# EPISODE 590

[INTRODUCTION]

**[0:00:00.3] JM:** This episode features recordings from our most recent meetup in Los Angeles. The first speaker is Kyle Polich, who is the host of Data Skeptic. Data Skeptic is a podcast about machine learning, data science and how software affects our lives. Kyle is one of the best explainers for machine learning concepts that I've met. For this episode, he presented some material that's perfect for this audience. It's a discussion of machine learning for software engineers.

The second speaker of this meetup was from Second Spectrum. Second Spectrum is a company that analyzes data from professional sports, turning that data into visualizations, reports and futuristic sports viewing experiences. We had a previous show about Second Spectrum, where we went into the company in detail. It was an excellent show and I wanted to have Kevin Squire, an engineer from Second Spectrum to come on this meetup and talk about how the company build machine learning models and tools to analyze sports data.

If you haven't seen any of the visualizations from Second Spectrum, stop what you're doing right now, watch a video on it. It's incredible. This year we had three Software Engineering Daily meetups so far in New York, Boston and Los Angeles. At each of these meetups, the listeners of the community got to know each other and talk about software. I was really happy to be in attendance at each of these and I'm posting the talks given by our presenters.

The audio quality, the experience of listening to these meetup talks is obviously not as well produced as the controlled experience that I have in the one-on-one interviews, but there are also no ads and these talks are fantastic. I hope you enjoy hearing from Kyle and from Kevin.

Thank you to TeleSign for graciously providing a space and some delicious food for this meetup. TeleSign has beautiful offices in Los Angeles, and they make SMS, voice and data solutions. If you're looking for secure and reliable communications APIs, check out TeleSign. Also, if you're looking for a job in Los Angeles, definitely check out TeleSign. They have some very nice offices.

We'd love to have you as part of our community. We're going to have more meetups over time, and you can be notified of these by signing up to our newsletter or going to softwareengineeringdaily.com/meetup. You can also come to softwaredaily.com and get involved in the discussions of our episodes and software projects.

With that, let's get to this episode.

[PRESENTATION]

**[0:02:43.4] KP:** Well, Jeff asked me to come talk. I thought well, I'm really a data scientist and there's a lot of overlap with software engineers that I knew would be here tonight, but I was trying to think, where is the middle ground, or what can be interesting, because machine learning is becoming so much more democratized. It still takes a certain practitioner for certain areas, but I know probably everyone in this room who writes code has dabbled in it at least.

I wanted to share a few thoughts on where I think machine learning is going. Might be of interest to you if you want to start dabbling in it yourself, maybe if you're already a practitioner, you'll agree or disagree with some of my thoughts and why not throw in some blockchain, because that seems to be a hot topic on Software Engineering Daily these days.

My outline, I'll give a very brief history and a whole review. I assume probably everyone already knows about machine learning at some level, so I'm not going to make it too basic. What I really wanted to get into is a few obvious directions about where machine learning is going and maybe a few theoretical ones that could be interesting.

Again, I knew I'd hit a bunch of software engineers, so this is the way I wanted to find machine learning for you guys, give a different to an audience of CEOs, but three simple lines; why do the teams take so long to build models? Looks really easy, right? A lot of magic happens in not only your loading step, but also in the labeling of data. For the most part, that is realistically three lines of code that might build you your model. Under the hood, a tremendous amount of other things are happening.

For the most part, we can think of machine learning as a black box, that data comes in from your database, from S3, from who knows where, you give it some project and I'll pop some answer. From a engineer's point of view, it's a prediction, a classification, something like that. Business owners, product owners are going to care, “Well, how do I know it's correct? How do I know it's not biased and all that sort of things?”

If you're the integrator, the question is how do I get the data it needs and where do I store its output? For any business people in the room, here's the way you can think of it too. Machine learning has benefited from and will continue to benefit, how do we take this thing and make it better? Well, we’re going to push it to the limits in every possible way. The first transistor was big and bulky and expensive and all that stuff, and we've progressively pushed that so small that we can't even really see them anymore.

Faster compute will make it possible, the compute models faster. Parallelization will allow us to compute models on larger datasets, which has been profoundly successful. More data and specifically better data will get you better models and better results. Future engineering is one of the keys to all of machine learning. Just thinking that you can throw all your data into a bucket and something magic comes out. Well, that does happen sometimes, but rarely does that actually work.

Usually, some amount of work has to be done by someone who understands the algorithm and figures out how to feed the data with just the right flavor and whatnot. We're entering an era when these concepts, like transfer learning are becoming very popular. Transfer learning is the idea that you see in computer vision a lot. Where very smart people built vision algorithms that could recognize that something's a chair, or a table, or things like that, but rarely would that same algorithm that a person built for general purposes be able to say that's an IKEA table, or that's a Model X table, or whatever the case may be.

Yet, inside of the neural network that does that is a lot of information content that can be repurposed. Transfer learning is a technique for saying, “Let me borrow that as a starting point and extend it,” and it allows people who don't have that depth of experience in computer vision to maybe custom tailor and get a head start on things. That's another way we're pushing this stuff to the limits, because to really get deep into some of those spaces, you might have to be a domain expert, but there are tools today where a software engineer who knows very little about machine learning could apply that technique and build some interesting classification results. The hotdog, not hotdog is a nice example of that.

There will be new algorithmic approaches, but honestly that maybe excites me the least. What seems to make data science and machine learning push forward is more about the quality and breadth of your data, than the fanciest of algorithm you use. A topic I won't cover today, but it's something that's important as we push machine learning to the limits is interpretability. Models today are becoming so complicated; people don't actually know how they're working, deep learning especially.

There are tools and techniques, things like the lime system that'll help us get some interpretability, but in the same way I can ask you a question and you can give me an answer, but I'm never sure if you're trying to get something over on me maybe, I don't know. At some point, we're going to have to give up on perfect interpretability of machine learning, but that's a topic for another day.

This is the perfect example you never encounter of what machine learning is. Your data is on one dimension here your x-axis, on the left axis the output you're looking for, your model is the red line, perfectly it does a pretty good job predicting that data. Rarely do you get this. Usually get something like this. Data is often has these nonlinear relationships. This is a challenge, because the plain vanilla techniques that are good at the thing on the left here don't generalize well. When you assume data is linear, as many things do, you're going to end up with models that don't quite fit your data. I can assure you real-world data always looks the right, if you're lucky, almost never looks like the left.

We can tackle this in a number of ways. We can try and transform the data and make it linear. We can come up with algorithms that are better at handling nonlinear data, so this isn't a death knell in any sense of the word, but data is often nonlinear; that's one of the challenges we face in machine learning.

Another challenge we face is that data is often highly dimensional. Think of something like on Amazon when they're doing product reviews. You buy a lot of products, but they have millions of products and millions of users, this very sparse matrix that you can never fit into memory. If you tried to solve it with vanilla machine learning techniques, you're not likely to get anywhere useful.

