Homework 5: SGD for Multiclass Linear SVM

Due: Wednesday, April 5, 2023 at 11:59PM EST

Instructions: Your answers to the questions below, including plots and mathematical work, should be submitted as a single PDF file. It's preferred that you write your answers using software that typesets mathematics (e.g.LaTeX, LyX, or MathJax via iPython), though if you need to you may scan handwritten work. You may find the minted package convenient for including source code in your LaTeX document. If you are using LyX, then the listings package tends to work better.

---

1 Bayesian Modeling

Bayesian Logistic Regression with Gaussian Priors

This question analyzes logistic regression in the Bayesian setting, where we introduce a prior $p(w)$ on $w \in \mathbb{R}^d$. Consider a binary classification setting with input space $\mathcal{X} = \mathbb{R}^d$, outcome space $\mathcal{Y}_\pm = \{-1, 1\}$, and a dataset $\mathcal{D} = \big((x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\big)$.

1. Give an expression for the posterior density $p(w \mid \mathcal{D})$ in terms of the negative log-likelihood function $\text{NLL}_\mathcal{D}(\text{w})$ and the prior density $p(w)$ (up to a proportionality constant is fine).

   - recall that the posterior distribution of our parameter w given out data set $\mathcal{D}$ is given by
   $$P(w|\mathcal{D}) \propto P(\mathcal{D}|w)P(w)$$
   - so now we want to think about $P(\mathcal{D}|w) = \mathcal{L}_\mathcal{D}(w) = P(y_1...y_n|w, x_1..x_n)$
   - then under the assumptions of logistic regression we can write this as $P(\mathcal{D}|w) = \mathcal{L}_\mathcal{D}(w) = P(y_1...y_n|w, x_1..x_n) = \Pi_{i=1}^n P(y^i|w, x_i)$
   - then we can take the log of this to get $\ell(w) = log(\mathcal{L}_\mathcal{D}(w)) = \frac{1}{2}\Sigma_{i=1}^n (y^i + 1)P(w = 1|x^i, w) + (y^i - 1)P(w = -1|x^i, w) = \frac{1}{2}\Sigma_{i=1}^n (y^i + 1)log(\frac{1}{1+e^{-w^t x^i}}) + (y^i - 1)log(1 - \frac{1}{1+e^{-w^t x^i}})$
   - so turning back to the question we can write
   $$P(w|\mathcal{D}) \propto P(\mathcal{D}|w)P(w) \propto e^{\frac{-1}{2}\Sigma_{i=1}^n (y^i+1)log(\frac{1}{1+e^{-w^t x^i}})+(y^i-1)log(1-\frac{1}{1+e^{-w^t x^i}})} p(w)$$
   which is the posterior expressed in terms of of the negative log likelihood function and posterior distribution
   - alternativly we could write it in more general terms as
   $$P(w|\mathcal{D}) \propto P(\mathcal{D}|w)P(w) = \mathcal{L}_\mathcal{D}(w)P(w) \propto e^{-log(\mathcal{L}_\mathcal{D}(w))}P(w)$$

2. Suppose we take a prior on w of the form $w \sim \mathcal{N}(0, \Sigma)$, that is in the Gaussian family. Is this a conjugate prior to the likelihood given by logistic regression?

   - a conjugate pair means that both the posterior and prior are from the same distribution family

