SVC classifier with 0% accuracy #9154

New issue

Closed **LBlnd** opened this issue on Jun 18, 2017 · 9 comments

LBlnd commented on Jun 18, 2017 • edited

Hello !

I'm using SVC from Sklearn to discriminate between different matrices.
The datas are 95 matrices of correlation, computed from IRM of patients with schizophrenia (50 matrices) and healty controls (45 matrices). They are pretty big (264*264), so I wasn't expecting perfect results, but 0% accuracy seems really low.

Code

Here is the code:

```
## Datas
#control_matrices: list of 45 matrices
#patient_matrices: list of 50 matrices

n_training = 25 #Number of matrices of control to train SVC (25 control and 25 patient)
indices = np.triu_indices(264,1) #Since the matrices are symetric, I just take the upper

perm_control = np.random.permutation(45) #Doing a permutation to take random matrices fo
contr_matrices = control_matrices[perm_control] #control_matrices is a list of matrices
perm_patient = np.random.permutation(50) #Same with the patient matrices
pat_matrices = patient_matrices[perm_patient]

x_control = [m[indices] for m in contr_matrices[:n_training]] #Data for training
x_patient = [m[indices] for m in pat_matrices[:n_training]]

test_control = [m[indices] for m in contr_matrices[n_training:]] #Data for test once the
test_patient = [m[indices] for m in pat_matrices[n_training:]]

X = np.concatenate((x_control, x_patient))
Y = np.asarray( n_training*["Control"] + n_training*["Patient"] )

## Training

clf = SVC()
clf.fit(X,Y)
```

Expected Results

Since the size of the data is huge compared to the number of matrices, I would have expected low results (something just a little bit better than 50%).

Actual Results

```
clf.score(np.concatenate((test_control, test_patient)), 20*['Control']+25*['Patient'])
```

```
0.0
```

The same happens whenever I run the code (so, with different permutations), and for `n_training` from 10 to 45. However the SVC does remember well the first matrices, for the training.

Try

I also tried this, with exactly the same results:

```
from sklearn.cross_validation import StratifiedKFold, cross_val_score
from nilearn import connectome

connectivity_coefs = connectome.sym_to_vec(matrices, ConnectivityMeasure)
```

Assignees

No one assigned

Labels

None yet

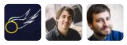
Projects

None yet

Milestone

No milestone

3 participants



```
cv = StratifiedKFold(Y, n_folds=3, shuffle=True)
svc = LinearSVC()

cv_scores = cross_val_score(svc, connectivity_coefs, Y, cv=cv, scoring='accuracy')

print('Score: %1.2f +- %1.2f' % (cv_scores.mean(), cv_scores.std()))


 Score: 0.00 +- 0.00
```

I completely fail to understand what is happening here.

Versions

```
Windows-8-6.2.9200
Python 3.4.1 |Continuum Analytics, Inc.| (default, May 19 2014, 13:02:30) [MSC v.1600
64 bit (AMD64)]
NumPy 1.9.1
SciPy 0.15.1
Scikit-Learn 0.15.2
```

rth commented on Jun 19, 2017 • edited — Member

With random `patient_matrices` and `control_matrices` I get a ~0.5 accuracy score using your first code sample, as expected. Did you normalize `X` to [0, 1] (or similar) for SVM? What is `clf.score(X, Y)` ? Have you tried a linear model first (e.g. `LogisticRegression`) to check that your code is working?

LBInd commented on Jun 19, 2017 • edited — Author

I didn't normalize anything; I'm using matrices with values in [-1,1].
`clf.score(X, Y)` gives the percentage of predictions (for `X`) that match the correct value (`Y`).
I have tried `LogisticRegression` , `SVC` and `LinearSVC` ; all of them give me the same result (0% accuracy on new matrices).

I also tried with simpler data : matrices `[0]` for Control and `[1]` for Patients. The SVC worked perfectly, so I suspect it has something to do with the size of the matrices I use (huge size and few samples).

rth commented on Jun 19, 2017 — Member

OK, thanks for the details! Yes, but what the actual value of `clf.score(X, Y)` : `1.0` ? Well your code works as expected with random data,

```
patient_matrices = np.random.rand(50, 264, 264)
control_matrices = np.random.rand(50, 264, 264)
```

I think. So the question is really, is there anything in the actual data you are using that could explain such difference in the results. Alternatively, trying to reproduce this issue with some synthetic data (or sharing your data) would also help understanding what is happening.

