Linearni modeli za klasifikacijo

Jure Žabkar
jure.zabkar@fri.uni-lj.si

20. 4. 2021

A.I.LAB
Ljubljana

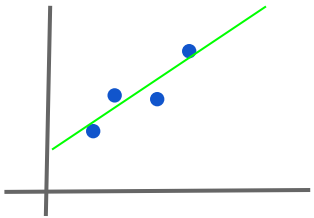
Vsebina

- Metoda K najbližjih sosedov, KNN
- Logistična regresija
- Diskriminantna analiza: LDA in QDA
- Praktični napotki za modeliranje

Strojno učenje

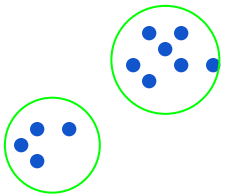
Nadzorovano

Regresija, Klasifikacija

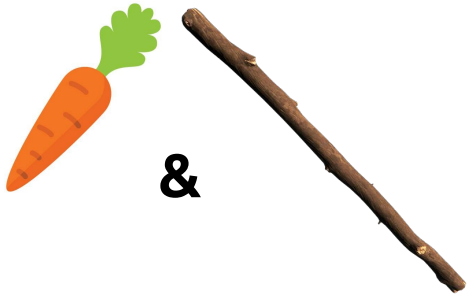


Nenadzorovano

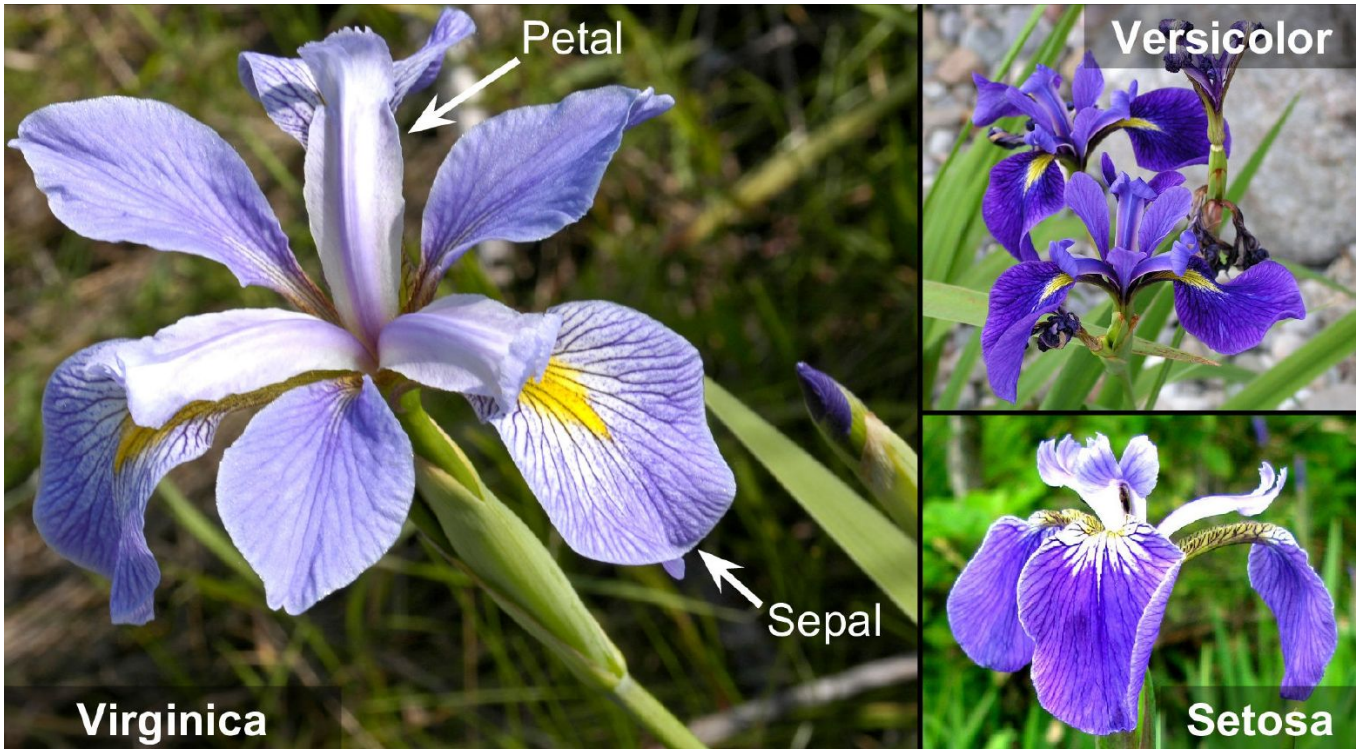
Gručenje, povezovalna pravila



Spodbujevano učenje

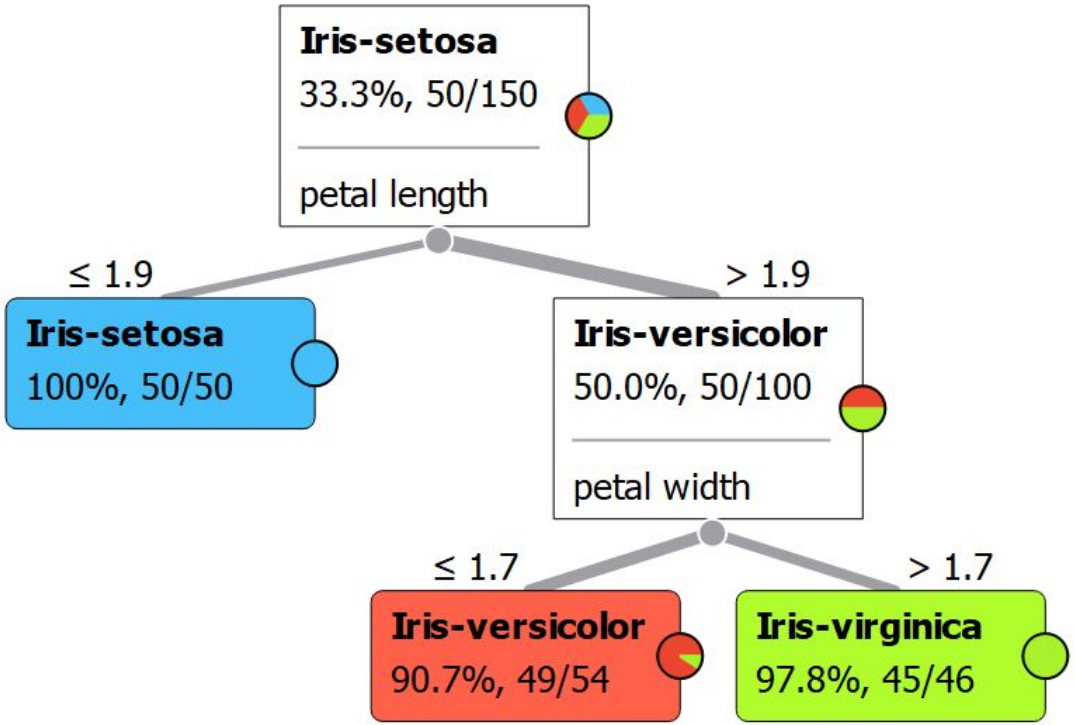


Iris

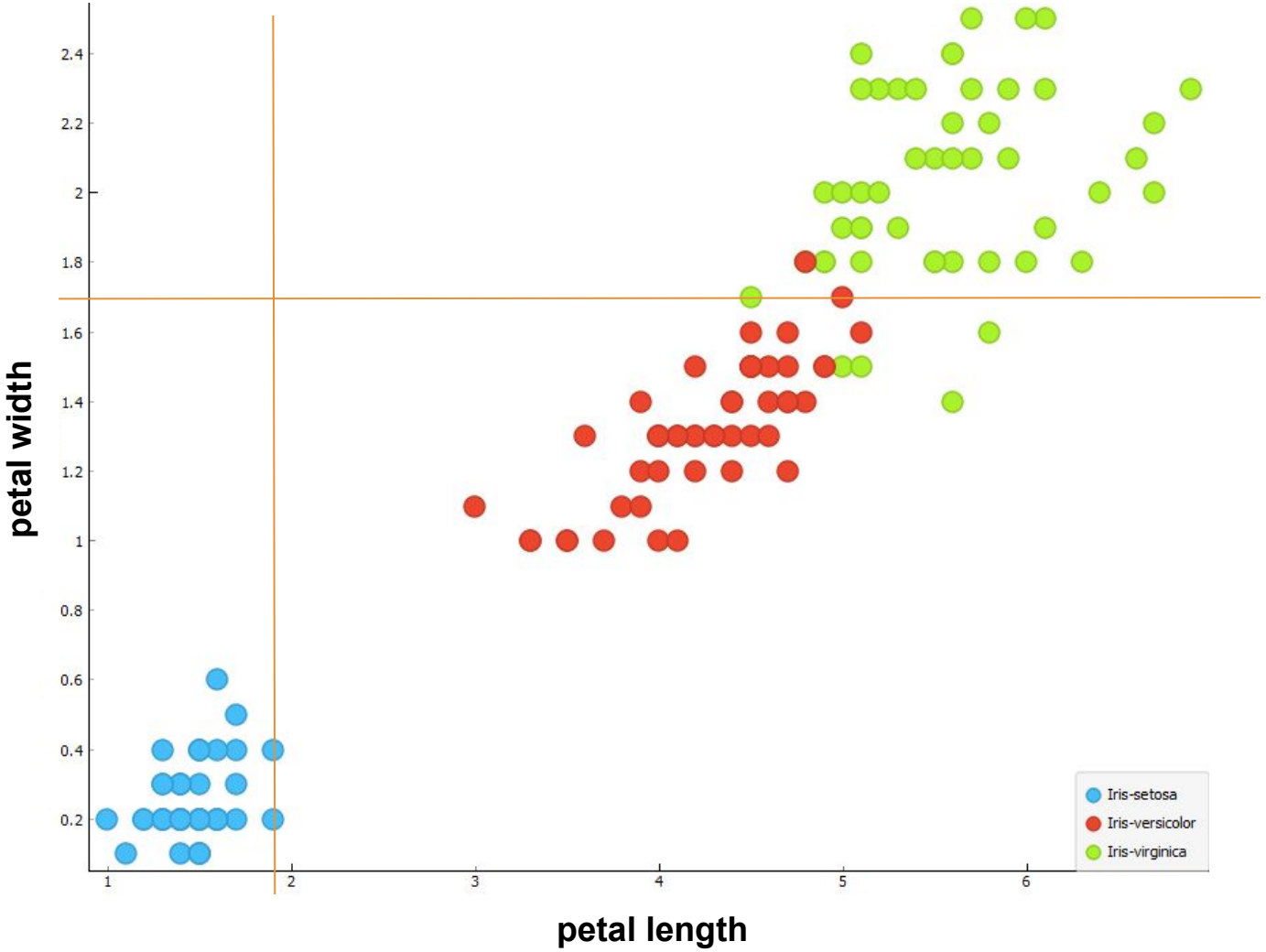


Photos reproduced from the corresponding Wikipedia pages. Iris-Virginica photo by Frank Mayfield (Creative Commons