학부모님 상담용,

이 프로그램은 정식 데이터 사이언스 과정이 아니고, 데이터 사이언스를 위한 전 단계인 파이썬 프로그래밍 과정입니다.

파이썬이나 자바 등 범용 프로그래밍 언어는 현재 광범위한 산업 분야에서 다양한 용도로 사용되고 있으므로 막연히 코딩 교육의 목적이 정확히 정의되지 않을 경우 코딩이라고 할 경우 코딩의 분야를 게임과 같이 흥미거리 위주나 학교수업 보조 등으로 국한시킬 수 있습니다. 이 경우 학생의 단기 흥미를 자극해서 초보 수준의 프로그래밍 교육을 시키기에는 적합할 수 있으나, 초보 단계를 벗어난 후 다음 단계에서 다른 분야를 학습하는 방식으로 널뛸 가능성이 많습니다. 물론 그러한 방법도 나름 장점이 있을 수도 있습니다.

엘리트 코딩 교육은 데이터 시대에 맞추어 K-12 학년 별 혹은 학생의 기 습득 지식 수준에 맞춰서.. 파이썬 기초부터 구글 텐서플로우를 사용한 딥러닝 응용 모델 까지 데이터 및 데이터 과학이라는 틀 내에서 학생들이 계속 성장할 수 있도록 구성되어 있습니다.
이는 1~2 년 단기에 끝낼 수 있는 것이 아니라.. 대학 입학 전까지 단계적으로 계속 학습해서 학생의 수준을 상당 레벨까지 올려서, 대학 입학 뿐 아니라 대학 입학 후 해당 전공분야를 공부할 때 실질적 도움을 주고 향후 취업에 직간접적인 도움을 줄 수 있도록 기획되고 있습니다.

모든 학생이 컴퓨터 사이언스를 전공할 수도 전공할 필요도 없습니다. 하지만 데이터 시대에 모든 전공 분야 모든 산업 분야에서 데이터 관련 프로그래밍 기술은 향후에도 지속적으로 요구되고 있습니다. 인공지능 시대에 접어들면서 지금 많은 소프트웨어 엔지니어들이 하고 있는 루틴한 코딩 업무는 인공지능에 의해 drag and drop 방식으로 바뀔 것입니다. 또한 향후 사회가 원하는 인재 역시 데이터에 대한 Above Algorithm 군과 Below Algorithm 군으로 나뉘어져 Above Algorithm 그룹이 이 사회를 리드해 나갈 것입니다.
엘리트의 코딩 교육은 지금 유행하고 있는 코딩 교육을 추종하기 보다는 우리 학생들을 데이터 시대에 맞는 미래의 기술 리더로 키워 내는 것에 보다 집중하고 있습니다.

인도와 미국 일부 지역에서는 이미 5 학년부터 데이터 과학 캠프에 참여해서 성공적으로 교육을 이수하고 있고, 한국 동탄고 김윤기 군은 고등학교 2 학년 때 시각장애인 안내용 AI 프로그램을 만들어 업계를 깜짝 놀라게 했습니다. 모든 학생들이 동일한 재능을 가질 수는 없습니다. 따라서 Data Oriented Programming Education 이라는 큰 길 위에서 학생들의 수준에 맞는 다양한 출발점과 목표점을 발굴해서 모두가 만족할 수 있도록 최선을 다 할 것입니다.

Data science for kids!

Two weeks back, we decided to experiment teaching data science to 6th-9th graders! We think it is important to introduce students to thinking in a data-driven way early on in their lives; also kids are way more fun to work with than higher-ed students, so it was an easier choice for us to make!

We sent out a form asking kids to apply to our cool Kids Data Camp - the first in the world?! We thought kids in 5th grade would have been too young and those in 10th would be more focused on school academics. We had 18 people apply to us, with most of them interested in science and math and a few in history/arts.

This weekend, we had 14 people turn up (there was no selection barring self-selection). This included one 10th grade student and one sophomore undergraduate who tagged with the group to learn!
Some kids came in early. We put on a Youtube video on Scratch for them. It was fun to discuss it with them and they related it to Lego right away. We asked kids to install scratch at home – and make a dancing Shah Rukh Khan (famous Bollywood movie actor) on it and also have him jump around from one building to another!

