Introduction:

About the dataset:

Features:

- Resource Allocation (RA): The amount of resource allocated to the employee to work, ie. number of working hours. On a scale of 1 to 10, strictly integers.
- Mental Fatigue Score (MS): The level of fatigue mentally the employee is facing. On a scale of 1 to 10, float datatype, where 0.0 means no fatigue and 10.0 means complete fatigue.

Target:

- Burn Rate (BR): The value we need to predict for each employee telling the rate of burn out while working. On a scale of 0 to 1, float datatype

Objectives:

- I am interested in predicting the values for burn rate using either or both of the independent variables (features)

Why does this matter ?

- It helps employers prevent foreseeable burn out, boost overall productivity and efficiency, and consequently save money for their companies.
- Specifically, burnout can lead to a range of costs for employers, including:

  Decreased Productivity: Burn out could hinder creativity and innovation within an organization.

  Employee Turnover: Employees experiencing burnout may leave the organization, resulting in recruitment and training costs for new hires.

  Impact on Company Culture: Burnout can have a negative impact on the overall company culture and employee morale.

Methodology:

- The data set comes with train.csv (for training) and test.csv (for testing). For methods 1 and 2 and their variants, I make 2 models.

- I train the first model with the entire train.csv and then test it with test.csv.
- I train the second model with 70% of train.csv and test it with the other 30% of the set.

- This is because test.csv does not come with target values (burn rate) and it makes it impossible to validate the results using MSE, R-squared or cross-validation.

1) How to plot learning curve:

```
# Define your linear regression model
model = LinearRegression()

# Create learning curve data
train_sizes, train_scores, test_scores = learning_curve(
    model, X_train, y_train, cv=5, scoring='neg_mean_squared_error', train_sizes=np.linspace(0.1, 1.0, 10)
)

# Calculate mean and standard deviation for train and test scores
train_scores_mean = -np.mean(train_scores, axis=1)
train_scores_std = np.std(train_scores, axis=1)
test_scores_mean = -np.mean(test_scores, axis=1)
test_scores_std = np.std(test_scores, axis=1)
```

2) How to compute regression equation for methods 1 and 2:

```
m = regr.coef_[0][0]  # Extract the coefficient
i = regr.intercept_[0]  # Extract the intercept
# The coefficients
print("Coefficients: \n", m)

print(f"Regression Equation: Burn Rate = {m:.10f} * {a} + {i:.10f}")
```

```
beta0 = regr.coef_[0][0]
beta1 = regr.coef_[0][1]
i = regr.intercept_[0]

print("Coeff of Resource Allocation \n", beta0)
print("Coeff of Mental Fatigue Score \n", beta1)
print("Intercept: \n", i)

print(f"Regression Equation: Burn Rate = {i:.10f} + {beta0:.10f} * {b} + {beta1:.10f} * {c} ")
```

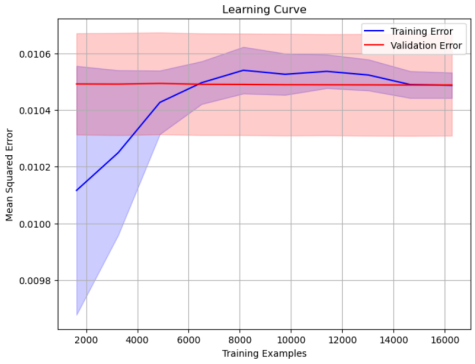
Method 1: Linear Regression

Resource Allocation

First model:

+ Regression Equation 1: (BR) = 0.0828905525 * (RA) + 0.0805349253

Learning curve:



+ Since the training and validation errors are converging to a low value (approx 0.0105), it indicates a well-fitted model.

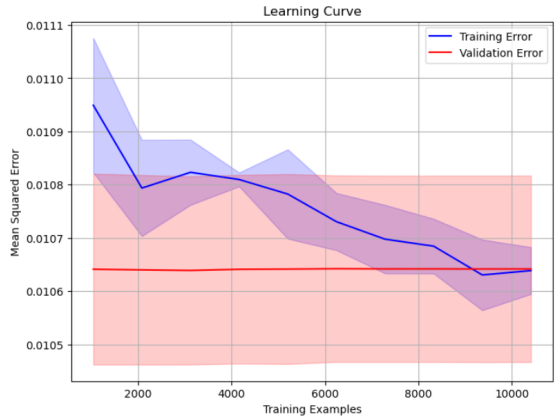
Second model:

+ Regression Equation 2: (BR) = 0.0826594136 * (RA) + 0.0819770611

Quantitative evaluation metrics:

+ Mean Squared Error (MSE): 0.010269068662474644
+ Root Mean Squared Error (RMSE): 0.10133641330970149
+ Mean Absolute Error (MAE): 0.08217012364576409
+ R-squared (Coefficient of Determination): 0.7345827796095579
+ The R-squared value is approximately 0.7346, which means that around 73.46% of the variance in the target variable is explained by the model.

