IT496: Introduction to Data Mining

Lecture 16

Logistic Regression

Arpit Rana

21st September 2023

Classification

We want to create programs that make predictions.

- We have been studying *regression*: regressors are programs that predict numeric target values.
- We turn now to *classification*: classifiers predict an object's class from a *finite set of classes*.

Example

Given a vector of feature values that describe an email, predict whether the email is *spam* or *ham*.

Notation

Our notation will be the same that we used for regression:

- $\boldsymbol{x}$ for an object, $\mathbf{y}$ for the actual class label, $\hat{\mathbf{y}}$ for the predicted class label.
- We assume we have a finite set of labels, $\mathbf{C}$, one per class.
  - Given an object $\boldsymbol{x}$, our task is to assign one of the labels $\hat{\mathbf{y}} \in \mathbf{C}$ to the object.
- We will often use integers for the labels.
  - E.g. given an email, a spam filter predicts $\hat{\mathbf{y}} \in \{0, 1\}$, where 0 means *ham* and 1 means *spam*.
  - But a classifier should not treat these as continuous, e.g. it should never output 0.5.

Notation

Furthermore, where there are more than two labels, we should not assume a relationship between the labels.

- Suppose there are three classes {1, 2, 3}.
- Suppose we are classifying object $\boldsymbol{x}$ and we happen to know that its actual class label is $\boldsymbol{y=3}$.
  - One classifier predicts $\boldsymbol{\hat{y}=1}$.
  - Another classifier predicts $\boldsymbol{\hat{y}=2}$.
- Which classifier has done better?

A Variation of Classification

Given an object $\boldsymbol{x}$, a classifier outputs a label, $\hat{\mathbf{y}} \in \mathbf{C}$.

- Instead, a classifier could output a probability distribution over the labels **C**.
- For Example,
    - Given an email $\boldsymbol{x}$, a spam filter might output $\langle 0.2, 0.8 \rangle$ meaning $P(y = ham \mid x) = 0.2$ and $P(y = spam \mid x) = 0.8$
    - The probabilities must sum to 1.
- We can convert such a classifier into a more traditional one by taking the probability distribution and selecting the class with the highest probability:

$$\operatorname*{arg\,max}_{\hat{y} \in C} P(\hat{y} \mid x)$$

Types of Classification

We distinguish three types of classification:

- **Binary classification**, in which there are just two classes, i.e. $|\mathbf{C}|=2$,
  e.g. fail/pass, ham/spam, benign/malignant.
- **Multiclass classification**, where there are more than two classes, i.e. $|\mathbf{C}|>2$,
  e.g. let's say that a post to a forum or discussion board can be *a question*,*an answer*,*a clarification or an irrelevance*.
- **Multilabel classification**, where the classifier can assign $\boldsymbol{x}$ to more than one class. I.e. it outputs a set of labels, $\hat{\mathbf{y}} \subseteq \mathbf{C}$.
  - E.g. consider a movie classifier where the classes are genres, e.g. $\mathbf{C}$ = {comedy, action, horror, documentary, romance, musical}.
  - The classifier's output for *The Blues Brothers* should be {comedy,action,musical}.

Do not confuse this with multiclass classification.

Types of Classification

In fact, there are even more types of classification, but we will not be studying them further:

- **Ordered classification**, there is an ordering defined on the classes.
  - The ordering matters in measuring the performance of the classifier.
- E.g. consider a classifier that predicts a student's overall grade, i.e. {A, B, C, D}.
  - Suppose for student $\boldsymbol{x}$, the actual class **y=A**.
  - One classifier predicts **ŷ=B**.
  - Another classifier predicts **ŷ=C**.
  - Which classifier has done better?

Binary Classification

In binary classification, there are two classes.

- It is common to refer to one class (the one labelled 0) as the **negative class** and the other (the one labelled 1) as the **positive class**.
- It doesn't really matter which is which.
  - But, usually, we treat the class we are trying to identify, or the class that requires special action, as the positive class.
  - E.g. in spam filtering, ham is the negative class; spam is the positive class.
  - What about *tumour* classification?
- This terminology is extended to other things too, e.g. we can refer to **negative examples** and **positive examples**.

Class Exercise

Consider:

- Predicting tomorrow's rainfall.
- Predicting whether Ireland will have a white Christmas.
- Predicting the sentiment of a tweet (negative, neutral or positive).
- Predicting a person's opinion of a movie on a rating scale of 1 star (rotten) to 5 stars (fab).

Answer the following:

- Which are regression and which classification?
- If classification, which are binary and which are multiclass?
- If binary, which is the positive class and which the negative?

