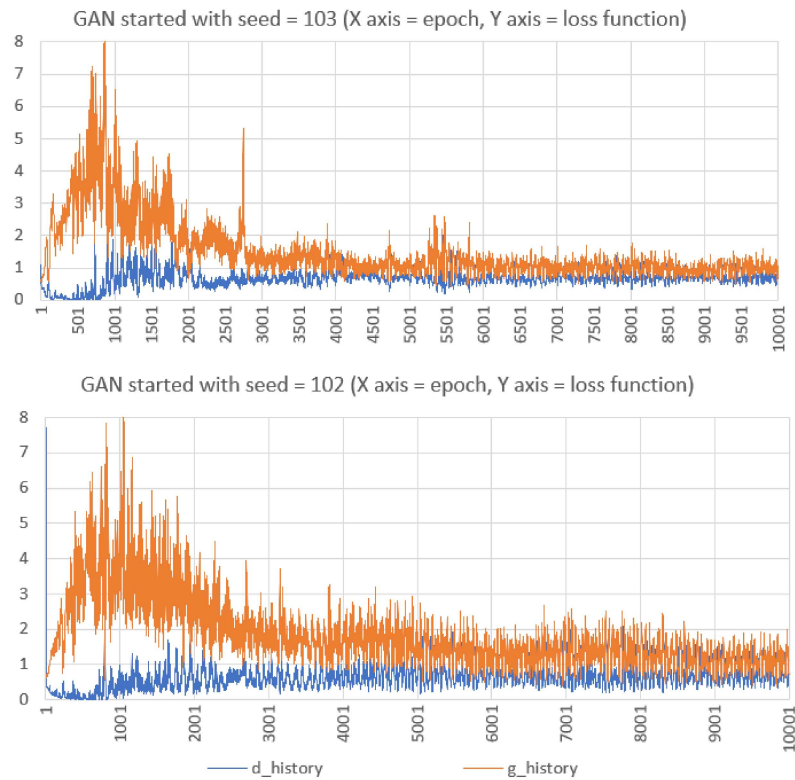

State-of-the-Art GenAI and LLMs Creative Projects, with Solutions



Preface

With tutorials, enterprise-grade projects with solutions, and real-world case studies, this coursebook covers state-of-the-art material on GenAI, generative adversarial networks (GAN), data synthetization, and much more, in a compact format with many recent references. It includes deep dives into innovative, ground-breaking AI technologies such as xLLMs (extreme LLMs), invented by the author.

The focus is on explainable AI with faster, simpler, high-performance, automated algorithms. For instance: NoGAN, new evaluation metrics, xLLM (self-tuned multi-LLM based on taxonomies with application to clustering and predictive analytics), variable-length embeddings, generating observations outside the training set range, fast probabilistic vector search, or Python-generated SQL queries. I also discuss alternatives to traditional methods, for instance to synthesize geo-spatial data or music.

Step 1: Summary tables. Besides **embeddings**, what are the summary tables and underlying data structure, for each of them? How are they linked to the output returned to a user query? How to adapt the methodology if the input source is Wikipedia rather than Wolfram?

Step 2: Issues with Python libraries. By looking at my comments in the code or by experimenting yourself, identify bad side effects from standard Python libraries such as **NLTK**. Examples include stop-words, auto-correct, and singularize. The problems arise due to the specialization of my xLLM, but is otherwise minor in generic applications targeted to non-experts. Suggest workarounds.

Step 3: Improving xLLM. I implemented several enhancements, for instance: ignoring **N -grams** (permutations of N tokens) [Wiki] not found in the crawled data, using normalized **PMI** in embeddings as the association metric or weight between two **tokens**, working with **variable-length embeddings** (VLE) rather than vectors, managing accented characters, not combining tokens separated by punctuation signs, and minimizing stemming such as singular form. Find other potential enhancements and how to implement them. For instance, making sure that “analysis of variance” and “ANOVA” return the same results, or “San Francisco” is not split into two tokens. Hint: use mappings (synonyms) and double-tokens treated as a single token. Treat uppercase and lowercase differently.

Figure 1: The first three steps in the main xLLM project

This textbook is an invaluable resource to instructors and professors teaching AI, or for corporate training. Also, it is useful to prepare for job interviews or to build a robust portfolio. And for hiring managers, there are plenty of original interview questions. The amount of Python code accompanying the solutions is considerable, using a vast array of libraries as well as home-made implementations showing the inner workings and improving existing black-box algorithms. By itself, this book constitutes a solid introduction to Python and scientific programming. The code is also on my GitHub repository.

About the author

Vincent Granville is a pioneering data scientist and machine learning expert, co-founder of Data Science Central (acquired by TechTarget), founder of [MLTechniques.com](https://mltechniques.com), former VC-funded executive, author and patent owner.



