Building a custom Estimator API: Task 2

As a part of the first task a predefined TensorFlow estimator - tf.estimator.DNNClassifier was used to determine if a customer would buy a term deposit product. Although the predefined estimator API is straightforward in its approach and can provide a quick predictions of the required classes these might not allow sufficient flexibility in modifying the hyperparameters associated with the model. A custom estimator API was built to train a tensorflow estimator with modified hyperparameters including - learning rate, activation functions, evaluation metrics, etc.

Training the Estimator API

The training set consisting of feature engineered data set created for the first task is used for building the estimator. In order to build the custom estimator API a model function is defined that specifies the characteristics of the estimator API with parameters including the number of hidden layers, the activation functions for each layer, the loss function to minimize, the optimization function to be used along with the evaluation metrics. The custom estimator consists of two hidden layers with each hidden layer containing 30 nodes/neurons each. The learning rate was set to 0.005. A few runs with different learning rates were carried out but a value of 0.005 was found to be optimal in terms of the results obtained. Activation functions including relu, relu6, crelu, elu and softplus were attempted, and crelu was chosen as it provided higher values of accuracy and F1 score over multiple runs. The sparse softmax cross entropy was chosen as the loss function to be minimized using a Gradient Descent Optimizer. The F1 score was chosen as the evaluation metric. Upon reading the data, the estimator object is built using the above defined model function and model parameters and fit with the data to be trained over 10000 steps.

By adjusting the learning rate, activation functions, weight parameters we were able to provide a model with significantly higher F1 score (0.4) than one without these specific parameters (F1 ~0.29).

Testing the Estimator API

The estimator API was tested on the test data set used from task 1. The results for this test set produces about ~82% accuracy across several runs with approximate F1 scores in the range ~0.39

Testing with New Examples

In addition to testing the model with the validation set, a new dataset was provided to obtain model predictions along with the probability score associated with the predicted class.