Applied Model Ensembling

William Steimel

Table of Contents

Sources

- Dietterich, T. G. (2000). Ensemble Methods in Machine Learning. *Multiple Classifier Systems Lecture Notes in Computer Science*, 1-15. doi:10.1007/3-540-45014-9_1
- Hendrik Jacob van Veen, Le Nguyen The Dat, Armando Segnini. (2015). Kaggle Ensembling Guide. [accessed 2018 Feb 6]. https://mlwave.com/kaggle-ensembling-guide/
- Ben Gorman. (2016). A Kaggler's Guide to Model Stacking in Practice, http://blog.kaggle.com/2016/12/27/a-kagglers-guide-to-model-stacking-in-practice/

Ensemble Methods in Machine Learning

Introduction

- Typically in supervised learning:
- **Training Set -** A learning program is given training samples of the form $\{(x_1, y_1), \ldots, (x_m, y_m)\}$ for some unknown function $y = f(x)$
- **Feature Vectors -** typically are represented as $\{x_{1,} x_{2,} \ldots, x_{n,}\}$
- **Target Class -** The y class values are typically represented as $\{1, \ldots, K\}$
- Typically given a set of training samples a learning algorithm outputs a classifier which is a hypothesis about the true function f
  - **Classifiers or models** are denoted by $\{h_{1,} \ldots, h_L\}$

Introduction

- Ensemble Learning takes individual models and combines them to form a stronger learner.
- “One of the most active areas of research in supervised learning has been to study methods for constructing good ensembles of classifiers.”
- There are two conditions for an ensemble classifier to be stronger than its individual members
  - Accuracy - A good classifier is one that has an error rate lower than random guessing (.5)
  - Diversity – If the classifiers make different errors on different data points

Introduction

- Example one- Consider we have 3 classifiers $\{h_{1,} h_2, h_3\}$
- Three classifiers are identical –
  - When $h_1(x)$ is wrong $h_2(x)$, $h_3(x)$ will also be wrong
- Three classifiers are Uncorrelated (diverse)-
  - Classifier $h_1(x)$ is wrong but $h_2(x)$ and $h_3(x)$ may be correct
  - A Majority vote will correctly classify x
- The area under the curve for 11 classifiers being simultaneously wrong is .026 which is much less than the individual error rates of .3

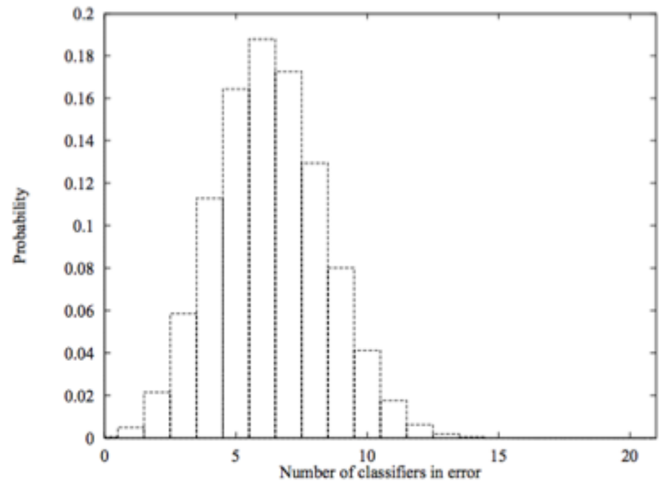


Fig. 1. The probability that exactly ℓ (of 21) hypotheses will make an error, assuming each hypothesis has an error rate of 0.3 and makes its errors independently of the other hypotheses.