The Ice Breaker
Once all the kids had assembled, we had a quick ice breaker. Parth and Abhishek, interns from IIT Kanpur, divided senior and junior students in two sets in order to pick one from each to form a group. A simple way to maximize group success- read here to learn how they did it !

Then Samarth, our intern from Harvard, introduced the idea of data science to kids. He started with the famous John Snow cholera outbreak example. Kids were very quick- by a show of hands, everyone had seen a Google map. They understood that infected people were clustering around one pump and there were other vacant pumps. Couple of questions – Why some dots are large and small? Why did someone not go to a pump which was farther away. We answered. We told them there were three learnings for them:
a. Don't waste water- it wasn't as easily available 100 years back and still not to many;
b. Don't run away from problems; try to solve them; else they will catch up with you (couple of them said that their way of solving the problem was to just run away from the city!),

c. You can solve problems with data – here is a medical problem you solved by plotting infected families on a map; you did not need any exposure to biology or medicine to come up with a preliminary inference.

The data exercise

We moved on and started with the key data set and experiment. Our aim was to give kids an idea of the whole cycle of data science – data collection, data entry/cleaning, feature extraction, visualization and model building (if we could get there, we had presumed we wouldn't due to a paucity of time) and also sensitize them to data security/permissions concerns.

We designed the following exercise: Every kid will get a set of 48 faces with names and their hobbies. The kids had to give a rating of 5 if they will make the person a friend, 1 if not and could choose other numbers in between. All the 7 groups completed the exercise with one mentor each. Out of these we pulled 16 samples out as a validation set :) The 'train' data sets were then exchanged among groups.

Introducing Features

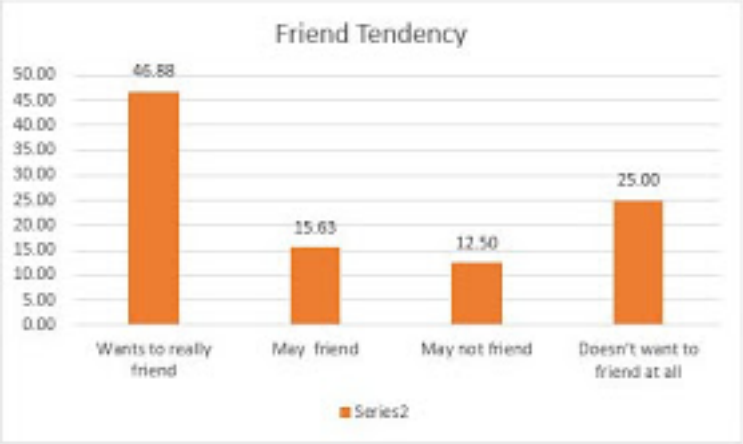
We then asked the following: if we wanted to know what kind of people does Raghav (one of the kids) prefer to make friends, how could they infer this by going through these sheets? One of the kids suggested that we could look at what kind of games his friends played and then tell accordingly. We asked what else? We then introduced that it could be that some of the kids prefer making friends with boys and some with girls; we asked a boy whom does he prefer to make friends with more often - he said boys; couple more said neutral.

Then we discussed two more features: we had smiling and neutral faces – would some people make smiling people friends more often? And also, we had old style names and new names – would some people prefer to make folks with new names friends more often? Kids seemed to have understood that people could possibly, not necessarily, make choices on this basis. For the workshop we decided to go with three features: gender, hobby and name style.

We used excel as the platform for all experiments. We had a sheet with features already entered for the data set. The kids had to enter the ratings and check the features. The kids did find some features wrongly entered and also some ambiguities: is squash indoor or outdoor, is Shilpy a new name or an old name? :)

Question 1: Is this kid a friendly person?

The first task of the kids was to find if the person they were analyzing was a friendly person or not - will s/he more often make friends than not. To get this right, kids had to simply count how many people were marked each as 5, 4, 3, 2 and 1. Some of the kids used filters to do this and others counted manually. They finally made a graph. Here is the first graph we discussed with the whole group, where the red bar depicted percentages and the blue bar depicted the actual number in each bin.



