6 Empirical risk minimization

Date	@October 29, 2024
Topic	Theory

Learning goals

Learning goals is about:

- know about notion of expected loss or risk
- know that average loss approximates risk
- know about empirical risk minimization
- know some design choices in ERM

This means learning an hypothesis $h \in H$ $h : x \to y$ such that $h(x) = y$ fro any data point.

In this context:

- data is the set of data points (x,y)
- model is the set H of hypothesis maps h(.)
- loss is the quality measure L((x,y),h)

Expected loss of risk

We have to interpret data points as realizations of independent and identically distributed variables with probability distribution p(x,y), and then to define loss incurred for any data point as the expected loss, which is also called expected risk or Bayes risk.

$$\mathbb{E}\{L((\mathbf{x}, y), h)\} := \int_{\mathbf{x}, y} L((\mathbf{x}, y), h) dp(\mathbf{x}, y).$$

To compute it we need to know the probability distribution p(**x**,y) of data points (**x**,y).

Empirical risk

The idea is to approximate expected loss by average loss on data points, estimating an empirical risk.

$$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})\}.$$

$$\widehat{L}(h|\mathcal{D}) = (1/m) \sum_{i=1}^{m} L((\mathbf{x}^{(i)}, y^{(i)}), h).$$

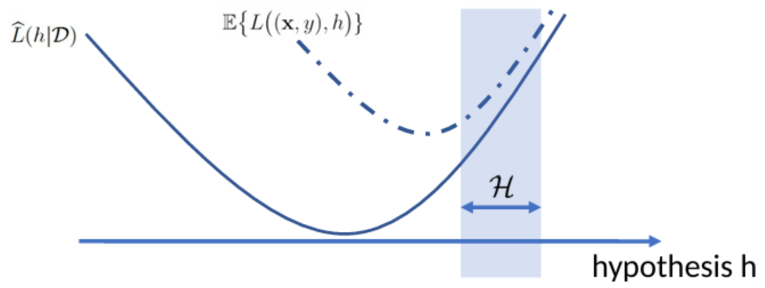
$$\mathbb{E}\{L((\mathbf{x}, y), h)\} \approx \widehat{L}(h|\mathcal{D}) \quad \text{for sufficiently large sample size } m.$$

Empirical risk minimization

$$\hat{h} \in \operatorname*{argmin}_{h \in \mathcal{H}} \widehat{L}(h|\mathcal{D}) = \operatorname*{argmin}_{h \in \mathcal{H}} (1/m) \sum_{i=1}^{m} L((\mathbf{x}^{(i)}, y^{(i)}), h).$$

Thus, any data point = training data points.

It means learning hypothesis out of a hypothesis space or model that incurs minimum average loss when predicting labels of training data points based on their features.


$\widehat{L}(h|\mathcal{D})$
$\mathbb{E}\{L((\mathbf{x},y),h)\}$
$\mathcal{H}$
hypothesis h

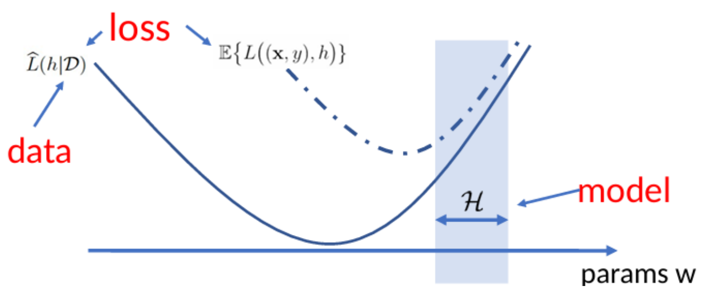
learnt (optimal) parameter vector

$$\widehat{\mathbf{w}} = \operatorname*{argmin}_{\mathbf{w} \in \mathbb{R}^n} f(\mathbf{w})$$

