Michal Grochmal

23a Beech Road, N11 2DA
London Borough of Enfield, United Kingdom
<mike@grochmal.org>

Likes to work with:

`python numpy pandas pytorch python-requests scikit-learn redis sqlalchemy flask javascript d3.js machine-learning linux linux-kernel archlinux archlinux-arm perl nginx c assembly disassembly unix http xss wireshark gdb scapy gimp inkscape imagemagick bash git amazon-web-services postgresql scipy`

Dislikes to work with:

`c# java visual-studio basic ms-access ms-word outlook windows sql-server`

Experience

Visiting Lecturer – City University London

`python, jupyter-notebook, ipython, numpy, pandas, matplotlib, scikit-learn`
Jan 2018 → Current

I'm teaching the short course in Data Analytics and Machine Learning at City University. The course starts from the basic skills of using numpy, pandas and family up to the automation of a machine learning project. Notably, we reach hyperparameter tuning methods and bias-variance and dataset size tradeoffs. The course is practical in nature, the objective is to demystify the idea that machine learning is a theoretical science with a lot of mathematical jargon. There is a good deal of mathematical jargon alright, since there is no way of escaping linear algebra when learning machine learning, but this is not the focus. Instead the focus of the course is in explaining that in real world projects one would try, and evaluate, several different models for a problem; and that it is untrue that more complex models are always the best solution. Over the last years we have completely rewritten the material for this short course, in two people. The old material was an agglomerate of several unconnected sources, after our changes we have a comprehensive and consistent material for the entire course. We are also writing a book based on this material, see the Projects section below for a link to it.

Assistant Vice President Programmer – Bank of America Merrill Lynch

`python, nosql, distributed-computing, microservices`
Jul 2017 → Sep 2019

We did ETL processing. ETL is always tricky, it involves a lot of requirements gathering and performance tuning. Data is always dirty, thinking that you will get clean data during ETL is always a mistake. BAML has a well integrated engine for risk processing, inside of which we perform the ETL to glue new and old systems of the bank together. I often call my job "welder" since my team is almost always writing the processes that make two separate systems share data. The risk engine stack is based on distributed python virtual machines running across thousands of machines, and has an integrated object database. Joining these two things together allows for building microservices quickly, and also allows to minimise boilerplate code on batch process scheduling and deployment.

Carer

Nov 2014 → Jun 2017

My mother got quite sick and I needed to stop my activities to care for her. Every second week she did undergo treatment which left her too weak to perform most activities, e.g. she could not cook food, make groceries or even get out of bed unaided on a couple of days after treatment. The examples also show my most common activity during this period.

Data Warehouse Architect – The MET Group
`postgresql, linux, amazon-web-services, redhat, django, python, bash, git, java`
Feb 2014 → Jul 2014

A data-warehouse is a project that needs knowledge on the full stack of technology, you need to optimise the low levels and you need a presentation on the high level. Loading and cleansing data is an easy task at first sight but nothing is as far from true. A data-warehouse often deals with real world data that is not crafted to abide by specific rules. The cleansing of data in a warehouse must handle errors gracefully, register and log errors including the entirety of erroneous data, and allow for semi-manual fixes of the encountered errors. Our small team (4 people) implemented the entire data-warehouse, including: Built administration processes for a data warehouse. Designed the data mining architecture, including visualisation. In a small team, we built a data warehouse from scratch, beginning from setting up the infrastructure.

Source Control and System Administrator – Fidelity National Information Services
`aix, hp-ux, redhat, c, perforce, perl, oracle`
Apr 2011 → Jun 2013

FIS was the company that acquired Metavante Technologies (see below), and, during the acquisition my role changed. FIS was composed of several teams that never talked to each other before, managing the source code for several teams was interesting in its own right in that environment. Moreover, the infrastructure our team had before the acquisition was alien to the sysadmins of the larger company. Quite understandable. I was responsible to keep it running for our own team. We used perforce as the Version Control System, perforce is a centralised version control engine. In most cases a decentralised VCS is preferred but under certain conditions a decentralised VCS might hinder rather than help. One of these conditions I discovered in this position: if you have teams that do not trust each other but shall deliver code together you can prevent a lot of trouble by separating their code. In the end that is a management issue but it ended being solved at the source control level. I worked on a couple of interesting projects during that time as well: Developed database migration scripts to normalize the database of our application with minimal downtime. Trained new teams to access and develop application code together with my existing team. Controlled the sources' access across a larger company.

C Developer – Metavante Technologies Inc.
`c, oracle, informix, plsql, aix, redhat, perl`
Apr 2010 → Apr 2011

My first real job after university, and I really cannot complain about even a single thing. I was thrown into real world SQL and low level optimisations, and thanks to many wonderful colleagues managed to learn a lot from it. I have learned the value of writing things in C: speed. Chipcard systems need to perform hundreds of transactions (and the verification PIN for most of these transaction) per second. Moreover, I got exposed to encryption, which lacked on my university path since I am originally a Physicist. The projects/highlights I worked with can be enumerated as: Developed active-active distributed architecture. Modernized system communications to new standards (new interfaces to other systems). Worked in a small team where each programmer needed to be self sufficient. Maintained a low level financial system, including direct communication with EMV chipcards. Maintained of a rule based system for fraud detection and fee calculation.

