Tyler: This is the healthcare.ai live broadcast with the Health Catalyst Data Science team, where we discuss the latest machine learning topics with hands-on examples. And here's your host, Levi Thatcher.

Levi (L): Hi I'm Levi Thatcher, thanks for joining us. This is the "Hands-On Healthcare Machine Learning Broadcast." This is my co-host, Mike Mastanduno. Hi, Mike.

Mike (M): Hey, Levi.

L: What do we got on tap for today?

M: Thanks for joining us everybody. We've got a great show planned. Today we're gonna be going through the mailbag, as usual. We've got a lot of user questions, and some of them are really interesting. Then we'll be doing a new segment, which we're calling "Machine Learning Legend."

L: Oo Yeah.

M: We want to kinda go back into the archives of machine learning.

L: Diggin deep.

M: Yeah and look at who came up with the techniques, how they got to where they are now, and what those techniques are doing today. Then we'll move into the main event, which is workflow. We're gonna be talking about how you train and build and deploy a machine learning model in a healthcare space.

L: Very practical.

M: Yeah specifically using healthcare.ai. We've had a lot of people asking [for] kind of a general walk-through of how that works.

L: Awesome.

M: So it'll be great to show that. Finally, we'll be talking, with the chat, interacting, hoping to get questions from the users. Before we start, we have the usual housekeeping reminders. You can log into YouTube and chat in the chatbox on the right or just below the video. If you want to watch on YouTube, you can click the YouTube logo in the lower right corner of your screen. We'll be definitely into the code this week, so if you want to change the resolution on YouTube, the gear is the way to do that. And then finally, please throw us a bone and subscribe. And you can be updated when we're gonna do a new one. Although, the shows are always this time, right here, right now.

L: Yeah we have blog posts going as well, which you'll get notified in the subscription there. Check that out.

M: Great. Without further ado, let's get going.

L: Yeah so what's in the mailbag? What do we have?

M: So the mailbag. We had a user [who] said they're just getting into data science and they're kinda curious about some of the use cases of data science in the healthcare domain: how we tailored healthcare.ai to meet those needs, what the project life cycle has been like and what the end goal of the project is.

L: Yeah the idea with machine learning, our stance, is to be super practical. So typically, a health system will want to improve x, y, or z. Typically these things are like readmission rates, infection rates, things like that. Maybe it's mortality rates. So when we interact with them, we help them to create a model to actually look forward, not only so they know *okay well who in the past has had an infection, but who in our hospital will have an infection today?* Whether it's something like central line infection or maybe sepsis. The idea is that using healthcare.ai, it's very easy for an analyst to both create the model on data the hospital has and then to actually deploy it so that each day the clinician gets updated predictions as to how to prioritize their limited resources.

M: That's great. That's actually exactly what today's show is gonna be about, right?

L: Perfect.

M: That's great. Thanks for that question. We had another question about masters programs [from] somebody wanting to get into data science. *What's our opinion on business analytics programs from business schools versus data science programs from tech schools? They're largely the same kind of course material, but how important is the spin to getting a job?*

L: That's a great question. And when you say tech school, that's like the computer science department?

M: Yeah exactly. So I think it comes down to what kind of job you want to get. I think next week we're actually gonna talk about different career paths in data science. Just a little teaser there. There are some jobs where your main role will be business intelligence, which is kind of like: you're taking data from within the company and presenting it in a format that helps business executives make decisions about whether they should buy a new company or build a new building, things like that. I think a business school degree might tailor well to that position. But, if you're looking to be a hardcore machine learning programmer, you might be better off with a tech school degree.

L: Meaning like get a CS, or Computer Science, degree and go towards maybe the route where you're learning how to build software.

M: The data science masters might be more tailored to machine learning than the business analytics program would. I think either program is a good way to go for any type of job. It's all about the experience you have and how you market yourself.

L: Yeah that is definitely important.

M: And we have one last question for today. People have been asking what we've been working on lately, like *what's going on with the healthcare.ai package? What's new?*

L: How nice of you to ask. Mike's been in the code a lot, doing some impressive stuff.