BY-SA 2.0), Iris-Versicolor photo by D. Gordon E. Robertson (Creative Commons BY-SA 3.0), and Iris-Setosa photo is public domain.

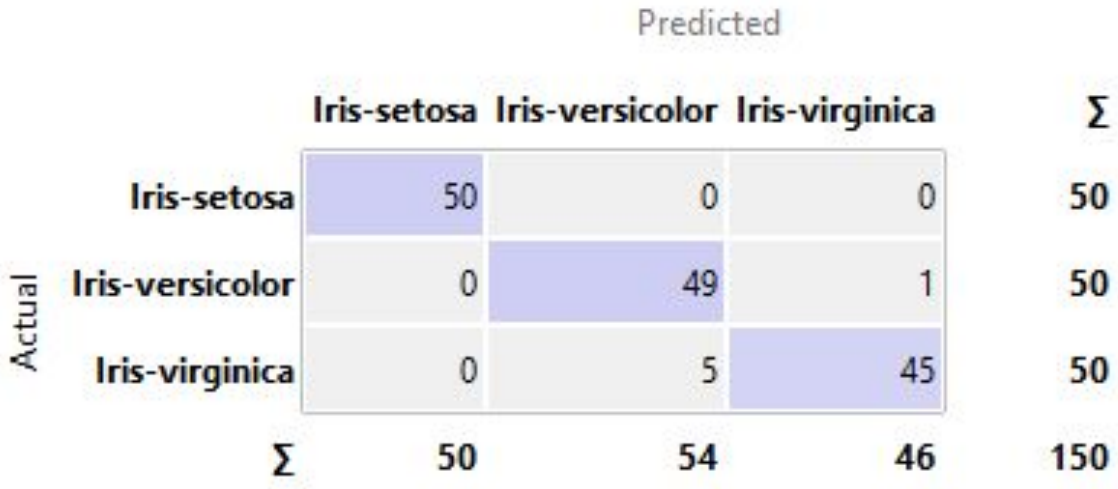
Iris


Iris-setosa
33.3%, 50/150
petal length
≤ 1.9
> 1.9
Iris-setosa
100%, 50/50
Iris-versicolor
50.0%, 50/100
petal width
≤ 1.7
> 1.7
Iris-versicolor
90.7%, 49/54
Iris-virginica
97.8%, 45/46

