Homework 4

COSC 6342: Machine Learning

University of Houston

Department of Computer Science

Sent on: Oct. 26, 2020; Due: Nov. 4, 2020 (midnight)

Name(s): Garima Singh (1793399)

Ruchi Shah (1800950)

General instructions:

Teams can be made of up to 3 members. Each team member should submit the same final one document. The document should have its code as an appendix. The names and peopleSoftIDs of all the members should be contained in the document (1 page).

Motivation:

The main goal of this assignment is to understand the difference between a Linear regression model and a Neural Network model. You can design a tool to train and monitor a neural network, using which, you should be able to compare and analyze the results with that of a Linear Regression model.

Implementation:

You can use any programming language such as Python, Matlab, etc. You can also rely on any existing toolbox (***Matlab, Tensorflow, Keras, Pytorch,*** etc.) to run a neural network. You can make use of ***Sklearn*** for the Linear Regression implementation. You should link your code to an existing toolbox that already implements a neural network.

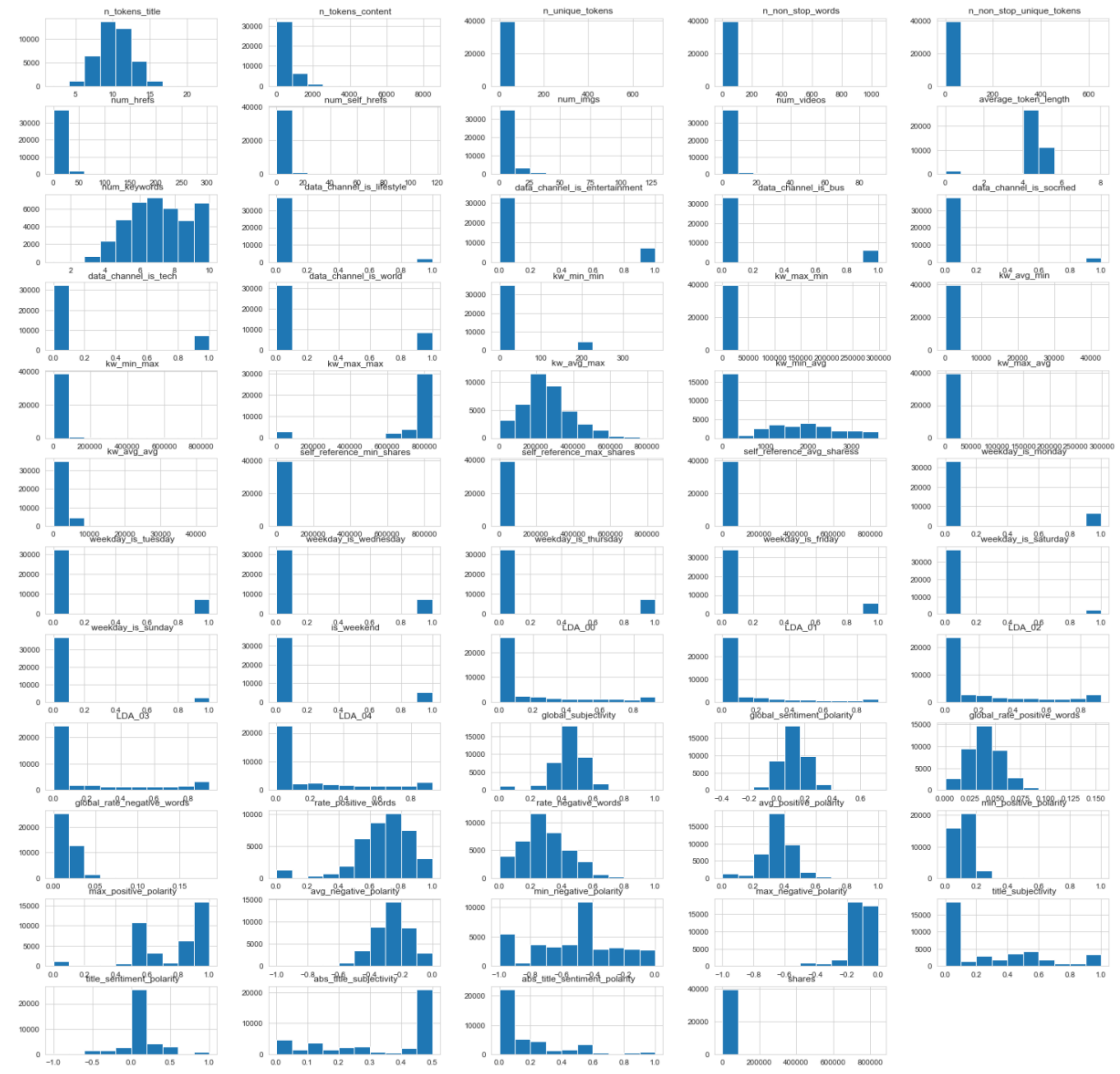
Dataset:

The dataset, its description, and all the attributes can be found here.

