Introduction to Machine Learning
NPFL 054

http://ufal.mff.cuni.cz/course/npfl054

Barbora Hladká
hladka@ufal.mff.cuni.cz

Martin Holub
holub@ufal.mff.cuni.cz

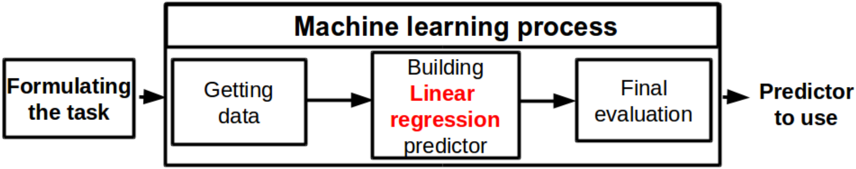
Charles University,
Faculty of Mathematics and Physics,
Institute of Formal and Applied Linguistics

Lecture #4

Outline

- **Linear regression**
  - `Auto` data set
- **Logistic regression**
  - `Auto` data set

Linear regression

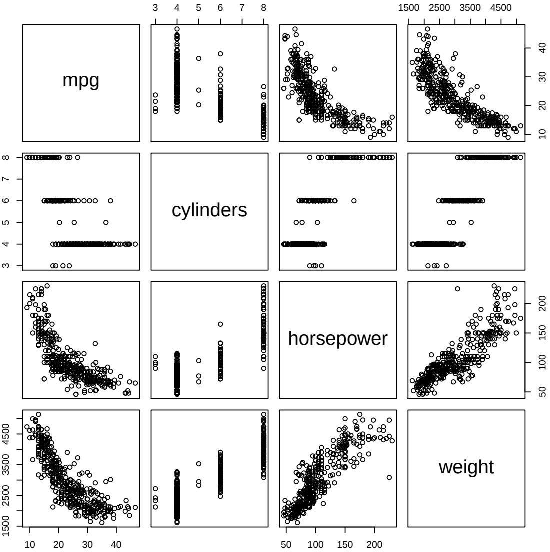


Dataset `Auto` from the `ISLR` package

392 instances on the following 9 features

mpg	Miles per gallon
cylinders	Number of cylinders between 4 and 8
displacement	Engine displacement (cu. inches)
horsepower	Engine horsepower
weight	Vehicle weight (lbs.)
acceleration	Time to accelerate from 0 to 60 mph (sec.)
year	Model year (modulo 100)
origin	Origin of car (1. American, 2. European, 3. Japanese)
name	Vehicle name

Dataset Auto from the ISLR package



Linear regression

Linear regression is a class of regression algorithms assuming that there is at least a linear dependence between a target attribute and features.

A target hypothesis f has a form of **linear function**

$$f(\mathbf{x}; \Theta) = \theta_0 + \theta_1 x_1 + \cdots + \theta_m x_m \tag{1}$$

– $\theta_0, \ldots, \theta_m$ are regression parameters

– we think of them as weights that determine how each feature affects the prediction

– **simple linear regression** if $m = 1$

Linear regression

Notation

$$\mathbf{y} = \begin{pmatrix} y_1 \\ \dots \\ y_n \end{pmatrix}, \ \Theta^\top = \begin{pmatrix} \theta_0 \\ \dots \\ \theta_m \end{pmatrix}, \ \mathbf{X} = \begin{pmatrix} 1 & x_{11} & \dots & x_{1m} \\ 1 & x_{21} & \dots & x_{2m} \\ \dots & \dots & \dots & \dots \\ 1 & x_{n1} & \dots & x_{nm} \end{pmatrix}$$

Now we can write $f(\mathbf{x}) = \Theta^\top \mathbf{x}$

Parameter interpretation

Numerical feature

θ_i is the average change in y for a unit change in A_i holding all other features fixed

