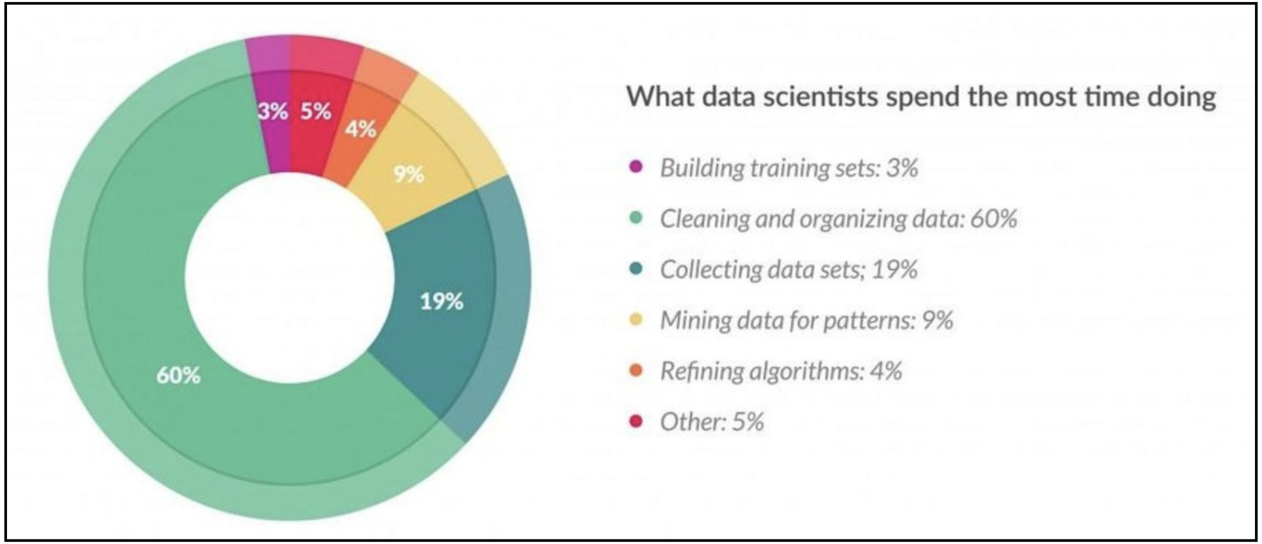
Preparing yourself for a Career in Data Science

The most important aspects of this job which you generally don’t learn in any class.

This deck is a distillation of insights from my work-ex. **- Anand Yati**

What’s the most important part of building an ML model?

***Data preparation** accounts for about 80% of the work of data scientists*



Source: Forbes survey of data scientists (2016) [Article]

Some of the best advice I ever got for ML modeling @ job:

“..Don’t use a missile to kill a mosquito..” Use the simplest possible solution to solve a business problem, don’t apply ML everywhere.

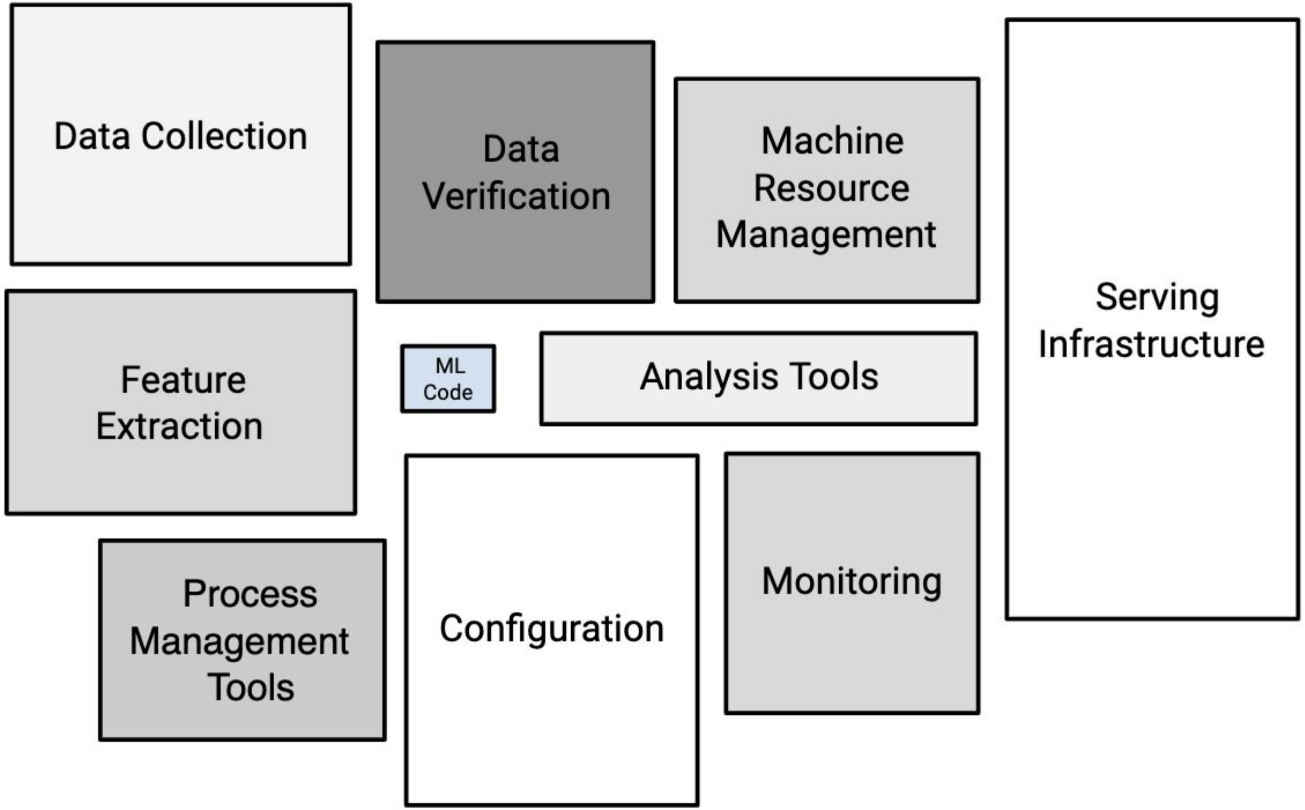
Some of the best advice I ever got for ML modeling @ job:

“..Spend time and frame the ML problem nicely before jumping into solution..”

Answer questions like:

- What problem I want to solve, and why ML?
- What would be performance metrics?
- How would success look like in production?
- Do I have right type, quality and amount of data?
- Who will do maintenance?
- Do I really need an ML model for this problem?

Real-world production ML system



ML code is at the heart of a real-world ML production system, but that box often **represents only 5% or less of the overall code** of that total ML production system.

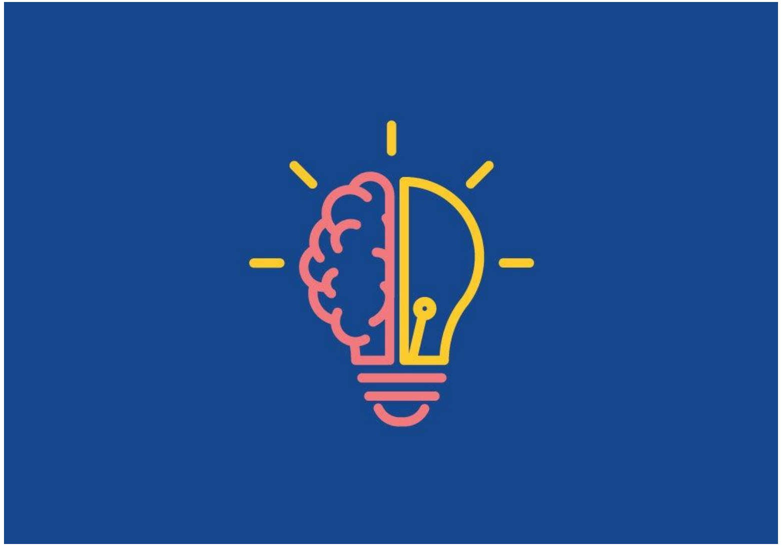
Thus, deploying an ML model in real world requires collaboration b/w multiple teams and a unified vision towards the end goal.

Source: https://developers.google.com/machine-learning/crash-course/production-ml-systems?hl=tr

What does that mean for a data analyst/scientist?