** These graphs were created by kids who participated in our exercise*

We made two inferences:

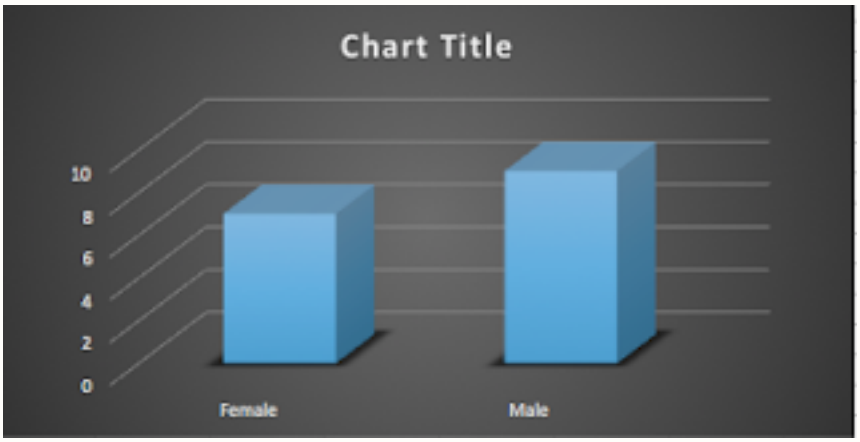
- K (anonymous) was a friendly person: s/he more often makes friends than not.
- K is clear-headed and a fast decision-maker. S/he doesn't have many may be/may be not cases. S/he either decides to make a person a friend or not.

Then we discussed couple of more graphs of other kids. We said statements positively ☺: V makes lesser friends, but that is because s/he likes to spend time studying. One group said, she

is confused since she had many may be/may be not: we corrected: not confused, she takes time to decide who to make a friend or not, because she could possibly be thinking deeply about it.

Question 2: What kind of people does s/he prefers making a friend?
This was fun! Our next exercise was that they had to find among the people the person chose to befriend, were there, say, more males than females? And similarly for other features. (We had created a balanced data set with 50-50 of each feature type; this created a simplification that we did not have to see the non-friends group) Again kids used filters and counted for input variables of the two types and plotted graphs. We had already inserted a template for the kids to put in their counts in their excel sheets; then plotted the graphs themselves.

Here is a set of graphs we discussed.



** These graphs were created by kids who participated in our exercise*

So, we learnt – the student for whom we'd made this graph definitely likes to make friends with people who plays outdoor games – this is a clear trend. Next we talked about gender – the person makes male friends slightly more often; but this trend was not completely clear, since the difference between males and females is too little. It needs further investigation. Same for the third feature.

The big take away was: we can find what kind of people each of us make friends with! Kids seem to understand and appreciate this. We told them that they could have done this differently, by

interviewing the person and then trying to say who he will make friends – but we do it differently – 'learning by example', we see who they make friends, analyze it to figure out trends and then be able to predict!

Making and validating a super simple predictor
Ideally we wanted kids to make a predictor with a simple point based system, but we didn't get there. We however went ahead and took the example of the kid just discussed, who had shown outdoor games as the key deciding factor, and considered that feature as the predictor – we took her 'validation' data from the envelope and saw how we well we did – it was only ok, honestly! But kids got the concept. They could predict unseen data based on a set of seen data.

[Edit - June 27th We decided to do a small follow-up session to close the loop and actually build the predictor. We worked with three kids this time on (rest were holidaying around the world, our ambassadors!). We re-did the whole exercise with the kids as a re-cap and came to the stage, where they identified 'features' - those that were overrepresented in the 'make friend' set (See table below-second column). For gamma, whereas there was a difference, it wasn't as much.

Person	Features Who to make a friend?	Train accuracy (32)	Test Accuracy (16)
ALPHA	'Outdoor' people	84.38	81.25
BETA	'Indoor' people	81.25	81.25
GAMMA	With 'new names'	62.50	75.00

Then kids made a simple predictor with one feature (such as, =if (c23="Indoor",5,1)). Each one wrote their own after we showed an example. Then they found the accuracy: simply typed in an adjoining column 1 if the predictor and the actual matched and 0 otherwise, then counted the 1s and got the percentage (Column 3 in table above). One kid actually predicted the accuracy saying it would be same as the percent they calculated to draw the feature graph -- smarter than we think!

