Cross Validation

Till Max David

June 25, 2013

Outline

- Cross Validation
  - Implementation
  - Efficient LOOCV
  - Profiling
- Receiver operating characteristic
- Classification

Outline

- Cross Validation
  - Implementation
  - Efficient LOOCV
  - Profiling
- Receiver operating characteristic
- Classification

Implementation

Star operator

```
param_vals = [params[i] for i in np.arange(1,len(params),2)]
combs = list(it.product(*param_vals))
...
for i,c in enumerate(combs):
   method.fit(X_train,y_train,*c)
```

- extract each second entry of the parameters (the actual values)
- use cross product to list all possible combinations
- call the individual fit method with an arbitrary number of components

Implementation

eLOOCV

```python
def fit(...):
    ...
    if self.regularization == 0:
        self.regularization = self.efficient_cv(y)
```

Thus, one can simply use

```python
results[dataset_name]["regularization"] = cvkrr.regularization
```

in all cases.

Profiling

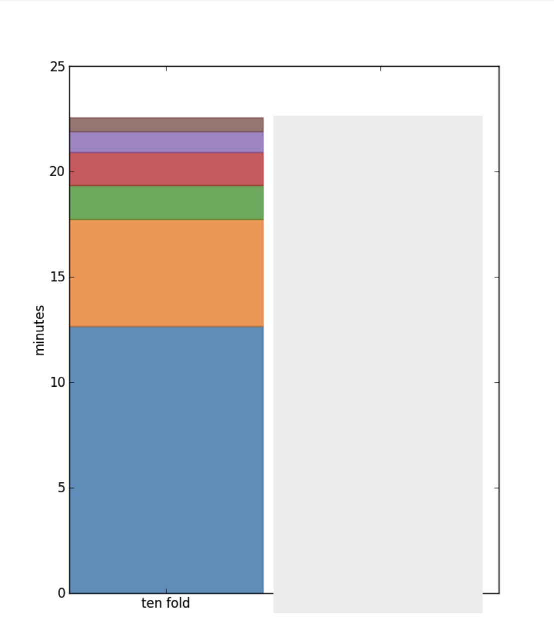
setup

```
import cProfile
...
def run_profile(method):
   if method == "tenfold":
      params = [ ... ,'regularization', np.logspace(-2,2,10)]
   else :
      params = [ ... ,'regularization', [0]]
   krr = imp.krr()
   imp.cv( ... ,nrepetitions=1000)
cProfile.run("run_profile('tenfold')")
cProfile.run("run_profile('efficientLOOCV')")
```

- run each method with 1000 repitITIONS
- run LOOCV with the *same* parameter range

