Mike: Hey, everybody, welcome to the healthcare.ai hands-on machine learning broadcast. It's great to be back another week. I'm your host, Mike Mastanduno. And we me, remote from Denver, I have [inaudible 00:00:14]

Taylor: [inaudible 00:00:14] having me.

Mike: Yeah, definitely. Glad you can make it. Glad you all could make it. Please—, you know, we want this broadcast—as always, we want this broadcast to be interactive so feel free to log in to the YouTube channel, healthcare.ai, and interact with us in the chat window.

We actually do have the chat up and running this time so we'll do our best to respond to your questions and try to work, you know, your ideas into our dialogue here. We've got a great show this week. [inaudible 00:00:47].

Taylor: Yeah, so we're going to five into some good topics. We're going to touch on some recent things we saw on the news that we found interesting and thought you all might as well. We're going to a dive into kind of the meat of today's topic which is the fact that we added XGBoost for multiclass [inaudible 00:01:08] to healthcare.ai.

And then, also, just wanted to mention, before we even get going, that there are going to be some hands-on learning sessions at HAS 17 this year. There's kind of a whole of Health Catalyst University track for multi-class [inaudible 00:01:27] to healthcare.ai.

And then, also, just wanted to mention before we even get going, that there are going to be some hands-on—

Mike: -- catalyst and— what's the goal of HAS?

Taylor: Yeah. I think that the goal is to bring some of the best ideas together and to know that we can learn a ton from both folks that interact with Health Catalyst, folks that don't, folks that are just doing some real cutting edge stuff. And so, there's a lot of different sessions that are featured with people that have been out there doing the work, whether it's analytics or new clinical methods, that tie back to the analytics.

Mike: Cool.

So great.

So it’s a two-day healthcare analytics conference. And then we also have a special two-day Health Catalyst University machine learning track that’s separate from that, but the two days preceding the conference. So either of those, it would be great to see you in person if you decide to come out.

In the meantime, keep watching the broadcast. You can subscribe in the YouTube community. On the channel, you can join the Slack Channel and keep the conversation going throughout the week. We love the community interaction, so it’s great to see all the participation.

Taylor, shall we do “In the News”?

Taylor: For sure.

So I just was going to touch on, real quick, this pretty cool article that we found. It was a blog post called “A year of Google and Apple maps.” And I think it’s just a really cool use of visualization. And so, basically, it’s just showing how Google Maps has evolved over a given year compared to how Apple maps hasn't.

I think it’s really neat to see that Google has a pretty fast pace of advancement and adding new features. And we also noticed that a lot of their features are kind of more social in nature, providing additional information on businesses, parks and things like that. And less so on just the need for car navigation. Even a bit more biker-friendly which is pretty cool.

Mike: Yeah. I thought that was cool too. It’s a nice story being told, just through screen shots of the maps over time.

Taylor: Yup.

Yup, because you just kind of scrolling in through and seeing both the actual Google Maps as well as charts kind of helping highlight what's going on and what’s changing.

Mike: Mm-hmm.

Mm-hmm, great.

Taylor: Yeah, another cool thing that was in the news was Data Science Poll 2017. One of the kind of top finishers for that Kaggle competition which was related to predicting lung cancer, kind of did a write up on their solution and posted that in a blog form. And I thought it was really neat just kind of detailing the different steps they have to take from profiling the data, all the way through doing a pre-

prediction. So predicting malignancies, so that then they can ultimately predict cancer. And so, taking that two-step approach before going into their ensemble was pretty neat.

Mike: Yeah. And they do a nice job with presenting all the dataset, and presenting the methodology, and the use of deep learning architecture which they kind of post as a block diagram. So it's really cool to see that, especially applied to a healthcare dataset.

Taylor: Yup, for sure.

Yeah. That was really neat. And speaking of healthcare datasets, one last thing, real quick, was that there was a new dataset put out there. We'll get the link out on the Slack Channel, before too long, that has 40,000 de-identified critical care patients. And so, that was really, you know, kind of a really nice large dataset that has everything from demographics, lab data, medication data, and some other really neat features that would be nice to play with and start practicing some of our skills on.

One thing, you do have to kind of go through a little sign up step to get access to that open dataset. I think they just want to make sure they've taken all the precautions necessary and given credit to those that put in a lot of work.

