Regresie logistică (Logistic Regression)

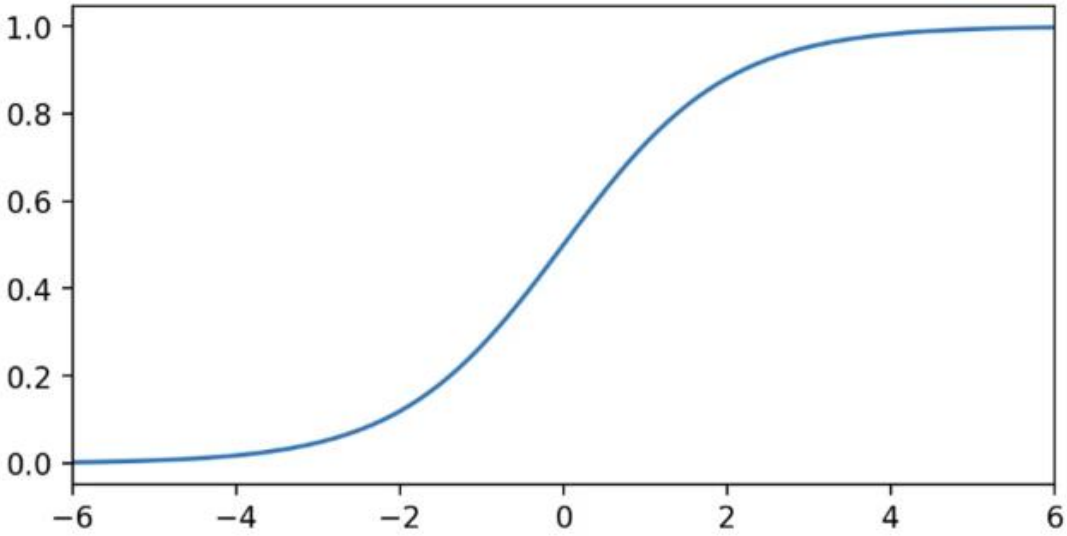
Ce ne așteaptă?

1. Funcția sigmoid
2. Aparatul matematic
3. Instrumente Scikit-Learn pentru regresia logistică

1. Funcția sigmoid

- **Funcția sigmoid**

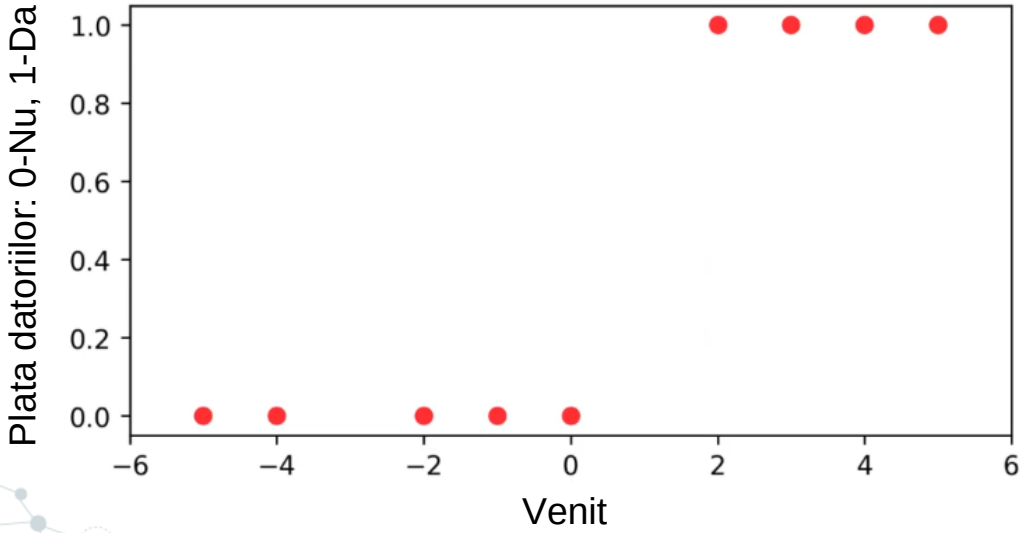
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