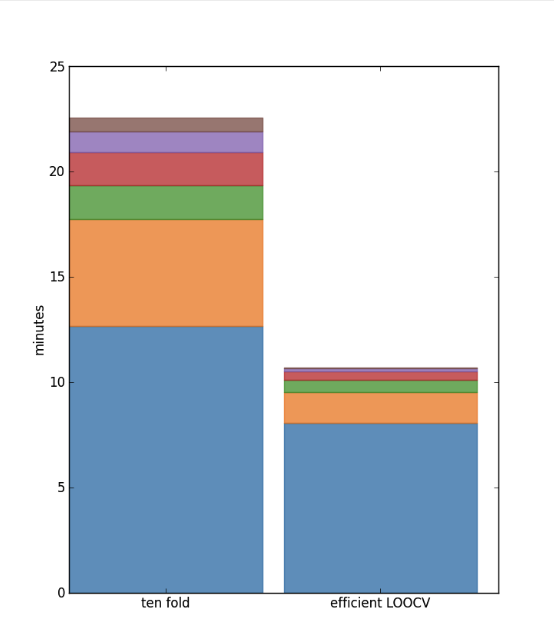
Profiling

10 Folds Cross Validation: Total time



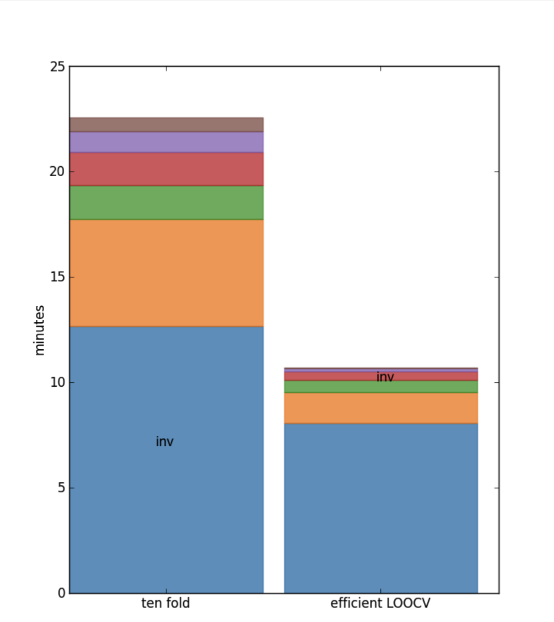
Profiling

Efficient Cross Validation: Total time



Profiling

critical computation: Total time



Profiling

Console

10 fold CV

ncalls	tottime	percall	cumtime	percall	filename:lineno(function)
1000001	759.195	0.001	871.458	0.001	basic.py:253(inv)
2000001	306.526	0.000	499.664	0.000	ps3_implementation.py:124(compute_kernel)
1	96.057	96.057	1615.387	1615.387	ps3_implementation.py:39(cv)
2000001	94.235	0.000	94.235	0.000	{scipy.spatial._distance_wrap.cdist_euclidean_wrap}
1000001	57.595	0.000	1337.970	0.001	ps3_implementation.py:100(fit)
2000002	40.103	0.000	40.103	0.000	{method 'any' of 'numpy.ndarray' objects}

efficient LOOCV

ncalls	tottime	percall	cumtime	percall	filename:lineno(function)
17166	483.351	0.028	626.219	0.036	ps3_implementation.py:148(efficient_cv)
8016064	88.382	0.000	88.382	0.000	{method 'dot' of 'numpy.ndarray' objects}
17166	33.691	0.002	35.661	0.002	decomp.py:196(eigh)
17165	24.105	0.001	26.129	0.002	basic.py:253(inv)
51496	9.221	0.000	14.852	0.000	ps3_implementation.py:124(compute_kernel)
51496	2.913	0.000	2.913	0.000	{scipy.spatial._distance_wrap.cdist_euclidean_wrap}

Outline

- Cross Validation
  - Implementation
  - Efficient LOOCV
  - Profiling
- Receiver operating characteristic
- Classification

1. ROC curve illustrates the quality of a binary classifier
2. Illustrates trade-off between false positive and false negative rate

1. Given two distinct probability distributions dependent on some random variable
2. E.g. population with disease and without
3. p= true positive classification
4. n= true negative classification
5. p’= predicted positive classification
6. n’= predicted negative classification

	p	n
p’	True Positive	False Positive
n’	False Negative	True Negative

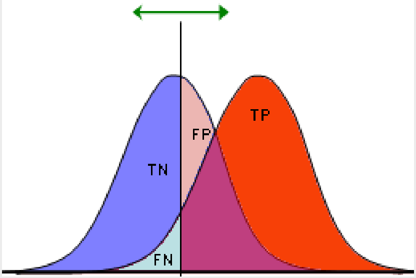


Figure: source: wikipedia.org

1. Classification depends on discrimination threshold
2. Classification quality is limited by the overlap of distributions
3. This classification performance can be shown with help of ROC-Curve
4. True positive Rate: $TPR = \frac{TP}{TP+FN} = \frac{TP}{P}$
5. False positive Rate: $FPR = \frac{FP}{FP+TN} = \frac{FP}{N}$

From exercise sheet we have:

1. $p(x|y=-1) \sim N(\mu = 0, \sigma^2 = 1)$
2. $p(x|y=-1) \sim N(\mu = 2, \sigma^2 = 1)$
3. $p(y=-1) = 0.5$
4. $p(y=1) = 0.5$

Implementation

empirical

```
for xs in x:
  dtn[i]=np.size(np.where(dn<=xs))
  dfn[i]=np.size(np.where(dp<=xs))
  dtp[i]=np.size(np.where(dp>=xs))
  dfp[i]=np.size(np.where(dn>=xs))
```

