CODAMETRIX

Lone Star HFMA Webinar

Revolutionizing Medical Coding with AI: Overcoming Challenges and Achieving Accuracy

Jay Aslam, PhD

CodaMetrix

Co-founder, Chief Data Science Officer

Outline

Motivation

- Why focus on coding and charge capture?
- Why use AI?
- Governing principles

Understanding the AI landscape

- AI, NLP, NLU, ML, DL, LLM/ChatGPT
- A gentle introduction to Machine Learning

An ML-based medical coding automation solution

Case-studies of real-world implementations

Why Focus on Coding and Charge Capture?

Coding is the most costly part of the RCM process

- Involving MDs does not make financial sense
- ~25% of RCM cost
- Coders are in high demand, highly compensated and in short supply
- Continuous training on new codes and regulations; certification and credentialing are basically required

It is high impact and getting it wrong affects cost and revenue

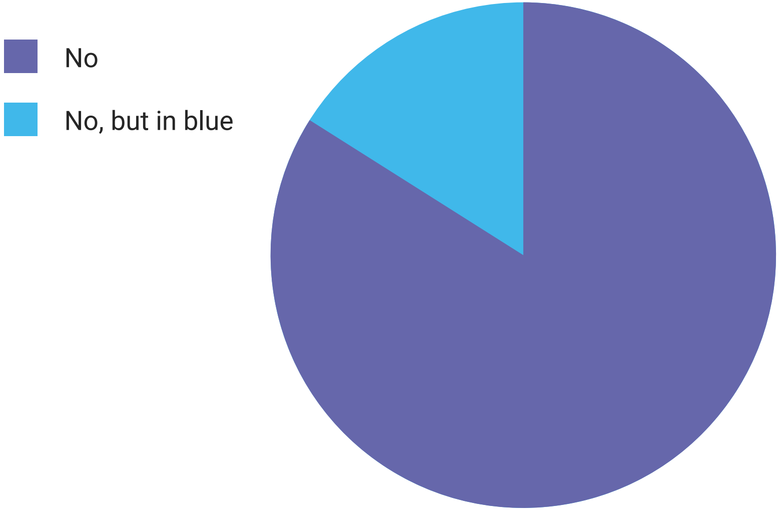
- Coding errors can lead to under-coding and lower revenue, and over-coding can lead to rejected claims, reprocessing or expensive penalties
- Coding is a part of the administrative burden experienced by providers, which results in less time with patients and lower yield

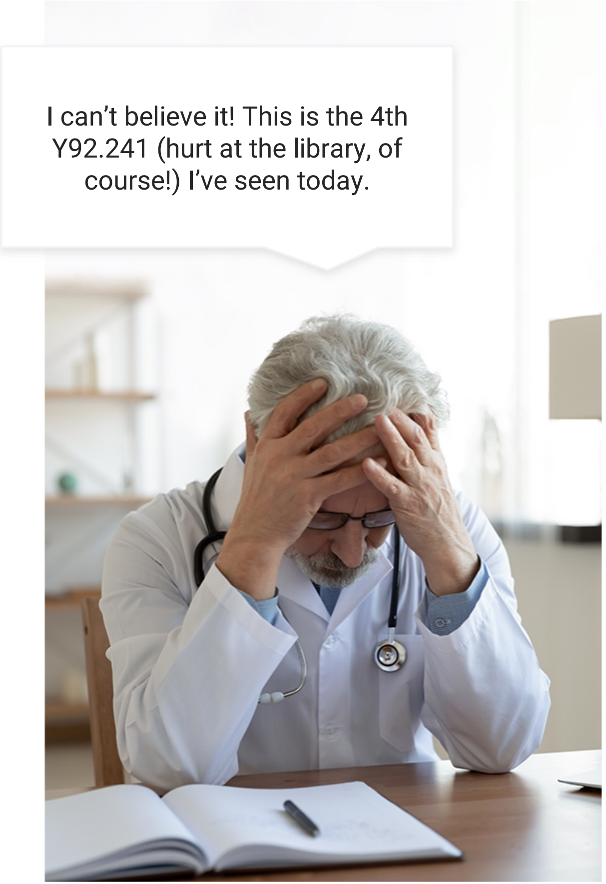
It is large and complex with an enormous number of claims processed each year

- ~ 12 billion claims per year are processed by our healthcare system
- ~ 30% or 3.6 billion claims per year are related to physician-patient encounters
- Manually coding every claim is impractical; studies show 5% of billable services are missed altogether

Data from Every Single MGH Physician Survey

Do physicians enjoy, like, look forward to, or want to be involved in charge capture in any way, shape, or form?