M: We've been building tools lately. Help people building models, evaluate their models, evaluate their data, make it better. So two functions we're excited about. One is when your model is in production, which we'll talk about later, and you want to evaluate the performance, now we have a handy dandy easy to use function that will evaluate the performance for you. All you have to do is feed it some columns that you've pulled down from SQL. And then the other one we've been working on is on the front end of model creation. So, null values in your data. You can count your nulls, and in your training data set, that might be consistent. But if you're using a data source that's constantly refreshing—like a real data source would—you might not have a sense of whether or not a piece of data is gonna be available an hour after a patient checks in, or if it's gonna take a week for that data to be available. We've written a tool that'll help users decide which columns they can pull in.

L: That is super handy. We run into that issue frequently. The idea is that the health system wants to predict readmissions, [so] they'll ask us. They'll say okay well, can we predict it upon patient arrival or can we predict it 24 or 48 hours after arrival? So what this function does is it goes and checks your data in the EMR and looks and sees 24 hours in, we have this many columns available. More than half of those columns are filled, or 75% of them are filled. So you can say, okay well maybe 24 hours in, we have enough data to actually create a prediction for clinicians at that point, and you move on from there.

M: So that's awesome.

L: Great work on that.

M: From the chat, we have somebody saying that they started the Jose Portilla class that we recommended last week. Glad to hear that's going really well and best of luck as [you continue]. Keep us updated on how it's going.

M: So, "Machine Learning Legend." It's a new segment.

L: Yeah so the idea is we want to change things up: have these segments, do some vignettes, talk about pillars in the field that we look up to, that we've learned a ton from, that have built the tools that we use in our jobs every day. The first person that we're gonna profile is Leo Breiman, who's actually a professor at UC Berkeley for decades. Have you ever been to Berkeley?

M: No I never have.

L: Lovely town, I hear. I'd love to make it out there. It's a nice little educational enclave, I hear. So, Leo was there for decades. Distinguished professor of statistics. Founded a lot of the field of machine learning. And what he's mainly known for, one of the many things, is helping found "CART." Mike what is CART?

M: CART is an acronym that stands for "Classification and Regression Trees." It's an ensemble-based machine learning method, where you take a whole lot of decision trees and lump them together with a subset of the data or a subset of the features that you have. It's kinda like crowdsourcing your machine learning.

L: Oh yeah.

M: Instead of a single decision tree, you're gonna use an ensemble of them and you gain a lot of statistical power and you get a lot of nice benefits by combining them.

L: That's right. From chaos comes order?

M: Exactly.

[*discussion about who's going to talk next*]

L: Oh so Leo, yeah. The RandomForest algorithm we've talked about a lot, and Professor Breiman was key to that, like Mike mentioned. He, of course, was building on other people. He stood on the shoulders of other giants. There was this other algorithm out there, that was sort of a proto-RandomForest, and what Professor Breiman did was add on something called bagging. This is aggregated boosting. Mike, you've got statistic courses: boosting, what is that? So many weird words.

M: They all kind of mean the same thing, where you're gonna take a sample, a subsample of your dataset, and use that as one group. Then you take a different subsample of the data, and you use that as another group. And then you keep taking subsamples and then throwing them back into the bag when you're done. Then you take the aggregate of [the results from the different samples].

L: Yeah exactly. That's amazing. Thanks to him and this work that he accomplished, we have the modern RandomForest algorithm which is one of our favorites. It's robust on many different types of data sets. It's very easy to use. There aren't a lot of hyper-parameters that you need to tune, if any. Sometimes you just say *okay well I want three hundred trees instead of two hundred trees,* if you get crazy—or maybe a thousand, depending on how crazy you are. But the idea is that it's super robust, it helps people get into machine learning. It teaches me that interesting math is going on all the time, so people are developing things in the math community that help us. So, when you're in your calculus courses or stats courses, it's amazing what can be accomplished from what you learn in there.

