LOVELY PROFESSIONAL UNIVERSITY

*Transforming Education Transforming India*

# Teaching Assistant Evaluation

# Data Set

**Submitted by -**

Name : Khushboo Soni

Reg No : 11615575

Roll No : B 40

**Submitted to -**

Dr. Gokulnath K

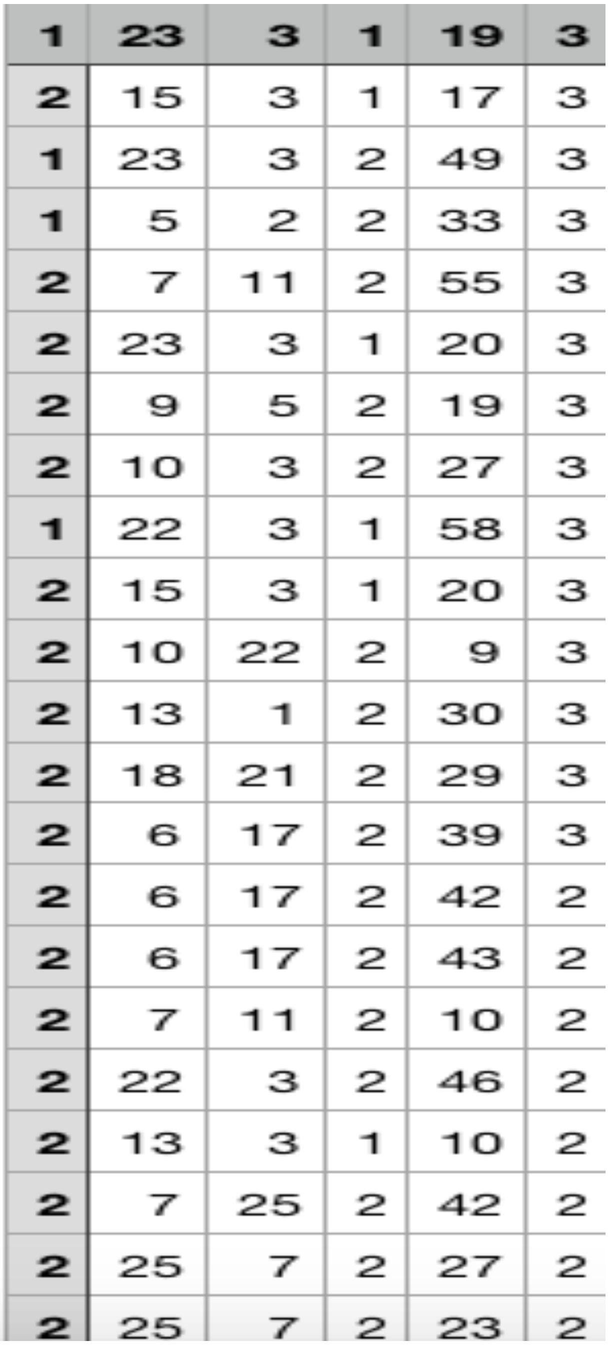
# 1.Abstract

Teaching performance evaluation can be done using multiple sources, like students, peers and teachers themselves. Even though only peers have the substantive expertise for a relevant evaluation, it is generally well-known that students are qualified to assess some of the classroom teaching aspects: clarity of the presentation, interpersonal rapport with students etc. The core idea of this research is to study if there can be built a computational model that uses past student evaluation in order to predict future teaching performance assessments. There can be designed different system based on supervised machine learning techniques. In this paper there are built several models based on classification techniques with the purpose of finding a model that has the smaller classification error of the new cases that means higher accuracy.

***Keyword*** : Teaching performance evaluation , Logistic regression , train-test-split

# 2.The Dataset

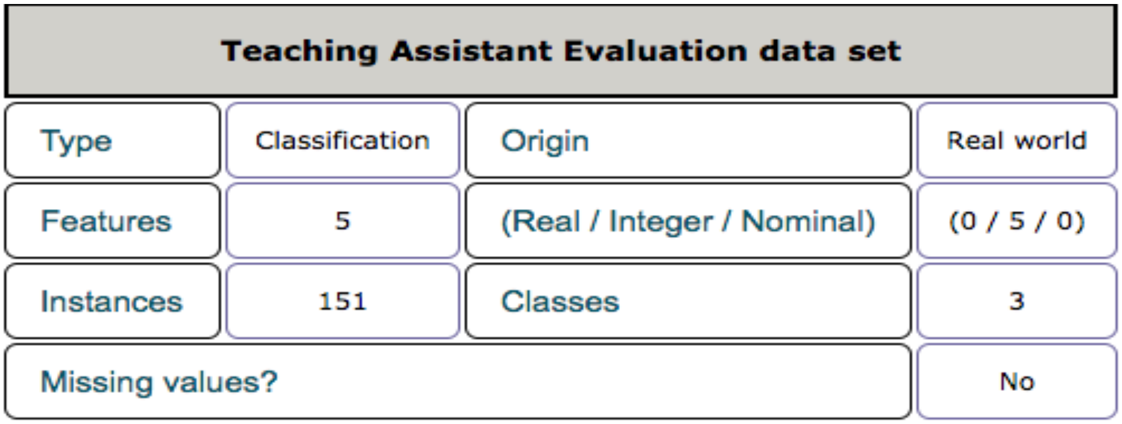
The data consist of evaluations of teaching performance over three regular semesters and two summer semesters of 151 teaching assistant (TA) assignments at the Statistics Department of the University of Wisconsin-Madison. The scores were divided into 3 roughly equal-sized categories ("low", "medium", and "high") to form the class variable.



