MASTER IN COMPUTER SCIENCE

Machine Learning

Final report

MATR 22-507-297

Author: *Loan Strübi*

MATR 20-506-572

Author: *Aurélie Wasem*

Instructor:	Christos Dimitrakakis
Assistant:	Andreas Athanasopoulos
Date:	January 11, 2026
Github:	Project page

Contents

1 Scientific question

Many articles assert that “more schooling leads to higher income. ”*On average across OECD countries, adults with a short-cycle tertiary degree earn 17% more than those with upper secondary attainment. This earnings advantage rises to 39% for those with a bachelor’s degree and 83% for those with a master’s or doctoral degree.*” [7] We will empirically test this claim on a large census-like dataset by quantifying how education level relates to earning $>$ \$50k/year after controlling for confounders (age, hours worked, occupation, marital status, etc.).

Therefore, the question we will answer in this project is : Does education level (e.g., HS-grad, Bachelors, Masters…) increase the probability of having an income $>$ \$50k?

2 Simulation and real-data approach

From the outset, our objective was to study the relationship between education level and income by performing a supervised classification task. Finding a real dataset that exactly matched the variables and dependencies of our initial conceptual model turned out to be more difficult than anticipated. Although several public datasets include income and education information, they often impose constraints on variable definitions, availability, or relationships. Simulating a dataset allowed us to:

- explicitly encode the causal assumptions represented in the DAG;
- control confounding factors and dependencies between variables;
- verify that the expected relationships (e.g., education → occupation → income) could be recovered by a classification model under ideal conditions.

In this first phase, the goal was not to mirror reality perfectly, but rather to create a coherent and interpretable environment in which the behavior of the model could be understood and validated.
To complement the simulation and ground our analysis in real data, we ultimately selected the **Adult / Census Income** dataset (UCI repository, Kaggle mirror) [18]
Adopting this dataset required adapting our initial model: some variables had to be removed, others redefined, and certain dependencies adjusted to match the available features.
For this reason, the project naturally split into two complementary parts:

- a **synthetic data simulation**, used to validate our modeling assumptions and dependency structure under controlled conditions;
- an **empirical analysis on an existing real-world dataset**, used to assess how well these assumptions transfer to observed data.

Together, these two complementary approaches —simulation and empirical analysis— provide both interpretability and realism, strengthening the overall conclusions of the project.

3 Methodology

To address the research question and ensure robust model selection, we followed a systematic methodology encompassing dataset preparation, feature encoding, model evaluation, and metric-based comparison. This section outlines the key steps undertaken and provides justification for the choice of the final model.

3.1 Pipeline and Model Testing Process

We built a supervised classification pipeline to predict whether an individual's yearly income exceeds $50'000 (`income`: `<=50K` vs. `>50K`) on our different datasets. The main steps are:

- **Feature construction and encoding:**
  - The target variable is defined as a binary label:

    $$y = \mathbb{1}\{\texttt{income} = \texttt{">50K"}\}.$$

  - The feature matrix X consists of all remaining columns.
  - Categorical predictors are transformed using one-hot encoding via `pd.get_dummies` with `drop_first=True` to avoid dummy-variable collinearity, all resulting features are cast to `float`.
- **Dataset splitting:**
  - The dataset is split into a training and a test set using `train_test_split` with a **20% test size**.
  - We use `random_state=42` for reproducibility and `stratify=y` to preserve the class balance in both splits.
- **Model fitting:**
  - We compare several standard classification algorithms, all using their scikit-learn implementations:
    - ∗ k-Nearest Neighbours (`KNeighborsClassifier`),
    - ∗ Linear Support Vector Machine (`LinearSVC`),
    - ∗ Gaussian Naive Bayes (`GaussianNB`),
    - ∗ Random Forest (`RandomForestClassifier`),
    - ∗ Decision Tree (`DecisionTreeClassifier`),
    - ∗ Gradient Boosting (`GradientBoostingClassifier`),
    - ∗ AdaBoost (`AdaBoostClassifier`).
  - Each model is trained on the training set $(X_{\text{train}}, y_{\text{train}})$ using the default or standard hyperparameters chosen for a fair baseline comparison.
- **Prediction:**

  - For each trained model, we compute predictions on the held-out test set:

$$\hat{y}_{\text{test}} = f(X_{\text{test}}).$$

  - When available, we also obtain a *score* for the positive class (`>50K`) via either `predict_proba` (class probability) or `decision_function` (decision score). These scores are used for ROC–AUC computation.

- **Metric calculation:**
  - To evaluate and compare models, we compute the following metrics:
    * **Test accuracy:** the proportion of correctly classified individuals on the test set.
    * **F1-score:** the harmonic mean of precision and recall for the positive class (`>50K`), capturing performance under class imbalance.
    * **ROC–AUC:** the area under the ROC curve, based on predicted probabilities or decision scores when available, this measures the ranking quality of the classifier.
    * **Cross-validated accuracy:** the mean accuracy from 5-fold cross-validation on the full dataset (X, y) using `cross_val_score`, providing a more stable estimate of generalisation performance.
    * **Bootstrap confidence interval on accuracy:** for each model, we estimate a 95% confidence interval on the training accuracy using a custom `bootstrap_accuracy` function. This function repeatedly resamples the training set with replacement, refits the model, records the accuracy on each bootstrap sample, and returns the 2.5th and 97.5th percentiles of the resulting distribution. This gives an empirical measure of the variability and reliability of the classifier.

The aggregated metrics for all models are stored in a Pandas dataframe (`results_df`), which we use to identify the best-performing classifier according to test performance and robustness (cross-validation and bootstrap intervals).

4 Simulation

To formalize our assumptions about the underlying causal structure, we first designed an initial conceptual model describing the expected dependencies between variables, illustrated by the Directed Acyclic Graph (DAG) shown in Figure 6.

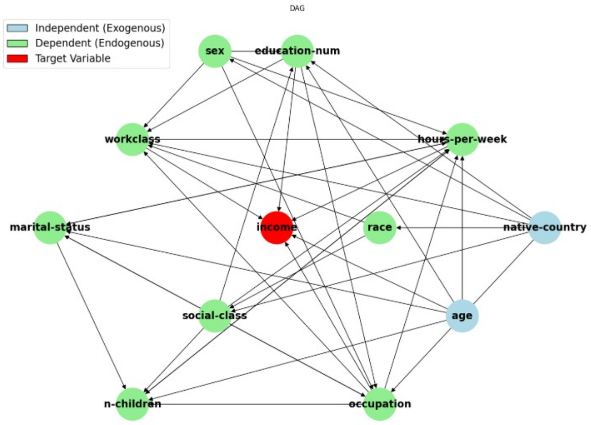


Figure 1: Directed Acyclic Graph (DAG) of the initial conceptual model

4.1 Choice of variables in the initial model

The selection of variables for the simulation was guided by empirical evidence and official statistics published by the Swiss Federal Statistical Office (OFS) [1–6]. These sources informed both the choice of variables (e.g., education, occupation, workclass, marital status, working hours) and the qualitative relationships between them.

