**Executive Summary:**

**Introduction:**

Goal:

Request from IU-Entre (i.e. restatement of the issue):

Summary of the steps involved:

1.) Building of a set of websites to include in the sample

2.) Location information on each website

3.) Industry classification of each website

4.) Measuring Changes in Websites

5.) Aggregating by time, place, and industry and look for correlations/prediction value with economic data.

Backcast and moving forward.

**Steps 1 and 2.) Building of a set of websites to include in the sample and acquiring location identifiers**

Tracking changes of company websites first requires the construction of a list of URLs (websites) that the analyst wants to include in the sample. Both the stock and flow of websites are key to include. New websites should be included. Extinct websites should be taken off the list after the closure has been documented.

*Issue: How to construct a list of urls of manageable size?*

As of 2019, there are over 1.8 billion websites, many with multiple urls. We could start from the list of ALL urls, scrape the websites, and attempt to use information from the scraped html to limit the sample by geography and industry. Clearly, the computing power and time required make this method impossible. Instead, we chose to use the services of a private firm that provides a database of all registered urls by time and place of registration. This allows us to complete both step 1 and step 2 simultaneously.

*Backcasting Step:*

We use the Security Trails API to create a database of all urls ever registered in the Southwest Ohio Nielsen Market Area. [Link to the Security Trails Product](https://securitytrails.com/). The API allows us to limit the search based off of the zip code of the owner address registered with the url and listed in the WHOIS database.[[1]](#footnote-1) For example, for this test case, we limited the search to urls ever registered with an address in any zipcode in the Southwest Ohio Nielsen Market Area. This method requires a Professional Subscription Account for which we paid $500 for a total of 20,000 queries.

We connected to the [Security Trails API](https://securitytrails.com/) to create the following two comma separated value (CSV) files:

1. [meta.csv](https://github.com/fmegahed/entrepreneurship-in-southwest-ohio/blob/master/Data/meta.csv), which stores the following meta data related to our API queries:

* **scrape\_date:** Date when we scrapped the WhoIs database using the Security Trails API;
* **post\_code:** The postal code for which websites were being scraped;
* **record\_count:** Number of websites/domain names that are registered for that postal code;
* **max\_page:** Number of page results on Security Trails that we have to iterate through for this post code. Not that if there are 6 pages, the first five pages will each contain 100 records and the sixth will contain up to 100 records.

1. [records.csv](https://github.com/fmegahed/entrepreneurship-in-southwest-ohio/blob/master/Data/records.csv), which stores information in the following variables:

* **scrape\_date:** Date when we scrapped the WhoIs database using the Security Trails API;
* **post\_code:** The postal code for which websites were being scraped;
* **hostName:** domain name for the website;
* **createdDate:** when was the website created;
* **expiresDate:** when does the current registration for the website expire;
* **companyName:** this is computed from the WhoIS information, and thus, it is not always exact and contains a lot of *None*.

The requests were made using *Python Programming Language*, as shown in the code snippet in the technical appendix. Note that we were able to use country as a part of our filter search since we had a **Professional Subscription account**.

The records list now contains all urls ever registered in the Southwest Ohio Nielsen Market Area, the date the url was registered (created), and the date the registration expired (closures). Unfortunately, the company name data is not particularly useful as the vast majority of registrations do not list a company name. For the test case, our query returned 131,723 urls, just for Southwest Ohio.

Note that the list contains ALL urls ever registered: blogs, personal websites, non-existent websites that were never created. Most of the activity is not directly capturing changes in the relevant industries. However, new website registrations, even if not in specific industries, may capture 1.) changes in total economic activity, and 2.) economic activity in the relevant industries to the extent that spillovers exist. We will return to this issue when discussing the prediction model later in step 5.