Parameter interpretation

Categorical feature with k values

- replace it with $k-1$ dummy variables DA^1, DA^2, ..., DA^{k-1}

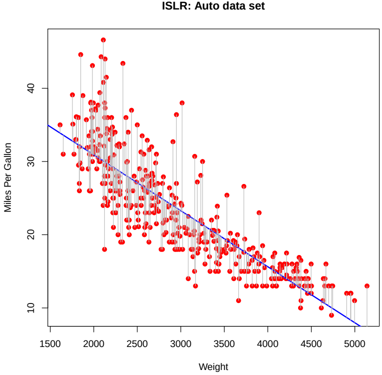
Example: run simple linear regression `mpg` $\sim$ `origin`

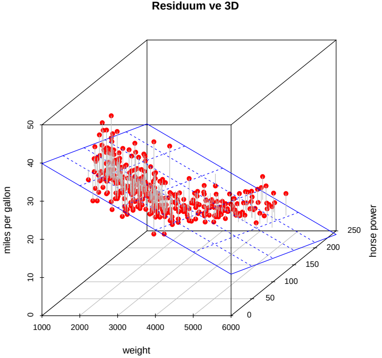
	DA^1	DA^1
American	0	0
European	1	0
Japanase	0	1

- $y = \theta_0 + \theta_1 \mathrm{DA}^1 + \theta_2 \mathrm{DA}^1$
- $y = \theta_0 + \theta_1$ if the car is European
- $y = \theta_0 + \theta_2$ if the car is Japanese
- $y = \theta_0$ if the car is American
- θ_0 as the average `mpg` for American cars
- θ_1 as the average difference in `mpg` between European and American cars
- θ_2 as the average difference in `mpg` between Japanese and American cars

Parameter estimates
Least Square Method

- residual $y_i - \hat{y}_i$, where $\hat{y}_i = \hat{f}(\mathbf{x}_i) = \hat{\Theta}^\top \mathbf{x}_i$
- **Loss function** Residual Sum of Squares $\mathrm{RSS}(\hat{\Theta}) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$





Parameter estimates
Least Square Method

Optimization problem

$$\Theta^\star = \mathrm{argmin}_\Theta \mathrm{RSS}(\Theta)$$

The `argmin` operator will give Θ for which $\mathrm{RSS}(\Theta)$ is minimal.

Parameter estimates
Least Square Method

Solving the optimization problem analytically

Normal Equations Calculus

Theorem

$\Theta^\star$ is a least square solution to $\mathbf{y} = \mathbf{X}\Theta^\top \Leftrightarrow \Theta^\star$ is a solution to the Normal equation $\mathbf{X}^\top\mathbf{X}\Theta = \mathbf{X}^\top\mathbf{y}$.

$$\Theta^\star = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Computational complexity of a $(m+1) \times (m+1)$ matrix inversion is $O(m+1)^3$:-(

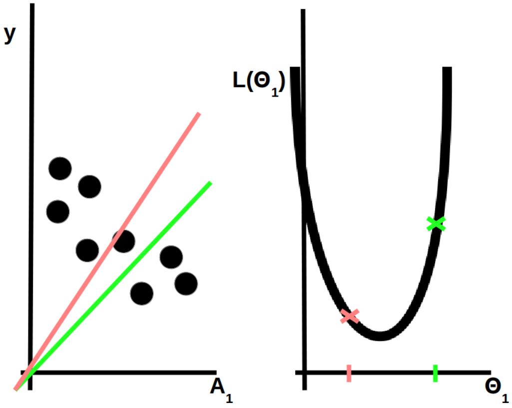
Parameter estimates
Least Square Method

Solving the optimization problem numerically

Gradient Descent Algorithm

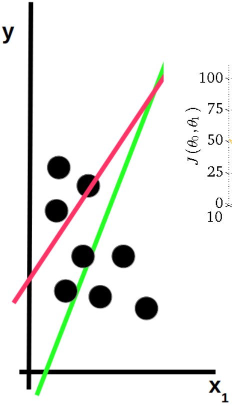
Gradient Descent Algorithm

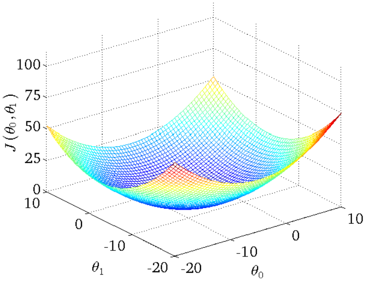
Assume: simple regression, $\theta_0 = 0$, $\theta_1 \neq 0$



