The Role of In-Context Examples in Regression with

Large Language Models

Learning vs Retrieval

Aliakbar Nafar[1], Kristen Brent Venable[2,3], Parisa Kordjamshidi[1]
[1]Michigan State University
[2]Florida Institute for Human and Machine Cognition
[3]University of West Florida

HUMANE LAB, 김건수

NAACL 2025 Outstanding

2025.06.24

Introduction: What Drives In-Context Learning?

- In-Context Learning: LLMs can perform tasks by observing examples directly in the prompt — no parameter updates needed.
- Two Dominant Views on How ICL Works
  - Meta-Learning: The model learns new patterns from in-context examples and generalizes.

  "Learning from data shown in the prompt."
  - Knowledge Retrieval: The model uses in-context examples as cues to retrieve relevant pre-trained knowledge.

  "Recalling what it already knows."
- Key question of this paper: "When do LLMs learn from in-context examples, and when do they simply recall?"

Experimental Setup

- Task Type: Regression
  - Predict Continuous numerical outputs from structured inputs
  - More complex than classification
- LLMs Used
  - GPT-3.5(OpenAI, 2020)
  - GPT-4(OPenAI, 2023)
  - LLaMA 3 70B(Meta, 2024)
- Why Regression?
  - Tests LLMs' ability to handle continuous outputs without classification shortcuts.

Datasets

1. Admission Chance
   - Predict graduate admission probability(India-based data)
   - Low prior exposure in LLM pretraining
2. Insurance Cost
   - Predict annual health insurance costs(USA)
   - Demographic + lifestyle features(e.g., smokre status)
3. Used Car Prices
   - Predict prices of used Toyota/Maserati cars in 2019
   - Real-world market features(mileage, fuel economy)

Prompt Configurations

1. Named Features
   - Shows real feature names(e.g., “Mileage”, “Smoker Status”)
   - Enables both learning from examples retrieving prioir knowledge
2. Anonymized Features
   - Replaces names with “Feature #1”, “Feature #2”, etc.
   - Removes domain-specific cues -> encourages pure learning

Prompt Configurations

3. Randomized Groud Truth

- Feature names intact, but labels replaced with random numbers
- Tests whether LLMs rely on outputs or just memorize format

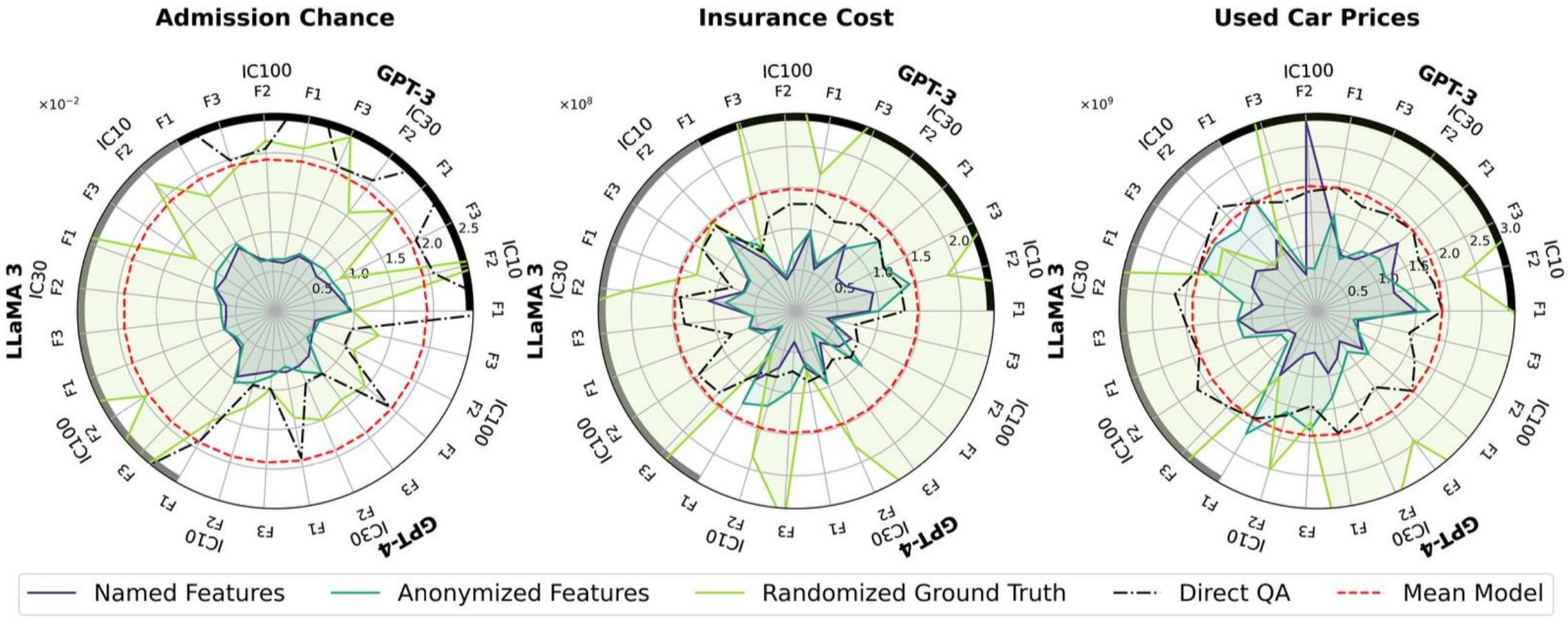
4. Direct QA(Zero-shot Baseline)

