Part 1 - Energy-efficiency dataset | Regression

Task: Build a deep neural network model using Back propagation and stochastic gradient descent algorithm:

Introduction
A stochastic gradient descent algorithm is the most efficient algorithm for training artificial neural networks, where weights are model parameters, and the target loss function is the prediction error averaged over a subset (batch) of the entire training dataset. Backpropagation calculates a loss function's gradient with respect to the variable values of a model. A neural network model is trained by backpropagation, which calculates the gradient for each weight. Using an optimization algorithm, the model weights are then updated using the gradient.

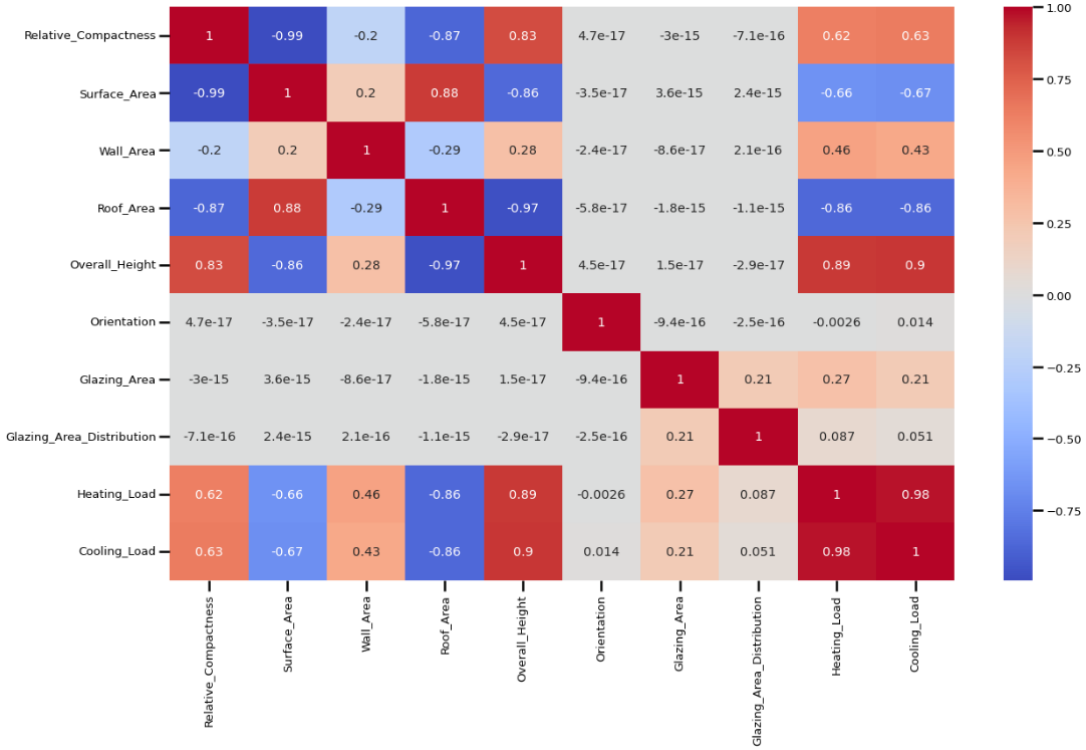
- Error Function: Loss function that is minimized when training a neural network.
- Weights: Parameters of the network taken as input values to the loss function.
- Error Gradients: First-order derivatives of the loss function with regard to the parameters.

Dataset Analysis

In order to conveniently navigate the dataset, those representation could be assumed throughout the Jupiter notebook.

X1 Relative Compactness
X2 Surface Area
X3 Wall Area
X4 Roof Area
X5 Overall Height
X6 Orientation
X7 Glazing Area
X8 Glazing Area Distribution
y1 Heating Load
y2 Cooling Load

Visualization Scheme



- First visualizing the relationship between the features and the responses. We notice immediately that Data is clean and does not have any rows with all the data points being the same
- We can infer from the data that an increase in relative compactness leads to a decrease in heating load. Using visualization library of python
- A decrease in relative compactness is associated with a unit increase in the heating load.
- Therefore, it is preferable for Relative compactness to be kept lower.
- In other hand Surface area is negativly correlated to Relative compactness and need to be kept high.
- Orientation has no significant p-value, in contrast to the other features (Reject the null hypothesis for those features with significant p-values), and fail to reject the null hypothesis for Orientation.
- All attributes are numerical and ranges are quite diffrent. Need to normalize before attempting any regression modelling.
- The attributes distribution is not following Gaussin distribution, hence linear regression kind of algorithms do not perform well.
- Since the range of Heating and Cooling loads are between 0 and 50, also aim is not to find the exact values.

