Amazon.com - Employee Access Challenge

Wednesday, May 29, 2013 $5,000 • 1,691 teams Finished Wednesday, July 31, 2013

Starter code in python with scikit-learn (AUC .885)

Start Watching

Hi everyone,

Since it seems we have a lot of Coursera students with us (of which I'm also a fervent user), I

Paul Duan

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

Since it seems we have a lot of Coursera students with us (of which I'm also a fervent user), I wanted to share some simple starter code in Python to help those who are new at machine learning. You might also want to read **Foxtrot's excellent post about beating the benchmark using Vowpal Wabbit.** In contrast, this code uses Python with scikit-learn, and aims at giving a base on which to expand for those who want to be a little bit more hands-on or have more flexibility in the algorithm design.

It provides an example on how to design a simple algorithm, including performing some pre-processing, training a logistic regression classifier on the data, and assessing its performance through cross-validation. I also added some comments to point at where to go next. The script assumes you have train.csv and test.csv in a folder named data in the same location as the classifier.py file.

The strategy itself is essentially the same as Foxtrot's, ie. training a linear model on the original data with nothing else changed excepted for the one-hot encoding. In this case, the model used is a regularized logistic regression. This will net you an AUC score of .885 -- have fun!

Edit: forgot to remove an import in the original file. Use classifier_corrected.py instead.

2 Attachments —

classifier.py (3.58 KB)
classifier_corrected.py (3.53 KB)

Thanked by **Sophia N.** , **sck_t_ths** , **davyzhu** , **Commander Data** , **Martin Beyer** , and 56 others

#1 / Posted 4 months ago / Edited 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Hi, is the "io_helper" library in the code a standard python module? If not, from where can I get it? My Python 2.7 on Windows does not recognize it.

sck_t_ths

Posts **4**
Joined **14 Mar '12**
Email User

#2 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Paul Duan

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

sck_t_ths wrote:

> Hi, is the "io_helper" library in the code a standard python module? If not, from where can I get it? My Python 2.7 on Windows does not recognize it.

Woops, I forgot to remove this line. I updated my original post with the correct file but you can also simply delete the line containing the import -- the load_data and save_results methods were originally in a separate file but I later decided to simplify things and just put everything in the same file. In addition, the script assumes your file structure is as follows:

    classifier.py

    data/

    -- train.csv

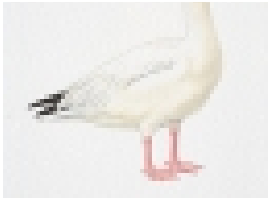
    -- test.csv

Thanked by **sck_t_ths**

#3 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Hi Paul, thank you for the starter code! In the code, you wrote "if you want to perform feature



davyzhu

Posts **5**
Joined **26 May '13**
Email User

selection / hyperparameter optimization,...", could you give us some suggestions on it?

#4 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Wen K Luo

Posts **47**
Thanks **28**
Joined **15 Apr '13**
Email User

First off, Thanks for the starter code. I personally avoid stuff like this. But I can see this is really helpful to a lot of our users, especially people who are interested in data science and have already a moderate understanding of it in the first place. I am however of the opinion, you should avoid stuff like this if you're a complete beginner like me.

When you start with a code that gets you within the top 25%. You're kind of pidgeon-holed to more specific things that work and stuck to more incremental improvements which to be frank you don't really understand for a while. This could really limit your understanding of how to implement or the data at hand. Personally, I'm treating this competition as a learning exercise and having went from .6 -> .75 -> .85 -> .87 by testing out various things. Creating personal validation models, trying out various methods, creating bad code, creating weird algorithm, and testing out so many things that did not work (you learn as much when you fail after all); I feel as if I learn more that way rather than using Foxtrot's or Paul's code. For the most part, I felt like this was effective in other competitions. Ex. I grinded out to first place in ~2 weeks for a Yelp business rating competition (still ongoing) at one point and I learned so much about every thing.

Then again, to play devil's advocate against myself. I was thinking of taking the datascience course on coursera, but the requirements wanted formal programming (I had 1 undergrad course in Java for non-majors where the prof asked if we think machines will take over on a test), and a whole hoard of other requirements that I did not meet. So maybe I'm the only

real beginner here. lol.

Just my 2 cents if it's about learning for beginners, not really for people who are probably moderate understanding.

Thanked by **davyzhu** and **Artem Yankov**

#5 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**
Email User

Thanks a lot Paul, starter code like this is perfect for newbies looking to get a grasp on how they should be writing code

#6 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Paul Duan

Wen: I definitely agree with you in that you learn the most by trying approaches that don't work, understanding why, then iterating. However, the main point of the starter code is to provide an example on how to design an algorithm from start to finish using scikit-learn, not just an out-of-the box solution (like if I told you to let Vowpal Wabbit do the whole work). The code itself is fairly generic and is designed to be modified; all you have to do is to change the preprocessing part or the line calling the LogisticRegression class if you want to try another method. Since I already provide a simple way to compute a cross-validation AUC score, you

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

will then be able to directly see how your changes affect performance, then react accordingly.

Also, in this specific case the fact the solution gives a rather high AUC right off the bat is just due to the characteristics of the dataset -- the method itself is fairly straightforward. I also don't think progress is necessarily incremental, since sometimes I'll try a technique that actually gives me worse performance than what I had. Sometimes it's worse but better in some cases, and realizing that can help you in tern improve your previous method. Having a good score already doesn't prevent you from experimenting.

davyzhu: Sure. In your case, I would advise to not spend too much effort on feature selection right now (although you are definitely welcome to try). The reason is that at this stage, your features are essentially based on the 8 columns you were provided at the beginning and each correspond to a specific manager, position, department and so on, so they pretty much all do provide some useful information (if you delete a feature corresponding to manager 23421, for example, you will most likely be worse off when trying to predict the outcome of a resource request made by someone with the same manager).

Sklearn provides a module called feature_selection, which implements quite a few classes and methods you can use for feature selection. Be careful with recursive feature selection algorithms though, since they're not appropriate for very high dimensional data.

For hyperparameter optimization, the simplest way to do it is just by modifying the parameters by hand, then looking at how this affects the cross validation score. This can work if you only have one parameter to optimize, but when you have several the usual way to do it is by trying a lot of different possibilities by grid search. Scikit-learn also provides classes for that: **http://scikit-learn.org/0.13/modules/grid_search.html** (in particular, look at GridSearchCV)

Also, the reason I put this comment inside the cross-validation loop is that feature selection and hyperparameter optimization basically fit the model itself to the data, so you'll end up getting a CV score that is way too optimistic otherwise. As a rule of thumb, any method that use some information from the labels should be done inside a CV loop -- in the case of the code I provided, you want to fit your GridSearchCV object to (X_train, y_train) and not directly (X, y).

Thanked by **davyzhu** and **GGuyZ**

#7 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**
Email User

Paul are you personally using Python or R for this competition?

#8 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Benoit Plante

Posts **135**
Thanks **26**
Joined **22 Jan '12**
Email User

Thanks for the code!

I find it very interesting that a simple logistic regression with regularisation could be that good.

```
encoder = preprocessing.OneHotEncoder()
    encoder.fit(np.vstack((X, X_test)))
    X = encoder.transform(X)  # Returns a sparse matrix (see numpy.sparse)
    X_test = encoder.transform(X_test)
```

This is very cool. It gives me interest to learn python :P

This is very cool. It gives me interest to learn python !!

#9 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Miroslaw Horbal

Rank **67th**
Posts **114**
Thanks **169**
Joined **27 Oct '12**

Email User

Paul Duan wrote:

> For hyperparameter optimization, the simplest way to do it is just by modifying the parameters by hand, then looking at how this affects the cross validation score. This can work if you only have one parameter to optimize, but when you have several the usual way to do it is by trying a lot of different possibilities by grid search. Scikit-learn also provides classes for that: **http://scikit-learn.org/0.13/modules/grid_search.html** (in particular, look at GridSearchCV)

Nice thing about Scikit-learn and grid search is you can define the scoring function to be AUC. This will let you directly optimize for high AUC instead of the default (which is usually accuracy for classification)

#10 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Paul Duan

Nitai Dean wrote:

> Paul are you personally using Python or R for this competition?

Python, but I like to do my data exploration in R since I find dataframes so convenient. I hear Pandas provides similar functionality in Python but I haven't tried it yet.

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

Pandas provides similar functionality in Python but I haven't tried it yet.

#11 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**
Email User

Yeah I'm forcing myself to use python instead of R this time around to learn new things (sigh...)

Pandas seems to be the way to go for data science so I'm using it. Seems nice... has a describe() function which is similar to R's summary(), which is dead useful IMO

#12 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

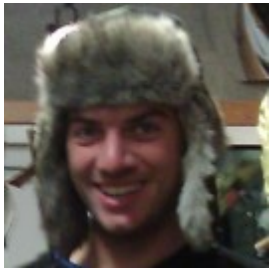
Miroslaw Horbal

Rank **67th**
Posts **114**
Thanks **169**

Pandas DataFrame is very nice, from my experiance it provides much of the functionality that you get out of R. Plus pandas.read_csv()... I love that function :D

Joined **27 Oct '12**
Email User

#13 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**



Daniel Velkov

Rank **85th**
Posts **13**
Thanks **4**
Joined **24 Jul '12**
Email User

Question for the people using sklearn. Is there any off the shelf way to generate feature pairs or triples? Something as easy as -q and --cubic in vw.