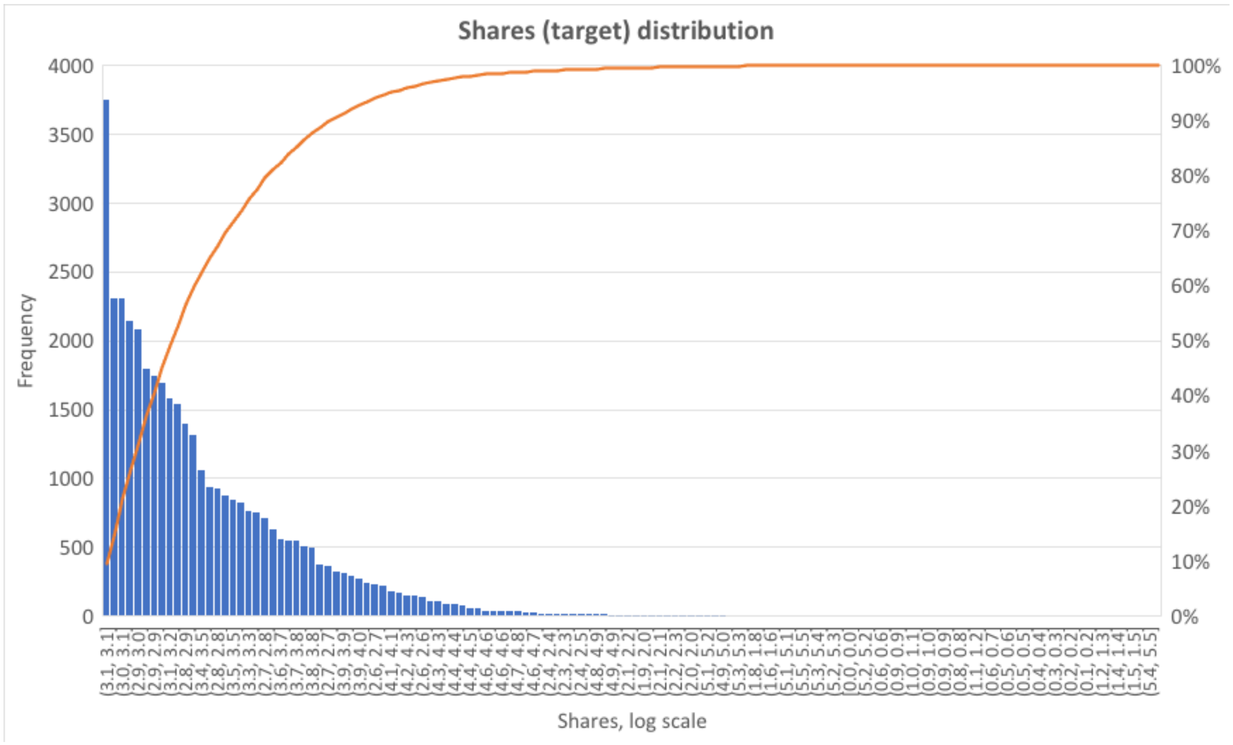
First, you need to divide the dataset into train, validation, and test sets with your own split ratio. Then, build a Linear Regression, and a Neural Network Regressor using the training set and tuned on the validation set. Then, test your model on the test set and report the results. You can use MSEloss for this dataset.

Questions:

1. Perform some exploratory analyses on the dataset (see the examples below but not limited to them).
   To begin with we plotted the histograms for all the features to see how that data is distributed across all the features. We observed that for many features have a normal distribution, some features were categorical others were skewed and the target itself (‘Shares’) have a highly skewed distribution. In fact, the target does not follow a Gaussian distribution.

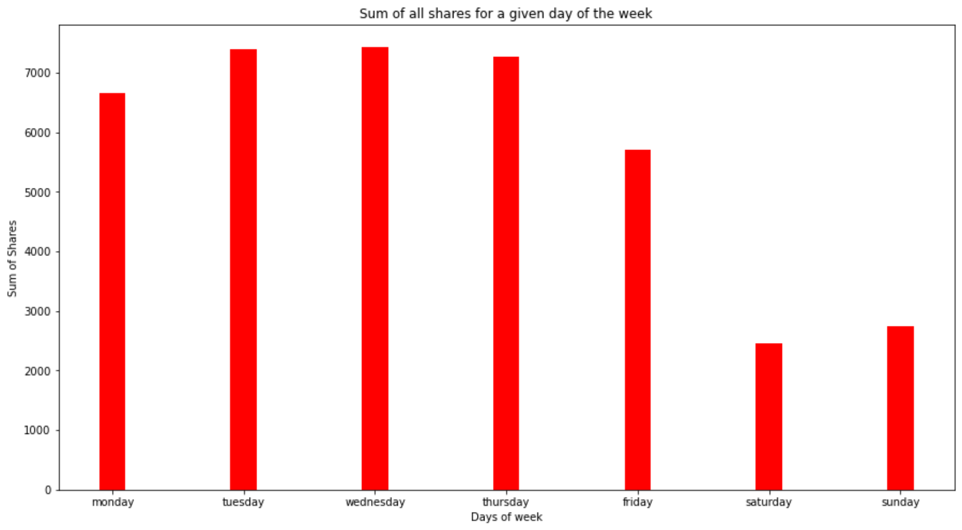


Please refer HW4Q1.py for the above plot.