- we know our prior is Gaussian such that $w \sim \mathcal{N}(0, \Sigma)$, we know the pdf of a Gaussian random vector is given by $f_W(x) = (2\pi)^{\frac{-k}{2}} det(\Sigma)^{\frac{-1}{2}} e^{-\frac{1}{2}(x-\mu)^t \Sigma^{-1}(x-\mu)}$ and given the information we have about our prior this becomes $f_W(x) = (2\pi)^{\frac{-k}{2}} det(\Sigma)^{\frac{-1}{2}} e^{-\frac{1}{2}(x-\mu)^t \Sigma^{-1}(x-\mu)} = (2\pi)^{\frac{-k}{2}} det(\Sigma)^{\frac{-1}{2}} e^{-\frac{1}{2}x^t \Sigma^{-1} x}$
- in this case we are only interested in out posterior up to a point of proportionality so we an ignore terms that do not relay on w so we can write our posterior distribution as $P(w|\mathcal{D}) \propto P(\mathcal{D}|w)P(w)P(w|\mathcal{D}) \propto e^{-log(\mathcal{L}_\mathcal{D}(w))} P(w) \propto e^{-log(\mathcal{L}_\mathcal{D}(w))} e^{-\frac{1}{2}w^t \Sigma^{-1} w}$ $= e^{-(log(\mathcal{D}|w) + \frac{1}{2}w^t \Sigma^{-1} w)}$
- so our pdf would be given by a logistic times the Gaussian pdf, and thus i don't think it would be Gaussian. so in other words no i don't believe it is a conjugate prior

3. Show that there exist a covariance matrix Σ such that MAP (maximum a posteriori) estimate for w after observing data $\mathcal{D}$ is the same as the minimizer of the regularized logistic regression function defined in Regularized Logistic Regression paragraph above, and give its value. [Hint: Consider minimizing the negative log posterior of w. Also, remember you can drop any terms from the objective function that don't depend on w. You may freely use results of previous problems.]

- as we showed in the last question our posterior can be written as $P(w|\mathcal{D}) \propto e^{-(log(\mathcal{D}|w) + \frac{1}{2}w^t \Sigma^{-1} w)}$
- note that the map estimate $\hat{w} \in argmax_w P(w|d)$ will not be effected by monotonic transformations
- thus we can write the negative log posterior as $log(P(w|\mathcal{D})) \propto (log(\mathcal{D}|w) + \frac{1}{2}w^t \Sigma w)$ and still achieve the same map estimator.
- so thus our map estimator will be $\hat{w} = \arg\max_w P(w|\mathcal{D}) \propto argmax_w(log(\mathcal{L}_\mathcal{D}(w)) + \frac{1}{2}w^t \Sigma^{-1} w)$
- we saw in the last homework $\hat{w} = argmax_w \in log(\mathcal{L}_\mathcal{D}(w)) = argmax_w \Sigma_{i=1}^n (y^i + 1)log(f(w^t x_i)) + (y^i - 1)log(1 - f(w^t x_i)) = \Sigma_{i=1}^n [\frac{y^i+1}{2} - f(w^t x^i)]x^i$
- so the optimal w for our regularized logistic expression can be written as $\hat{w} = argmax_w \in log(\mathcal{L}_\mathcal{D}(w)) = argmax_w \Sigma_{i=1}^n (y^i+1)log(f(w^t x_i)) + (y^i-1)log(1-f(w^t x_i)) + \lambda||w|| = \Sigma_{i=1}^n [\frac{y^i+1}{2} - f(w^t x^i)]x^i + \lambda w$
- so turning back to the map of the negative log of our posterior distribution we can write the problem as $\hat{w} = \arg\max_w P(w|\mathcal{D}) \propto argmax_w(log(\mathcal{L}_\mathcal{D}(w)) + \frac{1}{2}w^t \Sigma^{-1} w) = \Sigma_{i=1}^n (y^i + 1)log(f(w^t x_i)) + (y^i - 1)log(1 - f(w^t x_i)) + \frac{1}{2}w^t \Sigma^{-1} w$
- the gradient of our posterior can be written as $\nabla_{P(w|\mathcal{D})}(w) = \Sigma_{i=1}^n [\frac{y^i+1}{2} - f(w^t x^i)]x^i + \Sigma^{-1} w$
- so it is clear if we set $\Sigma = I\lambda \Rightarrow \Sigma^{-1} w = w$ and thus we will achive the same argmax $\hat{w}$ in both cases

4. In the Bayesian approach, the prior should reflect your beliefs about the parameters before seeing the data and, in particular, should be independent on the eventual size of your dataset. Imagine choosing a prior distribution $w \sim \mathcal{N}(0, I)$. For a dataset $\mathcal{D}$ of size n, how should you choose λ in our regularized logistic regression objective function so that the ERM is equal to the mode of the posterior distribution of w (i.e. is equal to the MAP estimator).

