SUPER DATASCIENCE
MAKING THE COMPLEX SIMPLE

SDS PODCAST EPISODE 79 WITH BALARAMAN RAVINDRAN

Kirill: This is episode number 79 with Associate Professor of Reinforcement Learning, Balaraman Ravindran.

(background music plays)

Welcome to the SuperDataScience podcast. My name is Kirill Eremenko, data science coach and lifestyle entrepreneur. And each week we bring you inspiring people and ideas to help you build your successful career in data science. Thanks for being here today and now let's make the complex simple.

(background music plays)

Hello and welcome to the SuperDataScience podcast. Today I've got a very interesting guest, Dr Balaraman Ravindran, who is an Associate Professor of Reinforcement Learning at the Indian Institute of Technology, or IIT, in Madras, which is also known as Chennai. This is in India, and I am very excited to have Ravi on the show because I have personally learnt a lot, a ton, from Ravi's online videos.

So what you need to know about Ravi is he teaches reinforcement learning at the Indian Institute of Technology in Madras, and at the same time, sometimes he records his lectures, which are actually running, so you can see him teach to students, and then it's published on YouTube and all these lectures are free, you can access them. And it is so engaging, and so inspiring, to sit and watch these on your computer, and it feels as if you're sitting in the classroom. And personally I've learned a lot of different topics, or upskilled myself in a lot of different topics, such as Thompson sampling, or Upper Confidence Bound, or Q learning, and all of these other areas, other aspects of reinforcement learning because Ravi has a lot to share in those areas and his teaching style is very, very good.

So even if for some reason you don't get to the end of the podcast, or you don't have the time to listen to everything here, I highly encourage you to check out his lectures, and we've included the links to Ravi's repository of lectures on the University website. So you can find that link at the show notes for this episode, which are at www.superdatascience.com/79. So if anything, I highly encourage you to go there and click that link and check out some of these videos so you can upskill yourself in reinforcement learning as well.

In this podcast, we talk about Ravi's journey, how he studied in India, how he then went and did his PhD in the US at the University of Massachusetts, was supervised by one of the founding fathers of reinforcement learning, Andrew Barto, You'll also find out what reinforcement learning is, and how it's different to supervised and unsupervised learning, and how it's driving progress that's happening in artificial intelligence.

So all of these things you hear about artificial intelligence, well most of the time, reinforcement learning is at the heart of it all. Plus we'll go into a bit more detailed reinforcement learning, you'll find out what types of reinforcement learning exist. So it's a great podcast if you want to get an overview of the world of reinforcement learning and get to know it a bit better.

And on that note, without further ado, I bring to you Dr Balaraman Ravindran from the Indian Institute of Technology in Madras.

(background music plays)

Hello everybody, welcome to the SuperDataScience podcast. Today I've got a very special guest calling in from India, Balaraman Ravindran, who is an Associate Professor at the Indian Institute of Technology in Madras. Ravi, welcome to the show. Thank you so much for taking some time to come here.

Ravindran: Thank you for having me on the show.

Kirill: It's so awesome, it's so great, to hear you and to have just now seen you. It's a great honour because, for those who don't know among our listeners, Ravi is an Associate Professor, and he not only teaches at the university, but you can also find a lot of his videos online on YouTube and other places as well, and I've personally seen so many of Ravi's videos. Ravi, I've watched your video on Upper Confidence Bound, on Thompson's sampling, on reinforcement learning in general, I think you have some videos on Q learning, and they've really helped me personally understand those subjects better and shape my view of what reinforcement learning is. So first and foremost, thank you so much for sharing all those great videos and all that content online.

Ravindran: You're welcome. I'm really happy that it helps.

Kirill: And can you tell us a bit about your background? You've got such an interesting background. So you're originally from India and you did your Bachelor's in India. Where did that take you, and how did you get into the field of reinforcement learning?

Ravindran: I did my Bachelor's actually in Electronics and Communications Engineering, it's not in Computer Science at all. But while I was doing my undergrad, I started getting into studying AI and especially the question of how do humans think and reason, embark on everyday problems. And then that first took me to neural networks. It was back in the early 90s, so neural networks were kind of hard, but on their way out. People learned about the history at that point.

But then I kind of felt a little disappointed, because at that point, neural networks were kind of far away from the biological motivations. Then I joined the Master’s program in the Institute of Science Bangalore, where I started looking at these issues in greater depth and that’s where I first came across reinforcement learning. Ironically, I came across this very much in a biological context. I looked at experiments that people have been doing on monkeys where they had used temporal difference learning rule to explain how dopamine, a neurotransmitter, or neuro-modulator, varies as monkeys go about learning different problems, learning to solve different problems, and then they had good mapping to what reinforcement learning algorithms predicted the variations should be.

