Exploring Classification ML Algorithms

By – pburra6

# Introduction

The purpose of this report is to apply five classification algorithms on two different dataset and summarize the pros and cons of the algorithm based on dataset types. This report is organised as follows:

1. Explain datasets properties and reason of choice.
2. Compare the algo performance for each dataset.
   1. KNN
   2. DecisionTree
   3. BoostedDecisionTree
   4. SVM
   5. Neural Network
3. Conclusion

# Datasets

Two datasets were picked from UC Irvine machine learning repo. The datasets were picked to be small in size (<50K) and number of features (<20) so the algorithm can train in reasonable time on a laptop. The datasets should also be on a field that does not require special expertise. They are diverse datasets that will help test algo behaviour for different feature and target variable types.

|  |  |  |
| --- | --- | --- |
|  | Dry Bean | Census Income |
| Classification type | Multi-class (7) | Binary |
| N features | 16 | 14 |
| N instances | 13,611 | 48,842 |
| Feature types | Numeric | Numeric, Categorical |
| Imbalance | No | Yes |
| Missing Data | No | Yes |

## EDA

EDA was performed on full dataset to detect any strong correlations in features and skewed distribution in features.

## Data prep

Dry Bean data features are numeric, so MinMax scaler has been applied to bring values to [0, 1] preserving the distribution of features.

Census Income had strongly skewed features and couple of features had dominant values. The missing value were dropped, and data filtered only for country US since this accounted for more than 95% of the instances. Filtering this did not impact the class distribution. MinMax was applied for numeric features and one hot encoding for categorical.

# Model

Standard set of process was setup as pipeline to ensure consistency across training for all algorithms. Gridsearch using 5 folds cross validation. 20% of data was set aside for test. GridSearch was performed on individual parameters to narrow down the best train/test score ranges on validation charts, followed by the combined grid search for all combinations of params in these potential ranges to get the best model. Learning plot was plotted based on the best model.

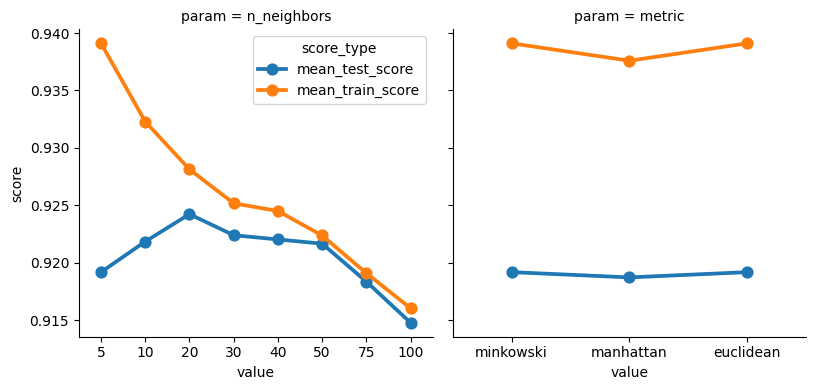
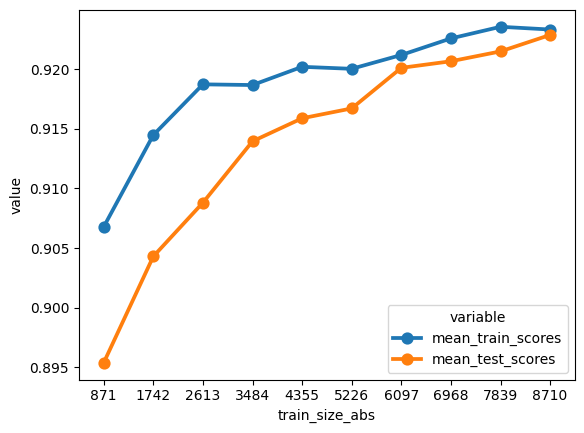
## Dry Beans Dataset

