2022 MEBDI Machine Learning Competition

Original version: December, 2021
Revised: March 29, 2022

This document describes the 2022 MEBDI ML competition and its rules. This document and all other competition-related material will be posted on MEBDI's ML Competition website at https://mebdi.org/ml-competition-2022. These rules may be modified to handle unforeseen contingencies, in which case the changes will be announced on the web site and via email to competing teams.

ML Problem Description

- **Goal:** Design a computer algorithm that provides the best out-of-sample prediction of the unemployment status one year ahead of individuals in a sample drawn from the matched Current Population Survey. More details below.
- **Eligibility:** To be eligible, you must have entered the program in Fall 2019 or before, be in good academic standing, and must not be on the job market this year (i.e, you must plan to continue as a full time student in Fall 2022). Each student can enter the competition individually or form a team with one other eligible student (team of two). To enter the competition, teams must notify their decision to participate to organizers via email by January 15, 2022. You are allowed to enter the competition again in future years if you satisfy the eligibility conditions for that year. Teams that submit identical or near identical algorithms will both be disqualified.
- **Deadline:** All deliverables listed below must be submitted by May 13, 2022 at Noon US Central Time. Late entries will not be considered.
- **Evaluation:** On May 17, 2022, all teams will participate in a live event where teams will run their code on the "test sample," which will be shared with the teams at that time. Afterward, the submissions will be reviewed by a committee of professors, who (if need be) will also have the final say on the interpretation of rules.
- **Prize:** $5,000

See next page for details.

1 Details

Problem Description

Objective: Design a Machine Learning algorithm with the best out-of-sample performance (as defined below) for predicting the unemployment status of a sample of workers one year ahead, using the set of predictors described below. Specifically, let $y_{i,t+1}$ be a binary indicator for unemployment status: $y_{i,t+1} = 1$ denotes "unemployed" and $y_{i,t+1} = 0$ denotes "employed" in $t + 1$. Similarly, the prediction, $\widehat{y}_{it+1}$, will be a binary assignment of a 0 (employed) or 1 (unemployed) to each individual for his/her predicted unemployment status in $t+1$. The set of allowable predictors is provided with the training sample as part of the competition. You may not use any variable that is not included in the data package posted on MEBDI's ML Competition web site.

Data and Variables

The official "training sample" for the competition is posted as a zip file on MEBDI competition web site (ML-Competition-2022-Data-Package.zip). A brief description of the dataset and variables is as follows:

- Current Population Survey, 2008-2015, CPS basic monthly and ASEC supplement.
- Downloaded from IPUMS. The panel is in *wide format* with corresponding person identifier `cpsid`.
- Each `cpsidp` forms one row of the dataset with all variable columns corresponding to year-t variables except for variable `y_tp1` (data counterpart of $y_{i,t+1}$) which corresponds to unemployment status (0 or 1) in year $t + 1$.
- The other variable definitions are given online in the IPUMS data dictionary listed below in Table 1.

Sample Details

The sample includes all individuals between ages 25 and 64 (inclusive) between 2008 and 2015. Employment status in t can be Employed, Unemployed or Not in Labor Force (E, U, or NILF). Individuals with missing employment status have been dropped (although they make up only 0.02% of the sample). To simplify the problem to a binary classification, in $t + 1$, E and NILF are treated as one group called NotU. So the classification problem is between U and NotU. Finally, the training sample is a randomly drawn subsample of the pooled sample over all years.

Table 1: CPS Data Description

Variable name	**Variable Definition**
y_tp1	year $t+1$, unemployment dummy ($y_{i,t+1}$. Equals 1 if `empstat` = 20, 21, 22 in year $t+1$.
cpsidp	year t, https://cps.ipums.org/cps-action/variables/cpsidp#codes_section
month	year t, https://cps.ipums.org/cps-action/variables/month#codes_section
year	year t, https://cps.ipums.org/cps-action/variables/year#codes_section
empstat	year t, https://cps.ipums.org/cps-action/variables/empstat#codes_section
earnweek	year t, https://cps.ipums.org/cps-action/variables/earnweek#codes_section
serial	year t, https://cps.ipums.org/cps-action/variables/serial#codes_section
hwtfinl	year t, https://cps.ipums.org/cps-action/variables/hwtfinl#codes_section
cpsid	year t, https://cps.ipums.org/cps-action/variables/cpsid#codes_section
statefip	year t, https://cps.ipums.org/cps-action/variables/statefip#codes_section
wtfinl	year t, https://cps.ipums.org/cps-action/variables/wtfinl#codes_section
relate	year t, https://cps.ipums.org/cps-action/variables/relate#codes_section
age	year t, https://cps.ipums.org/cps-action/variables/age#codes_section
sex	year t, https://cps.ipums.org/cps-action/variables/sex#codes_section
race	year t, https://cps.ipums.org/cps-action/variables/race#codes_section
marst	year t, https://cps.ipums.org/cps-action/variables/marst#codes_section
famsize	year t, https://cps.ipums.org/cps-action/variables/famsize#codes_section
nchild	year t, https://cps.ipums.org/cps-action/variables/nchild#codes_section
nativity	year t, https://cps.ipums.org/cps-action/variables/nativity#codes_section
hispan	year t, https://cps.ipums.org/cps-action/variables/hispan#codes_section
labforce	year t, https://cps.ipums.org/cps-action/variables/labforce#codes_section
occ1990	year t, https://cps.ipums.org/cps-action/variables/occ1990#codes_section
ind1990	year t, https://cps.ipums.org/cps-action/variables/ind1990#codes_section
classwkr	year t, https://cps.ipums.org/cps-action/variables/classwkr#codes_section
uhrsworkt	year t, https://cps.ipums.org/cps-action/variables/uhrsworkt#codes_section
ahrsworkt	year t, https://cps.ipums.org/cps-action/variables/ahrsworkt#codes_section
wkstat	year t, https://cps.ipums.org/cps-action/variables/wkstat#codes_section
educ	year t, https://cps.ipums.org/cps-action/variables/educ#codes_section
paidhour	year t, https://cps.ipums.org/cps-action/variables/paidhour#codes_section