Mike: Great.

Yeah, so if you're interested, sign up for the dataset. I wish I had known about this when I was scavenging for datasets. I feel like it's pretty hard to find, you know. Maybe, we'll have to go back and update our description for open datasets. But this definitely looks like a promising treasure trove of good stuff.

Taylor: For sure.

Mike: Great.

So thanks, Taylor, for the In the News.

I think there's a lot of good news this week. And we even left out some of the more exciting stuff such as the hotdog – not hotdog classifier that came out on the Apple App Store. But you can read about that on your own. I would highly encourage it. And it will help you decide where or not something is a hotdog or not.

But let’s get into the main event which is kind of talking about— what is it? We are talking about multiclass classification using XGBoost.

So we’ll kind of keep this at a high level and the idea is that we want to introduce all the different topics a little at a time and then go through it to get you actually using it.

So what is multiclass classification in healthcare? Taylor, could you give an example of a healthcare-related problem that might use multiclass? And maybe start even just by defining multiclass?

Taylor: Yeah.

So this is a quick definition for multiclass, I'll quickly mention just kind of the regular binary classification which is a lot of times we're doing, “Will this person be readmitted, yes or no? Will they have an extended length of stay? Will they pay their bill?” That kind of yes-no, 1 – 0 type of classification.

With the multiclassification, you're wanting to know which of these handful of categories or mini-categories is a patient or visit likely to fall into. And a good use case example of that would be assigning service lines in kind of a proactive way. So knowing that, based on all of these characteristics, which service line most appropriately reflects this client or this patient. And so, they're either treated correctly, billed correctly - although these types of things.

Mike: Mm-hmm. So that’s kind of like of like a less flashy use but definitely important for the financial aspects of the health system if they don’t have that information.

Taylor: Yup.

Mike: Another use case might be, going along the lines of those lung cancer preconditions, maybe you're trying to predict ”Is it benign? Is it a lung nodule? Is it a malignancy? What type of malignancy?” Those are each different categories which the algorithm would be able to assign a probability to.

And so, we added that feature to healthcare.ai because we had a lot of demand for different use cases that we made sense to have multiclass. And so, we ended up settling on this software package that's put out open source called XGBoost.

And XGBoost is a pretty interesting software package. It’s built on this algorithm called gradient boosted trees which is a tree ensemble, much like random forest. But where random forest is going to use a lot of independent but intelligent decision trees, XGBoost is going to use a lot of very stupid decision trees that work together to create something smart.

So in random forest, you build the best decision tree you can, independent of all the other decision trees and then you take a vote. With gradient-boosted trees or sometimes called gradient-boosted machines, every new tree you build is trying to augment the failures of the previous trees. So they're all kind of working together to solve the problem.

And so, the XGBoost is kind of just a packaging of that algorithm but it's a really nice one because it offers a lot in terms of scalability. It has GPU support. It's really good across cloud computing architectures. And a lot of people really like it, especially in Kaggle competitions. I think XGBoost has won like half of Kaggle competitions or something like that because it's just a really capable algorithm and package. So the package itself can do regression, classification and multiclass classification.

And so, our first step implementation has done only multiclass classification just because that's what we needed most and it made sense for XGBoost to do the heavy lifting there. But I wanted to kind of go through how we came to that conclusion. And in order to do that, I wrote a blog post which you can find at the healthcare.ai/blog website.

And I kind of just talk about the process of adding a new feature to healthcare.ai. So we have two packages. We have R and Python. So which one do we want to start with? We decided it would make sense to start just with R because Python is undergoing a fairly large re-architecture right now.

Then there's a lot of different ways to do multiclass classification - kind of, any of the tree-based algorithms or you could even do multivariate progressions. But we ended up settling on XGBoost just because it had a lot of niceties around scaling. And we thought we'd eventually like to have that algorithm in the package anways so it made sense to do that just for R, especially because the random forest package is in R are not really set up for multiclass which is an unfortunate shortcoming.

Then we kind of of had to decide how much functionality we wanted to give the user. One of the issues with XGBoost is that it has a lot of parameters to tune. And so it can be tiring, you have to tweak tall the parameters. So which ones can we sweep under the rug? And which ones do we really have to have the user able to play with?