Gradient Descent Algorithm

Assume: simple regression, $\theta_0 \neq 0$, $\theta_1 \neq 0$



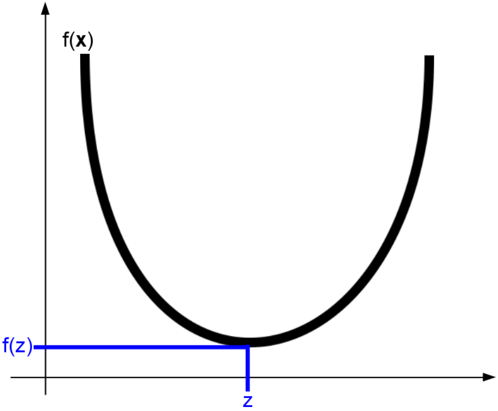


Credits: Andrew Ng

Note: In our notation $J(\theta_0, \theta_1) = L(\theta_0, \theta_1)$.

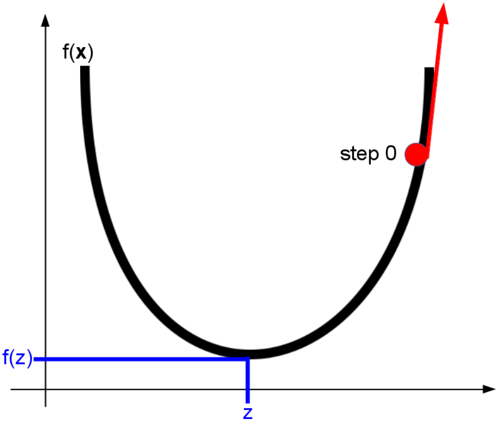
Gradient Descent Algorithm

Gradient descent algorithm is an optimization algorithm to find a local minimum of a function f.



