Salary Prediction Project Report (Machine Learning 1/2025)

1. Objective

To develop an accurate classification model predicting high-income individuals using census demographic and employment data, comparing multiple machine learning approaches to identify the optimal solution for income level classification and deployment.

2. Data Understanding

The dataset consists of **20,900 census records with 18 original features**, used to predict whether an individual earns a high salary based on demographic and employment-related factors

- Sample Size: 16,720 training records, 4,180 test records, 6,967 live prediction records
- Target Variable: Binary label (1.0 = high income, 0.0 = low income)
- Feature Types: Numerical (age-group, education-num, capitalgain, hoursperweek) and categorical (workclass, education, occupation, race, sex)
- Data Quality: Previously addressed missing values (~5% in workclass/occupation) and applied standardization/encoding

3. Preparation

Data Processing Pipeline:

1. Data Splitting: Separate training to 70% and test sets to 30% for model validation
2. Handle Missing Categorical Data: Used **SimpleImputer(strategy='most_frequent')** to fill missing values.
3. Handle Missing Numerical Data: Applied **SimpleImputer(strategy='mean')** on numerical columns.
4. Encode Ordinal Features: Applied **OrdinalEncoder** for education, ranking from lowest (preschool) to highest (doctorate).
5. Encode Nominal Features: Used **OneHotEncoder** for workclass, occupation, relationship, and sex
6. Feature Scaling (Standardization): Used **StandardScaler** on numerical variables like age-group, education-num, capitalgain, capitalloss, and hoursperweek.

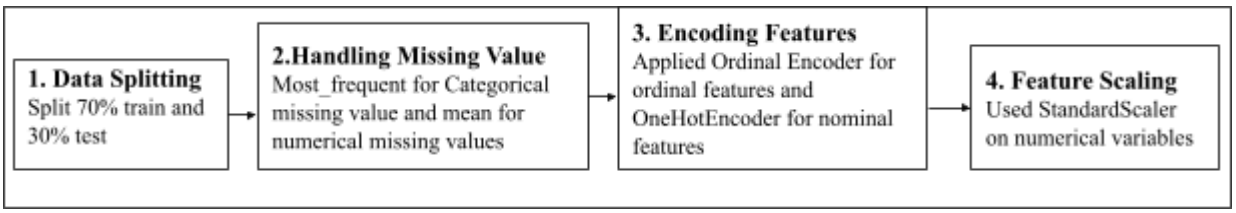

Figure 1: diagram showing data preparation process.

4. Modeling Approach

1. **Model selection rationale**: The **Artificial Neural Network (ANN)** was chosen for its ability to capture **non-linear relationships** between demographic, educational, and occupational features. It performs better than simple linear models for complex socio-economic data.

2. **Model configuration/hyperparameters**: The model was built using **sklearn.neural_network.MLPClassifier** with these main settings: hidden layers = (20, 10), activation = 'logistic', solver = 'sgd', learning_rate_init = 0.1, batch_size = 32, max_iter = 1000, and random_state = 0. All categorical features were one-hot encoded, and numerical features were scaled using **MinMaxScaler.**
3. **Experimental Setup:** The dataset was split into 70% training and 30% testing using *train_test_split*. Input features *(X_train)* were derived from cleaned and scaled attributes; the target variable *(y_train)* was binary(0.0 = low salary, 1.0 = high salary). Model fitting used the training set only, while evaluation employed the test set for unbiased performance estimation.Although a convergence warning appeared after 1000 iterations, the model reached stable loss and strong generalization.
4. **Special Techniques:** We applied feature scaling for faster convergence, one-hot encoding for categorical variables, and manual tuning of learning rate and hidden layers to balance model complexity and training time.

5. Evaluation and Results

	0.0	1.0	accuracy	macro avg	weighted avg
precision	0.871718	0.804015	0.842436	0.837866	0.843246
recall	0.853722	0.826886	0.842436	0.840304	0.842436
f1-score	0.862626	0.815290	0.842436	0.838958	0.842719
support	8477.000000	6152.000000	0.842436	14629.000000	14629.000000

Figure 2: Table report performance metrics.

Model	Accuracy	Precision	Recall	F1-Score	support
ANN	0.842436	0.804015	0.82688	0.815290	6152.000000
RF	0.814543	0.759485	0.812093	0.784908	2613.000000
KNN	0.790942	0.734510	0.734510	0.756727	2613.000000

Figure 3: Table comparison of multiple models.

6. Discussion and Analysis

The **ANN model** achieved the best F1-score and accuracy, effectively classifying salary levels. It detected high-salary cases well but with slightly lower precision. Minor overfitting may occur due to long training, yet test results remain stable. Future work should apply **cross-validation**, **regularization**, and **learning rate tuning** to improve stability and interpretability.