Classification

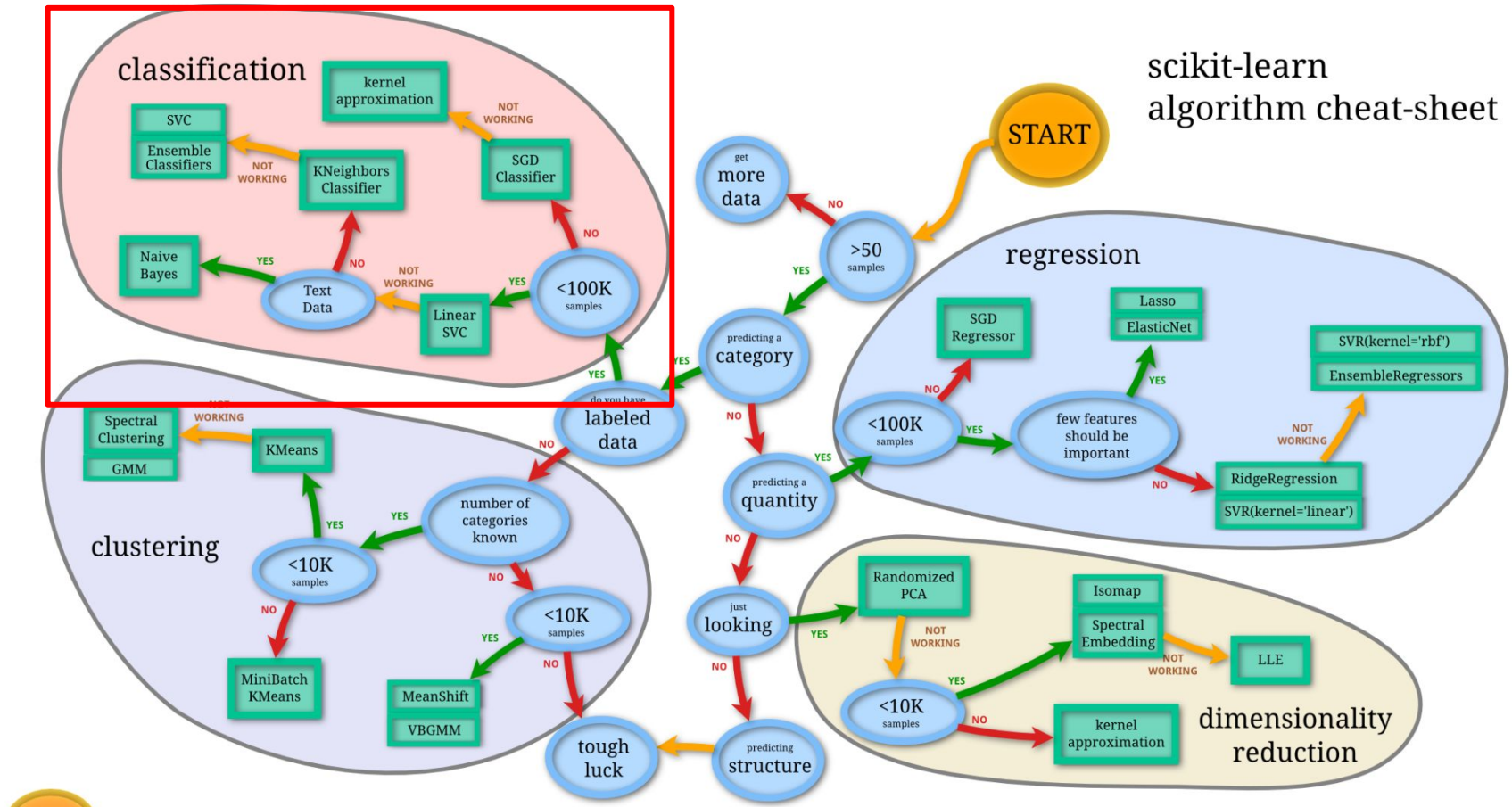
Chantri Polprasert
CPDSAI

Agenda

- Classification
- Logistic regression
- Regularization
- Multiclass classification using multiple binary classifiers (OvR & OvO)