L: Leo, Professor Breiman, also was helpful in the development of Lasso, another one of our favorite algorithms. Let me just pull up the name of what he did here, so he actually was, of course, standing on the shoulders of giants himself with Lasso as well. He developed something called the Nonnegative Garrote, which Rob Tibshir ani at Stanford later turned into the Lasso. So Professor Breiman contributed to two of the foundational algorithms for healthcare.ai, the Lasso and the RandomForest. In the show notes afterwards, we'll throw in a fantastic YouTube video where he describes how he had this bolt of lightning-sort of insight that helped him come up with the RandomForest algorithm. It's a fascinating story about using radar to detect what type of ships the Russians had in certain places during the cold war. We'll put that up in the show notes. Check it out.

L: So he's our first "Machine Learning Legend" that we're excited to talk about. Please read Wikipedia profile. Read his papers. There's a paper we'll show right here. It was cited 26,000 times. [*pulled up the Wikipedia page and article*] It looks pretty humble, from 2001. But, 26,000 reference citations?

M: That's a lot.

L: I come from academia, like I got a PhD. And five [references]? If anybody cites your work, it's like "Wooo! People care? People read it? Wow that's amazing." But 26,000 is unbelievable. So, we'll throw the paper up in the show notes afterwards, as well. But what's next on the docket?

M: Next we have the main event. We're gonna be looking at how to build a healthcare machine learning model using healthcare.ai. And I think it's useful to know that we're gonna be focusing on the high level, and then diving into how healthcare.ai can address each step in the process. We're trying to make the process [easier]—it's a long pipeline of things you have to do to go from [the model] that you're trying to build to actually having it serve up predictions. Healthcare.ai: one of the main goals is making that process easier.

L: Definitely, definitely. And please reach out in the chat or through email, we want to make sure you understand what we're working on. We'd love to have an interaction. We need you and your contributions and your questions to make this helpful to everybody. We're gonna look at the computer here for a little bit, and what we're gonna be starting with is RStudio. We've talked about this before. RStudio is the main environment in which you interact with R, and perhaps the best way to interact with R. How we're gonna start out is by loading in healthcare.ai, by [typing "library(healthcareai)."] You can refer back to this later, this link will be posted, perpetuity. So, don't feel like you need to follow all these steps, but just kinda enjoy and pick up what you can.

L: The idea is, whenever you start with healthcare.ai in RStudio, pull up these built in docs [by typing "?healthcareai"]. This is the main way to read up on how to do certain things. You'll see that we have a description as to each of the steps. And if you click through these links, you'll see that there are notes as to how to use the functions, argument definitions, examples. And these examples are checked every time we check in pieces of code, so that if we break something, it yells at us and we can fix the example so that it's awesome for you. So what's the first step in getting started with developing a model, Mike?

M: The first step is to figure out what your model's actually going to do. Are you trying to predict readmission rates? Are you trying to predict length of stay? You gotta figure out what's most valuable for your organization, or most fun for your project.

L: Yeah definitely, so what's the business question of interest? If you're working for a health system, of course the business question comes first, and that's what should drive the machine learning. So things we've been doing has been a lot of classification type work. So is somebody gonna get sepsis or not? Central-line infection or not? Be readmitted within thirty days or not.

M: And why have we chosen those things? Are those random choices or?

L: Well, great question. The idea is that this is what health systems are worried about. They want to decrease the infection rates or readmission rates because often times they get penalized from the government—from Medicare or CMS, if you're in the field. CMS issues penalties for exceeding certain thresholds, in terms of readmission rates. That's where they come to us. They say, *we've used descriptive statistics. Our analysts, using Qlik, haven't been able to achieve* *these gains that we need. So, let's use the machine learning and get this forward guidance*. That's where we start, [by] saying *okay let's focus on readmissions reductions*. Once you have a data set prepared—let's say you've gone and pulled the columns that are of interest, you have some fields describing who's been in your hospital, who's returning or not in the past—you want to do some feature engineering. You may have heard that [term]. We often call a column a feature in terms of machine learning. That's a word that'll pop up a lot. So Mike, what are some feature engineering steps that come up quite a bit?