We initially considered requesting a custom dataset from the OFS, which offers this service free of charge. However, after discussions with an OFS specialist in education data, we were informed that assembling a dataset combining multiple departments and sensitive variables would require several months, as well as the signing of privacy and security agreements. Given the time constraints of this project, this option was not feasible.

4.1.1 Describe the main variables

Independent Variables (Inputs):

- **Age (years):**
  - **Definition:** Age of the individual, ranging from 18 to 65.
  - **Type:** Discrete quantitative.
- **Sex:**
  - **Definition:** Biological sex category used in the simulation (`Male`, `Female`).
  - **Type:** Nominal categorical (binary).
- **Native Country:**
  - **Definition:** Country/region of origin (`Switzerland`, neighboring countries, and aggregated regions such as `Africa`, `Asia`, `The Americas`).
  - **Type:** Nominal categorical.
- **Race (proxy):**

  - **Definition:** Coarse proxy category derived from native country (e.g., `White`, `Black`, `Asian-Pac-Islander`, `Latin`, `Other`).
  - **Type:** Nominal categorical.

- **Social Class:**
  - **Definition:** Simplified socioeconomic category used as an intermediate variable (`working`, `middle`, `upper`).
  - **Type:** Ordinal categorical.

- **Education-num (years of education):**
  - **Definition:** Integer proxy for educational attainment (years), ranging from 9 to 18.
  - **Type:** Discrete quantitative.

- **Occupation:**
  - **Definition:** Job category (e.g., `Exec-managerial`, `Prof-specialty`, `Sales`, etc.).
  - **Type:** Nominal categorical.

- **Workclass:**
  - **Definition:** Employment arrangement (e.g., `Private`, `Self-emp-inc`, `Federal-gov`, `Without-pay`, `Never-worked`).
  - **Type:** Nominal categorical.

- **Marital-status:**
  - **Definition:** Marital situation (e.g., `Never-married`, `Married-civ-spouse`, `Divorced`, etc.).
  - **Type:** Nominal categorical.

- **Number of children (n-children):**
  - **Definition:** Number of children in the household, discretized to values in $\{0, 1, 2, 3, 4\}$.
  - **Type:** Discrete quantitative.

- **Hours-per-week:**
  - **Definition:** Weekly working hours (integer, bounded between 0 and 70).
  - **Type:** Discrete quantitative.

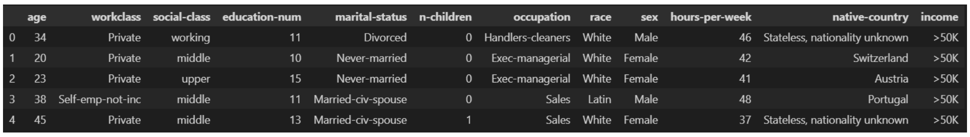
- **Income amount :**
  - **Definition:** Simulated annual income as an integer value (CHF-like), generated from education, age, occupation, workclass, and hours, plus random noise.
  - **Type:** Discrete quantitative.

Dependent Variable (Output):

- **Income (binary class):**
  - **Definition:** Target variable indicating whether annual income is above \$50k.
  - **Classes:** `<=50K` and `>50K`.
  - **Type:** Binary categorical (classification target).

4.2 Generation process of the synthetic variables

This section documents how each variable in our Swiss census-like synthetic dataset is generated, and how dependencies between variables are injected. We follow a dependency-aware sampling approach: we first sample *exogenous* variables (assumed independent), then generate *endogenous* variables by modifying either categorical logits (softmax sampling) or numeric means as a function of already-sampled variables.



	age	workclass	social-class	education-num	marital-status	n-children	occupation	race	sex	hours-per-week	native-country	income
0	34	Private	working	11	Divorced	0	Handlers-cleaners	White	Male	46	Stateless, nationality unknown	>50K
1	20	Private	middle	10	Never-married	0	Exec-managerial	White	Female	42	Switzerland	>50K
2	23	Private	upper	15	Never-married	0	Exec-managerial	White	Female	41	Austria	>50K
3	38	Self-emp-not-inc	middle	11	Married-civ-spouse	0	Sales	Latin	Male	48	Portugal	>50K
4	45	Private	middle	13	Married-civ-spouse	1	Sales	White	Female	37	Stateless, nationality unknown	>50K

Figure 2: head of the fully numeric simulated dataset

The output is a dataset of `n` samples with generated values for each 12 variables and no missing values.

4.2.1 Native Country

Definition

The **Native Country** variable represents the country/region of origin of an individual. It is a categorical variable taking values in the predefined vocabulary `COUNTRIES`, which includes Switzerland and aggregated regions (e.g., Africa, Asia, The Americas).

Generation Process

Random Sampling Native country is sampled first using a prior probability distribution calibrated to be Switzerland-dominant:

- **Switzerland:** approximately 73%
- **Other countries/regions:** approximately 27% distributed across neighboring countries and broader regions.

This ensures a dataset that resembles the structure of a Swiss population while still containing non-Swiss groups.

Assumptions

- The prior distribution over countries is fixed and does not depend on other variables (exogenous).
- Aggregated regions (e.g., Africa, Asia) are used to simplify the simulation.

4.2.2 Age

Definition

The **Age** variable represents the age of an individual in years, ranging from 18 to 65. It influences several downstream variables such as marital status, number of children, education completion, hours worked, and ultimately income.

Generation Process

Random Sampling Ages are generated using a discrete uniform distribution over the range 18–65. This provides a balanced representation across the working-age population.
Why Uniform Distribution? We use a uniform distribution to avoid introducing an additional age structure in the simulation. This keeps the age variable neutral and lets the dependencies (e.g., age → marital/children/hours) drive the observed patterns.

Assumptions

- The age distribution is uniform over 18–65 in this simulation.

4.2.3 Sex

Definition

The **Sex** variable is a binary categorical variable with values `Male` and `Female`. It is used as a mild modifier in education and working hours, and also enters the income generation as a small aggregate adjustment.

Generation Process

Random Sampling Sex is sampled independently using fixed probabilities:

- **Male:** 52%
- **Female:** 48%

Assumptions

- Sex is treated as binary for simplicity.
- Sex is exogenous and independent of other variables in the simulation.

4.2.4 Race

Definition

The **Race** variable is a categorical proxy variable (e.g., White, Black, Asian-Pac-Islander, Latin, Other). In our simulation, it is not sampled independently; instead, it is derived from *native country* using a coarse mapping.

Generation Process

Deterministic / Rule-Based Mapping Race is generated from native country through a simple rule-based function:

- `Asia` → `Asian-Pac-Islander`
- `Africa` → `Black`

- `Spain/Portugal` → `Latin`
- `The Americas` → sampled among `White/Black/Latin` with fixed probabilities
- Most European countries → `White`

Assumptions

- Race is a proxy derived from country and does not reflect real demographic complexity.
- The mapping is intentionally coarse and only exists to induce a dependency structure.

4.2.5 Social Class

Definition

The **Social Class** variable represents a simplified socioeconomic position with three categories: `working`, `middle`, and `upper`. It is an intermediate latent-like variable used to influence education, occupation, workclass, and family structure.

Generation Process

Dependency-Aware Softmax Sampling Social class is sampled from a baseline prior (working/middle/upper) using a logit model. Logits are adjusted based on:

- **Foreign origin:** increases probability of `working`, decreases `upper`.
- **Race:** small additive boosts to `upper` (kept mild).