Learning curve:



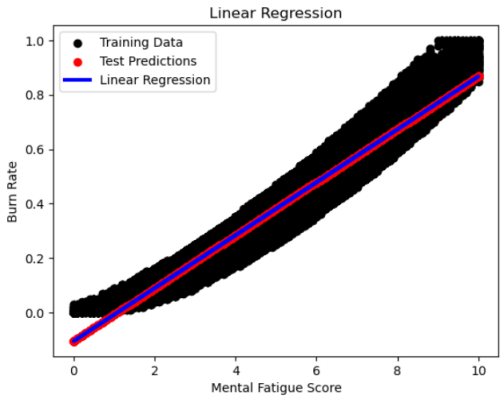
+ Since the training and validation errors are converging to a low value (approx 0.01067), it indicates a well-fitted model.

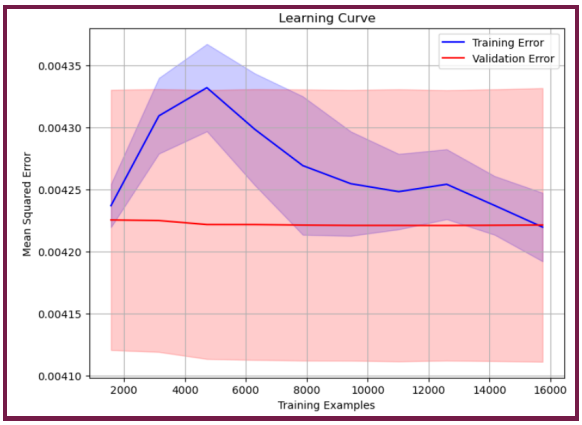
Mental Fatigue Score

First model:

Regression Equation 3: (BR) = 0.0972792965 * (MS) - 0.1054152958

Learning curve:





Second model

+ Regression Equation 4 : (BR) = 0.0973627735 * (MS) - 0.1056281247

Quantitative evaluation metrics:

+ Mean Squared Error (MSE): 0.004202475124432487
+ Root Mean Squared Error (RMSE): 0.06482650017109119
+ Mean Absolute Error (MAE): 0.054322735174858916
+ R-squared (Coefficient of Determination): 0.8913816527137665
+ The R-squared value is approximately 0.8914, which means that around 89.14% of the variance in the target variable is explained by the model.

Learning curve:

+ Since the training and validation errors are converging to a low value (approx 0.00425), it indicates a well-fitted model.

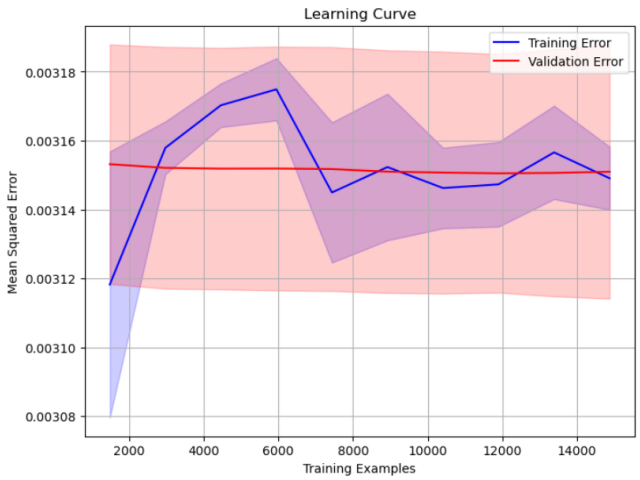
Conclusion:

Method 2: Multiple Linear Regression

First model:

+ Regression Equation 5: (BR): = -0.0946832721 + 0.0267990136 * (RA) + 0.0744730516 * (MS)

Learning curve:



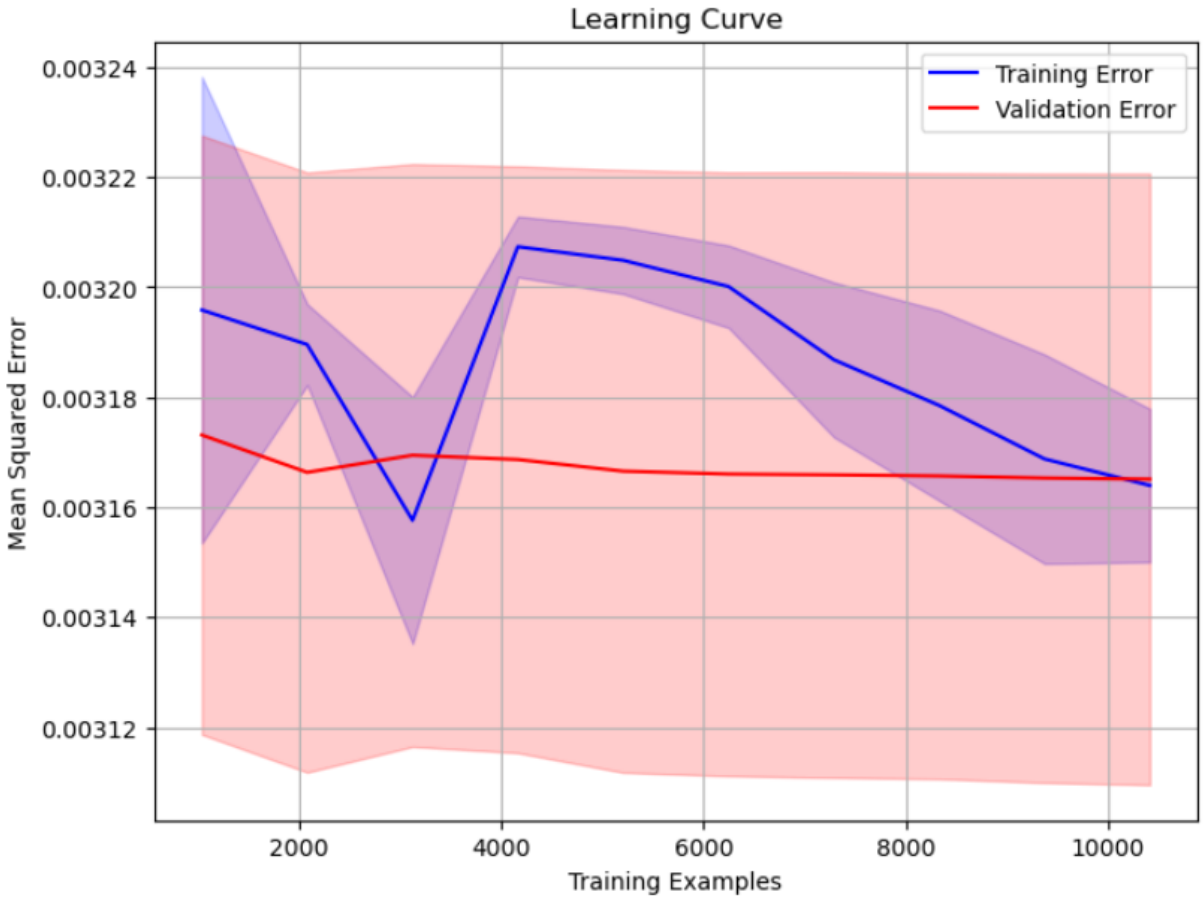
Second model:

+ Regression Equation 6: (BR): = -0.0952024031 + 0.0266647415 * (RA) + 0.0746786432 * (MS)

Quantitative evaluation metrics:

+ Mean Squared Error (MSE): 0.0031150585392848646
+ Root Mean Squared Error (RMSE): 0.05581270947808272
+ Mean Absolute Error (MAE): 0.04586910850999231
+ R-squared (Coefficient of Determination): 0.91948732586901
+ The R-squared value is approximately 0.9194, which means that around 91.948% of the variance in the target variable is explained by the model.

Learning curve:



Method 3: K-Nearest Neighbours (Regression)

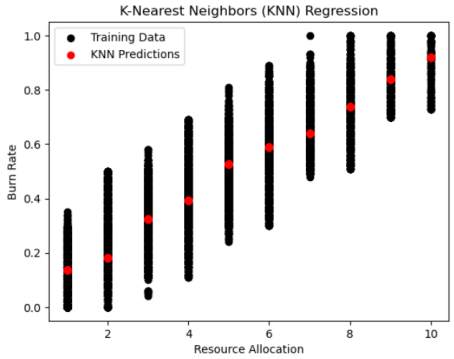
- Only 1 model was made with 70% of train.csv for training and 30% of train.csv for testing. Train.csv was unused

