Hands-on with **Python**

Tanujit Chakraborty

Indian Statistical Institute, Kolkata.

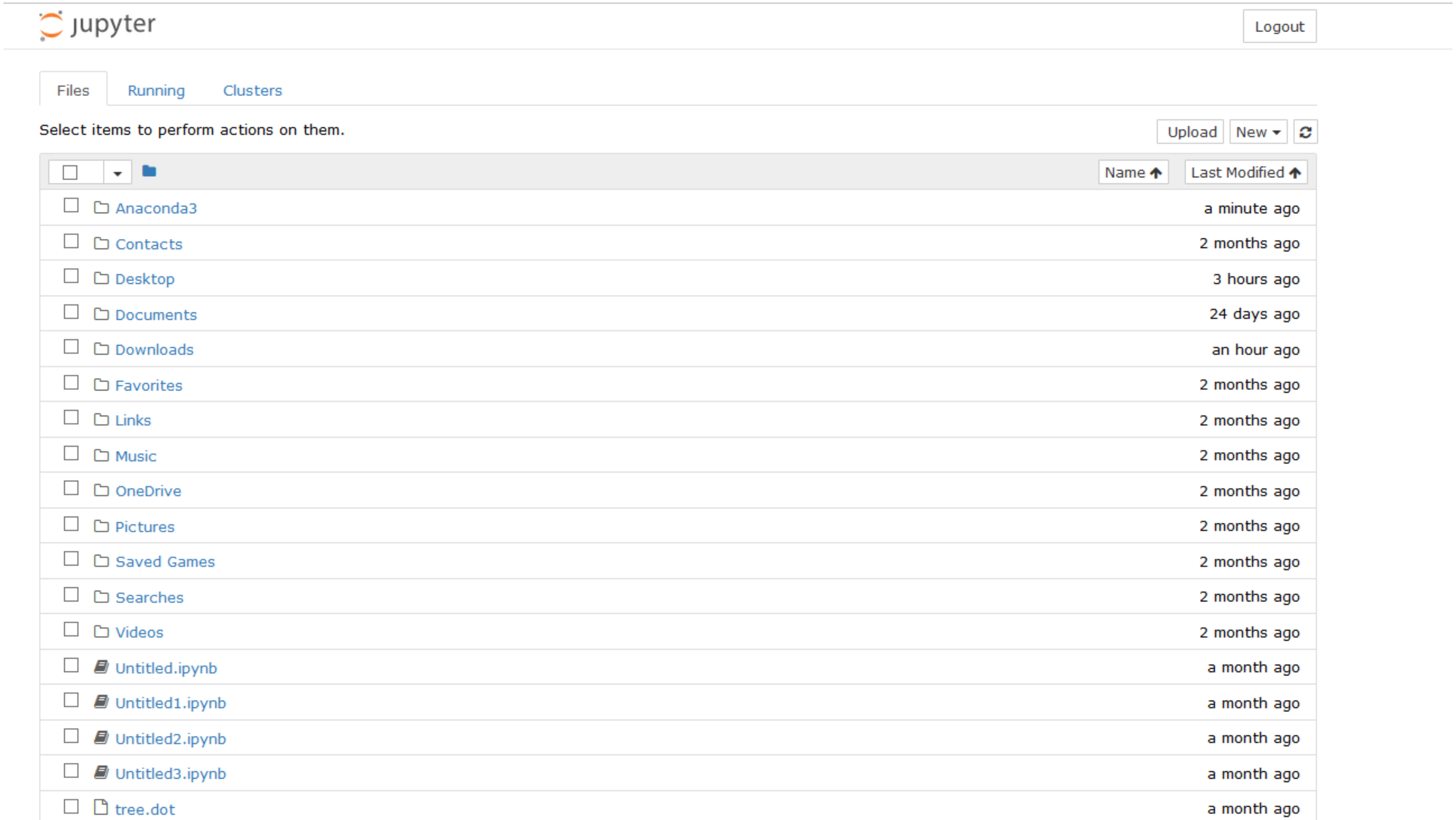
Introduction
to
Python

PYTHON INSTALLATION

1. Download Anaconda from http://jupyter.readthedocs.io/en/latest/install.html
2. Run the set up (exe) file and follow instructions
3. Check Jupyter notebook is installed