M: Oh well you might have the patient's birthdate, but the more relevant thing to get from that would be their age. So you'd have to do like a transformation to get today's date minus their birthday, and that'll give you their age.

L: Exactly, and that's the kind of thing that we've tried to make super simple in healthcare.ai. If you notice in this help screen, we have all sorts of functions down here. We'll be organizing these over time, but the idea is right here you have a function "countDaysSinceFirstDate," so that function will take the date field, like Mike mentioned, and transform it into "days since birth" "days since hired," things like that. And you can look and see these other functions, but what are some of the other ones that you've seen pop up frequently, Mike?

M: I don't think this one's in healthcare.ai, but it would be useful to convert a latitude/longitude coordinate to a zip code.

L: Oh yeah.

M: We could do demographic information. We use demographic information in healthcare.ai, so—

L: It would be a good one to have.

M: it would make sense to have.

L: Definitely.

M: We'll put it on the roadmap, I suppose.

L: Yeah and that's the thing. As you guys come up with these functions, let us know what we can do better and what we can put in the package to make your life simpler. We want to be able to help you.

L: And [there are] other things as well, imputations, sometimes you'll have columns that are missing data entirely or they're missing too much data, so you need to maybe take those out or fill in those columns with the column mean or something like that.

M: But that's different from feature engineering, right? That's more into the data cleaning?

L: Yeah, it's interesting. Sometimes you'll hear pre-processing mixed with feature engineering. We try to help with a little bit [of] both sides of that coin there.

M: What's the difference, generally, between those two?

L: Yeah it's somewhat fuzzy, but maybe we could call pre-processing is like doing some calculations to even get your—it's fuzzy. So pre-processing is maybe like cleaning the data, checking, making sure that there is actually data there in the columns, and there's not nulls. I think of feature engineering of more of like converting a [latitude/longitude] to a zip code, like something that lets the machine learn, rather than just cleansing.

M: And data cleaning being more like making sure that a string actually has a value in it that you want to use, and that all the dates are in date format. Things like that?

L: That's a great question. Yeah so we're, again, trying to put that all in here. Once you've done that step, and you have your data and you've transformed your columns into something that's helpful to the machine, so it can learn, what you do, is you can read the steps here. [*pulls up RStudio and directs viewers to the documents in the bottom right quadrant*]. And we say okay well, generally, after feature development, there's a two step process. In the first step, you're comparing two different algorithms, and we've talked about these a lot: Lasso and RandomForest. [They're] robust, fairly easy to use, and applied across a lot of different use cases. So, if we show just a little bit of code here, we won't show too much of this, but if we click in and show "RandomForestDevelopment" here. We'll walk you through just a little bit of the code. So, you'll notice as we scroll down, we have different arguments that we describe. We have some other links there, in case you need to go back to the home page. And then we have examples, which is where you want to spend most of your time. You'll notice in the examples here, we start with the "iris" example. You heard of the "iris" data set before?

M: [It] seems to be a popular one.

L: Yeah it's out there at every healthcare package, it seems like. And it's a way to easily get started because it comes with R. If you scroll down more, you'll see there's a csv example, so if you're working with flap files or CSVs, here's how to get started with those. Really quick, we'll talk about this SQL server example at the bottom here. What you'll want to do if you're following along at home, now or later, you simply copy this code and put it into this script window [in the top left quadrant of the window]. If you don't have that window, you just click "File" → "New File" → "RScript." [Then] copy that in there. We'll just talk about a couple of these steps here. So, at the start, we're connecting to SQL server, you might notice. Is that typically the case, Mike?

M: It seems like it is for healthcare organizations. A lot of them use SQL server and windows. That's kinda why that functionality is in there, and not something open source like Postgres. But we're excited to provide support for other SQL engines as well.

