OliverWyman

GPT-3 AND THE ACTUARIAL LANDSCAPE

An Overview of Large Language Models and Applications

CAS RPM Seminar
March 2023

A business of Marsh McLennan

1

CONFIDENTIALITY

2

MEET THE SPEAKERS

OLIVIER BROWN, FCAS, MAAA

P&C Insurance Practice: Principal
Olivier.Brown@oliverwyman.com

Olivier Brown leads P&C Actuarial Innovation within Oliver Wyman's Actuarial practice. Olivier helps P&C insurers modernize their actuarial functions. He has worked with many insurers to improve their pricing sophistication, ratemaking, reserving, and underwriting analytics.

HUGO LATENDRESSE, FCAS

P&C Insurance Practice: Senior Manager
Hugo.Latendresse@oliverwyman.com

Hugo Latendresse is a Senior Manager in Oliver Wyman's Actuarial practice. He specializes in machine learning and automation solutions designed to improve insurance processes. Hugo has seven years of predictive analytics experience in pricing, reserving, claims, and underwriting.

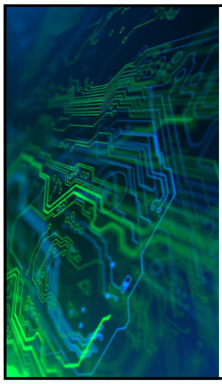
SABRINA TAN, ACAS

P&C Insurance Practice: Consultant
Sabrina.Tan@oliverwyman.com

Sabrina Tan is a Consultant in Oliver Wyman's Actuarial practice. She provides P&C actuarial consulting services to a variety of insurance organizations. She has worked on various projects in predictive analytics, process improvement, pricing, and reserving.

 3

3



OVERVIEW

1 Intro to Natural Language Processing

2 The Building Blocks of GPT

3 Actuarial Applications

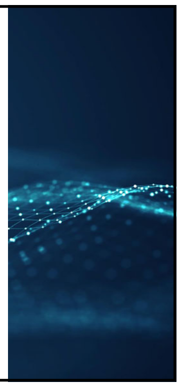
4 AI: Software 2.0

5 Recap

4

4

1
INTRO TO NATURAL LANGUAGE PROCESSING



5

RECENT KEY INNOVATIONS HAVE ACCELERATED ADVANCEMENTS IN NLP

PRE-2000

Rule-based systems to simple statistical models
- First application of natural language processing (NLP) was for machine translation
- Initial rule-based models required significant manual coding
- Machine learning and statistical models (N-grams, Markov models) and the first recurrent neural networks such as long short-term memory models replaced hard-coded rules.

2000-2017

Use of neural networks for language modeling
- Initial uses of neural networks for next word prediction
- First representations of words with dense vectors called *word embeddings* and algorithms capable of learning them efficiently (Word2Vec)

2017

Attention, transformers, and large language models
- The attention mechanism along with transformer architecture enable state-of-the-art performance on language tasks and efficient process of large datasets
- Ability to consider context in texts increased the ability to produce human-like texts

2018+

Generative pre-trained transformers (GPT)
- OpenAI releases first version of GPT language model (2018), with GPT-2 and GPT-3 released each year thereafter
- ChatGPT, fined-tuned on GPT-3.5, launched in 2022

Source: https://medium.com/nlplanet/a-brief-timeline-of-nlp-bc45b640f07d

6

6

ACCESS TO POWERFUL RESOURCES ENABLE LARGE LANGUAGE MODELS

NLP has achieved groundbreaking results through LLMs, enabled by various modern technology

- Increasing **availability of text data** from the internet
- Development of powerful **computational resources** (GPUs and TPUs)
- **Frameworks** for developing neural networks (TensorFlow and PyTorch)
- Advances in **ML algorithms** (transformers and attention)

	Number of parameters	Size of training dataset (Quantity of text)	Compute resources used for training
BERT	110M	16GB	
GPT	117M	40GB	
ROBERTA	125M	160GB	
GPT-2	1.5B	800GB	
GPT-3	175B	45TB	3,600+ GPU days 330+ MWh

