

WHITEPAPER

AI skills report: Meeting the demand for AI & ML engineers



Author: Albert Sahakyan, PhD
Engineering Manager @ CodeSignal



Executive summary

With a surging demand for artificial intelligence (AI) innovation across industries, employers today are experiencing significant skill gaps in AI and machine learning (ML) engineering skills. This whitepaper identifies the core skills required of AI, ML and prompt engineers. It then presents two solutions for filling these skills gaps on technical teams: first, a more effective process for identifying qualified AI and ML engineering candidates and second, best practices for upskilling and reskilling current employees with in-demand AI and ML skills.

Table of contents

Introduction: The demand for AI engineers today	1
Essential skills for AI, ML & prompt engineers	3
Discovering talent with the right AI skills	8
Developing AI skills on your teams	10
Conclusion	15
References	16

Introduction:

The demand for AI engineers today

The AI future has arrived. From tech and finance, to healthcare, retail, and manufacturing, nearly every industry today has begun to incorporate artificial intelligence (AI) into their technology platforms and business operations. The result is a surging talent demand for engineers who can design, implement, leverage and manage AI systems.

Over the next decade, the need for AI talent will only continue to grow. The US Bureau of Labor

Statistics expects demand for AI engineers to increase by 23 percent by 2030 [1] and demand for machine learning (ML) engineers [2], a subfield of AI, to grow by up to 22 percent.

In the tech industry, this demand is in full swing. Job postings that call for skills in generative AI increased by an incredible 1,848 percent in 2023, a recent labor market analysis shows [3]. The analysis also found that there were over 385,000 postings for AI roles in 2023.

Generative AI: Number of monthly jobs posted in the US

3-month rolling average

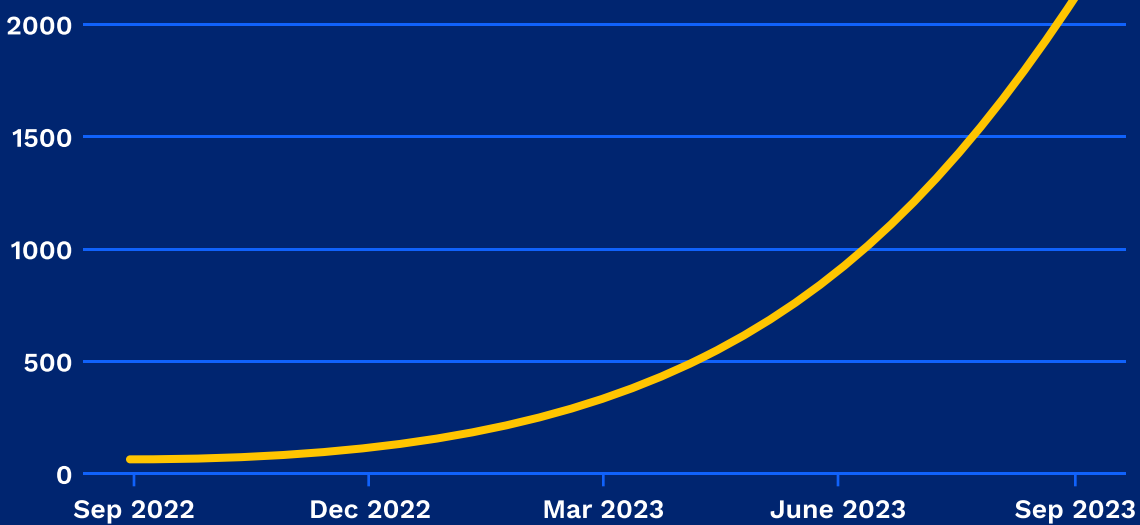


Figure 1: Growth of job postings requiring skills in generative AI, 2022-2023

There's also the rise of a field called prompt engineering: a role responsible for crafting the prompts for generative AI models, like GPT-4, to produce optimal outputs. A recent analysis of Glassdoor data found a growing demand for this role, with 2,100 open prompt engineer positions in December 2023 [4].