Education

PhD in Computer Science – Birkbeck, University of London
`pytorch, latex, slurm, numpy, jax, python, linux`
Nov 2019 → Current

I am continuing my PhD at Birkbeck. The research looks at the application of theoretical advances of Chaos theory. The notable applications can be seen in Connectivist Models. Connectivist Models present Chaotic behaviour. And recent advances to the training of such models is based on taming the Nonlinear and Nonextensive structure of the phase space of the learning of these models. Our objective is to build a framework where we can understand the training advances not as separate entities but as specific changes to the phase space of the learning paths.

MSc in Intelligent Technologies (distinction) – Birkbeck College University of London
`python, oracle, sql, r, libsvm, neural-network, numpy,`
`information-retrieval, cloud, data-warehouse, nosql, haskell, erlang`
2012 → 2014

An incredibly varied course, where the students could decide their path of learning. The Intelligent Technologies title is only one of the exit paths. Birkbeck is famous for the work on databases and we reviewed several of these. Moreover, we went through the future of databases: focusing on NoSQL options and map-reduce as complementing the use of classic relational options. For the machine learning part, instead of going into pure statistics we went all round the data science: data storage and collection, warehousing, feature retrieval and indexing of data that is not relational. Instead of making statistics on a known dataset we understood the implications of how the dataset is built. My project was about feature extraction from data that is not well structured: art. Based on several features I've built an SVM to classify paintings by style (school of art). With more work (and data) it will likely be capable to tell apart not only the school of art but also painters by their style. I got a distinction for the project and for the overall course. Birkbeck is a great environment for students. Rules are well defined, and so are deadlines. There is no micro-management of students, instead lecturers and tutors are there whenever you are stuck and ask for help. A really great place to study.

Faculty of Computational Physics – Jagiellonian University
`unix, linux, multithreading, pthreads`
2008 → 2009

This is where I really learned to program. The practical focus was on C programming on UNIX and all the UNIX system calls. That is a rather traditional approach but I'd argue that is a good approach to teach students how the computer actually works. Although the theoretical focus was in physics we did several topics common to the computer science curriculum, including concrete mathematics and operating systems. Computational Physics also deals with electronics, which might explain the focus on C programming.

Faculty of Physics – Federal University of Paraná
`unix, latex, matlab`
2006 → 2007

My introduction to programming happened on Debian, and I'm still very thankful to the guys at UFPR for this. Physics students were expected to use the scientific tools for physics, which meant emacs on the debian machines and write everything in LaTeX from the first day. Those were the days before stackoverflow and the best way to learn how to use a GNU/Linux machine was to have one at home. I still remember burning debian CDs and running home with them to install a system I never heard about before. I have not used MS Windows since.

Other Projects

Learn You some ML for great Good
https://learnyousomeml.com/
`jupyter, numpy, pandas, matplotlib, sklearn`

I am currently writing a book based on student feedback on the lectures I deliver. The attempt is to start from the Python libraries used for ML, instead of teaching the models and hoping that the reader will figure out the programming by himself. Data Scientists need to also be competent programmers, and know well the tools they use - that is the objective of the book.

fracdim
https://gitlab.com/grochmal/fracdim
`python, numpy, scipy`

Library on top of scipy to calculate the fractal dimension from the phase space path taken by a dynamic system. We use the correlation dimension method for counting. The library has a reasonably optimised regression and path sampling options. On top of fractal dimension the library is capable of estimating lyapunov coefficients for the dynamic systems. The lyapunov coefficients are counted by delay embedding.

lightviolin
https://gitlab.com/grochmal/lightviolin
`python, numpy, matplotlib`

A violin graph library on top of matplotlib. The intention is for a much lighter violin graph engine than the one used by seaborn, whilst keeping the option to graph different data on each side of a violin.

Personal Skills

Often I write code just for fun, and am often active on the GNU/Linux community. Otherwise I'm doing maths for fun, figuring out how statistics are defined in terms of linear algebra and calculus is my current challenge. I love *Vim* (vim is zen, vim is life, vim is great with pedals).

Machine Learning Knowledge

Good knowledge of Forests/Boosting, SVMs, ANNs, Genetic and Swarm techniques. Lots of experience with data mining and good experience with feature extraction. Some knowledge of the Semantic Web. I have been lecturing general Machine Learning and Statistics for quite while.

Research Tools

Python is one of my best programming languages, including numpy/scipy and flask. Good knowledge of publishing with LaTeX. Very good C skills and some experience with assembly, useful skills if heavy computations are needed.

General Programming

Administration of *nix systems, including web servers, and POSIX C programming on Linux, *BSD, AIX, HP-UX and Solaris systems. Good knowledge of Python, Haskell, Common Lisp, Scheme and Perl. Some knowledge of most open source languages (e.g. PHP, Clojure or Ruby). Full stack development of modern applications: from configuring HTTP parameters on apache/nginx to XSS flaws to jQuery. Experience on a plethora of database management systems.