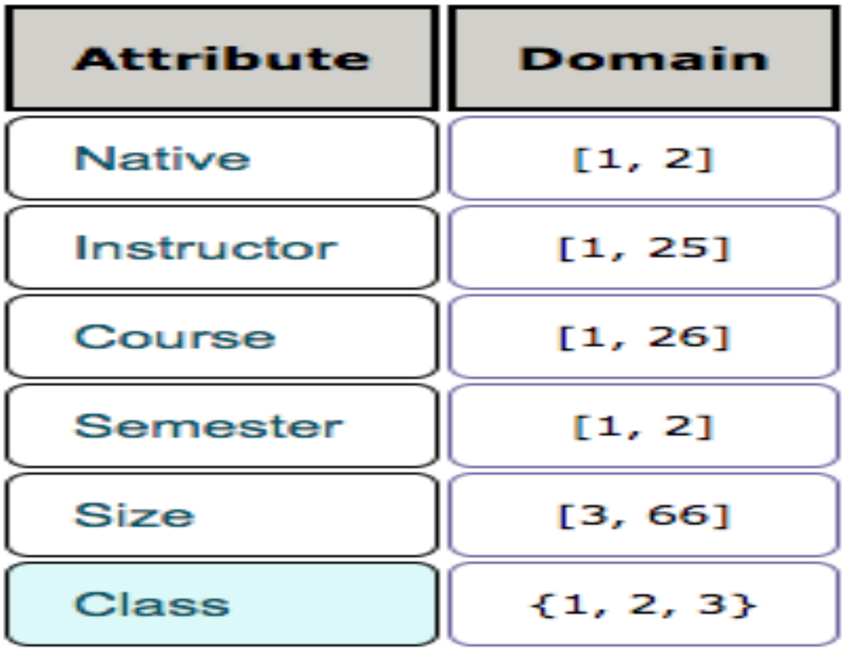
| 1 | 23 | 3 | 1 | 19 | 3 |
|---|---|---|---|---|---|
| 2 | 15 | 3 | 1 | 17 | 3 |
| 1 | 23 | 3 | 2 | 49 | 3 |
| 1 | 5 | 2 | 2 | 33 | 3 |
| 2 | 7 | 11 | 2 | 55 | 3 |
| 2 | 23 | 3 | 1 | 20 | 3 |
| 2 | 9 | 5 | 2 | 19 | 3 |
| 2 | 10 | 3 | 2 | 27 | 3 |
| 1 | 22 | 3 | 1 | 58 | 3 |
| 2 | 15 | 3 | 1 | 20 | 3 |
| 2 | 10 | 22 | 2 | 9 | 3 |
| 2 | 13 | 1 | 2 | 30 | 3 |
| 2 | 18 | 21 | 2 | 29 | 3 |
| 2 | 6 | 17 | 2 | 39 | 3 |
| 2 | 6 | 17 | 2 | 42 | 2 |
| 2 | 6 | 17 | 2 | 43 | 2 |
| 2 | 7 | 11 | 2 | 10 | 2 |
| 2 | 22 | 3 | 2 | 46 | 2 |
| 2 | 13 | 3 | 1 | 10 | 2 |
| 2 | 7 | 25 | 2 | 42 | 2 |
| 2 | 25 | 7 | 2 | 27 | 2 |
| 2 | 25 | 7 | 2 | 23 | 2 |

# 3. Attribute Information:

1. Whether of not the TA is a native English speaker (binary) :
   1=English speaker,
   2=non-English speaker
2. Course instructor (categorical, 25 categories)
3. Course (categorical, 26 categories)
4. Summer or regular semester (binary) 1=Summer, 2=Regular
5. Class size (numerical)
6. Class attribute (categorical) 1=Low, 2=Medium, 3=High



| Teaching Assistant Evaluation data set | | | |
|---|---|---|---|
| Type | Classification | Origin | Real world |
| Features | 5 | (Real / Integer / Nominal) | (0 / 5 / 0) |
| Instances | 151 | Classes | 3 |
| Missing values? | | | No |



| Attribute | Domain |
|---|---|
| Native | [1, 2] |
| Instructor | [1, 25] |
| Course | [1, 26] |
| Semester | [1, 2] |
| Size | [3, 66] |
| Class | {1, 2, 3} |

# 4. Software Requirement Analysis

- **Python** : Python is an interpreted high-level programming language for general-purpose programming. It is created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notaly using significant whitespace. It provides constructs that enable clear programming on both small and large scale.

