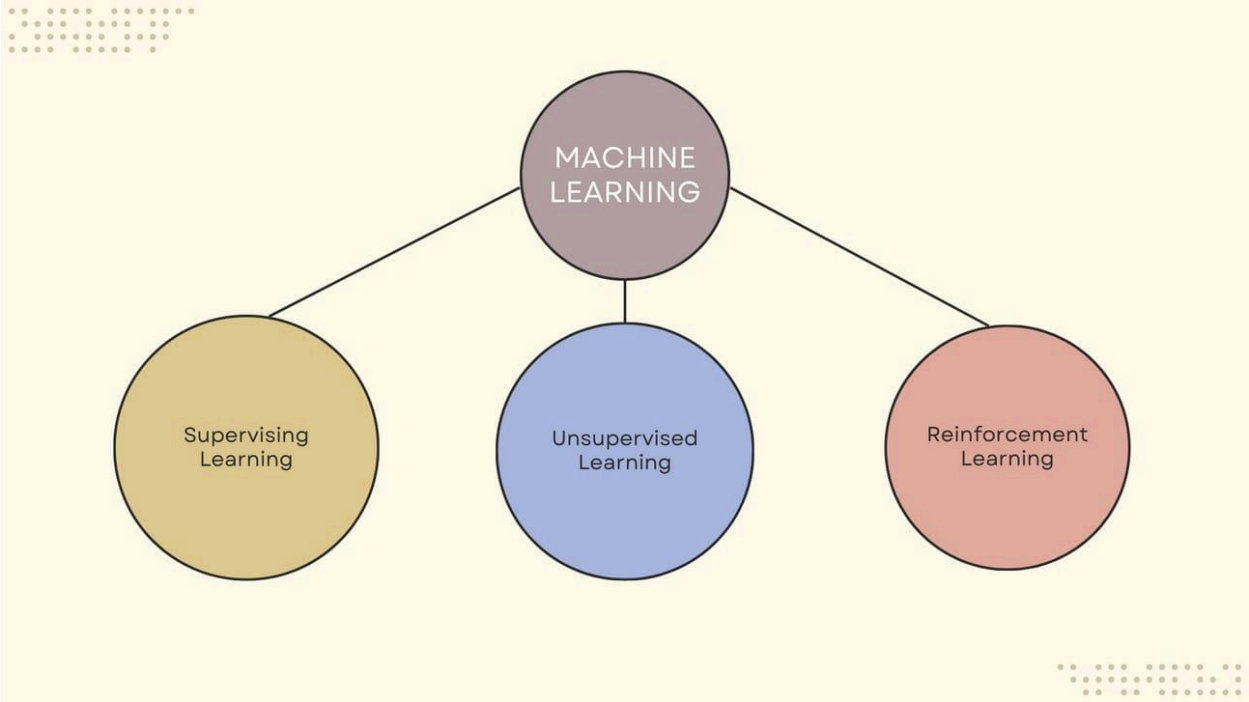
Machine Learning Terminology

Before we dive headfirst, it helps to know some of the lingo beforehand. These are all terms you will see often in ML, so feel free to reference this table often, as you read through the rest of this. Don't worry if this all doesn't sink in right away—we'll go more in depth in the future.

- Algorithm - a set of rules or steps the computer follows to learn from the data.
- Model - the final product after an algorithm learns from the data.
- Training Data - the examples you give to your model to learn from. (Like flash cards!)
- Testing Data - the new data you give to your model to see how well it performs. (Like a pop quiz!)
- Features - the input variables that describe your data. (If your data is a dating profile, the features would things like height, hobbies, etc.)
- Labels - the answers your machine is trying to learn. (Ex. Spam email or not? Sorry, grandma's forwards)
- Epoch - one full pass through the training data. (Like one full lap in a marathon).

Types Of Machine Learning

So we know that machine learning is a field of computer science that teaches computers to learn on it's own from data.
But that feels like a bit of a *broad* description, doesn't it? That's because it is.
In order to understand machine learning, we have to know the 3 main flavors it comes in: Supervised Learning, Unsupervised Learning, and Reinforcement Learning.



🎓 Supervised Learning (“The Honor Student”)

Let’s start with supervised learning.