On the left here is an example of the MNIST data set. It's a handwritten digit recognition corpus, where you literally represent that image as a, I guess 16 by 16, 256 featured Becker and intensities being the pixel colors, and that's how we do a lot of machine learning on images. Text the old-fashioned way was every single word in the document and its frequency was a feature. That's gone out of fashion things like word embeddings and word2vec have really taken things leaps and bounds forward their ways of fighting highly dimensional data, but this is another one of the problems machine learning people face.

Feature engineering is expensive and hard, so can we automate it? That's the dream of deep learning. I doesn't apply to every scenario, but this is a diagram from a research paper about computer vision where in the same way I was showing you here, that this image becomes just this bitmap. We can apply the same thing to photos and traditional machine learning logistic regression, random forest, you just give it these photos, it'll never converge to being able to predict something useful from it. Well, this doesn't seem like it would in any reasonable amount of time. If you introduce these intermediary layers, that's the deep in the deep learning, the hope is that at each layer, they're all learning something useful.

If I told you, I'm going to give you a data set, a CSV file and it's going to tell you if an image has a face in it. Yes or no? If it has two eyeballs in it, yes or no? You could probably very quickly come up with a rule-based system that's simple, that would say, “Is this a face, or a chair, or whatever the case may be?” Data sets like that don't exist. We have raw image data.

At the other end of it, I think everybody in this room could come up with a clever way of detecting edges somehow in an image. Where things transition from high-contrast to low-contrast, black to white, whatever the case may be. You could hard-code all these things and that's what people tried to do for a long time, until eventually people said, “Give up on that. Let's just let the machine figure it out for us.”

The magic that is deep learning figures out these intermediary representations at each level, where these edges and little swirly things are detected. From those are composed shapes and from the shapes are composed elements of faces, and from that we get to recovery of the original faces there, and all that's automated. Stata is big, this is a challenge. It's somewhat of a solved challenge today, through data parallelism, techniques like Hadoop and Spark, just spinning up a thousand little cluster, if you can afford it and feel like it's somewhere, even though big data is still a challenge and I don't mean to undermine the amount of work that goes into managing it, it's a problem that is solvable, but amongst these problems that machine learning practitioners face.

One that isn't quite as solvable, although we've got some good tools out there is that data happens in real-time; the traditional machine learning part practitioner would go and collect a massive training set, build their model and then somehow figure out a way to give that model to an engineer and get it to production. Today things shift radically and streaming technologies are becoming more prevalent.

Spark is one of my favorites. There's plenty of other ones I could have listed as well. Serverless plays our role in this and how elastic can be at managing data. And interesting algorithmic techniques, like the two I visualized here that's a bloom filter and a count them in sketch here, these are probabilistic data structures that take up a very small amount of memory, and in that compromise, by not giving it a big amount of memory, you have to accept maybe not perfect accuracy.

Bloomfilter for example, if you give it objects that you've hashed and then say, “Hey, have I seen this object before?” It can always tell you, “Yes, confidently,” but it'll sometimes make some false positives, or I might have that one backwards. Either way, you can insert the negative of the hash and do the same thing. With very small amounts of memory, a stream of data can be processed, and with just a little compromise on perfect accuracy, you can manage streaming data a little bit better for many scenarios. Data is real-time, still a big challenge, but there are some techniques and methods out there to help us deal with it.

Where is machine learning going to go next? There are some of these problems and maybe some solutions for it. Certainly it's going to go on to mobile devices. There is a lot of work going into how we deploy that deep learning technology, so that the image recognition you do today, which might take a picture, send it to the cloud and get the answer back, perhaps that should run on the device itself.

Robotics is a big component of this, self-driving cars obviously. The tooling for machine learning continues to evolve and get better to the point where people who don't understand machine learning in some applications can use the tools out a box for good effect. Where are we going? Many directions, of course. One thing I think a lot about is how models are really hard to build, but easy to copy. I compare this to pharmaceuticals. It takes a lot of money to develop a drug, to test it, to get FDA approval, all that kind of stuff, but out of it comes a chemical diagram you could write on a napkin, and probably a chemistry undergraduate could reverse engineer with basic lab equipment.

I'm not a fan of software patents, although teach there on that. I am a fan of medical patents, because there's no incentive to develop a new drug if someone can steal it in a second and then the millions you invested is just gone. That's not to say that the industry works perfectly, or there are no problems there. It's just that the same challenge is analogous in model building. You don't want to give someone your model, because you invested a lot in creating it, so can you protect that in some way?

Now the obvious answer, I'm sure a lot of engineers have already figured out is of course, make it a service. Send me a rest request, I'll never give you the model, I'll just give you the answer, you pay me whatever per use. That's a nice solution, except for the results of this gentleman Florian Tramer that I interviewed on Data Skeptic last year about stealing machine learning models from the cloud.

The morale to the story if you want the technical details, I'd refer you to his paper, or to our episode, is without too much effort, you can steal a cloud API. Now if that cloud API is streaming in real-time and that kind of thing, there could be some added challenges there, but a static model, it can be extracted and someone can take that assets you think you have protected behind your rest service.

Data is the new oil, everyone's heard that term. Federating that data is key to success. A lot of companies have what I like to refer to as the unfair advantage. Gmail pretty much solve spam. How do they do that? They became the best e-mail client and they got the majority of the users to use their e-mail, so they saw the spam coming in. Our gracious host today, TeleSign, they have an unfair advantage in their phone intelligence products. The more clients they add, the more data they see, and each of the individual clients couldn't provide the same intelligence, because they don't see the world of opportunities that this company sees.

Steam, just to throw a random one in, the video game company; if you develop an app, you can put some cool analytics in it, put in some tracking and stuff and you'll know about your users, but how do your users compare to other games? Who is going to take off or not? You have no idea, but Steam really knows already, if they want to study it. Waze is probably the best example yet. They got just enough people to be on that app and all your GPS is sending back data, so they have probably the best traffic, because you are all their real-time sensors deployed to the streets.

I assume everyone probably enjoys opting into that. There might be a way to block your data, or you could do something to not let Waze get it, but why would you? When we all contribute our data there, we all have better images of the street. I think I got one more. Oh, that's weird. I don't know how that happened, but it got redacted somehow.

Anyway, Cambridge analytic is in the news. A minute ago, I was like, “Yeah, give your data to Waze. That makes sense, right?” I think it probably does, because it's benign. Maybe I don't want Waze tracking what grocery stores I shop at, whatnot. Maybe I'm talking myself out of it, but we all have this issue now of data is good to share, data is powerful to federate, but the world is going to change, and how can we deal with that?

A bit on opting in and opting out; this is a sentiment I'm hearing a lot more. I don't want my data in your model, and most of us don't have a choice. All right, I guess you could run your own e-mail server, that gets you out of Gmail. You could run your own everything and at some point all you're doing is running your own things, you're never getting anything done. We will need to find some middle-ground.

Another option is I would like to in some cases anonymously be included in your model. Let's say someone called me and said, “Kyle, we know your family history. I'm calling you from this hospital research organization. Your family has a certain history. Will you give us a blood sample?” I would love to, if my blood could in some way contribute to medical advancement, especially in things that my family suffered from, how wonderful would that be? I'd love to give it, with the understanding that it's anonymous and all that, and not that company who took my blood is going to IPO and be sold in some other place, and then suddenly someone wants to raise my insurance rates, because they looked at this blood sample and what's included in it.

