CIS3200 Term Project Tutorial

Authors: Alan Liu

Instructor: Jongwook Woo

Date: 05/19/2019

Lab Tutorial

Alan Liu (aliu38@calstatela.edu)

05/19/2019

Predicting BMI from Food Availability

Objectives

This tutorial will teach you how to create the best-performing BMI prediction model of the ones tested in the full project. In this lab, you will learn how to:

- Get data manually from USDA ERS website
- Load data into Microsoft Azure ML Studio
- Transform data
- Train model
- Load data into Elasticsearch
- Visualization

Platform Specifications

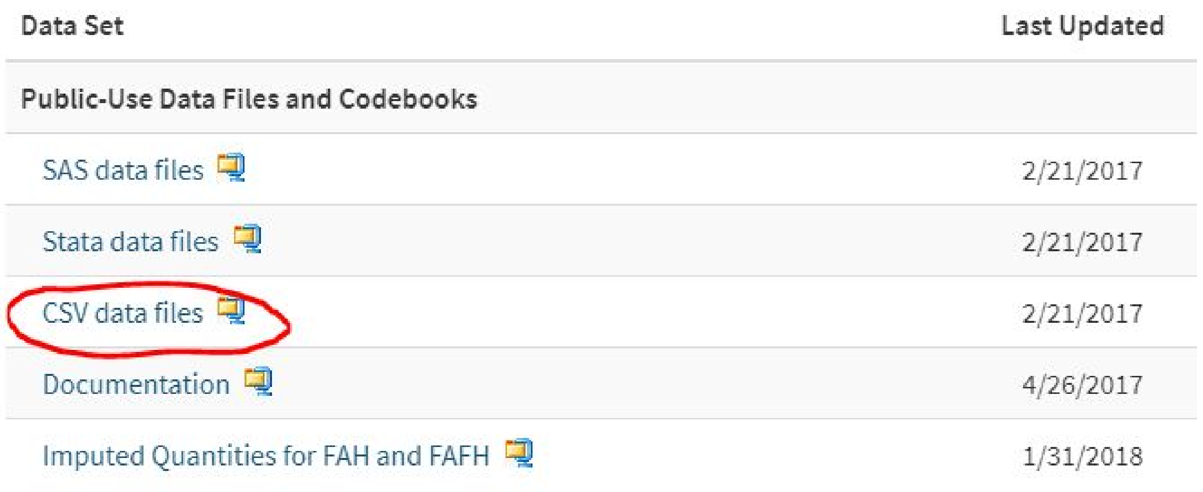
- Microsoft Azure Machine Learning Studio (Free tier)
    - Maximum Nodes per Experiment: 100
    - Maximum Experiment Duration: 1 hour
    - Node Execution Performance: Single Node
    - Maximum Storage Space: 10GB
- Elasticsearch Service (Deployment consisting of co-located master/data/ingest node, kibana, ml)
    - Instance/Configuration: 'gcp.data.highio.1'
        - CPU Cores: 12 cores
        - Memory: 78GB (1GB provisioned)
        - Storage: 3000GB (30GB provisioned)
    - Instance/Configuration: 'gcp.kibana.1'
        - CPU Cores: 22 cores
        - Memory: 140GB (1GB provisioned)
        - Storage: N/A
    - Instance/Configuration: 'gcp.ml.1'
        - CPU Cores: 64 cores
        - Memory: 270GB (1GB provisioned)
        - Storage: N/A

Step 1: Get data manually from USDA ERS website

The United States Department of Agriculture's Economic Research Service conducted a Food Acquisition and Purchase Survey across the nation and the participant's responses are available for download at the website:
https://www.ers.usda.gov/data-products/foodaps-national-household-food-acquisition-and-purchase-survey/

You can optionally also download the Documentation at the same link, which contains the codebook files explaining the columns and their meanings in each dataset.

1. Visit website and download csv dataset from link.



Data Set	Last Updated
Public-Use Data Files and Codebooks	
SAS data files	2/21/2017
Stata data files	2/21/2017
CSV data files	2/21/2017
Documentation	4/26/2017
Imputed Quantities for FAH and FAFH	1/31/2018

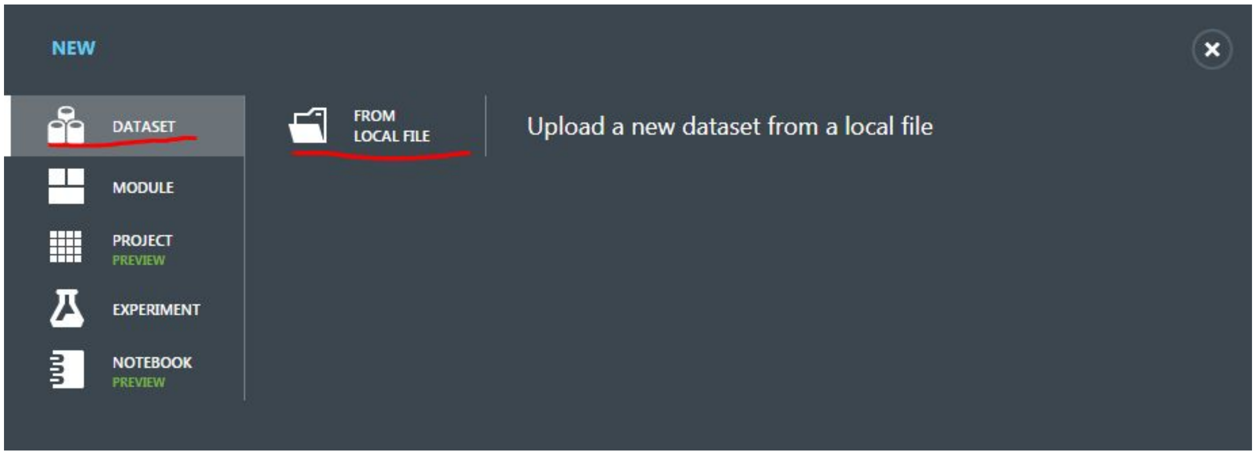
2. Unzip the files into a folder.

Step 2: Load data into Microsoft Azure ML Studio

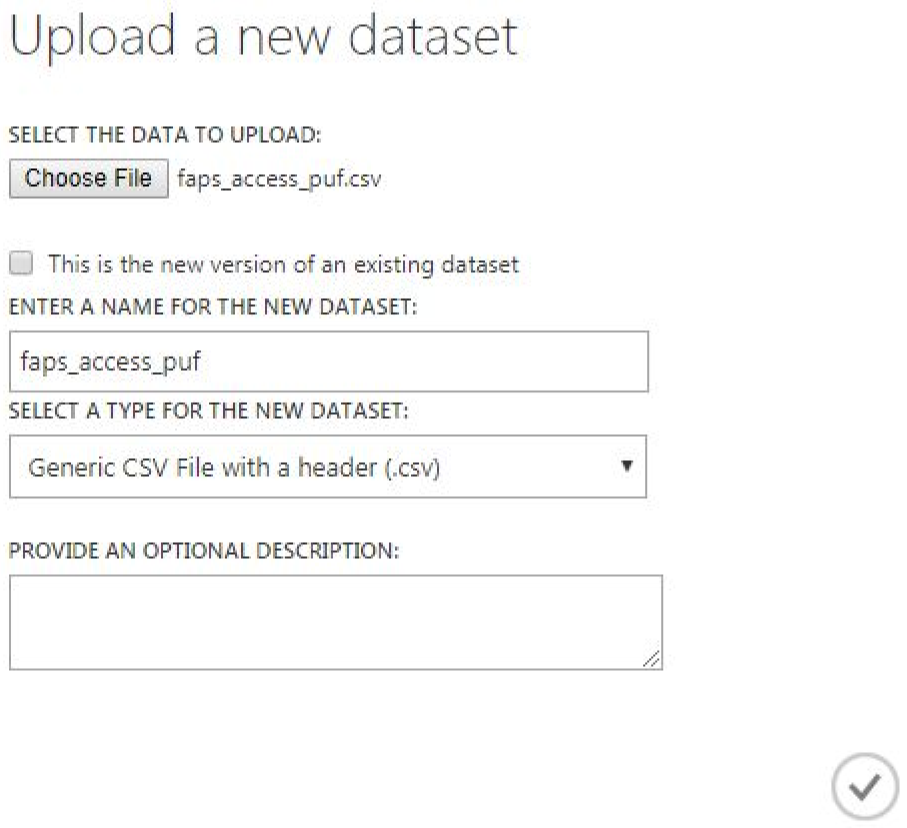
We need to upload the data into our Azure ML as a dataset before we can use them as modules.

1. Login to your Azure ML Studio account using your school email: https://studio.azureml.net

2. On the bottom left corner, click **New** and click on **Dataset** on the left sidebar that pops up, and upload your csv files by clicking **From Local File**:



3. Navigate to folder where you extracted the dataset, and select **faps_access_puf.csv** and enter the name for the data set "**faps_access_puf**" and click the checkmark.


Upload a new dataset

SELECT THE DATA TO UPLOAD:
Choose File faps_access_puf.csv

☐ This is the new version of an existing dataset

ENTER A NAME FOR THE NEW DATASET:
faps_access_puf

SELECT A TYPE FOR THE NEW DATASET:
Generic CSV File with a header (.csv)

PROVIDE AN OPTIONAL DESCRIPTION:

4. Repeat above step 3 for the csv files:
faps_fafhevent_puf.csv
faps_fafhitem_puf.csv
faps_individual_puf.csv

Step 3: Transforming the data

Before using the data for training the model, we have to clean the data and make sure we only the the columns we need. We also have to make sure the metadata for the columns are correct.

1. **Make a new experiment** by clicking on the **New** button and **Blank Experiment**. Name it something you will remember.

2. From your experiment items sidebar, drag and drop the **faps_access_puf** into the experiment.

3. Drag and drop a **Select Columns in Dataset** module into the project.

4. Connect the output port from the **faps_access_puf** module to the input port of the **Select Columns in Dataset** module.

5. Click on the **Select Columns in Dataset** module and in the properites panel, click **Launch Column Selector**.

6. Select the columns **By Name** from the available columns:

```
hhnum,snap1,snap2,snap3,snap4,snap5,snap6,snap8,
ss1,ss2,ss3,ss4,ss5,ss6,ss7,ss8,
sm1,sm2,sm3,sm4,sm5,sm6,sm7,sm8,
co1,co2,co3,co4,co5,co6,co7,co8,
cs1,cs2,cs3,cs4,cs5,cs6,cs7,cs8,
mlg1,mlg2,mlg3,mlg4,mlg5,mlg6,mlg7,mlg8,
dist_co,dist_cs,dist_mlg,
ff1,ff2,ff3,ff4,ff5,ff6,ff7,ff8,
nonff1,nonff2,nonff3,nonff4,nonff5,nonff6,nonff7,nonff8,
nearff_dist,nearnonff_dist
```

7. Drag and drop your **faps_individual_puf** dataset into the project, and another **Select Columns in Dataset** module, and connect them together like in step 4.

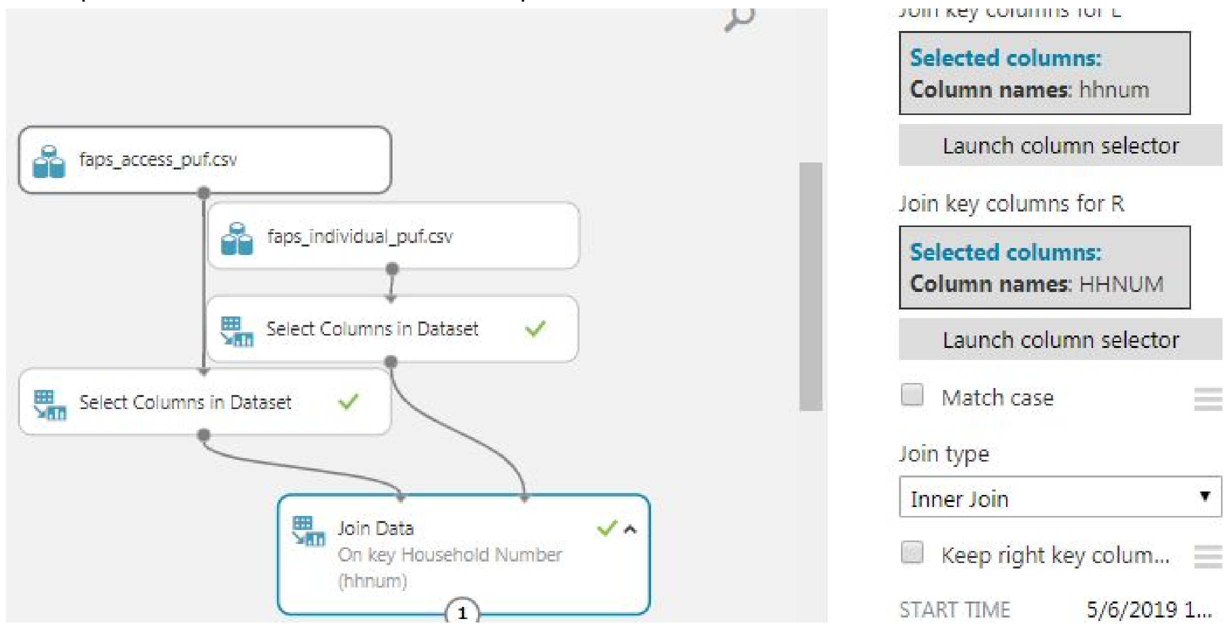
8. Click on the new **Select Columns in Dataset** module and **Launch Column Selector**.

