CS410 Fall 2020 Course Project

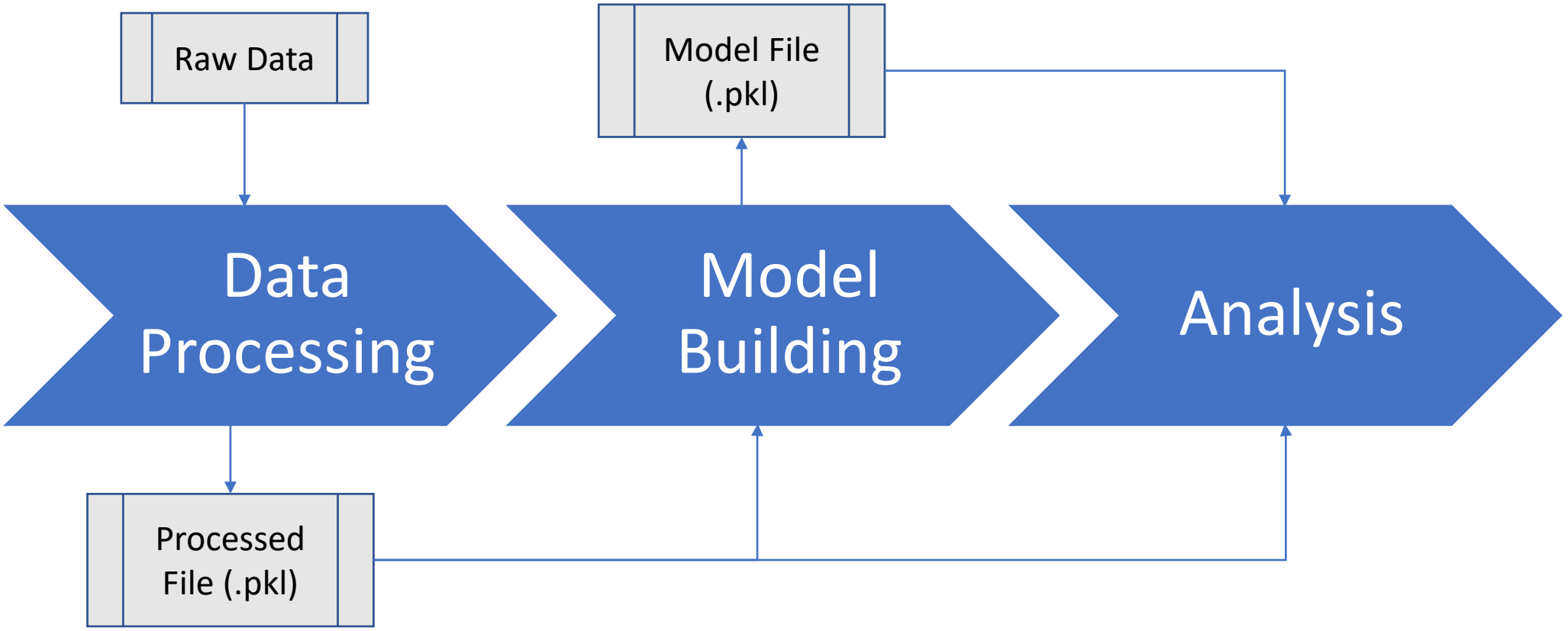
Thiago Seuaciuc-Osorio

netID: thiagos2

Topic Paper

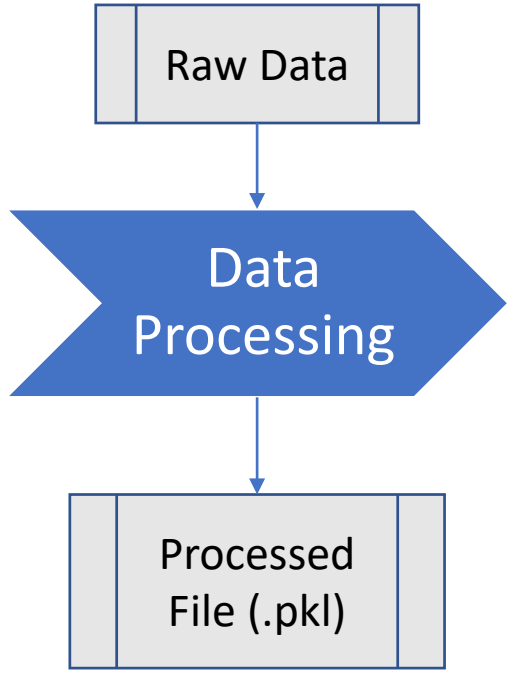
- Hongning Wang, Yue Lu, and ChengXiang Zhai. 2011. Latent aspect rating analysis without aspect keyword supervision. In Proceedings of ACM KDD 2011, pp. 618-626. DOI=10.1145/2020408.2020505
  - With considerable material from reference [3] in the paper on LDA: *D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. The Journal of Machine Learning Research, 3:993–1022, 2003*