*How to expand steps 1 and 2 of the backcast to the Midwest and entire nation.*

Creating a url list for any region or the entire country is relatively easy, procedurally. Additional queries to the Security Trails API are all that is needed. The more difficult aspect is the time and monetary cost of doing so. The API limits the number of records that be downloaded in a given month, with prices and limits varying by product tiers. [See the Security Trails pricing list.](https://securitytrails.com/corp/pricing)

Fadel help with this -

1.) Entire Nation – Estimate of number of U.S. hosted websites in 2012 – 505 million.[[2]](#footnote-2) There is likely many more today. Although, this figure could be drastically understated as the most solid number I could find is from 2012. We would know the true number without making the queries on the SecurityTrails API.

2.) Midwest *–* Suppose 20 percent of U.S. websites registered in the Midwest. Then the cost would be somewhere upwards of $10,000 - $20,000 (an estimate of the enterprise API access and/or obtaining a one time data dump from Security Trails API).

*Method for Continuation:*

The list of urls will need to be updated continuously to extend the method in to future dates. API queries will need to be conducted every quarter to get a list of newly registered and newly expired urls. The procedure will be exactly the same as that used for the backcast, except for the addition of a time period limiter, which the API allows for. Queries will need to be done for each zipcode limiting the time period to the previous quarter. The database of active urls will be updated quarterly to reflect the additions and closures.

*Issue: Location of establishment vs. location of headquarters vs. location of registration*

A major issue with using website activity to predict local economic activity is the definition of the location. One could imagine each establishment with its own website, for which local activity could be measured. However, multi-establishment firms use a single firm-level website. The only location information we have is from the registration address from the WhoIs data.

This might be the location of the headquarters, or it might not. Proctor and Gamble registers its urls in Cincinnati, the location of its headquarters. But, P&G could have just as easily registered with an address in Delaware or the address of the web hosting firm. Unfortunately, there is no scalable way to discern between the two. In the end, the website activity we measure will be allocated to locations based on the registration address of the url, however that is chosen.

*Issue: Multiple urls per firm*

Often multiproduct firms will have multiple websites associated with the firm. Using a specific example from our test case, Proctor and Gamble operates many websites: the P&G main corporate url, P&G careers url, and then a long series of product specific urls (e.g. [www.gillette.com](http://www.gillette.com), [www.pampers.com](http://www.pampers.com), etc.). Human judgment is required to decide which of the urls is the homepage for a multi-product firm, along with a knowledge of the product list. In our opinion, any method to limit the sample to firm homepages would be prohibitively expensive. As such, we do not exclude urls from multi-product, or more specifically, multi-url firms.

**Step 3.) Industry classification of each website**

Each website must be classified into an “industry”. Firstly, so that personal websites, blogs, newspapers can be excluded from the sample. Secondly, so that changes in a given industry website activity can be used to predict local economic activity in that same industry. The more detailed the classification the better.

*ISSUE #1: Should an algorithmic solution or a human solution be used?*

Absent an external and universal database of urls and industry codes, there are two possible methods to classify websites into industry categories:

1.) Human judgment – humans read each website and classify into 1.) Commerce (remove personal website, etc.), and then 2.) into NAICS code. The more detailed the NAICS code, the more human expertise and judgment would be required.

2.) Algorithmic approach – The use of some automated procedure to allocate websites to industry, potentially with some human involvement to train and validate the prediction model.

*Solution:* At this point, we do not have the manpower nor time to use the human judgment method. We only came up with this idea recently, and the time-frame for the project does not allow for testing the use of Amazon Turk workers. In any case, given the millions of websites that will need to be classified to roll out the model to national-level, human judgement might be too costly. Instead, our opinion is that an algorithmic method should be used.

This type of problem requires the use of topic modelling methods and seems to be an open research question. A topic modelling method would consist of scraping each website and using the text from the website to allocate websites industries. The idea is that some words, and combinations of words, are used more often by some industry websites than others. These groups of words could be used to predict the industry of a given website. The accuracy of the prediction would increase with the non-overlap of words across industries.