Introduction

- Previously, we've been focused on the regression problem, the supervised machine learning problem when our response variable is a continuous quantity.
- Machine learning algorithms that aim at predicting a categorical (AKA qualitative) response are referred to as classification techniques.
- In this section, rather than trying to predict a continuous responses, we will be trying to classify observations in to categories (AKA classes)


scikit-learn
algorithm cheat-sheet
START
get more data
NO
>50 samples
YES
predicting a category
YES
NO
predicting a quantity
YES
NO
just looking
YES
NO
predicting structure
tough luck
do you have labeled data
YES
NO
classification
kernel approximation
NOT WORKING
SVC
Ensemble Classifiers
NOT WORKING
KNeighbors Classifier
SGD Classifier
NO
Naive Bayes
YES
Text Data
NO
NOT WORKING
Linear SVC
YES
<100K samples
YES
clustering
Spectral Clustering
GMM
NOT WORKING
KMeans
YES
number of categories known
YES
<10K samples
NO
MiniBatch KMeans
NO
<10K samples
YES
MeanShift
VBGMM
NO
regression
SGD Regressor
NO
<100K samples
YES
few features should be important
YES
Lasso
ElasticNet
NO
RidgeRegression
SVR(kernel='linear')
NOT WORKING
SVR(kernel='rbf')
EnsembleRegressors
dimensionality reduction
Randomized PCA
YES
NOT WORKING
<10K samples
YES
Isomap
Spectral Embedding
NOT WORKING
LLE
NO
kernel approximation
Back
scikit learn

Objectives

- Distinguish different items into predefined categories based on their certain characteristics
- We have some prototype samples that teach us the characteristics of each categories
- In supervised learning framework, we have
  - Categories = label
  - Characteristics = features
  - Prototype samples = training set

Why not Linear Regression?

- Suppose that we are trying to predict the medical condition of a patient in the emergency room on the basis of her symptoms: stroke, drug overdose and epileptic seizure
- Consider encoding these values as a quantitative response variable, Y, as follows:

$$Y = \begin{cases} 1 & \text{if stroke;} \\ 2 & \text{if drug overdose;} \\ 3 & \text{if epileptic seizure.} \end{cases}$$

Question: What would a prediction of 1.5 signify?

Class levels

- Numeric codings for a response fed into a continuous model assumes a natural order, or hierarchy.
- For multi-class categorical data, this is simply inappropriate.
- But what about a binary classification problem?

$$Y = \begin{cases} 1, & \texttt{Stroke} \\ 0, & \texttt{No Stroke} \end{cases}$$

Can we use this in a linear model such that a predicted value of 0.5 suggests a 50-50 chance that a person's chance of stroke?

Logistic Regression

- A statistical method for predicting binary classes.
- The outcome or target variable is dichotomous in nature.
- A special case of linear regression where the target variable is categorical
- From linear regression equation:

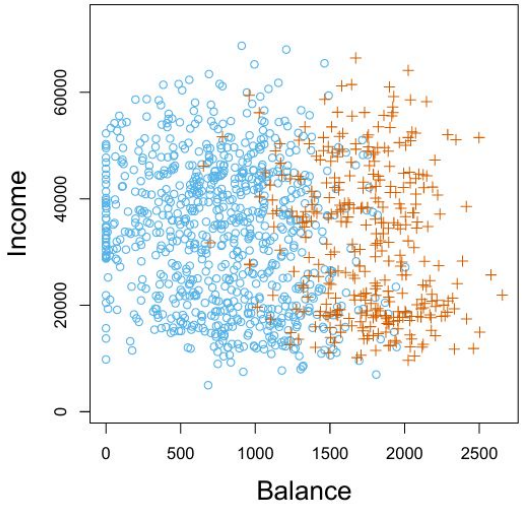
$$y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

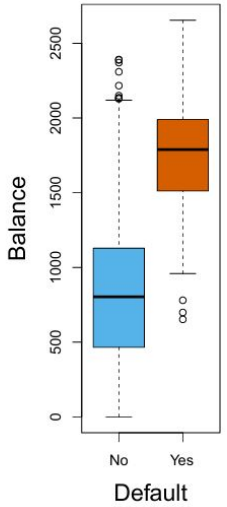
- Logistic Regression predicts the probability of the positive class p(Y=1|X) using by applying logistic function (sigmoid)

