Introduce the problem -

The problem I am investigating is can we predict whether a person earns more than $50,000 per year based on demographic and employment attributes.

Questions I want to find the answers to -
Which demographic and work related features are most influential in predicting income?

Can a machine learning model accurately classify whether or not a person's income exceeds $50K?

Are there biases in the data that affect outcomes?

Introduce the data -

https://archive.ics.uci.edu/dataset/2/adult

Extracted from the 1994 U.S census. It contains individual demographic and employment data

Features:
Age
Workclass (Type of employment)(Private, Federal Government)
Education (Highest level of education obtained)
Marital status
Occupation
Race
Sex
Capital gain/loss (Gains and losses from investments)
Hours per week
Income
Fnlwgt (Census weighting variable)

Pre- processing steps

Missing values - Some rows have missing values marked with "?" I am going to replace "?" with NaN and drop these rows

Irrelevant Features -
Fnlwgt - This is the sampling weight used by the Census Bureau and does nothing for classifying people

Encode Categorical Values - Convert categorical features such as workclass, education, and occupation to a numerical format using pd.get_dummies

Training/Testing Split
Split dataset into training (70%)and testing (30%) data

Model -

For this project, I chose to use Decision Tree and Random Forest models because both are well suited for classification tasks with mixed categorical and numerical features, such as the Adult Income dataset. Decision Trees provide easy readability, while Random Forest improves on accuracy and generalization by reducing overfitting. Using both allows me to compare a simple baseline model with a more complex model

Decision Tree Model

What it is:
A Decision Tree splits the data into smaller subsets based on feature values, creating a tree-like structure. Each internal node represents a decision rule (e.g., “education-num > 12”), and each leaf node represents a predicted class (<=50K or >50K).

How it works:
The algorithm recursively chooses the feature and threshold that best separates the data (often using Gini impurity or entropy).

Pros:

Very interpretable — easy to visualize and explain results.

Handles both categorical and numerical features without scaling.

Fast to train and predict.

Cons:

Prone to overfitting, especially with deep trees.

Can be sensitive to small changes in the data.

May not achieve the best accuracy compared to more advanced methods.
.

What it is:
Random Forest is an ensemble method that builds many decision trees and combines their predictions

How it works:
Each tree is trained on a random subset of the data (bootstrap sampling) and considers only a random subset of features at each split. This randomness ensures diversity among trees, which reduces variance and overfitting.

Pros:

Usually achieves higher accuracy than a single decision tree.

Less prone to overfitting due to averaging across many trees.

Provides feature importance scores, helping identify which variables most influence income predictions.

Cons:

Less interpretable than a single decision tree (harder to explain all trees).

Requires more computational resources.

Predictions are less transparent compared to a simple tree.

Evaluation -

Both the Decision Tree and Random Forest models performed well, achieving approximately 85% accuracy on the test set. The Decision Tree had slightly higher recall for the majority class but lower for the high-income class, while the Random Forest improved recall and F1 score for the high income group, showing better balance between classes. I used accuracy to measure overall correctness, precision to evaluate how many predicted high income individuals were actually correct, recall to assess how well the model identified all high income individuals, and F1 score to balance precision and recall. These metrics are important because the dataset is somewhat imbalanced, with fewer high-income individuals, so relying on accuracy alone could be misleading. Overall, Random Forest slightly outperforms the Decision Tree, especially in predicting the higher-income class, making it a more robust model for this task.

Storytelling -

Through this project, I learned that income is strongly influenced by several key factors, including education level, hours worked per week, capital gains, and marital status. The models show that higher education and more working hours significantly increase the likelihood of earning over $50K, while demographic factors such as sex and race also have an impact, highlighting potential biases in the data. By comparing the Decision Tree and Random Forest models, I saw that while a single tree is easy to interpret, the Random Forest provides better overall performance and more reliable predictions for the high-income class. These insights directly answer my initial questions: education, hours worked, and capital gains are the most influential features; machine learning models can predict income above $50K with reasonable accuracy; and there are noticeable biases in the data that could affect outcomes. Overall, the analysis tells a story about how economic and demographic factors combine to shape income levels, and it also emphasizes the importance of considering fairness when using predictive models.

Impact -

This project provides insights into which factors influence income and can inform education, career planning, and policy decisions. However, the models reflect biases present in the historical data, such as gender and racial disparities, and could reinforce inequalities if misused in hiring or lending. Using demographic information to predict income also raises privacy and ethical concerns. Overall, the analysis highlights the importance of applying machine learning responsibly, with attention to fairness and potential social impacts.

References

https://archive.ics.uci.edu/dataset/2/adult
https://www.geeksforgeeks.org/machine-learning/decision-tree-implementation-python
https://www.geeksforgeeks.org/machine-learning/random-forest-regression-in-python
https://www.kaggle.com/code/usamabajwa86/classification-report