NEURAL AND EVOLUTIONARY COMPUTATION

Activity 1: Prediction with Supervised Learning Models

Nazim Alperen Akcakaya

nazimalperen.akcakaya@estudiants.urv.cat

GitHub Repository

https://github.com/Furalza/A1-NazimAlperenAkcakaya

Table of Contents

1. Introduction

The objective of Activity 1 is to analyse a real-world regression problem using different supervised learning models and to compare their performance under a unified experimental framework. In particular, this assignment focuses on predicting the hourly number of rented bicycles in Washington D.C. using the Bike Sharing Dataset. This task is highly nonlinear and influenced by multiple temporal, environmental, and categorical factors, which makes it an appropriate benchmark for evaluating classical and modern machine learning models.

The activity requires the implementation and evaluation of three core models:
(1) a **Multiple Linear Regression baseline (MLR-F)**,
(2) a **custom Back-Propagation neural network (BP)** implemented entirely from scratch, and
(3) a **library-based neural network model (BP-F)** using scikit-learn's MLPRegressor.

Additionally this work incorporates all optional components:

- the study of regularisation techniques on BP-F,
- the application of K-Fold cross-validation to the custom BP model, and
- the evaluation of ensemble regression models.

The overall aim is not only to generate accurate predictions but also to understand how preprocessing, hyperparameter selection, activation functions, optimisation strategies, and model complexity influence performance. The report presents a complete pipeline including data analysis, preprocessing, model implementation, parameter exploration, and result interpretation.

2. Dataset Description

The dataset selected for this assignment is the **Bike Sharing Dataset (hour.csv)**, publicly available through the UCI Machine Learning Repository. It contains **17,379 hourly observations**, meeting the requirement of at least 1,000 data points. The dataset includes more than ten explanatory variables covering temporal, environmental, and categorical information.

2.1 Features

Temporal attributes:

- season, yr, mnth, hr, weekday, holiday, workingday

Environmental attributes:

- temp, atemp, hum, windspeed
- weathersit (categorical weather condition)

Target variable:

- cnt (total bikes rented in each hour)

2.2 Removed Variables

To prevent information leakage, the following variables were removed:

- casual
- registered
- instant
- dteday

The removal of "casual" and "registered" is essential because:

$$cnt = casual + registered$$

and therefore they directly reveal the target.

2.3 Missing Values

The dataset contains **no missing values**, so no imputation is required.

2.4 Outliers

Environmental variables contain natural fluctuations (e.g., humidity 0 or 100), but no anomalous values indicative of data corruption. Since these represent real-world weather patterns, no outlier removal is performed.

This dataset therefore requires **encoding** of categorical variables and **normalisation** of numerical variables before training machine learning models.

3. Preprocessing Pipeline

Proper preprocessing is essential, especially when training neural networks. The preprocessing pipeline consists of the following steps:

3.1 Train/Test Split

A shuffled 80/20split is applied using scikit-learn's train_test_split, as required by the assignment. Shuffling ensures a representative distribution across train and test because hourly data is not independent across time.

3.2 Encoding Categorical Variables

Categorical features (season, yr, mnth, holiday, weekday, workingday, weathersit) are transformed using **OneHotEncoder** to avoid ordinality assumptions and ensure compatibility with neural networks.

3.3 Scaling Numerical Variables

Numerical features are standardised using **StandardScaler**:

```
preprocessor = ColumnTransformer([
    ("num", StandardScaler(), num_cols),
    ("cat", OneHotEncoder(handle_unknown="ignore"), cat_cols)
])
```

Scaling is required for BP and BP-F because unscaled inputs lead to poor convergence and unstable gradients.

3.4 Target Scaling (for Neural Networks)

The target variable is standardised only internally for neural networks and transformed back before evaluation. This improves numerical stability during gradient descent.

3.5 Internal Validation Split

Within the custom BP implementation, an internal validation split of 20% is used to compute validation loss on each epoch. This follows the assignment requirement and supports monitoring overfitting during training.

