# Description

This function will automatically detect Simpson’s pairs (i.e. pairs of independent and conditioning variables) in a dataset with a pre-defined DV using regression models. If the user hasn’t specified which model type to use, the function will use logistic regression if the DV is binary and one-versus-all logistic regression if the DV is but not binary. Otherwise, the function will use linear regression.

This function will also do pre-processing steps prior to checking the dataset:

1. Removing user-defined irrelevant columns from the analysis
2. Encoding any non-numeric columns (i.e. string or Boolean data types)
3. Standardizing any non-discrete columns (i.e. columns with more than 10 categories)
4. Binning large conditioning variables if the user hasn’t specified any to bin

Only pairs with pre-defined minimum correlation (between the IV and CV, and between the CV and DV) will be checked by model building. If the DV is binary, only correlation between IV and CV is checked.

## Usage: Jupyter

1. Unzip *simpsons\_paradox.zip*
2. Open Anaconda Prompt and run the following commands:
   1. cd simpsons\_paradox
   2. conda env create -f environment.yml
   3. conda activate simpsons-paradox
   4. jupyter lab or jupyter notebook

## Usage: Scripts

1. Unzip *simpsons\_paradox.zip*
2. Open Anaconda Prompt and run the following commands:
   1. cd simpsons\_paradox
   2. conda env create -f environment.yml
   3. conda activate simpsons-paradox
3. Examples for running from the command line with sample datasets:

|  |
| --- |
| python main.py -input\_file="data/SP\_Data.csv" -dependent\_variable="Survived" -ignore\_columns "PatientId" -bin\_columns "" -standardize -weighting |
| python main.py -input\_file="data/coffee\_data.csv" -dependent\_variable="neuroticism" -ignore\_columns "Unnamed: 0" -max\_pvalue=1 -bin\_columns "" -standardize |
| python main.py -input\_file="data/small\_khancademy.csv" -dependent\_variable="performance" -ignore\_columns "user\_id" "problem\_id" -bin\_columns "timestamp" "solve\_time" "attempts" "tspp" "session\_num" "session\_index" "session\_length" "all\_first\_attempts" "signup\_duration" "total\_solving\_time" "all\_attempts" "all\_problems" "all\_sequences" "month" "join\_month" -standardize -weighting -model=”logistic” |
| python main.py -input\_file="data/conposcovidloc.csv" -dependent\_variable="Outcome1" -ignore\_columns "Row\_ID" -target\_category=1 -bin\_columns "" -standardize -weighting |

## Arguments

|  |  |  |  |
| --- | --- | --- | --- |
| **Argument** | **Type** | | **Description** |
| -infile | String | Required | Path to the CSV file. |
| -dependent\_variable | String | Required | Name of the dependent variable. |
| -model | String | Optional | Type of regression model to use.  Options are: ‘linear’, ‘logistic’ |
| -ignore\_columns | String(s) | Optional | List of columns that user wants to exclude from the analysis. To pass a list from the command line, pass multiple strings and add a space between each string. |
| -bin\_columns | String(s) | Optional | List of columns that user wants to bin prior to the analysis (i.e. columns with more than 10 categories). If user doesn’t pass any, the function will bin columns with many categories by default. |
| -standardize | Boolean | Optional | Include this argument in the command to standardize variables with more than 10 unique values. |
| -bin\_method | String | Optional | By default, the binning method is ‘quantile’, but user can also choose ‘kmeans’ or ‘uniform’. |
| -max\_pvalue | Float | Optional | Pairs with IVs that have large p-values in the simple regression model are filtered out. The default is 0.05. |
| -min\_coeff | Float | Optional | Pairs with IVs that have small coefficients in the simple model are excluded. This is done by filtering out coefficients close to zero using this specified value. If the DV is binary, this can also be done by evaluating the confidence intervals of the odds. The default for this argument is 0.00001. |
| -min\_corr | Float | Optional | \*For numeric variables, any pairs with correlations between the independent variable and conditioning variable, or between the conditioning variable and dependent variable, that are below this minimum are filtered out. The default is 0.01. |
| -output\_plots | Boolean | Optional | Include this argument in the command to output plots and model summary statistics tables (i.e. in a notebook environment) |
| -target\_category | Int | Optional | If the dataset DV is discrete and non-binary, the user can include the target category in order to bin all other categories of the DV into the non-target category (one-versus-all regression). |
| -weighting | Boolean | Optional | Include this argument in the command to exclude weak cases of Simpson’s Paradox, i.e. cases where the sign of the weighted average of the coefficients is the same as the sign of the simple model coefficient value. |

\*Based on the “Can you Trust the Trend: Discovering Simpson's Paradoxes in Social Data” paper: <https://arxiv.org/abs/1801.04385>. The paper “identifies two necessary conditions for the paradox to occur: (1) the independent and conditioning variables are correlated and (2) the value of the outcome variable differs within conditioning subgroups” so we try to implement these necessary conditions by filtering out pairs with little correlation prior to model building.

## Troubleshooting

Some warnings may be raised by the packages used in this function. Here is a description of those warnings and what we do about them:

|  |  |  |
| --- | --- | --- |
| Warning | Description | Suppress |
| Scikit-learn KBinsDiscretizer | This warning happens when the variable we're binning has mostly one value, and all of the records with that value will fall into one bin, and the variable will end up having less than 5 bins (which is the function's default number of bins), but the function handles this issue appropriately by ignoring those cases, so we suppress this warning. | Yes |
| Numpy Overflow warning | This warning is about taking exponents of values that are larger than 709. In this function, this happens when the odds confidence intervals are too large, and when a value of an independent variable is too large. We don’t suppress this warning since it can help users discover issues with the dataset where the models may produce bad results. The best solution is to do good data prep before running the model, and to carefully interpret the results of the model. | No |
| Pandas SettingWithCopyWarning | This warning happens when we try to overwrite values in a pandas series, which happens when converting discrete DV into binary DV (i.e. one-versus-all). We suppress this warning based on this page: https://exceptionshub.com/how-to-deal-with-settingwithcopywarning-in-pandas.html | Yes |

Some other things to be aware of:

* This function will also output a plot corresponding to the regression model, along with a table of summary statistics. To plot the trend lines for the aggregated model and disaggregated models, we use the unstandardized independent variables and the logistic regression model’s predictions to build a linear regression that “approximates” the logistic regression trend line.
* This function will use a regression depending on the DV of the full data or a subgroup of the data. It’s possible there are cases where the aggregated and disaggregated models differ e.g. the aggregate data has a non-binary DV but one of the subgroups for that data has a binary DV. In that case, the function will use a one-versus-all regression for the non-binary DV subgroup, and a binary logistic regression for the aggregate data.

## Data

Some good datasets which can be used for testing this function are from:

* Stack Exchange Dataset: <https://arxiv.org/abs/1801.04385>
* Khan Academy Dataset: <https://arxiv.org/abs/1805.03094>
* Coffee Dataset: <https://rdrr.io/cran/Simpsons/man/Simpsons.html>