Zadanie 1. **Krzywe walidacji krzyżowej i optymalizacja parametrów**  
Wyniki treningu otrzymywane przy podziale train/valid/test bywają niestabilne i zależne od stanu generatora losowości, zwłaszcza przy małych danych. Rozwiązaniem jest *walidacja krzyżowa*. Dokończ przykład poniżej, tak by zoptymalizować model <KNeighborsRegressor> dla zakresu parametru n\_neighbors':range(1,60). Pokaż wyniki przy pomocy *nowych wizualizacji* w scikit-learn v1.3.

import sklearn

import sklearn.datasets

from sklearn.model\_selection import GridSearchCV

from sklearn.pipeline import Pipeline

from sklearn.preprocessing import StandardScaler

from sklearn.neighbors import KNeighborsRegressor

# some data

X, y = sklearn.datasets.load\_diabetes(return\_X\_y=True, as\_frame=True)

# composite/pipeline model

full\_model = Pipeline([('preprocess',StandardScaler()),('predictor',KNeighborsRegressor())])

param\_grid = {'...n\_neighbors':range(1,60)} # search for how to name parameters in Pipelines

# grid-search using scikit-learn API

grid\_search = GridSearchCV(

full\_model,

cv=10,

n\_jobs=1,

param\_grid=param\_grid,

scoring="...", # turn RMSE loss into scoring (higher=better),

return\_train\_score=True,

)

results = grid\_search.fit(X, y)

#%pip install --upgrade scikit-learn>=1.3

from sklearn.model\_selection import ValidationCurveDisplay

import matplotlib.pyplot as plt

import numpy as np

# extract scores for all folds from <cv\_results\_> attribute, to use them in ValidationCurveDisplay

all\_test\_scores = …

all\_train\_scores = …

# plot validation curves along with standard errors

display = ValidationCurveDisplay(

param\_name='n\_neighbors',

param\_range=range(1,60),

train\_scores=all\_train\_scores,

test\_scores=all\_test\_scores,

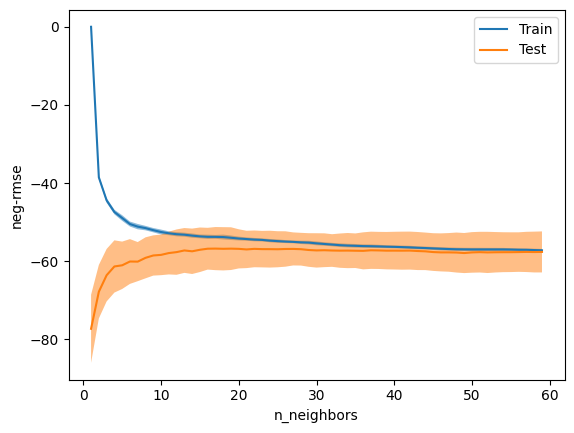
score\_name='neg-rmse'

)

display.plot(std\_display\_style='fill\_between')

plt.show()

Oczekiwany wynik:



Zadanie 2. **Elastyczność składni R**

Zbuduj model predykcyjny dla przeżywalności na Titanicu, *wprowadzając efekty interakcji* tak by pokazać *jak przeżywalność zależy od kombinacji* wieku i płci, klasy biletu i płci, lub innych par zmiennych. Postaraj się znaleźć prosty model z roc\_auc\_score powyżej 85%, zinterpretuj wyniki.

import seaborn as sns

import statsmodels.formula.api as smf

from sklearn.metrics import roc\_auc\_score

data = sns.load\_dataset('titanic')

data.fillna(data.median(), inplace=True)

model = smf.logit('survived ~ C(sex)', data=data).fit()

print(model.summary())

print(roc\_auc\_score(data['survived'], model.predict(data)))

Uwaga: o zmiennych typu kategorii i budowaniu ich interakcji można poczytać np. tu: <https://tilburgsciencehub.com/topics/analyze/regression/linear-regression/interaction-terms-r/>

Zadanie 3. **Ocena klasyfikacji binarnej**

Na potrzeby klasyfikacji binarnej budujemy najczęściej modele szacujące prawdopodobieństwo wystąpienia efektu (choroby / przeżycia / przestępstwa…) w skali ciągłej od 0 do 100%. Takie modele dają możliwość elastycznej decyzji w zależności od progu. Dla modelu poniżej znajdź próg decyzji przy którym **suma częstości błędów pierwszego i drugiego typu (FPR+FNR) jest najmniejsza** (można posłużyć się wizualizacją macierzy pomyłek).

import sklearn

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.pipeline import Pipeline

from sklearn.metrics import confusion\_matrix, ConfusionMatrixDisplay

X,y = sklearn.datasets.load\_breast\_cancer(return\_X\_y=True)

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.25,random\_state=41)

full\_model = Pipeline(

[('preprocess',StandardScaler()),

('predictor',LogisticRegression())]

)

model = full\_model.fit(X\_train,y\_train)

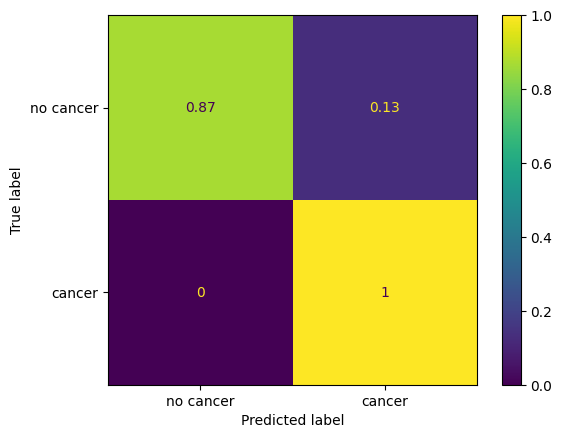
y\_score = model.predict\_proba(X\_train)[:,1] # risk/continous (!) predictions

t\_decision = 0.1 # try to optimize the decision threshold to achieve the lowest Type I + Type II error

conf\_matrix = confusion\_matrix(y\_train,y\_score>=t\_decision,normalize='true')

ConfusionMatrixDisplay(conf\_matrix,display\_labels=['no cancer','cancer']).plot()

Poniższa macierz jest poglądowa - należy ją poprawić tak by lepiej zająć się błędem typu II



Zadanie 4 - Scenariusz 6 z notebooka “Regression 1”.