- **Jupyter-Notebook** : The Jupyter Notebook is an open-source web application that allows you to create and share documents that contains live code, equations, visualizations and narrative text.

Uses includes : data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

- **Comma-separated values (CSV)** : In computing, a comma-separated values file is a delimited text file that uses a comma to separate values. A CSV file stores tabular data (numbers and text) in plain text. Each line of the file is a record. Each record consists of one or more fields, separated by commas. The use of the comma as a field separator is the source of the name for this file format.

- **matplotlib** : Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hard-copy formats and interactive environments across platforms. Matplotlib can be used in

Python scripts, the Python and the Jupyter notebook, Web application servers, and four graphical user interface toolkits.

- **Pandas** : Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.

- **Numpy :** NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python.

- **scikit-learn :** Scikit-learn (formerly scikits.learn) is a free software machine learning library for the Python programming language.[3] It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

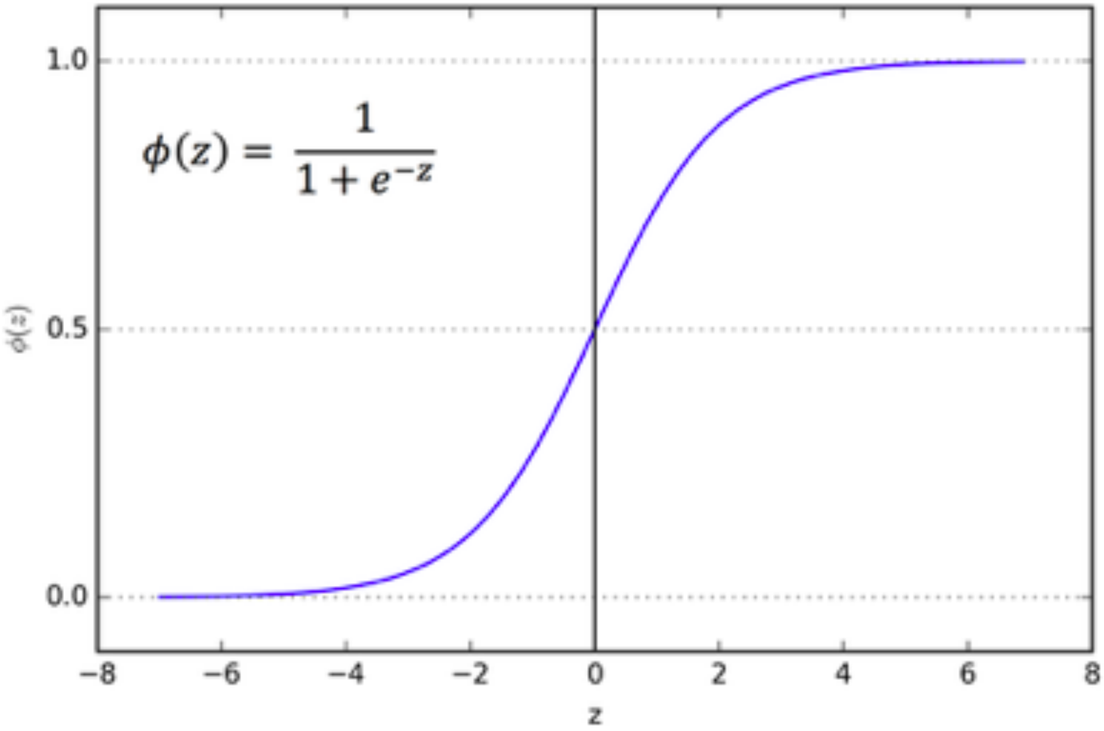
# 5. Logistic Regression

Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Unlike linear regression which outputs continuous number values, logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes.