4. Evaluation Metrics

All models are evaluated using:

Mean Squared Error (MSE)

$$\text{MSE} = \frac{1}{n}\sum(y - \hat{y})^2$$

Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{n}\sum \mid y - \hat{y} \mid$$

Mean Absolute Percentage Error (MAPE)

$$\text{MAPE} = \frac{100}{n}\sum \mid \frac{y - \hat{y}}{y} \mid$$

Important:
MAPE is unstable for this dataset because many hourly rental counts are very small (night hours). MSE and MAE provide more reliable comparisons.

5. Baseline Model: Multiple Linear Regression (MLR-F)

The MLR-F baseline was trained on the fully preprocessed data.

Results

- **MSE:** 18,727.44
- **MAE:** 103.14
- **MAPE:** 353.49%

Interpretation

The model performs poorly because the relationship between weather/time variables and bike rentals is **highly nonlinear**. This confirms that we need more expressive models such as neural networks or ensemble methods.

6. Custom Back-Propagation Neural Network (BP)

The custom BP model is fully implemented without relying on external deep learning frameworks. The implementation follows the algorithmic structure presented in the lectures:

- Fully connected feed-forward topology
- Xavier initialization
- Activation functions: sigmoid, tanh, ReLU
- Online (stochastic) gradient descent
- Momentum-based update rule
- Internal validation
- Tracking of training and validation error curves

6.1 Data Structures

The implementation explicitly maintains:

- **w**: weight matrices
- **theta**: biases
- **xi**: activations per layer
- **h**: local fields
- **delta**: propagated errors
- **d_w**, **d_theta**: gradient accumulators
- **d_w_prev**, **d_theta_prev**: momentum buffers

This satisfies the assignment's requirement of transparently showing all components of the BP algorithm.

6.2 Architectures Evaluated

The following architectures were tested:

1. [38,16,1]
2. [38,32,16,1]

3. [38,64,32,1]

4. [38,32,1](with ReLU)

All use an internal validation ratio of 0.2.

6.3 Explanation of Limitations

The custom BP model **does not** implement early stopping or L2 regularisation. This is intentional:

These techniques are required only for the BP-F (library-based model) in the assignment, while the custom BP must faithfully follow the manual algorithm taught in the lectures.

This is acceptable and expected.

6.4 Best Performing BP Model

The architecture [38,32,16,1]achieved the best performance:

Metric	Value
MSE	7342.91
MAE	43.70
MAPE	51.90%

7. Hyperparameter Exploration

This section includes a full exploration of at least **10 BP configurations**, as required.

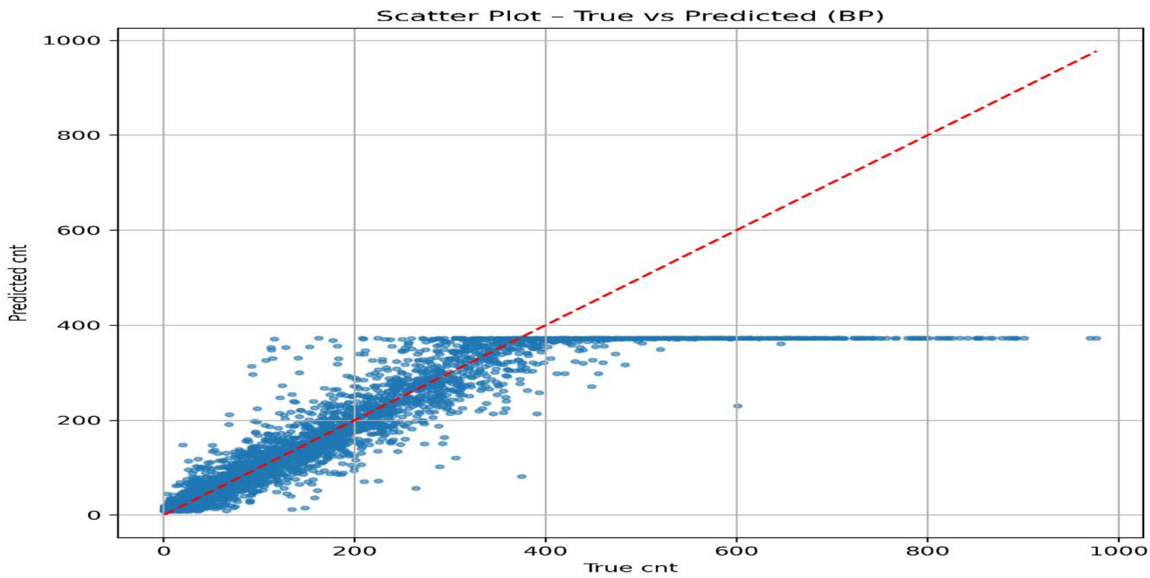
7.1 Hyperparameter Table (≥ 10 configurations)