Can I give you my data anonymously? I mean, you can promise me, it's anonymous but who knows what really happens? Or what one rogue employee could decide to do if they have a little bit too much access? I want my data included in your model, but maybe I want something back. Maybe I should say, “Hey, Waze. You got to pay me a penny a day, or something like that.” Lots of options and ways we can look at this. I want to present one that's becoming a little bit more popular.

Oh, before I get into that, a couple other ideas on privacy, just some interesting notes. This is a technique long known in the market research space. Here's a survey I might ask some of you to take, and what I really need to do is give this presentation again, where I ask a different audience to do the same thing. Everybody think, of those three options, I don't care which one, just tell me your number; zero, one, two or three, and I could collect that data and I could take the average.

Maybe the average for the room is 1.5 is the number everyone reports back. Okay, so then half my group in an AB test, I give them what they think is the same survey, but this is the survey. I've included one new option; drove while intoxicated, which I hope no one has done and I expect those that have are not readily willing to admit it. If you were such a person and you're asked to give this number, you probably count it. If you say your number is three, maybe I just assumed you did the other three options, or your number is two and you're even more anonymized.

If I take the average of this group, minus the average of that group, I know exactly how many drunk drivers are in the audience. I don't know who they are, but I have the number, assuming people answer these honestly. There are interesting methodological techniques that can give us anonymity and I want to see more things like this used, not that that's software engineer's responsibility, but interesting technique nonetheless.

Anonymity through averaging makes a lot of sense, although there's one guy there with a yellow shirt, so he's not quite so anonymous as everybody else. A couple quick side notes that are of interest here and maybe along the lines of privacy, but also somewhat impractical included for completion; wonderful topic, interactive zero knowledge proof systems. This is a way in which I could claim, “Hey, I have a way of solving a problem like the traveling salesman problem, but I don't want to show you how I do it, because that's my secret sauce.” If I had that, that would mean that I have proven that P equals NP and I definitely want to protect that, because I can make a lot of money if that were true and if I was the one who solved it.

Through zero knowledge proof systems, you can submit to me examples, and through a bit of trickery, I won't go into here, I can prove to you that I have the answer without you being able to know what the answer is. On paper, that sounds really awesome, right? That I can give you the data, but you can never know how I came up with the answer. The limitation there is that it really applies to some very specific use cases, mostly to NP-complete problems, or P space complete problems, or stuff like that. While this is interesting, I don't think this is the solution to our privacy debate.

I base that also on the fact that I just checked the proceedings for the upcoming computational complexity conference down in San Diego this year, which just came out this week and there is precisely one paper on proof systems. While it looks like a cool paper, I wouldn't call this some breakthrough things. Complexity theorists are not really pushing interactive crew systems forward. There are just some academic barriers there, either we're not investing enough in that research, or we're not making the breakthroughs in the same way and no one's making too much progress on P versus NP.

While this is interesting, I don't know that it's our solution. Can we do better? Users wish to keep their data private, but we could all benefit from train models. We benefit from ways, we don't want it to go away. Yet, data is necessary to make the machine learning models useful. The creators of those machine learning models, they want to keep the data private. They don't want someone else getting the model that they've created and invested in, and also you as individuals, you probably don't want your data that was in the training set to be included, in the same way I might give my data to that medical research, but I don't want it being passed along anywhere else.

The owner of that data can leak it, can get hacked, can get acquired, whatever the case may be. Do we have any options at all? Another one that comes up that probably some of you are at least a little bit familiar with this homomorphic encryption. It's the idea that computation can be done unencrypted data, useful computation, and that the result is an encrypted thing. Let's break that down a little bit.

Let's say you had some really awesome procedure for doing something to data, and I wanted you to give me that service. I can send you encrypted data, maybe a zip file with a password, but in order for you to do anything with it, you have to unzip it and now you have the raw data. Maybe you're a careful person, you zip it back up, you re-encrypt it, you send it back, so that's good as long as we follow all the procedures well. Ultimately, you saw my real data. Maybe I could avoid that, and that's the dream of homomorphic encryption, that I would pass you this encrypted data, you would do all your processing on it, you could even keep your algorithms private, you send me back an encrypted result and I get to decrypt it and look at it.

On paper, it sounds great. There is progress being made in this area, but it's overall very early. What we don't know is if we'll ever have a fully homomorphic encryption system without limitations. Every single one that's out there today has some limit that makes it impractical. There's always a got you. You have to be able to do as many addition and multiplication operations to do fully homomorphic encryption. That's as simple it is, adding and multiplying in an encrypted form.

We don't have a system that does both. We have systems that can do homomorphic encryption only with addition, or only with multiplication, but then they're not touring-complete. We can't solve any open problem. There are some, like there's one, I think it's this [inaudible 0:22:48.9] that I mentioned here, where it can do as many additions as you want and one multiplication. I'm not even clear on how that works, but that's the limit.

Imagine if I said you could write a bunch of if statements, but no else statements. Actually you could do that, so that's not a great example. This is early stage stuff. We're making progress, but it's not readily available. Most of all, it's computationally expensive. Blockchain has to be the answer, right? Well what are the benefits of blockchain? I'm sure if there's any enthusiasts in the crowd, I might have trouble getting past this slide if they want to add about a thousand other bullets, but as best I can tell, this is really the real six when you break it down. I'll return to these in a moment.

Benefits a blockchain-based machine learning. You remember earlier where I said, what machine learning could benefit from; faster compute, parallelization, more data, these sorts of things. Let's try and marry these together a little bit. Decentralized is not something that the blockchain needs to give to machine learning. We already have it. Immutability is something machine learning doesn't necessarily benefit from. High availability, yes, but we already have it.

Highly secure, well that one's interesting. Yes, we do have secure machine learning today and just the form we transfer everything over SSL. It does have to be decrypted somewhere, so we could say that's unsecure, maybe I shouldn't have grayed out high security. It's not the most profound thing we want to do, because for the most part if you're building a model, you can secure your own data.

The shared interesting, because we can share our data. Transparency and trust is certainly there, which goes hand in hand with interpretability. Really if I'm after anything, it's maybe trust transparency, interpretability and the access to more data. What does that look like? I found four people who are in the space doing stuff, and there's lots of others listed. Omissions are not me trying to insult anyone. It just limits to how many I could cover here.

I'm going to talk mostly about Google's research into federated learning and algorithm known as DanKu system, but I'll mention open mind and singularity.net are two others, people in the space trying to bring machine learning into the blockchain. What does that look like? Algorithm is idea with this project DanKu is that you would post your machine learning problem as a contract, but it's designed to run in the Ethereum blockchain. Some buyer says, “Here's a problem I have. I want you to do a classification problem for me, and I want you to achieve a certain level of accuracy.”

You can put that onto the blockchain, people can download that, contribute their data, build a machine, if the contract insists that the testing data set, the holdout data set, which is not there is used to verify it, so all the typical tropings of machine learning, testing and training data is included in this. If the person who's doing the machine learning here is able to meet that contract, able to achieve that, let's say greater than 90% accuracy, then the contract is closed.