To create an industry classification topic model from scratch is outside the scope of this project. Our solution is to use an off the shelf algorithmic classification system provided for a fee, the online commercial platform [Web Shrinker API](https://docs.webshrinker.com/v3/website-category-api.html#website-category-api-introduction). Web Shrinker has created a proprietary topic modelling classification method, and we decided to start with a method that we didn’t have to create ourselves. Among a number of providers, we believe the Web Shrinker API is the best choice, because:

1. The Web Shrinker Category API gets and downloads the content of a given *URL* prior to categorizing it to one of [42 categories](https://docs.webshrinker.com/v3/web-shrinker-categories.html#categories).
2. Provided more accurate results when compared to the [Website Categorization API from WhoisEXMLAPI](https://website-categorization-api.whoisxmlapi.com/api/) in our initial experiments.
3. The API is relatively cheap; $20 allows for 30,000 URL/domain name queries

In our test case for Southwest Ohio, of the 1,383 urls with Wayback Machine data and culled from the list created in step 1 and step 2, we were able to categorize over 1,100 domains using the Web Shrinker API (see [webshrinker.csv](https://github.com/fmegahed/entrepreneurship-in-southwest-ohio/blob/master/Data/webshrinker.csv) for details). The remaining URLs, which were not categorized, were primarily due to the URL expiring, which means that the page cannot be opened and downloaded for further analysis. Note that we have tried using the web archive link for categorization in the Web Shrinker API (i.e. by using the url from the [wayback\_changes.csv](https://github.com/fmegahed/entrepreneurship-in-southwest-ohio/blob/master/Data/wayback_changes.csv)) as input to the Python code below, however the results were not satisfactory. We believe that this is due to the structure of the Wayback Machine HTML pages. To examine the errors obtained from the Web Shrinker API, please see [webshrinker\_error.csv](https://github.com/fmegahed/entrepreneurship-in-southwest-ohio/blob/master/Data/webshrinker_error.csv).

*ISSUE #2: Which industry classification system to use?*

The ideal industry classification structure would be the one in which the local economic activity that we want to predict is classified in. Thus, either NAICS, SIC, or GICS. To my knowledge, the government statistical agencies are using NAICS as their primary industry structure moving forward. Thus, NAICS we would be the clear choice. A related issue is what level of detail to use. However, this is actually more of a prediction problem in the sense of whether the prediction models before better or worse for aggregate or more disaggregated NAICS industries.

Currently, however, we are forced to use the categories provided by the Web Shrinker API. These categories provide a surprising amount of detail, but not enough to track “high-tech” firms or “gazelles”. The table below presents a breakdown of the number of urls registered in Southwest Ohio that fall into each of the detailed Web Shrinker categories.

*Suggestions for future research:*

1. Explore using external databases to categorize as many urls as possible. For instance, D&B lists both the web address and a NAICS code. We were unable to determine the feasibility of this method. Miami University’s library license for D&B does not allow this type of query. Other databases might also exist to perform the same function.
2. A separate research project to build on the work of the proprietary APIs by using topic modelling to create a NAICS specific classification algorithm.
3. Explore using Amazon Turk to cost-effectively scale up the human judgement method.
4. Use a mixture of methods in order of increasing complexity or cost. (i.e. start with D&B or other database. Apply a topic-modelling prediction algorithm to remaining uncategorized URLs. Use Amazon Turk for remaining uncategorized URLs not capture by the predictive modeling.

**Step 4.) Measuring Changes in Websites**

Measuring the number and types of changes in website activity is also an area of open research. We propose using three measures of changes in activity: 1.) the number of new url registrations, 2.) the change in the stock of url registrations, 3.) and a count of any change in files based on the Wayback Machine definition.

