Otto Group Product Classification Challenge

1st PLACE SOLUTION - Gilberto Titericz & Stanislav Semenov

First, thanks to Organizers and Kaggle for such great competition.

Our solution is based in a 3-layer learning architecture as shown in the picture attached.
-1st level: there are about 33 models that we used their predictions as meta features for the 2nd level, also there are 8 engineered features.
-2nd level: there are 3 models trained using 33 meta features + 7 features from 1st level: **XGBOOST**, Neural Network(**NN**) and **ADABOOST** with ExtraTrees.
-3rd level: it's composed by a weighted mean of 2nd level predictions.
All models in 1st layers are trained using a 5 fold cross-validation technique using always the same fold indices.

The 2nd level we trainned using 4 Kfold random indices. It provided us the ability to calculate the score before submitting to the leader board. All our cross-validate scores are extremely correlated with LB scores, so we have a good estimate of performance locally and it enabled us the ability to discard useless models for the 2nd learning level.

Models and features used for 2nd level training:
X = Train and test sets

-Model 1: RandomForest(R). Dataset: X
-Model 2: Logistic Regression(scikit). Dataset: Log(X+1)
-Model 3: Extra Trees Classifier(scikit). Dataset: Log(X+1) (but could be raw)
-Model 4: KNeighborsClassifier(scikit). Dataset: Scale(Log(X+1))
-Model 5: libfm. Dataset: Sparse(X). Each feature value is a unique level.
-Model 6: H2O NN. Bag of 10 runs. Dataset: sqrt(X + 3/8)
-Model 7: Multinomial Naive Bayes(scikit). Dataset: Log(X+1)
-Model 8: Lasagne NN(CPU). Bag of 2 NN runs. First with Dataset Scale(Log(X+1)) and second with Dataset Scale(X)
-Model 9: Lasagne NN(CPU). Bag of 6 runs. Dataset: Scale(Log(X+1))
-Model 10: T-sne. Dimension reduction to 3 dimensions. Also stacked 2 kmeans features using the T-sne 3 dimensions. Dataset: Log(X+1)
-Model 11: Sofia(R). Dataset: one against all with learner_type="logreg-pegasos" and loop_type="balanced-stochastic". Dataset: Scale(X)
-Model 12: Sofia(R). Trainned one against all with learner_type="logreg-pegasos" and loop_type="balanced-stochastic". Dataset: Scale(X, T-sne Dimension, some 3 level interactions between 13 most important features based in randomForest importance)
-Model 13: Sofia(R). Trainned one against all with learner_type="logreg-pegasos" and

loop_type="combined-roc". Dataset: Log(1+X, T-sne Dimension, some 3 level interactions between 13 most important features based in randomForest importance)
-Model 14: Xgboost(R). Trainned one against all. Dataset: (X, feature sum(zeros) by row). Replaced zeros with NA.
-Model 15: Xgboost(R). Trainned Multiclass Soft-Prob. Dataset: (X, 7 Kmeans features with different number of clusters, rowSums(X==0), rowSums(Scale(X)>0.5), rowSums(Scale(X)< -0.5))
-Model 16: Xgboost(R). Trainned Multiclass Soft-Prob. Dataset: (X, T-sne features, Some Kmeans clusters of X)
-Model 17: Xgboost(R): Trainned Multiclass Soft-Prob. Dataset: (X, T-sne features, Some Kmeans clusters of log(1+X))
-Model 18: Xgboost(R): Trainned Multiclass Soft-Prob. Dataset: (X, T-sne features, Some Kmeans clusters of Scale(X))
-Model 19: Lasagne NN(GPU). 2-Layer. Bag of 120 NN runs with different number of epochs.
-Model 20: Lasagne NN(GPU). 3-Layer. Bag of 120 NN runs with different number of epochs.
-Model 21: XGboost. Trained on raw features. Extremely bagged (30 times averaged).
-Model 22: KNN on features X + int(X == 0)
-Model 23: KNN on features X + int(X == 0) + log(X + 1)
-Model 24: KNN on raw with 2 neighbours
-Model 25: KNN on raw with 4 neighbours
-Model 26: KNN on raw with 8 neighbours
-Model 27: KNN on raw with 16 neighbours
-Model 28: KNN on raw with 32 neighbours
-Model 29: KNN on raw with 64 neighbours
-Model 30: KNN on raw with 128 neighbours
-Model 31: KNN on raw with 256 neighbours
-Model 32: KNN on raw with 512 neighbours
-Model 33: KNN on raw with 1024 neighbours
-Feature 1: Distances to nearest neighbours of each classes
-Feature 2: Sum of distances of 2 nearest neighbours of each classes
-Feature 3: Sum of distances of 4 nearest neighbours of each classes
-Feature 4: Distances to nearest neighbours of each classes in TFIDF space
-Feature 5: Distances to nearest neighbours of each classed in T-SNE space (3 dimensions)
-Feature 6: Clustering features of original dataset
-Feature 7: Number of non-zeros elements in each row
-Feature 8: X (That feature was used only in **NN** 2nd level training)

The 2nd level we start training cross-validated just to choose best models, tune

hyperparameters and find optimum weights to average 3rd level.
After we found some good parameters, we trained 2nd level using entire trainset and bagged results.
The final model is a very stable 2nd level bagging of:
XGBOOST: 250 runs.
NN: 600 runs.
ADABOOST: 250 runs.

The average for the 3rd level we found better using a geometric mean of **XGBOOST** and **NN**. For **ET** we did an aritmetic mean with previous result: 0.85 * [**XGBOOST**^0.65 * **NN**^0.35] + 0.15 * [**ET**].

We tried a lot of training algorithms in first level as Vowpal Wabbit(many configurations), R glm, glmnet, scikit SVC, SVR, Ridge, SGD, etc... but none of these helped improving performance on second level.
Also we tried some preprocessing like PCA, ICA and FFT without improvement.
Also we tried Feature Selection without improvement. It seems that all features have positive prediction power.
Also we tried semi-supervised learning without relevant improvement and we discarded it due the fact that it have great potential to overfit our results.

Definetely the best algorithms to solve this problem are: Xgboost, NN and KNN. T-sne reduction also helped a lot. Other algorithm have a minor participation on performance. So we learn not to discard low performance algorithms, since it have enough predictive power to improve performance in a 2nd level training.Our final cross-validated solution scored around 0.3962. LB(Public): 0.38055 and LB(Private): 0.38243.

Gilberto & Stanislav =)