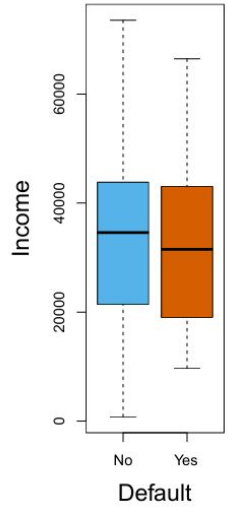
$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p)}}$$

Logistic Regression

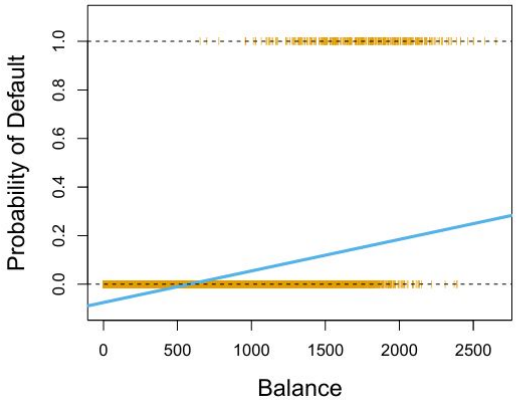
- Left: The annual incomes and monthly credit card balances of a number of individuals.
- Center: Boxplots of balance as a function of default status.
- Right: Box plots of income as a function of default status.

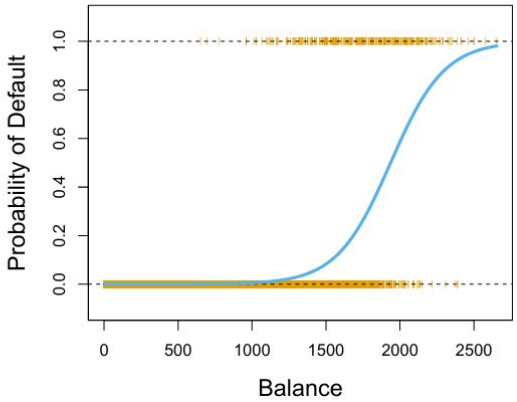






- Left: Estimated probability of default using linear regression. Some estimated probabilities are negative! The orange ticks indicate the 0/1 values coded for default (No or Yes).
- Right:Predicted probabilities of default using logistic regression. All probabilities lie between 0 and1.





Logistic Regression: Binary case

- The probability of the positive class:

$$\hat{p}(X_i) = \operatorname{expit}(X_i w + w_0) = \frac{1}{1 + \exp(-X_i w - w_0)}.$$

- The cost function with regularization term $r(w)$

$$\min_w \frac{1}{S} \sum_{i=1}^{n} s_i \left(-y_i \log(\hat{p}(X_i)) - (1 - y_i) \log(1 - \hat{p}(X_i))\right) + \frac{r(w)}{SC}$$

where s_i corresponds to the weights assigned by the user to a specific training sample

$$S = \sum_{i=1}^{n} s_i.$$

penalty	$r(w)$
None	0
ℓ_1	$\|w\|_1$
ℓ_2	$\frac{1}{2}\|w\|_2^2 = \frac{1}{2} w^T w$
ElasticNet	$\frac{1-\rho}{2} w^T w + \rho \|w\|_1$

ρ is l1_ratio

Issues using linear regression for classification

1. A regression method cannot accommodate a qualitative response with more than two classes
2. A regression method will not provide meaningful estimates of Pr(Y |X), even with just two classes.

Logistic Regression: Multinomial case (Softmax)

- The probability of predicting the kth class ($k \in K$)

$$\hat{p}_k(X_i) = \frac{\exp(X_i W_k + W_{0,k})}{\sum_{l=0}^{K-1} \exp(X_i W_l + W_{0,l})}.$$

- The cost function with regularization term $r(w)$

$$\min_W -\frac{1}{S} \sum_{i=1}^{n} \sum_{k=0}^{K-1} s_{ik} [y_i = k] \log(\hat{p}_k(X_i)) + \frac{r(W)}{SC}$$

where s_{ik} corresponds to the weights assigned by the user to a specific training sample

$$S = \sum_{i=1}^{n} \sum_{k=0}^{K-1} s_{ik}.$$

penalty	$r(W)$
None	0
ℓ_1	$\|W\|_{1,1} = \sum_{i=1}^{m} \sum_{j=1}^{K} \|W_{i,j}\|$
ℓ_2	$\frac{1}{2}\|W\|_F^2 = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{K} W_{i,j}^2$
ElasticNet	$\frac{1-\rho}{2}\|W\|_F^2 + \rho\|W\|_{1,1}$