Supervised Learning is when we train a machine learning model using labeled data to predict outputs from inputs.

Let’s use an example…

This is like giving a student a cheat sheet with all the correct answers for an exam. Let’s get bougie and say that the exam is all about being able to tell the difference between a Rolex and an Omega watch. We show the student thousands of pictures with Rolex watches labelled “Rolex” and Omega watched labelled “Omega”.

Eventually the student starts to recognize subtle differences between the two watches, because they are *learning by examples*. The more they see, the better they get!

Just think of the student as your machine learning model. Eventually, after having been given enough examples, it will get good enough that it won’t even need them. We’re telling the student:

“Here’s the answer. Now learn how to get there.”

In fact, I’m training you through a supervised learning process right now, by telling you about supervised learning. (Full circle moment).

🕊️ Unsupervised Learning (“The Free Spirit”)

Let’s quickly test your deductive skills…if supervised learning means we give labelled information to our model, then what could *unsupervised* learning mean?
Of course! Our data is now unlabelled.
Unsupervised Learning finds patterns or structures in data without labeled outcomes, often through clustering or grouping.
Here, the model gets a bunch of data that has no labels, and has to figure out patterns all by itself. Over time, the model starts to recognize certain features in the data and groups them together with other data samples that have the same or similar features.
Think about a typical high school environment. Ever notice how certain groups/cliques of people just seem to naturally form?
You see jocks gravitate together in a sea of letterman jackets, nerds sitting together collectively, nose-deep in their books, and maybe all the band kids hauling their massive instrument cases and sheet music everywhere. How did they all congregate? It came down to pattern recognition.
Yes, I know I’m being painfully stereotypical here, but bear with me.

🎮 Reinforcement Learning (“The Gamer Kid”)

Ah reinforcement learning. My personal favorite.
I nicknamed it the “Gamer Kid”, because it’s similar to how people get good at video games; reinforcement learning is all about learning through trial, error, and reward. The model presses buttons, learns moves, see what works and what doesn’t, and improves from there.
Reinforcement Learning trains an machine learning agent to make decisions by interacting with its environment through trial and error, using rewards or punishments to learn the best actions in an environment.
Reinforcement Learning is great for something like robotics (a topic that I’d love to get into in the future). It helps the robot learn how to walk, pick things up, perform tasks, all through trial and error — not unlike learning how to ride a bike by wobbling, failing and eventually balancing.

Recap:

Here’s a table to summarize what we just covered:

Supervised Learning	Unsupervised Learning	Reinforcement Learning
Learns from examples with answers.	Find patterns without answers.	Trial and error learning with rewards.
“Here’s the answer, now learn to get there”.	*“No answers, just vibes.”*	*“Learn by messing up a lot.”*

Machine Learning Workflow Loop

Remember the Scientific Method from school?

You’d start with a question, form a hypothesis, run experiments, analyze data, and adjust based on what you found.

Well, machine learning follows a similar rhythm, except it’s just dressed up in code. Instead of blowing up the lab like a mad scientist, you’re feeding data into algorithms to find patterns, make predictions, and improve over time.

Following this workflow is essential - without it, it’s like trying to bake a cake by throwing random ingredients into the oven and hoping for the best. It keeps everything organized and repeatable. It works like this:

1. Define The Problem

Identify what exactly you’re trying to solve. Are you predicting house prices? Trying to classify which emails spam and not spam? You’re basically Sherlock Holmes here, pinpointing what goal is.

2. Collect The Data

This is where we get all the ingredients we need. This might mean pulling from datasets, scraping the web, or using pre-existing datasets—all depends on what your problem is from step 1. Both quantity and quality matter here!

3. Clean & Prepare The Data

Raw data is messy! It often comes with missing values, typos, duplicates, etc. This means your data needs “cleaning”. It needs to be formatted, organized, and adjusted so it can be understood by your ML model.

4. Choose A Model

Here's the exciting part: picking an algorithm. Depending on whether it's supervised, unsupervised, or reinforcement learning, your algorithm will differ, but nonetheless you will need one: its the brains behind your operation.