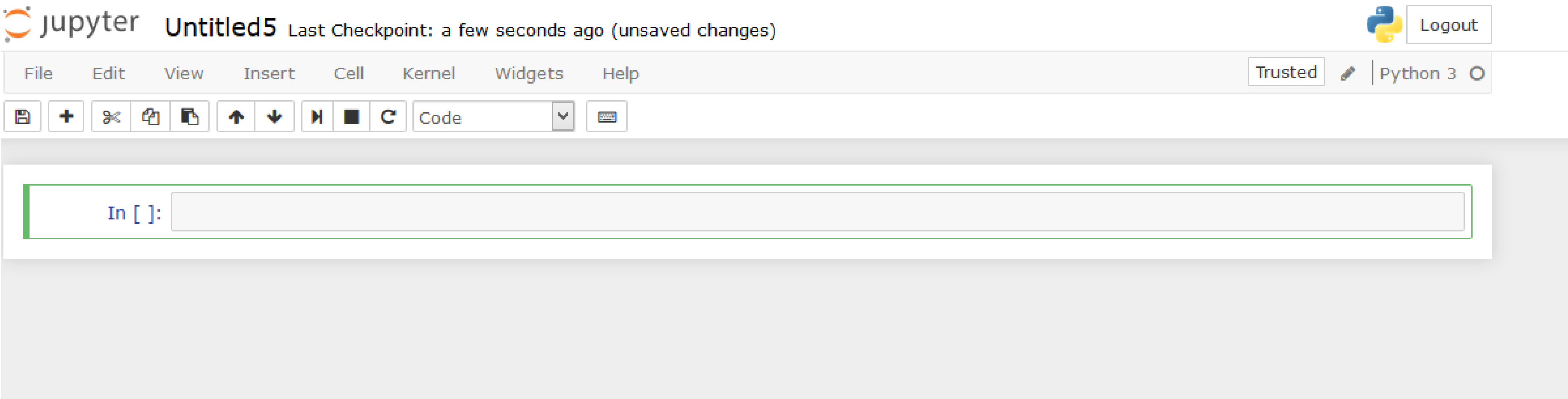
PYTHON INSTALLATION

3. Open Jupyter Notebook



PYTHON INSTALLATION

3. Open Jupyter Notebook



DESCRIPTIVE STATISTICS
using Python

DESCRIPTIVE STATISTICS

Exercise 1: The monthly credit card expenses of an individual in 1000 rupees is given in the file Credit_Card_Expenses.csv.

a. Read the dataset to Python

b. Compute mean, median minimum, maximum, range, variance, standard deviation, skewness, kurtosis and quantiles of Credit Card Expenses

c. Compute default summary of Credit Card Expenses

d. Draw Histogram of Credit Card Expenses

DESCRIPTIVE STATISTICS

Reading a csv file : Source code

```
import pandas as mypd
mydata = mypd.read_csv("E:/ISI/Data/Credit_Card_Expenses.csv")
mydata
```

To read a particular column or variable of data set to a ne variable

Example: Read CC_Expenses to CC

```
cc = mydata.CC_Expenses
cc
```

Operators – Arithmetic & Logical

Operator	Description
+	addition
-	subtraction
*	multiplication
/	division
**	exponentiation
%	modulus (x mod y) 5%2 is 1

Operator	Description
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	exactly equal to
! =	not equal to

DESCRIPTIVE STATISTICS

Descriptive Statistics

Computation of descriptive statistics for variable CC

Function	Code	Value
Mean	cc.mean()	59.2
Median	cc.median()	59
Mode	cc.mode()	59
Standard deviation	cc.std()	3.105
Variance	cc.var()	9.642
Minimum	cc.min()	53
Maximum	cc.max()	65
Percentile	cc.quantile(0.9)	63
Skewness	cc.skew()	-0.09
Kurtosis	cc.kurt()	-0.436

DESCRIPTIVE STATISTICS

Descriptive Statistics

Statistics	Code
Summary	cc.describe()

Statistics	Value
Count	20
Mean	59.2
Standard Deviation	3.1052
Minimum	53
Q1	57
Median	59
Q3	61
Maximum	65