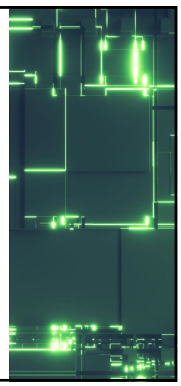
© Oliver Wyman

Source: https://huggingface.co/transformers/v2.4.0/pretrained_models.html 7

7

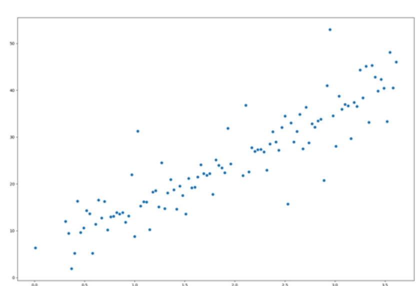
2

THE BUILDING BLOCKS OF GPT



8

MACHINE LEARNING 101: GRADIENT DESCENT



9

9

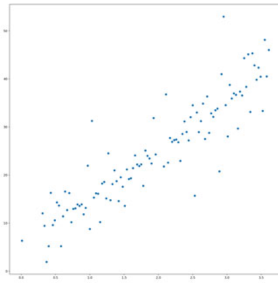
MACHINE LEARNING 101: GRADIENT DESCENT

Linear Regression Model Using Formulas

MODEL STRUCTURE $Y = \beta_0 + \beta_1 * X$

COST FUNCTION Cost = Σ(predicted – actual)^2

FORMULAS FOR β COEFFICIENTS
Slope
$\beta_1 = (n * \Sigma(x*y) - \Sigma(x) * \Sigma(y)) / (n * \Sigma(x^2) - (\Sigma(x))^2)$
Intercept
$\beta_0 = (\Sigma(y) - \beta_1 * \Sigma X) / N$



10

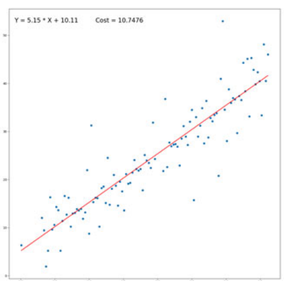
MACHINE LEARNING 101: GRADIENT DESCENT

Linear Regression Model Using Formulas

MODEL STRUCTURE $Y = \beta_0 + \beta_1 * X$

COST FUNCTION Cost = Σ(predicted – actual)^2 = 10.7476

FORMULAS FOR β COEFFICIENTS
Slope
$\beta_1 = (n * \Sigma(x*y) - \Sigma(x) * \Sigma(y)) / (n * \Sigma(x^2) - (\Sigma(x))^2) = 10.11$
Intercept
$\beta_0 = (\Sigma(y) - \beta_1 * \Sigma X) / N = 5.15$



11

MACHINE LEARNING 101: GRADIENT DESCENT

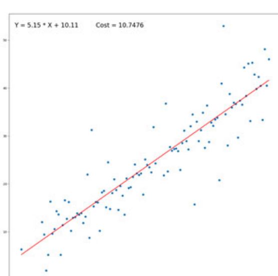
Linear Regression Model Using Formulas

MODEL STRUCTURE $Y = \beta_0 + \beta_1 * X$

COST FUNCTION Cost = Σ(predicted – actual)^2 = 10.7476

FORMULAS FOR β COEFFICIENTS
Slope
$\beta_1 = (n * \Sigma(x*y) - \Sigma(x) * \Sigma(y)) / (n * \Sigma(x^2) - (\Sigma(x))^2) = 10.11$
Intercept
$\beta_0 = (\Sigma(y) - \beta_1 * \Sigma X) / N = 5.15$

RESULTING MODEL Y = 10.11 * X + 5.15



12

MACHINE LEARNING 101: GRADIENT DESCENT

Linear Regression Model Using Formulas

MODEL STRUCTURE	$Y = \beta_0 + \beta_1 * X$
COST FUNCTION	Cost = Σ(predicted – actual)^2 = 10.7476
FORMULAS FOR β COEFFICIENTS	**Slope** $\beta_1 = (n * \Sigma(x*y) - \Sigma(x) * \Sigma(y)) / (n * \Sigma(x^2) - (\Sigma(x))^2) = 10.11$ **Intercept** $\beta_0 = (\Sigma(y) - \beta_1 * \Sigma X) / N = 5.15$
RESULTING MODEL	Y = 10.11 * X + 5.15

Without those formulas,

How can we find the coefficients?