*Measures 1 and 2: Changes in counts of registrations*

The data requirements for measures of url registrations comes directly from the Security Trails API data discussed in Steps 1 and 2. The number of new registrations in a given period and geographic area can be directly constructed from the createdDate variable in [records.csv](https://github.com/fmegahed/entrepreneurship-in-southwest-ohio/blob/master/Data/records.csv) that is associated with the initial registration date of a URL. The idea is that new registrations capture growth in local economic activity. The second measure would be the change in the stock of currently registered URLs: the running sum of new registrations minus expired registrations. The idea here is that the ups and downs of newly created websites captures something additional about local economic activity over just counting the number of new registrations. Sectoral counts and changes in registrations can be constructed by applying the industry categorization methods discussed in Step 4. Measures constructed for monthly, quarterly, or annual time periods can easily be constructed using the exact creation and expiration dates contained in the Security Trails API data.

*Measure 3: Substantive changes in website content*

The original impetus for this project was to attempt to apply a method to categorize website changes from the existing literature. Specifically, the suggested method was to implement that used in an article by Blazquez, Domenech, and Debon, “Do Corporate Websites’ Changes reflect firms’ survival?”, published in the *Online Information Review.* In this paper, the authors construct a list of 720 Spanish companies and query the homepage of each firm using the Wayback Machine Internet Archive over the period 2008 to 2014. To capture changes, the authors created a variable called *Web\_status*, which could take five different values:

(1) the website is down. This includes the websites that do not work (e.g. HTTP Error 404 Not Found) or whose domain name has expired or is for sale.

(2)  the website remains unchanged. This includes the cases in which the website remains exactly the same as its previous year’s version.

(3)  the website has undergone minor changes. These changes include the removal or addition of sections, options, pictures and contents.

(4)  the website has undergone major changes. These changes refer to a new website design, so that it completely differs from to the previous year’s version; this may imply a change in the technology used to build the website.

(5) the website has not been captured by the Wayback Machine. These cases were processed as missing data and were removed from the final sample as it was impossible to determine the website status.

We attempted to follow this established procedure from the literature, which we can break up into two separate steps: 1.) Wayback Machine data extraction, and 2.) Measuring website changes.

*Step 1.) Wayback Machine Data Extraction*

The Wayback Machine API allows for user generated algorithms to conduct large volumes of queries. [Information on the API can be found here](https://archive.org/help/wayback_api.php). A user generated set of extensions to the API, called the [waybackpack](https://github.com/jsvine/waybackpack), allows for easier queries to construct a dataset and download web pages from the archive. For each of the 132 thousand unique URLs registered in the Southwest Ohio region, we search the Wayback Machine to see if any scrapes were completed for that URL, the dates of each scrape, and whether there were any changes. The python code used can be found in the 02\_clean\_scrape.py file. Of the 132 thousand unique URLs, the Wayback Machine archive contains at least one scrape for 1,383 websites, which are contained in the wayback\_changes.csv file. *We currently have not downloaded the actual archived webpages, just the metadata from the archive.* We chose this because there is not yet a method to *use* the html directly, which we discuss below.

*Step 2.) Measuring Website Changes*

Given the large number of websites in the sample, 1,383 alone just in Southwest Ohio, and the large number of changes, an automated solution to determining website changes is required. Expanding the method to more geographic areas makes the necessity for an algorithmic solution even more clear. We read the Blazquez, Domenech, and Debon paper in detail, searching for the algorithm used to determine whether a firm’s homepage underwent major or minor changes – corresponding to codes 3 and 4 in the author’s terminology. Unfortunately, the article did not elaborate on how the author’s differentiated between no changes, minor or major changes. Personal communication with Dr. Blazquez returned the following:

*“… to categorize a website into code 1, 2 or 3, we did a visual inspection of the homepage of each website:*

* *If the homepage in year t looked exactly the same to year t-1, then we categorized the website with code 1.*
* *If we observed changes between year t and year t-1, then we used code 2 or 3:*
  + *Code 2 was used when minor changes were observed. Examples of these minor changes include: addition or removal of sections in the menu bar; addition or removal of options (such as including a form to contact the firm); addition or removal of pictures; and addition or removal of contents (such as text paragraphs).*
  + *Code 3 was used when major changes were observed. As major changes, we considered a completely different design of the website (instead of adding or removing options, contents, and so on, following the same website design, the website was completely redesigned).”*

Thus, the author’s of the paper used visual inspection and human judgement to categorize the existence and extent of any changes to a firm’s homepage. In our opinion, this method is not appropriate to scale up.