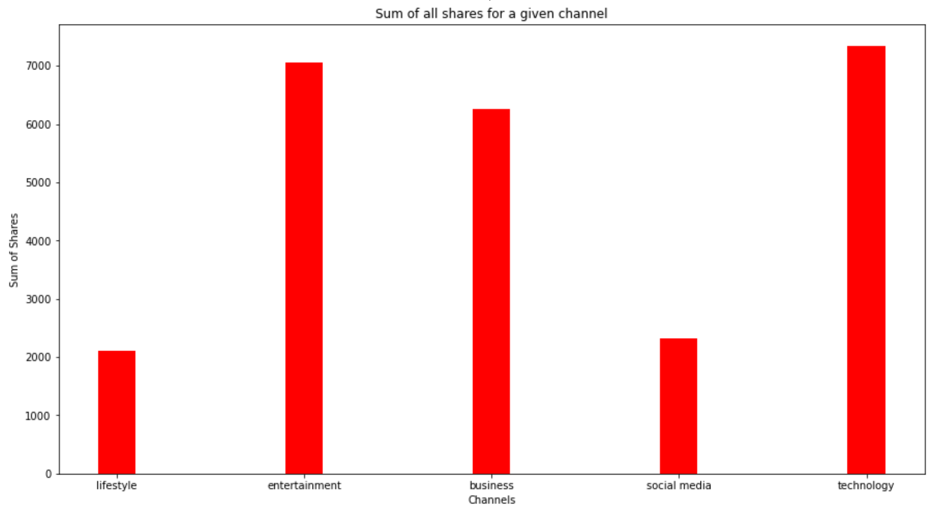
Please refer HW4.xlxs for the above plot.

a. You can plot the sum of shares for a given day of the week



Please refer program HW4Q1.py for the plots.
The above bar chart shows the highest number of shares for a given day of the week. The sum is the highest during the weekday and significantly drops over the weekend. During the week, the sum of shares is highest on Wednesday and we see a relative drop in the sum of shares on Friday.

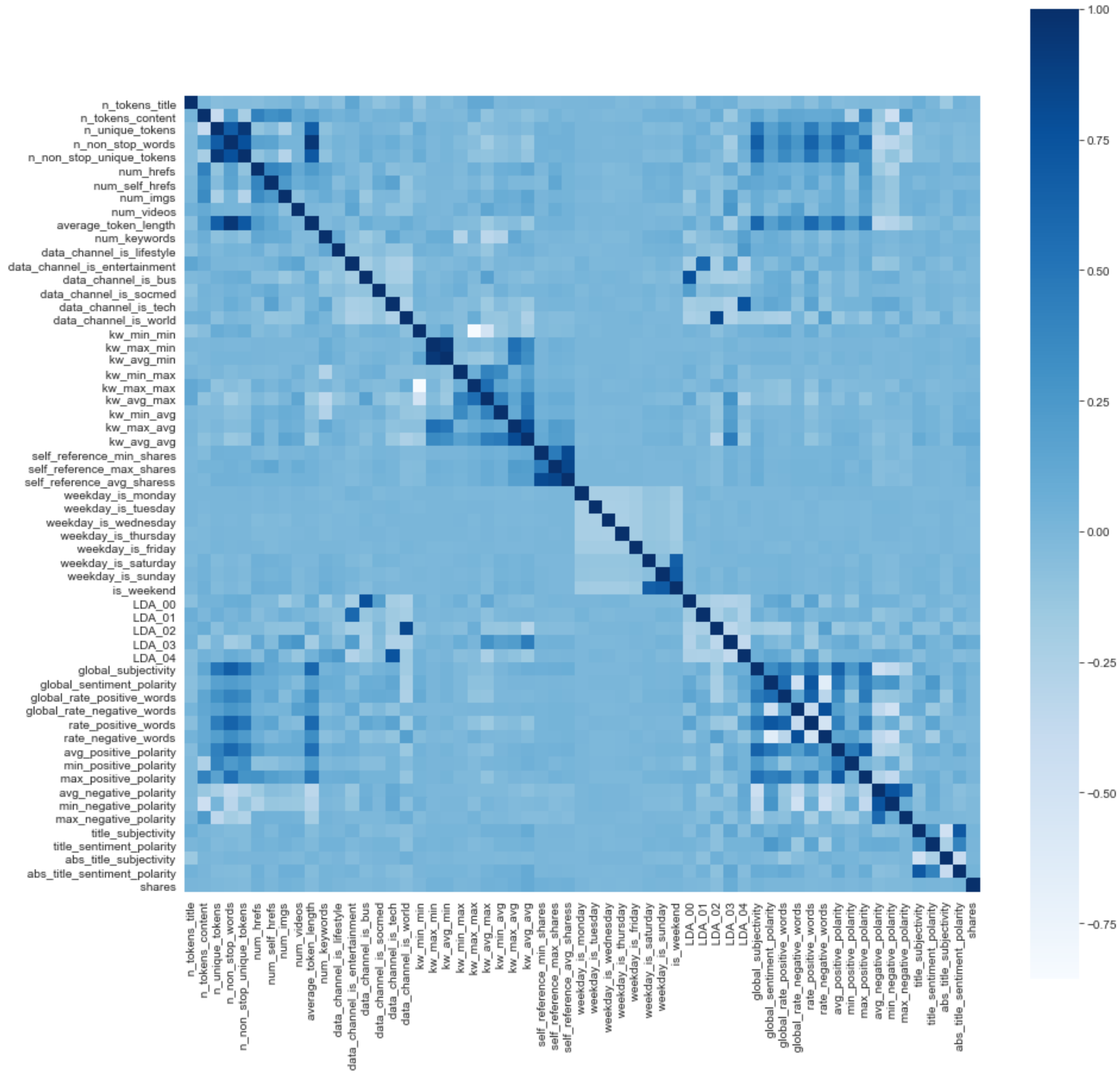
b. You can plot the sum of shares for a given channel (Life_style, entertainment, business, social_media, tech, world)



