### Žiga Fon:

LV02: Strojno učenje s scikit-learn

* Knjižnjica scikit-learn

# Scikit-learn knjižnjica

Namenjena je strojnemu učenju, torej učenju modelov, ki na podlagi podatkov izvajajo npr. klasifikacijo v razrede, regresijo (napoved vrednosti parametra), ali nenadzorovano učenje (rojenje - clustering).

<https://scikit-learn.org/stable/index.html>

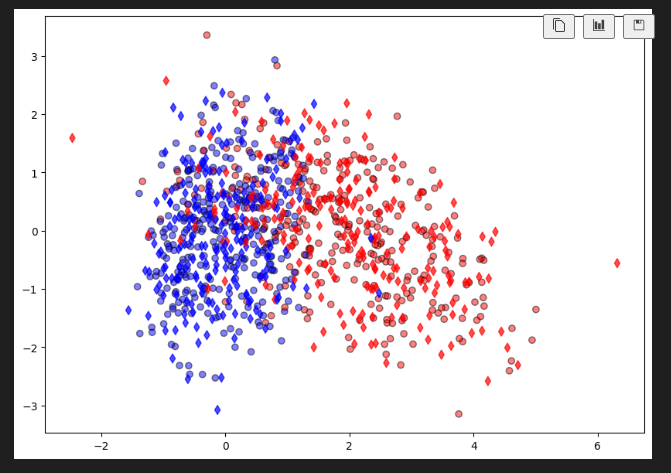
Slika, ki vsebuje besede besedilo, posnetek zaslona, oblikovanje

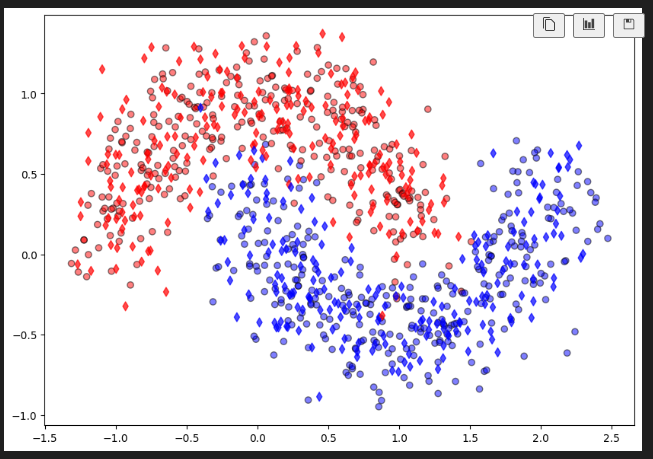
Opis je samodejno ustvarjen Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, tiskanje

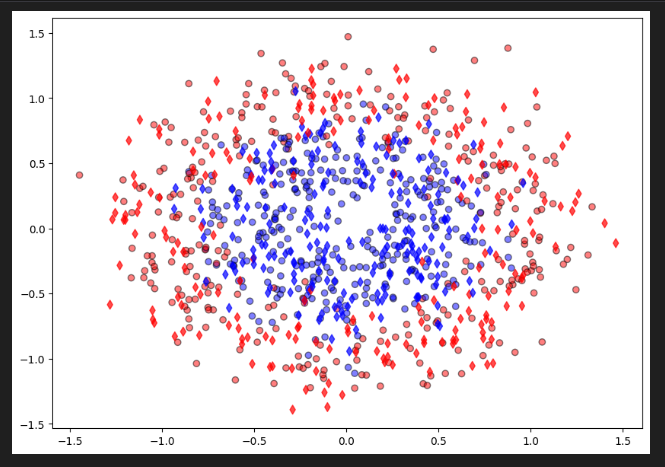
Opis je samodejno ustvarjen

## Podatki / Datasets

#### Preglej tri generatorje umetnih podatkov in te podatke izriši.







#### Opiši posamezni podatkovni set (razporeditev vzorcev, število razredov, število značilk – features, najpomembnejši parametri pri generiranju podatkov).

Razporeditev vzorcev -> Podatki so precej razpršeni, vizualno smo jih ločili v 3 razrede (naključna roja, krogce in nagnjeni elipsi).

**Število razredov:** Enako št ločenih skupin. Kar sta dve (rdeča in modra). Gre za klasifikacijo.

**Število značilk – features:** Značilke so posamezne merljive lastnosti ali značilnosti podatkov. To je ponavadi X podatkov. Torej vhodne spremenljivke. Značilki sta 2

**Najpomembnejši parametri pri generiranju podatkov:** so funkcija make moons and make circles

#### Kakšni so vzorci istega seta pri večkratnem generiranju ? Kako (s katerim parametrom) bi dosegel, da vedno dobiš enake podatke (vzorce)?