DESCRIPTIVE STATISTICS

Descriptive Statistics

Arithmetic functions for variable CC

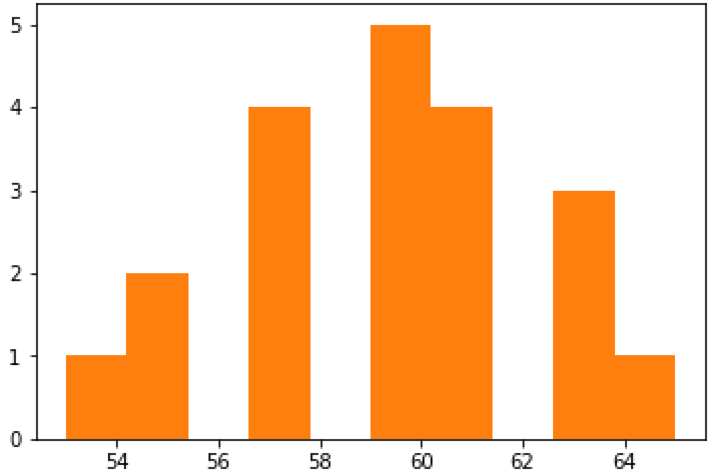
Function	Code	Value
Count	cc.count()	20
Sum	cc.sum()	1148
Product	cc.prod()	6.21447E+18

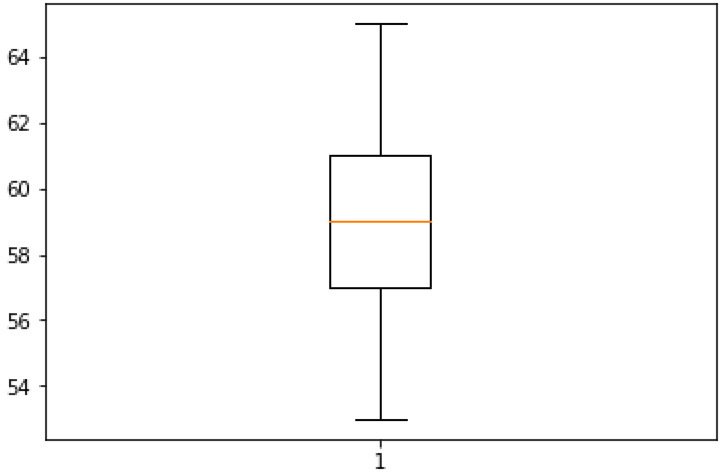
Function	Code	Value
Square root	Import math as mymath mymath.sqrt(49)	7
Sum of Squares	sum(cc**2)	70276

DESCRIPTIVE STATISTICS

Graphs:

Graph	Code
Histogram	import matplotlib.pyplot as myplot myplot.hist(cc) myplot.show()
Box Plot	myplot.boxplot(cc) myplot.show()





CLASSIFICATION and REGRESSION TREE

CLASSIFICATION AND REGRESSION TREE

Objective

To develop a predictive model to classify dependent or response metric (Y) in terms of independent or exploratory variables X's).

When to Use

X's : Continuous or discrete

Y : Discrete or continuous

CLASSIFICATION AND REGRESSION TREE

Classification Tree

When response y is discrete

Method = “DecisionTreeClassifier”

Regression Tree

When response y is numeric

Method = “DecisionTreeRegressor”

CLASSIFICATION AND REGRESSION TREE

Challenges

How to represent the entire information in the dataset using minimum number of rules?

How to develop the smallest tree?

Solution

Select the variable with maximum information (highest relation with y) for first split

CLASSIFICATION AND REGRESSION TREE

Example: A marketing company wants to optimize their mailing campaign by sending the brochure mail only to those customers who responded to previous mail campaigns. The profile of customers are given below. Can you develop a rule to identify the profile of customers who are likely to respond (Mail_Respond.csv)?

Profile Variable	Values
District	0:Urban, 1: Suburban & 2: Rural
House Type	0:Detached, 1: Semi Detached & 2: Terrace
Income	0:Low & 1: High
Previous Customer	0:No & 1:Yes

Output Variable	Value
Outcome	0:No & 1:Yes

CLASSIFICATION AND REGRESSION TREE

Example: A marketing company wants to optimize their mailing campaign by sending the brochure mail only to those customers who responded to previous mail campaigns. The profile of customers are given in Mail_respond.csv? Can you develop a rule to identify the profile of customers who are likely to respond?