ρ is l1_ratio

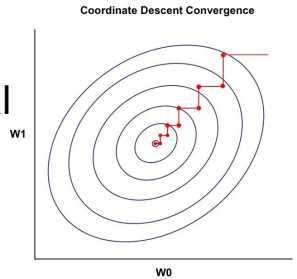
LogisticRegression

```
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='lbfgs', max_iter=100, multi_class='deprecated', verbose=0,
warm_start=False, n_jobs=None, l1_ratio=None)                                [source]
```

- Logistic Regression (aka logit, MaxEnt) classifier.
- **C**, default=1.0: Inverse of regularization strength; must be a positive float.
- Multi_class = ‘multinomial’.
- penalty{‘l1’, ‘l2’, ‘elasticnet’, None}, default=’l2’
- Class_weight: dict or ‘balanced’, default=None

Optimization algorithms for Logistic Regression in sklearn

- Limited-memory Broyden–Fletcher–Goldfarb–Shanno (lbfgs): (Default), belongs to quasi-Newton methods
  - May perform better when the dataset is small as it saves a lot of memory
  - Some serious drawbacks such that if it is not paid attention to, it may not converge to anything!
- A Library for Large Linear Classification (liblinear)
  - Use a Coordinate Descent (CD) algorithm that solves optimization problems by successively performing approximate minimization along coordinate directions or coordinate hyperplanes.
  - Recommended for high dimension dataset (recommended for solving large-scal classification problems).



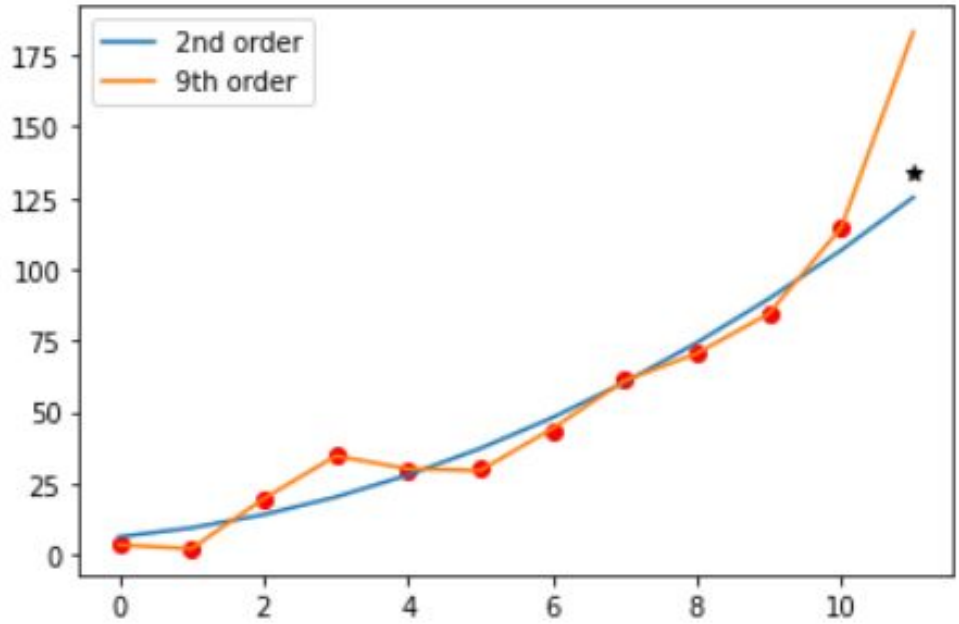
https://medium.com/@arnavr/scikit-learn-solvers-explained-780a17bc322d

Regularization

Overfitting revisits

The simplified story of Generalization, Overfitting, and Underfitting

High variance could lead to low training errors, but high prediction errors – the situation called overfitting