9. Select the columns **By Name** from the available columns:
`HHNUM,PNUM,NDINNERSOUT,HEALTHSTATUS,BMICAT,BMI`

10. Add a **Join Data** module to the experiment, and connect the output ports of both **Select Columns in Dataset** modules to the input ports of **Join Data**. Make sure the one for **faps_access_puf** is on the **left input** and the one for **faps_individual_puf** is on the **right input**.

11. In the **Join Data** property panel, select **hhnum** for the **Join key columns for L**, and select **HHNUM** for the **Join key columns for R**. Make sure **Match case** and **Keep right key column** is **unchecked**. Set the **Join type** to **Inner Join**.

The experiment should look like this at this point:



Join key columns for L

Selected columns:
Column names: hhnum

Join key columns for R

Selected columns:
Column names: HHNUM

☐ Match case

Join type

Inner Join

☐ Keep right key colum...

START TIME 5/6/2019 1...

12. Add the module **Execute Python Script** to the experiment. Connect the **Join Data** output to this module's input and use this code from Github: https://github.com/shozonu/usda-foodaps-bmi-model/blob/master/py/prep1.py

```
import pandas as pd

def azureml_main(dataframe1 = None, dataframe2 = None):
    # For each row
    for index, row in dataframe1.iterrows():
        hnum = dataframe1['hhnum'].iat[index]
        pnum = dataframe1['PNUM'].iat[index]

        # Append create unique pid by appending pnum to end of hhnum
        pid = int("{}{}".format(hnum, pnum), 10)
        dataframe1["hhnum"].iat[index] = pid

    # Rename columns
    dataframe1.rename(columns={"hhnum" : "pid"}, inplace = True)
    dataframe1.rename(columns={"NDINNERSOUT" : "ndinnersout"}, inplace = True)
    dataframe1.rename(columns={"HEALTHSTATUS" : "healthstatus"}, inplace = True)
    dataframe1.rename(columns={"BMICAT" : "bmicat"}, inplace = True)
    dataframe1.rename(columns={"BMI" : "bmi"}, inplace = True)

    # Drop pnum column
    dataframe1.drop(["PNUM"], axis=1, inplace = True)

    return dataframe1
```

This Python script will create a unique id from the household number and person number field. It will also rename the upper case columns, the new id column, and remove the extra pnum column.

13. Add a **Edit Metadata** module. Select the **pid** column. Leave the **Data type** and **Categorical** dropdown menu as **unchanged**. Change the **fields** dropdown to **Clear feature**.

14. Add a **Apply SQL Transformation** module. Connect the previous output to this input. We need to remove rows that contain the error flags specified in the official data codebook, using the SQL script from Github: https://github.com/shozonu/usda-foodaps-bmi-model/blob/master/sql/tran1.sql

```
-- Filter out rows with these columns
-- that contain error flags.
SELECT * FROM t1
WHERE bmi > 0
AND bmi != 'V'
AND bmicat != 'V'
AND bmicat != 'E'
AND healthstatus != 'D'
AND healthstatus != 'R'
AND ndinnersout != 'D'
AND ndinnersout != 'R';
```

15. Connect the previous output to the input of a new **Edit Metadata** module. Select the **bmi** column, and **uncheck allow duplicates**. Make the data type **Floating point**. Set it to **Make non-categorical**. Set the fields to **Label**.

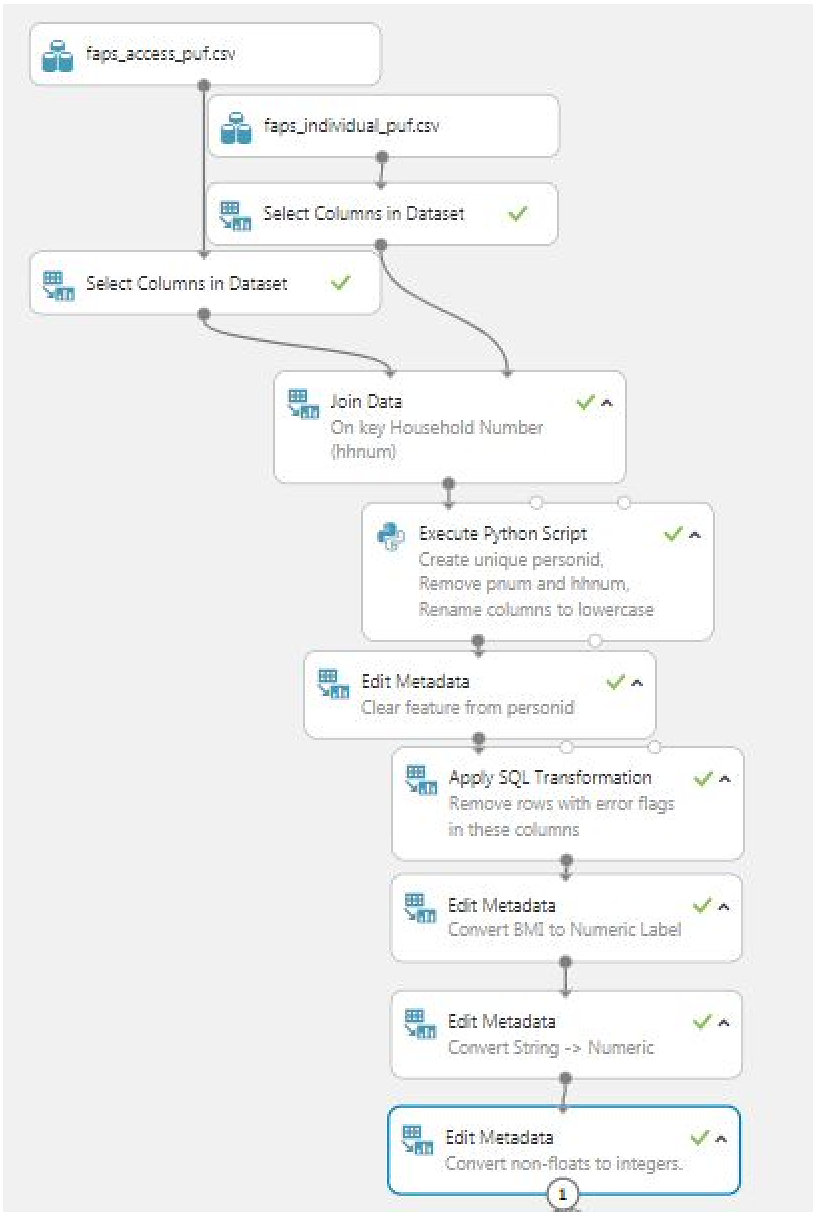
16. Connect the previous output to the input of a new **Edit Metadata** module. Select the columns:
`bmicat,ndinnersout,healthstatus`
Uncheck allow duplicates. Make the data type **Integer**. Set it to **Make categorical**. Set the fields to **Features**.

17. Connect the previous output to the input of a new **Edit Metadata** module. Select the columns:

```
snap1,snap2,snap3,snap4,snap5,snap6,snap8,
ss1,ss2,ss3,ss4,ss5,ss6,ss7,ss8,
sm1,sm2,sm3,sm4,sm5,sm6,sm7,sm8,
co1,co2,co3,co4,co5,co6,co7,co8,
cs1,cs2,cs3,cs4,cs5,cs6,cs7,cs8,
mlg1,mlg2,mlg3,mlg4,mlg5,mlg6,mlg7,mlg8,
ff1,ff2,ff3,ff4,ff5,ff6,ff7,ff8,
nonff1,nonff2,nonff3,nonff4,nonff5,nonff6,nonff7,nonff8
```

Uncheck allow duplicates. Make the data type **Integer**. Set it to **Make non-categorical**. Set the fields to **Unchanged**.

The experiment should look like this at this point:


faps_access_puf.csv
faps_individual_puf.csv
Select Columns in Dataset
Select Columns in Dataset
Join Data
On key Household Number (hhnum)
Execute Python Script
Create unique personid, Remove pnum and hhnum, Rename columns to lowercase
Edit Metadata
Clear feature from personid
Apply SQL Transformation
Remove rows with error flags in these columns
Edit Metadata
Convert BMI to Numeric Label
Edit Metadata
Convert String -> Numeric
Edit Metadata
Convert non-floats to integers.
1

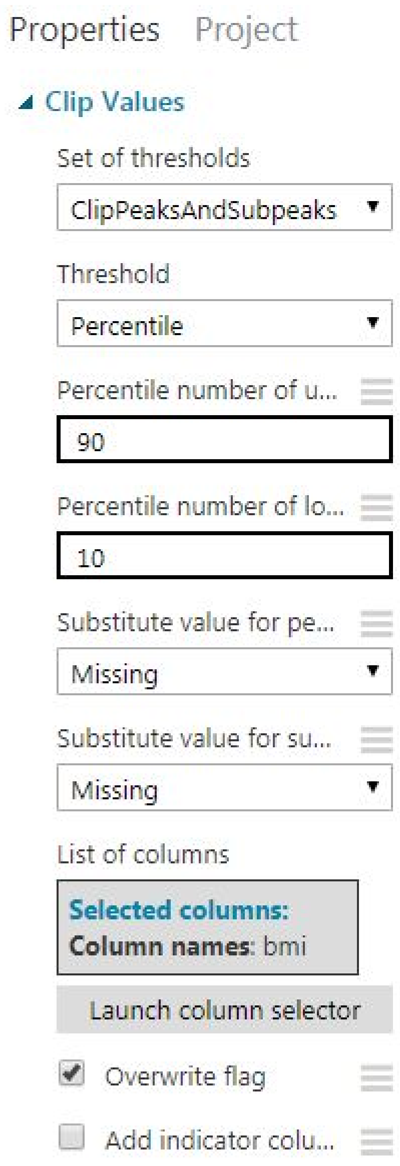
Step 4: Train model

Now that we're done cleaning the data, we can move onto the setting up the training portion for predicting the BMI.

1. First we want to remove the outliers so that we can improve overall model performance. Add a Clip Values module to the experiment.

2. In the properties panel, set the **Set of thresholds** to **ClipPeaksAndSubpeaks**. Set **Threshold** to **Percentile**. Set the upper percentile to **90** and the lower percentile to **10**. Set **substitute value for peak** to **Missing**. Set **substitute value for subpeak** to **Missing**. Lanch the column selector and select **bmi**. **Check** the **Overwrite flag** box and **uncheck** the **Add Indicator column** box.

It should look like this:



Set of thresholds: ClipPeaksAndSubpeaks
Threshold: Percentile
Percentile number of u...: 90
Percentile number of lo...: 10
Substitute value for pe...: Missing
Substitute value for su...: Missing
List of columns: **Selected columns:** **Column names:** bmi
[✓] Overwrite flag
[ ] Add indicator colu...

3. Add a **Clean Missing Data** module to the experiment and connect the last output to this input. Make sure that **All columns** are selected. Leave the minimum and maximum value ratio at 0 and 1. Make sure cleaning mode is set to **Remove entire row**.

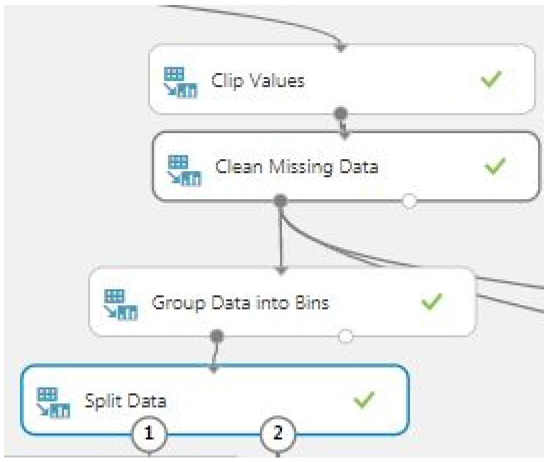
4. Connect the last module to a new **Group Data into Bins** module. In the properies panel, set the **Binning mode** to **Quantiles**. Set **Number of bins** to **10**. Set **Quantile normalization** to **percent**. Select the following columns:

```
snap1,snap2,snap3,snap4,snap5,snap6,snap8,
ss1,ss2,ss3,ss4,ss5,ss6,ss7,ss8,
sm1,sm2,sm3,sm4,sm5,sm6,sm7,sm8,
co1,co2,co3,co4,co5,co6,co7,co8,
cs1,cs2,cs3,cs4,cs5,cs6,cs7,cs8,
mlg1,mlg2,mlg3,mlg4,mlg5,mlg6,mlg7,mlg8,
ff1,ff2,ff3,ff4,ff5,ff6,ff7,ff8,
nonff1,nonff2,nonff3,nonff4,nonff5,nonff6,nonff7,nonff8
```

Set the **Output mode** to **Inplace**, and make sure the **Tag columns as categorical** is unchecked.

5. Connect the previous output to the input of a new **Split Data** module. Set the **Splitting mode** to **Split Rows**. Set the **Fraction of rows in the first output** to **0.66**. Make sure the **Randomized split** box is **checked**. Set the **Random seed** to **4321**. Set the **Stratified split** to **False**.

