Universidad Tecnológica de Bolívar
CARTAGENA DE INDIAS

Machine Learning

Logistic Regression

Edwin Puertas, Ph.D(c).
epuerta@utb.edu.co

Introduction

- Logit Regression
- It is commonly used to estimate the probability that an instance belongs to a particular class (e.g., what is the probability that this email is spam?).
- It's a binary classifier
  - If the estimated probability is greater than 50%, then the model predicts that the instance belongs to that class (called the positive class, labeled "1")
  - If the estimated probability is less than 50%, the model predicts that the instance does not belong to that class (belongs to the negative class, labeled "0").
- A Logistic Regression model computes a weighted sum of the input characteristics plus a bias term.

How does Logistic Regression work?

Logistic Regression model estimated probability

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

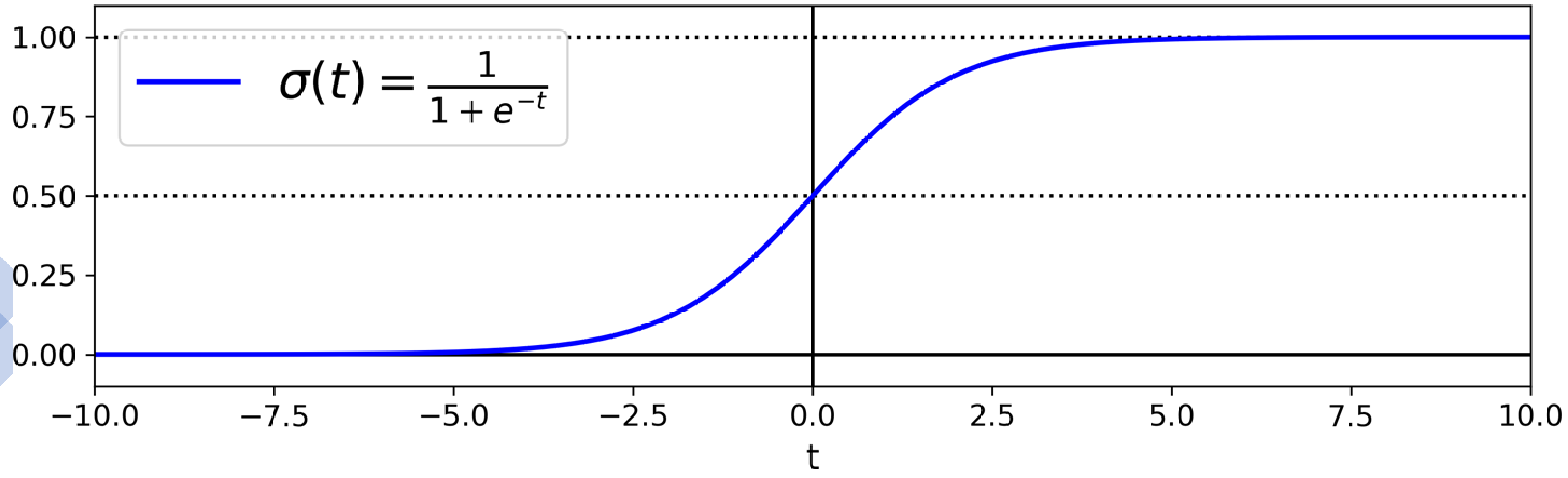
$\sigma(\cdot)$ is a *sigmoid function* (i.e., *S*-shaped) that outputs a number between 0 and 1.

$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^{\intercal}\mathbf{x})$$

Logistic function

Logistic Regression model has estimated the probability ***p^ = h$_\vartheta$(x)*** that an instance **x** belongs to the positive class, it can make its prediction *ŷ*

$$\hat{y} = \begin{cases} 0 & \text{if } \widehat{p} < 0.5 \\ 1 & \text{if } \widehat{p} \geq 0.5 \end{cases}$$