Instead, we use an query filter in the Wayback Machine API that allows the user to restrict information to “unique” files that were archived for a given website. Unique files were defined so that only one copy of the website’s URL will be scraped (as an initial simplification, we did not want to scrape multiple pages for the same company – this only utilized the Company name field from the WhoIs data). The python code creates a .csv file that contains the date of each “unique” scrape of a website by the Wayback Machine. We interpret the “unique” value to mean a “change” in website activity.

*Issue: Inconsistent crawling by the Wayback Machine spiders.*

The Wayback Machine does not scrape every webpage on a specific schedule. It uses a set of crawlers that start from a fixed list of URLs, and then follows the hyperlinks on a each page to a given set depth. This means that some websites are never archived, some websites are archived only intermittently, whereas others are archived every quarter. This can be a problem, in that as the analyst, we only observe the scrapes of website that did occur, and if any changes occurred relative to the previous scrape. The problem can be thought of as measurement error in some sense. When a website is not scrape for over a year, then we don’t observe whether or not there were any true changes in the website, and this would be recorded as a “No change” for that year, even if changes did occur. When a true change is captured by the Wayback Machine, the analyst does not observe the actual date of the change, only that a change occurred between the two scrape dates. For example, say a website was archived on July 1, 2017 and March 1, 2018 and a change is recorded. For the purposes of predicting local economic activity, should the change be recorded as occurring in 2017 or 2018? If predicting economic activity on a quarterly interval, this issue becomes thornier. For the lack of better solution, we record the change as occurring on the second scrape date. In the example above, a website change would be allocated to 2018. *When constructing a prediction model in Step 5.), we limit the interval to be annual changes because this might limit measurement error in the predictor variable.*

*Extending the Backcast Step:*

Extending this step to the entire nation is relatively straightforward. Once a list of URLs is constructed in Step 1, they python code will make a series of calls to the Wayback Machine API and create csv file with the same information.

*Method for Continuation:*

Putting the method into production would obviate the need for the Wayback Machine, and actually improve the measurement of website changes. The analyst could scrape homepages directly on a fixed schedule (monthly or quarterly) and record a more accurate date of changes made. The drawback of this method is the large amount of computational time and memory required to scrape hundreds of thousands (millions maybe?) of websites every month or quarter. Scripts could be written to automate this step, but without knowing the total number of websites that need to be followed it is difficult to estimate the time and cost. Moreover, a method would need to be develop to approximate how the Wayback Machine determines a “unique” scrape.

*Suggestions for future research:*

1. The biggest open research question is there additional algorithmic ways to determine useful substantive changes in homepages. The use of java, flash, measures of complexity, etc. Determining whether these have predictive value key.
2. The number of websites that need to be scraped each month can be usefully restricted to those that are for commercial enterprises. Blogs and personal websites, or government websites, do not need to be included. Once the list of URLs are collected in Step 1, the immediate next step should be to run them through the WebShrinker API to determine whether it is a commercial enterprise, or personal, etc. From our limited experience exploring the data for Southwest Ohio, the vast majority of registered URLs are not for commercial enterprises. Moreover, this step needs to be completed only once for a specific URL. Categorizing all URLs created between 1995 and 2019 will take some time, but categorizing the newly registered URLs each month would be less costly and time consuming.

**Step 5.) Aggregating by time, place, and industry and look for correlations/prediction value with economic data.**

Purpose is to determine whether and by how much additional information from website activity can improve predictions of local economic activity above using a simple lag of the dependent variable.