- we know from last question the mode of posterior that is the map estimator is given by $\hat{w} = \Sigma_{i=1}^{n}[\frac{y^i+1}{2} - f(w^t x^i)]x^i + \Sigma^{-1}w = \Sigma_{i=1}^{n}[\frac{y^i+1}{2} - f(w^t x^i)]x^i + Iw = \Sigma_{i=1}^{n}[\frac{y^i+1}{2} - f(w^t x^i)]x^i + w$
- we know from the last home work that our regularized erm can be expressed as $\hat{w} = argmax_w \in log(\mathcal{L}_{\mathcal{D}}(w)) = argmax_w \Sigma_{i=1}^{n}(y^i+1)log(f(w^t x_i)) + (y^i-1)log(1 - f(w^t x_i)) + \lambda||w|| = \Sigma_{i=1}^{n}[\frac{y^i+1}{2} - f(w^t x^i)]x^i + \lambda w$
- so these two expressions are equalized when $\lambda = 1$

Coin Flipping with Partial Observability

This is continuing your analysis done in HW4, you may use the results you obtained in HW4.

Consider flipping a biased coin where $p(z = H \mid \theta_1) = \theta_1$. However, we cannot directly observe the result z. Instead, someone reports the result to us, which we denote by x. Further, there is a chance that the result is reported incorrectly *if it's a head*. Specifically, we have $p(x = H \mid z = H, \theta_2) = \theta_2$ and $p(x = T \mid z = T) = 1$.

5. We additionally obtained a set of clean results $\mathcal{D}_c$ of size N_c, where x is directly observed without the reporter in the middle. Given that there are c_h heads and c_t tails, estimate θ_1 and θ_2 by MLE taking the two data sets into account. Note that the likelihood is $L(\theta_1, \theta_2) = p(\mathcal{D}_r, \mathcal{D}_c \mid \theta_1, \theta_2)$.

   - first things first we need to derive our likelihood expression for $\mathcal{D}_e$
     - we are told that our observations in $\mathcal{D}_e$ are the result with out any intermediary, thus we can say that $\forall x_i \in \mathcal{D}_e = P(x_i = H|x = H, \theta_1, \theta_2) = 1 = P(x_i = H|x = H, \theta_1, \theta_2)$
     - with this in mind we can write $\mathcal{L}_{\mathcal{D}_e}(\theta_1, \theta_2) = \Pi_{i=1}^{n} P(x_i|\theta_1, \theta_2) = P(x_i = h|\theta_1, \theta_2)^{c_h} P(x_i = t|\theta_1, \theta_2)^{c_t} = (P(x_i = h|z_i = h\theta_1, \theta_2)P(z_i = h|\theta_1, \theta_2) + P(x_i = h|z_i = T, \theta_1, \theta_2)P(z_i = T|\theta_1, \theta_2)^{c_h} (P(x_i = t|, z_i = h, \theta_1, \theta_2)P(z_i = h|\theta_1, \theta_2) + P(x_i = t|, z_i = t, \theta_1, \theta_2)P(z_i = t|\theta_1, \theta_2))^{c_t} = P(z_i = H|\theta_1, \theta_2)^{c_h} P(z_i = T|\theta_1, \theta_2)^{c_t} = P(z_i = H|\theta_1)^{c_h} P(z_i = T|\theta_1)^{c_t} = \theta_1^{c_h}(1 - \theta_1)^{c_t}$
   - now we can derive our joint likelihood of both data sets that is $\mathcal{L}_{\mathcal{D}_e, \mathcal{D}_c}(\theta_1, \theta_2)$
     - given both data sets are sampled iid we can say $\mathcal{L}_{\mathcal{D}_e, \mathcal{D}_c}(\theta_1, \theta_2) = \mathcal{L}_{\mathcal{D}_e}(\theta_1, \theta_2) * \mathcal{L}_{\mathcal{D}_c}(\theta_1, \theta_2)$
     - then using what we computed above as well as question 10 from homework 4 we can write $\mathcal{L}_{\mathcal{D}_e, \mathcal{D}_c}(\theta_1, \theta_2) = \mathcal{L}_{\mathcal{D}_e}(\theta_1, \theta_2) * \mathcal{L}_{\mathcal{D}_c}(\theta_1, \theta_2) = \theta_1^{c_h}(1 - \theta_1)^{c_t}(\theta_1\theta_2)^{n_h}(1 - \theta_1\theta_2)^{n_t}$
     - then to make the optimization easier and more stable we can write the log likelihood as