Thanked by **sanilg01**

#14 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

densonsmith

Posts **66**
Thanks **14**
Joined **11 Oct '12**
Email User

Thanks for the sample code! I've been using scikit for about a year now but I started out as a total Python newbie so your code is much cleaner than my junk.

My two cents on the competition and people posting stuff like this. On the one hand I already had figured out a way to get about 0.86 on my own, so I guess this is making it much harder to finish in the top 25%. On the other hand, getting that last 0.05 is what the competition is all about.

This problem is deceptive. It is easy to get 0.85 if you just know how to use a machine learning/statistical package reasonably well. However, you have to really know what you are doing to improve that result.

The hardest part about learning scikit is that there are too few examples out there and the

examples there are are too simple. Releasing code like this will make it much easier for people to learn the basics.

#15 / Posted 4 months ago Reply / Quote / Thank / Flag / Email User

densonsmith

Posts **66**
Thanks **14**
Joined **11 Oct '12**
Email User

Daniel Velkov wrote:

> Question for the people using sklearn. Is there any off the shelf way to generate feature pairs or triples? Something as easy as -q and --cubic in vw.

I'm not sure exactly what you are trying to do but you can probably do it with lists in python:

http://effbot.org/zone/python-list.htm

Notice that you can have a list of lists or other data types and iterate over it in various ways, including selecting various pairs, 3's, 4's etc.

As someone already mentioned, GridSearchCV is the built in scikit way to optimize parameters.

#16 / Posted 4 months ago Reply / Quote / Thank / Flag / Email User

Miroslaw Horbal

Daniel Velkov wrote:

> Question for the people using sklearn. Is there any off the shelf way to generate feature pairs or triples? Something as easy as -q and --cubic in vw.

It doesn't look like it does but can easily accomplish this with a nested for-loop, tuples, and a

Rank **67th**
Posts **114**
Thanks **169**
Joined **27 Oct '12**
Email User

hash function with a few lines of code. The downside with this method is that if your hash function produces negative values - such as python's hash() - sklearn's OneHotEncoder will not work since it only accepts non-negative integers (which is silly IMO).

I've attached the code I am using to accomplish this.

1 Attachment —

data_utils.py (1.61 KB)

Thanked by **Paul Duan**

#17 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Wen K Luo

Posts **47**
Thanks **28**
Joined **15 Apr '13**
Email User

Paul Duan wrote:

> **Wen:** I definitely agree with you in that you learn the most by trying approaches that don't work, understanding why, then iterating. However, the main point of the starter code is to provide an example on how to design an algorithm from start to finish using scikit-learn, not just an out-of-the box solution (like if I told you to let Vowpal Wabbit do the whole work). The code itself is fairly generic and is designed to be modified; all you have to do is to change the preprocessing part or the line calling the LogisticRegression class if you want to try another method. Since I already provide a simple way to compute a cross-validation AUC score, you will then be able to directly see how your changes affect performance, then react accordingly.
>
> Also, in this specific case the fact the solution gives a rather high AUC right off the bat is just due to the characteristics of the dataset -- the method itself is fairly straightforward. I also don't think progress is necessarily incremental, since sometimes I'll try a technique that actually gives me worse performance than what I had. Sometimes it's worse but better in some cases, and realizing that can help you in tern improve your previous method. Having a good score already doesn't prevent you from experimenting.

3 points I'll make and these are just my opinions, not facts:

1) Sorry if I caused any confusion, but rather than directing my comment at you, I was hoping to direct my comment at absolute beginners who would just take your code without

completely understanding why it works. Something fairly simple and generic that can be easily modified like you say can still be confusing to someone who just started. If someone looks at your code and doesn't feel like it's self-explanatory when it comes to further improving and still need further guidance to optimize, then they should spend some more time exploring the data before using your code as a framework.

2) I say high score because it got you within the top 25% as it is (remember high is always relative). When you start off with a low-ish score, you tend to experiment more since you recognize that you would probably need a major change in thought because you KNOW other people are using approaches that are better than yours. As in using something completely different from what you're doing now. That broadens your exploration and experiences. Whereas a high-ish score limits you to trying to work in a certain framework. **This isn't to say your framework forces people to be pidgeon-holed, but that beginners when they get such a high score, they pidgeon-hole themselves and are less willing to experiment with out the box thinking**.

3) I say incremental because improvements to scores that are at the top 25% are much smaller than someone in the 30th percentile. And incremental also includes going down, not sure what you meant by incremental only going up... sorry if I mislead you. Usually these improvements tend to be optimizations rather than a whole new broad approach.

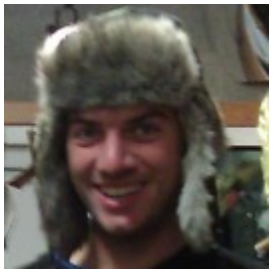
To reiterate, your code is great. I just think if you're a absolute beginner, it's better if you make a few personal forays before trying out a framework to work around. Thanks again for the code. I don't want to take anything away from you. I just want to send a warning to absolute beginners if their sole purpose is to learn. That maybe they should try a few things out themselves before using your code.

tl;dr: Not really aimed at you, but at certain people who might just take your code and fall into certain traps.

Thanked by **Satya**

#18 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User



Daniel Velkov

Rank **85th**
Posts **13**
Thanks **4**
Joined **24 Jul '12**
Email User

Miroslaw Horbal wrote:

> **Daniel Velkov wrote:**
>
> > Question for the people using sklearn. Is there any off the shelf way to generate feature pairs or triples? Something as easy as -q and --cubic in vw.
>
> It doesn't look like it does but can easily accomplish this with a nested for-loop, tuples, and a hash function with a few lines of code. The downside with this method is that if your hash function produces negative values - such as python's hash() - sklearn's OneHotEncoder will not work since it only accepts non-negative integers (which is silly IMO).
>
> I've attached the code I am using to accomplish this.

Yeah, similar to what I figured.

#19 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

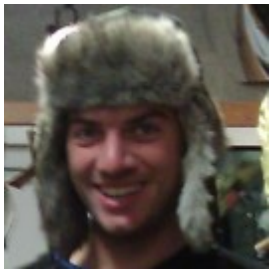
I'm new to sklearn. Your code outputs a float b/w 0 and 1. To convert this to a 0 or 1 we can probably threshold. Is there a best/good way to determine a threshold value in sklearn?

MB1

Posts **3**
Joined **5 Jun '13**
Email User

#20 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**



Daniel Velkov

Rank **85th**
Posts **13**
Thanks **4**
Joined **24 Jul '12**
Email User

Mike Basilyan wrote:

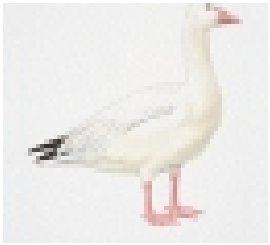
> I'm new to sklearn. Your code outputs a float b/w 0 and 1. To convert this to a 0 or 1 we can probably threshold. Is there a best/good way to determine a threshold value in sklearn?

There is **http://scikit-learn.org/stable/modules/preprocessing.html#feature-binarization** which does what you need. But for this competition your submissions doesn't have to be only in {0,1}.

#21 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Daniel Velkov wrote:



MB1

Posts **3**
Joined **5 Jun '13**
Email User

> But for this competition your submissions doesn't have to be only in {0,1}.

Hmm... the sample submission is in {0,1} and I couldn't find anything to contradict that.

#22 / Posted 4 months ago — Reply / Quote / Thank / Flag / Email User

Paul Duan

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

Mike Basilyan wrote:

> I'm new to sklearn. Your code outputs a float b/w 0 and 1. To convert this to a 0 or 1 we can probably threshold. Is there a best/good way to determine a threshold value in sklearn?

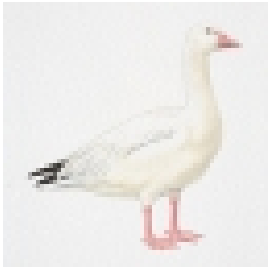
The classic value for the threshold is .5 -- when fitting a logistic regression, this is the value that minimizes the loss on the training set. In sklearn, you can also simply use the predict method instead of predict_proba, which would give you exactly that (y = 1 if y >= .5 else 0).

Moving it up or down would allow you to improve precision at the expense of recall and vice-versa, but in this case you don't need to (and shouldn't) output a binary decision.

The AUC metric is a ranking metric, so what counts is that you scored denied requests as less likely to be approved than approved ones.