L: Yeah and the nice thing is, if you look at the CSV example, you can write your own little Postgres or MySQL connector such that you can work with those databases if you want. Again, just look for that CSV example. If we step through here, you'll notice we're connected to the SQL server, pulling in the data. And we're setting up some arguments here, so we're *saying okay "classification" what does that mean?*

M: So classification would be: are we trying to predict a 'yes' or a 'no' value? And just so we can see how it fits into context. So far the steps we've covered are,

- We picked our business question, so maybe it was, a user suggested that maybe we try prediction the patients that are likely to no show their appointments. That's actually a great use case because it can really encourage efficiency within a practice because they can double book a slot or use extra reminder calls. So that's a great use case.
- We can move forward because it's classification. We'd have to go find a data set somewhere. That's step two.
- Then, we'd have to clean the data and engineer features, so that we actually get some use out of that.
- Then we're ready to come into the code and define that it's a classification model predicting "yes" or "no." The patient will or will not no show.

L: Yeah don't be intimidated, so if you've played with various languages in the past or if you're fairly tech savvy, you don't have to worry too much about the data cleansing and feature engineering step. You can maybe go back to that a little later and work on that a little more if your model's not as accurate as you would have hoped. Just dive in and you can kind of iterate from there. Just dive in. So the idea is, again, we're specifying classification because we're predicting thirty day readmit flags. You'll notice in this argument list right here, we're setting up the model that we're developing. We're saying okay yes, we want to do imputations, we want to handle those null cells [and] put something in there. We have a grain column. Grain column. Some people raise their eyebrows at that. What's a grain column?

M: A grain column is sometimes called a key column, right?

L: Mhmm

M: It's kind of like the patient ID, but not necessarily the patient ID visit. If patient one has come to the clinic three different times, they'll have unique visit IDs, but their patient ID will always be "1." So the grain idea is the way that you tie those three lines from the "1" patient to that one patient.

L: Yeah, yeah. Exactly. Great point, Mike.

M: Some of the models don't take advantage of it, but some of them do.

L: Yeah really this isn't necessary, so if you have it in your data set, you can specify it there, but you can also just delete it and it works just fine. So, let's just simplify that for right now. We're predicting "ThirtyDayReadmitFLG," so again this is the "yes/no" column that we are predicting. Pretty straight forward. We're gonna use one of the processors on the machine. That's just the default. Down below gets a little bit more exciting, get to the algorithms themselves. We're trying to compare Lasso and RandomForest.

M: Levi, before we get there, can I interject? We had someone suggest that maybe we use weather data to predict no shows.

L: Yeah yeah!

M: So like what's the functionality of healthcare.ai to use data from external sources?

L: Yeah that's a great question, so the idea is that, in healthcare, it's typical to have a data warehouse, or they're fairly common Health Catalyst does a lot of data warehousing. The point of that is to pull in all of those disparate data sources, so things like claims data from your health insurer or census data or even weather data. We're actually working with some people that are interested in looking at air quality data and how that might affect things like asthma exacerbations. If you can pull it into your data set, healthcare.ai will quickly help you see *okay well is air quality data affecting my readmission rate or not? Should that be part of the model?* It's a great question.

M: On a related note, we said it's SQL server primarily. But does the version of SQL server matter? Whether it's 2016 or 2014?

L: I think we're fairly robust. I haven't played around with anything prior to 2012, but give this a whirl. Let us know and we'll fix it up from there. That's a great question. Keep them coming. That's super helpful, by the way. This is meant to be interactive, so we love that. Please keep them coming.

L: So we have two algorithms here. We're gonna say *okay algorithm, meet data, give us model.* That's kinda the equation, if you think about it that way.

M: That is a good way to think about it.

L: Matching an algorithm with the data gives you a model. If you think of a model back from your high school or college stats class, a model can be something as simple as like a line, some linear thing that compares square footage with house prices, or something like that. We're doing something akin to that, just a little more complicated. But healthcare.ai helps kinda take away the rough spots, and the complicated pieces. So what we do here is we compare these two different algorithms and we say okay well, let's pull in the data and see how well these algorithms do. We'll make this screen a little bit bigger there [*adjusts the size of the quadrant he's using in RStudio*]. What happens is you have an actual accuracy measure or performance measure. First, is lasso, as you notice right there [*highlights the point on the screen that says "lasso$run()"]. What we see is, okay for Lasso, the area under the ROC curve is point seven four, and we're gonna compare that to the RandomForest result. If you scroll down, the area under the ROC curve for RandomForest is point nine seven. So what does that mean: ROC curve? I guess we don't have to go too much into it, but higher is better?