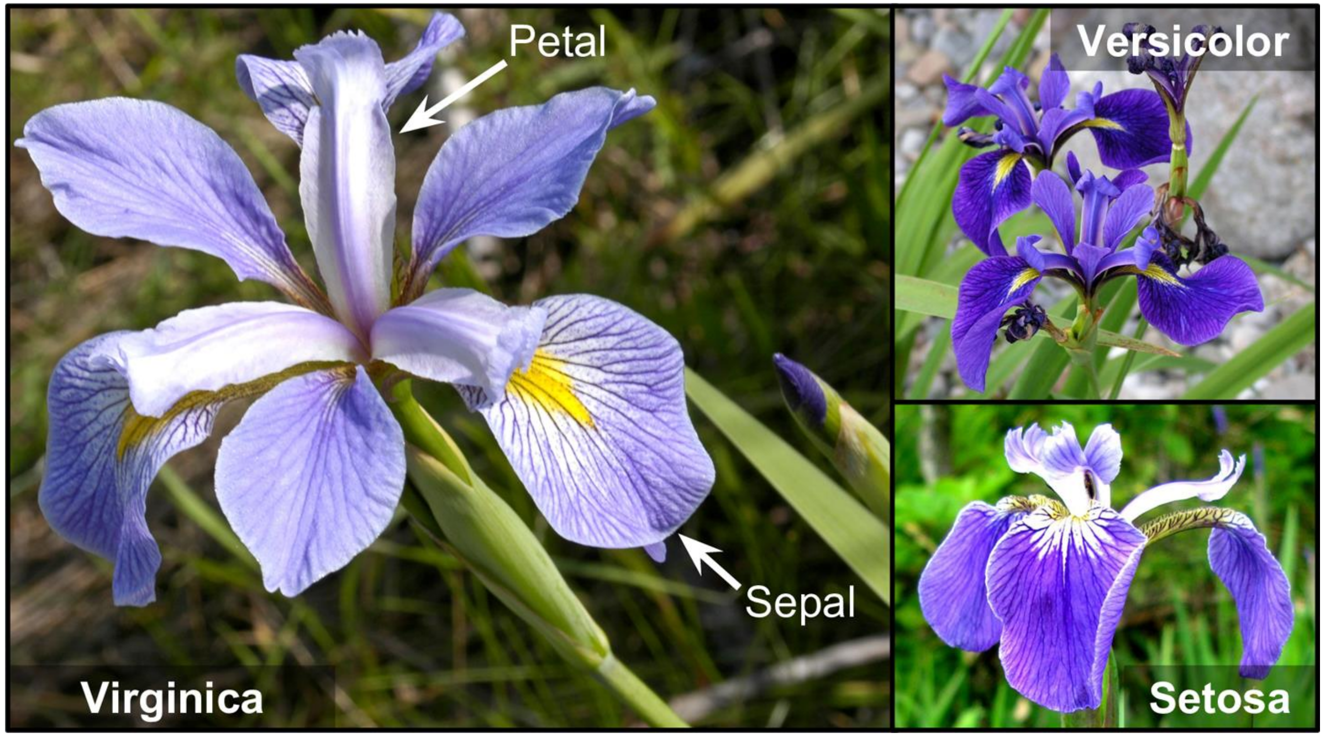


Training and Cost Function

- Logistic Regression model estimates probabilities and makes predictions.
- The objective of training is to set the parameter vector θ so that the model estimates high probabilities for positive instances (y = 1) and low probabilities for negative instances (y = 0)

$$c(\boldsymbol{\theta}) = \begin{cases} -\log(\hat{p}) & \text{if } y = 1 \\ -\log(1 - \hat{p}) & \text{if } y = 0 \end{cases}$$