Organizations looking to boost their adoption of AI must take a two-pronged approach to fill the AI-related skills gaps on their technical teams:

1. First, they need better ways to discover and hire engineers who have the required AI, ML, and prompt engineering skills;

2. Second, they need to upskill and reskill their existing engineers to address the global shortage of AI engineering talent.

This whitepaper will demonstrate how to do both of these successfully to fill AI skills gaps on your technical teams. First, we will describe the core skills in AI, ML, and prompt engineering that teams should assess for when hiring and develop when upskilling. We will also give practical examples of best practices for questions that evaluate & build engineers' skills in AI and ML.

Essential skills for AI, ML, and prompt engineers

Now, let's dive deeper into the core skills required by AI, ML, and prompt engineering roles. These are the skills that employers will want to assess, when hiring for these roles, or to develop among their current technical teams.

Core skills for AI and ML engineers

AI and its subfields, machine learning (ML) and natural language processing (NLP), all involve training algorithms on large sets

of data to produce models that can perform complex tasks. As a result, different types of AI engineering roles require many of the same core skills.

CodeSignal's [Talent Science team](#) and technical subject matter experts have conducted extensive skills mapping of AI engineering roles to define the skills required of these roles. These are the core skills they identified for two popular AI roles: ML engineering and NLP engineering.



Machine learning (ML) engineering core skills

Skill	Description	Importance
Mathematics & statistics	Creating algorithms & understanding data patterns in AI/ML	Foundational
Coding & data algorithms	Essential for implementing efficient, scalable ML solutions	Foundational
Data querying & retrieval	Accessing & manipulating large datasets in AI applications	Foundational
Data cleaning & preprocessing	Improving data quality & model performance by removing or correcting anomalies	Essential
Feature engineering	Involves creating informative attributes to improve model predictions	Essential
Machine learning model development	Writing algorithms capable of learning from & making predictions on data	Essential
Model validation & selection	Techniques to ensure models are reliable & choosing the best one for deployment	Essential
Deep learning & neural networks	Utilizing multi-layered neural networks to model complex patterns in data for advanced AI applications	Specialized
Big data processing	Handling and analyzing vast datasets efficiently to derive insights & inform AI model training	Specialized
Model deployment processes	Implementing models into production environments to ensure they operate efficiently & reliably	Specialized
Data privacy, governance & AI ethics	Ensuring AI systems are designed & used in ways that protect privacy, comply with regulations & uphold ethical standards	Specialized

Table 1: Core skills in ML engineering, listed by importance

Natural language processing (NLP) engineering core skills

Skill	Description	Importance
Mathematics & statistics	Creating algorithms & understanding data patterns in AI/ML	Foundational
Programming skills	Essential for implementing efficient, scalable ML solutions	Foundational
Linguistics & semantics	Critical for natural language processing (NLP)	Foundational
Data collection & preparation	The initial step in creating a robust dataset for ML models	Essential
Feature engineering & text representation	Developing meaningful text features & representations that capture linguistic nuances for improved NLP model performance	Essential
Modeling for NLP	Building models to understand & generate human language	Essential
Model evaluation & optimization	Continuous improvement of models for enhanced accuracy & efficiency	Essential
Deep learning for NLP	Applying deep learning techniques to process & understand human language	Specialized
Large language models	Developing expansive models that can understand, generate & interpret human language at scale	Specialized
Multilingual NLP	Creating NLP systems that can understand & process multiple languages to enhance global applicability	Specialized
Data privacy, governance & AI ethics	Ensuring AI systems are designed & used in ways that protect privacy, comply with regulations & uphold ethical standards	Specialized

Table 2: Core skills in NLP engineering, listed by importance

Core skills for prompt engineers

Prompt engineers serve as intermediaries between AI or ML models and the humans who use them, and are in high demand with organizations that want to

leverage and integrate third-party AI models into their products.

They are responsible for developing inputs and training the models to deliver the best and most relevant outputs back to the user.