Thanked by **davyzhu**

#23 / Posted 4 months ago — Reply / Quote / Thank / Flag / Email User



MB1

Posts **3**
Joined **5 Jun '13**
Email User

I just looked closer at the sample submission on the site:

```
id,ACTION
1,1
2,0.2
3,1
4,0
5,2
```

I see that they have 1, 0, 0.2 and 2 for id=5. So that means we can return 0..1 for a given id. The question is what's the deal with id=5? What does 2 mean? Doubly allowed? Or is that just a typo?

#24 / Posted 4 months ago — Reply / Quote / Thank / Flag / Email User

Paul Duan

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

No, it just means that in our ACTION needs not be a probability -- all the AUC metric cares about is the relative ranking of the predictions, so (as Daniel said) your outcome variable doesn't have to be between 0 and 1. However, if you do want to output "probabilities" (as is the case in the provided code) this will of course work too.

#25 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

William Cukierski
Kaggle Admin

Posts **682**
Thanks **392**
Joined **13 Oct '10**
Email User

k

MB1 wrote:

> I just looked closer at the sample submission on the site:
>
> ```
> id,ACTION
> 1,1
> 2,0.2
> 3,1
> 4,0
> 5,2
> ```
>
> I see that they have 1, 0, 0.2 and 2 for id=5. So that means we can return 0..1 for a given id.
> The question is what's the deal with id=5? What does 2 mean? Doubly allowed? Or is that just a typo?

See here: **https://www.kaggle.com/c/amazon-employee-access-challenge/forums/t/4705/do-we-predict-probabilities**

#26 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Can someone explain what encoding does in the starter code? If you remove encoding the AUC drops to .5.

tylerbrabham

Posts **1**
Joined **9 Jun '13**
Email User

#27 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Miroslaw Horbal

Rank **67th**
Posts **114**
Thanks **169**
Joined **27 Oct '12**
Email User

It is a One Hot Encoder for categorical data.

Say you had a feature with 3 categories (a, b, c), you could map each of those to a binary variable

a -> 1 0 0

b -> 0 1 0

c -> 0 0 1

That's what the encoder does.

#28 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Paul Duan

Scikit-learn is rather well-documented -- you can find the doc for this specific class here:

http://scikit-learn.org/dev/modules/generated/sklearn.preprocessing.OneHotEncoder.html

Basically, the one-hot encoding method transforms your categorical variables into dummy (0 or 1) variables. In this code I am using a logistic regression which can't use categorical variables. It will try to find a direct linear relationship between say, the manager ID and the

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

variables. It will try to find a direct linear relationship between say, the manager ID and the outcome, which will of course fail since manager IDs are meaningless by themselves. This is why it'll give an AUC of .5 (which is equivalent to the random benchmark).

Thanked by **sanilg01**

#29 / Posted 4 months ago — Reply / Quote / Thank / Flag / Email User

Benoit Plante

Posts **135**
Thanks **26**
Joined **22 Jan '12**
Email User

This one-hot encoding code is great. I do not use python, and I would code in R a similar thing by hand.

Thanks, I learned something!

#30 / Posted 4 months ago — Reply / Quote / Thank / Flag / Email User

Paul Duan

Rank **1st**
Posts **69**
Thanks **197**

Benoit Plante wrote:

> This one-hot encoding code is great. I do not use python, and I would code in R a similar thing by hand.
>
> Thanks, I learned something!

There is in package in R that lets you expand categorical features into dummy variables:
http://cran.r-project.org/web/packages/dummies/

Joined **3 Jun '12**
Email User

This might save you some time.

Thanked by **Benoit Plante**

#31 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Dylan Friedmann

Posts **50**
Thanks **47**
Joined **15 Nov '12**
Email User

Paul Duan wrote:

> **Benoit Plante wrote:**
>
> > This one-hot encoding code is great. I do not use python, and I would code in R a similar thing by hand.
> >
> > Thanks, I learned something!
>
> There is in package in R that lets you expand categorical features into dummy variables: **http://cran.r-project.org/web/packages/dummies/**
>
> This might save you some time.

also available in the Matrix package is the sparse.model.matrix function, which creates dummies of any categorical variable and stores the result as a sparse matrix

```
library(Matrix)
amazon_train = sparse.model.matrix(~. - 1, data = amazon_train) # -1 = no intercept column

amazon_train = amazon_train[,-1] #remove split of ACTION0 , ACTION1
```

Thanked by **Benoit Plante**

#32 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Benoit Plante

Posts **135**
Thanks **26**
Joined **22 Jan '12**
Email User

Dylan Friedmann wrote:

> also available in the Matrix package is the sparse.model.matrix function, which creates dummies of any categorical variable and stores the result as a sparse matrix
>
> library(Matrix)
> amazon_train = sparse.model.matrix(~. - 1, data = amazon_train) # -1 = no intercept column
>
> amazon_train = amazon_train[,-1] #remove split of ACTION0 , ACTION1

Thanks.

And may I ask how to encode the test set using the same dummy variables as the training?

The package help file is not clear.

#33 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Dylan Friedmann

Posts **50**
Thanks **47**
Joined **15 Nov '12**

Benoit Plante wrote:

> **Dylan Friedmann wrote:**
>
> > also available in the Matrix package is the sparse.model.matrix function, which creates dummies of any categorical variable and stores the result as a sparse matrix
> >
> > library(Matrix)

Email User

> amazon_train = sparse.model.matrix(~. - 1, data = amazon_train) # -1 = no intercept column
>
> amazon_train = amazon_train[,-1] #remove split of ACTION0 , ACTION1

Thanks.

And may I ask how to encode the test set using the same dummy variables as the training?

The package help file is not clear.

This may be a hacky way, but whenever I have a problem with getting dimensions equal, I typically turn to the ?match function. In this case:

https://gist.github.com/dylanjf/5774446

To be honest, I'm not sure if filtering this way hurts the end results, but it did allow me to fit my model fit since the dimensions were equal.

#34 / Posted 4 months ago — Reply / Quote / Thank / Flag / Email User

Paul Duan

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**

Benoit Plante wrote:

> **Dylan Friedmann wrote:**
>
> > also available in the Matrix package is the sparse.model.matrix function, which creates dummies of any categorical variable and stores the result as a sparse matrix
> >
> > library(Matrix)
> > amazon_train = sparse.model.matrix(~. - 1, data = amazon_train) # -1 = no

Email User

> intercept column
>
> amazon_train = amazon_train[,-1] #remove split of ACTION0 , ACTION1

Thanks.

And may I ask how to encode the test set using the same dummy variables as the training?

The package help file is not clear.

An easier way to do this is to encode both at the same time, then retrieving the individual dataframes afterwards:

> > foo = dummy.data.frame(rbind(Xtrain, Xtest), dummy.classes="ALL")
> >
> > Xtrain = foo[1:nrow(Xtrain),]
> >
> > Xtest = foo[-nrow(Xtrain):-1,]
> >
> > rm(foo)

Thanked by **Dylan Friedmann** and **Benoit Plante**

#35 / Posted 4 months ago — Reply / Quote / Thank / Flag / Email User

Elliot Dawson

Paul Duan wrote:

> **Benoit Plante wrote:**
>
> > **Dylan Friedmann wrote:**
> >
> > > also available in the Matrix package is the sparse.model.matrix function.

Posts **6**
Joined **25 Dec '11**
Email User

> which creates dummies of any categorical variable and stores the result as a sparse matrix
>
> library(Matrix)
> amazon_train = sparse.model.matrix(~. - 1, data = amazon_train) # -1 = no intercept column
>
> amazon_train = amazon_train[,-1] #remove split of ACTION0 , ACTION1

Thanks.

And may I ask how to encode the test set using the same dummy variables as the training?

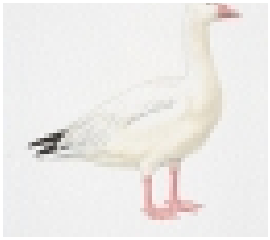
The package help file is not clear.

An easier way to do this is to encode both at the same time, then retrieving the individual dataframes afterwards:

> > foo = dummy.data.frame(rbind(Xtrain, Xtest), dummy.classes="ALL")
> >
> > Xtrain = foo[1:nrow(Xtrain),]
> >
> > Xtest = foo[-nrow(Xtrain):-1,]
> >
> > rm(foo)

Did anybody else run out of memory attempting this? My machine quickly ran out of its possible 16249Mb when I tried.

#36 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**



densonsmith

Posts **66**
Thanks **14**
Joined **11 Oct '12**
Email User

Elliot Dawson wrote:

> **Paul Duan wrote:**
>
> > **Benoit Plante wrote:**
> >
> > > **Dylan Friedmann wrote:**
> > >
> > > > also available in the Matrix package is the sparse.model.matrix function, which creates dummies of any categorical variable and stores the result as a sparse matrix
> > > >
> > > > library(Matrix)
> > > > amazon_train = sparse.model.matrix(~. - 1, data = amazon_train) # -1 = no intercept column
> > > >
> > > > amazon_train = amazon_train[,-1] #remove split of ACTION0 , ACTION1
> > >
> > > Thanks.
> > >
> > > And may I ask how to encode the test set using the same dummy variables as the training?
> > >
> > > The package help file is not clear.
> >
> > An easier way to do this is to encode both at the same time, then retrieving the individual dataframes afterwards:
> >
> > > foo = dummy.data.frame(rbind(Xtrain, Xtest), dummy.classes="ALL")
> > >
> > > Xtrain = foo[1:nrow(Xtrain),]
> > >
> > > Xtest = foo[-nrow(Xtrain):-1,]

> xtest = foo[-nrow(xtrain),-1,]

> rm(foo)

Did anybody else run out of memory attempting this? My machine quickly ran out of its possible 16249Mb when I tried.