- No in-context examples
- Asks for a direct numerical prediction
- Given mean and standard deviation of dataseet for celibration

Result1: Knowledge Retrieval Baseline

- Direct QA(Zero-shot)
  - LLMs predict output without seeing any in-context examples.
- Key Obsebations
  - Better than random: LLMs outperform mean-only baselines in many cases
  - Dataset-dependent:
    - Best on familiar datasets(e.g., U.S. Insurance)
    - Worst on low-exposure data(e.g., Admission Chance – Indian students)
- Interpretation
  - LLMs already “know” quite a bit – even without examples.
  - But Performances varies with prior knowledge embedded during pretraining

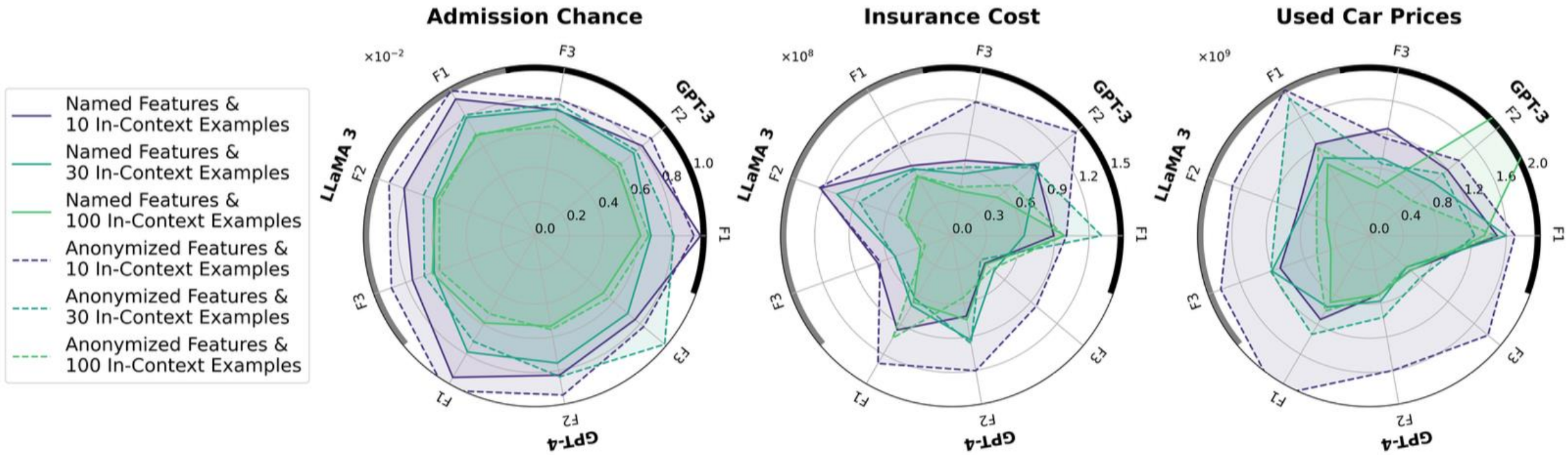
Result2: Learning VS Retrieval



Result2: Learning VS Retrieval

- Key Insight
  - Models do learn from outputs
  - Performace decreases with random outputs -> Output labels matter
- More Examples -> More Learning
  - Performance drop is larger with 100 randomized examples
  - Suggets LLMs increasingly rely on example outputs as m increases