       $$\ell(\theta_1, \theta_2) = log(\mathcal{L}_{\mathcal{D}_e \mathcal{D}_c}(\theta_1, \theta_2) = c_h log(\theta_1) + c_t log(1-\theta_1) + n_h log(\theta_1\theta_2) + n_t log(1-\theta_1\theta_2)$$

   - now we can being our mle which can be expressed as

     $$\hat{\theta} = argmax_{\theta \in \mathbb{R}^2} \ell(\theta_1, \theta_2)$$

   - so first lets find the optimal value of θ_2

  - given the log likelihood function $\ell(\theta_1, \theta_2) = c_h log(\theta_1) + c_t log(1-\theta_1) + n_h log(\theta_1\theta_2) + n_t log(1-\theta_1\theta_2)$
  - we can calculate the partial with respect to θ_2 as $\frac{\partial \ell}{\partial \theta_2} = \frac{nh}{\theta_1\theta_2} - \frac{n_t}{1-\theta_1\theta_2}$
  - we know optima will be when the partial derivative equals zero that is
    $\frac{nh}{\theta_1\theta_2} - \frac{n_t}{1-\theta_1\theta_2} = 0$
    $n_h(1-\theta_1\theta_2) - n_t(\theta_1\theta_2) = 0$
    $n_h - \theta_1\theta_2 n_h = n_t\theta_1\theta_2$
    $n_h = (nh_+ n_t)\theta_1\theta_2$
    $\theta_2^* = \frac{n_h}{(n_h+n_t)\theta_1}$
- now we want to find the optimal value of θ_1
  - we know the optima will have $\theta_2 = \theta_2^* = \frac{n_h}{(n_h+n_t)\theta_1}$
  - so we can now think of our likelihood function as 1 dimensional optimization problem $\ell(\theta_1, \theta_2^*) = \ell(\theta_1) = c_h log(\theta_1) + c_t log(1-\theta_1) + n_h log(\theta_1\theta_2^*) + n_t log(1-\theta_1\theta_2^*) = c_h log(\theta_1) + c_t log(1-\theta_1) + n_h log(\theta_1 \frac{n_h}{(n_h+n_t)\theta_1}) + n_t log(1-\theta_1 \frac{n_h}{(n_h+n_t)\theta_1}) = c_h log(\theta_1) + c_t log(1-\theta_1) + n_h log(\frac{n_h}{(n_h+n_t)}) + n_t log(1-\frac{n_h}{(n_h+n_t)})$
  - now we can take the partial derivative of this function with respect to θ_1 as $\frac{\ell(\theta_1)}{\partial\theta_1} = \frac{c_h}{\theta_1} - \frac{c_t}{1-\theta_1}$
  - setting this equal to zero yields
    $\frac{c_h}{\theta_1} = \frac{c_t}{1-\theta_1}$
    $c_h - \theta_1 c_h = \theta_1 c_t$
    $c_h = \theta_1(c_h + c_t)$
    $\theta_1^* = \frac{c_h}{c_h+c_t}$
- this overall makes sense as we are more or less saying that our mle estimator of θ in each case is just the number of heads we observe in each data set.

