Widzenie Komputerowe - Klasyfikacja

Wykład 11.

Magdalena Mazur-Milecka

Katedra Inżynierii Biomedycznej, WETI, PG

11 kwietnia 2019

Spis treści

1. Wprowadzenie do klasyfikacji
2. kNN - k Nearest Neighbours
3. Regresja logistyczna
4. Naive Bayes
5. SVM
6. Drzewa Decyzyjne
7. Random Forest

Wprowadzenie

Klasyfikacja - realny cel rozpoznania, widzenia komputerowego.
Oszacowanie prawdopodobieństwa tego, że obraz analizowany jest obiektem, o którym posiadamy informacje w systemie.
Dąży się do osiągnięcia 100% pewności co do zaklasyfikowania analizowanego obiektu.

Wprowadzenie

Czym jest klasyfikacja w odniesieniu do umiejętności człowieka?

Wprowadzenie

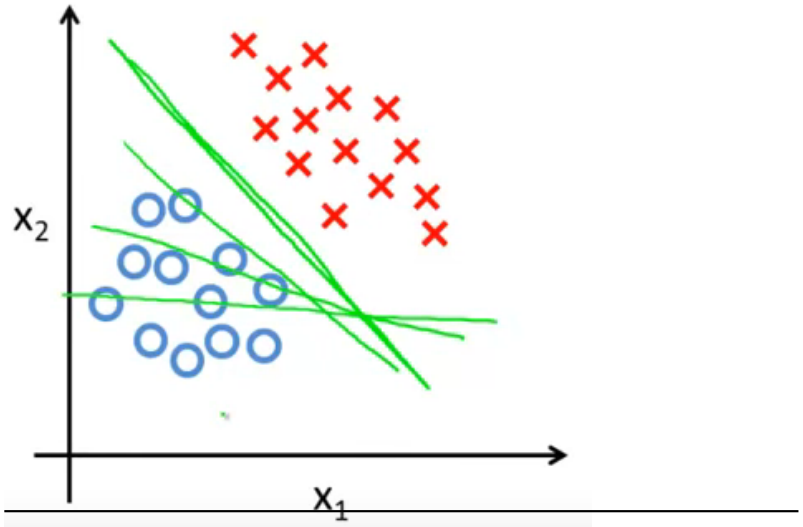
Podział klasyfikacji obrazów:

Stanford OpenClassroom, Andrew Ng

Wprowadzenie

Podział klasyfikacji obrazów:

- podejście całościowe - wzorce i obiekt reprezentowane są przez wektor cech. Wektor cech umieszcza obiekt w przestrzeni (o wymiarze takim, jak ilość cech). Przestrzeń cech jest podzielona na obszary decyzyjne. Przykładem jest regresja logistyczna.

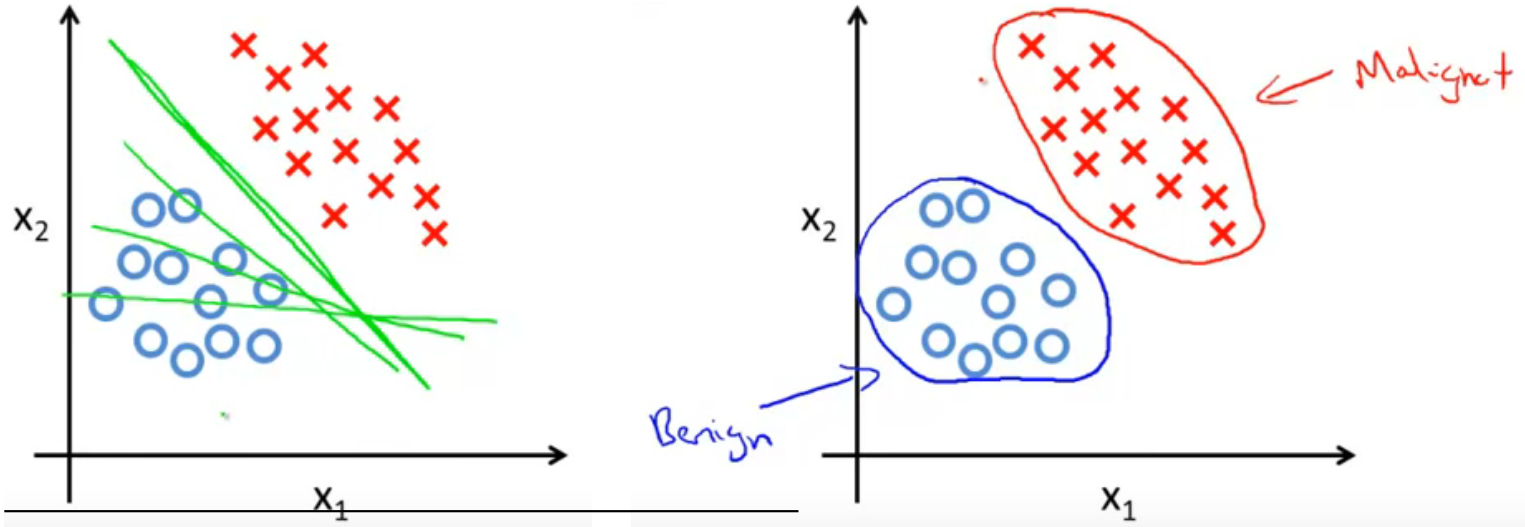


Stanford OpenClassroom, Andrew Ng

Wprowadzenie

Podział klasyfikacji obrazów:

- podejście całościowe - wzorce i obiekt reprezentowane są przez wektor cech. Wektor cech umieszcza obiekt w przestrzeni (o wymiarze takim, jak ilość cech). Przestrzeń cech jest podzielona na obszary decyzyjne. Przykładem jest regresja logistyczna.
- podejście strukturalne (Generative) - wzorce reprezentowane są przez elementy bazowe oraz ich relacje, budowany jest model klas. Przykładem jest Naive Bayes.



Stanford OpenClassroom, Andrew Ng

Wprowadzenie

Podział klasyfikacji obrazów:

- podejście całościowe - wzorce i obiekt reprezentowane są przez wektor cech. Wektor cech umieszcza obiekt w przestrzeni (o wymiarze takim, jak ilość cech). Przestrzeń cech jest podzielona na obszary decyzyjne. Przykładem jest regresja logistyczna.
- podejście strukturalne (Generative) - wzorce reprezentowane są przez elementy bazowe oraz ich relacje, budowany jest model klas. Przykładem jest Naive Bayes.

Podejście strukturalne jest bardziej złożone, ale pozwala na analizę bardziej złożonych obrazów

---

Stanford OpenClassroom, Andrew Ng

Miary oceny klasyfikatorów

Miary oceny klasyfikatorów

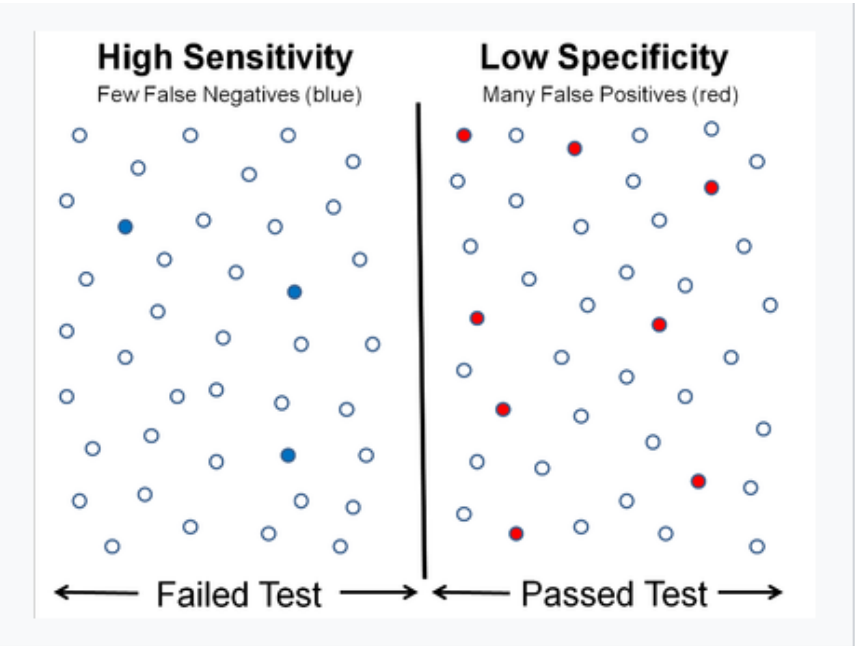
- Czułość (sensitivity) $= \frac{TP}{TP+FN}$

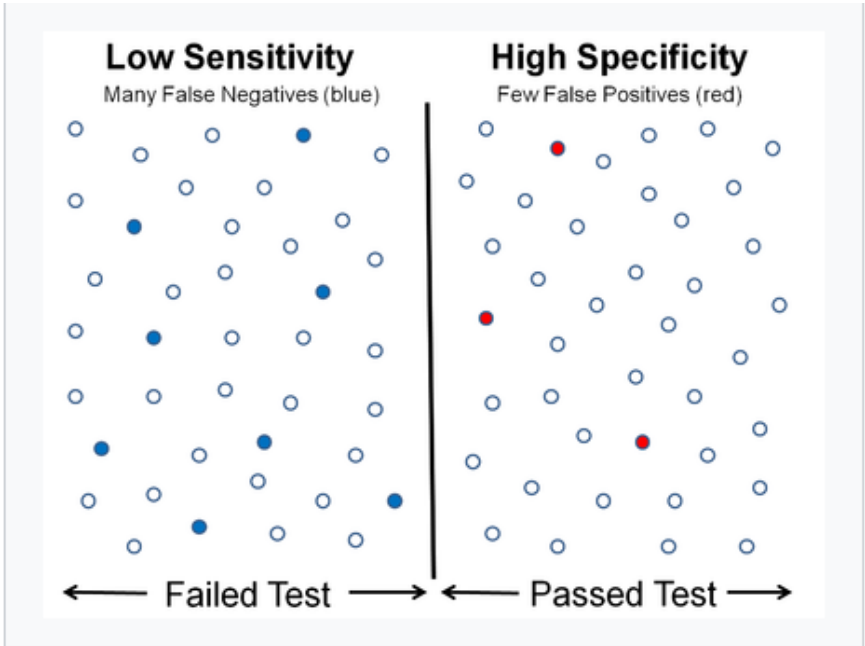
Miary oceny klasyfikatorów

- Czułość (sensitivity) = $\frac{TP}{TP+FN}$
- Specyficzność (specificity, precision) = $\frac{TN}{TN+FP}$

Miary oceny klasyfikatorów

- Czułość (sensitivity) $= \frac{TP}{TP+FN}$
- Specyficzność (specificity, precision) $= \frac{TN}{TN+FP}$





Miary oceny klasyfikatorów

- Czułość (sensitivity) = $\frac{TP}{TP+FN}$
- Specyficzność (specificity, precision) = $\frac{TN}{TN+FP}$
- Ilość pozytywnej predykcji = $\frac{TP}{TP+FP}$

Miary oceny klasyfikatorów

- Czułość (sensitivity) = $\frac{TP}{TP+FN}$
- Specyficzność (specificity, precision) = $\frac{TN}{TN+FP}$
- Ilość pozytywnej predykcji = $\frac{TP}{TP+FP}$
- Ilość negatywnej predykcji = $\frac{TN}{TN+FN}$

Miary oceny klasyfikatorów

- Czułość (sensitivity) = $\frac{TP}{TP+FN}$
- Specyficzność (specificity, precision) = $\frac{TN}{TN+FP}$
- Ilość pozytywnej predykcji = $\frac{TP}{TP+FP}$
- Ilość negatywnej predykcji = $\frac{TN}{TN+FN}$
- Dokładność (accuracy) = $\frac{TP+TN}{TP+TN+FP+FN}$

Trenowanie klasyfikatorów

1. **Leve-one-out** - w przypadkach małej ilości przypadków. Jeżeli liczba przypadków = N, trenowanie odbywa się na N-1 przypadkach, testowanie na 1, następnie wybierany jest kolejny pojedynczy przypadek i uczenie jest powtarzane. Na końcu wynik zostaje uśredniony.

Trenowanie klasyfikatorów

1. **Leve-one-out** - w przypadkach małej ilości przypadków. Jeżeli liczba przypadków = N, trenowanie odbywa się na N-1 przypadkach, testowanie na 1, następnie wybierany jest kolejny pojedynczy przypadek i uczenie jest powtarzane. Na końcu wynik zostaje uśredniony.
2. **Powtarzalne losowanie podpróby** - w przypadkach większej ilości przypadków.
   Losowanie zbioru testowego i uczącego z zadanym parametrem jest powtarzane n-krotnie. Na końcu wynik zostaje uśredniony.

Spis treści

1. Wprowadzenie do klasyfikacji
2. **kNN - k Nearest Neighbours**
3. Regresja logistyczna
4. Naive Bayes
5. SVM
6. Drzewa Decyzyjne
7. Random Forest

kNN - k najbliższych sąsiadów

Założenia:

- Dany jest zbiór uczący(?) zawierający obserwacje, które składają się z wektora cech $(X_1, ..., X_n)$ oraz wartość zmiennej objaśnianej Y,

Wikipedia

kNN - k najbliższych sąsiadów

Założenia:

- Dany jest zbiór uczący(?) zawierający obserwacje, które składają się z wektora cech $(X_1, ..., X_n)$ oraz wartość zmiennej objaśnianej Y,
- Dana jest obserwacja C opisana wektorem cech, dla której chcemy prognozować wartość Y

Wikipedia

kNN - k najbliższych sąsiadów

Założenia:

- Dany jest zbiór uczący(?) zawierający obserwacje, które składają się z wektora cech $(X_1, ..., X_n)$ oraz wartość zmiennej objaśnianej Y,
- Dana jest obserwacja C opisana wektorem cech, dla której chcemy prognozować wartość Y

Algorytm polega na:

Wikipedia

kNN - k najbliższych sąsiadów

Założenia:

- Dany jest zbiór uczący(?) zawierający obserwacje, które składają się z wektora cech $(X_1, ..., X_n)$ oraz wartość zmiennej objaśnianej Y,
- Dana jest obserwacja C opisana wektorem cech, dla której chcemy prognozować wartość Y

Algorytm polega na:

1. porównaniu wartości cech dla obserwacji C z wartościami cech dla obserwacji ze zbioru uczącego (usytuowanie obserwacji C w przestrzeni cech uczących),

Wikipedia

kNN - k najbliższych sąsiadów

Założenia:

- Dany jest zbiór uczący(?) zawierający obserwacje, które składają się z wektora cech $(X_1, ..., X_n)$ oraz wartość zmiennej objaśnianej Y,
- Dana jest obserwacja C opisana wektorem cech, dla której chcemy prognozować wartość Y

Algorytm polega na:

1. porównaniu wartości cech dla obserwacji C z wartościami cech dla obserwacji ze zbioru uczącego (usytuowanie obserwacji C w przestrzeni cech uczących),
2. wyborze parametru k, który powinien być dobierany odpowiednio do ilości klas,

Wikipedia

kNN - k najbliższych sąsiadów

Założenia:

- Dany jest zbiór uczący(?) zawierający obserwacje, które składają się z wektora cech $(X_1, ..., X_n)$ oraz wartość zmiennej objaśnianej Y,
- Dana jest obserwacja C opisana wektorem cech, dla której chcemy prognozować wartość Y

Algorytm polega na:

1. porównaniu wartości cech dla obserwacji C z wartościami cech dla obserwacji ze zbioru uczącego (usytuowanie obserwacji C w przestrzeni cech uczących),
2. wyborze parametru k, który powinien być dobierany odpowiednio do ilości klas,
3. sprawdzeniu wartości Y dla k najbliższych obserwacji ze zbioru uczącego,

---

Wikipedia

kNN - k najbliższych sąsiadów

Założenia:

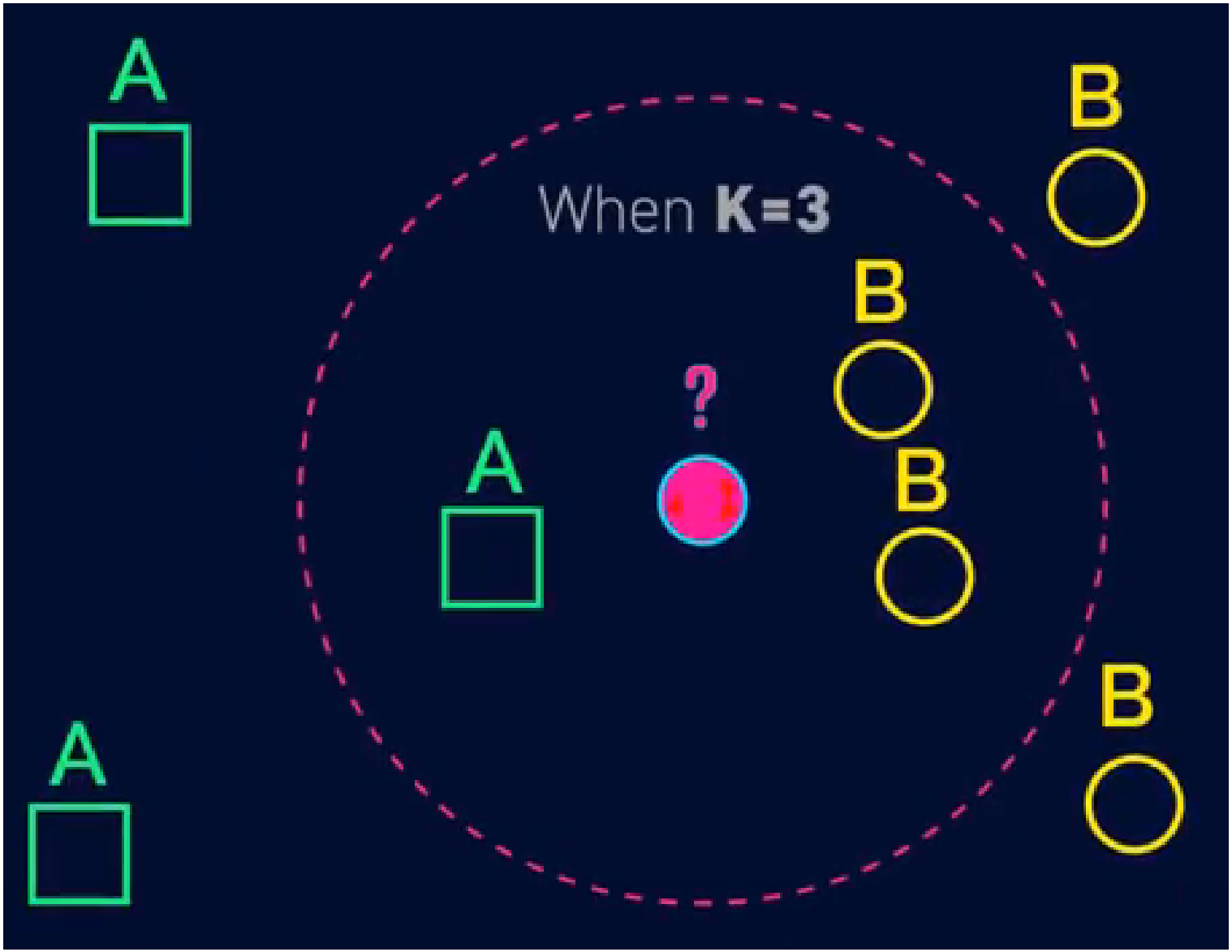
- Dany jest zbiór uczący(?) zawierający obserwacje, które składają się z wektora cech $(X_1, ..., X_n)$ oraz wartość zmiennej objaśnianej Y,
- Dana jest obserwacja C opisana wektorem cech, dla której chcemy prognozować wartość Y

Algorytm polega na:

1. porównaniu wartości cech dla obserwacji C z wartościami cech dla obserwacji ze zbioru uczącego (usytuowanie obserwacji C w przestrzeni cech uczących),
2. wyborze parametru k, który powinien być dobierany odpowiednio do ilości klas,
3. sprawdzeniu wartości Y dla k najbliższych obserwacji ze zbioru uczącego,
4. uśrednieniu (lub wybranie większości) wartości Y dla wybranych obserwacji, co daje prognozę.

Wikipedia

kNN



Analytics India Magazine

kNN - zalety i wady

- Uczenie nie jest skomplikowane,
- Predykcja jest czasochłonna,
- Algorytm jest prosty,
- Zwiększenie k powoduje zmniejszenie wydajności,
- Zmniejszenie k powoduje zwiększenie podatności na błędy (zmniejszenie dokładności),
- Algorytm nie zawiera modelu danych.

Spis treści

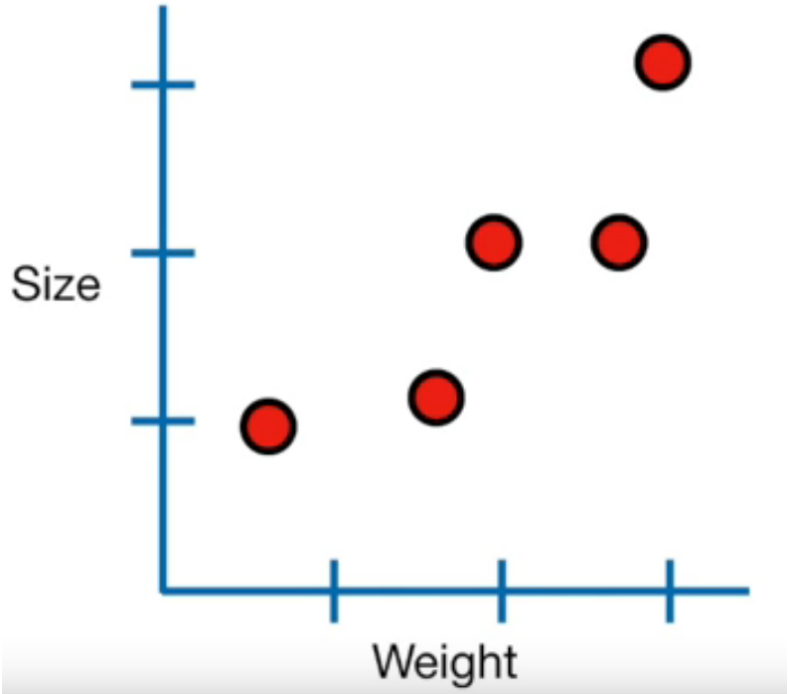
1. Wprowadzenie do klasyfikacji
2. kNN - k Nearest Neighbours
3. **Regresja logistyczna**
4. Naive Bayes
5. SVM
6. Drzewa Decyzyjne
7. Random Forest

Regresja logistyczna

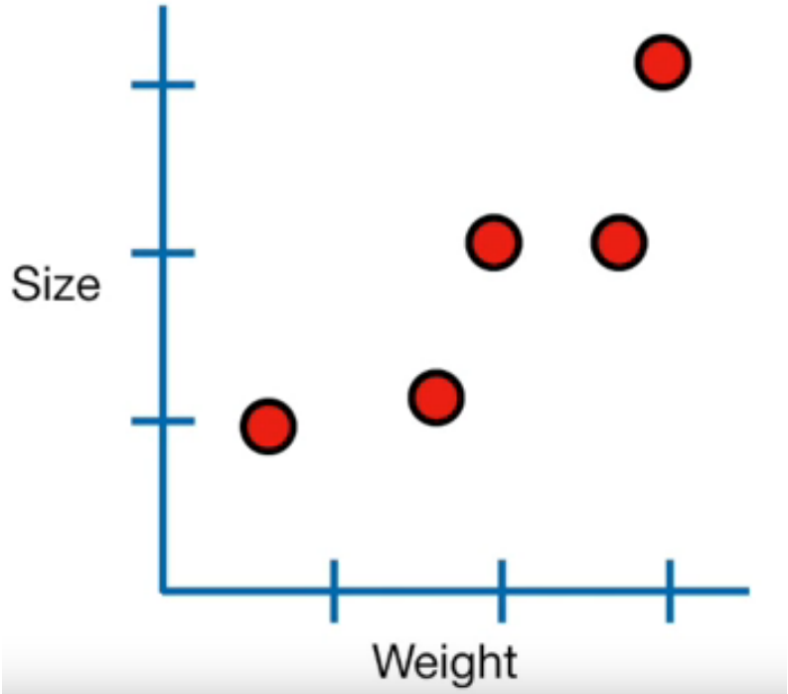
Jest to szczególny przypadek uogólnionej regresji liniowej, gdy zmienna zależna jest dychotomiczna - przyjmuje tylko dwie wartości (sukces, porażka)

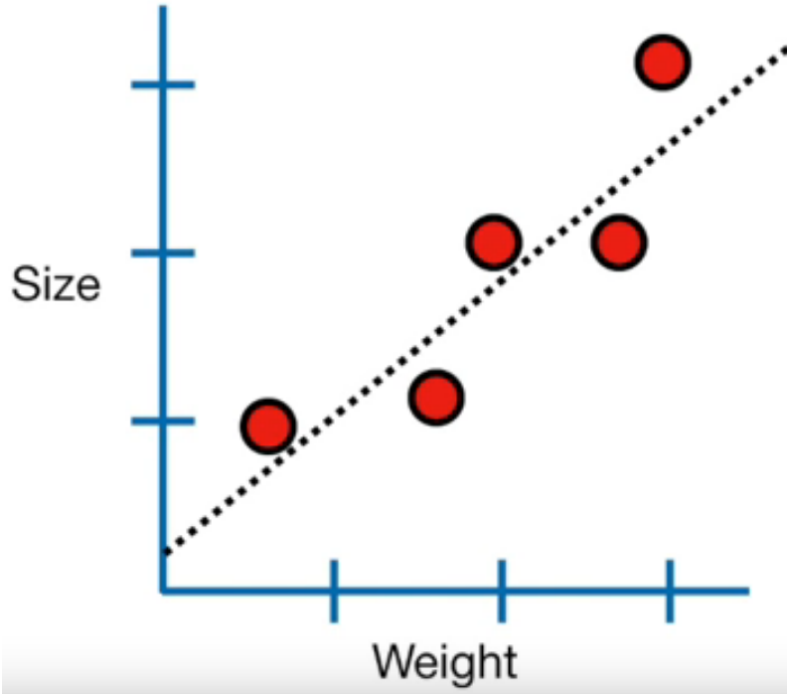
$$y \in 0, 1$$

Regresja liniowa

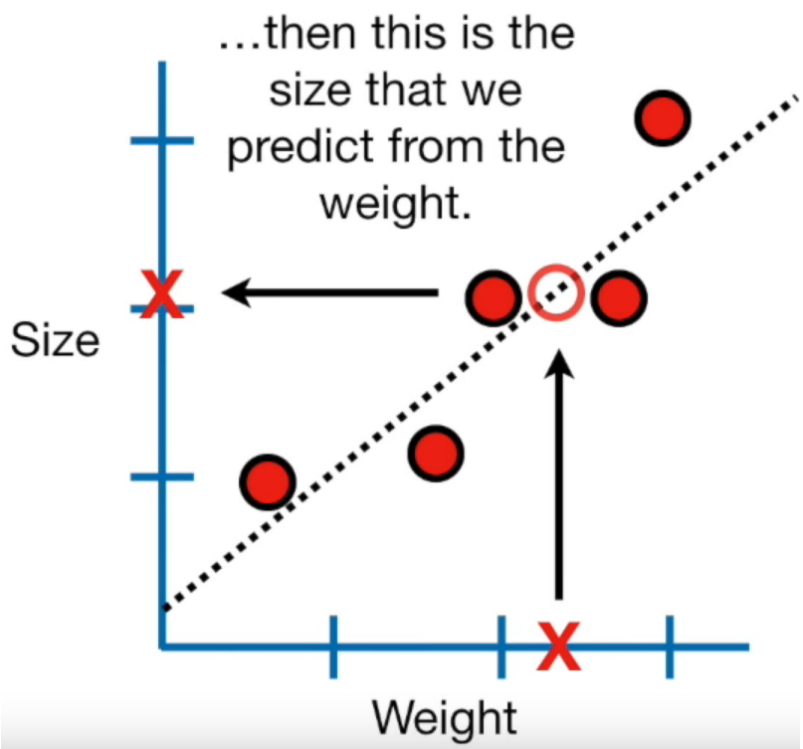


Regresja liniowa





Regresja liniowa

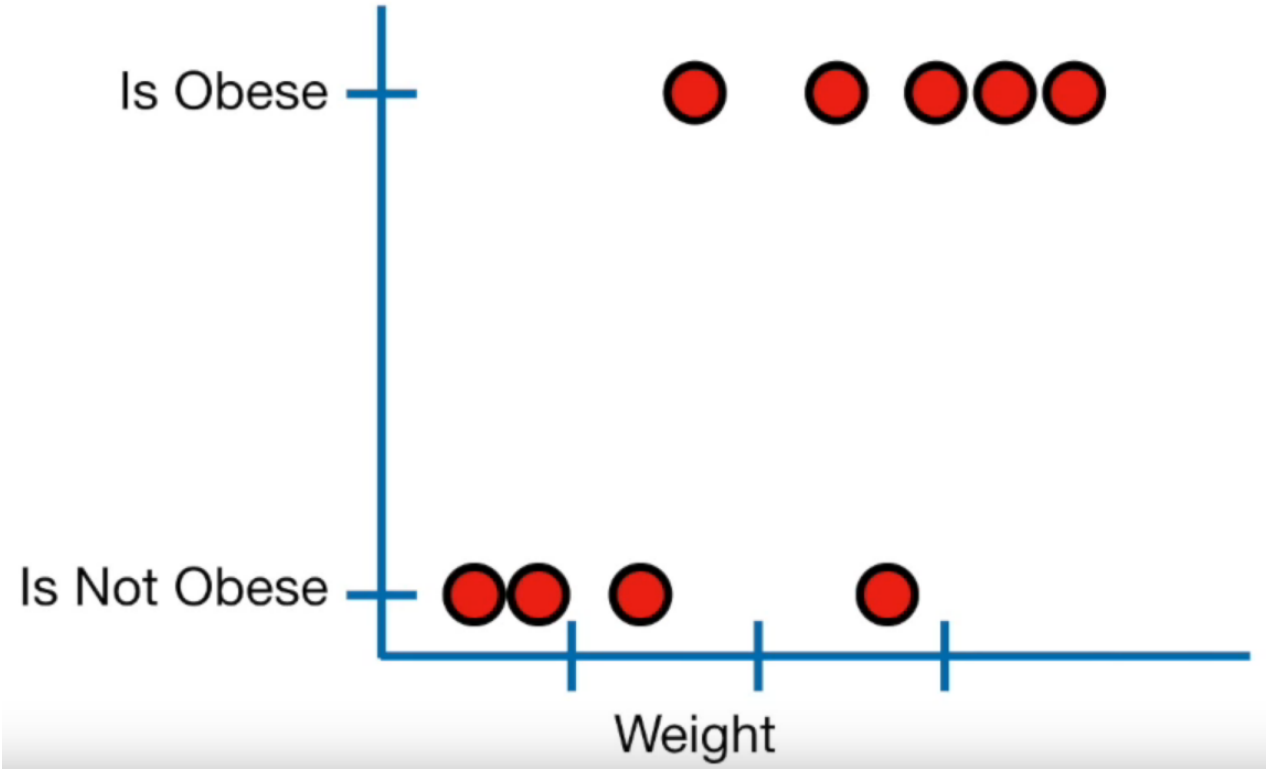


$$h_\theta(x) = \theta^T x$$

h - hipoteza θ - parametr

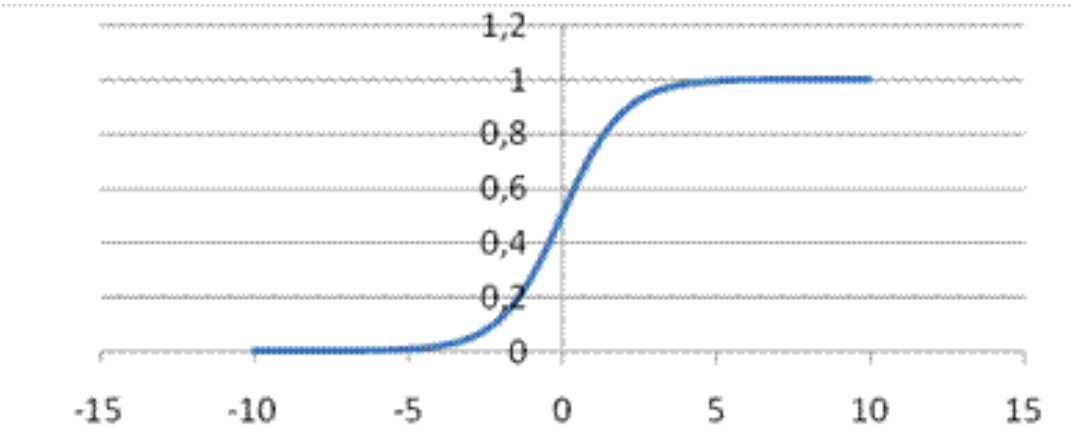
Regresja logistyczna

Jest to szczególny przypadek uogólnionej regresji liniowej, gdy **zmienna zależna jest dychotomiczna** - przyjmuje tylko dwie wartości (sukces, porażka)