Iris

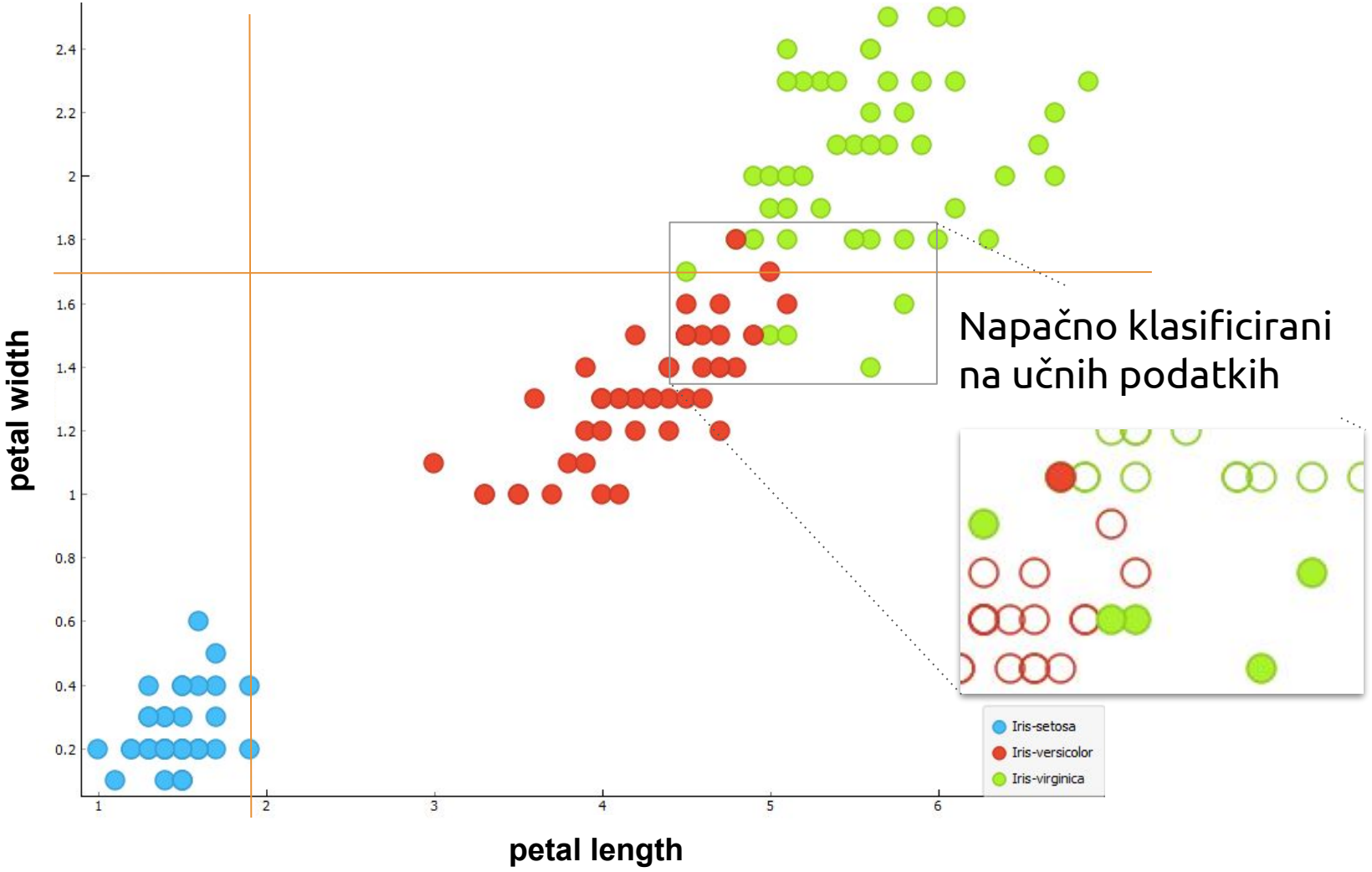

petal width
petal length
2.4
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
1
2
3
4
5
6
Iris-setosa
Iris-versicolor
Iris-virginica