In principle, sounds good, right? There are some challenges here that make this a bit infeasible. First and foremost, that anyone who promises you, "Hey, I'll come in and I'll solve your machine learning problem and I guarantee you this accuracy," that person is a liar. No one can know how accurate they can make a model, or how much they can improve it without starting on the data, because it has only a certain amount to do with how smart and clever you are as a machine learning person. Machine learning is just a tool that helps you work with the data you have.

A master carpenter given rotten lumber with a bunch of termites in it is not going to build nice furniture, and the same thing is true of machine learning. I as a person who does machine learning would not be particularly interested in joining in one of these, because I don't know if I can actually satisfy that contract. It's a big risk for me, but there are risk takers in the world, so maybe they're onto something, maybe I don't have enough vision.

However, quoting their own paper, if they were to do some process on the famous MNIST dataset, that's the handwritten character recognition data set that I gave you an example of earlier, it would cost about $300,000 to come up with that model. I've built models on MNIST and I also bought a house. I couldn't have done both if I was doing it on the Ethereum blockchain. That is quite an expensive trade-off. I think that's a non-starter.

Now, certainly these people might keep working on it. I know there's ways where we're trying to do things, where you put a token in the blockchain and the storage is elsewhere. There's work in this space, but it's early days and I myself I'm not particularly hopeful. One of their other incentives they think is cool is that well, this democratized what mining means, someone who has a bank of GPUs and wants to do some mining that could mine for Bitcoin, or they could mine for models that are out there. You have two opportunities to do useful things with your hardware.

However, the mining is very predictable in a statistical way. You could pretty much figure out how much money you're going to make from the hardware you have, and you'll be small margin of error that's quite predictable. The data science machine learning stuff doesn't have that same predictability, the amount of variance in from model to model and contract to contract is really unpredictable.

I don't think this is a real incentive. If I owned a fleet of GPU machines, I'd stay mining Bitcoin. I wouldn't participate in these products. It would be a more stable business and probably a more profitable one, but I could be wrong. The implementation, I've had a lot of gloom and doom here as to why maybe blockchain is irrelevant for ML. I am excited though about federated learning, and this comes out of a prototype that Google has live right now.

The idea of federated learning is that you pass the model you have to an end-user. In this case, all of us with our phones, or I think maybe only Android phones here. You have user data on it and that user data is very valuable to Google, but we don't want to give it to them, but we can let their software run locally, improve the model they have and then send back not the full model and not the original training data, but what the improvements to the model are.

A machine learning model can be serialized in some way. Usually it's just a set of parameters, a vector of weights, something along those lines. With the training you've done locally, you've updated those weights, those can be pushed back to Google. Now in the same way, I told you earlier a machine learning model can be stolen from the cloud, someone could reverse-engineer well, what training data would have led to the improvement you sent me?

You're not as anonymous as people might like. However, Google says as soon as you send your data back, we immediately average it. That averaging does totally anonymize you. It destroys whatever identity you had in there and it puts all the mush of training data into a pile. That is a good step if you trust that Google will immediately do that averaging process. I think a lot of people would, if there's one company that's trustworthy, Google's probably on the list. Federated learning could be deployed elsewhere at places we wouldn't necessarily have the same trust for.

I do think this is really cool though. Their implementation that's live is basically with this, I think it's called the Gbar, or something. It's the self-recommended typing and also suggest you things that you might be looking for. That is trying to learn context. For example, if I was at home and I search for restaurants, maybe it knows my history, I go to Michelin star restaurants, I’m probably planning something, I see my anniversary coming up, there's all these fancy things and says, “Hey, let's take you to a site where you can search menus and book.” That's the scenario when I'm in my living room.

Other than that, I'm driving down the street on the passenger and I'm looking at restaurants, location is tremendously important to be there, and immediacy, whether it's open or not. Different context, different recommendation would be appropriate. Google would like to know that without having to basically have all of your data shipped off to the cloud. They do the training locally. They try and learn the context by which your searches are valid and they send back only the pieces that updates that are better than the model they already have.

It's a neat idea. We should paste more attention to it in terms of scrutiny and privacy and what the possible hacks are here, but this is very exciting stuff. Is there a future where ML is doing things in the blockchain? It's possible, but there are some challenges and issues. The use cases are not obvious. The example from DanKu tells us that the overhead and the cost of mining and all these sorts of things make it impractical as is today for anyone to really use the Ethereum VM, or anything like it.

Training offline simply has orders of magnitude, cheaper costs. Even though that can maybe be improved with time, it seems like such a ceiling. I don't know that we're ever going to meet in the middle there. I do think there are some practical cases, probably some very interesting corner cases that this is perfect for. I really like that medical one. I would love to be contributing a lot of health data to researchers if I could – if they wanted it for starters, and then if I could trust that it was not being saved, or labeled as me, or someone else is going to get access to it. I know very well that those aren't commitments anyone can give me today.

Machine learning generally requires some finesse, not a sledgehammer, this idea that will train models on encrypted data and they'll just work without knowing what the data is going to look like. That can work in some situations, but in general, a lot of the novel machine learning problems that people work on require a lot of finesse. You have to get to know the data. There's this famous quote that data scientists spend 80% of their time cleaning the data. While that might be true and it sounds negative, you're also getting to know the data at the same time and you're generating the insights required to build a good model. That goes away if everything's encrypted.

There's some large class of interesting problems that can't be solved in this way, but certainly a class of problems that can’t. Homomorphic encryption is a necessary step. It's very slow and it's not clear that we'll ever have true full homomorphic encryption, so that could be a full stop. There's a lot of other overheads, just the delays in blocks being updated, not getting in to change, these sorts of anomalies and just issues with the blockchain that are pervasive elsewhere. That will probably get solved with time. Any machine learning on that system is going to suffer the same things until solutions arrive.

The transaction delays and those other known issues that come up, those will hurt model training and today model training can be a very slow process. Machine learning people don't want slower process. That puts a little bit of a challenge on it. I think there's hope and interesting things, but in corner cases mostly the opportunities I really see is in more federated data. I love the idea of deploying the model, someone running it locally and maybe even approving what changes they're willing to send back and that we can all globally in this distributed fashion make improvements to a model we all benefit from.

I also think there's some neat stuff with what I would call data stewardship. Like I said, I'd be weary to give my data to some hospital, even though I want to, because I don't know who else is going to get a hold of it. What if there was a middleman I trusted, and they just managed my requests and they took the code from the hospital, and they took my data and they ran it and they were as impartial to it as Amazon is impartial to what I store in S3? If I could trust that someone was a steward of my data, in that way, I think it could advance this idea of federated learning.

Last but not least, I have really undersold the ability to have data provenance via the blockchain, to track where data came from, who has seen it, what was done with it. There are some neat opportunities there that I haven't seen too much research on. That's my take on where we’ve come from, where we might be going and machine learning and how it's going to relate to the blockchain if and or maybe. I feel more of me, that's me on Twitter, we do a weekly show all about it. It’s just said machine learning and stuff.

We can do questions, or we can move on.

