**Sprint-1**

**Introduction**

In this Sprint, the purpose was to create a application that can ingest data and add windowing features. The following sections contain the User Stories I worked on with a detailed description of the Tasks I worked on.

**User Stories**

I worked on the following User Stories:

[**XGB: Attention-CNN-LSTM + XGBoost Hybrid Forecasting #589**](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/589)

**Conditions of Satisfiability:**

* Accuracy: Model outputs match expected shapes and value ranges; embeddings dimension is correct.
* Speed: Full inference (window → CNN-LSTM → XGBoost → JSON) completes in ≤ 5 s on sample data.
* Robustness: Pipeline handles edge cases (e.g., missing bars) without errors.
* Integration: Crew AI agent consistently applies probability thresholds and returns valid recommendations.

**Definition of Done:**

* Data ingestion, cleaning, normalization, and windowing scripts are in place and tested.
* CNN+Attention+LSTM model code, training loop, and checkpointing are implemented.
* Embedding extraction utility correctly dumps hidden states with labels.
* XGBoost fine-tuning pipeline (hyperparameter search, training, evaluation) is complete.
* Inference script loads both models and produces the required JSON.
* Crew AI decision agent returns the correct BUY/SELL/HOLD payload.
* One end-to-end integration test passes under performance budget.

**Tasks**

[XGB.2 CNN-Attention-LSTM Pretraining #616](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/616)

[XGB.2.1 Implement 1D-CNN encoder (PyTorch or Keras) (8 ph) #644](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/644)

[XGB.2.2 Add self-attention layer over time steps (6 ph) #645](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/645)

[XGB.2.3 Stack an LSTM decoder with regression/classification head (6 ph) #646](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/646)

[XGB.2.4 Write training loop with loss, optimizer, checkpointing, and early stopping (10 ph) #647](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/647)

[XGB.2.5 Log training/validation metrics and save best model (4 ph) #648](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/648)

[XGB.3 Embedding Extraction #638](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/638)

[XGB.3.1 Freeze the trained CNN+Attention+LSTM backbone (4 ph) #649](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/649)

[XGB.3.2 Forward all windows through the network, extracting final hidden-state vectors (4 ph) #650](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/650)

[XGB.3.3 Save embeddings and corresponding labels to disk (e.g., CSV or HDF5) (4 ph) #651](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/651)

[XGB.4 XGBoost Fine-tuning #639](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/639)

[XGB.4.1 Load embeddings + labels into a pandas DataFrame (2 ph) #652](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/652)

[XGB.4.2 Perform train/validation split (2 ph) #653](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/653)

[XGB.4.3 Grid-search n\_estimators, max\_depth, learning\_rate (8 ph) #654](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/654)

[XGB.4.4 Train the best XGBRegressor or XGBClassifier; evaluate and save the model (4 ph) #655](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/655)

[XGB.5 Inference Wrapper #640](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/640)

[XGB.5.1 Write a Python script that loads both the deep model and XGBoost (4 ph) #656](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/656)

[XGB.5.2 Given the latest window, compute cnn\_lstm\_score and xgb\_prob (4 ph) #657](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/657)

[XGB.5.3 Emit JSON with date, scores, and probabilities (2 ph) #658](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/658)

[XGB.6 Crew AI Agent Development #641](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/641)

[XGB.6.1 DecisionAgent: Implement prompt logic to map xgb\_prob to BUY/SELL/HOLD (4 ph) #659](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/659)

[XGB.6.2 ExplainAgent: Build an agent that ingests the same JSON plus feature importances and outputs a concise rationale (4 ph) #660](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/660)

[XGB.6.3 End-to-end test chaining InferenceAgent → DecisionAgent → ExplainAgent (6 ph) #661](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/661)

[XGB.7 Integration Testing & Performance #642](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/642)

[XGB.7.1 Write integration tests covering data fetch → model inference → Crew AI decision (6 ph) #662](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/662)

[XGB.7.2 Benchmark end-to-end latency; optimize any slow steps to meet the ≤ 5 s target (6 ph) #663](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/663)

[XGB.8 Backtesting & Evaluation #643](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/643)

[XGB.8.1 Prepare historical test sets and data conversion (4 ph) #664](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/664)

[XGB.8.2 Implement backtesting harness to simulate the full inference + decision pipeline over history (6 ph) #665](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/665)

[XGB.8.3 Calculate performance metrics (e.g., accuracy, return, drawdown) and analysis (4 ph) #666](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/666)

[XGB.8.4 Generate a backtesting report and visualizations (4 ph) #667](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/667)

**Tasks I Worked On**

[XGB.1 Data Ingestion & Windowing #603](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/603)

I have developed the application to fetch OHLCV via yfinance and drop missing data and then normalize features and then slice into overlapping windows. The task is estimated at 20 person hours but took 26 to complete.

**Summary Table of Work**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| UserStory GitHub Issue ID | User Story | Story Points | Task GitHub Issue ID | Task | Task Hours | Status | Actual Hours |
| [XGB](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/589) | Attention-CNN-LSTM + XGBoost Hybrid Forecasting |  | [XGB.1](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/603) | [Data Ingestion & Windowing](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/603) | 20 | Complete | 26 |

**Summary Table of Commits**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Commit Number | Commit Description (exactly as in github) | User Story | Task |
| June 1st, 2025 | 14d037ee180fe9986738a03fc08d9e3ef0239d0b | [XGB data ingestion and windowing](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/pull/623/commits/14d037ee180fe9986738a03fc08d9e3ef0239d0b) | [XGB](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/589) | [XGB.1](https://github.com/Rivier-Computer-Science/AI-Agent-Stock-Prediction/issues/603) |