Machine learning

Logistic Regression

Wouter Gevaert & Marie Dewitte

Inhoud

Introductie

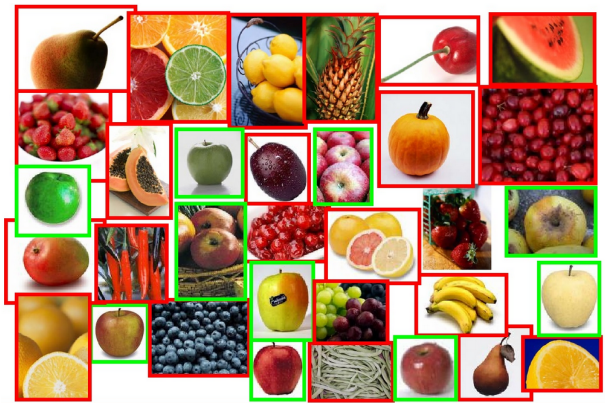
Logistic regression

Evaluatie van een classifier

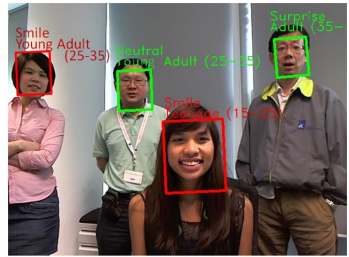
Introductie

Wat is classificatie?

Classificatie is een supervised learning techniek waarbij een getraind model niet geziene inputs toewijst aan één of meerdere gelabelde categorieën (classes)



- Gezichtsherkenning
- Nummerplaatherkenning
- Spam detectie
- Medische diagnoses
- Voorspelling of een klant op een advertentie zal klikken
- Kwaliteitscontrole
- ...

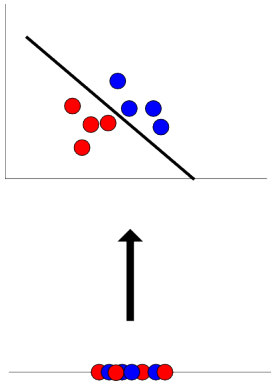


Types van classifiers

Binairy (binomial) classifier

Verdeel de samples in **twee verschillende klasses**

Voorbeeld: bepaal of een kanker goedaardig of kwaadaardig is

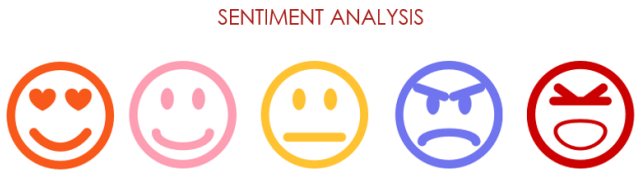


Types van classifiers

Multiclass classifier

Verdeel de samples in **drie of meerdere verschillende klasses**

Voorbeelden: gezichtsherkenning, sentiment analyse



Types van classifiers

Multilabel classifier

Er kunnen meerdere labels aan een sample toegewezen worden. Een sample kan tot meerdere klasses behoren

Voorbeelden: image content analysis, een film kan tot meerdere genres behoren.

Huis	Boom	Strand	Wolken	Bergen	Dieren
Ja	Ja	Nee	Ja	Nee	Nee

Voorbeeld

	rondheid	groenheid	appel
0	9	8	1
1	10	7	1
2	2	3	0
3	1	2	0
4	5	8	1
5	7	7	1
6	6	3	0
7	3	3	0
8	9	5	1
9	9	3	1
10	4	5	0
11	6	1	0
12	5	7	1
13	8	8	1
14	2	7	0