A softmax converts logits to probabilities, then one category is sampled per individual.

Assumptions

- Foreign origin is associated with slightly lower social-class category probabilities (heuristic).
- Race effects are intentionally kept small to avoid hard-coded strong bias.

4.2.6 Education Level (education-num)

Definition

The **Education-num** variable is an integer proxy for educational attainment (years of education), ranging from 9 to 18. It is a core variable for our scientific question because it directly affects occupation and income.

Generation Process

Mean-Based Sampling with Noise We compute an expected education value μ for each individual, then sample around it:

- **Social class:** working ≈ 12, middle ≈ 14, upper ≈ 16 years.
- **Age:** individuals under 25 receive a downward adjustment (not completed studies).
- **Sex:** a tiny upward adjustment for females (+0.1) is used (kept minimal).
- **Native country:** non-CH and non-EU-15 origins are slightly reduced.

Finally, we add Gaussian noise and round to an integer within [9,18].

Assumptions

- Education is strongly driven by social class in this simulation.
- The under-25 adjustment models incomplete education.
- Country effects are simplified to a EU-15 vs non-EU-15 distinction.

4.2.7 Occupation

Definition

The **Occupation** variable is categorical (e.g., `Exec-managerial`, `Prof-specialty`, etc.). It is generated as a function of education, social class, sex, and country, and it strongly impacts income via multiplicative wage factors.

Generation Process

Logit Shifts + Softmax Sampling We start from a Swiss-ish prior distribution over occupations. We then adjust logits:

- **Education:** increases probabilities of professional/managerial categories and decreases low-skill categories.
- **Social class:** shifts mass toward higher-status occupations for `upper`.
- **Foreign origin:** mild penalties on some high-status occupations.
- **Sex:** small adjustments for a few categories (kept small).

A softmax is applied and one occupation is sampled per individual.

Assumptions

- Education is the primary driver of occupation in this simulation.
- Sex and origin effects are mild and only serve to create variability.

4.2.8 Workclass

Definition

The **Workclass** variable describes the employment arrangement (e.g., `Private`, `Self-emp-inc`, `Federal-gov`, etc.). It is generated from occupation, social class, and native country and later influences working hours and income.

Generation Process

Dependency-Aware Softmax Sampling We start from a prior distribution and shift logits:

- **Occupation:** executive and professional roles increase self-employment and private sector probabilities.
- **Social class:** upper increases `Self-emp-inc`; working increases `Private`.
- **Foreign origin:** reduces government categories (`Federal/Local/State-gov`).

Then sample with softmax.

Assumptions

- Workclass is partly explained by occupation and social class.
- Non-Swiss origin reduces the probability of government categories (heuristic).

4.2.9 Marital Status

Definition

The **Marital-status** variable (e.g., `Never-married`, `Married-civ-spouse`, etc.) is generated in two passes: a main pass driven by age and social class, then a tiny adjustment to create a weak dependency from hours worked.

Generation Process

Pass 1: Age-Driven Softmax Sampling We start from priors, then:

- Older age increases probability of `Married-civ-spouse`
- Older age decreases probability of `Never-married`
- Minor effects from social class and a small occupation-based boost

Pass 2: Small Post-Adjustment using Hours After working hours are generated, we apply a small probability of switching:

- `Never-married` → `Married-civ-spouse` for ages 28–55 with high hours (≥ 45)
- `Married-civ-spouse` → `Never-married` for young individuals with very low hours (≤ 10)

These probabilities are low and only used to introduce a weak feedback pattern.

Assumptions

- Age is the dominant driver of marital status.
- The hours-based adjustment is intentionally small.

4.2.10 Number of Children (n-children)

Definition

The **n-children** variable is a discrete variable in $\{0, 1, 2, 3, 4\}$. It depends mainly on age and marital status, with small effects from social class and country.

Generation Process

Categorical Sampling with Age/Marital Shifts We start from a baseline distribution over $\{0, \ldots, 4\}$ and shift logits:

- **Age:** older individuals are more likely to have 2–4 children.
- **Marital status:** married increases 2+; never-married increases 0.
- **Social class:** small shifts (working slightly higher for 2–3).
- **Foreign origin:** tiny upward adjustments.

Then we sample one category.

Assumptions

- Age and marital status explain most of the variation in number of children.
- Country and social effects are intentionally mild.

4.2.11 Hours per Week

Definition

The **Hours-per-week** variable is the number of hours worked per week (0–70). It depends on age, workclass, social class, sex, and number of children.

Generation Process

Mean-Based Sampling with Noise We compute a mean μ per individual and sample around it:

- **Age:** quadratic "mid-life hump" (peak around working prime years).
- **Workclass:** self-employed increases hours; government slightly decreases; without-pay $\rightarrow 8$; never-worked $\rightarrow 0$.
- **Social class:** small upward shift for working and upper.
- **Children and sex:** more children reduces hours more strongly for females.

We then add Gaussian noise and clip to [0,70].

Assumptions

- Workclass is a major driver of working hours.
- Children reduce hours, especially for females (heuristic average effect).

4.2.12 Income Amount and Income Class

Definition

We generate first a numeric **Income Amount** (annual CHF-like value), then convert it into the categorical target **Income**:

- **Income Amount:** integer annual income ≥ 0
- **Income class (target):** `<=50K` vs `>50K`

Generation Process

Deterministic Wage Model + Noise Income amount is computed as:

- A base hourly wage
- **Education premium:** increases with education years (capped)
- **Age premium:** quadratic hump (career peak)
- **Sex adjustment:** small constant shift (kept modest)
- Multiplicative factors for **occupation** and **workclass**

- Additive Gaussian noise (to create realistic overlap between classes)

This produces a numeric annual income.
Thresholding into a Binary Target Finally, the numeric income is mapped into the target class:

$$\texttt{income} = \begin{cases} \texttt{>50K} & \text{if income\_amount} > 50\,000 \\ \texttt{<=50K} & \text{otherwise} \end{cases}$$

Assumptions

- Education and occupation are key drivers of income in the simulation.
- Race effects are kept near-zero to avoid encoding strong bias.
- Thresholding at 50K produces a classification setting with overlap (non-separable classes).

4.3 Evaluation of different models on the simulated dataset

Table 1 reports the performance of the evaluated classifiers on the synthetic dataset.

Table 1: Performance of the evaluated models on the simulated dataset.

