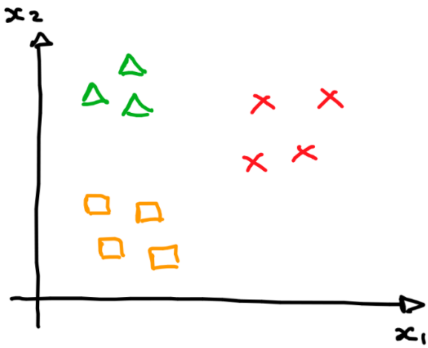
Multiclass logistic regression

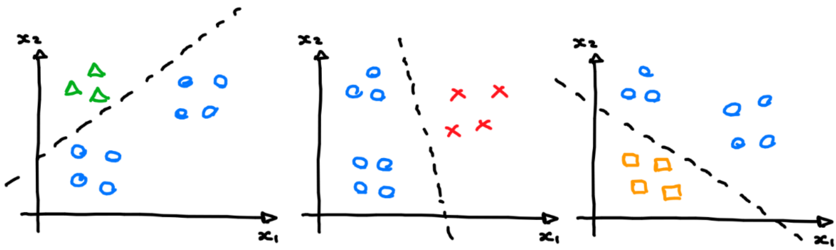
Herman Kamper

2024-01,

One-vs-rest classification



Strategy: Train three classifiers with $y \in \{0, 1)$ where each classifier considers one class as the positive class and the others as negative.



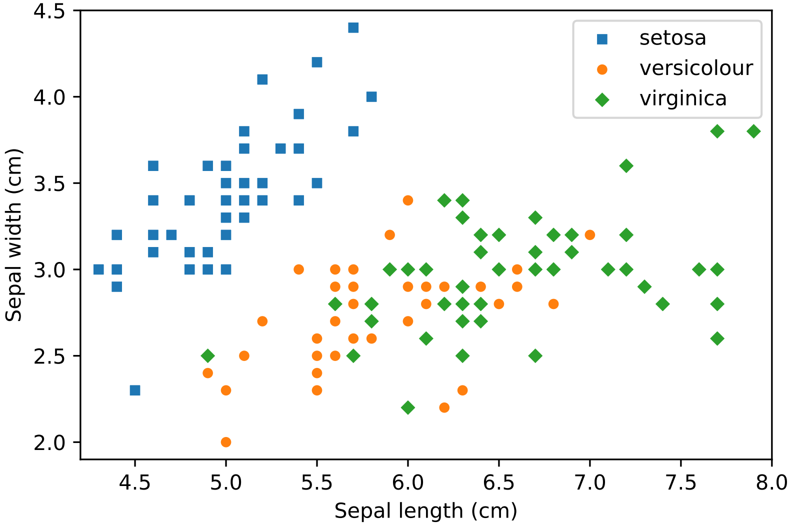
We then get three classifiers:

$$f_1(\mathbf{x}; \mathbf{w}_1)$$
$$f_2(\mathbf{x}; \mathbf{w}_2)$$
$$f_3(\mathbf{x}; \mathbf{w}_3)$$

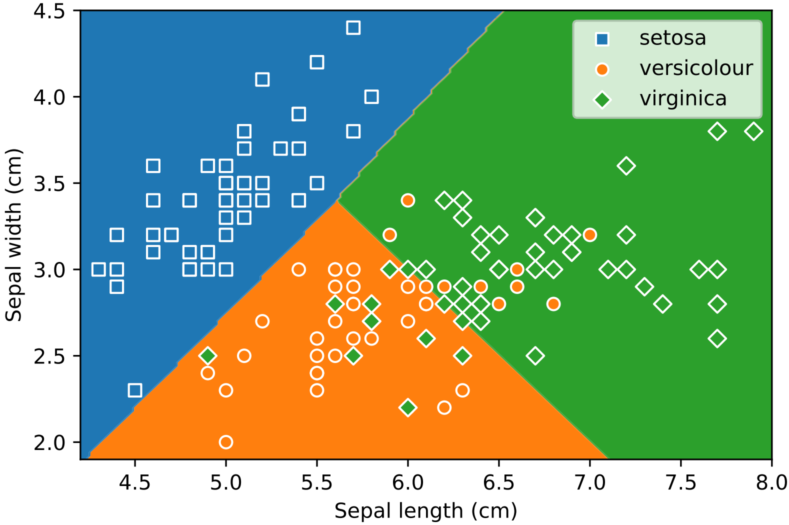
Final prediction:

$$\arg\max_k f_k(\mathbf{x}; \mathbf{w}_k)$$

One-vs-rest on iris dataset



One-vs-rest on iris dataset



Softmax regression

For binary logistic regression we had

$$f(\mathbf{x}; \mathbf{w}) = \sigma(\mathbf{w}^\top \mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}}}$$

with $y \in \{0, 1\}$.

We interpreted the output as $P(y = 1 | \mathbf{x}; \mathbf{w})$, implying $P(y = 0 | \mathbf{x}; \mathbf{w}) = 1 - f(\mathbf{x}; \mathbf{w})$.

For the multiclass setting we now have $y \in \{1, 2, \ldots, K\}$.

Idea: Instead of just outputting a single value for the positive class, let us output a vector of probabilities for each class:

$$\boldsymbol{f}(\mathbf{x}; \mathbf{W}) = \begin{bmatrix} P(y = 1 | \mathbf{x}; \mathbf{W}) \\ P(y = 2 | \mathbf{x}; \mathbf{W}) \\ \vdots \\ P(y = K | \mathbf{x}; \mathbf{W}) \end{bmatrix}$$

Below we build up to a model that does this.