Your experiment should look like this at this point:

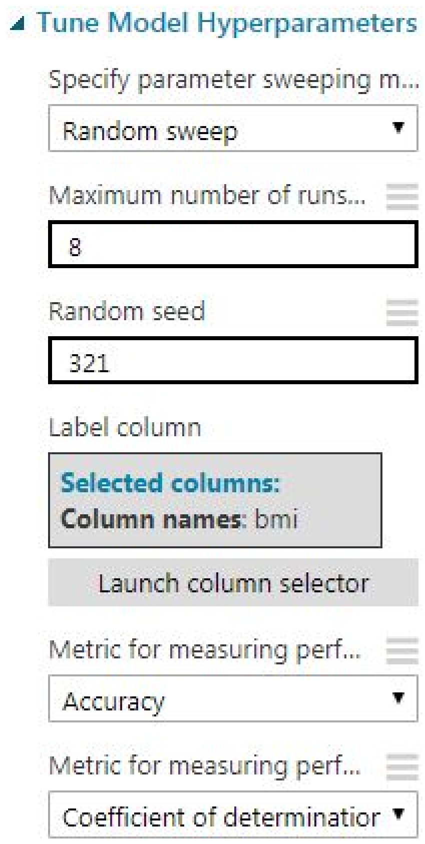


6. Add a **Bayesian Linear Regression** module.

7. Add a **Tune Model Hyperparameters** module and connect the **Bayesian Linear Regression** to the untrained model input.

8. In the **Tune Model Hyperparameters** properties panel, set it to **Random sweep** with a maximum number of runs of **8**. Set the **Random seed** to **321**. Select the **bmi** column. Set the **Metric for measuring performance for regression** to **Coefficient of determination**.

It should look like this:



Tune Model Hyperparameters

Specify parameter sweeping m...

Random sweep

Maximum number of runs...

8

Random seed

321

Label column

Selected columns:
Column names: bmi

Metric for measuring perf...

Accuracy

Metric for measuring perf...

Coefficient of determinatior

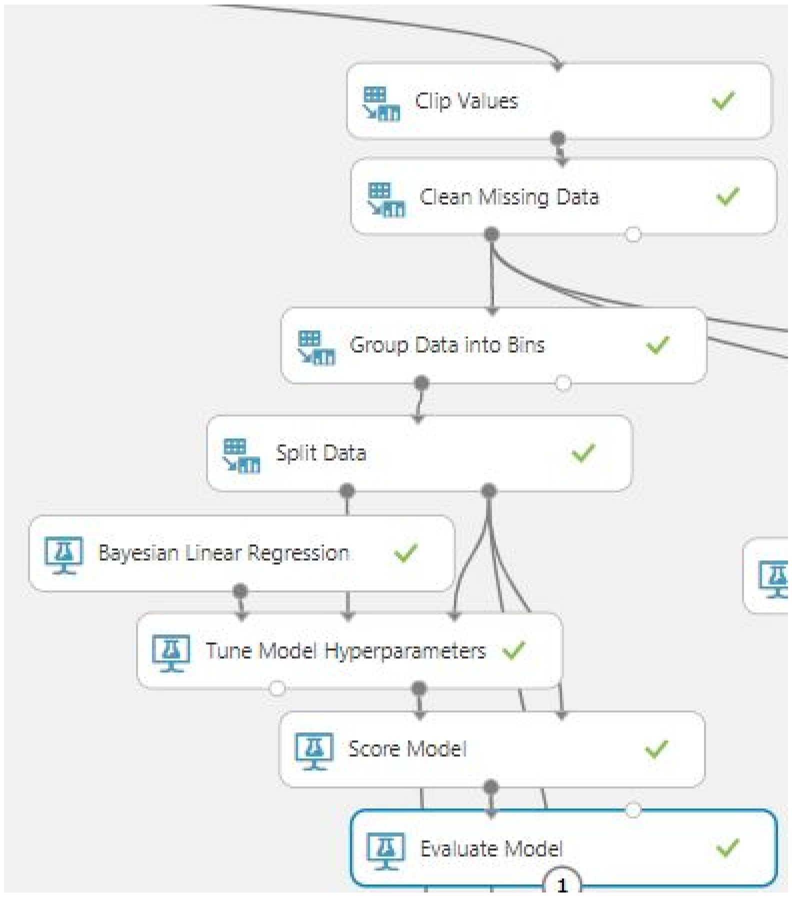
9. Connect the first output of the **Split Data** to the second input of **Tune Model Hyperparameters**.

10. Connect the second output of **Tune Model Hyperparameters** to the first input of a new **Score Model** module.

11. Connect the third output of the **Split Data** module to the third input of **Tune Model Hyperparameters** and the second input of **Score Model**.

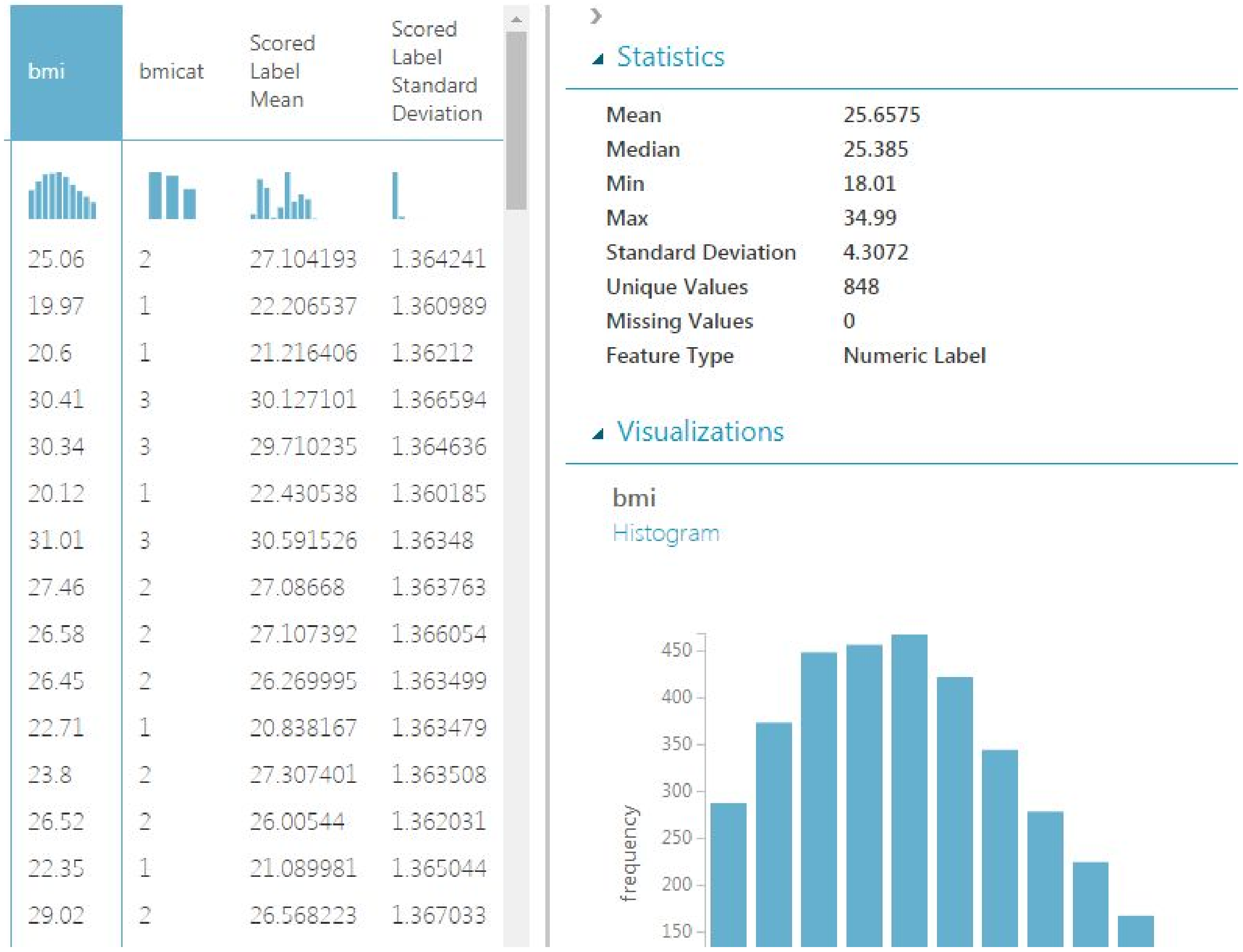
12. Connect the output of **Score Model** to the first input of a new **Evaluate Model** module.

The training section of the experiment should look like this:



13. Save and run the experiment.

14. Visualize the output of **Score Model** to see the predicted BMI.



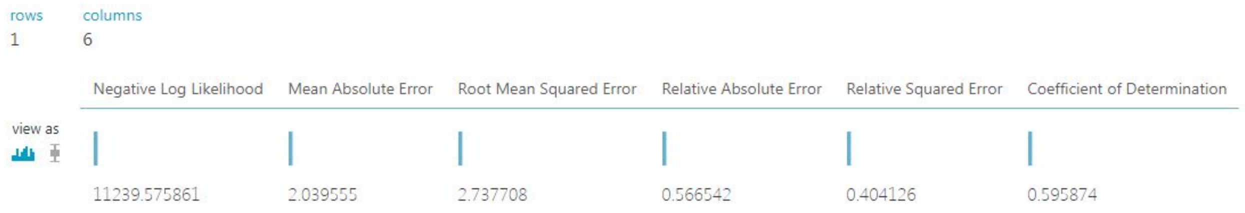
bmi	bmicat	Scored Label Mean	Scored Label Standard Deviation
25.06	2	27.104193	1.364241
19.97	1	22.206537	1.360989
20.6	1	21.216406	1.36212
30.41	3	30.127101	1.366594
30.34	3	29.710235	1.364636
20.12	1	22.430538	1.360185
31.01	3	30.591526	1.36348
27.46	2	27.08668	1.363763
26.58	2	27.107392	1.366054
26.45	2	26.269995	1.363499
22.71	1	20.838167	1.363479
23.8	2	27.307401	1.363508
26.52	2	26.00544	1.362031
22.35	1	21.089981	1.365044
29.02	2	26.568223	1.367033

Statistics

Mean	25.6575
Median	25.385
Min	18.01
Max	34.99
Standard Deviation	4.3072
Unique Values	848
Missing Values	0
Feature Type	Numeric Label

We can see that the predicted BMI rows are fairly close to the true BMI most of the time.

15. Visualize the output of **Evaluate Model** to see the performance of the model.



rows 1 columns 6

Negative Log Likelihood	Mean Absolute Error	Root Mean Squared Error	Relative Absolute Error	Relative Squared Error	Coefficient of Determination
11239.575861	2.039555	2.737708	0.566542	0.404126	0.595874

Step 3: Visualization

To better visualize the difference between the true BMI and the predicted BMI, we will export the scored dataset to Elasticsearch and visualize it in a graph.

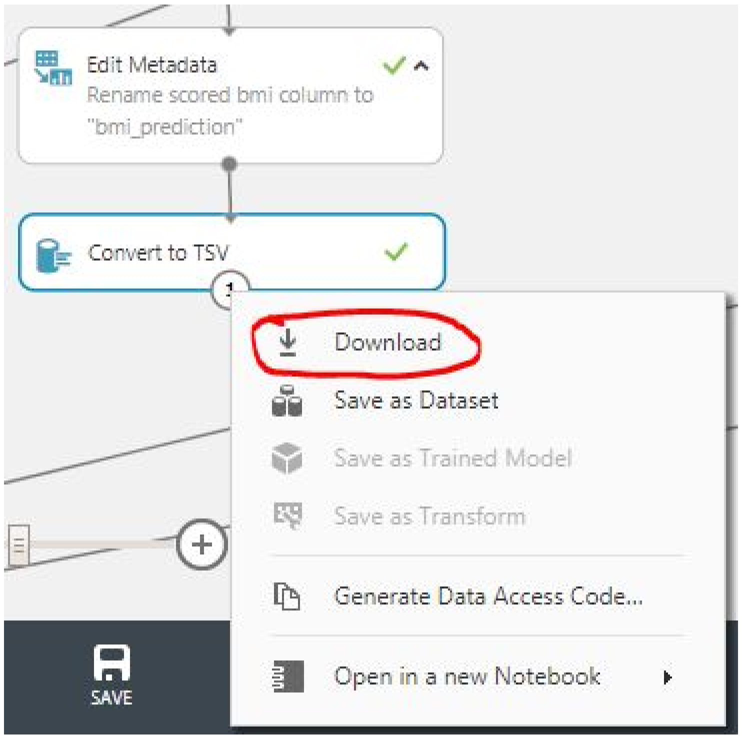
1. Connect the output of **Score Model** to the input of a new **Edit Metadata** module.

2. In the **Edit Metadata** properties, select the **Scored Label Mean** column. Leave everything unchanged except for **New column names**, set it to **bmi_prediction**.

