Using machine learning to try and predict taxi availability

Hari Allamraju
https://github.com/hari-allamraju

About me …

What this talk is about?

- Examples of machine learning with Python, using the taxi availability as a real world problem
- Code samples and walkthrough of multiple examples
- Assumes knowledge of Python and a general idea about working with data in Python

What we will not do

- We will not dive into the mathematics behind all the algorithms used - it is not necessary to know all the math to be able to implement simple machine learning
- We will not go into any neural networks or deep learning approaches - that would make this talk too complicated
- There are no examples of running the analysis on cloud machine learning platforms - this keeps it simple enough to run locally and later scale to any platform as you learn and add more

Let’s get started!

What is machine learning?

Quoting Wikipedia

Machine learning is the subfield of computer science that, according to Arthur Samuel, gives "computers the ability to learn without being explicitly programmed.” Samuel, an American pioneer in the field of computer gaming and artificial intelligence, coined the term "machine learning" in 1959 while at IBM.

Evolved from the study of pattern recognition and computational learning theory in artificial intelligence,machine learning explores the study and construction of algorithms that can learn from and make predictions on data – such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions, through building a model from sample inputs.

It’s already part of our daily life

How?

- We see machine learning in action whenever
    - We benefit from spam detection in email
    - We see targeted ads in social media and e-commerce
    - We experience surge pricing or receive coupons in ride sharing apps
    - Credit card fraud detection
    - And many such data driven user experiences

Doing your own machine learning

- Machine learning sounds very complex and daunting, but with some mathematical or a good technical background it is pretty easy to get started
- You don't need large computing power if you are dealing with small data sets
- You don't need to be a mathematical genius and come up with new algorithms
- And you don't need a lot of complicated data either if you are not trying to save the world
- There are tools which can give you a fairly good experience with the basics and get you prepped to take on bigger problems later

Tools in Python

- scikit-learn is probably the simplest Python library which you can use to get started with machine learning
- Along with numpy and matplotlib you can easily analyse and visualise the data and the algorithms
- This talk is based on analysis done using scikit-learn

But how do we get the data?

- Data is the key for any experiment with machine learning and you need good quality data to get started
- Fortunately there are a lot of open data sets that you can use, like the Divvy Bike Share data from Chicago or other data sets shared by various private and government agencies.
- You could even work with a local organization which is looking to provide data driven user experiences or make the best use of data that they might have collected as part of their operations
- But for this talk we will use something closer home - the Singapore Taxi Availability data