Voorspel of een stuk fruit een appel is op basis van vorm en kleur

features: rondheid en groenheid

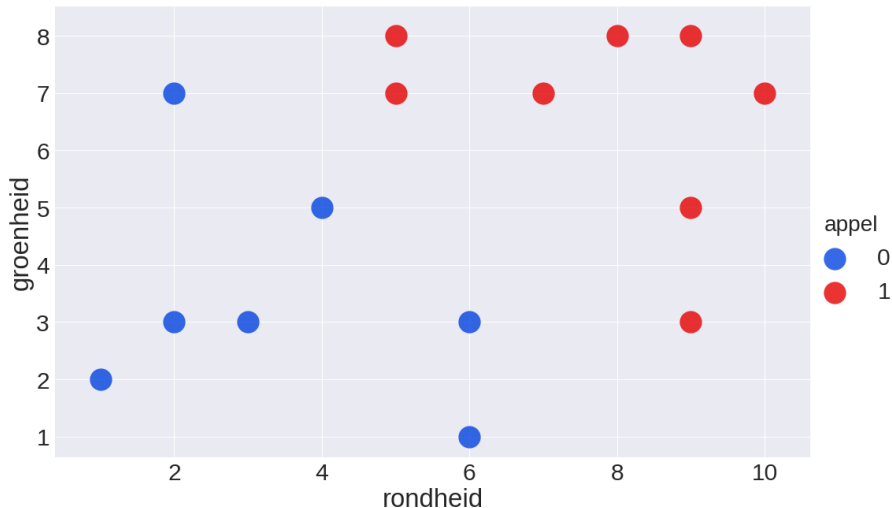
target: appel: ja/nee

trainingset met 15 training examples

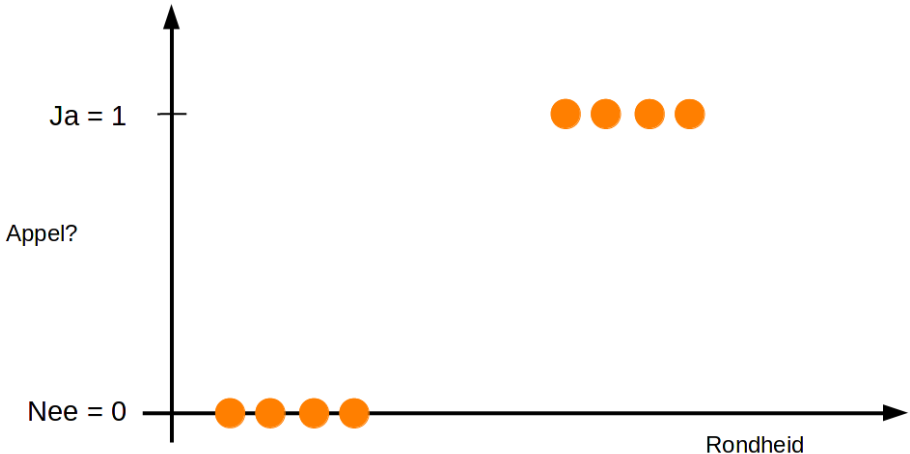
Classificatie

Bij classificatie is de output/target een (discrete) variabele / klasse

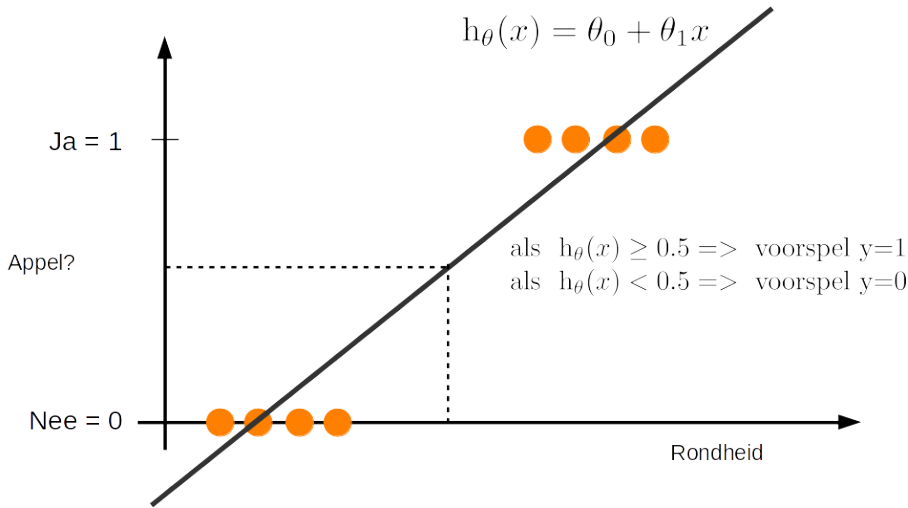
Voorbeeld



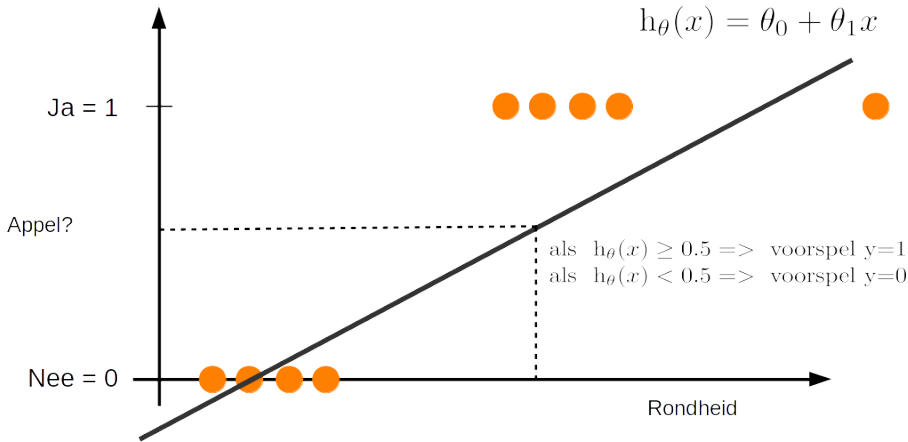
Waarom regressie geen goede optie is



Waarom regressie geen goede optie is



Waarom regressie geen goede optie is



Logistic regression

Logistic regression

Het model

We willen dat het model $h_\theta(x)$ voldoet aan

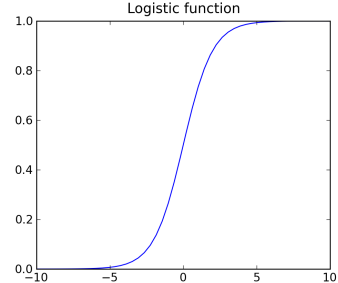
$$0 \leq h_\theta(x) \leq 1$$

- $h_\theta(x)$ = de geschatte kans dat y=1 bij input x
- Voorbeeld: $h_\theta(x) = 0.80$
  Het model is voor 80% zeker dat het om een appel gaat

Logistic regression

Het model

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



$y = 1$ als $h_\theta(x) \geq 0.5 \Rightarrow \theta^T x \geq 0$