Metric for training – Accuracy. KNN was the easiest to train with SVM and BoostedTree the most time consuming. BoostedTree generalised the best.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Best Model Params | Train score | Test score |
| KNN | Metric: minkowski, p=2, n\_neighbors: 45 | 0.9229 | 0.9144 |
| DecisionTree | criterion: entropy, max\_depth: 10, min\_samples\_leaf: 10, min\_samples\_split: 2 | 0.9103 | 0.9034 |
| GradientBoostedTree | learning\_rate: 0.1, n\_estimators: 50 | 0.9192 | 0.9260 |
| SVM | C: 0.1,  gamma: 0.005, kernel: linear | 0.9105 | 0.9100 |
| Neural Network | activation: tanh, alpha: 0.001, hidden\_layer\_sizes: (50, 50), solver: adam | 0.9211 | 0.9133 |

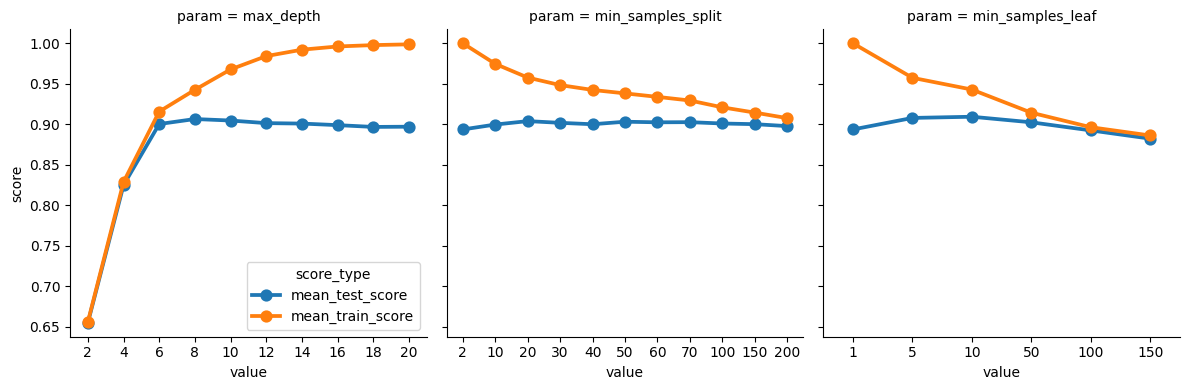
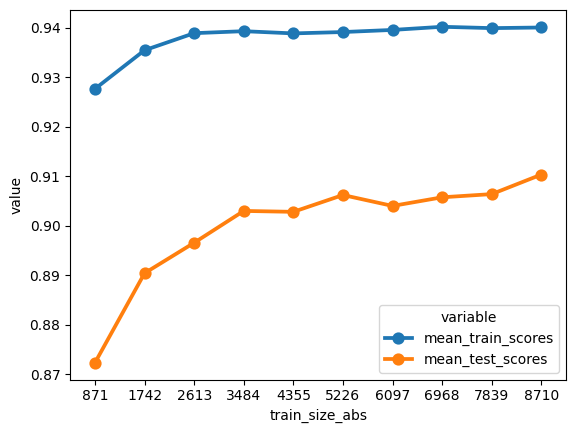
### KNN

The distance metric used did not seem to make a difference – especially with Minkowski being the generalised form of the Manhattan and Euclidean distance. The algo appeared to generalise well when using 50 neighbours, which is not a problem considering the sample size. Increasing neighbours further leads to a drop as we now breach the other clusters. Learning improves with sample size as the dataset is balanced.

**

### Decision Tree

For the decision tree the score did not improve significantly with training size perhaps owing to the numeric features. This also explains the pruning at max depth 6 in the valuation curve. Accuracy does not change much based on min samples split or min sample leaf.