5. Train The Model

Our model can't just receive the data, it needs to learn from it. This is where we use the training data that we collected earlier. Using this data, our model will adjust itself based on patterns, and try to get better with each pass (epoch). Think of our model as Rocky Balboa preparing hard for the big final fight.

6. Evaluate The Model

This is where we make sure is doing what we want it to do. We feed it testing data this time, which is data that our model has never seen before, and evaluate how it performs. For instance, if our training data was training our model to identify a kangaroo, we should hope to expect for the model to correctly identify a kangaroo upon seeing a brand new picture of one that it's never seen before.

7. Tune & Optimize

Sometimes a chef will do a taste test of their dish to see if it's too salty, sweet, etc. In this step, you're the chef doing your taste test. We tweak settings (called hyperparameters), engineer new features, or even try new models to improve our results. In ML, sometimes the small changes can make a big difference!

8. Deploy The Model

Once you finish the previous steps (and often times more than once!), your model is ready to be introduced to the real world. This is like serving your top quality meal on a fancy platter.

9. Monitor & Maintain

Just like how food left alone can degrade, so can models. You'll want to monitor your performance, retrain the model with new data, and make sure your model is behaving as desired, and even better over time.

That's it! One important thing however…

Notice how the picture above shows a loop? That's because the steps I listed in the workflow are often revisited and repeated. The fact is, the very first model rarely works

perfectly. It needs constant adjusting, revisioning, and fine-tuning in order to really make the most out of it.

Common Challenges

Although sometimes machine learning might seem like magic, that's not to say it doesn't come with it's fair share of headaches.

One of them was mentioned above, and that's messy data. Think of missing values, duplicates, or just straight up gibberish that has no value for the model. If your data is garbage, then your results will be too (*Garbage In, Garbage Out rule*).

But there's a couple other big challenges that have to be mentioned: Overfitting and Underfitting.

Overfitting

Ever wore an outfit that was *wayyy* too tight? Maybe it was a shirt squeezing you so tight that it felt like your circulation was getting cut off, or a pair of jeans hugging so close that they revealed more than you intended.

That's kind of what overfitting is like in machine learning.

Overfitting is when a model learns the training data *too perfectly* (yes there is such a thing). It picks up on every tiny detail, even the useless ones, because it's overly focused on the training data. This is an issue because it underperforms when it's given new data.

Here's Breaking Bad's Tuco expressing his opinion on overfitting:

Underfitting

Underfitting is the opposite problem; it’s like when you’re wearing clothes that are three sizes too big. Technically, yes your body is covered, but it doesn’t fit your form at all and looks incredibly sloppy.

Underfitting is when a model doesn’t learn enough from the training data. It’s like a toddler reading a complex book — it misses the main patterns and performs badly on both the training data and the new data because it’s too basic.

An adorable depiction of under-fitting.

Recap:

Here’s how overfitting, underfitting, and well-performing fit looks like in machine learning:

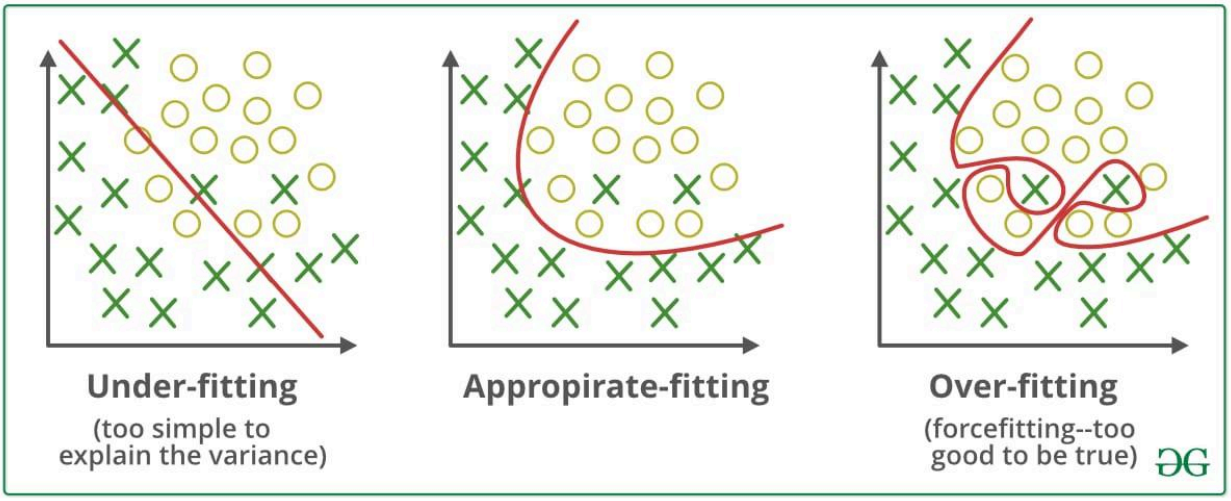


Image Credit to GeeksForGeeks

Conclusion

To wrap things, up remember this: machine learning might sound fancy, but at the end of the day, it's just a bunch of computers trying their best to learn—kind of like us cramming for a test.

Sometimes they do really well, sometimes they totally miss the point, and sometimes they just memorize everything without really *understanding* (looking at you, overfitting). The data can be messy, the results can be weird, but that's all part of the fun. Just keep feeding them good info, give them a few tries, and with enough practice (and maybe a little luck), they'll get the hang of it.

Organizational Readiness
60% of employers define themselves as transformative
86% of employers expect AI to revamp their business
Applicability of AI across all industries
Productivity growth uplift 1.5-2.5% YoY

Market Growth
$807 projected rise in global AI by 203020
U.S. NLP market to reach 38.70% CAGR by 2030
Computer Vision Engineering market to rise at CAGR of 16% by 2030

Workforce Demand
67% YoY increase in talent sourcing for AI professionals in 2025
77% of employers strategic goal to upskill their employees in AI-related competencies
87% surge in AI and Big Data skilled knowledge worker
AI and Big Data proficiency is fastest growing skill
Organizations competing intensely for top talent

Catalysts
Massive data growth driven by unstructured data
Computational capabilities
Broadening digital access
Real-time processing

Role and Skill Demand
170 million new jobs will be created by 2030 with AI and technology leading roles as leaders
12 million in the U.S.
Prompt Engineering LLM interaction 220% growth YoY
Machine Learning Engineering consistently ranked in top 3 fastest growing jobs
LLM Operations is one of the highest-paying AI skilled categories
FinTech engineers rank second fastest growing job category
AI Engineers specializing in industrial applications and edge computing
Autonomous and electric vehicle specialist among 15 fastest growing jobs
MLOPs and AI Design uptick 16% to 24% signaling investment in scalable infrastructure
NLP grew from 29% to 36%
Computer vision increased from 14% to 22%
Neural Network adoption rose 10% and now sits at 55%
Generative AI is leading in growth with more than double demand at 83%

Significant Growth in GenAI for Specific to Software Roles 2024 to 2025
Job descriptions for Software Engineers has more than doubled from 11% in 2024 to 23%
Engineering Managers demand increased from 7% to 18%

Product Manager mentions increased from 12% to 17%
Software Development Engineers in Test (SDET) is now meaningful at 8%
Cloud Solution Architects rose from 22% to 32%
GenAI job descriptions leaped from 7% to 17%

Industry-Specific Patterns
Technology sector leading AI adoption with 86%

Financial Services has strong use cases for fraud detection, algorithmic trading, and risk management