Framework



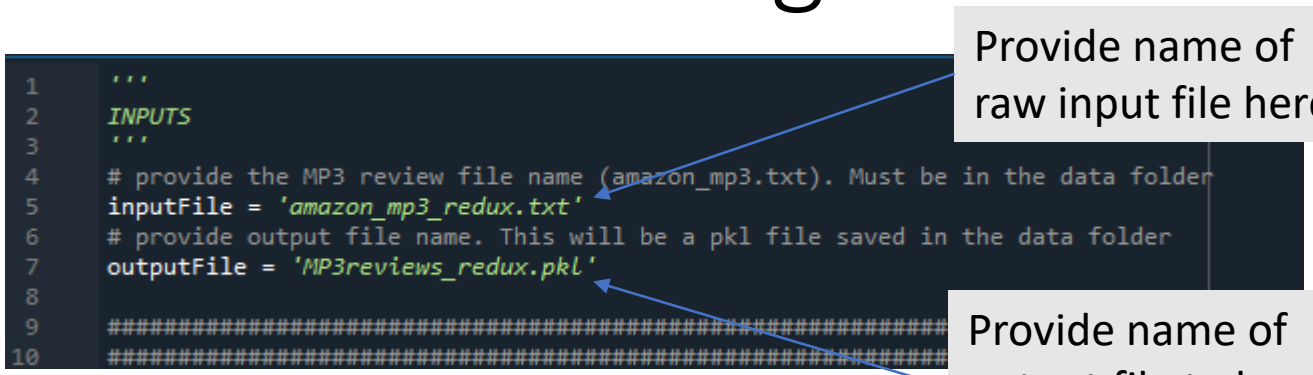
- Data transfer through files
- No command line interface: inputs in first lines of scripts

Data Processing



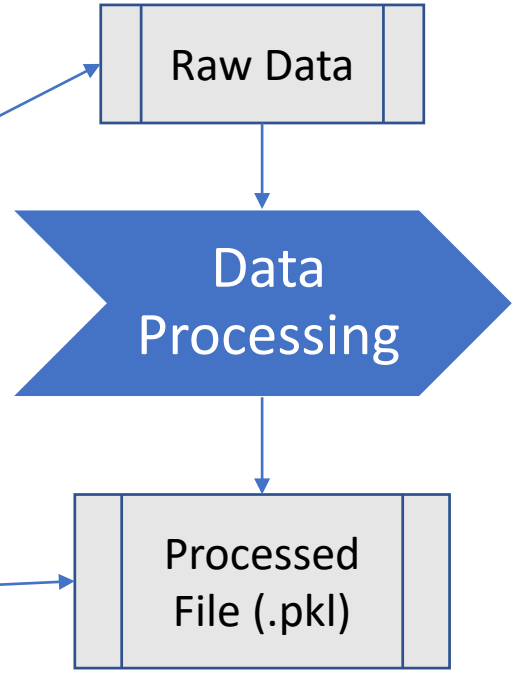
- Input: raw file
- Output: processed file
  - Reviews: list of dictionaries
    - ReviewText: review content (list of words)
    - 'Author' (if available)
    - 'Product'
    - 'Date'
    - 'Rating': list of floats; first is overall rating
  - Vocabulary: list of words in corpus
  - Term-Document Matrix: count of each term (column) in each document (row)
- Processing:
  - Lower case ⇒ remove punctuation ⇒ word tokenize ⇒ remove stopwords (NLTK) ⇒ remove non-alphabetical terms
  - Filter reviews with less than 50 words, and terms appearing in less than 10 reviews
- All files in the *data* folder