StatQuest

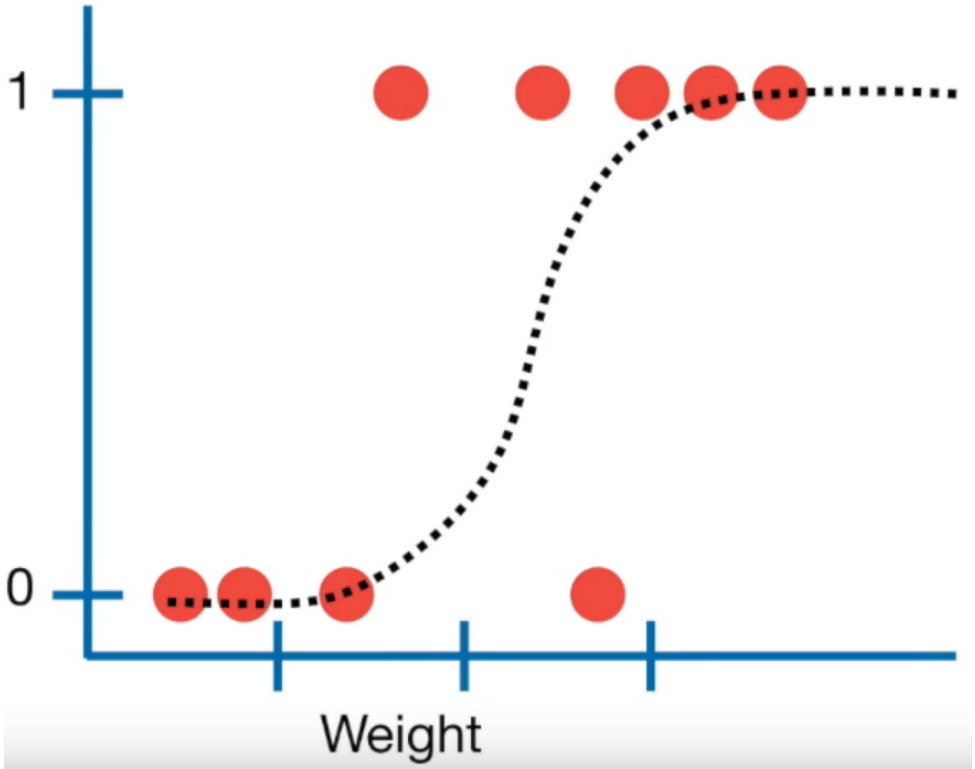
Regresja logistyczna



funkcja sigmoidalna, logistyczna

$$g(x) = \frac{1}{1+e^{-x}}$$

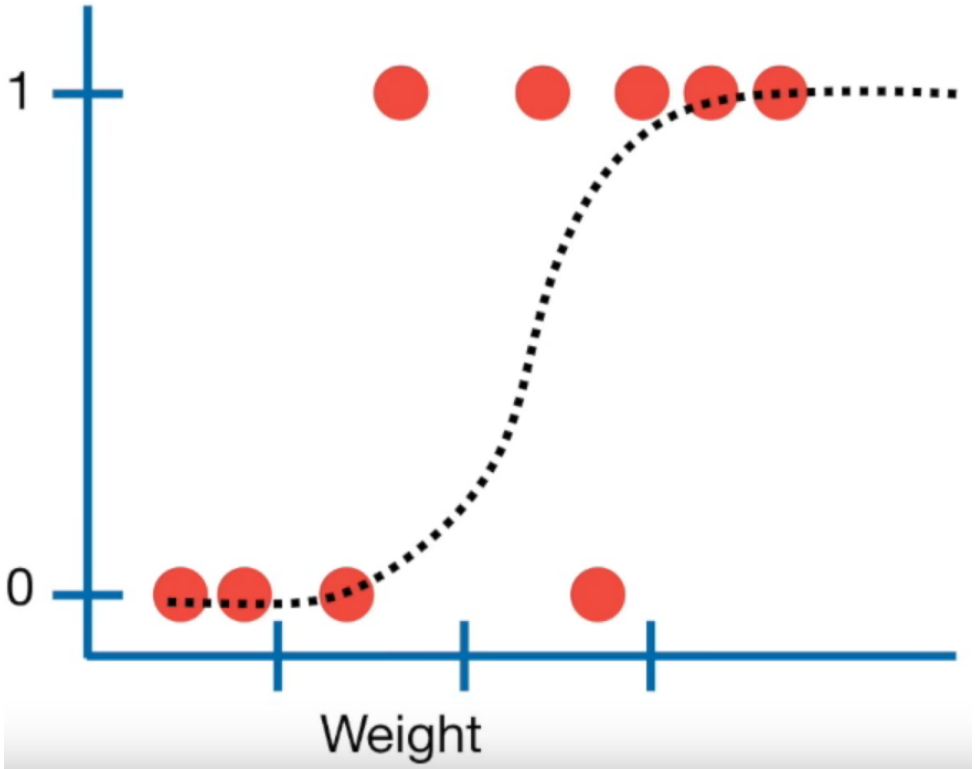
Regresja logistyczna



$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

StatQuest

Regresja logistyczna

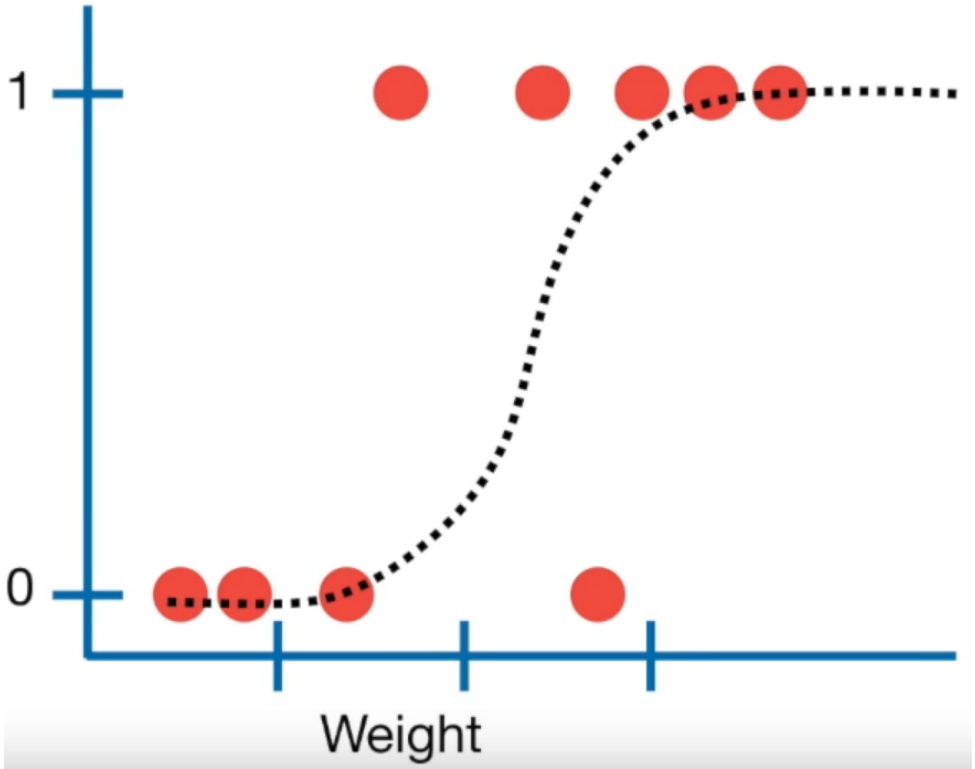


$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

$h_\theta(x)$ - estymowane prawdopodobieństwo tego, że y=1 dla danego x

StatQuest

Regresja logistyczna



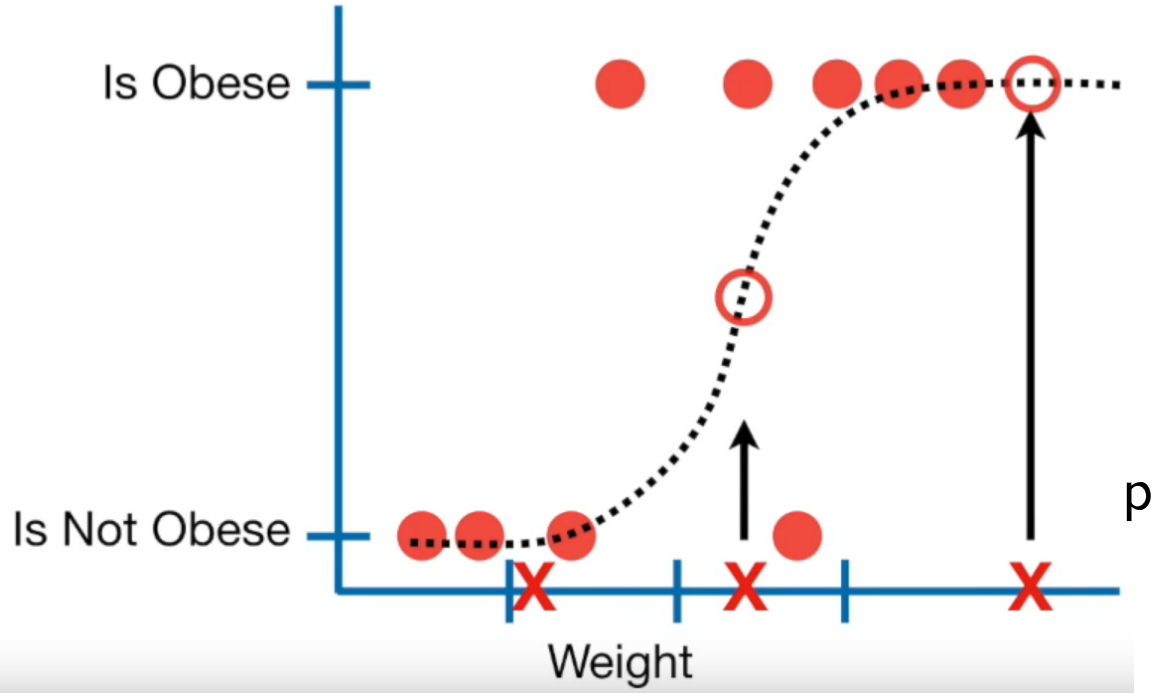
$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

$$0 \leqslant h_\theta(x) \leqslant 1$$

$h_\theta(x)$ - estymowane prawdopodobieństwo tego, że y=1 dla danego x

StatQuest

Regresja logistyczna

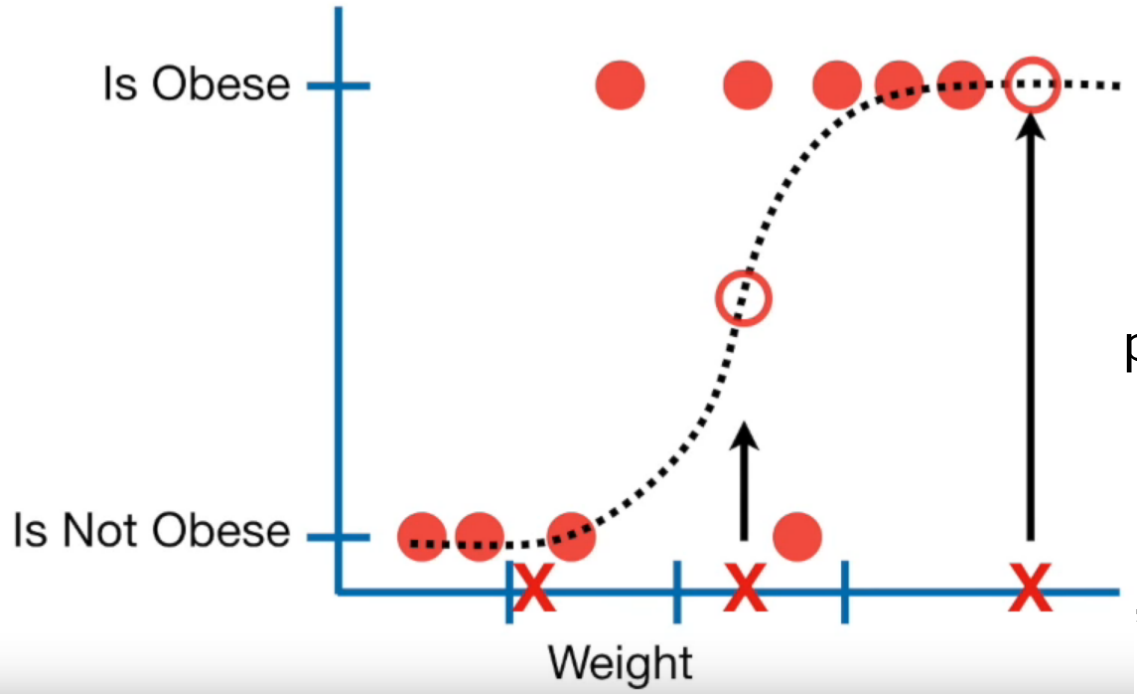


$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

$$0 \leqslant h_\theta(x) \leqslant 1$$

$h_\theta(x)$ - estymowane prawdopodobieństwo tego, że y=1 dla danego x

Regresja logistyczna



$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

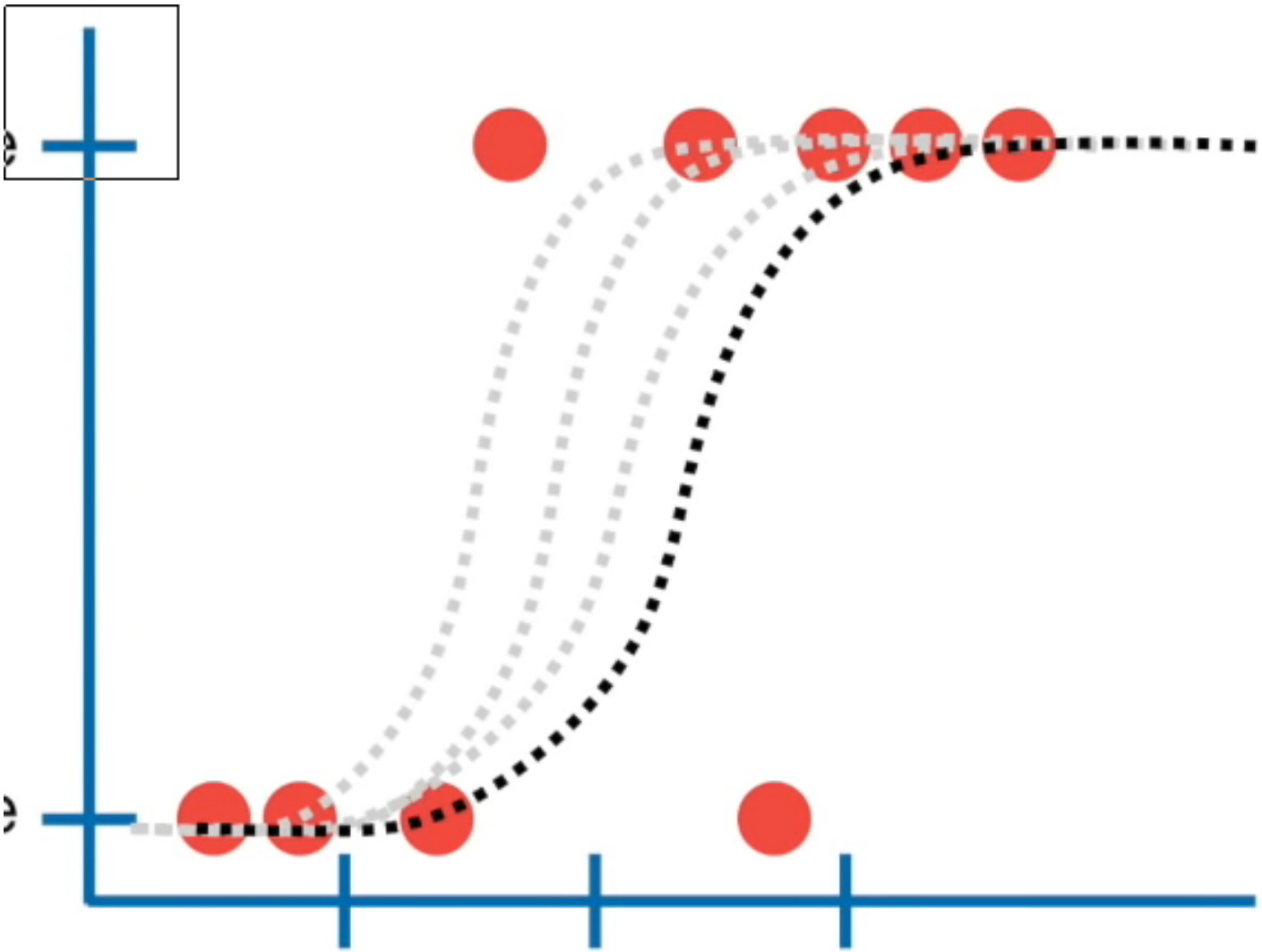
$$0 \leqslant h_\theta(x) \leqslant 1$$

$h_\theta(x)$ - estymowane prawdopodobieństwo tego, że y=1 dla danego x

$$h_\theta(x) = P(y = 1|x; \theta)$$

"prawdopodobieństwo, że y=1, mając x, przy określonym parametrze θ"

Regresja logistyczna



StatQuest

Dopasowanie parametru θ

Zbiór uczący składa się z m par x, y:

$$(x^1, y^1), (x^2, y^2), ..., (x^m, y^m),$$

gdzie każda para posiada n cech wejściowych $x \in \begin{vmatrix} x_0 \\ x_1 \\ ... \\ x_n \end{vmatrix}$ oraz

wyjście: $y \in 0, 1$

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

Dopasowanie parametru θ - Funkcja kosztu

Funkcja kosztu dla regresji liniowej:
$Cost(h_\theta(x), y) = (h_\theta(x) - y)^2$

Dopasowanie parametru θ - Funkcja kosztu

Funkcja kosztu dla regresji liniowej:
$Cost(h_\theta(x), y) = (h_\theta(x) - y)^2$

Funkcja kosztu dla regresji logistycznej:

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{gdy } y = 1 \\ -log(1 - h_\theta(x)) & \text{gdy } y = 0 \end{cases}$$

Dopasowanie parametru θ - Funkcja kosztu

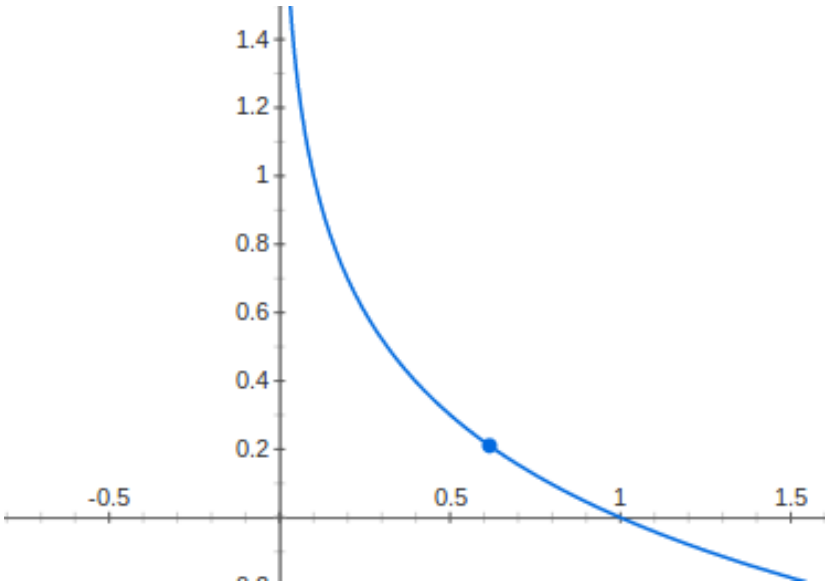
Funkcja kosztu dla regresji logistycznej:

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{gdy } y = 1 \\ -log(1 - h_\theta(x)) & \text{gdy } y = 0 \end{cases}$$

Dopasowanie parametru θ - Funkcja kosztu

Funkcja kosztu dla regresji logistycznej:

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{gdy } y = 1 \\ -log(1 - h_\theta(x)) & \text{gdy } y = 0 \end{cases}$$

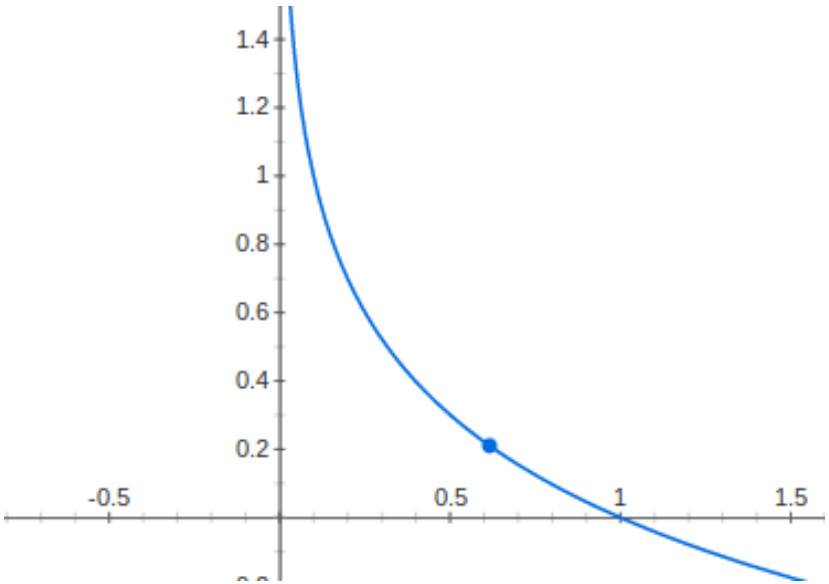


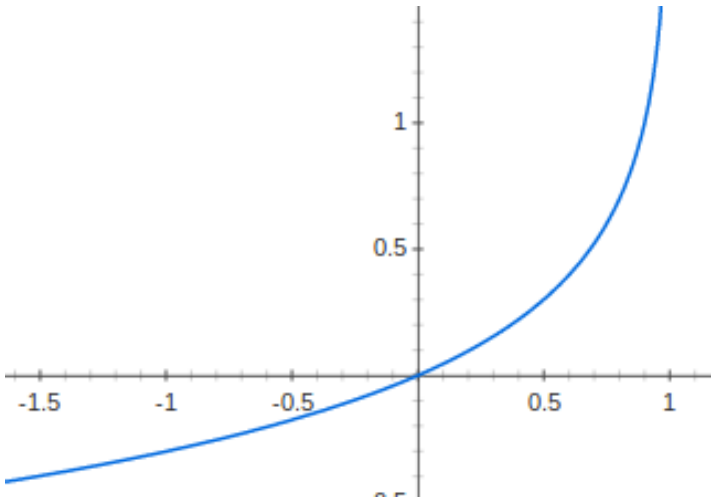
Machine Learning—Andrew Ng, Stanford University

Dopasowanie parametru θ - Funkcja kosztu

Funkcja kosztu dla regresji logistycznej:

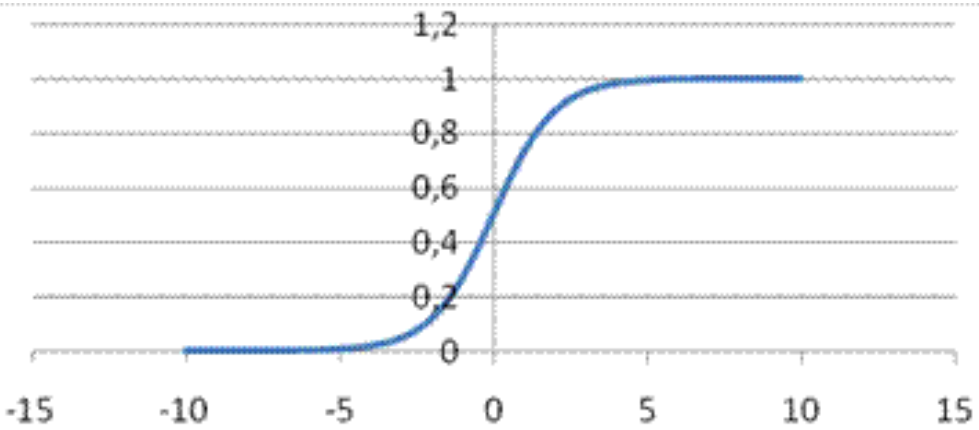
$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{gdy } y = 1 \\ -log(1 - h_\theta(x)) & \text{gdy } y = 0 \end{cases}$$



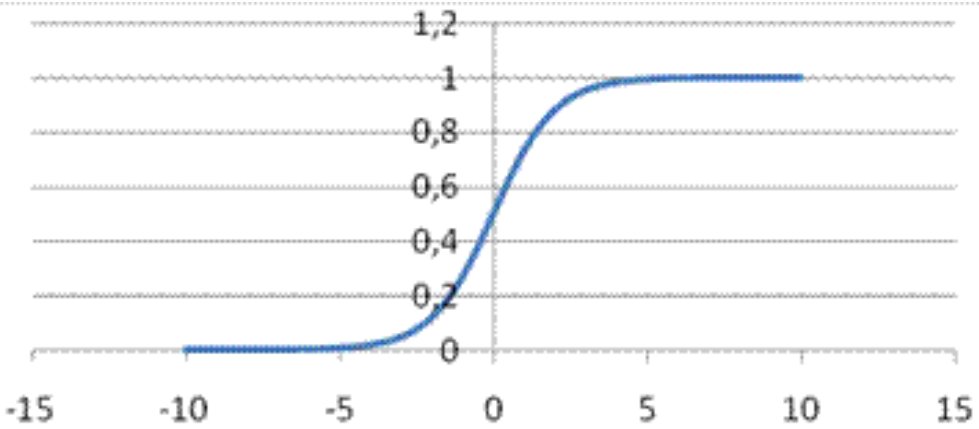


Machine Learning—Andrew Ng, Stanford University

Decision Boundary

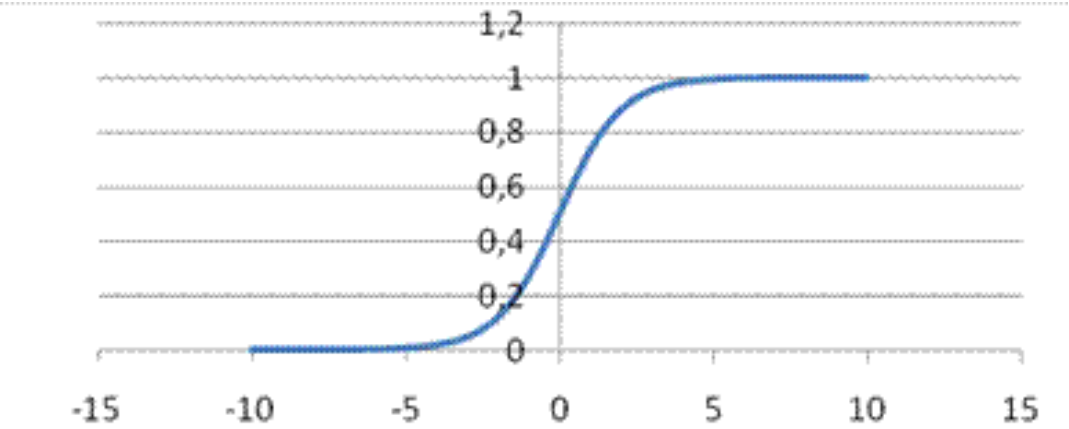


Decision Boundary



$$g(z) = \frac{1}{1+e^{-z}}$$
$$h_\theta(x) = g(\theta^T x)$$

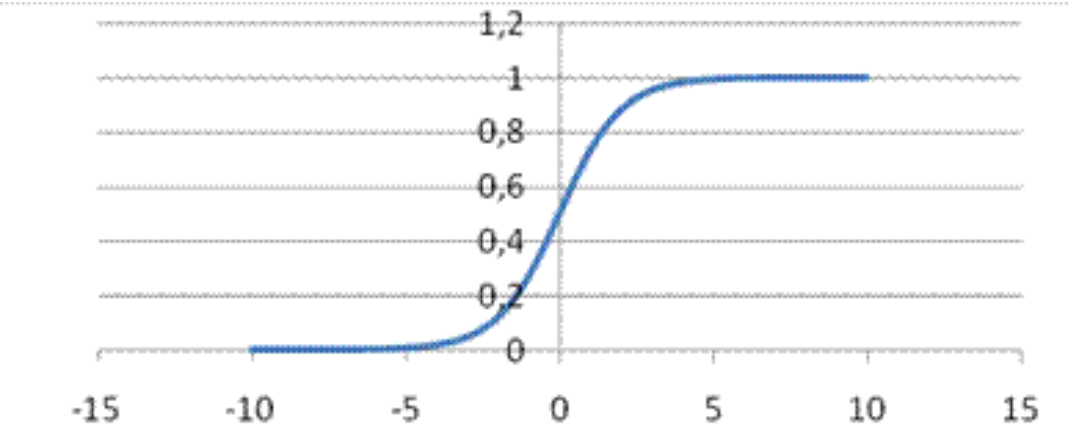
Decision Boundary



$$g(z) = \frac{1}{1+e^{-z}}$$
$$h_\theta(x) = g(\theta^T x)$$

Załóżmy predykcję y=1 jeśli $h_\theta(x) \geqslant 0.5$

Decision Boundary

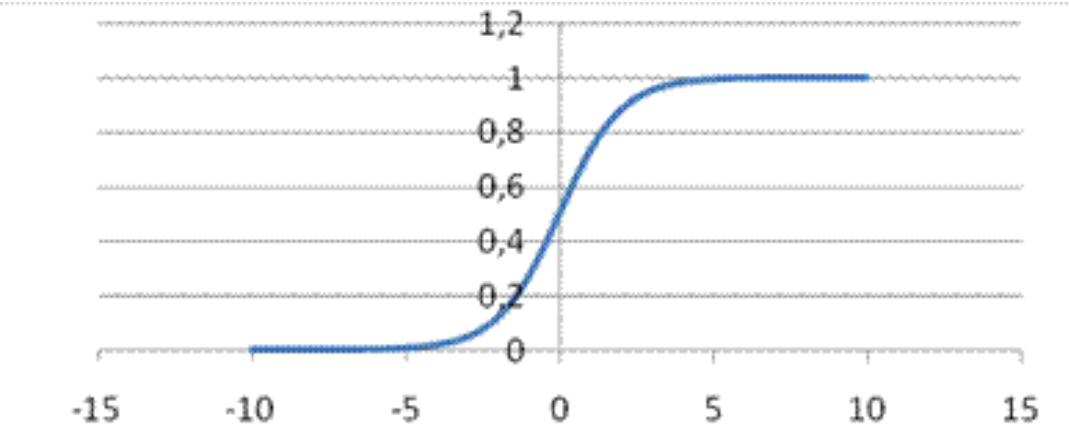


$$g(z) = \frac{1}{1+e^{-z}}$$
$$h_\theta(x) = g(\theta^T x)$$

Załóżmy predykcję y=1 jeśli $h_\theta(x) \geqslant 0.5$

$g(z) \geqslant 0.5$ gdy $z \geqslant 0$

Decision Boundary

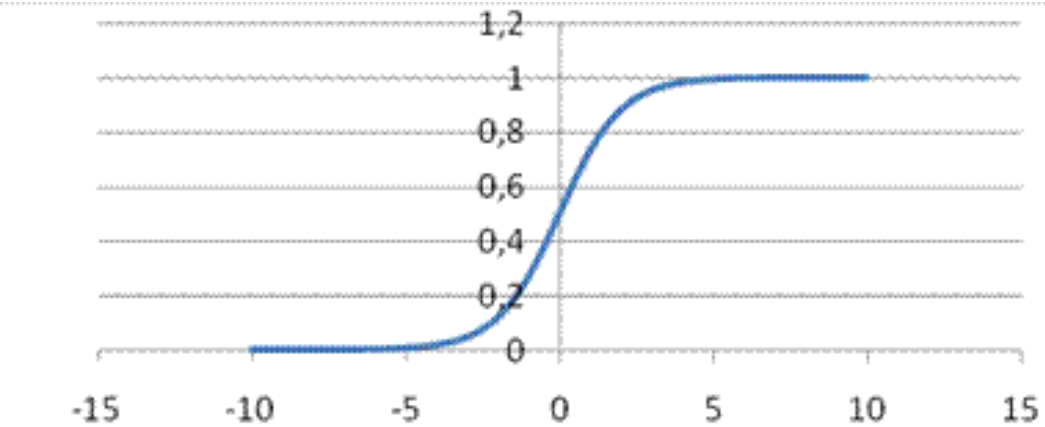


Załóżmy predykcję y=1 jeśli $h_\theta(x) \geqslant 0.5$

$$g(z) = \frac{1}{1+e^{-z}}$$
$$h_\theta(x) = g(\theta^T x)$$

$g(z) \geqslant 0.5$ gdy $z \geqslant 0$
$h_\theta(x) \geqslant 0.5$ gdy $\theta^T x \geqslant 0$

Decision Boundary



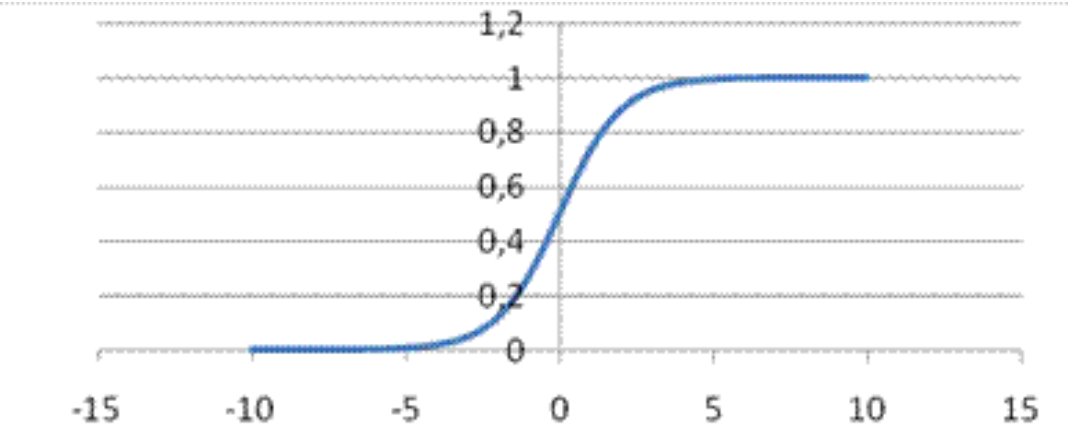
$$g(z) = \frac{1}{1+e^{-z}}$$
$$h_\theta(x) = g(\theta^T x)$$

Załóżmy predykcję y=1 jeśli $h_\theta(x) \geqslant 0.5$

predykcję y=0 jeśli $h_\theta(x) < 0.5$

$g(z) \geqslant 0.5$ gdy $z \geqslant 0$
$h_\theta(x) \geqslant 0.5$ gdy $\theta^T x \geqslant 0$

Decision Boundary



$$g(z) = \frac{1}{1+e^{-z}}$$
$$h_\theta(x) = g(\theta^T x)$$

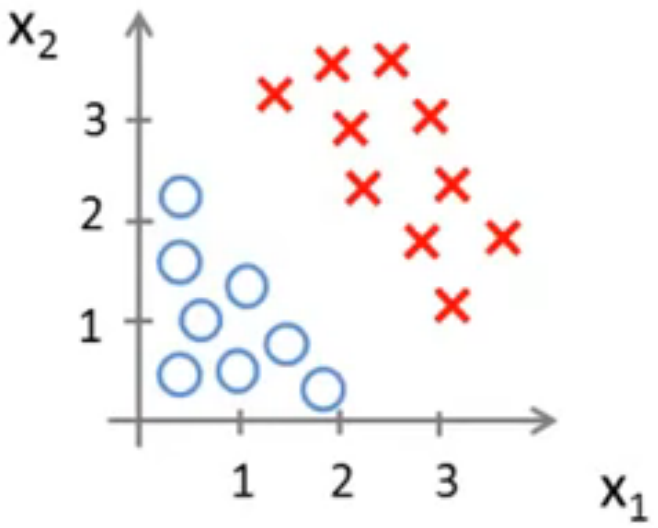
Załóżmy predykcję y=1 jeśli $h_\theta(x) \geqslant 0.5$

$$g(z) \geqslant 0.5 \text{ gdy } z \geqslant 0$$
$$h_\theta(x) \geqslant 0.5 \text{ gdy } \theta^T x \geqslant 0$$

predykcję y=0 jeśli $h_\theta(x) < 0.5$

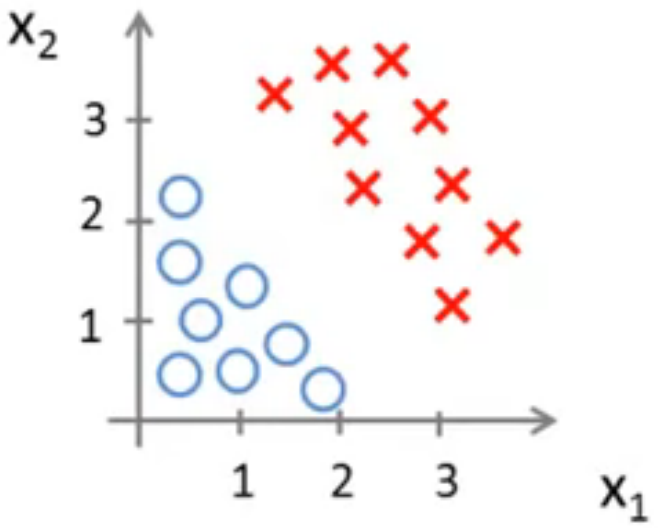
$$h_\theta(x) < 0.5 \text{ gdy } \theta^T x < 0$$