**

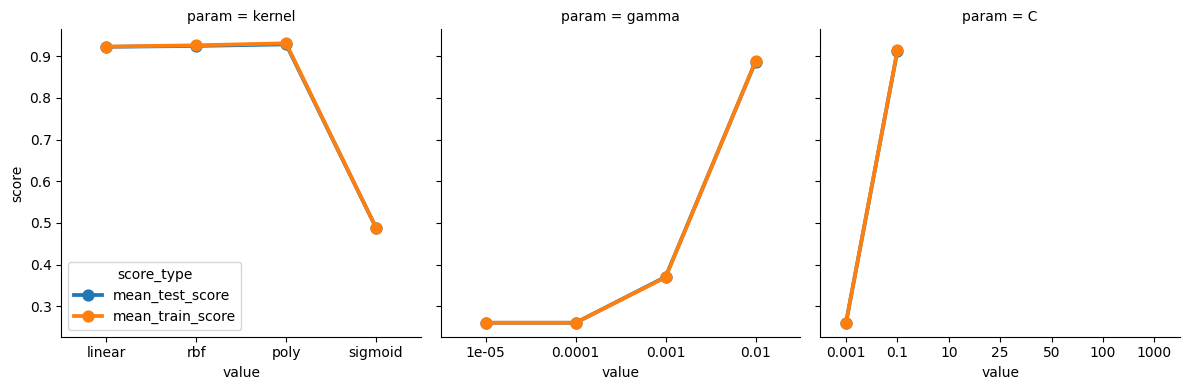
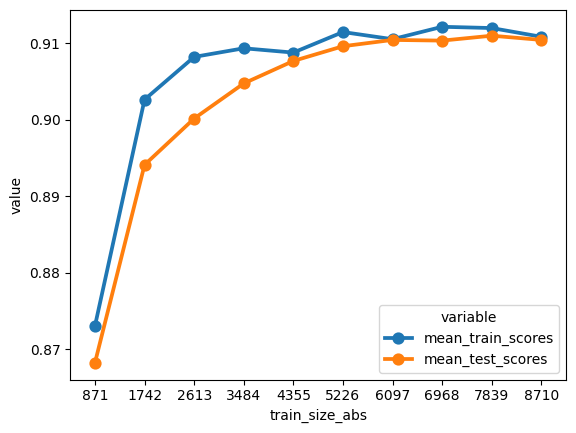
Gradient Boosted Tree

The model is expensive to train but performs better than DecisionTree as is expected for ensemble method. Using lower end of estimators 50 implied overfitting may not be a concern. The score does not improve with change in learners over a broad range which implies the best model could be found quickly. This is evident from DecisonTree that has accuracy of 0.91 from a single tree.

### 

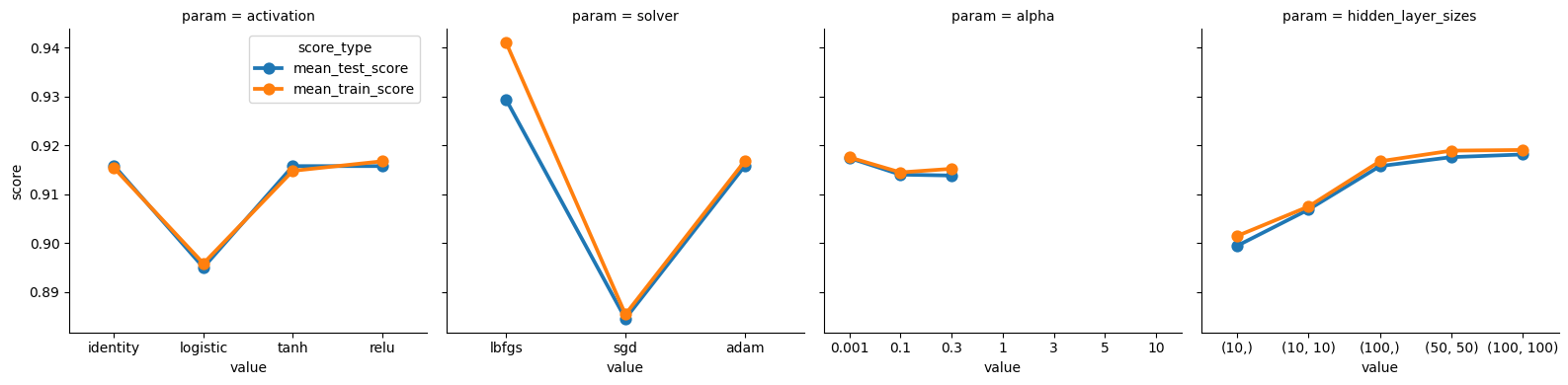
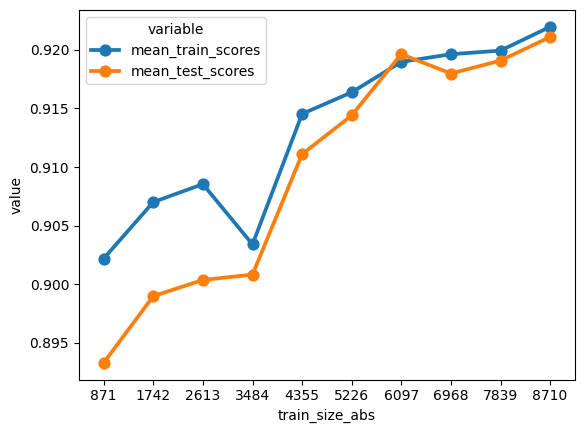
### SVM