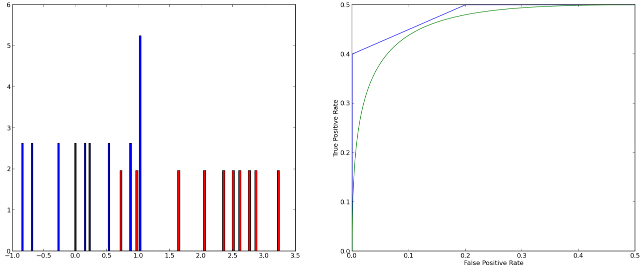


Figure: n=10

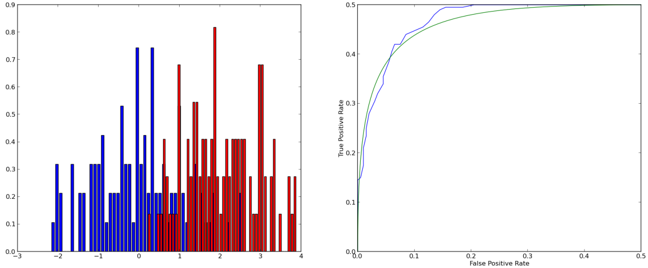

0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3
4
True Positive Rate
False Positive Rate
0.0
0.1
0.2
0.3
0.4
0.5

Figure: n=100

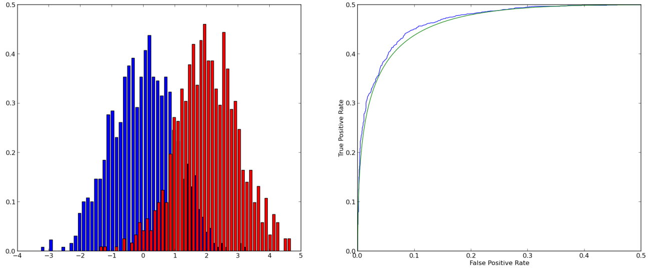


Figure: n=1000

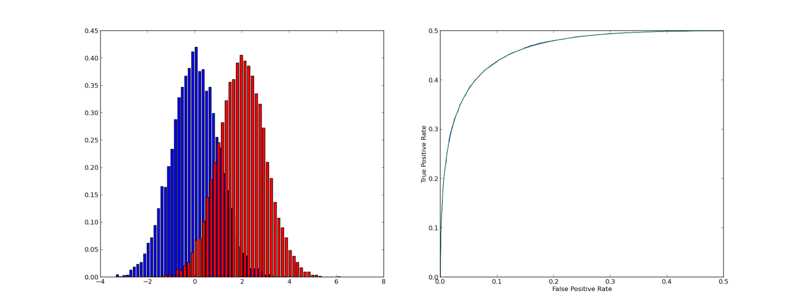


Figure: n=10000

Outline

Cross Validation
- Implementation
- Efficient LOOCV
- Profiling

Receiver operating characteristic

Classification

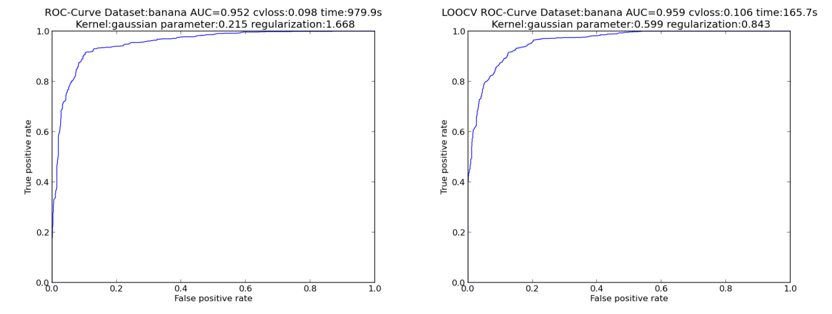
Assignment 4

- Classification of data sets `image`, `ringnorm`, `flare-solar`, `banana` and `diabetes` using `cv`
- Cross-validation: `nfolds = 10` and `nrepetitions = 5`
- Parameter ranges:

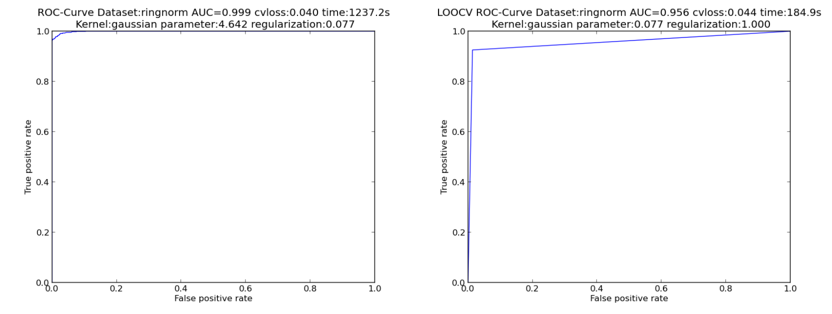
Kernel	Kernel parameter	Regularization
linear	[0]	`np.logspace(-2,2,10)`
polynomial	`np.arange(1,10)`	`np.logspace(-2,2,10)`
gaussian	`np.logspace(-2,2,10)`	`np.logspace(-2,2,10)`

- KRR with LOOCV $\Rightarrow$ regularization set to `[0]`

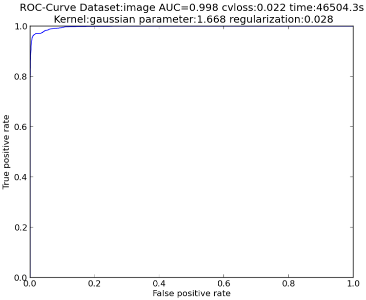
ROC-Curves banana

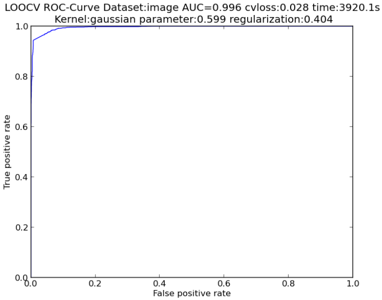


ROC-Curves `ringnorm`

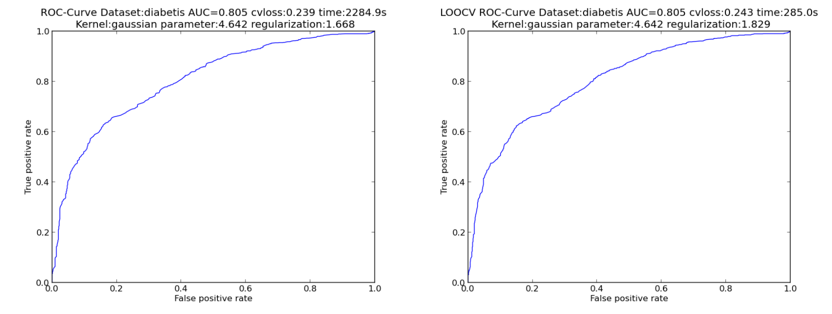


ROC-Curves `image`

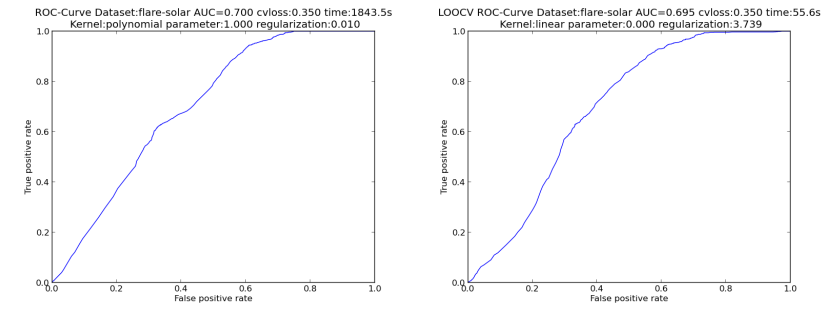




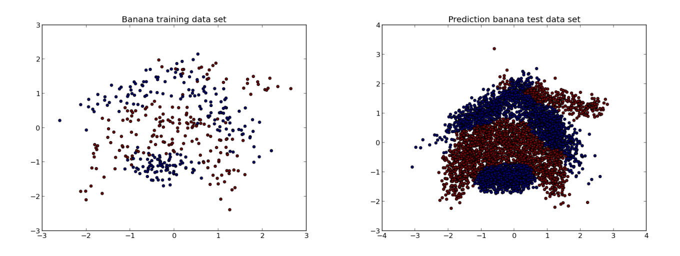
ROC-Curves `diabetes`

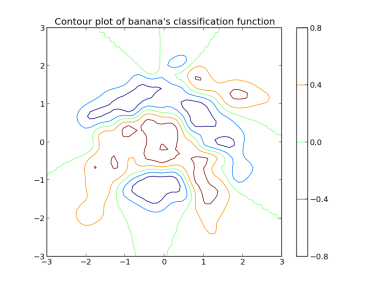


ROC-Curves `flare-solar`



banana in detail





Cross-validation results

Table: Normal `cv`

Data set	cv loss	AUC	Kernel	Parameter	Regularization
image	0.0225	0.998	Gaussian	1.6681	0.0278
ringnorm	0.0395	0.999	Gaussian	4.6416	0.0774
flare-solar	0.3499	0.7	polynomial	1	0.01
banana	0.0985	0.952	Gaussian	0.2154	1.6681
diabetes	0.2385	0.805	Gaussian	4.6416	1.6681

Table: LOOCV

Data set	cv loss	AUC	Kernel	Parameter	Regularization