Logistic Regression

The Model

Logistic Regression

We can use *model-based learning* for classification.

- There are all sorts of model, but the simplest again is a linear model.
- For **regression**, we wanted to find the line/plane/hyperplane that best *fits* the training examples.
- For **classification**, we want to find the line/plane/hyperplane that best *separates* training examples of different classes.

For two features, if it is possible to find a line that separates the data (only **positive examples** on one side, only **negative examples** on the other), we say the dataset is *linearly separable*.

This generalizes from straight lines to planes and hyperplanes in the case of more features.

Logistic Regression

Despite its name, logistic regression is used for classification.

- At heart, it predicts a number (and turns it into a probability), and perhaps this is why its name mentions regression.
- At heart, it builds linear models.

Logistic Regression for Binary Classification

Let's start with logistic regression for binary classification.

- In this case, logistic regression predicts the probability that $\boldsymbol{x}$ belongs to the positive class.
- This is what logistic regression does:

$$\hat{y} = \begin{cases} 0 & if\ P(\hat{y} = 1 \,|\, x) < 0.5 \\ 1 & if\ P(\hat{y} = 1 \,|\, x) \geq 0.5 \end{cases}$$

Thresholding: so it outputs one of the two class labels

Where, $P(\hat{y} = 1 \,|\, x) = \sigma(x\beta)$

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

and $\boldsymbol{x}\beta$ is familiar from linear regression (and assumes that $\boldsymbol{x}$ has an extra 'feature', $\boldsymbol{x}_0 = 1$)

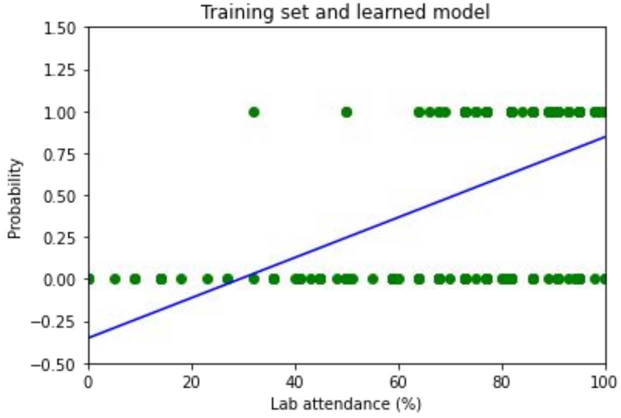
Why Not Just Linear Regression

In Linear Regression, each hypothesis h_β was of the form

$$\begin{aligned} h_\beta(x) &= \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n \\ &= x\beta \end{aligned}$$

where $\boldsymbol{x}$ is a row vector with (n+1) elements (with $\boldsymbol{x}_0$ = 1), and β is a (column) vector of coefficients.

Why can't we just use this directly to predict probabilities?



Probability of passing IT492 course given students' lab attendance

Why Not Just Linear Regression

The Logistic Function

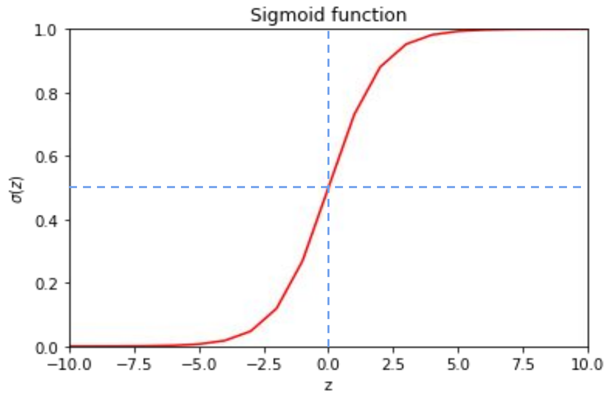
To 'squash' the values of $\boldsymbol{x}\beta$ to [0, 1] so we can treat them as probabilities

$$h_\beta(x) = \sigma(x\beta)$$

where $\boldsymbol{\sigma}$ is the **logistic function** (also called the 'logit'):

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

The logistic function is also called the **sigmoid function** (which is what we will call it) because it is S-shaped:



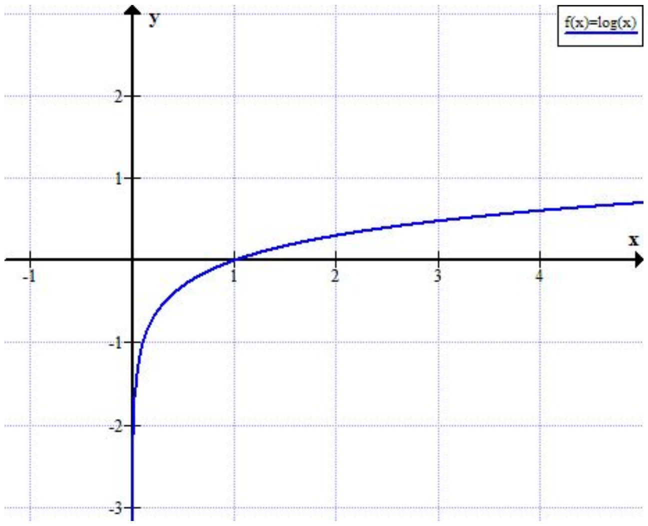
A minor point: the sigmoid function asymptotically approaches 0 and 1.

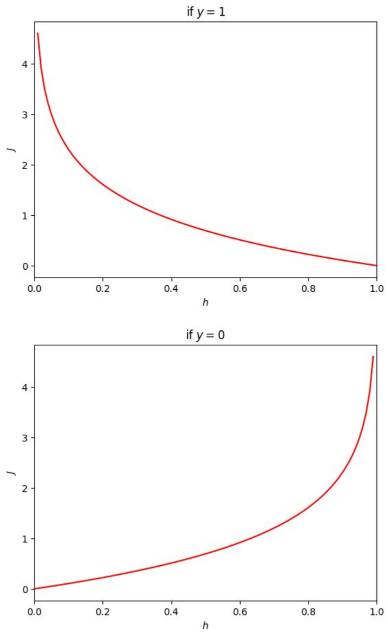
- So, in fact its values are in (0, 1) and not [0, 1].

Logistic Regression: Loss Function

For a given example $\boldsymbol{x}$, we propose the **binary cross-entropy loss** as the loss function -

$$J(x, y, \beta) = -(y \log(h_\beta(x)) + (1 - y)\log(1 - h_\beta(x)))$$





Logistic Regression: Loss Function

The overall loss function, *J*, is simply the average of this over all the training examples

$$J(X, y, \beta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(h_\beta\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right)\log\left(1 - h_\beta\left(x^{(i)}\right)\right)\right]$$

This loss function is sometimes called the **log loss** function.

Logistic Regression: Gradient Descent

So how do we find the hypothesis that minimizes this log loss function?

- Happily, this function is convex.
- But there is no equivalent to the Normal Equation, so we must use **Gradient Descent.**
- Not that it matters, but here is the partial derivative of its loss function with respect to β_j

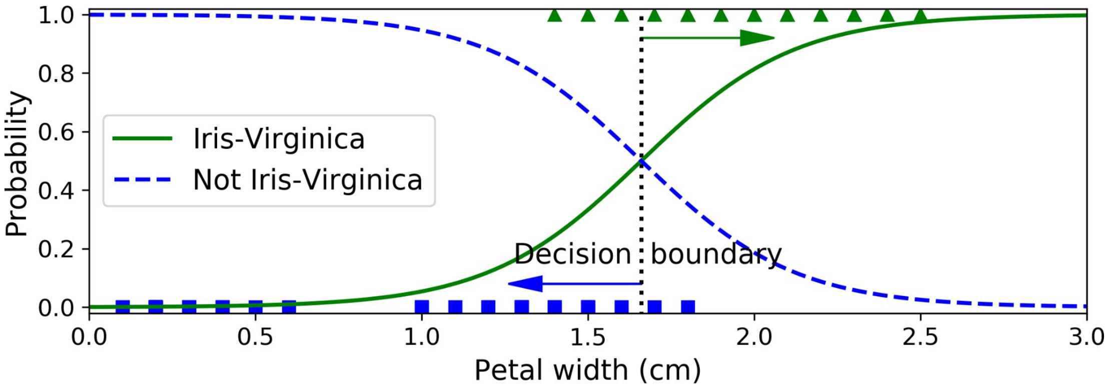
$$\frac{\partial J(X, y, \beta)}{\partial \beta_j} = \frac{1}{m} \sum_{i=1}^{m} \left(x^{(i)} \beta_j - y^{(i)} \right) \times x_j^{(i)}$$

Since we'll be using Gradient Descent, we must remember to scale our data.

