In [13]:
```
import numpy as np
import pandas as pd
```

In [14]:
```
from matplotlib import pyplot as plt
```

In [15]:
```
from sklearn import svm
from sklearn import model_selection
from sklearn import metrics
from sklearn import datasets
from sklearn import preprocessing
```

In [16]:
```
data = datasets.load_breast_cancer()
```

In [17]:
```
X = data.data
y = data.target
```

Kada želimo da odredimo optimalne vrednosti hiperparametara, iz skupa za treniranje izdvajamo jedan manji skup podataka koji zovemo validacionim skupom. Nad njime dalje proveravamo kako se ponaša model koji razvijamo za različite vrednosti hiperparameta i biramo one hiperparametre koji daju najbolje rezultate u nekom smislu. Pogrešili bismo da za ovaj zadatak iskoristimo skup sa testiranje jer bi ocena koju bismo dobili kasnije prilikom testiranja modela sa odabranim vrednostima hiperparametara bila pristrasna.

Trening, validacioni i test skup ćemo u ovom primeru podeliti u odnosu 56:14:30.

In [18]:
```
X_train_and_val, X_test, y_train_and_val, y_test = model_selection.train_test_split(
    X, y, test_size = 0.3, random_state = 42, stratify = y)
```

In [19]:
```
X_train, X_val, y_train, y_val = model_selection.train_test_split(
    X_train_and_val, y_train_and_val, train_size = 0.8,
    random_state = 42, stratify = y_train_and_val)
```

In [20]:
```
scaler = preprocessing.StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_validation = scaler.transform(X_val)
X_test = scaler.transform(X_test)
```

Hiperparametri SVM modela koje ćemo podešavati su C i γ. C kontroliše jačinu regularizacije, a γ širinu Gausovog radijalnog jezgra (RBF kernela).

In [21]:
```
# jačina regularizacije je inverzno proporcionalna parametru C
# C mora biti strogo pozitivan broj
# podrazumevano se koristi kvadrat l2 regularizacije
Cs = [0.0001, 0.001, 0.01, 0.05, 0.1, 0.5, 1, 5, 10, 100, 1000]
```

In [22]:
```
gammas = [0.0001, 0.001, 0.01, 0.05, 0.1, 0.5, 1, 5, 10, 100, 1000]
```

Inicijalizujemo promenljive u kojima ćemo čuvati najbolje ocene i optimalne vrednosti hiperparametara.

In [23]:
```
best_f1_score = 0
best_C = None
best_gamma = None
```

Za svaki par vrednosti C i γ treniramo model na trening skupu i pratimo njegovu ocenu na validacionom

skupu.

In [24]:
```
for C in Cs:
    for gamma in gammas:
        model = svm.SVC(kernel='rbf', gamma=gamma, C = C)
        model.fit(X_train, y_train)
        f1_score = metrics.f1_score(y_val, model.predict(X_val))
        if f1_score > best_f1_score:
            best_f1_score = f1_score
            best_C = C
            best_gamma = gamma
```

In [25]:
```
print('Najbolji f1-score na validacionom skupu je: ', best_f1_score)
```

```
Najbolji f1-score na validacionom skupu je:  0.7692307692307693
```

In [26]:
```
print('Najbolji hiperparametri modela su: ', best_C, best_gamma)
```

```
Najbolji hiperparametri modela su:  0.0001 0.0001
```

Kreiramo konačni model, koristeći trening skup:

In [27]:
```
best_model = svm.SVC(kernel='rbf', gamma=best_gamma, C=best_C)
best_model.fit(X_train, y_train)
```

Out[27]:

▾ SVC

```
SVC(C=0.0001, gamma=0.0001)
```

I na kraju testiramo model koristeći test skup:

In [28]:
```
f1_score = metrics.f1_score(y_test, best_model.predict(X_test))
```

In [29]:
```
print('f1-score na test skupu je: ', f1_score)
```

```
f1-score na test skupu je:  0.7697841726618705
```