Data Processing



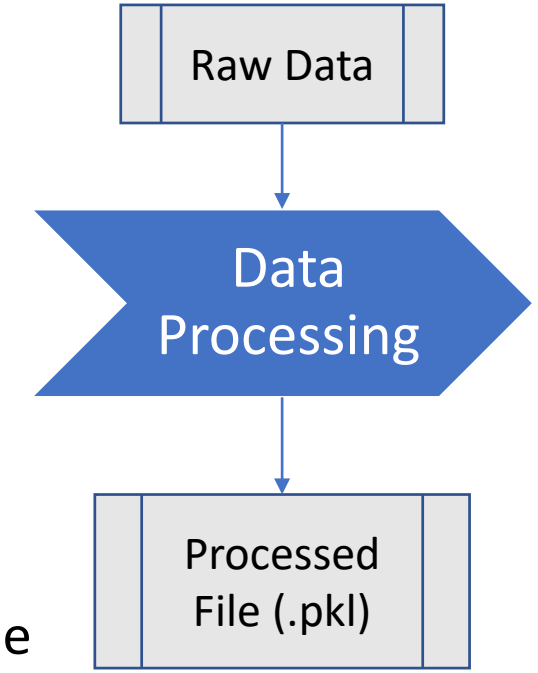
Provide name of raw input file here

Provide name of output file to be created here



- MP3 & Hotels reviews in different formats; need different processing
  - processHOTELreviews.py
  - processMP3reviews.py
  - processMP3reviews_split.py
    - Splits reviews based on rating (low/high)

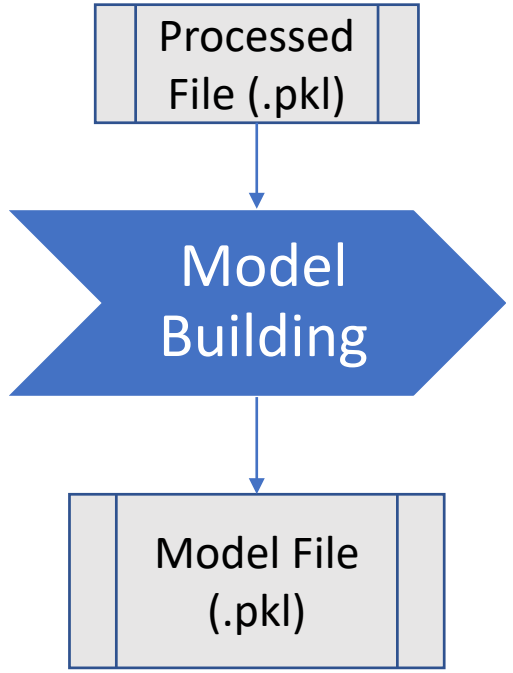
Data Processing



- Main available processed data files:
  - *MP3reviews_low_100.pkl*: processed data of a random sub-sample of 100 reviews with low rating (3 or lower).
  - *MP3reviews_high_100.pkl*: processed data of a random sub-sample of 100 reviews with high rating (higher than 3).
  - *HotelReviews_100.pkl*: processed data of a random sub-sample of 100 hotel reviews
- Main available raw data files:
  - *amazon_mp3_redux.txt*: a small sub-sample of the amazon review dataset in its raw format that can be used to test the data processing codes.
    - Associated processing script: *processMP3reviews.py* or *processMP3reviews_split.py*
  - *Test_redux* (folder): a small sub-sample of the hotel review dataset in its raw format that can be used to test the data processing codes.
    - Associated processing script: *processHOTELreviews.py*

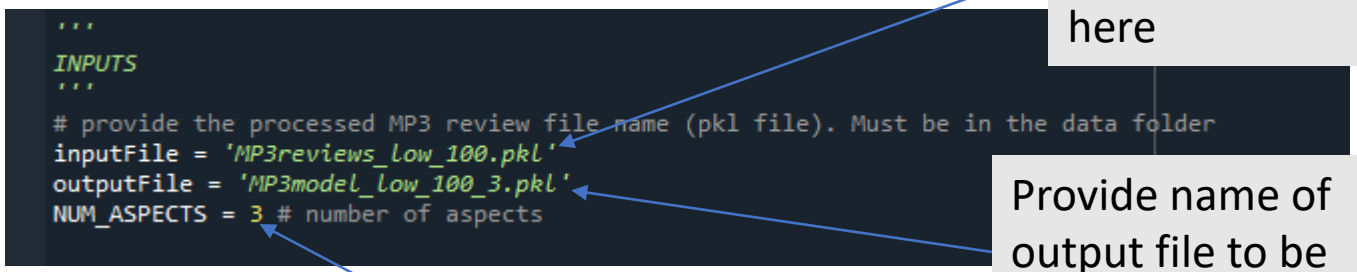
Model Building

- Follows process in subject paper*:*
  - Initialize corpus-level parameters
  - Compute review-level parameters
    - Initialize review-level parameters
    - Iteratively update until convergence
    - Corpus-level parameters are held constant in this process
  - Update corpus-level parameters
    - Review-level parameters are held constant
  - Recompute log-likelihood
  - Iterate until convergence
- Input: processed data file (in *data* folder)
- Output: model file saved to *models* folder
  - Corpus-level parameters
  - Review-level parameters



