

Machine Learning

Textbook: Machine Learning Refined, by Watt, Borhani, and Katsaggelos, 2nd edition:
MLRefined.com

Previous Lecture

- ♦ Binary classification predicts one of two class labels rather than a numerical response.
- ♦ A linear classifier first produces a real-valued score $f(\mathbf{x}; \mathbf{w})$.
- ♦ The sign of the score gives the predicted class, and $f(\mathbf{x}; \mathbf{w}) = 0$ defines the decision boundary.
- ♦ The product $y_p f_p$ is positive for a correctly classified observation and negative for a misclassified one.
- ♦ Zero-one loss directly counts classification mistakes but is difficult to optimize w/gradient methods.
- ♦ Least-squares loss is not naturally aligned with classification, motivating a new smooth classification loss.



§6.2, 6.3 - Logistic Regression

Last time:

a linear classifier produced a score

$$f(\mathbf{x}; \mathbf{w}) = w_0 + \mathbf{v}^T \mathbf{x}.$$

Its sign gave the predicted class.

But the score was not a probability.

Today:

turn that score into a probability.

What We Want

For class +1,

we want a probability

$$p(\mathbf{x})$$

satisfying

$$0 < p(\mathbf{x}) < 1.$$

We want:

very negative score $\Rightarrow p \approx 0$;

score 0 $\Rightarrow p = 1/2$;

very positive score $\Rightarrow p \approx 1$.

The logistic sigmoid does exactly this.

The Logistic Sigmoid

Define

$$\sigma(t) = \frac{1}{1+e^{-t}}.$$

Domain:

$$-\infty < t < \infty.$$

Range:

$$0 < \sigma(t) < 1.$$

Important values:

$$\sigma(0) = \frac{1}{2},$$

$$\sigma(t) \rightarrow 1 \text{ as } t \rightarrow \infty,$$

$$\sigma(t) \rightarrow 0 \text{ as } t \rightarrow -\infty.$$

Some Sigmoid Values

t	-4	-2	0	2	4
$\sigma(t)$	0.018	0.119	0.500	0.881	0.982

The sigmoid compresses the entire real line into the probability interval (0, 1).

The transition occurs smoothly near $t = 0$.

A Useful Symmetry

Compute

$$\sigma(-t) = \frac{1}{1+e^t}.$$

Also

$$\begin{aligned} 1 - \sigma(t) &= 1 - \frac{1}{1+e^{-t}} \\ &= \frac{e^{-t}}{1+e^{-t}} \\ &= \frac{1}{1+e^t}. \end{aligned}$$

Therefore

$$\sigma(-t) = 1 - \sigma(t).$$

Positive and negative scores are treated symmetrically.

Derivative of the Sigmoid

Differentiate:

$$\sigma(t) = (1 + e^{-t})^{-1}.$$

Then

$$\sigma'(t) = \frac{e^{-t}}{(1+e^{-t})^2}.$$

But

$$\sigma(t) = \frac{1}{1+e^{-t}}$$

and

$$1 - \sigma(t) = \frac{e^{-t}}{1+e^{-t}}.$$

So

$$\sigma'(t) = \sigma(t)[1 - \sigma(t)].$$

This derivative will simplify our optimization dramatically.

Logistic Regression

Begin with the linear score

$$f(\mathbf{x}; \mathbf{w}) = w_0 + \mathbf{v}^T \mathbf{x}.$$

Then define

$$P(Y = +1 \mid \mathbf{x}) = \sigma(f(\mathbf{x}; \mathbf{w})).$$

By the sigmoid symmetry:

$$P(Y = -1 \mid \mathbf{x}) = \sigma(-f(\mathbf{x}; \mathbf{w})).$$

The model is called **logistic regression**.

Despite its name:

it is normally used for classification.

Why the Word “Regression”?

Logistic regression does not regress directly on the class labels.

Instead:

the linear score

$$f(\mathbf{x}; \mathbf{w})$$

models the **log-odds** of class +1.

We will derive this shortly.

The final output is a class probability,
not an unrestricted numerical response.

Return to Our One-Dimensional Classifier

From Lesson 1.14:

$$f(x) = x - 3.5.$$

Now define

$$p(x) = \sigma(x - 3.5).$$

Then

x	1	2	3	4	5	6
$f(x)$	-2.5	-1.5	-0.5	0.5	1.5	2.5
$p(+1 \mid x)$	.076	.182	.378	.622	.818	.924

The hard boundary is still at
 $x = 3.5$.