6. Since the clean results are expensive, we only have a small number of those and we are worried that we may overfit the data. To mitigate overfitting we can use a prior distribution on θ_1 if available. Let's imagine that an oracle gave use the prior $p(\theta_1) = \text{Beta}(h, t)$. Derive the MAP estimates for θ_1 and θ_2.

   - first lets reason about our prior
     - as we showed in class the pdf of beta distributed prior with parameters α, β can be written as $\theta \sim beta(\alpha, \beta) \Rightarrow P(\theta) \propto (\theta)^{\alpha-1}(1-\theta)^{\beta-1}$
     - so given what we are told in the problem we know our prior of θ_1 is $\theta_1 \sim beta(\text{h }, \text{t}) \Rightarrow P(\theta_1) \propto (\theta_1)^{h-1}(1-\theta_1)^{t-1}$
   - next we want to derive our posterior distribution (or something proportional to it)
     - we know our posterior distribution of the parameters θ_1, θ_2 given the observed data $\mathcal{D}_r, \mathcal{D}_c$ can be written as $P(\theta_1, \theta_2 | \mathcal{D}_r, \mathcal{D}_c) \propto P(\mathcal{D}_R, \mathcal{D}_c | \theta_1, \theta_2) P(\theta_1) P(\theta_2)$
     - then using what we derived in the last section and above we can write $P(\theta_1, \theta_2 | \mathcal{D}_r, \mathcal{D}_c) \propto P(\mathcal{D}_R, \mathcal{D}_c | \theta_1, \theta_2) P(\theta_1) P(\theta_2) = \theta_1^{c_h}(1-\theta_1)^{c_t}(\theta_1\theta_2)^{n_h}(1-\theta_1\theta_2)^{n_t}(\theta_1)^{h-1}(1-\theta_1)^{t-1}P(\theta_2) = (\theta_1)^{h-1+c_h}(1-\theta_1)^{t-1+c_t}(\theta_1\theta_2)^{n_h}(1-\theta_1\theta_2)^{nt}P(\theta_2)$
     - we make no assumptions about the prior of θ_2 so i think we can ignore $P(\theta_2)$ and just write $P(\theta_1, \theta_2 | \mathcal{D}_r, \mathcal{D}_c) \propto (\theta_1)^{h-1+c_h}(1-\theta_1)^{t-1+c_t}(\theta_1\theta_2)^{n_h}(1-\theta_1\theta_2)^{nt}P(\theta_2) \propto (\theta_1)^{h-1+c_h}(1-\theta_1)^{t-1+c_t}(\theta_1\theta_2)^{n_h}(1-\theta_1\theta_2)^{nt}$

  - then as we know the log is a monotonic function we can take the log of our posterior and still achieve the same optimal values of θ while making our computation more simple and stable $log(P(\theta_1, \theta_2|\mathcal{D}_r, \mathcal{D}_c)) \propto log((\theta_1)^{h-1+c_h}(1-\theta_1)^{t-1+c_t}(\theta_1\theta_2)^{n_h}(1-\theta_1\theta_2)^{nt}P(\theta_2) \propto (\theta_1)^{h-1+c_h}(1-\theta_1)^{t-1+c_t}(\theta_1\theta_2)^{n_h}(1-\theta_1\theta_2)^{nt}) = (h-1+c_h)log(\theta_1)+(t-1+c_t)log(1-\theta_1)+n_hlog(\theta_1\theta_2)+n_t(\theta_1\theta_2)$

