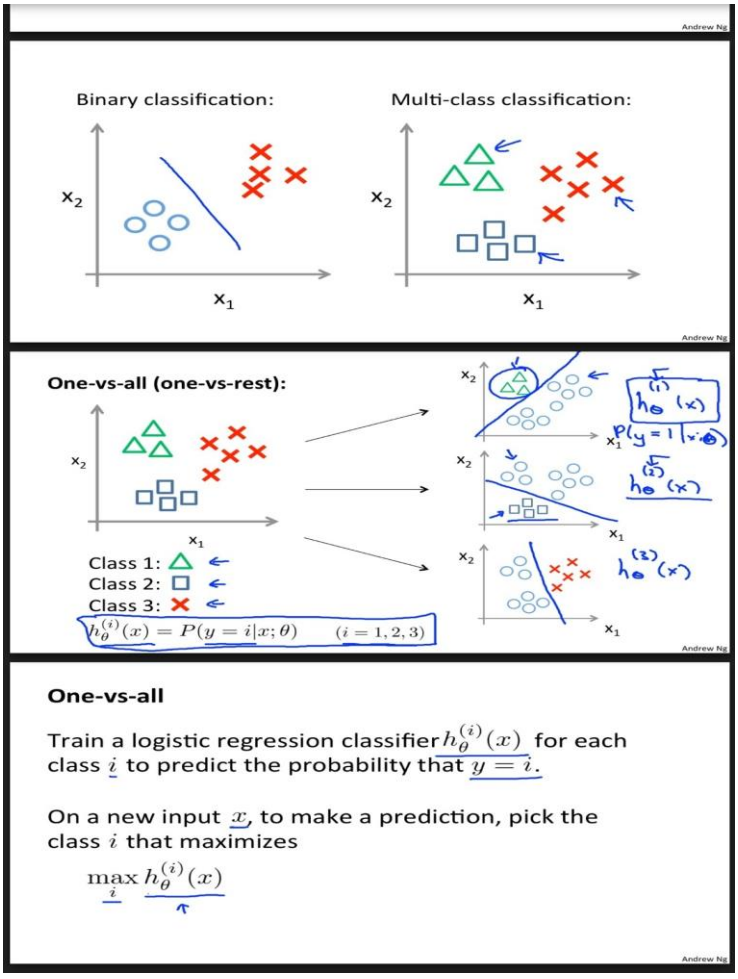
Multinomial logistic regression

Logistic regression is one of the most popular supervised classification algorithms. This classification algorithm mostly used for solving binary classification problems. People follow the myth that logistic regression is only useful for the binary classification problems.

Which is not true. Logistic regression algorithm can also use to solve the multi-classification problems.

Example:
https://dataaspirant.com/2017/05/15/implement-multinomial-logistic-regression-python/

Coursera: Machine Learning Andrew NG



Learning Phase: One vs all classification

Multi class classification is implemented by training multiple logistic regression classifiers, one for each of the K classes in the training dataset.

In the above Multi Class classification example, there are 3 classes. Hence, we need to train 3 different logistic regression classifiers.

When training the classifier for Class 1, we will treat input data with class 1 labels as +ve samples (y==1) and all other classes as -ve samples (y==0).

When training the classifier for Class 2, we will treat input data with class 2 labels as +ve samples (y==1) and all other classes as -ve samples (y==0).

This will continue for all the classes.

Prediction phase: One vs all prediction

Once training all our classifiers, we can now use it to predict which class the test data belongs to.

For the test input, we should compute the "probability" that it belongs to each class using the trained logistic regression classifiers. Your one-vs-all prediction function will pick the class for which the corresponding logistic regression classifier outputs the highest probability and return the class label (1, 2,..., or K) as the prediction for the input example.

For example,

```
1.          P(y==Class1) = 0.3
2.          P(y==Class2) = 0.5
3.          P(y==Class3) = 0.2
```

Then, the one-vs-all function will choose the label as Class 2 as it has higher probability than the other classes.

"mul_lr = linear_model.LogisticRegression(multi_class='multinomial', solver='newton-cg').fit(train_x, train_y)"