$y = 0$ als $h_\theta(x) < 0.5 \Rightarrow \theta^T x < 0$

Logistic regression

Het model - interpretatie via voorbeeld appels

Het model is van de vorm: $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

met x_1 = rondheid en x_2 = groenheid

Veronderstel na training: $\theta_0 = -40,\ \theta_1 = 4,\ \ \theta_2 = 4$

Voorspel $y = 1$ als $-40 + 4x_1 + 4x_2 \geq 0$

Voorspel $y = 0$ als $-40 + 4x_1 + 4x_2 < 0$

Gegeven een rondheid van 8 en een groenheid van 6:

$-40 + 4 \times 8 + 4 \times 6 = 16 \Rightarrow$ Appel

$$h_\theta(x) = \frac{1}{1 + e^{-16}} = 0.999999887$$

Logistic regression

Het model - interpretatie via voorbeeld appels

Gegeven een rondheid van 5 en een groenheid van 4.5:

$-40 + 4 \times 5 + 4 \times 4.5 = -2 \Rightarrow$ Geen appel

$$h_\theta(x) = \frac{1}{1 + e^{+2}} = 0.12$$

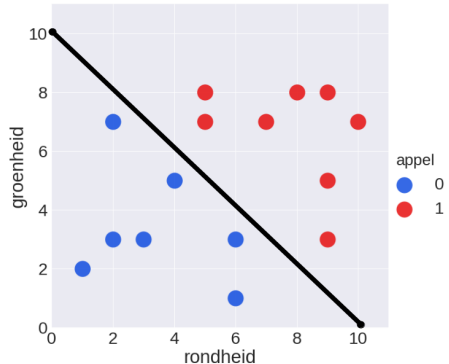
- Het model is maar voor 12% zeker dat het om een appel gaat
- Met 88% zekerheid gaat het volgens het model niet om een appel