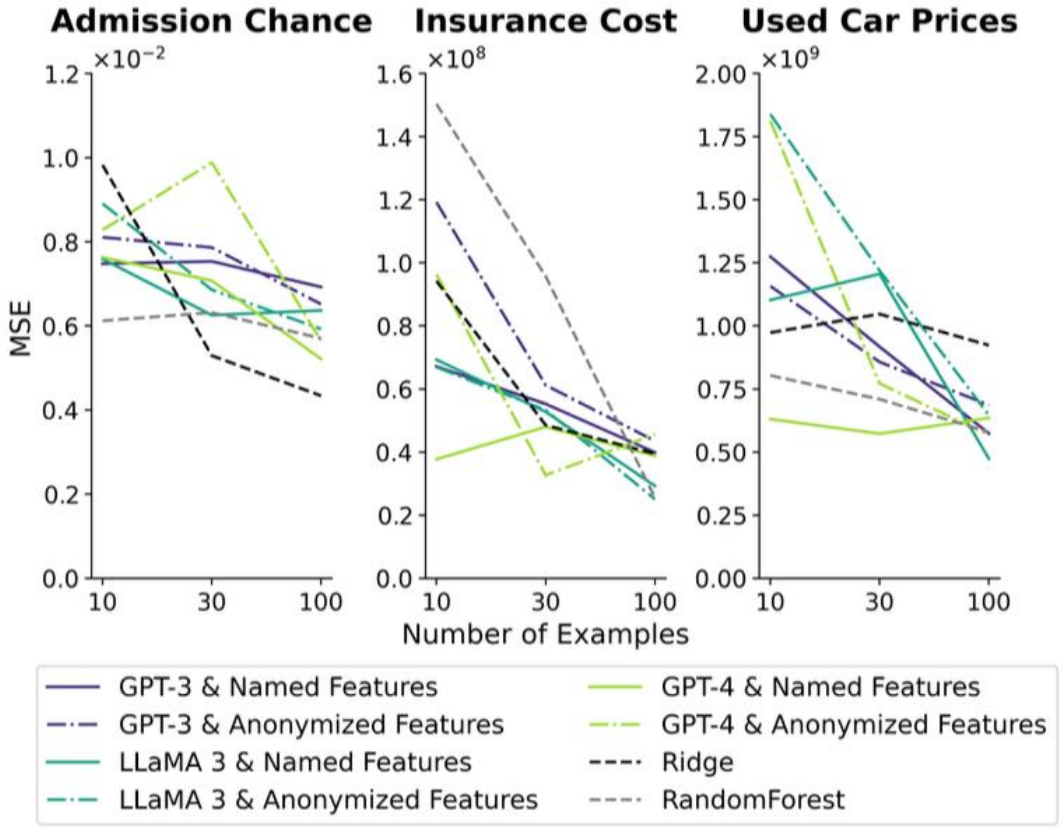
Results3: The Role of In-context Example Quantity

