Repository and Mining of Temporal Data
Milestone 4 Evaluation

Team Members:

Jessica Nguy	jnguy2014@my.fit.edu
Siomara Nieves	snieves2014@my.fit.edu

Faculty Sponsor and Client:

Philip Chan	pkc@fit.edu

Meetings With the Client:

Jan 28, Feb 12

Progress of Current Milestone:

Task	Completion %	Jessica	Siomara	To-Do
Q3	0%	0%	0%	Pseudocode, Implement, Test
NarrowData	80%	70%	10%	Test Cases, Tag Search
Django Website	80%	0%	80%	Q2 graphs, Q3 graphs
Save File Uploads	100%	10%	90%	Improve code
Database Input Handling	100%	60%	40%	N/A
Improve Current Code	80%	45%	35%	Speed and Optimization, Compatibility
Project Plan, Evaluation Document, Presentation	100%	50%	50%	N/A

Discussion of Each Accomplished Task:

Q3: Has not been worked on during this Milestone; will be expanded and focused on more during Milestone 5.

NarrowData: The program has several functions, the main one which returns a list of filenames that have their timestamps occur before the first timestamp of the target variable and whose last timestamp occurs after or on the last timestamp of the target variable. This is to ensure that when calculating correlations for various lag times that there will be a value to map to each of the values in the target. The function connects with the database to query whether or not each file in the database matches the requirements to be compared to the target variable.

The function starts by finding the ID number of the target variable, then retrieving the start timestamp and end timestamp of the target. These timestamps are then compared to the start and last columns of the granularity table in the database; the files that are less than the start timestamp and equal to or greater than the last timestamp have their ID numbers returned. From the list of ID numbers the program then compares the numbers to the IDs in the database, which the filenames are then returned as a list.

Currently there is a plan to implement a function that retrieves filenames that have at least one matching tag, however it is not implemented. Further testing, especially with actual data, is needed.

Django Website: The backend for the application has been written for Q1 and moving on to Q2. The website consists of the home page which allows the user to either only upload a file to contribute to the system or to upload and analyze the user's file. If the user has chosen to analyze their file, the system saves the target variable along with the metadata (tags, granularity, etc) into the database and calculates the graphs for answering Q1 (time vs data; time vs zscore of data; time vs change in values; time vs zscore change in values). After analyzing the file, the system redirects to a new page showing the results. Part of the backend code for Q2 has been written on the framework, as of now it searches through all other files that have been uploaded to the system but is yet to create and show any of the graphs for this section (top-k variables, correlation). Moreover, Q2 already has part of its views written down as well as its urls and front-end templates. As for Q3, it will be further expanded on the next milestone.

Save File Uploads: The files uploaded by users are now saved on a local directory and can be accessed and changed on the admin page. Before this milestone the files were only opened and closed while calculating Q1. When the application is switched from a local host to a hosting site, the files will have to be migrated to a cloud computing service such as AWS. Each file has an ID that makes thing easier for narrowing data when calculating Q2 and Q3.

Database Input Handling: The framework is linked to an sqlite database which prompts the user to choose a file, enter the target variable, and 4 tags to describe the variable; when the user uploads the file, all this input is saved in the database. The application raises an error to the user in case the file was not in .csv format, if the user did not specify any of the metadata, or if there is a mistake in the document. While reading the file, the first and last timestamp are calculated by the program and saved in the database in another table for granularity and narrowing data purposes. The primary key is retrieved by the program when each file is uploaded, incremented, and then the new number is assigned to the uploaded file. This is to ensure that when traversing tables when narrowing data that all the files are in order.

Improve Current Code: Q1's 'Is there a significant change in the target variable?' has been completed and has been implemented on the web application. The user uploads a file with data and timestamps and the system processes it and loads 4 graphs on the next page with the information. The graphs are time vs data; time vs z-score of data; time vs change in values; and time vs z-score of change in values. The drawback of doing this implementation was that the system went from loading the first 2 graphs in less than 5 seconds to loading all 4 graphs in almost 1 minute. The system allows the user to only upload data and to upload and analyze for Q1.

Q2's 'What are the top-k variables affecting the data?' has been written on an IDE to later be implemented on the Django framework. The IDE is Pycharm, and as of now is working correctly with NarrowData. Q2 calculates the correlation of all lag times; as of now the lag time has been set to 5 and the top-k number has been set to 3. Changes to the two variables to be user-defined will be allowed and adjusted in a future milestone. The three visualizatons that Q2 presents is a graph of the target versus the top-k variables on a line graph; the second is a bar graph depicting the correlations of the top-k variables against the target variable; and the last graph is a bar graph depicting all the lag times of the top-1 variable. The third figure is to explain the reasons why that variable was chosen at that specific lag point.

An additional class, ParseData, has been created to help with function flow and organization. ParseData is the class that prepares the data to calculate correlation. It accepts two lists and the lag number - the target list and the list to be prepared. ParseData searches through the variable list for the timestamp that matches the target's first timestamp, from there it deletes items from the beginning of the variable list until there is n-number of items before the target variable timestamp, which is the lag. ParseData also contains a trimData, which removes variables from the variable list until the length matches the target list. These two functions are to ensure that when graphing to matplotlib and calculating correlation that there are no errors.

As of this milestone, the framework has Q2's views, urls, templates, and searches through the database in order to gather the files that have been uploaded before to later show Q2's graphs.

Evaluation Document, Presentation: The evaluation document was written collectively by using Google Docs and the presentation by using Google Slides.