And then we kind of considered steps for development, how do we want to develop the algorithm? Do we want to do it in one huge chunk with all the functionality? Or did we want to roll it out a little bit at a time? And so, we decided to roll it out a little bit at a time using version control and the GitHub that we've been talking about the past few weeks.

And so, if users want to get into it right now, they can go to GitHub and get the latest version of the code, which we'll show you how to do in this broadcast, and then use the functionality as it comes out. And if you don't want to, you can just wait until all the functionalities have been released and we've found all the bugs, Then it will be a new release on CRAN and you can install using install.packages.

So with that, I guess, we are up to "How do you get it?" So if you head over to our GitHub repo or our main Read the Docs page, with the main documentation about installation, there is a line here called "Install the bleeding edge version for folks providing contributions." You don't have to provide a contribution. You just have to recognize that that's the working development end of the package, so it might have some bugs. It might be changing a little bit faster than the rest of it. And the documentation might lag a little bit behind the functionality but that is the place to get the most recent code that we're using.

And so, right down there, you can see, there's a few lines of code that you can type into RStudio. And those will get you the most recent version directly from GitHub. So why don't we head over to RStudio? Let's head over to RStudio and check out what that looks like.

Taylor, were there any questions? Or do you have anything to add to that section?

Taylor: No. I think this is just really exciting with a lot of good use cases. I know that some concerns I've heard expressed that you kind of addressed were related to scalability with random forests on its own being known as a much more scalable algorithm. So wondering about multiclass classification using XGBoost if that's as scalable. But it sounds like, the way it was designed, it's accounted for a lot of those things.

Mike: Yeah. That's great.

And it looks like somebody's wondering what Kaggle is. And I'm glad that the chat has been responding to you.

If you're looking to get into data science , you know, we've said this in previous broadcasts but I truly believe that doing well in Kaggle competitions and getting hands-on experience is the best way to learning the field and learning the techniques. I think MOOCs**,** the online courses can only take you so far. Eventually, you've got to get the boots on the ground and start working. So, Kaggle is a great way to work with people, collaborate, see their solutions, what they did differently than you, and you might end up with an awesome blog post like the lung cancer guys did, Team Deep Breath, so.

Taylor: Yup, for sure.

Even diving into that new critical care dataset, using some of the new functionality from healthcare.ai would be another great option so.

Mike: Yeah. That's great.

Okay. So, from this point on, we're going to be working in RStudio. And if you didn't set your resolution high enough, if the screen looks grainy, you can do that on YouTube. There's a gear in the lower right corner. You can make sure you're streaming a higher definition and hopefully this text looks big enough for you.

So to start, as always, we're going to load the library. If I can type— trying to type and talk. You think I'd get better at it but I don't know if I really am.

So now we have library. And just like all the other— [inaudible 00:16:26] here.

Taylor: Here come the hands-on part.

Mike: Yup.

Taylor: This is going to be fun.

Mike: Watch out, guys.

So here is the main health documentation for healthcare.ai which we get just by typing "?healthcare.ai".

And so, in this development, we have links to lasso development and random forest development. And since this XGBoost stuff is brand new, I haven't updated the main documentation yet. I want to wait until it's fledged out a little bit. But the file structure is the same. So if we do XGBoost, there is a development and a deployment version. So let's take a look at what the development doc looks like.

Taylor: And notice, you have to load the library first before you can get the docs or use any of the functions from the package.

Mm-hmm. Mm-hmm. And so, I think— I tried to write this as much as like the other algorithms as I could. So the health files should look familiar. The examples should look familiar.

This particular example is going to use a dataset from the University of California Irvine's machine learning repository. And it's a dermatology dataset that you can download. I've put a link to it right there in the documentation. And then I also put a link to the XGBoost parameters that you can fiddle with in the documentation as well.

And so, why don't we grab this example. Actually, I'm going to go and grab the deployment example because it does the same thing as the development example but it goes one step further. So, let's load deployment. Again, it looks the same. And head down to the example. And I'll just copy and paste that into a script so that we can run it and see what we're doing.

So if I make a new R script and put that in there.

So here we have an example csv dataset which is going to be that dermatology dataset I was talking about that comes with the package now, if you were to update. So, it knows where to find it.