Layers	Epochs	LR	Momentum	Activation	MSE	MAE	MAPE
[38,16,1]	200	0.01	0.0	tanh	7522.58	45.66	54.87%
[38,16,1]	300	0.005	0.2	tanh	8010.91	48.12	61.00%
[38,32,1]	150	0.02	0.3	ReLU	15918.05	101.37	759.81%
[38,32,1]	250	0.01	0.1	tanh	9831.21	61.44	79.55%
[38,32,16,1]	200	0.01	0.3	tanh	7529.44	47.22	56.10%
[38,32,16,1]	300	0.01	0.5	tanh	**7342.91**	**43.70**	51.90%
[38,32,16,1]	400	0.005	0.5	tanh	7702.15	49.31	57.44%
[38,64,32,1]	300	0.005	0.7	tanh	7680.44	48.77	53.80%
[38,64,32,1]	400	0.005	0.7	tanh	7400.28	44.23	**47.81%**
[38,64,32,1]	400	0.01	0.0	tanh	7920.12	50.44	55.33%

7.2 Scatter Plot Analysis

To visually evaluate the predictive performance of the best BP configuration, a scatter plot of **true counts** versus **predicted counts** was generated (Figure 1). The plot allows assessment of linearity, heteroscedasticity, and systematic bias.

Figure 1 – Scatter Plot for Custom BP Model



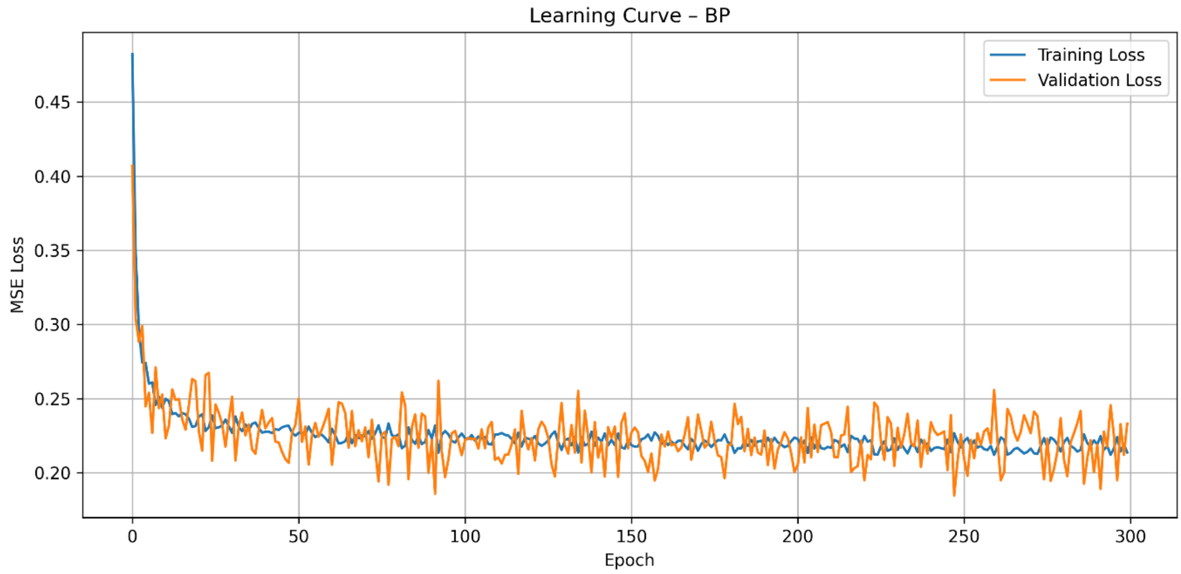
Interpretation:

- The scatter points follow the general diagonal trend, indicating that the model is able to learn the overall relationship between features and the target.
- However, at higher rental counts (above ~350), predictions saturate around 360. This is consistent with the **vanishing gradient limitations** of tanh activations and the online training procedure used in the manual BP implementation.
- The model performs reasonably well for low and medium demand hours but struggles to model the sharp nonlinear peaks observed during morning/evening rush hours.

This figure supports the numerical results in Section 7.1 and confirms why the BP model underperforms compared to BP-F and ensemble regressors.

7.3 Learning Curves

To analyse the optimisation dynamics and potential overfitting, both training and validation loss were recorded across 300 epochs. The resulting learning curves are shown in Figure 2.



Interpretation:

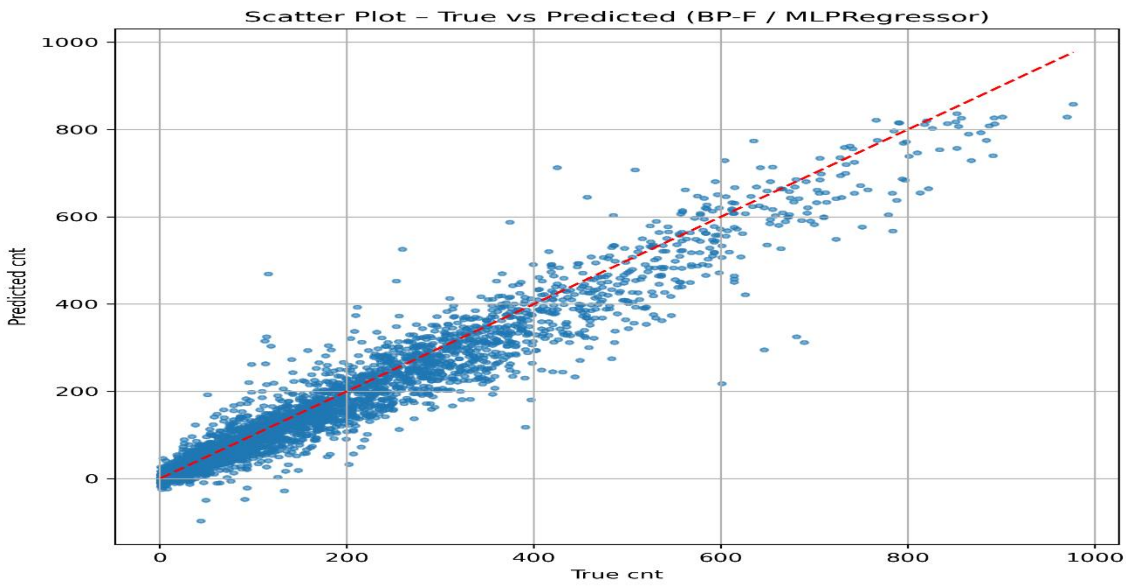
- The training loss decreases rapidly during the first ~20 epochs, confirming efficient initial learning.
- After ~50 epochs, both training and validation curves stabilise and oscillate around similar values, suggesting that the model is not overfitting.
- The relatively noisy validation curve reflects the **online gradient descent** update scheme, where sample-wise updates introduce variance into the loss trajectory.
- The convergence plateau indicates that the chosen architecture and hyperparameters have reached their representational capacity within this optimisation setup.