Number of variables = 4

SL No	Variable Name	Number of values
1	District	3
2	House Type	3
3	Income	2
4	Previous Customer	2

Total Combination of Customer Profiles = 3 x 3 x 2 x 2 = 36

CLASSIFICATION AND REGRESSION TREE

Read file and variables

```
import pandas as mypd
from sklearn import tree
mydata = mypd.read_csv("E:/ISI/Data/Mail_Respond.csv")
x = mydata["District", "House_Type", "Income", "Previous_Customer"]
y = mydata.Outcome
```

CLASSIFICATION AND REGRESSION TREE

Develop the model

```
mymodel = tree.DecisionTreeClassifier(min_samples_split = 10)
mymodel.fit(x,y)
mymodel.score(x,y)
```

Statistics	Value (%)
Accuracy	100
Misclassification Error	0.00

CLASSIFICATION AND REGRESSION TREE

Model Accuracy measures

```
pred = mymodel.predict(x)

mytable = mypd.crosstab(y, pred)

mytable
```

Actual Vs predicted: %

Actual	Predicted	
	No	Yes
No	34	0
Yes	0	66

Accuracy = 34 + 66 = 100%

CLASSIFICATION AND REGRESSION TREE

Exercise 1: Develop a tree based model for predicting whether the customer will take pep (0: No & 1: Yes) using the customer profile data given in bank-data.csv? Use 80% of data to develop the model and validate the model using the remaining 20% of data?

Variables	Values
Age	Numeric
Sex	0:Male & 1: Female
Region	0: Inner City, 1: Rural, 2: Suburban & 3: Town
Income	Numeric
Married	0: No, 1: Yes
Children	Numeric
Car	0: No, 1: Yes
Saving Account	0: No, 1: Yes
Current Account	0: No, 1: Yes
Mortgage	0: No, 1: Yes

Exercise 1: Develop a tree based model for predicting whether the customer will take pep using the customer profile data given in bank-data.csv? Use 80% of data to develop the model and validate the model using the remaining 20% of data?

Reading data

```
import pandas as mypd
from sklearn import tree
from sklearn.cross_validation import train_test_split

mydata = mypd.read_csv("E:/ISI/PM-01/Data/bank-data.csv")
x = mydata.values[:, 0:9]
y = mydata.values[:, 10]
```

CLASSIFICATION AND REGRESSION TREE

Exercise 1: Develop a tree based model for predicting whether the customer will take pep using the customer profile data given in bank-data.csv? Use 80% of data to develop the model and validate the model using the remaining 20% of data?

Split data into training and test data

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2,
random_state = 100)
```

Develop model using training data

```
mymodel = tree.DecisionTreeClassifier(min_samples_split=50)
mymodel.fit(x_train, y_train)
mymodel.score(x_train, y_train)
```

Statistics	Value (%)
Accuracy	83.3
Misclassification Error	16.7

CLASSIFICATION AND REGRESSION TREE

Exercise 1: Develop a tree based model for predicting whether the customer will take pep using the customer profile data given in bank-data.csv? Use 80% of data to develop the model and validate the model using the remaining 20% of data?

```
pred = mymodel.predict(x_train)
mytable = mypd.crosstab(y_train, pred)
mytable
```

Actual vs Predicted

Actual	Predicted	
	No	Yes
No	232	30
Yes	50	168

CLASSIFICATION AND REGRESSION TREE

Exercise 1: Develop a tree based model for predicting whether the customer will take pep using the customer profile data given in bank-data.csv? Use 80% of data to develop the model and validate the model using the remaining 20% of data?

Validating the Model using test data

```
pred_test = mymodel.predict(x_test)
mytesttable = mypd.crosstab(y_test, pred_test)
mytesttable
```

Actual Vs predicted: %

Actual	Predicted	
	No	Yes
No	58	6
Yes	15	41

Accuracy = (58 + 41)/(58 + 6 + 15 +41) = 82.5 %

Exercise 1: Develop a tree based model for predicting whether the customer will take pep using the customer profile data given in bank-data.csv? Use 80% of data to develop the model and validate the model using the remaining 20% of data?

Data	Accuracy	Misclassification Error
Training	83.33	16.67
Test	82.5	17.5

RANDOM FOREST

and

BAGGING

RANDOM FOREST

Improves predictive accuracy

Generates large number of bootstrapped trees

Classifies a new case using each tree in the new forest of trees

Final predicted outcome by combining the results across all of the trees

Regression tree – average

Classification tree – majority vote

RANDOM FOREST

- Uses trees as building blocks to construct more powerful prediction models
- Decision trees suffer from high variance

  If we split the data into two parts and construct two different trees for each half of the data, the trees can be quite different

- In contrast, a proceedure with low varaince will yield similar results if applied repeatedly to distinct datasets
- Bagging is a general purpose procedure for reducing the variance of a statistiocal learning method

RANDOM FOREST

Procedure

- Take many training sets from the population
- Build seperate prediction models using each training set
- Average the resulting predictions
- Averaging of a set of observatins reduce variance
- Different training datasets are taken using bootstrap sampling
- Generally bootstraped sample consists of two third of the observations and the model is tested on the remainng one third ofthe out of the bag observations

For discrete response – will take the majority vote instead of average

Major difference between bagging and Random Forest

Bagging generally uses all the *p* predictors while random forest uses $\sqrt{p}$ predictors

RANDOM FOREST

Example

Develop a model to predict the medain value of owner occupied homes using Boston_Housing_Data ? Use 80% of the data to develop the model and validate the model using remaining 20% of the data?

Python Code

Call libraries and import data