Resource Allocation

Quantitative evaluation metrics:

+ Mean Squared Error (MSE): 0.011077164783934015
+ Root Mean Squared Error (RMSE): 0.10524811059555424
+ Mean Absolute Error (MAE): 0.08497077281692667
+ R-squared (Coefficient of Determination): 0.7136965012706264
+ Average cross-validation score: 69.1%

```
Cross-validation scores: [0.70512772 0.66793524 0.68021156 0.68126168 0.72047114]
Average cross-validation score: 0.6910014682424871
```

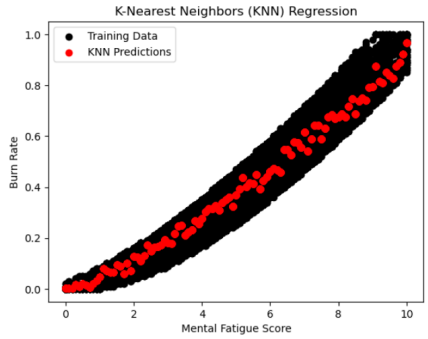


Mental Fatigue Score

Quantitative evaluation metrics:

+ Mean Squared Error (MSE): 0.0044612415277030665
+ Root Mean Squared Error (RMSE): 0.06679252598684277
+ Mean Absolute Error (MAE): 0.055177335485027804
+ R-squared (Coefficient of Determination): 0.8846935038909346
+ Average cross-validation score: 88.262%

```
Cross-validation scores: [0.8804761  0.87866211 0.88545131 0.87762986 0.89088719]
Average cross-validation score: 0.8826213140904947
```



Both Features

Quantitative evaluation metrics:

+ Cross-validation scores: [0.91111952 0.9076529 0.90889705 0.90599689 0.91706834]
+ Average cross-validation score: 0.910146941919734
+ Mean Squared Error (MSE): 0.003520083916083916

+ Root Mean Squared Error (RMSE): 0.05933029509520339
+ Mean Absolute Error (MAE): 0.04746745562130177
+ R-squared (Coefficient of Determination): 0.9090189267151175

Conclusion:

Notes: LR = Linear Regression ; MLR = Multiple Linear Regression ; RA = Resource Allocation ; MS = Mental Fatigue Score
Method 1 with Resource Allocation:

+ Regression Equation 1: (BR) = 0.0828905525 * (RA) + 0.0805349253
+ Regression Equation 2: (BR) = 0.0826594136 * (RA) + 0.0819770611

Method 1 with Mental Fatigue Score:

+ Regression Equation 3: (BR) = 0.0972792965 * (MS) - 0.1054152958
+ Regression Equation 4: (BR) = 0.0973627735 * (MS) - 0.1056281247

Method 2: Multiple Linear Regression:

+ Regression Equation 5: (BR): = 0.0267990136 * (RA) + 0.0744730516 * (MS) -0.0946832721
+ Regression Equation 6: (BR): = 0.0266647415 * (RA) + 0.0746786432 * (MS) -0.0952024031

For methods 1 and 2, the two regression equations from each variant are similar (look at the intercept and coefficient(s)).

Therefore, it is sufficient to conclude that the first and second models in each method and its variants are so similar that we can assume that all quantitative evaluation metrics found for the second model are also applicable to the first model.

Model's name and variable(s)	R-squared value	Mean Squared Error	Ranking (based on R-squared / MSE)
(1) LR + RA	0.7345827796095579	0.01026906866247464	5th / 5th
(2) LR + MS	0.8913816527137665	0.00420247512443248	3rd / 3rd
(3) MLR	0.91948732586901	0.00311505853928486	1st / 1st
(4) KNN + RA	0.7136965012706264	0.01107716478393401	6th / 6th
(5) KNN + MS	0.8846935038909346	0.00446124152770306	4th / 4th
(6) KNN + MS + RA	0.9090189267151175	0.003520083916083916	2nd / 2nd

Additionally, there is a similarity in the regression plots for between the KNN and LR models.

It may indicate that the KNN model is performing similarly to the linear regression model for the given data. The similarity in the plots suggests that both models are capturing a similar relationship between "Mental Fatigue Score"/ “Resource Allocation” and "Burn Rate" in the data.

Overall, it’s best to use both features to predict burn rate. If one could not collect a sufficiently large sample of either of the features, it’s best to use Mental Fatigue Score as the independent variable. Resource Allocation should only be the last resort if Mental Fatigue Score is not available.