Our task is to Predict heating load as well as cooling load while minimizing the sum-of-squares error function and evaluate the performance by RMS.

Code (hw1_0810976_part1.pynb):
The first function in our code is “build model” which takes the as a parameter the input shape as well as the number of classes which is set to None by default.
This input receives all the preprocessed features which then uses the activation function ‘relu’ to return our model it is also noted that: the 'inputs' parameter of the model must contain the full list of inputs used in the architecture.

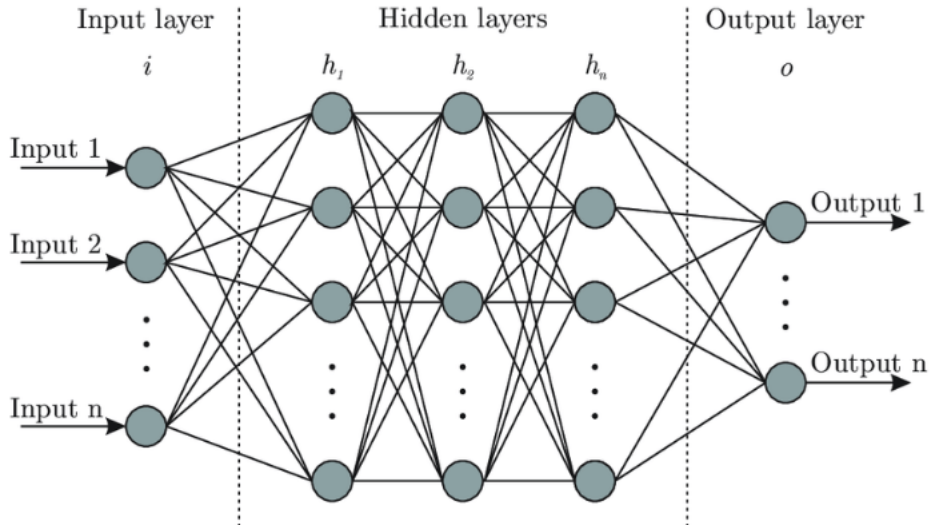
The Normalize function simply perform the normalizing of the array data, which is translate the values to be between 0 and 1 range instead.

The Gradient function takes as an input our x and y as well as theta, which is an array of zeros, as well as the learning rate that we set at 0.1 and the number of epochs that I set to 10. Minibatch size is assumed to be 1.

The function Cost functions takes in x and y and return the cost function formula.

b) Show

Network architecture



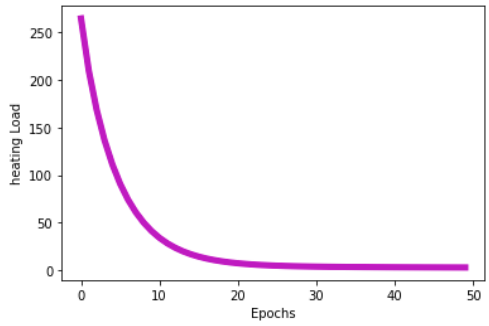
As n = 8 in our case.

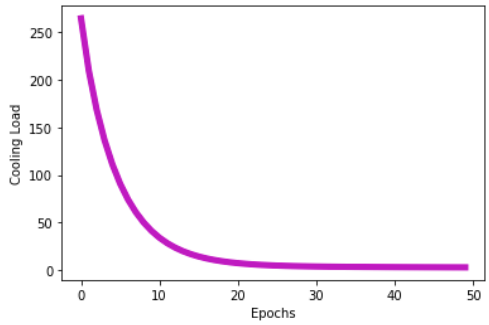
With 2 outputs

And 8 input features.