I have a macbook pro with 16gb of ram and I have about 4gb to spare AFTER the encoding is complete and I've deleted all unnecessary variables. During encoding my memory is full. Mac OS is leaner than Windows I think but you should still be able to make it work by carefully cleaning up unneeded arrays, lists etc.

#37 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Paul Duan

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

Elliot Dawson wrote:

Did anybody else run out of memory attempting this? My machine quickly ran out of its possible 16249Mb when I tried.

You can also try replacing the first line in the example I gave with the sparse.model.matrix() function proposed by Dylan instead of dummy.data.frame(). Since by definition the encoded data will be overwhelmingly filled with 0s using a sparse representation should be much more memory efficient.

Thanked by **Elliot Dawson**

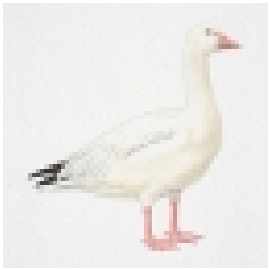
#38 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

as a side question: how come that your python sklearn example works with sparse matrices?

I tried using random forests and they only accept dense matrices.

Maybe just some predictors, like logistic, can handle the sparsity

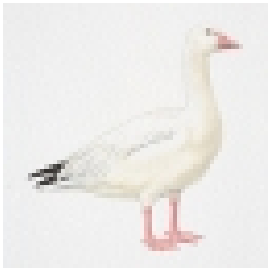


Dieselboy

Rank **83rd**
Posts **20**
Thanks **4**
Joined **8 Jan '13**
Email User

#39 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User



densonsmith

Posts **66**
Thanks **14**
Joined **11 Oct '12**
Email User

Dieselboy wrote:

> as a side question: how come that your python sklearn example works with sparse matrices?
>
> I tried using random forests and they only accept dense matrices.
>
> Maybe just some predictors, like logistic, can handle the sparsity

I found a thread somewhere that the scikit development team is considering creating a version of random forests that accept sparse matrices as input. However, it not a high priority and it seems to me that it would not be a simple thing to accomplish.

Getting the model to accept a sparse matrix would be not problem but if it had to just convert it to dense internally it wouldn't accomplish much.

#40 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Linda Otero

Posts **2**
Joined **8 Jun '13**
Email User

Thanks very much!, I'm trying to run the code, and I'm getting this error:

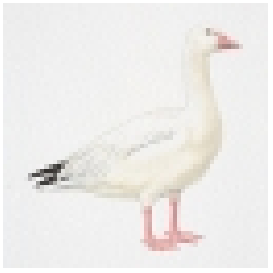
AttributeError: 'Module' object has no attribute 'OneHotEncoder'

any ideas what am I doing wrong?

thanks

#41 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User



rc3623

Posts **1**
Joined **8 Jun '13**
Email User

Linda -

Got the same error and it appears that I was using an older version of scikit-learn (0.10.1) downloaded from Ubuntu which does not have the OneHotEncoder module. Follow the instructions in (**http://scikit-learn.org/dev/install.html**) to download and build the latest version (0.13.1) using 'sudo pip install -U scikit-learn' and you should be good!

#42 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

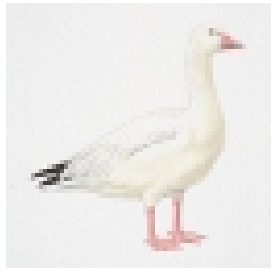
Linda Otero

Posts **2**
Joined **8 Jun '13**
Email User

Great **rc3623** thanks very much! I'll try that :)

#43 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**



Robert Graves

Posts **1**
Joined **16 May '13**
Email User

Daniel, check out **itertools.combinations** (standard python library, no download needed) for generating tuples.

#44 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Paul Duan wrote:

Dylan Friedmann

Posts **50**
Thanks **47**
Joined **15 Nov '12**
Email User

Elliot Dawson wrote:

> Did anybody else run out of memory attempting this? My machine quickly ran out of its possible 16249Mb when I tried.

You can also try replacing the first line in the example I gave with the sparse.model.matrix() function proposed by Dylan instead of dummy.data.frame(). Since by definition the encoded data will be overwhelmingly filled with 0s using a sparse representation should be much more memory efficient.

Still, there seems to be a big limitation in the types of ML algorithms that you can apply when you go about this way. Namely, in R I have only seen logistic regression be able to handle this format (which is stored as an S4 object... S3 objects are what are typically used in R). I'm probably just ignorant of a way to work around this... but it seems to hamstring a lot of typical creative techniques (feature creation, ensembling, etc). Might have to try something like octave to implement other methods... or I should just man up and learn python :)

#45 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**

Python aint perfect either... I'm trying to run libsvm on it now and in a fashion very reminiscent of R, it just crashes with too many features/rows

Email User

#46 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Miroslaw Horbal

Rank **67th**
Posts **114**
Thanks **169**
Joined **27 Oct '12**
Email User

Nitai Dean wrote:

> Python aint perfect either... I'm trying to run libsvm on it now and in a fashion very reminiscent of R, it just crashes with too many features/rows

Have you tried the **SVC class** from scikit learn? Also, if you're using a linear kernel I'd look into **LinearSVC** which scales much more gracefully to more features.

#47 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**
Email User

exactly that one... but with 30000 rows and 16000 columns (in sparse array, naturally), it still chokes up. I'm currently trying to reduce the feature space with randomized logistic regression, down to about 2000 columns - but even that it has a hard time swallowing.

any suggestions? it keeps telling me: "Warning: using -h 0 may be faster", but I'm not really sure what that means and how to make the change.

#48 / Posted 4 months ago Reply / Quote / Thank / Flag / Email User

Paul Duan

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

The -h parameter controls whether you use the shrinking heuristic or not. It's supposed to help speed up the computation but sometimes it doesn't. In your case I don't think it should make a significant difference if your problem is the memory usage. If you don't care about a nonlinear kernel you should use the SVM implementation in liblinear (in scikit-learn, call LinearSVC instead) which is much faster -- it runs in a couple seconds while using libsvm takes me about an hour (but it does work for me given enough time).

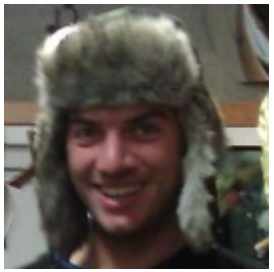
#49 / Posted 4 months ago Reply / Quote / Thank / Flag / Email User

Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**
Email User

I've found that a linear SVM does not offer much additional information over other linear models, which doesnt improve the stacking very much. the idea is to find as many diverse algorithms as possible, right?

#50 / Posted 4 months ago Reply / Quote / Thank / Flag / Email User



Daniel Velkov

Rank **85th**
Posts **13**
Thanks **4**
Joined **24 Jul '12**
Email User

And talking about stacking (or blending) models, is there a function which does that in sklearn?

#51 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

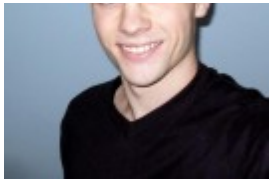
Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**
Email User

As far as i know, you can just implement it yourself using a simple learner like logistic regression as the meta learner that combines the predictions of multiple base learners on a hold-out set

#52 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Nitai Dean wrote:



Miroslaw Horbal

Rank **67th**
Posts **114**
Thanks **169**
Joined **27 Oct '12**
Email User

> I've found that a linear SVM does not offer much additional information over other linear models, which doesnt improve the stacking very much. the idea is to find as many diverse algorithms as possible, right?

As a suggestion, you should be able to achieve an AUC of .90 using a single linear model, feature selection and feature transformations. Model averaging is probably something you should focus on after you know that your individual models are performing at their limit.

#53 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

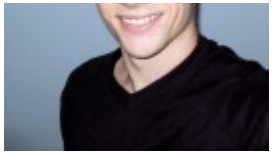
Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**
Email User

Is that so? Hmm I haven't had much luck in this department. Honestly, I don't have too many ideas of how feature selection or transformation would help a linear model improve over what it can do on the dummy encoded matrix. PCA/LSI didn't seem to help much... also, removing rare features (where one might guess that there are not enough samples to accurately discern a pattern) only made matters worse.