Logistic regression

Het model - grafische interpretatie via voorbeeld appels

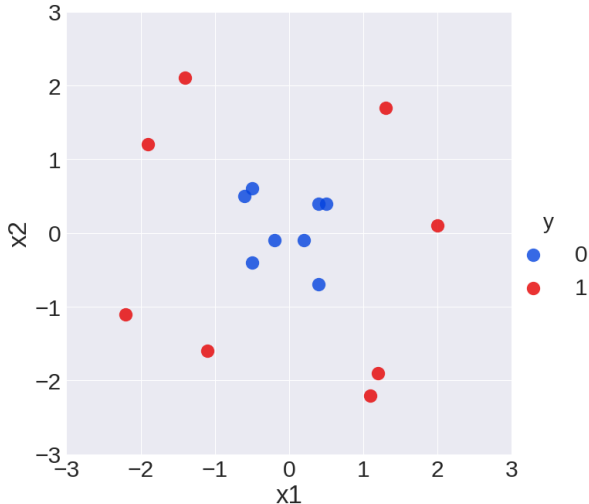
Op de scheidingslijn is: $\theta_0 + \theta_1 x_1 + \theta_2 x_2 = 0$

In het voorbeeld: $-40 + 4x_1 + 4x_2 = 0$



Logistic regression

Het model - grafische interpretatie - niet lineair



Logistic regression

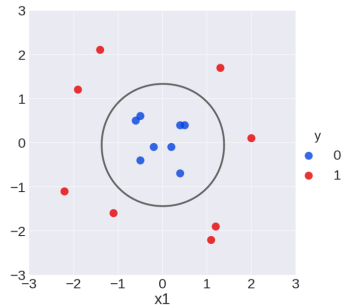
Het model - grafische interpretatie - niet lineair

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

Veronderstel: $\theta_0 = -2,\ \theta_1 = 0,\ \theta_2 = 0,\ \theta_3 = 1,\ \theta_4 = 1$

Voorspel $y = 1$ als $-2 + x_1^2 + x_2^2 \geq 0$

$x_1^2 + x_2^2 \geq 2$ (vergelijking van een circel met straal $\sqrt{2}$)



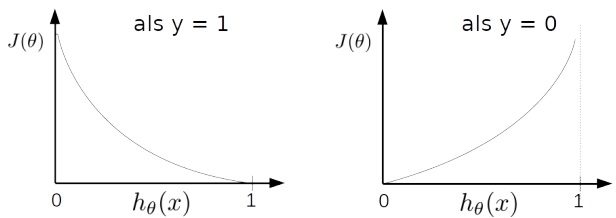
Logistic regression

Het model

De kostenfunctie wordt:

$$J(\theta) = \begin{cases} -\ln(h_\theta(x)) & \text{als } y = 1 \\ -\ln(1 - h_\theta(x)) & \text{als } y = 0 \end{cases}$$

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \ln(h_\theta(x^{(i)})) + (1 - y^{(i)}) \ln(1 - h_\theta(x^{(i)}))\right]$$



Zoek de waarden voor θ die de kostenfucntie $J(\theta)$ minimaliseert via Gradiënt Descent (GDS).

Logistic regression met Sklearn

Preprocessing van de data

Analoog aan preprocessing bij lineaire regressie:

- Data inlezen
- Check op inconsistenties
- Check uitschieters
- Plot de data
- Splits op in features en targets
- Verdeel in een training en test set

Logistic regression met Sklearn

Preprocessing van de data

```
# Importeer de dataset
dataset = pd.read_csv('appels.csv')
features = list(dataset.columns[:dataset.columns.size-1])
X = dataset[features].values
y= dataset['appel'].values
sns.set(font_scale = 2) # lettergrootte van de axis labels
colors = ["blue", "red", "greyish", "faded green",
 "dusty purple"]
sns.lmplot(x='rondheid',y='groenheid',data=dataset,fit_reg=False,
 hue='appel',palette =sns.xkcd_palette(colors),
scatter_kws={'s':500}, size=7, aspect=1.5)
```

