

Human-AI Co-Mentorship in Project-Based Learning: A Case Study in Financial Forecasting

Freyaa Chawla

Ahan Chawla, University of Notre Dame

Rishi Singh

Joe Germino

Mr. Grigorii Khvatskii, University of Notre Dame

Grigorii Khvatskii is a 4th year PhD student at the University of Notre Dame. His research interests include natural language processing and Large Language models. Specific to this project, he advised three early-undergraduate and high school students on data science fundamentals and usage of AI.

Human-AI Co-Mentorship in Project-Based Learning: A Case Study in Financial Forecasting

abstract

This paper reflects on a AI research project carried out by a team of high-school and early-undergraduate students under the mentorship of graduate researchers and ably assisted by AI tools. We share our experience in not only on the learning experience for the high school students, but also on how AI tools accelerated the process that enabled the high school students to focus on higher order problem formulation and solution. Although the participants entered the project with limited background in both AI and finance, they showed strong enthusiasm for technical market analysis and ETF price prediction. Traditional learning settings would first teach the necessary methods in a classroom setting and only later let students apply them. In contrast, our project emphasized workflow design: students identified the sequence of steps needed to address the problem and then used AI-driven tools to execute each step.

During the project the team tackled the challenge of forecasting ETF price movements from historical news data. They began by constructing a web scraper that collected the full text of financial articles together with key metadata (title, URL, and publication date). The scraped articles were subsequently processed with large language models `gpt-5-mini`, which assigned sentiment scores on a -10 to $+10$ scale and provided a brief justification for each rating. These sentiment time series were used to train a number of forecasting models, including both classical statistical approaches and deep-learning architectures such as LSTM networks.

We note that the high school students developed the necessary code through iterating with the AI tools, and we used our daily stand-ups to debug and answer conceptual questions. Each of the student was able to dig deeper into their area of interest whether computer science or finance, while collaboratively making a significant advance over the summer of 2025. This project was an important pedagogical exercise on how AI tools can be used for mentoring high school students, allowing them to focus on their specific interests and using the daily stand-ups to focus on problem definition and conceptual understanding. Despite their limited technical qualifications, the students were able to leverage AI tools to build meaningful models with real-world application.

introduction

Generative artificial intelligence (AI) tools based on large language models (LLMs) are playing an important role in shaping education and training. Yet they create an apparent tension centered on

how and when to use these AI tools within a classroom setting. The use of AI introduces challenges that were not previously encountered by either educators or students. In this sense, AI use can be described as a "wicked problem" [1] that is difficult to conceptualize or address. One key challenge for educators is finding an appropriate balance between the use of AI tools and more traditional approaches to teaching.

In this work, we address this challenge by exploring what happens when AI is not merely a supplementary tool but a *co-mentor*, that is, actively guiding students alongside human advisors throughout an extended research project. We reflect on a ten-week project conducted by a team of two high school students (with one entering college over the course of the project) and early undergraduate students, mentored by a graduate student and a postdoctoral scholar in computer science at the University of Notre Dame, a US-based educational institution. Throughout the project, students were encouraged to use AI tools not only to complete tasks but to learn the concepts and skills required to complete them.

The project's technical goal, building an AI system to forecast ETF prices using historical price data and news sentiment, provided a rigorous test bed for this pedagogical approach. Because the participating students had limited background in programming and finance, the project demanded substantial skill acquisition. In a traditional learning environment, we would have begun with classroom instruction to teach the requisite theories and techniques before students worked independently to apply this knowledge. Instead, we adopted a workflow-driven approach: under the scholars' supervision, students identified the steps required to complete the project and used AI tools to execute and learn from those steps in real time.

This approach structured our ten weeks as a series of daily stand-ups (around 30 minutes each) focused on problems to be solved and tasks to be accomplished, with students pursuing specific learning exercises, including doing so through dialogues with the AI tools, between sessions. Through this human-machine co-mentorship model, we observed both accelerated progress and instructive failures that shaped our understanding of AI's role in education.

This dual focus, on both the learning process and technical outcomes, motivates the following research questions (RQs).

- RQ1** How does AI-augmented mentorship influence the learning process compared to traditional methods of instruction?
- RQ2** What are the most frequent failure modes encountered when using AI in a project-based learning context?
- RQ3** Do LLM-generated sentiment scores improve the predictive performance of ETF price forecasting models?

Our observations suggest human-AI co-mentorship model accelerated both development and learning, allowing students to engage with higher-level concepts rather than getting mired in implementation details. At the same time, we observed recurring failure modes: LLMs often produced subtle errors in code or analysis that were difficult to detect without domain expertise. These failures, however, became valuable learning opportunities, and students gradually developed the judgment to identify and correct such errors independently.

