1. domača naloga: Linearna regresija

Jernej Henigman (Kaggle: JernejHenigman)

4. april 2016

1 Uvod

V 3. domači nalogi je bilo potrebno implementirati multinomsko logistično regresijo (softmax regression).

2 Metode

- **cost(theta, X, y)**
  Implementiramo kriterijsko funkcijo :

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right]$$

Slika 1: Kriterijske funkcija

- **approx_grad(theta, X, y)**
  Podobno, kot v prejšnih dveh domačih nalogah tudi tu implementiramo aproksimacijo gradienta, z metodo končnih diferenc.

- **grad(theta, X, y)**
  Implementiramo gradient kriterijske funkcije:

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[x^{(i)}\left(1\{y^{(i)} = j\} - p(y^{(i)} = j | x^{(i)}; \theta)\right)\right]$$

Slika 2: Gradient kriterijske funkcije

- **fit(X, y)**
  S pomočjo L-BFGS vrnemo matriko parametrov modela Softmax.

- **predict(theta, X)**

Implementiramo funkcijo napovedi, ki za dane primere vrne matriko verjetnosti, s katero pripada primer ustreznemu razredu.

- **Regularizacija**
  Implementiramo vse metode podobno kot zgoraj, s tem dodatkom, da vključimo tudi regularizacijo.

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right] + \frac{\lambda}{2}\sum_{i=1}^{k}\sum_{j=0}^{n}\theta_{ij}^2$$

Slika 3: Kriterijska funkcija z regularizacijo

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[x^{(i)}(1\{y^{(i)} = j\} - p(y^{(i)} = j | x^{(i)}; \theta))\right] + \lambda\theta_j$$

Slika 4: Gradient kriterijske funkcije z regularizacijo

3 Rezultati

Najboljša stopnja regularizacije je podobna kot pri prejšni domači nalogi in se giblje med 0.01 in 0.10, odvisno od velikosti učne množice. Ko določimo najboljšo stopnjo regularizacije, poženemo softmax metodo, nad celo učno množico in oddamo rezultate na kagglu. Rezultati so nekoliko slabši, kot pri 2. domači nalogi in sicer dosežemo točnst 0.895.

V nalogi nisem implementiral 4. točke, zato algoritem deluje nekoliko počasneje