FinTech Unit 10 Time Series Homework Grading Rubric

Criteria	Ratings			
Time-Series Forecasting • **Hodrick-Prescott Filte**r utilized to decompose the **settle** price into trend and noise. • **ARMA** Model used to forecast returns. • **ARIMA** Model used to forecast settle price. • **GARCH** Model used forecast volatility. **Time Series analysis** • Purchase of the yen analyzed for or against. • Risk of the yen analyzed. • Confidence of models as a basis for trading analyzed.	**35 Points Mastery** • Completed 7 out of 7 requirements • Code runs without error and produces the assigned results • Code accounts for all possible scenario • Code is free of bugs	**34 > 28 Points Approaching Mastery** • Completed 5 out of 7 of requirements • Code runs without error • Code produces results as expected 80% of the time	**28 > 23 Points Progressing** • Completed fewer than 3 out of 7 requirements • Code runs without error • Code produces results, but not necessarily the correct results	**23 > 0 Emerging** • Completed 1 out of 7 requirements • No submission • Code runs with error
Linear Regression Forecasting • Data prepared, returns and lagged returns created and data split into training and testing. • Linear Regression model fitted. • Predictions made using testing data. • Out-of-sample performance. • In-sample performance. **Linear Regression Analysis** • Model performance analyzed for out-of-sample and in-sample data.	**35 Points Mastery** • Completed 6 out of 6 requirements • Code runs without error and produces the assigned results • Code accounts for all possible scenario • Code is free of bugs	**34 > 28 Points Approaching Mastery** • Completed 4 out of 6 of requirements • Code runs without error • Code produces results as expected 80% of the time	**28 > 23 Points Progressing** • Completed 2 out of 6 requirements • Code runs without error • Code produces results, but not necessarily the correct results	**23 > 0 Emerging** • Completed 1 or none out of the 6 requirements • No submission • Code runs with error
Coding Conventions/Formatting • Appropriate header, name, short description at top of the notebook • Imports are at the top of the file, just after any headers or subheads. • Files read in from relative file path • Functions and variable names are descriptive, lowercase, with words separated by underscores • Clean code, no repetition, maintainable and highly reusable code.	**10 Points Mastery**	**9 Points Approaching Mastery**	**8 Points Progressing**	**8 > 0 Emerging**

• Appropriate code wrapping and cell sizes • Appropriate subheads as needed				
Deployment/Submission • Files submitted in personal repo • Appropriate directory structure with correct files needed to run scripts • Appropriate commit messages • Appropriate README	**10 Points Mastery**	**9 Points Approaching Mastery**	**8 Points Progressing**	**8 > 0 Emerging**
Documentation/Comments • Code is well commented with concise, relevant comments	**10 Points Mastery**	**9 Points Approaching Mastery**	**8 Points Progressing**	**8 > 0 Emerging**