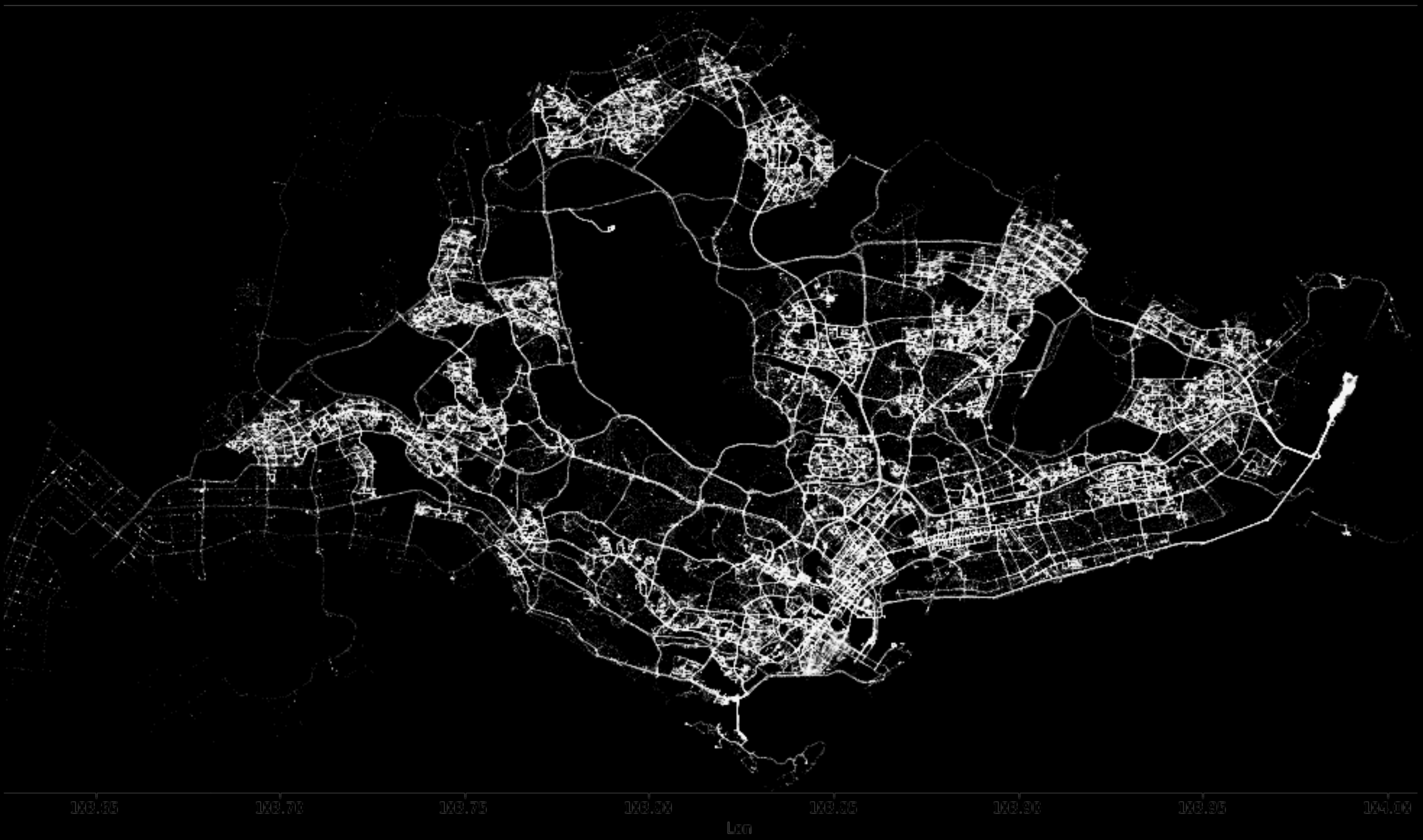
What data do we have for Singapore?

- Taxi availability is a very real world problem for which we wish there was an accurate answer. So its a good candidate to machine learn

- The Singapore LTA provides an API which we can invoke to get the locations of all free taxis in Singapore, as a list of coordinates at any given time

- This near real time (30 second or so delayed) data is pretty useful and at the same time not very complicated and fits easily for an exploration with machine learning

- The API is well documented on their website - https://data.gov.sg/dataset/taxi-availability



Taxi availability scatter plot

Data for a period of 20 mins

Preparing the data

- The data returned from the API is json and was stored to flat files.
- Data was collected for a period of one week, making the API call once every 5 mins
- The data was then parsed into a grid of approximately 20x35, grouping the taxi coordinates into cells with the count for that small area
- This data was then loaded to an SQLite DB with a simple schema that allowed to load into Pandas data frames, and eventually use in scikit-learn
- The raw data was about 500 MB in files and 100 MB when loaded to the database grid format

Links to the material

- The code used to fetch the data, load to the DB and the various utility methods we will see can be found here - https://github.com/hari-allamraju/sg-taxidata
- There are some utility scripts that can be put into cron straightaway
- There is a setup.py, so it can be installed from GitHub
- The notebooks we will walk through now can be found at - https://github.com/hari-allamraju/pycon-talk-taxidata

Enough slides, lets see code

What did we learn from the analysis?

- Its easy to get started with basic machine learning and predictions with small data sets
- There is no one golden algorithm and it varies with the data sets and subsets used
- The accuracy depends largely on the data that we are able to get and the variables we are able to capture in the data
- We would need other related information like weather, or was there an MRT delay that day or was it a peak holiday season etc

Can we get 100% accuracy?

Why not?

- Although we say our simple models will better fit the data as we add more features or use better algorithms, we might over fit the model to the data that we have. If we over fit the data, then it might not be able to accurately predict future data; small natural variations in the data that occur over a period of time will not fit with the rigid model and we will see errors

- There will always be a degree of error no matter how many features we collect - for processes that involve human beings, the general variation in behaviour creates a different result, and that cannot be measured 100%; and for fully automated processes there might be edge cases or external dependencies that cause different results.

- We can reduce the error as we accumulate more data to train the model and as we learn from the results, thus minimising future errors - deep learning and other neural network methods make use of this as they can have a feedback loop to learn from the prediction results

Questions!