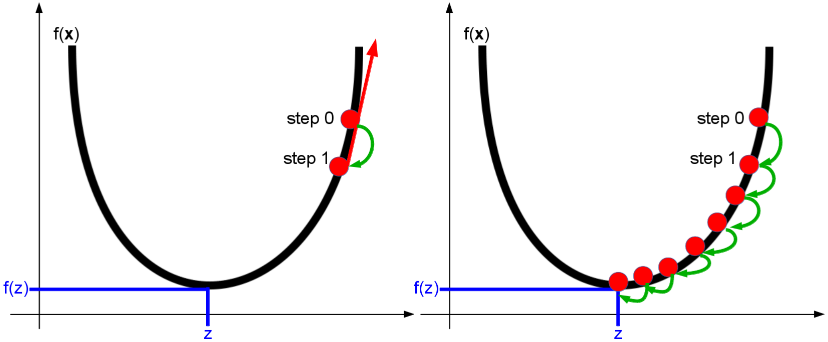
Gradient Descent Algorithm

1. Start with some $\mathbf{x}_0$.

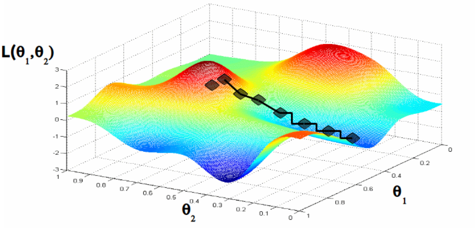


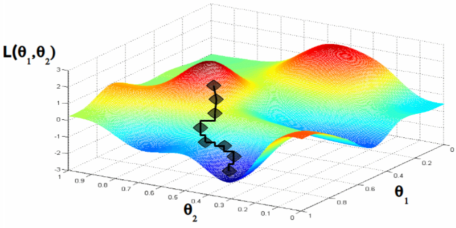
Gradient Descent Algorithm

2. Keep changing $\mathbf{x}_i$ to reduce $f(\mathbf{x}_i)$



Gradient Descent Algorithm





Credits: Andrew Ng

Gradient Descent Algorithm

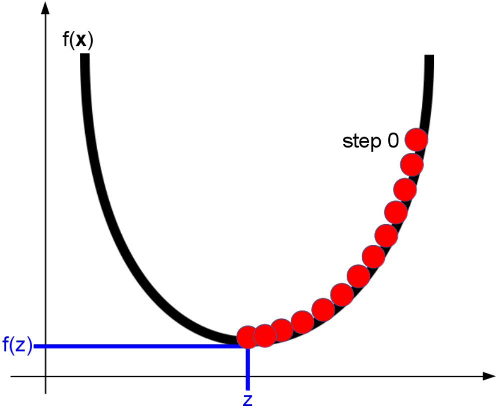
- We are seeking the solution to the minimum of a function $f(\mathbf{x})$. Given some initial value $\mathbf{x}_0$, we can change its value in many directions.
- What is the best direction to minimize f? We take the **gradient** ∇f of f

$$\nabla f(x_1, x_2, \ldots, x_m) = \langle \frac{\partial f(x_1, x_2, \ldots, x_m)}{\partial x_1}, \ldots, \frac{\partial f(x_1, x_2, \ldots, x_m)}{\partial x_m} \rangle$$

- Intuitively, the gradient of f at any point tells which direction is the steepest from that point and how steep it is. So we change $\mathbf{x}$ in the opposite direction to lower the function value.

Gradient Descent Algorithm

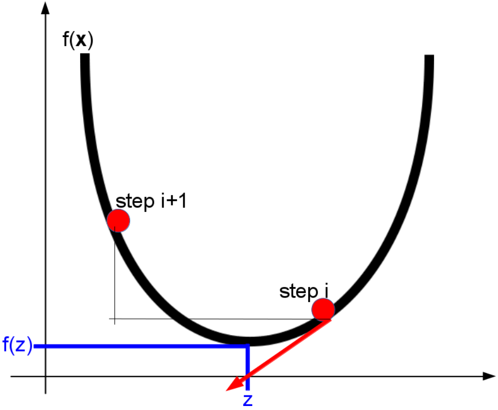
Choice of the step



If the step is too small, GDA can be slow.

Gradient Descent Algorithm

Choice of the step



If the step is too large, GDA can overshoot the minimum.
It mail fail to converge, or even diverge.
If α is too small, GDA can be slow.

Gradient Descent Algorithm

repeat until convergence $\{$

$$\Theta^{K+1} := \Theta^{K} - \alpha \nabla f(\Theta^{K})$$

$\}$

– α is a positive step-size hyperparameter

I.e. simultaneously update θ_j, $j = 1, \ldots m$

$$\theta_j^{K+1} := \theta_j^{K} - \alpha \frac{1}{n} \sum_{i=1}^{n} (f(\mathbf{x}_i; \Theta^{K}) - y_i) x_{ij}$$

For linear regression $f = RSS$

Linear regression
Gradient Descent Algorithm

RSS is a convex function, so there is no local optimum, just global minimum.

Polynomial regression

Polynomial regression is an extension of linear regression where the relationship between features and target value is modelled as a *d*-th order polynomial.

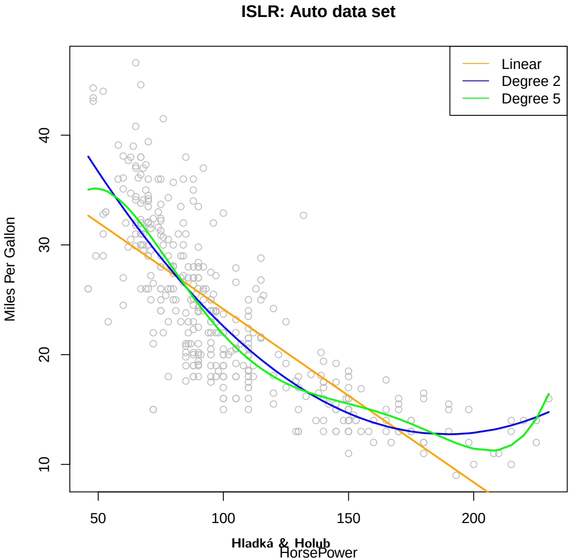
Simple regression
$y = \Theta_0 + \Theta_1 x_1$