3. Connect the output of the last **Edit Metadata** to the input of a new **Convert to TSV** module.

4. Save and run the experiment.

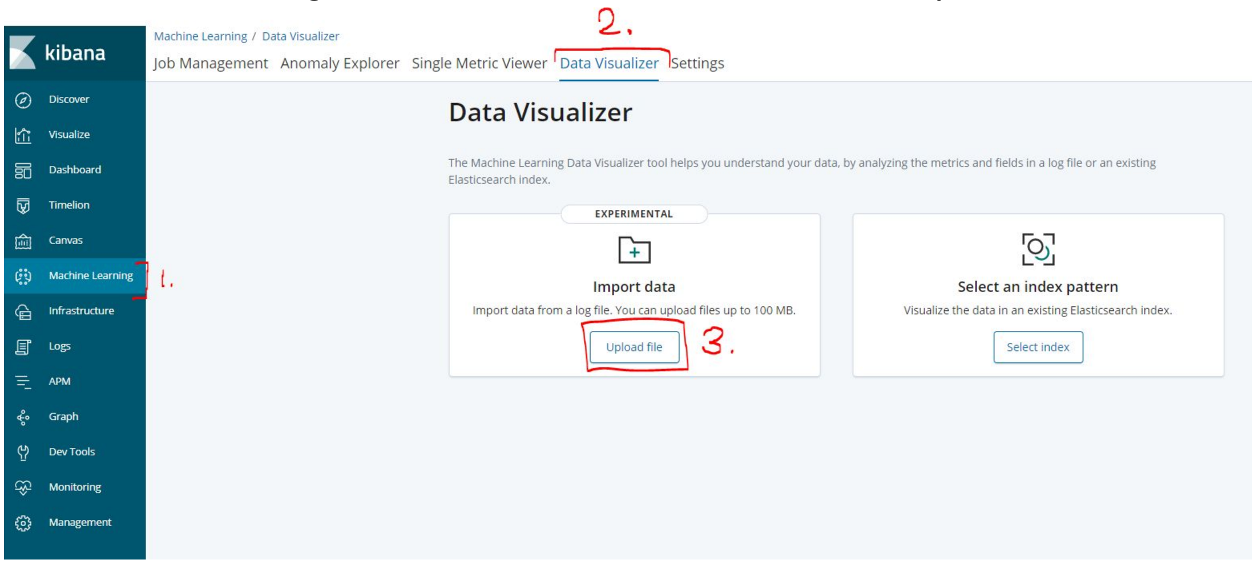
5. Right-click on the output of **Convert to TSV** and select **Download** to download the scored dataset. Make sure to rename the downloaded file to something you can remember.



6. Log into your Elasticsearch Service dashboard at https://cloud.elastic.co

7. Under your deployments, click **Launch Kibana**, and login to your deployment's Kibana with your credentials.

8. In the **Machine Learning** sidebar, click on the **Data Visualizer** tab and click **Upload File**.



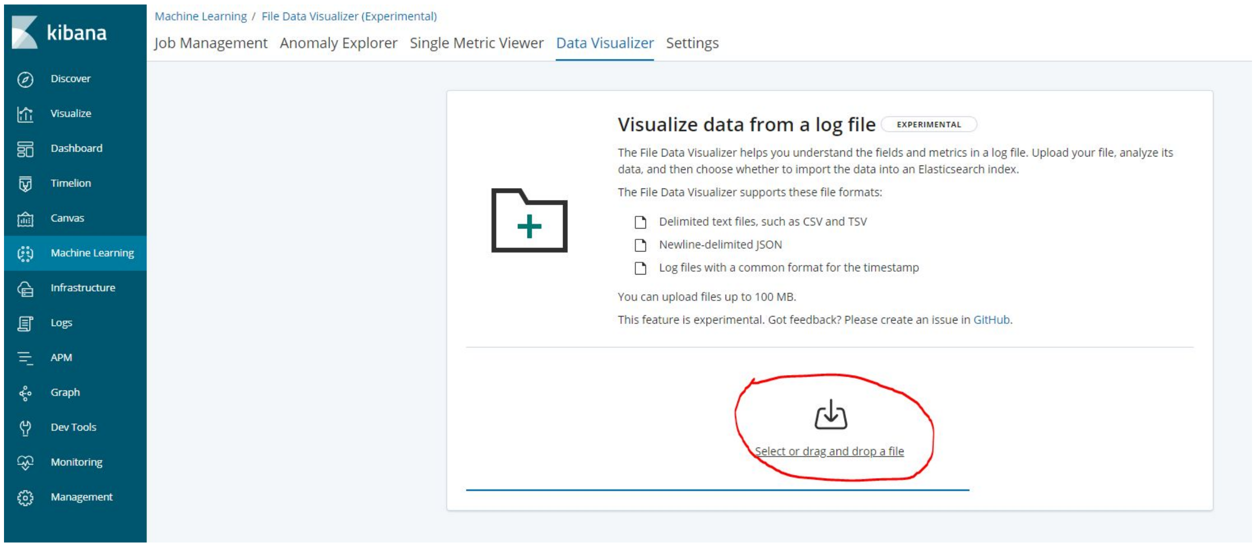
Data Visualizer

The Machine Learning Data Visualizer tool helps you understand your data, by analyzing the metrics and fields in a log file or an existing Elasticsearch index.

Import data from a log file. You can upload files up to 100 MB.

Visualize the data in an existing Elasticsearch index.

9. Select the file you've downloaded from Azure ML and upload it.



Visualize data from a log file

The File Data Visualizer helps you understand the fields and metrics in a log file. Upload your file, analyze its data, and then choose whether to import the data into an Elasticsearch index.

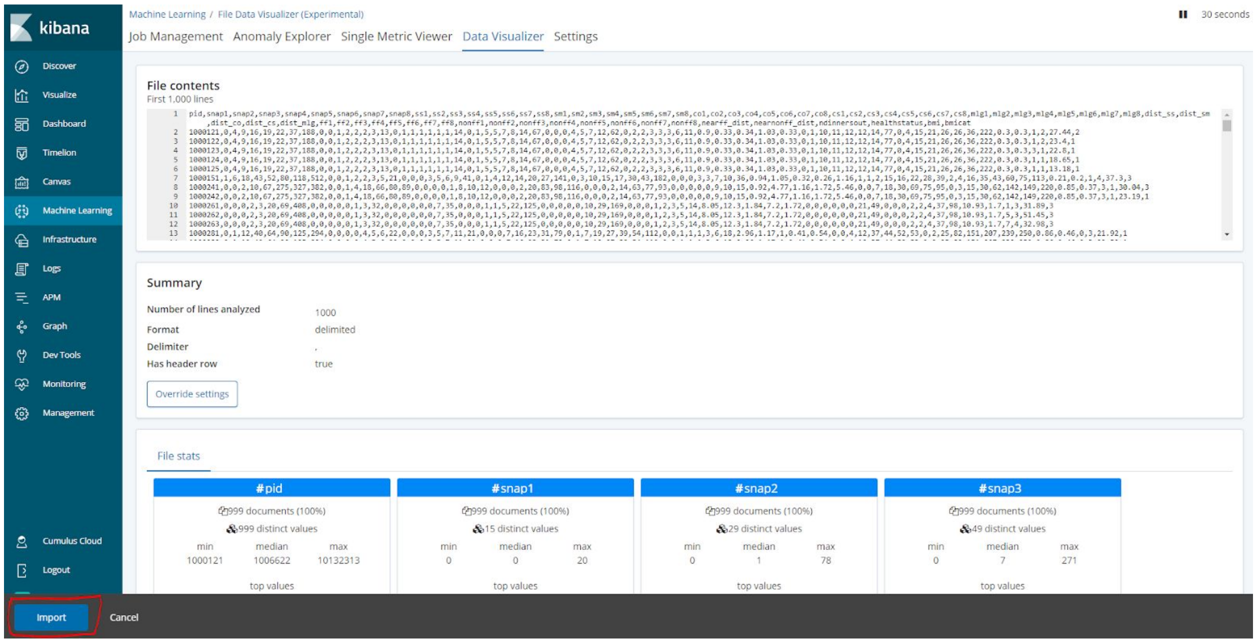
The File Data Visualizer supports these file formats:

- Delimited text files, such as CSV and TSV
- Newline-delimited JSON
- Log files with a common format for the timestamp

You can upload files up to 100 MB.

This feature is experimental. Got feedback? Please create an issue in GitHub.

10. It will only read the first 1000 lines at this point, click the **Import** button on the bottom left.



File contents

First 1,000 lines

```
pid,snap1,snap2,snap3,snap4,snap5,snap6,snap7,snap8,ss1,ss2,ss3,ss4,ss5,ss6,ss7,ss8,sm1,sm2,sm3,sm4,sm5,sm6,sm7,sm8,co1,co2,co3,co4,co5,co6,co7,co8,cs1,cs2,cs3,cs4,cs5,cs6,cs7,cs8,mlg1,mlg2,mlg3,mlg4,mlg5,mlg6,mlg7,mlg8,dist_ss,dist_sm
    ,dist_co,dist_cs,dist_mlg,ff1,ff2,ff3,ff4,ff5,ff6,ff7,ff8,nonff1,nonff2,nonff3,nonff4,nonff5,nonff6,nonff7,nonff8,nearff_dist,nearnonff_dist,ndinnersout,healthstatus,bmi,bmicat
1000121,0,4,9,16,19,22,37,188,0,0,1,2,2,2,3,13,0,1,1,1,1,1,1,14,0,1,5,5,7,8,14,67,0,0,0,4,5,7,12,62,0,2,2,3,3,3,6,11,0.9,0.33,0.34,1.03,0.33,0,1,10,11,12,12,14,77,0,4,15,21,26,26,36,222,0.3,0.3,1,2,27.44,2
1000122,0,4,9,16,19,22,37,188,0,0,1,2,2,2,3,13,0,1,1,1,1,1,1,14,0,1,5,5,7,8,14,67,0,0,0,4,5,7,12,62,0,2,2,3,3,3,6,11,0.9,0.33,0.34,1.03,0.33,0,1,10,11,12,12,14,77,0,4,15,21,26,26,36,222,0.3,0.3,1,2,23.4,1
1000123,0,4,9,16,19,22,37,188,0,0,1,2,2,2,3,13,0,1,1,1,1,1,1,14,0,1,5,5,7,8,14,67,0,0,0,4,5,7,12,62,0,2,2,3,3,3,6,11,0.9,0.33,0.34,1.03,0.33,0,1,10,11,12,12,14,77,0,4,15,21,26,26,36,222,0.3,0.3,3,1,22.8,1
1000124,0,4,9,16,19,22,37,188,0,0,1,2,2,2,3,13,0,1,1,1,1,1,1,14,0,1,5,5,7,8,14,67,0,0,0,4,5,7,12,62,0,2,2,3,3,3,6,11,0.9,0.33,0.34,1.03,0.33,0,1,10,11,12,12,14,77,0,4,15,21,26,26,36,222,0.3,0.3,1,1,18.65,1
1000125,0,4,9,16,19,22,37,188,0,0,1,2,2,2,3,13,0,1,1,1,1,1,1,14,0,1,5,5,7,8,14,67,0,0,0,4,5,7,12,62,0,2,2,3,3,3,6,11,0.9,0.33,0.34,1.03,0.33,0,1,10,11,12,12,14,77,0,4,15,21,26,26,36,222,0.3,0.3,1,1,13.18,1
1000151,1,6,18,43,52,80,118,512,0,0,1,2,2,3,5,21,0,0,0,3,5,6,9,41,0,1,4,12,14,20,27,141,0,3,10,15,17,30,43,182,0,0,0,3,3,7,10,36,0.94,1.05,0.32,0.26,1.16,1,1,2,15,16,22,28,39,2,4,16,35,43,60,75,113,0.21,0.2,1,4,37.3,3
1000241,0,0,2,10,67,275,327,382,0,0,1,4,18,66,80,89,0,0,0,0,1,8,10,12,0,0,0,2,20,83,98,116,0,0,0,2,14,63,77,93,0,0,0,0,0,9,10,15,0.92,4.77,1.16,1.72,5.46,0,0,7,18,30,69,75,95,0,3,15,30,62,142,149,220,0.85,0.37,3,1,30.04,3
1000242,0,0,2,10,67,275,327,382,0,0,1,4,18,66,80,89,0,0,0,0,1,8,10,12,0,0,0,2,20,83,98,116,0,0,0,2,14,63,77,93,0,0,0,0,0,9,10,15,0.92,4.77,1.16,1.72,5.46,0,0,7,18,30,69,75,95,0,3,15,30,62,142,149,220,0.85,0.37,3,1,23.19,1
1000261,0,0,0,2,3,20,69,408,0,0,0,0,0,1,3,32,0,0,0,0,0,0,7,35,0,0,0,1,1,5,22,125,0,0,0,0,0,10,29,169,0,0,0,1,2,3,5,14,8.05,12.3,1.84,7.2,1.72,0,0,0,0,0,0,21,49,0,0,0,2,2,4,37,98,10.93,1.7,1,3,31.89,3
1000262,0,0,0,2,3,20,69,408,0,0,0,0,0,1,3,32,0,0,0,0,0,0,7,35,0,0,0,1,1,5,22,125,0,0,0,0,0,10,29,169,0,0,0,1,2,3,5,14,8.05,12.3,1.84,7.2,1.72,0,0,0,0,0,0,21,49,0,0,0,2,2,4,37,98,10.93,1.7,5,3,51.45,3
1000263,0,0,0,2,3,20,69,408,0,0,0,0,0,1,3,32,0,0,0,0,0,0,7,35,0,0,0,1,1,5,22,125,0,0,0,0,0,10,29,169,0,0,0,1,2,3,5,14,8.05,12.3,1.84,7.2,1.72,0,0,0,0,0,0,21,49,0,0,0,2,2,4,37,98,10.93,1.7,7,4,32.98,3
1000281,0,1,12,40,64,90,125,294,0,0,0,0,4,5,6,22,0,0,0,3,5,7,11,21,0,0,0,7,16,23,31,79,0,1,7,19,27,39,54,112,0,0,1,1,1,3,6,18,2.96,1.17,1,0.41,0.54,0,0,4,12,37,44,52,53,0,2,25,82,151,207,239,250,0.86,0.46,0,3,21.92,1
```

