DataRoot courses research project

Gender Recognition by Voice

Made by Honcharov Danylo

Tasks:

- Explore the data
- Choose 2 models and metrics for quality assurance.
- Implement the models and compare them.
- Make some conclusions and ways to improve result
- Prepare iPython notebook and presentation.

Exploring data. Questions:

Is data balanced?

Statistical properties of data.

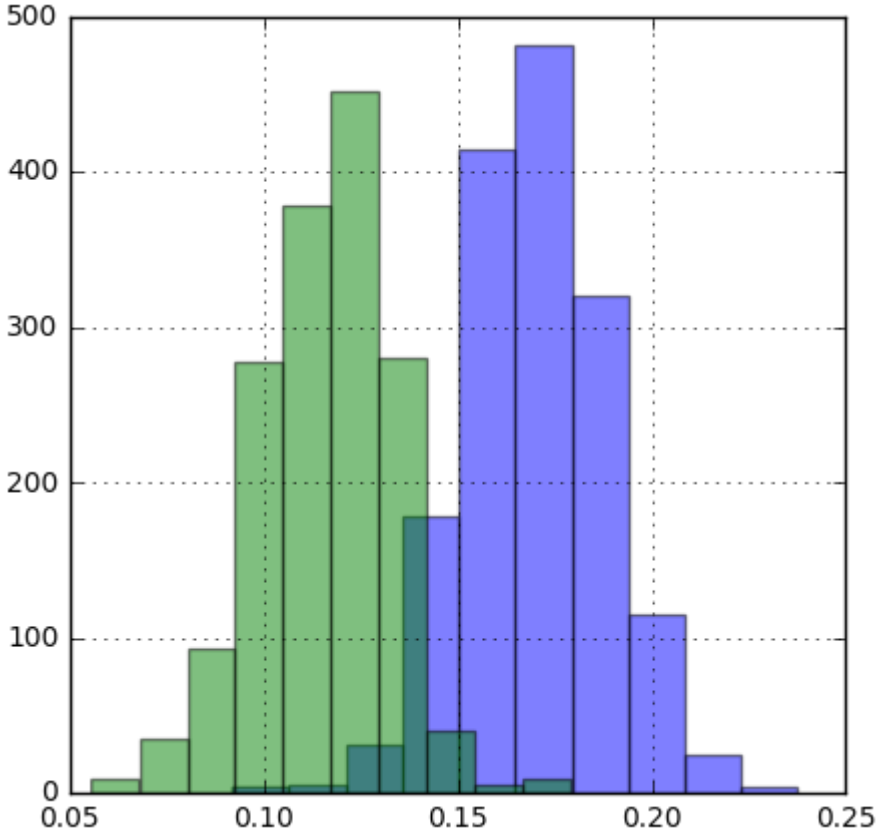
Possible correlations?

Have we some features that more useful than others?