SVM takes longer time for multiclass since it effectively breaks the problem into multiple binary SVM problems. It is surprising that the test and train align so closely in the learning and valuation curve. C=0.1 implies model will strongly choose the hyperplane to avoid misclassification. Linear kernel sufficed here implying no interaction between features. Since linear kernel has been doing well, gamma is inapplicable for our best model.



### NN

Logistic activation function performed poorly owing to the continuous nature of features. The neural network with more nodes performed better due the range values allowed in inputs and the multiclass output.



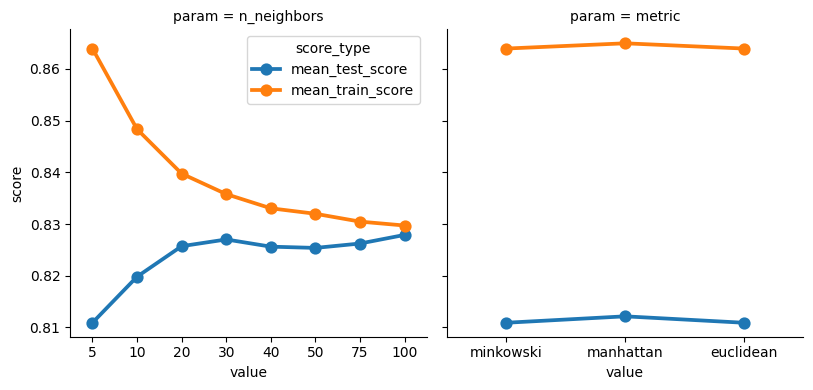
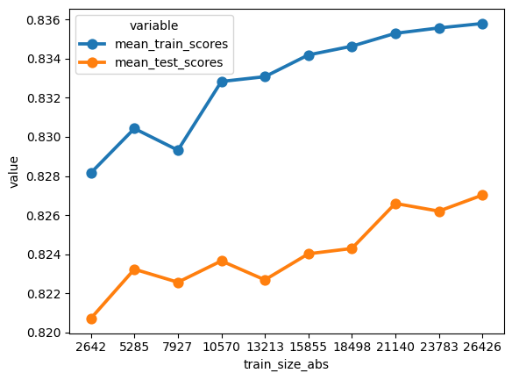
## Census Income

Accuracy metric is reasonable here since both the classes are equally important. F1 score could be used but we would not know what we are optimising for. KNN was the easiest to train with SVM and BoostedTree the most time consuming. BoostedTree generalised the best. SVM and DecisionTree are not optimal due to imbalanced dataset.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Best Model Params | Train score | Test score |
| KNN | metric: minkowski, p=2 n\_neighbors: 30 | 0.8270 | 0.8275 |
| DecisionTree | criterion: gini, max\_depth: 8, min\_samples\_leaf: 75, min\_samples\_split: 200 | 0.7984 | 0.8008 |
| GradientBoostedTree | learning\_rate: 0.1, n\_estimators: 150 | 0.8368 | 0.8344 |
| SVM | C: 0.2, gamma: 1e-05, kernel: 'poly' degree 3 | 0.7603 | 0.7681 |
| Neural Network | activation: 'relu', alpha: 0.1, hidden\_layer\_sizes: (100,), solver: 'adam' | 0.8318 | 0.8313 |