M: Yeah higher is better. At a high level, a ROC curve is a really robust way to evaluate machine learning classification algorithms. The value spans anywhere from point five, which is like you flipped a coin, to one, which is perfect.

L: Yeah, so the RandomForest—Thank you—the RandomForest is doing pretty well. What you would do at this point is say *okay well maybe I want to get closer to one, so maybe I'm gonna pull in some new columns. I think I can get more columns based around length of stay or* *readmissions.* So you can pull those in, see if RandomForest gets you even closer to one by redoing what we just did here. What happens after that?

M: After that?

L: Yeah you have a good model, you're close to one. In reality, you don't get that close to one. Maybe point eight is considered fairly normal.

M: Yeah I actually had a question about that. So you have your Lasso ROC curve is point seven four, is that good or bad? Should you be worried?

L: Yeah great question. It depends. It's hard to say. If you're doing something that's very clinical and you're dealing with cancer patients and treatment optimizations, then you want something point nine and up. If you're predicting no shows and its okay if you are calling people that maybe would have showed up anyhow. If you're missing with no show prediction, it's not the end of the world. When you're doing cancer treatment and a lot of these clinical issues, then you want to have a lot more performant model, up into the point nine arena. And really point eight is really as far as you can get with a lot of the clinical use cases. If you talk to your physicians, maybe it's an accuracy they're okay with.

M: Yeah and I think it's worth keeping in mind, too, that a lot of times, point seven four is gonna be leaps and bounds ahead of what they're currently using.

L: Yeah exactly. You have to compare against: *well, what do we have now*? That's a great point.

M: So you asked what we do next. We got a good model, and so we've come up with our question, we've loaded our data, we've cleaned our data, we've played with the different algorithms, and we have a model with the feature set? Did we prune feature selection?

L: Kinda. So let me just interject really quick. That's a great point. So the nice thing about Lasso and RandomForest is, you'll notice, it gives you an idea as to what factors most influence your prediction. So right here we see "GenderFLG" didn't help at all [*overall importance for "GenderFLG" was zero*], so let's take that out of the model if we can. It's not really helping, so why leave it in there? Let's imagine you get to the point where you have a type feature set in your model's performance, where do you go from there?

M: So from there, you're ready to go onto the production server and start surfacing predictions every day. You can actually just save the model to your desktop and then put it up on the production server as a model. Then, whenever a new patient comes in, their row gets entered into the table, and it gets run through the model and prediction is generated for them.

L: Yeah so that's where a lot of people get stuck, a lot of papers come out saying such and such research group created a model and it was performant and it was on this cool medical data, but often times, they get stuck. You hear later on such and such health system didn't actually deploy the model or it's not being used to enhance patient care.

M: Part of that is because it's academic research, right?

L: Yeah.

M: So there's hoops they have to jump through before it gets surfaced to actual patients. But then the other part is that it's just hard to do that.

L: Yeah it's hard to integrate within the ETL servers and get the IT folks involved, and know how to wrangle all the folks that are irrelevant to that question, and then just have the software to do it. That's one of the things that this package helps out with. So, let's quickly jump to that deploy step, and then we can take some user questions after that. Notice in RStudio, it's very easy to jump back. Here's where we were and we're clicking back to the main page and you see we have "RandomForestDeployment." So let's say we want to deploy a model, so you click through that. You'll notice again it describes what it does, we have a description of each of the arguments, and most importantly example code. So if you scroll down, you can see an example with csv data, you can see an example with SQL server. We'll just stick with that SQL server bent real quick.

M: It's nice. Those examples are just so helpful because writing all this—it's kind of like when you're learning a language, for instance—it's much easier to see a foreign word and know what it means than it is to come up with that organically. And it's kind of like that with the code. So it's hard to write this whole code block, but if you can see the prediction column needs to be length of stay instead of readmission. All you have to do is follow the syntax there.