image	0.0283	0.996	Gaussian	0.5995	0.4043
ringnorm	0.0475	0.956	Gaussian	0.0774	1.0
flare-solar	0.3505	0.695	linear		3.7391
banana	0.106	0.959	Gaussian	0.5995	0.8431
diabetes	0.2431	0.805	Gaussian	4.6416	1.8293

Accuracy comparison

- Reference loss taken from Machine Learning 2 lecture
- Normal `cv` slightly better than LOOCV

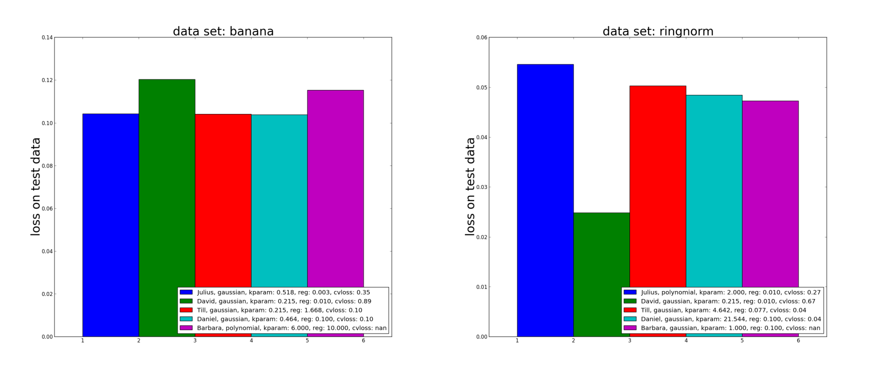
Data set	cv loss	cv loss LOOCV	reference loss
image	0.0225	0.0283	0.028
ringnorm	0.0395	0.0475	0.047
flare-solar	0.3499	0.3505	0.341
banana	0.0985	0.106	0.106
diabetes	0.2385	0.2431	0.232

Runtime comparison

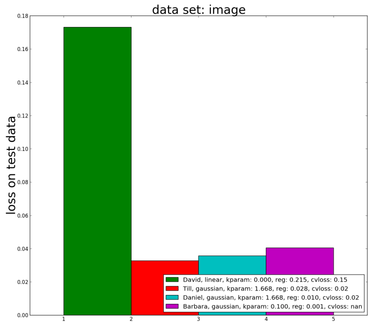
- Speed-up between $6 - 12$
- `flare-solar` data set is an outlier
- Linear kernel has fewer parameter values

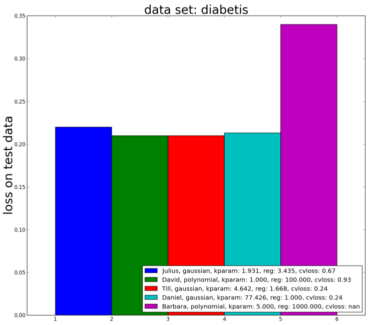
Data set	`cv` in s	`cv` with LOOCV in s	speed-up
image	46504	3920	11.9
ringnorm	1237	185	6.7
flare-solar	1843	56	33
banana	980	166	6
diabetes	2284	285	8

Comparison test data set



Comparison test data set contd.





Comparison test data set contd.