So this got me really excited to say, “Okay, here is something that really explains how humans learn.” And then I started working on reinforcement learning. In fact, I was the first student with my adviser who actually looked at reinforcement learning. And then I really wanted to continue doing my PhD in reinforcement learning. So I applied to UMass Amherst, which is where reinforcement learning actually started.

My adviser, Andrew Barto, one of the founders of the field of reinforcement learning; he and his students wrote the first paper on reinforcement learning back in '83. Then he and his student, Richard Sutton, who's the student who originally proposed the temporal difference rule, they wrote the book on reinforcement learning. So Richard and Andy are widely regarded as the pioneers of the field of RL, so I really wanted to work with them and I was really, really fortunate that I managed to join my PhD with Andy. That's how I got into reinforcement learning. So I've been doing reinforcement learning since '94.

Kirill: Yeah, and that is a fantastic story. Just for our listeners, I want to emphasize how—this is just crazy for me, listening to this, because the book "Reinforcement Learning: An Introduction" is written by Richard Sutton and Andrew Barto and I've been through this book, through many chapters of this book, I think I actually remember chapter 7 is on – what's it called – the temporal difference or something like that. Every chapter has a separate huge part, it's a huge book, so it's just crazy to hear that the author of this book, Andrew Barto, there's a Wikipedia page about him, he is one of the founders of the whole of reinforcement learning and, Ravi, he was your PhD adviser. It's a wonderful opportunity and I'm sure, like you said, you were very lucky. You probably learned so much from Andrew Barto.

Ravindran: Yeah. Andy is amazing. It's not that he knows so much about reinforcement learning. He knows so much about AI in general and he's a very, very deep thinker, so it's always enlightening to have conversations with him. And as an adviser, he was amazing to work with. Despite all his accomplishments, he never ended up overwhelming the students. He gives you enough room so you can explore and go out and do whatever you want. He never forces you to work on his ideas. Amazing advisor to have.

Kirill: Fantastic. So, you did your Bachelor's in India and then you did your Master's and PhD—

Ravindan: I did my Master's in India and the PhD at UMass Amherst.

Kirill: Okay, so PhD at University of Massachusetts. And where did that take you from there?

Ravindan: I actually took my time finishing my PhD, so at the end of it I thought I was ready to take up a job and I've always wanted to be in academia because I'm a third generation teacher from my family so I was applying, looking for academic positions. And luckily enough, a position opened up in the IITs back in India, so I applied to IIT Madras and IIT Bombay, and I got an offer from both places and I chose to come to Madras. So I've been in IIT Madras since 2004.

Kirill: 2004? So you've been teaching at IIT Madras for 13 years now.

Ravindan: 13 years, yeah. And doing research and building a fantastic group here.

Kirill: That's so great. And how many students have you taught so far in 13 years?

Ravindan: Taught? Here we have a good amount of undergraduate teaching, so probably a huge number of students. My machine learning classes typically tend to have 100 plus students every year. I think the highest I've had was 160, 170 students in one semester. In Machine Learning. So that would be a large number. I have graduated – about 27 graduate students so far from my group. Several of them with their Master's and a few of them with a PhD degree.

Kirill: Okay, that's pretty cool. So you're teaching, you're running these lectures, which are fantastic, but what made you decide to record your videos and put them online?

Ravindan: There are a couple of reasons for that. You know, India is a very large country, so getting high quality teaching to each and every college and every student is very hard. Recruiting high calibre teachers is hard, so the Indian government started this program called National Programme on Technology Enhanced Learning, or NPTEL. NPTEL started off several years ago, they just recorded the videos and then made the videos available online. But more recently, they started getting into this MOOC more, the online courses more, and we have been running several classes online with a live chat room where students can ask doubts, and weekly exercises which are graded, and exams.

In fact, we offer proctored exams if the students are willing to pay for it. So we do have proctored exams so that the certificates the students get from the NPTEL courses are highly valued. So through the NPTEL platform I started offering these courses, I actually taught three courses through the NPTEL platform. One is Introduction to Data

Science, another one is Introduction to Machine Learning course, and then the Introduction to Reinforcement Learning course, the videos of which you referred to at the beginning.

The other reason was really—when I decided to record these 'Introduction to RL' lectures, there were not that many high quality reinforcement learning video lectures that were available online globally, not just in India. I mean, there were some machine learning and data analytics. We had too much material out there. In fact, we have an overkill of material out there. But for reinforcement learning there were really not many good online courses. I'm really passionate about RL and I wanted more and more people to be aware of this field, so I thought I would do this. Of course, now there are other videos that are available online, not just mine. One of the reasons I decided to do the RL course was because there was not enough material available.

Kirill: I agree that there's quite a lot right now. I did some substantial research into this when we were preparing our courses on artificial intelligence, which just came out recently, and I must say your videos are some of the best ones.

Ravindran: Thank you.