Če imamo enak seed so dobljeni naključni podatki enaki. Če ga nimamo pa ne

## Učenje in evalvacija modela

Metrike za oceno kvalitete modela : <https://scikit-learn.org/stable/modules/model_evaluation.html#classification-metrics>

#### Kaj pomenijo spremenljivke, in kako jih generiramo : X\_train, X\_test, y\_train, y\_test

X je spremenljivka. Y je rezultat funkcije. X\_train in Y\_train skupaj so podatki s katerimi treniramo model. Y\_test in X\_test so podatki za testiranje našega modela

#### Pomen in vpliv parametrov: test\_size, random\_state ?

Test size je zelo pomemben, ker z večjo testno množico dobimo lahko bolje natreniram model. Random state je funkcija s katero generiramo naključne podatke. Bolj natančno določa seed za random funckijo.

#### Izvedi učenje izbranega modela in vstavi rezultate:

precision recall f1-score support

0 0.771 0.846 0.807 195

1 0.839 0.761 0.798 205

accuracy 0.802 400

macro avg 0.805 0.804 0.802 400

weighted avg 0.806 0.802 0.802 400

#### Izpiši y\_pred. Kaj predstavlja?

Naše predikcije klasifikacij v strojnem učenju.

[1 1 1 0 1 1 0 1 1 0 0 0 0 1 0 1 0 0 0 0 0 1 0 0 0…]

#### Dodatno: izračunaj delež, koliko napovedi je pravilnih (napiši metodo)

#### Vstavi in interpretiraj pomen matrike Confusion matrix.

A chart of a number of colored squares

Description automatically generated with medium confidence

Iz matrike confusion lahko ugotovimo kako dobro klasifikacijo smo naredili.

True Positive (TP) False Negative (FN)

False Positive (FP) True Negative (TN)

#### Razloži pomen metrik: precision, recall, f1-score, accuracy ( support )

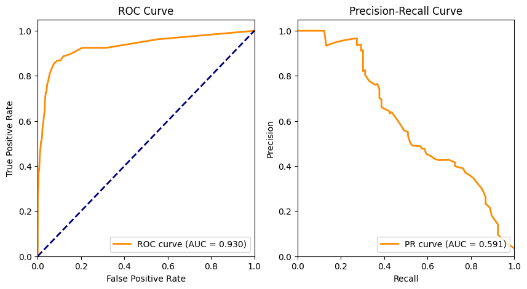
Precision je število verjetnost True Positive. Torej dejanskih 1 klasificiranih kot 1

Recal je število verjetnost True Negative. Torej dejanskih 0 klasificiranih kot 0

F1 je geometrijsko povprečje obeh

Accuracy: je verjetnost točnega rezultata

#### Preskusi print\_stats metodo. Komentiraj Precision-Recall in ROC krivulje.



Precision-Recall je v katerem nas zanima desni kot zgoraj.

ROC v njem nas zanima levi zgornji kot.

AUC Kvaliteta razpoznave (želimo čim bližje 1). Vrednosti so od nič do 1

**Primerjaj vsaj 3 ML modele, prikaži njihove rezultate in jih komentiraj**

**(glede na zahtevnost izbranega testnega seta podatkov)**

**Izbrani model poženi večkrat, pri čemer mu nastavljaj parametre na različne vrednosti. Vstavi in komentiraj rezultate.**

AdaBoost deluje izjemno dobro z visoko natančnostjo, priklicem in rezultati F1 za oba razreda. Med tremi modeli dosega največjo natančnost.

{'0': {'precision': 0.9851485148514851, 'recall': 0.9521531100478469, 'f1-score': 0.9683698296836983, 'support': 209.0}, '1': {'precision': 0.9494949494949495, 'recall': 0.9842931937172775, 'f1-score': 0.9665809768637532, 'support': 191.0}, 'accuracy': 0.9675, 'macro avg': {'precision': 0.9673217321732173, 'recall': 0.9682231518825621, 'f1-score': 0.9674754032737258, 'support': 400.0}, 'weighted avg': {'precision': 0.9681239373937394, 'recall': 0.9675, 'f1-score': 0.9675156524621745, 'support': 400.0}}