loss incurred by h(.) for i-th data point

$$\text{with } f(\mathbf{w}) := \underbrace{(1/m) \sum_{i=1}^{m} L((\mathbf{x}^{(i)}, y^{(i)}), h^{(\mathbf{w})})}_{\widehat{L}(h^{(\mathbf{w})}|\mathcal{D})}.$$

average loss or empirical risk

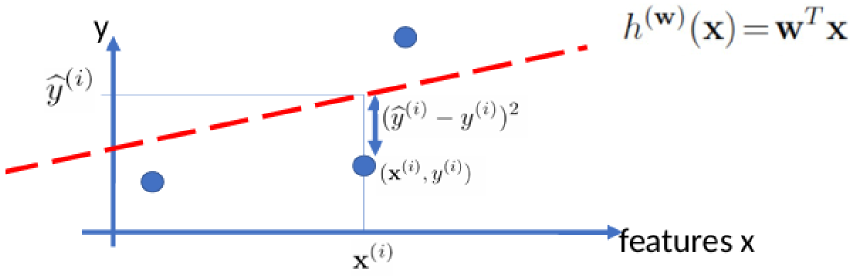

loss
$\widehat{L}(h|\mathcal{D})$
$\mathbb{E}\{L((\mathbf{x},y),h)\}$
data
$\mathcal{H}$
model
params w

ERM is learning a hypothesis in model that incurs in smallest empirical risk (loss) when predicting labels of training data points.

ERM for regression

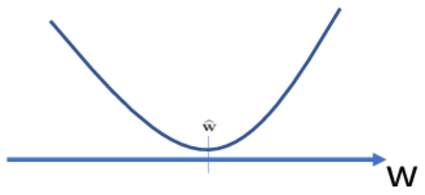
Linear regression

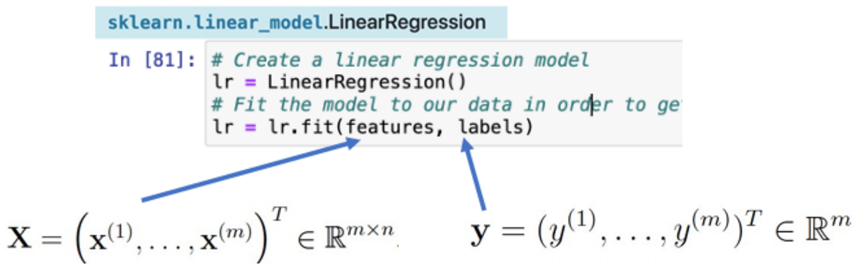
- Data is data points characterized by numeric feature vector and numeric label
- Model consists of linear hypothesis maps
- Loss is the squared error loss


y
$\hat{y}^{(i)}$
$h^{(\mathbf{w})}(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$
$(\hat{y}^{(i)} - y^{(i)})^2$
$(\mathbf{x}^{(i)}, y^{(i)})$
features x
$\mathbf{x}^{(i)}$

We have to choose parameter/weight vector w in order to minimize average squared error loss.

$$\boxed{\widehat{\mathbf{w}} = \underset{\mathbf{w}\in\mathbb{R}^n}{\operatorname{argmin}} (1/m) \sum_{i=1}^{m} \big(y^{(i)} - \mathbf{w}^T\mathbf{x}^{(i)}\big)^2. \qquad (4.5)}$$

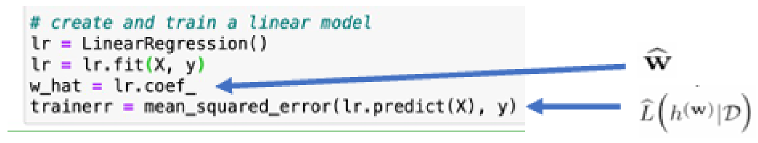




sklearn.linear_model.LinearRegression

```
In [81]: # Create a linear regression model
         lr = LinearRegression()
         # Fit the model to our data in order to ge
         lr = lr.fit(features, labels)
```