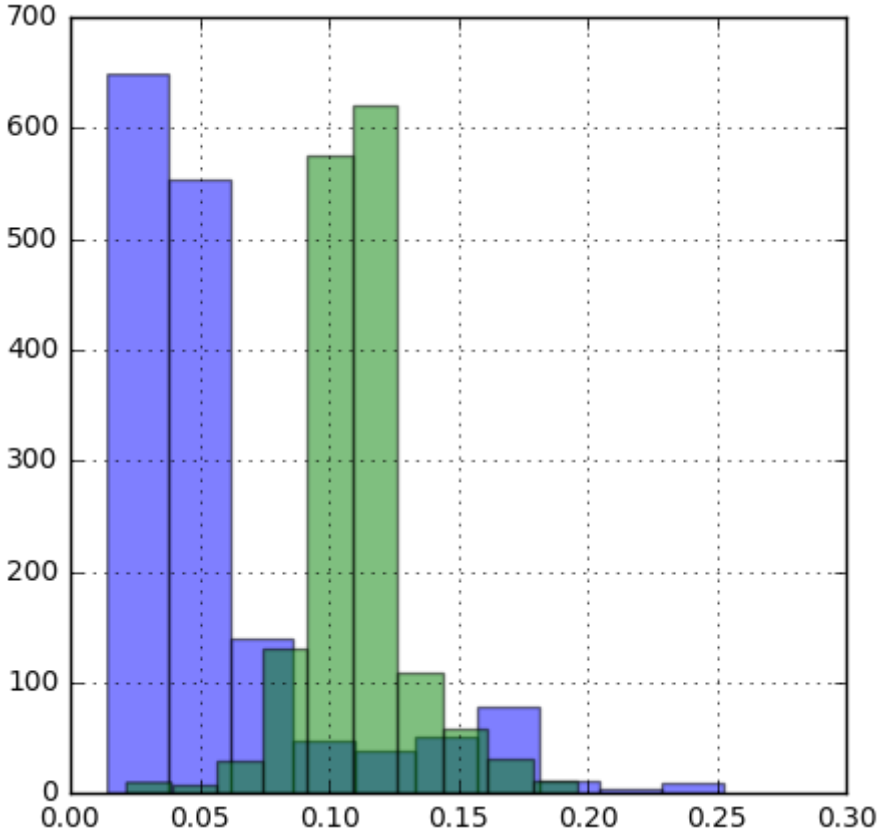
Graphical representation of data

Exploring data. Illustrations:

- **Meanfun feature:**

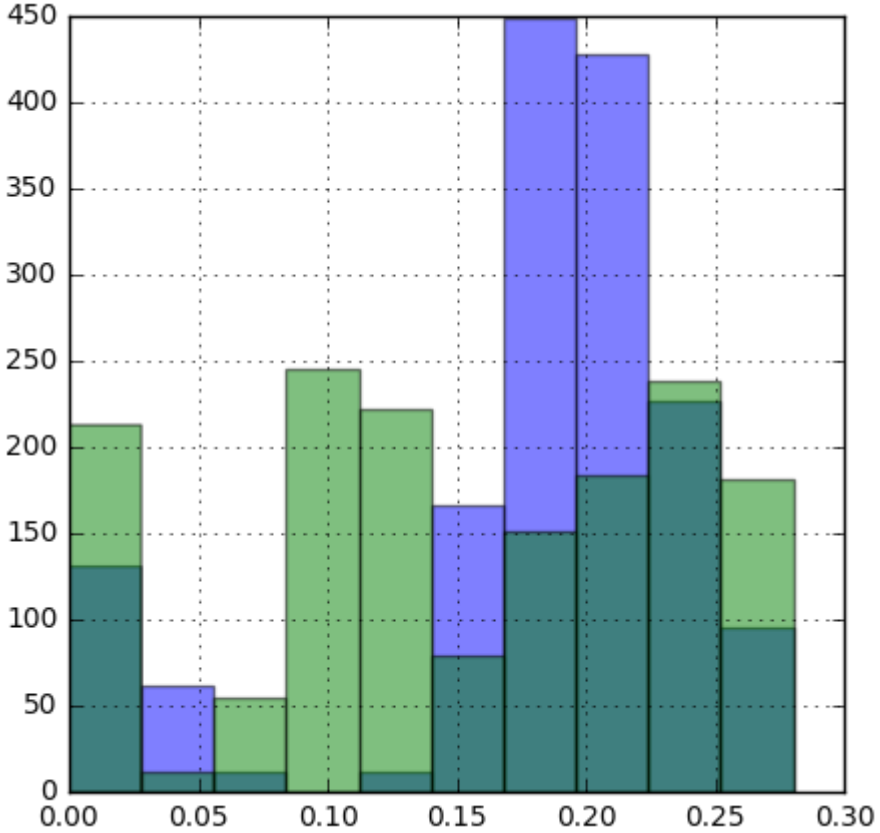


- **IRQ feature:**

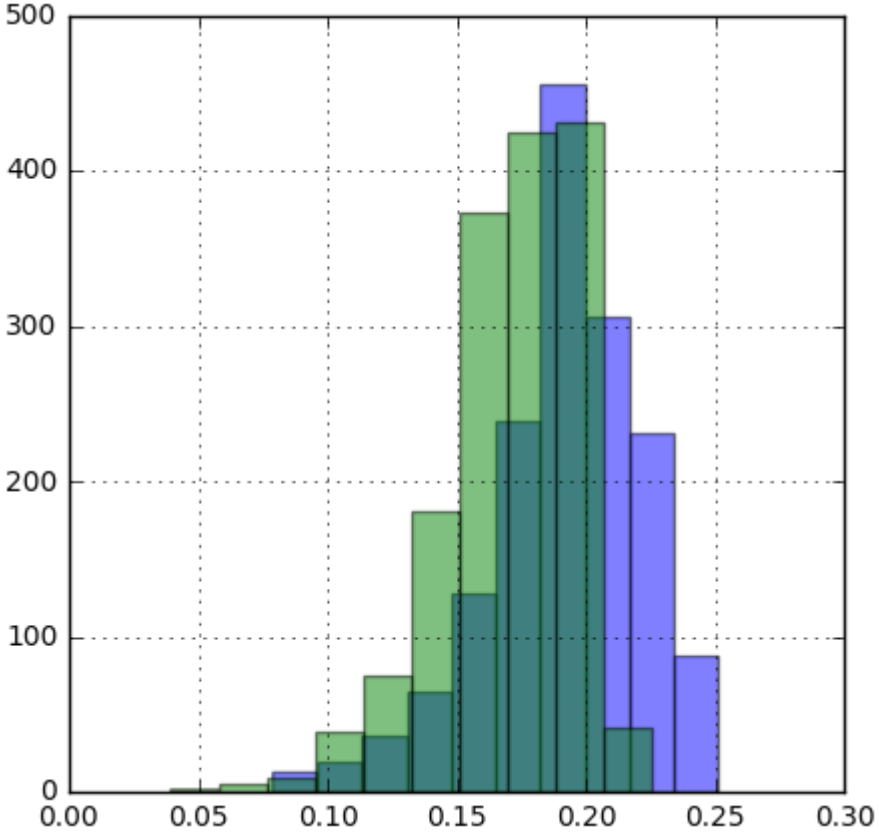


Exploring data. Illustrations:

- **Mode feature:**



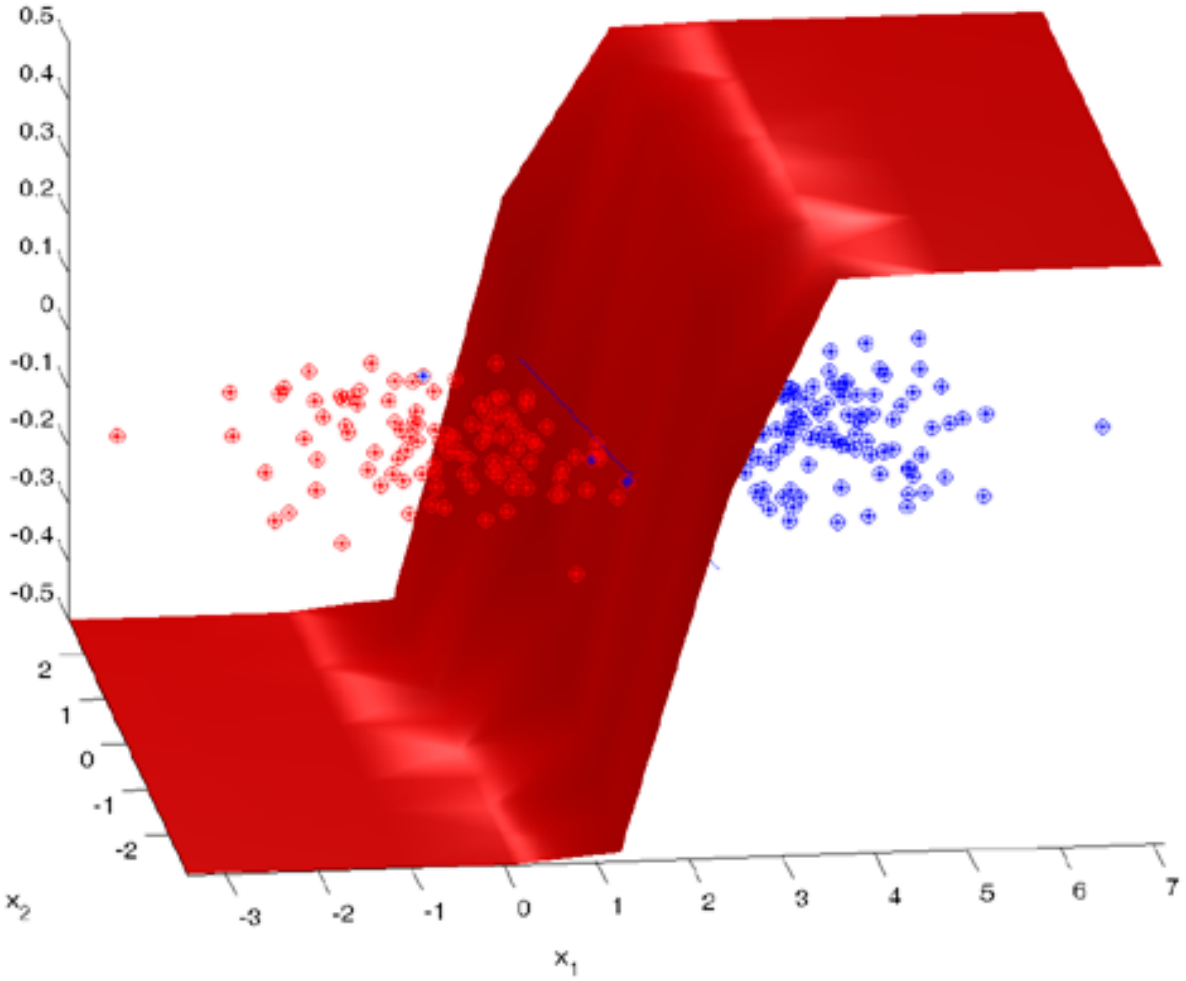
- **Meanfreq feature:**