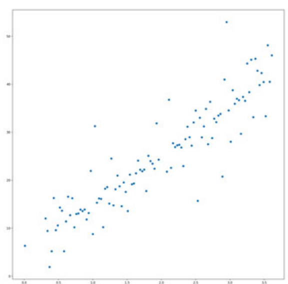
13

13

MACHINE LEARNING 101: GRADIENT DESCENT

Linear Regression Model Using Gradient Descent

MODEL STRUCTURE	$Y = \beta_0 + \beta_1 * X$
COST FUNCTION	Cost = Σ(predicted – actual)^2
FORMULAS FOR β COEFFICIENTS	None!

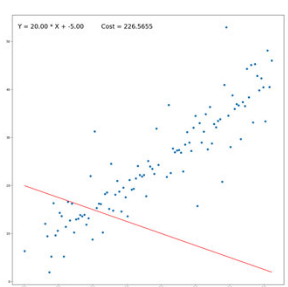


14

MACHINE LEARNING 101: GRADIENT DESCENT

Linear Regression Model Using Gradient Descent

MODEL STRUCTURE	$Y = \beta_0 + \beta_1 * X$
COST FUNCTION	Cost = Σ(predicted – actual)^2
FORMULAS FOR β COEFFICIENTS	None!

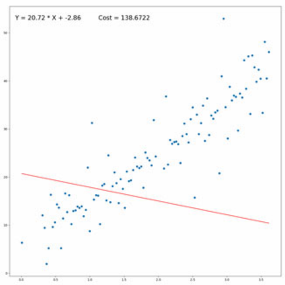


15

MACHINE LEARNING 101: GRADIENT DESCENT

Linear Regression Model Using Gradient Descent

MODEL STRUCTURE	$Y = \beta_0 + \beta_1 * X$
COST FUNCTION	Cost = Σ(predicted – actual)^2
FORMULAS FOR β COEFFICIENTS	None!



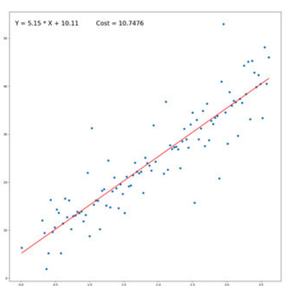
16

MACHINE LEARNING 101: GRADIENT DESCENT

Linear Regression Model Using Gradient Descent

MODEL STRUCTURE	$Y = \beta_0 + \beta_1 * X$
COST FUNCTION	Cost = Σ(predicted – actual)^2
FORMULAS FOR β COEFFICIENTS	None!
RESULTING MODEL	Y = 10.11 * X + 5.15

Through the two coefficients, the model "remembers" the data.



17

REGRESSION VS CLASSIFICATION

	Linear Regression	Multivariate Linear Regression	Logistic Regression
MODEL STRUCTURE	$Y = \beta_0 + \beta_1 * X$	$Y = \beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \beta_2 * X_2 + \beta_3 * X_3 + \ldots$	$Y = 1 / (1 + \exp(-(\beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \ldots)))$ $Y = \text{sigmoid}(\beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \ldots)$
COST FUNCTION	**Sum of squared error** Σ(predicted – actual)^2	**Sum of squared error** Σ(predicted – actual)^2	**Cross-entropy** Σ(-(actual * log(predicted) + (1 - actual) * log(1 - predicted)))
	Prediction Y can be any number	Prediction Y can be any number	Prediction Y is between 0 and 1
	Example: predict the sell price of a house using one variable (such as square footage)	**Example:** predict the sell price of a house using multiple variables (square footage, year of construction, etc.)	**Example:** predict whether a flower is of a certain species based on petal length and width

 18

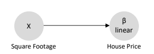
18

REGRESSION VS CLASSIFICATION

Linear Regression

MODEL STRUCTURE $Y = \beta_0 + \beta_1 * X$

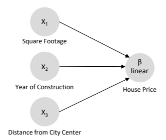
Input | **Output**



Multivariate Linear Regression

MODEL STRUCTURE $Y = \beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \beta_2 * X_2 + \beta_3 * X_3 + \dots$