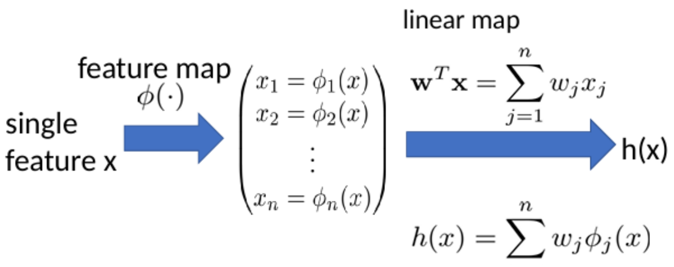
$$\mathbf{X} = \left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right)^T \in \mathbb{R}^{m \times n} \qquad \mathbf{y} = \left(y^{(1)}, \ldots, y^{(m)}\right)^T \in \mathbb{R}^m$$



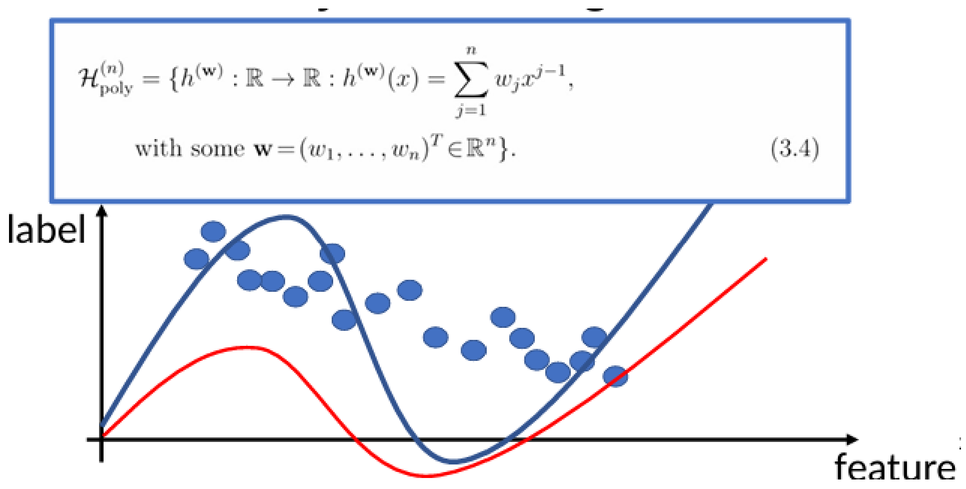
```
# create and train a linear model
lr = LinearRegression()
lr = lr.fit(X, y)
w_hat = lr.coef_
trainerr = mean_squared_error(lr.predict(X), y)
```

$$\widehat{\mathbf{w}} \qquad \widehat{L}\left(h^{(\mathbf{w})}|\mathcal{D}\right)$$

Feature map + linear model



Polinomial regression



$$\mathcal{H}^{(n)}_{\text{poly}} = \{h^{(\mathbf{w})} : \mathbb{R} \rightarrow \mathbb{R} : h^{(\mathbf{w})}(x) = \sum_{j=1}^{n} w_j x^{j-1}, \text{ with some } \mathbf{w} = (w_1, \ldots, w_n)^T \in \mathbb{R}^n\}. \tag{3.4}$$

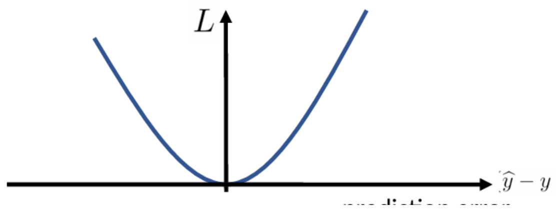
Polynomial regression is linear regression with feature transformation.

Measuring error via loss function

Loss function is also design choice.

