Lecture Notes for

Machine Learning in Python

⟦👨‍🏫, 👨‍💻, 🐍, 👨‍🔬⟧

Professor Eric Larson

Optimization Techniques for Logistic Regression

Class Logistics and Agenda

- Logistic: grading!
- Agenda
  - Finish Logistic Regression
  - Numerical Optimization Techniques
    - Types of Optimization
    - Programming the Optimization
- **Whirlwind Lecture Alert**
  - Get an intuition, program it, maybe you don’t follow every mathematical concept in lecture
  - But you know how to approach it outside lecture

Class Overview, by topic

Table Data Visualization
Numpy, Pandas, Seaborn
Overviews with some in-depth discussion

Dimension Reduction and Image Processing
Scikit-learn, Scikit Image,
Intuition only, Some mathematics

Linear and Logistic Regression
Numpy, Recreate API for Scikit-learn
Detailed mathematics for simple optimization
intuition for advanced optimization

Neural Networks and Back Prop.
Numpy
Detailed mathematics for NN operations

Wide and Deep Networks

Convolutional Networks

Recurrent Networks

Keras, Tensorflow
Intuition, Detailed implement.

Ethics in Language Models

ConceptNet
Case studies

Review

Objective Function: $l(\mathbf{w}) = \sum_i y^{(i)} \ln\left(g(\mathbf{w}^T\mathbf{x}^{(i)})\right) + (1 - y^{(i)})\ln\left(1 - g(\mathbf{w}^T\mathbf{x}^{(i)})\right)$

$$\underbrace{w_j}_{\text{new value}} \leftarrow \underbrace{w_j}_{\text{old value}} + \underbrace{\frac{\eta}{M}\sum_{i=1}^{M}(y^{(i)} - g(\mathbf{w}^T\mathbf{x}^{(i)}))x_j^{(i)}}_{\text{gradient}}$$

now $\mathbf{x}$, $\mathbf{w}$ are vectors

weighted sum of $\mathbf{x}$

$$\underbrace{\mathbf{w}}_{\text{new vect}} \leftarrow \underbrace{\mathbf{w}}_{\text{old vect}} + \frac{\eta}{M}\sum_{i=1}^{M}(y^{(i)} - g(\mathbf{w}^T\mathbf{x}^{(i)})) \cdot \mathbf{x}^{(i)}$$

$$\mathbf{y}_{diff} = \begin{bmatrix} y^{(1)} - g(\mathbf{w}^T\mathbf{x}^{(1)}) \\ y^{(2)} - g(\mathbf{w}^T\mathbf{x}^{(2)}) \\ \vdots \\ y^{(M)} - g(\mathbf{w}^T\mathbf{x}^{(M)}) \end{bmatrix} = \begin{bmatrix} y_{diff}^{(1)} \\ y_{diff}^{(2)} \\ \vdots \\ y_{diff}^{(M)} \end{bmatrix}$$

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \cdot \text{mean}(\mathbf{y}_{diff} \odot \mathbf{X})_{columns}$$

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \cdot \text{mean}\left(\begin{bmatrix} y_{diff}^{(1)} \\ y_{diff}^{(2)} \\ \vdots \\ y_{diff}^{(M)} \end{bmatrix} \odot \begin{bmatrix} \leftarrow & \mathbf{x}^{(1)} & \rightarrow \\ \leftarrow & \mathbf{x}^{(2)} & \rightarrow \\ & \vdots & \\ \leftarrow & \mathbf{x}^{(M)} & \rightarrow \end{bmatrix} \right)$$

mean across columns

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \cdot \text{mean}\left(\begin{bmatrix} (\leftarrow & \mathbf{x}^{(1)} & \rightarrow) \cdot y_{diff}^{(1)} \\ (\leftarrow & \mathbf{x}^{(2)} & \rightarrow) \cdot y_{diff}^{(2)} \\ & \vdots & \\ (\leftarrow & \mathbf{x}^{(M)} & \rightarrow) \cdot y_{diff}^{(M)} \end{bmatrix} \right)$$