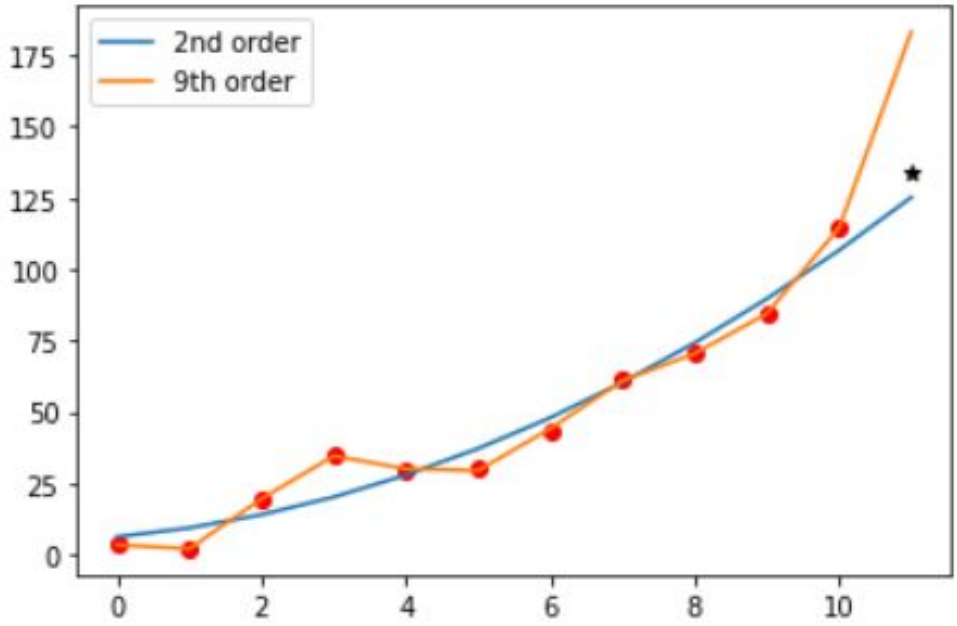


This situation indicates that the model cannot generalize the training data effectively

Overfitting revisits

To improve the predictive ability, we can neglect some errors during training, in order to allow the model to better generalize the training data

For example, we might choose the model with less complexity (here, 2nd order model)



Regulalization

- Normally, we do not measure the model complexity explicitly
- The strategy is to choose model with relatively high complexity in order to reduce bias
- If the size of the data is not sufficiently large for the model, overfitting would occur
- We perform regularization to improve generalizability of the model
- There are a variety of regularization methods– here, in linear model, we use loss function regularization

Regularization on loss function

- Loss function for classification model

$$\min_{w} \frac{1}{S} \sum_{i=1}^{n} s_i \left(-y_i \log(\hat{p}(X_i)) - (1 - y_i) \log(1 - \hat{p}(X_i)) \right) + \frac{r(w)}{SC}$$

- The regularization term $r(w)$ is added to the loss function. It can be $||w||_1$ or $||w||_2$
- *C* is the penalty term (similar to alpha in Linear regression)

High C -> weak regularization
Low C -> strong regularization

Rationales for minimizing size of the weights

- Adding regularization term reduces the freedom in choosing the coefficient values (only small values are allowed), we have less choices in choosing the coefficient, thus less model complexity
- Lower values of coefficients lead to less variance: instances that are closed to each other might be assigned with different values

Smaller $|w|$, less variance

- $x_1 = [1,1.2]$; and $x_2 = [1,0.9]$ suppose that b=0
- Suppose that $w = [0.1, -0.1]$, we get
  - $f(x_1) = 0.1 * 1 - 0.1 * 1.2 = -0.02$ and $f(x_2) = 0.1 * 1 - 0.1 * 0.9 = 0.01$
- Suppose that $w = [100, -100]$, we get
  - $f(x_1) = 100 * 1 - 100 * 1.2 = -20$ and $f(x_2) = 100 * 1 - 100 * 0.9 = 10$
- We can see that although x_1 and x_2 are closed to each other, the resulting decision functions are highly different in the 2nd case.
- This is how small $|w|$ causes less variance

Multiclass classification using multiple binary classifiers

Multiclass classification using binary class model

Instead of using generalized cross entropy model (Softmax), we can decompose multiple classes problem into multiple binary-class sub-problems

- One vs Rest (OneVsRestClassifier)
- One vs One (OneVsOneClassifier)