Here are the core skills CodeSignal has identified for prompt engineering roles:

Skill	Description	Importance
NLP & machine learning fundamentals	Creating algorithms & understanding data patterns in AI/ML	Foundational
Basic programming & scripting skills	Crucial for automating prompt generation & interaction with AI models	Foundational
Written communication	Key for crafting clear, concise prompts that guide AI responses effectively	Foundational
Prompt design & development	Involves creating targeted prompts to elicit specific responses or behaviors from AI models	Essential
Prompt testing & iteration	Refining prompts based on performance & feedback to improve outcomes	Essential
Task analysis & outcome definition	Involves understanding the end goal of an AI interaction & defining success criteria for prompts	Essential
LLM-specific prompt design	Deep understanding of specific large language models (LLMs) to optimize prompt design & interaction	Specialized
Advanced prompt engineering techniques	Employing sophisticated methods to refine prompts for more effective & nuanced AI responses	Specialized
Ethical AI design & user experience	Ensuring prompts are crafted to promote ethical AI use & create positive, inclusive user experiences	Specialized

Table 3: Core skills in prompt engineering, listed by importance

Discovering talent with the right AI skills

Hiring the right talent for AI and ML engineering roles all comes down to evaluating the skills that matter for the role. This requires asking candidates to solve the kinds of problems that they would encounter on the job—and do so in a realistic coding environment—to simulate real AI and ML engineering work.

Technical skills evaluations for AI and ML engineering roles should assess candidates' proficiency in the following core skill areas:

- **AI fundamentals:** First, evaluate candidates' understanding of theories behind common machine learning algorithms, models, and concepts. Assess their knowledge of, and ability to apply, topics like: L1 and L2 regularizations, reasons for overfitting, limitations of the Bayes rule, and the fundamentals of neural networks.
- **Data manipulation:** Next, assess candidates' skills in working with data in Python, R, or another relevant programming language. Key data skills to evaluate include performing basic operations with numbers, manipulating strings and arrays, and using nested loops.
- **Machine learning algorithms implementation:** Lastly, hiring teams will need to evaluate candidates' ability to implement a complete ML model, components of an ML model, or an ML concept based on a high-level overview of how the algorithm or concept works. Concepts may include: k-nearest neighbors, k-means clustering, decision trees, and Gaussian mixture models, among others.

Deep dive:

Smarter questions for evaluating ML skills

Below is an example of a strong question for evaluating ML engineers that has been developed by SMEs and validated by IO Psychologists. This question comes from CodeSignal's [Certified Evaluation](#) for assessing skills in ML engineering [5].

Example question: Training ML models

This assessment question evaluates a candidate's ability to train three different ML models—logistic regression, a decision tree, and a random forest—on actual data, and then evaluate the results.

Prompt: *Your challenge involves developing machine learning models to predict the likelihood of ad clicks. Effective models can significantly enhance targeted advertising strategies by utilizing available data on user demographics and ad attributes.*

The screenshot displays the CodeSignal IDE interface for a challenge titled "Question 3 of 4". The left sidebar contains navigation options: DESC, HISTORY, RULES, README, and SETTINGS. The main content area is split into two panes. The left pane shows the "Description" and "Acceptance Criteria". The right pane shows a code editor with Python code for training and evaluating models, and a terminal window displaying the output of unit tests.

Description: Your challenge involves developing machine learning models to predict the likelihood of ad clicks. Effective models can significantly enhance targeted advertising strategies by utilizing available data on user demographics and ad attributes.