The rest of the paper is organized as follows: **Related Work** reviews prior work on AI-assisted project-based learning and ETF forecasting techniques. After that, **Section Data, Methods and Analysis** details the data collection pipeline, sentiment extraction, model architecture (regression plus classification), and presents performance results in tables. Next, **Reflections on AI Use** reflects on how generative AI supported programming, writing, and analysis while highlighting both benefits and pitfalls. After that, **Discussion** discusses pedagogical implications of our findings and outlines future research directions. Finally, **Limitations and Future Work** addresses study limitations and suggests avenues for broader validation. **Please note** that unless specified on AI as a mentor or assistant, the use of the word mentor will imply human mentor, which in this case are the graduate and postdoctoral scholars.

related work

We review how our study relates to prior work regarding our two main components: (1) the use of AI in project-based education and (2) the methods used for ETF price modeling.

ai in project-based education

AI for project based learning was widely explored in the literature. Findings reveal that LLMs tend to underperform human experts in tasks that closely resemble grading when benchmarked [2]. At the same time, other studies demonstrate that LLMs can outperform humans in other tasks, such as when generating feedback, as evidenced by [3] and further validated in [4] for even stricter and more sensitive settings such as K-12 learning. These results suggest that while LLMs may lag in some domains, they hold promise for use in broader educational contexts.

Additionally, research has explored how LLM-based platforms can support project-based learning. One study showed that platform-integrated LLMs can act as autonomous tutors, guiding students through self-directed projects and yielding better educational outcomes [5]. Additional evidence indicates that LLMs can function as mentors in traditional classroom contexts, such as teaching software architecture, thereby extending their utility beyond informal learning environments [6].

Studies on the practical adoption of AI tools argue that widespread student use of generative AI necessitates early curriculum integration. Researchers noted that "guerrilla" usage of AI by students is expected to grow and recommended embedding AI from the outset of courses [7]. Following similar recommendations, frameworks for integrating AI into project-based learning scenarios have been proposed [8]. Real-world practice of integrating AI into educational contexts confirms that following such recommendations and embracing AI increases course satisfaction in traditional academic settings [9], enhances student satisfaction and overall learning outcomes in project-based courses [10], and demonstrates the usability of LLMs even in sensitive contexts such as middle-school projects [11].

Collectively, these studies illustrate both the potential benefits and challenges of incorporating AI into project-based learning environments. At the same time, no deep case study of AI use for project based learning and its possible failure modes was conducted, highlighting the significance and novelty of our study.

etf price modeling

ETF price modeling has become a well-studied area of data science. Over the years a wide range of methods and models have been applied to this problem. Classical machine-learning approaches such as SVM, KNN, and Naive Bayes have appeared in studies since at least the early 2010s [12] and continue to surface in more recent work [13]. Traditional time-series models (e.g. ARIMA) remain popular for financial data [14]. More advanced tree-based algorithms such as XGBoost [15] also find use in the domain. Finally, deep-learning techniques have been extensively explored for ETF price prediction [16], [17].

Unlike many investigations that aim to construct automated trading robots, our focus is not on building a production system. Instead, we intend to guide students through a real-world project that employs AI for ETF price modeling and observe how they apply the techniques in practice. Having situated our work within the literatures on AI-assisted education and ETF forecasting, we now turn to the methods employed in our study. In describing these methods, we attend not only to technical details but also to how each phase of the project created opportunities for student learning.

data, methods and analysis

We describe the methods we used for ETF price modeling, as well as the results and performance metrics of the resulting models. All this work was led by the high school students with support from the undergraduate student (specifically on the price modeling work), and mentorship by the scholars. Importantly, we designed each phase of the pipeline not only to produce functional outputs but also to create structured learning opportunities. Students encountered each technical challenge first through AI-assisted exploration, then refined their understanding through scholar mentor-guided debugging and discussion.

web scraping and sentiment modeling

For this study we compiled a historical price dataset for 29 exchange-traded funds (ETFs), covering January 2, 2024 through September 23, 2025. The data were harvested via Yahoo Finance API [18], from which we have downloaded daily closing prices for the ETFs of interest. The ETFs under study were chosen because they collectively span the major sectors of the U.S. equity market, including, but not limited to technology, energy, banking & financials and healthcare. Each ETF was assigned to its primary sector by examining the underlying holdings list provided by the fund sponsor; for example, XLF was mapped to financials, while XOP fell under oil and gas.

This data collection phase served as an introduction to working with APIs and structured financial data. Students used AI tools to generate initial API query code, which worked for the basic requests but required mentor intervention to handle rate limiting and error recovery, which served as an early lesson in the gap between AI-generated scaffolding and production-ready code.