## Activation Function used in Logistic Regression

Sigmoidal Activation function is used for logistic regression. The range for sigmoid activation function is from 0 to 1.
Sigmoidal Activation function

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}.$$



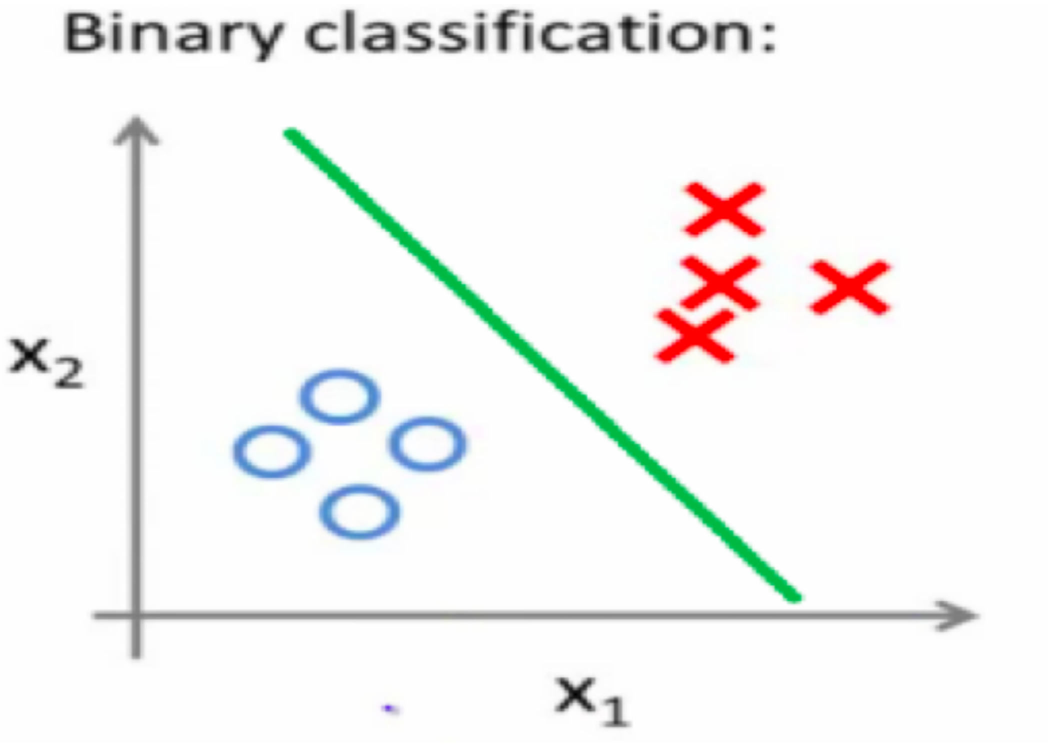
# Types of Logistic Regression

- Binary Logistic Regression
- Multi Logistic Regression

## Binary Logistic Regression

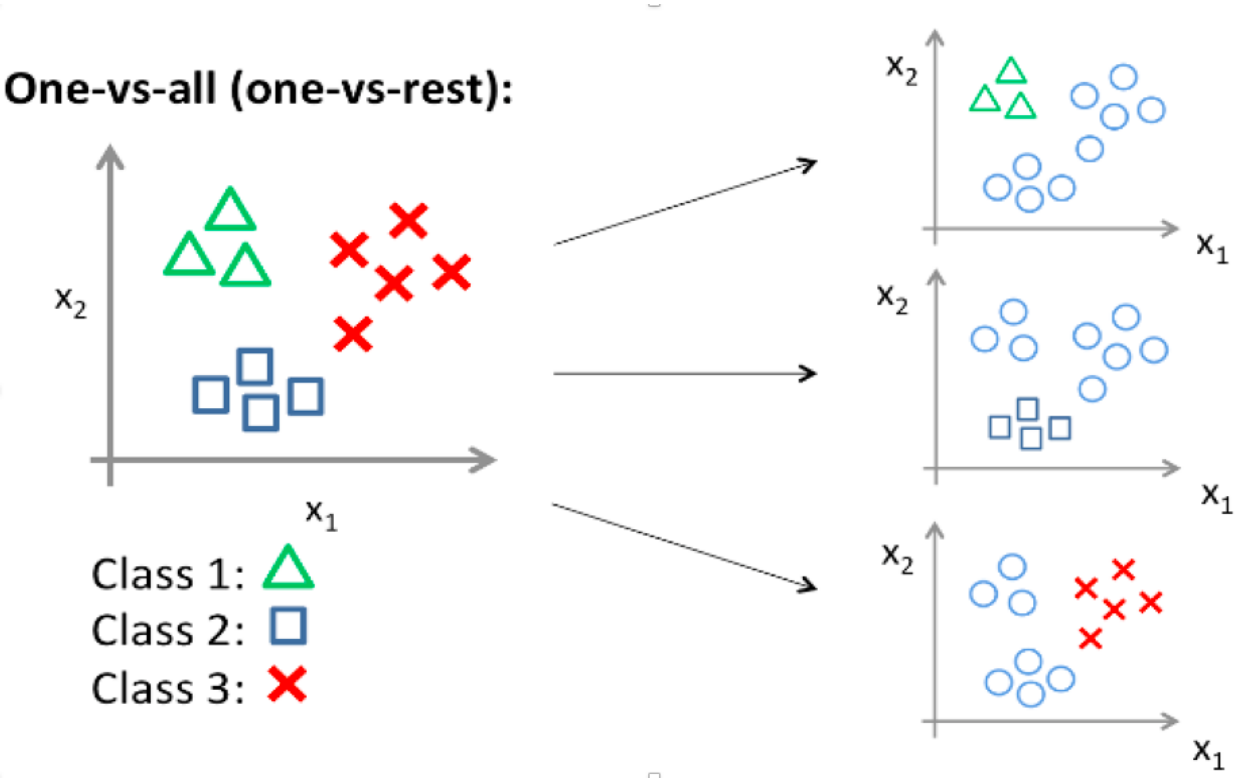
In binary logistic regression the data is to be classified into classes. Each feature set will belong to either one of this classes. For example, for the dataset having size of tumour the algorithm has to predict whether the tumour in benign or malignant. In this case the algorithm will classify the data either to benign class or to malignant class. Another example is to filter out spam mails in that case each email will be classified either as spam or not.