Please refer program HW4Q1.py for the plots.
The above bar chart gives the best performing channel based on highest number of shares for that channel. We can see that technology is the best performing channel and lifestyle, social media are the least performing channel.

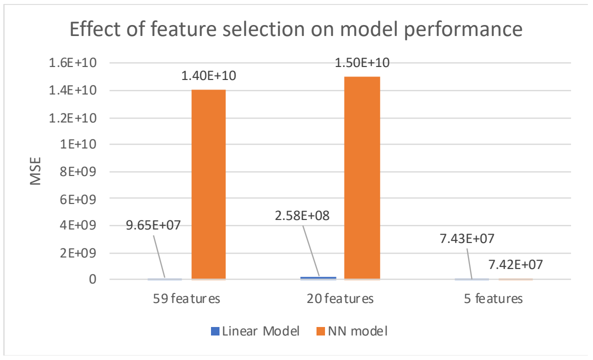
2. Does this dataset require feature selection or in other words, does it need a dimensionality reduction to achieve better performance? If yes, provide some plots to show the relationship of the attributes and briefly explain how you decide on using a subset of attributes for your models.
    a. You can make use of correlation plots from the seaborn package's diverging_palette, heatmap methods in python

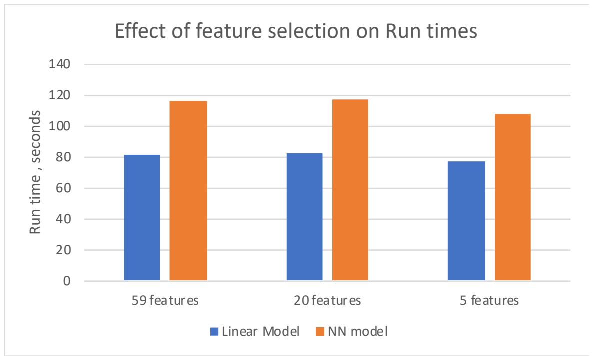
We have used seaborn heatmap to plot a correlation matrix ('Pearson's method') between **all the features** of the dataset. The observations are as follows:



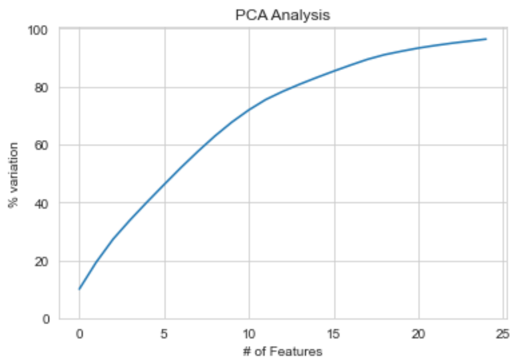
- The first column of the matrix gives an idea of correlation of all features with the target 'shares. We notice they have a very lean correlation (absolute values < 0.1).
- We see some darker clusters which implies some features have a strong correlation with each other as high as 0.95.

- We ranked the absolute values of the correlation for the first column and raked them in descending order using: corr_shares.reindex(corr_shares.abs().sort_values(ascending=False).index)
- We next chose top 20 features and top 5 features to build the linear regression and neural network model and compared the performance in each case using the MSE as the criteria. The comparison for each case is shown in bar charts below.