Iris, odločitveno drevo

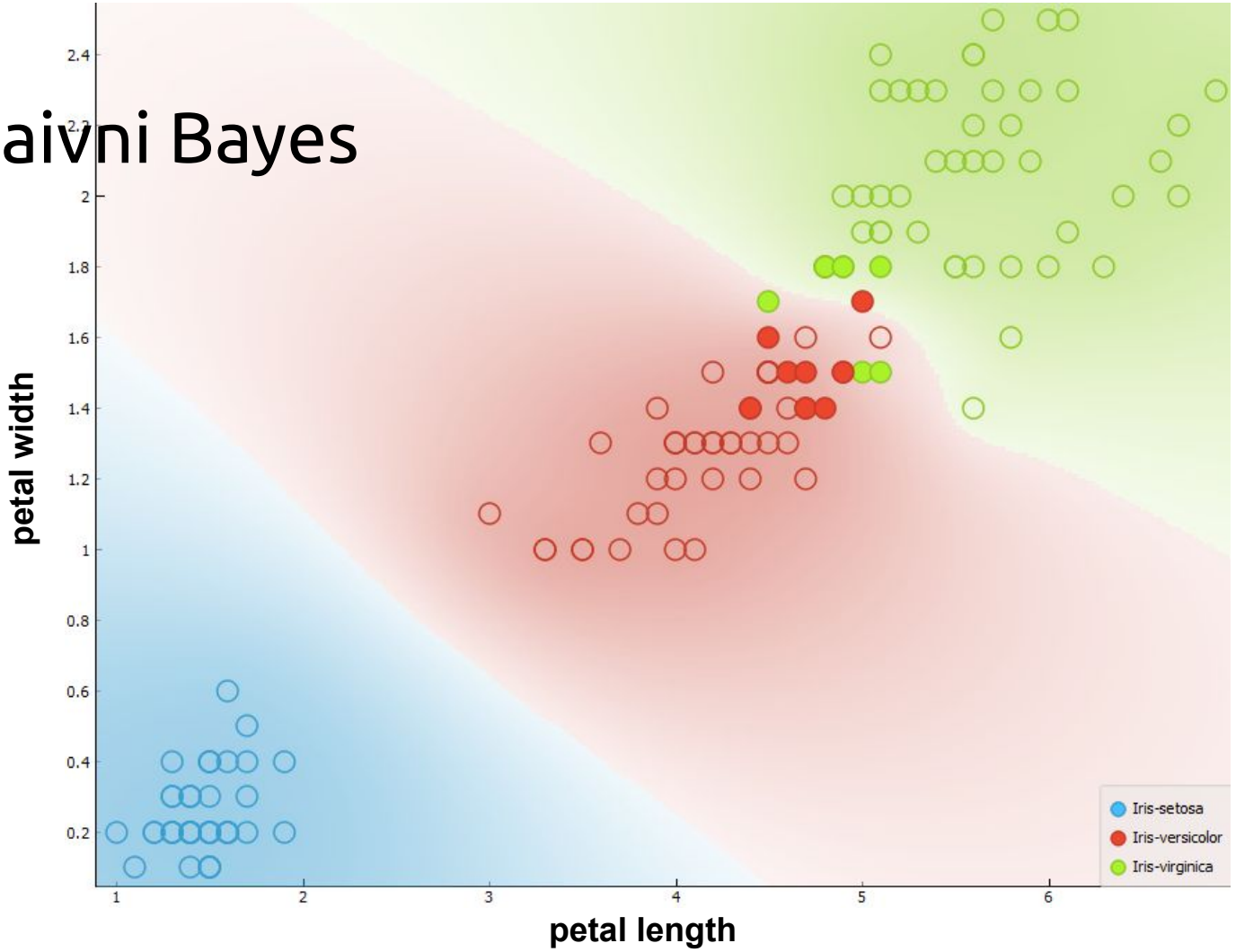


		Predicted			
		Iris-setosa	**Iris-versicolor**	**Iris-virginica**	**Σ**
Actual	**Iris-setosa**	50	0	0	**50**
	Iris-versicolor	0	49	1	**50**
	Iris-virginica	0	5	45	**50**
	Σ	**50**	**54**	**46**	**150**

Iris


petal width
petal length
2.4
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
1
2
3
4
5
6
Napačno klasificirani
na učnih podatkih
Iris-setosa
Iris-versicolor
Iris-virginica