- we know our map estimator is given by
  $\hat{\theta} = argmax_{\theta_1,\theta_2}P(\theta_1, \theta_2|\mathcal{D}_r, \mathcal{D}_e)$
  $= argmax_{\theta_1,\theta_2}log(P(\theta_1, \theta_2|\mathcal{D}_r, \mathcal{D}_e))$
  $= argmax_{\theta_1,\theta_2}(h-1+c_h)log(\theta_1)+(t-1+c_t)log(1-\theta_1)+n_hlog(\theta_1\theta_2)+n_t(\theta_1\theta_2)$
- so call $j(\theta_1, \theta_2)$the function we are looking to optimize that is $j(_1, \theta_2) = argmax_{\theta_1,\theta_2}(h-1+c_h)log(\theta_1)+(t-1+c_t)log(1-\theta_1)+n_hlog(\theta_1\theta_2)+n_t(\theta_1\theta_2)$
- so first we can find the optimal value of θ_2
  - so we first want to take the partial derivative of j with respect to θ_2 that is $\frac{\partial j(\theta_1,\theta_2)}{\theta_2} = \frac{nh}{\theta_1\theta_2} - \frac{n_t}{1-\theta_1\theta_2}$
  - we know optima will be when the partial derivative equals zero that is
    $\frac{nh}{\theta_1\theta_2} - \frac{n_t}{1-\theta_1\theta_2} = 0$
    $n_h(1-\theta_1\theta_2) - n_t(\theta_1\theta_2) = 0$
    $n_h - \theta_1\theta_2 n_h = n_t\theta_1\theta_2$
    $n_h = (nh_+n_t)\theta_1\theta_2$
    $\theta_2^* = \frac{n_h}{(n_h+n_t)\theta_1}$
- next we want to find the optimal value of θ_1
  - we know our optimal tuple (θ_1^*, θ_2^*) must have $\theta_2 = \theta_2^* = \frac{n_h}{(n_h+n_t)\theta_1}$
  - thus we can express $j(\theta_1, \theta_2)$ as a 1 dimensional function of θ_1
  - that is $j(\theta_1, \theta_2^*) = j(\theta_1) = (h-1+c_h)log(\theta_1)+(t-1+c_t)log(1-\theta_1)+n_hlog(\theta_1\theta_2^*)+n_t(\theta_1\theta_2^*) = (h-1+c_h)log(\theta_1)+(t-1+c_t)log(1-\theta_1)+n_hlog(\theta_1\frac{n_h}{(n_h+n_t)\theta_1})+n_t(\theta_1\frac{n_h}{(n_h+n_t)\theta_1}) = (h-1+c_h)log(\theta_1)+(t-1+c_t)log(1-\theta_1)+n_hlog(\frac{n_h}{(n_h+n_t)})+n_t(\frac{n_h}{(n_h+n_t)})$
  - now we want to take the partial derivative of this function with respect to θ_1 that is $\frac{\partial J(\theta_1)}{\partial\theta_1} = \frac{h-1+c_h}{\theta_1} - \frac{t-1+c_t}{1-\theta_1}$
  - setting our partial derivative equal to zero we see that
    $\frac{h-1+c_h}{\theta_1} - \frac{t-1+c_t}{1-\theta_1} = 0$
    $\frac{h-1+c_h}{\theta_1} = \frac{t-1+c_t}{1-\theta_1}$
    $(h-1+c_h)(1-\theta_1) = (t-1+c_t)(\theta_1)$
    $(h-1+c_h) - \theta_1(h-1+c_h) = (t-1+c_t)\theta_1$
    $(h-1+c_h) = (t-1+c_t)\theta_1 + \theta_1(h-1+c_h)$
    $(h-1+c_h) = \theta_1(t+h-2+ch+c_t)$
    $\theta_1^* = \frac{h-1+c_h}{t+h+2+c_{h\,c\,t}}$