Governing Principles

At least two logical approaches to tackling our problem:

1 **Build a system and**

- *train doctors* to document for that system (input-side) and/or
- *encode rules* to determine codes and make automation decisions from documentation (output-side)

Generally takes form as a rules-based expert system

- Cumbersome for coders/docs
- Brittle
- Difficult to maintain
- Difficult to determine accurate automation decisions

2 **Build a system and**

- *train that system* to understand doctors' documentation and
- *train that system* to automatically determine codes and make automation decisions

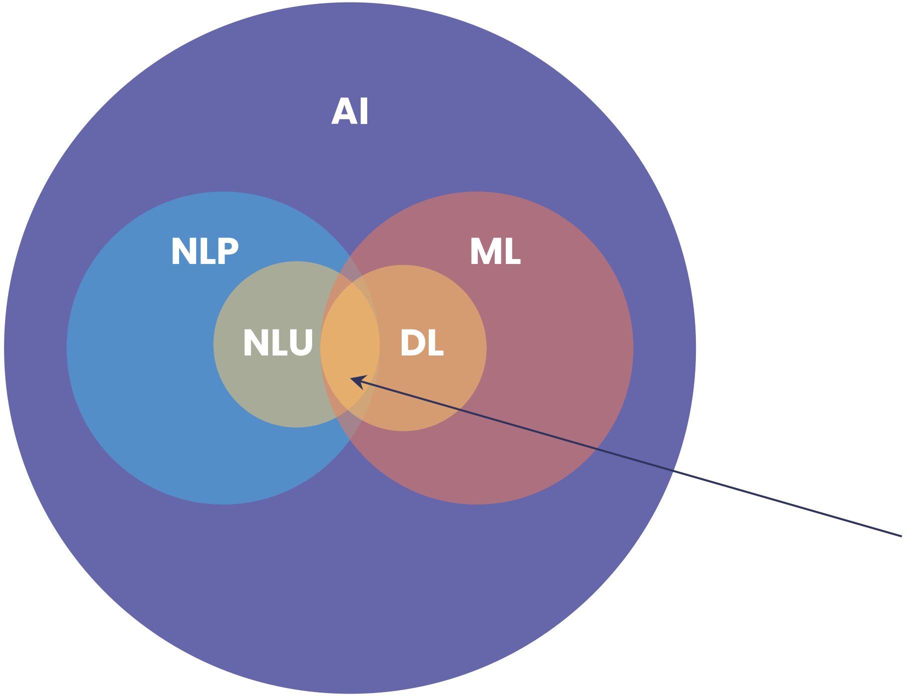
Generally takes form as an ML-based approach

- Much less cumbersome
- Much more robust
- Much more accurate

CMX Technology: Governing Principles

- We chose to go the modern ML-based route
- We had to devise a unique ML solution and invent many of its components
- But how does ML work and how can it be used to solve our task?
- And how is an ML-based solution different than an NLP or rules-based solution?

The AI Landscape



AI: Artificial Intelligence

NLP: Natural Language Processing

NLU: Natural Language Understanding

ML: Machine Learning

DL: Deep Learning

LLM: Large Language Models (e.g. ChatGPT)

NLP vs. NLU vs. ML

NLP **Natural Language Processing**	**NLU** **Natural Language Understanding**	**ML** **Machine Learning**
"Pleural effusion" is a statistically interesting phrase	"Pleural effusion" is a lung condition	If pleural effusion with malignancy, code J91.0 If pleural effusion in other conditions, code J91.8

NLP vs. NLU vs. ML

NLP & NLU are largely about *extracting* and *assigning meaning* to words and phrases in the note.

- “pleural effusion” is a ”lung condition”

But you need rules to act on that information…
…what to do when you see the lung condition pleural effusion?

- In a traditional NLP/NLU-based system, you *manually craft* the rules
- In an ML-based system, you *automatically learn* the rules from data

But how does ML do this?

Machine Learning Intro

Let’s think about how ML works

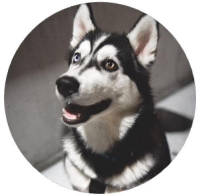
Consider a game where we try to come up with characteristics that distinguish various kinds of object.

ML Intro

Binary Classification

What characteristics distinguish birds from dogs?