### KNN

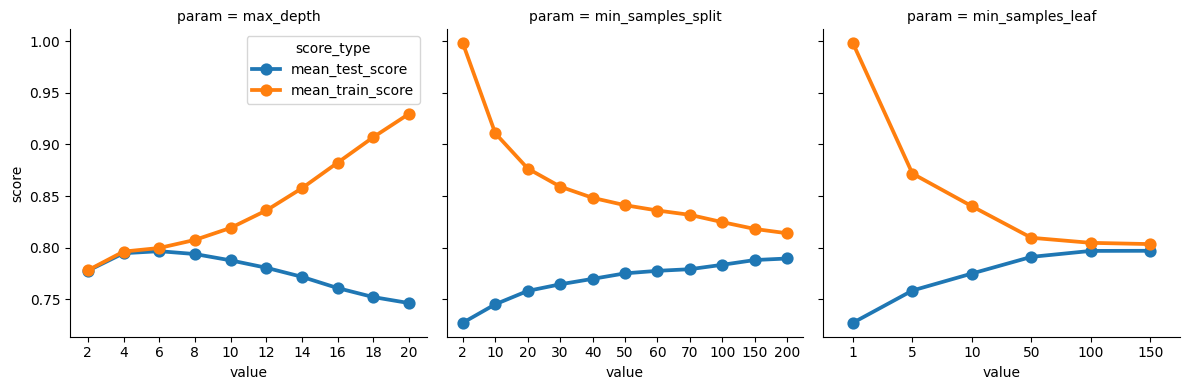
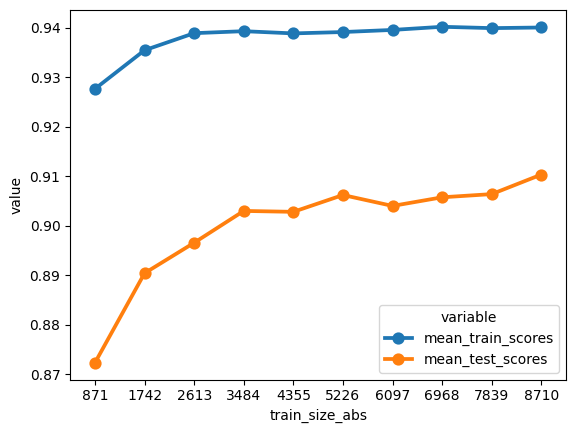
We observe the same behaviour as the previous dataset on the distance metric. 30 neighbours seem good to help classify. How ever test does not catch up to train scores with sample size due to the imbalanced dataset.



### DecisionTree

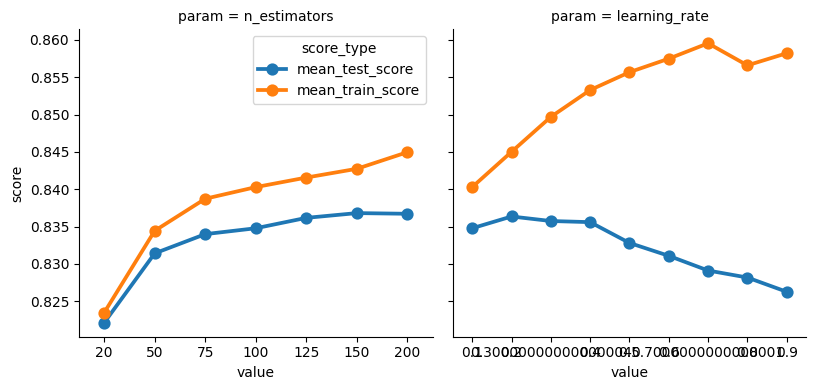
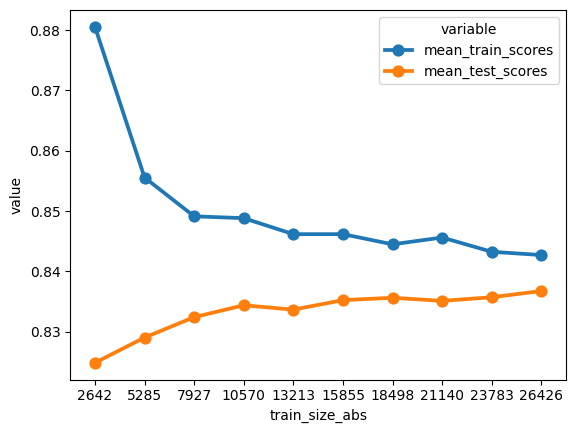
DT might not perform well for imbalanced data set. But the valuation charts show good patterns of generalistaion for min\_samples\_split and min\_sample\_leaf. The tree size is also small for the number of features, this could be due to the features being correlated, hence information can be captured in smaller tress. The learning curve shows that test struggles to catch up with training score despite large sample owing to the imbalanced dataset.

