CPSC 340 Assignment 4 (due Friday, March 9 at 9:00pm)

1 Convex Functions

1. $f'(w) = 2\alpha w - \beta$, $f''(w) = 2\alpha$. $\alpha \geq 0$, and thus $f''(w) \geq 0$. As such, $f(w)$ is convex.
2. $f'(w) = \log(w) + 1$, $f''(w) = \frac{1}{w}$. Since $w > 0$, $f''(w) > 0$. As such, $f(w)$ is convex.
3. Since squared norms are convex, $\|Xw - y\|^2$ is convex. Since norms are convex and it is multiplied by a non-negative $\lambda, \lambda\|w\|_1$ is convex. $f(w)$ is thus convex because it is a sum of convex functions.
4. Replacing $-y_i w^T x_i$ with z_i, we get $f(w) = \sum_{i=1}^{n} \log(1 + \exp(z_i))$. Since $f(w)$ is a summation and the sum of convex functions is also convex, it would be sufficient to prove that $g(w) = \log(1 + \exp(z))$ is convex. $g'(w) = \frac{\exp(z)}{1+\exp(z)} = \frac{1}{1+\exp(-z)}$. $g''(w) = \frac{\exp(w)}{(1+\exp(w))^2}$. Since both the numerator and denominator of $g''(w)$ is positive, $g''(w)$ is positive and thus $g(w)$ is convex. Since the composition of a convex function and the linear function z is convex, f(w) is also convex.
5. $f(w, w_0)$ involves two terms $\sum_{i=1}^{N}[\max\{0, w_0 - w^T x_i\} - w_0]$ and $\frac{\lambda}{2}\|w\|_2^2$. Looking at the summation term, there are two terms in it, $\max\{0, w_0 - w^T x_i\}$ and w_0. Looking at the max function, the second differential of 0 is just zero, and the second differential of $w_0 - w^T x_i$ is also zero. Hence, both terms are convex as its second differential is greater than or equal to zero. Since the max of two convex functions is also convex, $\max\{0, w_0 - w^T x_i\}$ is also convex. For the w_0 term, f‘$(w) = 0$ and f “$(w) = 0$. Hence, w_0 is also convex. Thus, the whole term $[\max\{0, w_0 - w^T x_i\} - w_0]$ is convex, and the summation of the terms will also result in a convex function. Since squared norms are convex and $\lambda \geq 0$ is non-negative, $\frac{\lambda}{2}\|w\|_2^2$ is also convex. Hence, the summation of $\sum_{i=1}^{N}[\max\{0, w_0 - w^T x_i\} - w_0]$ and $\frac{\lambda}{2}\|w\|_2^2$ would result in a convex function.

2 Logistic Regression with Sparse Regularization

2.1 L2-Regularization

logRegL2 is implemented in *linear_model.py*. Training error increases from 0 to 0.002. Validation error decreases from 0.084 to 0.074. Number of features used remains the same at 101. Number of iterations decreased from 121 to 36.

2.2 L1-Regularization

logRegL1 is implemented in *linear_model.py*. Training error remains the same at 0. Validation error decreases from 0.084 to 0.052. Number of features used decreases from 101 to 71. Number of iterations decreased from 121 to 78.

2.3 L0-Regularization

logRegL0 is implemented in *linear_model.py*. Training error is 0.112. Validation error is 0.114. Number of features selected is 12.

2.4 Discussion

For the same value of $\lambda = 1$, L0-regularization has the highest training and validation error and the lowest number of features selected, followed by L2-regularization and finally L1-regularization. This is possibly because L0-regularization regularizes the number of relevant features directly, and thus tends to prevent overfitting the model more as compared to L1 and L2-regularization. Furthermore, L2-regularization does not have feature selection and thus all features are retained.

2.5 Comparison with scikit-learn

Implemented in *main.py*. With scikit-learn, L2-regularization training error is 0.002, validation error is 0.074 and number of features selected is 101. This is exactly the same result as we achieved with our implementation.

L1-regularization training error is 0, validation error is 0.052 and number of features selected is 71. This is exactly the same result as achieved with our implementation as well.

3 Multi-Class Logistic

3.1 Softmax Classification, toy example

With the given $\hat{x}$ and W, we obtain $\hat{y} = \begin{bmatrix} +1 \\ +4 \\ +2 \end{bmatrix}$. Since $\hat{y_2}$ has the largest value, we will assign the test example to class 2.

3.2 One-vs-all Logistic Regression

logLinearClassifier is implemented in *linear_model.py*. The validation error is 0.070 and the training error is 0.084.

3.3 Softmax Classifier Implementation

softmaxClassifier is implemented in *linear_model.py*. Validation error is 0.008.

3.4 Comparison with scikit-learn, again

For scikit-learn implementation, we get a training error of 0.000 and a validation error of 0.012. Our implementation of logistic regression using the One-vs-all method has much worse validation error of 0.07, but our implementation using the softmax classifier has a better validation error of 0.008.

3.5 Cost of Multinomial Logistic Regression

Rubric: {reasoning:2}

Assuming that we have

- n training examples.
- d features.
- k classes.
- t testing examples.
- T iterations of gradient descent for training.

1. $O(Tnkd)$. We run T iterations and calculate f and g with each iteration which costs $O(nkd)$.
2. $O(tdk)$, which is the cost of calculating the matrix multiplication between (t x d) and (d x k) matrices

4 Very-Short Answer Questions

Rubric: {reasoning:9}

1. We use a score BIC instead of validation error for feature selection when we want to minimize the complexity of our model. Using BIC, a new model have to reduce the training error by a certain amount to justify increasing its complexity.
2. Forward selection considers $O(d^2)$ models, whereas in search and score, there are 2^d possible sets. Forward selection is a cheaper process, overfits less, and has fewer false positives. In search and score, the process is costly, optimization bias is high and models are prone to false positives.
3. A lower λ results in less regularization, and thus the model tends to overfit. This leads to a lower training error but higher test error. However, with a higher λ, there is more regularization and simpler models are favoured. This leads to a higher training error but lower test error.
4. L1-regularization performs feature selection whereas L2-regularization does not. L1-regularization cannot be used in gradient-based approaches since it is not differentiable unlike L2-regularization.
5. The main problem is that least squares penalizes classifications that are "too right" as they are far from the actual value.
6. A classifier found from the perceptron algorithm finds some classifier with zero error, whereas the linear SVM finds the maximum-margin classifier.
7. Least squares, the perceptron algorithm, SVMs, and logistic regression all produce linear classifiers.
8. In multi-label classification, y is a n x k matrix, where in each row, the column values are 1 for correct labels and -1 for incorrect labels. There can be more than 1 correct class label. For multi-class classification, y is a vector of size n. Each row value is simply the label assigned to the example.
9. For one-vs-all multi-class logistic regression, we are solving k optimization problem(s) of dimension d. On the other hand, for softmax logistic regression, we are solving 1 optimization problem(s) of dimension c x d.