Then came the real test -- we took out the envelopes containing the unseen test and we manually marked each one we got right (ideally to be done in excel!) - kids were so happy everytime we did it right and low when we didn't :) - we were nervous! Finally, kids were super-happy seeing the high accuracy each of their predictors had. We asked why did Gamma have a low accuracy -- after some struggle one did say, because the difference for 'new names' feature wasn't high.

Thus the kid predictors shined doing much better than the predictors we make all the time!!! ***Edit end]***

We then got a data release form signed from them and explained to them that they have the right that their data isn't publicly disclosed and we seek their permission – we will anonymize their data. One girl opted out. Rest of the data can be found here.

When one of the authors asked with a wink how many from the gathering would like to come over for a part 2 of the data camp the following week, ten of them raised their hands :) A good test for us. See the kids' blog entries here and mentor experiences here! Harsh also suggested to them that they should start making data entries of their expenditure and pocket money! Some really interesting suggestions came from kids regarding what they would do with this knowledge.

Do note, that we were using lot of assumptions to simplify this – correlation vs. causality, balanced sets, no significance testing, small sample size, etc. Our aim was to lead them to a naiver naïve bayes. We think this is a fine approach like the famous Arundhati nyaya.

Learnings:

- We need 5-6 hours to run this right and we would have done the model too and explained things a lot better.
- We didn't have a what-next? A strong take away, resource sheet and continuity.
- Kids need to know the concept of percentages – we think 7th to 9th might be a better target.
- Currently, we have 1 mentor for every 2 kids. We need this to be more scalable. Should be possible.
- Would want to emphasize explaining data science vs. other ways of doing things through some examples. We give them a problem, they try it and then we give the data way of doing it.
- More visualizations to share.

And of course, this is just the TIP OF THE ICE BERG!

Thanks Harsh, Bhanu, Nishant, Gursimran (for the photos also!), Parth, Abhishek, Vishal, Samarth – good show. Thanks Una-May for the encouragement and helpful ideas!

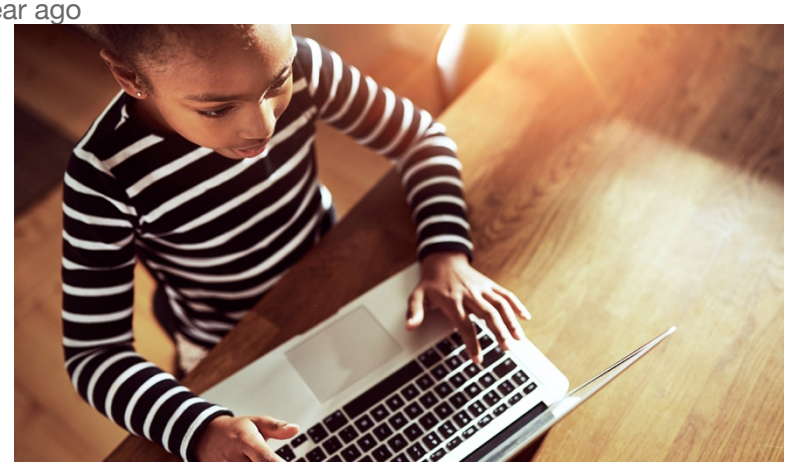
-Varun & Shashank
2015

PS - If you want to replicate this exercise in your backyard, feel free to use all our material. We've rigorously documented the entire workflow here. Do write to us to let us know how it panned out!

Skip Coding, Teach Data Science

- Tom Vander Ark

1 year ago



Computer programing can be a valuable part of the K-12 curriculum. Pleasanton USD superintendent David Haglund thinks "programming is a mindset and a great foundation for math."

Haglund uses the first of three rationales for coding: it's a good way to teach computational thinking, a skill likely to be valuable in college and careers.

Code.org advocates that "Every student in every school should have the opportunity to learn computer science." Let's call this the access and equity argument. They argue that computer science drives innovation and as a career pathway it remains marginalized in K-12, particularly at schools serving underrepresented minorities.