#54 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Nitai Dean wrote:



Miroslaw Horbal

Rank **67th**
Posts **114**
Thanks **169**
Joined **27 Oct '12**
Email User

> Is that so? Hmm I haven't had much luck in this department. Honestly, I don't have too many ideas of how feature selection or transformation would help a linear model improve over what it can do on the dummy encoded matrix. PCA/LSI didn't seem to help much... also, removing rare features (where one might guess that there are not enough samples to accurately discern a pattern) only made matters worse.

Two things that worked for me:
1. Transform the dataset into higher order features (ie, group them into pairs, triples, etc). I have some python code in this thread that will accomplish that. I transformed my dataset into a full combination of all features of degree 1, 2, 3.
2. Perform greedy forward selection on the feature set. That is: train a model on all individual features in your dataset and select the feature that gives you the highest CV score using K-fold CV. Next, train a model using the features selected in the previous iteration and iterate over all the remaining features. Repeat this process until your cross validation score begins to decline. Then use those selected features to train your full model.

The model I used for this was Scikit Learn's Logistic regression, and have gotten similar (but poorer) results using LinearSVC

EDIT: I'm also hitting my head against the wall with things to do with linear models. All other avenues for feature construction I have tried so far have netted poorer results. Model averaging didn't improve things by much either, I only got a boost of 0.00014 to my AUC score on the leaderboard.

#55 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Miroslaw: I'd be curious to know how long it takes for you to run your greedy feature

selection. I would expect generating all pairs and triples to result in a very high dimensional feature space?

Paul Duan

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

#56 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

That's actually pretty clever... I feel silly for not having thought of that. For some reason I forgot that adding higher order features can improve a linear model. I just assumed that the linear model was already taking that information into account, but theres no reason why that should be true.

Is there a python function that implements greedy forward selection or did you just write it yourself?

Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**
Email User

#57 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

jaberg

Posts **4**
Joined **15 Oct '12**
Email User

Hi Paul, thanks for uploading the starter code!

Would you approve of it being included in **SkData**? (**github**)

It is precisely the kind of thing that I would like to see more of in skdata: how to load data, a standard pre-processing a demo classifier.

- James

#58 / Posted 4 months ago **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Miroslaw Horbal

Rank **67th**
Posts **114**
Thanks **169**
Joined **27 Oct '12**
Email User

Paul Duan wrote:

> Miroslaw: I'd be curious to know how long it takes for you to run your greedy feature selection. I would expect generating all pairs and triples to result in a very high dimensional feature space?

The dimension is massive... for ALL features it's above 1 million... but with feature selection I get it around 100,000 (still massive). Thankfully sklearn's linear models handle sparse data so things still run very quickly (about 1s to train the model)

EDIT: I lied, with all features up to 3rd order it's in the hundreds of thousands of dimensions. I tried to go to 4th order to see the results and that was when the dimensionality went up into the millions.

As for greedy feature selection, it takes about 4 hours to run. But I didn't do any exact timings. I just let it run overnight and it'd be finished by the morning.

To note, the greedy selection is performed on the categorical features, not the One Hot

Encoded features (that would be insane). With all features up to degree 3 this results in 93 features so greedy selection is still tractable.

Nitai Dean wrote:

> That's actually pretty clever... I feel silly for not having thought of that. For some reason I forgot that adding higher order features can improve a linear model. I just assumed that the linear model was already taking that information into account, but theres no reason why that should be true.
>
> Is there a python function that implements greedy forward selection or did you just write it yourself?

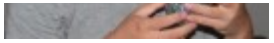
I wrote the code myself but perhaps something exists out there. It's essentially a while loop to maintain when the CV score begins to drop, a for loop to loop over the features, and an inner CV loop performing K-fold CV.

There is also another super-greedy variant of greedy selection where you only perform one pass over all individual features, sort the scores, and select the K best features. I haven't evaluated this technique yet so can't say anything about it's performance, but it will run much faster.

#59 / Posted 4 months ago / Edited 4 months ago **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Nitai Dean wrote:

> Is there a python function that implements greedy forward selection or did you just write it yourself?



Paul Duan

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

Check out the feature_selection package in sklearn, and in particular the **RFE** and **RFECV** documentation.

jaberg wrote:

> Hi Paul, thanks for uploading the starter code!
>
> Would you approve of it being included in **SkData**? (**github**)
>
> It is precisely the kind of thing that I would like to see more of in skdata: how to load
>
> data, a standard pre-processing a demo classifier.
>
> - James

Yes, absolutely. I'm happy to help.

Miroslaw Horbal wrote:

> There is also another super-greedy variant of greedy selection where you only perform one pass over all individual features, sort the scores, and select the K best features. I haven't evaluated this technique yet so can't say anything about it's performance, but it will run much faster.

A good compromise is to select (or eliminate) features in steps of 5, 10, etc. It's better than performing only one pass since you take into account interactions between features but it is still much faster to train.

By the way, did you see a significant improvement before and after using feature selection? In my case it was fairly minor so given the high computational cost I haven't been using it, but since I engineered my features differently I didn't need it that much.

#60 / Posted 4 months ago

Miroslaw Horbal

Rank **67th**
Posts **114**
Thanks **169**
Joined **27 Oct '12**
Email User

Paul Duan wrote:

> Check out the feature_selection package in sklearn, and in particular the **RFE** and **RFECV** documentation.

I find that these methods don't work as well for One Hot Encoded data, particularly when the dimensionality of the categorical variables is very high. This is primarily because the features that are eliminated are the individual binary features which should be interpreted as an entire group (IMO).

Paul Duan wrote:

> By the way, did you see a significant improvement before and after using feature selection? In my case it was fairly minor so given the high computational cost I haven't been using it, but since I engineered my features differently I didn't need it that much.

Depends on what you would call significant. My AUC went up from 0.89465 when I trained on all my features to 0.90491 after feature selection which boosted my spot on the leaderborads by 22 positions at the time. I also tried performing annealed greedy forward selection, but those gains were pretty insignificant increasing my AUC to 0.90537.

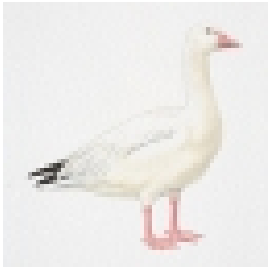
Paul Duan wrote:

> A good compromise is to select (or eliminate) features in steps of 5, 10, etc. It's better than performing only one pass since you take into account interactions between features but it is still much faster to train.

Another technique that gives comparable results while speeding up the forward selection process is to perform greedy forward selection for K steps while maintaining the order that the features fall in. Then at the Kth step drop the N worst performing feature(s) and repeat the process.

#61 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User



densonsmith

Posts **66**
Thanks **14**
Joined **11 Oct '12**
Email User

Dylan Friedmann wrote:

Paul Duan wrote:

Elliot Dawson wrote:

Did anybody else run out of memory attempting this? My machine quickly ran out of its possible 16249Mb when I tried.

You can also try replacing the first line in the example I gave with the sparse.model.matrix() function proposed by Dylan instead of dummy.data.frame(). Since by definition the encoded data will be overwhelmingly filled with 0s using a sparse representation should be much more memory efficient.

Still, there seems to be a big limitation in the types of ML algorithms that you can apply when you go about this way. Namely, in R I have only seen logistic regression be able to handle this format (which is stored as an S4 object... S3 objects are what are typically used in R). I'm probably just ignorant of a way to work around this... but it seems to hamstring a lot of typical creative techniques (feature creation, ensembling, etc). Might have to try something like octave to implement other methods... or I should just man up and learn python :)

> and learn python :)

Man up and learn python. Sorry, but python is more memory efficient. I'm trying throwing many different model types at the problem. So far, the more memory intensive is random

forest but it still runs on the encoded data with some small amount of memory to spare.

Random forest in scikit won't take sparse as input but there are versions of most of the other models that will. Note that some methods might take a sparse matrix as input but then internally convert it to dense. If that happens you can get a giant increase in demand for memory after calling the model.

#62 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Leustagos

Rank **3rd**
Posts **386**
Thanks **224**
Joined **22 Nov '11**
Email User

This forum is heating! Just by looking at the posted solutions here one could easily get to top 10%, just by ensenbling them all.

In this competition many people are working hard to do feature engineering, i never got good resultd by spending many time on feature engineering, its always blending for me.

And this competition is good for this, i don't even need to think about differents approachs, just pick some in this forum! :)

#63 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Leustagos wrote:

Benoit Plante

Posts **135**
Thanks **26**
Joined **22 Jan '12**
Email User

> This forum is heating! Just by looking at the posted solutions here one could easily get to top 10%, just by ensenbling them all.
>
> In this competition many people are working hard to do feature engineering, i never got good resultd by spending many time on feature engineering, its always blending for me.
>
> And this competition is good for this, i don't even need to think about differents approachs, just pick some in this forum! :)

Then at least you should share something ;)

#64 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Leustagos

Rank **3rd**
Posts **386**
Thanks **224**
Joined **22 Nov '11**
Email User

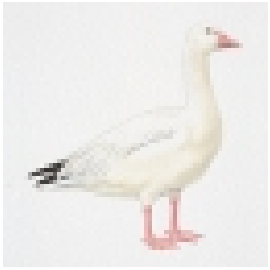
Benoit Plante wrote:

> **Leustagos wrote:**
>
> > This forum is heating! Just by looking at the posted solutions here one could easily get to top 10%, just by ensenbling them all.
> >
> > In this competition many people are working hard to do feature engineering, i never got good resultd by spending many time on feature engineering, its always blending for me.
> >
> > And this competition is good for this, i don't even need to think about differents approachs, just pick some in this forum! :)
>
> Then at least you should share something ;)

I usually do share my suboptmal approach. Not my hotest (that i do after the competition), but something useful. But right now, i'm more in a poistion to listen in this competition.