```
import pandas as mypd
from sklearn.ensemble import RandomForestRegressor
from sklearn.cross_validation import train_test_split
import math as mymath

mydata = mypd.read_csv("E:/ISI/PM-01/Data/Boston_Housing_Data.csv“)
x = mydata.values[:, 0:12]
y = mydata.values[:,13]
```

RANDOM FOREST

Example

Develop a model to predict the medain value of owner occupied homes using Boston_Housing_Data ? Use 80% of the data to develop the model and validate the model using remaining 20% of the data?

Python Code

Split data into training and test

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2,
random_state = 100)
```

Develop the model using training data - Bagging

```
mymodel = RandomForestRegressor(n_estimators = 500,
                  min_sample_split = 40, max_features = None)
mymodel.fit(x_train,y_train)
```

n_estimators : Number of trees

max_features = None, include all (p) explanatory variable (x’s)

max_features = ‘auto’, include subset ($\sqrt{p}$) explanatory variable (x’s)

RANDOM FOREST

Example

Develop a model to predict the medain value of owner occupied homes using Boston_Housing_Data ? Use 80% of the data to develop the model and validate the model using remaining 20% of the data?

Python Code

```
mymodel.score(x_train, y_train)
pred = mymodel.predict(x_train)
res = y_train – pred
res_sq = res**2
res_ss = res_sq.sum()
total_ss = y_train.var()*404
r_sq = 1 - res_ss/total_ss
mse = res_sq.mean()
rmse = mymath.sqrt(mse)
```

RANDOM FOREST

Example

Develop a model to predict the medain value of owner occupied homes using Boston_Housing_Data ? Use 80% of the data to develop the model and validate the model using remaining 20% of the data?

Statistics	Value
MSE	3.733
RMSE	1.932
R^2	95.41

RANDOM FOREST

Example

Develop a model to predict the medain value of owner occupied homes using Boston_Housing_Data ? Use 80% of the data to develop the model and validate the model using remaining 20% of the data?

Python Code

Validate the model using test data

```
pred_test = mymodel.predict(x_test)
res_test = y_test- pred_test
res_test_sq = res_test**2
res_test_ss = res_test_sq.sum()
total_test_ss = t_test.var()*101
r_test_sq = 1 - res_test_ss/total_test_ss
mse = res_test_sq.mean()
rmse = mymath.sqrt(mse)
```

Example

Develop a model to predict the medain value of owner occupied homes using Boston_Housing_Data ? Use 80% of the data to develop the model and validate the model using remaining 20% of the data?

Statistics	Training	Test
MSE	3.733	18.007
RMSE	1.932	4.243
R^2	95.41	81.17

RANDOM FOREST

Example

Develop a model to predict the medain value of owner occupied homes using Boston_Housing_Data ? Use 80% of the data to develop the model and validate the model using remaining 20% of the data?

Developing model with random forest

mymodel = RandomForestRegressor(n_estimators = 500, min_samples_split = 40, max_features= 'auto’]

Developing model with CART

mymodel = tree.DecisiontreeRegressor(min_samples_split=40)

Statistics	Bagging		Random Forest		Regression Tree	
	Training	Test	Training	Test	Training	Test
MSE	3.733	18.007	4.449	20.169	13.287	28.879
RMSE	1.932	4.243	2.109	4.491	3.645	5.373
R^2	95.41	81.17	94.52	78.91	83.65	69.81

ARTIFICIAL NEURAL NETWORKS

ARTIFICAL NEURAL NETWORKS

Introduction

One of the most fascinating machine learning modeling technique

Generally uses back propagation algorithm

Relatively complex (due to deep learning with many hidden layers)

Structure is inspired by brain functioning

Generally computationally expensive

Instructions

1. Normalize the data – Use Min – Max transformation (optional)

   Normalized data = Data – Minimum / (Maximum – Minimum)

2. Number of hidden layers required = 1 for vast number of application

3. Number of neurons required = 2/3 of the number of predictor variables or input layers

Remark: The optimum number of layers and neurons are the ones which would minimize mean square error or misclassification error which can be obtained by testing again and again

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file. The factors and response considered are given below. Use 80% of the data to develop the model and validate the model using remaining 20% of the data?