Kirill: My pleasure. You have a great way of engaging students when talking. Like, when I'm watching the video, I feel like I'm in the classroom because you sometimes pause and you ask questions and you wait for the students to answer. And it's actually an interesting experience because the camera is on you, but there's still students in the classroom. You can hear them, so it's as if you are sitting among the students.

And sometimes I caught myself—one time I remember I was watching I think your video on Q-learning, and you were going through something and you asked a question and I answered it, I was the first one to answer, even faster than your students in the classroom. I was like, "Yes! I got this one." So that was really fun.

Ravindran: It helps because I'm actually teaching the class while we're recording the video. You know, it was a little bit of a challenge to begin with, so I had to kind of curtail my movements in the classroom so that the camera could track me easily. So there were a few things, but it was actually the class that I was teaching during the semester that we were taping, and then we went back and did some editing to fit it into the MOOC format. So it's not like I did a standalone recording for the online course, which I did for the Intro to Data Science course. I didn't like the way the videos for that turned out. It kind of changes your style of teaching if you don't have an audience in front of you. When you teach to the classroom, there's this two-way communication going on and that helps a lot.

Kirill: Yeah. And I guess everybody has their own preferences and style. That's something that you're very good at and you can tell from the videos that that's something you're really enjoying when you get into it. What I wanted to ask you was — so, you're teaching reinforcement learning. Can you tell us a bit more about what reinforcement learning is? You're obviously so passionate about this field. There are so many listeners eager to find out more about it. Can you give us a quick intro to reinforcement learning and this whole area of analytics?

Ravindran: Sure. So, many of the listeners would probably be familiar with learning problems like classification or regression. So where you're given kind of a labelled dataset, data is given to you offline, and then you build a predictor that tells you what the labels should be for an unseen dataset that's not part of your training set. So the whole problem here is one of generalization, to go from data that was given to you during training time to being able to predict labels for data at test time.

But if you stop and think about how you did learning yourself, let's say how you learned to ride a bicycle, it's certainly not through supervised learning. If somebody just sits there and tells you "Push down your left leg," or "Push down your right leg." "This is the thing that you have to do when your bicycle is tilting so many degrees to the vertical and you're moving at so many meters per second," and so on. We don't even process those kind of information when we're trying to learn to cycle. You just get on the bicycle and then you do a whole bunch of trial and error learning.

The kind of feedback you get is what I would call more of an evaluative feedback. So there might be a parent or an uncle or some family member standing there and telling you, "Hey, be careful! Don't fall of the bicycle. No, don't do that." So those are the kinds of things. You're basically evaluated on what you do. You're not instructed really on what you should do. So reinforcement learning is all about learning from such evaluative feedback of what you have done. So you're given an input, you respond to that first, and then you get an evaluation of your response and then you are learning to improve your performance based on the evaluation you’re getting.

So the computing field of reinforcement learning is essentially looking at mathematical formulisms for this kind of a trial and error learning problem and coming up with efficient algorithms for solving this. So one way of thinking of what reinforcement learning actually does, is tell you how to change your behaviour in response to the evaluation that you’re getting. That’s what reinforcement learning algorithms do. You mentioned Q-learning. That’s one very popular reinforcement learning algorithm. Q-learning essentially does that: you perform actions, you get some feedback which is in form of the evaluation of these actions, and you use that evaluation to change how you behave when faced with similar situations in the future. That way reinforcement learning significantly differs from the other learning formulisms like classification, regression, and classical supervised learning.

Kirill: Okay. So, reinforcement learning is a form of unsupervised learning for machines to learn how to complete a certain task without being explicitly pre-programmed to do them. So we need to draw a line here between you saying you have an if/else type of logic and you tell the machine how to do certain tasks, versus you just give it a certain environment where it can operate and try different things on its own, get some feedback, and based on that feedback, it can learn to do that task that it has, that utility function that it’s trying to maximize. It learns how to do that better.

Ravindran: Yeah. Just one slightly nit-picky thing here: I wouldn't call reinforced learning as being unsupervised because you are getting an evaluation.

Kirill: Yes. I was actually waiting for that because there's supervised, there's unsupervised. I mean, reinforcement learning is somewhere like a third, on its own type of thing, right?

Ravindran: Exactly.

Kirill: Sorry for interrupting. Tell us a bit more. Why is it not an unsupervised?

Ravindran: If you think about unsupervised learning, at least the mathematical formulation of unsupervised learning, it's more about finding patterns in the data that's given to you. So, if you think about something like clustering — I'm pretty sure there's a data science savvy audience, data analytics savvy audience who know what clustering is — it's essentially given a set of data points and trying to find groups of data among them so that the points that belong to the same group are more similar and somewhat dissimilar from points that belong to another group. That's the whole idea of clustering, so it's basically finding patterns in the data that's given to you.