Decision Boundary - przykład



Machine Learning—Andrew Ng, Stanford University

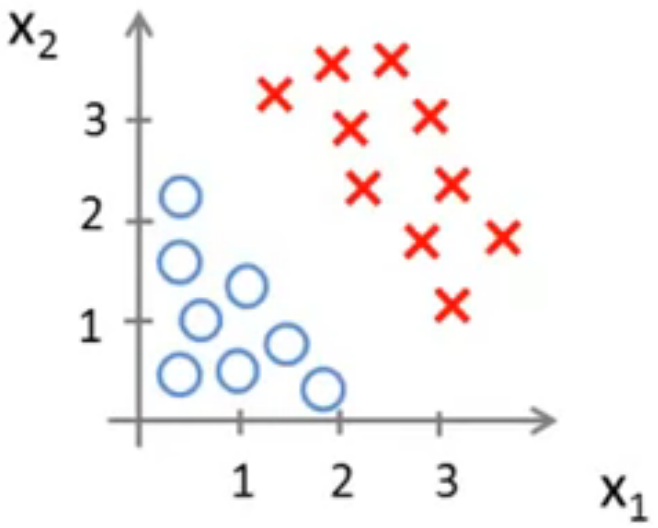
Decision Boundary - przykład



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

Machine Learning—Andrew Ng, Stanford University

Decision Boundary - przykład

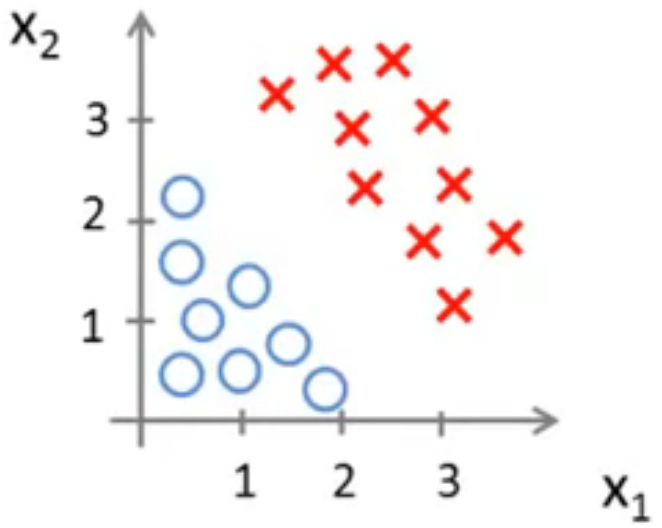


$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$$\theta = \begin{vmatrix} -3 \\ 1 \\ 1 \end{vmatrix}$$

Machine Learning—Andrew Ng, Stanford University

Decision Boundary - przykład

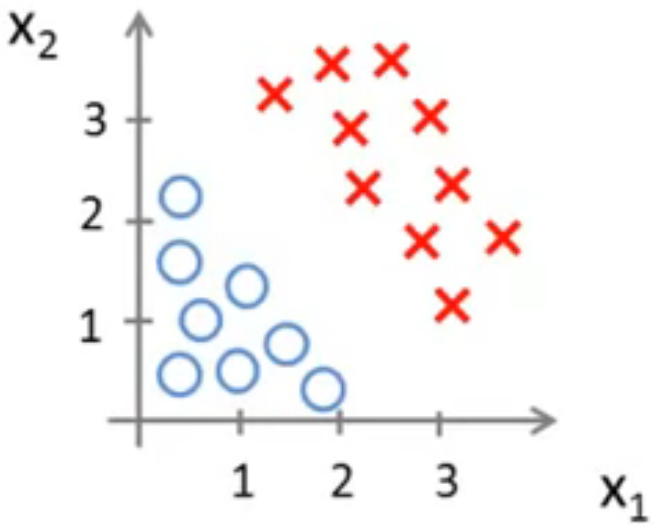


$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$$\theta = \begin{vmatrix} -3 \\ 1 \\ 1 \end{vmatrix}$$

$$-3 + x_1 + x_2 \geqslant 0$$

Decision Boundary - przykład



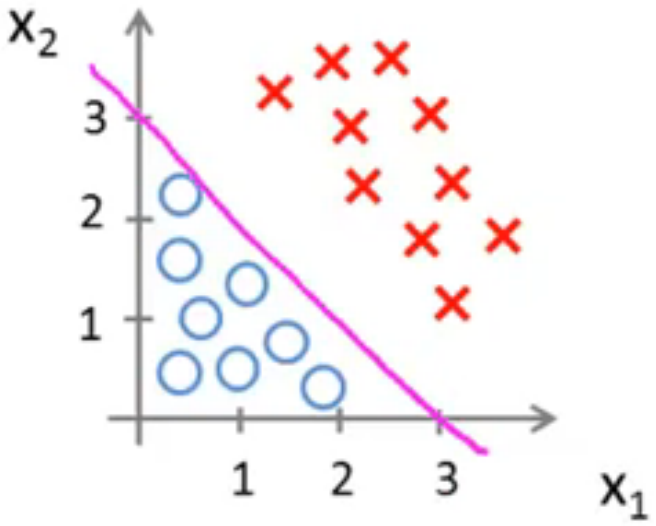
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$$\theta = \begin{vmatrix} -3 \\ 1 \\ 1 \end{vmatrix}$$

$$-3 + x_1 + x_2 \geqslant 0$$

$$x_1 + x_2 = 3 \Rightarrow h_\theta(x) = 0.5$$

Decision Boundary - przykład



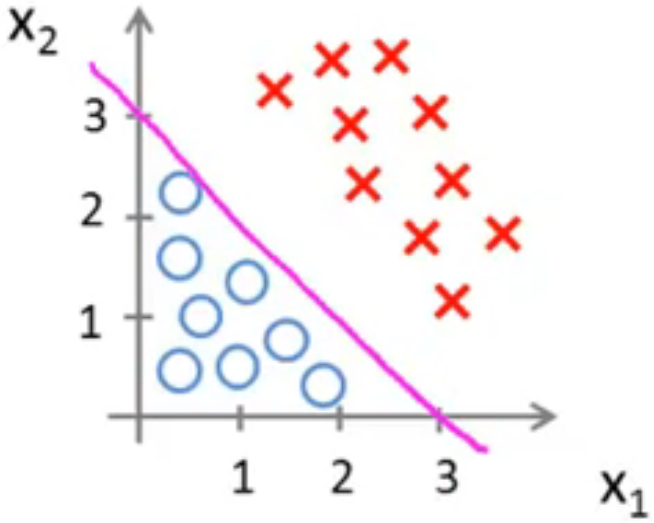
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$$\theta = \begin{vmatrix} -3 \\ 1 \\ 1 \end{vmatrix}$$

$$-3 + x_1 + x_2 \geqslant 0$$

$$x_1 + x_2 = 3 \Rightarrow h_\theta(x) = 0.5$$

Decision Boundary - przykład



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

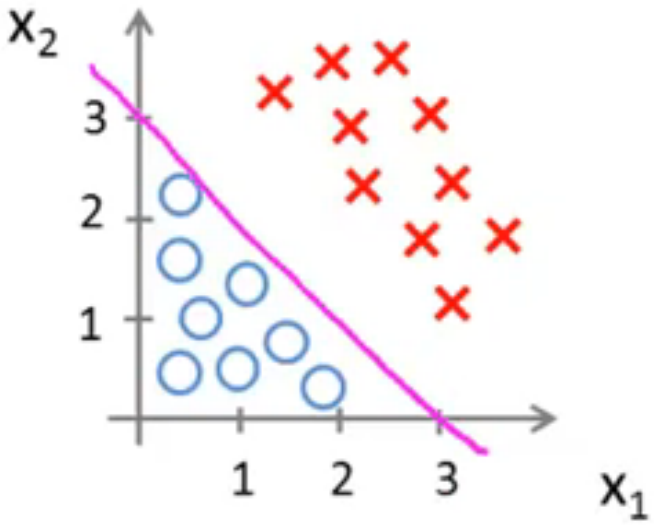
$$\theta = \begin{vmatrix} -3 \\ 1 \\ 1 \end{vmatrix}$$

$$-3 + x_1 + x_2 \geqslant 0$$

$$x_1 + x_2 = 3 \Rightarrow h_\theta(x) = 0.5$$

Predykcja y=1 dla $x_1 + x_2 \geqslant 3$

Decision Boundary - przykład



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$$\theta = \begin{vmatrix} -3 \\ 1 \\ 1 \end{vmatrix}$$

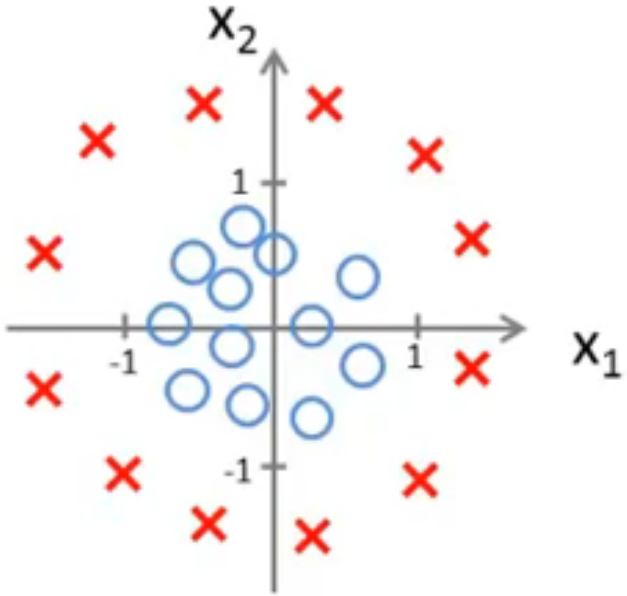
$$-3 + x_1 + x_2 \geqslant 0$$

$$x_1 + x_2 = 3 \Rightarrow h_\theta(x) = 0.5$$

Predykcja y=1 dla $x_1 + x_2 \geqslant 3$

Predykcja y=0 dla $x_1 + x_2 < 3$

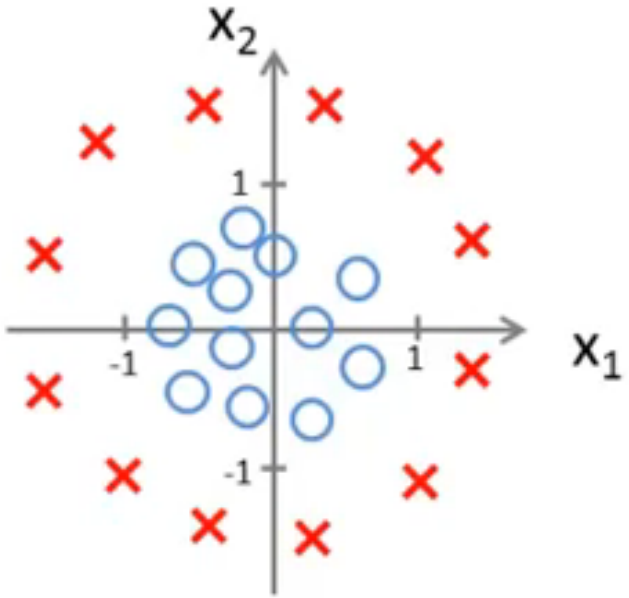
Decision Boundary - przykład nieliniowy



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

Machine Learning—Andrew Ng, Stanford University

Decision Boundary - przykład nieliniowy



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

Uwaga: Granice decyzyjne nie są obliczane na podstawie zbioru uczącego, zbiór uczący optymalizuje parametr θ.

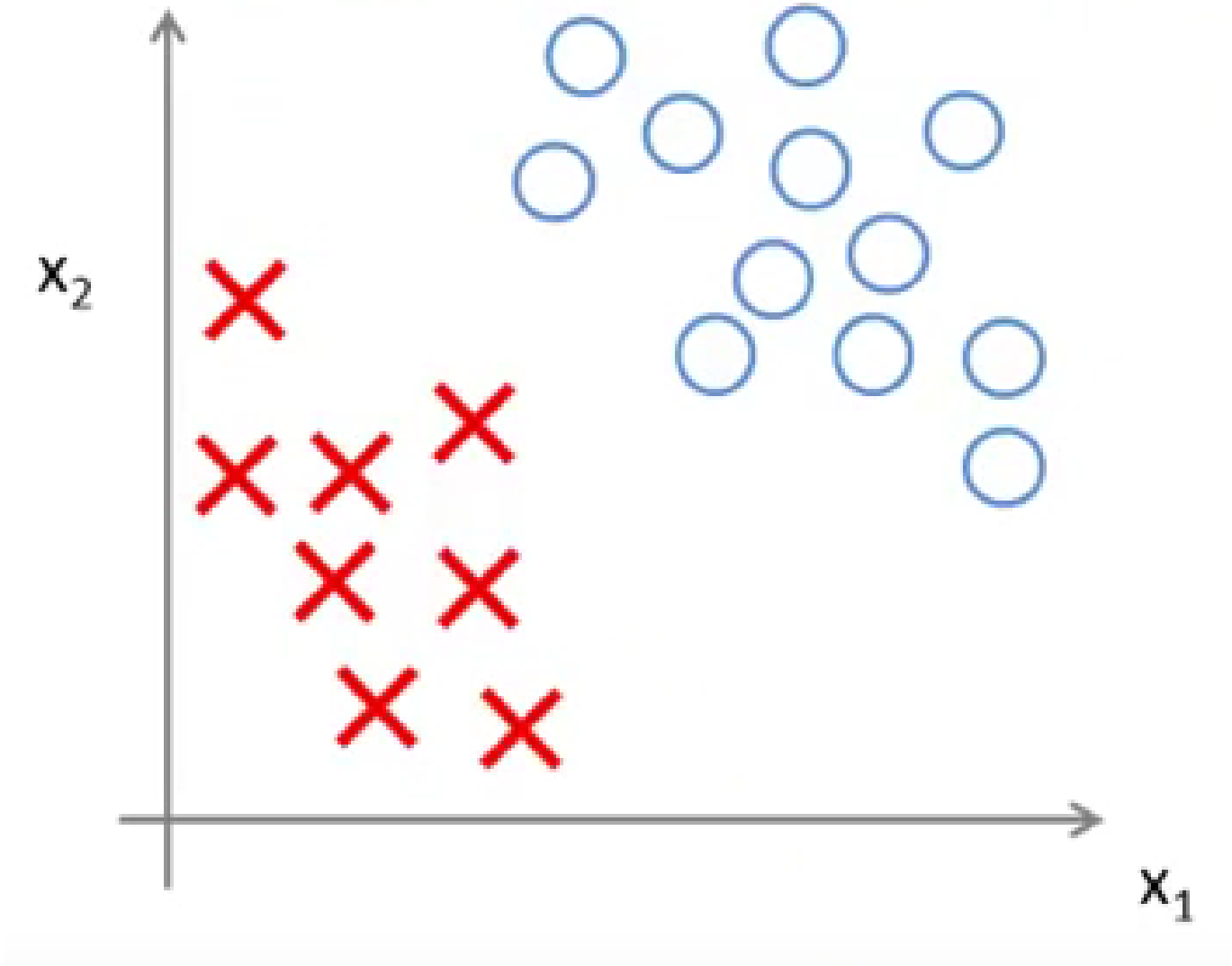
Machine Learning—Andrew Ng, Stanford University

Spis treści

1. Wprowadzenie do klasyfikacji
2. kNN - k Nearest Neighbours
3. Regresja logistyczna
4. **SVM**
5. Naive Bayes
6. Drzewa Decyzyjne
7. Random Forest

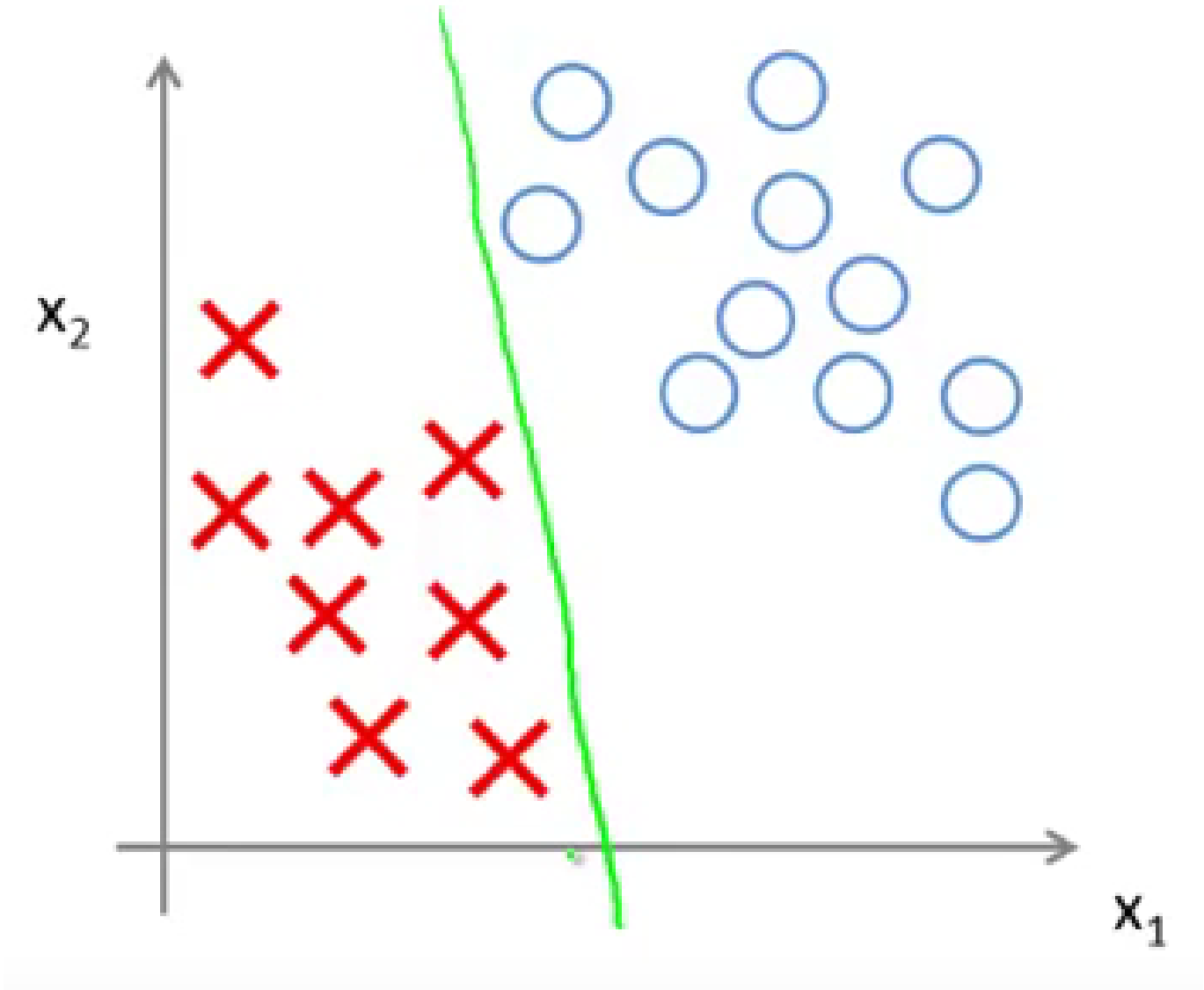
Problem regresji logistycznej

Problem regresji logistycznej



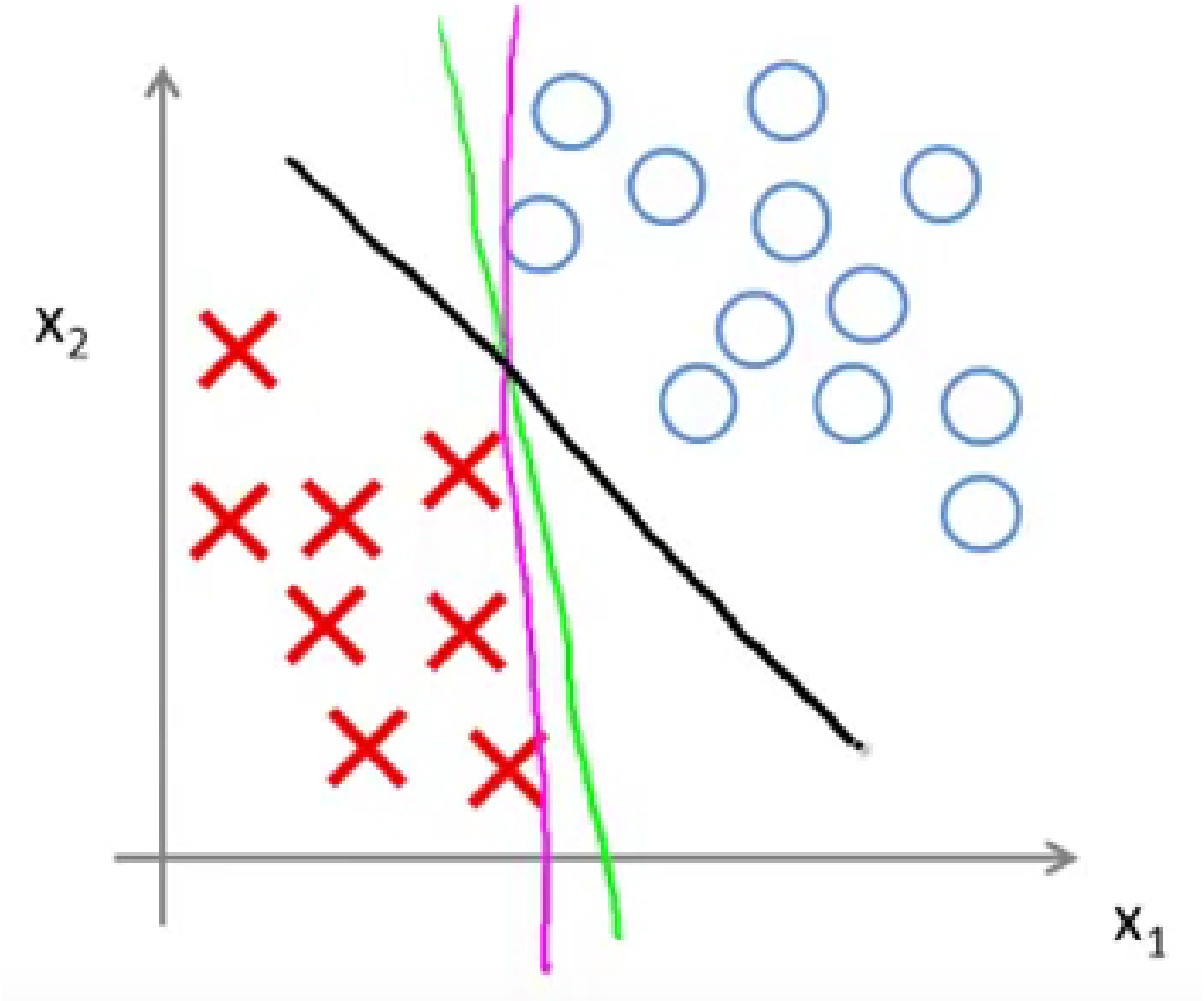
Machine Learning - Andrew Ng, Stanford University

Problem regresji logistycznej



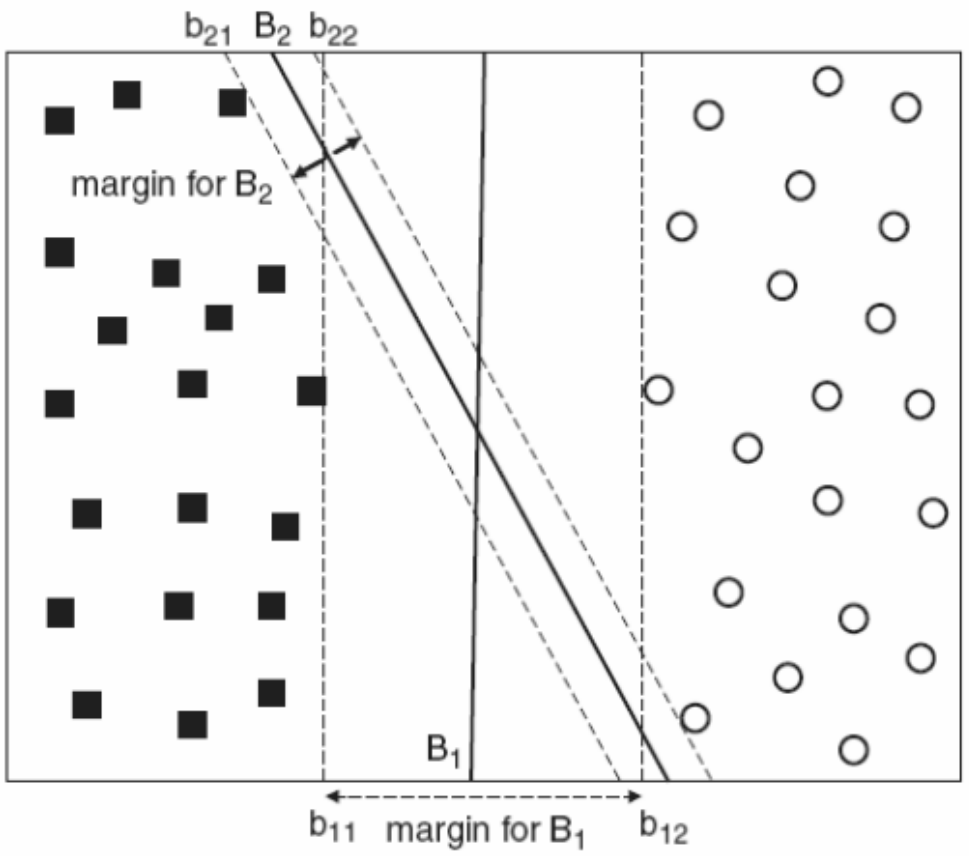
Machine Learning - Andrew Ng, Stanford University

Problem regresji logistycznej



Machine Learning - Andrew Ng, Stanford University

Metoda Wektorów nośnych - Support Vector Machine



Marginesy są to proste (hiperpłaszczyzny) otrzymane przez równoległe przesuwanie granicy aż do pierwszych punktów z obu klas.

Odległość między nimi to **marginesy** klasyfikatora.

Szerszy margines to lepsze własności generalizacji oraz mniejsza podatność na przeuczenie.

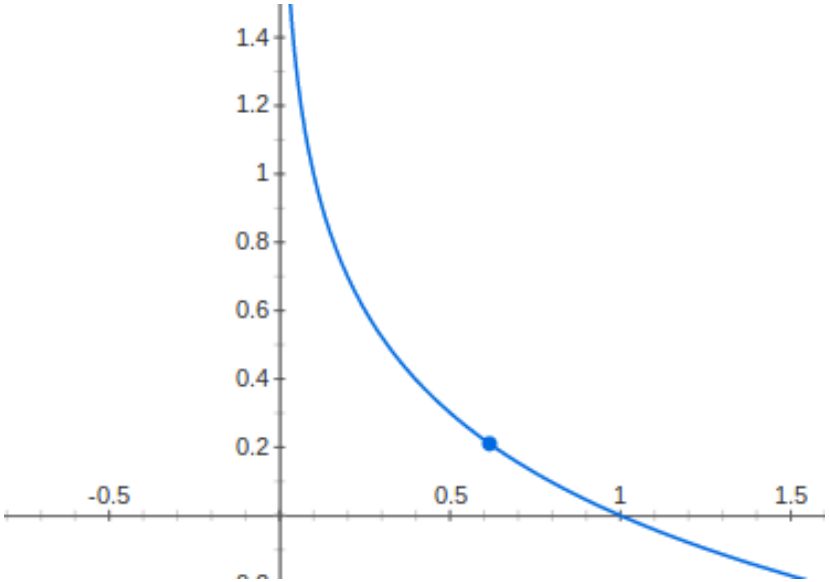
Support Vectors

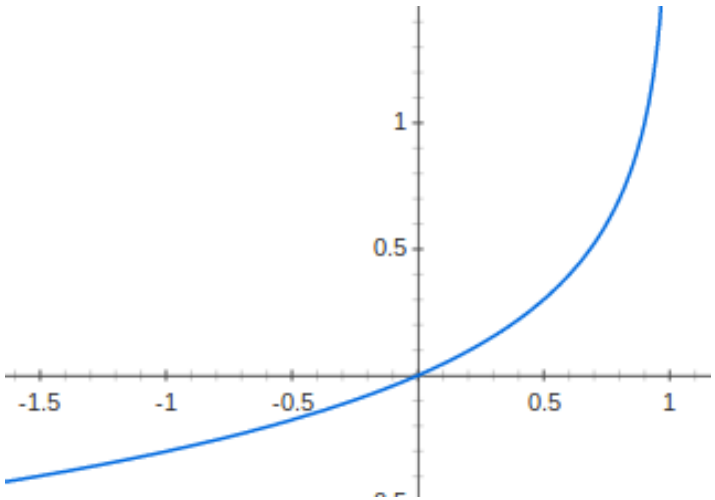
Alice Zhao

Funkcja kosztu

Funkcja kosztu dla regresji logistycznej:

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{gdy } y = 1 \\ -log(1 - h_\theta(x)) & \text{gdy } y = 0 \end{cases}$$





Machine Learning—Andrew Ng, Stanford University

Funkcja kosztu

Funkcja kosztu dla regresji logistycznej:

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{gdy } y = 1 \\ -log(1 - h_\theta(x)) & \text{gdy } y = 0 \end{cases}$$

$$Cost = -(y \log h_\theta(x) + (1 - y) \log(1 - h_\theta(x)))$$

Funkcja kosztu

Funkcja kosztu dla regresji logistycznej:

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{gdy } y = 1 \\ -log(1 - h_\theta(x)) & \text{gdy } y = 0 \end{cases}$$

$$Cost = -(y \log h_\theta(x) + (1 - y) \log(1 - h_\theta(x)))$$
$$= -y \log \frac{1}{1+e^{-\theta^T x}} - (1 - y) \log(1 - \frac{1}{1+e^{-\theta^T x}})$$

Funkcja kosztu

Funkcja kosztu dla regresji logistycznej:

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{gdy } y = 1 \\ -log(1 - h_\theta(x)) & \text{gdy } y = 0 \end{cases}$$

$$Cost = -(y \log h_\theta(x) + (1 - y) \log(1 - h_\theta(x)))$$
$$= -y \log \frac{1}{1+e^{-\theta^T x}} - (1 - y) \log(1 - \frac{1}{1+e^{-\theta^T x}})$$

$$min_\theta \underbrace{\frac{1}{m}[\sum_{i=1}^{m} y^{(i)}(-\log h_\theta(x^{(i)})) + (1 - y^{(i)})((-\log(1 - h_\theta(x^{(i)})))]}_{J(\theta)} +$$

Funkcja kosztu

$$Cost = -(y \log h_\theta(x) + (1-y)\log(1-h_\theta(x)))$$
$$= -y \log \frac{1}{1+e^{-\theta^T x}} - (1-y)\log(1-\frac{1}{1+e^{-\theta^T x}})$$

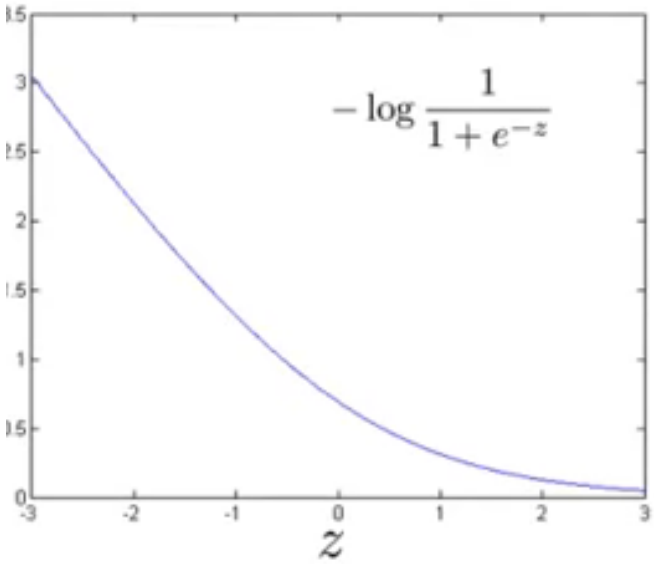
$$min_\theta \underbrace{\frac{1}{m}[\sum_{i=1}^{m} y^{(i)}(-\log h_\theta(x^{(i)})) + (1-y^{(i)})((-\log(1-h_\theta(x^{(i)})))]}_{J(\theta)} +$$

$$+\lambda \underbrace{\frac{\partial J(\theta)}{\partial \theta_j}}_{B}$$

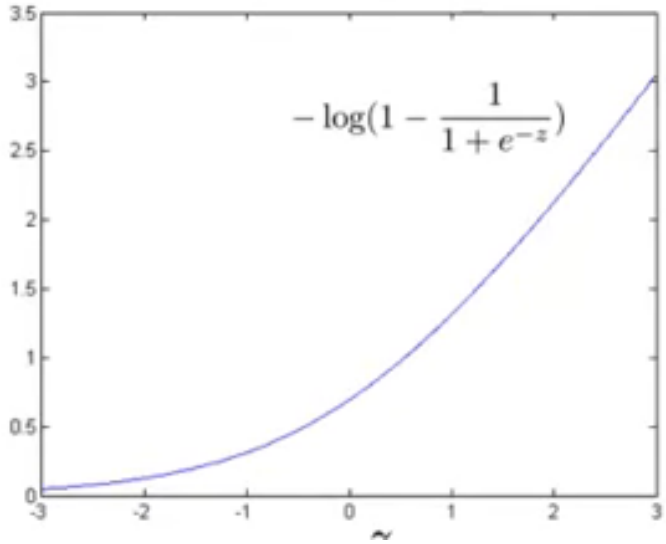
Machine Learning—Andrew Ng, Stanford University