Polynomial regression
$y = \Theta_0 + \Theta_1 x_1 + \Theta_2 x_1^2 + \dots \Theta_d x_1^d$

It is still a linear model with features $A_1, A_1^2, \dots, A_1^d$.

The *linear* in linear model refers to the hypothesis parameters, not to the features. Thus, the parameters $\Theta_0, \Theta_1, \dots, \Theta_d$ can be easily estimated using least squares linear regression.

Polynomial regression
Auto data set



Assessing the accuracy of the model

- **Coefficient of determination** R^2 measures the proportion of variance in a target value that is reduced by taking into account **x**

$$\mathrm{R}^2 = \frac{\mathrm{TSS} - \mathrm{RSS}}{\mathrm{TSS}} = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}}$$

where $\mathrm{TSS} = \sum_{i=1}^{n}(y_i - \overline{y})^2$; $R^2 \in (0, 1)$

- **Mean Squared Error** MSE

$$\mathrm{MSE} = \frac{1}{n} \cdot \mathrm{RSS}$$

Binary classification
Decision boundary

$Y = \{0, 1\}$

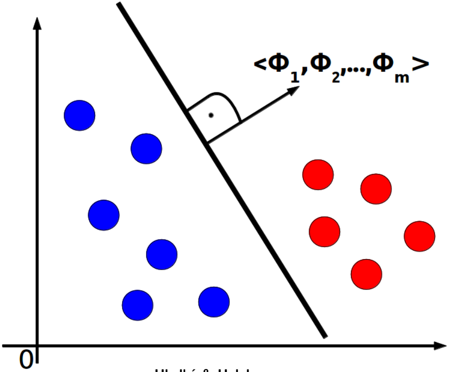
Decision boundary takes a form of function f and partitions a feature space into two sets, one for each class.

Binary classification
Hyperplane

Hyperplane is a linear decision boundary of the form

$$\Theta^\top \mathbf{x} = 0$$

where direction of $\langle \theta_1, \theta_2, \ldots, \theta_m \rangle$ is perpendicular to the hyperplane and θ_0 determines position of the hyperplane with respect to the origin



Binary classification
Hyperplane

- point if $m = 1$, line if $m = 2$, plane if $m = 3$, ...
- **separating hyperplane** separates data perfectly, i.e.

$$y_i \cdot (\theta_0 + \theta_1 x_1 + \dots + \theta_m x_m) > 0 \quad \forall i = 1, \dots, n$$

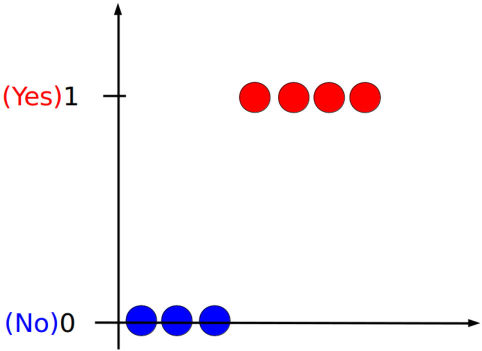
- we can use hyperplane for classification so that

$$f(\mathbf{x}) = \begin{cases} 1 \text{ if } \theta_0 + \theta_1 x_1 + \dots + \theta_m x_m < 0 \\ 0 \text{ if } \theta_0 + \theta_1 x_1 + \dots + \theta_m x_m \geq 0 \end{cases}$$

- **linear classifiers** classify examples using hyperplane.