And there's really no decision, no output that is expected from the unsupervised learning algorithm on a per input basis, while reinforcement learning is all about decision making. So given an input, given some kind of a situation that you find yourself in, we expect the agent to be able to give us decisions that in some sense optimize the evaluation. In unsupervised learning it's more about finding patterns in the data. Reinforcement learning agent can use unsupervised learning to find patterns in the input data and then learn to associate decisions with those patterns. But the decision-making part is what we learn through reinforcement learning and that crucially relies on getting this evaluated feedback from the world.

Kirill: Okay, thank you for that clarification. It makes sense. And you mentioned Q-learning. What other types of reinforcement learning algorithms exist out there?

Ravindran: Oh, many. When I typically teach my course, I kind of group them into three categories. So, there are classes of methods that relay on what I call the value functions, utility functions, because value functions essentially tell you what is the expected evaluation you're going to get when you perform an action. So these are an indirect way of learning a solution to the problem, because once I know what is the best possible evaluation I can get, then I can go back and try to recover the behaviour that will give me this best possible evaluation.

These are called—in the literature they're called value function-based methods. They're largely dependent on what are called the temporal difference error, it's called the TD-error. There is the TD-lambda learning algorithm, that is Q-learning, and that is something called SARSA, which is a variant of Q-learning. And then there are actor-critic algorithms which are very popular nowadays.

Kirill: A3C?

Ravindran: Well, A3C is one such implementation of an actor-critic algorithm. They added a little bit more to the basic actor-critic setup to get something that is very scalable and runs very fast. It can be increasingly the algorithm of choice for people doing deep RL, but yeah, there are other forms of actor-critic algorithm. In fact, the very first paper published in 1983 on reinforcement learning was an actor-critic algorithm.

Kirill: Oh, that's so interesting. So it's like history is repeating itself?

Ravindran: Yeah. In fact, this is something that I point to my students. The very first paper also used the neural network. So that really illustrates history repeating itself. And then the second class of algorithms are those that explicitly just search through the space of behaviours. So they don't look to estimated value functions, so there are algorithms called policy gradient algorithms, and one of the most famous of the policy gradient algorithms is something called REINFORCE. It's actually an acronym, don't ask me to expand it, but it was proposed by Williams in '92 and it has made a huge comeback in the new era of deep learning, so a lot of deep learning architectures use this REINFORCE algorithm.

The thing with REINFORCE is it doesn't maintain any value function, it just searches directly in the space of behaviours by perturbing the behaviours and trying to find what's the best behaviour. And then you could also use something like genetic algorithms to search through the space of behaviours. That is the class of approaches that are based on genetic algorithms.

And then there are these more classical ways of solving the reinforcement learning problem when you have knowledge of the system parameters, so when you know exactly how the world is behaving. If you have a closed-form model, then you can try to solve it using stochastic dynamic programming methods. Those have been around forever, they're not new, so you can think of using value iteration, or policy iteration, some classes of algorithms that come from OR, and that can also be used to solve the decision-making problem that reinforcement learning tries to solve.

But I would stop short of calling them reinforcement learning algorithms because they don't have the same kind of interactive feel that reinforcement learning algorithms have. They work, like, offline algorithms that just solve the problem.

Kirill: Okay. So just to recap, we have the value function-based methods, which include the TD-lambda, Q-learning, SARSA, A3C and so on; we've got policy gradient methods, where you mentioned the REINFORCE method from 1992; and we have the genetic algorithms or the evolutionary methods. So those are three main classes and then you have the offline algorithms, where you've got other methods of solving the same problem which aren't really reinforcement. Where would you put Thompson sampling and Upper Confidence Bound?

Ravindran: One way of thinking about what Thompson sampling does is to think of it as estimating a model of the problem. So all of this has to do with stochastic systems, so you could think of first taking your interaction with the world, converting that into a probabilistic model of the world, and then trying to solve the probabilistic model using dynamic programming techniques, offline techniques I was telling you about, and then using that to gather more data from the world. So this kind of iterating approach is what Thompson sampling would do.

So Thompson sampling in some sense gathers data from the world – uses that to refine a model of the world – then solves the model, not solving the real world problem, but solves the model and uses that to go back and behave in the world to gather more data.

Kirill: Okay. So, that would fall into that last category that you spoke about, the offline category?

Ravindran: Partly, yeah. It's not completely offline, because you are going online to gather data, but when you are actually solving the problem, you are doing it offline with the data that you've gathered already, and then you have an online phase where you go back and gather data again and then you go offline and then solve it. That is the iterative nature of the Thompson sampling.

Kirill: Okay, that makes sense. And could you give us an example of genetic algorithm? I think that's a very interesting field and we actually had a guest previously on the podcast who created her own genetic algorithm based on what she observed from plants and that was very interesting. Can you give us a quick example of a genetic algorithm and how that would work?