Funkcja kosztu

dla y=1 ($\theta^T x \gg 0$)



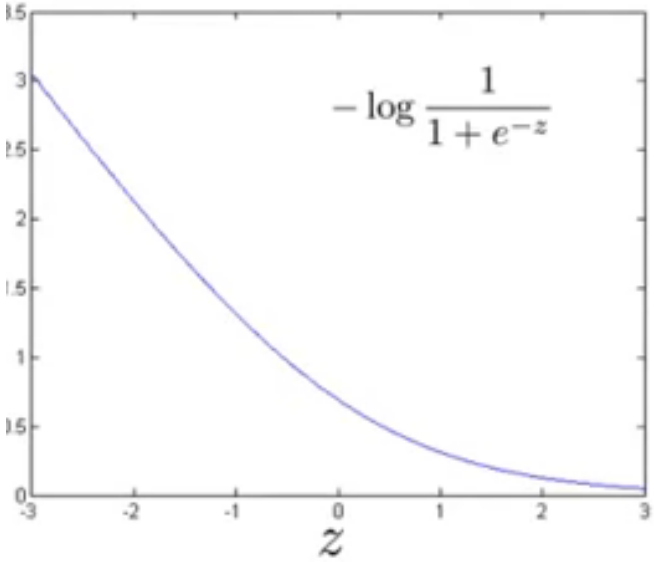
dla y=0 ($\theta^T x \ll 0$)

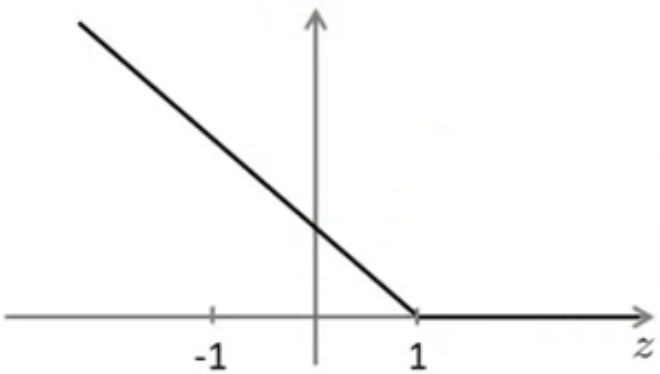


Machine Learning—Andrew Ng, Stanford University

Funkcja kosztu

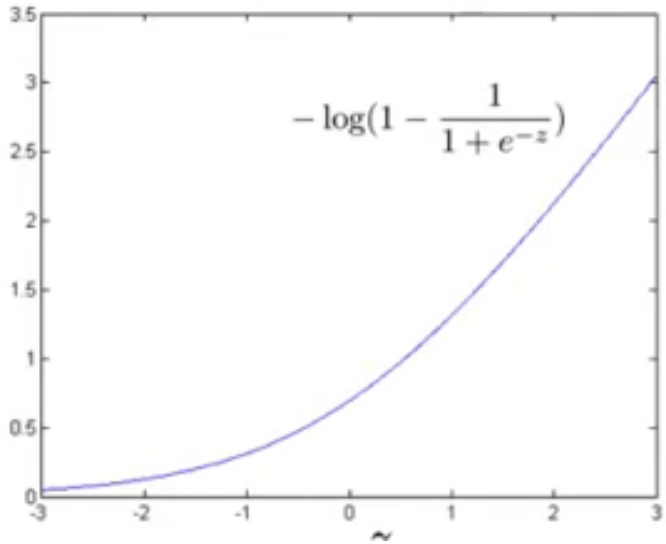
dla y=1 ($\theta^T x \gg 0$)

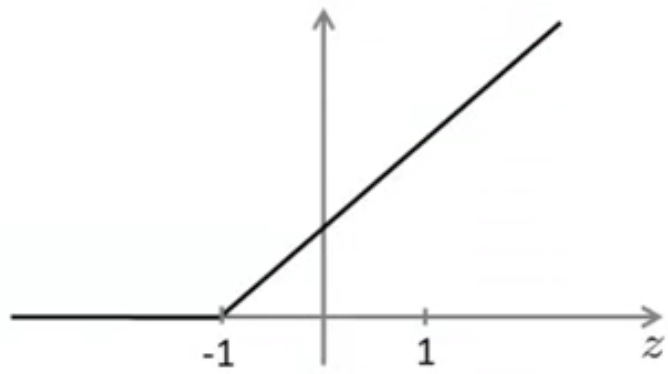




$\theta^T x \geqslant 1$

dla y=0 ($\theta^T x \ll 0$)





$\theta^T x \leqslant -1$

Machine Learning—Andrew Ng, Stanford University

Funkcja kosztu

Regresja logistyczna:

$$min_\theta \frac{1}{m}[\sum_{i=1}^{m} y^{(i)}(-\log h_\theta(x^{(i)}))+(1-y^{(i)})((-\log(1-h_\theta(x^{(i)})))]+\lambda B$$

Funkcja kosztu

Regresja logistyczna:

$$min_\theta \frac{1}{m}[\sum_{i=1}^{m} y^{(i)}(-\log h_\theta(x^{(i)}))+(1-y^{(i)})((-\log(1-h_\theta(x^{(i)})))]+\lambda B$$

$$J(\theta) + \lambda B$$

gdzie λ- parametr regularyzacji

Funkcja kosztu

Regresja logistyczna:

$$min_\theta \frac{1}{m}[\sum_{i=1}^{m} y^{(i)}(-\log h_\theta(x^{(i)}))+(1-y^{(i)})((-\log(1-h_\theta(x^{(i)})))]+\lambda B$$

$$J(\theta) + \lambda B$$

SVM:

$$CJ(\theta) + B$$

Funkcja kosztu

Regresja logistyczna:

$$min_\theta \frac{1}{m}[\sum_{i=1}^{m} y^{(i)}(-\log h_\theta(x^{(i)}))+(1-y^{(i)})((-\log(1-h_\theta(x^{(i)})))]+\lambda B$$

$$J(\theta) + \lambda B$$

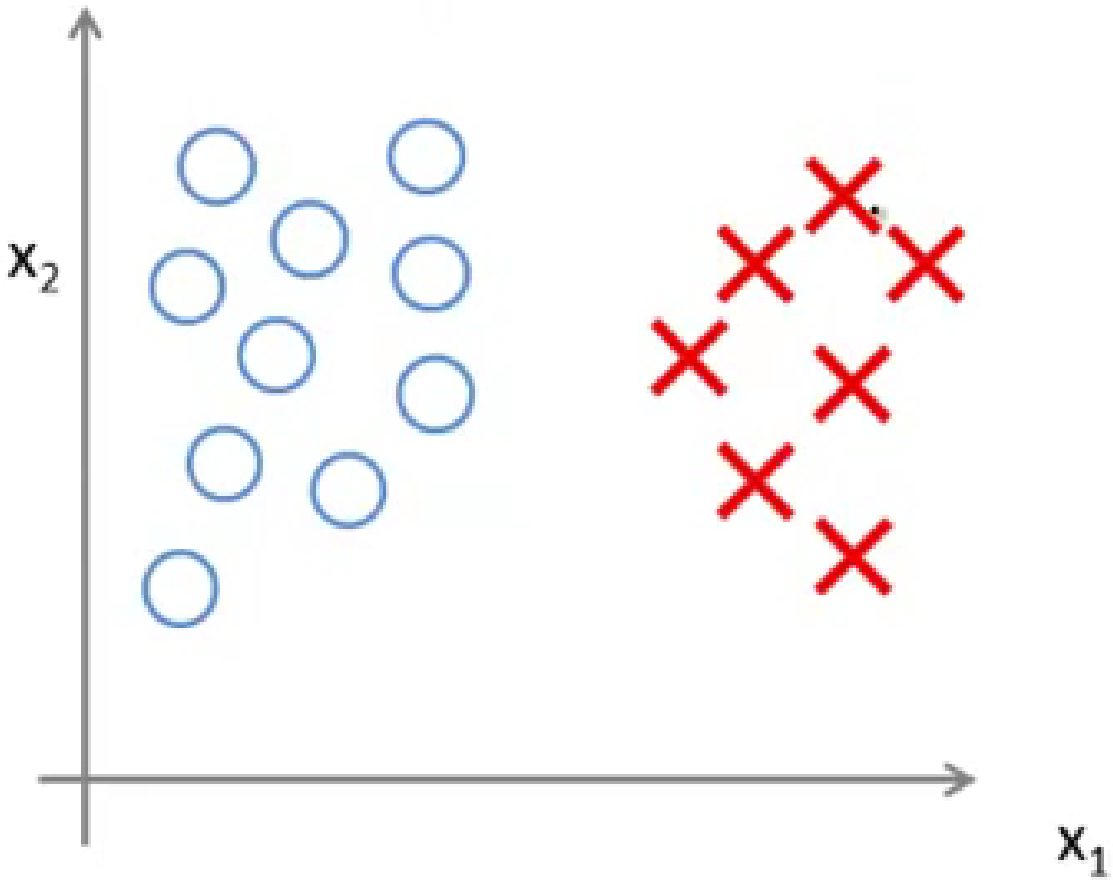
SVM:

$$CJ(\theta) + B$$

$$min_\theta C[\sum_{i=1}^{m} y^{(i)}(cost_1(\theta^T x^{(i)}) + (1 - y^{(i)})(cost_0(\theta^T x^{(i)})] + B$$

Funkcja kosztu

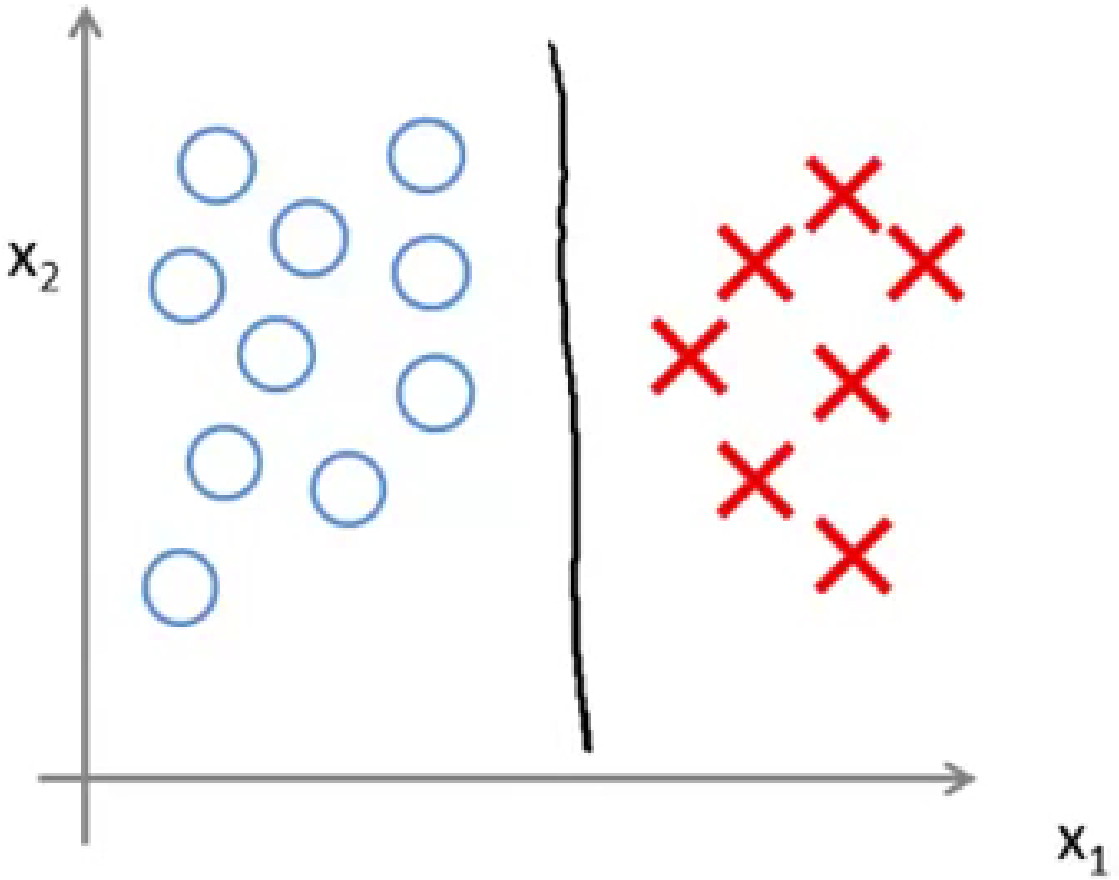
$$min_\theta CJ(\theta) + B$$



Machine Learning - Andrew Ng, Stanford University

Funkcja kosztu

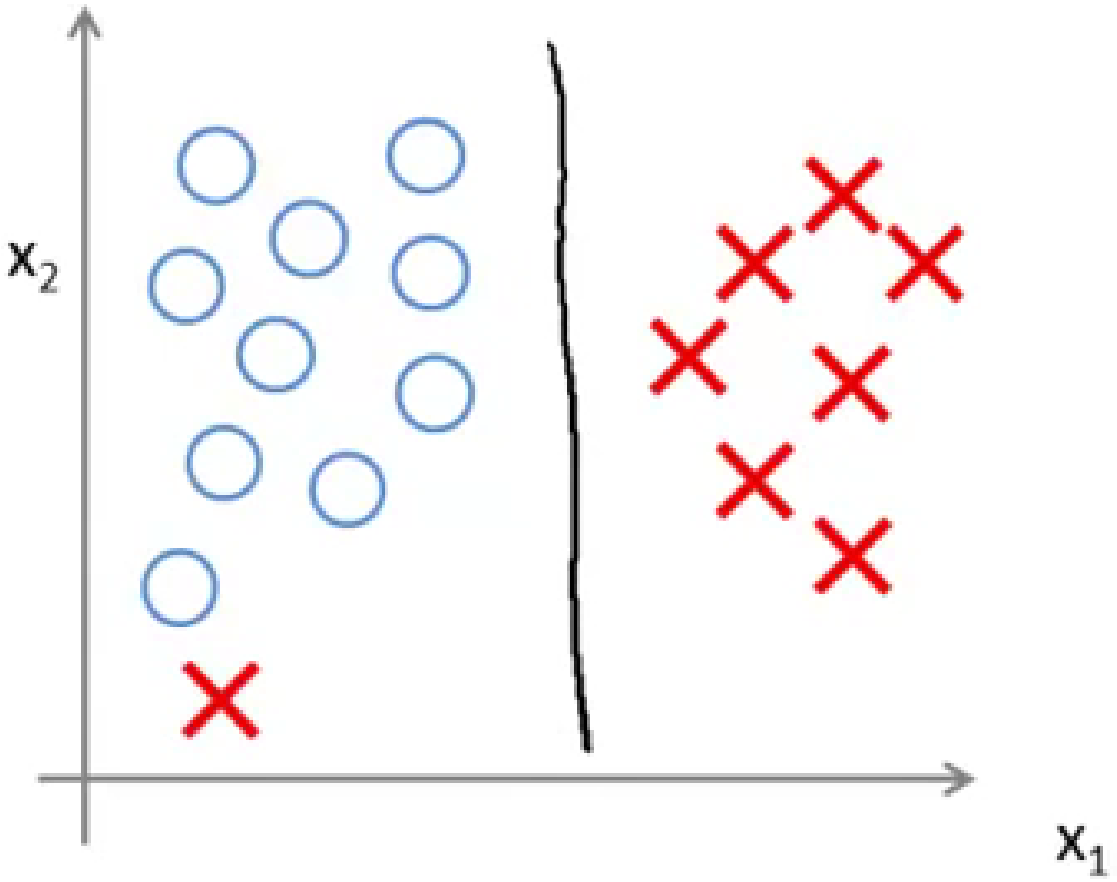
$$min_\theta CJ(\theta) + B$$



Machine Learning - Andrew Ng, Stanford University

Funkcja kosztu

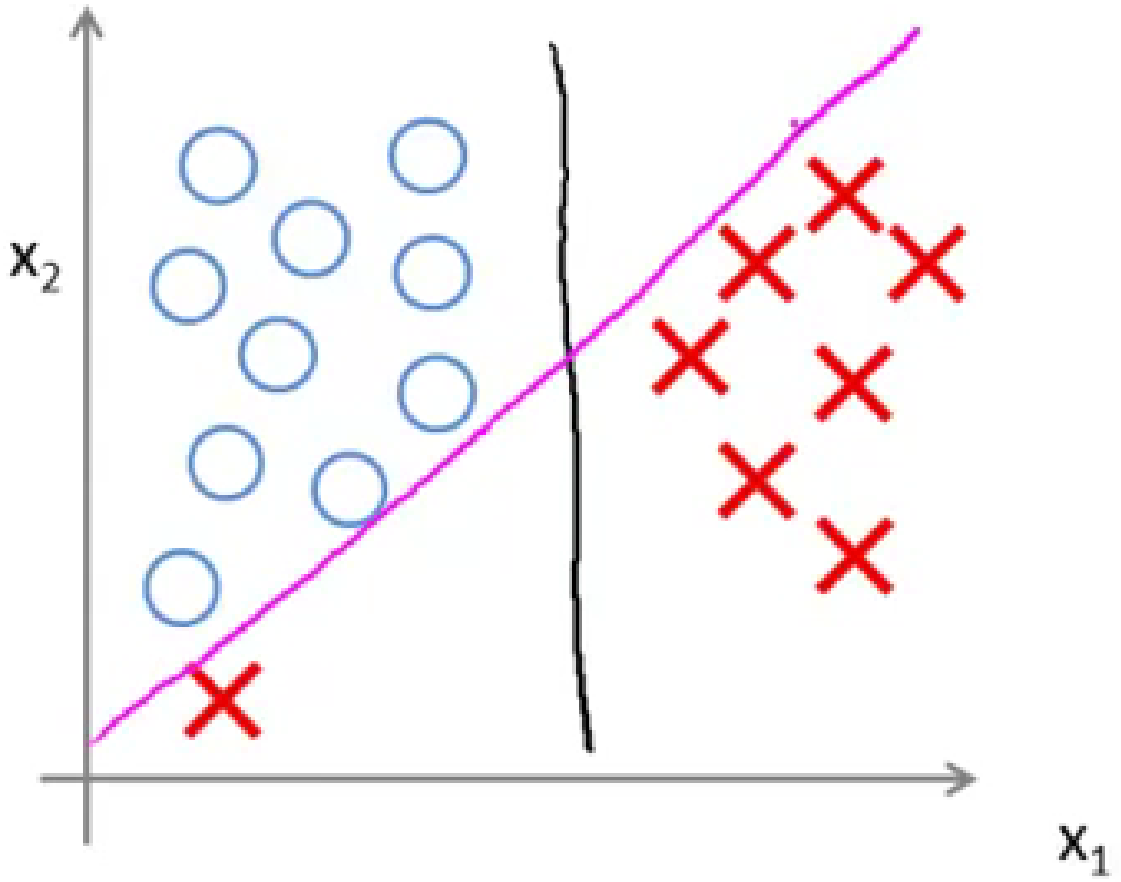
$$min_\theta CJ(\theta) + B$$



Machine Learning - Andrew Ng, Stanford University

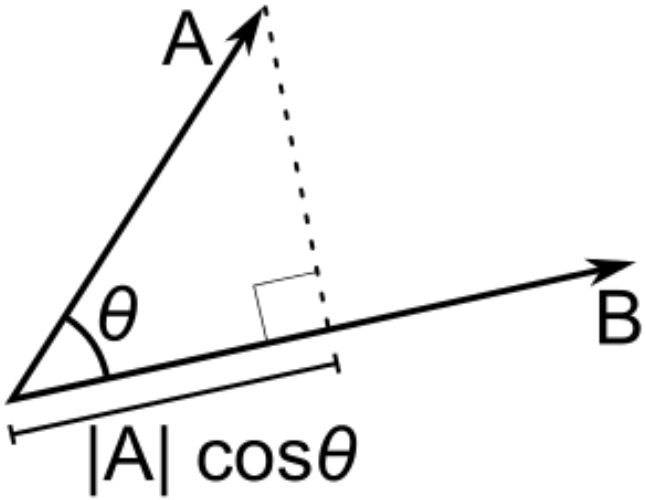
Funkcja kosztu

$$min_\theta CJ(\theta) + B$$



Machine Learning - Andrew Ng, Stanford University

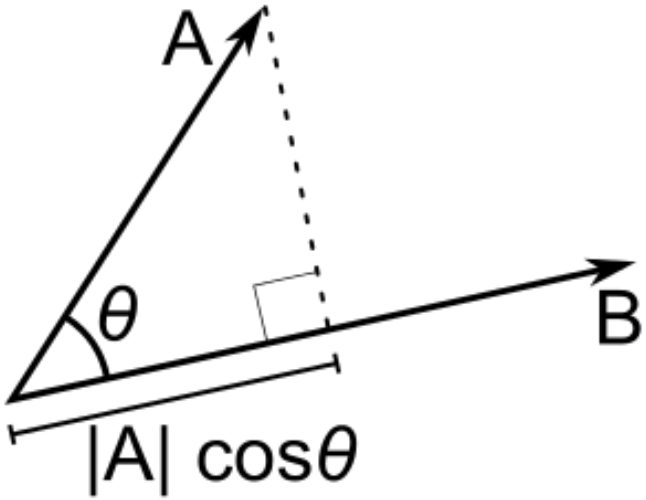
Wektory nośne - Support Vectors



$$\mathbf{A} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \mathbf{B} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}$$

Wikipedia

Wektory nośne - Support Vectors

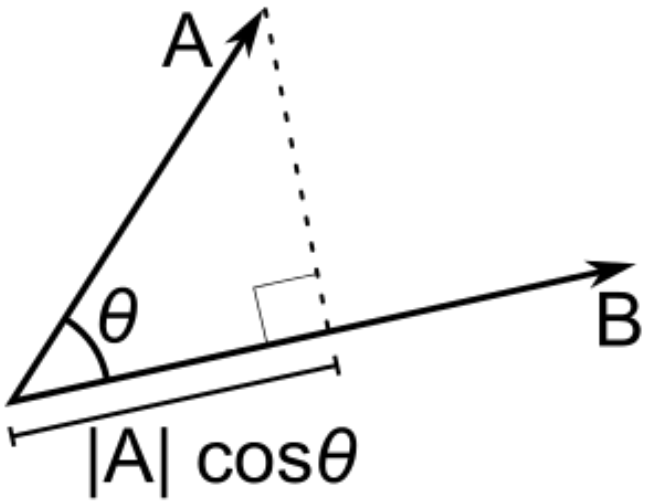


$$\mathbf{A} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \mathbf{B} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}$$

$$\mathbf{B}^T\mathbf{A} = B_1A_1 + B_2A_2 = |\mathbf{B}||\mathbf{A}| \cos\theta$$

Wikipedia

Wektory nośne - Support Vectors



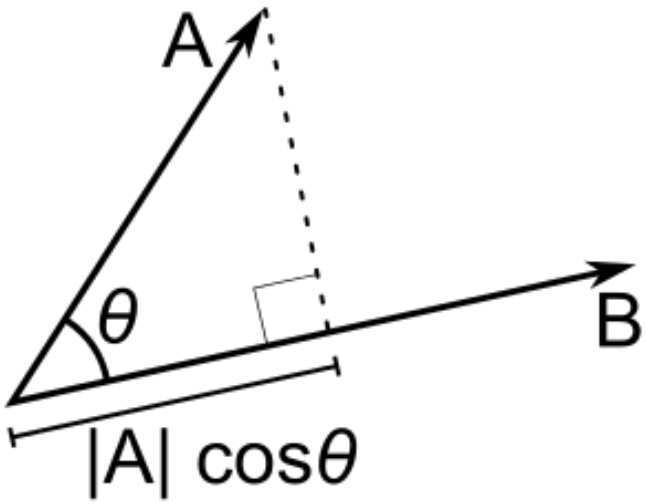
$$\mathbf{A} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \mathbf{B} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}$$

$$\mathbf{B}^T\mathbf{A} = B_1A_1 + B_2A_2 = |\mathbf{B}||\mathbf{A}|\cos\theta = p|\mathbf{B}|$$

$$|\mathbf{A}|\cos\theta = p$$

Wikipedia

Wektory nośne - Support Vectors



$$\mathbf{A} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \mathbf{B} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}$$

$$\mathbf{B}^T\mathbf{A} = B_1A_1 + B_2A_2 = |\mathbf{B}||\mathbf{A}|\cos\theta = p|\mathbf{B}|$$

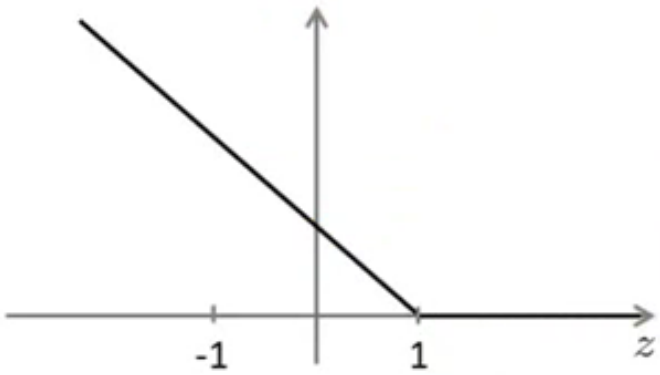
$$|\mathbf{A}|\cos\theta = p$$

p jest < 0 dla $\theta > 90^\circ$

Wikipedia

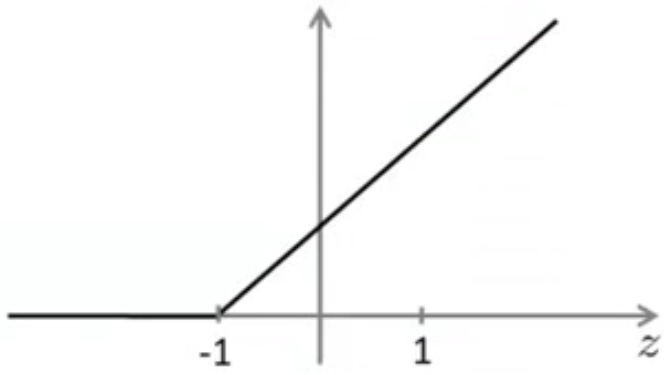
Wektory nośne

dla y=1



$\theta^T \mathbf{x} \geqslant 1$

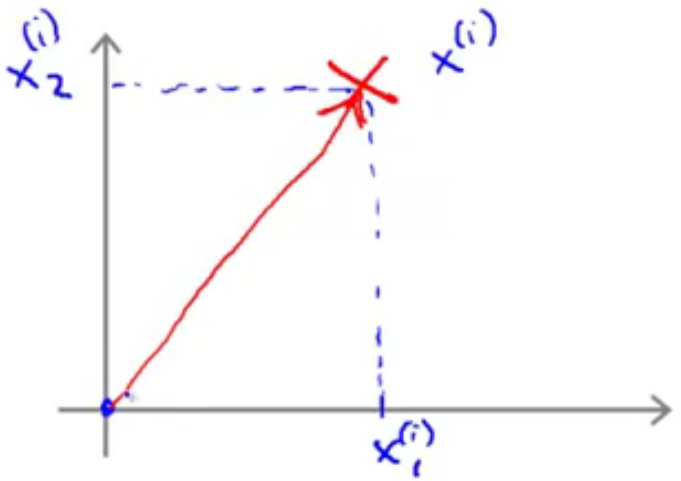
dla y=0



$\theta^T x \leqslant -1$

Wektory nośne

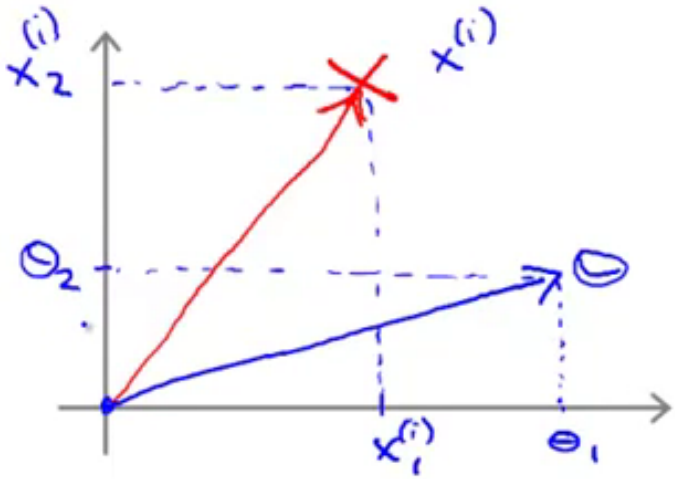
$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$



Machine Learning—Andrew Ng, Stanford University

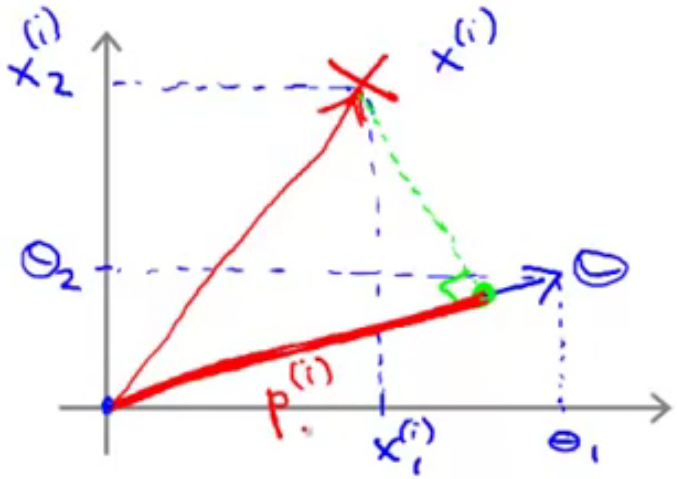
Wektory nośne

$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$



Wektory nośne

$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$

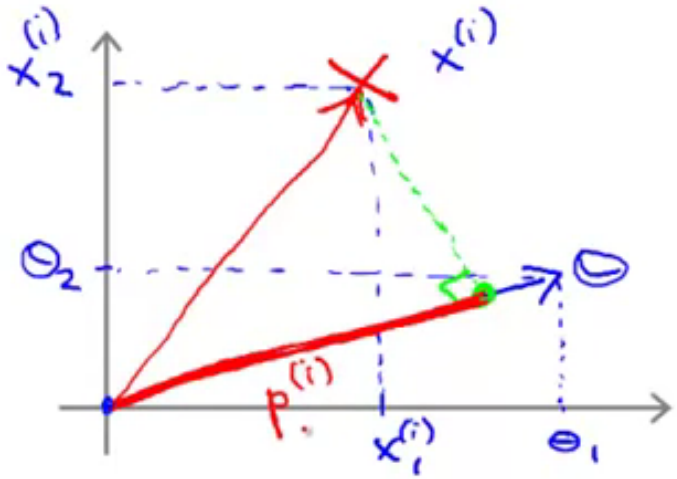


Wektory nośne

$\theta^T \mathbf{x}^{(i)} \geqslant 1$ dla $y^{(i)} = 1$
$\theta^T \mathbf{x}^{(i)} \leqslant -1$ dla $y^{(i)} = 0$

$\theta^T \mathbf{x}^{(i)} = p^{(i)} |\theta|$

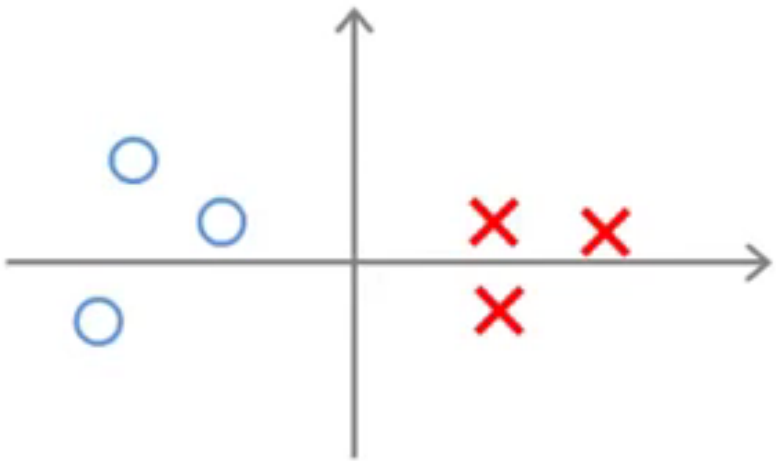
$p^{(i)}$ - rzut wektora $\mathbf{x}^{(i)}$ na θ

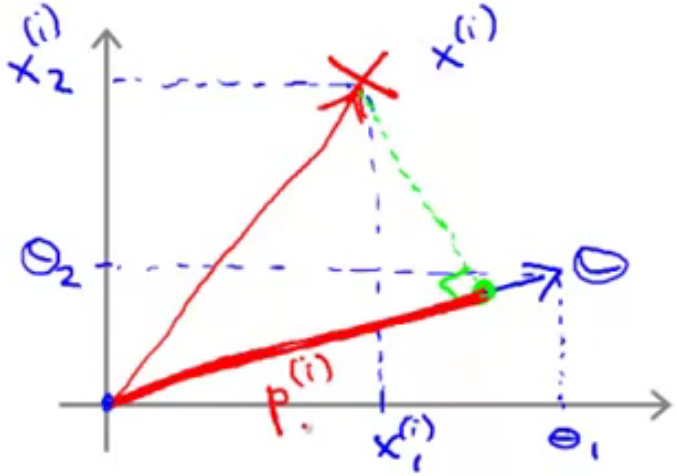


Wektory nośne

$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$

$$\theta^T \mathbf{x}^{(i)} = p^{(i)} |\theta|$$



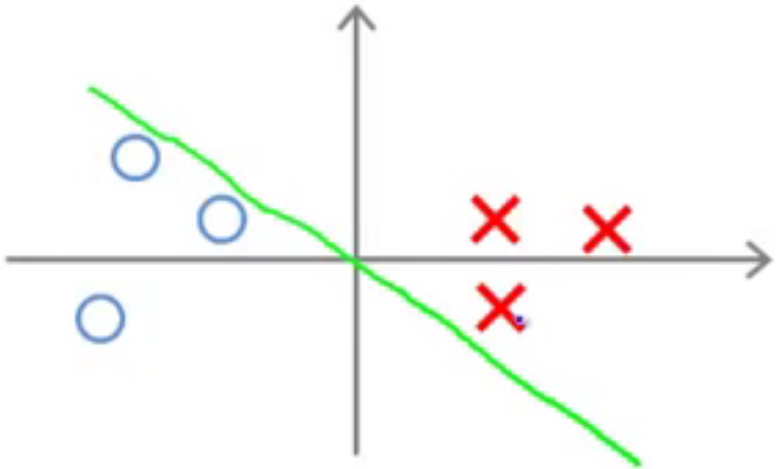


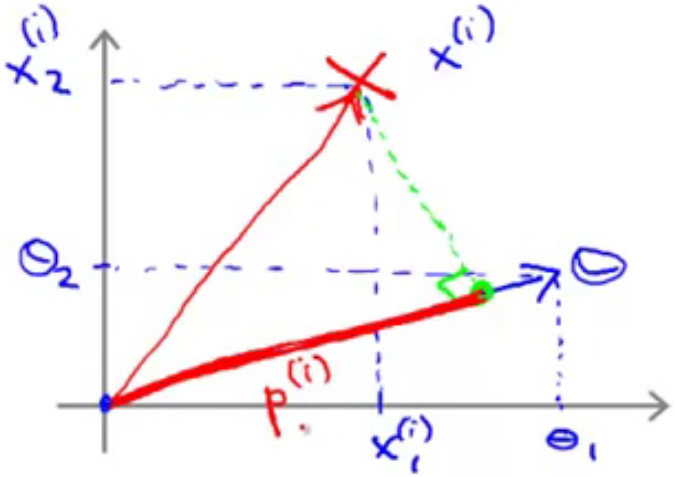
Machine Learning—Andrew Ng, Stanford University

