   
from sklearn import cross\_validation ##   
#features=['Item\_Weight','Item\_Fat\_Content','Item\_Visibility','Item\_Type','Item\_MRP','Outlet\_Identifier',  
 # 'Outlet\_Establishment\_Year','Outlet\_Size','Outlet\_Location\_Typ','Outlet\_Type']  
features=['Item\_Weight','Item\_Visibility','Outlet\_Type','Item\_MRP','Outlet\_Identifier','Outlet\_Establishment\_Year',  
 'Outlet\_Size','Outlet\_Location\_Type','Outlet\_Type'] ## list of features  
X\_train,X\_test,y\_train,y\_test = cross\_validation.train\_test\_split(train[features],train['Item\_Outlet\_Sales'], test\_size=0.3, random\_state=0)  
  
  
##%%  
  
### use Linear Model for regression  
from sklearn import linear\_model  
from sklearn import cross\_validation ##   
features=['Item\_Weight','Item\_Visibility','Outlet\_Type','Item\_MRP','Outlet\_Identifier','Outlet\_Establishment\_Year',  
 'Outlet\_Size','Outlet\_Location\_Type','Outlet\_Type'] ## list of features  
X\_train,X\_test,y\_train,y\_test = cross\_validation.train\_test\_split(train[features],train['Item\_Outlet\_Sales'], test\_size=0.3, random\_state=0)  
  
regr = linear\_model.LinearRegression() ### Create linear regression object  
regr.fit(X\_train, y\_train) ## fit train/fit the model  
regr.score(X\_test, y\_test) ### check the accuracy on test set  
scores = cross\_validation.cross\_val\_score(regr, train[features],train['Item\_Outlet\_Sales'], cv=5) ##scoring='mean\_squared\_error'  
scores  
  
  
##%%  
  
pred=regr.predict(X\_test) ## predict the sales / target variable on new dataset  
pred  
  
##%%  
  
### Decision Tree model  
from sklearn import tree  
DT = tree.DecisionTreeRegressor(max\_depth=3)

Pandas packages information is available in youtube link