Acceptance Criteria:

- Split the data into training and testing sets.
 - The size of the training data should be approximately 80% of the given dataset, and the testing data should be approximately 20%.
 - It is highly recommended to set the seed for random evaluations to 1 to ensure reproducibility.
 - Implement this in the `split_data(seed)` function.
- Train three different machine learning models: logistic regression, a decision tree, and a random forest.
 - Use default hyperparameters for the models. Do not tune them in this exercise.
 - Return the trained models as a dictionary with the model's name as the key and the trained model as the value. The names should be as follows:
 - "Logistic Regression"
 - "Decision Tree"
 - "Random Forest"
 - Implement this in the `train_models(X_train, y_train)` function.
- Evaluate the performance of each model on the testing set.
 - The evaluation should result in a `pandas.DataFrame` containing the accuracy scores of each model.
 - The DataFrame should have two columns: "Model" and "Accuracy Score", with each row corresponding to a model's performance.
 - The model's class name should be used for the "Model" column, and its accuracy score on the test set for the

Code Editor:

```
def train_models(X_train: pd.DataFrame, y_train: pd.Series) -> Dict[str, BaseEstimator]:
    models = {
        "Logistic Regression": LogisticRegression(max_iter=1000),
        "Decision Tree": DecisionTreeClassifier(max_depth=20, max_features=10, min_samples_leaf=10),
        "Random Forest": RandomForestClassifier()
    }
    for model in models.values():
        model.fit(X_train, y_train)
    return models

def evaluate_models(models: Dict[str, BaseEstimator], X_test: pd.DataFrame, y_test: pd.Series) -> pd.DataFrame:
    results = []
    for model_name, model in models.items():
        y_pred = model.predict(X_test)
        accuracy = accuracy_score(y_test, y_pred)
        results.append({"Model": model_name, "Accuracy Score": accuracy})
    return pd.DataFrame(results)

def save_data(df, file_name):
    df.to_csv(file_name, index=False)
```

Terminal Output:

```
UNIT TESTS TERMINAL + In Terminal RUN
2764 0
4767 0
3914 1
3499 0
2735 0
Names clicked, dtype: int64
Models:
dict_items([('Logistic Regression', LogisticRegression(max_iter=1000)), ('Decision Tree', DecisionTreeClassifier(max_depth=20, max_features=10, min_samples_leaf=10)), ('Random Forest', RandomForestClassifier())])
Model Performance:
  Model Accuracy Score
0 Logistic Regression    0.706
1 Decision Tree          0.670
2 Random Forest         0.703
/usercode/FILESYSTEM#
```

Figure 2: An example question that assesses candidates' skills in training ML models, as it appears in the CodeSignal Hire IDE

Developing AI skills on your teams

Discovering new talent with the right AI skills only solves one piece of the puzzle, however. A recent McKinsey report finds that, despite huge advancements in AI technologies over the past few years, many organizations have plateaued in their adoption of AI [6]. The reason: there aren't enough AI and ML engineers.

To address this talent shortage,

the report finds, organizations have turned to **reskilling and upskilling** as their key strategy for filling AI skill gaps at their organization.

So: what is the best way to develop the AI skills you need within your existing teams? To answer that, we first need to dive deeper into how humans learn new skills.

“ When it comes to sourcing AI talent, the most popular strategy among all respondents is **reskilling existing employees**. Nearly half of the companies we surveyed are doing so. ”

- **Alexander Sukharevsky**, Senior Partner @ McKinsey

Components of effective skills development

Most corporate learning programs today use the model of traditional classroom learning where one teacher, with one lesson, serves many learners. An employee starts by choosing a program, often with little guidance. Once they begin the course, lessons likely use videos to deliver instruction and are followed by quizzes to gauge their retention of the information.

There are several problems with this model:

- Decades of research show that the traditional, one-to-many model of learning is not the most effective way to learn. Educational psychologist Benjamin Bloom observed that students who learned through one-on-one tutoring outperformed their peers by two standard deviations; that is, they performed better than 98 percent of those who

learned in traditional classroom environments [7].

The superiority of one-on-one tutoring over classroom learning has been dubbed the 2-sigma problem in education (see Figure 3 on next page).

- Multiple-choice quizzes provide a poor signal of employees' skills—especially for specialized technical skills like AI and ML engineering. Quizzes also do not give learners the opportunity to apply what they've learned in a realistic context or in the flow of their work.
- Without guidance grounded in their current skills, strengths, and goals—as well as their team's needs—employees may choose courses or learning programs that are mismatched to their level of skill proficiency or goals.