Perhaps using F1 metric might improve performance.



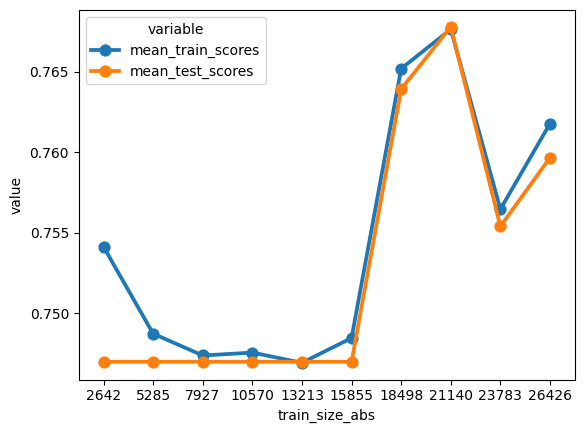
### Gradient Boosted Tree

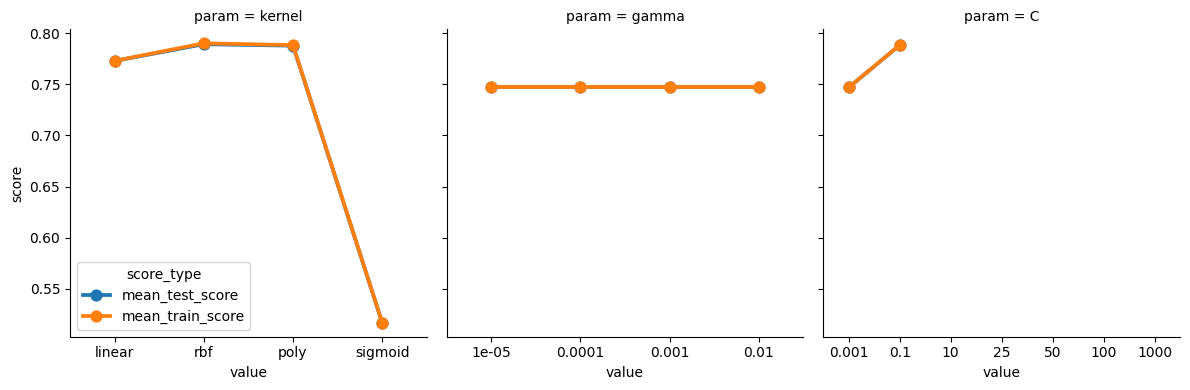
The algorithm can handle imbalanced data and the results shows. For this dataset this algo is an exception where the learning curve converges for test and train sets. The model was more time consuming than rest to train and at 150 learners, there may be chance that there is overfitting. The deviation of train/test scores for learning rate also exhibits overfitting for higher learning rates,