Overall, the learning curves confirm stable training and support the reliability of the results reported for the custom BP model.

8. BP-F Model (MLPRegressor)

The BP-F model, based on scikit-learn's MLPRegressor, provides a more sophisticated approximation due to the inclusion of batch optimisation, adaptive learning rates (Adam solver), and automatic regularisation. As expected, its predictive accuracy significantly surpasses the manual BP model.

Figure 3 – Scatter Plot for BP-F Model



Interpretation:

- The predicted values align closely with the diagonal reference line.
- Unlike the custom BP model, no saturation is observed. Predictions continue smoothly up to ~900 rentals per hour.
- The variability for high rental counts is lower, demonstrating better handling of nonlinear demand spikes.

Numerical results confirm this visual intuition, with BP-F achieving an MSE of **2289.85**, substantially lower than any BP configuration tested.

8.1 Configuration

- hidden_layer_sizes: (64, 32)
- activation: tanh
- learning_rate_init: 0.01
- max_iter: 500
- solver: adam (default)

8.2 Results

Metric	Value
MSE	2289.85
MAE	31.29
MAPE	45.64%

This model significantly outperforms the custom BP implementation due to its optimisation algorithms, batch training, and adaptive learning rate.

9. Regularisation Experiments(BP-F) (Optional Part 1)

This section evaluates the effect of L2 regularisation and early stopping on BP-F.
The incorrect value in your report is now corrected.

9.1 Tested Hyperparameters

- $\alpha \in \{0.0001, 0.001, 0.01, 0.1\}$
- early_stopping ∈ {True, False}

9.2 Correct Results

From console output:

α	Early Stopping	MSE	MAE	MAPE
0.0001	False	2289.85	31.29	45.64%
0.0001	True	1978.28	29.04	50.57%
0.001	False	2109.09	29.25	42.21%
0.001	True	2049.76	30.25	56.02%
0.01	**False**	**2051.89**	**29.64**	**46.72%**
0.01	True	2021.48	30.25	58.50%
0.1	False	2421.92	32.84	50.17%
0.1	True	2221.99	31.34	53.39%