**[0:34:25.9] JM:** Yeah, let’s do questions.

**[0:34:26.5] KP:** Cool.

**[0:34:27.2] JM:** I'm going to start off. There is some ways of solving that hospital problem that you're talking about today and where you do that KN on enmity, where if I want to give my dataset to a hospital and I've got a bunch of friends that also want to give their data to the hospital, we all want to do that, and we can do that if we can anonymize the name field and anonymize the other fields of that data to make it sufficiently anonymous, so that the only thing that is not anonymous is our gender, our age, and whether or not we have some type of Leukemia, or whether we develop some type of leukemias you can start to develop correlations there.

The problem is and this has happened with the Netflix prize, the whole Netflix public data set thing, where people took this public dataset and they found – they made their own algorithms around it and they found what – they found a better model for recommending movies. The unfortunate side effect of that was that data set got de-anonymized, because people cross-referenced it with well, would like I am –

**[0:35:43.3] KP:** Something like that, yeah.

**[0:35:44.4] JM:** IMDB data –

**[0:35:45.1] KP:** Very clever idea.

**[0:35:46.2] JM:** The IMDB public accounts got a mapped to private Netflix data. Then some people found out about it. My long-winded way of asking, how well developed is that field of K anonymity?

**[0:36:00.7] KP:** It's developed enough that the experts will probably tell you there's no such thing as anonymity in that respect. There are just techniques that get us a little bit closer. Even my example of the drunk driver thing, that's very anonymous, but there's something you could do to back into that, in the same way people backed into IMDB. That was just a clever isomorphic mapping thing.

There's techniques I've seen where on facial recognition algorithms, this one guy was able to actually recover some original faces that were trained on it, by doing an adversarial approach. Information is weird and it gets embedded in things in unexpected ways. What might seem anonymous is only anonymous until a very clever person figures out how to make a reverse direction of that function.

**[0:36:44.5] PARTICIPANT:** I heard a little bit about differential privacy is we need to safeguard information that you can do queries on a dataset and get data on the aggregate, but when you try and get any data on an individual, there's too much noise to get anything valuable. Was that related to the encryption that you were talking about, or is that a separate path?

**[0:37:04.5] KP:** It's a related idea. I left it out just in terms of time. That Facebook has been pushing that label. Google has the similar idea in federated learning. It's a nice concept and it works to a point. If you average out data, you put a big pool of people and you stir them all up, that does contribute a great deal of anonymization. Aggregates hide things, but they don't perfectly guarantee that they've given you privacy. They've given you what looks an awful lot like privacy and may in many cases be so, but there's no proof that says they're for sure private, that there can't be a very clever person that comes along and figures out how to sync it to IMDB, or there's some correlation they can reverse out.

For example, if maybe I can only make queries in aggregate, but if I make a bunch of them in different partitions of the data, now I can divide and conquer. You're never going to get – it's never going to be the perfect thing, like the CSI enhance, where you take this tiny image and you see somebody's fingernails. They actually are working on that and it's a little bit less science fiction. It's never going to be perfect, but in that same way that smart things will learn to reconstruct data. There's always clever people who can try and de-anonymize. You have no guarantees with techniques like that, but those techniques are the best things we know about, so we should be using them.

Cool. Thanks everybody.

**[0:38:37.9] KS:** Good evening, everyone. Nice to see you all. I'm going to take a quick survey just to calibrate where I should be in this talk. How many people feel they are machine learning expert in the room? How many people use machine learning at all in your job? Some people, a few people. Okay, how many people know very little about machine learning, you feel? Okay, that calibrates me, I think. Okay, very good.

All right. Today, I'm going to talk to you about machine learning at Second Spectrum, how we use it to solve problems that we're interested in. I missed, how many people have actually heard of Second Spectrum? A few people? Okay. Did you hear about it through just podcast? Okay. That also works.

Second Spectrum is a sports analytics company. Our goal is to revolutionize sports through intelligence, that's our tagline right here. What does that actually mean? Well, people – we're looking to change the way that people will play and watch sports, and think of all the people that play sports, that includes teams, leagues, individuals as well, of course, but we're more focused on teams and leagues. If you look at the people that watch sports, we might be able to influence them through the media that is presented to them, or directly to the fans themselves. Those are the areas that we're interested in.

Focusing on teams and leagues for a moment, right now we're the – as of this year, this season, we're the official NBA tracking and analytics provider. We have cameras installed in all 29 NBA stadiums throughout the country right now that are used in all the NBA games, or at least all the ones in those stadiums, to track all the players and the ball throughout the game. We take that data and build things on top of it, that we used to make useful tools for teams and for fans.

One of the things that we do for teams in particular, so we provide a lot of things for teams. We make reports for them, we give them tools to help them rank players in certain categories, a few of the things that they like. One of the tools that we provide to teams right now is the ability to ask questions and answer them in ways, who's – what's the most common pick and roll combination that the Golden State Warriors run and what's the best – what's the most common effective defense against that, for example?

At any rate, those are things we can ask them. Then on top of that, one critical tool that we provide them is the ability to query video. We do that by indexing the video by the content of the video, what actually is happening there. What I'm going to ask you to do here is to watch the video, and in your mind index what you think is actually going on in this video. Just how would you tag it, such that it could be found inside of a database?

Now, maybe I should preface that by how many basketball fans do we have in the room? All right, a few people. Okay, good. Especially you, there we go. All right, what’s going on, the Golden State brought the ball up, they're passing the ball around, come to the top of the key, there's a pick set there, no one's open over there, okay throwing off to the left and take a shot, there we go. All right. Oka, now you have in your mind how you might have – I mentioned a few things, maybe you found a few others, maybe you didn't.

This is what a computer might see. Ah, there's a drop screen over there. What cut point on the top? Some space inside of the corners, the lower right, so you got coming off a wide pin. There's another whip cut in the middle. Same spacing, getting off ball defense, holding ,shooting a jumper and voila. Okay. How many of those things that I talked about did you know? Not too many.

The fact is that, I don't know half the things I talked about. I really don't know. I don't know what the terms are. Those are things that NBA coaches and their staff are interested in a lot of those terms. We've built a computer that can really understand what's going on the floor, “understand.” I use a very anthropomorphic term, right? I understand, “understand” what's going on in the floor. The level of what an NBA coach is interested in.

We can then use any of those things to index the video, and that's been a really great tool for us and it's the reason that teams really like. They used too, and some teams still do have rooms full of interns that sit around and watch video all day long. All they do is watch video and cut up interesting things and label them and put them together and provide them to the coaching staff.

That they spend hours and hours and hours just doing this. Now, if you're an intern who likes and has passion in basketball, maybe that's not a bad job. In our case, we can do all that in a few seconds. Go on.

**[0:43:21.9] PARTICIPANT:** One point. Erik Spoelstra, the coach of the Miami Heat, that was his first job was going through all the tapes.

**[0:43:30.4] KS:** There you go.

**[0:43:31.1] PARTICIPANT:** I guess, these days you guys, I guess, also you can get a text dump of what happened in that scene, right?

**[0:43:37.8] KS:** Absolutely, yes. I mean, that's what our –

**[0:43:39.9] PARTICIPANT:** I guess, in some ways your system doesn't replace that job and just – it keeps those interns a lot and they're working.