Model Building

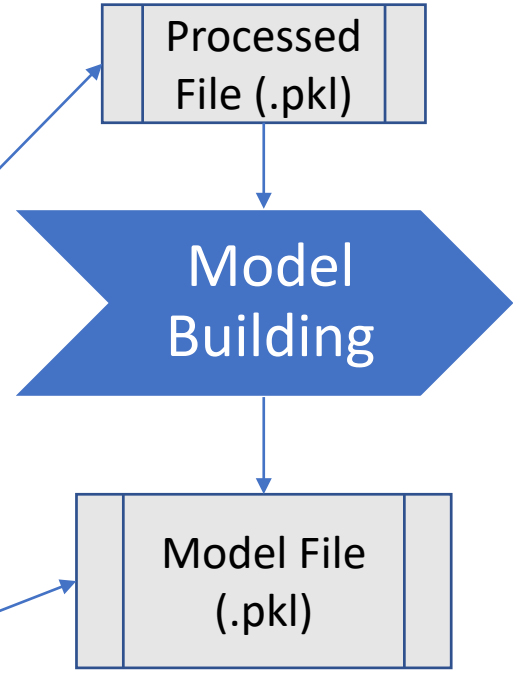
- *estimateModel.py*



Provide name of processed data file here

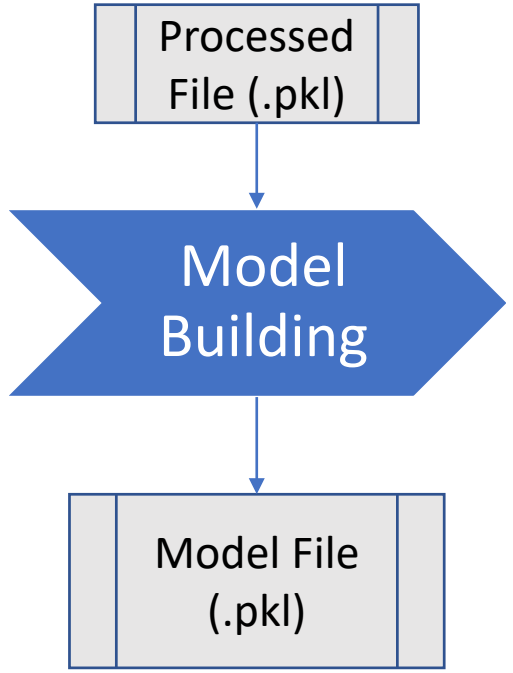
Provide name of output file to be created here

Define number of aspects here



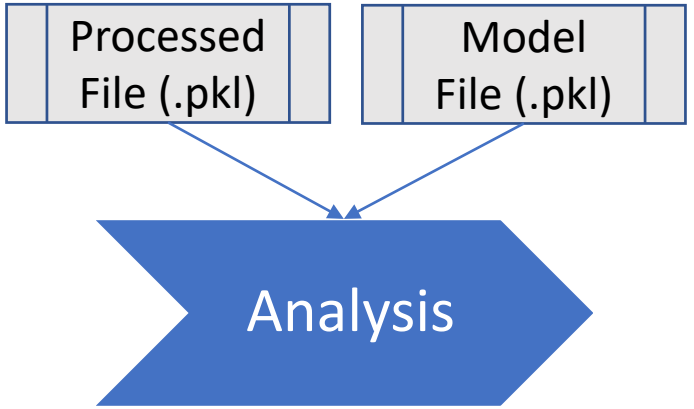
- Data files should be in the *data* folder
- Model file will be saved to the *models* folder