Model	Test Acc.	Test F1	ROC–AUC	CV Acc.	Acc CI 2.5	Acc CI 97.5
KNN	0.942	0.969	0.838	0.944	0.967	0.973
Linear SVM	0.963	0.980	0.978	0.960	0.960	0.970
Naive Bayes	0.784	0.871	0.909	0.750	0.603	0.836
Random Forest	0.961	0.979	0.975	0.961	1.000	1.000
Decision Tree	0.955	0.976	0.810	0.950	1.000	1.000
Gradient Boosting	**0.966**	**0.982**	**0.985**	0.967	0.976	0.981
AdaBoost	0.947	0.972	0.976	0.946	0.943	0.951

Several observations can be drawn from these results:

- **Very high overall performance across models.** Most classifiers achieve test accuracies above 94%, which is expected given that the simulated dataset follows a coherent and relatively low-noise generative process. The underlying signal linking predictors to income is therefore strong and learnable.
- **Gradient Boosting as the best-performing model.** Gradient Boosting achieves the highest test accuracy (0.966), F1-score (0.982) and ROC–AUC (0.985). This indicates excellent discrimination between the two income classes and confirms that ensemble methods are particularly well suited to capture the non-linear dependencies encoded in the simulation.
- **Linear models perform surprisingly well.** The Linear SVM also performs extremely well (ROC–AUC 0.978), suggesting that a large part of the signal in the synthetic data is close to linearly separable. This reflects the fact that income was generated from mostly monotonic effects of education, age and working hours.

- **Overfitting in single-tree models.** Both the Decision Tree and Random Forest reach perfect or near-perfect bootstrap confidence intervals (CI upper bounds equal to 1.0), while their test ROC–AUC values are slightly lower. This gap indicates overfitting, which is expected when flexible tree-based models are applied to a dataset with strong deterministic patterns.
- **Stability of ensemble methods.** For Gradient Boosting and AdaBoost, cross-validated accuracies and bootstrap confidence intervals are close to test accuracies, indicating stable and robust performance across different splits.

4.3.1 Choice of the final model

Based on these results, **Gradient Boosting** is selected as the final model for the simulated dataset. It consistently dominates the other classifiers across all metrics while maintaining strong stability. The Linear SVM is retained as a useful reference model, but Gradient Boosting provides the best balance between flexibility and generalisation in the simulated setting.

4.4 Feature importance analysis on the simulated data

Because the simulation explicitly encodes causal relationships, feature importance analysis provides a useful sanity check: important variables identified by the model should align with the data-generating process.

4.4.1 Impurity-based importance (dummy level)

Tree-based methods provide a built-in measure of feature importance based on the total decrease in impurity (here, the reduction in classification loss) each feature contributes across all splits in the ensemble. For the Gradient Boosting model, we extract the vector of importances `model.feature_importances_` and associate it with the corresponding dummy or numeric feature names:

$$\texttt{Importance}(\texttt{Feature}) = \texttt{model.feature\_importances\_}.$$

Table 2 reports the top 20 important features at this dummy level.
At this level, `hours-per-week` clearly dominates, followed by several occupation categories and `age`. The variable `education-num` also appears among the top contributors, confirming that education plays a direct role in income determination in the simulation, but remains mediated by downstream variables such as occupation and working hours.

4.4.2 Impurity-based importance aggregated by original variable

Because many original variables are expanded into several dummies, dummy-level importances can be difficult to interpret. To obtain a more meaningful ranking, we aggregate importances by *original* variable. We define a mapping `original_var_name` that extracts the variable name before the first underscore (e.g. `"workclass_Private"` ↦ `"workclass"`, `"education-num"` ↦ `"education-num"`), and group the importance scores accordingly:

$$\texttt{Importance}(\text{Variable}) = \sum_{\text{dummies of Variable}} \texttt{Importance}(\text{Feature}).$$

Table 2: Top features according to impurity-based importance (dummy level) on the simulated dataset.

Feature	Importance
hours-per-week	0.437864
occupation_Other-service	0.123699
occupation_Handlers-cleaners	0.114992
age	0.096476
occupation_Farming-fishing	0.067494
occupation_Priv-house-serv	0.039654
occupation_Prof-specialty	0.025926
education-num	0.023986
occupation_Exec-managerial	0.019143
sex_Male	0.018721
occupation_Machine-op-inspct	0.005633
social-class_working	0.005413
workclass_Private	0.005156
occupation_Tech-support	0.004089
occupation_Sales	0.002888
marital-status_Widowed	0.001666
workclass_Local-gov	0.001354
native-country_Kosovo	0.001225
occupation_Transport-moving	0.001132
native-country_Ukraine	0.000989

The resulting aggregated importances are summarised in Table 3.

Table 3: Impurity-based importance aggregated by original variable on the simulated dataset.

Variable	Importance
hours-per-week	0.437864
occupation	0.404651
age	0.096476
education-num	0.023986
sex	0.018721
workclass	0.007213
social-class	0.005413
native-country	0.003263
marital-status	0.001666
race	0.000493
n-children	0.000256

- **Hours-per-week** and **occupation** are by far the most important predictors.
- **Age** plays a secondary but still substantial role.

- **Education level** contributes positively, but less directly than working hours and occupation, reflecting the hierarchical structure of the simulation.

This ranking is consistent with the DAG: education influences income mainly through occupation and working hours, rather than acting as a dominant direct predictor.

4.4.3 Permutation importance aggregated by original variable

To validate these findings with a model-agnostic measure, we also compute *permutation importance* on the test set. For each feature, we repeatedly permute its values in X_{test}, recompute the accuracy, and measure the average drop in performance. We then aggregate these drops by original variable using the same `original_var_name` mapping. The resulting permutation importances are shown in Table 4.

Table 4: Permutation importance aggregated by original variable on the simulated dataset (scoring: test accuracy).

Variable	Perm_Importance
`hours-per-week`	0.041700
`occupation`	0.038867
`age`	0.014433
`sex`	0.004767
`education-num`	0.004067
`social-class`	0.000367
`marital-status`	0.000000
`n-children`	0.000000
`race`	0.000000
`workclass`	-0.000200
`native-country`	-0.000367

Permuting `hours-per-week` or `occupation` leads to the largest drops in test accuracy (around 4 percentage points), followed by `age` and `education-num`. In contrast, variables such as `marital-status`, `race`, `native-country` and `n-children` have negligible or even slightly negative permutation importance.
This confirms that the model primarily relies on the variables that were explicitly designed to drive income in the simulation, while demographic variables with weaker or indirect effects play a marginal role.

4.4.4 Interpretation with respect to the scientific question

The synthetic data was generated according to a predefined causal structure in which education level influences income both directly and indirectly, mainly through occupation and working hours. The feature importance analysis allows us to verify whether the trained models recovered the intended structure.
Both impurity-based and permutation importance rankings are strongly aligned with the simulation design. Variables that were explicitly constructed as the main drivers of income
—`hours-per-week` and `occupation`—emerge as the most influential predictors. This is

consistent with the income generation mechanism, where education affects income primarily by increasing access to higher-paying occupations and by influencing work intensity.
Education level (`education-num`) appears as a secondary but non-negligible predictor. Its relatively lower importance compared to `hours-per-week` and `occupation` is expected, as its effect on income was deliberately modeled to be mostly mediated rather than purely direct. This confirms that the model does not spuriously overemphasize education, but instead captures its role within the hierarchical dependency structure encoded in the DAG.
Overall, the correspondence between simulated dependencies and learned feature importances provides a strong internal validation of the simulation. It confirms that **education level increases the probability of earning more than $50K a year**, primarily through indirect pathways involving occupation and working hours. The simulation therefore serves as a controlled benchmark, against which deviations observed in real-world data can later be meaningfully interpreted.