Ravindran: Sure. So, in genetic algorithms — I'm not talking about any specific approach, I mean, they have so many of these now, evolutionary programming, evolutionary computing, I'm just talking about it at a very abstract level of how the genetic algorithm setup would work. So you have a solution that is described in terms of what people call a chromosome. This is not like the human chromosome, it's just like any kind of an abstract representation of the solution.

In this case you could think of a function that describes a behaviour. That is your solution. It could be—let us say I'm trying to solve a small navigation problem, so I'm a robot that's supposed to get around in a world, in a world space. So I could decide to go up, down, left, right and then I could orient myself in some direction and go forward in that direction as well. So that's my behaviours, right?

So the genetic algorithm is going to say, "Okay, here is a function that describes how you should behave given a specific input. So I give you the reading of all your sensors and this is how you should behave." That's a function that's going to tell you about it. And then we use your usual genetic algorithm operators like mutation and crossover and so on and so forth, but then you try to perturb this solution, come up with a new solution which is essentially like the chromosome of one behaviour kind of crossing over with the chromosome of another behaviour. So you could say that, "Okay, if you are in the lower half of the space, you use policy 1; if you're in the upper half of the space, you use policy 2." So this could be one way of mixing the two up.

And then you come up with a new solution and then you go ahead and behave according to the solution and see how well you performed. And like that, you generate multiple solutions by doing this mutation and crossovers on your existing set of solutions. And then you use your reinforcement learning algorithms to evaluate how well you are doing and use that as your reward, what's called the fitness function. And then you keep searching through this space of policies by randomly doing these mutations and cross-overs until you're satisfied with the fitness of the solutions that you achieve. And then you stop and return the solution with the best fitness as your policy.

Kirill: That's interesting. It's actually very different to Q-learning.

Ravindran: Oh, it's very different to Q-learning.

Kirill: Okay. And in Q-learning you explore the environment and you try to model what kind of reward to expect of performing certain actions. And once you've explored enough, you can start to exploit that and you can start to look for the best chains of actions to take in order to maximize your reward.

Ravindran: Yeah, that's essentially what it is. So reinforcement learning is one of those value function-based methods where the reward expectation is essentially what's called the Q-function, so you're learning the Q-function it became Q-learning.

Kirill: Okay, interesting. And I'm just trying to get my head around this — would it be correct to say that in Q-learning you explore the individual steps and then you adjust your algorithm or your policy, how the agent, let's say it's a robot that's navigating a room and needs to not bump into the walls and needs to understand what actions to take — just so that the listeners have something in mind what kind of application we can talk about here — and then it gets to a door and it doesn't know what to do, it bumps into the door and then it realizes that if it opens the handle, then it gets a reward because it gets into a room.

So that robot, or that artificial intelligence, is going to be adjusting its policy of actions that it's taking so it's going to be making the choice of actions that it's taking along the way. So it's going to be, like, halfway through the solution, that actually this action is better and it'll adjust to that. So that's kind of Q-learning.

Whereas in genetic algorithm – this is the way I understand it, correct me if I'm wrong – you're not assessing individual actions, you're assessing the whole algorithm, or the whole behaviour in general, and then you're like, "Oh, no, this one was bad, let's throw it away. Now we're going to go in this direction." It's not individual steps direction, but "We're going to try this whole pattern of events, whole pattern of behaviours." Is that about right?

Ravindran: Yes. The most basic genetic algorithms, of course, do that. There have been recent attempts at trying to do more fine grind exploration using genetic algorithm as well, but the philosophy behind how you do this is that genetic algorithms evaluate whole policies, whole behaviours, while in Q-learning you're exploring at the levels of individual actions.

Kirill: Awesome, thank you. And we won't go into policy gradient on the show here. I'll leave the listeners to go check out your videos on that and they can get some info from there. But what I would like to talk about is application. We've touched on one just now with the robot. Can you give us some applications of reinforcement learning that exist in the real world? Because a lot of our listeners are probably listening to this show and reinforcement learning sounds great, but until we can justify the value that it can bring to the world, a lot of people won't see it necessary to explore this field.

Ravindran: Sure. Some of the most exciting recent breakthroughs in AI have happened due to reinforcement learning. I'm pretty sure a lot of people have heard about the game of Go, which is very popular in Asia and for a long time it was considered a very hard game for computers to solve because the number of possible outcomes are just too many. The game has a lot more variety; in fact, it has a higher what's called a branching factor, a higher set of possible outcomes for every board position, than even chess.

