Multi-Class Classification

Tuesday 30 April 2024 9:00 PM

* No concept of threshold setting in multi-class classification

→ More than 2 classes

→ 2 options
- OVR (One Vs Rest) ① (not efficient with large datasets with high number of classes)
- Multinomial Logistic Regression (Softmax Regression) ②

① internal working of algorithm [Multi-class problem → Multiple Binary Classification problems]

training [each class gets its dedicated model]

→ K number of LogisticRegression models will be trained.
where K is the number of classes present in the target column.

1) Data is transformed by one hot encoding the target column.
2) Data is split up into K parts → we have converted a multi-class classification problem into K binary class classification problems.
3) Logistic Regression is applied on the K datasets independently
4) For each model we will obtain a corresponding $\beta_0, \beta_1, \beta_2$

prediction

1) test query point
2) send the test query point to each of the K models from which we will obtain K probabilities.
3) Normalize the output → $P(\text{class}_k) = \dfrac{P(\text{model}_k)}{\sum_{i=1}^{K} P(\text{model}_i)}$ sum of normalized probabilities is 1.
4) choose the class with the highest normalized probability.

② internal working of algorithm

→ SoftMax function

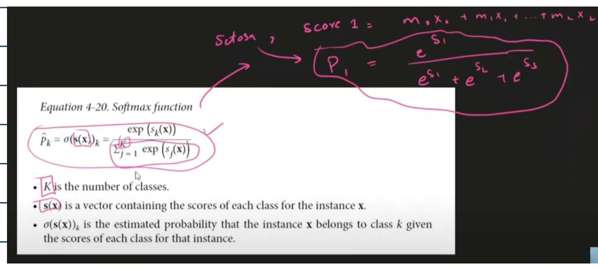
$$\boxed{\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}}$$

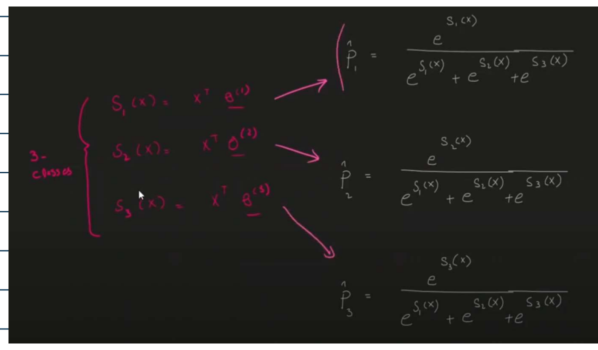
→ Provides a set of probabilities that sum up to one.

→ For K classes it will try to draw K lines to create the decision regions.

→ each line will have $(m+1)$ parameters, where m is the # of features.

→ in softmax regression, we will try to find $(m+1)$ parameters for each of the K lines. So total of $K(m+1)$ parameters.





training

1) Data is transformed by one hot encoding the target column.
2) Loss function $= -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} y_i^k \log \hat{y}_i^k$ ← minimize
   (categorical loss entropy)
   k represents the class [don't confuse with power]
   i for each row, k for each category
   → For each row we will obtain 1 term
3) $\hat{y}_1^{\text{class}} = \sigma(\vec{z}_i) = \dfrac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$
   ↳ row 1

   where $z_i = [1 \; x_{n1} \; x_{n2} \ldots x_{nm}] \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \end{bmatrix}$

row 1 $\quad j=1$

where $z_i = [1 \ x_{n1} \ x_{n2} \dots x_{nm}] \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{bmatrix}$

single record in data

corresponding line parameters to a particular class.

4) the Loss function depends on $\beta_{k \times (m+1)}$

Start with K random lines

Apply gradient descent [K(m+1) differentiations]

for each parameter of each line → for ___ epochs

$$\boxed{\beta_n^{k'} = \beta_n^k - \eta \frac{\partial L}{\partial \beta_n^k}}$$

line

parameter #

prediction

1) we have found out the β matrix for which loss is minimum

2) test query point

$$\beta = \begin{bmatrix} \beta_0^1 & \beta_1^1 & \dots & \beta_m^1 \\ \beta_0^2 & \beta_1^2 & \dots & \beta_m^2 \\ \vdots & \vdots & \dots & \vdots \\ \beta_0^K & \beta_1^K & \dots & \beta_m^K \end{bmatrix}_{K \times (m+1)} \qquad Q = [1 \ f_1 \ f_2 \ f_3 \dots f_m]_{1 \times (m+1)}$$

3) dot product β & Q

4) send each value to softmax function

5) choose class with highest probability

Usage

softmax regression
- Computationally efficiency
- Classes are mutually exclusive (not suitable for multi-label multi class classification problem)
- Interpretability is important

OVR
- Classes are non-mutually exclusive
- Dealing with imbalanced data

→ Deriving sigmoid from softmax
→ Deriving binary cross entropy from categorical cross entropy
→ Find derivative of softmax function
→ Find the gradients of cross entropy error.