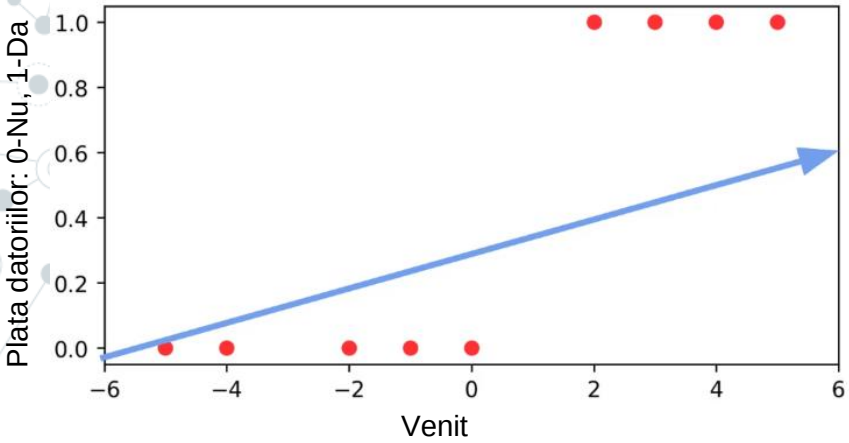
Datele istorice

Venit	-5	-4	-2	-1	0	2	3	4	5
Plata datoriilor	0	0	0	0	0	1	1	1	1

Graficul datelor istorice

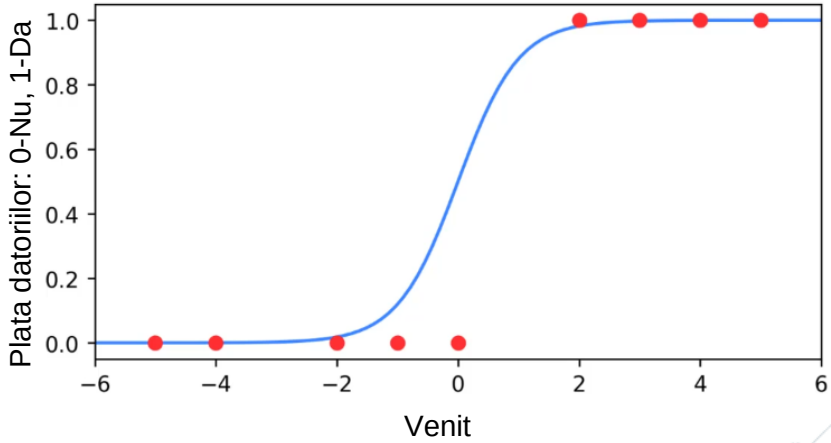


Regresia liniară simplă



$$Plata\ datorii = \beta_0 + \beta_1 * Venit$$

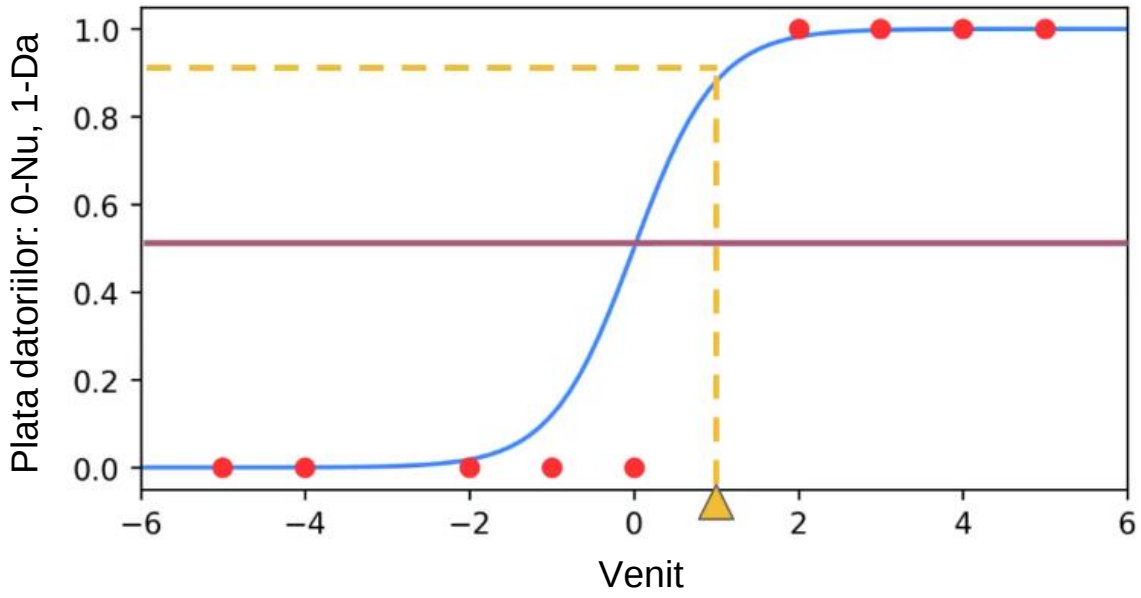
Regresie logistică simplă



$$p(Plata\ datorii) = \frac{1}{1 + e^{-Venit}}$$

$$p() - probabilitatea$$

- **Care este probabilitatea că va plăti datoriile dacă venitul este 1?**