So people are considering it a very hard game for computers to solve, and a couple of years back this company, DeepMind, built a reinforcement learning agent that played Go really, really well. That essentially demonstrated that it can solve very hard decision-making problems using reinforcement learning, even when humans really do not know how to communicate to the program, how they solve the problem. This essentially did not use any human expert input in designing the solution. It just learned to play this game from scratch. And this was just not one example. It's been repeatedly shown that very hard decision-making problems — we can't expect a human to give very detailed instructions because humans themselves don't understand how to solve this problem well.

Another example would be a decade ago or so, we had people train a helicopter, train a helicopter pilot using reinforcement learning. That was for a long time one of the marquee examples of how well reinforcement learning works. And more related to everyday things—you're not going to fly a helicopter anytime soon, but there are things like recommendations systems. So you could think of when a user comes to a website, you could recommend products to the users based on their past behaviour, but then you could soon become pretty focused on just one thing. If you're going to repeat things based on the past behaviour, you really do not know how to vary stuff.

So what people have looked at is they looked at the recommender system that treats the problem as if it is a reinforcement learning problem in the sense that the actions you could take or different recommendations you can give, and the evaluation is, if somebody actually accepts your recommendation, you think of that as a positive evaluation; if somebody ignores your recommendation, think of that as a negative evaluation. That way, you could go beyond what you've seen as established behaviours from people who've come to your site in the past. So you can actually vary your recommendation, introduce more variety into it, beyond what you've seen just in the past behaviour. These kinds of things, again, are something that reinforcement learning allows you to do.

We have done some work on using reinforcement learning for design on loan applications in banks. So there is this tension between gathering more information about your customer base and then there is also this risk of losing money because the person is not able to repay and it's very hard to know where to draw the line. So reinforcement learning methods allow us to do that. And we have worked with banks in India where we have actually experimented with their data, their customer data, and built algorithms that allow them to make better decisions about loans.

But there's lot of work that has been done on experiment design. You have many, many, many different parameters, you don't know which one to choose. One example is — this is actually a scenario my colleagues actually worked on. The city has a fixed budget and they have to design on how they're going to spend it on different kinds of parks. There are a few things they can do in a city park. Which is the right things to do. They can't set aside "Do I put a swing in this park?" versus "Do I strengthen the fence around the park?" There are many ways in which you can spend the money in the park. There are just too many combinations for me to try out everything, so we can design reinforcement learning agents that very quickly narrow down on the right combination of factors that you should be looking at. So that is another place where we could use reinforcement learning.

So from a data analytics point of view, these kinds of work are sometimes called prescriptive analytics. You don't just predict something or describe something. People must have heard of predictive analytics and descriptive analytics. But in prescriptive analytics, you actually suggest decision based on analysis that they're doing with the data and reinforcement learning is one tool to use for prescriptive analytics.

Kirill:	That was a great overview, some interesting applications. So you can use it not just for robotics. For those listening out there, yes, you definitely can use it in robotics and you can use it in artificial intelligence, even the component that doesn't have robotics, so that's the Alpha Go example, how Alpha Go from Google DeepMind beat Lee Sedol, the—
Ravindran:	Lee Sedol was last year. It beat the number one Chinese player. I can look up his name for you, but he beat the number one Chinese player very recently. And they've announced that they're retiring from active competition because—
Kirill:	They're making the game boring. (Laughs)
Ravindran:	Yeah, exactly.
Kirill:	And with that example, that was so cool. It beat Lee Sedol last year, maybe March or something last year, out of 5 games it won 4 and Lee Sedol won one game. But then they were like, "How did it loose that one game?" So they improved the algorithm even further and recently – I think it was January/February this year – AlphaGo beat the top 60 players of Go, 60-0. So it beat each one of them and it never lost even a single game in all those years. So no wonder they're retiring it. You can't win against that thing any more.
Ravindran:	Yeah, you can't win against that thing and they want to move their resources on to other things and they feel that— Ke Jie was the Chinese champion.
Kirill:	Yeah. The other thing that I liked about AlphaGo was, in one of the games against Lee Sedol, I think it was like the third game, AlphaGo—so, in the game of Go you place these black

and white stones onto these boards 16 by 16, you place them across the lines, and at the end, if you surround an enemy stone, like white stones surround a black one, then the black one is captured. And at the end of the game you calculate the territory which is counted as what's within the borders of adjacent stones. So it's pretty straightforward, those are the rules. But it's a very intuitive game because it's not like in chess, you know, pieces have their own values and so on.

But here in game 3 AlphaGo put a stone not on a third line from the side, but on the fourth. That's something nobody has done ever at the start of the game, and Go has been played for like 3,000 years. Human players never ever do that. And then that move, that single move helped AlphaGo win the game further on. So that one move has now got into the history of Go and all of these players around the world are analysing that one move and a lot of the grandmasters have said, "This one move has opened up a new world of opportunity for me in my mind to what Go can be." And it's just an example of how AI is not just used to complete certain functions, but also progress the human race, even though it's in the game of Go. I think that was pretty cool.