But we now also have a graded measure of confidence.

The 0.5 Decision Rule

Predict class +1 when
 $P(Y = +1 \mid \mathbf{x}) \geq \frac{1}{2}$.

Because the sigmoid is increasing:
 $\sigma(f) \geq \frac{1}{2}$
exactly when
 $f \geq 0$.

Thus
 $P(+1 \mid \mathbf{x}) \geq \frac{1}{2} \iff f(\mathbf{x}; \mathbf{w}) \geq 0$.

So logistic regression keeps the same linear decision boundary:
 $f(\mathbf{x}; \mathbf{w}) = 0$.

The sigmoid adds probabilities
without changing the boundary at threshold 1/2.

Odds

If an event has probability p ,
its odds are
 $\frac{p}{1-p}$.

Examples:
 $p = 1/2$ gives odds 1;
 $p = 3/4$ gives odds 3;
 $p = 1/5$ gives odds 1/4.

Odds compare
probability of the event
with probability of its complement.

Sigmoid Odds

Let

$$p = \sigma(f) = \frac{1}{1+e^{-f}}.$$

Then

$$1 - p = \frac{e^{-f}}{1+e^{-f}}.$$

Therefore

$$\begin{aligned} &\frac{p}{1-p} \\ &= \frac{1/(1+e^{-f})}{e^{-f}/(1+e^{-f})} \\ &= e^f. \end{aligned}$$

So

$$\frac{p}{1-p} = e^f.$$

Log-Odds Are Linear

Take logarithms:

$$\log\left(\frac{p}{1-p}\right) = f(\mathbf{x}; \mathbf{w}).$$

For logistic regression:

$$\log\left(\frac{p}{1-p}\right) = w_0 + \mathbf{v}^T \mathbf{x}.$$

This quantity is called the **logit**.

Thus:

logistic regression assumes that
the log-odds are linear in the features.

Interpret a Coefficient Through Odds

Suppose

$$f(x) = w_0 + w_1 x.$$

Increase x by one unit.

The log-odds increase by

$$w_1.$$

Therefore the odds are multiplied by

$$e^{w_1}.$$

Example:

$$\text{if } w_1 = \log 2,$$

each one-unit increase in x
doubles the odds of class +1.

This interpretation is specific to odds,
not probability itself.

How Do We Learn the Parameters?

In regression:
choose parameters by minimizing a loss.

Same idea here.

But now each observation has a class label
 $y_p \in \{-1, +1\}$.

We want the model to assign high probability
to the observed class.

For observation p :
 $P(Y = y_p \mid \mathbf{x}_p) = \sigma(y_p f_p),$

where
 $f_p = f(\mathbf{x}_p; \mathbf{w}).$

Why Does $\sigma(y_p f_p)$ Work?

If
 $y_p = +1,$

then
 $\sigma(y_p f_p) = \sigma(f_p) = P(Y = +1 \mid \mathbf{x}_p).$

If
 $y_p = -1,$

then
 $\sigma(y_p f_p) = \sigma(-f_p) = P(Y = -1 \mid \mathbf{x}_p).$

So the same expression gives
the probability assigned to the observed class
in both cases.

Again:
the product $y_p f_p$
is the key signed quantity.

Likelihood of the Training Data

Assume observations are conditionally independent.

Then the probability assigned to all observed labels is
 $L(\mathbf{w}) = \prod_{p=1}^P \sigma(y_p f_p).$

This is the likelihood.

We want parameters that make

the observed labels as probable as possible.

So:

maximize $L(\mathbf{w})$.

As in probabilistic regression:

products are inconvenient.

Take logarithms.

Negative Log-Likelihood

Maximizing L

is equivalent to minimizing

$$-\log L.$$

Since

$$\log \prod_p a_p = \sum_p \log a_p,$$

we obtain

$$-\log L(\mathbf{w}) = -\sum_{p=1}^P \log \sigma(y_p f_p).$$

Now simplify one term.