SL No	Factor
1	Individual expected level of activity score
2	Transaction speed score
3	Peer comparison score in terms of transaction volume

Response	Values
Outcome	0: Not Paid and 1: Paid

Example

Importing packages

```
import pandas as mypd
from sklearn.cross_validation import train_test_split
from sklearn.neural_network import MLPClassifier
```

Reading the data

```
mydata = mypd.read_csv("E:/ISI/PM03/Course_Material/Data/Logistic_Reg.csv")
x = mydata.values[:, 0:3]
y = mydata.Outcome
```

Splitting the data into training and test

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state
= 100)
```

Example

Develop the model

```
mymodel =MLPClassifier(solver = 'lbfgs', alpha = 1e-5, hidden_layer_sizes = (2), random_state = 100)

mymodel.fit(x_train, y_train)
```

Note:

Classification problem: Use MLPCLassifier

Value estimation: Use MLPRegressor

Solver:
'lbfgs' : Uses quasi-Newton method optimization algorithm.
'sgd'   :Uses stochastic gradient descent optimization algorithm.
'adam' :Uses stochastic gradient-based optimizer

Example: Interpretation

hidden_layer_sizes : a vector representing hidden layers and hidden neurons in each layer

hidden_layer_sizes = (l) : one hidden layers with *l* hidden neurons

Output

mymodel.score(x_train, y_train)

Statistics	Value
% Accuracy	96.81
% Error	3.19

mymodel.predict_proba(x_train)

Output: Validation

```
predtest = mymodel.predict(x_test)
mytable = mypd.crosstab(y_test, predtest)
mytable
```

Actual Vs Predicted

		Predicted	
		0	1
Actual	0	54	4
	1	0	138

Output: Validation

Actual Vs Predicted (%)

		Predicted	
		0	1
Actual	0	27.55	2.04
	1	0.00	70.41

Statistics	Training	Test
% Accuracy	96.81	97.96
% Error	3.19	2.04

Output

```
> mse = mean(res^2)
> rmse = sqrt(mse)
> residual_ss = sum(res^2)
> total_ss = var(myzdata$Conversion)*15
> r_sq = 1 – residual_ss / total_ss
```

Statistics	Value
Mean Square Error	0.0009994
Root Mean Square Error	0.0316128
R Square	0.9905

Prediction for new data set

```
> test <- read_csv("E:/ISI/output.csv“)
> output = compute(mymodel, test)
> output$net.result
```

Temperature	Time	Kappa_Number	Conversion	Predicted Conversion
1	0.0058	0.1243	0.9577	0.9882
1	0.0058	0.2090	0.9915	0.9813
1	0.0000	0.3220	1.0000	0.9782
1	0.0173	0.4633	0.9437	0.9269
1	0.0231	0.6610	0.9155	0.8871

Exercise 1

Develop a model to predict the medain value of owner occupied homes using Boston_Housing_data ? Use 80% of the data to develop the model and validate the model using remaining 20% of the data?

Exercise 1

Python Code – Import the packages

```
import pandas as mypd
from sklearn.cross_validation import train_test_split
from sklearn.neural_network import MLPRegressor
```

Import the data

```
mydata = mypd.read_csv("E:/ISI/PM- 03/Course_Material/Data/ Boston_Housing_Data.csv")
x = mydata.values[:, 0:12]
y = mydata.values[:,13]
```

Split data into training and test

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 100)
```

Exercise 1

Develop the model

```
mymodel = MLPRegressor(solver = 'lbfgs', alpha = 0.001, hidden_layer_sizes =
          (6), random_state= 100)
mymodel.fit(x_train, y_train)
mymodel.score(x_train,y_train)
```

Statistic	Value
R^2	66.76

Validation: Test data

```
pred = mymodel.predict(x_test)
res = y_test – pred
res_sq = res**2
res_ss = sum(res_sq)
total_ss = y_test.var()*100

rsq = 1 - res_ss/total_ss
rsq
```

Statistic	Training	Test
R^2	66.76	63.43

Thank You