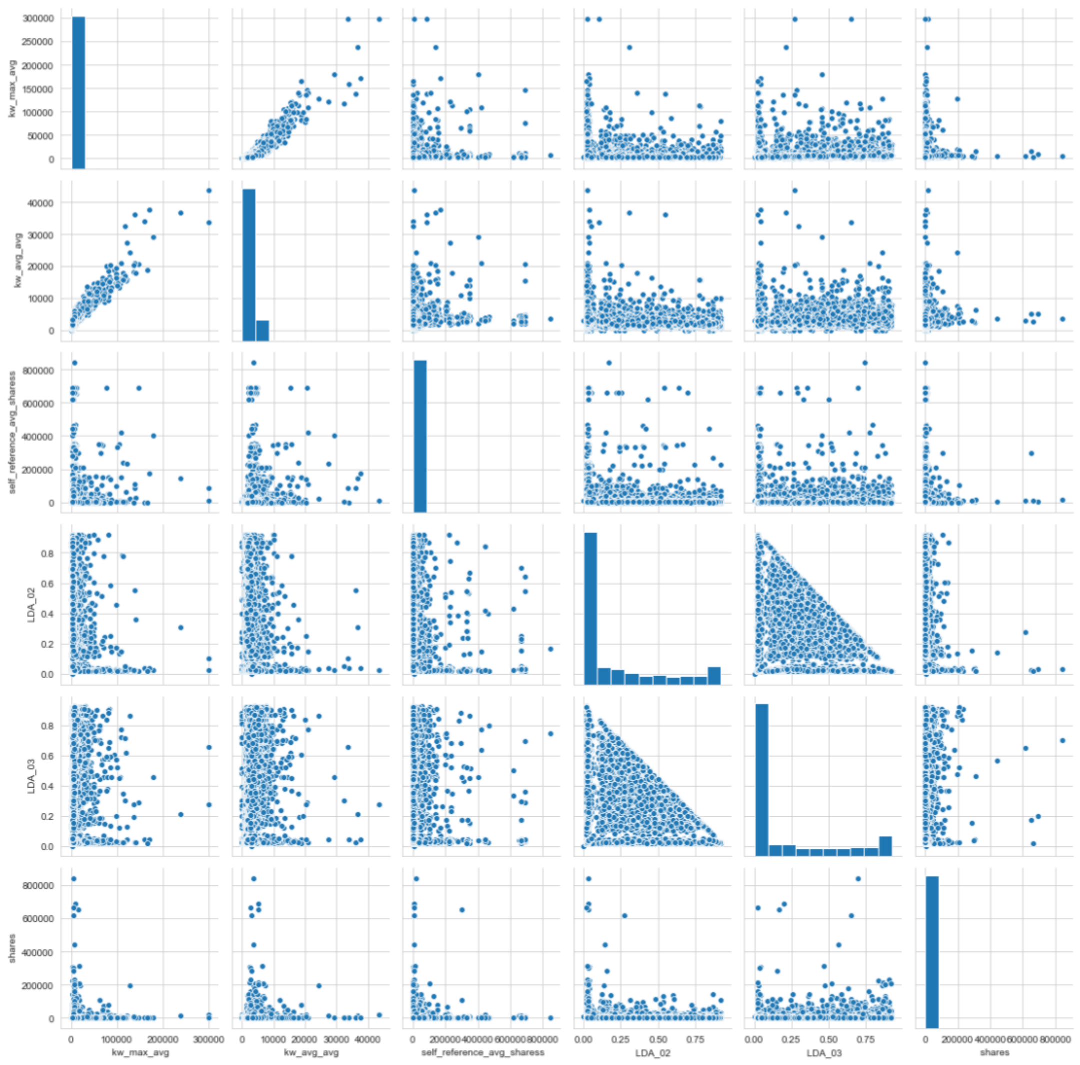




- We find that reducing the features to 5 improves the model performance significantly. Also, feature selection has more impact on the NN model compared to the Linear regression model. Also, the run times reduce slightly for as we reduce the number of features.
- We also considered PCA analysis for dimension reduction, and as seen from the cumulative line plot below, PCA does not help in significant dimension reduction implying the dataset has a very huge co-variance. To capture 90% covariance in the dataset we need 20 dimensions (reducing the dimension of dataset by one-third). To capture ~80% covariance in the dataset we need as many as 10 dimensions.



- Pair wise Scatter plots of top 5 features (shares) using seaborn's pairplot(). It is clear that none of them has a linear correlation with the 'shares'. We note that only 'kw_avg_avg' has a strong linear relationship with 'kw_max_avg'

