CSC215-01 Artificial Intelligence (Spring 2019)

Mini-Project 2: Yelp Business Rating Prediction using Tensorflow

Due at 4 pm, Wednesday, February 27, 2019

Team Members :

Palak Patel (219681801)

Pallavi Yadkikar (220264292)

Problem Statement :

In this project, we aim to predict a business's stars rating using the reviews of that business and review count based on neural network implementation in Tensorflow.

Methodology :

- **Data Pre-processing to gain better accuracy.**
- How to do feature normalization (zscore and min-max scaling).
- Tf-IDf vectorizer for text data.
- Applied Linear Regression, Logistic Regression, KNN, SVM and Multinomial Naïve Bayes.
- **Tensorflow regression neural network model**
- **Tensorflow classification neural network model**
- Applying the models and generating their scores and comparing their performance.
- Parameter tuning to compare model with different parameters.

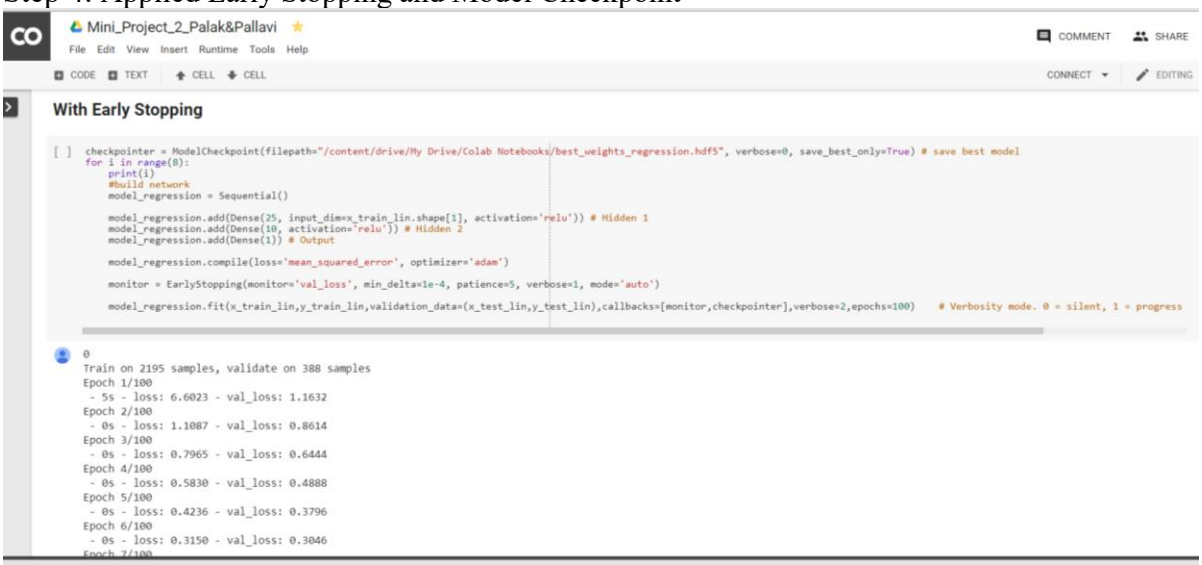
Experimental Results and Analysis

Step 1 : Followed all steps of Project-1 with Additional Feature(Data Pre-Processing).

Step 2: Applied Linear Regression, Logistic Regression, KNN, SVM and Multinomial Naïve Bayes.

Step 3: **Tensorflow regression neural network model**

Step 4: Applied Early Stopping and Model Checkpoint



Step 5 : Parameter tuning for regression

Step 6: Getting X and Y(One Hot Encoding) ready for Tensorflow Classification model.

```
# pass in array and columns
df_y_class = pd.DataFrame(normal_y , columns=columns_new)
```

In [92]:
```
encode_text_dummy(df_y_class,'normal_y')
```

In [94]:
```
df_y_class[:5]
```

Out[94]:

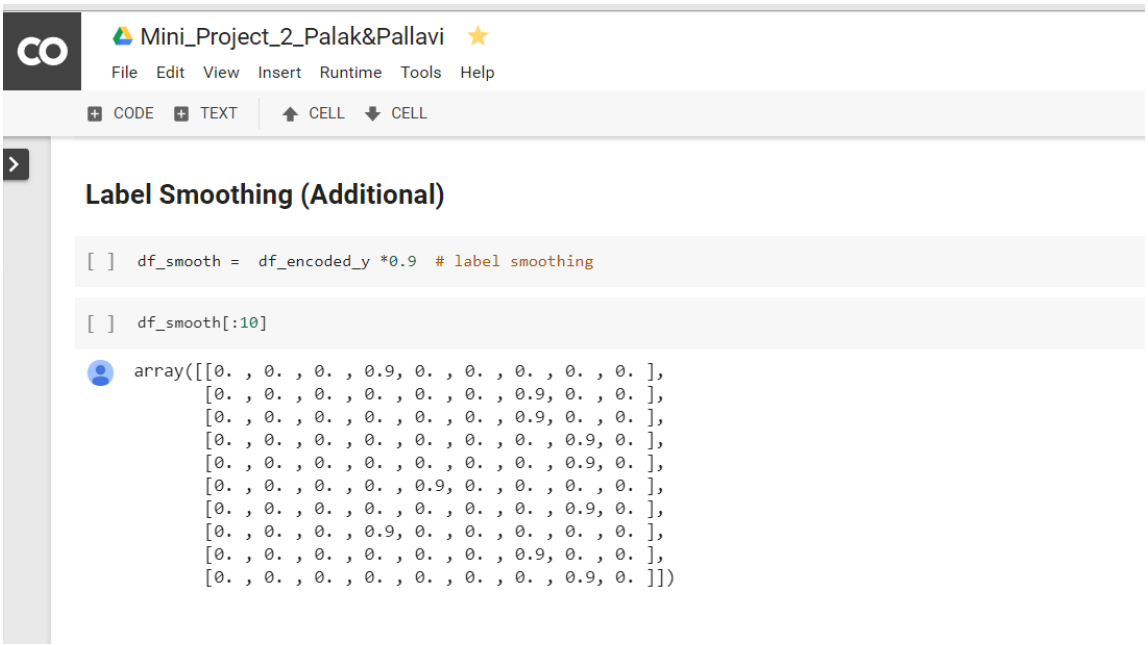
	normal_y-0	normal_y-1	normal_y-2	normal_y-3	normal_y-4	normal_y-5	normal_y-6	normal_y-7	normal_y-8
0	0	0	0	1	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0
2	0	0	0	0	0	0	1	0	0
3	0	0	0	0	0	0	0	1	0
4	0	0	0	0	0	0	0	1	0

In [96]:
```
df_encoded_y = df_y_class.values
```

In [97]:
```
df_encoded_y[:10]
```

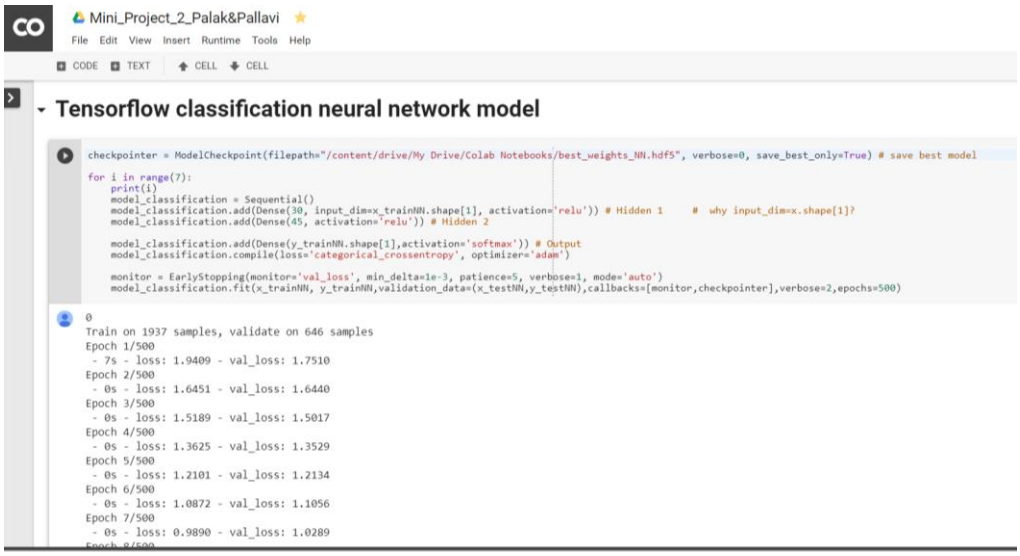
Out[97]:
```
array([[0, 0, 0, 1, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 1, 0],
       [0, 0, 0, 0, 0, 0, 0, 1, 0],
       [0, 0, 0, 0, 1, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 1, 0],
       [0, 0, 0, 1, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 1, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 1, 0]], dtype=uint8)
```

Step 7: Label Smoothing **(Additional Feature)**



Step 8: **Tensorflow classification neural network model**

Step 9 : Applied Early stopping and Model Checkpoint



Step 10 : Parameter tuning for classification

Step 11 : Applying the models and generating their scores and comparing their performance.