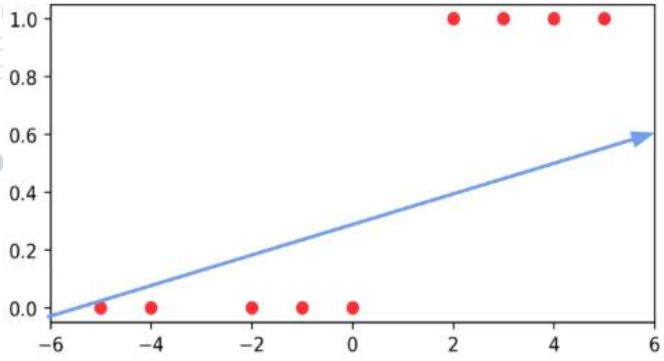


0.5 – nivel de decizie

$$p(Plata\ datorii)_{Venit=1} = 0{,}9 \rightarrow 1$$

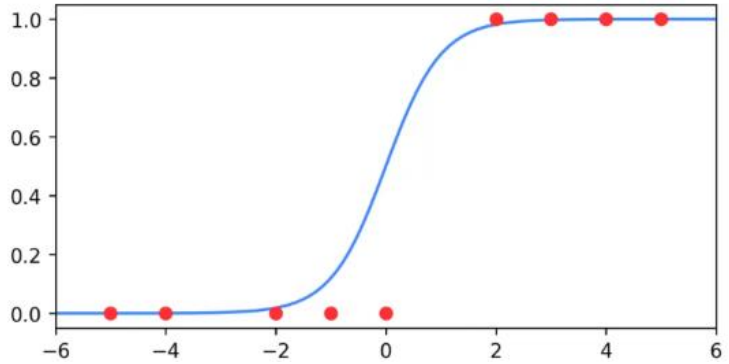
2. Aparatul matematic

- **Regresia liniară simplă**



$$\hat{y} = \beta_0 + \beta_1 x$$

- **Regresie logistică simplă**



$$\hat{y} = \sigma(\beta_0 + \beta_1 x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

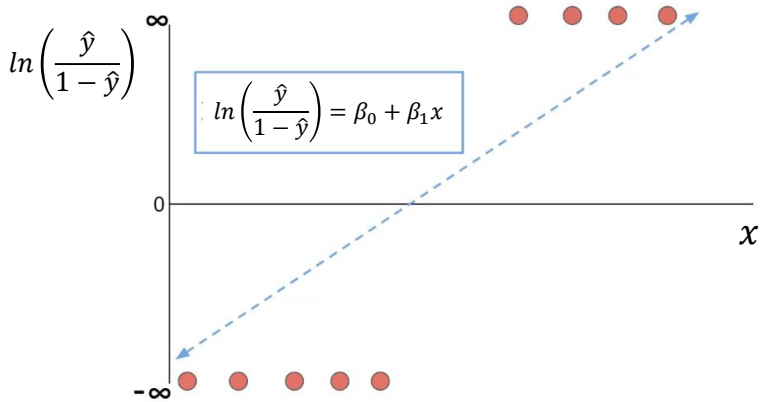
- **Cota unui eveniment *e* daca se știe probabilitatea acestuia *p(e)***