Model Building



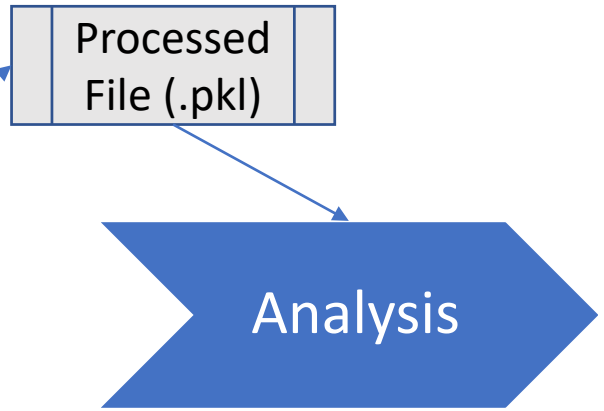
- Available models:
  - MP3model_low_100_3.pkl
    - Model built with 3 aspects on dataset with 100 MP3 reviews with low rating
    - Associated processed data file: *MP3reviews_low_100.pkl*
  - MP3model_high_100_3.pkl
    - Model built with 3 aspects on dataset with 100 MP3 reviews with high rating
    - Associated processed data file: *MP3reviews_high_100.pkl*
  - HotelModel_100_7.pkl
    - Model being built with 7 aspects on dataset with 100 hotel reviews
    - Associated processed data file: *HotelReviews_100.pkl*

Analysis



- For various purposes
- May take either or both of:
  - Processed data file
  - Model file
- Available codes:
  - *getStats.py*: computes basic stats on the review data
  - *getTopAspectWords.py*: retrieves the top words of each aspect (based on aspect word distribution)

Analysis



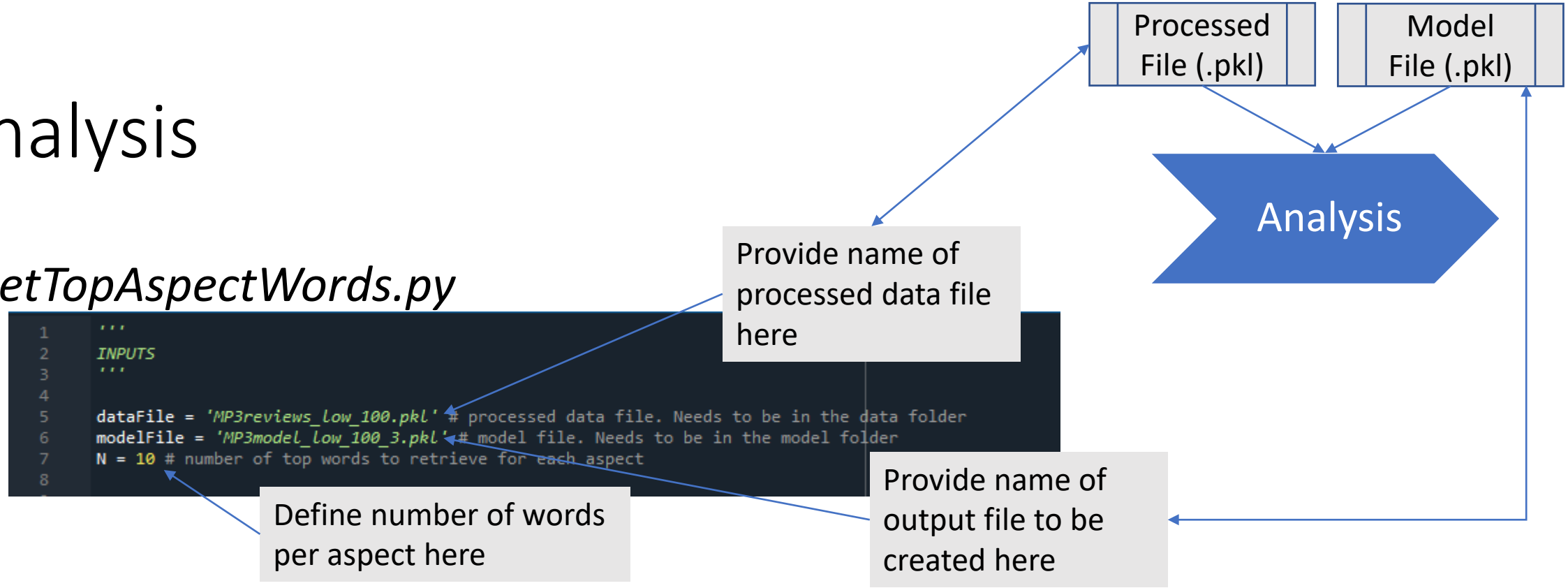
- *getStats.py*

Provide name of processed data file here

```
'''
INPUTS
'''
# provide the processed MP3 review file name (pkl file). Must be in the data folder
inputFile = 'MP3reviews_high_100.pkl'
```