Wektory nośne

$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$

$$\theta^T \mathbf{x}^{(i)} = p^{(i)} |\theta|$$

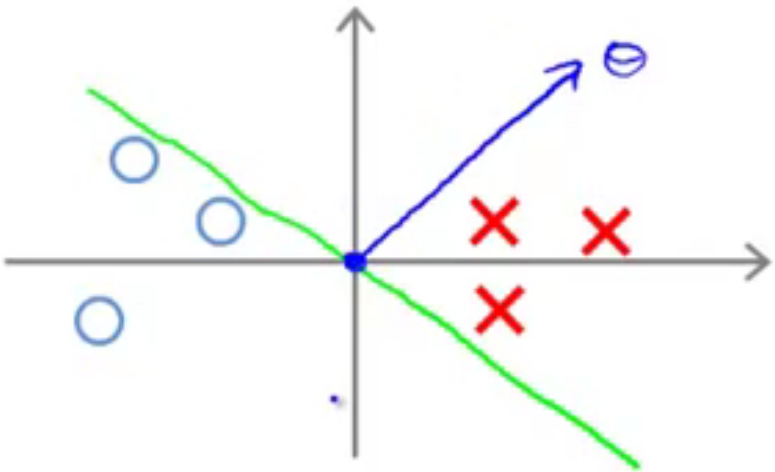


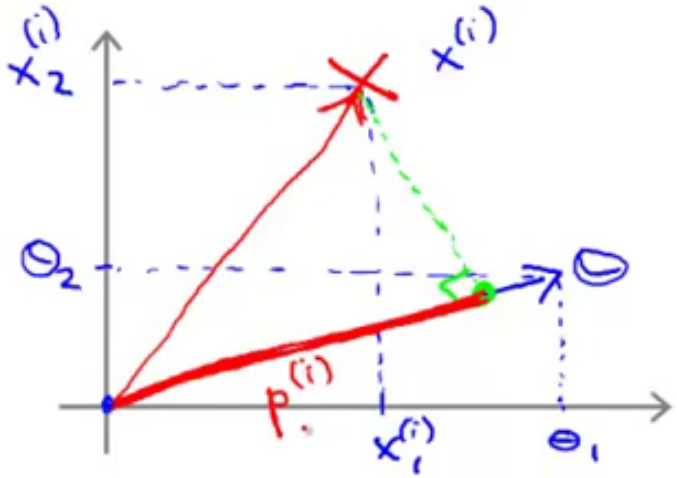


Wektory nośne

$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$

$$\theta^T \mathbf{x}^{(i)} = p^{(i)} |\theta|$$



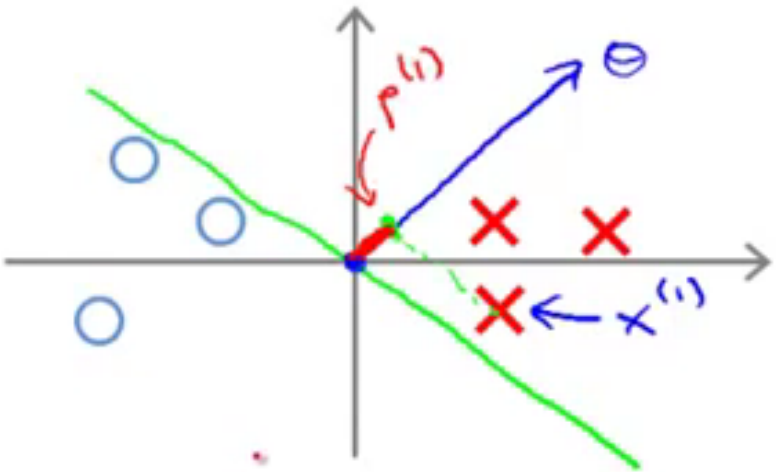


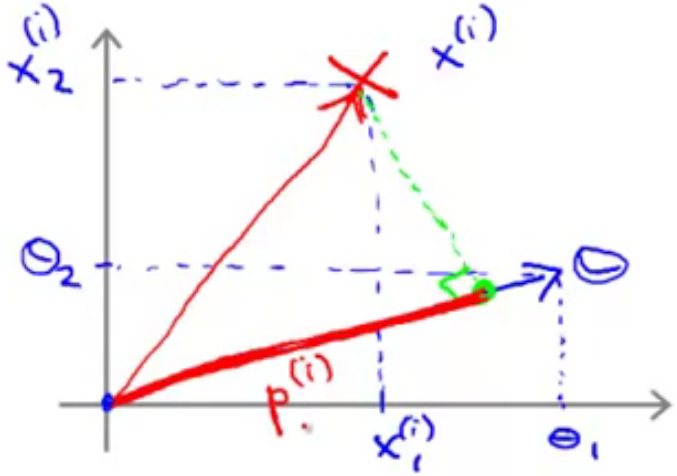
Machine Learning—Andrew Ng, Stanford University

Wektory nośne

$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$

$$\theta^T \mathbf{x}^{(i)} = p^{(i)} |\theta|$$



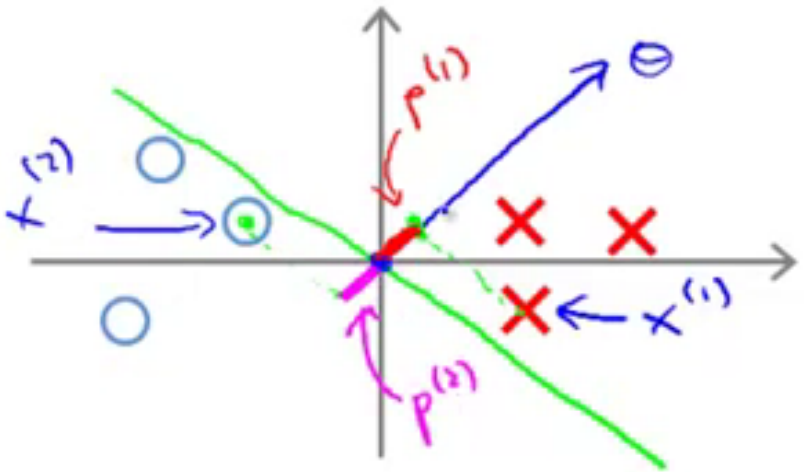


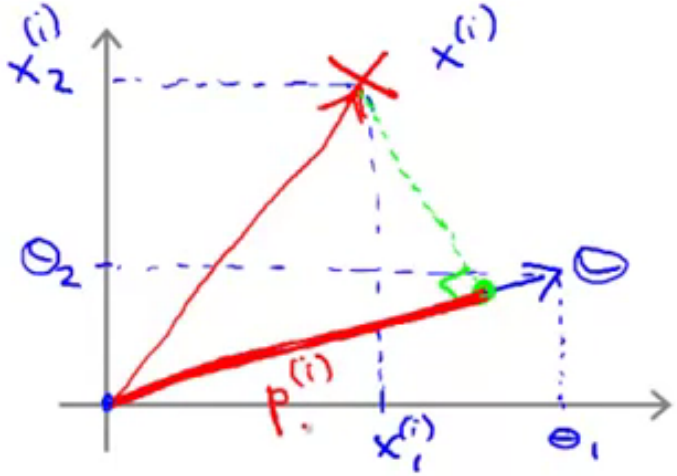
Machine Learning—Andrew Ng, Stanford University

Wektory nośne

$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$

$$\theta^T \mathbf{x}^{(i)} = p^{(i)} |\theta|$$



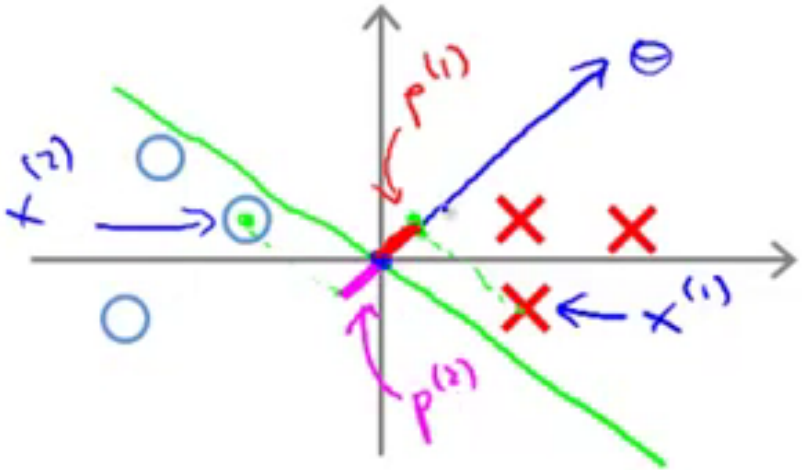


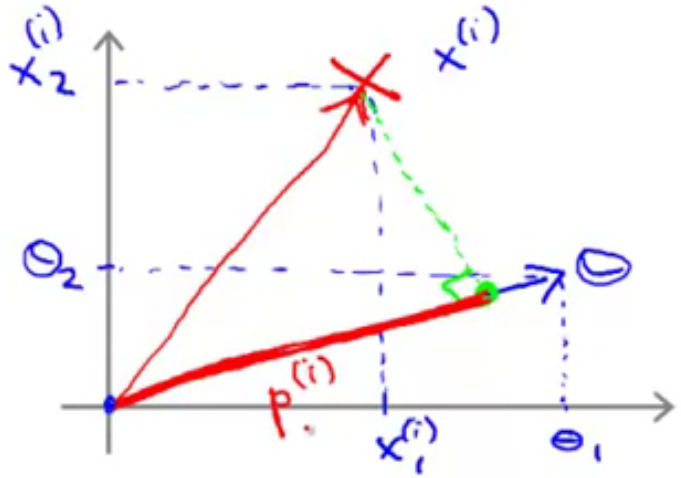
Machine Learning—Andrew Ng, Stanford University

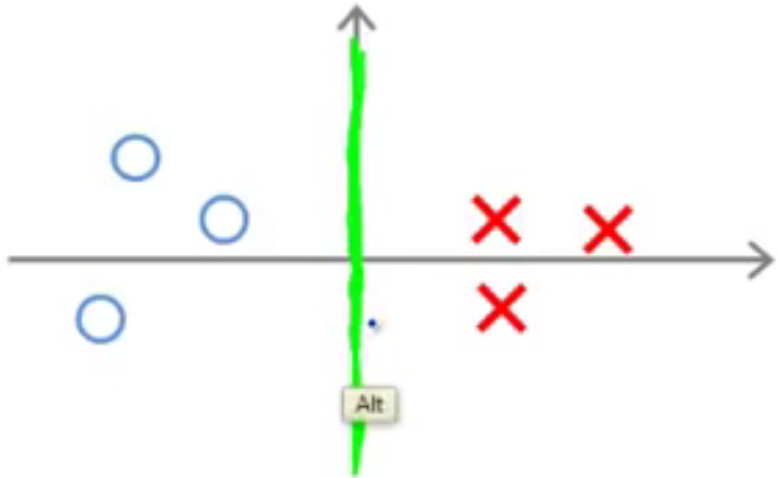
Wektory nośne

$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$

$$\theta^T \mathbf{x}^{(i)} = p^{(i)} |\theta|$$





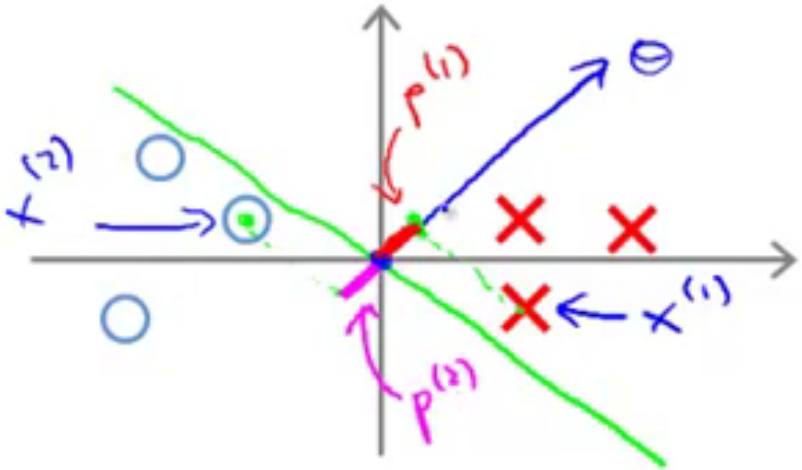


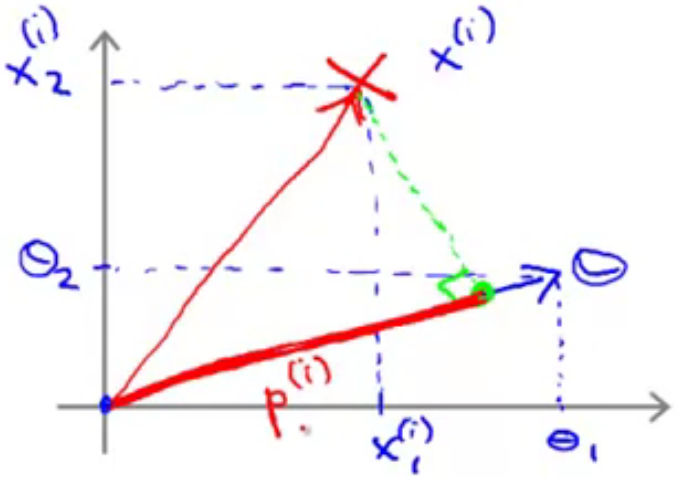
Machine Learning—Andrew Ng, Stanford University

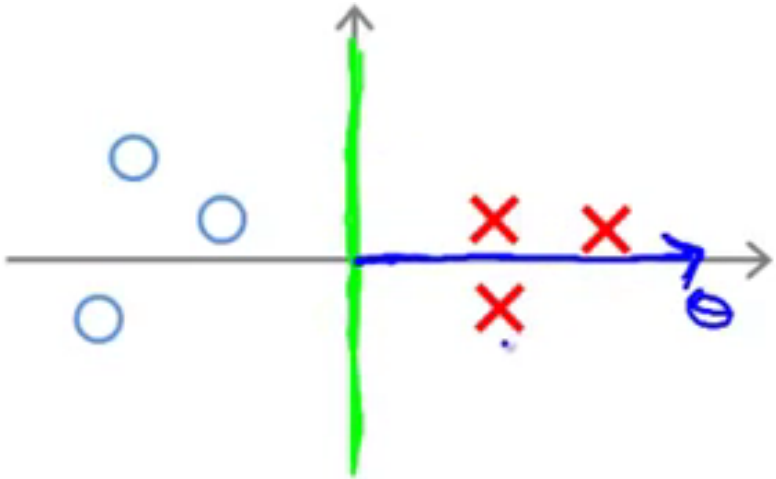
Wektory nośne

$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$

$$\theta^T \mathbf{x}^{(i)} = p^{(i)} |\theta|$$



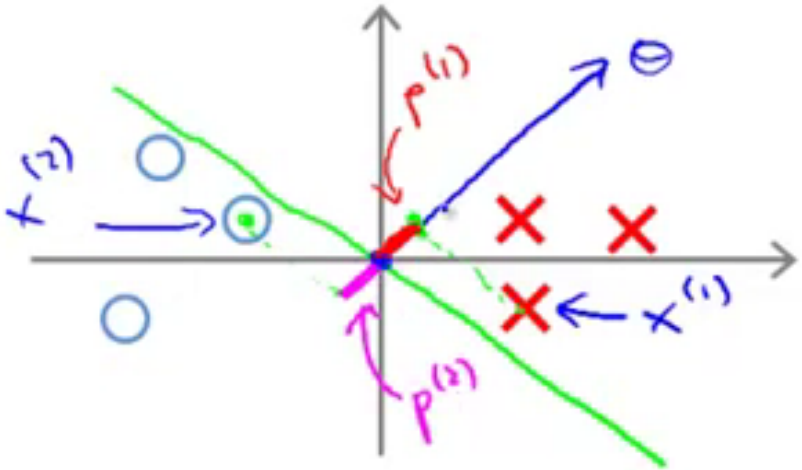


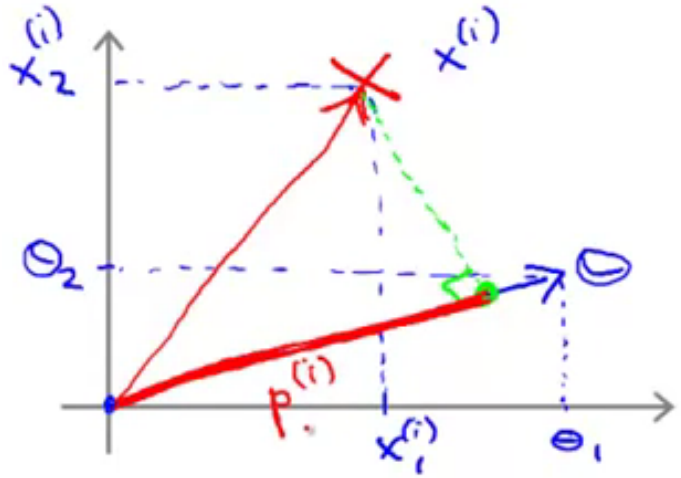


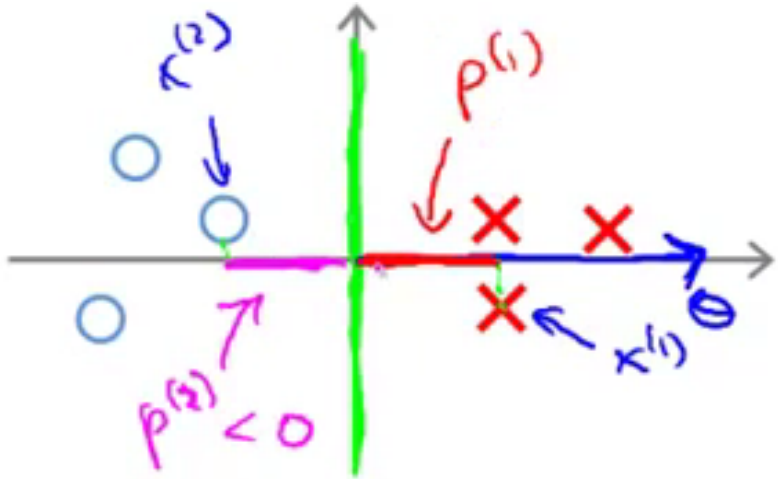
Wektory nośne

$$\theta^T \mathbf{x}^{(i)} \geqslant 1 \text{ dla } y^{(i)} = 1$$
$$\theta^T \mathbf{x}^{(i)} \leqslant -1 \text{ dla } y^{(i)} = 0$$

$$\theta^T \mathbf{x}^{(i)} = p^{(i)} |\theta|$$







Jądro SVM

Używane podczas poszukiwania nieliniowych granic decyzyjnych.

$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \ldots$$

Machine Learning—Andrew Ng, Stanford University

Jądro SVM

Używane podczas poszukiwania nieliniowych granic decyzyjnych.

$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \ldots$$

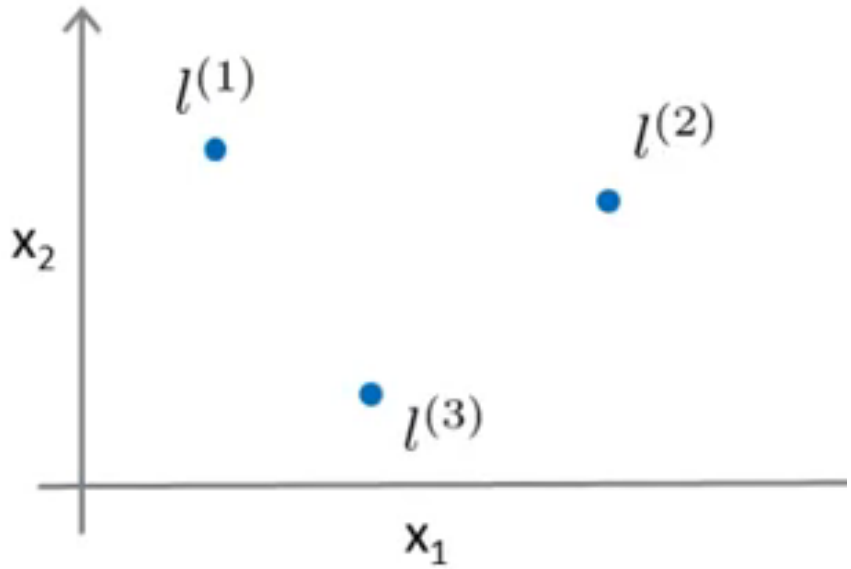
$$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \theta_4 f_4 + \theta_5 f_5 + \ldots$$

gdzie f-funkcja podobieństwa danej do znaczników (landmarks) używająca jądra

$f = similarity(x, l^{(i)}) = k(x, l^{(i)})$

Jądro

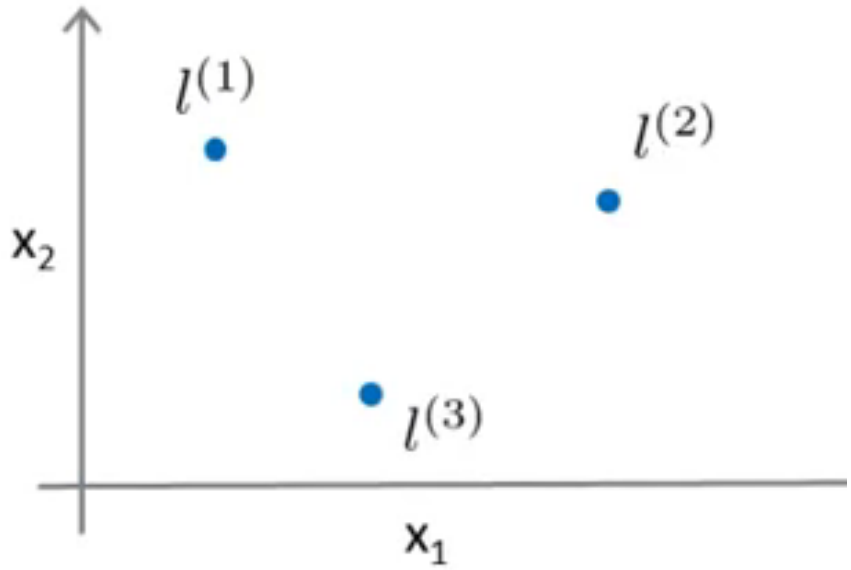
$$f = similarity(x, l^{(i)}) = k(x, l^{(i)})$$



Machine Learning—Andrew Ng, Stanford University

Jądro

$$f = similarity(x, l^{(i)}) = k(x, l^{(i)})$$



Jądro przekształca wektor przypadku x w nowy wektor f m-wymiarowy, gdzie m-wielkość zbioru treningowego.

Machine Learning—Andrew Ng, Stanford University

Jądro

Przykład jądra: gaussowskie

$$f_i = similarity(x, l^{(i)}) = \exp(-\frac{|x - l^{(i)}|^2}{2\sigma^2})$$

Jądro

Przykład jądra: gaussowskie

$$f_i = similarity(x, l^{(i)}) = \exp(-\frac{|x - l^{(i)}|^2}{2\sigma^2})$$

dla $x \approx l^{(i)}$
$f_i \approx 1$

Jądro

Przykład jądra: gaussowskie

$$f_i = similarity(x, l^{(i)}) = \exp(-\frac{|x - l^{(i)}|^2}{2\sigma^2})$$

dla $x \approx l^{(i)}$
$f_i \approx 1$

dla x dalekiego od $l^{(i)}$
$f_i \approx 0$

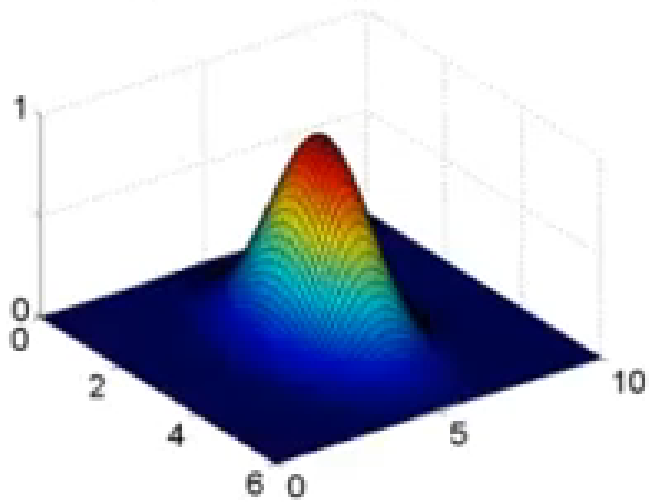
Jądro

Przykład jądra: gaussowskie

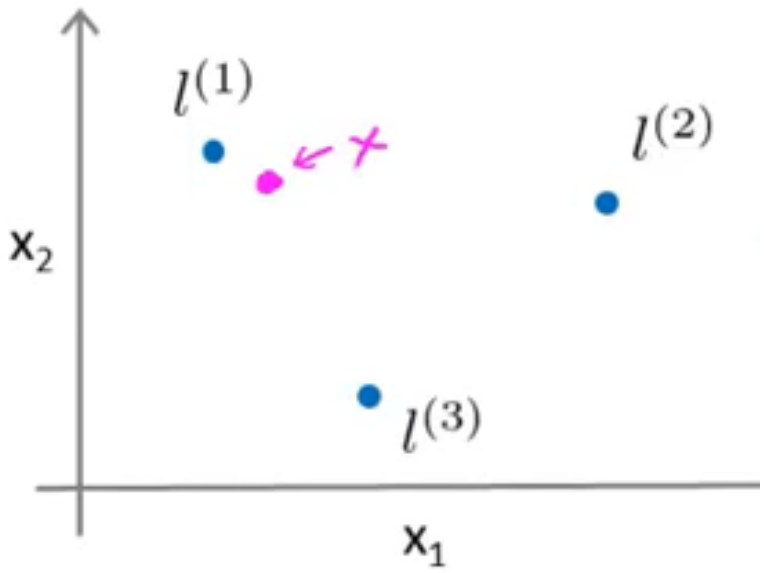
$$f_i = similarity(x, l^{(i)}) = \exp(-\frac{|x - l^{(i)}|^2}{2\sigma^2})$$

dla $x \approx l^{(i)}$
$f_i \approx 1$

dla x dalekiego od $l^{(i)}$
$f_i \approx 0$

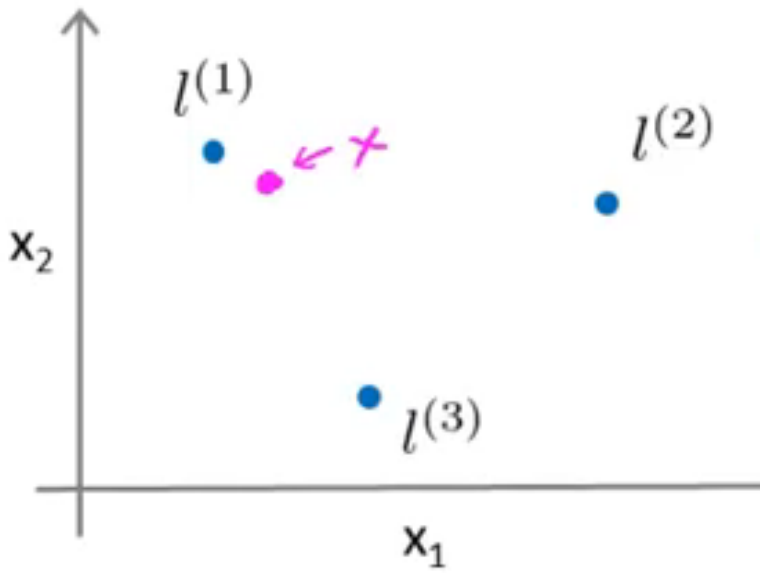


Jądro



Machine Learning—Andrew Ng, Stanford University

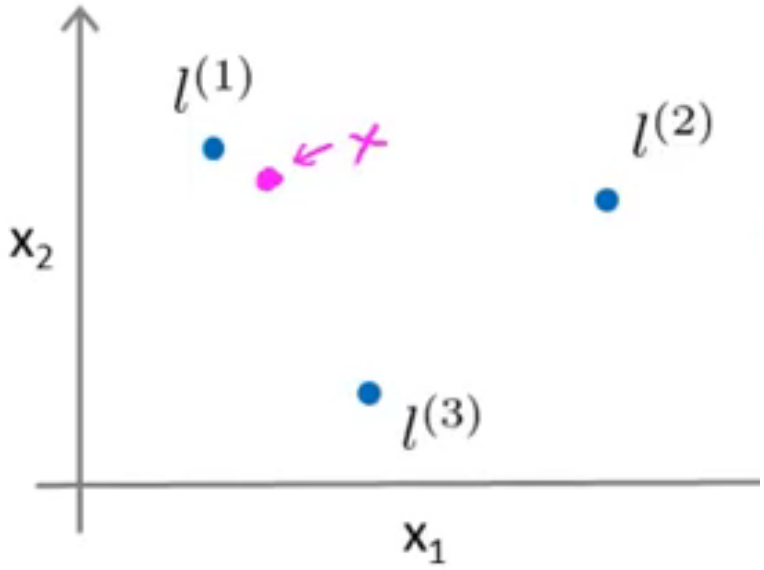
Jądro



$$x^{(i)} = \begin{bmatrix} f_1^{(i)} \\ f_2^{(i)} \\ \dots \\ f_m^{(i)} \end{bmatrix}$$

Machine Learning—Andrew Ng, Stanford University

Jądro



$$x^{(i)} = \begin{bmatrix} f_1^{(i)} \\ f_2^{(i)} \\ \dots \\ f_m^{(i)} \end{bmatrix}$$

Uwaga: wszystkie założenia zmieniają się z $\theta^T x$ na: $\theta^T f$

Machine Learning—Andrew Ng, Stanford University

SVM- Wybór parametrów

- C - duże C=overfitting, małe C=niedopasowanie,

SVM- Wybór parametrów

- C - duże C=overfitting, małe C=niedopasowanie,
- jądro (lub jego brak)

SVM- Wybór parametrów

- C - duże C=overfitting, małe C=niedopasowanie,
- jądro (lub jego brak)
- σ w jądrze gaussowskim,

Mocne strony SVM

Słabe strony SVM

Mocne strony SVM

- Potrafi poradzić sobie z outliersami i danymi nieseparowalnymi liniowo

Słabe strony SVM

Mocne strony SVM

- Potrafi poradzić sobie z outliersami i danymi nieseparowalnymi liniowo
- Minimalizacja funkcji kosztu wymaga mniej obliczeń niż w przypadku regresji logistycznej

Słabe strony SVM

Mocne strony SVM

- Potrafi poradzić sobie z outliersami i danymi nieseparowalnymi liniowo
- Minimalizacja funkcji kosztu wymaga mniej obliczeń niż w przypadku regresji logistycznej
- Stopień skomplikowania jest niezależny od liczby wymiarów

Słabe strony SVM

Mocne strony SVM

- Potrafi poradzić sobie z outliersami i danymi nieseparowalnymi liniowo
- Minimalizacja funkcji kosztu wymaga mniej obliczeń niż w przypadku regresji logistycznej
- Stopień skomplikowania jest niezależny od liczby wymiarów
- Nie jest czuły na przetrenowanie

Słabe strony SVM

Mocne strony SVM

- Potrafi poradzić sobie z outliersami i danymi nieseparowalnymi liniowo
- Minimalizacja funkcji kosztu wymaga mniej obliczeń niż w przypadku regresji logistycznej
- Stopień skomplikowania jest niezależny od liczby wymiarów
- Nie jest czuły na przetrenowanie
- Dzięki dopasowaniu jądra, algorytm osiąga dużą skuteczność w praktyce

Słabe strony SVM

Mocne strony SVM

- Potrafi poradzić sobie z outliersami i danymi nieseparowalnymi liniowo
- Minimalizacja funkcji kosztu wymaga mniej obliczeń niż w przypadku regresji logistycznej
- Stopień skomplikowania jest niezależny od liczby wymiarów
- Nie jest czuły na przetrenowanie
- Dzięki dopasowaniu jądra, algorytm osiąga dużą skuteczność w praktyce

Słabe strony SVM

- Długotrwały trening przez minimalizację funkcji

Mocne strony SVM

- Potrafi poradzić sobie z outliersami i danymi nieseparowalnymi liniowo
- Minimalizacja funkcji kosztu wymaga mniej obliczeń niż w przypadku regresji logistycznej
- Stopień skomplikowania jest niezależny od liczby wymiarów
- Nie jest czuły na przetrenowanie
- Dzięki dopasowaniu jądra, algorytm osiąga dużą skuteczność w praktyce

Słabe strony SVM

- Długotrwały trening przez minimalizację funkcji
- Brak możliwości wprowadzenia prior knowledge

Spis treści

1. Wprowadzenie do klasyfikacji
2. kNN - k Nearest Neighbours
3. Regresja logistyczna
4. SVM
5. **Naive Bayes**
6. Drzewa Decyzyjne
7. Random Forest

Naiwny klasyfikator Bayesa

Prosty klasyfikator probabilistyczny. Model prawdopodobieństwa można wprowadzić korzystając z **twierdzenia Bayesa**.

Naiwny klasyfikator Bayesa

Prosty klasyfikator probabilistyczny. Model prawdopodobieństwa można wprowadzić korzystając z **twierdzenia Bayesa**.

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

gdzie: $P(A|B)$ - prawdopodobieństwo warunkowe, prawdopodobieństwo zajścia zdarzenia A jeśli zajdzie zdarzenie B.

Naiwny klasyfikator Bayesa

Prosty klasyfikator probabilistyczny. Model prawdopodobieństwa można wprowadzić korzystając z **twierdzenia Bayesa**.

Naiwność - założenie o *wzajemnej niezależności predyktorów* (cech) - często błędne.

Naiwny klasyfikator Bayesa

Prosty klasyfikator probabilistyczny. Model prawdopodobieństwa można wprowadzić korzystając z **twierdzenia Bayesa**.