One vs Rest

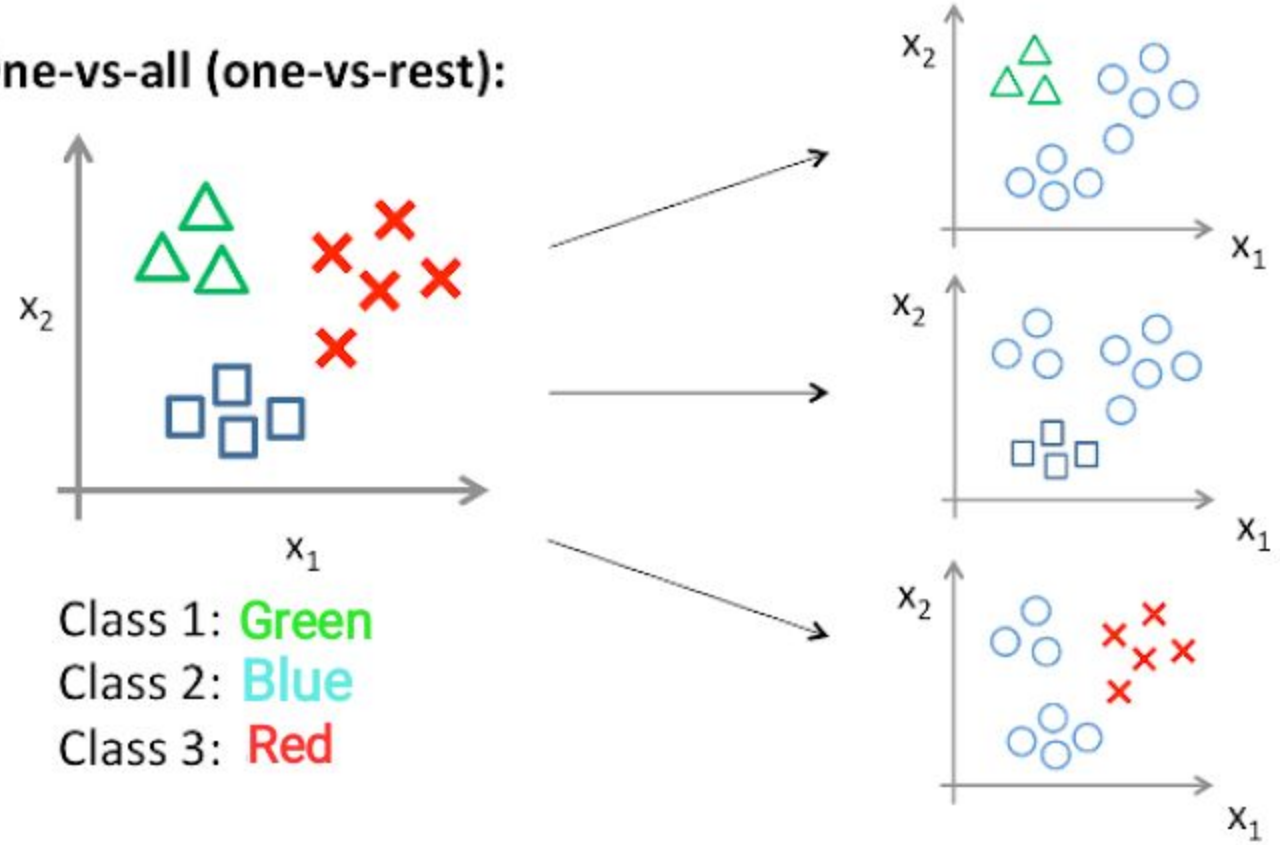
- Assume that for 4 classes: {1,2,3,4}
- 4 Binary Classifiers
  - 1 vs {2,3,4} Classifier : Model#1 {w1,b1}
  - 2 vs {1,3,4} Classifier : Model#2 {w2,b2}
  - 3 vs {1,2,4} Classifier : Model#3 {w3,b3}
  - 4 vs {1,2,3} Classifier : Model#4 {w4,b4}
- In general (n classes):
  - *n* binary classifiers - each on all data
- prediction in one vs rest
  - Class with highest score

- To make a prediction, compute the decision function of all classifiers (4) on a new data point. Choose the class with the highest score:

$$\widehat{y} = \arg\max_i (\mathbf{w}_i^T \mathbf{x} + b_i)$$

- Unclear why it works but yields acceptable performance
- We have one coefficient vector (***w***) and one bias (*b)* for each class

One-vs-all (one-vs-rest):