The first step is to just load the data. And we will do that. So, I can do CTRL+Alt+B to run up to where my cursor is. And so, you can see it loaded the data. The data frame is these nondescript variable names X1 through X34. Most of them are characters. I think these are assigned by dermatologists, describing the patient's, you know, the lesion or whatever the description is and the feature.

And one of the reasons I've provided the link— oh, I should add the link to this doc. But in the development documentation, the link will tell you what all the features are and that'll help so. And then there's a target column which has the classes, so class 2, class 1, class 3 associated with each row.

So to develop the model, we're going to develop and save the model. And the saving is automatic. This, again, looks pretty similar to all the other algorithms where we're initializing a new develop, supervise model, params. And then we are adding our data that we want to train on. We're going to use the type multiclass instead of classification regression. Everything else is the same.

We now, can add this list xgb_params. And we can put anything we want. I think, currently, it has to be this list here but you can change these values. And eventually we'll upgrade it so that you can add any parameter on the whole parameters list.

But again, that's kind of just a consideration of how we're rolling out the feature. We want to be able to roll it out a bit at a time. We want to give you the access to it as soon as possible, even if it's not completely done, with the idea that we can always circle back and improve it.

So, then we run the XGBoost Development and the run. And so, I'll do that right now. And I'll just run everything up to my cursor. And we can see kind of what it outputs for us.

So it starts by telling us what parameters we're using for our model training. And that's good because maybe we forgot what we were using. And then we start to run it.

And so, it's going to do the automatic train test splitting. It'll train on 80% of the data, evaluate on 20%. And since it's multiclass classification, the metrics for evaluating it are a little different, the easiest way to start is with a confusion matrix. So we can see that on the reference we have class five and prediction class five.

Okay. So we've got-- the diagonal of this matrix is basically, "Where did we get right? Where did the reference agree with the prediction?" And then, as you go down, you can start to look at where you might have gotten some wrong.

So it turns out this dataset is pretty easy to predict. So, here, we had a reference class four that got a prediction of 2. So that was one case where it was wrong. But then it also gives you— so it gives you the overall accuracy. And we've poopooed on accuracy a lot in the past but in this case, it's actually a good one because you are just interested in knowing how many times you got the class right and you don't really care about the distribution. But if you do, you get the normal statistics, sensitivity and specificity, precision and recall for each of the classes versus all the ones. So you do get that deep analysis tool in training.

So since this was the develop method, it's going to save that model automatically. And we can go over and look at my ugly files. There you see the— oh wait, that's not it. Where is it? Let's go to my working directory. Oh, that is my working directory. Well, I don't know where it went but— oh, there it is. R model probability XGB with a timestamp from just a minute ago. So that is our model.

Let’s see what that looks like to load it off in the deploy. So you’ll notice up here. This is a little different from the previous examples. I actually took 20 rows off of the dataframe that came from that csv file just so that we could have 20 rows to do a prediction in the deploy step.

Taylor: Those are 20 rows that your model hasn’t seen yet which is good for applying to.

Mike: Right.

So we really have the data divided into three parts, right? We have the 80% training. We have the 20% testing. And then we have those 20 rows that I've reserved, pretending that they're kind of new data in the wild or something like that.

So if we were to run the deploy section, which is right here. We're going to initialize a new supervised model of deployment. It’s going to be type multiclass, same as before. And then we’ll do the deployment and then use this getOutDf to get the output ready for your writing to csv, or something, or SQL.

Taylor: And Mike, real quickly, we’ve got a question in the chat asking about how the multiclass model is different from just creating six or however many different classification models?

Mike: Yes. That’s a great question.

So creating six or however many different classification models, it's one way to do multiclass. It’s kind of a hacky way to do it because you're kind of just saying like one versus all for every class. Whereas in this case, it's actually training one model to learn the patterns of all the classes at once. So you can imagine that that way of doing it one versus all with different models would be fine if you only had like three or four classes to predict. But we’ve been working on this service line project where we're trying to predict service lines and there’s almost 50 of them. So it just doesn’t seem scalable to use individual classifiers for every class versus a using a multiclass that’ll kind of do it all in one fell swoop, especially when XGBoost is so great and so much fun to implement.

Taylor: Yeah. And the question also mentioned just picking the highest probability from whichever of those separate classification models. But, essentially, that is what the multiclass classification is doing, right? As you kind of showed on your output, it’s still showing that there was a probability that it may have been one of those other classes, but there was a higher probability that it was the class that it chose.