LBInd commented on Jun 19, 2017 — Author

`clf.score(X,Y)` always return `0.0` when `X` is a list of new matrices (not the ones used for training). Well, the datas are matrices computed from IRM from an open dataset, but I have a lot of preprocessing. I'll share the previous code if that helps.

Thanks for your answers!

rth commented on Jun 19, 2017 — Member

`y` *are* the matrices used for training) ..

> Well, the datas are matrices computed from IRM from an open dataset, but I have a lot of preprocessing. I'll share the previous code if that helps.

Or maybe just the result of your preprocessing (those 2 matrices `(50, 256, 256)` ..)?

LBlnd commented on Jun 19, 2017 Author

Here is how to get the same matrices that I use:

```
from nilearn import datasets
from nilearn import input_data
from nilearn.connectome import ConnectivityMeasure
import numpy as np
from sklearn.svm import SVC, LinearSVC
from sklearn.cross_validation import StratifiedKFold, cross_val_score
from nilearn import connectome


## Atlas for the parcellation and Dataset

power = datasets.fetch_coords_power_2011()
coords = np.vstack((power.rois['x'], power.rois['y'], power.rois['z'])).T
datas = datasets.fetch_cobre(n_subjects=None, verbose=0)

spheres_masker = input_data.NiftiSpheresMasker(
                    seeds=coords, smoothing_fwhm=4, radius=5.,
                    detrend=True, standardize=True,
                    high_pass=0.01, t_r=2, verbose=0)


## Extracting useful IRM

list_time_series = []
i = 0
for fmri_filenames, confounds_file in zip(datas.func, datas.confounds): #Might take a fe
    print("Sujet %s" % i)
    if i != 38 and i != 41: #Subjects removed from the study
        conf = np.genfromtxt(confounds_file)
        conf = np.delete(conf, obj = 16, axis = 1) #Remove Global Signal
        conf = np.delete(conf, obj = 0, axis = 0) #Remove labels
        scrub = [i for i in range(150) if conf[i,7]==1]
        conf = np.delete(conf, obj = 7, axis = 1) #Remove Scrub
        if len(scrub) < 90: #Keep at least 60 non scrub
            time_series = spheres_masker.fit_transform(fmri_filenames, confounds=conf)
            time_series = np.delete(time_series, obj = scrub, axis = 0) #Remove scrub
            list_time_series.append(time_series)
        else:
            list_time_series.append([])
    else:
        list_time_series.append([])
    i+=1


## Computing correlation matrices

N = len(datas.phenotypic)
control_subjects = []
patient_subjects = []
for i in range(N):
    t = list_time_series[i]
    if type(t) != list :
        subject = datas.phenotypic[i]
        if str(subject[4])=='b\'Control\'':
            control_subjects.append(t)
        else:
            patient_subjects.append(t)
control_subjects = np.asarray(control_subjects)
patient_subjects = np.asarray(patient_subjects)

connect_measure = ConnectivityMeasure(kind='tangent')
control_matrices=connect_measure.fit_transform(control_subjects)
patient_matrices=connect_measure.fit_transform(patient_subjects)

matrices = np.concatenate((control_matrices, patient_matrices))
```

```
cv = StratifiedKFold(Y, n_folds=3, shuffle=True)
svc = LinearSVC()
connectivity_coefs = connectome.sym_to_vec(matrices, ConnectivityMeasure)
cv_scores = cross_val_score(svc, connectivity_coefs, Y, cv=cv, scoring='accuracy')

print('Score: %1.2f +- %1.2f' % (cv_scores.mean(), cv_scores.std()))
```

LBInd commented on Jun 19, 2017 • edited Author

Ooooh ok I finally get it, sorry I took me so long ^^'
`clf.score(X,Y)` is `1.0`.

With `matrices = np.random.rand(95,264,264)`, I get `Score: 0.58 +- 0.03`.

The matrices I used: control_matrices.zip

LBInd commented on Jun 19, 2017 Author

And: patient_matrices.zip

amueller commented on Jun 19, 2017 Member

The issue tracker is not for usage questions, use stackoverflow please.

amueller closed this on Jun 19, 2017