2-Sigma Problem in Education

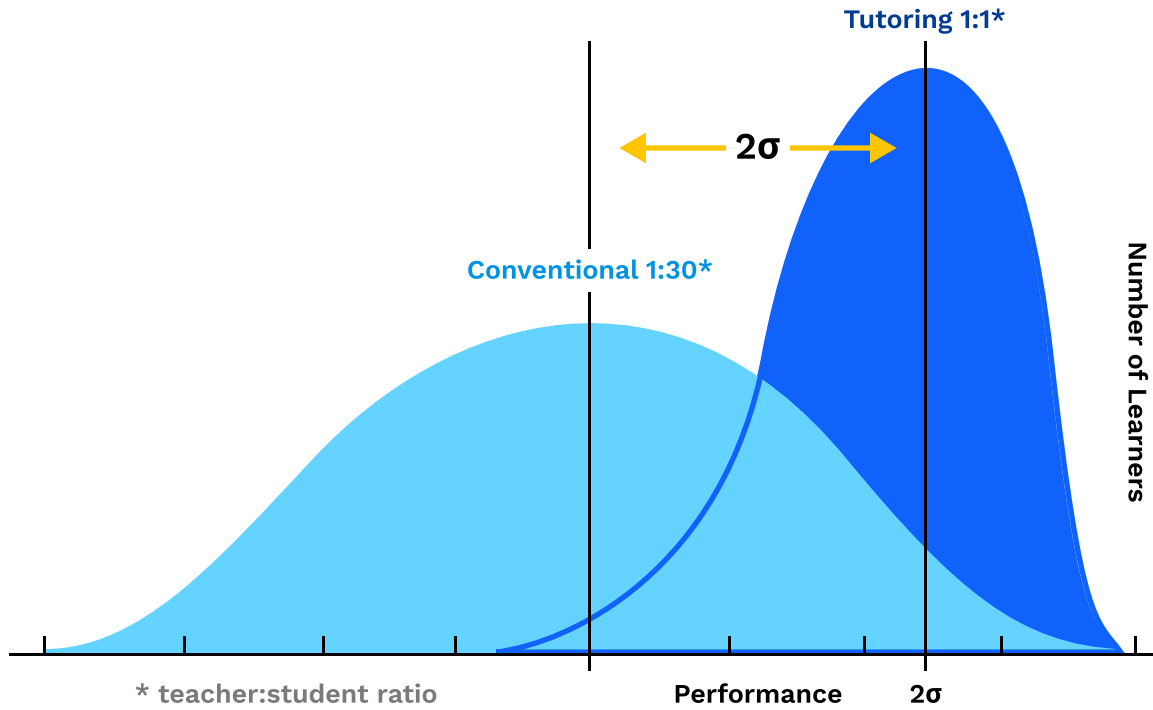


Figure 3: Comparison of the distributions of student performance by instructional style shows a 2 sigma difference in mean performance scores.

Developing your team members' mastery of the AI and ML skills your team needs requires a learning program that delivers the following:

- **One-on-one tutoring.** Today's best-in-class technical learning programs use AI-powered assistants that are contextually aware and fully integrated with the learning environment to deliver personalized, one-on-one guidance and feedback to learners at scale.

The use of AI to support their learning will come as no surprise to your developers and other technical employees: a recent survey shows that 81 percent of developers already use AI tools in their work—and of those, 76 percent use them to learn new knowledge and skills [8].

- **Practice-based learning.** Decades of research show that people learn best with active practice, not passive intake of information [9]. The learning program you use to level up your team's skills in AI and ML should be practice-centered and make use of coding exercises that simulate real AI and ML engineering work.
- **Outcome-driven tools.** Lastly, the best technical upskilling programs ensure employees actually build relevant skills (not just check a box) and apply what they learn on the job. Learning programs should also give managers visibility into their team members' skill growth and mastery. Your platform should include benchmarking data, to allow you to compare your team's skills to the larger population of technical talent, as well as integrations with your existing learning systems.