**[0:43:48.6] KS:** Well, okay we provide tools to the interns that also the interns, or whoever, the coaching staff really at this point, that let them ask queries, that say for example, I want to know all a coach wants to know, all of the pick-and-rolls that involve Steph Curry and Harrison Barnes, whatever, some to two players on there. We have we have a system that you can put those two things in and it will just pull up all those videos immediately. You don't need to watch through the whole game. We can give you all that video for the whole season as of right now.

One example is this one right here, where I did a query where I said give me all of the passes over 30 feet that ended with a lob. Now, if you ask anyone else to do this, there's no data out there that really has this. The data is solid within us right now. We're not the only ones that have it, but it's rare that you have the length of a pass in the data. We have all that. It's rare that you can actually sequence these things and ask the question and produce this video. Now this one's obviously not something a coach would really be interested in. It's more interesting for someone like you or me to actually watch the lobs and the jams.

The coaches can't ask very similar questions and get answers pretty much immediately, or they'll most likely that's a coach, not to coach themselves, but someone on their staff that can ask those questions. The interns are, they're still there, but they're doing other things now. All right.

Going on to media and fans, I could talk a little bit about that, but I'm just going to let you watch a video. Then I will actually get into the actually the machine learning stuff in a moment, but this is just one last spiel and then we'll actually talk more about the algorithms and stuffs that we do. How is air going to affect basketball? Some of the things that you'll see and actually in this video, you'll see in the video at the very least, and may actually hearing the video, are some of the ways that we are hoping to affect the fan experience. I can actually just talk over this, if I need to. All right, we'll do that.

I'm not as a low point, as our CEO is talking over this thing. Generally speaking, when you're watching a basketball game right now, for example, or any sports match, you are given – everyone watches the same thing. You're watching it on ESPN, you're watching – or some local Fox sports, or some local network. You're watching what they want you to show, and that's fine. Up to a certain point, that's great. What if I want to do a little more? I want to know what's going on in the game. I want to be able to get a little more information about what's actually happening. I might want some statistics while the game's going on. What's the likelihood of certain things happening?

I might want to track my fantasy team, again see what's going on in there, I maybe have a new player that I'm interested in watching. Maybe I want watch my kids and I want something fun on the screen. These are the kinds of things that we can start to do once we have tracking data, and once we actually understand what's going on inside the game. It gives a lot of possibilities for personalization, that suddenly everyone can watch the game in the way that they want to watch it. You can get highlights in ways that you weren't able to get before, and we can provide all that with the tools that we're doing right now.

That's a rough estimate of what was being said in the actual video. Here we go. That's really those two things, you know working with teams and providing them with tools and ways of preparing for games are the kind – then trying to provide an experience for users that allows them to get an – have an individual experience, their own personalized experience for watching a basketball game, or some other sport, are really the target. These are the things that we're working on right now.

The thing about that is that it's all driven by machine learning. Our tagline is revolutionize it. We want to revolutionize sports through intelligence, let's actually get into some of the intelligence, at least at a high-level; our technology stack. Starting at the bottom, we work on in all these areas. We start with player tracking, so I mentioned already that we have cameras set up in all the stadiums, we also have a few college stadiums and a number of football – sorry, American football, soccer, fields, pitches, whatever. Yeah, for soccer stadiums as well, where we have cameras set up to track the action that's going on right now.

If all you had is player tracks, that isn't very useful. In the early days, the early days of tracking, which was only a few years ago, the NBA that's pretty much what they were given. They're given a pile of X-Y coordinates for their players and X, Y, Z for the ball and said, "Okay, maybe a few things like a shot happened here, something else happened here," and that's all you get. That's not that interesting, not that useful, not yet anyway.

One of the things where we really made our mark initially was in adding the semantics layer, the stuff that I started talking over the video for before, where we went add a lot of information about what's actually happening on there. That lets us do – that lets us move forward to the other places. That's what allows us to make it useful for teams and coaches, and where allows us to build these video augmentations and streaming platforms and make the data actually useful for either teams or for consumers. Along the way, we've built up competencies in various areas, so we have a computer vision and machine learning, which are the two things I'll really focus on after this.

Some amount of AI, which is for an in-store generation, mostly this is targeted at the consumer applications is how do we take this data and present it again in ways that is useful for telling a story, to make it interesting for users. When we're generating the augmented reality, we need to have some computer vision and ways of projecting that information onto the screen. Finally, we're working largely without outside vendors, but developing our own competency in streaming video streaming.

Focusing on player tracking for a moment, for a moment; now we start with a stadium. We have cameras set up in stadiums. Our pipeline looks something like this, the raw video comes in from cameras, we're using computer vision IP cameras, mostly it was up until recently, those have mostly been used for industrial applications. They work pretty well for our location as well. Those are trained over the court. We take and encode the video, chunk it up and send it up into the cloud, S3 in our case, although it's not – there's nothing, restricted to that.

As the data is uploaded, we use RabbitMQ in most of our pipeline actually right now. Something very simple messaging is all we need at this point. Send a message saying, “Okay, this is now available. This data is now available. Start processing it.” Messages get sent over to our CV tracking pipeline, which I'll go into a little a more detail in a moment. It says, “Okay, start processing this data.”

CV pipeline reads video and data from S3 and writes back results, and the whole pipeline works this way in a very modular way. We get that repeats over and over again, and we get player and ball tracks are made available. At this point, about 75 seconds after they appear. It still takes a little bit of time. It's not close to – not quite real-time yet. That was as of a little – as of recently, why the light turn. That has actually been coming down slowly.

One of the one of the things that we're really trying to do is make this as real-time as possible. A couple of months ago, that was at 2 to 3 minutes, and now we're down at 75 seconds and we're really shooting to get down to just a few seconds. That's changing the way we're thinking about the problem, trying to move into hardware; we’re going to talking about exploring FVGAs and things like that. Anyway, that's the direction that we're heading right now.

Still even within the current framework, there's still some work that we can do to make it faster than it is right now. The tracking itself, the pipeline is entirely written in Python and C++. Pythons most has – as with many applications, Python is the wrapper and C++ - most of the core algorithms are written in the C++, some on the GPU. We do use open CV, which is a very common computer vision library, if you're not familiar with it, and we also use Tensorflow, which is again, a very common machine learning framework, if you're not familiar with it.

At the same time, about 90% of the algorithms that we use are actually – were actually developed by us. It's not that we're doing – some things that we're doing are really unique, I think. There’s a few tips and tricks and things that we have to compensate. We have the ability to do to make – to make our code really fast, and a few tricks that aren't published anywhere as far as we know that actually help us do things.

On the same time, just being able to even take grills of the common algorithms and speed them up in a way – speed them up on our own, just having control of the source code there, gives us a lot of flexibility in what we do. The point is we have that capability right now, which is the – it's important to us. The algorithms that we – if we take the pipeline and break it down just a little bit, we have a bunch of modular CB algorithms. We have algorithm for player detection and tracking, player identification, of course, ball detection and tracking.