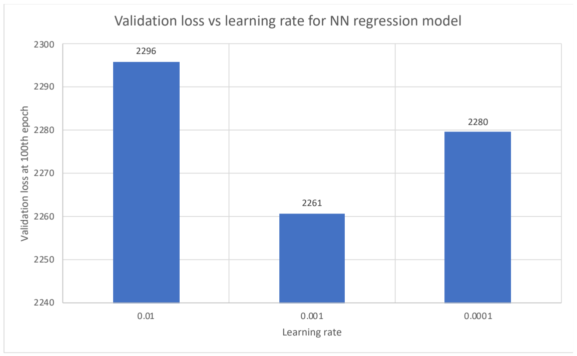


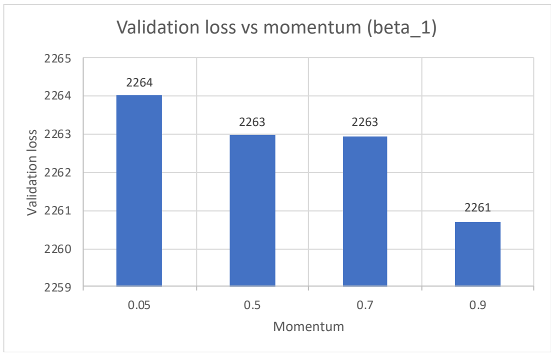
Refer files HW4Q2_scatterplot.py, HW4Q2_PCA.py and HW4.xlxs for the above plots.

3. For the NN model, include
    a. the parameter settings such as the nonlinearity function used, learning rate, momentum, etc.

We have used keras for building the NN model and followed the tutorial for the same.

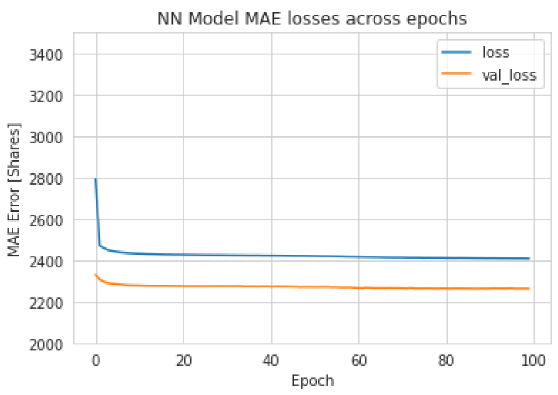
- We used ‘Relu’ activation function and ‘Adam’ optimizer.
- We have tested three learning rates. As can been seen in the figure the learning rate of 0.001 gave the minimum validation loses and was selected for further comparison with the Linear regression model.
- We also tested four different values of momentum by varying the values of parameter ‘beta_1’. As can been seen in the figure the default value of 0.9 gave the minimum validation losses and was selected for further comparison with the Linear regression model.

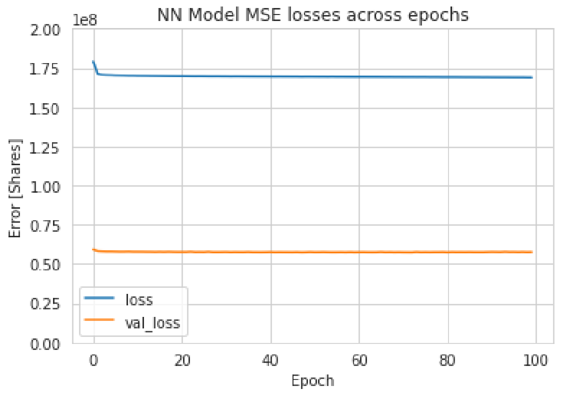


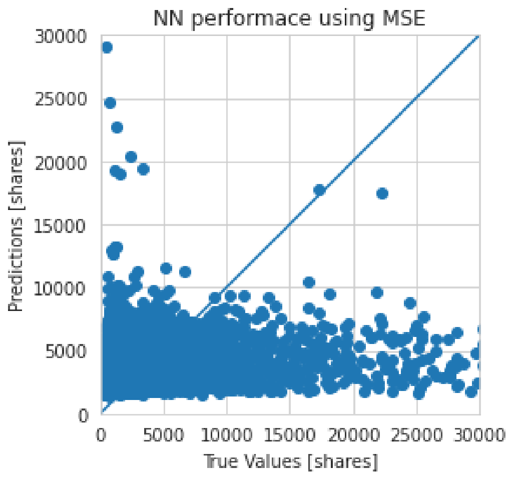


    b. the plots for train and test errors across epochs

- We have run the NN model for 100 epochs and compared the train losses (loss) and validation losses (val_loss).
- We notice that though the losses are huge, but we attain the steady loss quickly in ~20 epochs. We have mean absolute error (MAE) and mean squared error (MSE) for computing the losses.