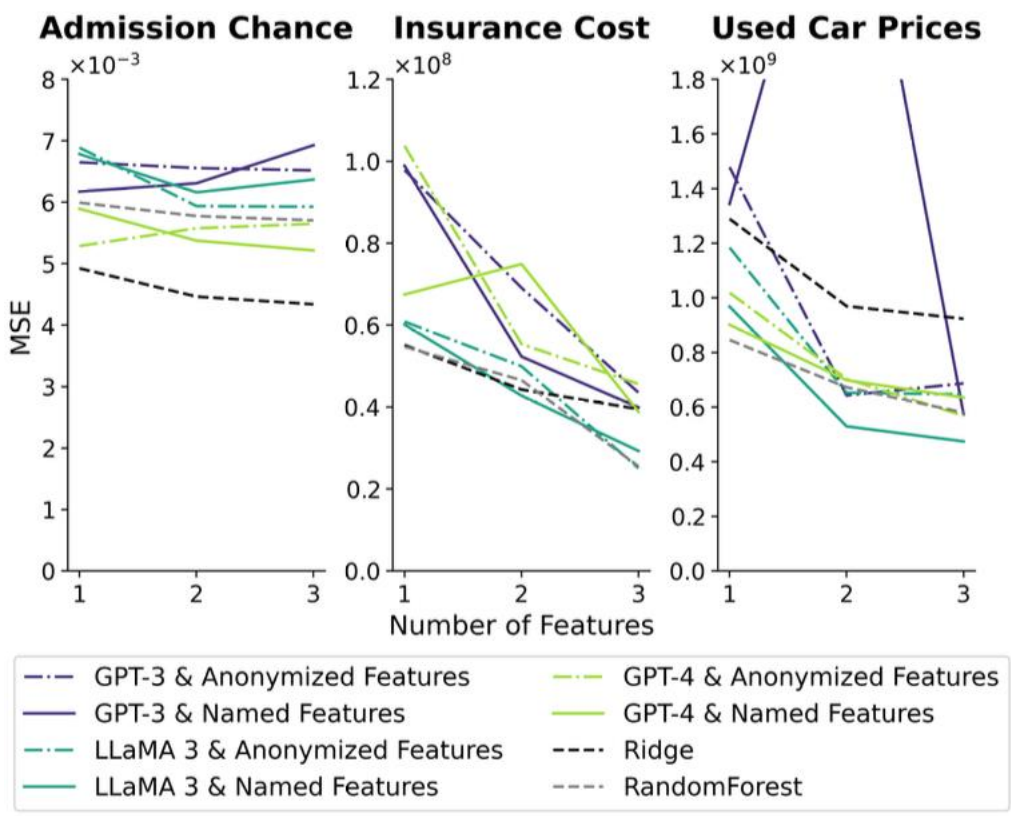


Results3: The Role of In-context Example Quantity

- Observation
  - More in-context examples -> More learning
  - But: Too many bad examples(e.g., randomized) degrade performance
- Conclusion
  - Example quantity strengthens learning, but quality still matters.
  - In informative prompts, fewer examples, can be more effective

Result4: The Role of Feature Quantity(F1/F2/F3)



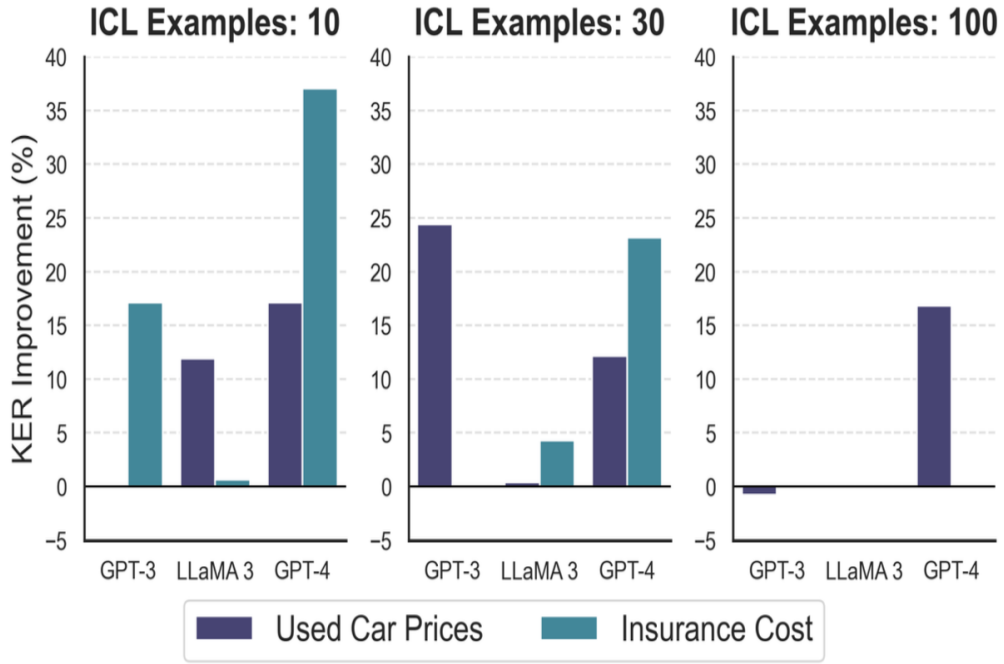


Result4: The Role of Feature Quantity(F1/F2/F3)

- Key Findings
  - Named: More features -> more knowledge retrieval
  - Anonymized: More features -> better learning from examples
  - Randomized: More features don’t help(no signal to learn)
- Conclusion
  - More features enhance both learning and retrieval
  - But only when outputs are meaningful

Quantifying Knowledge Use: Knowledge Effect Ratio(KER)

- What is KER?
  - KER measures how much adding feature names improves prediction accuracy.
  - $$\text{KER} = \frac{|Y_{AF} - Y_{GT}| - |Y_{NF} - Y_{GT}|}{|Y_{AF} - Y_{GT}|} \times 100$$
- What does it tell us?
  - KER ↑ = More benefit from knowledge retrieval
  - Higher KER with fewer in-context examples
  - KER ≈ 0 for Admission dataset -> little prior knowledge
- Conclusion
  - Named features significantly help LLMs – especially in low-data settings.



Discussion & Takeaways

- Key Takeaways
  1. ICL is not binary – It's spectrum
  2. Prompt design controls that balance
  3. LLMs are data-efficient
  4. Bad examples hurt performance
- Practical Implications
  - Use feature names wisely in prompt-based tasks
  - Avoid long, misleading examples
  - In low-data regimes, retrieval can replace extra examples