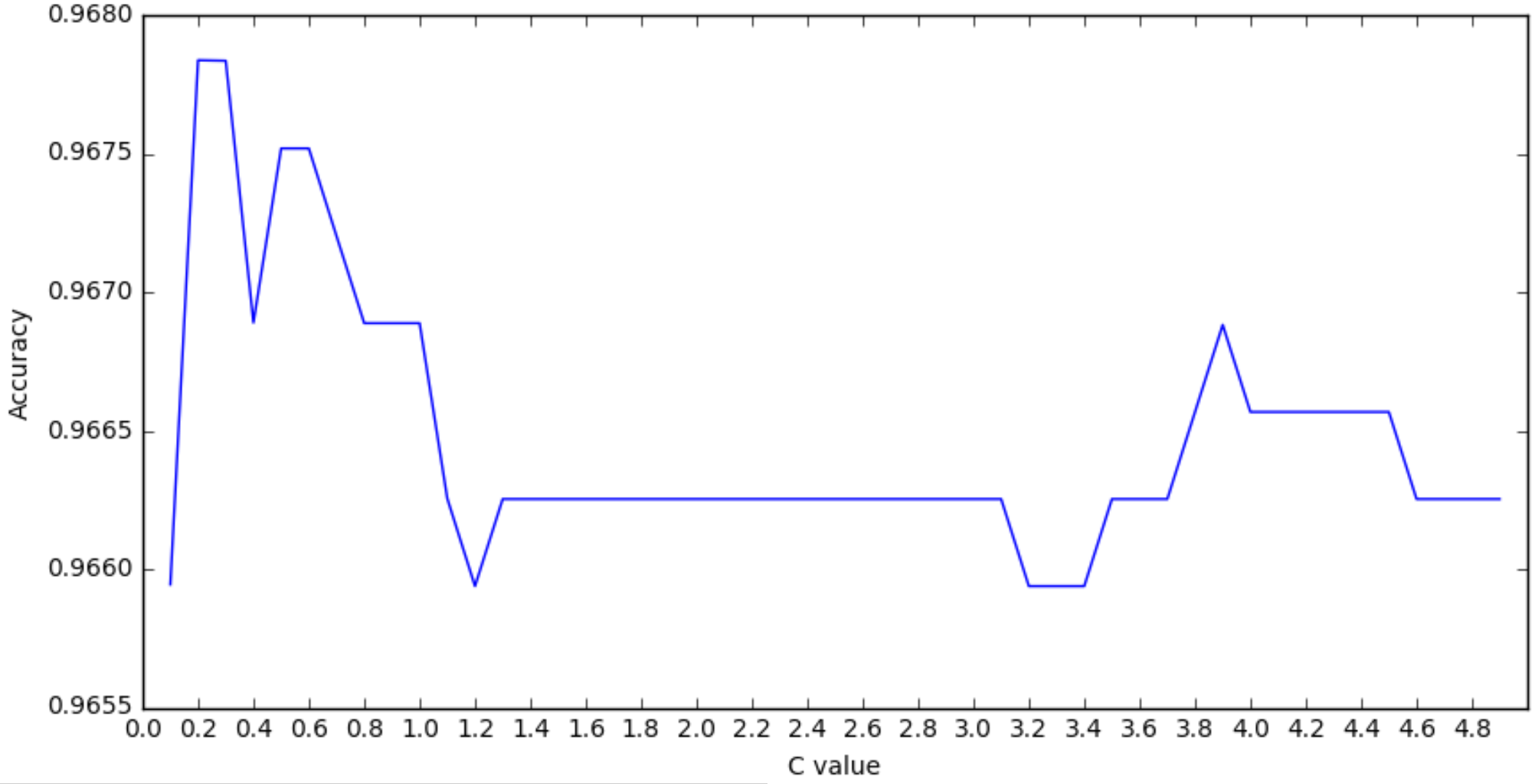
Researching models and metrics:

- **Logistic regression**
- **Support Vector Machines**

Logistic regression:



Finding best regularization for logistic regression in train set:




```
Best accuracy: 0.967836955656  best C: 0.2
```

How evaluate quality of classification:

```
Confusion matrix:
        Male    Female
Male:   156      7
Female: 4        150


Male f_measure 0.9659442724458206
Female f_measure 0.9646302250803858
```

1. Precision
2. Recall
3. Accuracy
4. F-measure
5. Confusion matrix

SVM. Default kernels:

RBF - Radial basis function:

$$K(x, x') = \exp(-\frac{\|x - x'\|^2}{(2\sigma^2)})$$

Linear kernel function:

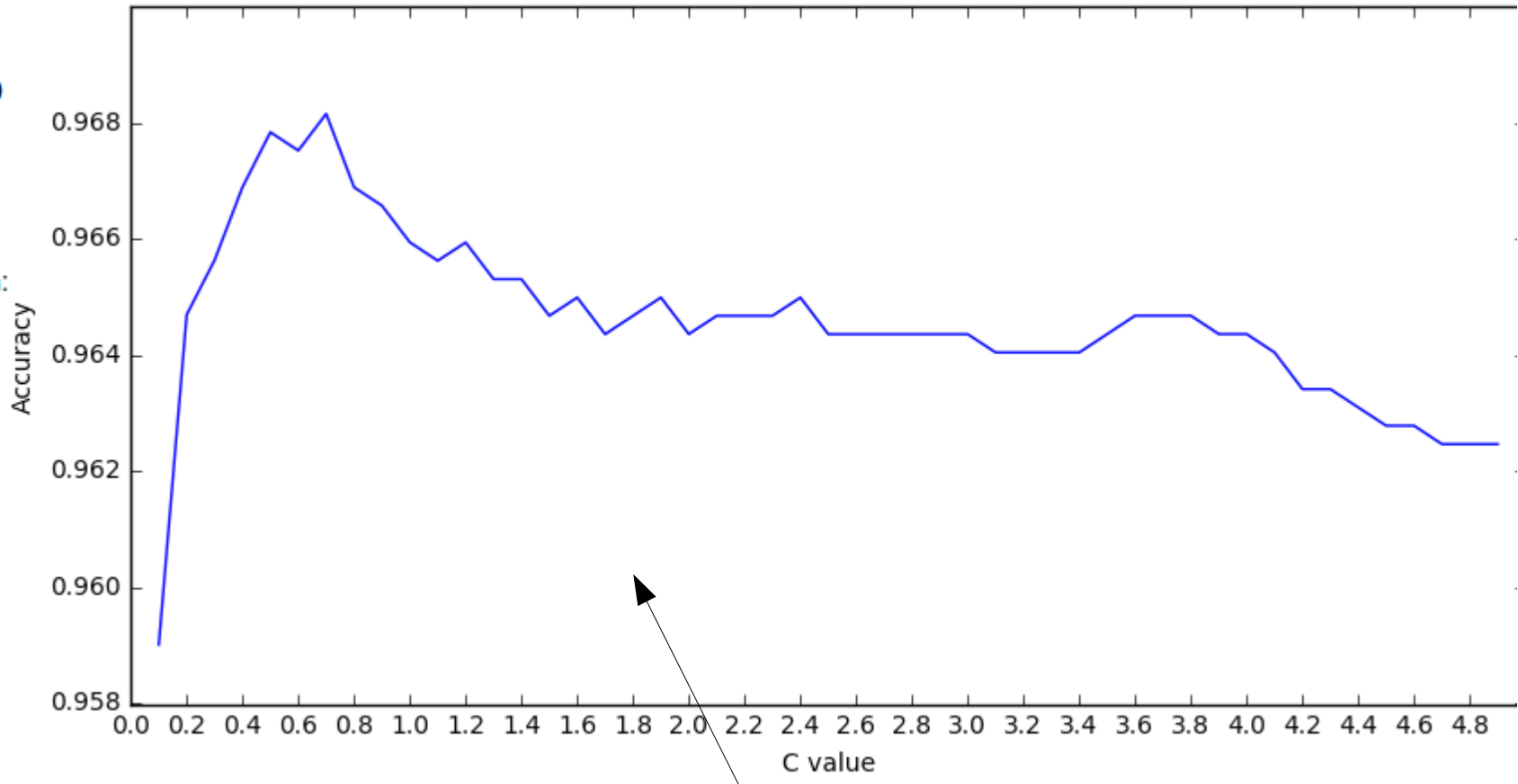
$$K(x, x') = x^T x'$$

Polynomial (poly) kernel function:

$$K(x, x') = (x^T x' + r)^n$$

Sigmoid kernel function:

$$K(x, x') = \tanh(\gamma x^T x' + r)$$




```
Default kernel (RBF) result: 0.990536277603
Linear kernel result: 0.977917981073
Poly kernel result: 0.958990536278
Sigmoid kernel result: 0.798107255521
```

Accuracy and regularization for RBF kernel.

SVM. GridSearch:

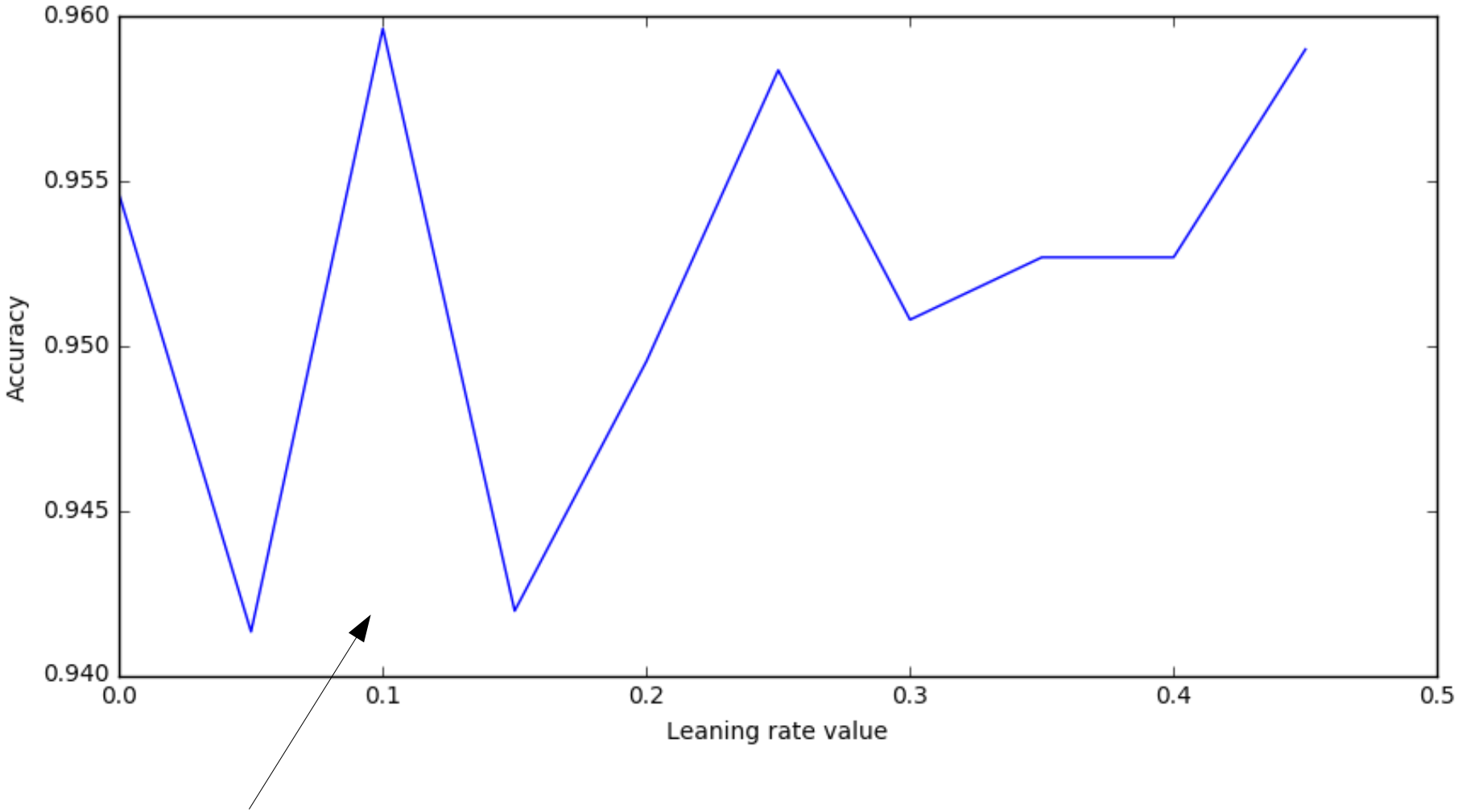
- **Characteristics of best kernel,found by GrindSearch:**

```
{'C': 1.1000000000000001, 'degree': 1.0, 'gamma': 0.151, 'kernel': 'poly'}
0.981072555205
             precision    recall  f1-score   support

          0       1.00      0.96      0.98       166
          1       0.96      1.00      0.98       151

avg / total       0.98      0.98      0.98       317
```

Simple perceptron.



Starting learning rate and accuracy for custom perceptron

Conslusions:

Exploring:

- Data is balanced
- Data have some useful features, that we can use
- We have nice visualization of data
- Data have some correlations between features.

Simple perceptron:

- This archaic thing worked
- We even find starting learning rate (which not influence much, cause we adjust rate to epochs)

SVM:

- Best default kernels - linear and rbf
- Best parametrized kernel,found by GridSearch - polynomial kernel
- Parametres of svm was explored
- Have some vizualization.

Logistic regression:

- Worked well
- Best C value - 0.2
- Result of classification was explored

Global:

- Numpy is lifesaver. Really.
- Pandas - as cool, as pandas. 🐼
- Scikit-learn - like a knife. Swiss knife.
- Matplotlib. I thought, it can nothing. I was wrong. Easy in use,good-looking plots.

Ways to improve:

Logistic regression:

- This model pretty simple, but have a good result. Nothing to improve, I think.

Simple perceptron:

- Make more complicated version (or maybe versions, many of them) with backpropagation
- Realize it with other libs, like Theano

SVM:

- Realize own version, that will be working with big dataset
- Search widely in space of parameters

Global:

Learn:

- Theano
- TensorFlow
- OpenCV
- xgboost

Thank you for attention.

Panda