For high school students, coding may be a career skill–the Career and Technical Education (CTE) argument. During or shortly after high school, young people can deploy coding skills to build websites (using front-end, or user interface, languages like HTML, CSS and JavaScript). Or they may want to build new apps or tackle big back-end problems (using languages like PHP, Python, C++, Java and Ruby).

School districts have rushed to adding coding to the curriculum using a mixture of these three arguments–computational thinking, access and equity to lucrative career pathways, and valuable CTE skills.

On the last two points (short and long term employability) about 60,000 U.S. students graduated bachelor degrees in computer and information services. There are probably 10 times as many computing jobs open and that could double by 2020.

The old Advanced Placement Computer Science class focuses on Java, a widely used object-oriented programming language–it's a CTE approach. The new AP Computer Science Principles course (launched fall 2016) "offers a multidisciplinary approach to teaching the underlying principles of computation." It's an extension of Haglund's computational thinking argument.

The first point of this post is to *get clear about your objectives for adding coding to the curriculum.* That will bring focus to the effort, staff it appropriately, prioritize it in the master schedule, and frame the size and nature of the investment.

Typical Problems With Teaching Coding In School

"Computer programming and computational thinking are the new foundation for our ever-increasing technological society—a society that calls for innovation, entrepreneurship, teamwork and creative thinking," said Idit Harel, CEO of Globaloria, an advocate for computer programming for all.

Harel is not a fan of the "pop computing" apps kids see in schools. She thinks students "aren't doing the hard thinking that defines what the subject of computing is today, and most importantly, what it can be in the future." Rather than a quick fix, Harel would like to see students engaged in computing projects every day.

A 2014 UK study on teaching computing (now required) highlighted three challenges: teacher subject knowledge, the high need for differentiation and technical problems. They concluded that "one specific focus should be on supporting students to develop resilience and the ability to learn from mistakes." (It always comes back to social and emotional learning, right?)

The Computer Science Teachers Association predicts there will be more than 9 million U.S. STEM jobs by 2020 with half requiring computer science knowledge and skills. They published a report listing reasons why schools in the United States are failing at teaching computer programming skills. They find the certification processes deeply flawed and (not surprisingly) want a lot more certified CS teachers with multiple roots to the classroom. They note that CS credits only count for core graduation requirements in 14 states.

Below is a diagram of how and where CSTA would like to see subjects introduced–it's pretty well aligned with the new AP CS class.