*Outcome Data*

*Predictive Model*

We choose to start with a simple linear OLS model. More complicated models seem overkill with, effectively, only 17 observations. The base model includes a constant and the lag of the dependent variable to capture the fact of substantial serial correlation.

Model 1:

Model 2:

(Add change in stock of website registrations)

Model 3:

(Add in total number of website changes)

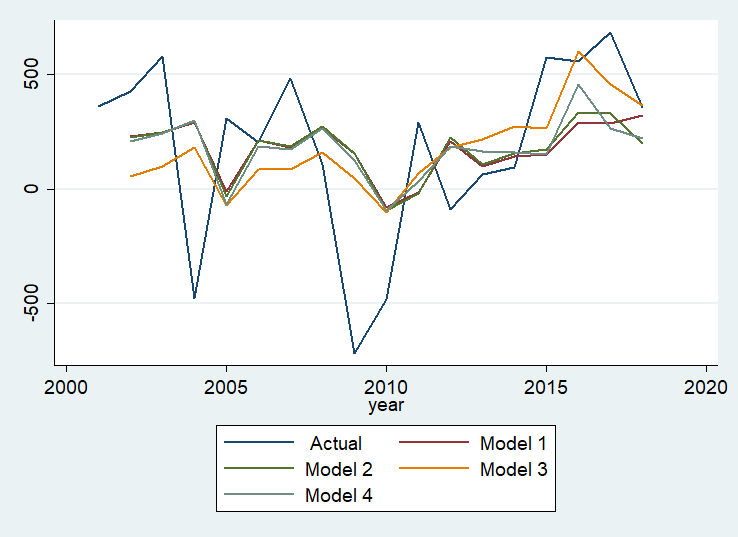
Model 4:

(Replace total number of changes with change in number of website changes)

We will select the model with the highest R-squared. The additional information from website changes will be useful to the extent that it increases the R-squared over the lag of the dependent variable (Model 1). We include a graph of model performance relative to actual data as well.

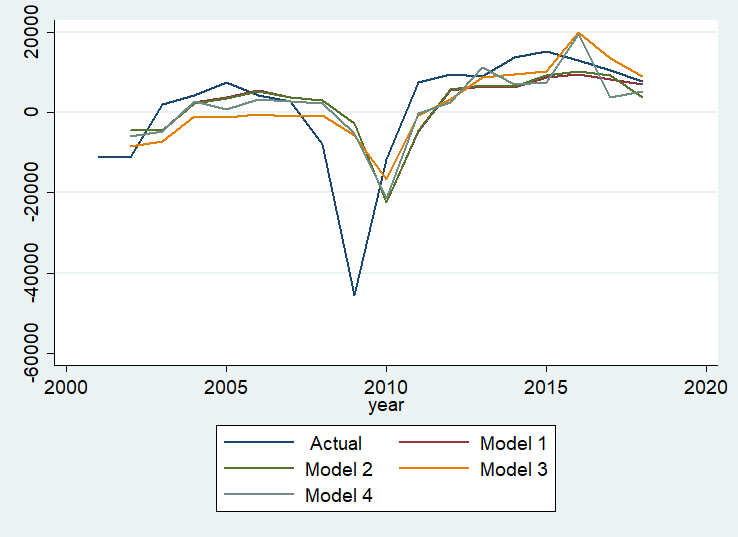
*Establishments - Total Covered Private Industries*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Model 1 | *Model 2* | *Model 3* | *Model 4* |
| Lagged Dependent Variable | 0.287  (0.247) | 0.311  (0.264) | *0.219*  *(0.0268)* | *0314*  *(0.273)* |
| Change in Stock of Registrations |  | 0.079  (0.233) | *0.179*  *(0.241)* | *0.142*  *(0.285)* |
| Total number of website changes |  |  | *0.063*  *(0.049)* |  |
| Change in number of website changes |  |  |  | *0.034*  *(0.082)* |
|  |  |  |  |  |
| R-squared | 0.0825 | 0.0900 | *0.1923* | *0.1016* |
|  |  |  |  |  |
| Mean Dep Var. |  |  |  |  |