Squared error loss

$$L := (\widehat{y} - y)^2$$

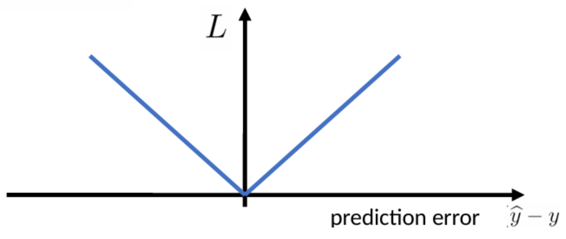


Squared error loss is sensitive to outliers. Minimize squared error loss forces predictor towards outlier.

Minimize squared error loss forces predictor towards outlier.

Absolute error loss

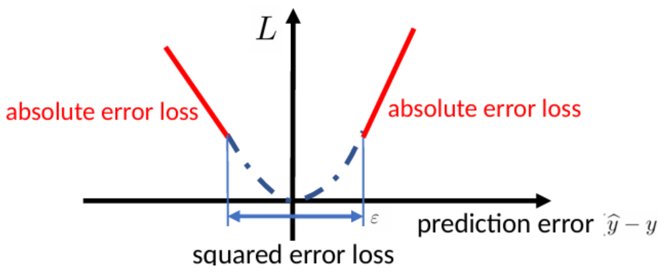
$$L := |\widehat{y} - y|$$



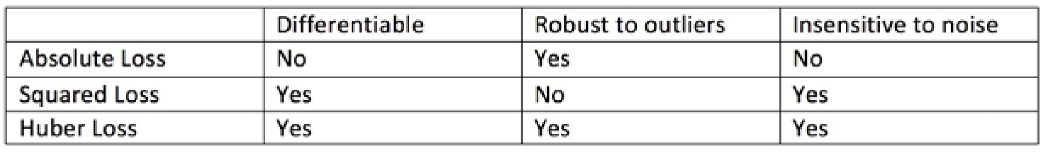
Absolute error loss is robust to outliers; it “tolerates” few outliers.

Huber loss

$$L\left((\mathbf{x}, y), h\right) = \begin{cases} (1/2)(y - h(\mathbf{x}))^2 & \text{for } |y - h(\mathbf{x})| \leq \varepsilon \\ \varepsilon(|y - h(\mathbf{x})| - \varepsilon/2) & \text{else.} \end{cases}$$



Loss comparison



	Differentiable	Robust to outliers	Insensitive to noise
Absolute Loss	No	Yes	No
Squared Loss	Yes	No	Yes
Huber Loss	Yes	Yes	Yes

All of them are convex functions.

Non-convex and non-differentiable objective functions are more difficult to minimize.

ERM for classification

While regression involves numeric labels and loss functions obtained from distance between numbers, classification involves categorical discrete-valued labels, which distinguishes between binary and multi-class classification, and loss function obtained from confidence measures.

Logistic regression

For logistic regression:

- data points include numeric features, same as in linear regression
- model is the space of linear maps, same as in linear regression
- loss is logistic loss, different from liner regression

It consists of a linear hypothesis $h(x) = w^t x$ and the sign of $h(x)$ is used for label prediction, since:

- $h(x) > 0$ means $sign(h(x)) = \hat{y} = 1$
- $h(x) < 0$ means $sign(h(x)) = \hat{y} = -1$

The absolute value $|h(x)|$ is used as confidence measure:

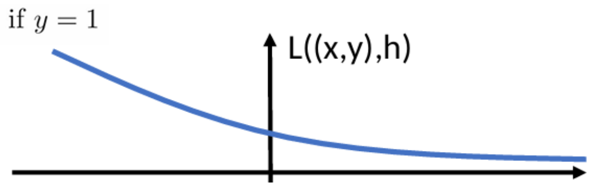
- $h(x) = 1000000 > 0$ means very confident in $\hat{y} = 1$
- $h(x) = 1000000 > 0$ means very confident in $\hat{y} = 1$