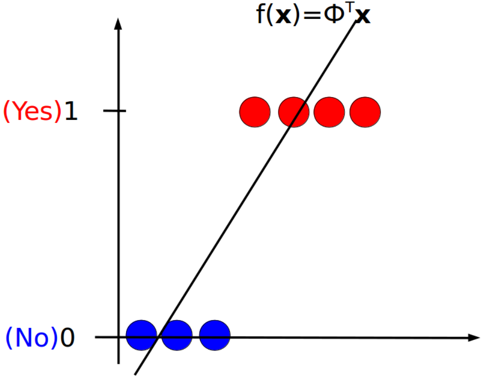
Binary classification
Can we use linear regression?

We are heading logistic regresession.



Binary classification
Can we use linear regression?

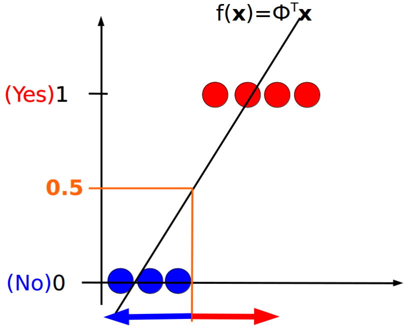
Fit the data with a linear function f



Binary classification
Can we use linear regression?

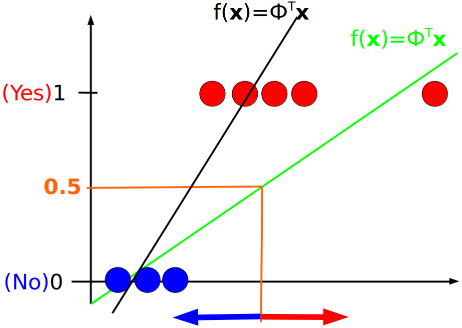
Classify

- if $f(\mathbf{x}) \geq 0.5$, predict 1
- if $f(\mathbf{x}) < 0.5$, predict 0



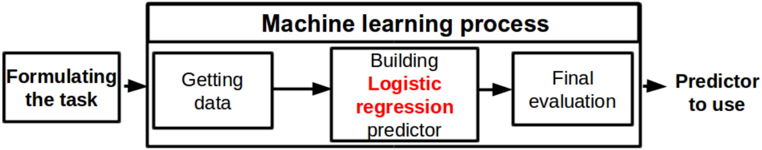
Binary classification
Can we use linear regression?

Add one more training instance



What to do if $f(\mathbf{x}) > 1$ or $f(\mathbf{x}) < 0$?

Logistic regression

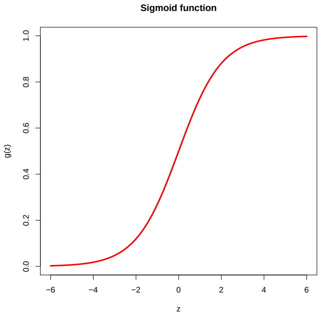


Logistic regression

Logistic regression is a classification algorithm.

Its target hypothesis f for a binary classification has a form of **sigmoid function**

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

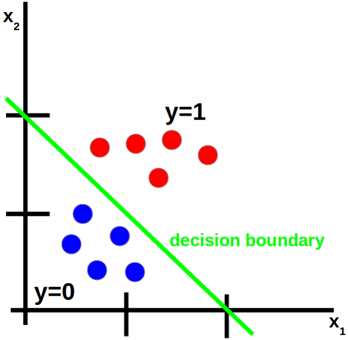


- $f(z) = g(z) = \frac{1}{1+e^{-z}}$
- $\lim_{z \to +\infty} g(z) = 1$
- $\lim_{z \to -\infty} g(z) = 0$

Classification rule

Predict a target value using $\hat{f}(\mathbf{x}; \hat{\Theta})$ so that

- if $\hat{f}(\mathbf{x}; \hat{\Theta}) \geq 0.5$, i.e. $\hat{\Theta}^\top \mathbf{x} \geq 0$, predict 1
- if $\hat{f}(\mathbf{x}; \hat{\Theta}) < 0.5$, i.e. $\hat{\Theta}^\top \mathbf{x} < 0$, predict 0