Task Division and Project Reflection :

Pallavi Yadkikar :
Worked on CoLab.
Applied Tensorflow classification and Regression models
Parameter Tuning
Worked on Additional Feature

Palak Patel :
Worked on System and Complete Data Set
Applied Tensorflow classification and Regression models
Parameter Tuning
Worked on Additional Feature

Challenges :

SGD was taking a lot of time to execute, so we had to do data preprocessing

What we have learned from the project as a team :

- How to manage the time efficiently and dividing the work and collaboration
- How to deal with Tensorflow and regarding models
- How to avoid overfitting of models
- Why and how to do label smoothing

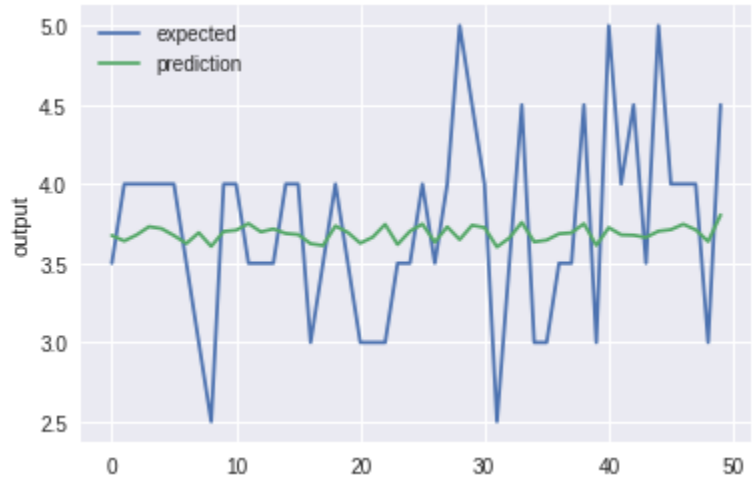
Additional Features :

- Regression:
  - Parameter Tuning

- Classification:
  - Label Smoothing
  - Parameter Tuning

- **L1 & L2 Regularization in TensorFlow**

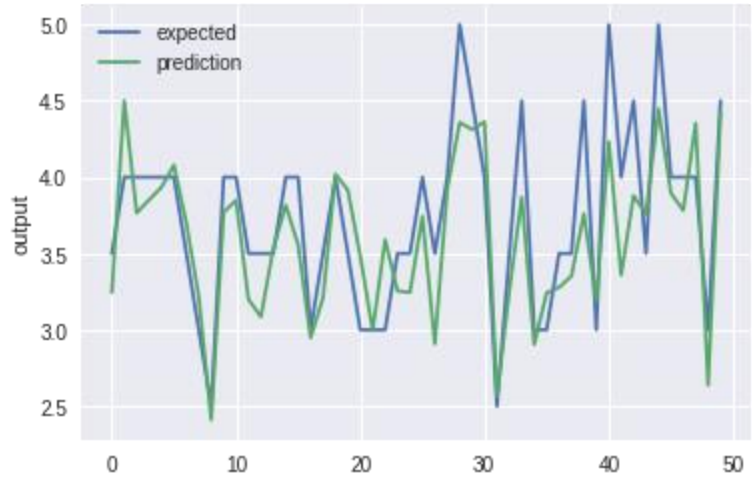
  - L1 and L2 regularization work by adding a weight penalty to the neural network training. This penalty push the connection weights to small values.



It does not fit well in our project that it gave us RMSE as 0.669, which is not so good.

- **Implementing Dropout Layer**

  - Each dropout layer will drop neurons in its previous layer. A dropout layer can be added between any two hidden layers to reduce overfitting.) It turned out to be the best model parameters so far. **(RMSE : 0.318)**



L1/L2 with TensorFlow(DropOut Layer)	**RMSE** **0.318**

- Calculate F1, Precision and Recall Score for data to check accuracy before and after preprocessing it.
- Plotting test data vs prediction using Matplotlib library.

Model Evaluation for <u>Regression</u> with Tensor Flow:

Parameter Type	RMSE
Adam & Relu (Without Early Stopping)	0.347
Adam & Relu	**0.320**
SGD & Sigmoid	0.641
SGD and tanh	0.636

Model Evaluation for Classification with Tensor Flow:

Parameter Type	Precision Score	Recall Score	F1 Score
Adam & Relu	**0.691**	**0.698**	**0.693**
Label Smoothed Y	0.678	0.682	0.677
Research Paper(4 Hidden Layers with Adam and Relu)	0.565	0.628	0.587
RMSPROP with Relu	0.680	0.682	0.677
RMSPROP with Sigmoid	0.669	0.687	0.676
SGD with Sigmoid	0.092	0.304	0.142
Adagrad with Relu	0.675	0.687	0.679

Analysis:

1. The best regression neural network model is having RMSE score as **0.318**. It was generated from activation = **RELU** and optimizer = **adam** with 3 hidden layers having 50, 25 and 1 neuron respectively, Its included with **dropout** function.
   Whereas Linear regression of SciKit Learn is having RMSE 0.369.

2. The best classification for neural network model is having precision score as **0.691** was generated from **2 hidden layers with activation= RELU and optimizer= adam**. Hidden layers were having 30 and 45 neurons respectively.

3. **SVM model was the best among all SciKit Learn models and was having precision score as 0.610**

4. **We experienced that SGD optimizer was taking much more time while training the model, It is evident from the no. of epochs it took before early stopping.**