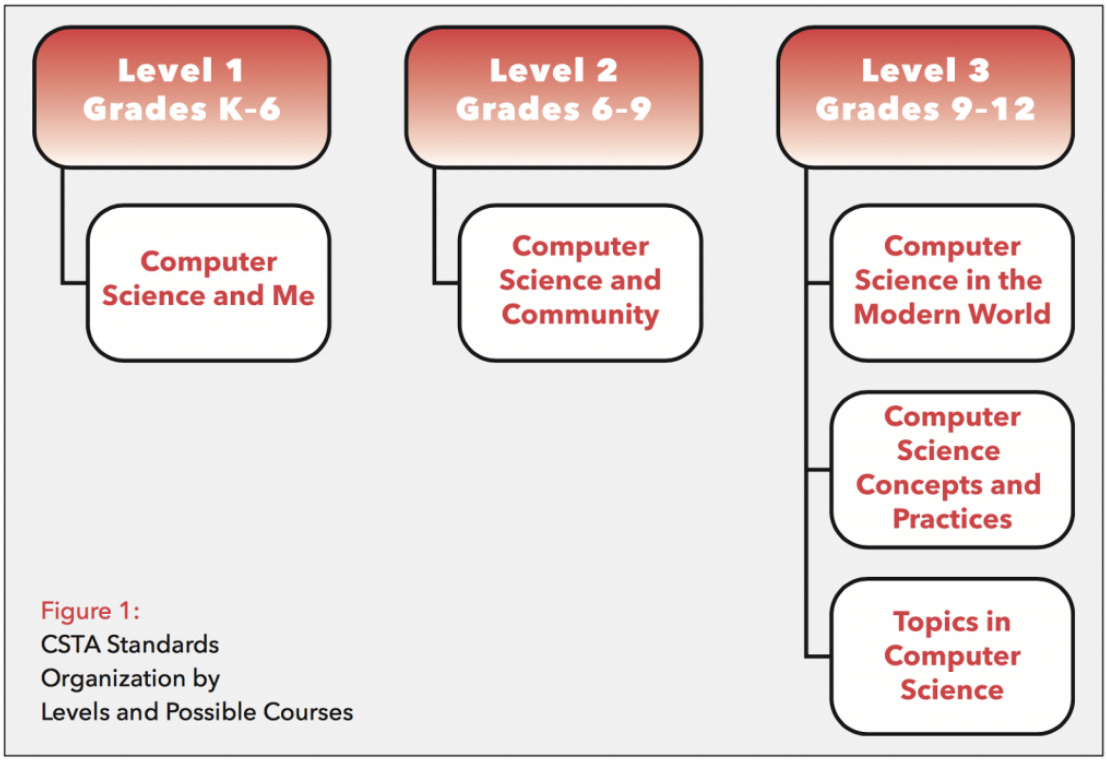


Figure 1:
CSTA Standards Organization by Levels and Possible Courses

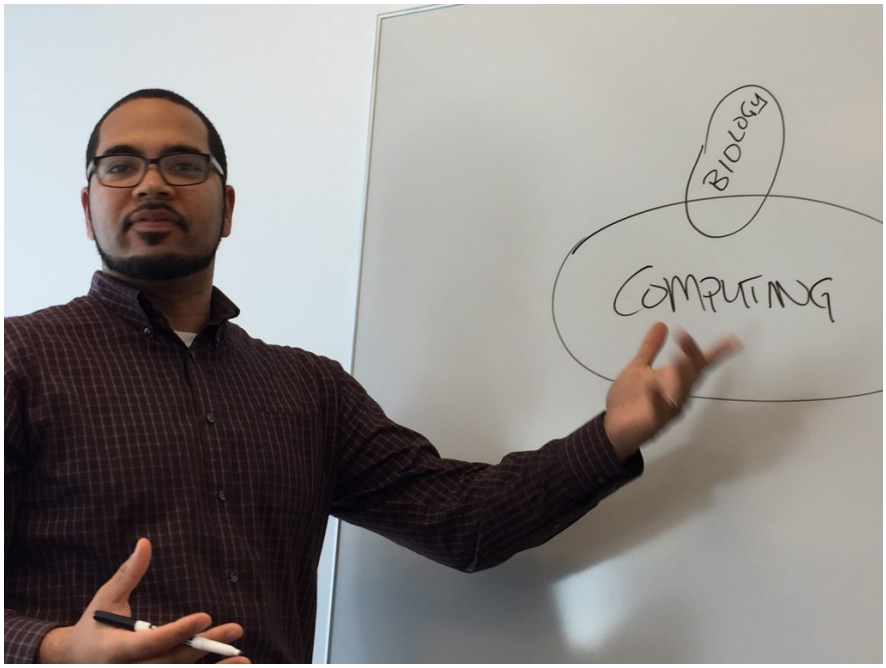
The central problem with the CTE approach to coding is that it's tough to keep up with the hottest languages for current job opportunities. And it appears that in the near future, artificial intelligence (AI) will begin replacing coding jobs.

Gottfried Sehringer, marketing VP for a web development firm, said, “Don’t teach everyone how to code. Teach them how to identify and understand needs, as well as how to visually express logic. Teach them how technology works, so they can understand the realm of possibility and then envision game-changing innovations.”

Just-In-Time Coding

The best way to avoid teaching soon to be obsolete computer languages to waves of young people that may never use them is to lead with problems worth solving and teach just-in-time (JIT) coding.

From week one at Olin College, students are conducting projects. Amon Millner, an assistant professor of computing and innovation at Olin, teaches computing across the curriculum–like a roving coding ninja. “All Olin students are exposed to computing in their first semester, in the context of a class about simulating and modeling real world problems,” said Millner.



This just-in-time approach isn’t easy to pull off without talent like Dr. Millner’s, but it’s the best way to integrate computing into a PBL approach. Olin also encourages freshmen to learn new maker skills from anyone on campus who has earned relevant certification.