To enrich this pricing dataset with market-sentiment information, we scraped news stories relevant to the same set of ETFs from the NASDAQ website. We first automated the website's internal

ETF	#Sentiment	#Price	Coverage	ETF	#Sentiment	#Price	Coverage
BBJP	55	436	12.61%	SPY	55	436	12.61%
CLOU	0	433	0.00%	SRVR	25	434	5.76%
DXJ	55	436	12.61%	VDE	104	449	23.16%
EUFN	55	436	12.61%	VFH	55	436	12.61%
EWJ	55	436	12.61%	VGK	55	436	12.61%
EZU	55	436	12.61%	VGT	0	433	0.00%
FEZ	55	436	12.61%	VHT	1	433	0.23%
FLJP	55	436	12.61%	VOO	55	436	12.61%
IEUR	55	436	12.61%	VTI	55	436	12.61%
IVLU	55	436	12.61%	XLE	104	449	23.16%
IVV	55	436	12.61%	XLF	55	436	12.61%
IXJ	1	433	0.23%	XLRE	25	434	5.76%
KRE	55	436	12.61%	XLV	1	433	0.23%
QQQ	0	433	0.00%	XOP	104	449	23.16%
REZ	25	434	5.76%				

Table 1: Sentiment Coverage by ETF

search API to download the metadata (publication dates, titles, and links to the full text) of the news articles related to each of the ETFs under analysis. Next, we downloaded the full text of each of the news articles. In total, 260 articles were captured.

This web scraping task proved to be the most challenging for AI assistance. While AI tools could generate generic scraping templates, students discovered that these templates could fail on specific NASDAQ pages due to dynamic content loading and anti-bot measures. This failure also became a learning moment: mentors introduced students to browser developer tools for inspecting page structure and to specialized libraries like curl-impersonate for mimicking authentic browser traffic. Students reported that debugging the scraper and iterating between AI and humans as co-mentors in the task gave them deeper hands-on learning experience into how web technologies work.

For each article we employed gpt-5-mini via OpenAI API with constrained generation, forcing the model to output a sentiment score on a fixed scale from -10 to 10, along with the reason for each of the given scores. After scoring, we linked each article to the sectors represented in its associated ETFs. When an article was relevant to multiple ETFs, we recorded the sentiment for each relevant sector. The sentiment scores were then aggregated on a daily basis within each sector: the mean of all articles posted that day produced an average sentiment value per date-sector pair. We did not implement ETF-specific linking in this iteration due to already low sector-level coverage.

Finally, the sentiment data was merged with the ETF price data by date and sector. For days lacking news coverage in a given sector we set the sentiment to 0, indicating neutral sentiment. The resulting dataset contained synchronized price trajectories and contemporaneous sentiment indicators for each sector, and was used for the subsequent price movement modeling. The final coverage between ETF prices and sentiment scores is shown in Table 1. We acknowledge that this

imputation of neutral sentiment may introduce bias into the model, as a lack of news does not necessarily equate to neutrality.

In summary, the students found the sentiment modeling phase the most accessible for AI as a co-mentor. The task for prompting an LLM to score sentiment aligned naturally with their intuition about language, and they iterated on prompt designs with minimal human mentor (scholar) input. However, this accessibility also introduced a risk: students initially trusted the sentiment scores. Mentors used this as an opportunity to discuss the limitations of LLM-based annotation, including inconsistency across similar articles and the lack of ground-truth validation.

price movement modeling

The price modeling framework is formulated as a two-stage predictive system designed to separately capture the magnitude and the direction of price movements in exchange-traded funds (ETFs). In the first stage, regression-based models are employed to estimate the absolute magnitude of price changes, commonly referred to as price deltas. In the second stage, classification-based models are utilized to predict the directional movement of prices. This modular design allows the framework to disentangle the intensity of market fluctuations from their directional tendencies, thereby providing a more nuanced and interpretable representation of market dynamics.

This two stage architecture was not immediately intuitive to students, who initially proposed predicting raw prices directly. Through a series of stand-up discussions, mentors explained why modeling deltas improves stationarity and why separating magnitude from direction actually simplifies the machine learning problem. The students were then able to use the AI tools to implement both stages, iterating with the mentors, resulting in a cycle that reinforced the conceptual understanding.

To support large-scale experimentation across multiple ETFs, the pipeline incorporates reliability mechanisms applicable to both stages. Model checkpoints and optimal hyperparameters are persistently stored to enable seamless recovery from interruptions. All model outputs and performance metrics are logged in structured JSON files, and individual test-set predictions are archived in CSV format to support detailed post-hoc analysis. Additionally, automated visualizations are generated for each ETF, overlaying predicted and actual price movements alongside sentiment trends to facilitate comparative analysis and interpretation. We discuss the regression and classification stages now.

regression stage

The regression component focuses on predicting absolute price changes (deltas) rather than raw price levels. Modeling price deltas improves the stationarity of the time series (that is, it makes certain statistical properties, like the mean, constant with respect to time), enhances numerical stability, and aligns more closely with the objectives of financial forecasting. To this end, the modeling pipeline evaluates a broad spectrum of approaches, including traditional statistical models, machine learning algorithms, and deep learning architectures, with particular emphasis

on quantifying the incremental predictive value of sentiment information.

To transform the time-series data into a supervised learning framework, the original price series is converted into a delta series representing changes over successive time steps. A fixed lookback window (5 days) is then applied to capture temporal dependencies. For non-sequential models such as Support Vector Regression (SVR) and XGBoost, lagged values of both price deltas and sentiment scores are explicitly constructed as input features. In contrast, for sequential models such as Long Short-Term Memory (LSTM) networks, the data is reshaped into three-dimensional tensors of the form (samples times lookback window times features), enabling the model to learn temporal patterns across multiple time steps.