Logistic regression is best used for binary classification even for multi class classification we will go for multiple binary classifiers.



# Multi Logistic Regression

In multi logistic regression the data is to be classified into more than 2 classes. For example, in the dataset having rating of cricket player in batting, bowling, fielding classifying each player as all-rounder, batsman, bowler or fielder. In this case we are having 4 classes and each player must be classified into one of these class.

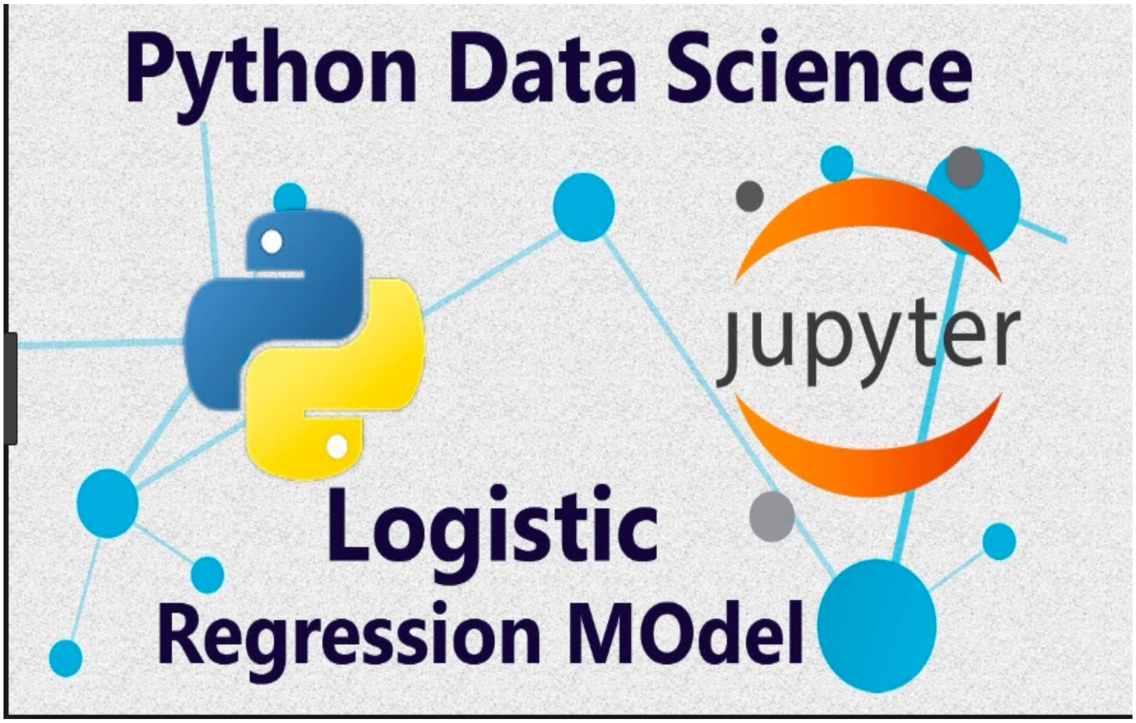


For solving multi-class classification, the one vs rest approach is used, in this approach if there are n classes, n classifier. The first classifier classifies the data in class one or not in class one, the second classifier classifies as belong to class 2 or not, and so on for all the classifiers. When a new test data comes for prediction the data is classified for all the n classifiers and one which gives maximum probability for classification is selected.

# Scikit-Learn Logistic Regression Model

There are several Python libraries which provide solid implementations of a range of machine learning algorithms. One of the best known is Scikit-Learn, a package that provides efficient versions of many common algorithms. Scikit-Learn is characterized by a clean, uniform, and streamlined API, as well as by very useful and complete online documentation.
The Logistic regression model lies under linear_model in sklearn (i.e Scikit-learn) library. It takes number of parameters if none of them is explicitly specified the default values are taken for these parameters.