Summary

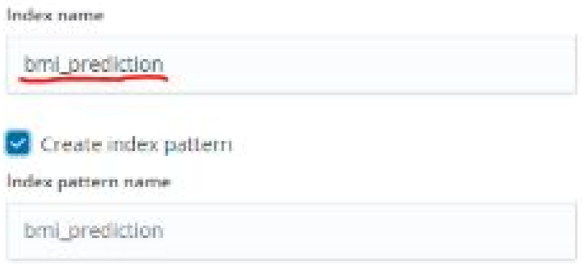
Number of lines analyzed	1000
Format	delimited
Delimiter	,
Has header row	true

Override settings

File stats

#pid	#snap1	#snap2	#snap3
999 documents (100%)	999 documents (100%)	999 documents (100%)	999 documents (100%)
999 distinct values	15 distinct values	29 distinct values	49 distinct values
min 1000121	min 0	min 0	min 0
median 1006622	median 0	median 1	median 7
max 10132313	max 20	max 78	max 271
top values	top values	top values	top values

11. Name the index **bmi_prediction** and click **Import**.



Index name

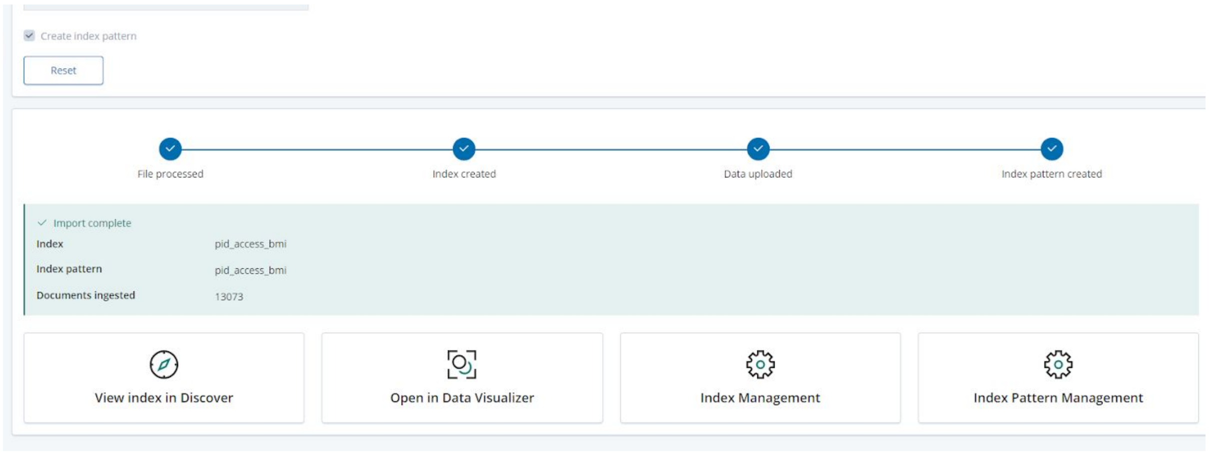
bmi_prediction

✓ Create index pattern

Index pattern name

bmi_prediction

Wait for the upload to finish, it may take a while. It it freezes or hangs, try again and not switch away from the browser window or tab.

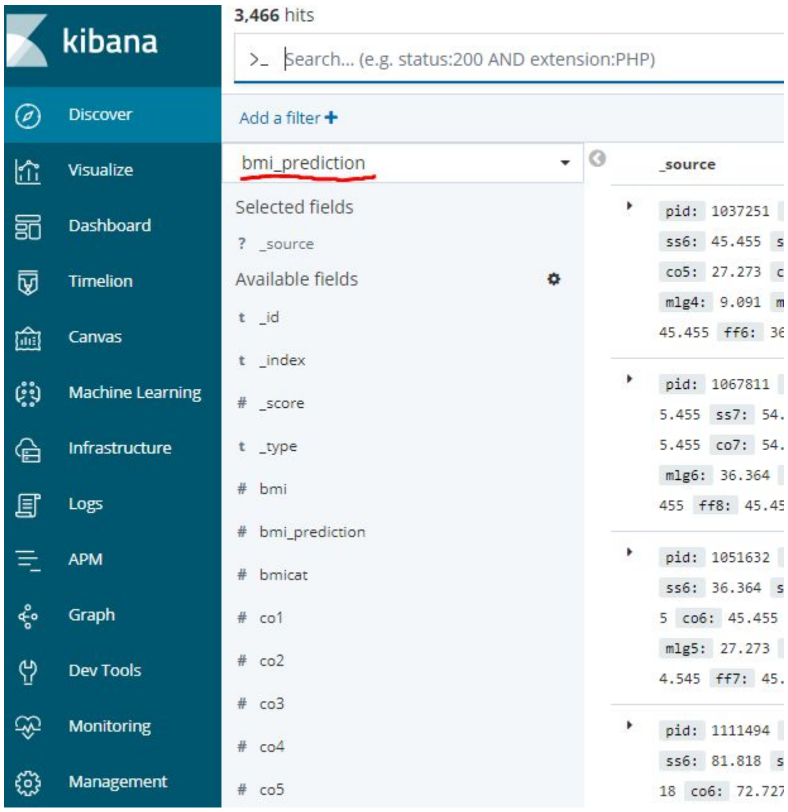


✓ Create index pattern

✓ Import complete

Index	pid_access_bmi
Index pattern	pid_access_bmi
Documents ingested	13073

12. Verify that the data was successfully imported by going to **Discover** and selecting the **bmi_prediction** index.



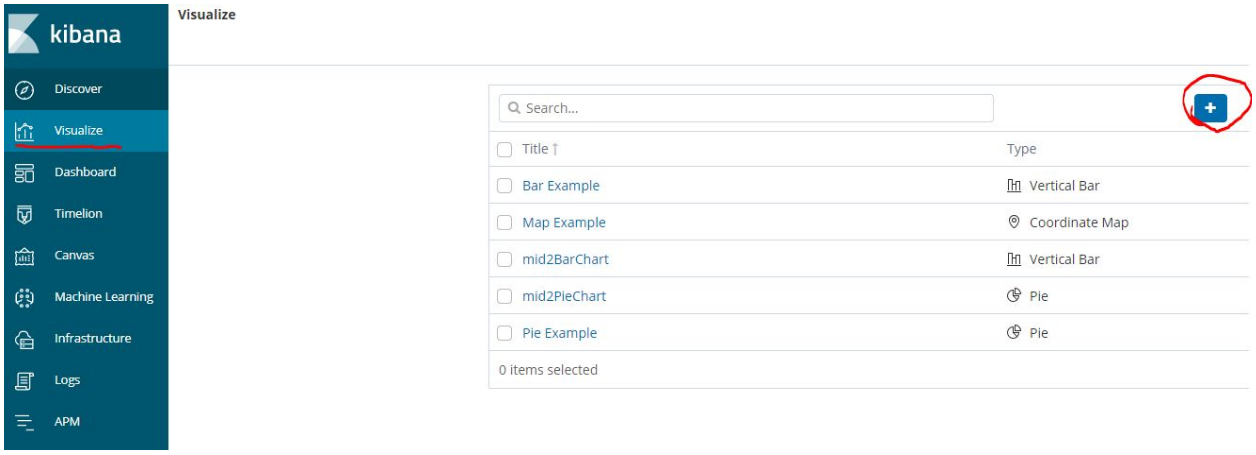
```
pid: 1037251
ss6: 45.455
co5: 27.273
mlg4: 9.091
45.455 ff6: 36

pid: 1067811
5.455 ss7: 54.
5.455 co7: 54.
mlg6: 36.364
455 ff8: 45.45

pid: 1051632
ss6: 36.364
5 co6: 45.455
mlg5: 27.273
4.545 ff7: 45.

pid: 1111494
ss6: 81.818
18 co6: 72.727
```

13. Go to **Visualize** and create a new visualization

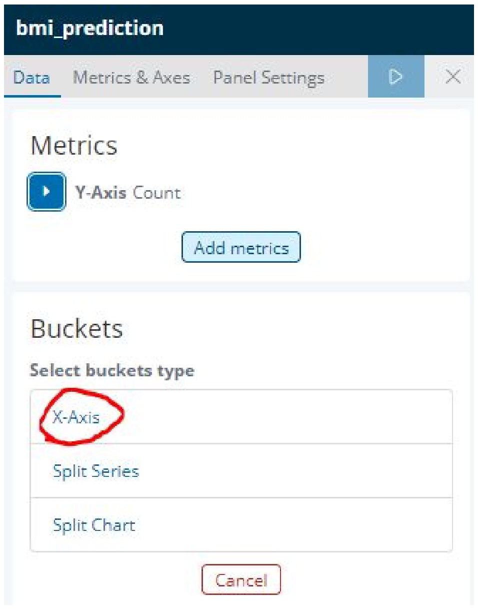


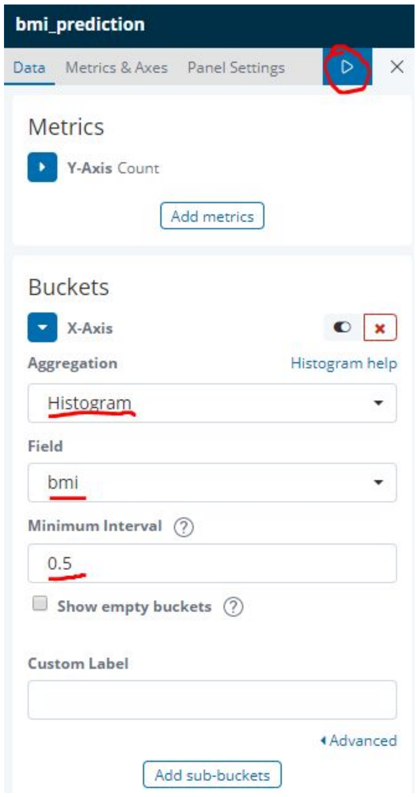
Title ↑	Type
Bar Example	Vertical Bar
Map Example	Coordinate Map
mid2BarChart	Vertical Bar
mid2PieChart	Pie
Pie Example	Pie

0 items selected

14. For the visualization type, select **Area** and select the **bmi_prediction** index.

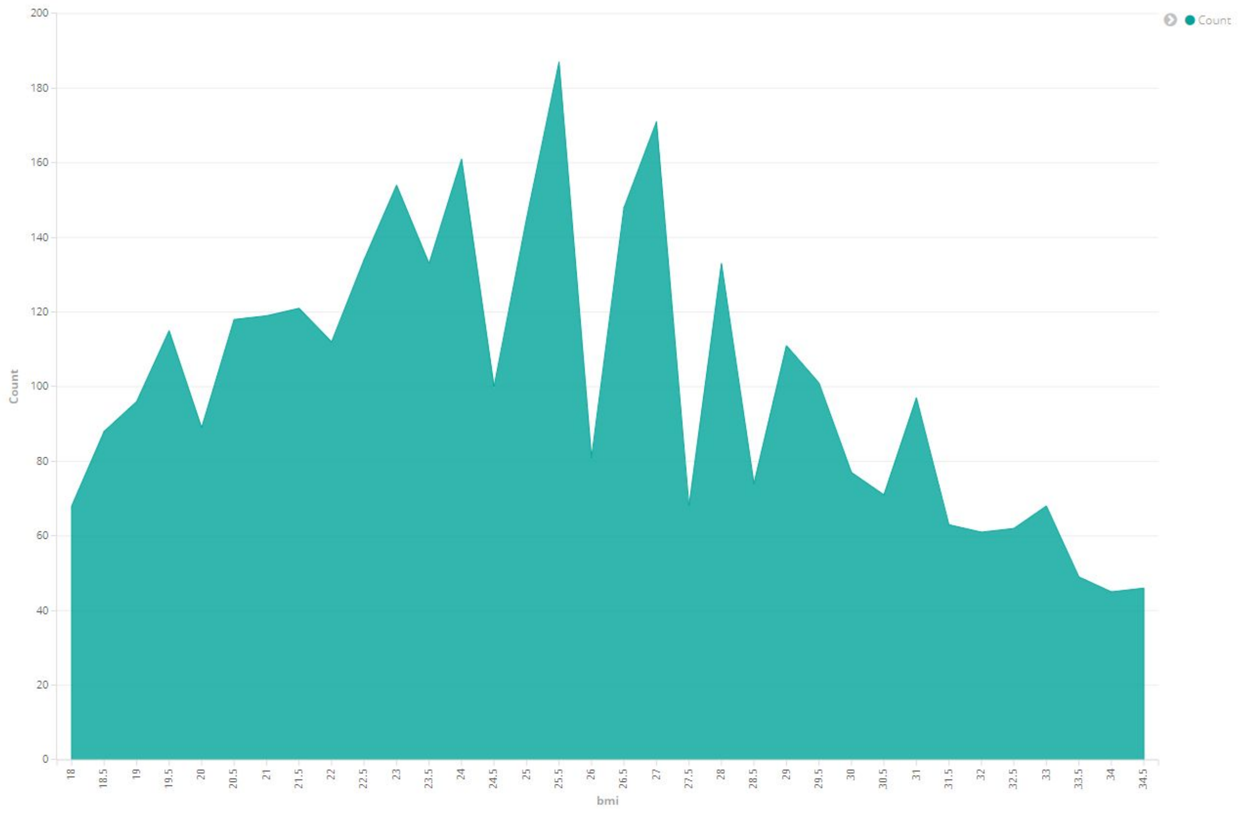
15. Make sure the **Y-Axis** is set to Count. Add a bucket for **X-Axis** and set the aggregation to **Histogram** and field **bmi**. Set the minimum interval to **0.5**. Click the play button to update the graph.