Naiwność - założenie o *wzajemnej niezależności predyktorów* (cech) - często błędne.

Naiwne klasyfikatory bayesowskie można uczyć z nadzorem. Estymacja parametru używa często **metody maksymalnego prawdopodobieństwa a posteriori**.

Naiwny klasyfikator Bayesa

Prosty klasyfikator probabilistyczny. Model prawdopodobieństwa można wprowadzić korzystając z **twierdzenia Bayesa**.

Naiwność - założenie o *wzajemnej niezależności predyktorów* (cech) - często błędne.

Naiwne klasyfikatory bayesowskie można uczyć z nadzorem. Estymacja parametru używa często **metody maksymalnego prawdopodobieństwa a posteriori**.

Pomimo uproszczonych założeń, klasyfikatory te często dają bardzo dobre wyniki.

Naiwny model probabilistyczny Bayesa

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Naiwny model probabilistyczny Bayesa

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

$$P(C|F_1, ..., F_n) = \frac{P(F_1, ..., F_n|C)P(C)}{P(F_1, ..., F_n)}$$

gdzie $P(C|F_1, ..., F_n)$ - prawdopodobieństwo przynależności do klasy C pod warunkiem cech $F_1, ...F_n$

Naiwny model probabilistyczny Bayesa

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

$$P(C|F_1, ..., F_n) = \frac{P(F_1, ..., F_n|C)P(C)}{P(F_1, ..., F_n)}$$

gdzie $P(C|F_1, ..., F_n)$ - prawdopodobieństwo przynależności do klasy C pod warunkiem cech $F_1, ...F_n$

$$P(C|F_1, ..., F_n) = P(F_1|C)P(F_2|C)..., P(F_n|C)P(C) = P(C)\prod_{i=1}^{n} P(F_i|C)$$

Naiwny model probabilistyczny Bayesa

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

$$P(C|F_1, ..., F_n) = \frac{P(F_1, ..., F_n|C)P(C)}{P(F_1, ..., F_n)}$$

gdzie $P(C|F_1, ..., F_n)$ - prawdopodobieństwo przynależności do klasy C pod warunkiem cech $F_1, ...F_n$

$$P(C|F_1, ..., F_n) = P(F_1|C)P(F_2|C)..., P(F_n|C)P(C) = P(C)\prod_{i=1}^{n} P(F_i|C)$$

$$P(C|F_1, ..., F_n) = \frac{1}{Z}P(C)\prod_{i=1}^{n} P(F_i|C)$$

gdzie Z - współczynnik skalowania zależny od $F_1, ..., F_n$

Naiwny model probabilistyczny Bayesa

Model uczy się:
$P(F|C)$ oraz $P(C)$ - inf. a priori

Na podstawie nauczonych prawdopodobieństw można obliczyć prawdopodobieństwo przynależności do klasy $C = j$ w danym przypadku (dla danych cech $F = F_1, ..., F_n$):

$$P(C = j|F) = \frac{P(F|C = j)P(C = j)}{P(F)}$$

Naiwny model probabilistyczny Bayesa - Przykład

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

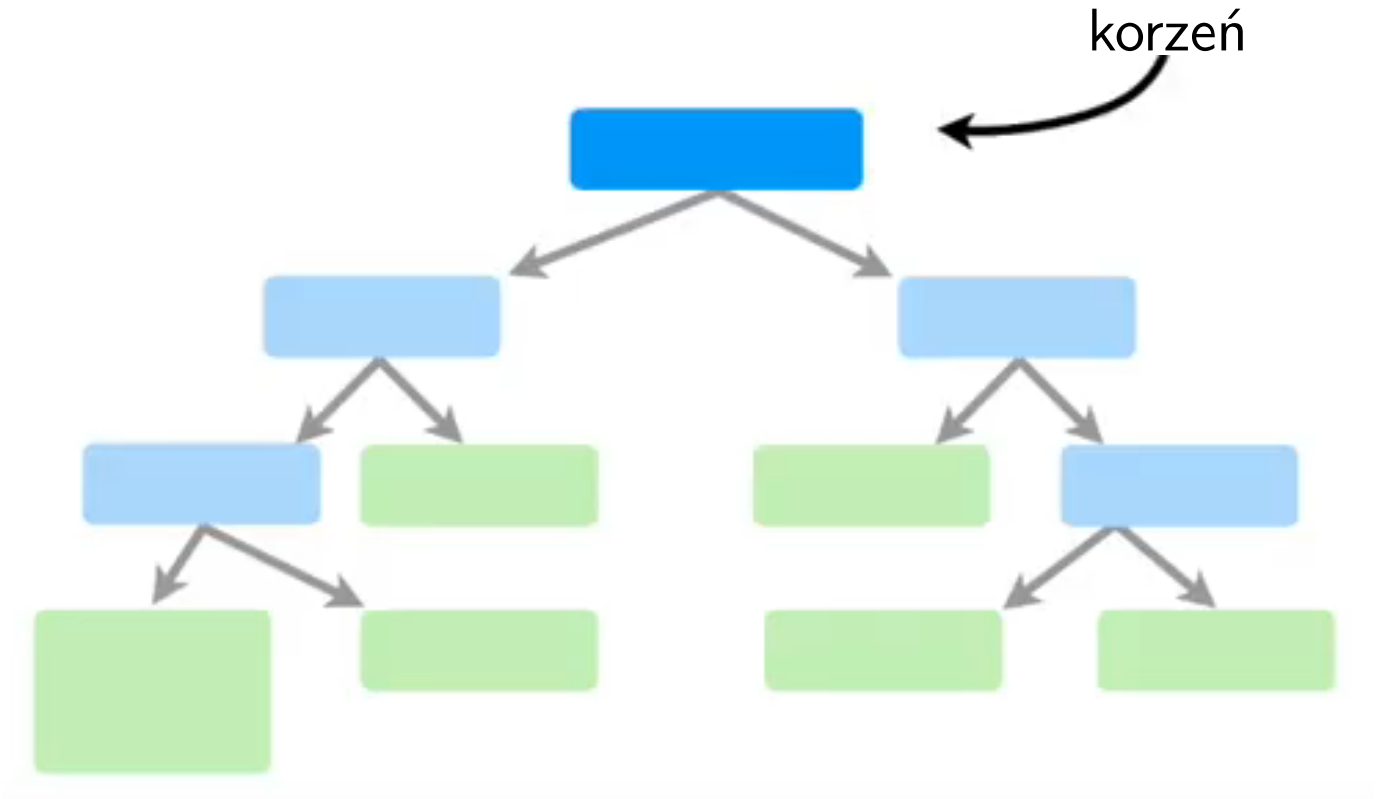
Spis treści

1. Wprowadzenie do klasyfikacji
2. kNN - k Nearest Neighbours
3. Regresja logistyczna
4. SVM
5. Naive Bayes
6. **Drzewa decyzyjne**
7. Random Forest

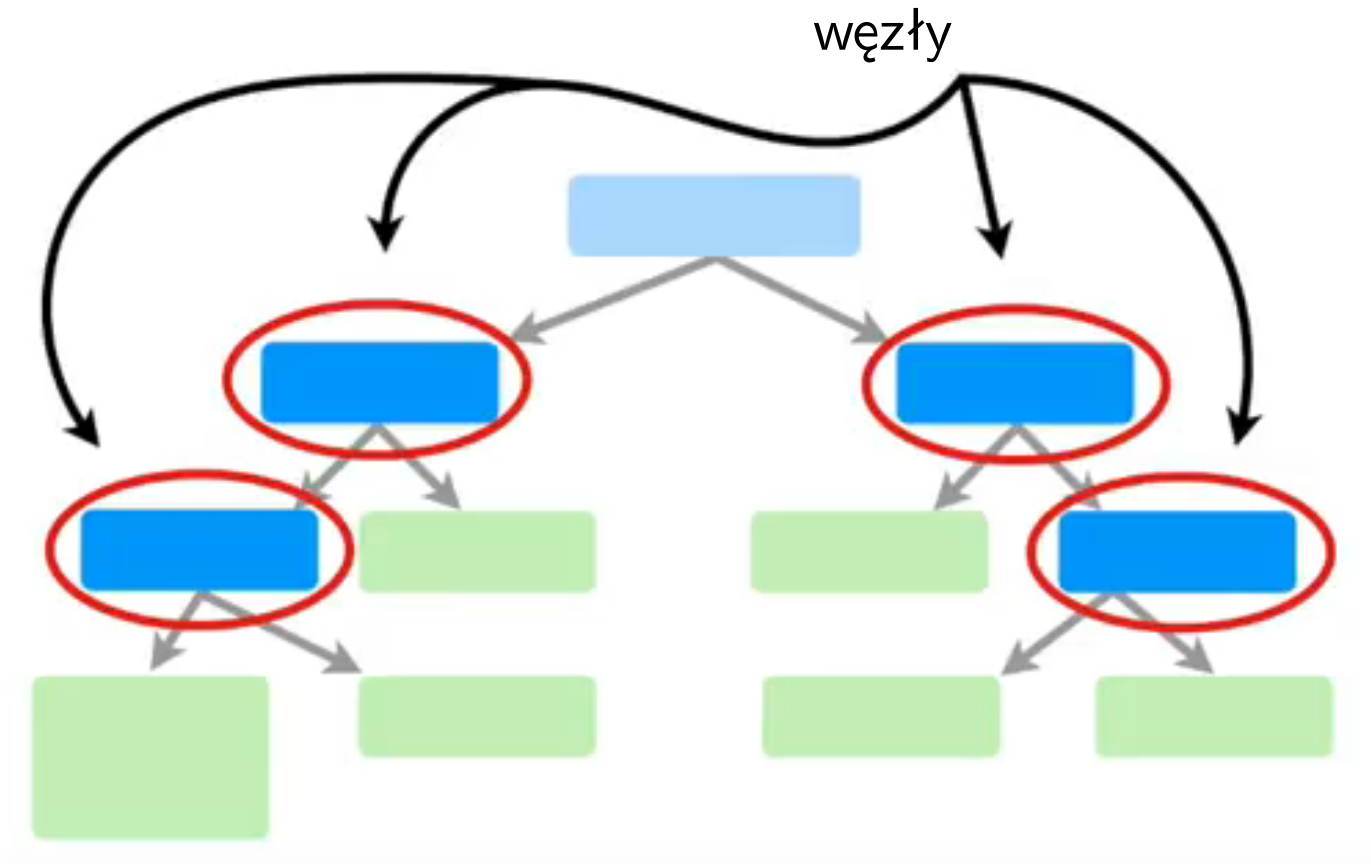
Drzewa decyzyjne

Algorytm zaczerpnięty z teorii decyzji, gdzie wspomaga procesy decyzyjne. W uczeniu maszynowym służy do pozyskiwania wiedzy ze zboru uczącego.

Budowa drzewa

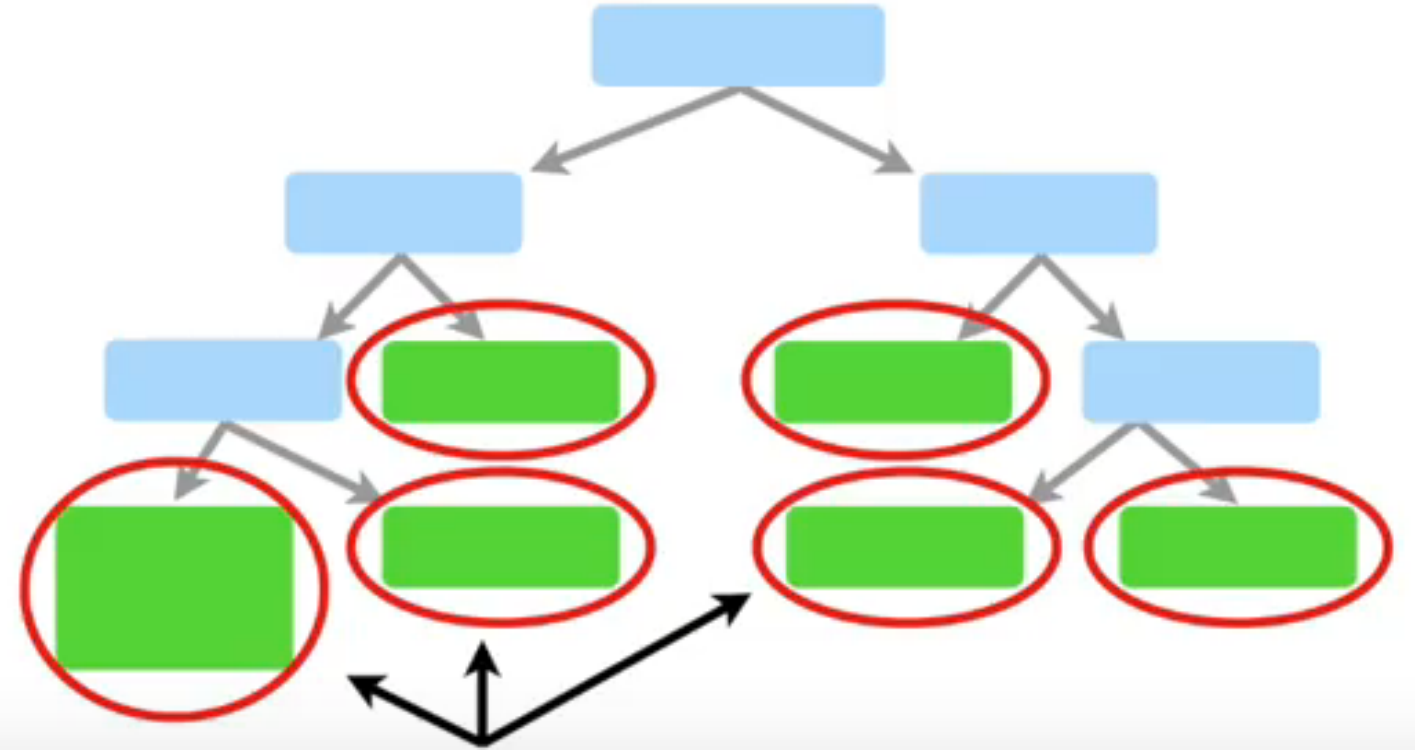


Budowa drzewa



StatQuest: Decision Trees

Budowa drzewa



liście

StatQuest: Decision Trees

Reguły drzew decyzyjnych

- Drzewo nie ma pętli (cykli),
- Istnieje tylko jedna ścieżka między dwoma różnymi węzłami,
- Reprezentuje podział zbioru na klasy:
  - węzły opisują sposób podziału,
  - liście to klasy
  - krawędzie reprezentują wartości cech, na podstawie których dokonano podziału

Drzewa decyzyjne

Rodzaje cech możliwych do klasyfikacji przez drzewa:

- cechy binarne (tak, nie)
- cechy ilościowe
- cechy jakościowe (ranking)
- cechy wielokrotnego wyboru.

Drzewa decyzyjne - Algorytm

1. utwórz korzeń drzewa

Drzewa decyzyjne - Algorytm

1. utwórz korzeń drzewa
2. utwórz element drzewa, który:

Drzewa decyzyjne - Algorytm

1. utwórz korzeń drzewa
2. utwórz element drzewa, który:
3. jest liściem, jeżeli wszystkie przykłady ze zbioru uczącego są pozytywne lub negatywne

Drzewa decyzyjne - Algorytm

1. utwórz korzeń drzewa
2. utwórz element drzewa, który:
3. jest liściem, jeżeli wszystkie przykłady ze zbioru uczącego są pozytywne lub negatywne
4. jest węzłem w przeciwnym wypadku

Drzewa decyzyjne - Algorytm

1. utwórz korzeń drzewa
2. utwórz element drzewa, który:
3. jest liściem, jeżeli wszystkie przykłady ze zbioru uczącego są pozytywne lub negatywne
4. jest węzłem w przeciwnym wypadku
   - na węzeł wybierz cechę, który najlepiej klasyfikuje zbiór uczący

Drzewa decyzyjne - Algorytm

1. utwórz korzeń drzewa
2. utwórz element drzewa, który:
3. jest liściem, jeżeli wszystkie przykłady ze zbioru uczącego są pozytywne lub negatywne
4. jest węzłem w przeciwnym wypadku
   - na węzeł wybierz cechę, który najlepiej klasyfikuje zbiór uczący
   - dla każdej wartości cechy (węzła) stwórz gałąź i rozdziel zbiór uczący według tej cechy

Drzewa decyzyjne - Algorytm

1. utwórz korzeń drzewa
2. utwórz element drzewa, który:
3. jest liściem, jeżeli wszystkie przykłady ze zbioru uczącego są pozytywne lub negatywne
4. jest węzłem w przeciwnym wypadku
   - na węzeł wybierz cechę, który najlepiej klasyfikuje zbiór uczący
   - dla każdej wartości cechy (węzła) stwórz gałąź i rozdziel zbiór uczący według tej cechy
   - jeśli gałąź jest pusta, utwórz liść z wartością, jaką najczęściej przyjmuje cecha

Drzewa decyzyjne - Algorytm

1. utwórz korzeń drzewa
2. utwórz element drzewa, który:
3. jest liściem, jeżeli wszystkie przykłady ze zbioru uczącego są pozytywne lub negatywne
4. jest węzłem w przeciwnym wypadku
   - na węzeł wybierz cechę, który najlepiej klasyfikuje zbiór uczący
   - dla każdej wartości cechy (węzła) stwórz gałąź i rozdziel zbiór uczący według tej cechy
   - jeśli gałąź jest pusta, utwórz liść z wartością, jaką najczęściej przyjmuje cecha
   - w przeciwnym wypadku przejdź do kroku 4.

Drzewa decyzyjne - Algorytm

Miara wyboru cechy, która najlepiej klasyfikuje zbiór uczący:

Miara wyboru cechy, która najlepiej klasyfikuje zbiór uczący:

- **Entropia (Z)**
  charakteryzuje zanieczyszczenie zbioru uczącego
  $Entropia(Z) = -p_+ log_2 p_+ - p_- log_2 p_-$

Miara wyboru cechy, która najlepiej klasyfikuje zbiór uczący:

- **Entropia (Z)**
  charakteryzuje zanieczyszczenie zbioru uczącego
  $Entropia(Z) = -p_+ log_2 p_+ - p_- log_2 p_-$

  Przykład: dla 14 przykładów (9 pozytywnych i 5 negatywnych)
  $Entropia([9+, 5-]) = -\frac{9}{14} log_2(\frac{9}{14}) - \frac{5}{14} log_2(\frac{5}{14}) = 0,94$

Drzewa decyzyjne - Algorytm

Miara wyboru cechy, która najlepiej klasyfikuje zbiór uczący:

- **Entropia (Z)**
  charakteryzuje zanieczyszczenie zbioru uczącego
  $Entropia(Z) = -p_+ log_2 p_+ - p_- log_2 p_-$

  Przykład: dla 14 przykładów (9 pozytywnych i 5 negatywnych)
  $Entropia([9+, 5-]) = -\frac{9}{14} log_2(\frac{9}{14}) - \frac{5}{14} log_2(\frac{5}{14}) = 0,94$

  Własności entropii:
  - Entropia=0 - Wszystkie rozpatrywane przykłady należą do tej samej grupy
  - Entropia=1 - przykładów pozytywnych jest tyle co negatywnych
  - Entropia<1 - liczby pozytywnych i negatywnych przykładów nie są sobie równe

Drzewa decyzyjne - Algorytm

Miara wyboru cechy, która najlepiej klasyfikuje zbiór uczący:

- **Entropia (Z)**
- **Współczynnik Czerwińskiego**

Drzewa decyzyjne - Algorytm

Miara wyboru cechy, która najlepiej klasyfikuje zbiór uczący:

- **Entropia (Z)**
- **Współczynnik Czerwińskiego**
  Obliczany z tablice wielodzielczych
  Wsp. Czerwińskiego przyjmuje wartości ze zbioru 0,1.

Miara wyboru cechy, która najlepiej klasyfikuje zbiór uczący:

- **Entropia (Z)**
- **Współczynnik Czerwińskiego**
  Obliczany z tablice wielodzielczych
  Wsp. Czerwińskiego przyjmuje wartości ze zbioru 0,1.

  Własności:
  - wartość 1 - pełna zależność między cechami
  - wartość 0 - brak zależności między cechami

Drzewa decyzyjne - Algorytm

Miara wyboru cechy, która najlepiej klasyfikuje zbiór uczący:

- **Entropia (Z)**
- **Współczynnik Czerwińskiego**
- **Współczynnik Giniego**
  Jakość podziału (zanieczyszczenie) obliczana dla każdego z liści ze wzoru:
  $Wspczynnik_{Giniego} = 1 - \sum_{j=1}^{n}(p_j)^2$

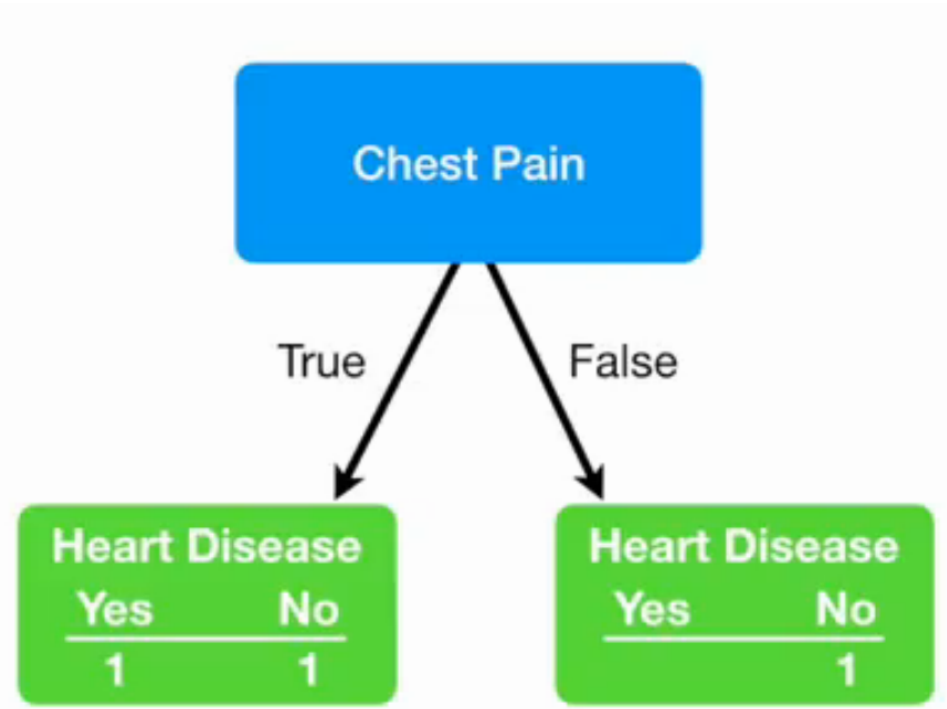
Drzewa decyzyjne - Przykład

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...

StatQuest: Decision Trees

Drzewa decyzyjne - Przykład

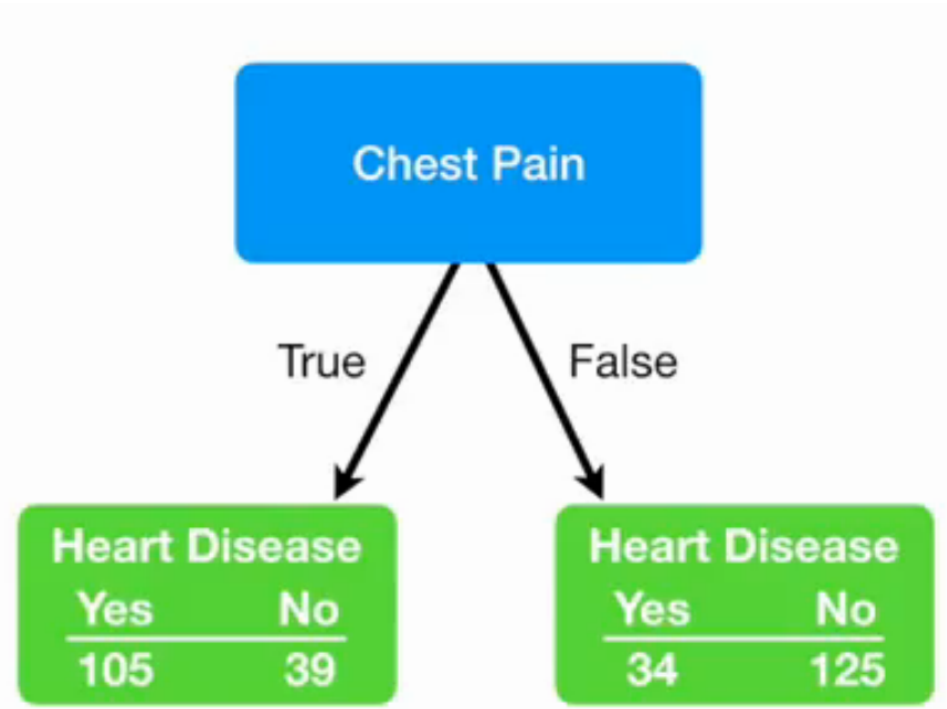
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



StatQuest: Decision Trees

Drzewa decyzyjne - Przykład

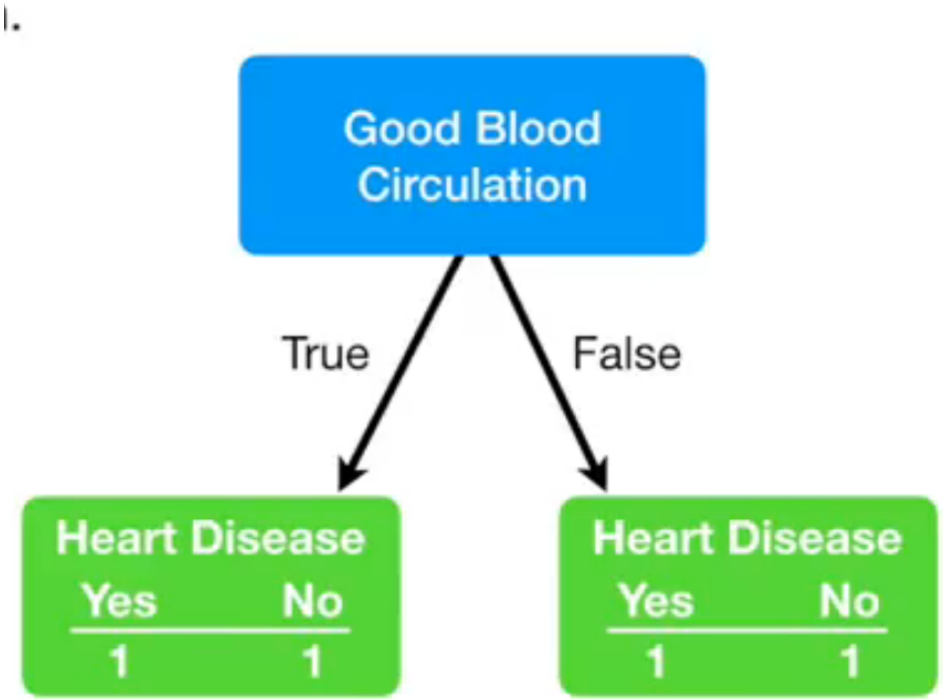
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



StatQuest: Decision Trees

Drzewa decyzyjne - Przykład

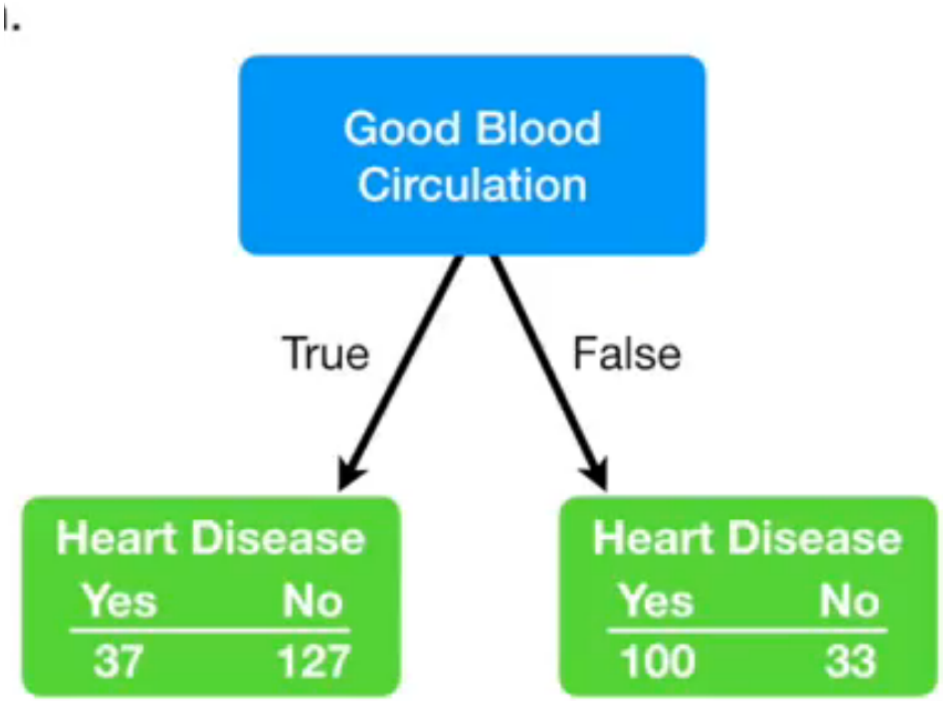
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



StatQuest: Decision Trees

Drzewa decyzyjne - Przykład

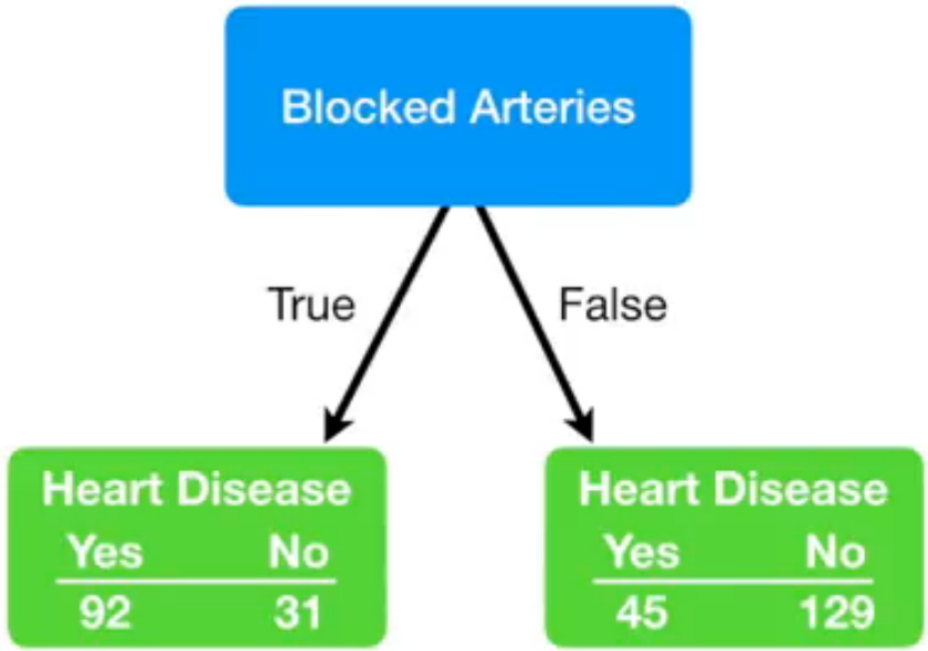
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