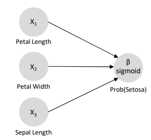
Input Layer | **Output**



Logistic Regression

MODEL STRUCTURE $Y = 1 / (1 + \exp(-(\beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \dots)))$
$Y = \text{sigmoid}(\beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \dots)$

Input Layer | **Output**



 19

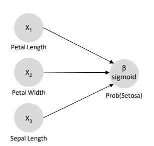
19

SINGLE-LABEL VS MULTI-LABEL CLASSIFICATION

Single-Label Classification

MODEL STRUCTURE $Y = \text{sigmoid}(\beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \dots)$

Input Layer | **Output**



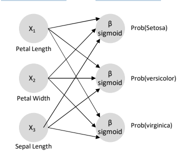
Multi-Label Classification Training Data Example

Petal Length	Petal Width	Sepal Length	Species
5.4	3.9	1.3	Setosa
4.5	2.3	1.3	Setosa
4.4	3.2	1.3	Setosa
4.8	3.0	1.4	Setosa
5.1	3.8	1.6	Setosa
4.6	3.2	1.4	Setosa
5.3	3.7	1.5	Setosa
5.0	3.3	1.4	Setosa
7.0	3.2	4.7	Versicolor
6.4	3.2	4.5	Versicolor
6.9	3.1	4.9	Versicolor
5.6	2.7	4.2	Versicolor
5.7	3.0	4.2	Versicolor
5.7	2.9	4.2	Versicolor
6.2	2.9	4.3	Versicolor
5.1	2.5	3.0	Versicolor
5.7	2.8	4.1	Versicolor
6.3	2.5	5.0	Virginica
6.5	3.0	5.2	Virginica
6.2	3.4	5.4	Virginica
...	...	...	...
5.9	3.0	5.1	Virginica

Multi-Classification Training Data Example

MODEL STRUCTURE $Y_1 = \text{sigmoid}(\beta_{10} + \beta_{11} * X_1 + \beta_{12} * X_2 + \dots)$
$Y_2 = \text{sigmoid}(\beta_{20} + \beta_{21} * X_1 + \beta_{22} * X_2 + \dots)$
$Y_3 = \text{sigmoid}(\beta_{30} + \beta_{31} * X_1 + \beta_{32} * X_2 + \dots)$

Input Layer | **Output Layer**



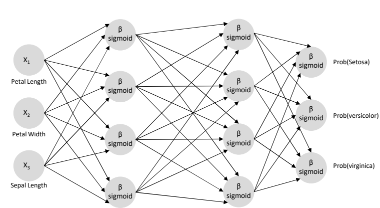
 20

20

NEURAL NETWORKS

Multi-Layer Perceptron

Input Layer | **Hidden Layer #1** | **Hidden Layer #2** | **Output Layer**



MODEL STRUCTURE The number of nodes in the hidden layer is chosen by the modeler.

COST FUNCTION **Cross-entropy**
Σ(-(actual * log(predicted) + (1 - actual) * log(1 - predicted)))

MODEL PARAMETERS All β parameters are initially set a random. The model adjusts those parameters to minimize the cost function.

 21

21

BUT WHAT ABOUT PREDICTING WORDS?

22

NEXT WORD PREDICTION

Fundamentally, GPT-3 and ChatGPT are neural networks that constantly give a probability to what should be the next outputted word. That's why ChatGPT types one word at a time!

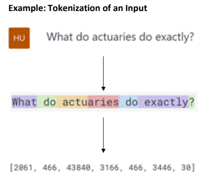
First Step: Tokenization

- First step of NLP any model is to convert text into numbers, or "tokens".
- GPT-3's tokenizer assign integers to chunks of characters.
- It's a one-to-one mapping, fixed mapping.
  - In the input layer, "exactly" will always be mapped to the number 3446
  - In the output layer, 3446 will always be mapped to "exactly"

Classification Problem

- Next word prediction becomes a classification problem
- Input: series of tokens (a sentence)
- Output: probability distribution over all tokens
- Vocab size of GPT-3 = 50,257
- The problem becomes a classification problem with 50,257 labels

Example: Tokenization of an Input



Source: https://platform.openai.com/tokenizer

 23

23

SUMMARIZING MEANING AND REDUCING DIMENSIONALITY WITH WORD EMBEDDINGS

How to quantify meanings of words?