Discussion of Contribution for Each Task:

Jessica: Programmed and tested NarrowData with ‘dummy’ database; created new version of Q2 to be compatible with NarrowData including correlation calculations, lag time calculations, and visualizations for the three figures. Implemented ParseData to help with Q2 and NarrowData analysis. Created a program called ParseComma to be an upgraded and simplified version of parseCSV. Created an Upload code to accept user input and automatically calculate necessary information to populate database with metadata. Updated database with a new column for target variable search to be implemented in future Milestones. Contributed to writing the Milestone 4 evaluation, and Milestone 4 presentation.

Siomara: Programmed new view for users to be able to only upload file and not analyze it in order to populate the system, wrote the code for graphs to be able to be shown on website by using JSON objects and mpld3 library, rewrote part of Q1’s previous code to make it compatible with Django, JSON, and mpld3, added front-end design for the home page, the upload page, and the results page, updated sqlite database with new model fields, wrote part of Q2 for sqlite database, added Q2’s templates, urls, view, added showing errors when metadata is missing, wrote code for uploaded files to be saved on local directory. Contributed to writing the Milestone 4 evaluation, and Milestone 4 presentation.

Plan for the Next Milestone:

Task	**Jessica**	**Siomara**
Q3	Pseudocode, Implement, Ensure compatibility with other classes, Test with dummy database for accuracy	Brainstorm, research, Django framework
Narrow Data	Implement Tag search, Further testing	Implement on framework
Showcase Documents	Prepare and turn in documents before due date	Prepare and turn in documents before due date
Target Variable Search	Implement and Test user	Implement and Test, User

	registration, edit Upload Code	registration on Django
Upload more files	Find more .csv files, test dummy database for accuracy, test with actual database for accuracy	Find more .csv files, Django display search results
Q2	Ensure that NarrowData changes are reflected correctly in Q2, add z-score graphs to Q2	Backend for analysis of Q2, show graphs
Improve Current Code	Speed, Optimization, Accuracy, Compatibility	Speed, Optimization, Accuracy, Compatibility
Poster	Poster Sketch, Research	Design Poster
Evaluation Document, Presentation	Write evaluation document, Create presentation, put code on GitHub repository.	Write evaluation document, Create presentation, put code on GitHub repository.

Discussion of Each Planned Task:

Q3: Create pseudocode and plan Q3 before implementing it. Q3.py will use a linear regression library; further research is needed on this topic. Testing the API is necessary to determine the return types the library uses and how to handle them. If necessary, a coding of the linear regression formula will be done with a function in Q3. Write, test, and implement Q3 on the website as well as in Pycharm.

After performing Q2, the top-k variables will be put on a list that will include the lag number, correlation value, and ID number of the item in the database. These values will help in calculating linear regression as well as retrieving values from the database. To test that the linear regression formula is working, files will be split up 80:20; 80% of the file will be the earlier timestamps, whereas the 20% is the timestamps that occur later and will be used with linear regression to predict the values to determine accuracy. Changing the number of top-k values retrieved from Q2 will also be necessary to ensure accuracy in Q3.

Narrow Data: Implement the tag search function to provide additional narrowing of data. The tag search will be used in conjunction with the granularity search option. The program will retrieve the list of tags assigned to the target variable and compare each to the tags in the database. Matching tags will have the ID number of the file returned in a list; these numbers will be compared to the numbers retrieved when performing the granularity search. Duplicates will be removed, and the final list that is returned and used in Q2 will be the list of file names.

Showcase Documents: There are several documents due during the course of Milestone 5; there is the Exhibit Requirements document due on February 14th; further questioning is necessary and a meeting with the faculty sponsor is scheduled for February 12th. The Poster for the Showcase is due on March 17th and will need to be designed and a rough sketch be completed and presented during the faculty sponsor meeting on February 26th.

Target Variable Search: Implement the search of a target variable within the system. Allow users to see answers to Q1, Q2, Q3 without having to upload any files. Allow user to download the information as well only if the privacy setting for this data is set to public viewing. Implement user privacy settings. Change Upload.py's privacy setting to username. Add inputs for user to be able to indicate how many top-k variables they would like, as well as how many lag times they would like to see. Also ensure that target variable search is working.

Upload more files: Search for more .csv files to add to the system; search for files with different timestamps other than m-d-Y data. Look for files with data other than stock market information (president's approval rating, university data, etc.). Populate 'dummy' database with more information to ensure that NarrowData works correctly.

Q2: finish implementation on the backend side in order to show the graphs and analysis to the user on the website. Work along with NarrowData and the metadata stored in the database to find which variables can be compared to the chosen target variable. Decide whether to show all graphs from all three questions in one page or have a separate page for each one so the website doesn't look cluttered.

Improve Current Code: There are three important factors to keep in mind while implementing the project: speed, optimization, and accuracy. The web application is aiming to process the files and show the results in the shortest time possible, as well as showing the user accurate information; rewrite code that could be simplified. If necessary, edit Python code in IDE to ensure that there is a smooth transition to Django and JSON.

Poster: Design project poster for Showcase. Research past projects to determine the best layout or format of Poster. Double-check with faculty advisor that the Poster meets Showcase standards.

Evaluation Document: : Creation of an evaluation document to indicate what was completed in Milestone 5, future requirements in Milestone 6. Create detailed explanations of completed milestones and detailed plans of future milestones. Create a Presentation for Milestone 5. Upload complete and/or current documents/code to GitHub repository and update links on main page. Write the Milestone 5 Evaluation and create the Presentation before the Milestone 6 due date.

Sponsor Feedback:

Task 1:

Task 2:

Task 3:

Task 4:

Task 5:

Task 6:

Task 7:

Task 8:

Task 9:

Sponsor Signature: ____ Date: ___________

Sponsor Evaluation

- Sponsor: detach and return this page to Dr. Chan (HC 322)
- Score (0-10) for each member: circle a score (or circle two adjacent scores for .25 or write down a real number between 0 and 10)

Jessica	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Siomara	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

Sponsor Signature: ____ Date: ___________