5 Dataset and Variables choice

5.1 Dataset selection

After reseach we've decided to settled down for the Adult / Census Income dataset (UCI/Kaggle mirror)[1]. We had to adapt our model to this new dataset, dropping some variables, adopting new ones and changing some of their relationships.

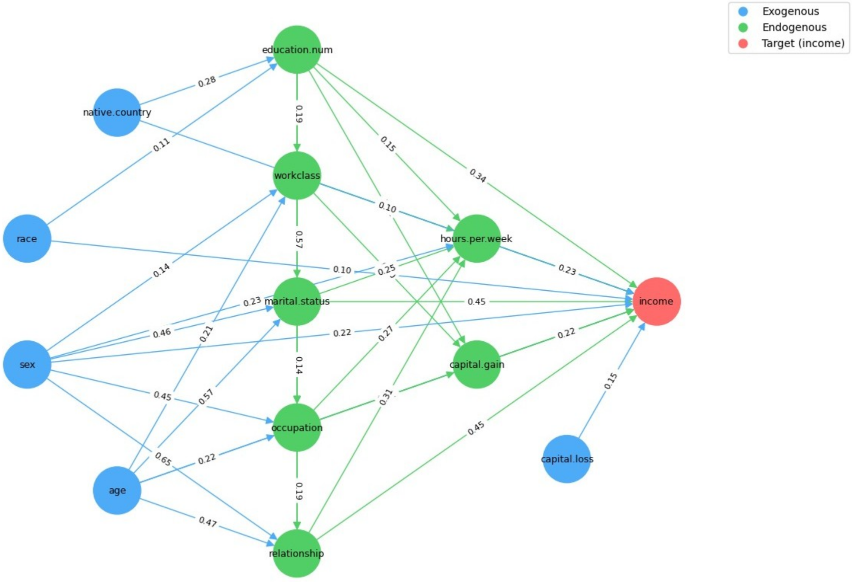


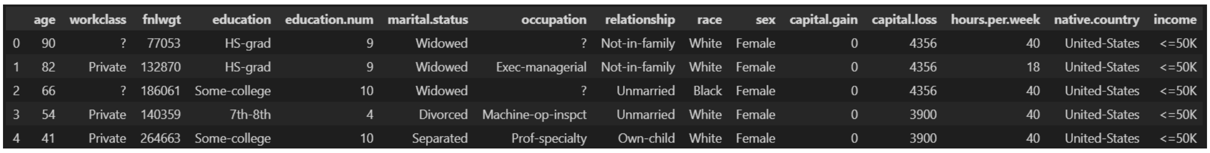
Figure 3: New DAG for the UCI/Kaggle dataset

5.2 Main changes from simulation

- `n-children` : removed, not in the Adult/Census dataset.

[1] UCI Machine Learning, Adult Census Income, Kaggle, accessed October 22, 2025, https://www.kaggle.com/datasets/uciml/adult-census-income

- `capital.loss` and `capital.gain` : added, we thought their impact on income was too important to not be used in our model.
- Changes in variables roles : `race` and `sex` are now exogenous



	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country	income
0	90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female	0	4356	40	United-States	<=50K
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female	0	4356	18	United-States	<=50K
2	66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female	0	4356	40	United-States	<=50K
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female	0	3900	40	United-States	<=50K
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female	0	3900	40	United-States	<=50K

Figure 4: Dataset Adult Census Income

Here we have the head of our new dataset. We can already see missing values,we'll go over how we managed these in the pipeline.

5.3 Variable choice

- `age` — integer age in years.
- `workclass` — type of employer / employment status (e.g., Private, Self-emp-not-inc, Self-emp-inc, Federal-gov,...).
- `education-num` — numeric encoding of education (approx. years of schooling).
- `marital.status` — marital situation (e.g., Never-married, Married-civ-spouse, Divorced,...).
- `occupation` — job category (e.g., Tech-support, Craft-repair, Sales,...).
- `relationship` — role within one's household/family (e.g., Husband, Wife, ...).
- `race` — race category (e.g., White, Black, Asian-Pac-Islander,...).
- `sex` — Male / Female.
- `capital.gain` — capital gains in the previous year (numerical; many zeros).
- `capital.loss` — capital losses in the previous year (numerical; many zeros).
- `hours.per.week` — usual hours worked per week (integer).
- `native.country` — country of origin (United-States plus many others).
- `income` — target label indicating earnings: $\leq$`50K` or >`50K`.

Two colums are dropped at this stage:

- `education`: a redundant categorical version of the numeric education level.
- `fnlwgt`: the sampling weight, which we do not use in our predictive modelling.

5.4 Pipeline

5.4.1 Handling missing values

In the raw file, missing values are encoded as the string “?” in several categorical columns. We first replace all occurrences of “?” with `NaN` and then inspect the amount of missing data per column. Three columns contain missing values:

- `workclass`: 1 836 missing values (5.64%),
- `occupation`: 1 843 missing values (5.66%),
- `native.country`: 583 missing values (1.79%).

To impute these missing entries, we use a *conditional mode* strategy implemented in a helper function `fill_with_conditional_mode`. The idea is to fill each missing value in a target column using the most frequent category among “similar” rows, defined by a set of conditioning features.

More precisely, for a target categorical column Y and a set of conditioning columns $X_1, \ldots, X_k$:

1. Compute the global mode of Y (most frequent category) to use as a fallback.
2. For each row i where Y_i is missing:
   (a) Select all rows j in the dataframe such that:
   - Y_j is not missing, and
   - for every conditioning column X_ℓ, the value $X_{\ell j}$ matches $X_{\ell i}$ (when $X_{\ell i}$ is not missing).

   (b) If at least one such candidate exists, fill Y_i with the empirical mode of Y among these candidates.

   (c) Otherwise, fill Y_i with the global mode of Y.

We apply this imputation scheme to the three columns with missing values:

- **Workclass.** We impute `workclass` based on a socio-demographic and occupational profile, using the following conditioning variables:
  {`education.num`, `marital.status`, `occupation`, `relationship`, `race`, `sex`, `income`}.
- **Occupation.** We impute `occupation` using:
  {`education.num`, `marital.status`, `workclass`, `relationship`, `race`, `sex`, `income`}.
  This couples the inferred occupation with the (possibly imputed) workclass and the same socio-demographic variables.
- **Native country.** We impute `native.country` primarily from race, sex and income, complemented by education and marital status:
  {`race`, `sex`, `income`, `education.num`, `marital.status`}.

After this step, all missing values in the dataset are imputed.

5.4.2 Exploratory analysis

We perform an initial exploratory data analysis (EDA) to inspect the distributions of both numeric and categorical variables.

Numeric variables. For the main numeric features
{`age`, `education.num`, `capital.gain`,`capital.loss`, `hours.per.week`},
we plot histograms to visualise their empirical distributions.