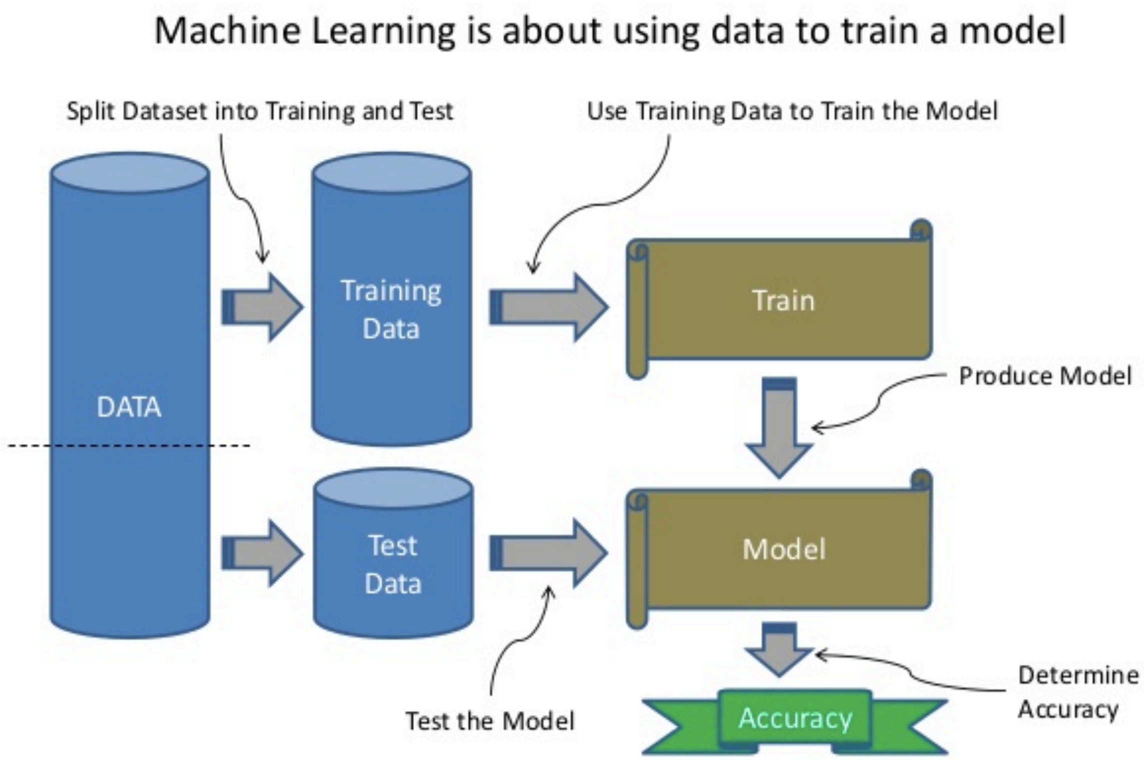
class sklearn.linear_model.**LogisticRegression**(penalty=’l2’, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None,solver=’warn’, max_iter=100, multi_class=’warn’, verbose=0, warm_start=False, n_jobs=None)



# 6. Training and Test Sets : Splitting Data

The idea of dividing your data set into two subsets:

- **training set —** a subset to train a model
- **test set —** a subset to test the trained model


Machine Learning is about using data to train a model
Split Dataset into Training and Test
Use Training Data to Train the Model
DATA
Training Data
Train
Produce Model
Test Data
Model
Test the Model
Accuracy
Determine Accuracy

**Slicing a single data set into a training set and test set.**

Make sure that your test set meets the following two conditions:

- Is large enough to yield statistically meaningful results.
- Is representative of the data set as a whole. In other words, don't pick a test set with different characteristics than the training set.

Assuming that your test set meets the preceding two conditions, your goal is to create a model that generalizes well to new data.

Our test set serves as a proxy for new data. For example, consider the following figure. Notice that the model learned for the training data is very simple. This model doesn't do a perfect job— a few predictions are wrong. However, this model does about as well on the test data as it does on the training data. In other words, this simple model does not overfits the training data.

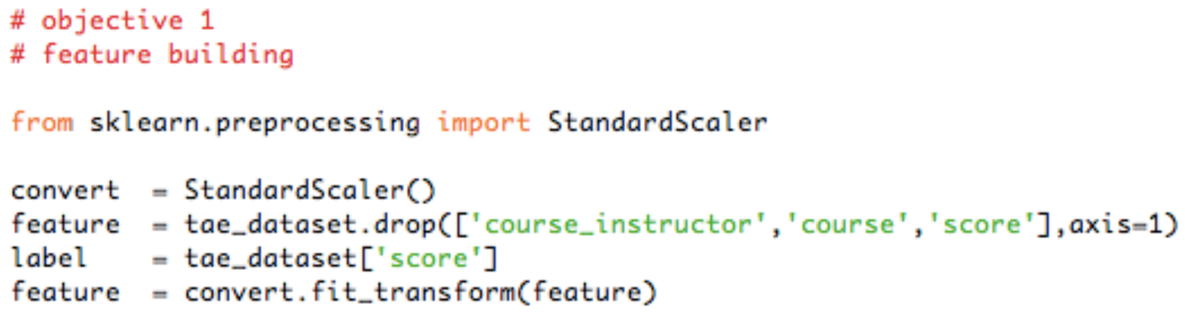
**Validating the trained model against test data.**

**Never train on test data.** If you are seeing surprisingly good results on your evaluation metrics, it might be a sign that you are accidentally training on the test set. For example, high accuracy might indicate that test data has leaked into the training set.

