**Background**

Srini’s class and Desheng’s class used the HUG-FM web application. 13 people from Srini’s class. 42 People from Desheng’s class. Data collected is split across two different databases. The schema for the databases is identical. User’s marked 15 pixels in each of 3 images (Houston early 2016 flood, Lumberton, NC before Hurricane Matthew, Lumberton, NC after Hurricane Matthew).

**Files**

dbM1COPY.sqlite3: This database holds all of the data from Srini’s class, and data from 11 people from Desheng’s class. The user Id’s of people from Srini’s class are 119-131. The user ids of those from Desheng’s class are 240-250.

dbM2COPY.sqlite3: This database holds the rest of the data from Desheng’s class. The relevant user ids are 247-277.

(directory) houston\_Validation: Directory containing the validation for the Houston data set

(directory) images: Contains the 3 images being classified

(directory) m1Results: Contains the resultant classifications produced by the HUG-FM application from users in dbM1COPY

(directory) m2Results: Contains the resultant classifications produced by the HUG-FM application from users in dbM2COPY

sampleQuery.py: A sample script illustrating the usage of the database. For all users who received a perfect success rate on test pixels, this script gathers all of the answers to Houston pixel questions, and also all Houston results for these users.

SCHEMA.jpg: A visualization of the relevant part of the schema of the database.

**Detailed Schema Description**

IMPORTANT NOTES:

* Although not displayed in SCHEMA.jpg, every table has a primary key called id
* The name of every table in the database has prefix getlabels\_. So e.g the table image, is really getlabels\_image in the database
* In the below description of columns, the ordering of the columns doesn’t nescessarily match the ordering of the columns in the database. Try pragma table\_info(‘TABLE’) to get correct ordering of columns for table named TABLE in the db
* There are many other tables in the database besides those described here. But they are not relevant to data analysis. They were only relevant to the Django application.

**IMAGE [Info on images being classified by the application]**

* Id: primary key
* Name: Name of the image
* Height: Height in pixels of the image
* Imgfile: path to the image
* Width: Width in pixels of the image
* clusterFile: IGNORE
* featureFile: IGNORE
* mixtureFile: IGNORE
* querysetFile: IGNORE
* Task\_id: IGNORE

**USER [Info on those using the application]**

* Id: primary key
* successRate: User success Rate on test pixels [See testPixels section for more details]
* firstName:
* lastName:
* startTime: When they started using the application
* endTime: When they stopped using the application
* Surv1: What was the longest time you had to wait during the experiment
  + 5: 1-5 seconds, 10: 6-10 seconds, 20: 11-20 seconds, 30: 21-30 seconds, 60: greater than 30 seconds
* Surv2: The website is easy to use
  + 5: Strongly Agree, 4: Agree, 3: Neutral, 2: Disagree, 1: Strongly Disagree
* Surv3: The instructions are very clear
  + 5: Strongly Agree, 4: Agree, 3: Neutral, 2: Disagree, 1: Strongly Disagree
* Surv4: Please express any other feedback or suggestions that you have about the web application

**ANSWER [USER answers to pixel questions on IMAGES]**

* Id: primary key
* Water: 1 if user thinks it is water; 0 otherwise
* Land: 1 if user thinks it is land; 0 otherwise
* Unknown: 1 if user thinks it is unknown; 0 otherwise
* Img\_id: The id of the image from which this pixel comes from [FOREIGN KEY]
* User\_id: The id of the user who is labeling this pixel [FOREIGN KEY]
* Imgcol: The col of the image this pixel comes from
* Imgrow: The row of the image this pixel comes from
* Certainty: The Gaussian Mixture Model certainty level for the classification of this pixel
  + 7: Uncertain, 8: Certain, 9: Very Certain

**RESULT: [A HUG\_FM result based on a USER’s pixel labels for an IMAGE]**

* Id: primary key
* Image\_id: The image that this classification is for [FOREIGN KEY]
* User\_id: The user whose pixel labels were used to generate this classification [FOREIGN KEY]
* Height: The height of the classification [in pxls]
* Width: The width of the classification [in pxls]
* Resultfile: Path to the result

**USERRATING: [A rating by a USER on the classification produced using the pixels they labeled on the IMAGE]**

* Id: primary key
* Rating: The water extent delineation results appear accurate
  + 5: Strongly Agree, 4: Agree, 3: Neutral, 2: Disagree, 1: Strongly Disagree
* Image\_id: the image that the classification is for (that is being rated) [FOREIGN KEY]
* User\_id: The user who is rating this classification that was produced using the labels they gave [FOREIGN KEY]

**TESTPIXELS: [There are 6 total test pixels. 2 for each image. Every user sees all 6 test pixels. The test pixels are pixels we have pre-classified and that are obvious. If the user mis-labels these pixels, we have a hunch that they might not have been paying very much attention when using the application. The successRate for each user on these test pixels is given as a column in the user table]**

* Id: primary key
* Relrow: IGNORE
* Relcol: IGNORE
* Water: 1 if this pixel is definitely water; 0 otherwise
* Land: 1 if this pixel is definitely land; 0 otherwise
* Unknown: Always zero in this table
* Image\_id: The image that this test pixel comes from [FOREIGN KEY]
* Imgpart\_id: IGNORE
* Abscol: The column position (in pxls) of this pixel in the image
* Absrow: The row position (in pxls) of this pixel in the image