#65 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User



Gert

Posts **24**
Thanks **18**
Joined **19 Apr '11**
Email User

I can't figure out how to append a feature to the sparse matrix that is returned by the encoder in Paul's code.

Could someone share code how to append one of the features in X as a numeric predictor?

(this could be useful to test whether a linear trend in the category encodings improves the auc)

#66 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Miroslaw Horbal

Rank **67th**

Posts **114**
Thanks **169**

Gert wrote:

> I can't figure out how to append a feature to the sparse matrix that is returned by the encoder in Paul's code.
>
> Could someone share code how to append one of the features in X as a numeric predictor?
>
> (this could be useful to test whether a linear trend in the category encodings improves the auc)

Joined **27 Oct '12**
Email User

You'll want to use sparse.hstack from the scipy.sparse package.

Example:

```
from scipy import sparse

# Assuming X is your current data and F is the features you want to append
X = sparse.hstack((X, F))
```

EDIT:
Forgot to mention that sparse.hstack will convert the matrix to COO format. You'll want to convert it back to CSR to do computation on by running X.tocsr() after running hstack

Thanked by **Gert** and **pbnyc**

#67 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Paul Duan

Rank **1st**
Posts **69**
Thanks **197**
Joined **3 Jun '12**
Email User

Miroslaw Horbal wrote:

> Forgot to mention that sparse.hstack will convert the matrix to COO format. You'll want to convert it back to CSR to do computation on by running X.tocsr() after running hstack

You can also just specify 'csr' as the second argument to have it directly in CSR format (the matrices will tend to be very large, so the overhead of converting to a different format may not be insignificant):

http://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.hstack.html

#68 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Miroslaw Horbal

Rank **67th**
Posts **114**
Thanks **169**
Joined **27 Oct '12**
Email User

Paul Duan wrote:

> You can also just specify 'csr' as the second argument to have it directly in CSR format (the matrices will tend to be very large, so the overhead of converting to a different format may not be insignificant):
>
> **http://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.hstack.html**

It is expensive to change the sparsity structure of CSR and CSC matrices anyway. My guess would be that scipy does the conversion to COO under the hood when applying hstack or vstack and then converts back to your desired format if a format parameter is passed. A quick inspection of the **source code** confirms that guess. Conversions to COO from CSR/CSC are always linear time. That being said, it is cleaner to just pass the format with the hstack command instead of doing the conversion yourself.

#69 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

I am wondering: even if you convert the data sets into some sparse matrices then you are very limited in what you can do since, as they say here

http://bit.ly/1cJrcGG

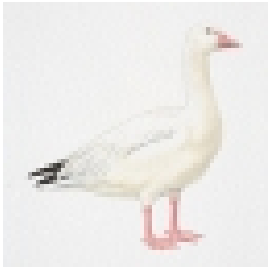
then you cannot use this data structure into a random forest or a GBM (at least in R)....so how can we move forward here?

larry77

Posts **26**
Thanks **2**
Joined **25 Sep '12**
Email User

#70 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User



densonsmith

Posts **66**
Thanks **14**
Joined **11 Oct '12**
Email User

larry77 wrote:

> I am wondering: even if you convert the data sets into some sparse matrices then you are very limited in what you can do since, as they say here
>
> **http://bit.ly/1cJrcGG**
>
> then you cannot use this data structure into a random forest or a GBM (at least in R)....so how can we move forward here?

I'm not sure about R but there are several models in scikit-learn that do take sparse matrices as input. Not random forests though.

#71 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Leustagos

which models in scikit-learn takes sparse matrixes as input?

Rank **3rd**
Posts **386**
Thanks **224**
Joined **22 Nov '11**
Email User

#72 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**
Email User

Logistic Regression, Naive Bayes, SGD

#73 / Posted 4 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

densonsmith

Posts **66**
Thanks **14**
Joined **11 Oct '12**

Leustagos wrote:

> which models in scikit-learn takes sparse matrixes as input?

Also, some versions of SVM...

Email User

#74 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Dylan Friedmann

Posts **50**
Thanks **47**
Joined **15 Nov '12**
Email User

densonsmith wrote:

> **larry77 wrote:**
>
> > I am wondering: even if you convert the data sets into some sparse matrices then you are very limited in what you can do since, as they say here
> >
> > **http://bit.ly/1cJrcGG**
> >
> > then you cannot use this data structure into a random forest or a GBM (at least in R)....so how can we move forward here?
>
> I'm not sure about R but there are several models in scikit-learn that do take sparse matrices as input. Not random forests though.

I believe only glmnet has been adapted to handle sparse matrices in R. To overcome this and open up some variety, I'm trying to pre-rank the features using Gini Index. There's plenty of other ranking metrics (chi sq, information gain, etc) that could help reduce dimensionality while maintaining interpretability.

Alternatively, R has an interesting package called irlba which can perform partial Singular Value Decomposition... so you could potentially explain a high level of variance but get it down to say 500 features or so, then take advantage of other models. The downside is since its still binary data, I don't think you're going to get anything significant from using other models, plus SVD is computationally expensive. Might be worth a shot though. **http://cran.r-project.org/web/packages/irlba/vignettes/irlba.pdf**

#75 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

Benoit Plante

Posts **135**
Thanks **26**
Joined **22 Jan '12**
Email User

The problem with PCA is that the variables need to be standardized first, but you can't standardize a sparse matrix...

#76 / Posted 4 months ago

Reply / Quote / Thank / Flag / Email User

rOcelot

Posts **17**
Thanks **2**
Joined **20 Sep '12**
Email User

larry77 wrote:

> I am wondering: even if you convert the data sets into some sparse matrices then you are very limited in what you can do since, as they say here
>
> **http://bit.ly/1cJrcGG**
>
> then you cannot use this data structure into a random forest or a GBM (at least in R)....so how can we move forward here?

I have also run into a similar issue with R. After some exploration I found that [glmnet] can handle sparse matrices. [glmnet] is a good package that can do Elastic Net variations (ridge and LASSO), It has some nice cross validation functions build into it. You can also run regular OLS models if you set alpha = 0 and use a penalty of 0 (lambda = 0). For this competition I think that elastic net would be especially useful because it is a good way of identifying important features. Unimportant features have their coefficient in the linear model driven to

important features. Unimportant features have their coefficient in the linear model driven to 0 as others take its place, I think that applying this to tuples would be a great way to move forward. I think ENET is especially good because of the high amounts of multicollinearity present in the dataset. (how could there not be ?).

Besides that there are a few packages listed in the link below that you may find useful. Several of which support sparse matrices.

http://cran.r-project.org/web/views/HighPerformanceComputing.html

#77 / Posted 4 months ago / Edited 4 months ago Reply / Quote / Thank / Flag / Email User

Leustagos

Rank **3rd**
Posts **386**
Thanks **224**
Joined **22 Nov '11**
Email User

Benoit Plante wrote:

> The problem with PCA is that the variables need to be standardized first, but you can't
>
> standardize a sparse matrix...

I don't think we need to to standardie a matrix full of binary columns. they are already in the same range...

Thanked by **mendrika**

#78 / Posted 4 months ago Reply / Quote / Thank / Flag / Email User

Benoit Plante

Posts **135**
Thanks **26**
Joined **22 Jan '12**
Email User

I am not a PCA expert, but I think I read somewhere that if the mean of each column is not zero, then the result will be incorrect.

edit: found where I saw this:

http://stats.stackexchange.com/questions/35185/dimensionality-reduction-svd-or-pca-on-a-large-sparse-matrix

#79 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Nitai Dean

Rank **30th**
Posts **122**
Thanks **66**
Joined **20 Jun '12**
Email User

If you dont center the matrix then PCA is invalid, you're right, but PCA without centering is actually a different technique called LSI (latent semantic indexing) which is effective for binary sparse matrices such as term document matrices.

In short, using PCA without centering has value

#80 / Posted 4 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Has anybody tried logistic regression in Matlab for this task? Seems that it works too far with such a sparse data...

adveboy

Posts **20**
Thanks **19**
Joined **5 Apr '11**
Email User

#81 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User

Artem Yankov

Posts **10**
Joined **13 Apr '13**
Email User

Can you guys explain how do you go about which metrics to use? Why AUC metrics was used in this example?

Also I've never seen examples with predict_proba yet (got very little experience though), as far as I understand it predicts probabilities. If I change it to just #predict score drops to 0.6, but why?

