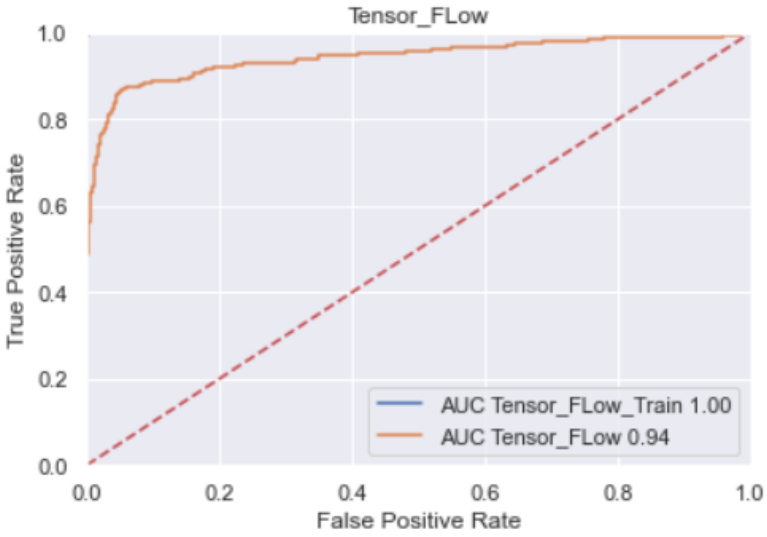
MSDS-422-SEC56_Assignment 4 Daniel Zaremba

TensorFlow Observations

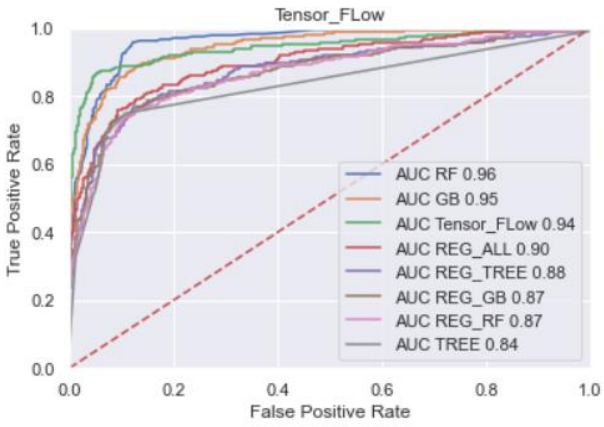
TensorFlow model using all variables:




```
Tensor_FLow CLASSIFICATION ACCURACY
======
Tensor_FLow_Train  =  0.9989513422818792
Tensor_FLow  =  0.9328859060402684
------
```

TensorFlow model using Gradient Boosting model variables:

```
Tensor_FLow RMSE ACCURACY
======
Tensor_FLow_Train  =  4320.05542647871
Tensor_FLow  =  4910.844546906779
------
```



```
ALL CLASSIFICATION ACCURACY
======
Tensor_FLow  =  0.9328859060402684
RF  =  0.910234899328859
GB  =  0.9085570469798657
REG_TREE  =  0.8833892617449665
REG_ALL  =  0.8825503355704698
TREE  =  0.8791946308724832
REG_GB  =  0.87248322147651
REG_RF  =  0.8666107382550335
------


ALL DAMAGE MODEL ACCURACY
======
GB  =  3177.9937332803356
RF  =  3506.6713918019404
REG_ALL  =  3926.760224661266
REG_TREE  =  4678.000953486434
REG_RF  =  4730.411757005278
REG_GB  =  4730.411757005278
Tensor_FLow  =  4910.844546906779
TREE  =  6133.05498303148
------
```

Observations

Having played around with the TensorFlow parameters, I initially had the units at two times the gradient boosting model variable count divided by three and the epoch at 100 this resulted in a test accuracy of .875, changing the unit to be just two times the variable count, increased the epoch to 500 and fed the TensorFlow model all variables resulted in a lift in test accuracy to .933. I would recommend using the TensorFlow model as it has the highest classification accuracy. I would also recommend using the Gradient Boosting model for predicting Loss Amounts. I tried different activation functions like softmax and sigmoid but these both performed worse than relu.

The Default predictive variables for the most part make sense, M_DEBTINC makes sense as individuals that don't provide their debt to income ratio may be hiding high debt and be more risky, on the other hand I'm not sure about the M_VALUE field, I would reach out to a subject matter expert to see if a missing home value is predictive of default. With respect to the Loss Amount variables again we see M_DEBTINC and also TRUNC_IMP_DEBTINC which makes sense, also showing as predictive is TRUN_IMP_CLNO which is a double edge sword, the fewer the tradelines the riskier but any individual may have a lot of tradelines and run up a lot of debt which is also risky.