Deep dive:

Practice-based learning for AI skills

Below is an example of an advanced practice exercise from the [Introduction to Neural Networks with TensorFlow course](#) in [CodeSignal Develop](#).

Example practice: Implementing layers in a neural network

In this practice exercise, learners build their skills in designing neural network layers to improve the performance of the network. Learners implement their solution in a realistic IDE and built-in terminal in the right side of the screen, and interact with Cosmo, an AI-powered tutor and guide, in the panel on the left side of the screen.

Practice description: Now that you have trained a model with additional epochs, let's tweak the neural network's architecture. Your task is to implement a second dense layer in the neural network to potentially improve its learning capabilities.

The screenshot displays the CodeSignal Develop platform interface. On the left, a chat window shows a user asking "what's the math formula for the loss function?" and the AI tutor (Cosmo) responding with the categorical cross-entropy loss formula and a detailed explanation. The central code editor shows Python code for a neural network with two dense layers. The right terminal window shows the output of the training process, including loss and accuracy metrics for each epoch.

Chat Window:

Seobhan • Just now
what's the math formula for the loss function?

Cosmo • Just now
The loss function used in the given code is `categorical_crossentropy`. This is commonly used in classification problems where you're predicting multiple classes. The mathematical formula for categorical cross-entropy loss is given by:

$$L = - \sum_{c=1}^M y_{oc} \log(p_{oc})$$

Here:

- L is the loss.
- M is the number of classes.
- y_{oc} is a binary indicator of whether class c is the correct classification for observation o .
- p_{oc} is the predicted probability that observation o is of class c .

This formula essentially sums up the product of the actual distribution (y_{oc}) and the logarithm of the predicted probability (p_{oc}) across all classes. The negative sign ensures that the loss is a positive number. The goal of training is to minimize this loss, leading the predicted probabilities to closely match the actual distribution.

Ask me anything

Code Editor (solution.py):11
12 # Convert to one-hot encoding
13 y = to_categorical(y)
14
15 # Split the data into training and test sets
16 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
17
18 # Create model
19 model = Sequential()
20 model.add(Dense(64, input_dim=len(X[0]), activation='relu'))
21 model.add(Dense(len(y[0]), activation='softmax'))
22
23 # Compile model
24 model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
25
26 # Train the model with initial epochs
27 history = model.fit(X_train, y_train, epochs=5, batch_size=32, validation_split=0.2)
28
29 # Print the training history
30 print("Training History: ", history.history)

Terminal/Output:1/36 [.....] - ETA: 0s - loss: 1.3330 - accuracy: 0.4688
17/36 [=====>] - ETA: 0s - loss: 1.2103 - accuracy: 0.6213
32/36 [=====>] - ETA: 0s - loss: 1.0732 - accuracy: 0.6650
36/36 [=====>] - ETA: 0s - loss: 1.0741 - accuracy: 0.6710 - val_loss: 0.6994 - val_accuracy: 0.7778
Epoch 3/5
1/36 [.....] - ETA: 0s - loss: 0.4641 - accuracy: 0.9062
17/36 [=====>] - ETA: 0s - loss: 0.2876 - accuracy: 0.9217
32/36 [=====>] - ETA: 0s - loss: 0.5243 - accuracy: 0.8243
36/36 [=====>] - ETA: 0s - loss: 0.5289 - accuracy: 0.8259 - val_loss: 0.4211 - val_accuracy: 0.8403
Epoch 4/5
1/36 [.....] - ETA: 0s - loss: 0.2994 - accuracy: 0.8750
17/36 [=====>] - ETA: 0s - loss: 0.3815 - accuracy: 0.8906
32/36 [=====>] - ETA: 0s - loss: 0.3458 - accuracy: 0.8952
36/36 [=====>] - ETA: 0s - loss: 0.3424 - accuracy: 0.8964 - val_loss: 0.3158 - val_accuracy: 0.8889
Epoch 5/5
1/36 [.....] - ETA: 0s - loss: 0.1589 - accuracy: 0.9688
17/36 [=====>] - ETA: 0s - loss: 0.2443 - accuracy: 0.9210
32/36 [=====>] - ETA: 0s - loss: 0.2410 - accuracy: 0.9277
36/36 [=====>] - ETA: 0s - loss: 0.2531 - accuracy: 0.9243 - val_loss: 0.2616 - val_accuracy: 0.9028
Training History: {'loss': [4.059823203277588, 1.0741262435913086, 0.5289433002471924, 0.3425773978233374, 0.2510929744243622], 'accuracy': [0.2685291130467004, 0.671013324464443, 0.625335022881104, 0.6944316444940186, 0.824281547463138], 'val_loss': [1.579204133974268, ...]}