- Token IDs cannot be used as-is.
- Word Embedding: a large vector assigned to each token
- Values in the vector are initially assigned at random

Word Embedding Examples

Token	Token ID	One-Hot Encoded Vector (50,000 dimensions)	Word Embedding Vector (fewer dimensions)
round	35634	(0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, ..., 0, 0, 0, 0, 0)	(0.932, 0.321, 0.456, 0.571, 0.984, ..., 0.654)
ball	1894	(0, 0, 0, 0, 0, 0, ..., 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0)	(0.524, 0.329, 0.132, 0.134, 0.952, ..., 0.213)
net	3262	(0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0)	(0.187, 0.818, 0.118, 0.901, 0.347, ..., 0.221)

 24

24

REPRESENTING ORDER OF WORDS WITH POSITIONAL ENCODING

Network nodes need to consider multiple tokens at once. How to do that?

A naïve approach of simply taking an average or a sum of all word embedding vectors would be wrong for two reasons.

- First, obvious reason: the order of the tokens need to be considered.

Solution: Positional Encoding (see below)

- Second, less obvious reason: some words "care" more about each other than others.

Solution: Self-Attention (see next slides)

Token	Word Embedding		Positional Encoding		*The resulting vectors represent both the meaning and position of tokens.*
Name	(0.638, 0.759, 0.905, 0.243, 0.189, ..., 0.900)	+	(0, 1, 0, 1, 0, ..., 0)	=	(0.638, 1.759, 0.905, 1.243, 0.189, ..., 0.900)
the	(0.655, 0.325, 0.599, 0.91, 0.49, ..., 0.726)	+	(0.031, 1.000, 0.003, 1.000, 0, ..., 0)	=	(0.686, 1.324, 0.602, 1.909, 0.490, ..., 0.726)
capital	(0.082, 0.326, 0.622, 0.418, 0.136, ..., 0.344)	+	(0.062, 0.998, 0.000, 1.000, 0, ..., 0)	=	(0.144, 1.324, 0.622, 1.418, 0.136, ..., 0.344)
of	(0.194, 0.294, 0.796, 0.07, 0.726, ..., 0.56)	+	(0.094, 0.995, 0.000, 1.000, 0, ..., 0)	=	(0.288, 1.289, 0.796, 1.07, 0.726, ..., 0.560)
Peru	(0.825, 0.943, 0.828, 0.611, 0.912, ..., 0.962)	+	(0.125, 0.992, 0.000, 1.000, 0, ..., 0)	=	(0.95, 1.935, 0.828, 1.611, 0.912, ..., 0.962)

 25

25

ATTENTION IS ALL YOU NEED

Self-Attention is the mechanism used by transformer models to weigh the importance of difference words in a sentence or piece of text based on their relationships to other words.

Motivation for Self-Attention

"I can enjoy almost any music genre, but I was never enthusiastic about heavy ____."
"I run instead of lifting, because my apartment building's gym doesn't have heavy ____."

In the two sentences above:

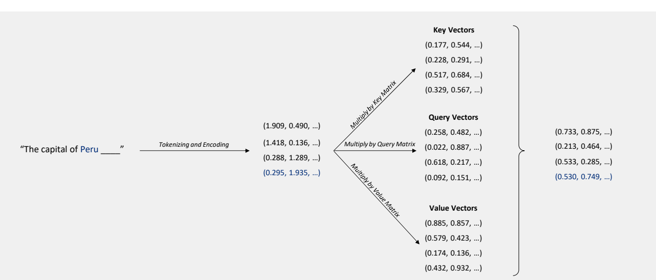
- The words "music" and "lifting" give a lot of meaning to the token "heavy", since those tokens help specify the context.
- The words "enthusiastic" and "apartment", however are not very useful in finding out what is "heavy".

Therefore, we want the next word predictions to highly depend on "music" and "lifting" and not so much on "enthusiastic" and "apartment".