L: It's nice to have a place to start, really. If we look at this, again, you're loading from SQL server, this will look fairly familiar. You have a query. You have a database, and a server you're pulling from. If we scroll down, let's say we've got the data into RStudio, what you'll notice is okay well you can remove a column if you need to. Maybe you can do that in the query if you want. But the idea is you have this argument list, like Mike was mentioning, where we populate it for you for a typical case here. [*looking at lines 32-43 in the code, the lines are auto-populated. Levi will explain what each of those mean in this situation.*] In this scenario, we're predicting thirty day readmissions, and we're using the "PatientEncounterID" as our key column. Plausible scenarios, right? We're doing classification again, but what's the main difference here? [*highlights row 40, which is 'p$useSavedModel < - FALSE'*] We have something called the "UseSavedModel Flag," so what is this about saving a model?

M: So saving the model is, after we've decided that our model is a good one, we don't want to have to retrain it every time we want to generate a prediction.

L: Well wait, why though?

M: Because retraining takes time and it takes computational power and you don't want to be using the servers resources to do that. The other thing is consistency. If you're using a saved model, you know what's in that model because you've evaluated on test data set. We had a user ask if that AUC was calculated based on the training data set. That's kind of a sticky situation. You don't really want to do that because you might not be able to generalize to new data well.

M: So the area under the curve, to answer that question, was calculated from the test data set, which was held out during model training.

L: That's a good point.

M: We have some idea of how that model behaves. So, if we were to retrain it every time we had new data, we'd also have to reevaluate it, and that would just be a headache.

L: Yeah you don't want to be doing that every night. The idea is you train it [then] save it so that it's on a file on your hard drive and then you can put that wherever you want, like Mike was mentioning earlier. Once you've saved it in that file that you associate with the model—So there's kinda two different files. You have the script that has the R code, then the file that's holding the model. In the R script here, you'll specify as to where this prediction is going. [*Highlights row 43, which is 'p$destSchemaTable <-"dbo.HCRDeployClassificationBASE"'*]. So typically, you know in healthcare, often you're going to things like a SQL server database. You'll point to the particular schema and column, or table, rather. The idea is that these arguments are all set up for you right here. It's very simple for an analyst to get started, or even a data scientist to save a lot of time. Then, each night, after you save the model and put it wherever you need to, you'll simply be putting new rows against the model.

L: So, let's say that Joe comes into the hospital. Joe.

M: We always use Joe.

L: Yeah, Joe's got a lot of issues, see, so he's in the hospital a lot. So, Joe comes in. Maybe we know Joe's weight, his cholesterol numbers, his blood pressure, and we want a score as to his likelihood of being readmitted to that hospital within thirty days. So we run his row against that model, and that's where you get that prediction. That's what drives more efficient clinical care. The physician is able to look at this and say *okay well maybe he needs more resources than these other folks because he's very likely to be readmitted*. That's kind of the full circle picture there. Did we leave anything significant out?

M: I guess the only other thing is, I'll go through the process one more time just to keep the high level.

L: Yeah kinda wrap it up there.

M: We picked a question, it was thirty day readmissions here; we found a dataset we liked; we cleaned it up; we engineered features; and then we did this iterative model selection process where we tried Lasso, we tried RandomForest. We found that the RandomForest was better because maybe the data has some non-linearity to it. RandomForest handles that well. Then we saved the model, we moved it to the production server, and now it's serving up predictions any time a new patient comes in. At this point what do we need to do?

L: That's a great spot. One thing that we should mention is that it's often, the model's a lot different in terms of performance, like on retrospective data versus in the wild. The performance you see using healthcare.ai may degrade after you put it up on the production server and get these new rows coming in each night. What we often recommend is when you put it into production, wait some time before letting the clinicians know to use it for guidance. If you're doing thirty day readmissions, leave it up there for more than thirty days so that you get the actual results coming in, you can see whether Joe came back within thirty days, and Sally and whoever else. Then you can see okay well yeah we are actually at that performance level that we thought we were. The package has helper functions for that as well. Then you're able to check and say okay it was performing like we thought, so let's surface it in QuikView or Tableau and let the physicians use the output.