$$cota(e) = \frac{p(e)}{1 - p(e)}$$

- **Dependența lui $\hat{y}$ de coeficienți β_i**

$$\hat{y} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \rightarrow \hat{y} + \hat{y} * e^{-(\beta_0 + \beta_1 x)} = 1 \rightarrow \hat{y} * e^{-(\beta_0 + \beta_1 x)} = 1 - \hat{y} \rightarrow \frac{\hat{y}}{1 - \hat{y}} = e^{(\beta_0 + \beta_1 x)}$$

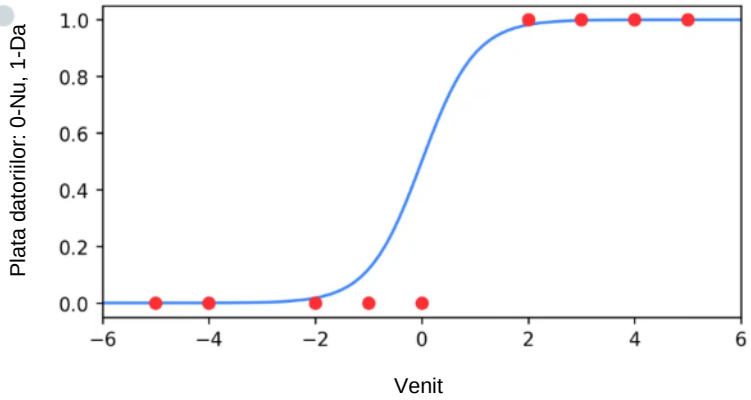
$$ln\left(\frac{\hat{y}}{1 - \hat{y}}\right) = \beta_0 + \beta_1 x$$



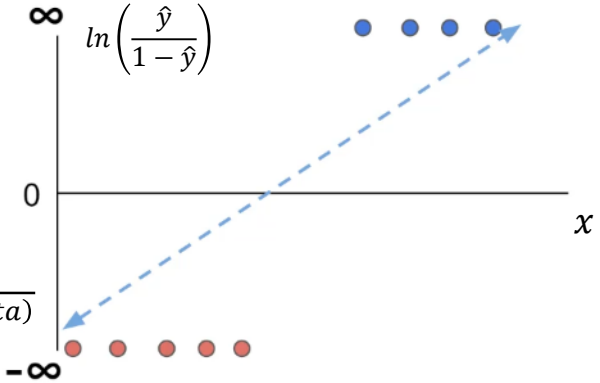
Conversia logaritmului cotei în probabilitate

$$ln\left(\frac{p}{1-p}\right) = ln(cota) \rightarrow \frac{p}{1-p} = e^{ln(cota)} \rightarrow p = (1-p)e^{ln(cota)} \rightarrow p = e^{ln(cota)} - p * e^{ln(cota)} \rightarrow$$