Iris, naivni Bayes



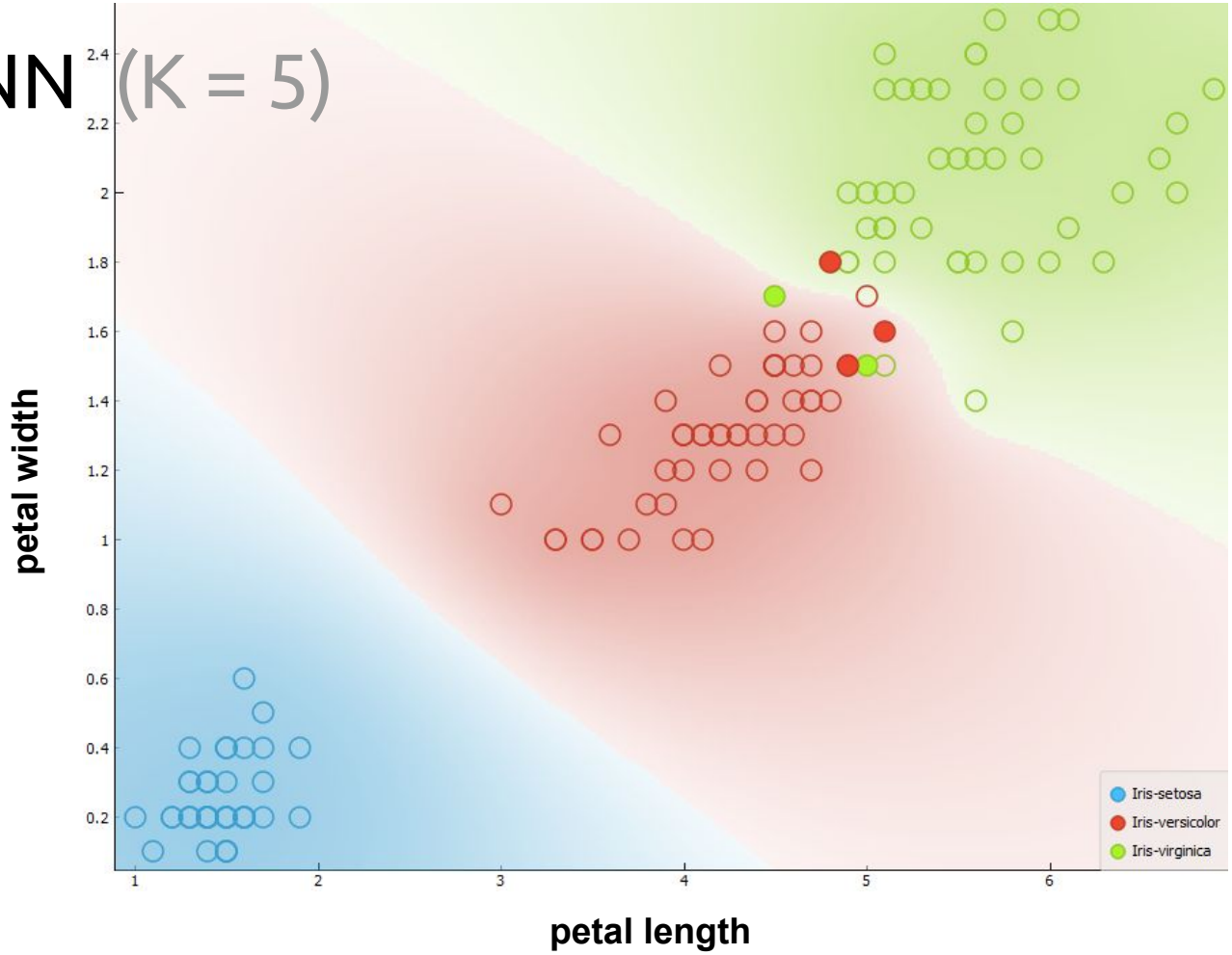
K najbližjih sosedov, KNN

Dana sta:

- *K* (število sosedov, ki določa velikost okolice $\mathcal{N}_0$)
- Testni primer x_0

Računamo verjetnost $P(Y=y \mid X=x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j)$

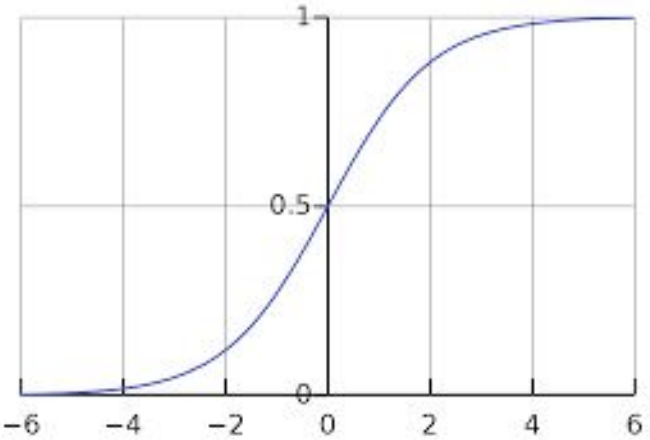
Iris, KNN (K = 5)


petal width
petal length
2.4
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
1
2
3
4
5
6
Iris-setosa
Iris-versicolor
Iris-virginica

Logistična regresija

Podobno kot pri linearni regresiji, le da namesto *Y* modeliramo verjetnost, da *Y* pripada določenemu razredu.

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$$



Logistična regresija, učenje modela

Za en učni primer:

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

Logistična regresija, učenje modela

Za celo učno množico vzamemo povprečje:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}log\left(\hat{p}^{(i)}\right) + \left(1 - y^{(i)}\right)log\left(1 - \hat{p}^{(i)}\right)\right]$$

$$\frac{\partial}{\partial\theta_j}\mathrm{J}(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\sigma\left(\theta^T \cdot \mathbf{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)}$$

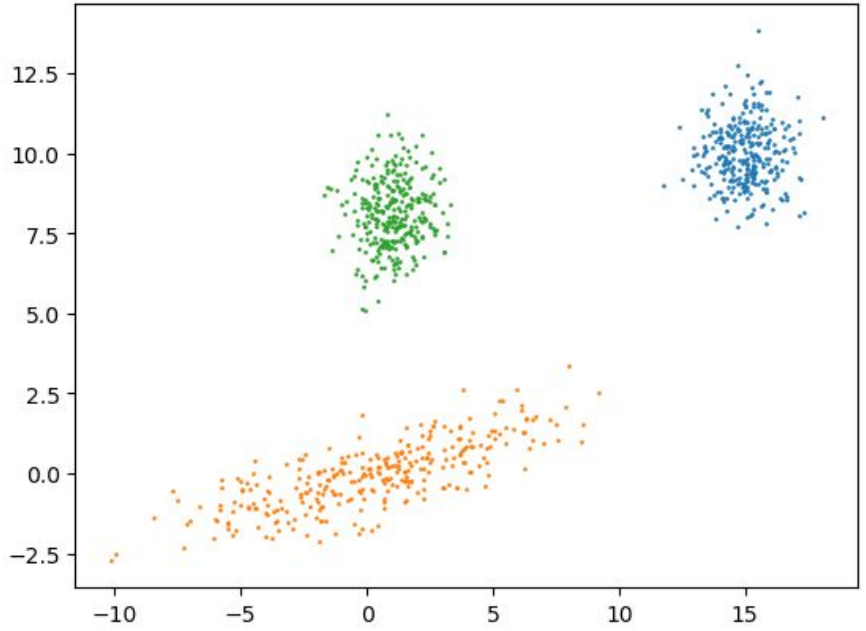
Rešimo z metodo gradientnega spusta.