M: You mentioned that sometimes the performance drops off when it goes into production and we don't want to make healthcare.ai look bad. But sometimes that's just because the data source is different, right?

L: Yeah so we do that eighty twenty split to be able to get accuracy and perhaps that's not representative of data that we'll be using in the wild. Things like that. Sometimes there's target leak, where in the EMR, sometimes the field you're predicting is populated before these other fields that are driving the prediction. Sometimes you don't really learn that until later. In future episodes, we'll go into tricks and tips that you can use to make sure that your performance in the wild is actually as good as performance on past data.

M: Then what about retraining the model? We don't want to retrain the model every night, but it does make sense to retrain it every couple months, right?

L: Yeah, so it's good to check in. Put a note up on your calendar, maybe every quarter. You can even automate this, so you could have a script running nightly that just shoots back some performance statistics. That's how you're able to see: *okay has my performance degraded substantively? Do we need to retrain? Was there some change in the underlying data?* Things like that.

M: Great.

M: I guess that's the process. We've seen it at a high level. If you want further resources on what that pipeline looks like, a lot of the MOOCs that we recommended last week kinda structure their courses based on what the data science process looks like, or the model-building pipeline. A lot of them have overviews. There's that great overview course from the show notes last week.

L: Oh that's fantastic. Check it out.

M: That was almost entirely just based on the pipeline. So you can check those out if you want a cover. We have a few questions from users—

L: Oh do it.

M: if you're still feeling up to it, Levi. I know I am.

L: Alright.

M: Paul is asking if we're working at all with genomic data.

L: Oh great question. That's kind of like the Holy Grail: precision medicine. That's the future. Imagine not only using EMR data on things like cholesterol and BMI to predict readmissions, but actually knowing that person's DNA for cancer predictions. We're not doing that yet. That is on the road map. It's really based on health systems being able to get at that data more and more. It's just a matter of the data availability.

M: I gotta say though, coming from a radiology background, I think radiology is the Holy Grail.

L: Everybody has their Holy Grail, I suppose.

M: We'll have to agree to disagree there. We've also had a question on getting the models actually into use. It's one thing to have it in production. Paul also asked, *Do we ever encounter resistance from clinicians not wanting the model to influence their decisions? How do we deal with that?*

L: That's a great question. The idea is that you want to help build trust in your model. Clinicians are skeptical, which is good. If you don't provide an explanation as to why your model said Joe's high risk, then they're not likely to accept it and to use that guidance. What we've done with healthcare.ai is you'll notice—we won't show it here—the idea is that not only are you surfacing a prediction as to Joe's risk of thirty-day readmission, but also the reasons why. The top three columns [show] why that risk is so high, so the clinician gets *okay that makes sense for Joe*. Maybe he was older, or maybe he did have high cholesterol, things like that.

M: I think we have time for one more question.

L: Let's do it.

M: It's a good one. Jonas is looking for if we could describe what a production server or environment might look like.

L: Yeah do you want me to take it?

M: Sure.

L: Often times these large health systems will have these Windows servers, so it's different from Windows10. It's actually a different operating system. They're often called ETL machines or EDW, for Enterprise Data Warehouse machines. They'll have maybe a hundred gigs of RAM, maybe sixteen processors, which is a lot different from our laptops, typically. That's kinda the hardware and software side of things. But typically, they'll be refreshing these data sources each night. Maybe they have a warehouse and they're pulling from the EMR, they're pulling from claims data. These are the machines that are doing this compute each night to get the data sources ready that feed the dashboards that administrators use, that clinicians use.

M: They're not optimized for web surfing like our computers, right? They're optimized for moving data around.

L: Yeah doing compute, exactly.

M: Great, well thanks everyone for joining us. Don't forget to subscribe. We'll be here next week. Levi, thanks for the great show.

L: Thanks, Mike! Great work.

M: Alright. See you next time.