**Autonomous Merging of Datasets (AMoD)**

Ritika De1, Ashutosh Deoghare2, Harsimrat Kaur3, Julie Wang4, Cheng Zhang5

**Abstract**

Data analysis requires observation, integration, examination, and interpretation of copious amounts of data. However, it is often the case that even numerous features do not provide valuable insight useful for problem solving. Thus, in this paper we explore the idea of designing smaller datasets with key features that significantly add value to the analysis.

The paper describes several experiments designed using machine learning and natural language processing techniques. Upon evaluation it is noted that these experiments merge independent datasets, identify patterns, and effectively shrink feature space to form relatively smaller and more valuable datasets, while improving the results.

**Introduction**

The research addresses the notion that big data does not mean good data as the result of such a time consuming and tedious process of interpreting and evaluating big data is not always rewarding as the features, which were initially assumed to be significant, provide little to no valuable insight to the analysis. Thus, the idea of smaller dataset comprising of key features needs to be explored.

We have designed four experiments that use machine learning and natural language processing (NLP) methodologies and techniques which aid in identifying patterns, merging autonomous datasets and reducing feature space, in order to provide smaller and more meaningful datasets to users; consequently, making the process of data analysis a little less challenging.

In the following section account the data used, along with detailed descriptions for each experiment designed, assumptions, graphical architecture, and output. Further the experiments are evaluated.

**Data**

In order to design and implement the experiments, some common prediction problems were considered, such as, prediction of house prices and crime rate. For instance, if one wants to predict the prices of houses in San Francisco area, perhaps one could expect to see patterns in crime rates or types of crime in the same area. This could potentially offer some valuable insights to the users. Based on such prediction problems, the experiments are designed using eight datasets. The following is the list of datasets along with details of each:

**San Francisco Datasets:**

These datasets were gathered from Kaggle, and data.sfgov.org.

* Inclusionary Housing Dataset- It includes columns such as dev\_year, price, price Zillow, etc., most of the data types are float, integers and objects.
* Police Dataset- This contains 14 columns with float integer and object data types, namely, incident number, descript, time, date, X & Y co-ordinates, category, etc.
* Fire Incident Dataset- This is one of the bigger datasets that includes 34 columns and 4.7 million rows, with information about zip code, call date, incident number, no. of alarms, fire prevention district, etc. Again, the data types are float, integer, and object.
* School Dataset- consists of 17 columns that give information about category, campus address, latitude, longitude, etc.
* Recreation & Parks Department Dataset- has 12 columns with park name, park type, zip code, location, etc. information.

**District of Columbia Datasets:**

These datasets were downloaded from Kaggle, crimemap.dc.gov, and usa.ipums.org.

* DC Residential Properties- comprises of 26 columns with float, integer, and object data types. Some of the column names are bathroom, heat, stories, kitchens, latitude, longitude, and price.
* DC Metro Crime Data- this provides information about type of offense, method, time, date, Xblock, and YBlock. It has 27221 rows of data, which can be very useful in the prediction.
* DC Income Data- it contains 18 columns and 44752 rows. Some of the columns are year, age, famsize, education, etc.

**Data Conditioning and Quality Assessment**

As the data sources are well scrutinized, documented, and were hosted government agencies, it did not require a lot of cleaning and preprocessing. However, as some of the datasets included dummy variables, they were dropped, and any missing values were either replaces with standardized values or with values that were available after some research. Additionally, as inclusionary housing dataset, and fire incident dataset were too large, and consumed approximately 1.5+ GB of memory, the models were designed using only samples for training and testing.

**Tools Used**

For the most part, Python is used for data cleaning, pre-processing, and designing the experiments. However, R and Tableau are also used for in certain experiments, when needed. These programming languages and software were chosen as they are flexible, open source, and offer toolkits and packages that are required to successfully implement the experimental models.

**Experimental Design**

In this section, all the experiments are presented, along with their flowcharts for better understanding to the readers.

Following are the assumptions for the experiments:

1. Clean datasets are used as input.
2. Number of rows of both the datasets must be equal. (Applicable to first 3 experiments)
3. For the experiments that use tree-based models (classification or regression models), the target variable has to be in the last column. Also the dataset that has the target variable should be the input as the second dataframe in the model. For instance, if user wants to predict housing prices, then housing data will be the second dataframe and the house prices will be treated as the target variable, and must be in the last column of the housing dataset.
4. Datasets used in fourth experiment are that of the same location. That is, either both datasets are San Francisco related, or DC related.