AdaBoost

precision recall f1-score support

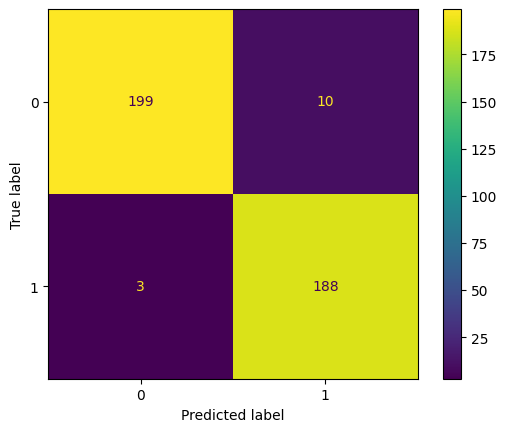
0 0.985 0.952 0.968 209

1 0.949 0.984 0.967 191

accuracy 0.968 400

macro avg 0.967 0.968 0.967 400

weighted avg 0.968 0.968 0.968 400



Random Forest kaže spodobno zmogljivost, vendar je bistveno nižja od AdaBoost. Ima uravnoteženo natančnost in priklic, vendar ne deluje tako dobro kot AdaBoost.

Random Forest

precision recall f1-score support

0 0.785 0.841 0.812 195

1 0.838 0.780 0.808 205

accuracy 0.810 400

macro avg 0.811 0.811 0.810 400

weighted avg 0.812 0.810 0.810 400

A yellow and purple squares with numbers

Description automatically generated

Nevronska mreža deluje podobno kot naključni gozd, z nekoliko boljšo natančnostjo in priklicem. Vendar pa še vedno ne uspe v primerjavi z AdaBoostom.

Neural Net

precision recall f1-score support

0 0.792 0.841 0.816 195

1 0.839 0.790 0.814 205

accuracy 0.815 400

macro avg 0.816 0.816 0.815 400

weighted avg 0.816 0.815 0.815 400

A yellow and purple squares with numbers

Description automatically generated

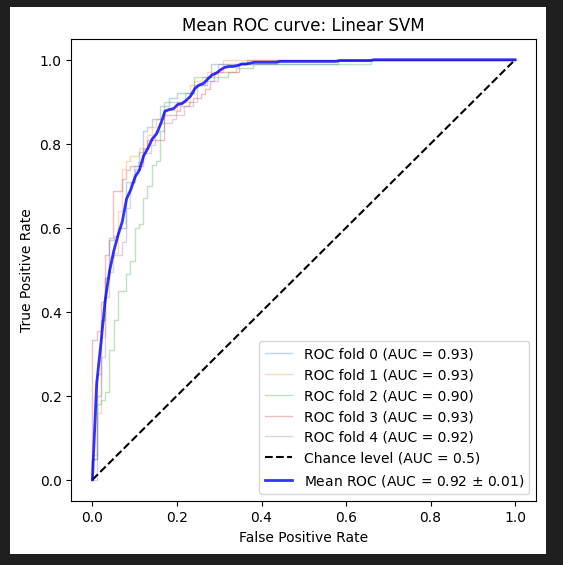
## Evalvacija s križno validacijo (Cross-validation)

#### Kako poteka K-Fold križna validacija, kaj pomeni n\_splits ?

Da vzamemo različne dele dataseta, 4/5 za training in 1/5 za testiranje. N splits se navizuje na ločevanje množice v omenjeni dve podmnožici. Podatkovno množico razdelimo na n\_splits delov, kjer učimo model na n\_splits-1 množicah in pustimo eni za testiranje podatkov.

#### Vstavi in komentiraj ROC krivuljo.

Če izberemo različne modele dobimo zelo različne krivulje. Smiselno je izbrati najboljši model. V grafu vidimo ROC za vse dele. Vsi deli so dobor ocenjeni in je razvidno, da je bil 4. del najboljši ter da je bil povprečni ROC 0.92.



## Dodatne naloge

#### Vizualizacija napovedanih točk :

Napiši funkcijo plot\_test\_points(), ki izriše testne podatke (točke) na DecisionBoundary, vendar naj pravilno klasificirane točke prikaže kot kroge, nepravilno pa kot križce.

#### Primerjava modelov z vizualizacijo:

Generiraj sliko (matplotlib, z subplot generiraj posamezne grafe), ki za vse testne sete izriše Decision boundary in testne točke za vsak posamezni model (ena vrstica grafov za en testni set).

#### Izvedi optimizacijo parametrov izbranega modela z GridSearchCV:

Napiši kodo, ki optimizira parametre modela, za najboljši rezultat izbrane evalvacije.

#### Uporabi drug (realni) podatkovni set in primerjaj modele:

Uporabi realne podatke izbranega dataseta, za klasifikacijo ciljne spremenljivke. Primerjaj uspešnost izbranih modelov.