 26

26

CREATING KEYS, QUERIES, AND VALUES TO ALLOW SELF-ATTENTION CALCULATION



 27

27

COMBINING KEYS, QUERIES, AND VALUES IN SELF-ATTENTION

	A. Key	B. Query	C. Value	D. Unnormalized Weights	E. Normalized Weights	F. New Representation of "Peru"
Preceding Tokens	Key Matrix x Previous Representation	Query Matrix x Previous Representation	Value Matrix x Previous Representation	Key x "Peru" Query	softmax(D.)	weighted average of C.
The	(0.177, 0.544, ...)	(0.258, 0.482, ...)	(0.885, 0.857, ...)	1.798	14%	(0.530, 0.749, ...)
capital	(0.228, 0.291, ...)	(0.022, 0.887, ...)	(0.579, 0.423, ...)	2.501	29%	
of	(0.517, 0.684, ...)	(0.618, 0.217, ...)	(0.174, 0.136, ...)	0.421	4%	
Peru	(0.329, 0.567, ...)	(0.092, 0.151, ...)	(0.432, 0.932, ...)	3.113	53%	

- Matrices used to obtain keys, queries, and values are common to all tokens.
- They are initialized at random and trained using gradient descent.

Queries: vector describing what each token cares about

Keys: vector describing what each token can inform about

Value: vector describing information each token has to offer

 28

28

TRANSFORMER ARCHITECTURE EXAMPLE

HU Why actuaries

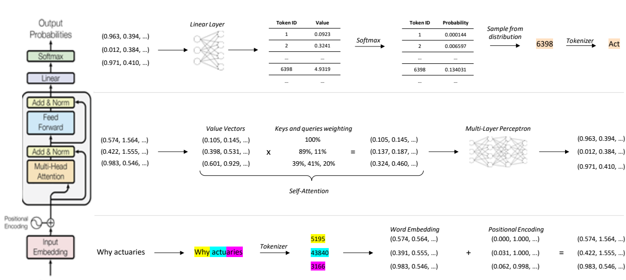
Actuaries are professionals who use mathematical and statistical techniques to analyze financial risk and uncertainty. They are often employed by insurance companies, pension funds, and other financial institutions to assess the likelihood of future events and the associated financial impact.

Source: chat.openai.com/chat

 29

29

TRANSFORMER ARCHITECTURE EXAMPLE



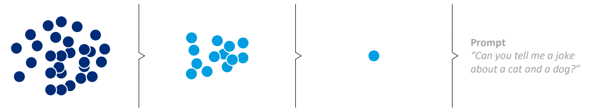
Source: arxiv.org/pdf/1706.03762.pdf

 30

30

GPT IS CAPABLE OF ZERO-SHOT LEARNING

FINE-TUNING	FEW-SHOT LEARNERS	ONE-SHOT LEARNERS	ZERO-SHOT LEARNERS
Update weights of pre-trained model by training on a dataset specific to the desired task	Model is given a few demonstrations of the task as conditioning, but no weight updates are allowed	Same as few-shot but only one demonstration is allowed	No demonstrations are allowed – the model is only given a natural language description of the task

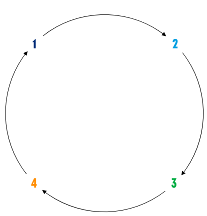


Source: https://arxiv.org/pdf/2005.14165.pdf

 31

31

ADDING REINFORCEMENT LEARNING LAYERS AND A MODERATION API ENABLES THE TRANSITION FROM GPT TO CHATGPT



1 **Fine-tune GPT-3.5**
ChatGPT focused language model that has been fine-tuned on conversational data such as short, informal sentences and specific conversational conventions.

2 **Train a reward model**
A labeler ranks possible responses to prompts, and this data is used to train a reward model to determine the final response.

3 **Use reinforcement learning to optimize reward**
An agent learns to choose the best response to a prompt by receiving feedback in the form of the rewards from step 2.

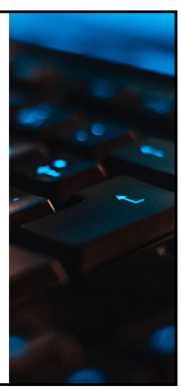
4 **Moderation endpoint**
A separate language model is used to classify text as whether they violate content policy by being "sexual, hateful, violent, or promoting self-harm".