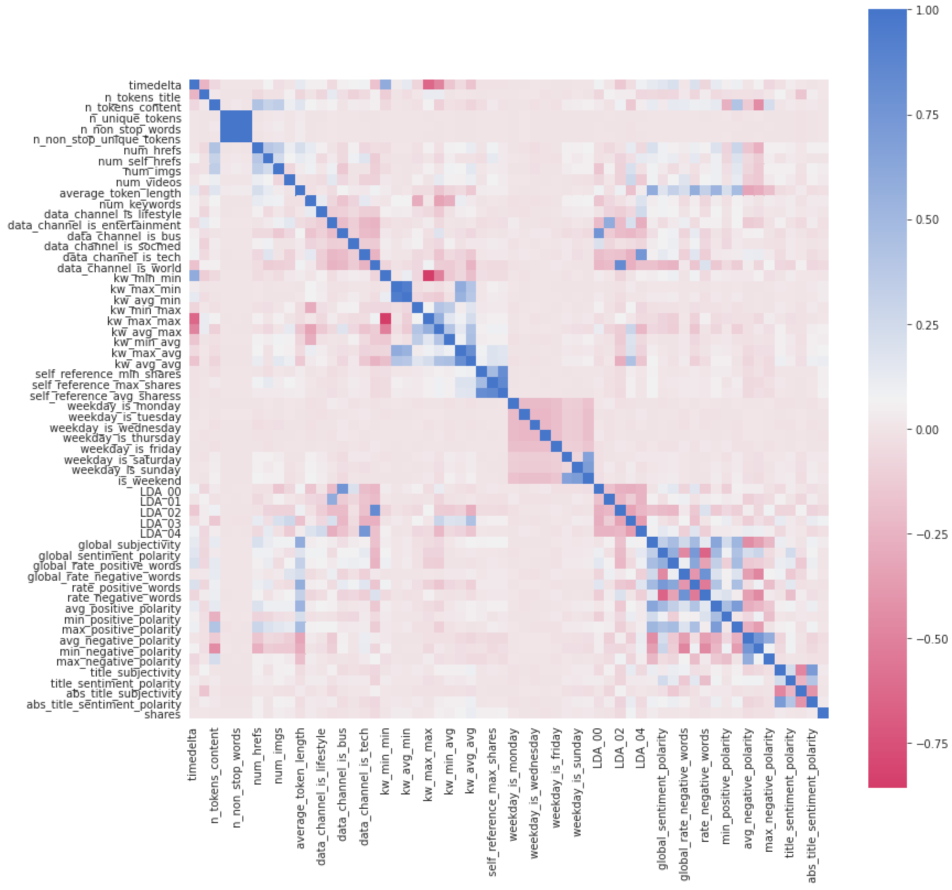




As seen in the scatter plot of predicted values vs true values of ‘shares’ we notice that the actual data has a long tail while the model predicts comparatively lower values. One of the possible reasons we believe is due to the class imbalance, that is there is lot more data for lower share values compared to the high share values (as was evident from the histogram of the target feature ‘shares’.

Please refer HW4.xlxs, HW4Q4_MSE.py, HW4Q4_MSE.py plots.

4. Compare and comment on the output performance of the Linear Regression model with that of the Neural Networks model

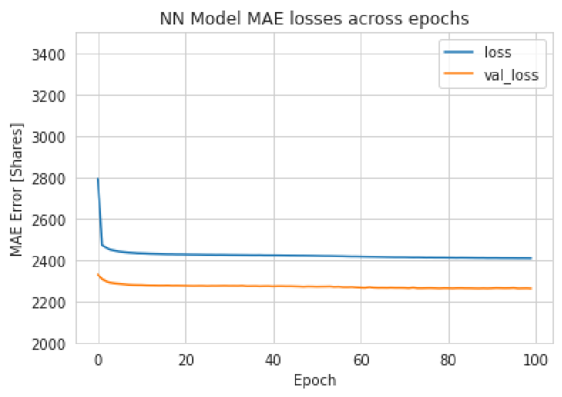


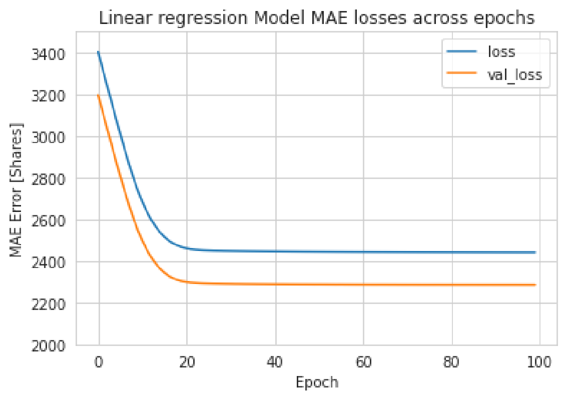
We have used seaborn heatmap to plot a correlation matrix ('Pearson's method') between all the features of the dataset. Based on observation from the heatmap, for this question we have considered only those features which have correlation, 95% or more to perform the regression analysis. This map helped us identify strong linear relationships within the data. This dataset consists of maximum non-linear relationships; therefore, it is nearly impossible to get a good score on our Linear Regression analysis using sklearn linear model

Please refer to program HW4Q4.py. In our simple linear regression model (scikit-learn), we have scaled the input data using the MinMaxScaler. The regression analysis score on test data is really bad. The variation is ~99% after scaling the data and considering only the highly correlated features. The MSELoss is also very minuscule.

