Applied Artificial Intelligence Project -5

Nickname: BohemianStarman
Domain: Otto Product Classification Challenge

Background: The Otto Group is one of the world's biggest e-commerce companies and they set a challenge in Kaggle to find a machine learning model that can classify their products into the 9 categories provided.
Winners Approach: The winner of this competition had a cross-validated score of 0.38243. It was a team comprising of two senior data scientists that used 33 models and 8 engineered meta features in the first layer and fed all of these into a XGBoost, Neural Network and Adaboost ExtraTree Classifier in second layer and then took the results and weighted their averages with both exponential and multiplicative weights.

My approach: I took a much simpler and less computationally intensive approach. Here are the steps that I undertook to try and solve this problem:

1) Initially, I explored the dataset to get an understanding of the features and nature of the dataset.
2) I wrote a R script that found the **PCA dimensionality reduction.** On plotting the feature importance and eigenvector scree and score plot, I realized that it isn't possible to reduce features for this dataset.
3) For dimensionality reduction, I also tried **t-sne dimensionality reduction** that produced some interesting plots that gave actionable insight into the data.
4) After this, I decided to use on of the most popular methods in data science (especially in categorical classification of data) called XGBoost (Extreme Gradient Boosting). It is a variant of gradient boosted trees. I created **5 separate models of XGBoost** with varying parameters with lowest giving me 0.45 loss on the test set.
5) In other decision tree algorithms, I tried **random forest classification(with different seeds)** with an CV score of 0.7.
6) I also used **ExtraTree Classifier** and got a cross-validated score of 0.7.
7) Moving over to **K Nearest Neighbour**, I tried KNN with varying number of neighbours **(from 2 to 1024) and manhattan and euclidean distances.**
8) Moving over to deep learning, I used neural networks to try and train the dataset for classification. After running around **10+ neural network models**, the lowest CV score was in the range of 1.8. However, I was not able to tune the model very well and I stopped with the score of 1.8.
9) My idea was to use all the above models as meta-models to a XGBoost to create an **ensemble model** and then average the results with other xgboost models. This resulted in the predictions being used as features to a XGBoost. Interestingly, I found that XGBoost with Random Forests as my meta-models gave me better CV score than XGBoost. (between 0.37 - 0.40). However, I suspect this is due to overfitting. I did not get this cross validated with kaggle submission yet.

Libraries required (Python): Keras, Tensorflow, scikit-learn, xgboost, numpy and pandas.
R libraries can be imported in RStudio (mentioned on top of script).
Softwares required: Python 2.7, R and RStudio

Final cross-validated score :

Future Work:

1) Fine tuning the parameters of neural network should give a boost in accuracy.
2) The code for ensembling is included and can be run. Various configurations yield reduced CV scores. Validating by submission has not been added yet.
3) More important meta-features and a deeper ensembling technique can push to a lower error.