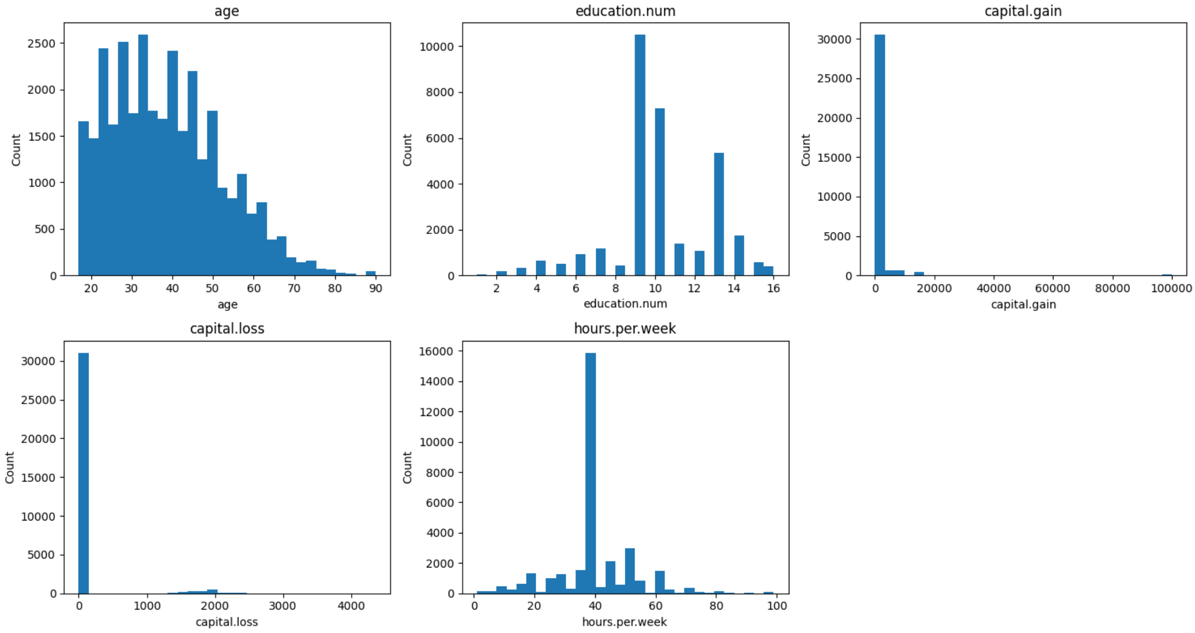


Figure 5: Numerical variables distributions

We also compute summary statistics via `df.describe()`, which yields:

	age	education.num	capital.gain	capital.loss	hours.per.week
count	32 561	32 561	32 561	32 561	32 561
mean	38.58	10.08	1 077.65	87.30	40.44
std	13.64	2.57	7 385.29	402.96	12.35
min	17	1	0	0	1
25%	28	9	0	0	40
50%	37	10	0	0	40
75%	48	12	0	0	45
max	90	16	99 999	4 356	99

These summaries show, for instance, that `capital.gain` and `capital.loss` are extremely skewed: most observations are zero, with a small number of very large values.

Categorical variables. For the main categorical features
{`marital.status`, `occupation`, `workclass`, `relationship`, `sex`, `race`,
`native.country`},

we compute plot bar charts (sorted by decreasing frequency). This allows us to identify dominant categories (e.g. the majority of individuals are from the United States, and `Male`/`Female` are balanced but not perfectly symmetric) and to check for very rare categories.

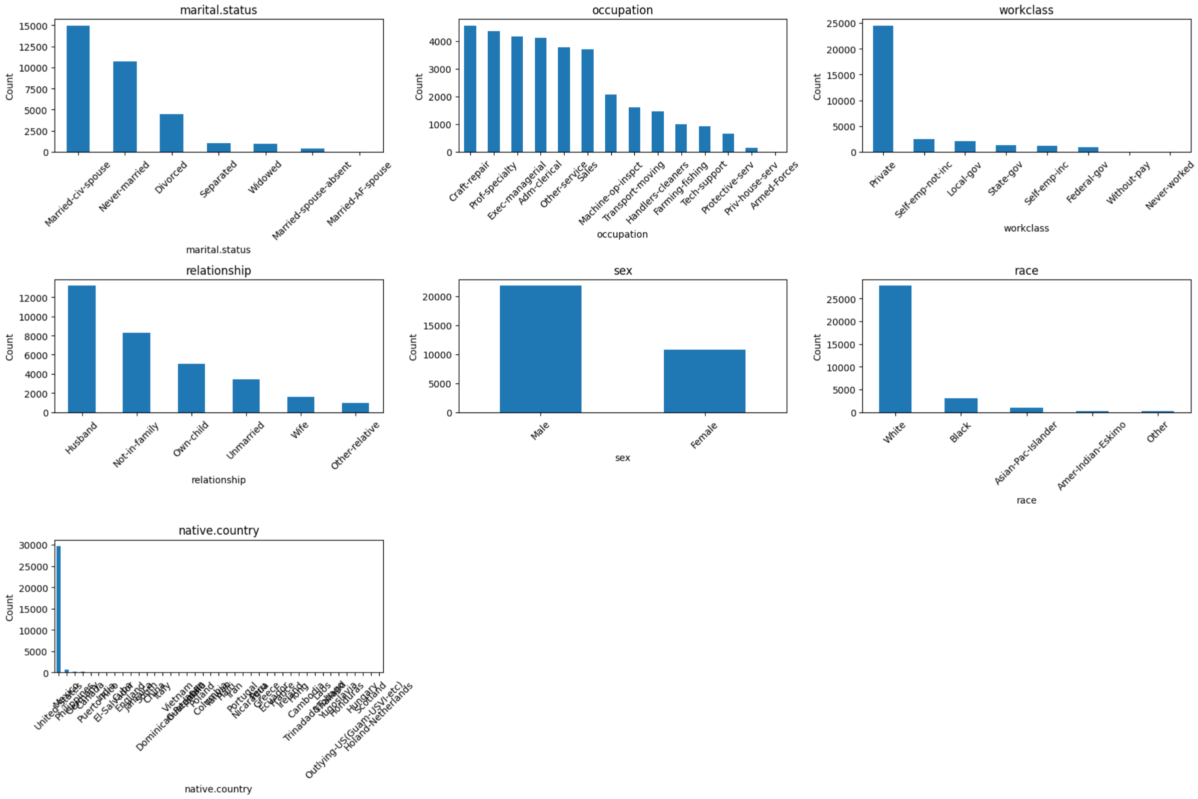


Figure 6: Categorical variables distributions

5.4.3 Outlier screening

To screen for potential outliers in the numeric features, we use the standard *1.5 × IQR rule*. For each numeric column X we compute:

$$Q_1 = \text{25th percentile of } X, \quad Q_3 = \text{75th percentile of } X, \quad \text{IQR} = Q_3 - Q_1,$$

and define the lower and upper fences as

$$L = Q_1 - 1.5 \cdot \text{IQR}, \qquad U = Q_3 + 1.5 \cdot \text{IQR}.$$

Observations outside the interval $[L, U]$ are flagged as potential outliers.
Applying this rule to the numeric subset {`age, education.num, capital.gain, capital.loss, hours.per.week`}, we obtain the following results:

- **`age`.** We obtain $L \approx -2$ and $U \approx 78$. No values fall below the lower fence, while 143 observations have `age` > 78. These correspond to very old individuals (up to 90 years) but remain realistic values in a census dataset. Removing them would artificially truncate the upper tail of the age distribution; we therefore keep them.