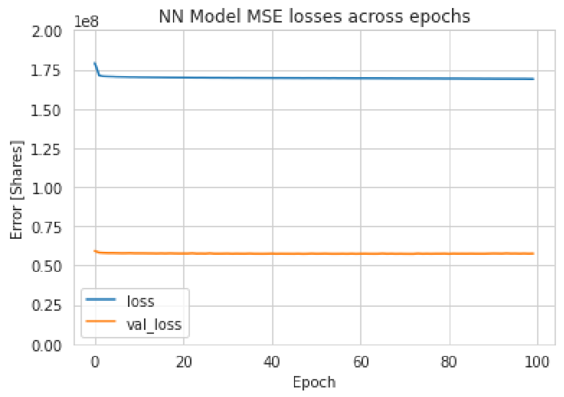
```
Linear Regression Analysis
Regression Analysis score on validation data : 1.0
Regression Analysis score on test data : 0.9999999999892222
MSELoss on validation data : 2.98754906798985e-21
MSELoss on test data : 0.00082022622451592
```

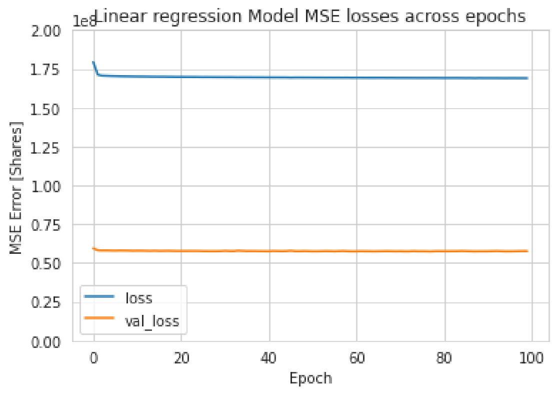
We then performed a simple regression using the Neural Networks (NN) using the tensorflow keras package. This Sequential NN model is also a sequence of linear operations however, due to the non-linear component (the activation function), it allows identification of linear relationships in a non-linearly separable dataset. The activation function used here is ‘Relu’. We have also normalized the input data prior to training to get better results.





When **MAE** is used as a performance metrics, NN model comparatively has lower validation loss in the initial epochs than the linear model and hence achieves the final value much faster (<10 epochs) while for Linear model similar value of loss is achieved in > 20 epochs.
The final validation losses for the NN is 0.53% lower than that of Linear regression model.

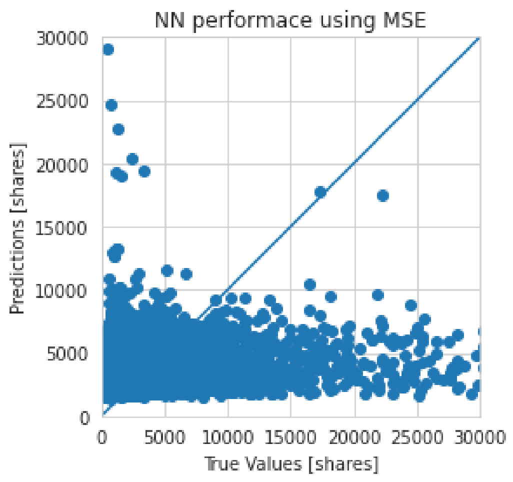


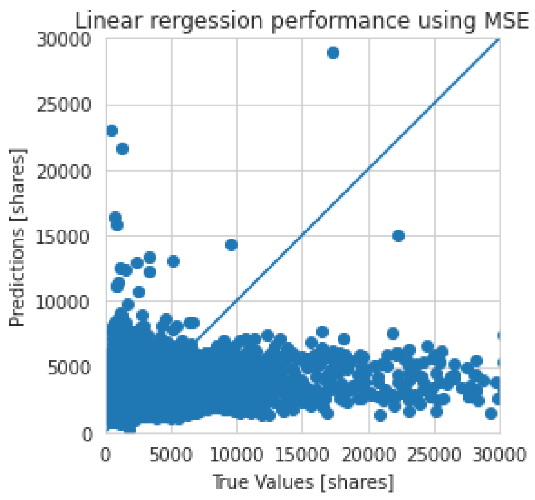


When **MSE** is used as performance metrics, NN and LR both have similar performance in terms of error and convergence across epochs (< 10) as can be seen from the graphs above.

The final validation losses for the NN is 0.15% lower than that of Linear regression model. he loss value.

Please refer files HW4Q4_MSE.py and HW4_MAE.py for the above epoch plots.





Deliverables:

Option1: an IPython notebook or an IPython HTML file with all the questions answered in it

Option 2: A PDF file with all the questions answered along with a file that has the code

Helpful readings and links:

Tutorial for using Pytorch to build a NN Regression model,

Tutorial for using Keras to build a NN Regression Model

You can use Googl e Colab to implement your models and you can also exploit the GPU resources that it provides for your Neural Networks implementation. Finally, you can download the IPython file or the Python file, or an HTML file from the colab notebook, which you can use as a final submission. Please note that the Colab notebook's session terminated after a certain time of inactivity.

Also, you can take this blog post as a reference for this homework.