Figure 4: An example practice exercise that develops learners' skills in working with neural networks, as it appears in the CodeSignal Develop platform

Conclusion

The demand for AI and ML engineers is here, and will continue to grow over the coming years as AI technologies become critical to more and more organizations across all industries. Companies seeking to fill AI and ML skills gaps on their teams need to hone their ability to identify and hire qualified candidates for these specialized roles. Hiring, however, is not enough to fill the gap.

Companies must also invest in upskilling and reskilling their existing technical teams with crucial AI and ML skills.

[CodeSignal](#), the leading skills platform, empowers companies to discover and develop the skills that will shape the future. Founded in 2015, CodeSignal is trusted by top companies like Netflix, Capital One, Meta, and Dropbox.

References

- [1] Bureau of Labor Statistics, U.S. Department of Labor, "Computer and Information Research Scientists," Occupational Outlook Handbook, [Online]. Available: <https://www.bls.gov/ooh/computer-and-information-technology/computer-and-information-research-scientists.htm>.
- [2] Bureau of Labor Statistics, U.S. Department of Labor, "Computer and Information Technology Occupations," Occupational Outlook Handbook, [Online]. Available: <https://www.bls.gov/ooh/computer-and-information-technology>.
- [3] Lightcast Press Office, "Generative AI Demand Soars 1,800% for US Employers," Lightcast, Oct. 20, 2023. [Online]. Available: <https://lightcast.io/resources/blog/generative-ai-10-19-2023>.
- [4] A. Woodie, "And the Top Paying AI Job Is...," EnterpriseAI, 15-Dec-2023. [Online]. Available: <https://www.enterpriseai.news/2023/12/15/and-the-top-paying-ai-job-is/>.
- [5] CodeSignal. "Machine Learning Engineering Core Skills Evaluation Framework," CodeSignal Skills Evaluation Lab, 2021. [Online]. Available: <https://discover.codesignal.com/rs/659-AFH-023/images/Machine-Learning-Engineering-Core-Skills-Evaluation-Framework-CodeSignal-Skills-Evaluation-Lab.pdf>.
- [6] McKinsey & Company, "AI reinvents tech talent opportunities," New at McKinsey Blog, [Online]. Available: <https://www.mckinsey.com/about-us/new-at-mckinsey-blog/ai-reinvents-tech-talent-opportunities>.
- [7] B. S. Bloom, "The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring," Educational Researcher, vol. 13, no. 6, pp. 4-16, 1984. [Online]. Available: <https://web.mit.edu/5.95/readings/bloom-two-sigma.pdf>.
- [8] CodeSignal, "Developers and AI Coding Assistant Trends," CodeSignal Reports, [Online]. Available: <https://codesignal.com/report-developers-and-ai-coding-assistant-trends/>.
- [9] S. Freeman, S. L. Eddy, M. McDonough, et al., "Active learning increases student performance in science, engineering, and mathematics," edited by B. Alberts, approved April 15, 2014, received for review October 8, 2013, published in the Proceedings of the National Academy of Sciences of the United States of America, vol. 111, no. 23, pp. 8410-8415, May 12, 2014. [Online]. Available: <https://doi.org/10.1073/pnas.1319030111>.



Ready to close AI skills gaps?

Schedule a call to explore how you can hire talented AI and ML engineers and upskill your existing team members, all with CodeSignal.

[LET'S TALK](#)