For example, consider a model that predicts whether an email is spam, using the subject line, email body, and sender's email address as features. We apportion the data into training and test sets, with an 80-20 split. After training, the model achieves 99% precision on both the training set and the test set. We'd expect a lower precision on the test set, so we take another look at the data and discover that many of the examples in the test set are duplicates of examples in the training set (we neglected to scrub duplicate entries for the same spam email from our input database before splitting the data). We've inadvertently trained on some of our test data, and as a result, we're no longer accurately measuring how well our model generalizes to new data.

# 7.Objectives of project

### Objective 1:

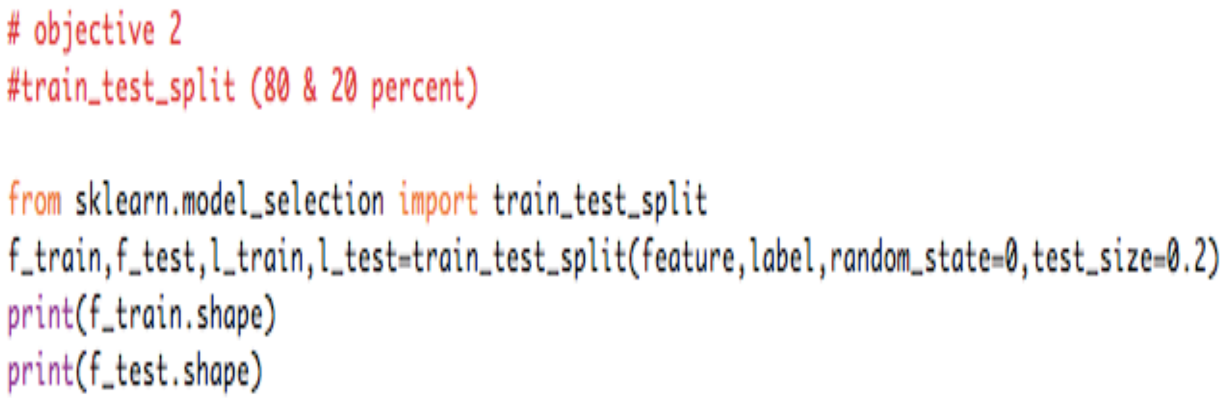


```
# objective 1
# feature building

from sklearn.preprocessing import StandardScaler

convert  = StandardScaler()
feature  = tae_dataset.drop(['course_instructor','course','score'],axis=1)
label    = tae_dataset['score']
feature  = convert.fit_transform(feature)
```

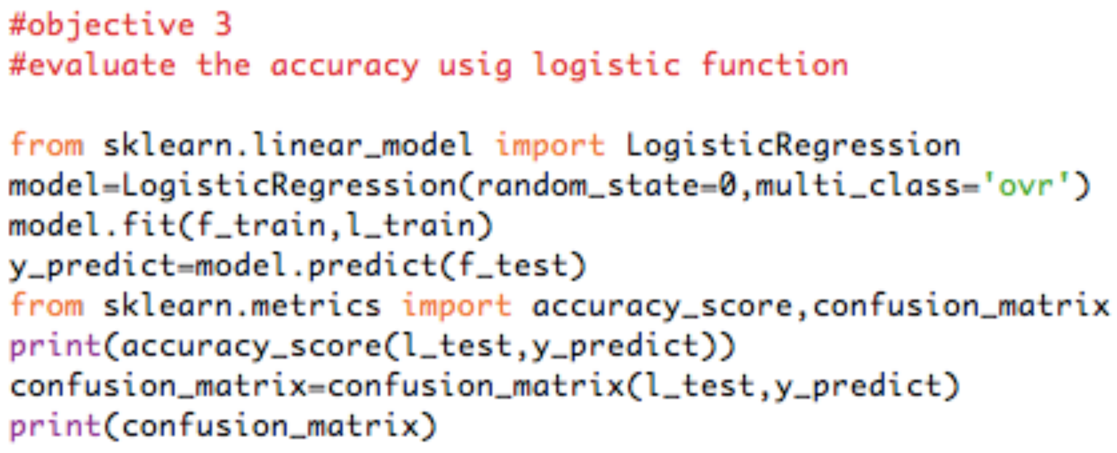
### Objective 2:



```
# objective 2
#train_test_split (80 & 20 percent)

from sklearn.model_selection import train_test_split
f_train,f_test,l_train,l_test=train_test_split(feature,label,random_state=0,test_size=0.2)
print(f_train.shape)
print(f_test.shape)
```