2 Derivation for multi-class modeling

Suppose our output space and our action space are given as follows: $\mathcal{Y} = \mathcal{A} = \{1, \ldots, k\}$. Given a non-negative class-sensitive loss function $\Delta : \mathcal{Y} \times \mathcal{A} \to [0, \infty)$ and a class-sensitive feature mapping $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$. Our prediction function $f : \mathcal{X} \to \mathcal{Y}$ is given by

$$f_w(x) = \arg\max_{y \in \mathcal{Y}} \langle w, \Psi(x, y) \rangle .$$

For training data $(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$, let $J(w)$ be the ℓ_2-regularized empirical risk function for the multiclass hinge loss. We can write this as

$$J(w) = \lambda \|w\|^2 + \frac{1}{n} \sum_{i=1}^{n} \max_{y \in \mathcal{Y}} \left[\Delta\left(y_i, y\right) + \langle w, \Psi(x_i, y) - \Psi(x_i, y_i)\rangle\right]$$

for some $\lambda > 0$.

7. Show that $J(w)$ is a convex function of w. You may use any of the rules about convex functions described in our notes on Convex Optimization, in previous assignments, or in the Boyd and Vandenberghe book, though you should cite the general facts you are using. [Hint: If $f_1, \ldots, f_m : \mathbb{R}^n \to \mathbb{R}$ are convex, then their pointwise maximum $f(x) = \max\{f_1(x), \ldots, f_m(x)\}$ is also convex.]

8. Since $J(w)$ is convex, it has a subgradient at every point. Give an expression for a subgradient of $J(w)$. You may use any standard results about subgradients, including the result from an earlier homework about subgradients of the pointwise maxima of functions. (Hint: It may be helpful to refer to $\hat{y}_i = \arg\max_{y \in \mathcal{Y}} \left[\Delta\left(y_i, y\right) + \langle w, \Psi(x_i, y) - \Psi(x_i, y_i)\rangle\right]$.)

9. Give an expression for the stochastic subgradient based on the point (x_i, y_i).

10. Give an expression for a minibatch subgradient, based on the points $(x_i, y_i), \ldots, (x_{i+m-1}, y_{i+m-1})$.

(Optional) Hinge Loss is a Special Case of Generalized Hinge Loss

Let $\mathcal{Y} = \{-1, 1\}$. Let $\Delta(y, \hat{y}) = \mathbb{1}y \neq \hat{y}$. If $g(x)$ is the score function in our binary classification setting, then define our compatibility function as

$$\begin{aligned} h(x, 1) &= g(x)/2 \\ h(x, -1) &= -g(x)/2. \end{aligned}$$

11. Show that for this choice of h, the multiclass hinge loss reduces to hinge loss:

$$\ell\left(h, (x, y)\right) = \max_{y' \in \mathcal{Y}} \left[\Delta\left(y, y'\right)\right) + h(x, y') - h(x, y)] = \max\{0, 1 - yg(x)\}$$

3 Implementation

In this problem we will work on a simple three-class classification example. The data is generated and plotted for you in the skeleton code.

One-vs-All (also known as One-vs-Rest)

First we will implement one-vs-all multiclass classification. Our approach will assume we have a binary base classifier that returns a score, and we will predict the class that has the highest score.

12. Complete the methods `fit`, `decision_function` and `predict` from `OneVsAllClassifier` in the skeleton code. Following the `OneVsAllClassifier` code is a cell that extracts the results of the fit and plots the decision region. You can have a look at it first to make sure you understand how the class will be used.

13. Include the results of the test cell in your submission.

Multiclass SVM

In this question, we will implement stochastic subgradient descent for the linear multiclass SVM, as described in class and in this problem set. We will use the class-sensitive feature mapping approach with the "multivector construction", as described in the multiclass lecture.

14. Complete the function `featureMap` in the skeleton code.
15. Complete the function `sgd`.
16. Complete the methods `subgradient`, `decision_function` and `predict` from the class `MulticlassSVM`.
17. Following the multiclass SVM implementation, we have included another block of test code. Make sure to include the results from these tests in your assignment, along with your code.