Sources:
https://openai.com/blog/chatgpt/
https://openai.com/blog/new-and-improved-content-moderation-tooling/

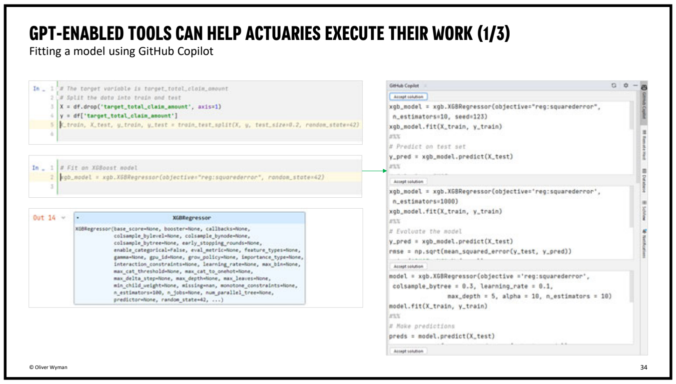
 32

32

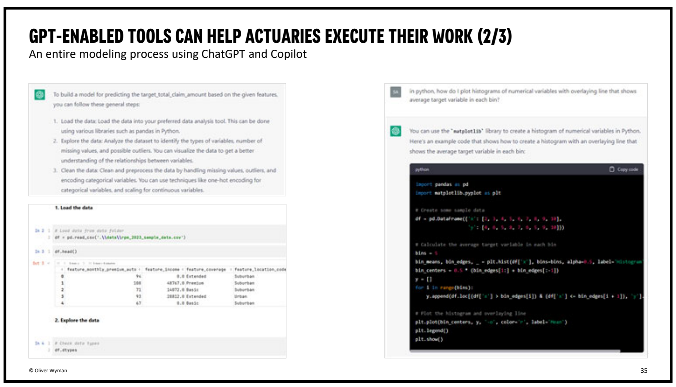
3
ACTUARIAL APPLICATIONS



33


GPT-ENABLED TOOLS CAN HELP ACTUARIES EXECUTE THEIR WORK (1/3)
Fitting a model using GitHub Copilot

34


GPT-ENABLED TOOLS CAN HELP ACTUARIES EXECUTE THEIR WORK (2/3)
An entire modeling process using ChatGPT and Copilot

35


GPT-ENABLED TOOLS CAN HELP ACTUARIES EXECUTE THEIR WORK (3/3)
Using ChatGPT to debug code

36

GPT OPENS THE DOOR TO INNOVATIVE SOLUTIONS FOR SEVERAL INSURANCE PROCESSES

Webscraping for commercial lines underwriting
Streamline quoting process by automating capture of potential policyholder information

Analysis of unstructured claims data
Classify/label unstructured data in claims to gather insights from documents such as medical reports

Summarizing and searching policy contracts
Identify key provisions and search to identify specific clauses or provisions

Fraud detection
Analyze data from alternative sources to identify potential risks through detection of anomalies

Customer service
Chatbots powered by GPT models can understand natural language and provide personalized responses

Actuarial communication and report generation
Generate text to support actuarial analyses and draft reports

 37

37

LIMITATIONS

GPT-SPECIFIC LIMITATIONS

- GPT-3 is proprietary. It would be expensive to use the API in production if thousands of requests are made per day
- Insurance data often private and data can be sensitive/restricted
- Output of a general purpose LLM can rarely be used as-is. Additional layers have to be built. Classification into specific categories, checks for model inaccuracy, conversion of model output (English sentences) into tabular data

CHATGPT LIMITATIONS

- ChatGPT can be confidently wrong; the system can write "plausible-sounding but incorrect or nonsensical answers"
- Can be sensitive to the phrasing of the prompt
- Models do not ask clarifying questions when a prompt is unclear and instead guesses the intent of the user
- It is possible for the model to respond to "harmful instructions or exhibit biased behavior"
- Supervision and adjustments are often needed

GENERAL LIMITATIONS OF LLMS

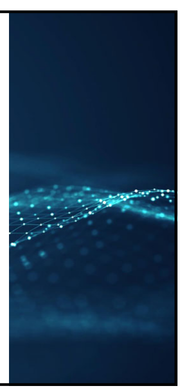
- LLMs are computationally expensive to train and run and require vast amounts of resources
- Explainability and interpretability: can be considered a black box since these models are highly complex
- Can perpetuate biases present in the data they are trained on, which can lead to unfair or inaccurate predictions
- Requires high level of technical expertise to implement, maintain and use

