**Университет ИТМО**

Кафедра информатики и прикладной математики

**Лабораторная работа №4**

Выполнила:

Бриль Марина P4217

# 

# 

### **Входные данные**

### <https://archive.ics.uci.edu/ml/datasets/Image+Segmentation>

Данные отражают характеристики сегментов 7 разных изображений - листвы, кирпича, тропинки, окна, неба, цемента и травы. Подробнее о данных можно прочитать в файле data/segmentation.names.

### **Ход работы**

Загрузим данные:

data = load\_data()

Сравним следующие ядра:

* SVC with linear kernel,
* LinearSVC,
* SVC with rbf kernel,
* SVC with sigmoid kernel,
* SVC with poly kernel,

Accuracy for SVC with linear kernel: 84.13%

Accuracy for LinearSVC (linear kernel): 53.97%

Accuracy for SVC with rbf kernel: 69.84%

Accuracy for SVC with sigmoid kernel: 12.70%

Accuracy for SVC with poly kernel: 87.30%

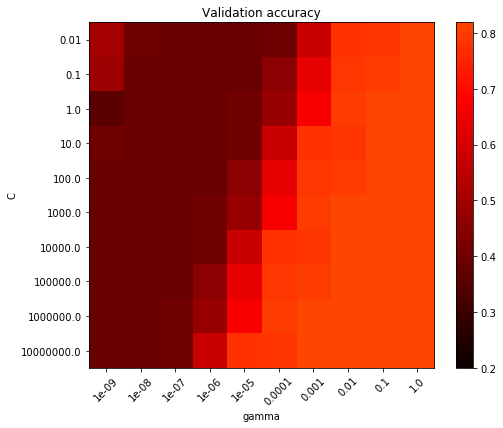
Из результатов сравнения видно, что лучше всего показало себя полиномиальное ядро. Полиномиальное ядро имеет следующий вид:

где: *d* - degree, *r* - coef0, *y* - gamma,

Есть ещё параметр *С* общий для всех ядер SVM. Является регулирующей константой, разрешающей конфликт между двумя задачами: максимизацией зазора и минимизацией потерь.

Посмотрим, как влияют изменения *С* и *y* на способность предсказания классов:

poly\_c\_gamma\_test(data)



Теперь, как влияют изменения *С* и *y* на способность предсказания классов:

In [10]:

poly\_c\_coef\_test(data)



Теперь, как влияют изменения *С* и *d* на способность предсказания классов:

poly\_c\_degre\_test(data)



Посмотрим на влияние коэффициентов *d* и *y*:

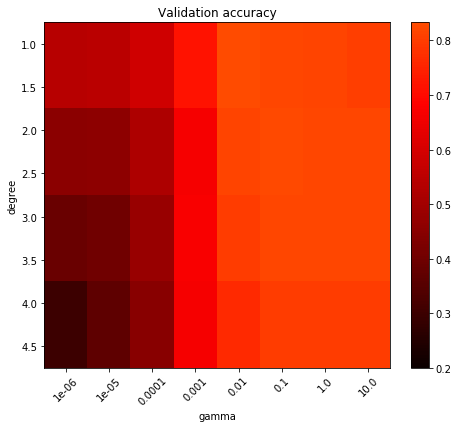
degree\_range = np.linspace(1, 4.5, 8)

gamma\_range = np.logspace(-6, 1, 8)

param\_grid = dict(gamma=gamma\_range, degree=degree\_range)

scores = get\_svc\_accuracy(param\_grid, len(gamma\_range), len(degree\_range))

draw(scores, gamma\_range, degree\_range, 'gamma', 'degree')



Теперь проверим показатели полиномиального ядра с наилучшими показателями коэффициентов:

In [25]:

poly\_svc = svm.SVC(kernel='poly', coef0=0.5, degree=2, C=1, gamma=1).fit(data.train\_x, data.train\_y)

pred = poly\_svc.predict(data.test\_x)

**print**('Accuracy: {:.2%}'.format(metrics.accuracy\_score(data.test\_y, pred)))

Accuracy: 87.30%

### **Вывод**

Для данного набора данных наиболее оптимальным будет использование полиномиального ядра SVC с *gamma*=1, *C*=1, degree=2 и *coef0*=0.5, хотя все манипуляции с параметрами не смогли особо увеличить точность классификатора