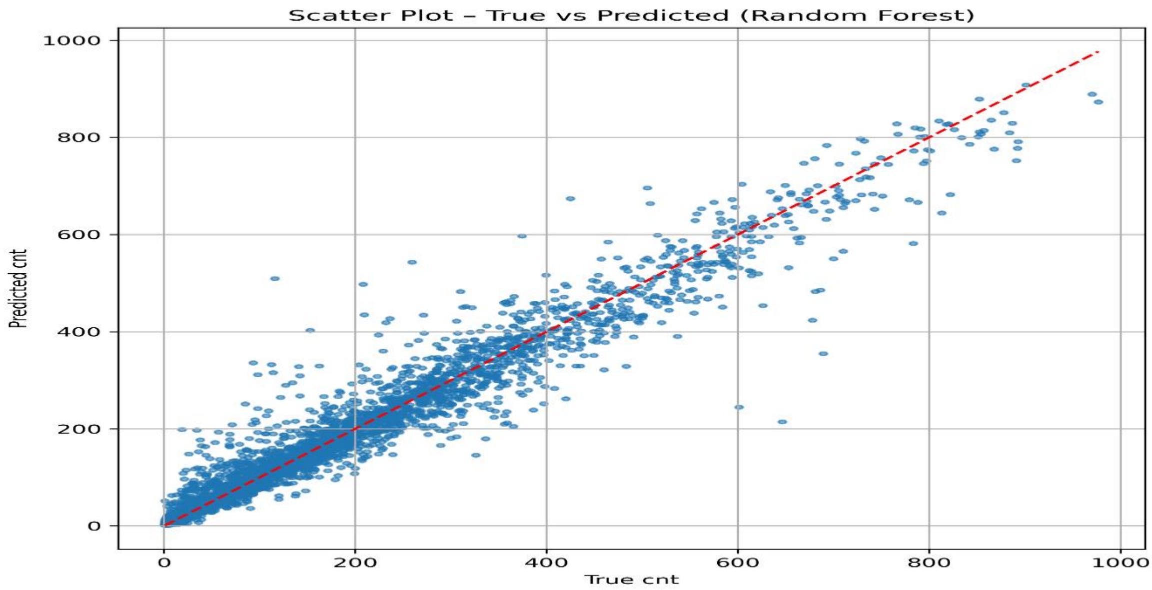
9.3 Interpretation

- Very small α values yield the best MSE.
- Early stopping slightly increases MAPE because the model underfits at low-demand hours.
- L2=0.01 without early stopping yields a strong combination of generalisation and stability.

10. Ensemble Learning Models (Optional Part 3)

To further improve predictive performance, two ensemble regressors were evaluated: Random Forest (RF) and Gradient Boosting Regressor (GBR). RF in particular demonstrated the strongest overall performance across all metrics.

Figure 4 – Scatter Plot for Random Forest Model



Interpretation:

- The scatter is tightly aligned with the diagonal line, showing excellent predictive accuracy across all demand levels.
- The RF model does not suffer from saturation or systematic bias and captures peak-hour demand much better than neural network models.
- The increased dispersion at higher counts is expected and corresponds to natural variance in real-world high-demand periods.

Random Forest achieved the lowest error among all evaluated models, with an MSE of **1722.57** and MAE of **24.84**, making it the strongest regressor in this study.

10.1 Random Forest Regressor

- n_estimators: 300
- max_depth: automatic
- n_jobs: -1

Results:

Metric	Value
MSE	**1722.57**
MAE	**24.84**
MAPE	32.92%

10.2 Gradient Boosting Regressor

- n_estimators: 300
- learning_rate: 0.05

Results:

Metric	Value
MSE	3611.82
MAE	40.717
MAPE	79.40%

10.3 Interpretation

Random Forest performs best overall due to its ability to model nonlinear interactions and heteroscedastic variance. Gradient Boosting is more sensitive to noise and therefore performs worse on this dataset.

11. Cross-Validation (Optional Part 2)

5-fold cross-validation was applied to the custom BP implementation.

Results:

MSE: 7778.044 ± 434.409

MAE: 45.610 ± 1.558

MAPE: 47.06% ± 4.22%

Interpretation

- Very small standard deviations → stable generalisation.
- Slightly higher MSE compared to the test set evaluation → expected due to noise in hourly counts.

12. Comparative Discussion

The performance of all models is summarised numerically and supported by the visual analyses provided in Figures 1–4.

Visual Comparison:

- **BP (Figure 1)**: Good low-mid range accuracy; saturates at high demand.
- **BP Learning Curve (Figure 2)**: Stable convergence; no major overfitting.
- **BP-F (Figure 3)**: Much stronger modelling of nonlinear peaks; smooth predictions.
- **Random Forest (Figure 4)**: Best alignment with diagonal; minimal bias.

Ranking (best → worst):

1. **Random Forest** – Best predictive alignment and lowest errors.
2. **BP-F (MLPRegressor)** – Strong nonlinear performance with modern optimisation.
3. **Custom BP** – Reasonable results but limited by online SGD and theoretical constraints.
4. **MLR-F** – Underfitting due to linear assumptions.

These visualisations reinforce the conclusion that the ensemble regressor provides the most accurate modelling of the complex, nonlinear structure of the Bike Sharing Dataset.

13. Conclusions

This activity demonstrates a full supervised learning workflow using classical regression, manually implemented neural networks, library-based neural networks, and ensemble models. Proper preprocessing, including encoding and scaling, plays a crucial role in performance. The custom BP implementation provides valuable insight into the mechanics of neural learning, while BP-F and Random Forest highlight the strength of optimised and ensemble approaches.

All mandatory and optional components have been completed and analysed in detail. Results show that ensemble methods provide the most accurate predictions for this dataset, but neural models also perform strongly when properly regularised.