Manufacturing showing significant adoption of computer vision and robotics

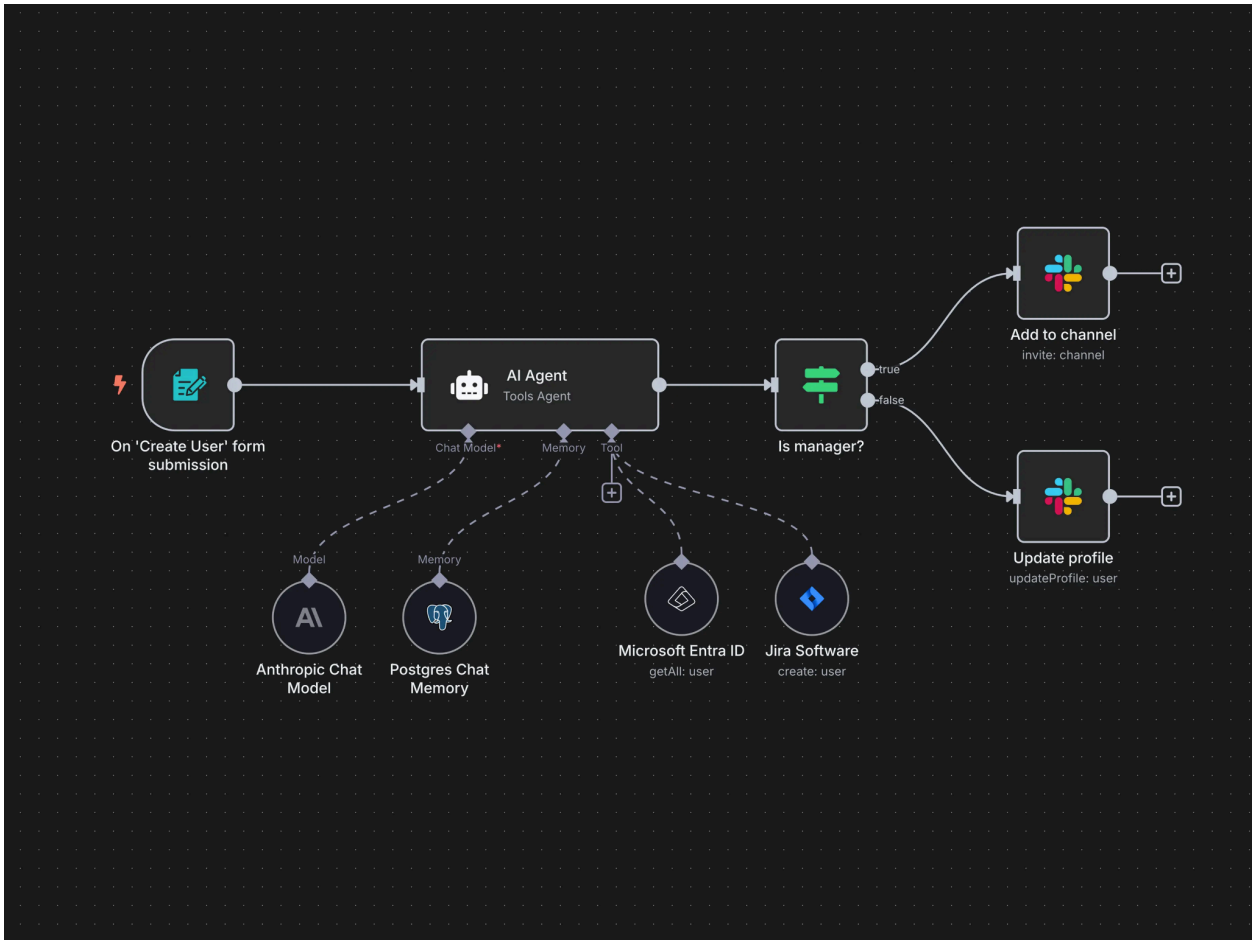
Increase need for AI specialists in medical imaging, drug discovery, and patient care optimization

Salary Impact
Professionals with generative AI skills can earn 47% higher salaries
Sales and marketing 43% higher salaries
Finance 42% higher salaries
Business operations 41% higher salaries