Lots of single characteristics would seem to do the trick, for example, wings or fur.

Dogs have fur but no wings.

Birds have wings but no fur.

ML Intro

Multi-class Classification

But what if I throw in bats?

Hmm… Bats have fur like dogs and wings like birds. Need both characteristics.

Dogs have fur, but no wings.

Birds have wings, but no fur.

Bats have wings and fur.

ML Intro

Multi-class Classification

And now what if I throw in bears, snakes, and plants?






We'll need lots more distinguishing characteristics, but we can find them

Wings, fur, teeth, beaks, four legs, green, leaves, warm-blooded, etc.

What have we done? We've come up with a list of *features* that we can use for *classification*.

ML Intro

Model Generation

An ML algorithm takes *labeled training data* as input and determines a *function of the features* that produces a *score* which is *correlated with the label*. This function is a *predictor*.

birdScore = 5.36 x [beak?] + 3.1 x [wings?] – 4.8 x [leaves?] – 1.2 x [green?] …

High scores: likely to be birds **Low scores:** unlikely to be birds

Coding Example

J91.0 score = 6.7 x [pleural effusion?] + 2.3 x [malignancy?] …

J91.8 score = 7.5 x [pleural effusion?] – 1.3 x [malignancy?] …

These predictors essentially encode *rules*, and these rules have been *automatically learned from data*.

But how do we make the output of these predictors *actionable*?

ML Intro

Calibration

Calibration takes these predictor scores and maps them to calibrated confidences, e.g.,

- birdScores in range [10, 12] are 90% often birds
- birdScores in range [2, 4] are only 60% often birds
- birdScores in range [-8, -6] are merely 10% likely to be birds.

ML Intro

Automation

Calibrated confidences can be used for *automation*.

- birdScores in range [10, 12] are 90% often birds
- birdScores in range [2, 4] are only 60% often birds
- birdScores in range [-8, -6] are merely 10% likely to be birds.

If I want to automatically tag images as birds, but with at most 10% errors, then I can safely do so as long as my birdScore is in the range [10, 12] (or presumably higher).

We build *machine learning* predictors and use calibrated confidences to make *automation decisions*.

Thresholding these calibrated confidences allow us to intelligently make *automation vs. quality tradeoffs* to satisfy customer needs.

ML Intro

Multi-label Classification

Our true problem is actually much, much more complex; it's closer to picking out the exact correct set of animals from a picture, given thousands of possible animals, at scale, with confidence, and in the presence of noisily labeled images…

What About LLMs (e.g. ChatGPT)?

Language Models are designed to understand human language.

- A practical example we see every day is *auto-completion*: “Thank you very ___”

Large Language Models (LLMs) are trained on *enormous* amounts of data, and they understand human language, computer programming, art & images, and a host of other things very well.

- LLMs, out-of-the-box, are not particularly good at *medical coding* (“hallucinations”).

However, LLMs are extremely good at *information extraction*, and with further specific *pre-training* and *fine-tuning*, they will likely perform quite well in (parts of) medical coding.

LLM Example: Findings Section of a Colonoscopy

The perianal and digital rectal examinations were normal. A moderate amount of stool was found in the entire colon, interfering with visualization. Lavage of the area was performed using 1 liter of sterile water, resulting in clearance with good visualization. Many small and large-mouthed diverticula were found in the sigmoid colon.

A 1 mm polyp was found in the cecum. The polyp was sessile. The polyp was removed with a jumbo cold forceps. Resection and retrieval were complete. Verification of patient identification for the specimen was done.

A 8 mm polyp was found in the ascending colon. The polyp was sessile. The polyp was removed with a saline injection-lift technique using a hot snare. Resection and retrieval were complete. Verification of patient identification for the specimen was done. Estimated blood loss was minimal.

A 4 mm polyp was found in the descending colon. The polyp was sessile. The polyp was removed with a cold snare. Resection and retrieval were complete. Verification of patient identification for the specimen was done. Internal hemorrhoids were found during retroflexion.

LLM Example: Prompt Engineering for ChatGPT

In the following medical report, find all mentions of polyp.

For each mention, return size of the polyp, body location of the polyp, the technique used to remove the polyp, and the original text span from which the answer is extracted, and the character ranges for the text span.

The answer should be in a json format. The keys in the json output should be "size", "body_location", "removal_technique", "text_span" and "character_span”.