Logistic regression met Sklearn

Trainen van het logistic regression model

```
# Train een logistic regression classifier

logreg = linear_model.LogisticRegression(C=1e5)
# C= Inverse of regularization strength;
# must be a positive float. Like in support vector machines,
# smaller values specify stronger regularization.
logreg.fit(X, y)

print('coefficienten: ',logreg.coef_)
print('intercept: ',logreg.intercept_)
```

coëfficiënten: $\theta_1 = 4.28747762$ en $\theta_2 = 4.06244327$

intercept: $\theta_0 = -43.94116677$

Logistic regression met Sklearn

Classificeren van een nieuwe sample

```
#voorspel de klasse van een rondheid=8 en een groenheid van 6

print(logreg.predict(np.array([8,6]).reshape(1,-1)))

print('kans op een appel/geen appel',
logreg.predict_proba(np.array([8,6]).reshape(1,-1)))

#voorspel de klasse van een rondheid=4 en een groenheid van 4

print(logreg.predict(np.array([4,4]).reshape(1,-1)))
print('kans op een appel/geen appel',
logreg.predict_proba(np.array([4,4]).reshape(1,-1)))
```

[1] kans op een appel/geen appel 3.99395302e-07 | 9.99999601e-01

[0] kans op een appel/geen appel 9.99973583e-01 | 2.64168196e-05

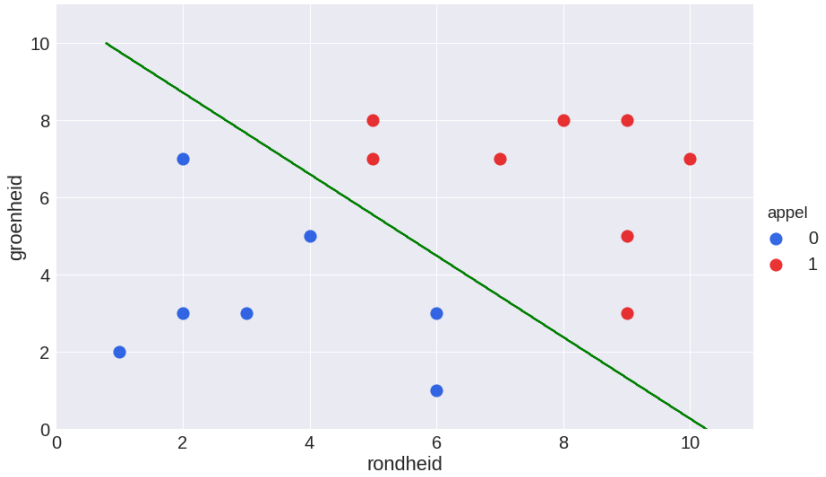
Logistic regression met Sklearn

Visualiseer de decision boundary

```
h = 0.01
rond_min = X[:,0].min()−2
rond_max = X[:,0].max()+2
groen_min = X[:,1].min()−2
groen_max = X[:,1].max()+2
xx, yy = np.meshgrid(np.arange(rond_min, rond_max, h),
np.arange(groen_min, groen_max, h))
Z = logreg.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)
sns.set(font_scale = 2)
colors = ["blue", "red"]
sns.lmplot(x='rondheid',y='groenheid',data=dataset,
fit_reg=False,hue='appel',palette =sns.xkcd_palette(colors),
scatter_kws={'s':200}, size=8, aspect=1.5)
sns.plt.ylim(0, 11)
sns.plt.xlim(0, 11)
plt.contour(xx, yy, Z, colors='green')
```