A diverse set of regression models is evaluated to establish a robust performance hierarchy. A five-day moving average serves as a naive baseline to benchmark predictive improvements. Statistical models, including ARIMA and SARIMAX, are employed for univariate and multivariate forecasting, respectively, with sentiment incorporated as an exogenous variable where applicable. Deep learning models based on LSTM architectures are trained under two configurations, using price data alone and using a combination of price and sentiment data, to isolate the contribution of sentiment signals. Additionally, machine learning models such as SVR and XGBoost leverage lagged price and sentiment features to capture nonlinear relationships inherent in financial time series.

To ensure fair and reliable model comparison, all regression models undergo systematic hyperparameter optimization. Grid search procedures are used to tune key parameters, such as network depth and optimizers for LSTM models, and regularization strength and tree depth for SVR and XGBoost. Model validation and hyperparameter grid search were conducted using a walk-forward time-series cross-validation strategy, which prevents look-ahead bias and ensures realistic out-of-sample performance evaluation.

A key objective of the regression framework is to assess the predictive utility of sentiment information. Accordingly, each model is trained in two variants: a univariate version relying solely on historical price deltas, and a multivariate version that augments price information with sentiment scores. Performance differences, evaluated using metrics such as Mean Squared Error (MSE) [19], directly indicate whether the inclusion of sentiment data yields statistically meaningful improvements in predictive accuracy. Furthermore, automated visualizations are generated for each ETF, overlaying predicted and actual price movements alongside sentiment trends to facilitate comparative analysis and interpretation.

As the students leaned on AI tools to help develop the models, they also learned the important lessons of validation, overfitting, and other implementation lessons rapidly.

classification stage

While the regression stage estimated the magnitude of price changes, the classification stage is dedicated to predicting the direction of price movement. This component formulates the problem as a binary classification task, reflecting the practical importance of directional forecasts in trading and risk management applications. Price changes are transformed into a binary target variable, where positive price changes are labeled as upward movements and non-positive changes

are labeled as downward or neutral movements. This formulation emphasizes directional accuracy, which is often more actionable than precise magnitude estimates in financial decision-making contexts. The predictions from the regression model are then merged with the classifier predictions, where the magnitude of the delta is predicted via the regression model, the sign is predicted via the classifier, and both then multiplied together.

A hierarchical benchmarking strategy evaluated a wide range of classification models. Naive classifiers, such as “all-up” and “all-down” predictors, are included to diagnose class imbalance and establish lower-bound performance baselines. Classical machine learning models, including Logistic Regression, Support Vector Machines, Decision Trees, Random Forests, and XGBoost, are then evaluated. In addition, deep learning models based on LSTM architectures are trained to exploit sequential dependencies in the data. LSTM classifiers operate on sliding windows of historical observations. To predict the price direction on a given day, the model processes data from the preceding lookback period, enabling it to learn short-term trends, momentum effects, and temporal patterns that are not captured by static classifiers.

Several strategies are adopted to enhance model robustness and generalization. Feature standardization ensures that price and sentiment variables contribute proportionately to the learning process. Hyperparameters are optimized using cross-validated grid search procedures, and model selection prioritizes the F1-score rather than raw accuracy to account for potential class imbalance in financial time series.

By the classification stage, students had developed noticeably stronger debugging instincts. When an XGBoost classifier produced suspiciously high accuracy, students independently hypothesized data leakage before consulting mentors—a marked contrast to their earlier reliance on mentor diagnosis. This progression suggests that repeated exposure to AI-generated errors, coupled with guided debugging, can accelerate the development of critical evaluation skills.

results and model performance

The overall model performance comparison is shown in Table 2. ARIMA based regressors generally underperformed and achieved the lowest errors when paired with an XGBoost classifier while they incurred the highest errors when combined with baseline classifiers. In contrast, SVM based regressors consistently outperformed other regressors especially when coupled with either decision tree or random forest classifiers suggesting effective capture of temporal dependencies and enhanced predictive stability. LSTM based models also performed well, benefiting from classifiers that model nonlinear or probabilistic decision boundaries such as decision trees yet they remained inferior to SVM regressors overall.

Across all regression models, the baseline classification strategies consistently produced inferior results whereas learning based classifiers, especially decision trees and random forests systematically improved performance, underscoring the importance of directional price movement modeling.