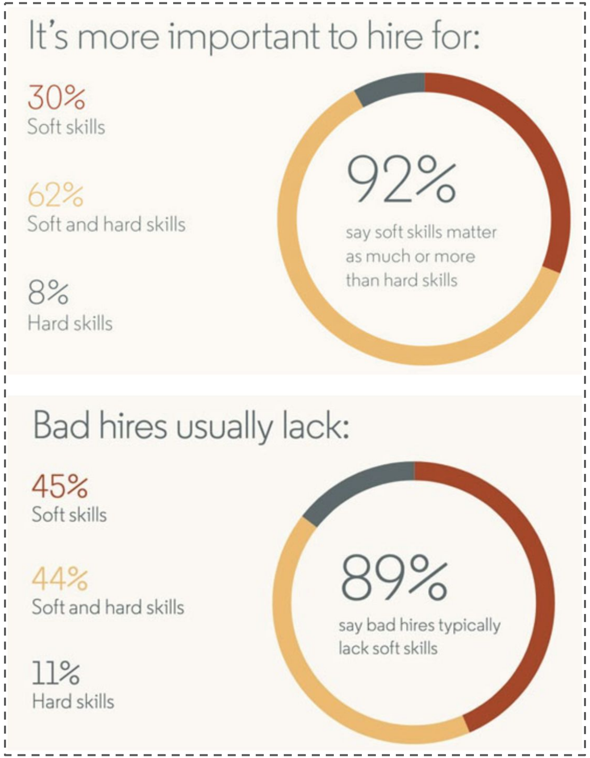
Wisdom I got from a senior analytics veteran: **“Good solution = High quality + Acceptability”**

If you cannot convince your team, engineers, management/leadership about a good solution, no matter how good your solution is, it will never land into production, and thus will never exist.



Softer aspects of being a successful data professional:

- Don’t chase best possible solution from the beginning, if you can start off from a working solution with acceptable quality.
- At times there can be multiple possible solutions to a problem, choose the solution with highest feasibility having acceptable quality. Be flexible, always think about improving from current state and reach the end goal iteratively.
- Put in hard-work while developing solutions, but don’t stick to your solutions on a personal front. Any solution can be improved, altered , implemented in phases if that improves its feasibility.
- When engaging with peers, keep in mind the needs and incentives for other people working with you. Think it from there perspective as well: Why should that person help you? What’s in it for her/him personally? Try to distribute tasks which are impactful across peers.
- Convert people who can shoot down your solution into stakeholders in your solution. If not possible, rally people who have decision making authority or influence with your solution. This improves acceptability of solution.
- Build long lasting partnerships, and get buy-in from concerned teams to maintain (commit bandwidth) systems surrounding your ML model. Model building/deployment is not a 1 time activity.



Source: Linkedin talent blog

Explainable AI

Build ML systems which are more explainable. For example a bank's fraud detection system can have 2 configurations of model:

- A mixed model with demographic features from users and transaction type.
- Scores from 2 separate models being combined:
  - A user profile risk classification system
  - A transaction risk classification system

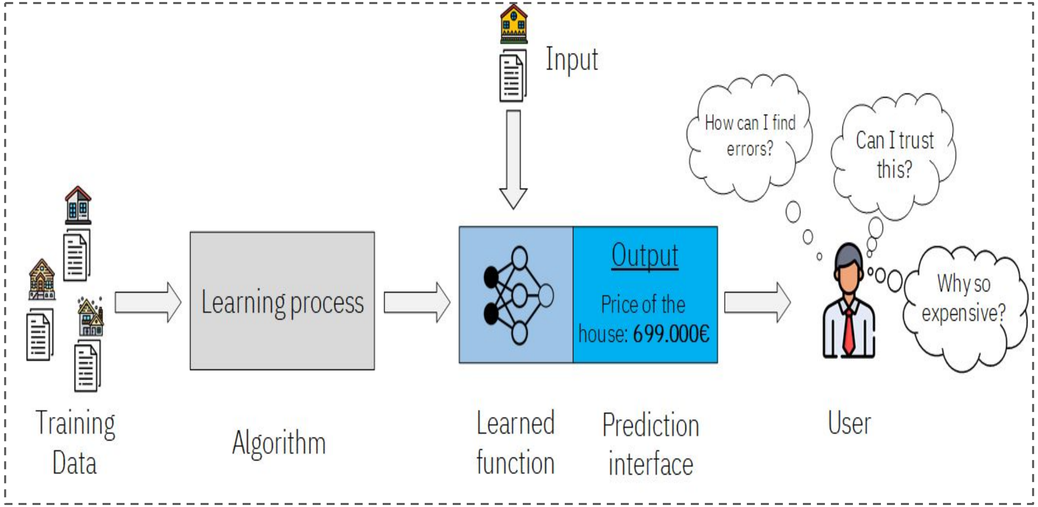


Image source: link

While 1st model can give better result in training, but think of what that model is learning at the back end, it might allow same transactions from a particular demographic/city and block it for another demographic/city, where as both could have been legit users/transactions and one of the city/demographic could have been under-represented in data. Also, how will you explain, feature crosses like account_age-city-transaction_type? What happens if the same user makes same transaction from a high risk city?