Logistic regression met Sklearn

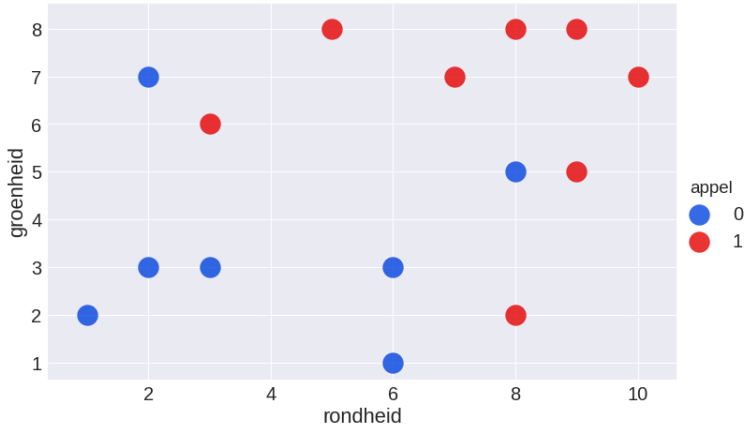
Visualiseer de decision boundary



Logistic regression met Sklearn

Feature engineering

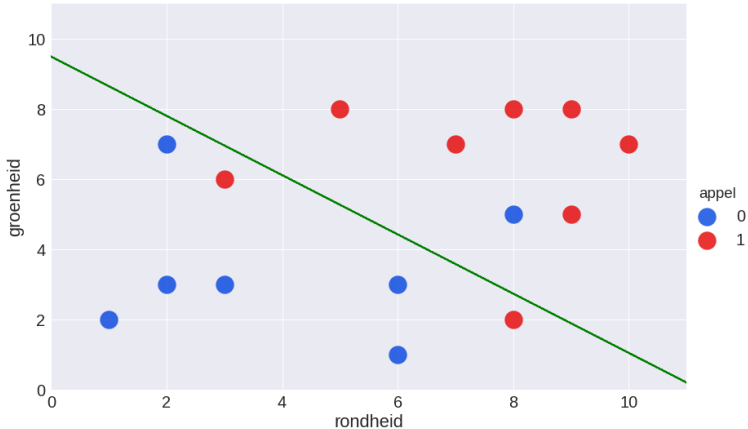
Niet lineair scheidbare dataset



Logistic regression met Sklearn

Feature engineering

Niet lineair scheidbare dataset



Logistic regression met Sklearn

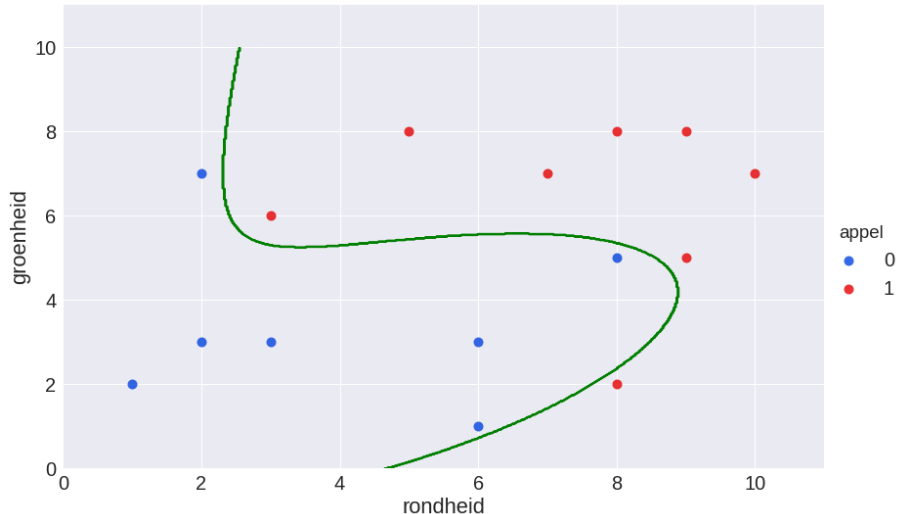
Feature engineering

Toevoegen van hogere orde features:

```
# Aanmaken van de hogere orde features
graad = 3

poly = PolynomialFeatures(graad)
Xp = poly.fit_transform(X)


# Train model op hogere orde features en visualiseer de decision boundary

logreg_poly = linear_model.LogisticRegression(C=1)
logreg_poly.fit(Xp, y)
```