LLM Example: ChatGPT Result


```
{
  "mentions": [
    {
      "size": "1 mm",
      "body_location": "cecum",
      "removal_technique": "jumbo cold forceps",
      "text_span": "A 1 mm polyp was found in the cecum. The polyp was sessile. The polyp was removed with a jumbo cold forceps.",
      "character_span": {
        "start": 817,
        "end": 920
      }
    },
    {
      "size": "8 mm",
      "body_location": "ascending colon",
      "removal_technique": "saline injection-lift and hot snare",
      "text_span": "A 8 mm polyp was found in the ascending colon. The polyp was sessile. The polyp was removed with a saline injection-lift technique using a hot snare. Resection and retrieval were complete. Verification of patient identification for the specimen was done. Estimated blood loss was minimal.",
      "character_span": {
        "start": 946,
        "end": 1138
      }
    },
    {
      "size": "4 mm",
      "body_location": "descending colon",
      "removal_technique": "cold snare",
      "text_span": "A 4 mm polyp was found in the descending colon. The polyp was sessile. The polyp was removed with a cold snare. Resection and retrieval were complete. Verification of patient identification for the specimen was done.",
      "character_span": {
        "start": 1164,
        "end": 1300
      }
    }
  ]
}
```

An ML-based Solution

We solve the massive multi-label prediction problem for autonomous code prediction using machine learning.

- We use NLP/NLU/LLM to *extract information* from the notes
- We make that information *actionable* by *rules automatically learned via ML* from data
- Our ML is *glass-box* in the sense that every prediction can be traced back to the information in the note that caused that prediction to be made
- Every prediction has an associated *interpretable calibrated confidence* that enables user-defined automation at quality targets
    - *automation vs. quality trade-off*