Additionally, it should be noted that in all cases, the models that included the sentiment as one of their inputs underperformed when compared to their counterparts that had no access to the sentiment information. We hypothesize two explanations: first, sentiment signals may already be

CM	RM	w/o Sentiment		with Sentiment	
		MSE	MAE	MSE	MAE
Baseline (always down)	ARIMA	14057.96	69.80	12236.56	63.31
	LSTM	4829.41	43.72	5333.01	45.17
	MA5	4275.37	41.95		
	SVM	4434.59	37.67	4698.75	41.21
	XGBoost	6151.31	46.90	6178.09	47.08
Baseline (always up)	ARIMA	24242.56	87.57	21465.47	81.07
	LSTM	11611.38	61.51	12347.92	62.95
	MA5	10706.56	59.75		
	SVM	10797.27	55.00	11158.01	58.46
	XGBoost	13550.30	64.30	13616.42	64.54
Decision Tree	ARIMA	8575.77	45.10	7252.22	39.70
	LSTM	<u>2588.23</u>	27.94	3205.64	29.60
	MA5	2301.90	27.06		
	SVM	2911.68	25.03	2890.48	26.75
	XGBoost	3562.57	30.59	3687.00	31.47
Logistic Regression	ARIMA	13978.66	68.82	12163.43	62.31
	LSTM	4821.64	43.33	5319.63	44.72
	MA5	4265.12	41.49		
	SVM	4445.98	37.46	4663.03	40.78
	XGBoost	6133.73	46.45	6151.83	46.56
LSTM	ARIMA	10768.45	58.77	9746.93	54.62
	LSTM	4343.57	38.47	4676.50	39.20
	MA5	3959.76	37.27		
	SVM	4279.30	34.20	4412.85	36.99
	XGBoost	5469.65	41.55	5469.75	41.63
Random Forest	ARIMA	9210.49	47.63	7566.95	41.49
	LSTM	3090.49	30.86	3440.74	31.75
	MA5	2755.15	29.64		
	SVM	2966.56	<u>26.82</u>	<u>3041.15</u>	<u>28.70</u>
	XGBoost	4150.48	<u>33.55</u>	<u>4237.55</u>	<u>33.96</u>
SVM-RBF	ARIMA	13176.76	65.68	11172.10	58.80
	LSTM	4611.89	41.68	5086.47	42.98
	MA5	4058.14	39.93		
	SVM	4309.10	36.12	4529.77	39.39
	XGBoost	5933.58	44.82	5953.84	44.95
XGBoost	ARIMA	9432.30	47.88	7935.71	42.25
	LSTM	3017.13	30.43	3621.75	32.05
	MA5	2706.25	29.44		
	SVM	3256.85	27.34	3262.03	29.15
	XGBoost	4166.37	33.57	4273.45	34.35

Table 2: Average performance measures for classifier model (CM) and regression model (RM) combinations. The best result is shown in **bold**, the second best is underlined.

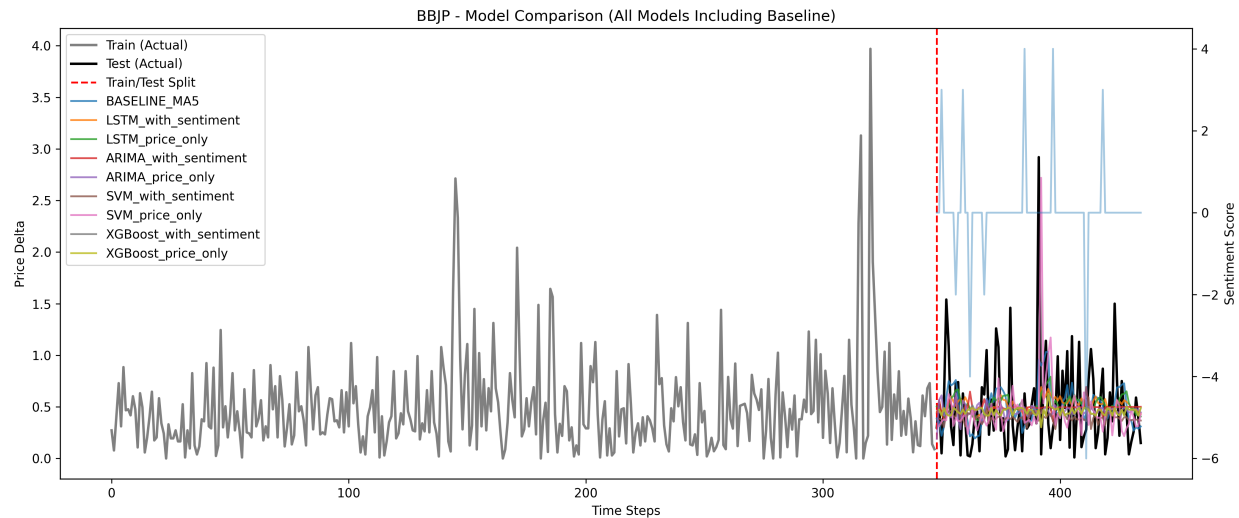


Figure 1: Price delta prediction example (BBJP)

incorporated into price movements by the time they are observable in news; second, the explicit sentiment scores introduce noise that degrades model performance. Figure 1 illustrates this behavior: the price-based SVM model captured a large peak representing rapid price change, whereas the sentiment-augmented model did not.

This negative finding, that sentiment hurt rather than helped, was initially disappointing to students, who had invested significant effort in the sentiment pipeline. However, mentors reframed it as a valuable lesson in hypothesis testing: the purpose of the experiment was not to confirm a predetermined outcome but to discover what the data actually revealed. Students reflected that this was one of the most memorable insights of the project, shifting their understanding of research from "proving ideas right" to "finding out what's true."

