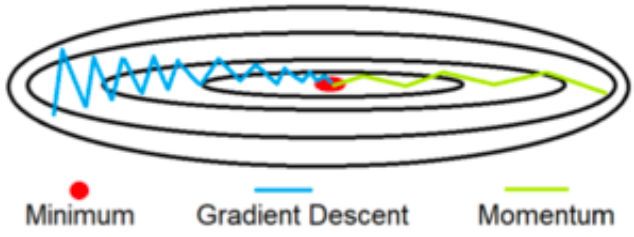
Assignment 5

1. Multi-class classification
    - One-vs-Rest (`sklearn.multiclass.OneVsRestClassifier`)
        - N-classes:
            - N binary classifiers
            - N-split of the training set, each includes all data points but divided into class i v.s. others)
        - Prediction: `argmax` of scores produced by a bunch of binary classifiers
    - One-vs-One (`sklearn.multiclass.OneVsOneClassifier`)
        - N classes:
            - $\binom{N}{2}$ binary classifiers
            - $\binom{N}{2}$ split of the training set, each includes only data points with the label i and j
        - Prediction:
            - majority votes
            - Or the class with the most sum score is taken as the class label
    - Compare:
        - the number of classifiers?
        - data used for training each classifier?
        - what if there is a specific class dominates the training set (i.e. imbalanced classes)?
        - when N is large, what will be the greatest issues for the two algorithms respectively?
2. SGD & BGD & mini-BGD

    Suppose the size of the training set is N
    - Number of data points used in each iteration?
    - Fluctuation? Who converges faster?
    - Flexibility? Who's more likely to get stuck at a certain local minimum?

    How to make the search for the minima more efficient?
    - Momentum: record the parameter update history



(ref: https://optimization.cbe.cornell.edu/index.php?title=Momentum)

- Method: include some amount (controlled by a hyperparameter) into the update equation.
  - common choices of the hyperparameter: 0.8, 0.9, 0.99...
  - q: what if the hyperparameter equals zero?
- Algorithm:
  - without momentum: $w_{t+1} = w_t - \lambda \frac{\partial L}{\partial w_t}$
  - with momentum:

$$v_{t+1} = \beta v_t + \frac{\partial L}{\partial w_t}$$
$$w_{t+1} = w_t - \lambda v_{t+1}$$

or $\Delta w_{t+1} = -\lambda \sum_{k=1}^{t} (\beta^{t-k} \nabla w_k)$

where β is the hyperparameter that controls the 'amount' of history (momentum constant)

Discussion Questions

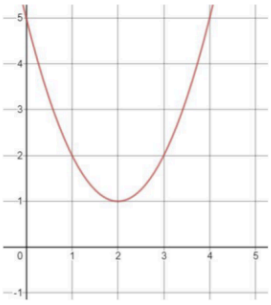
1. k-fold cross-validation
   1. Compare (A)) 100-Folds (B) 5 × 20 repeated K-Fold validation, over the parameter grid of C = [0.1, 1, 10, 100]
      1. Do the two methods fit the model for the same number of times?
      2. For a dataset of 100 datapoints. Do the two methods have the same size of the training dataset?
      3. bias and variance of the fit of the model?
      4. bias and variance of the estimate of test error?
2. Gradient Descent

Given the function $f(w)$:

$$f(w) = (w-2)^2 + 1 \tag{1}$$

1. Compute the gradient of $f(w)$ with respect to w.
2. State the weight update rule used in gradient descent.
3. Starting with initial weight $w_0 = 0$, run gradient descent for the following function for 5 iterations. Use learning rate $\alpha = 3,\ 0.2,\ 0.01$.
4. Compare and contrast $\frac{\partial f(w)}{\partial w}$ after 5 iterations using the three different learning rates. What can you infer from these values? Which learning rate should we select as the best one?

Gradient Descent Example #1



$f(w) = (w-2)^2 + 1$

$d(f(w)) / dw = 2w - 4$

$W_{n+1} = W_n - \alpha\ (df(w) / dw)$

$W_0 = 0$

	Approach 1 : α = 3		Approach 2 : α = 0.2		Approach 3 : α = 0.01	
	w	**d(f(w)) / dw**	**w**	**d(f(w)) / dw**	**w**	**d(f(w)) / dw**
Iteration 0	0	-4	0	-4	0	-4
Iteration 1	12	20	0.8	-2.4	0.04	-3.92
Iteration 2	-48	-100	1.28	-1.44	0.079	-3.84
Iteration 3	252	500	1.568	-0.864	0.118	-3.76
Iteration 4	-1248	-2500	1.7408	-0.5184	0.155	-3.69
	Will not Converge		About Right		Take too long	

3. SGD v.s. BGD v.s. mini-BGD
    1. If I want perfect repeatability of the descent path given the same initial point, which GD method should I use? Why?
    2. If I want the best memory efficiency during model training, which GD method should I use? Why?