mean across columns

```
05. Logistic Regression.ipynb
```

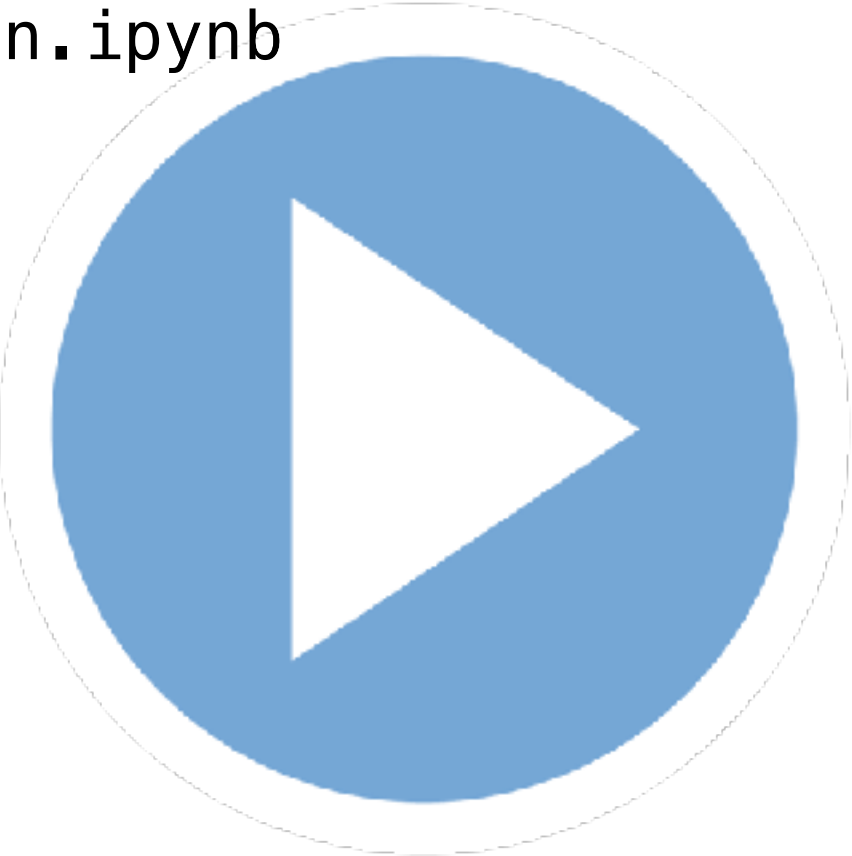
“Finish”

Programming

Vectorization

Regularization

Multi-class extension



Demo Lecture

`06. Optimization`

Line Optimization

Intuition for the Hessian

Lecture Notes for

Machine Learning in Python

⟦👨‍🏫, 👨‍💻, 🐍, 👨‍🔬⟧

Professor Eric Larson

Optimization Techniques for Logistic Regression Continued

Class Logistics and Agenda

- Agenda
  - Numerical Optimization Techniques
    - Types of Optimization
    - Programming the Optimization
- **Last Time:**
  - Logistic regression update equations
  - Line Searches
  - Stochastic small batches
  - Hessian-based methods

Class Overview, by topic

Table Data Visualization

Numpy, Pandas, Seaborn
Overviews with some in-depth discussion

Dimension Reduction and Image Processing

Scikit-learn, Scikit Image,
Intuition only, Some mathematics

Linear and Logistic Regression

Numpy, Recreate API for Scikit-learn
Detailed mathematics for simple optimization
intuition for advanced optimization

Neural Networks and Back Prop.

Numpy
Detailed mathematics for NN operations

Wide and Deep Networks

Convolutional Networks

Recurrent Networks

Keras, Tensorflow
Intuition, Detailed implement.

Ethics in Language Models

ConceptNet
Case studies

Demo Lecture, Continued

$$\mathbf{H}_{j,k}(\mathbf{w}) = \frac{\partial}{\partial w_k}\boxed{\frac{\partial}{\partial w_j} l(\mathbf{w})} \longrightarrow \frac{\partial}{\partial w_j} l(\mathbf{w}) = \sum_i \left(y^{(i)} - g(\mathbf{w}^T \cdot \mathbf{x}^{(i)})\right) x_j^{(i)}$$

$$\mathbf{H}_{j,k}(\mathbf{w}) = \frac{\partial}{\partial w_k} \sum_i \left(y^{(i)} - g(\mathbf{w}^T \cdot \mathbf{x}^{(i)})\right) x_j^{(i)}$$

$$= \cancel{\sum_i \frac{\partial}{\partial w_k} y^{(i)} x_j^{(i)}} - \sum_i \frac{\partial}{\partial w_k} g(\mathbf{w}^T \cdot \mathbf{x}^{(i)}) x_j^{(i)}$$

no dependence on w_k, zero

$$= -\sum_i x_j^{(i)} \boxed{\frac{\partial}{\partial w_k} g(\mathbf{w}^T \cdot \mathbf{x}^{(i)})}$$

already know this as $g(1-g)x_k$

$$\mathbf{H}_{j,k}(\mathbf{w}) = -\sum_{i=1}^{M} \left[g(\mathbf{w}^T\mathbf{x}^{(i)})[1 - g(\mathbf{w}^T\mathbf{x}^{(i)})]\right] \cdot x_k{}^{(i)} x_j{}^{(i)}$$

for each j,k pair

$$L_2 = C\sum_j w_j^2$$

```
penalty = '12'
```

$$L_1 = C\sum_j |w_j|$$

```
penalty = 'l1'
```

$$L_{12} = C_1\sum_j |w_j| + C_2\sum_j w_j^2$$

```
penalty = 'elasticnet'
```

Warning: The choice of the algorithm depends on the penalty chosen. Supported penalties by solver:

- 'lbfgs' - ['l2', None]
- 'liblinear' - ['l1', 'l2']
- 'newton-cg' - ['l2', None]
- 'newton-cholesky' - ['l2', None]
- 'sag' - ['l2', None]
- 'saga' - ['elasticnet', 'l1', 'l2', None]