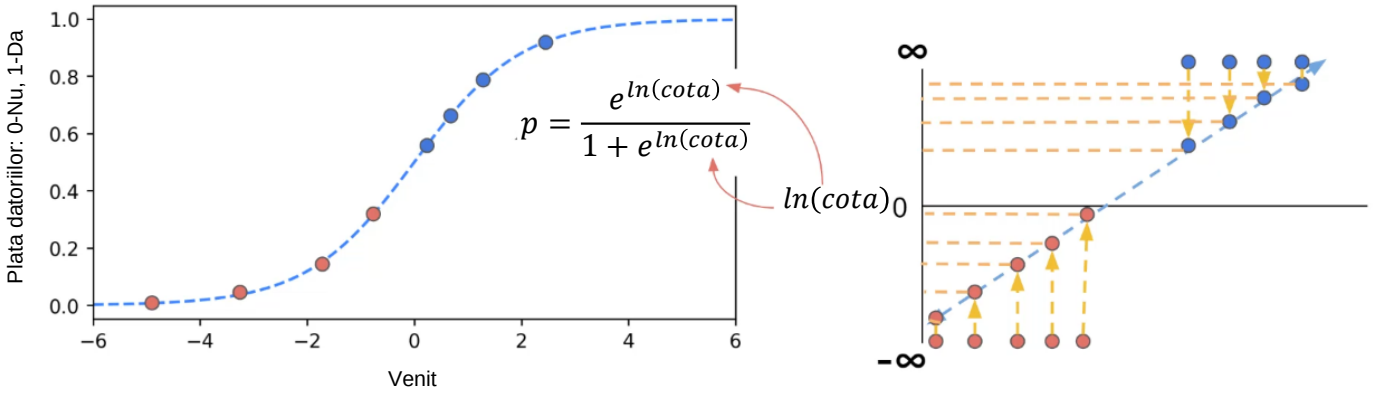
$$\rightarrow p + p * e^{ln(cota)} = e^{ln(cota)} \rightarrow p = \frac{e^{ln(cota)}}{1 + e^{ln(cota)}}$$



$$p = \frac{e^{ln(cota)}}{1 + e^{ln(cota)}}$$

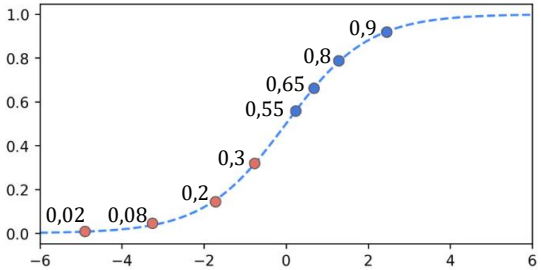


- **Dependență probabilităților de valorile coeficienților β_i**



- **Cei mai buni coeficienți β_i vor corespunde valorii maxima a probabilității p calculată ca produsul probabilităților conform exemplului**

$$p = 0{,}9 * 0{,}8 * 0{,}65 * 0{,}55 * (1 - 0{,}3) * (1 - 0{,}2) * (1 - 0{,}08) * (1 - 0{,}02) = 0{,}129$$



3. Instrumente Scikit-Learn pentru regresia logistică

- Se importa clasa algoritmului LogisticRegression

```
from sklearn.linear_model import LogisticRegression
```

- Se creează un model cu fixarea valorilor hiper-parametrilor:
  - penalty – tipul penalității la regularizare (valori posibile = {'l1', 'l2', 'elasticnet', 'none'}, implicit ='l2')
  - C - coeficientul invers de includere a penalizării (valori posibile = float, implicit =1.0)
  - solver – algoritmul utilizat pentru minimizarea sumei erorilor la realizarea trainingului (valori posibile = {'newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'}, implicit='lbfgs')
  - multi_class – specifica dacă sunt 2 clase sau mai multe (valori posibile = {'auto', 'ovr' (pentru binar), 'multinomial' (pentru multi-class)}, implicit='auto')
  - Max_iter - numarul maxim de iterații pentru minimizarea sumei erorilor (valori posibile = int, implicit =100)
  - L1_ratio – (doar daca penalty='elasticnet') gradul de includere a regularizării l1

```
model = LogisticRegression(solver='saga', multi_class='multinomial',
penalty='l1', C=120, max_iter=1000000)
```

- Se realizează trainingul modelului pe datele de training

```
model.fit(X_train, y_train)
```

- Se vizualizează coeficienții modelului

```
model.coef_
```

- Se realizează predicția pe datele de test

```
y_pred = model.predict(X_test)
```

- Se realizează predicția pe datele de test cu afișarea probabilităților

```
y_pred = model.predict_proba(X_test)
```

- Se evaluează performantele modelului
  - Se determina acuratețea modelului

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

  - Se vizualizează matricea confuziilor

```
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test, y_pred)
cm
```

  - Se vizualizează grafic matricea confuziilor

```
from sklearn.metrics import ConfusionMatrixDisplay
ConfusionMatrixDisplay(cm, display_labels=model.classes_).plot();
```

  - Se afișează raportul de clasificare - classification_report

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```