Then implicit in all this is really the ability to fuse inputs from multiple cameras. You need to do that. If you want to find locations, you need to be able to triangulate. This is machine learning, okay, this is the same thing everyone else does. You collect a bunch of data, you split your data up, you train it on some of it, you evaluate it on the rest. The data collection part is labor-intensive, but it's something that we've taken the time to actually do.

We're going to explore with the computer – in the computer vision area, we look at that, we look at the areas that where we're having problems. We have tools that say, okay, this is where it got right, this is where you got it wrong. I can go right to the video there and say, “What actually happened here? What was going on here? Oh, there was an extra shadow over here. Oh, he stepped out of bounds. Oh, his head looks like a ball.” It happens. We repeat trying new things.

We're very data-driven and we're very – we have a good testing infrastructure set up, where with every change that we make to the computer vision code and with – and at the very least, also every week, we rerun all of our testing, test games to see what effects any changes that may have had on our results. Have they made it better? Have they made it worse for some reason? If they get it worst, let's figure out what's going on.

In most in most cases, it’s either staying the same, or it's improving and we're always going through that. Our first goal was to make it as accurate as possible, and right now our accuracy is quite high, 98-ish% accuracy, or more depending on the stadium, the game, the so on and so forth, which is good enough for the rest of our tasks. It gets the job done right quite well. Now we're focusing a lot more on making it faster.

When it's done right, this is what it looks like. Obviously if you're on the podcast, you can't see this, but you can see the ball going in there. It's moving a little faster than real-time. We do the same thing in soccer. It's essentially the same pipeline for soccer, and essentially this is much of the same code as well. We can do the same thing with hockey. We can do the same thing on your back basketball court, if you happen to have multiple cameras set up around your court, which we did for that exercise.

As an aside, our CEO did give a TED talk, and that video is actually one that was used, was prepared for and is used in the in the TED talk, the last one; examples of some issues that we've run across. One of the biggest things, a lot of the work that we do, if you look in the literature for tracking for people tracking, you're not going to find much in the area for sports, at least when we we’re looking at this a couple years ago, last time I looked, there's not much for sports. There's a lot of things for pedestrian tracking.

In pedestrian tracking, it's great. Everyone looks a little different, most of the time. On a basketball court, everyone looks alike. Well, there’s some taller, some shorter, some darker, some lighter, but everyone really – they're all wearing the same uniform. Just from that – there's ways of distinguishing them, but at a first pass everyone looks alike. It's actually early on, especially in the development of our system, was actually very common that we would swap players back and forth on the same team, just as they cross from one another, in front of another. That was one of the biggest problems.

At this point, that hasn't been a problem as much anymore. We've improved our algorithms enough, but that was certainly a big thing. Players or balls being included, or missing, this is a common problem. You get a bunch of people under a basket waiting for a basketball, both the ball and the various players. It's hard to actually distinguish them from one another. Bench players celebrating and jumping onto the court happens on occasion, and we start tracking them along with the players and we realized, “Oh, wait a second. Tey're not supposed to be in the game right now.”

Lighting and reflections at various times are doing problems. Again, these are all things that we've mostly addressed and still occasionally crop up. Player heads being attracted to the ball. Yeah, well I heard that this was actually, for this particular game that his head was tracked as the ball most of the game, unfortunately. It happens. It's pretty rare these days. We actually do a really good job overall, and we have ways of correcting that as we go along. That's now a negative turn example.

Moving on. That's our player tracking, obviously we need a little bit more than that. I've already hinted at you that we will – that we have over more than that. Player in ball tracks, then they get passed on to our – what we call our semantics pipeline. Semantics I mean, it's a big fancy word. Really what we're doing is labeling video, or taking it, or any number of other things that you might want to say along the way there.

Did a pass just happen here? What kind of pass happened? I used the simple names in this diagram. You saw the more complicated names. There are at least 14 different varieties of pick-and-roll that we recognize to varying degrees and accuracy. Some of them are very similar to one another. The coaches and the teams don't mind too much if the two varieties of pick-and-roll are very similar and we confuse one for the other. They’ll say, “Okay, that's fine. It's not a big deal.” Okay, this is the feedback that we get from teams that we work with.

We’ve spent a lot of time interacting with the teams and making these as useful as possible. When we were – the stack, sorry. The stack that we work with is almost entirely written in Python. We use a lot of libraries that are written in other things. Socket learn is a pretty common machine learning framework that's out there. We use a lot of things from there. We use Keras and Tensorflow, again framework, so I've mentioned Tensorflow as a framework. Keras is a – I think actually – I take that back. We actually use Keras in our open CV pipeline as well. I'd forgotten that.

It’s a meta framework. It makes things easier; Tensorflow a little easier to use. Makes a generic, so you can have – in theory, you can swap out the backend practice. I don't know how easy this really is, but in theory, you can swap out Tensorflow for some other deep learning framework or things like that. Again, one of the things that we've been able to do and has been really important for the tasks that we work on, is we spend a lot of time with the problems itself.

We take whatever knowledge we have of basketball and incorporate those into the algorithms. Often what that means is doing feature generation saying, “I understand what this particular pick-and-roll happens to mean. I'm going to make it easy for the machine learning algorithm to do that by engineering some features that make it possible first, and then easy for the machine learning algorithm to learn what's actually going on there.”

There weren’t a lot of people who are very familiar with machine learning in the audience here, so I'll have to decode some of these algorithms – these names perhaps. We use a lot of different models in our algorithms. Depending on what we're trying to do, depending on if it's working with soccer, or basketball, depending if we’re interested in a sequence of events that are happening, or just looking at a very short small slice of data. We very commonly used support vector machines, that's the first SVM up there, and random forests which are both – each which has their strengths and weaknesses.

For sequential data, we use both hidden markup models and conditional random fields. There are libraries for all these things. If you know how to code them up yourself, you can find libraries that will do this, or you can code them up yourself. In some cases, we've done both. Then we use neural networks for various parts of this, sometimes in combination with one another. Same thing for training data.

Again, because I've talked about twice already now, but we spend a lot of time, especially early on, but even now consulting with teams, okay. They have a new marketing, they have a new something else that they want that we don't quite provide them with yet, that they want us to provide.

Okay, they come with a request. We say, “Okay, let's see what we can do.” We find a bunch of examples of it. What we think our examples, or maybe the teams already provide some of those examples, we see how well we can find those. Then we go and test them. We say, “Okay, these are everything we found. How do we do? What do you think? Is this useful to you?” They come back with some feedback. That's a relatively slow loop. We do a lot of iteration on our own just to find what we can.

Some of the things that they found are pretty esoteric, that might happen once or twice a game. Those are hard, honestly. It's hard to train something that does that. We spend a lot of time and we do those to varying degrees of success. Well, as before, lather, rinse repeat, do it again, collect more data, consult again. We end up with something that – with again, this is the same video as before, which coach shows all of the various actions that are going on the court in that video, and we do this for every single NBA game, minus the ones that are overseas, where we don't have cameras, all throughout the season. We just concluded over the season tracking and providing this data for gosh, over a thousand games.