Logistic Regression: Decision Boundary

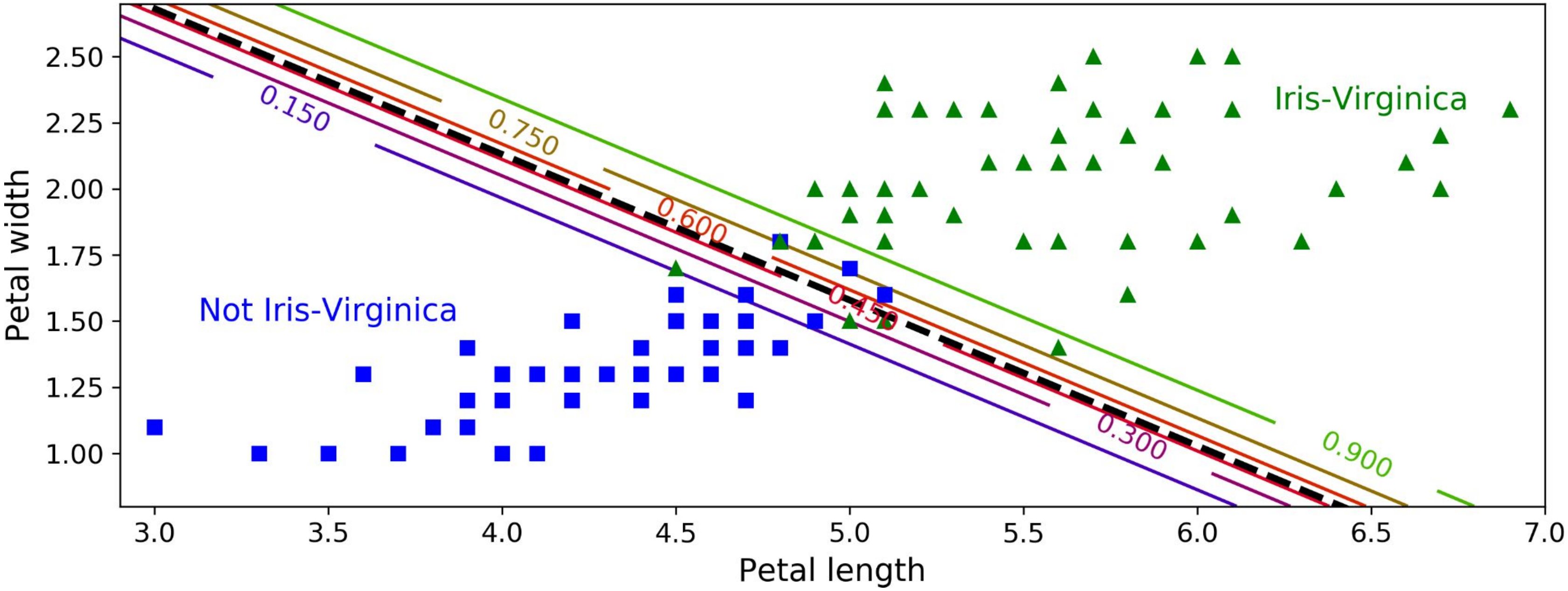
The examples for which logistic regression predicts probabilities of 0.5, $P(\hat{y} = 1 \mid x) = 0.5$ lie on what is called the *decision boundary*.

- If you look at the graph of the sigmoid function, its output is 0.5 when its input (z) is zero. It follows that the decision boundary are examples where $\boldsymbol{x}\beta = 0$.
- For example, we use the iris dataset that contains the *sepal* and *petal length* and *width* of 150 iris flowers of three different species: *Iris-Setosa*, *Iris-Versicolor*, and *Iris-Virginica*.
- Let's try to build a classifier to detect the Iris-Virginica type based only on the *petal width* feature.



Logistic Regression: Decision Boundary

- Let's try to build a classifier to detect the Iris-Virginica type using the same dataset but this time displaying two features: *petal width* and *length*.



Logistic Regression in Scikit-learn

FYR:

- If you want fine-grained control over the learning rate and so on, then scikit-learn offers you the `SGDClassifier` class.
- But most people use the `LogisticRegression` class, which sits on top of the `SGDClassifier` class.
- If you want to use regularization with Logistic Regression, then there is a separate scikit-learn class for ridge classification (`RidgeClassifier` but *none* for lasso).
- But you can instead use `LogisticRegression`, which has an argument called *`penalty`*, whose possible values include "`l1`" and "`l2`". The amount of regularization is usually controlled by a hyperparameter called *`alpha`* in the `Lasso` and `Ridge` classes.
- For `LogisticRegression`, this hyperparameter is called **`C`** and **`C`** is the inverse of *`alpha`*, so small values means more regularization!

Next lecture

Multinomial Logistic Regression

22^{nd} September 2023