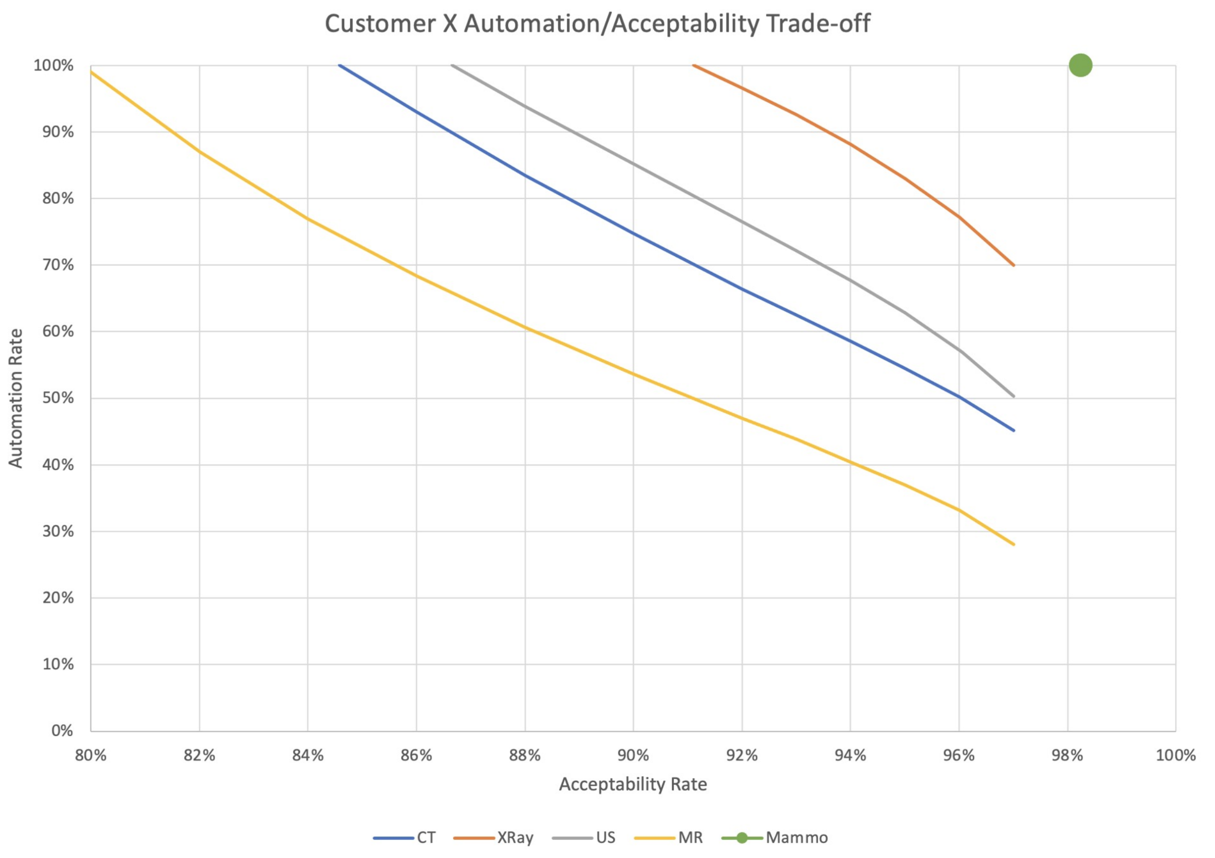
ML Intro

Calibration

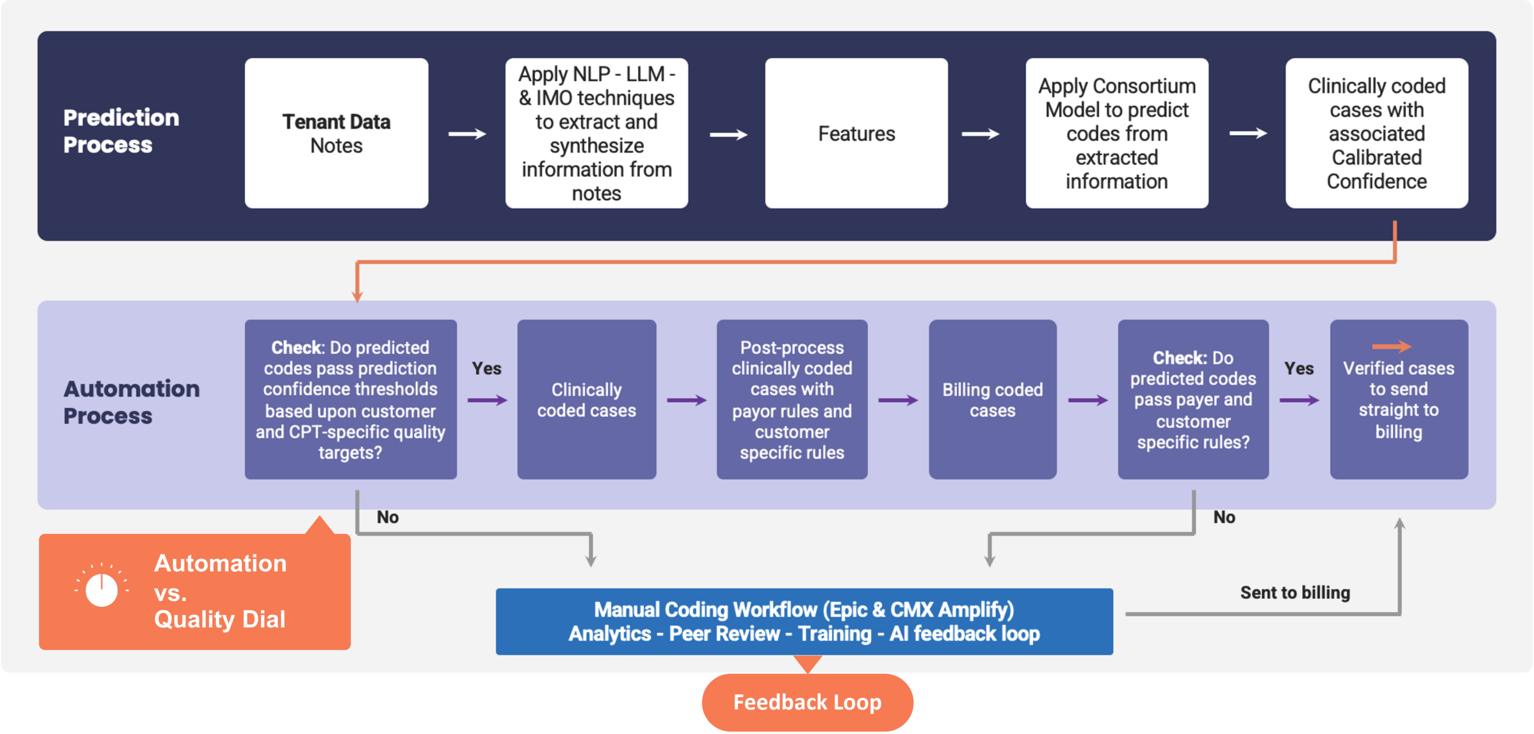
Calibration takes these predictor scores and maps them to calibrated confidences, e.g.,

- birdScores in range [10, 12] are 90% often birds
- birdScores in range [2, 4] are only 60% often birds
- birdScores in range [-8, -6] are merely 10% likely to be birds.