**[1:01:21.7] JM:** What’s that number that’s next to the circles?

**[1:01:24.6] KS:** Good. Thank you for asking that. That's actually our model for shot probability. There's no ground truth for that, but you can tell when – just from the data, you can extract whether, or how frequently shots are made from that place on the floor, using different – depending on the configuration of the players, or things like that.

**[1:01:41.8] PARTICIPANT:** Is that per player? It’s like Golden State or whatever, those guys you just see them knocking down.

**[1:01:47.9] KS:** Right. We have models that are both generic, which are still useful for players that don't play very often, and player-specific models for players that play often enough, where we –

**[1:01:55.8] PARTICIPANT:** The Steph Curry mod, the Steph [inaudible 1:01:57.4].

**[1:01:57.6] KS:** Exactly. We do have a Steph Curry power building, which is really rather high. Yes. Yes, we do.

**[1:02:03.4] JM:** Sorry for asking that, a little big question, but for all the noise that you could have added to this very clean, make sure want to do that shot probability of those people.

**[1:02:15.4] KS:** Those are the people that were open at that particular time, perhaps.

**[1:02:18.5] JM:** No, I'm just saying you could've added other steps, right? Like –

**[1:02:22.5] KS:** This is the one that's actually the most easily interpretable. I mean, there's all kinds of other stats that we can do. We can do a passing power ability, but yeah, who really cares about that really. Even on the teams, this is – we want to know what really teams are interested in scoring. This is basketball. We want to know what kind of quality – how was the quality of the shots that that players are actually doing? We have models that do that.

**[1:02:47.1] PARTICIPANT:** Because they can figure, mainly they put another second defender on the guy outside the line.

**[1:02:52.6] KS:** You do that, the shot probability changes if you had a second defender.

**[1:02:55.2] PARTICIPANT:** Steph Curry is going to knock it down anyway. They'll block.

**[1:02:58.6] KS:** That's also quite possible. At that point, someone else is open and their shot probability is higher. For a team like Golden State, for example, which is a very good basketball team, just passing it to other players actually may be the better option in that case. Yeah. That's the general overview of what we do in machine learning and computer vision at our company.

Now, just to be clear at our company right now, we have probably 50 or 60 engineers, or so. Only about 20% to 30% of them are in machine learning computer vision. We have a lot of full-stack engineers, people doing UIUX infrastructure and DevOps and we're always looking for good people. If you, or if you know people who are looking for jobs, do feel free to reach out and contact us. Let us know what's going on. You can send an e-mail to work@secondspectrum, where you can check out our website for things that we're looking for.

I will end with, and it's really too bad. If you don't see, or hear the audio for this. It's actually fun, but these are some of the things that we do. Up until now, almost everything's been basketball, but it does actually has a backdrop. It actually works pretty well for asking questions, and we're going to stick with that.

We do work in soccer. Maybe just add to that, the soccer is I think it’s a concept video, but they can use based off real data, but then we've also worked with Sky Sports in Europe, the ESPN of Europe and provided videos for them to use in their broadcasts. We've worked with ESPN here, especially in basketball and continue to do so, to provide them with various things. We've done some work in football. Nothing that's gone on the air at this point yet, but these are the kinds of stats and stuff that we've been able to get from the data that we've had access to there.

Then the ones at the bottom are basketball. These are the more fun things that either we can do, or have ambitions to do. The one on the lower-right is actually – it's not exactly the same these days. That's a little bit of an older video, but we actually have a consumer app that we're developing, that doesn't do exactly those things, but the ideas are the same, that you can take the data that you have right now and provide more information. I showed some of that in some earlier videos as well. Any questions?

**[1:05:17.5] PARTICIPANT:** How do you guys deal with like confidence intervals? Because you have that shot probability, but depending on how confident you want to be dealing with, like 95% or something.

**[1:05:28.6] KS:** How do we deal with confidence intervals? The answer is we really don't right now. The model, we take a measure of how close we are. The answer is we don't. We actually don't do that right now. We do set a minimum for the number of shots you had to have taken from a particular part of the floor, or we don't even bother trying to calculate it. Once you're above a threshold for either total number of shots, or our threshold for shots in that particular situation, then we can start doing something.

Yeah, I don't know what the confidence interval is, or the actual cutoff is. I haven't seen that part of the code, but I know that that's what we do. If you have someone who only dunks the ball, DeAndre Jordan works is still in the Clippers. He's the only Clipper that's in those videos that's still left on the Clippers right now. We have no way of calculating a shot probability for him beyond 10 feet away, maybe even five feet away, okay, because he doesn't take shots out there, things like that. We can calculate it, but it's meaningless. I guess, the answer is that we deal with it by not calculating it when we don't have enough data. Other questions? Yeah.

**[1:06:31.3] PARTICIPANT:** You mentioned that you have ability to query videos. What does it mean, you actually have an API? You have some kind of DSL company?

**[1:06:39.1] KS:** Right now when I say – so good. Question was we have an ability to query videos. How do we actually provide that? This is something that's provided to teams, and the places like ESPN, or other media outlets right now. It's really just done through a web interface, so we actually control the query entirely on the back end. They do it through some web interface where they specify the kinds of things that they're looking for, and we take that, turn that into a query into our system, go get the data associated with that, and then those have links to – the data that we have they're all aligned into the video using the techniques that I just talked about.

**[1:07:17.1] PARTICIPANT:** You've mentioned developing some in-house algorithms. Can you talk more about the motivation for that? What were some of the sure things, like OpenCD and SK Learn that you also do.

**[1:07:26.2] KS:** Okay. The question was, what was the motivation for developing in-house algorithms as opposed to simply using OpenCD, or SK Learn and things like that? Part of the motivation is just the – having the ability to incorporate domain-specific knowledge into those algorithms. I mean, let's see what's a good example of that.

One example I talked a little earlier about feature engineering. That in our case, we have a lot of people that work for the company who know a lot about basketball, and we can also draw on the knowledge of teams. We draw a lot of the knowledge of teams in order to generate the features for that. That just feature engineering. That doesn't that doesn't change the model itself. The fundamental model still the same, it's just what features you have going on.

Regarding other reasons for doing that, we found in some cases that the open source models are complete, but they're not very fast, and this gives us – even if they're open source, we couldn't in theory take their source to figure out what it's doing, try and make it faster. We've done that at times. I've taken some open source projects and added parallelization to them. Then spin them up a little bit. Then we threw that code away, because we moved on to something a little better. That's fine. It was it was good work at the time to do and we found a better algorithm, or developed [inaudible 1:08:43.8] off on our own.

In many cases, we we're familiar with the code base, then we can actually go in and modify it and change it in ways that's useful. I guess, most importantly, we have the expertise in-house to do it, so why not do that? Let's take advantage of some of the people that we have who are really good at coding some of the stuff up.

I won't go into too many details, but for example even in computer vision, we know we're looking at a basketball court. We can use that, we know about the dimensions of the court, or other things like that, we can use that knowledge to make our algorithms simpler and faster, saying we know we're not going to see a generic something other than this court, so I know the court I'm looking at right now. That can make my life a little easier.

I might not change the code to that, but I can code – I won't change the code for every court, but I'll change the code to take into account the fact that I have a court and I know what that court looks like, things like that.

[END]