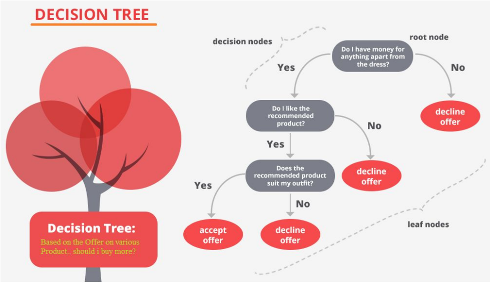
Now think of profiling user risk and transaction risk separately. You will have very clear decision boundary/understanding of why a transaction or user is being considered risky. It will help you build a modular system, which can be improved individually in future, and you will always exactly know where the problem/improvement-opportunity exists.

But how do people achieve explainable AI in neural networks?

One of the approach can be building 2 different models on the same data:

1. A neural network based model for production
2. A linear/logistic/tree model for explaining feature interactions/importance

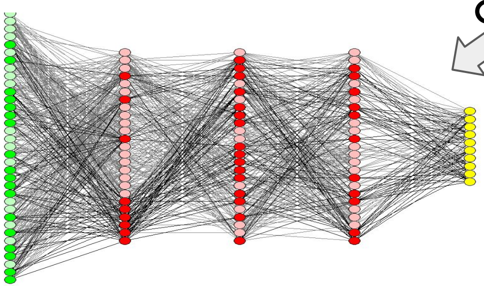
Features & Training data


DECISION TREE
decision nodes
root node
Do I have money for anything apart from the dress?
Yes
No
decline offer
Do I like the recommended product?
No
Yes
decline offer
Does the recommended product suit my outfit?
Yes
No
accept offer
decline offer
leaf nodes
Decision Tree:
Based on the Offer on various Product.. should i buy more?

A relatively weaker but explainable model, e.g. Decision Tree

Used to generate insights into what ML model is actually learning (rules). Used to identify any data/learning issues and improve it.

Optimise

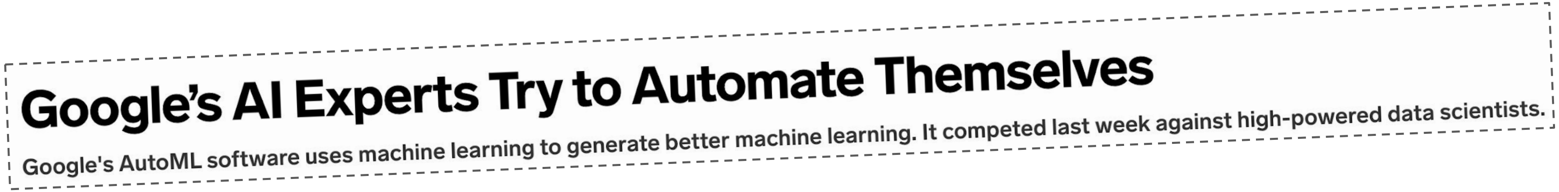


A high performance blackbox deep neural net

Used to generate production recommendations/decisions consumed by ML pipeline.

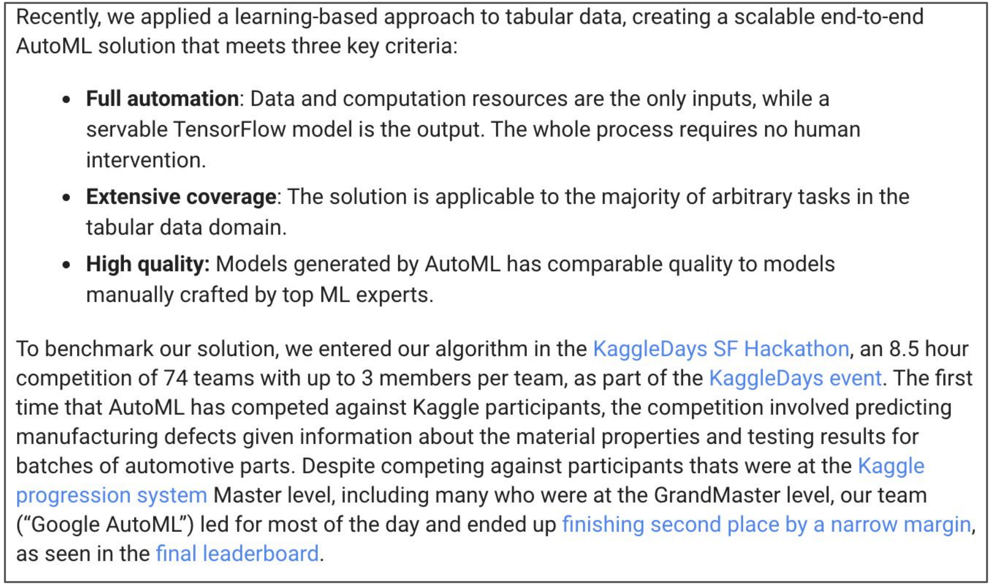
Where are we headed? The rise of AutoML

Automated machine learning (AutoML) is the process of automating the process of applying machine learning to real-world problems. AutoML covers the complete pipeline from the raw dataset to the deployable machine learning model.