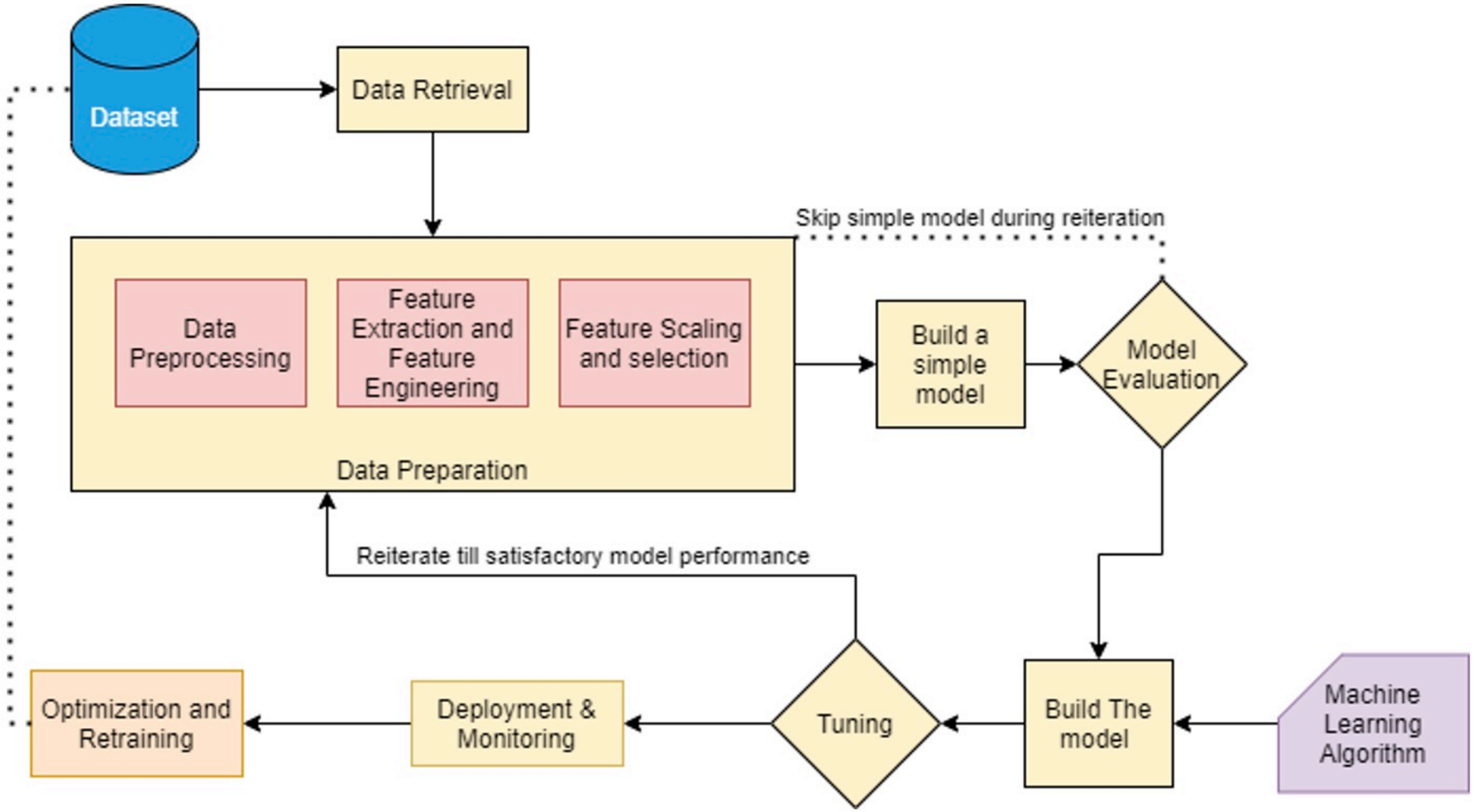
Projections
Generative AI integration expansion beyond text to multimodal applications
Growing demand for AI on edge devices
Greater visibility and accountability on responsible AI
New domain specific applications
Desired understanding of how AI creates business value

Job Displacement Projections by 2030
60-70% of white-collar roles
15-25% of knowledge workers

Source: World Economic Forum Future of Jobs Report 2025, industry surveys, labor market data, Interview Kickstart


On 'Create User' form submission
AI Agent
Tools Agent
Chat Model*
Memory
Tool
Is manager?
true
false
Add to channel
invite: channel
Update profile
updateProfile: user
Model
Anthropic Chat Model
Memory
Postgres Chat Memory
Microsoft Entra ID
getAll: user
Jira Software
create: user

The AI Stack


Dataset
Data Retrieval
Skip simple model during reiteration
Data Preprocessing
Feature Extraction and Feature Engineering
Feature Scaling and selection
Data Preparation
Build a simple model
Model Evaluation
Reiterate till satisfactory model performance
Optimization and Retraining
Deployment & Monitoring
Tuning
Build The model
Machine Learning Algorithm