Diskriminantna analiza

- Alternativa logistični regresiji

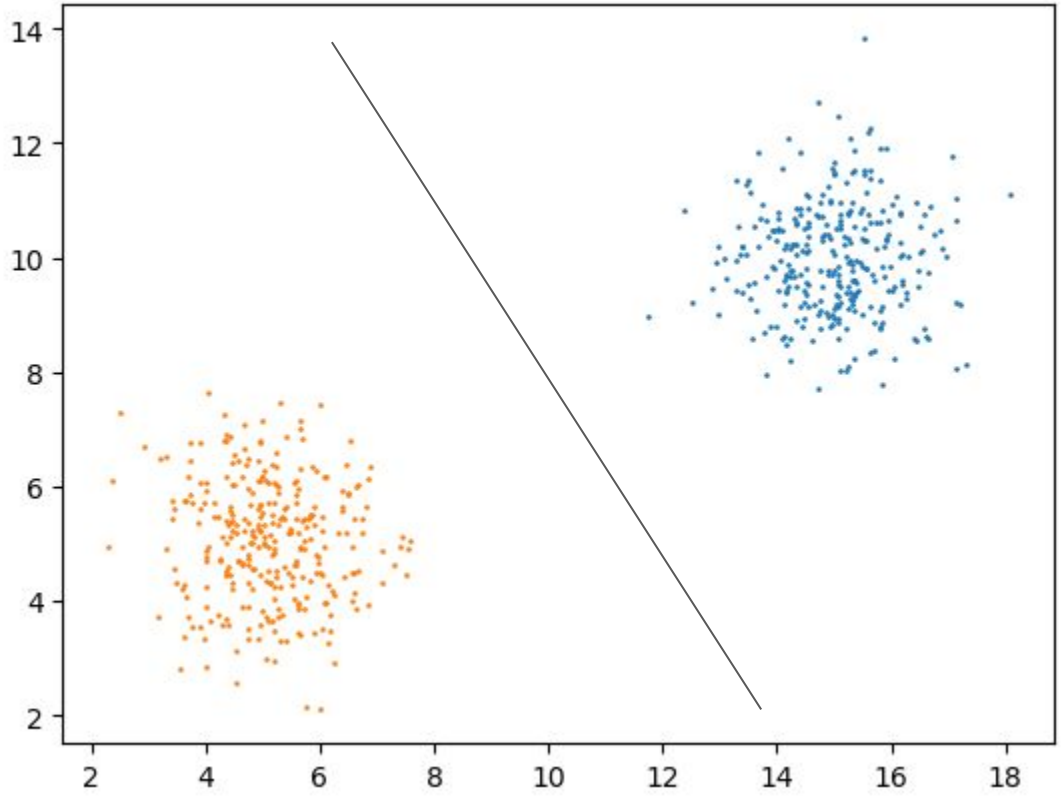
- modeliramo porazdelitve posameznih razredov in uporabimo Bayesov izrek za izračun $Pr(Y = k|X = x)$

- Log. reg. ima probleme pri majhnem vzorcu in ko so razredi dobro ločeni.



LDA (linearna diskriminantna analiza)

Predpostavlja, da so učni primeri posameznega razreda iz normalne porazdelitve s svojim povprečjem in isto kovariančno matriko (za vse razrede).

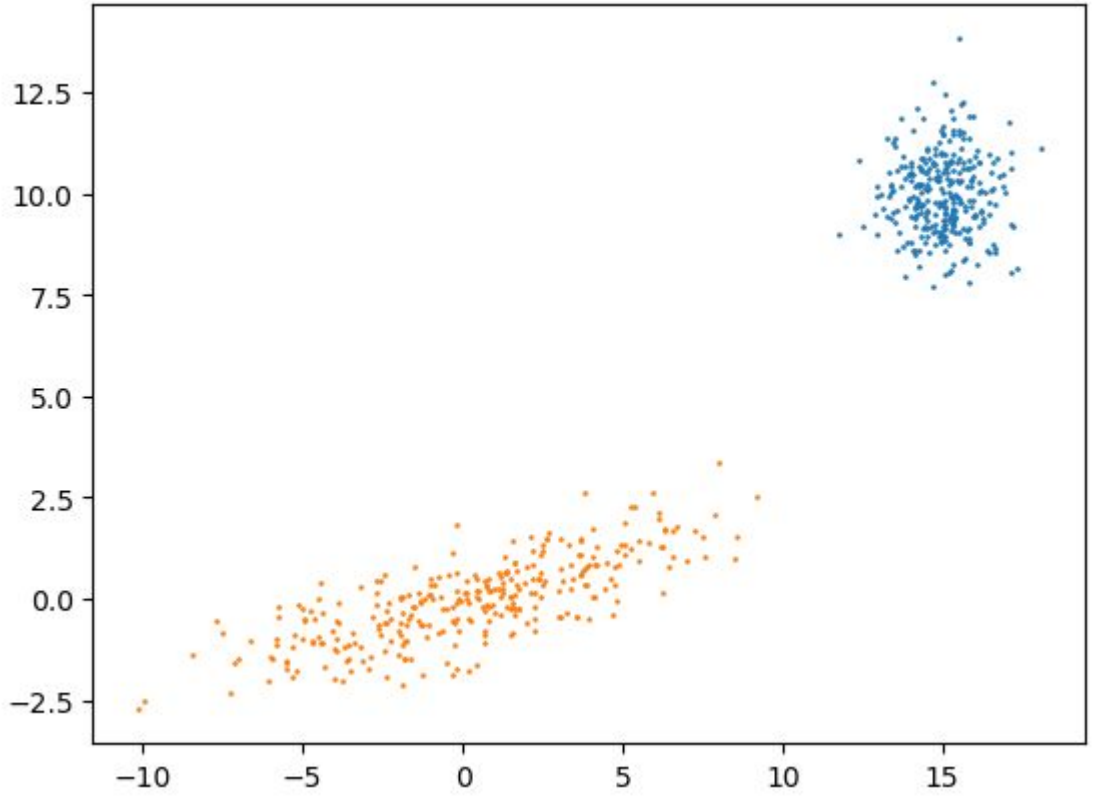


QDA (Kvadratna diskriminantna analiza)