Modeling conditional probabilities

Logistic regression models the conditional probability $\Pr(y = 1|\mathbf{x}; \Theta)$

$$f(\mathbf{x}; \Theta) = \Pr(y = 1|\mathbf{x}; \Theta) = \frac{1}{1 + e^{-\Theta^\top \mathbf{x}}}$$

Algebraic manipulation results in

$$\frac{\Pr(y = 1|\mathbf{x}; \Theta)}{1 - \Pr(y = 1|\mathbf{x}; \Theta)} = e^{\Theta^\top \mathbf{x}} \in (0, +\infty)$$

Take logarithm

$$\ln \frac{f(\mathbf{x}; \Theta)}{1 - f(\mathbf{x}; \Theta)} = \Theta^\top \mathbf{x} \in (-\infty, +\infty)$$

Modeling conditional probabilities

- $\text{odds} = \Pr(y = 1|\mathbf{x}; \Theta) / \Pr(y = 0|\mathbf{x}; \Theta)$
- $\text{log-odds} = \text{logit}$
- $\text{logit}(p) = \ln \frac{p}{1-p}$
- logit is linear in $\mathbf{x}$

Parameter interpretation
Numerical features

θ_i gives an average change in $\mathrm{logit}(f(\mathbf{x}))$ with one-unit change in A_i holding all other features fixed

Parameter interpretation
Binary features

Example:

disease	female		
	0 (male)	1 (female)	Total
no	74	77	151
yes	17	32	49
Total	91	109	200

- the odds of having the disease for male:
  $\Pr(\text{disease} = \textit{yes} | \text{female} = 0) / \Pr(\text{disease} = \textit{no} | \text{female} = 0) = \frac{17/91}{74/91} = 0.23$
- the odds of having the disease for female:
  $\Pr(\text{disease} = \textit{yes} | \text{female} = 1) / \Pr(\text{disease} = \textit{no} | \text{female} = 1) = \frac{32/109}{77/109} = 0.42$
- the ratio of the odds for female to the odds for male $0.42/0.23 = 1.81$, i.e. the odds for female are about 81% higher than the odds for males

Parameter interpretation
Binary features

$\ln \frac{p}{1-p} = \theta_0 + \theta_1 * \text{female}$

If $\text{female} == 0$

- $p = p_1 \rightarrow \frac{p_1}{1-p_1} = e^{\theta_0}$
- the intercept θ_0 is the log odds for men

If $\text{female} == 1$

- $p = p_2 \rightarrow \frac{p_2}{1-p_2} = e^{\theta_0+\theta_1}$
- odds ratio $= \frac{p_2}{1-p_2} / \frac{p_1}{1-p_1} = e^{\theta_1}$
- the parameter e^{θ_1} is the of odds ratio between women and men

Assume the output of logistic regression $\theta_0 = -1.471$, $\theta_1 = 0.593$. Then relate the odds for males and females and the parameters: $-1.471 = \ln 0.23$, $0.593 = \ln 1.81$

Parameter estimates

- **Loss function**

$$\mathrm{L}(\boldsymbol{\Theta}) = -\sum_{i=1}^{n} y_i \log \mathrm{P}(y_i|\mathbf{x_i};\Theta) + (1-y_i)\log(1-\mathrm{P}(y_i|\mathbf{x_i};\Theta))$$

See Maximum Likelihood Principle for derivation of this loss function.