#82 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User

Miroslaw Horbal

Artem Yankov wrote:

> Can you guys explain how do you go about which metrics to use? Why AUC metrics was used in this example?
>
> Also I've never seen examples with predict_proba yet (got very little experience though),

Rank **67th**
Posts **114**
Thanks **169**
Joined **27 Oct '12**
Email User

> as far as I understand it predicts probabilities. If I change it to just #predict score drops to 0.6, but why?

AUC is the metric that is being used to score on this competition. So when running cross validation, you want to use that as the metric to give the best idea of how the model will generalized to the test set.

predict_proba will generate the probabilities instead of just making the class label decision like predict. In fact the results of predict are most likely just a selection of which probability is higher for the class labels.

Thanked by **Artem Yankov**

#83 / Posted 3 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Tiago Zortea

Posts **34**
Thanks **2**
Joined **14 Sep '12**
Email User

Does anybody know why glmnet Linear Regression in R performs so much worse than sklearn in python?

I mean, even if you don't perform any feature selection, in R I get around 0.87 AUC, while Python already scores almost 0.9.

I rather program in R than python, so this have been puzzling me for days.

#84 / Posted 3 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Silogram

Rank **56th**

Posts **4**
Joined **9 Dec '12**
Email User

Tiago Zortea wrote:

> Does anybody know why glmnet Linear Regression in R performs so much worse than sklearn in python?
>
> I mean, even if you don't perform any feature selection, in R I get around 0.87 AUC, while Python already scores almost 0.9.
>
> I rather program in R than python, so this have been puzzling me for days.

I was wondering the same thing. Even if you use the python greedy feature selection code, and then run those selected features through glmnet, the cv score stays at about .87. I don't know if it has to do with the regression algorithms being slightly different or possibly the R sparse matrix encoding doing something different from the python one hot encoder.

Also, why is logistic regression (and rf for that matter) so much slower than in python?

#85 / Posted 3 months ago — **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Tiago Zortea

Posts **34**
Thanks **2**
Joined **14 Sep '12**
Email User

Silogram wrote:

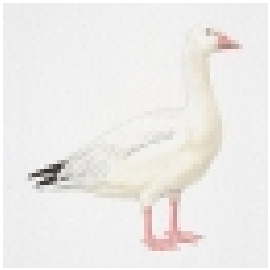
> I was wondering the same thing. Even if you use the python greedy feature selection code, and then run those selected features through glmnet, the cv score stays at about .87. I don't know if it has to do with the regression algorithms being slightly different or possibly the R sparse matrix encoding doing something different from the python one hot encoder.
>
> Also, why is logistic regression (and rf for that matter) so much slower than in python?

I'm glad someone else has the same problem, it meas I didn't messed up :)

I don't see how logistic regression can be implemented in a way to give different results, so I believe it is something about the regularization R performs

I'm still shocked by the massive difference. I usually see very different speeds across implementations, but not in results.

#86 / Posted 3 months ago Reply / Quote / Thank / Flag / Email User



densonsmith

Posts **66**
Thanks **14**
Joined **11 Oct '12**
Email User

Silogram wrote:

> **Tiago Zortea wrote:**
>
> > Does anybody know why glmnet Linear Regression in R performs so much worse than sklearn in python?
> >
> > I mean, even if you don't perform any feature selection, in R I get around 0.87 AUC, while Python already scores almost 0.9.
> >
> > I rather program in R than python, so this have been puzzling me for days.
>
> I was wondering the same thing. Even if you use the python greedy feature selection code, and then run those selected features through glmnet, the cv score stays at about .87. I don't know if it has to do with the regression algorithms being slightly different or possibly the R sparse matrix encoding doing something different from the python one hot encoder.
>
> Also, why is logistic regression (and rf for that matter) so much slower than in python?

I'm not sure about the difference in results but I the speed difference is probably because the python module is really calling compiled code written in C or FORTRAN for the number

crunching. I'm not sure what underlies R but if it is Java or poorly optimized C or FORTRAN that would be the answer.

#87 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User

EgyFirst

Posts **7**
Joined **6 Apr '12**
Email User

Paul Duan wrote:

> Hi everyone,
>
> Since it seems we have a lot of Coursera students with us (of which I'm also a fervent user), I wanted to share some simple starter code in Python to help those who are new at machine learning. You might also want to read **Foxtrot's excellent post about beating the benchmark using Vowpal Wabbit.** In contrast, this code uses Python with scikit-learn, and aims at giving a base on which to expand for those who want to be a little bit more hands-on or have more flexibility in the algorithm design. It provides an example on how to design a simple algorithm, including performing some pre-processing, training a logistic regression classifier on the data, and assessing its performance through cross-validation. I also added some comments to point at where to go next. The script assumes you have train.csv and test.csv in a folder named data in the same location as the classifier.py file. The strategy itself is essentially the same as Foxtrot's, ie. training a linear model on the original data with nothing else changed excepted for the one-hot encoding. In this case, the model used is a regularized logistic regression. This will net you an AUC score of .885 -- have fun! Edit: forgot to remove an import in the original file. Use classifier_corrected.py instead.

I receive this error

AttributeError: 'module' object has no attribute 'OneHotEncoder'

#88 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User

Piotr Kuchta

Rank **21st**
Posts **9**
Thanks **17**
Joined **2 May '13**
Email User

I guess you have an old version of scikit-learn installed on your system. I would strongly recommend installing the latest development version from github, perhaps in a virtualenv, so in case of problems you can easily try different versions (in a different virtualenv).

#89 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User

Dylan Friedmann

Posts **50**
Thanks **47**
Joined **15 Nov '12**
Email User

has anyone looked into Combined Regression Ranking? I was reading over some KDD winning papers that mention this method performs especially well when a ranking metric like AUC is to be optimized. If I understand correctly, the algorithm alternates between taking a SGD step and optimizing regression loss, with tuneable regularization parameter and SGD step probability.

There's an implementation in R I'll try to work, looks like it writes and reads its own sparse data format. here's the source code and the cran link: **https://code.google.com/p/sofia-ml/ http://cran.r-project.org/web/packages/RSofia/RSofia.pdf**

#90 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User

Jared Huling

Rank **37th**
Posts **61**
Thanks **26**
Joined **23 Sep '12**
Email User

This seems interesting. Either I'm using it incorrectly or it's unconscionably slow. It's been running for an hour on only 1000 observations from the training data

#91 / Posted 3 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

rOcelot

Posts **17**
Thanks **2**
Joined **20 Sep '12**
Email User

Tiago Zortea wrote:

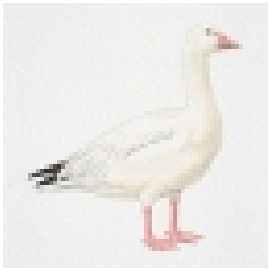
> **Silogram wrote:**
>
> > I was wondering the same thing. Even if you use the python greedy feature selection code, and then run those selected features through glmnet, the cv score stays at about .87. I don't know if it has to do with the regression algorithms being slightly different or possibly the R sparse matrix encoding doing something different from the python one hot encoder.
> >
> > Also, why is logistic regression (and rf for that matter) so much slower than in python?
>
> I'm glad someone else has the same problem, it meas I didn't messed up :)
>
> I don't see how logistic regression can be implemented in a way to give different results, so I believe it is something about the regularization R performs

> I'm still shocked by the massive difference. I usually see very different speeds across implementations, but not in results.

I was able to minimally improve upon Paul's starter model by using R only (.88840). I basically used sparse matrices, fit 6 different cv.glmnet models with varying regularization parameters and ensembled those predictions with the predictions from a naive random forest with numeric features. No feature selection was performed other than the regularization inherently performed by glmnet.

#92 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User



Davit

Posts **22**
Thanks **1**
Joined **11 May '12**
Email User

Sorry for being a little off topic but I need some tips from you for the current situation I am facing. I want to implement PCA on the OneHotEncodod sparse matrix. When using SVD function in Octave([U S] = svd(sigma)) it gives "memory exhausted" error. The problem with R's svd() and irlba() functions is that they don't return S diagonal matrix which we have in Octave's svd(). S diagonal matrix is essential for calculating k eigen vectors to retain. Without S it is hard to know what k value to choose to retain significant variance while reducing the dimensionality of data.

Secondly, is it reasonable to center the encoded matrix by subtracting the mean values of columns?

Any thoughts on this?

#93 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User

I wonder if it's possible to replace cross validation loop with using cross_val_score in this situation?

Artem Yankov

Posts **10**
Joined **13 Apr '13**
Email User

I'm trying to simplify the original code doing the following:

cvk = cv.ShuffleSplit(len(target), n_iter = 10, test_size=.20, indices=True, random_state=0)

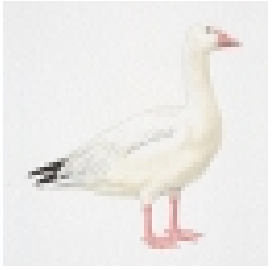
scores = cv.cross_val_score(classifier, train, target, cv=cvk, score_func=metrics.auc_score)

print scores.mean()

I think it should have done the same thing as a cross validation loop in code provided by Paul, but the mean of scores I'm getting is about .60. What am I doing wrong?