Decision Boundaries

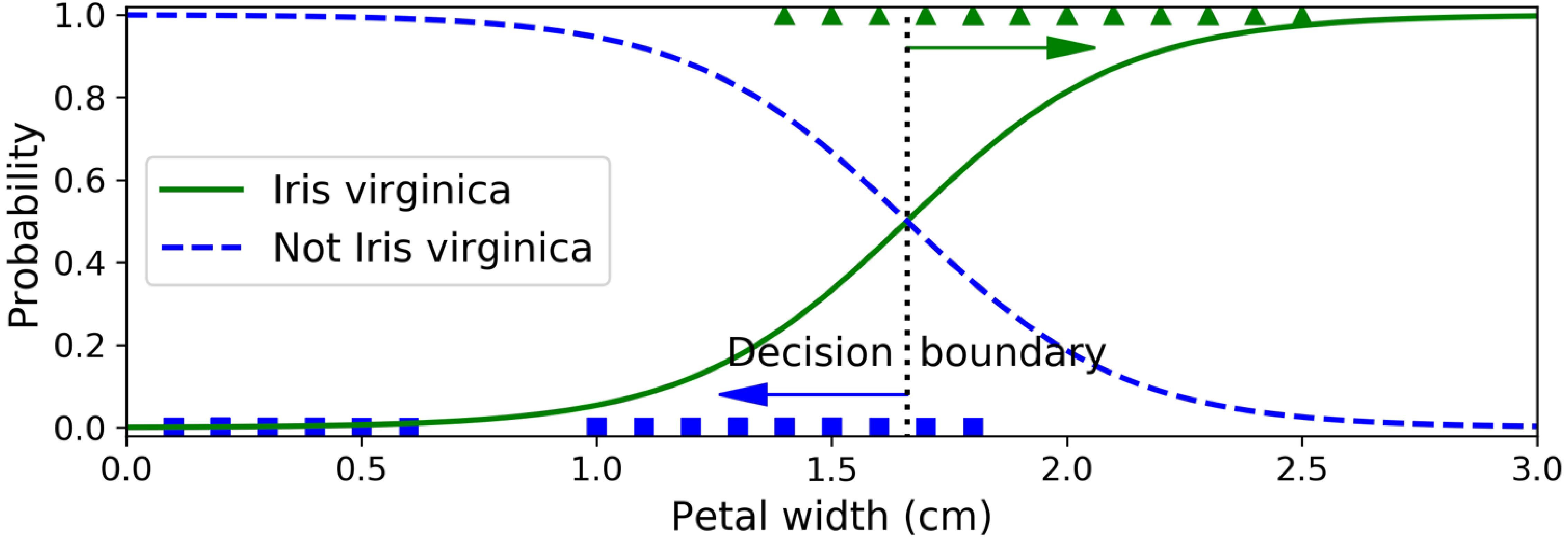


- Iris dataset
- contains the sepal and petal length and width of 150 iris flowers of three different species.
  - Iris setosa
  - Iris versicolor
  - Iris virginica

Training

```
#1. Load the data

from sklearn import datasets
iris = datasets.load_iris()
list(iris.keys())
['data', 'target', 'target_names', 'DESCR', 'feature_names', 'filename']
X = iris["data"][:, 3:]  # petal width
y = (iris["target"] == 2).astype(np.int)  # 1 if Iris virginica, else 0

#2. Logistic Regression model:
from sklearn.linear_model import LogisticRegression

log_reg = LogisticRegression()
log_reg.fit(X, y)

#3.The model's estimated probabilities
X_new = np.linspace(0, 3, 1000).reshape(-1, 1)
y_proba = log_reg.predict_proba(X_new)
plt.plot(X_new, y_proba[:, 1], "g-", label="Iris virginica")
plt.plot(X_new, y_proba[:, 0], "b--", label="Not Iris virginica")
```

Logistic Regression model



Predict

- The petal width of Iris virginica flowers (represented by triangles) ranges from 1.4 cm to 2.5 cm.
- The other iris flowers (represented by squares) generally have a smaller petal width, ranging from 0.1 cm to 1.8 cm.
- Above about 2 cm the classifier is highly confident that the flower is an Iris virginica (it outputs a high probability for that class)
- Below 1 cm it is highly confident that it is not an Iris virginica (high probability for the "Not Iris virginica" class).

Predict

- if you ask it to predict the class (using the predict() method rather than the predict_proba() method)
- Therefore, there is a decision boundary at around 1.6 cm where both probabilities are equal to 50%
  - if the petal width is higher than 1.6 cm, the classifier will predict that the flower is an Iris virginica
  - otherwise, it will predict that it is not (even if it is not very confident):

```
log_reg.predict([[1.7], [1.5]])

array([1, 0])
```

Softmax Regression

- The Logistic Regression model can be generalized to support multiple classes directly.
- This is called Softmax Regression, or Multinomial Logistic Regression.
- The Softmax Regression model first computes a score sk(x) for each class k, then estimates the probability of each class by applying the softmax function (also called the normalized exponential) to the scores.

$$s_k(\mathbf{x}) = \left(\boldsymbol{\theta}^{(k)}\right)^\top \mathbf{x}$$

Softmax function

In this equation:

- K is the number of classes.
- s(x) is a vector containing the scores of each class for the instance x.
- σ(s(x))k is the estimated probability that the instance x belongs to class k, given the scores of each class for that instance.

$$\hat{p}_k = \sigma(\mathbf{s}(\mathbf{x}))_k = \frac{\exp\left(s_k\left(\mathbf{x}\right)\right)}{\sum_{j=1}^{K} \exp\left(s_j\left(\mathbf{x}\right)\right)}$$

Softmax function

```
X = iris["data"][:, (2, 3)]  # petal length, petal width
y = iris["target"]

softmax_reg = LogisticRegression(multi_class="multinomial",solver="lbfgs", C=10)
softmax_reg.fit(X, y)

softmax_reg.predict([[5, 2]])
array([2])
softmax_reg.predict_proba([[5, 2]])
array([[6.38014896e-07, 5.74929995e-02, 9.42506362e-01]])
```

Softmax Regression decision boundaries