PLACES CATEGORIZATION APPROACH

The overall approach to creating model to categorize places can be broken down into three parts –

1) Data Inspection and Exploration
2) Classical Machine Learning Model
3) Deep Learning Models

1 – Data Inspection and Exploration

The data looked a little unclean at first glance and upon further inspection there were many outliers in terms of length of the characters. Place Names are not that long, and the max character length was 487 which seemed a little odd. Further drill down revealed that certain rows had their place names in multiple languages clubbed together. So only the first entry from each row was extracted. Then regular cleaning of text data was done by removing unnecessary whitespaces, special characters and converting all text to Unicode.

2 – Classical Machine Learning Model

When it comes to classical machine learning models, two approaches can be taken on place names as they do not contain many words

a) Char level model
b) Word level model

Both approaches were tried by creating tf-idf document term matrix on char level n-grams and word level n-grams. Then the data was fed through multiple models to see their performance. A split of 80-20 was taken to train and test the model. Since the classes were uniformly distributed, accuracy was chosen as the metric to evaluate models.

The machine learning models used were

1) Naïve Bayes – Advantage – Speed, Disadvantage – Assumes the variables are independent of each other
2) Random Forest – Advantage – Reduces variance by averaging results of multiple bagged trees, Disadvantage – Takes very long to train, especially when number of trees are large
3) Xgboost – Advantage – Learns sequentially and usually outperforms naïve bayes or Random Forest, Disadvantage – Prone to overfitting and requires extensive hyper-parameter tuning.

We found comparable performance when it came to both char level model and word level models on all three models.

3- Deep Learning Methods

Two approaches were compared in deep learning.

1) Building LSTM model from scratch using Keras
2) Using Transfer learning on pre-trained NLP models
    a. Fastai’s ULMFit
    b. Google’s BERT

LSTM model from scratch barely outperformed the word/char level machine learning models. With enough time and computational resources, the results achieved by Keras would

have been matched by Xgboost also. However, we saw significant improvements when we used transfer learning with NLP. Both ULMFit and BERT achieved good accuracy on test sets and even performed well on categories on which classical models were struggling.

Finally, an ensemble will be created on the best performing models to see if the ensemble can beat the standalone model.