Ravindran: That was amazing, yes. In fact, I've been reading a lot from Go commentators who were actually writing about that game. He said, for the first time in his life, he felt like he was watching an alien intelligence in action because he could not explain what was happening when looking at the game and then retroactively they get out that that was the really, really killer move. So they just couldn't reason about it when they looked at it the first time.

Kirill: Yeah, it was crazy. And just for those listening out there, if you're not impressed by AI or reinforcement learning algorithms winning in a game against humans, well, that same team from Google DeepMind – I'm not sure if they used the same algorithm or not, but they created an algorithm, also for reinforcement learning, which they then applied to one of Google's data centres where they have all these servers. And obviously there's a lot of servers, people are searching, they heat up and then it has to be cooling, so they applied it to the air conditioning system and the reinforcement learning algorithm was allowed to control the air conditioning and control if the windows are open or closed and stuff like that and it was monitoring the temperature.

So that reinforcement learning algorithm, it was able to reduce the electricity consumption in that facility by 40%, which resulted in a 15% reduction of the electricity bill for Google. You can just imagine like tens of millions of dollars for that specific location, so that alone is impressive.

Ravindran: Yes, and I do believe they're going to be productizing that as well.

Kirill: Yeah, I totally agree with that. And I like how you mentioned that commentator said it was like he was observing artificial intelligence playing a game. I was actually reading a discussion yesterday between Yann LeCun and somebody else on Facebook, it was live and it was very interesting. Yann LeCun is the head of AI research at Facebook, and he's one of the founding fathers of deep learning and convolutional networks. He was saying—and I'd really like to get your opinion on this question—this goes back to what we were talking about, the history repeating itself, that even back in the 80s, some of these algorithms already existed, like actor-critic, and he was even using neural networks. Now it's all happening again, but you have more power and so on.

Well, what Yann LeCun's opinion was, this whole hype around artificial intelligence, reinforcement learning, how these technologies are progressing and how they're going to change the world and robots are going to be all over the place in 20-30 years, he says it's all just because of a coincidence. That we were at the stage where we should have been in our natural technological development. We just stopped exploring those algorithms in the 80s and now we've come back to them and they look really cool to us. But had we been exploring them continuously from that period of time, this would have been just a natural progression. And it feels like an explosion right now, but it's just a natural course of things and it's going to take much, much longer for AI, like general AI and strong AI to come on our planet, into our society, into our lives and so on. What is your opinion on that, based on from what you've seen in the field of reinforcement learning and how quickly it's developing?

Ravindran: I largely agree with the sentiment that strong AI is still a while off. We are able to do a lot of very, very interesting things, but like I always say in my lectures on deep RL, reinforcement learning for 20 years was largely limited to small game domains. And what deep RL has allowed us to do is move reinforcement learning to large game domains. So we still are some ways off from solving hard AI problems in the sense of getting an AI agent that can get a true understanding of the world. It's still something we don't have a handle on, but then we are solving so many interesting problems which for a long time we have been thinking only in the domains of human problem solving. So it does look like the bar is getting closer and closer, but it's still some ways off.

From a reinforcement learning point of view, there have been some significant changes, I would say, and we started looking at all these kinds of differentiable models which deep learning allows you to use, so we are thinking of ways of using our interaction data in ways that were not possible, say, 10 years back. So we are thinking about looking at gradients through models and using that for learning value functions, which was something which we didn't conceptualize. I should say that there have been some geniuses who actually thought about this way in the past.

For example, Paul Werbos has a paper from the early 80s where he talks about using differentiable models and making reinforcement learning more efficient. But he really did not have the tools and the capabilities that deep learning brings to the table now, so they could not really solve hard problems at that point. But the ideas themselves, they're not that novel, what they're doing now. In some sense, RL — we are reinventing the wheel, but the wheel as was drawn on paper now actually runs. That's a significant improvement that's happened.

In some sense, yes, I largely agree with the sentiment in the sense that there were ideas that were there in the 80s, and sometimes in the early 90s, which were abandoned when things seemed very hard to achieve, but now we are going back and revisiting them and we're able to achieve them now. So I wouldn't strictly say that we shouldn't have abandoned them at that point because frustration is a factor. If we're trying to solve things and we don't see an immediate solution on the horizon, it sometimes makes sense to toss that, do something else and come back to it. That's exactly what had happened to the AI community now. We've come back to problems that we were looking at earlier. It's an absolutely exciting time to be in AI. That is for sure something that has happened in the last 2-3 years. Deep reinforcement learning is now acting in some sense at the forefront of all this excitement about AI.