StatQuest: Decision Trees

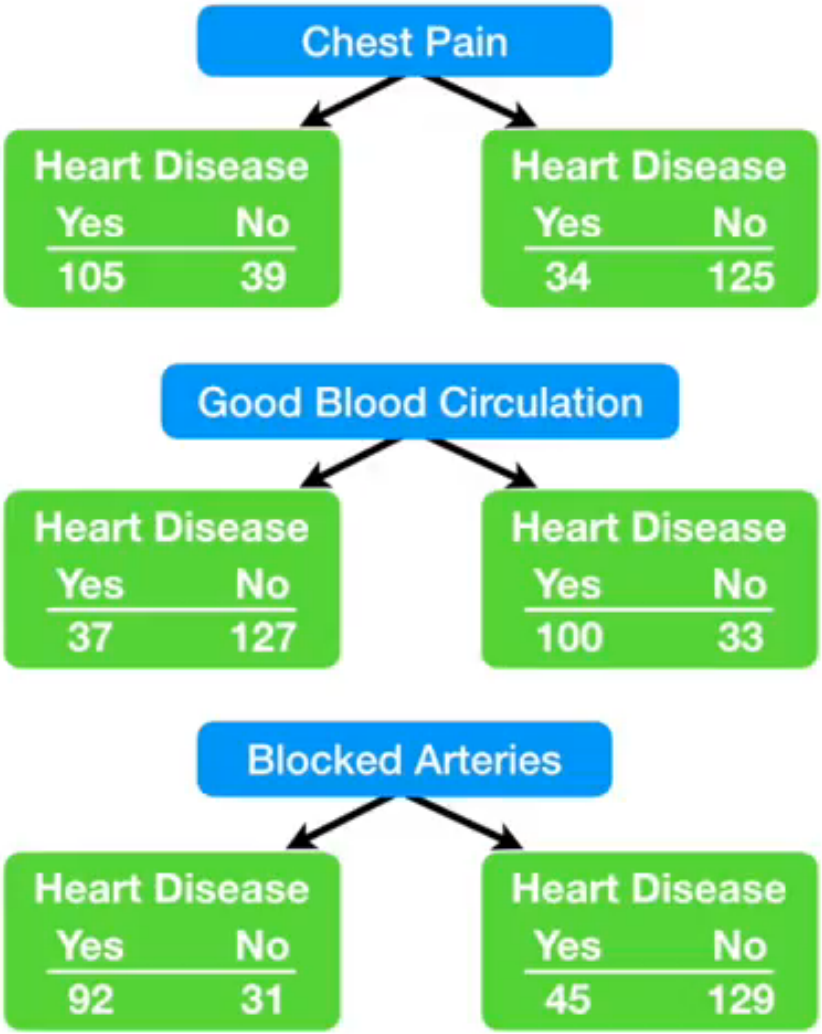
Drzewa decyzyjne - Przykład

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



StatQuest: Decision Trees

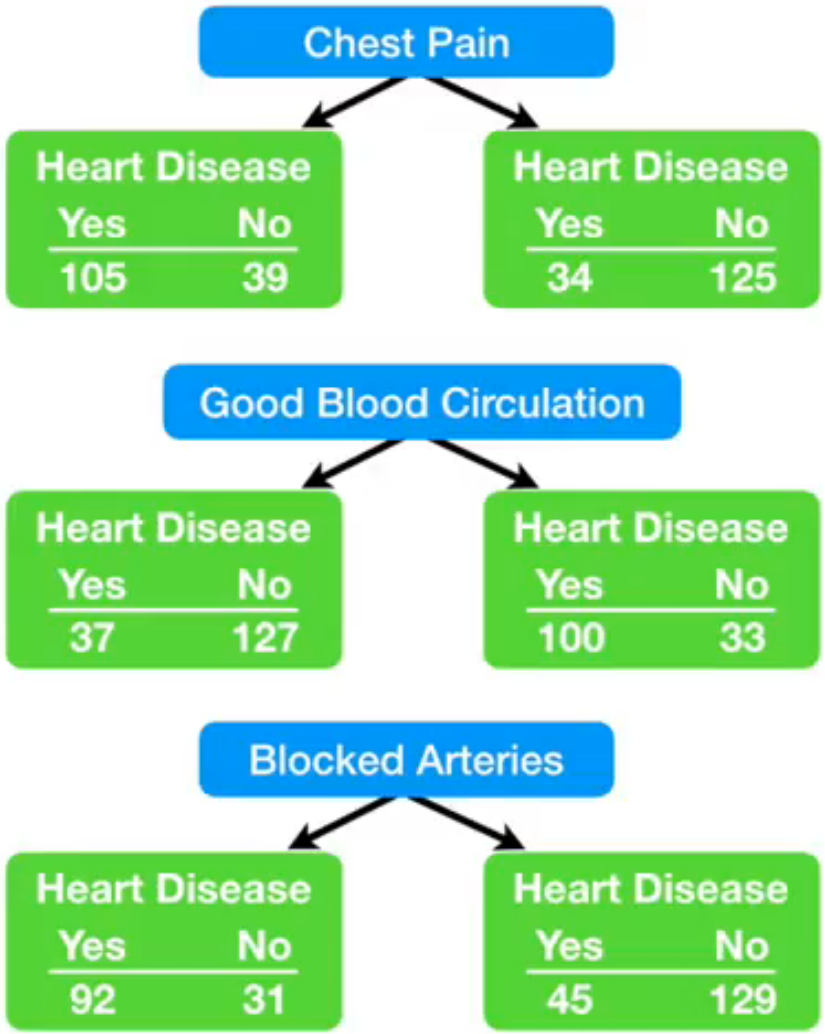
Drzewa decyzyjne - Przykład



StatQuest: Decision Trees

Drzewa decyzyjne - Przykład

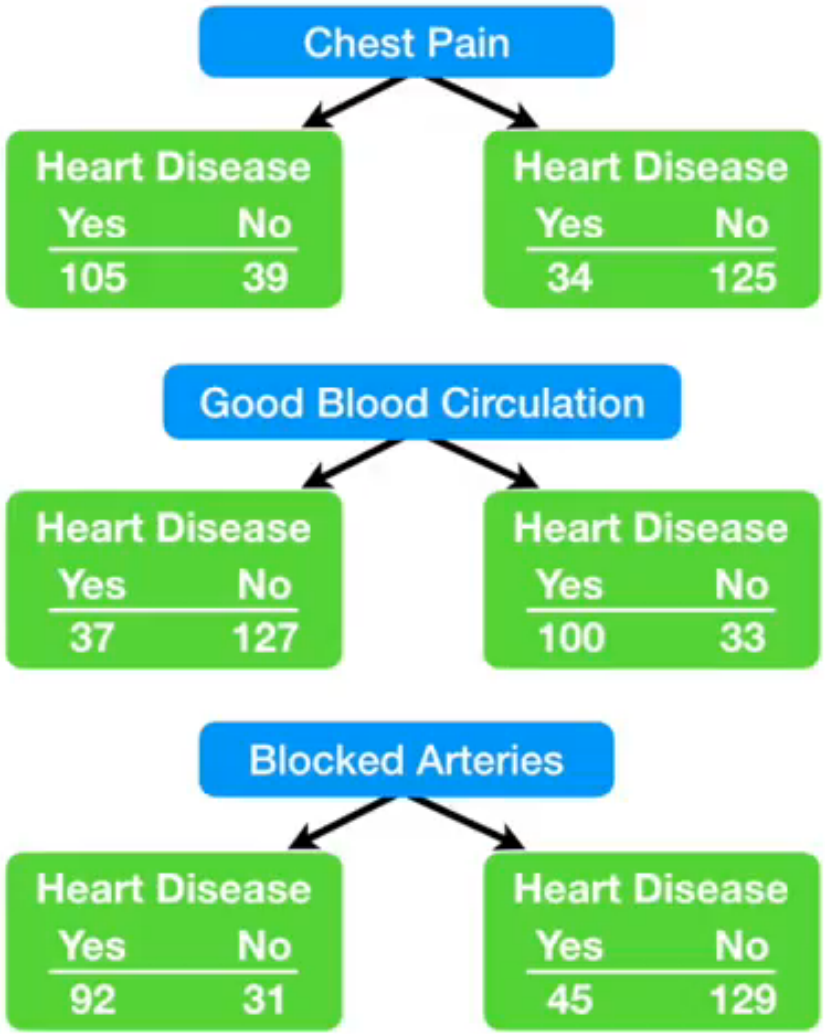
$$Wspczynnik_{Giniego} =$$
$$1 - (prawdopodobiestwo"\ Tak"\ )^2 - (Prawdopodobiestwo"\ Nie"\ )^2$$



$$1 - (\frac{105}{105+39})^2 - (\frac{39}{105+39})^2) = 0,395$$

Drzewa decyzyjne - Przykład

$$Wspczynnik_{Giniego} =$$
$$1 - (prawdopodobiestwo"\ Tak"\ )^2 - (Prawdopodobiestwo"\ Nie"\ )^2$$

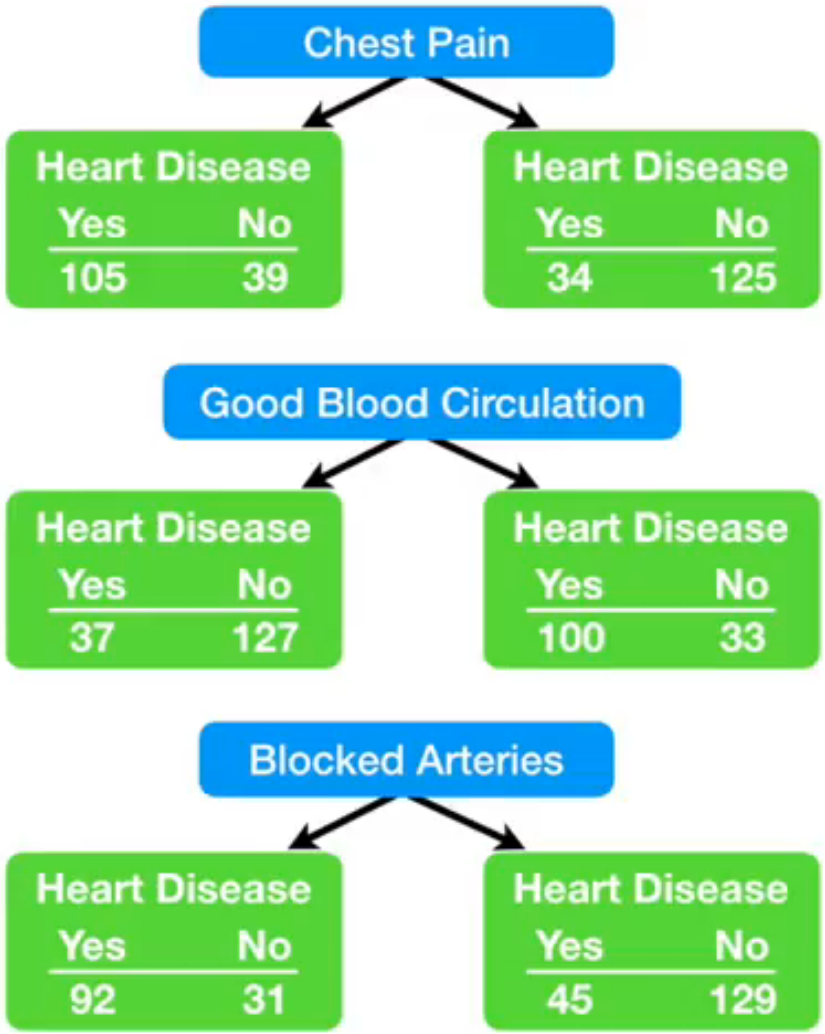


$$1 - (\frac{105}{105+39})^2 - (\frac{39}{105+39})^2) = 0,395$$

$$1 - (\frac{34}{34+125})^2 - (\frac{125}{34+125})^2) = 0,336$$

Drzewa decyzyjne - Przykład

$Wspczynnik_{Giniego} =$
$1 - (prawdopodobiestwo"\ Tak")^2 - (Prawdopodobiestwo"\ Nie")^2$



$1 - (\frac{105}{105+39})^2 - (\frac{39}{105+39})^2) = 0,395$

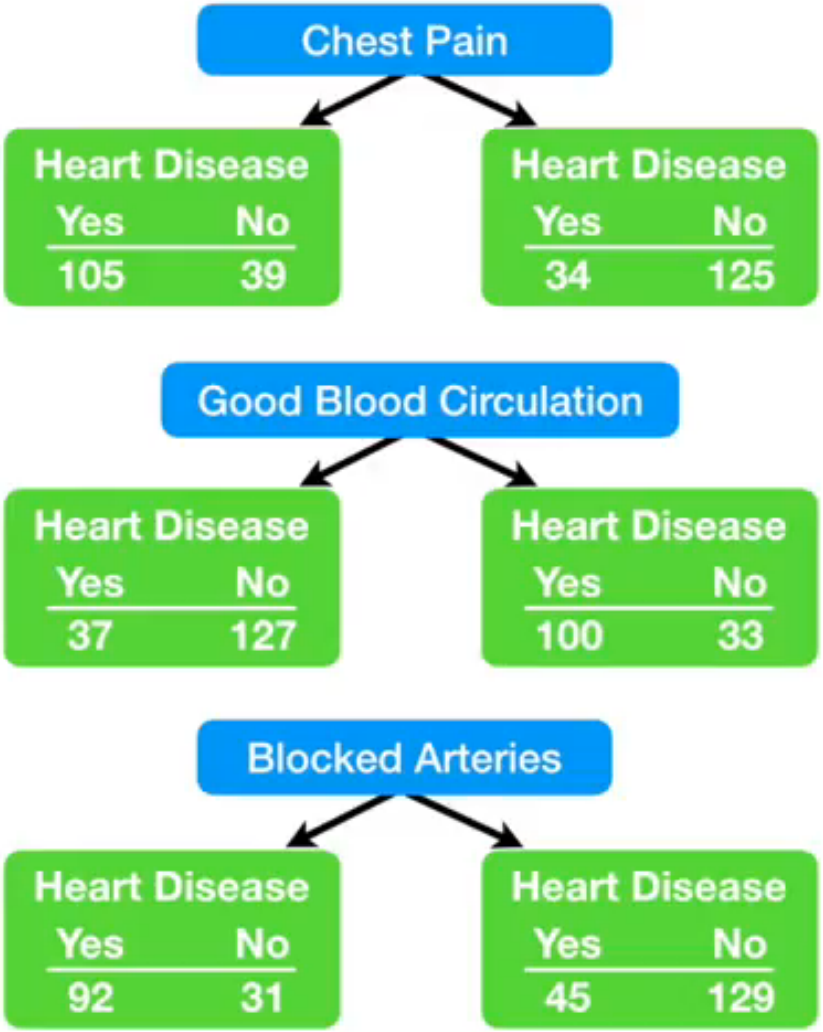
$1 - (\frac{34}{34+125})^2 - (\frac{125}{34+125})^2) = 0,336$

Całkowita jakość podziału to ważona średnia jakości podziału obu liści:

$(\frac{144}{144+159})0,395 +$
$(\frac{159}{144+159})0,336 = 0,364$

Drzewa decyzyjne - Przykład

$$Wspczynnik_{Giniego} = 1 - (prawdopodobiestwo" Tak")^2 - (Prawdopodobiestwo" Nie")^2$$



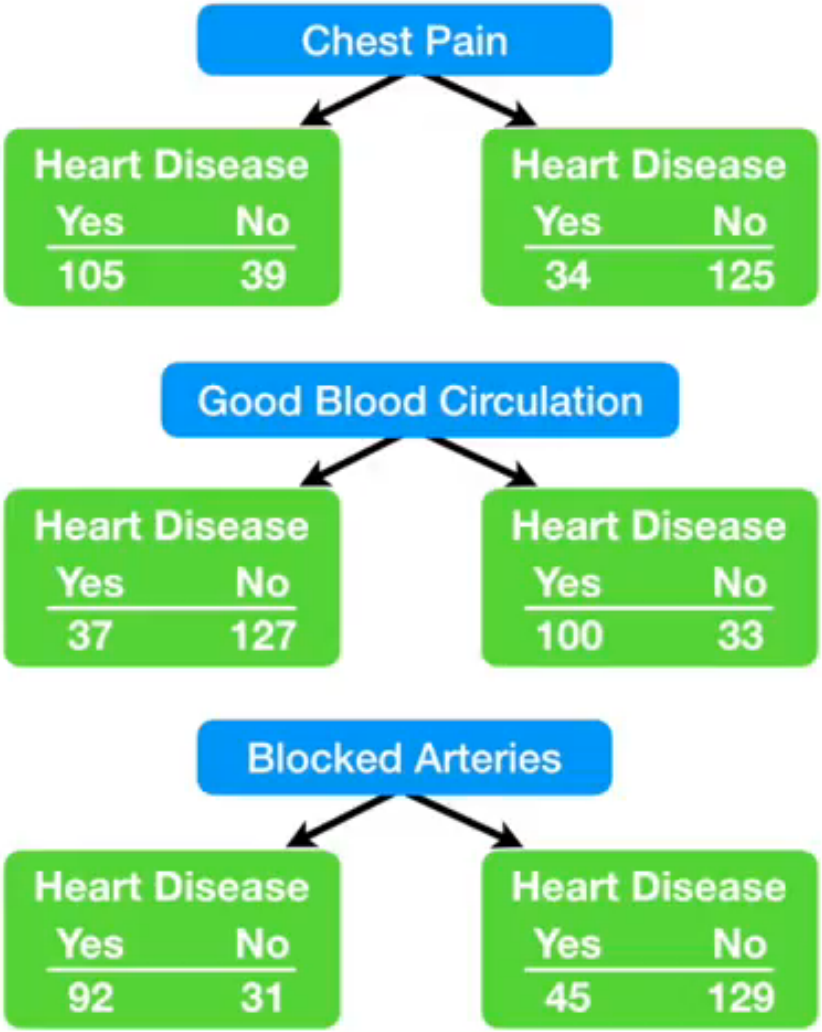
0,395

0,336

0,364

Drzewa decyzyjne - Przykład

$$Wspczynnik_{Giniego} = 1 - (prawdopodobiestwo" Tak")^2 - (Prawdopodobiestwo" Nie")^2$$



0,395

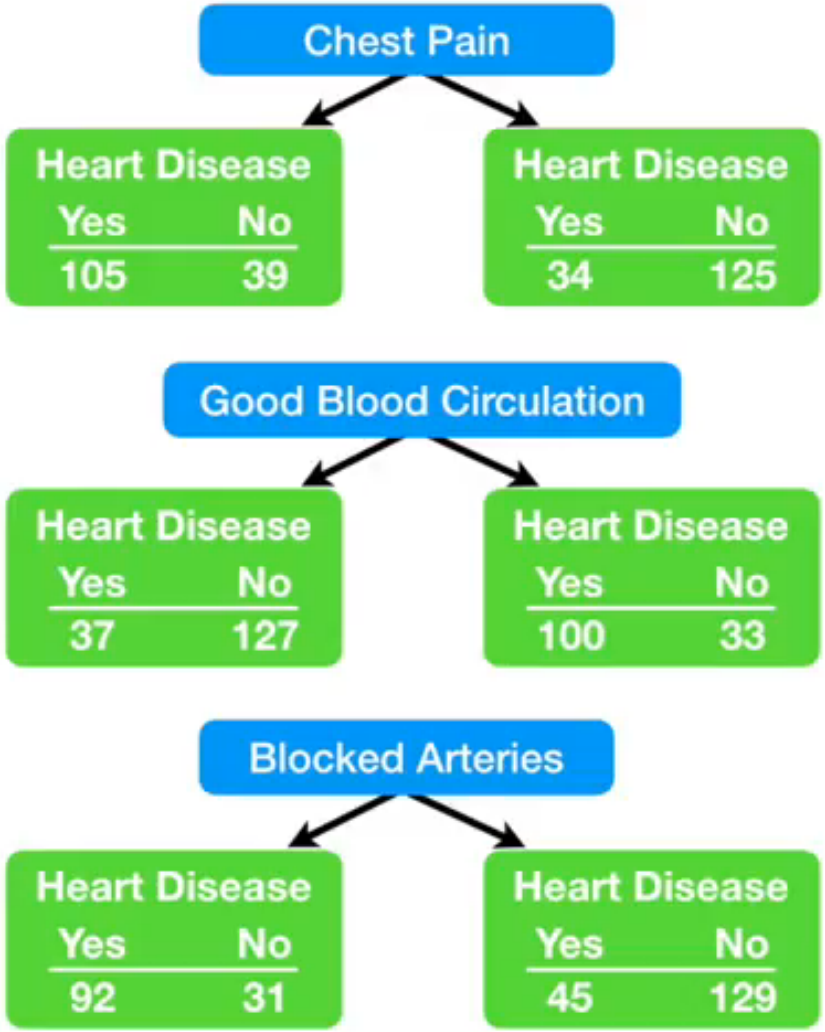
0,336

0,364

0,360

Drzewa decyzyjne - Przykład

$$Wspczynnik_{Giniego} = 1 - (prawdopodobiestwo" Tak")^2 - (Prawdopodobiestwo" Nie")^2$$



0,395

0,336

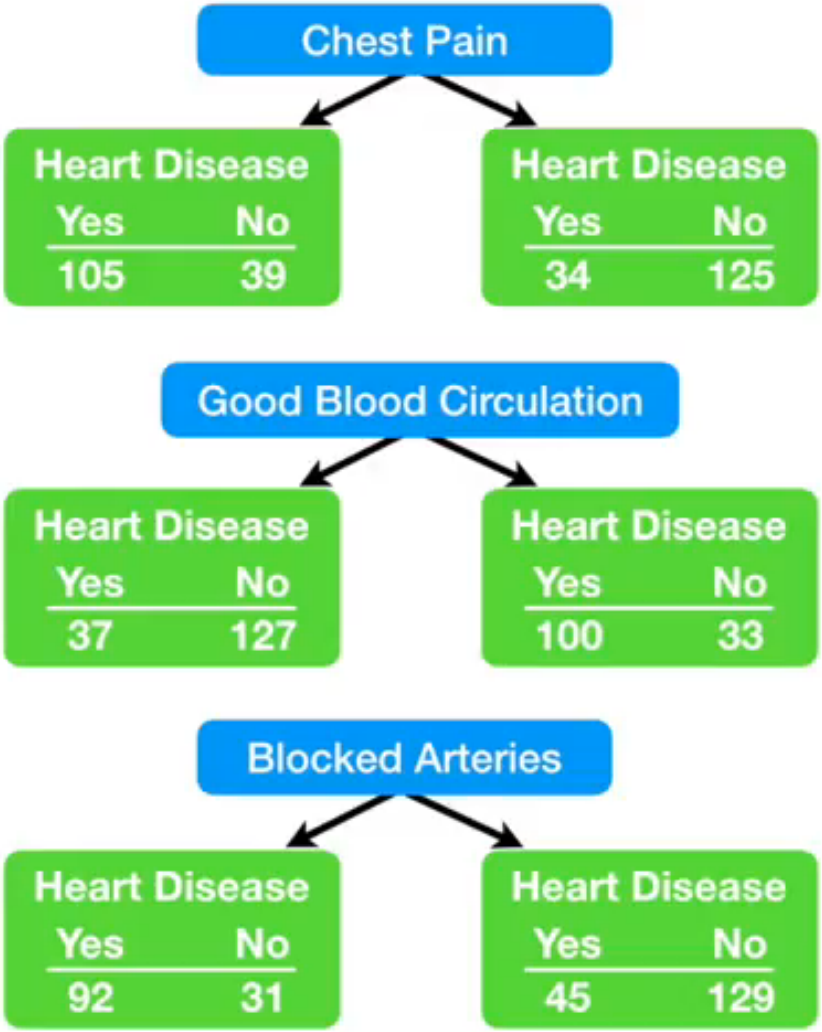
0,364

0,360

0,381

Drzewa decyzyjne - Przykład

$$Wspczynnik_{Giniego} = 1 - (prawdopodobiestwo"\ Tak"\ )^2 - (Prawdopodobiestwo"\ Nie"\ )^2$$



0,395

0,336

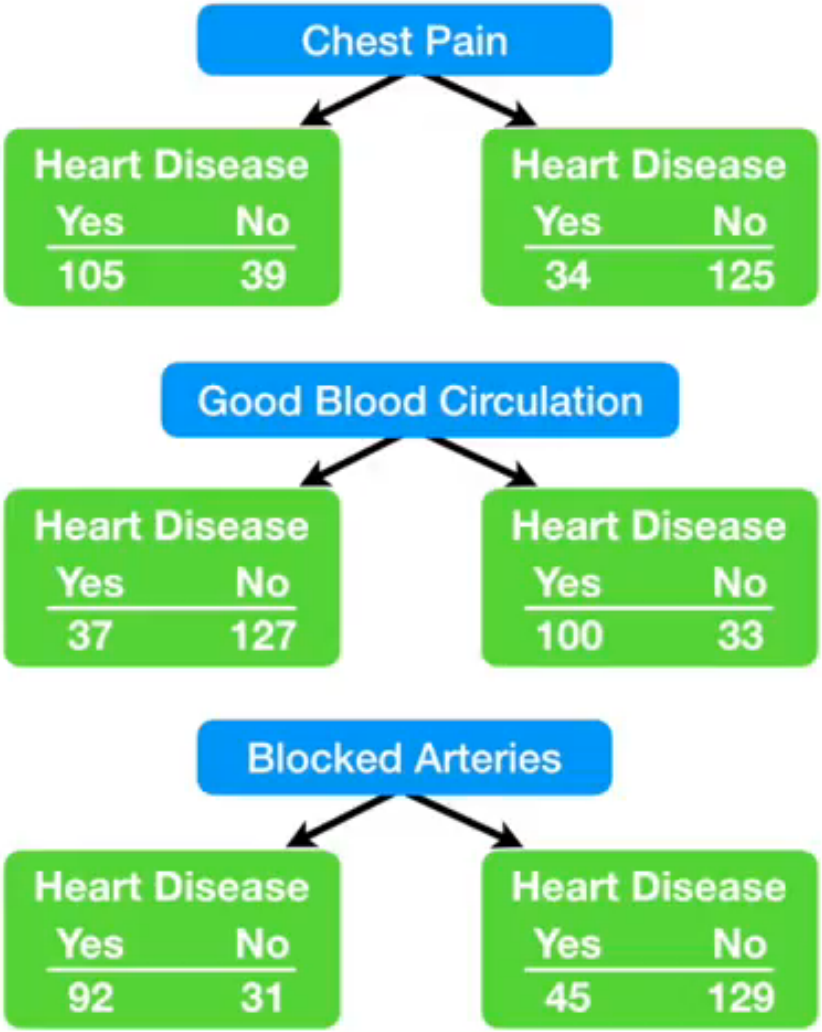
0,364

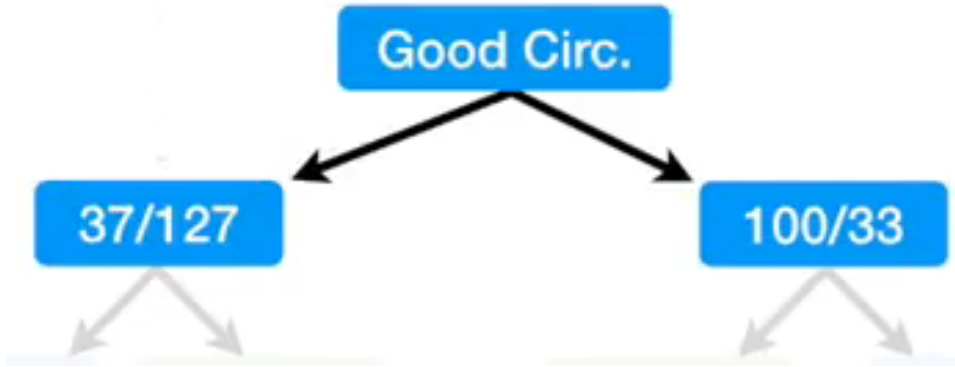
0,360

0,381

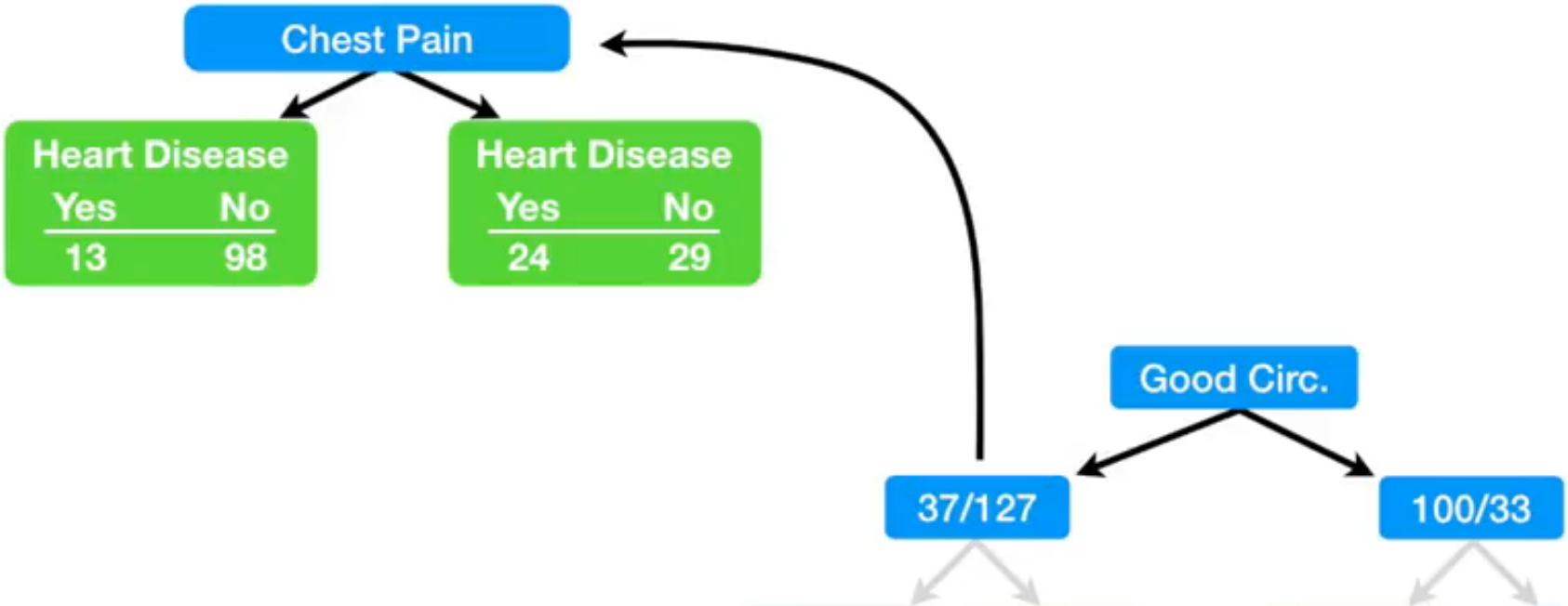
Drzewa decyzyjne - Przykład

$$Wspczynnik_{Giniego} =$$
$$1 - (prawdopodobiestwo" Tak")^2 - (Prawdopodobiestwo" Nie")^2$$