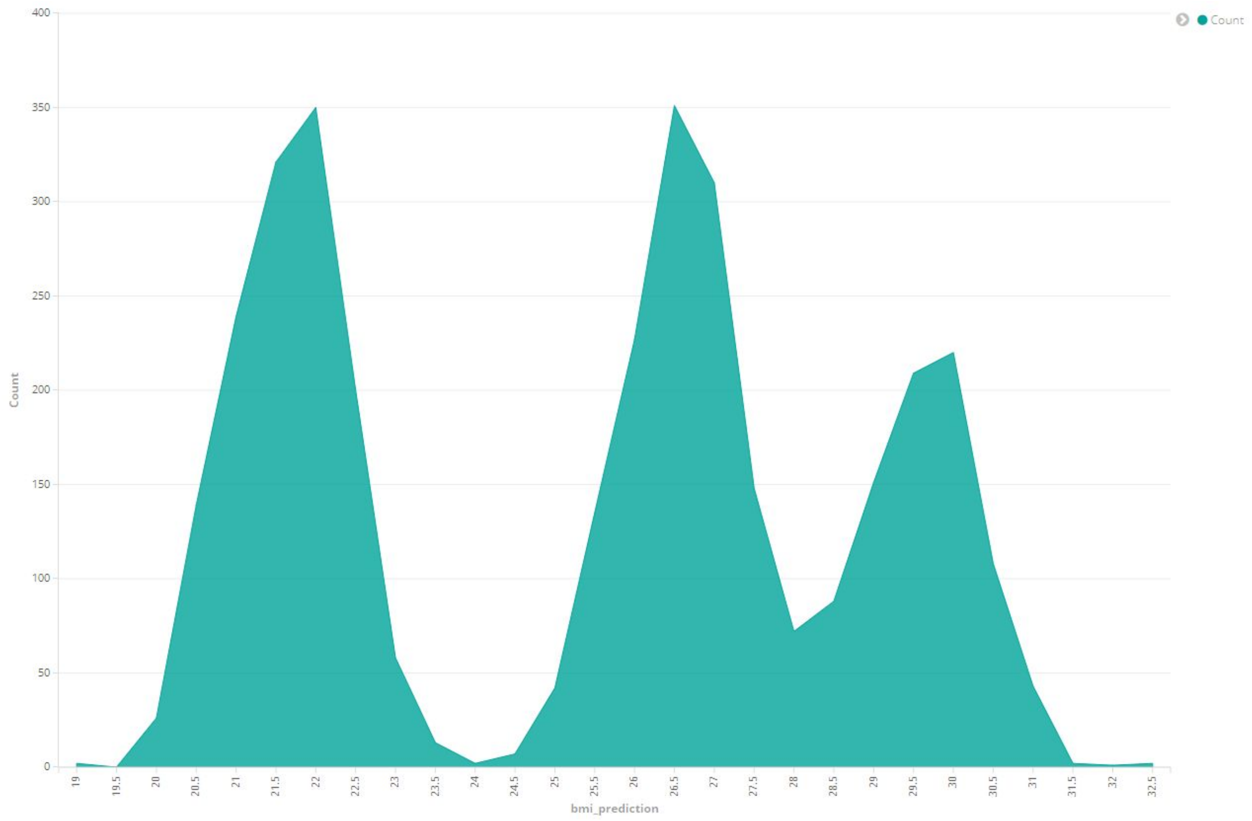


Aggregation: Histogram
Field: bmi
Minimum Interval: 0.5
Show empty buckets

This results in a graph of the true BMI values in the dataset:

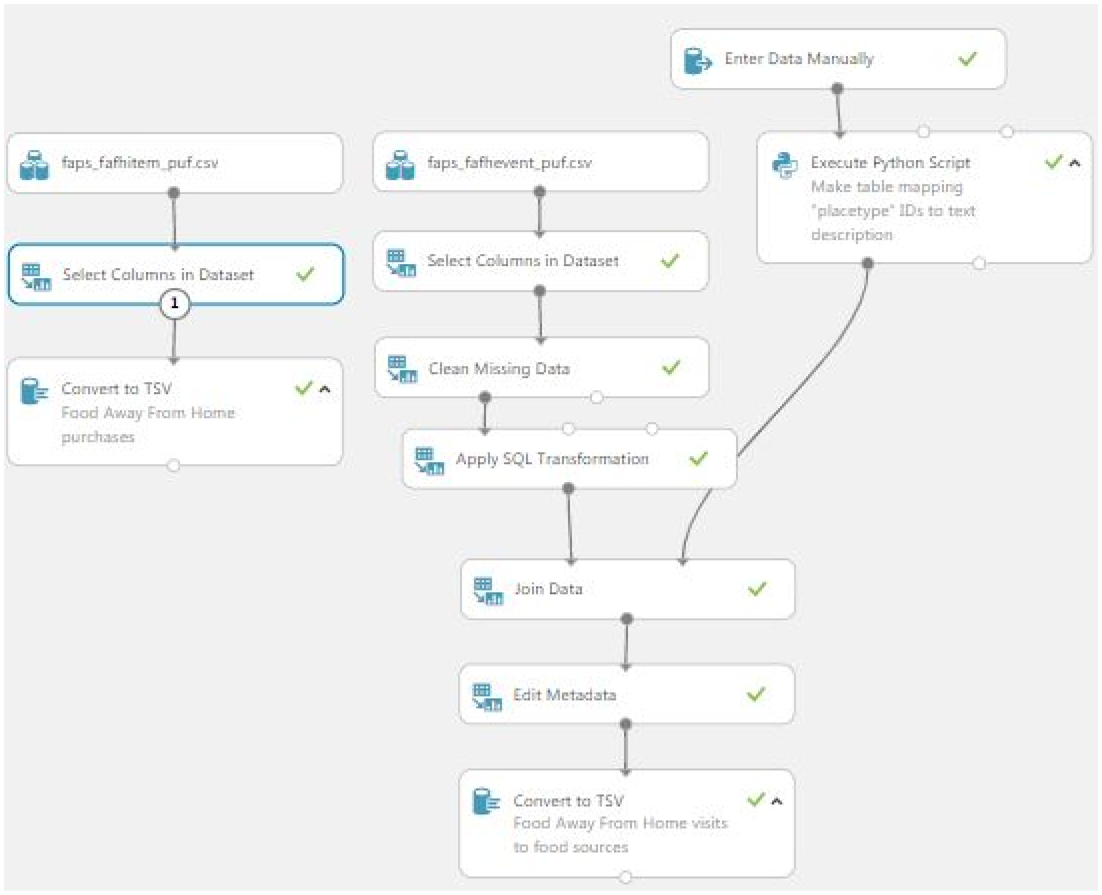


16. Change the field to **bmi_prediction** to see the graph of the predicted bmi:



Step 4 (Optional): Other Visualizations

1. If you want to make a word cloud of restaurants visited and food bought, or a timeline, you can make a new mini-experiment to transform the **faps_fafhitem_puf** and **faps_fafhevent_puf** datasets we uploaded into something we can export to Elasticsearch as well. Set up the modules like so:



7. In the **Select Columns in Dataset** for **faps_fafhitem_puf**, select columns:
`hhnum,eventid,itemnum,itemdesc`

8. In the **Select Columns in Dataset** for **faps_fafhevent_puf**, select columns:
`eventid,hhnum,date,placetype,placedist_s,totalpaid,`
`breakfast,lunch,dinner_supper,snack_drink,whogotpnum`

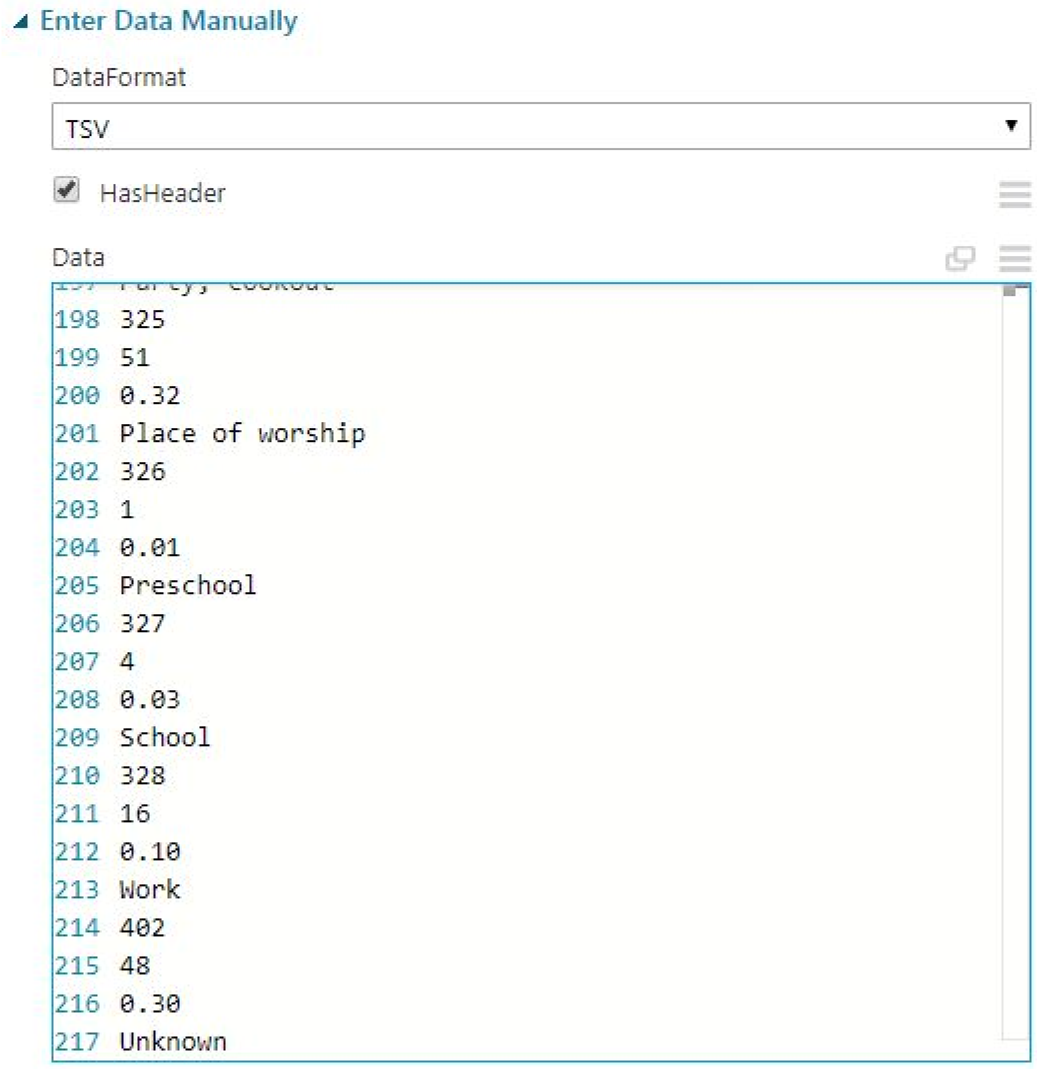
9. In the **Clean Missing Data** module, select the columns:
`breakfast,lunch,dinner_supper,snack_drink`
and set the **Cleaning mode** to **Custom substitution value**. Set the **Replacement value** to **0**.

10. In the **Apply SQL Transformation** module, use the SQL script from Github:
https://github.com/shozonu/usda-foodaps-bmi-model/blob/master/sql/fafh_event_clean.sql

```
update t1 set breakfast = 0 where breakfast < 0;
update t1 set lunch = 0 where lunch < 0;
update t1 set dinner_supper = 0 where dinner_supper < 0;
update t1 set snack_drink = 0 where snack_drink < 0;
update t1 set placedist_s = null where placedist_s < 0;
select * from t1;
```

11. In **faps_fafhevent_puf**, the **placetype** column is an integer coded to a specific text description. The code table is in the official data handbook at the USDA ERS dataset link. We can copy the entire table from the handbook but it is not formatted correctly and there are some irrelevant lines. Copy/paste the text from the Github into the **Enter Data Manually** node:
https://github.com/shozonu/usda-foodaps-bmi-model/blob/master/Data/event_placetype_code_raw.txt

And set the **DataFormat** to **TSV**. **Check** the **HasHeader** box.