- **`education.num`.** The interval is approximately $[4.5, 16.5]$, and 1 198 observations fall below the lower fence. However, `education.num` is a discrete code for education level: low values simply represent individuals with low education, not data errors. Dropping them would systematically remove a specific socio-economic group and introduce selection bias. We thus retain all observations.

- **`capital.gain` and `capital.loss`.** For both variables, $Q_1 = Q_2 = Q_3 = 0$, so the IQR is zero. In this case the 1.5 × IQR rule is not informative: almost all observations are equal to 0, and only a small fraction have strictly positive values (some of them very large, e.g. 99 999). These large gains and losses are precisely the signal we would want for income prediction. Removing or capping them would destroy important information, so we keep all values.

- **`hours.per.week`.** The IQR rule yields $L \approx 32.5$ and $U \approx 52.5$, flagging 5 516 observations below the lower fence and 3 492 above the upper fence. This means that about 27% of the dataset would be labelled as "outliers". In reality, these values correspond to real variability in working time (part-time work, long work weeks, multiple jobs) rather than anomalies. Filtering them out would heavily distort the relationship between working hours and income. We therefore treat them as valid observations.

Overall, we use the IQR rule as a *diagnostic tool* to understand the tails of the numeric distributions, but we do not remove any data points solely on this basis. Since all flagged values are plausible in the context of a census and carry meaningful information for the prediction task, we decide to keep the full dataset and simply document the presence of extreme values instead of filtering them out.

5.5 Evaluation of Different Models

Table 5 reports the test performance of all evaluated models on the held-out test set.

Table 5: Performance of the evaluated models on the Adult Census Income dataset.

Model	Test Acc.	Test F1	ROC–AUC	CV Acc.	Acc CI 2.5	Acc CI 97.5
KNN	0.840	0.666	0.869	0.780	0.918	0.923
Linear SVM	0.852	0.660	0.900	0.821	0.846	0.854
Naive Bayes	0.820	0.670	0.886	0.810	0.815	0.828
Random Forest	0.846	0.658	0.894	0.800	0.984	0.986
Decision Tree	0.823	0.624	0.769	0.766	0.984	0.986
Gradient Boosting	**0.861**	**0.675**	**0.916**	0.810	0.868	0.876
AdaBoost	0.851	0.631	0.902	0.807	0.853	0.857

Several observations can be made from these results:

- **Overall best performer.** Gradient Boosting achieves the highest test accuracy (0.861), the best F1-score (0.675) and the highest ROC–AUC (0.916). This indicates that it not only predicts the correct class most often, but also provides the best trade-off between precision and recall for the positive class (`>50K`) and the best ranking quality among all classifiers.

- **Linear SVM and AdaBoost as strong competitors.** The Linear SVM reaches a test accuracy of 0.852 and ROC–AUC of 0.900 with the highest cross-validated accuracy (0.821), suggesting good and stable generalisation. AdaBoost also performs well (test accuracy 0.851, ROC–AUC 0.902), but with a lower F1-score (0.631), indicating that it is slightly less balanced on the minority class than Gradient Boosting.

- **Naive Bayes and KNN.** Naive Bayes obtains a relatively high F1-score (0.670) and ROC–AUC of 0.886 despite its simplicity, making it an interesting baseline model. KNN achieves reasonable performance (test accuracy 0.840, ROC–AUC 0.869) but is dominated by Gradient Boosting and Linear SVM on all key metrics.

- **Tree-based overfitting behaviour.** The single Decision Tree and the Random Forest show very high bootstrap training accuracies (CIs around 0.984–0.986), while their test accuracies remain noticeably lower (0.823 and 0.846 respectively). This gap is a clear sign of overfitting: these models fit the training data extremely well, but do not generalise as well as Gradient Boosting or Linear SVM.

- **Stability according to cross-validation and bootstrap CIs.** For the best models (Gradient Boosting, Linear SVM, AdaBoost), the cross-validated accuracies (around 0.81–0.82) and the bootstrap confidence intervals are close to their test accuracies, suggesting that their performance is relatively stable and not overly sensitive to a particular train–test split.

5.5.1 Choice of the final model

Based on these results, we select **Gradient Boosting** as our final model. It delivers the best overall performance in terms of accuracy, F1-score and ROC–AUC on the test set, while its cross-validated accuracy and bootstrap confidence interval indicate good robustness. Linear SVM is retained as a strong and interpretable baseline, but Gradient Boosting offers the best predictive performance for the Adult Income classification task.

5.6 Feature importance analysis

To assess the impact of education level on income, we analyzed the coefficients obtained using the Gradient Boosting.

5.6.1 Impurity-based importance (dummy level)

Table 6 reports the top 20 features according to impurity-based importance at the dummy level. At this level, one specific category, `marital.status_Married-civ-spouse`, stands out as by far the most important feature for the model, followed by the numeric variables `education.num`, `capital.gain`, `age`, `capital.loss` and `hours.per.week`.

5.6.2 Impurity-based importance aggregated by original variable

To improve interpretability, dummy-level importances are aggregated by original variable. The results are shown in Table 7.

Table 6: Top features according to impurity-based importance (dummy level).

Feature	Importance
marital.status_Married-civ-spouse	0.3830
education.num	0.1987
capital.gain	0.1978
age	0.0643
capital.loss	0.0609
hours.per.week	0.0382
occupation_Exec-managerial	0.0179
occupation_Prof-specialty	0.0072
occupation_Other-service	0.0061
occupation_Farming-fishing	0.0054
relationship_Wife	0.0054
workclass_Self-emp-not-inc	0.0025
sex_Male	0.0020
occupation_Tech-support	0.0018
marital.status_Married-AF-spouse	0.0009
relationship_Own-child	0.0008
occupation_Sales	0.0008
workclass_Local-gov	0.0008
occupation_Protective-serv	0.0008
occupation_Machine-op-inspct	0.0007

Table 7: Impurity-based importance aggregated by original variable.

Variable	Importance
marital.status	0.3839
education.num	0.1987
capital.gain	0.1978
age	0.0643
capital.loss	0.0609
occupation	0.0414
hours.per.week	0.0382
relationship	0.0067
workclass	0.0038
sex	0.0020
native.country	0.0018
race	0.0003

This ranking shows that **marital status**, **education level** (education.num) and **capital gains** are the dominant predictors for the Gradient Boosting model, with additional contributions from age, capital.loss and hours.per.week. In contrast, race, sex, workclass and native.country have very low aggregated importance.

5.6.3 Permutation importance aggregated by original variable

Permutation importance results further validate these findings as shown on Tables 8.

Table 8: Permutation importance aggregated by original variable (test accuracy as scoring).

Variable	Perm_Importance
marital.status	0.0435
capital.gain	0.0425
education.num	0.0311
age	0.0159
capital.loss	0.0118
occupation	0.0077
hours.per.week	0.0069
relationship	0.0020
workclass	0.0016
sex	0.0005
race	0.0001
native.country	−0.0002

Permutation importance shows that marital.status, capital.gain and education are the three key drivers of prediction performance: permuting any of these variables on the test set reduces accuracy by about 3–4.5 percentage points. age, capital.loss, occupation and hours.per.week have a smaller but still noticeable impact, whereas variables such as relationship, workclass, sex, race and native.country have almost no effect on accuracy. This indicates that the Gradient Boosting model relies primarily on marital status, education and capital gains to distinguish individuals earning more than $50K.