Google's AI Experts Try to Automate Themselves

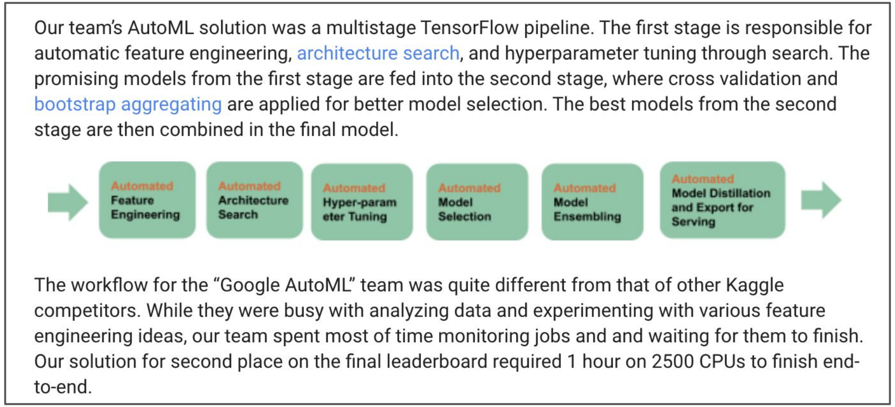
Google's AutoML software uses machine learning to generate better machine learning. It competed last week against high-powered data scientists.



Recently, we applied a learning-based approach to tabular data, creating a scalable end-to-end AutoML solution that meets three key criteria:

- **Full automation**: Data and computation resources are the only inputs, while a servable TensorFlow model is the output. The whole process requires no human intervention.
- **Extensive coverage**: The solution is applicable to the majority of arbitrary tasks in the tabular data domain.
- **High quality:** Models generated by AutoML has comparable quality to models manually crafted by top ML experts.

To benchmark our solution, we entered our algorithm in the KaggleDays SF Hackathon, an 8.5 hour competition of 74 teams with up to 3 members per team, as part of the KaggleDays event. The first time that AutoML has competed against Kaggle participants, the competition involved predicting manufacturing defects given information about the material properties and testing results for batches of automotive parts. Despite competing against participants thats were at the Kaggle progression system Master level, including many who were at the GrandMaster level, our team ("Google AutoML") led for most of the day and ended up finishing second place by a narrow margin, as seen in the final leaderboard.



Our team's AutoML solution was a multistage TensorFlow pipeline. The first stage is responsible for automatic feature engineering, architecture search, and hyperparameter tuning through search. The promising models from the first stage are fed into the second stage, where cross validation and bootstrap aggregating are applied for better model selection. The best models from the second stage are then combined in the final model.

Automated Feature Engineering
Automated Architecture Search
Automated Hyper-param eter Tuning
Automated Model Selection
Automated Model Ensembling
Automated Model Distillation and Export for Serving

The workflow for the "Google AutoML" team was quite different from that of other Kaggle competitors. While they were busy with analyzing data and experimenting with various feature engineering ideas, our team spent most of time monitoring jobs and and waiting for them to finish. Our solution for second place on the final leaderboard required 1 hour on 2500 CPUs to finish end-to-end.

AutoML has the potential to enhance the efforts of human developers and address a broad range of ML problems.

Source: Google AI blog, external coverage

Building a well shaped Data Science profile/resume

- Practicing problems @ Kaggle is good, but that should not be the only thing or the most sought after thing. As you have seen real world ML problems go beyond well-available, pre-cleaned data set. You need to go beyond boston housing, titanic survival, MNIST projects. Those are good to practice, but shouldn’t be the only thing in CV.
- Familiarise yourself with github and code versioning. Have a github profile with regular commits and add its link to your resume !
- The power of Linkedin !
- Publish research papers.
- Work on real world problems, e.g. ESA’s annual competition for advancements in space.
- Try to gain knowledge of gathering data, cleaning, building a model and productionising it [end to end ML].
- Go after internships, if it’s difficult to land a data science internship try virtual internships.
- Do certifications backed by universities (like on coursera) and/or companies (like MLCC).
- Python vs R vs SAS

Image source: link

Use of ML in cybersecurity

References:

- Kaspersky ML models
- Microsoft Defender Advanced Threat Protection ML model, variant 2
- Using static and dynamic analysis based features [link]