The preceding section documented our technical pipeline and its outcomes. We now step back to reflect more broadly on how AI shaped the learning process throughout the project, including its benefits, its failure modes, and the evolving relationship between students, AI-mentors and assistants, and human mentors.

reflections on ai use

We present our reflections on using AI across the project, both as a tool for modeling and analysis, and for co-mentorship between AI and humans. AI proved valuable for code organization, debugging, sentiment analysis and as a writing assistant but also introduced maintenance issues, factual errors, and a need for human oversight. We discuss these findings in two subsections that examine AI's roles as a programming assistant and an analytical tool.

ai as a programming assistant

Beyond sentiment generation, AI proved valuable in improving the structural quality and organization of the experimental codebase. In our study, the students have used the ChatGPT chat interface to aid them in writing code and debugging errors. From the students' perspective, AI tools have assisted them in using established programming patterns and best practices. This way, it enhanced code maintainability, reduced redundancy, and improved reproducibility.

Additionally, AI support was instrumental in debugging, as it helped identify and correct errors, allowing for faster development. AI further contributed by suggesting efficient pipelines for data pre- and post-processing, enabling easier model comparison and evaluation. Notably, students converged on using the standard ChatGPT interface and primarily utilized its free version, though this occasionally limited access to updated model capabilities. From a data science standpoint, AI was also effective in identifying suitable hyperparameter grids for different classes of models. By proposing reasonable hyperparameter ranges, it helped us tighten the feedback loop which led to faster iteration. At the same time, from the mentors' perspective, the use of AI has allowed us to shift our attention from teaching low-level programming patterns and concepts to explaining and discussing higher-level concepts that were required to build the predictive system.

Despite these advantages, AI exhibited notable limitations, stemming from the fact that it lacked full awareness of the overall research objectives and experimental constraints. As a result, human oversight was required to validate architectural decisions and ensure methodological correctness; AI-generated code occasionally required manual refinement and correction for it to be usable. Moreover, while AI could propose hyperparameters and model structures, it could not independently assess their financial interpretability or empirical suitability without human intervention. Additionally, using AI sometimes resulted in code that was hard to maintain and contained subtle bugs that were difficult for non-experts to detect. But we also converted all these limitations into learning moments for the students, giving us an opportunity to dig deeper into concepts, offer specific reading exercises, and do real-time debugging together.

From the students' perspective, given that they lacked specialized programming knowledge, it meant that sometimes the behavior of their code changed in an unexpected way, without them having a way to fix the problem without mentor intervention initially as they improved over time. Conversely, from the mentors' perspective, it meant that at certain points in the project, they had to spend additional time on understanding and corrected hard to maintain, AI-generated code. Overall, the debugging sometimes increased the time spent for both, but in retrospect, it also became a deeper engagement experience between scholars and students.

For example, in programming the news scraper, while AI was able to output generic scaffolding and boilerplate code, the resulting scripts often omitted subtle but essential details, such as the correct code to analyze the source code of the news pages to extract the relevant information. The model also lacked awareness of niche libraries and tools that proved critical in our workflow: for instance, we relied on curl-impersonate to mimic authentic browser traffic, and the AI we used lacked the knowledge of this particular tool. As a result, the practical usefulness of AI for scraping news stories was limited but offered a good starting point for mentors to engage with students on specific conceptual understanding and doing live coding together. Similarly to what we said above, this meant that while students did have an opportunity to try their hand at building the scraper,

deeper engagement with mentors was ultimately required to bring it to a usable state.

In summary, AI functioned most effectively as an intelligent programming assistant rather than an autonomous developer. It excelled in tasks related to code organization, sentiment processing, debugging, and documentation, while strategic decisions, including model design, experimental validation, and domain-specific reasoning needed to remain under human oversight and control. This complementary interaction points to a potential use of AI as an augmentation tool in research programming, rather than a replacement for human judgment.

ai as an analysis tool

While AI played a supportive yet non-autonomous role in the programming workflow of this research, its primary and most effective application was in transforming unstructured news statements into structured sentiment scores. This task benefited significantly from AI's natural language understanding capabilities, as it enabled consistent and scalable sentiment extraction from large volumes of textual data, something that would be hard to achieve using more classical machine learning methods.

These reflections provide the raw material for addressing our research questions. In the following section, we synthesize our observations into answers to RQ1–RQ3 and draw out implications for educators considering similar approaches.

discussion

We synthesize our findings by returning to the three RQs that motivated the study, and then discuss the broader implications for AI-augmented education.

RQ1: How does AI-augmented mentorship influence the learning process compared to traditional methods of instruction?

Our experiment suggests that, at least for our specific cohort, LLM-based tools can serve as an effective co-mentor for students with limited domain expertise, but its role is fundamentally different from that of a human mentor. Within the context of our study, AI excelled at providing just-in-time tactical support: generating code scaffolding, suggesting hyperparameter ranges, and offering quick explanations of unfamiliar concepts. This enabled students to engage with the project immediately rather than waiting for prerequisite instruction.