Each element in $\boldsymbol{f}(\mathbf{x}; \mathbf{W})$ should be a "score" for how well input $\mathbf{x}$ matches that class.

For input $\mathbf{x}$, we set the score for class k to

$$\mathbf{w}_k^\top \mathbf{x}$$

But probabilities need to be positive. So we take the exponential:

$$e^{\mathbf{w}_k^\top \mathbf{x}}$$

But probabilities need to sum to one. So we normalise:

$$P(y = k | \mathbf{x}; \mathbf{W}) = \frac{\exp(\mathbf{w}_k^\top \mathbf{x})}{\sum_{j=1}^{K} \exp(\mathbf{w}_j^\top \mathbf{x})}$$

This gives us the *softmax regression* model:

Optimisation

We fit the model using maximum likelihood. This is equivalent to minimising the negative log likelihood:

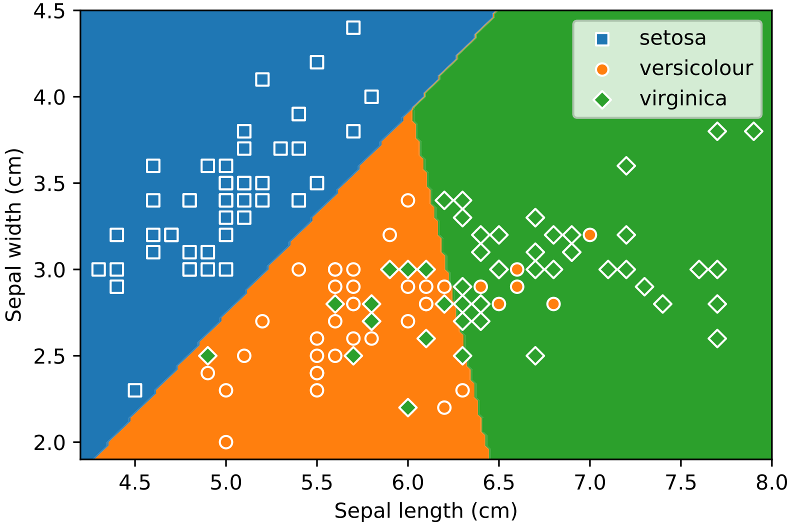
$$
\begin{aligned}
J(\mathbf{W}) &= -\log L(\mathbf{W}) \\
&= -\sum_{n=1}^{N} \log P(y^{(n)} | \mathbf{x}^{(n)}; \mathbf{W}) \\
&= -\sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{I}\{y^{(n)} = k\} \log \frac{\exp(\mathbf{w}_k^\top \mathbf{x}^{(n)})}{\sum_{j=1}^{K} \exp(\mathbf{w}_j^\top \mathbf{x}^{(n)})}
\end{aligned}
$$

Derivatives:

$$
\frac{\partial J(\mathbf{W})}{\partial \mathbf{w}_k} = -\sum_{n=1}^{N} \Big(\mathbb{I}\{y^{(n)} = k\} - f_k(\mathbf{x}^{(n)}; \mathbf{W}) \Big) \, \mathbf{x}^{(n)}
$$

Using these derivatives, we can minimise the loss using gradient descent.

Softmax regression on iris dataset



Output representation

Sometimes it is convenient to represent the target output as a *one-hot vector*:

$$\mathbf{y}^{(n)} = \begin{bmatrix}0 & 0 & \dots & 0 & 1 & 0 & \dots & 0\end{bmatrix}^\top$$

This one-hot vector has a one in the position $y_k^{(n)}$ if $\mathbf{x}^{(n)}$ is of class k, with zeros everywhere else. This is a convenient representation for the target output, since it allows us to vectorise algorithms.

We can then write the loss and gradient as:

$$\begin{aligned} J(\mathbf{W}) &= -\sum_{n=1}^{N}\sum_{k=1}^{K} y_k^{(n)} \log \frac{\exp(\mathbf{w}_k^\top \mathbf{x}^{(n)})}{\sum_{j=1}^{K} \exp(\mathbf{w}_j^\top \mathbf{x}^{(n)})} \\ \frac{\partial J(\mathbf{W})}{\partial \mathbf{w}_k} &= -\sum_{n=1}^{N} \left(y_k^{(n)} - f_k(\mathbf{x}^{(n)}; \mathbf{W})\right) \mathbf{x}^{(n)} \end{aligned}$$

This is mathematically exactly equivalent to using the versions with the indicator function.

(We will look at one-hot encodings for categorical *input* later.)

Relationship between softmax and binary logistic regression

For the special case that $K = 2$, you can show that softmax regression reduces to:

$$\boldsymbol{f}(\mathbf{x}; \mathbf{W}) = \begin{bmatrix} \frac{1}{1+\exp\left((\mathbf{w}_1 - \mathbf{w}_2)^\top \mathbf{x}\right)} \\ 1 - \frac{1}{1+\exp\left((\mathbf{w}_1 - \mathbf{w}_2)^\top \mathbf{x}\right)} \end{bmatrix}$$

So the model only depends on $\mathbf{w}_2 - \mathbf{w}_1$, a single vector.

We can replace this vector with $\mathbf{w}' = \mathbf{w}_2 - \mathbf{w}_1$, and only need to fit $\mathbf{w}'$.

This is equivalent to binary logistic regression.

Videos covered in this note

- Logistic regression 5.1: Multiclass - One-vs-rest classification (5 min)
- Logistic regression 5.2: Multiclass - Softmax regression (15 min)

Reading

- ISLR 4.3.5
- UFLDL Tutorial: Softmax Regression