Enter Data Manually

DataFormat: TSV

[✓] HasHeader

Data

```
325
51
0.32
Place of worship
326
1
0.01
Preschool
327
4
0.03
School
328
16
0.10
Work
402
48
0.30
Unknown
```

12. In the **Execute Python Script** module, use the code from Github to replace the placetype integer code with the string description:
https://github.com/shozonu/usda-foodaps-bmi-model/blob/master/py/fafh_event_placeid_desc.py

```
import pandas as pd

def azureml_main(df = None):
    # Create dictionary to hold values
    arr = {"desc" : [] ,"id" : []}
    for index, row in df.iterrows():
        # Place description occurs every nth row,
        # Corresponding place id occurs every n-3th row
        if((index + 1) % 4 == 0):
            arr["id"].append(int(df["raw"].iat[index - 3]))
            arr["desc"].append(df["raw"].iat[index])

    # Construct new dataframe from dictionary
    data = pd.DataFrame.from_dict(arr)

    # Rearrange columns so that id is first
    cols = data.columns.tolist()
    cols = cols[-1:] + cols[:-1]
    data = data[cols]
    return data
```

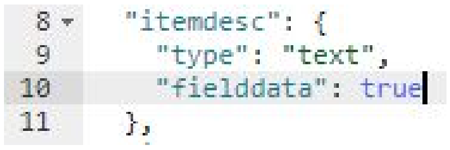
13. In the **Join Data** module, set the **Join key columns for L** to **placetype** and **Join key columns for R** to **id**. Make sure the **Match case** box is **checked**. Set the **Join type** to **Inner Join**. Make sure the **Keep right key column** is **unchecked**.

14. In the **Edit Metadata** module, select **whogotpnum** and type **pnum** for the **New column names**.

15. After saving and running the experiment, download the two datasets by right-clicking on the two **Convert to TSV** modules' output and clicking **download**. Make sure to rename the downloaded files to something you can remember.

16. Follow the same basic steps in Step 3: Visualization for uploading the data with Kibana:
Machine Learning > Data Visualizer > Import
But for these two datasets, you need to change the import settings when you get to the **Import** stage.

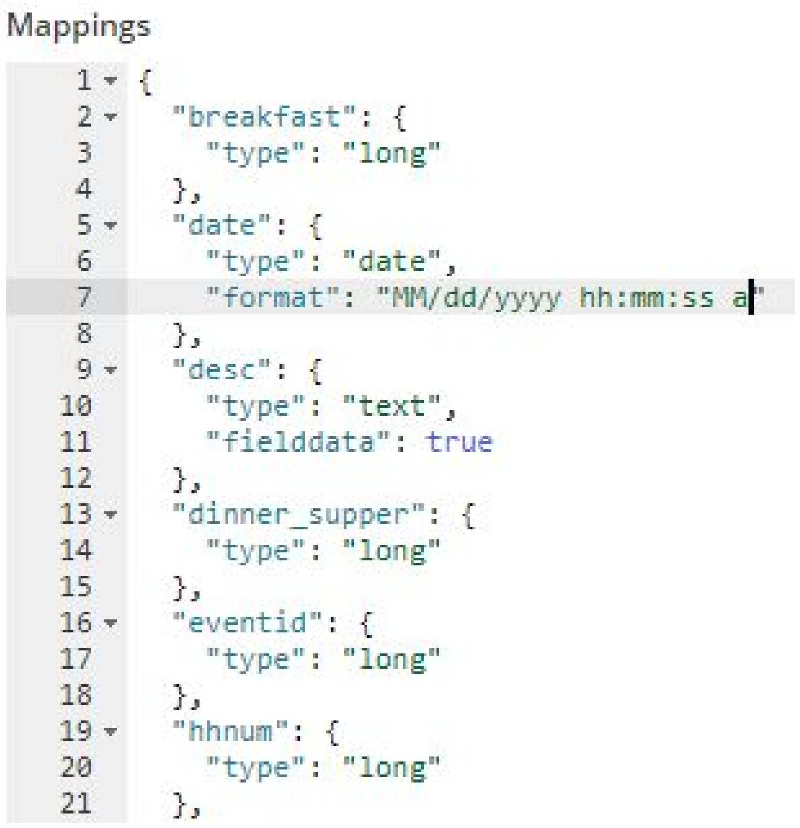
17. For **fafh_item**, click the **Advanced** tab and edit the mappings. Under **itemdesc**, add a comma to the end of the existing line, and add a line:



```
"itemdesc": {
  "type": "text",
  "fielddata": true
},
```

This allows the text field to be aggregatable for the purposes of making the word cloud.

18. We need to change **fafh_event** as well. Change the **desc** field and **date** field like so:

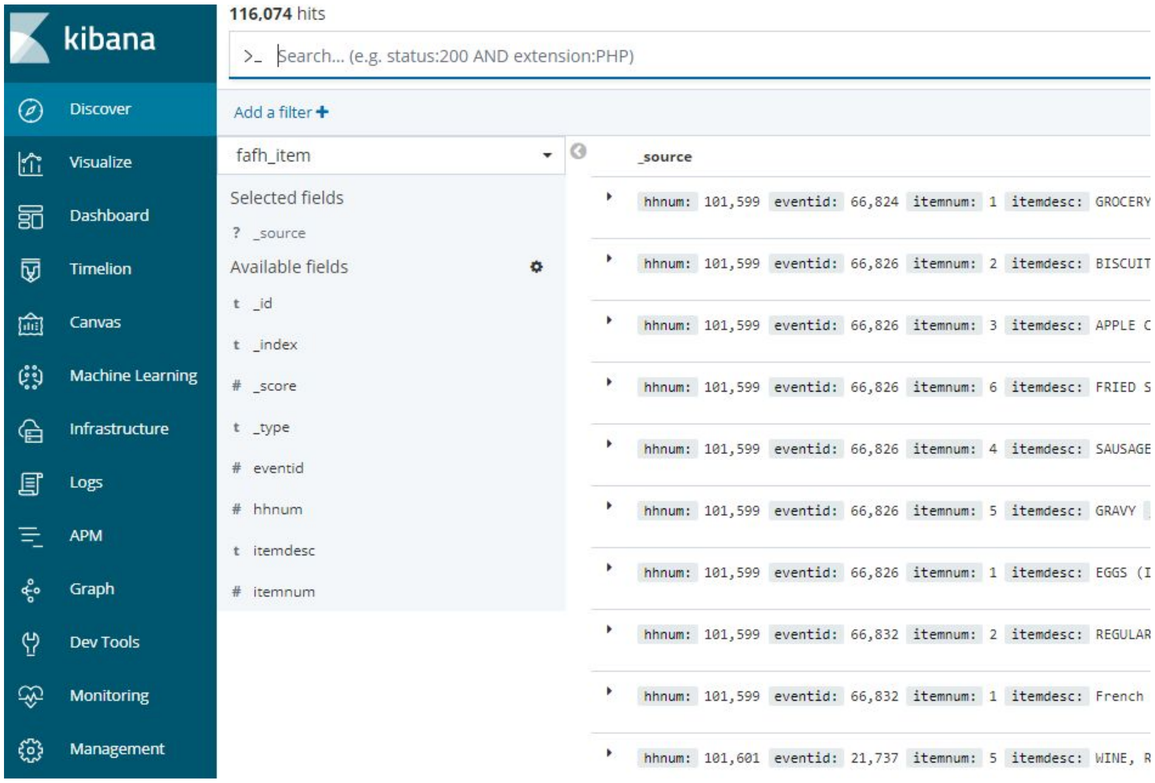


Mappings

```
{
  "breakfast": {
    "type": "long"
  },
  "date": {
    "type": "date",
    "format": "MM/dd/yyyy hh:mm:ss a"
  },
  "desc": {
    "type": "text",
    "fielddata": true
  },
  "dinner_supper": {
    "type": "long"
  },
  "eventid": {
    "type": "long"
  },
  "hhnum": {
    "type": "long"
  },
```

In order to properly import the date for our time visualization, we needed to change the type to "date" and specify the date format that our dataset used.

19. Once you successfully uploaded both of them with the modified mappings, check them in **Discover**.



116,074 hits

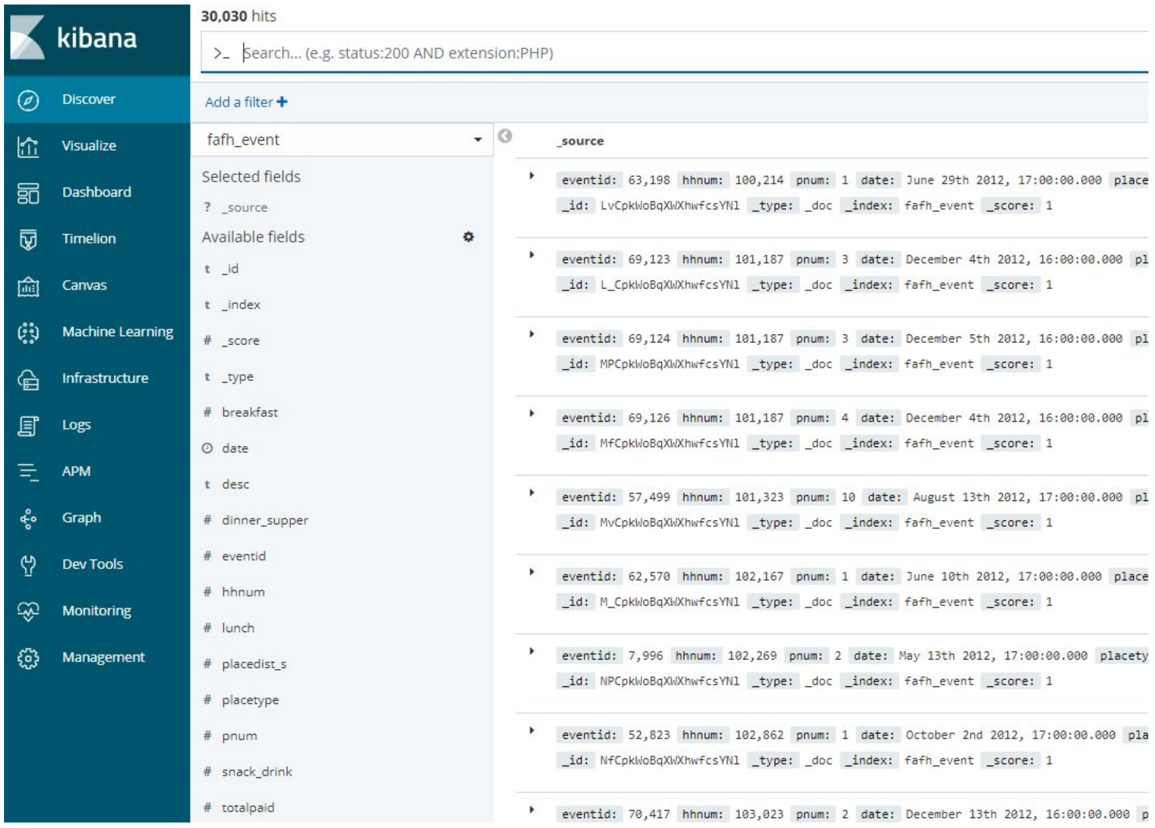
>_ Search... (e.g. status:200 AND extension:PHP)

fafh_item

Selected fields
? _source

Available fields
t _id
t _index
_score
t _type
# eventid
# hhnum
t itemdesc
# itemnum

```
_source
hhnum: 101,599 eventid: 66,824 itemnum: 1 itemdesc: GROCERY
hhnum: 101,599 eventid: 66,826 itemnum: 2 itemdesc: BISCUIT
hhnum: 101,599 eventid: 66,826 itemnum: 3 itemdesc: APPLE C
hhnum: 101,599 eventid: 66,826 itemnum: 6 itemdesc: FRIED S
hhnum: 101,599 eventid: 66,826 itemnum: 4 itemdesc: SAUSAGE
hhnum: 101,599 eventid: 66,826 itemnum: 5 itemdesc: GRAVY
hhnum: 101,599 eventid: 66,826 itemnum: 1 itemdesc: EGGS (I
hhnum: 101,599 eventid: 66,832 itemnum: 2 itemdesc: REGULAR
hhnum: 101,599 eventid: 66,832 itemnum: 1 itemdesc: French
hhnum: 101,601 eventid: 21,737 itemnum: 5 itemdesc: WINE, R
```



30,030 hits

>_ Search... (e.g. status:200 AND extension:PHP)