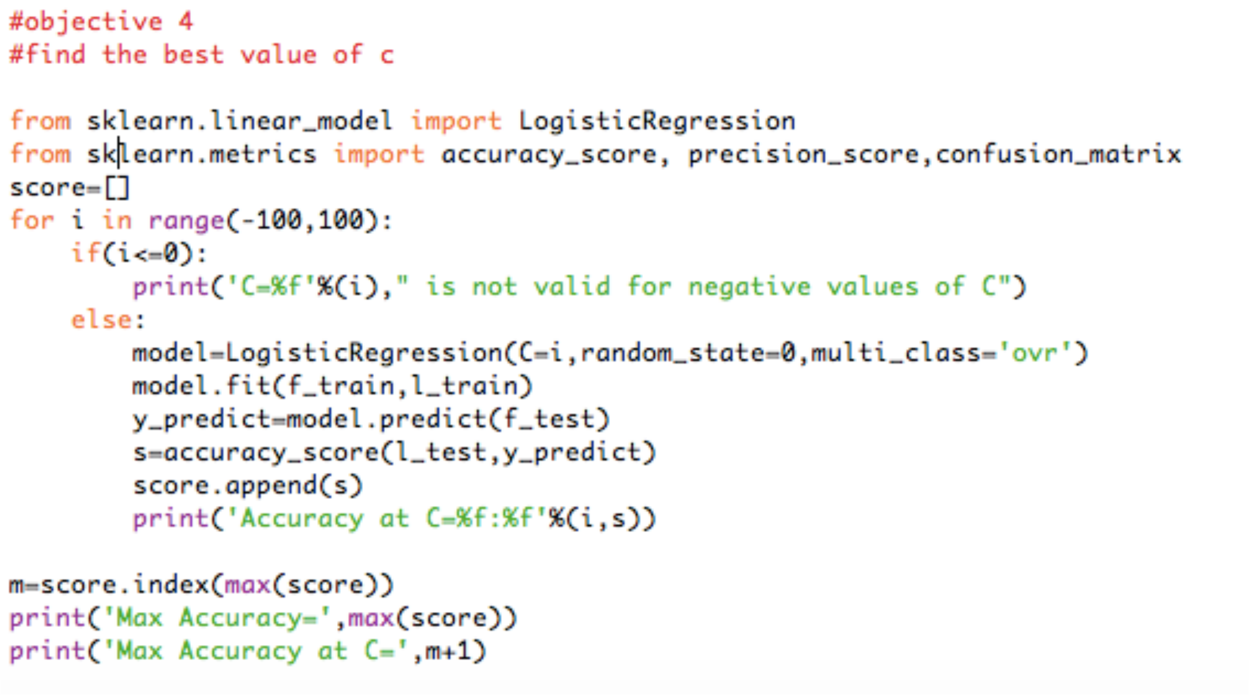
### Objective 3:



```
#objective 3
#evaluate the accuracy usig logistic function

from sklearn.linear_model import LogisticRegression
model=LogisticRegression(random_state=0,multi_class='ovr')
model.fit(f_train,l_train)
y_predict=model.predict(f_test)
from sklearn.metrics import accuracy_score,confusion_matrix
print(accuracy_score(l_test,y_predict))
confusion_matrix=confusion_matrix(l_test,y_predict)
print(confusion_matrix)
```

# Objective 4 :



```
#objective 4
#find the best value of c

from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score,confusion_matrix
score=[]
for i in range(-100,100):
    if(i<=0):
        print('C=%f'%(i)," is not valid for negative values of C")
    else:
        model=LogisticRegression(C=i,random_state=0,multi_class='ovr')
        model.fit(f_train,l_train)
        y_predict=model.predict(f_test)
        s=accuracy_score(l_test,y_predict)
        score.append(s)
        print('Accuracy at C=%f:%f'%(i,s))

m=score.index(max(score))
print('Max Accuracy=',max(score))
print('Max Accuracy at C=',m+1)
```

# 7. References

- http://archive.ics.uci.edu/ml/datasets/Teaching+Assistant+Evaluation

- Loh, W.-Y. & Shih, Y.-S. (1997). Split Selection Methods for Classification Trees, Statistica Sinica 7: 815-840.

- Lim, T.-S., Loh, W.-Y. & Shih, Y.-S. (1999). A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-three Old and New Classification Algorithms. Machine Learning.

- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

- https://en.wikipedia.org/wiki/Logistic_regression

- http://cs231n.github.io/python-numpy-tutorial/

- https://www.learnpython.org/en/Pandas_Basics