Logistic Regression Project

In this application we will be working on dataset called exam scores. We will try to create a model that will predict whether or not the candiate admitted or not admitted based on two exam scores.

Import Libraries

Import a required libraries

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
%matplotlib inline
```

Get the Data

Read in the examscores.csv file and set it to a data frame called examscores.

In [3]:
```
df =pd.read_csv("examscores.csv")
```

Check the head of examscores

In [5]:
```
df.head()
```

Out[5]:

	score1	score2	result
0	34.623660	78.024693	0
1	30.286711	43.894998	0
2	35.847409	72.902198	0
3	60.182599	86.308552	1
4	79.032736	75.344376	1

Use info and describe() on examscores

In [ ]:
```

```

In [ ]:
```

```

Logistic Regression

Now it's time to do a train test split, and train our model!

Split the data into training set and testing set using train_test_split

1. Define X with score1 and score2 and Y data with result
2. Use train_test_split() with X and Y

In [13]:
```
X=df[["score1", "score2"]]
y=df["result"]
```

In [14]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
```

Train and fit a logistic regression model on the training set.

In [15]:
```
logmodel = LogisticRegression()
logmodel.fit(X_train,y_train)
```

Out[15]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

In [ ]:

Predictions and Evaluations

Now predict values for the testing data.

In [16]:
```
predictions = logmodel.predict(X_test)
```

Create a classification report and confusion matrix for the model.

In [17]:
```
print(confusion_matrix(y_test, predictions))
```

```
[[ 7  3]
 [ 0 23]]
```

In [18]:
```
print(classification_report(y_test,predictions))
```

```
             precision    recall  f1-score   support

          0       1.00      0.70      0.82        10
          1       0.88      1.00      0.94        23

avg / total       0.92      0.91      0.90        33
```