6 Conclusion

Our initial scientific question was *does the eduction level increase the probability of earning an higher income* $> 50k$. Our results clearly support this hypothesis. Education level (education.num) consistently emerges as an important predictor of income, although it is not the only contributing factor.
To answer that question we first tried a classification model on a generated dataset, but then switched to an actual dataset and adapted the model to test it in a real scenario. The methodology applied on the two was the same (evaluation of model and feature importance analysis)

In the initial generated dataset, our variables, relationships and the impacts of the different variables were based on statistical articles published by the OFS. The variables were mainly socio-demographic (e.g. race, social class, marital-status, native country and so on) and a few were more directly linked to the income (hours-per-week, workclass, etc).

To test our model in a real senario, we used the UCI/Kaggle Adult Census Income [18] dataset and a supervised classification pipeline. We handled missing values with a conditional-mode imputation function (`fill_with_conditional_mode`) filling each missing value in a target column using the category that is the most represented in similar rows. We ran analysis to see the distribution of our variables, allowing us to better understand which categories were predominant. Using the *IQR* rule multiplied by a 1.5 factor we looked for potential outliers to remove them if needed, but since all screened values were realistic we decided to not remove any of them.

For the methodology, we had a binary target variable defined by : $y = \mathbb{1}\{\texttt{income} = \texttt{">50K"}\}$. We used one-hot encoding for our categorical variables and divided the dataset in two train/test splits (80/20, stratified). We then compared a list of algorithms, from which **Gradient Boosting** came out the best. The algorithms were compared on multiple metrics (accuracy, F1, ROC-AUC, cross-validated accuracy and Bootstrap confidence interval on accuracy). These comparisons allowed us to make multiple observation on the algorithms: the strength of **Linear SVM** and **AdaBoost** which both performed really well, just not a well as the **Gradient Boosting** for all three of these algorithms we also observed a great stability in **cross-validation** and **bootstrap CIs**; The overfitting behaviors of Tree-based algorithms. Accordingly to our results we chose Gradient Boosting for our final model.
A feature importance analysis was then done. We ranked the top features according to impurity-based importance first at dummy level, then aggregated by original variable and last permutation importance aggregated by original variables.

The simulation phase plays a crucial role in validating the approach, as it allows direct comparison between estimated effects and the known ground truth. The feature-importance analysis on the Adults Census dataset showed that the `education.num` is one of the top predictors for the income. Other features also showed great importance (`marital.status, capital.gains, age, hours.per.week`). We observed that the main important features were not all the same between our generated dataset and the Adult Census one, for example the martial status is the most important feature in the kaggle dataset while in our simulated one its effects are basically null. This is probably due to the nature of our simulated dataset in which we decided how features impacted each others when generating the data.
The sensitive variables like `sex` and `race` ended up contributing to almost nothing to the final model. This suggest that the model relies more on economic and demographic factors. The education level is one of the main drivers for higher income, but it is important to note that it is not the only factor with a high contribution.

Finally, ethical considerations must be taken into account. Sensitive attributes such as sex and race were included to control for confounding effects, but they contribute very little to the final predictions, suggesting that the model primarily relies on economic and demographic factors rather than protected characteristics. Nevertheless, the use of census data raises privacy concerns, and results should not be interpreted or used for individual-level decision-making. Any real-world application would require careful attention to fairness, transparency, and responsible use of the model outputs.

References

[1] Federal Statistical Office (FSO). Total self-employed persons and self-employed persons without employees – 2023, 2023. Chart, Federal Statistical Office (Switzerland). URL: https://www.bfs.admin.ch/bfs/en/home.assetdetail.33748377.html.

[2] Federal Statistical Office (FSO). Total fertility rate per woman – 1876–2023, 2024. Chart on fertility, Federal Statistical Office (Switzerland). URL: https://www.bfs.admin.ch/asset/en/32486285.

[3] Federal Statistical Office (FSO). Usual weekly working hours of full-time employed persons by work status – 2024, 2024. Chart, Federal Statistical Office (Switzerland). URL: https://www.bfs.admin.ch/asset/en/35507770.

[4] Federal Statistical Office (FSO). Composition of the foreign population, 2025. Statistics on the structure of the foreign resident population. URL: https://www.bfs.admin.ch/bfs/en/home/statistics/population/migration-integration/foreign/composition.html.

[5] Federal Statistical Office (FSO). Economic sector and branch, 2025. Topic page on labour force characteristics. URL: https://www.bfs.admin.ch/bfs/en/home/statistics/work-income/employment-working-hours/labour-force-characteristics/economic-sector.html.

[6] Federal Statistical Office (FSO). Marital status, 2025. Statistics on resident population by marital status. URL: https://www.bfs.admin.ch/bfs/en/home/statistics/population/effectif-change/marital-status.html.

[7] Organisation for Economic Co-operation and Development. Education at a glance 2025: Oecd indicators, chapter a4 “what are the earnings advantages to education?”, 2025. URL: https://www.oecd.org/en/publications/education-at-a-glance-2025_1c0d9c79-en/full-report/what-are-the-earnings-advantages-to-education_7a7e64e0.html.

[8] OECD. Government at a glance 2023: Switzerland, 2023. Country note, OECD Publishing, Paris. URL: https://www.oecd.org/en/publications/government-at-a-glance-2023_c4200b14-en/switzerland_8f88ce47-en.html.

[9] olethrosdc. machine-learning-neuch: Msc materials, 2025. GitHub repository. URL: https://github.com/olethrosdc/machine-learning-neuch/tree/main/MSc.

[10] scikit-learn developers. 1.6. nearest neighbors — scikit-learn documentation, 2025. Documentation of KNeighborsClassifier. URL: https://scikit-learn.org/stable/modules/neighbors.html.

[11] scikit-learn developers. sklearn.ensemble.adaboostclassifier — scikit-learn documentation, 2025. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html.

[12] scikit-learn developers. sklearn.ensemble.gradientboostingclassifier — scikit-learn documentation, 2025. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html.

[13] scikit-learn developers. sklearn.ensemble.randomforestclassifier — scikit-learn documentation, 2025. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

[14] scikit-learn developers. sklearn.naive_bayes.gaussiannb — scikit-learn documentation, 2025. URL: https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html.

[15] scikit-learn developers. sklearn.svm.linearsvc — scikit-learn documentation, 2025. URL: https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html.

[16] scikit-learn developers. sklearn.tree.decisiontreeclassifier — scikit-learn documentation, 2025. URL: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html.

[17] State Secretariat for Migration (SEM). Foreign population and asylum statistics 2024, 2025. Report, State Secretariat for Migration, Bern. URL: https://www.sem.admin.ch/dam/sem/en/data/publiservice/statistik/bestellung/auslaender-asylstatistik-2024.pdf.download.pdf/auslaender-asylstatistik-2024-e.pdf.

[18] UCI Machine Learning and Kaggle. Adult census income dataset, 2016. URL: https://www.kaggle.com/datasets/uciml/adult-census-income.