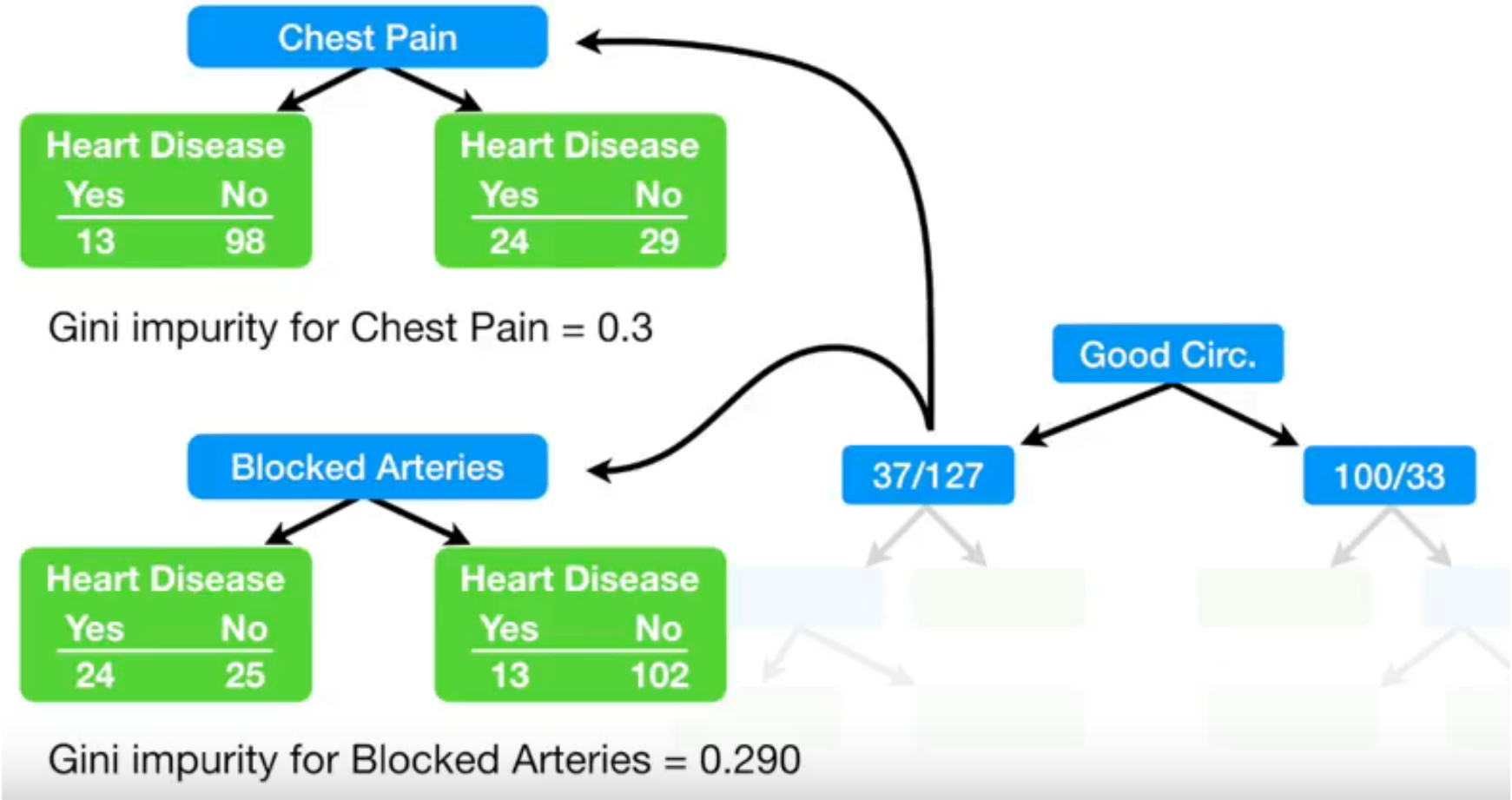


Drzewa decyzyjne - Przykład



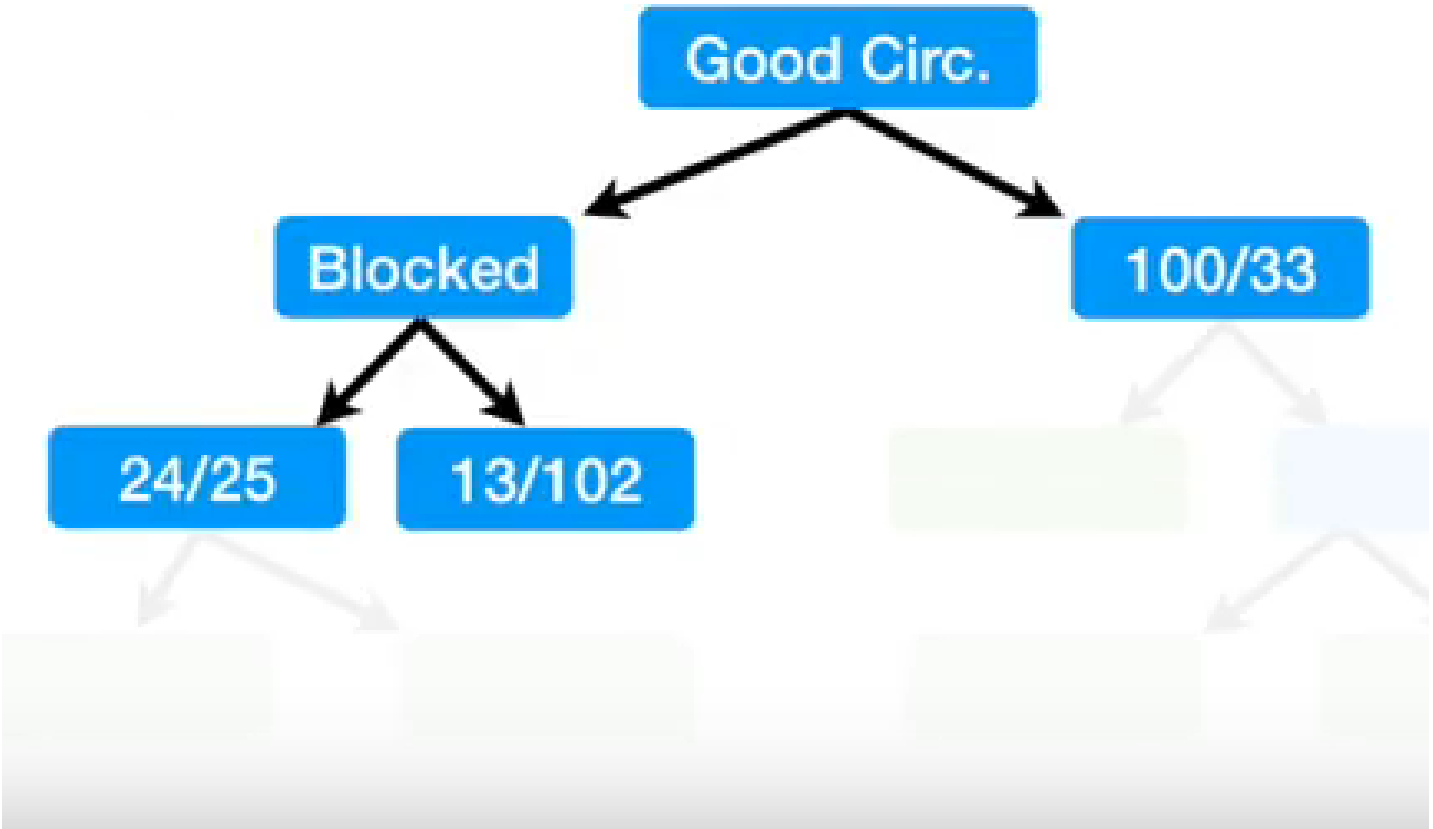
StatQuest: Decision Trees

Drzewa decyzyjne - Przykład



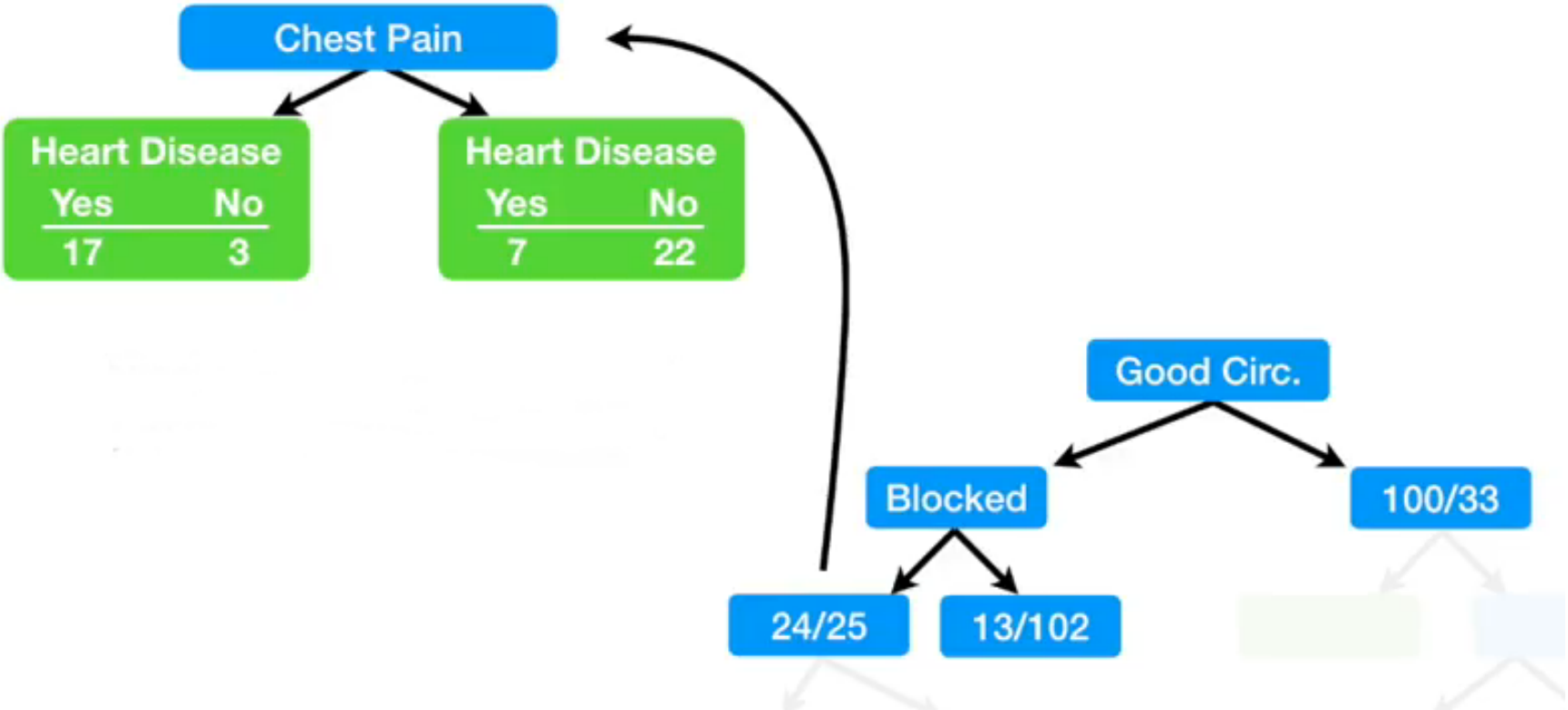
StatQuest: Decision Trees

Drzewa decyzyjne - Przykład



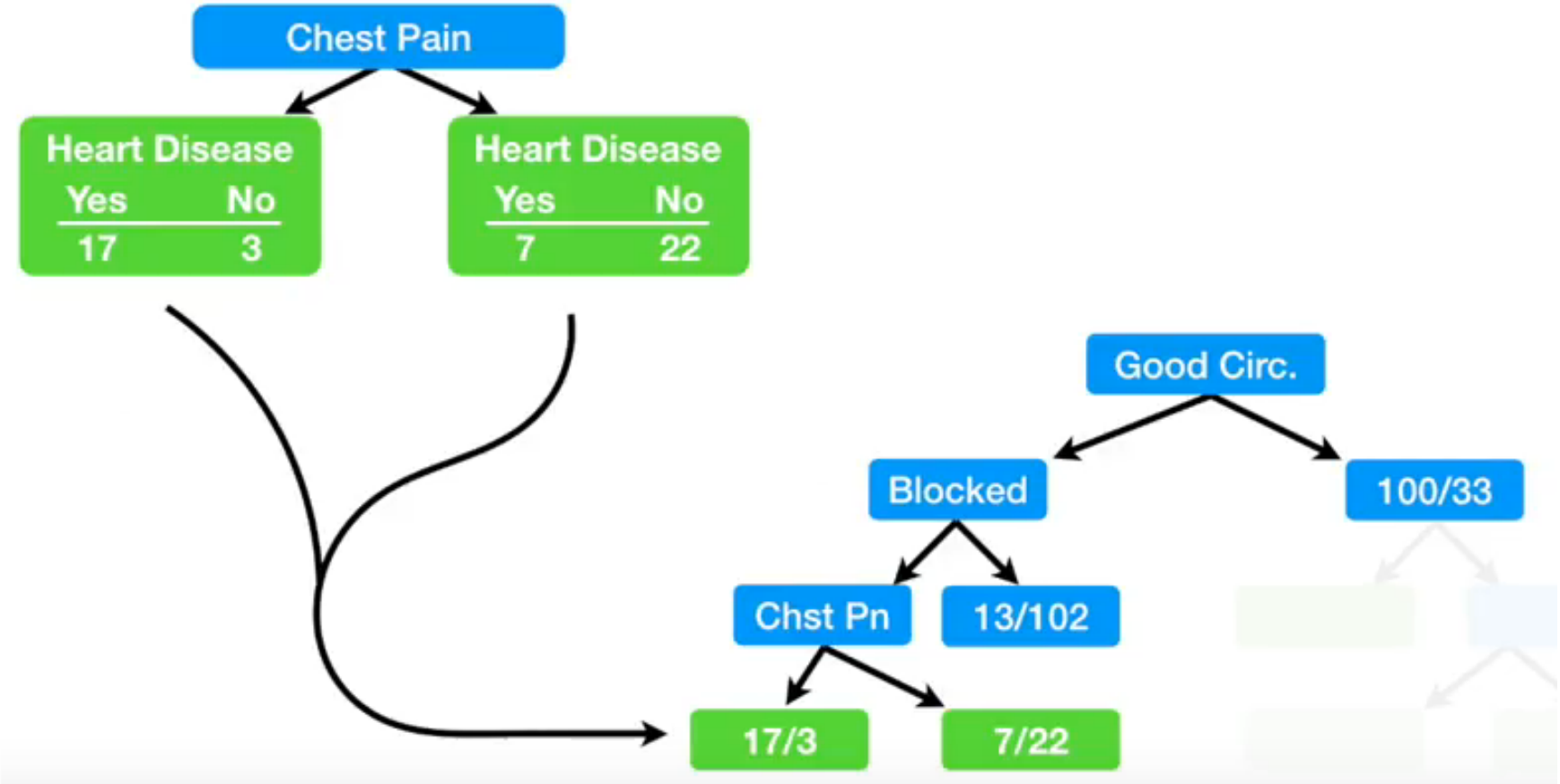
StatQuest: Decision Trees

Drzewa decyzyjne - Przykład



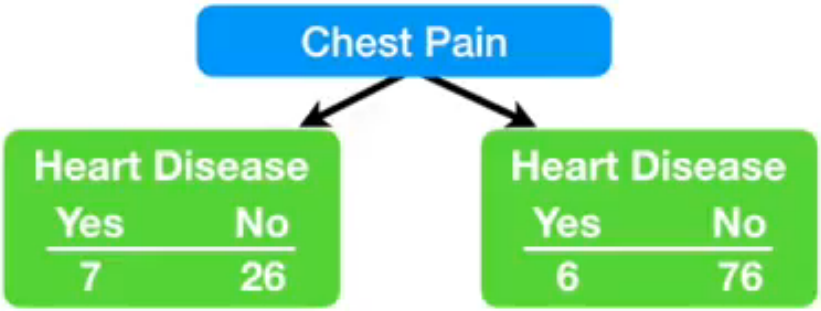
StatQuest: Decision Trees

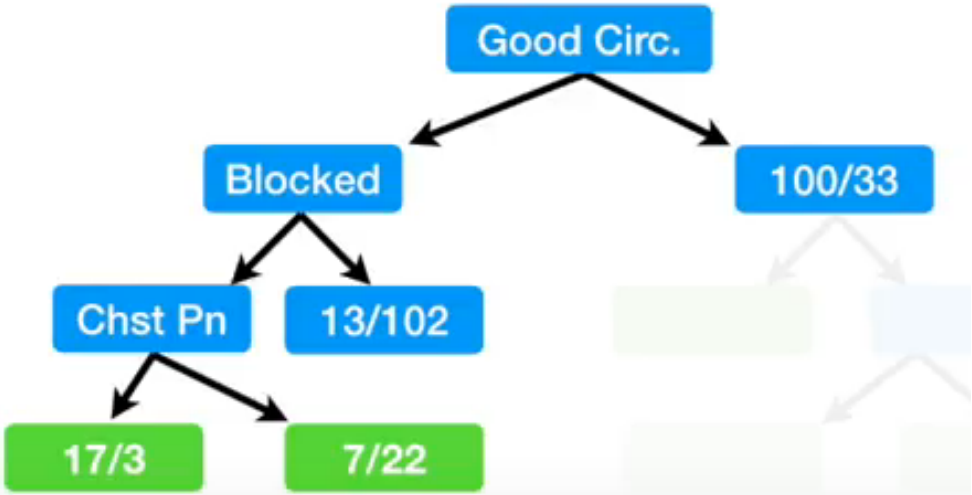
Drzewa decyzyjne - Przykład



StatQuest: Decision Trees

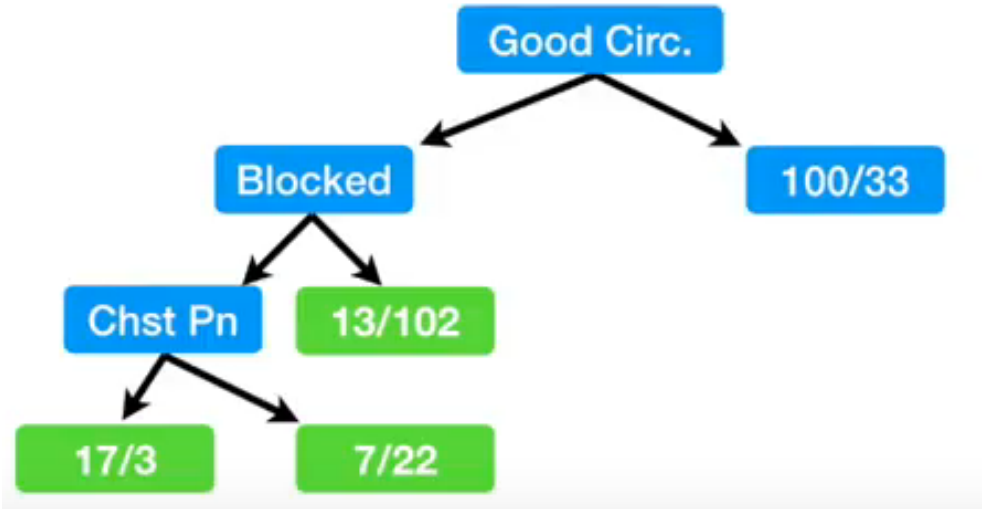
Drzewa decyzyjne - Przykład





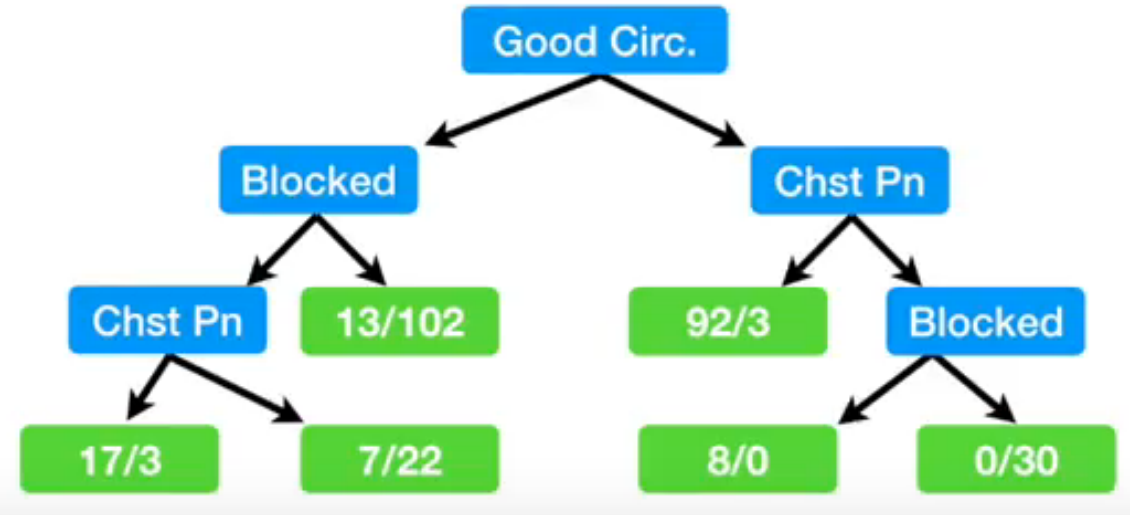
StatQuest: Decision Trees

Drzewa decyzyjne - Przykład



StatQuest: Decision Trees

Drzewa decyzyjne - Przykład



StatQuest: Decision Trees

Przykład - Dane numeryczne

Kroki algorytmu w przypadku danych numerycznych:

Weight	Heart Disease
220	Yes
180	Yes
225	Yes
190	No
155	No

Przykład - Dane numeryczne

Kroki algorytmu w przypadku danych numerycznych:

1. sortowanie

	Weight	Heart Disease
Lowest	155	No
	180	Yes
	190	No
	220	Yes
Highest	225	Yes

Przykład - Dane numeryczne

Kroki algorytmu w przypadku danych numerycznych:

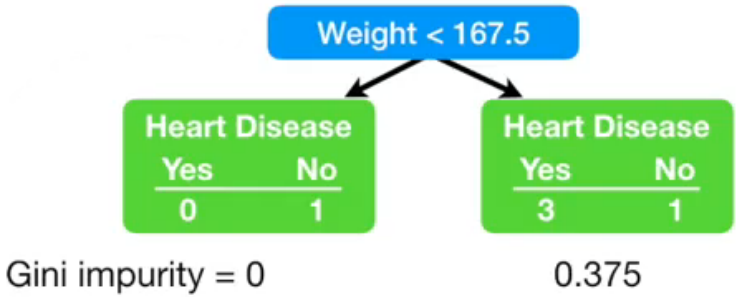
1. sortowanie
2. obliczenie średniej sąsiednich wartości

Weight	Heart Disease
155	No
167.5	
180	Yes
185	
190	No
205	
220	Yes
222.5	
225	Yes

Przykład - Dane numeryczne

Kroki algorytmu w przypadku danych numerycznych:

1. sortowanie
2. obliczenie średniej sąsiednich wartości
3. obliczenie jakości podziału dla każdej średniej wartości



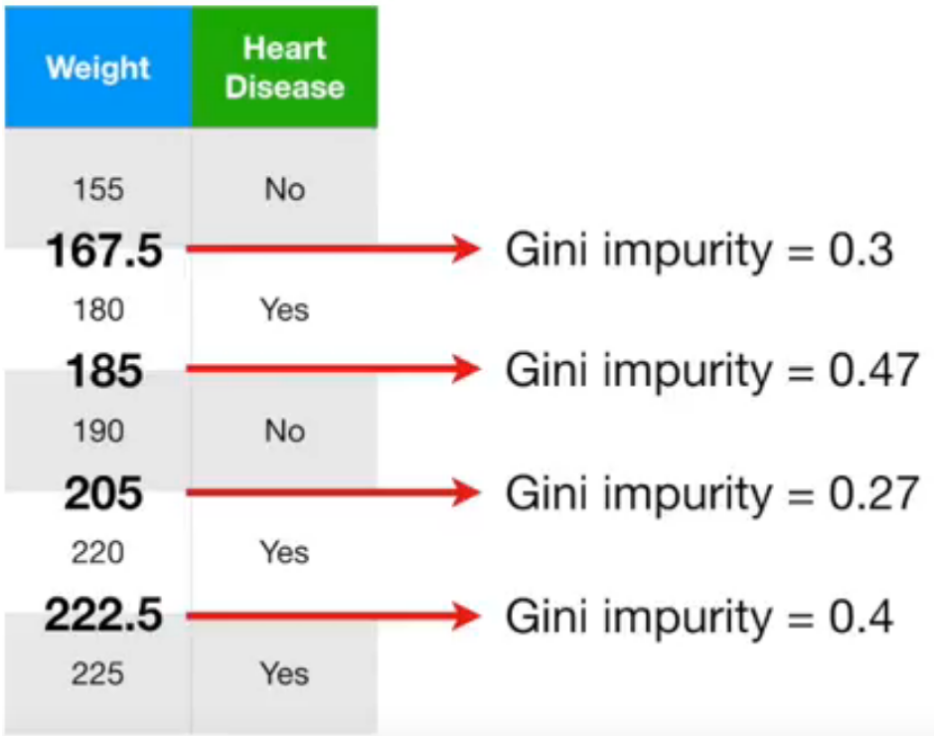
Weight	Heart Disease
155	No
167.5	
180	Yes
185	
190	No
205	
220	Yes
222.5	
225	Yes

StatQuest: Decision Trees

Przykład - Dane numeryczne

Kroki algorytmu w przypadku danych numerycznych:

1. sortowanie
2. obliczenie średniej sąsiednich wartości
3. obliczenie jakości podziału dla każdej średniej wartości



Zalety i wady

Zalety:

- Efektywny pamięciowo zapis wiedzy,
- Cechy ilościowe, binarne, rankingowe, wielokrotnego wyboru,
- Nie są czułe na błędy i braki danych,
- Szybki proces klasyfikacji,
- Zapis intuicyjny dla człowieka.

Wady:

- Brak możliwości uczenia adaptacyjnego,
- Możliwe duże rozmiary i poziom skomplikowania,
- Niewielka zmiana wymaga uruchomienia algorytmu budowy od początku,
- Zbyt duże dopasowanie do zbioru uczącego, duża niedokładność.

Spis treści

1. Wprowadzenie do klasyfikacji
2. kNN - k Nearest Neighbours
3. Regresja logistyczna
4. SVM
5. Naive Bayes
6. Drzewa decyzyjne
7. **Random Forest**

Random Forest

Idea:

- opary na zbiorze drzew decyzyjnych
- losowanie (ze zwracaniem) podzbioru z całego zbioru uczącego
- losowanie (ze zwracaniem) cech

Random Forest - Algorytm uczenia

1. Losowanie z n-elementowej próby n wektorów obserwacji,

Original Dataset

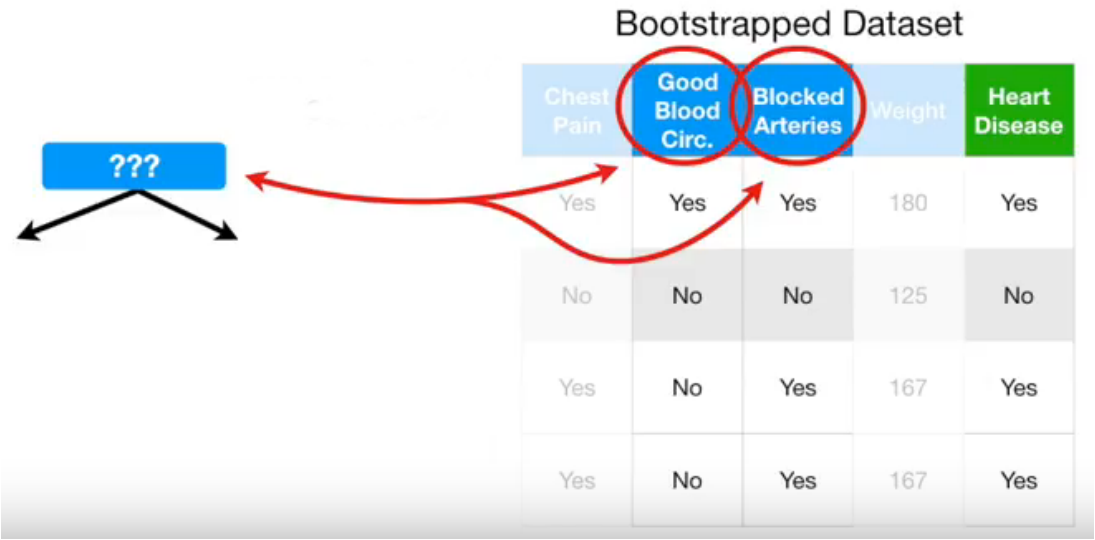
Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Random Forest - Algorytm uczenia

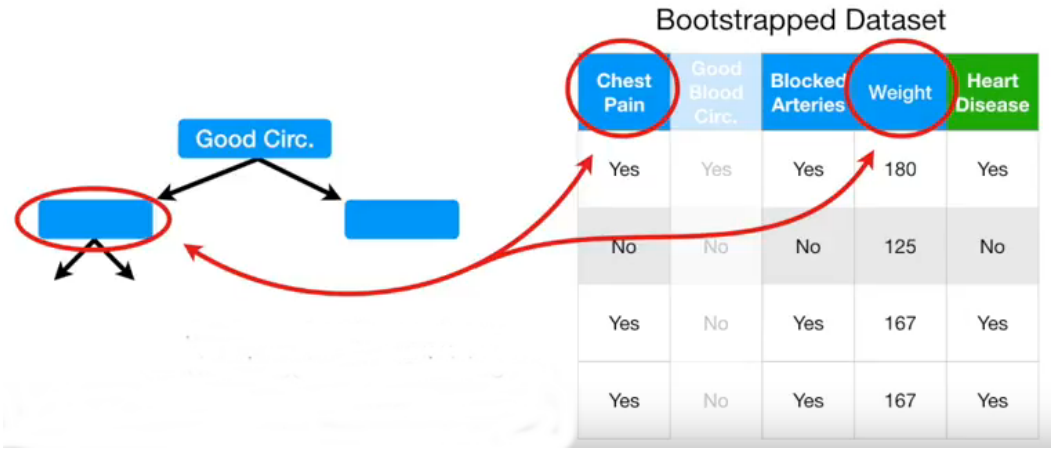
1. Losowanie z n-elementowej próby n wektorów obserwacji,
2. Tworzenie drzewa decyzyjnego, gdzie dla każdego węzła losowane jest m z p cech (potem k z m, gdzie $p > m > k$)



StatQuest: Random Forest

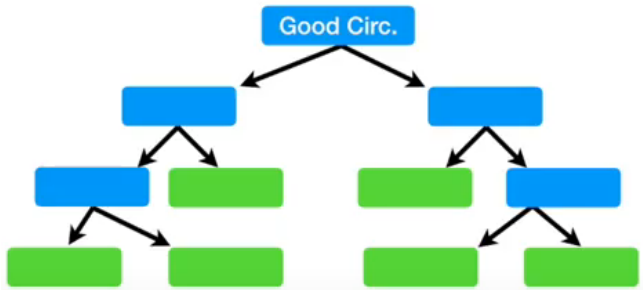
Random Forest - Algorytm uczenia

1. Losowanie z n-elementowej próby n wektorów obserwacji,
2. Tworzenie drzewa decyzyjnego, gdzie dla każdego węzła losowane jest m z p cech (potem k z m, gdzie $p > m > k$)



Random Forest - Algorytm uczenia

1. Losowanie z n-elementowej próby n wektorów obserwacji,
2. Tworzenie drzewa decyzyjnego, gdzie dla każdego węzła losowane jest m z p cech (potem k z m, gdzie $p > m > k$)



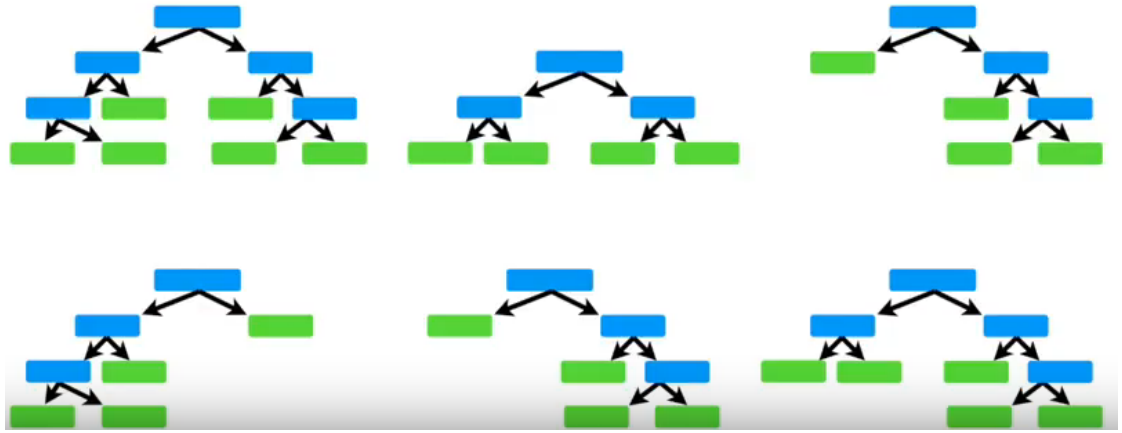
Bootstrapped Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

StatQuest: Random Forest

Random Forest - Algorytm uczenia

1. Losowanie z n-elementowej próby n wektorów obserwacji,
2. Tworzenie drzewa decyzyjnego, gdzie dla każdego węzła losowane jest m z p cech (potem k z m, gdzie $p > m > k$)
3. Powrót do kroku 1. i stworzenie kolejnych losowych drzew.



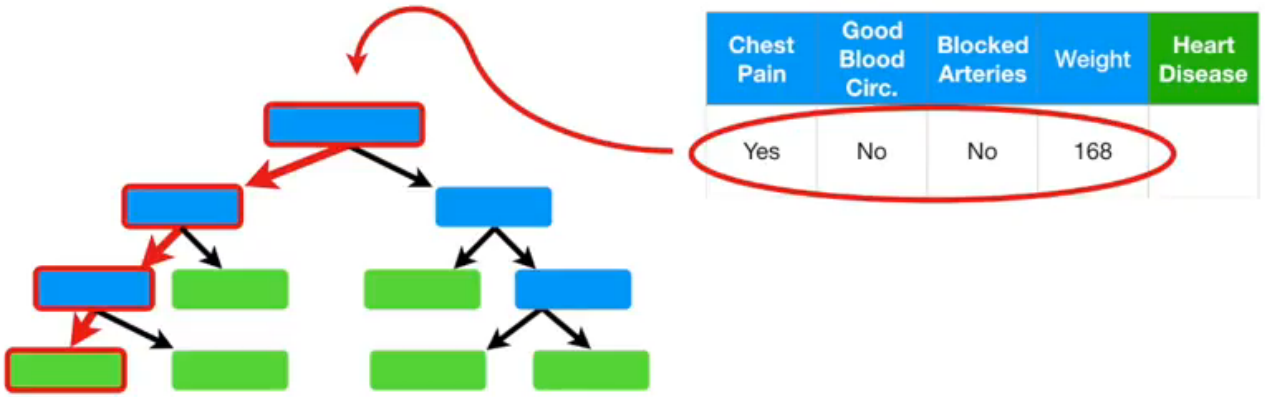
StatQuest: Random Forest

Random Forest - Klasyfikacja

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	

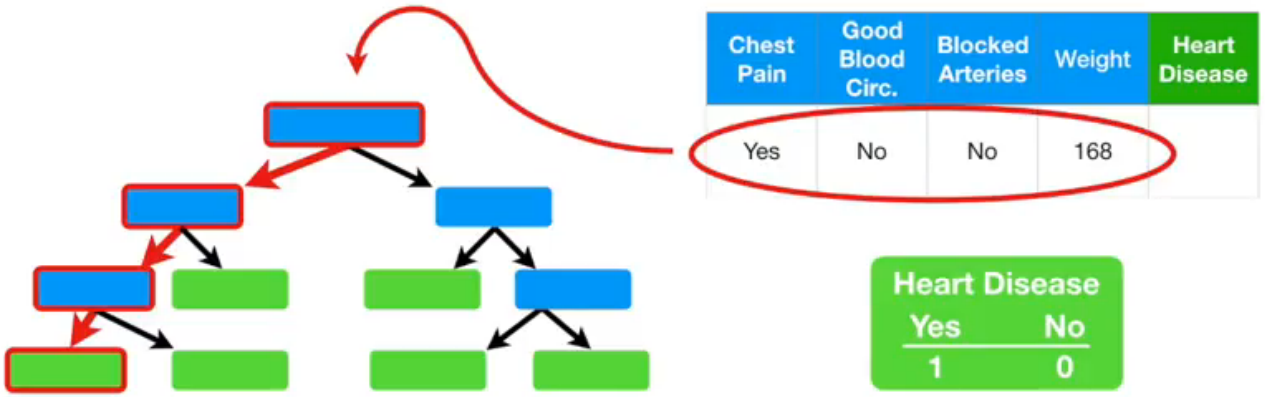
Random Forest - Klasyfikacja

1. Klasyfikacja obserwacji przez jedno drzewo,



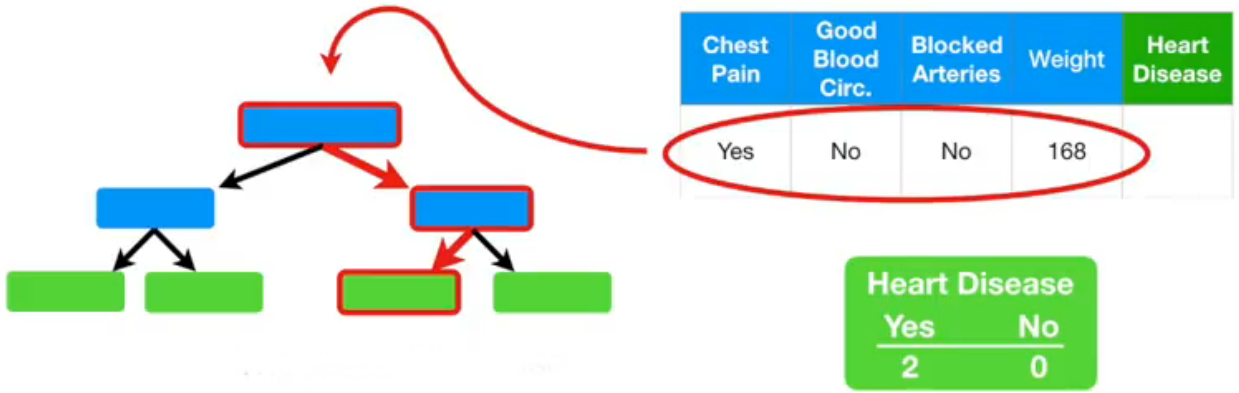
Random Forest - Klasyfikacja

1. Klasyfikacja obserwacji przez jedno drzewo,
2. Zapisanie wyniku,



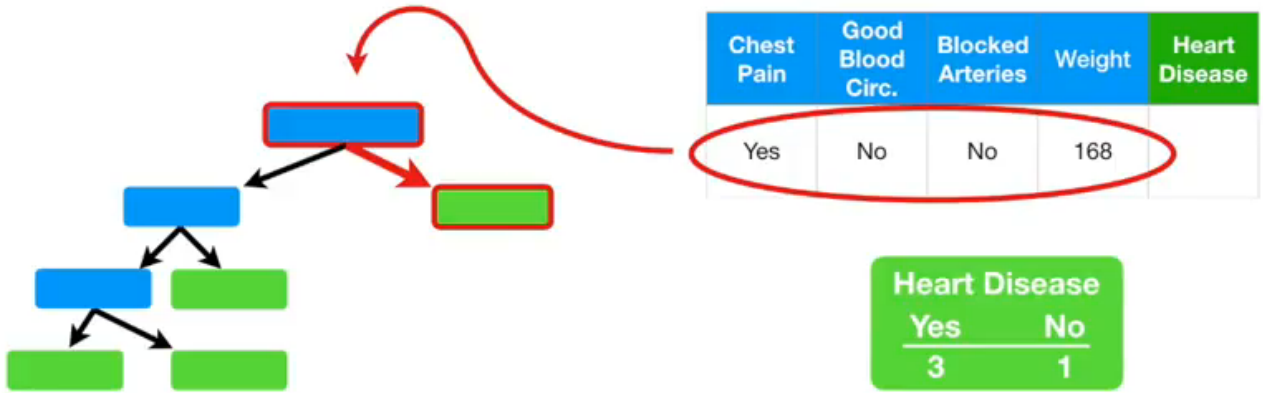
Random Forest - Klasyfikacja

1. Klasyfikacja obserwacji przez jedno drzewo,
2. Zapisanie wyniku,
3. Powtórzenie kroku 1. dla wszystkich drzew,



Random Forest - Klasyfikacja

1. Klasyfikacja obserwacji przez jedno drzewo,
2. Zapisanie wyniku,
3. Powtórzenie kroku 1. dla wszystkich drzew,



Random Forest - Klasyfikacja

1. Klasyfikacja obserwacji przez jedno drzewo,
2. Zapisanie wyniku,
3. Powtórzenie kroku 1. dla wszystkich drzew,
4. Wybór klasy, która wystąpiła najczęściej.

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	**YES**

Heart Disease	
Yes	No
5	1

Random Forest

Cechy:

- jest dokładny
- jest skuteczny na dużych bazach danych
- szacuje dane w przypadku ich braku
- nie wymaga wiedzy eksperckiej
- nie jest podatny na overfitting
- określa przydatność cech
- umożliwia klasteryzację
- umożliwia oszacowanie błędu klasyfikacji