Vincent’s past corporate experience includes Visa, Wells Fargo, eBay, NBC, Microsoft, and CNET. Vincent is also a former post-doc at Cambridge University, and the National Institute of Statistical Sciences (NISS). He published in *Journal of Number Theory*, *Journal of the Royal Statistical Society* (Series B), and *IEEE Transactions on Pattern Analysis and Machine Intelligence*. He is also the author of multiple books, available [here](#). He lives in Washington state, and enjoys doing research on stochastic processes, dynamical systems, experimental math and probabilistic number theory.

Contents

1	Getting Started	5
1.1	Python, Jupyter Notebook, and Google Colab	5
1.1.1	Online Resources and Discussion Forums	6
1.1.2	Beyond Python	7
1.2	Automated data cleaning and exploratory analysis	7
1.3	Tips to quickly solve new problems	8
1.3.1	Original solution to visualization problem	8
1.3.2	New solution, after doing some research	10
2	Machine Learning Optimization	12
2.1	Fast, high-quality NoGAN synthesizer for tabular data	12
2.1.1	Project description	12
2.1.2	Solution	14
2.1.3	Python implementation	15
2.2	Cybersecurity: balancing data with automated SQL queries	21
2.2.1	Project description	22
2.2.2	Solution	22
2.2.3	Python code with SQL queries	23
2.3	Good GenAI evaluation, fast LLM search, and real randomness	24
2.3.1	Project description	25
2.3.2	Solution	27
2.3.3	Python implementation	29
2.4	Spectacular use of GenAI evaluation as an adaptive loss function	35
2.4.1	Project and solution	37
2.4.2	Python code	40
3	Time Series and Spatial Processes	46
3.1	Time series interpolation: ocean tides	46
3.1.1	Project description	47
3.1.2	Note on time series comparison	48
3.1.3	Solution	49
3.2	Temperature data: geospatial smoothness and interpolation	53
3.2.1	Project description	54
3.2.2	Solution	55
4	Scientific Computing	64
4.1	The music of the Riemann Hypothesis: sound generation	64
4.1.1	Solution	65
4.2	Cross-correlations in binary digits of irrational numbers	67
4.2.1	Project and solution	67
4.3	Longest runs of zeros in binary digits of $\sqrt{2}$	68
4.3.1	Surprising result about the longest runs	69
4.3.2	Project and solution	70
4.4	Quantum derivatives, GenAI, and the Riemann Hypothesis	72
4.4.1	Cornerstone result to bypass the roadblocks	73
4.4.2	Quantum derivative of functions nowhere differentiable	74
4.4.3	Project and solution	75
4.4.4	Python code	79
5	Generative AI	84
5.1	Holdout method to evaluate synthetizations	84
5.1.1	Project description	85

5.1.2	Solution	86
5.2	Enhanced synthetizations with GANs and copulas	89
5.2.1	Project description	89
5.2.2	Solution	91
5.2.3	Python code	92
5.3	Difference between synthetization and simulation	102
5.3.1	Frequently asked questions	102
5.3.2	Project: synthetizations with categorical features	103
5.3.3	Solution	104
5.4	Music, synthetic graphs, LLM, and agent-based models	105
6	Data Visualizations and Animations	106
6.1	Synthesizing data outside the observation range	106
6.1.1	Animated histograms for extrapolated quantiles	106
6.1.2	Python code: video, thumbnails	107
6.2	Curve fitting in bulk	110
6.2.1	Regression without dependent variable	111
6.2.2	Python code	111
6.3	Gradient descent: grid, orthogonal trajectories	116
6.4	More spectacular videos and image generation	121
7	NLP and Large Language Models	123
7.1	Synthesizing DNA sequences with LLM techniques	123
7.1.1	Project and solution	123
7.1.2	Python code	126
7.2	Creating high quality LLM embeddings	130
7.2.1	Smart, efficient, and scalable crawling	130
7.2.2	User queries, category-specific embeddings and related tables	134
7.2.3	RAG: retrieval augmented generation using book catalogs	143
8	Miscellaneous Projects	150
8.1	Fast probabilistic nearest neighbor search (pANN)	150
8.1.1	Motivation and architecture	151
8.1.2	Applications	152
8.1.3	Project and solution	154
8.1.4	Python code	155
8.2	Building and evaluating a taxonomy-enriched LLM	159
8.2.1	Project and solution	161
8.2.2	Python code	162
8.3	Predicting article performance and clustering using LLMs	169
8.3.1	Project and solution	170
8.3.2	Visualizations and discussion	172
8.3.3	Python code	173
A	Glossary: GAN and Tabular Data Synthetization	182
B	Glossary: GenAI and LLMs	186
C	Introduction to Extreme LLM and Customized GPT	189
C.1	Best practices	189
C.2	Python utility for xLLM	192
C.3	Comparing xLLM with standard LLMs	198
C.4	Comparing xLLM5 with xLLM6	199
C.4.1	Conclusion	200
	Bibliography	202
	Index	204