Source: https://openai.com/blog/chatgpt

 38

38

4
AI: SOFTWARE 2.0



39

SOFTWARE IS EATING THE WORLD, AI IS EATING SOFTWARE

“

The “classical stack” of **Software 1.0** is what we’re all familiar with — it is written in languages such as Python, C++, etc. It consists of explicit instructions to the computer written by a programmer. By writing each line of code, the programmer identifies a specific point in program space with some desirable behavior. […]

In contrast, **Software 2.0** is written in much more abstract, human unfriendly language, such as the weights of a neural network. […]

Software (1.0) is eating the world, and now AI (Software 2.0) is eating software.

ANDREJ KARPATHY
Founding Member of OpenAI
Former Director of AI at Tesla

HOWEVER...

- Applying AI to insurance and actuarial problems does not happen automatically.
- There is a lot of work needed to convert our industry to Software 2.0.
- Who will do this work?
- We think actuaries are ideally suited to lead this work.

Source: https://karpathy.medium.com/software-2-0-a64152b37c35

 40

40

WHERE TO START?

Modern software development practices are the foundation; Actuaries can learn a lot from the software world

Design: “Simplicity is the ultimate sophistication”
One should fall in love with the problem rather than any given solution. Once the problem is understood, drafts should be presented to users before rushing to the development phase.

Agility: Learn to “fail fast” and adapt
It is essential to interact frequently with end users and adjust the trajectory based on their feedback.

Testing: Foresee bugs and defects before users
Automatic and timely testing of the whole code base for compliance with expected behavior should be in place.

Version Control: Keep track of all changes
Allow collaborative development by tracking changes of individual contributors and setting frameworks for integration.

Modularity: Reduce work duplication
Maximize code understandability and reusability by spreading functionalities into independent components.

Continuous Integration: Scale the collaboration
Frequent integration of all new code that compose the application, leveraging automated testing and building functionalities.

Put together, these best practices ensure that code will remain easy to understand and maintain over time. It makes it easier to implement new functionality and integrate new technologies.

 41

41

5
RECAP

42

RECAP

- We've seen exponential growth in the complexity of machine learning models, which is largely attributable to the use of deep learning techniques.
- Transformer models, including GPTs, have resulted in breakthrough performance on NLP tasks; the process of "self-attention" has been pivotal to this breakthrough.
- These breakthroughs impact all fields of work, including insurance and actuarial work.
- Converting our industry to a Software 2.0 world will require a lot of work. Actuaries are well suited to lead this work but need to modernize their skillset.

 43

43

QUESTIONS

44

QUALIFICATIONS, ASSUMPTIONS, AND LIMITING CONDITIONS

This report is for the exclusive use of the Oliver Wyman client named herein. This report is not intended for general circulation or publication, nor is it to be reproduced, quoted, or distributed for any purpose without the prior written permission of Oliver Wyman. There are no third-party beneficiaries with respect to this report, and Oliver Wyman does not accept any liability to any third party.

Information furnished by others, upon which all or portions of this report are based, is believed to be reliable but has not been independently verified, unless otherwise expressly indicated. Public information and industry and statistical data are from sources we deem to be reliable; however, we make no representation as to the accuracy or completeness of such information. The findings contained in this report may contain predictions based on current data and historical trends. Any such predictions are subject to inherent risks and uncertainties. Oliver Wyman accepts no responsibility for actual results or future events.

The opinions expressed in this report are valid only for the purpose stated herein and as of the date of this report. No obligation is assumed to revise this report to reflect changes, events, or conditions, which occur subsequent to the date hereof.

All decisions in connection with the implementation or use of advice or recommendations contained in this report are the sole responsibility of the client. This report does not represent investment advice nor does it provide an opinion regarding the fairness of any transaction to any and all parties. In addition, this report does not represent legal, medical, accounting, safety, or other specialized advice. For any such advice, Oliver Wyman recommends seeking and obtaining advice from a qualified professional.

45

OliverWyman

A business of Marsh McLennan

46