8 Hidden layers.

Learning curve





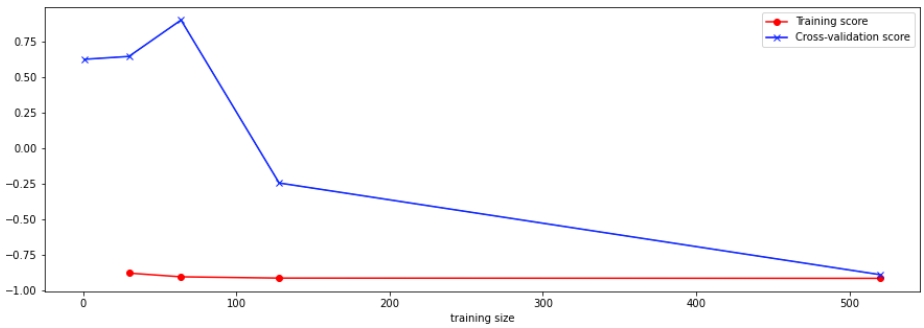
Training RMS error

- 0.916

Test RMS error

- 0.8402

Regression result test (cross validation) labels vs Regression result with training labels



c) Design a feature selection to find which input features influence the energy load, explain why compare using different features:

The energy load is influenced by Use model to make predictions (manually).
A 10% increase in overall relative compactness
translates to a decrease in heating load. Which is: -23.053014+59.359053*10 =
A given amount of the additional features (surface are glazing area dist.), are amongst the input feature which influences the energy load. The p-value for Relative Compactness is less than 0.05, which implies, there is a correlation between Relative Compactness and Heating Load.

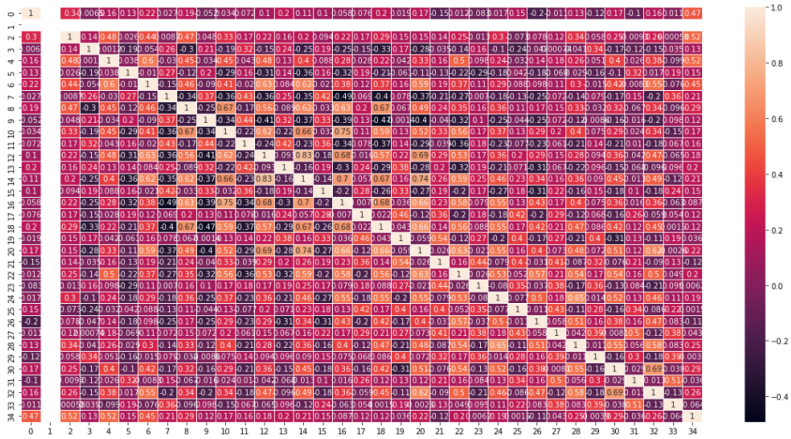
Conclusion

In overall this data set is interesting in its way. All its attributes are categorical and still, the numerical value is a meaningful quantity. The amount of data is not very less or more. And the same time the target variables look continuos and once rounded, their unique values go around 40. The data set is given in CSV format is imported as a pandas data frame. After observations, the data is further converted to NumPy arrays. As we have quantified data in attributes with very different ranges, all are brought to the same range of 0 to 1 using normalization. After this, the data set is split into train and test with 25% of total data going into the test dataset.
The target data is further divided into cooling load and heating load separately to have more control over the final model or models. After data preparation, 6 tree-based models were run on the dataset I run regression functions and cost function to check their performance using the best R2 score. We have so much freedom as the data set is not big. These four models further trained and tuned to their respective target variables in train data (heating and cooling loads). Residuals are visualized to check for any patterns and where exactly the highest errors are occurring. And to finish the various results are highlighted.

Part 2: Ionosphere dataset | Classification

1. Please try to classify the Ionosphere data by minimizing the cross-entropy error function.

Correlation matrix for visualization of dataset



Code (hw1_0810976_part2.pynb):

After loading the dataset and import the relevant libraries, next step is split the data 80/20 for train and test respectively. The data needs to be randomly split, in order to achieve the most unbiased results possible, so we take special care. Next, we create copy of data set and then create training set and test set. The next part is to implement the functions to achieve classification in our code, we have Cross Entropy Loss Function, gradient descent function, fit function, predict function, and evaluate accuracy function.