However, LLM-based tools could not replace the strategic guidance that human mentors provided. Students needed mentors to frame problems, validate architectural decisions, and help them recognize when AI-generated outputs were incorrect. The most effective co-mentoring configuration was rather *complementary*: AI handled routine implementation tasks, freeing human mentors to focus on higher-level reasoning, debugging and conceptual explanation.

Overall, AI tools were effective at tactical guidance, as well as helping with the repetitive tasks of software development: they were always available to the students, and were able to read and understand large codebases faster than the human mentor. At the same time, human mentors were required for making larger, strategic decisions: they had domain knowledge and intuition available, coupled with having a complete vision for the overall project.

We also observed an interesting developmental arc. Early in the project, students accepted LLM outputs with little scrutiny; by the end, they had developed skepticism and debugging skills that allowed them to identify errors independently. This progression suggests that working with imperfect AI outputs, albeit under mentor supervision, may itself be a valuable pedagogical intervention.

RQ2: What are the most frequent failure modes encountered when using AI in a project-based learning context?

We identified three recurring categories of AI failure:

1. **Plausible but incorrect code:** AI-generated code often ran without errors but could contain errors or methodological flaws, including issues associated with assumptions around modeling, feature construction, etc. This is where the expertise of the human mentors became essential, but also where the students could participate in learning through debugging exercise.
2. **Lack of specialized knowledge:** LLM-based tools were great at generating standard templates, but could miss niche libraries that were critical to performing the task at hand. This again required mentors intervention to identify such issues and also walk the students through the nuances of web scraping.
3. **Long term context windows:** As the project lasted over a period of time, it became clear that the awareness of overall research objectives with the AI system became limited, which carried the risk of incorrect advice. We re-oriented ourselves to regular meetings during the analysis and paper writing to ensure alignment.

In our experience, each of these failure modes actually became a teaching moment, and opportunity for hands-on learning. These are occasions for deeper engagement with underlying concepts.

RQ3: Do LLM-generated sentiment scores improve the predictive performance of ETF price forecasting models?

Contrary to our expectations, sentiment features did not improve model performance. We offer two interpretations: 1) sentiment information is already priced into ETF movements by the time it appears in the news articles, rendering explicit sentiment features redundant; and 2) the LLM-generated scores were not completely reliable, warranting additional research.

In summary, AI functions best as a powerful augmenting tool for novices in educational settings: it assists with tactical decisions, routine programming tasks, and basic writing support. Beyond these roles, effective use continues to depend on expert or mentor guidance, particularly for higher-level strategic choices and specialized intuition. The experiment suggests that AI can help students explore diverse programming patterns and make small-scale decisions, but mentorship remains crucial for ensuring rigorous research outcomes. For educators, this means treating AI as a guided support tool rather than a replacement for mentorship, and explicitly building verification and skepticism into the learning goals. While we started with a project idea on developing machine learning models for predicting ETF movements, this project an important learning experience for the students to know how and when to even use AI, and how creative and critical thinking, writing, and discernment are critical skills even in the age of AI. Simple routines such as

regular check-ins and shared documentation also help keep AI-assisted work transparent and allow errors to be caught early.

limitations and future work

Our study was carried out with a single cohort of high-school and early-undergraduate students from US-style educational institutions over the summer, so caution is warranted when extrapolating these findings to other contexts or institutions. Additionally, the primary objective was to investigate how AI can act as a mentoring companion for novices in research and model performance metrics on the actual modeling were secondary to this goal. As such, predictive accuracy should be viewed as exploratory rather than conclusive evidence of any kind of model performance. Additionally, it is important to note that the capabilities of LLMs and LLM-based tools evolve rapidly; therefore, our findings are specific to the versions of the tools available during the summer of 2025.

On the forecasting side, results may be affected by sparse and uneven news coverage, sector-level sentiment aggregation that can dilute ETF-specific signals. While we took steps to reduce leakage risk, alternative sentiment extraction methods remain an important direction for future work. News coverage gaps left about one-tenth of days without sector-specific sentiment, and we imputed zero sentiment for those dates, potentially biasing the signal. Moreover, LLM-based sentiment scoring is not necessarily robust, introducing noise that could mislead downstream models, and for this work, we did not validate the LLM scores against human-coded ground truth. These limitations point to the importance of contextualizing our results and pursuing further research to validate the educational benefits of AI support.

Future work should broaden the participant base to include more diverse cohorts. Longitudinal studies tracking student learning outcomes over time would also clarify the lasting impact of AI co-mentorship.

references

- [1] T. Corbin, M. Bearman, D. Boud, and P. Dawson, “The wicked problem of ai and assessment,” *Assessment & evaluation in higher education*, pp. 1–17, 2025.
- [2] X. Wu, Y. Jia, Q. Zhang, Y. Qin, L. Xiao, and S. Zhao, “Towards robust evaluation of stem education: Leveraging mllms in project-based learning,” *arXiv preprint arXiv:2505.17050*, 2025.
- [3] K. Sasaki and T. Inoue, “Evaluation of llm-based feedback generation for distance project-based learning,” in *International Conference on Collaboration Technologies and Social Computing*, Springer, 2025, pp. 144–160.
- [4] S. C. E. Fung, M. F. Wong, and C. W. Tan, “Automatic feedback generation on k-12 students’ data science education by prompting cloud-based large language models,” in *Proceedings of the Eleventh ACM Conference on Learning@ Scale*, 2024, pp. 255–258.