**Experiment 1:**

For this experiment, we are assuming there is one common column in two different datasets. For this purpose, we have divided one dataset into two, intentionally keeping one column in common.

The first step here is to merge these two datasets based on a common column. Doing so results in a new dataset that has all the features of the initial two datasets. As our dataset contain both continuous and categorical data, se need to dummy encode categorical data to convert it into continuous. This is because, as we know machine learning models can’t process categorical type of data. Next, these variables needed to scale and transformed so that its mean should become zero and standard deviation one. Later, this data is fed to random forest model based on the problem whether classification and regression. What this random forest model will do, it will rank all features according to its importance to solve the problem. If the problem is of classification, then the data is fed to random forest classifier and if the data is continuous then the data is fed to random forest regressor.

Using the feature importance method of random forest, we can get all these ranked features. The instance of this model is then fed to the SelectFromModel function of scikit-learn. With the model we are passing some threshold. This threshold has been set so that the output of the SelectFromModel must be the features whose importance is greater then that threshold. This gives us lesser number of features to solve our original problem. Hence, with this we get our smaller dataset. The threshold od this SelectFromModel can be set to anything but for our experiment we kept it 0.05.

Following is the graphical representation of the above-described experiment, for better understanding:

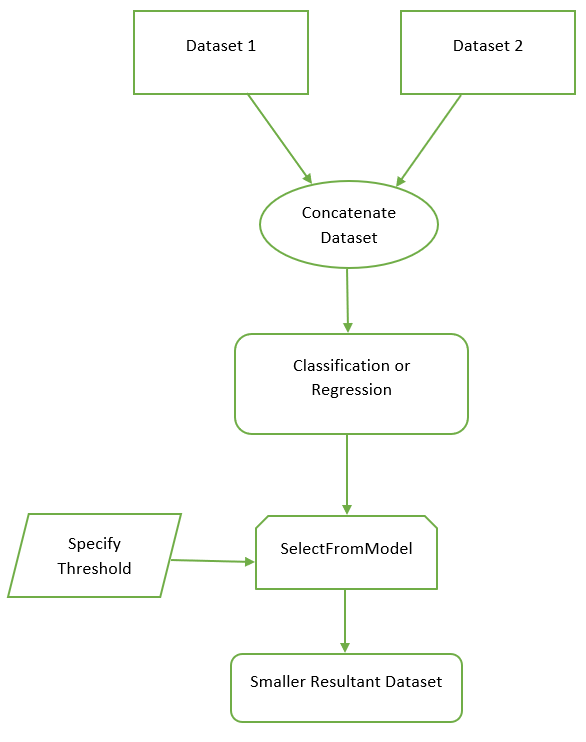


Fig. 1

**Experiment 2**

In this experiment, the first step is to run the machine learning models separately on two independent datasets depending on the problem. As discussed on the above assumption the problem here depends on what is type of variable present at the last column of second input dataset. If the datatype is continuous then the problem is regression and is the dataset is categorical then the problem is classification. This problem can be solved by model random forest regression or random forest classification. The target variable of the first input dataset is being dropped here. The features of the dataset which is being fed to these models is first converted to continuous type. After that, this dataset must be scaled and transformed to get mean zero and standard deviation one.

This data is then fed to these machine learning models to get the important features. Again, the next step over here is to use the SelectFromModel function to get the best features according to the threshold. After performing above process in the end we get important features for both the datasets which means we get small dataset for different problems. Later the idea over here is to concatenate these important features of both the dataset. Doing so will result in getting important features from both the dataset and will result in smaller dataset.

Following is the graphical representation of the above-described experiment, for better understanding:

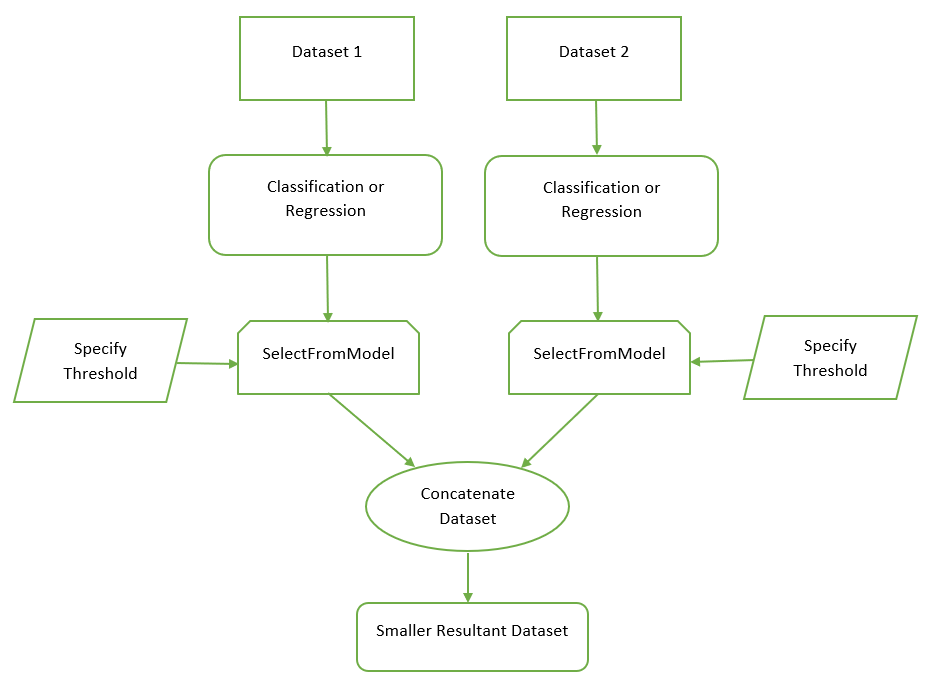


Fig. 2

**Experiment 3:**

In this experiment the first step is to concatenate two datasets irrespective of any common column. Again, the problem for which we want the smaller dataset must be at the last column of second input dataset. After concatenating we are dropping the problem for the first input dataset that is target variable of first dataset. What we are left is with features of both the datasets which are of continuous and categorical type. The next step in this experiment is to separate the continuous and categorical type features. Again, after separating the idea here is to run machine learning model on both continuous and categorical dataset depending on the problem. If the problem is of regression, then run random forest regressor and if its classification then the model used is random forest classification. Before feeding these continuous and categorical datasets into the models the categorical type dataset must be converted into continuous type and both continuous and categorical dataset must be scaled and transformed so that the mean should become zero and standard deviation to one. Next after feeding these data into models we get ranked important features of both continuous type and categorical type. Next, to extract the essential features, SelectFromModel is used with some threshold. Lastly, the extracted features are joined once again, to form a smaller dataset.

Following is the graphical representation of the above-described experiment, for better understanding:

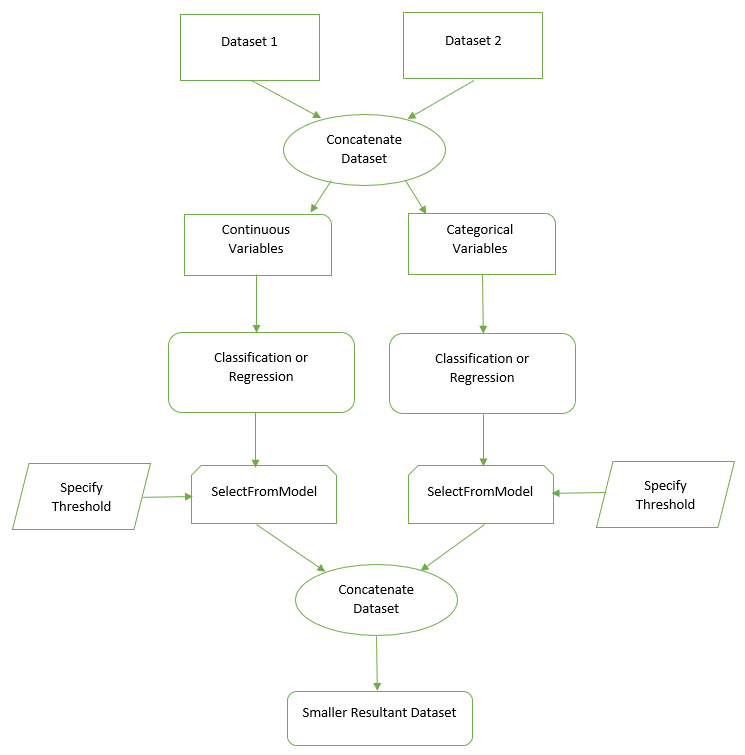


Fig. 3

**Experiment 4:**

In this particular experiment, it is assumed that datasets belonging to the same geographical locations are used. Furthermore, with the aid of a natural language processing (NLP) technique called semantics to identify common features in two or more autonomous datasets. Using Python, first step of the experiment requires creating a list of column names (i.e. headers) for each of the datasets involved. Next, with the help of NLTK, a toolkit specifically designed to offer NLP functions, similar words in the lists of headers are identified. Further, similarity based on cosine-distances between the column names is calculated. The smaller the cosine-distance, the more similar the column names; this means, if the cosine distance between two column names is calculated to be small, then they both may be similar, and can even be considered common. Based on these common (similar) columns, the datasets are merged.