### SVM

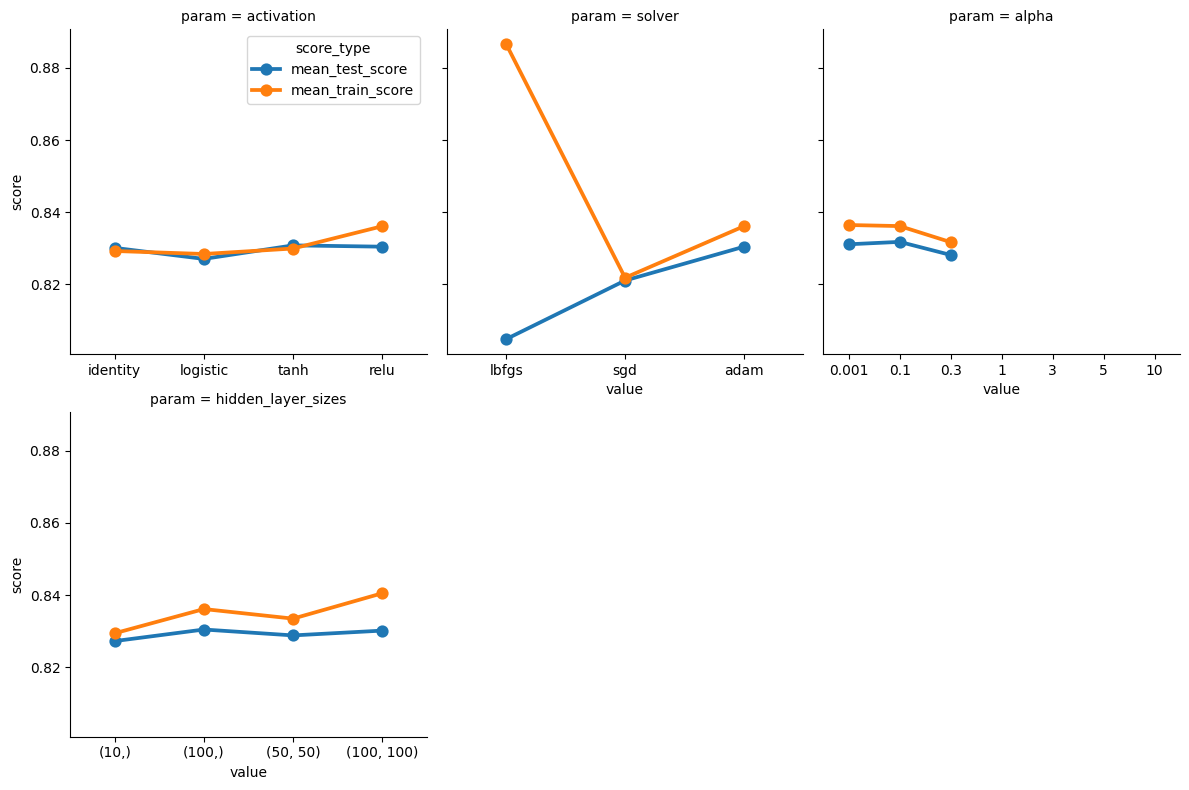
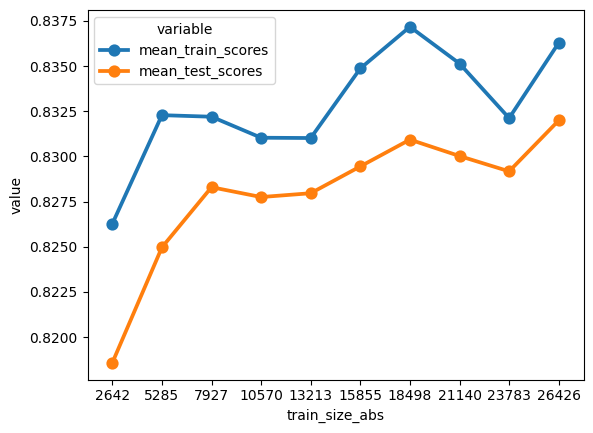
SVM is expected to perform well when there is binary classification but doesn’t not seem to be the case here due to imbalance dataset. Model also trains slower due to larger data size and features. Since the classification is binary one would expect sigmoid kernel to work well but does not seem to be the case. Poly and RBF kernel do better implying interaction between features (e.g Age, marital-status, relationship). For this case Poly kernel was chosen for speed.





### NN

Intrerpreting NN is more difficult, but a fully connected network of 10 nodes seemed to perform the same as 100 perhaps due to binary classification and categorical features. While LBGFGS otpimsier is expected to give better accuracy, it is slow to train and seems to have overfitted. It also required 5000 iterations to converge. Chose ADAM optimser for speed.



# References

The following websites were referred to while training the models

1. StatQuest - <https://www.youtube.com/c/joshstarmer>
2. <https://papers.nips.cc/paper_files/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html>
3. <https://www.freecodecamp.org/news/machine-learning-pipeline/>
4. <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html>
5. <https://stackoverflow.com/questions/37161563/how-to-graph-grid-scores-from-gridsearchcv>
6. <https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html>
7. <https://scikit-learn.org/stable/auto_examples/model_selection/plot_grid_search_digits.html>
8. <https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/>
9. <https://medium.com/cuenex/advanced-evaluation-metrics-for-imbalanced-classification-models-ee6f248c90ca>
10. <https://seaborn.pydata.org/tutorial/relational.html>
11. <https://www.analyticsvidhya.com/blog/2021/07/using-seaborns-facetgrid-based-methods-for-exploratory-data-analysis/>
12. <https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/>