- [5] Y. Zhu, Z. Ye, Y. Yuan, W. Tang, C. Yu, and Y. Shi, "Autopbl: An llm-powered platform to guide and support individual learners through self project-based learning," in *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, 2025, pp. 1–26.
- [6] S. Gürtl, D. Scharf, C. Thrainer, C. Gütl, and A. Steinmaurer, "Design and evaluation of an llm-based mentor for software architecture in higher education project management classes," in *International Conference on Interactive Collaborative Learning*, Springer, 2024, pp. 375–386.
- [7] L. Al Labadi and A. Ly, "Enhancing statistics education through project-based learning (pbl) and the emergence of chatgpt," *Teaching Statistics*, vol. 47, no. 3, pp. 200–218, 2025. doi: <https://doi.org/10.1111/test.12405>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/test.12405>. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1111/test.12405>.
- [8] T. Wu and M. Chang, "Application of generative artificial intelligence to assessment and curriculum design for project-based learning," in *2023 International Conference on Engineering and Emerging Technologies (ICEET)*, 2023, pp. 1–6. doi: [10.1109/ICEET60227.2023.10525933](https://doi.org/10.1109/ICEET60227.2023.10525933).
- [9] A. Vadaparty et al., "Cs1-llm: Integrating llms into cs1 instruction," in *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*, ser. ITiCSE 2024, Milan, Italy: Association for Computing Machinery, 2024, pp. 297–303, ISBN: 9798400706004. doi: [10.1145/3649217.3653584](https://doi.org/10.1145/3649217.3653584). [Online]. Available: <https://doi.org/10.1145/3649217.3653584>.
- [10] L. Zhang and W. Zhang, "Integrating large language models into project-based learning based on self-determination theory," *Interactive Learning Environments*, vol. 33, no. 5, pp. 3580–3592, 2025.
- [11] S. Zha, Y. Qiao, Q. Hu, Z. Li, J. Gong, and Y. Xu, *Designing child-centric ai learning environments: Insights from llm-enhanced creative project-based learning*, 2024. arXiv: [2403.16159](https://arxiv.org/abs/2403.16159) [cs.HC]. [Online]. Available: <https://arxiv.org/abs/2403.16159>.
- [12] M. Ucar, I. Bayram, and A. M. Ozbayoglu, "A two-level cascade evolutionary computation based covered call trading model," *Procedia Computer Science*, vol. 20, pp. 472–477, 2013.
- [13] R. P. B. Piovezan, P. P. de Andrade Junior, and S. L. Ávila, "Machine learning method for return direction forecast of exchange traded funds (etfs) using classification and regression models," *Computational Economics*, vol. 63, no. 5, pp. 1827–1852, 2024.
- [14] S. Sharma, U. Anand, V. Aggarwal, P. Jain, and A. Anand, "Multimodel forecasting of technology-focused etfs using arima and lstm," in *2025 3rd International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS)*, IEEE, 2025, pp. 559–565.
- [15] X. Li, L. Yang, C. Zha, and Y. Xu, "Xgboost algorithm-based model for predicting mutual fund returns," *Computational Economics*, pp. 1–26, 2026.
- [16] T. Kundu and E. Pinsky, "Predicting daily stock price directions with deep learning models," *Machine Learning with Applications*, vol. 22, p. 100744, 2025.
- [17] M. Hong, Z. Chen, W. Mahmoud Soliman, and K. Zhang, "A comparative study of lstm, lightgbm, and autoregressive model in narrow-based etf market prediction with multi-ticker models," in *Proceedings of the 6th International Conference on Machine Learning and Machine Intelligence*, ser. MLMI '23, Chongqing, China: Association for Computing Machinery, 2024, pp. 10–16, ISBN: 9798400709456. doi: [10.1145/3635638.3635640](https://doi.org/10.1145/3635638.3635640). [Online]. Available: <https://doi.org/10.1145/3635638.3635640>.

- [18] R. Aroussi, *Ranaroussi/yfinance*, original-date: 2017-05-21T10:16:15Z, Jan. 2026. Accessed: Jan. 21, 2026. [Online]. Available: <https://github.com/ranaroussi/yfinance>.
- [19] M. D. Schluchter, “Mean Square Error,” en, in *Wiley StatsRef: Statistics Reference Online*, R. S. Kenett, N. T. Longford, W. W. Piegorsch, and F. Ruggeri, Eds., 1st ed., Wiley, Sep. 2014, ISBN: 9781118445112. DOI: 10.1002/9781118445112.stat05906. Accessed: Jan. 21, 2026. [Online]. Available: <https://onlinelibrary.wiley.com/doi/10.1002/9781118445112.stat05906>.