Automation at Quality



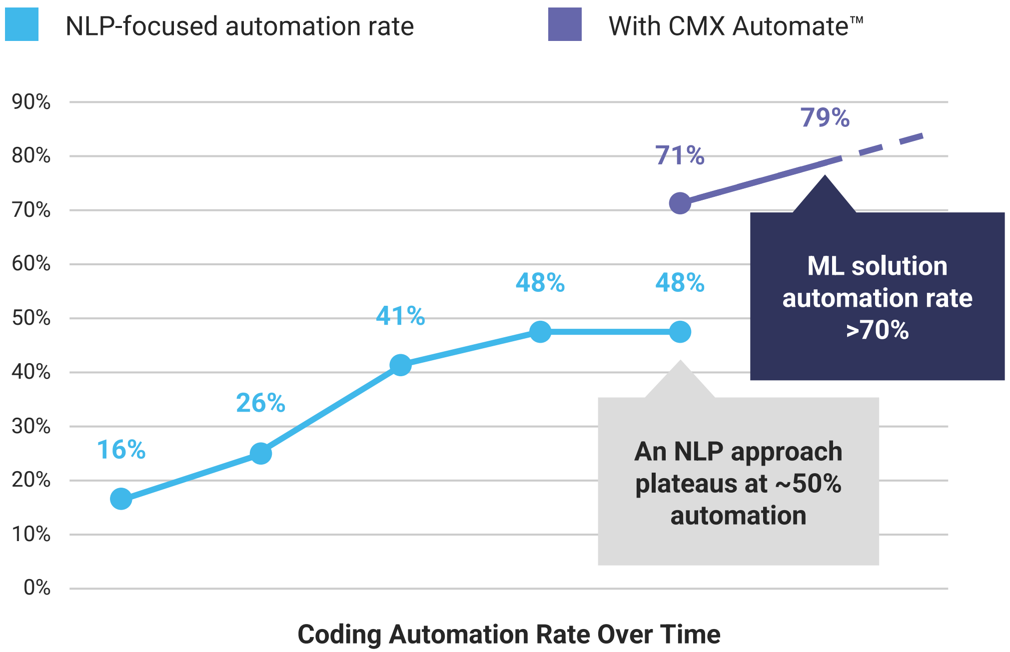
A Complete ML-based Automation Solution



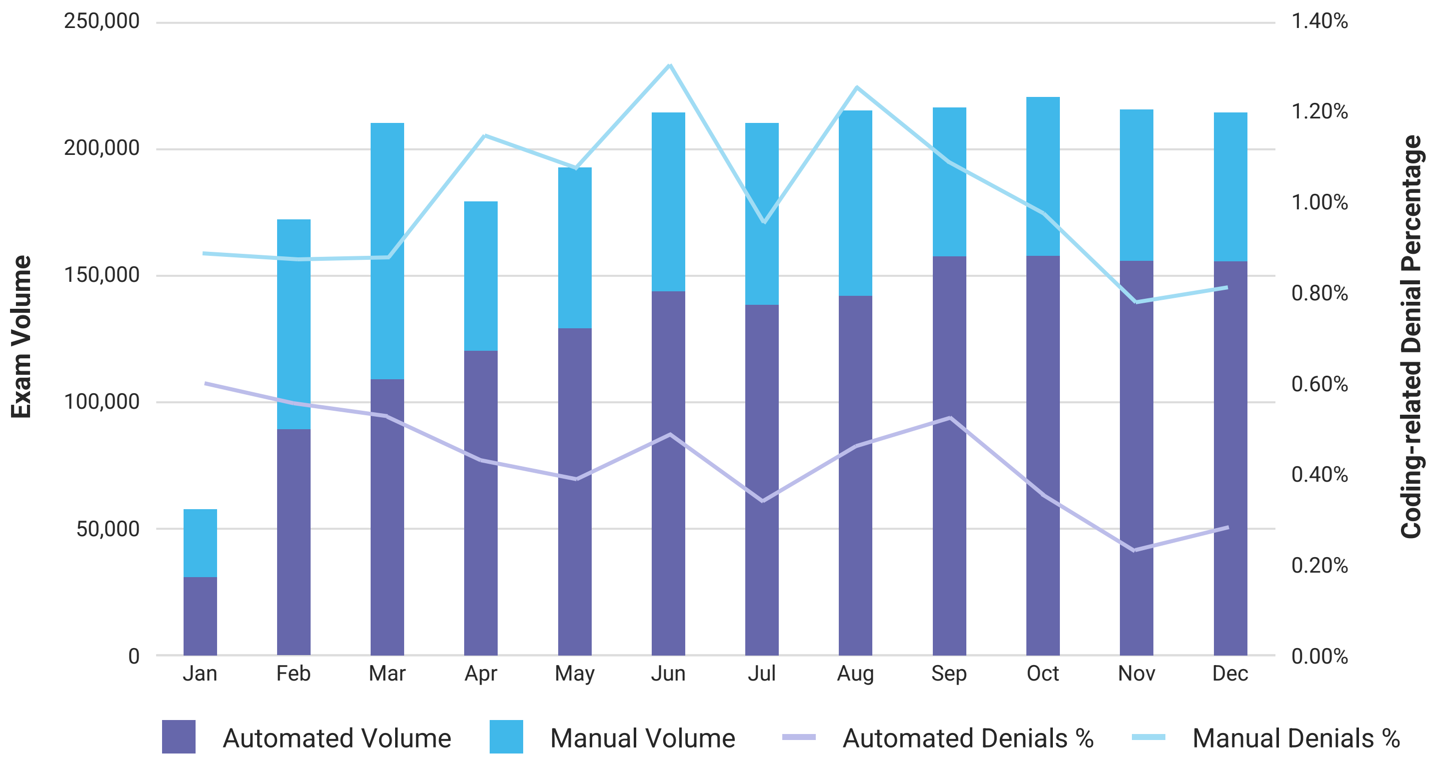
Recent Results of Automation

The CMX Autonomous-Coding platform outperforms the NLP-centric solutions by combining the power of AI: NLP, NLU, DL, ML.