Because

$$\sigma(z) = \frac{1}{1+e^{-z}},$$

we have

$$-\log \sigma(z) = \log(1 + e^{-z}).$$

The Logistic Loss

For one observation:

$$\ell(y, f) = \log(1 + e^{-yf}).$$

For the whole dataset:

$$g(\mathbf{w}) = \frac{1}{P} \sum_{p=1}^P \log(1 + e^{-y_p f_p}).$$

This is the **logistic loss**.

It came directly from maximum likelihood.

It is smooth.

So gradient-based optimization becomes possible again.

Logistic Loss Depends Only on the Margin

Let

$$m = yf.$$

Then

$$\ell(m) = \log(1 + e^{-m}).$$

If

m is large and positive:
correct with high confidence,
loss near 0.

If

$m = 0$:
 $\ell(0) = \log 2$.

If

m is negative:
misclassified,
loss grows.

Some Logistic Loss Values

Margin m	-4	-2	0	2	4
$\log(1 + e^{-m})$	4.018	2.127	0.693	0.127	0.018

Compare with zero-one loss:

Margin m	-4	-2	0	2	4
Zero-one loss	1	1	1	0	0

Zero-one loss only records right / wrong.

Logistic loss also records confidence.

Derivative With Respect to the Margin

Let

$$\ell(m) = \log(1 + e^{-m}).$$

Then

$$\begin{aligned} \ell'(m) &= \frac{-e^{-m}}{1+e^{-m}} \\ &= -\frac{1}{1+e^m} \\ &= -\sigma(-m). \end{aligned}$$

This derivative is always negative.

Increasing the signed margin
always lowers logistic loss.

Gradient for One Observation

Let

$$f_p = \tilde{\mathbf{x}}_p^T \mathbf{w},$$

where the leading entry of $\tilde{\mathbf{x}}_p$ is 1 for the intercept.

Then

$$m_p = y_p \tilde{\mathbf{x}}_p^T \mathbf{w}.$$

By the chain rule:

$$\begin{aligned} \nabla_{\mathbf{w}} \ell_p &= -\sigma(-m_p) y_p \tilde{\mathbf{x}}_p. \end{aligned}$$

Thus

$$\nabla_{\mathbf{w}} \ell_p = -y_p \sigma(-y_p f_p) \tilde{\mathbf{x}}_p.$$

Misclassified or uncertain points receive the larger gradient magnitudes.

Gradient of the Average Logistic Loss

For

$$g(\mathbf{w}) = \frac{1}{P} \sum_{p=1}^P \ell_p,$$

we get

$$\nabla g(\mathbf{w}) = -\frac{1}{P} \sum_{p=1}^P y_p \sigma(-y_p f_p) \tilde{\mathbf{x}}_p.$$

Now gradient descent can train the classifier:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha \nabla g(\mathbf{w}_k).$$

The optimization machinery from our first unit returns.

A Small Numerical Example

Use

$$x = (1, 2, 3, 4, 5, 6),$$

with

$$y = (-1, -1, -1, +1, +1, +1).$$

Take

$$f(x) = x - 3.5.$$

Margins:

$$2.5, \ 1.5, \ 0.5, \ 0.5, \ 1.5, \ 2.5.$$

All margins are positive.

Average logistic loss:

$$\frac{1}{6} \sum_{p=1}^6 \log(1 + e^{-m_p})$$

$$\approx 0.251.$$

Classification error is already 0,

but logistic loss is not zero.

It still rewards larger correct margins.

An Important Subtlety: Separable Data

Our six observations are perfectly separable.

Consider scores

$$f_c(x) = c(x - 3.5),$$

where

$$c > 0.$$

Every value of c
gives the same classification boundary.

But increasing c
makes every positive margin larger.

Therefore
logistic loss keeps decreasing as
 $c \rightarrow \infty$.

So for perfectly separable data:
unregularized logistic regression may have no finite minimizing coefficient vector.

The likelihood approaches its supremum as the coefficients grow.

This will motivate regularization later.

Switching to 0/1 Labels

Probability notation is often written with

$$z_p \in \{0, 1\}.$$

Let

$$p_p = P(Z_p = 1 \mid \mathbf{x}_p) = \sigma(f_p).$$

Then

$$P(Z_p = 0 \mid \mathbf{x}_p) = 1 - p_p.$$

For one observed label:

$$P(Z_p = z_p \mid \mathbf{x}_p) = p_p^{z_p} (1 - p_p)^{1-z_p}.$$

This compact Bernoulli formula handles both labels.