At the innovative global college Minerva, the ability to code in Python is an entrance requirement, and incoming students find a way to learn it before coming to campus. Minerva requires Python because it is useful for back end coding, data science and using machine learning programs like Google’s open TensorFlow.

Olin and Minerva point to the opportunity to make coding a required microcredential but allowing students and teachers to identify how and when to incorporate it into projects.

Teaching Data Science

While there is high demand for coding skills today, AI will increasingly take over routine coding jobs as it grows in sophistication. In the way that web site development has become a drag and drop affair for many new sites, AI will automate code development in many fields.

The longer term and larger employment and impact opportunities lie in data science–assembling, scrubbing and analyzing big data sets. Data science is a subset of computer science. (It's not in the old AP CS course but an important part of the new AP CS Principles course.)

What is data science? It's a set of methods and tools used to extract knowledge and insights from data sets which may be highly structured or very unstructured. Draws heavily on statistics and data analysis.

Why have I never heard of this field? It's new. You would not have seen many jost postings for it in the last decade. But five years ago, Tom Davenport in HRB called the "The Sexiest Job of the 21st Century."

How is data science different than coding? Computer programming is usually a very directed task with specific functionality in mind. Data science is usually more discovery oriented. As Davenport explained, "Data scientists make discoveries while swimming in data."

What is data wrangling? It's the messy process of sourcing, mapping and transforming raw data into a clean data set worth analyzing, visualizing and extracting new knowledge.

What about data analytics? It's the discovery, interpretation and communication of meaningful patterns in data.

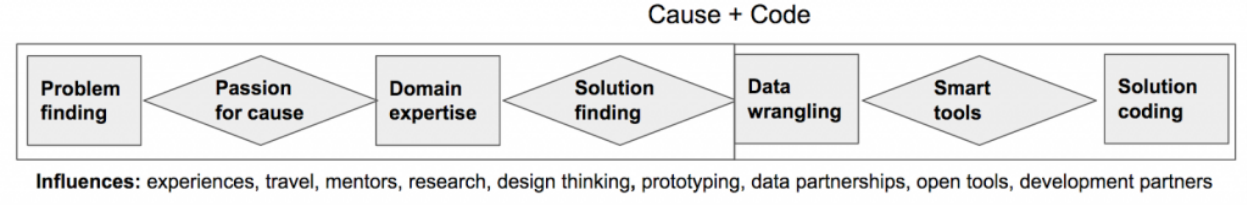
How do we move from coding to machine learning and data science? Two recent podcasts discuss the transition:

- Linear Digressions: From Software Engineering to Data Science
- Talking Machines: Getting a Start in ML

Eric Lander, who runs the Broad Institute, said in a few years every biologist will be computational. It looks like the same will be true for doctors, mechanics, economists, water managers and soldiers. As every profession becomes computational, wrangling big data sets will become more important.

Because these big data sets will often require partnerships with multiple sources, data science is becoming a social science requiring strong collaboration skills. For example, young people who want to study local water quality could set up hundreds of sensors, survey households and convince the local water department to give them access to their water quality data.

As the data analysis tools get better and easier to use, the new challenge will be problem finding–identifying and framing problems worth solving–and then funding, staffing and managing projects around solutions. We call this *Cause + Code*, the new impact formula. It combines design thinking (problem finding/framing, user empathy), data wrangling, some coding and a lot of project management.



While exponential technology and AI are mentioned these days, it's often in association with likely job losses. The good news is that there's never been a better time to make a difference. The new lever is wrangling big datasets associated with a big problem and aiming smart tools at them to find or build big solutions.

K-12 Conclusions

1. Clarify your objectives for adding coding to the curriculum (e.g., computational thinking, career path, specific CTE skills) and then support those goals with staffing, investment and student support.
2. If you do offer programming classes, make sure they are very current and linked to employment opportunities (high schools may want to rely on career centers, community colleges or coding bootcamps for this).
3. In the same way schools have integrated writing across the curriculum, look for ways to integrate data science across the curriculum, because every problem worth solving has a data set associated with it. A couple big integrated community connected projects each year can be a great place for a data science integration.
4. Create a business or university partnership to support computing and data science across the curriculum.