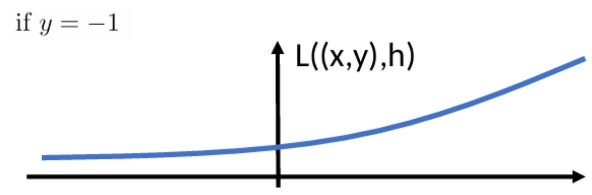
Logistic loss

When using -1 and 1 as label values, the formula is

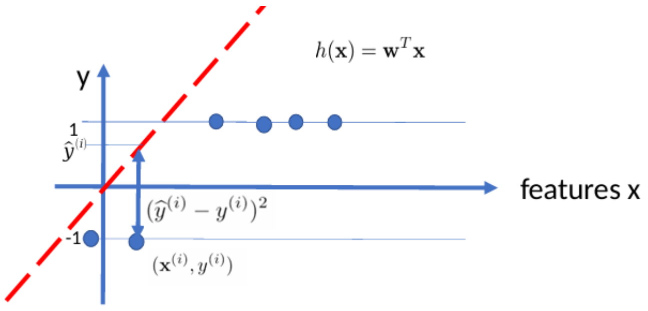
$$L\big((\mathbf{x}, y), h\big) := \log(1 + \exp(-yh(\mathbf{x}))).$$

It is differentiable and convex as function of h(x) and, in turn, of weight w for linear $h(x) = w^t x$.

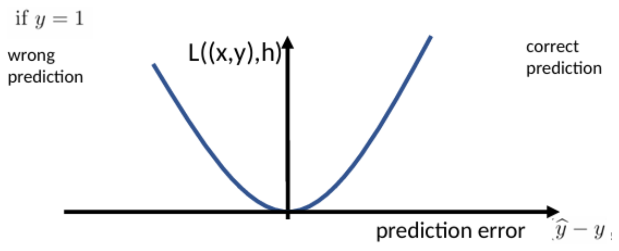


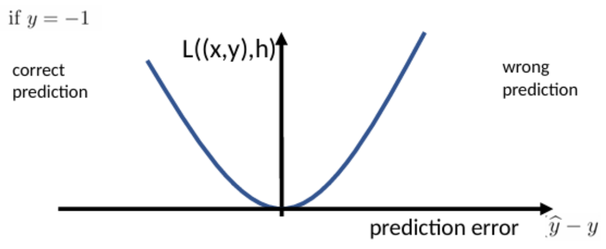


We don't use average squared loss because if $y^{(i)}$ is -1 and $x^{(i)}$ is positive, $sign(h(x^{(i)})) = +1$, which is an error.



$h(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$
y
1
$\hat{y}^{(i)}$
features x
$(\hat{y}^{(i)} - y^{(i)})^2$
-1
$(\mathbf{x}^{(i)}, y^{(i)})$





Decision boundqary in 2D

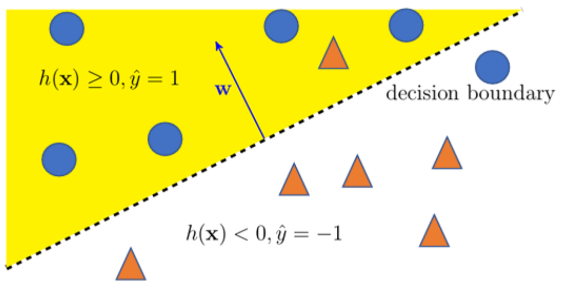
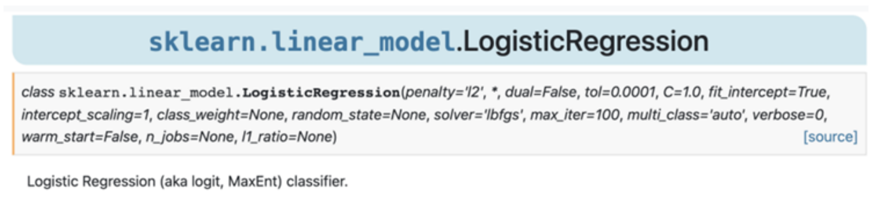