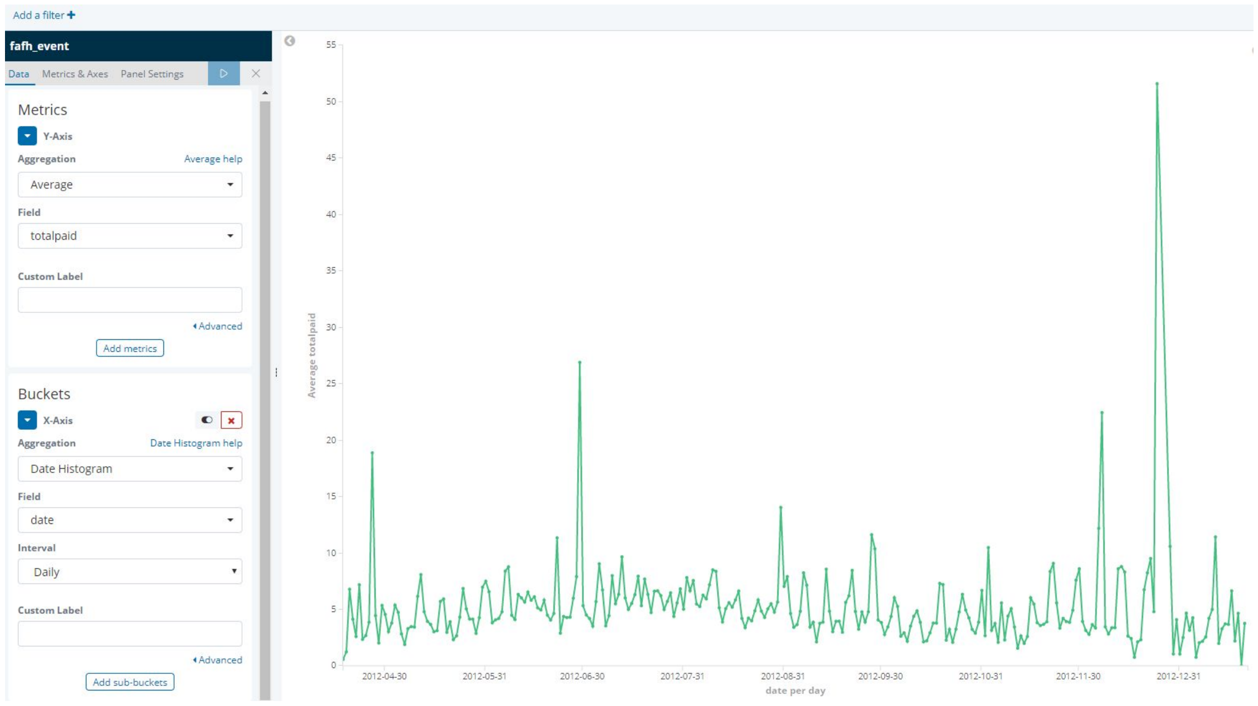
fafh_event

Selected fields
? _source

Available fields
t _id
t _index
_score
t _type
# breakfast
date
t desc
# dinner_supper
# eventid
# hhnum
# lunch
# placedist_s
# placetype
# pnum
# snack_drink
# totalpaid

```
_source
eventid: 63,198 hhnum: 100,214 pnum: 1 date: June 29th 2012, 17:00:00.000 place
_id: LvCpkWoBqXWXhwfcsYNl _type: _doc _index: fafh_event _score: 1
eventid: 69,123 hhnum: 101,187 pnum: 3 date: December 4th 2012, 16:00:00.000 pl
_id: L_CpkWoBqXWXhwfcsYNl _type: _doc _index: fafh_event _score: 1
eventid: 69,124 hhnum: 101,187 pnum: 3 date: December 5th 2012, 16:00:00.000 pl
_id: MPCpkWoBqXWXhwfcsYNl _type: _doc _index: fafh_event _score: 1
eventid: 69,126 hhnum: 101,187 pnum: 4 date: December 4th 2012, 16:00:00.000 pl
_id: MfCpkWoBqXWXhwfcsYNl _type: _doc _index: fafh_event _score: 1
eventid: 57,499 hhnum: 101,323 pnum: 10 date: August 13th 2012, 17:00:00.000 pl
_id: MvCpkWoBqXWXhwfcsYNl _type: _doc _index: fafh_event _score: 1
eventid: 62,570 hhnum: 102,167 pnum: 1 date: June 10th 2012, 17:00:00.000 place
_id: M_CpkWoBqXWXhwfcsYNl _type: _doc _index: fafh_event _score: 1
eventid: 7,996 hhnum: 102,269 pnum: 2 date: May 13th 2012, 17:00:00.000 placety
_id: NPCpkWoBqXWXhwfcsYNl _type: _doc _index: fafh_event _score: 1
eventid: 52,823 hhnum: 102,862 pnum: 1 date: October 2nd 2012, 17:00:00.000 pla
_id: NfCpkWoBqXWXhwfcsYNl _type: _doc _index: fafh_event _score: 1
eventid: 70,417 hhnum: 103,023 pnum: 2 date: December 13th 2012, 16:00:00.000 p
```

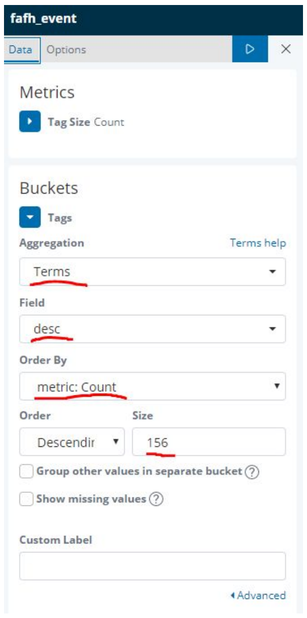
20. In **Visualize**, make a new **Line Graph** with the **fafh_event** index and set their settings as follows to create the time-series graph of average money spent on food during the period. You can set the time filter to **Absolute**, between **April 2012** and **January 2013**.



Aggregation: Average
Field: totalpaid
Custom Label:

Aggregation: Date Histogram
Field: date
Interval: Daily
Custom Label:

21. In **Visualize**, make a new **Tag Cloud** with the **fafh_event** index. Set their settings as follows to create a word cloud describing food-away-from-home events. You can add a filter and set it for the **desc** field to be **not one of** any of the words you specify. This can help remove stop words and other non-descriptive tags from showing up in the word cloud.

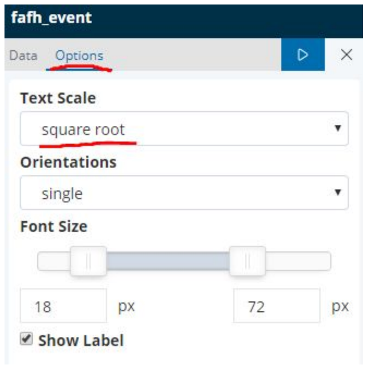


fafh_event

Data | Options

Metrics
Tag Size Count

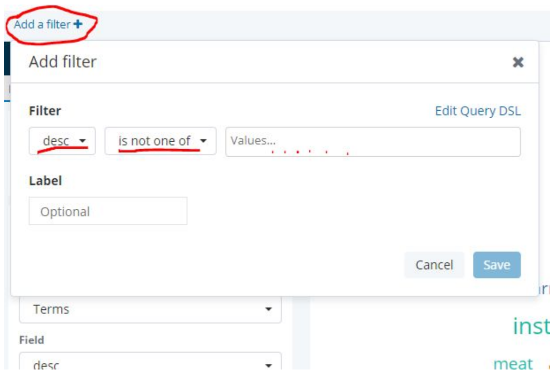
Buckets
Tags
Aggregation: Terms
Field: desc
Order By: metric: Count
Order: Descending
Size: 156
☐ Group other values in separate bucket
☐ Show missing values
Custom Label:



fafh_event

Data | Options

Text Scale: square root
Orientations: single
Font Size: 18 px – 72 px
☑ Show Label

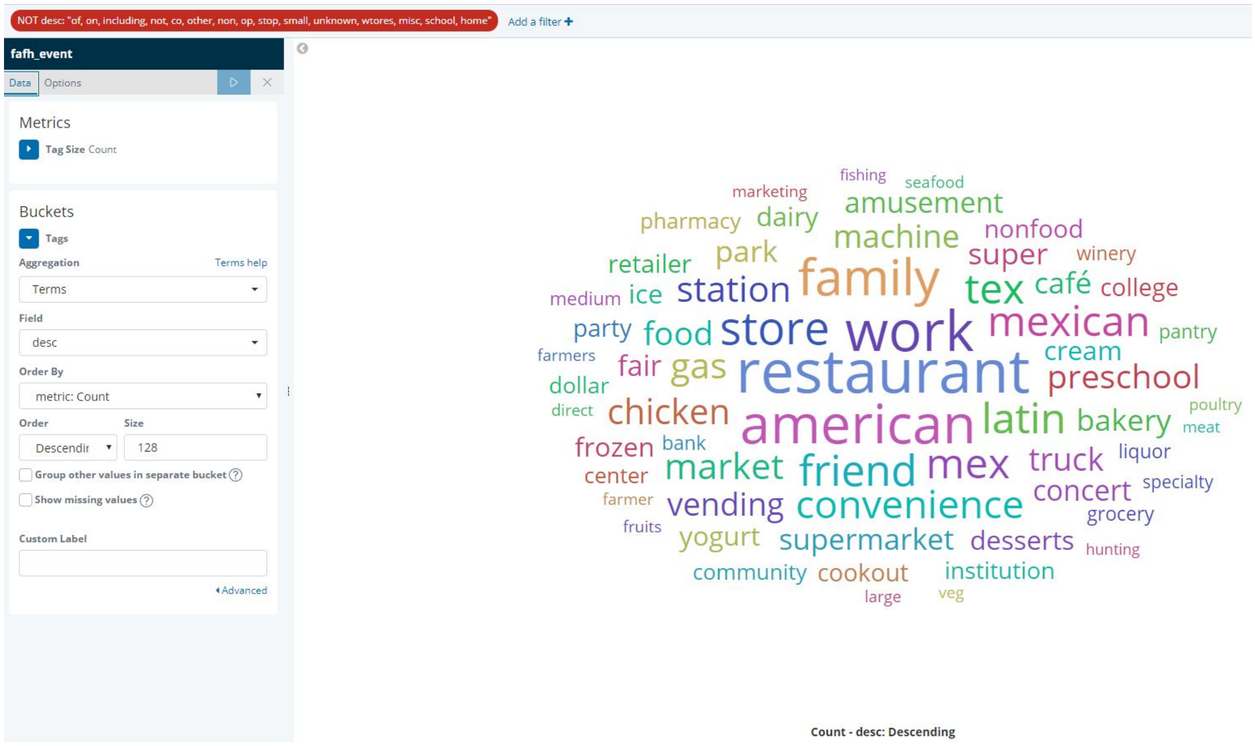


Add a filter +

Add filter

Filter: desc | is not one of | Values...
Label: Optional

Cancel | Save



NOT desc: "of, on, including, not, co, other, non, op, stop, small, unknown, wtores, misc, school, home"   Add a filter +

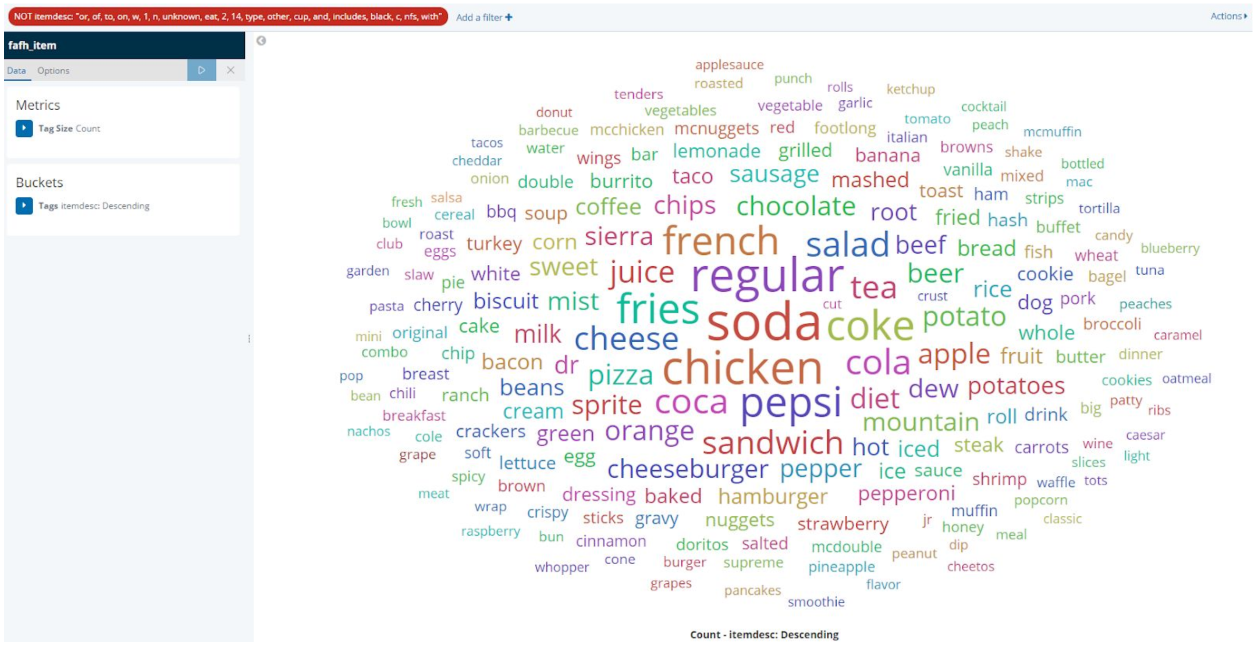
fafh_event

Data | Options

Metrics
Tag Size Count

Buckets
Tags
Aggregation: Terms
Field: desc
Order By: metric: Count
Order: Descending
Size: 128
☐ Group other values in separate bucket
☐ Show missing values
Custom Label:

22. Repeat the same basic steps as step 21 for **fafh_item** to create a word cloud describing food bought away from home. Set the filter words to any words you see in the word cloud that are not helpful.



NOT itemdesc: "or, of, to, on, w, 1, n, unknown, eat, 2, 14, type, other, cup, and, includes, black, c, nfs, with"

fafh_item

Metrics

Tag Size Count

Buckets

Tags itemdesc: Descending

Count - itemdesc: Descending

References

1. Data Source:
   https://www.ers.usda.gov/data-products/foodaps-national-household-food-acquisition-and-purchase-survey/
2. Project Github:
   https://github.com/shozonu/usda-foodaps-bmi-model
3. Azure ML Studio Specifications:
   https://azure.microsoft.com/en-us/pricing/details/machine-learning-studio/
4. Elasticsearch Service Specifications:
   https://www.elastic.co/guide/en/cloud/current/ec-reference-hardware.html
5. Azure ML Project Gallery:
   https://gallery.cortanaintelligence.com/Experiment/USDA-FoodAPS-BMI-Modeling