*Employment - Total Covered Private Industries*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Model 1 | *Model 2* | *Model 3* | *Model 4* |
| Lagged Dependent Variable |  |  |  |  |
| Change in Stock of Registrations |  |  |  |  |
| Total number of website changes |  |  |  |  |
| Change in number of website changes |  |  |  |  |
|  |  |  |  |  |
| R-squared |  |  |  |  |
|  |  |  |  |  |
| Mean Dep Var. |  |  |  |  |



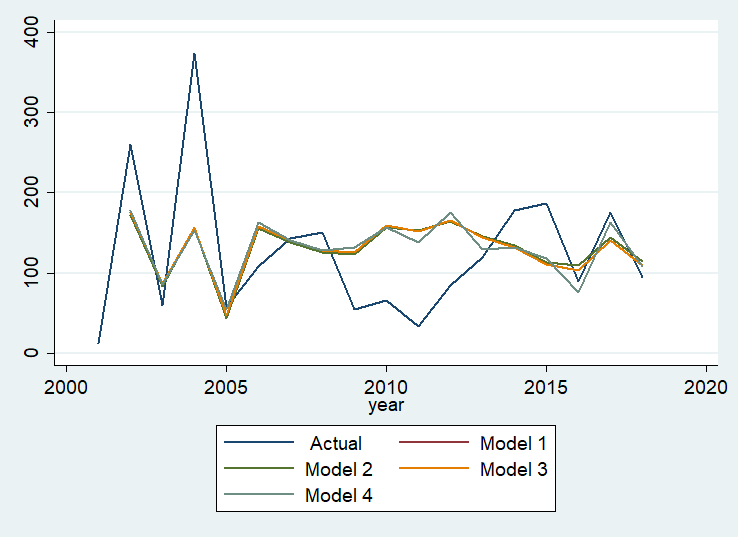
Relevant Industries











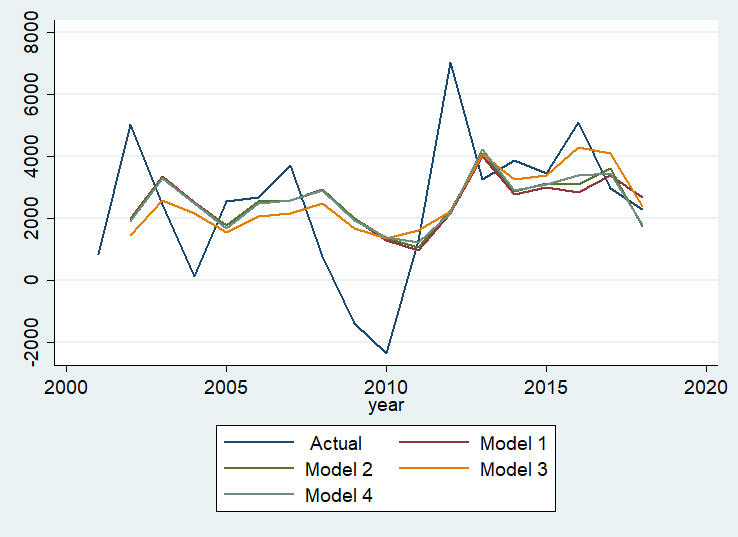
Employment in relevant industries











1. **WHOIS** (pronounced as the phrase "who is") is a query and response protocol that is widely used for querying databases that store the registered users or assignees of an Internet resource, such as a domain name, an IP address block or an autonomous system, but is also used for a wider range of other information. [↑](#footnote-ref-1)
2. <https://www.cia.gov/library/publications/the-world-factbook/rankorder/2184rank.html> [↑](#footnote-ref-2)