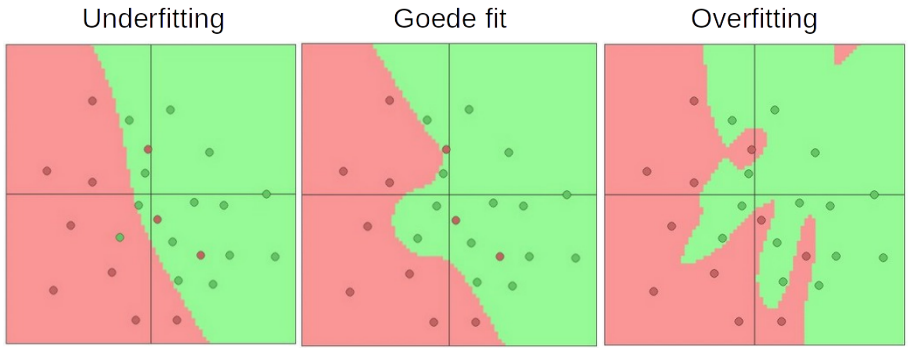
Logistic regression met Sklearn

Feature engineering - voorbeeld appels met derde-orde features



Logistic regression met SKlearn

Underfitting en overfitting



Regelen tussen underfitting en overfitting via regularisatie

Logistic regression met SKlearn

Underfitting en overfitting

Via regularisatie een goed evenwicht zoeken tussen underfitting en overfitting

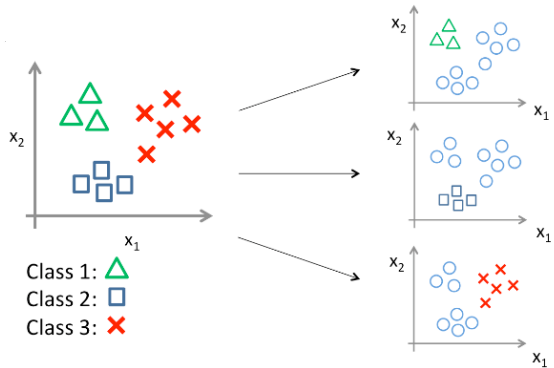
In Scikit Learn linear_model.LogisticRegression

- C = inverse regularisatie sterkte
    - kleine waarden voor C zorgen voor een sterke regularisatie (underfitting)
    - grote waarden voor C zorgen voor een zwakke regularisatie (overfitting)

```
logreg = linear_model.LogisticRegression(C=100)
logreg.fit(Xf, y)
```

Multi-class classification

One-vs-All

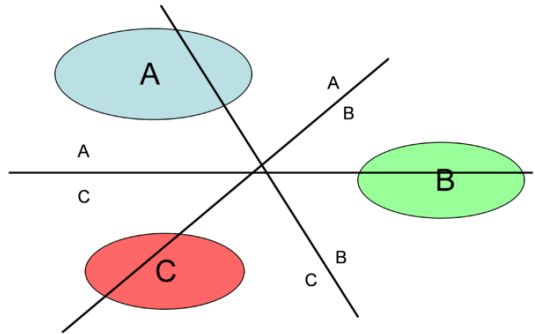


Je hebt 1 classifier per klasse. Totaal aantal classifiers: N

Gevoeliger voor niet-gebalanceerde data

Multi-class classification

One-vs-One



Je hebt $\frac{N(N-1)}{2}$ classifiers (met N aantal klasses) $\Rightarrow$ Rekenintensief

Minder gevoelig voor niet-gebalanceerde data

Evaluatie van een classifier

Evaluatie van een classifier

Accuracy

	p' (Predicted)	n' (Predicted)
p (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

Evaluatie van een classifier

Recall = Sensitivity = Hit rate = True Positive Rate (TPR)

	p' (Predicted)	n' (Predicted)
P (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

$$\text{Recall} = \frac{TP}{TP + FN}$$

Evaluatie van een classifier

Precision = Positive Predictive Value (PPV)

	p' (Predicted)	n' (Predicted)
P (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

$$\text{Precision} = \frac{TP}{TP + FP}$$

Evaluatie van een classifier

F1 score

	p' (Predicted)	n' (Predicted)
p (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

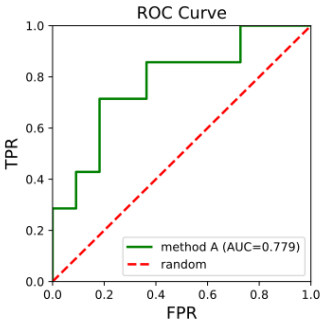
$$\text{F1 score} = \frac{2*(\text{Recall} * \text{Precision})}{(\text{Recall} + \text{Precision})}$$

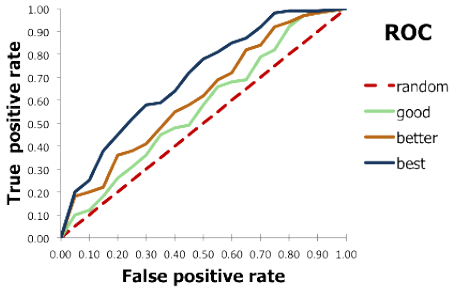
Is een harmonisch gewogen gemiddelde van de recall en de precision