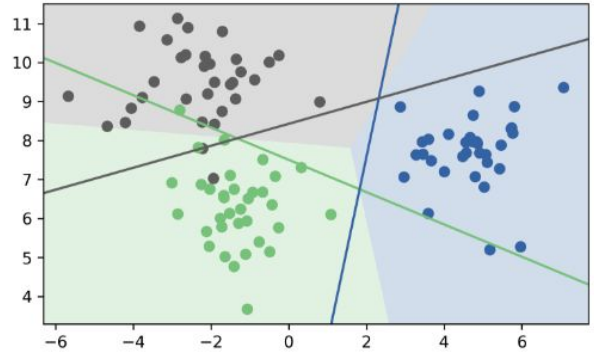
https://towardsdatascience.com/multi-class-classification-one-vs-all-one-vs-one-94daed32a87b

One vs One

- 1vs2, 1vs3, 1vs4, 2vs3, 2vs4, 3vs4
- Given 4 classes, how many binary classifiers do we need?
- n * (n-1) / 2 binary classifiers - each trained on a fraction of the data
  - Classify by all classifiers.
  - Count how often each class was predicted.
  - Return majorly predicted class.
  - Again - just a heuristic.

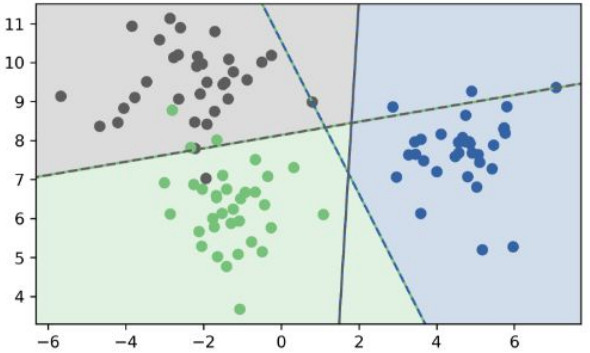
One vs Rest

- n_classes classifiers
- trained on imbalanced datasets of original size
- Retains some uncertainty?



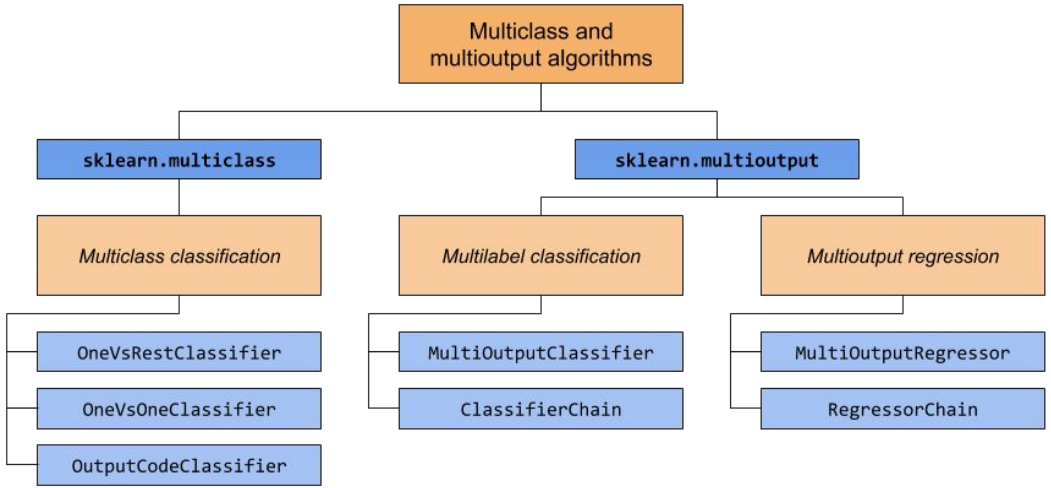
One vs One

- n_classes * (n_classes - 1)/2 classifiers
- trained on balanced subsets
- No uncertainty propagated



Multinomial logistic regression

- Scikit provides the option that automatically determine if the labels are binary or multiclass
- For Logistic Regression, the training algorithm uses the one-vs-rest (OvR) scheme



https://scikit-learn.org/stable/modules/multiclass.html

Summary

- Classifier outputs an integer
- Classifier utilizes log-loss as a cost function
- Logistic regression maps the output into [0,1], can be analyzed using probability value
- Regularization goal is to minimize error from overfit problem (similar to regression analysis)
- k-Multiclass classification can be simplified to k binary classifiers

References

- https://scikit-learn.org/1.5/modules/multiclass.html
- https://scikit-learn.org/1.5/modules/generated/sklearn.linear_model.LogisticRegression.html#
- ISLR Chapter 4