Algoritam K najbližih suseda

Na primeru algoritma k najbližih suseda (KNN) prikazaćemo značajnost standardizacije u slučaju kada koristimo euklidsku metriku. Algoritam k najbližih suseda klasifikuje nepoznatu instancu tako što pronalazi k instanci iz skupa za obučavanje koje su joj najbliže u smislu neke izabrane metrike i pridružuje joj klasu koja se najčešće javlja medu tih k instanci. U slučaju regresije, za predvidjanje se uzima prosečna vrednost k najbližih suseda iz skupa za obučavanje. Ovaj algoritam retko predstavlja najbolji izbor za rešavanje nekog problema, ali neretko daje relativno dobre rezultate, a izuzetno lako se implementira i primenjuje. Broj suseda k predstavlja hiperparametar modela.
Glavni razlog zašto je ovaj algoritam loš je taj što je teško naći odgovarajuću metriku. Najjednostavnija metrika je euklidska metrika primenjena na standardizovane podatke. Napredniji pristup je da se prvo nauči optimalna metrika (**metric learning**), a zatim pomoću nje implementira algoritam k najbližih suseda.

U primeru koji sledi model k najbližih suseda biće iskorišćen za klasifikaciji tumora dojke na maligne i benigne.

In [63]:
```
from sklearn import model_selection
from sklearn import metrics
from sklearn import preprocessing
from sklearn import datasets
from sklearn.neighbors import KNeighborsClassifier
import pandas as pd
```

In [64]:
```
data = datasets.load_breast_cancer() #ucitavamo podatke
```

In [65]:
```
X = pd.DataFrame(data.data, columns=data.feature_names)
y = data.target
```

In [66]:
```
X_train, X_test, y_train, y_test = model_selection.train_test_split(
    X, y, test_size=0.33, random_state=7, stratify=y)
# pomocu parametra stratify biramo da li hocemo da
# odnos klasa u trening i test skupu bude isti
```

Pravićemo model sa 5 suseda.

In [67]:
```
model1 = KNeighborsClassifier(n_neighbors=5)
```

In [68]:
```
model1.fit(X_train, y_train);
```

In [69]:
```
y_pred = model1.predict(X_test)
```

In [70]:
```
metrics.accuracy_score(y_test, y_pred)
```

Out[70]: 0.9468085106382979

In [71]:
```
metrics.f1_score(y_test, y_pred)
```

Out[71]: 0.957983193277311

Sada pravimo model sa standardizovanim podacima.

In [72]:
```
scaler = preprocessing.StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

Opet ćemo koristiti 5 najbližih suseda

In [73]:
```
model2 = KNeighborsClassifier(n_neighbors=5)
```

In [74]:
```
model2.fit(X_train, y_train);
```

In [75]:
```
y_pred = model2.predict(X_test)
```

In [76]:
```
metrics.accuracy_score(y_test, y_pred)
```

Out[76]: 0.9787234042553191

In [77]:
```
metrics.f1_score(y_test, y_pred)
```

Out[77]: 0.9833333333333333

Dobili smo bolje rezultate.