Kirill: Yeah, I totally agree. And whichever way it goes, whether it's an explosion or it's just going to keep progressing naturally slower now, as Yann LeCun put it, in any case it's going to bring value to the world. And with that in mind, for those listening out there, where would you recommend to get started? Because there are people for whom this is a brand-new field, first time they're hearing about reinforcement learning and the part it plays in artificial intelligence and of course all of the incredible applications that we're seeing all around, it's not limited to AlphaGo, you've got self-driving cars, you've got recommender systems or in banking, like you mentioned, AI is popping everywhere. Where would you recommend for somebody to get started? How can they get started in learning more about this field and approaching it in a manner that won't scare them away, that they won't find it too complex or unbearably heavy in terms of mathematics and things like that?

Ravindran: There is a course on Udacity that's taught by Michael Littman, which is a pretty lightweight introduction to reinforcement learning. That would be a good place to start if you just want to get a quick idea. I think it's very short, like 5 hours or 6 hours or something. So that's a pretty good place to start if you want a really lightweight introduction to reinforcement learning.

And if you want to really knuckle down and get into the nitty gritty details, you always have my online course. And then of course the Sutton & Barto book. It's pretty light on the math, it's more heavy on the intuition, so that is an easy place to start as well. But you do really need to invest some amount of time, a month or so, to go through the book. But the best place to start would be the Udacity course and then—

Kirill: Udacity course by Michael Littman?

Ravindran: Yeah. And then, of course, you should graduate looking at my videos.

Kirill: Okay. And your videos, if I'm not mistaken, they're free? Is that correct?

Ravindran: Yeah, they're free.

Kirill: Yeah. So everybody can look them up. I attest to that, Ravi's videos are definitely a great place to start. And what I also like about them is that even if you don't want to learn the whole of reinforcement learning, everything, you just want to learn about this one specific thing, then you can just go watch that one video, whether it's the Upper Confidence Bound algorithm, or Thompson sampling, or Q-learning. Just watch that one video and you will already get a good overview. They're interconnected, but you can still get a lot of value from individual videos. So that's a good place to start as well.

And the Sutton-Barto book, of course, is the foundation for reinforcement learning. Thank you so much. We're coming to the end of the show, thank you so much for coming on and sharing your expertise. If our listeners would like to follow you or contact you with maybe more questions or just understand how your career is going to be progressing, where is the best place to get in touch?

Ravindran: I'm on LinkedIn. So you can find me on LinkedIn or you can just e-mail me directly. You can easily find me by googling 'Ravindran' and 'IIT Madras.' All my contact information is online so they could do that.

Kirill: Okay, perfect. Thank you. And I have one final question for you: What is your one book that you could recommend to our listeners to help them become better data scientists?

Ravindran: This is a book which I recommend all my students read when they start my data mining course that I teach. It's a very popular book called "Freakonomics" that tells you that it's the right question to ask that matters in data analytics more than the techniques that you're using to solve them. So that's one place that I would recommend.

Kirill: Yeah. I've heard some fantastic things about "Freakonomics." I even met a person who has worked with their team, so I definitely can attest to that. I haven't read the book myself, but it's a great team that's written that book. Yeah, check it out – "Freakonomics." Once again, Ravi, thank you so much for coming on the show, it was a pleasure having you on and we've definitely learned a lot from your vast experience in the space of reinforcement learning.

Ravindran: Thank you, Kirill. Thank you for having me on the show.

Kirill: So there you have it. That was Dr. Balaraman Ravindran, or Ravi, from the Indian Institute of Technology in Madras. Hopefully you enjoyed this episode; hopefully you got some good takeaways from here. My personal favourite takeaway was the description of different reinforcement learning algorithms so that a lot of the time, especially if you've taken our course on artificial intelligence, you can get kind of carried away thinking that Q-learning is the main reinforcement learning algorithm, that's all it is. But it was great, it was refreshing to hear about the value function-based methods, one of which Q-learning is, the policy gradient method, which we didn't go into detail in this podcast, the genetic algorithm methods, and then the separate branch of the dynamic programming methods that also exist.

So, a great overview of reinforcement learning, and of course you can feel the passion from Ravi with which he was explaining these things. You can tell, you can see how it would be very exciting to be in his classroom. By the way, if you are interested in checking out those lectures, they're all free. You can find them online plus after the podcast Ravi messaged me saying that he's setting up a repository of his lectures where you can find all of them together in one place, which I think is very, very valuable and it's something that probably is a good idea to bookmark even if you're not going to watch them now.

So you can find that at the show notes at www.superdatascience.com/79, plus there you'll also find a link to something that we didn't cover off on the show, but something that you might be excited to learn about, some of the exciting and inspiring work that Ravi and his team are doing in the space of reinforcement learning. Plus, of course, we'll also include all of the resources mentioned in this episode as well as the transcript for this episode. So there you go, I hope you enjoyed this episode, make sure to connect with Ravi on LinkedIn and follow his career. And I can't wait to see you next time. Until then, happy analyzing.