#94 / Posted 3 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**



Joe Regan

Posts **8**
Thanks **4**
Joined **1 Feb '13**
Email User

Tiago Zortea wrote:

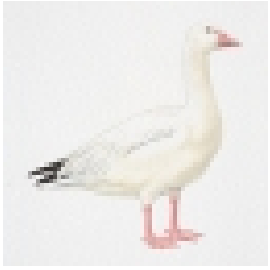
> Does anybody know why glmnet Linear Regression in R performs so much worse than sklearn in python?
>
> I mean, even if you don't perform any feature selection, in R I get around 0.87 AUC, while Python already scores almost 0.9.
>
> I rather program in R than python, so this have been puzzling me for days.

I have noticed that a lot of the top guys on the leaderboard wrote their code in Python, while the 2nd tier algorithms (0.88-0.9 AUC) seem to be heavily weighted towards users of R. OTOH, glmnet is unbelievably fast.

I'm surprised no one has discussed neural nets w/ ML estimation as a potential solution to this problem. Then again, in my experience, neural networks on a skewed classifier have produced some less-than-awesome results (converges to 1 or 0 for all points, not exactly useful).

#95 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User



Miner

Posts **9**
Thanks **3**
Joined **7 Apr '11**
Email User

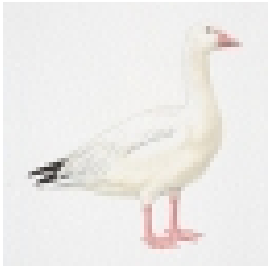
Curious if you had any luck with RSofia? I got very poor results trying out several of the algorithms.

#96 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User

JMA wrote:

> Curious if you had any luck with RSofia? I got very poor results trying out several of the algorithms.



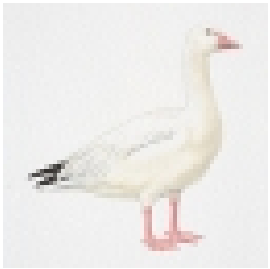
Joe Regan

Posts **8**
Thanks **4**
Joined **1 Feb '13**
Email User

I tried glmnet for my modeling, as well as some other methods (Naive Bayes, a small NN, etc). Of everything I've tried, glmnet has worked by far the best; unfortunately, it requires some real feature engineering to create interactions in the data. SVD could also accomplish this, to an extent, but that also removes the benefit of sparsity from the data.

#97 / Posted 3 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**



Davit

Posts **22**
Thanks **1**
Joined **11 May '12**
Email User

rOcelot wrote:

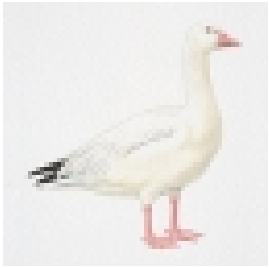
> I was able to minimally improve upon Paul's starter model by using R only (.88840). I basically used sparse matrices, fit 6 different cv.glmnet models with varying regularization parameters and ensembled those predictions with the predictions from a naive random forest with numeric features. No feature selection was performed other than the regularization inherently performed by glmnet.

Hello. Can you give a hint, please, on how to do the ensembling part?

#98 / Posted 3 months ago

Reply / **Quote** / **Thank** / **Flag** / **Email User**

Joe Regan wrote:



Jared Huling

Rank **37th**
Posts **61**
Thanks **26**
Joined **23 Sep '12**
Email User

Tiago Zortea wrote:

> Does anybody know why glmnet Linear Regression in R performs so much worse than sklearn in python?
>
> I mean, even if you don't perform any feature selection, in R I get around 0.87 AUC, while Python already scores almost 0.9.
>
> I rather program in R than python, so this have been puzzling me for days.

I have noticed that a lot of the top guys on the leaderboard wrote their code in Python, while the 2nd tier algorithms (0.88-0.9 AUC) seem to be heavily weighted towards users of R. OTOH, glmnet is unbelievably fast.

I'm surprised no one has discussed neural nets w/ ML estimation as a potential solution to this problem. Then again, in my experience, neural networks on a skewed classifier have produced some less-than-awesome results (converges to 1 or 0 for all points, not exactly useful).

glmnet is fast because it uses a quadratic approximation to the log-likelihood when fitting. for the same reason its performance is sub-optimal

#99 / Posted 3 months ago **Reply** / **Quote** / **Thank** / **Flag** / **Email User**

Gilberto Titericz Junior

Rank **19th**
Posts **43**
Thanks **50**
Joined **23 Aug '12**
Email User

Joe Regan wrote:

> **Tiago Zortea wrote:**
>
> > Does anybody know why glmnet Linear Regression in R performs so much worse than sklearn in python?
> >
> > I mean, even if you don't perform any feature selection, in R I get around 0.87 AUC, while Python already scores almost 0.9.
> >
> > I rather program in R than python, so this have been puzzling me for days.
>
> I have noticed that a lot of the top guys on the leaderboard wrote their code in Python, while the 2nd tier algorithms (0.88-0.9 AUC) seem to be heavily weighted towards users of R. OTOH, glmnet is unbelievably fast.
>
> I'm surprised no one has discussed neural nets w/ ML estimation as a potential solution to this problem. Then again, in my experience, neural networks on a skewed classifier have produced some less-than-awesome results (converges to 1 or 0 for all points, not exactly useful).

Well...I was able to score 0.90+ only using data preprocessing + glmnet in R

rOcelot

Posts **17**
Thanks **2**
Joined **20 Sep '12**
Email User

Gilberto Titericz Junior wrote:

> **Joe Regan wrote:**
>
> > **Tiago Zortea wrote:**
> >
> > > Does anybody know why glmnet Linear Regression in R performs so much worse than sklearn in python?
> > >
> > > I mean, even if you don't perform any feature selection, in R I get around 0.87 AUC, while Python already scores almost 0.9.
> > >
> > > I rather program in R than python, so this have been puzzling me for days.
> >
> > I have noticed that a lot of the top guys on the leaderboard wrote their code in Python, while the 2nd tier algorithms (0.88-0.9 AUC) seem to be heavily weighted towards users of R. OTOH, glmnet is unbelievably fast.
> >
> > I'm surprised no one has discussed neural nets w/ ML estimation as a potential solution to this problem. Then again, in my experience, neural networks on a skewed classifier have produced some less-than-awesome results (converges to 1 or 0 for all points, not exactly useful).
>
> Well...I was able to score 0.90+ only using data preprocessing + glmnet in R

.8840 averaging a couple glmnets with with a numeric RF.

Paul Duan uses R, and im almost positive that Xavier Conort uses R.

Restricted Boltzmann machines would be an interesting choice for this competition.

#101 / Posted 3 months ago / Edited 3 months ago

Reply / Quote / Thank / Flag / Email User

Dylan Friedmann

Posts **50**
Thanks **47**
Joined **15 Nov '12**
Email User

Davit wrote:

> **rOcelot wrote:**
>
> > I was able to minimally improve upon Paul's starter model by using R only (.88840). I basically used sparse matrices, fit 6 different cv.glmnet models with varying regularization parameters and ensembled those predictions with the predictions from a naive random forest with numeric features. No feature selection was performed other than the regularization inherently performed by glmnet.
>
> Hello. Can you give a hint, please, on how to do the ensembling part?

1) create a K fold CV of your training data

2) for i in 1 to K (repeat this for multiple models)

- label all folds not equal to [i] as fold_training, label fold [i] = fold_test

-fit any suitable model using your fold_training set

-predict on your fold_test set

3) order your predictions for each model from fold_test as one column against the dependant variable (ACTION) in the training data

4) train a model (i.e. linear regression) using ACTION against each of your model's predictions.

-record the coefficient estimates. these are your model weights.

5) fit original training data to each model, run predictions on your real test data, weigh each model's prediction by its weight, and sum up the total for each observation.

Thanked by **Davit**

#102 / Posted 3 months ago Reply / Quote / Thank / Flag / Email User

rOcelot

Posts **17**
Thanks **2**
Joined **20 Sep '12**
Email User

Davit wrote:

> **rOcelot wrote:**
>
> > I was able to minimally improve upon Paul's starter model by using R only (.88840). I basically used sparse matrices, fit 6 different cv.glmnet models with varying regularization parameters and ensembled those predictions with the predictions from a naive random forest with numeric features. No feature selection was performed other than the regularization inherently performed by glmnet.
>
> Hello. Can you give a hint, please, on how to do the ensembling part?

Dylan gave the correct procedure above, that is if you have the time to get fancy.

I just averaged my predictions :X

#103 / Posted 3 months ago

Reply / Quote / Thank / Flag / Email User

Reply

File Edit Insert View Format

Formats B I

p Words: 0

+ Add attachment(s) ...

Post Reply ☑ Email me when someone replies