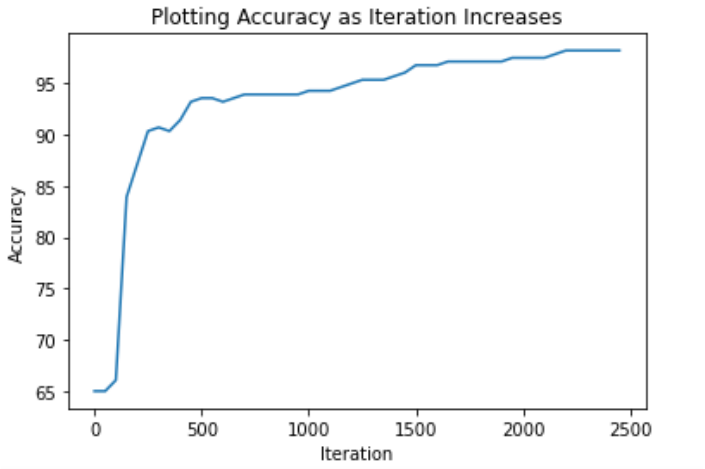
We use the logistic regression classification technique to perform the statistical method and analyzing the dataset as well as predict the different outcomes.

a) Answers

- The size of the input layer is: = 34
- The size of the hidden layer is: = 34
- The size of the output layer is: = 1
- average cross validation model accuracy = 84.0%
- Minimum Cost (J) = 0.32772157249851447
- Recall: 100.0 %

- Accuracy Train: 98.21428571428571%
- Accuracy Test: 88.73239436619718%

Learning accuracy/ iteration curve

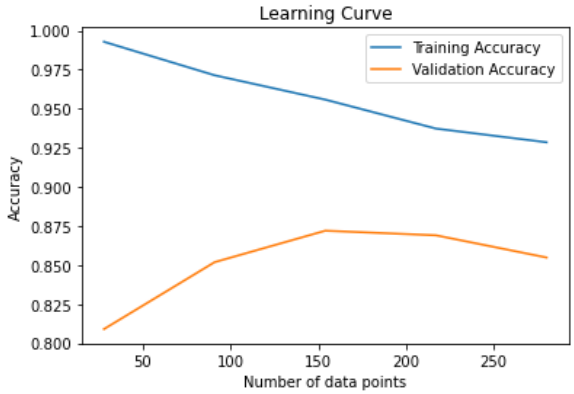


Results of choosing different numbers of nodes in the layer before the output.

Logistic Regression classification Report with Feature Selection

	precision	recall	f1-score	support
b	0.97	0.70	0.82	44
g	0.82	0.98	0.90	62
accuracy			0.87	106
macro avg	0.90	0.84	0.86	106
weighted avg	0.88	0.87	0.86	106

Distribution of latent features at different training stage



Conclusion:

In binary classification, we use one output neuron for every positive class, whose output represents its probability. In multi-class classification, however, we have one output neuron for every class and use softmax activation on the output layer to ensure the final probabilities add up to 1. The most used loss function to optimize is mean squared error unless there are many outliers. The Huber loss or mean absolute error should be used in this case. In contrast, Cross-entropy is most used for classification.

References:

Brownlee, J. (2021). Difference Between Backpropagation and Stochastic Gradient Descent. [online] MachineLearningMastery.com. Available at: https://machinelearningmastery.com/difference-between-backpropagation-and-stochastic-gradient-descent/#:~:text=The%20Stochastic%20Gradient%20Descent%20algorithm.

builtin.com. (n.d.). Why Is Logistic Regression a Classification Algorithm? | Built In. [online] Available at: https://builtin.com/machine-learning/logistic-regression-classification-algorithm.

KDnuggets. (n.d.). Designing Your Neural Networks. [online] Available at: https://www.kdnuggets.com/2019/11/designing-neural-networks.html.