Evaluatie van een classifier

Receiver Operating Characteristic (ROC)

- Gebruikt bij binaire classifiers
- Gebruikt om betere modellen te selecteren en minder goede te verwerpen





Evaluatie van een classifier

Receiver Operating Characteristic (ROC)

	p' (Predicted)	n' (Predicted)
p (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

True Positive Rate (TPR) = Sensitivity = Recall = $\frac{TP}{TP + FN}$

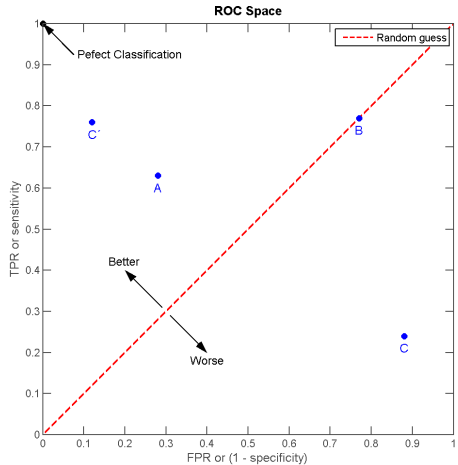
False Positive Rate (FPR) = 1 - specificity = $\frac{FP}{FP + TN}$

Specificity = $\frac{TN}{FP + TN}$

Precision = $\frac{TP}{TP + FP}$

Evaluatie van een classifier

Receiver Operating Characteristic (ROC)



Evaluatie van een classifier

ROC curve en AUC (Area Under ROC Curve)

ROC bij verschillende threshold settings voor $p(y = 1|x)$ - **threshold = 0.5**

Example nr.	y_true	P(y=1|x)
1	1	0.93
2	0	0.55
3	0	0.30
4	0	0.53
5	1	0.81
6	1	0.69
7	0	0.42
8	1	0.28
9	1	0.96
10	0	0.51

	Predicted (y=1)	Predicted (y=0)
Actual (y=1)	4	1
Actual (y=0)	3	2

$$\text{True Positive Rate (TPR)} = \frac{TP}{TP + FN} = \frac{4}{4 + 1} = 0.8$$

$$\text{False Positive Rate (FPR)} = \frac{FP}{FP + TN} = \frac{3}{3 + 2} = 0.6$$

Evaluatie van een classifier

ROC curve en AUC (Area Under ROC Curve)

ROC bij verschillende threshold settings voor $p(y = 1|x)$ - **threshold = 0.6**

Example nr.	y_true	P(y=1|x)
1	1	0.93
2	0	0.55
3	0	0.30
4	0	0.53
5	1	0.81
6	1	0.69
7	0	0.42
8	1	0.28
9	1	0.96
10	0	0.51

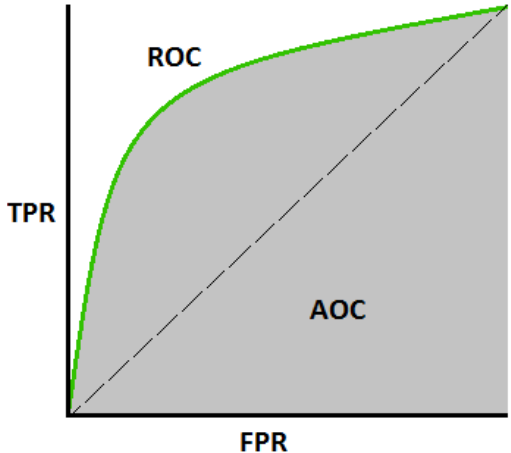
	Predicted (y=1)	Predicted (y=0)
Actual (y=1)	4	1
Actual (y=0)	0	5

$$\text{True Positive Rate (TPR)} = \frac{TP}{TP + FN} = \frac{4}{4 + 1} = 0.8$$

$$\text{False Positive Rate (FPR)} = \frac{FP}{FP + TN} = \frac{0}{0 + 5} = 0$$

Evaluatie van een classifier

AuROC (Area under ROC)



Evaluatie van een classifier

Vuistregel