Software and Packages

- You are allowed to use two compiled languages (Fortran or versions of C) or the following five high-level programming languages: Matlab, Python, R, Stata, or Julia, or any combination of them. You can also use outside libraries and packages that work with these languages as long as the source code is freely and easily available (for inspection). Libraries that are part of a language, such as Matlab's Statistics and Machine Learning Toolbox, are allowed. Because it is impossible to anticipate every contingency, if you are unsure, please email Kyle if the software is eligible before you start using them.

- If you are using any extra packages or libraries that are not part of the base programming language, they must be included with the source code or if that is not feasible, they must be described in the report with all the information necessary for the committee to access those packages and libraries.

2 Deliverables and Criteria for Win

Deliverables

To be admissible, your submission by May 13, 2022 must include the following:

1. A clearly written, concise **report** (ideally between 3 and 5 pages) that

   (a) contains a half-page executive summary of the method you propose to use and the 2 goodness of fit measures described below computed in your cross-validation analysis. This is not the final goodness of fit statistic for the competition, but it's a useful benchmark to include for completeness.

   (b) A detailed description of the ML algorithm(s) you propose, describing all the necessary modifications and all the specific choices you have made in every step. Someone who reads this report (and the committee will read it) should be able to write a code based on your description and replicate exactly what you did and get the same out-of-sample goodness of fit measures that you report.[1]

   (c) All the source code for your submission (in one zip file) and the executable program if you are using a compiled language. If you are using any extra packages or libraries that is not part of the base programming language, they must be included with the source code or if that is not feasible, they must be described in the report with all the information necessary for the committee to access those packages and libraries.

During the live run on May 17, teams will run their code on the test sample that will be shared at that time to produce the final goodness-of-fit statistic from their algorithm that will be used as the official figures for each team for the competition.

[1] For example: "We use the R code for elastic-net regularized linear models written by Robert Tibshirani et al available for download at https://cran.r-project.org/web/packages/glmnet/index.html." Then describe all the user-specific choices you made, etc.

Success Criteria for Out-Sample-Prediction

There are 2 performance measures of out-of-sample prediction that will be considered.

1. **Goodness of fit (GF)**: let $\overline{y}$ be the sample average of y_{t+1} in the training sample, which is equal to 0.0514. Let $\mathcal{T}$ be the set of individuals in the test sample. The GF measure rewards "correct" classifications in the following fashion:

$$\text{GF} = \frac{1}{2}\left[\frac{\sum_{i\in\mathcal{T}} \widehat{y}_{it+1} * I(y_{it+1}=1)}{\sum_{i\in\mathcal{T}} I(y_{it+1}=1)}\right] + \frac{1}{2}\left[\frac{\sum_{i\in\mathcal{T}}(1-\widehat{y}_{it+1}) * I(y_{it+1}=0)}{\sum_{i\in\mathcal{T}} I(y_{it+1}=0)}\right],$$

   where $\widehat{y}_{it+1}$ is the predicted unemployment status (0 or 1) and $I()$ is an indicator function. The two expressions in square brackets are the empirical counterparts of $\Pr(\widehat{y}_{it+1}=1|y_{it+1}=1)$ and $\Pr(\widehat{y}_{it+1}=0|y_{it+1}=0)$, respectively. Because the data is unbalanced by its nature (many more employed than unemployed in $t+1$), the weighting proportional to $\overline{y}$ ensures that the GF measure puts equal weight on predicting employment and unemployment in the overall sample. GF lies between 0 and 1, with a perfect classifier (with no misclassification) attaining the upper bound, GF = 1.

2. **Goodness of fit, Unemployment, GFU:** This measure is simply the first term of the GF measure in (1) without scaling:

$$\text{GFU} = \left[\frac{\sum_{i\in\mathcal{T}} \widehat{y}_{it+1} * I(y_{it+1}=1)}{\sum_{i\in\mathcal{T}} I(y_{it+1}=1)}\right].$$

Win Rule:

To win the competition, an algorithm must satisfy two conditions:

1. Deliver a GF measure for the test sample that is at least 0.005 (0.5 percentage points) higher than the next best entry.

2. Deliver a GF measure for the test sample that is at least 0.03 (3 percentage points) higher than a classifier based on a logit regression that includes log(earnweek), age dummies, and all the remaining allowable predictors in Table 1 without transformation or interaction terms. The classifier assigns $\widehat{y}_{it+1}=1$ when the predicted probability is greater than 1/2.

Tie-breaker: If there is more than one team within 0.005 of the highest GF measure, then the team with the highest GFU measure wins IF their GFU measure is at least 0.005 higher than the next best team. If a team has the highest GF and GFU scores and its combined GF + GFU measure is at least 0.005 higher than the second highest combined score, the top team wins. ***Tie After the Tiebreaker:*** If two or more teams are still tied after the tiebreaker, they will be declared joint winners and will share the prize.

The submissions will be evaluated by a committee of faculty members who will also have the authority to interpret the rules if needed.