- Script will print on the terminal:
  - The number of reviews in the file
  - The number of unique items (products) reviewed
  - The average length (and standard deviation) of the reviews
  - The average and standard deviation of the overall ratings
- No other outputs or files created
- Data file should be in the *data* folder

Analysis



- *getTopAspectWords.py*

```
'''
INPUTS
'''

dataFile = 'MP3reviews_low_100.pkl' # processed data file. Needs to be in the data folder
modelFile = 'MP3model_low_100_3.pkl' # model file. Needs to be in the model folder
N = 10 # number of top words to retrieve for each aspect

```

- Script print array of words to the terminal
  - No other outputs or files created
- Data file should be in the *data* folder
- Model file should be in the *models* folder

Some Results – Review Statistics

Dataset	# Items	# Reviews	Avg Length (std)	Avg Rating (std)
MP3	676	16012	123.06 (98.00)	3.75 (1.42)
Hotels	1849	47750	125.73 (99.02)	3.96 (1.22)

- Compared to the reported results (see Table 1 in subject report):
  - Slightly fewer reviews. For instance, Table 1 reports 2,232 reviewed hotels, but the available dataset only contains files for 1,850 (one is filtered out)
  - Higher average review length. This is likely because of the slightly different processing. For instance, here the NLTK stopwords were used, which will differ a little from that was used in the paper.
  - Ratings statistics are very similar.

Some Results: Top Words in MP3 Reviews

Low Rating			High Rating		
Zune	Like	**Warranty**	Would	Hours	Software
Bought	Good	Software	Use	Great	Also
Everything	**Time**	Work	New	Use	One
Get	Product	Apple	Zune	Itunes	Like
People	One	Buy	Device	Still	**Sound**
Like	Get	Im	Great	Player	**Quality**
Problem	Screen	Music	**Easy**	Battery	Get
One	Use	Battery	Good	Like	Much
Ipod	Ipod	Unit	Player	One	Good
Player	player	would	ipod	music	player

Some Results: Top Words in MP3 Reviews

- Few similarities with Table 2 in the subject paper:
  - *Low*: problem, time, warranty
  - *High*: easy, sound, quality
- Despite a few similarities, this list is mainly different from the one in Table 2 in the subject paper. Reasons could be:
  - Fewer aspects: the paper modeled 20 aspects and displayed the top 3, while here only 3 aspects were modeled. The higher number of aspects in the paper allow for better topic definition for each aspect, whereas here, the shown 3 topics need to account for all the content in the corpus.
  - The much smaller dataset. This was built on 100 reviews, while the paper used 16,680 divided between the two sub-groups (low/high rating)
  - The different list of stopwords; the list used here may have left more common English words than the list used in the paper.

Note: *time for model computation is the reason to have reduced the dataset (number of reviews) and the number of aspects modeled.*

Challenges

- Model complexity
  - A number of high dimensional numerical optimizations at repeated iterations leads to high computation time to build models
  - This gets worse as the dataset and number of aspects increase
- This has hampered the development of the model on the hotel dataset, where at least 7 aspects are needed for meaning assessment
  - Model build code has not finished running (on set with 100 reviews)
  - Not possible yet to perform the aspect rating analysis
  - Even when the model finishes, the low number of reviews will likely make the results not very robust
- To solve this, efforts are needed for code optimization
  - Most of the time in the project was spent on research to understand the model and approach to be able to make an initial implementation of it
  - Better computation resources would also help

Suggested Testing Procedure

The steps below allow for all scripts provided to be tested without taking too much time. They are already set with the inputs corresponding to this, so they can be run without changes.

- Data processing scripts
  - Run *processMP3reviews.py* to process the MP3 reviews in *amazon_mp3_redux.txt*. This will generate the *MP3reviews_redux.pkl*. All these files are in the *data* folder.
  - Run *processHOTELreviews.py* to process the hotels reviews in the folder *Texts_redux*. This will generate the *HotelReviews_redux.pkl*. All these files are in the *data* folder.
- Model building scripts
  - Run *estimateModel.py* on the reduced MP3 dataset (*MP3reviews_redux.pkl*) to generate the model file *MP3model_redux.pkl*. The suggested number of aspects is 3.
- Analysis scripts
  - Run the *getStats.py* on *MP3reviews_high_100.pkl* to obtain the statistics on that smaller set of reviews.
  - Run the *getTopAspectWords.py* using the data *MP3reviews_high_100.pkl* and model *MP3model_high_100_3.pkl* to obtain the top 10 words in each of the 3 aspects of this model on these reviews.