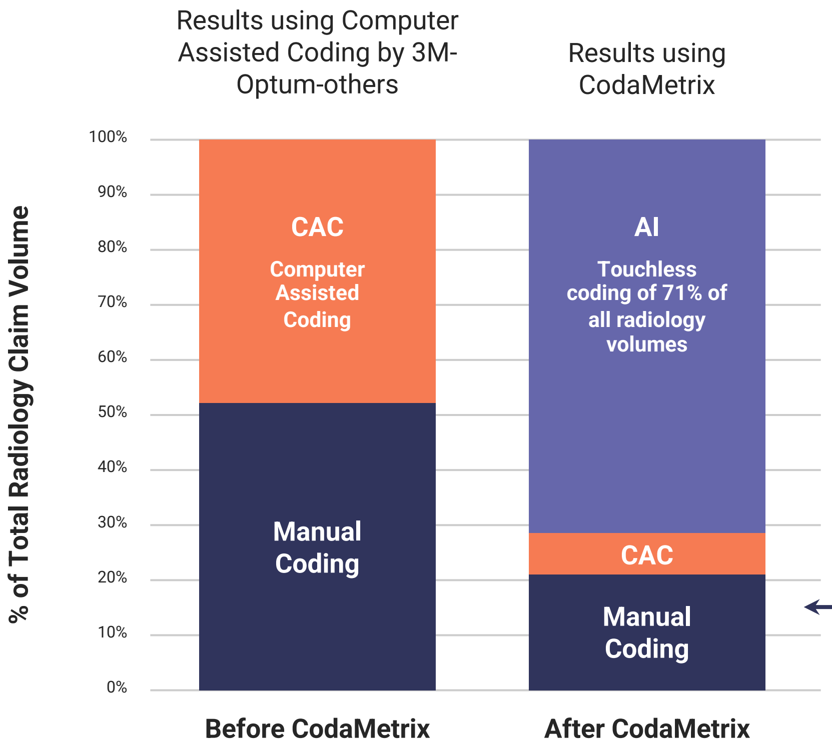
CU Medicine Implementation of CodaMetrix



Mass General Brigham's CY2021 Total Accessions (~2.1M)



CU Medicine CMX Results



5x improvement
in lag time

64% reduction
in manual coding

Questions

CODAMETRIX

Thank you!

www.codametrix.com (978) 906-1183 jill@codametrix.com