$h(\mathbf{x}) \geq 0, \hat{y} = 1$
$\mathbf{w}$
decision boundary
$h(\mathbf{x}) < 0, \hat{y} = -1$

Figure 2.9: A hypothesis $h : \mathcal{X} \rightarrow \mathcal{Y}$ for a binary classification problem, with label space $\mathcal{Y} = \{-1, 1\}$ and feature space $\mathcal{X} = \mathbb{R}^2$, can be represented conveniently via the decision boundary (dashed line) which separates all feature vectors $\mathbf{x}$ with $h(\mathbf{x}) \geq 0$ from the region of feature vectors with $h(\mathbf{x}) < 0$. If the decision boundary is a hyperplane $\{\mathbf{x} : \mathbf{w}^T\mathbf{x} = b\}$ (with normal vector $\mathbf{w} \in \mathbb{R}^n$), we refer to the map h as a linear classifier.

Logistic regression in python



sklearn.linear_model.LogisticRegression

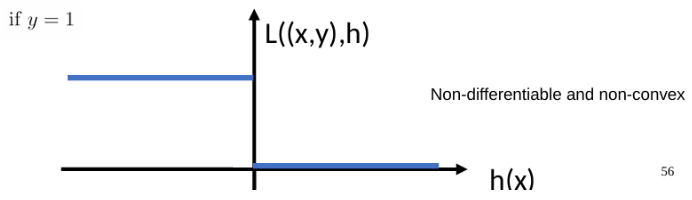
class `sklearn.linear_model.LogisticRegression`*(penalty='l2', *, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None)* [source]

Logistic Regression (aka logit, MaxEnt) classifier.

Losses in classification

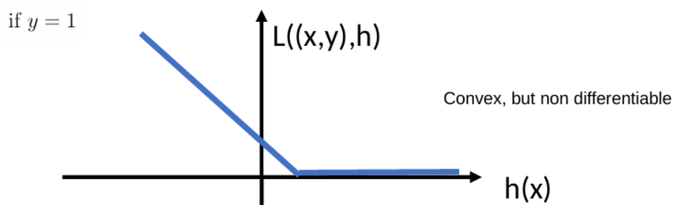
0/1 loss

$$L\left((\mathbf{x}, y), h\right) := \begin{cases} 1 & \text{if } y \neq \hat{y} \\ 0 & \text{else,} \end{cases} \quad \text{with } \hat{y} = 1 \text{ for } h(\mathbf{x}) \geq 0, \text{ and } \hat{y} = -1 \text{ for } h(\mathbf{x}) < 0$$



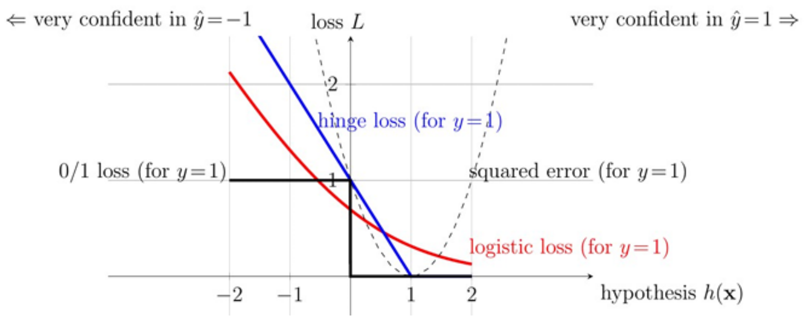
Hinge loss

$$L\left((\mathbf{x}, y), h\right) := \max\{0, 1 - yh(\mathbf{x})\}.$$



Loss comparison

if $y = 1$


$\Leftarrow$ very confident in $\hat{y}=-1$
loss L
very confident in $\hat{y}=1 \Rightarrow$
hinge loss (for $y=1$)
0/1 loss (for $y=1$)
squared error (for $y=1$)
logistic loss (for $y=1$)
-2
-1
hypothesis $h(\mathbf{x})$