Podobna predpostavka kot pri LDA

Razlika:

vsak razred ima svojo kovariančno matriko.



Strojno učenje = Koda + Podatki

```
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.neighbors import KNeighborsClassifier


def prepare_country_stats(oecd_bli, gdp_per_capita):
    oecd_bli = oecd_bli[oecd_bli["INEQUALITY"]=="TOT"]
    oecd_bli = oecd_bli.pivot(index="Country", columns="Indicator", values="Value")
    gdp_per_capita.rename(columns={"2015": "GDP per capita"}, inplace=True)
    gdp_per_capita.set_index("Country", inplace=True)
    full_country_stats = pd.merge(left=oecd_bli, right=gdp_per_capita,
                                  left_index=True, right_index=True)
    full_country_stats.sort_values(by="GDP per capita", inplace=True)
    remove_indices = [0, 1, 6, 8, 33, 34, 35]
    keep_indices = list(set(range(36)) - set(remove_indices))
    return full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[keep_indices]

# Load the data
oecd_bli = pd.read_csv("oecd_bli_2015.csv", thousands=',')
gdp_per_capita = pd.read_csv("gdp_per_capita.csv",thousands=',',delimiter='\t', encoding=
```

	iris	sepal length	sepal width	petal length	petal width
1	Iris-setosa	5.1	3.5	1.4	0.2
2	Iris-setosa	4.9	3.0	1.4	0.2
3	Iris-setosa	4.7	3.2	1.3	0.2
4	Iris-setosa	4.6	3.1	1.5	0.2
5	Iris-setosa	5.0	3.6	1.4	0.2
6	Iris-setosa	5.4	3.9	1.7	0.4
7	Iris-setosa	4.6	3.4	1.4	0.3
8	Iris-setosa	5.0	3.4	1.5	0.2
9	Iris-setosa	4.4	2.9	1.4	0.2
10	Iris-setosa	4.9	3.1	1.5	0.1

Iskanje napak na zviti pločevini

Foto: Tomo Jeseničnik/SIJ Acroni (prva slika) & Andrew Ng (druga slika)

Iskanje napak na zviti pločevini

- 39 različnih vrst napak
- **Osnovni model**:
  naučena NN na teh podatkih ima 76,2% točnost
- Cilj je doseči **90% točnost**
  to točnost pri tem problemu dosega človek

Vir: Andrew Ng

Iskanje napak na zviti pločevini

Vprašanje:

Kaj je potrebno narediti, da bi se približali točnosti 90%?

Izboljšamo kodo ali podatke?

Vir: Andrew Ng

Osredotočenost na **model**

- Zbiranje podatkov
- Učenje modela, ki dovolj dobro

  obravnava šum v podatkih
- Iterativna izboljšava modela na istih podatkih

Osredotočenost na **podatke**

Izboljšava **konsistentnosti podatkov**

in **gradnja novih značilk**

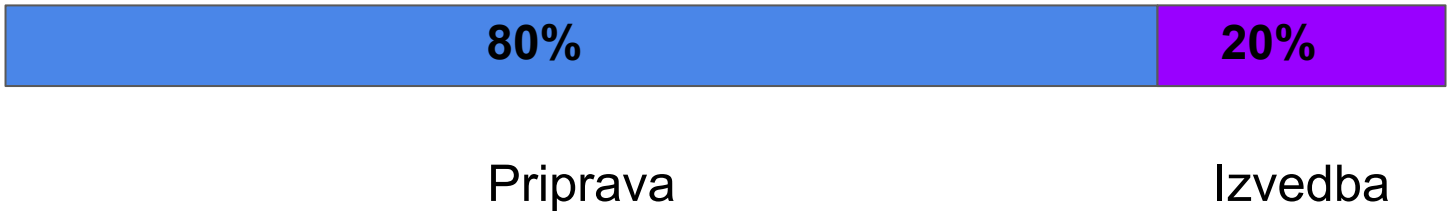
dopušča boljše rezultate **različnim modelom**.

Iskanje napak na zviti pločevini

	Klasifikacijska točnost
Izhodišče (nevronska mreža)	76,2%
Izboljšava modela	**+ 0%** => 76,2%
Izboljšava podatkov (v 2 tednih)	**+ 16,9%** => 93,1%

Vir: Andrew Ng

Podatki so *“hrana”* za strojno učenje



Vir: Andrew Ng