- **Optimization problem**

$$\Theta^\star = \mathrm{argmin}_{\Theta}\mathrm{L}(\Theta)$$

Parameter estimates
Gradient Descent Algorithm

repeat until convergence $\{$

$$\Theta^{K+1} := \Theta^K - \alpha \nabla f(\Theta^K)$$

$\}$

– α is a positive step-size hyperparameter

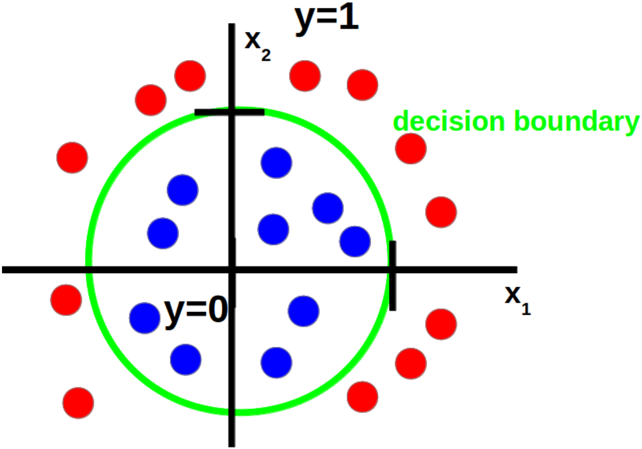
I.e. simultaneously update θ_j, $j = 1, \ldots m$

$$\theta_j^{K+1} := \theta_j^K - \alpha \frac{1}{n} \sum_{i=1}^{n} (f(\mathbf{x}_i; \Theta^K) - y_i) x_{ij}$$

Non-linear decision boundary

- Let $f(\mathbf{x}) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$ (a higher degree polynomial)
- Assume $\theta_0 = -1$, $\theta_1 = 0, \theta_2 = 0$, $\theta_3 = 1, \theta_4 = 1$
- Predict $y = 1$ if $-1 + x_1^2 + x_2^2 \geq 0$, i.e. $x_1^2 + x_2^2 \geq 1$

Non-linear decision boundary



Logistic regression
Summary

Classification of $\mathbf{x}$ by $\hat{f}^{\star}$

1. Project $\mathbf{x}$ onto $\hat{\Theta}^{\star}$ to convert it into a real number z in the range $\langle -\infty, +\infty \rangle$
   - i.e. $z = {\hat{\Theta}^{\star}}^{\top} \mathbf{x}$
2. Map z to the range $\langle 0, 1 \rangle$ using the sigmoid function $g(z) = 1/(1 + e^{-z})$
3. Classify $\mathbf{x}$ using a classification rule

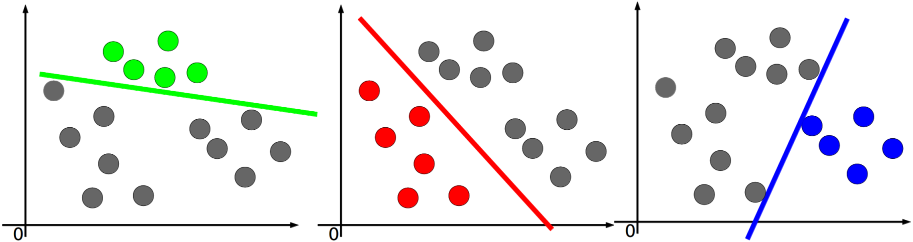
Multi-class classification

$|Y| = N$, $N \geq 3$

- **One-to-all**
  - train N predictors f_k for the pair k-th class and $\{1, \cdots, N\} \setminus \{k\}$ classes
  - classify $\mathbf{x}$ into the class $k^\star = \mathrm{argmax}_k f_k(\mathbf{x})$
- **One-to-one**
  - train $\binom{N}{2}$ classifiers f_i
  - classify $\mathbf{x}$ into the class $k^\star = \max_{k=1,\ldots,N} \sum_{i=1}^{\binom{N}{2}} \delta(f_i(\mathbf{x}) = k)$

Logistic regression
Multi-class classification

One-to-all



Summary of Examination Requirements

- Linear regression, simple linear regression, polynomial regression
- Parameter interpretation
- Least Square Method
- Gradient Descent Algorithm
- Coefficient of Determination, Mean Squared Error
- Decision boundary, classification rule
- Logistic regression, sigmoid function, probabilistic formulation
- Parameter interpretation
- Multi-class classification