Mike: Yeah, exactly.

So, if I do a–

Let me just head out. Let me just do this for a better output. So, here, this is what we might write to the SQL table or to our csv. It has the patient ID, and then it’s got the highest probability for that patient ID, and the class associated with that probability. Some of these cases, like this one here, it’s a 0.98 probability. It's probably that class.

Taylor: Yeah.

Mike: But this one is 0.85. It’s still pretty likely but it’s not guaranteed. So when you included the second most likely class, which in this case was a 0.11 and the third most probable class as well. And so, when the number of classes goes up, sometimes those probabilities start to go down, but you can-use that as a way to kind of see the confidence you’d have in a given prediction.

And then I actually wanted to run one other bit of code which is this. Here, we also gave a method in this class to return just the raw predictions. So let me do that. Raw.

Taylor: And also, one other question about data preparation—

Mike: Mm-hmm.

Taylor: For the model that you were just showing, did you just kind of pull that straight from the UC dataset that you mentioned or—

Mike: Yup.

Taylor: --did you have to get things ready to be a classification dataset?

Mike: Nope. I just pulled that right from UC website. I changed the class numbers. They have these as numbers but I changed them to the numbers spelled out, just to make sure the functionality was working properly. And I think that's fine.

But the only difference in data prep for this type of model versus a binary model is, in the binary model, you have to make sure your target, or your predicted column, or your target column has either yes's or no's. "Yes, this patient got sepsis. No, this patient did not get sepsis." You have to put it to different categories. So you put moderates sepsis, high sepsis, or no sepsis - that kind of thing.

Taylor: Sure.

Mike: And then it'll take care of all the rest for you.

So I tried to do as best a job I could to play along with the existing architecture so that it would be very little extra learning or burden on the user to implement this and get going on your data as quick as you could.

The last thing I wanted to show was this. You can get these raw predictions which just shows you every class' probability for every patient with a predicted label and a true label. And so, we have a pretty good model, they match for the most part.

So, I'd encourage you to download the package. Go to GitHub and get the most recent version, if you want to play with this. If you can be patient, we'll have a CRAN release for you probably within a month. But we don't know exactly how long that'll be. But, again, you can always get the source version from GitHub.

Taylor: One more really good question from the chat window kind of related to whether or not the classes need to be mutually exclusive when you're trying to build the model. So the example was if there was a patient that is both has HIV and is diabetic and you're trying to treat those as two separate classes but in fact they are both, so how would multiclass handle something like that?

Mike: So I think— yeah, that's a great question and definitely a practical one.

I think, for the multiclass, here, you have to have a single row associated with a single class. But maybe you could put a secondary target row and just kind of know that if the algorithm gave a toss up between diabetes and HIV, maybe it would get that primary diagnosis wrong, but maybe it would get the secondary diagnosis right. But that's definitely a more advanced bit functionality. And we'd encourage you to just pick one class to represent that patient for now.

Taylor: Definitely.

Mike: It's a great question.

And it’s great to see the interaction and the excitement about this. Glad to see that everybody’s participating. We’d love to continues this, get your feedback and know what you want us to talk about, what would help you.

We're still planning on getting a homework out for you. Just, with Levi and Taylor both on paternity leave, it’s been a little rough on Taylor here and I. So haven’t exactly had a lot of time but we're trying to do the best we can.

Please join the Slack Community. We've had a lot of good discussion on that with just like in the chat, people helping each other out to get healthcare machine learning implemented in their systems. And it’s been really cool to watch that as well.

So, thanks for watching, everybody.

Taylor, do you have any more questions? Or do you just want to give a preview of next week?

Taylor: Yeah, a quick preview on next week.

Actually, I spaced on that. If you don’t mind doing that for me, that would be great.

Mike: Not at all. Great to be honest.

Next week, we’ve got Risa Myers who’s a data scientist at Houston Methodist Hospital. She’ll be coming on to talk about challenges in healthcare and what her day-to-day life is like working within a single hospital. So I think that’ll be a really exciting interview and I'm certainly looking forward to meeting her.

So, thanks again for watching everybody. And we will see you next week.

Narrator: Thank you for joining us today. Remember to like, share and subscribe. Comment below and click the links below to join our Slack Channel and our healthcare.ai Community.