Next lesson we will study its negative log-likelihood in detail.

The Decision Boundary Is Still Linear

Even though

$$p(\mathbf{x}) = \sigma(w_0 + \mathbf{v}^T \mathbf{x})$$

is nonlinear in $\mathbf{x}$,

the 0.5 boundary satisfies

$$p(\mathbf{x}) = \frac{1}{2}$$

$$\Leftrightarrow$$

$$w_0 + \mathbf{v}^T \mathbf{x} = 0.$$

Therefore basic logistic regression is still a **linear classifier**.

The sigmoid changes the output scale,
not the geometry of the basic boundary.

Logistic Regression in Python

Define the sigmoid:

```
def sigmoid(t):  
    return 1 / (1 + np.exp(-t))
```

New Python idea:

`np`

applies the exponential function element by element.

Use the score from Lesson 1.14:

```
x = np.array([1., 2., 3., 4., 5., 6.])  
  
scores = x - 3.5  
probabilities = sigmoid(scores)  
  
print(probabilities)
```

Approximately:

```
[0.0759 0.1824 0.3775 0.6225 0.8176 0.9241]
```

Probability to Class in Python

Use threshold 0.5:

```
y_pred = np.where(probabilities >= 0.5, 1, -1)
```

This gives the same labels as

```
y_pred = np.where(scores >= 0, 1, -1)
```

because

$$\sigma(f) \geq 0.5$$

exactly when

$$f \geq 0.$$

Probability rule and sign rule agree.

Logistic Loss in Python

For labels

```
y ∈ {-1,+1}:
def logistic_loss(w, X, y):
    scores = X @ w
    margins = y * scores

    return np.mean(np.logaddexp(0, -margins))
```

New Python idea:

```
np(a,b)
```

computes

```
log(ea + eb)
```

in a numerically stable way.

Thus

```
np(0,-m)
```

computes

```
log(1 + e-m).
```

This is safer than directly forming

```
np(1 + np(-m))
```

for very large magnitudes.

Gradient in Python

For augmented design matrix *X*:

```
def grad_logistic_loss(w, X, y):
    scores = X @ w
    margins = y * scores

    factors = -y * sigmoid(-margins)

    return np.mean(
        factors[:, None] * X,
        axis=0
    )
```

New Python idea:

```
factors[:, None]
```

turns a one-dimensional array into a column

```
so each row of X
```

is multiplied by its own factor.

Then

```
np(...,axis = 0)
```

averages down the rows.

A Gradient-Descent Training Loop

For a dataset that is not perfectly separable:

```
w = np.zeros(X.shape[1])
alpha = 0.1

for k in range(1000):
    grad = grad_logistic_loss(w, X, y)
    w = w - alpha*grad
```

Same pattern we have seen repeatedly:

- compute gradient;
- update parameters;
- repeat.

The model changed.

The optimization framework did not.

Regression and Logistic Regression

	Linear Regression	Logistic Regression
Score	$f = \mathbf{x}^T \mathbf{w}$	$f = \mathbf{x}^T \mathbf{w}$
Output	f	$\sigma(f)$
Output range	all real numbers	$(0, 1)$
Typical target	numerical	binary class
Probabilistic model	Gaussian	Bernoulli
Typical loss	squared error	logistic / log loss

Same linear score.

Different output model.

Different probability distribution.

Different loss.

What did we learn?

- ◆ The logistic sigmoid maps any real-valued score into the interval $(0, 1)$.
- ◆ Logistic regression models $P(Y = +1 \mid \mathbf{x}) = \sigma(f(\mathbf{x}; \mathbf{w}))$.
- ◆ A probability threshold of $1/2$ gives the same linear decision boundary $f(\mathbf{x}; \mathbf{w}) = 0$.
- ◆ The linear score equals the log-odds: $\log[p/(1 - p)] = f(\mathbf{x}; \mathbf{w})$.
- ◆ Maximum likelihood with binary labels produces the smooth logistic loss $\log(1 + e^{-y f})$.
- ◆ Logistic loss depends on the signed margin and can be optimized with gradient methods.



Next

- Logistic regression gives a probability for each binary class.

- Maximum likelihood produces a logarithmic loss.
- Next we rewrite that loss in the standard 0/1 probability language.
- We will see why it is called cross-entropy and derive its gradient.
- **1.16 - Cross-Entropy**