**Experiment Results**

Following are the experimental results of each experiment:

**Experiment 1:**

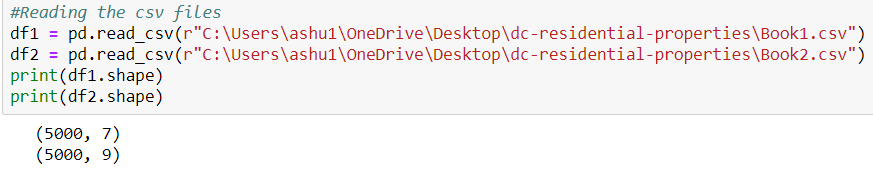


Fig. (a) Input shape of datasets

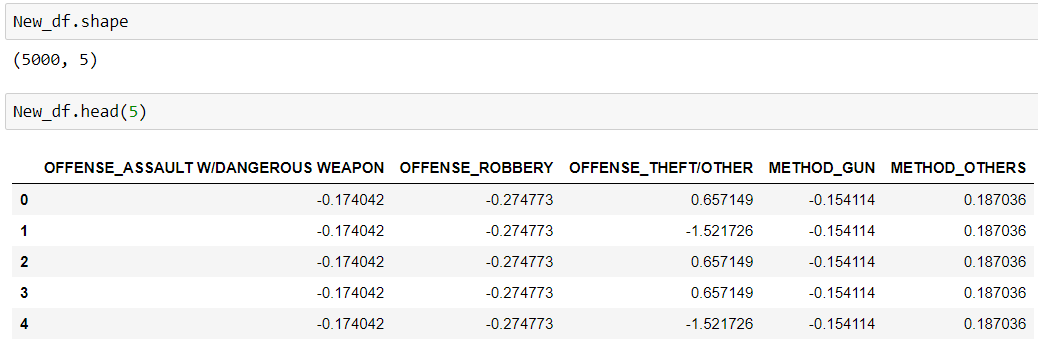


Fig. (b) Output of experiment with shape

**Experiment 2:**

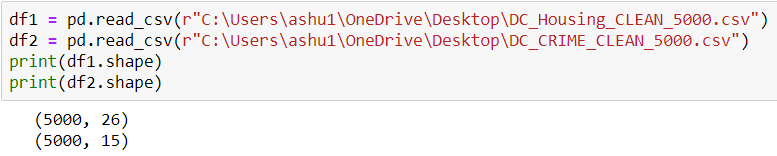


Fig. (a) Input shape of datasets

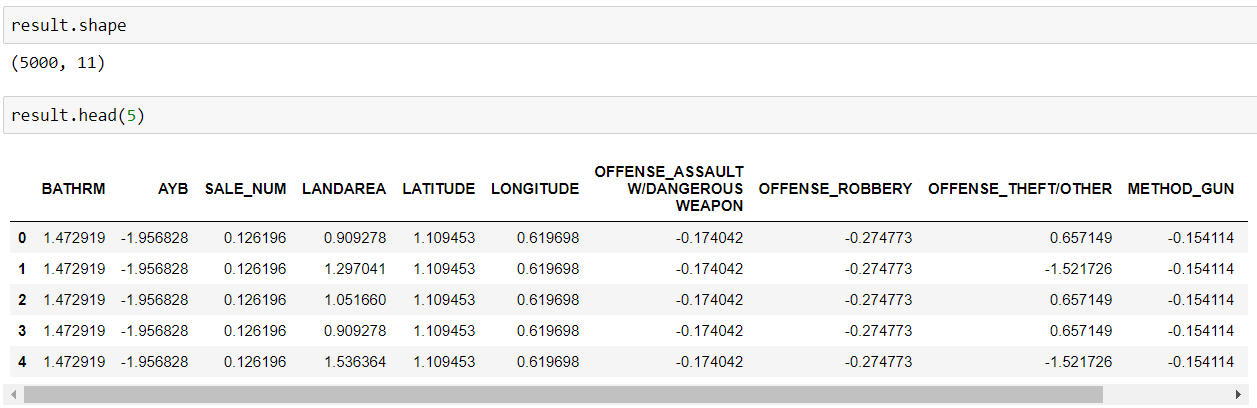


Fig. (b) Output of experiment with shape

**Experiment 3:**

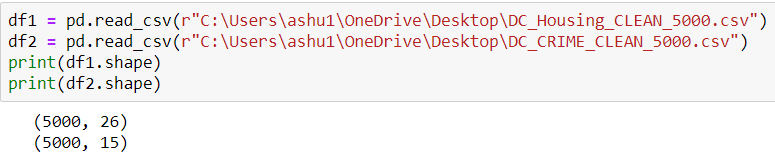


Fig. (a) Input shape of datasets

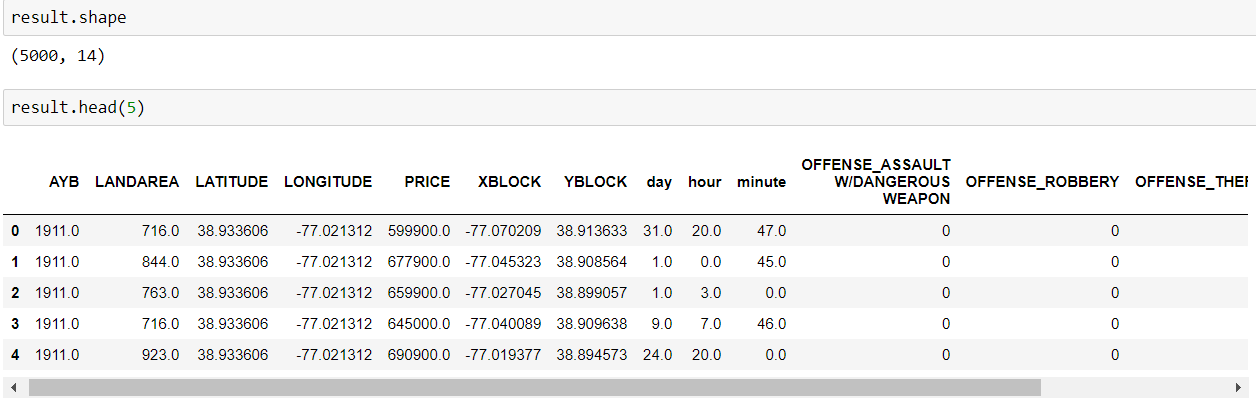
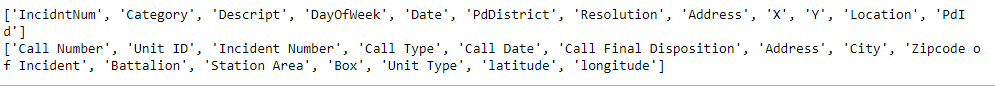


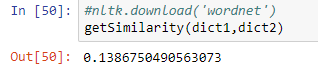
Fig. (b) Output of experiment with shape

**Experiment 4:**

As seen the following screenshot, the column names of two datasets are included in two separate lists:



Further, with the help of NLTK, similarities are calculated:



Lastly, the similar column names can be printed in the following fashion: https://lh6.googleusercontent.com/f3-rlf_YUvsPryHXU1v0Y5eFSyVJDxRYkrOsMf3sZdqkwqVvQ4Mm7dEuBUVu1IQaumKrttyC5ApXqrM-bfbPfYWojFAD94naS75c-LCBJDmwxQpEXgsN3msCg8so4oX4pKhjAv78sPo

**Conclusion and Future Work**

The process of data analysis is a time consuming and tedious one. Often, it entails observing and analyzing datasets that contain mixed variables, belong to multiple domains, or they vary in scale. Our research work so far has allowed us to use multiple techniques, and functions that aid in merging datasets and reducing the feature space. However, the concept of merging any number of independent datasets, and producing one resultant datasets that consists of only the most valuable features is very fascinating.

We are currently working towards gradually building a model that is generic enough to be able to process several kinds of datasets (cross domain & of mixed variables). We will be using the techniques mentioned in the aforementioned experiments. Furthermore, we want to look into deep neural networks, and multi kernel view data fusion techniques, too, as we believe they could potentially provide us with a more generalized model.