Introduction

- There are three reasons it is possible to construct good ensembles
  - Statistical
  - Computational
  - Representational

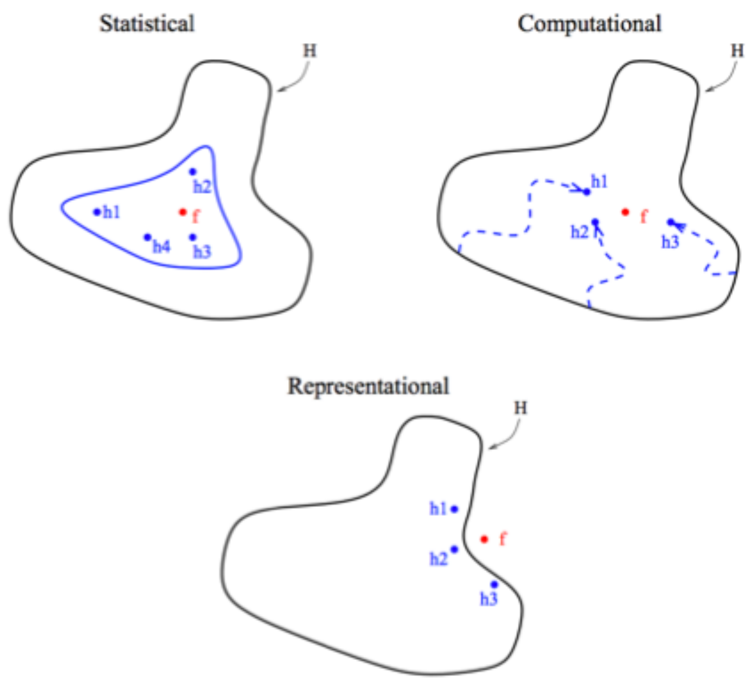
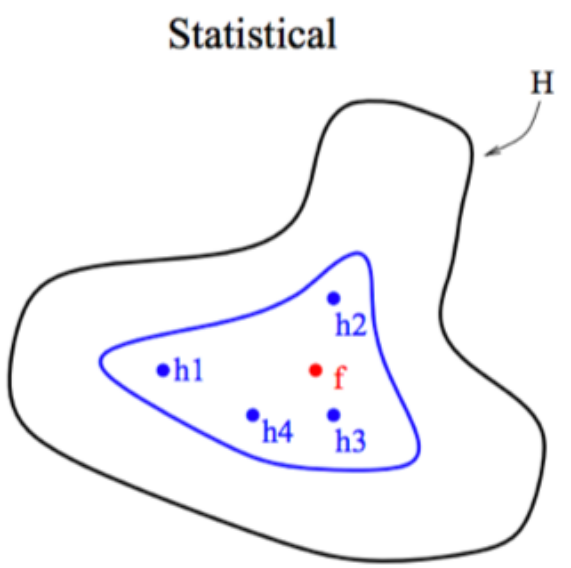


Fig. 2. Three fundamental reasons why an ensemble may work better than a single classifier

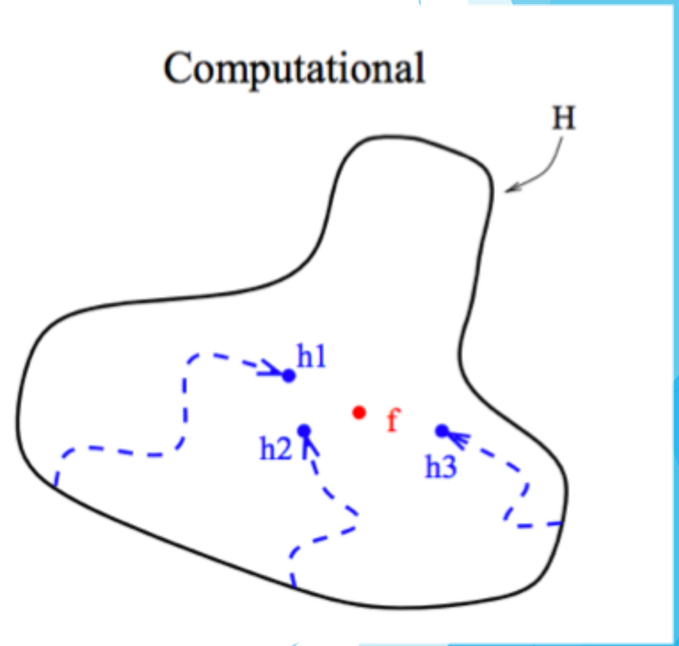
Statistical

- Statistical – A learning algorithm can be viewed as searching a space H for hypotheses to find the best hypothesis in space.
- There are some challenges where one may not have enough data leading to different hypotheses in H that all give the same accuracy on training data.
- Averaging the accurate hypotheses we can find a good approximation of f



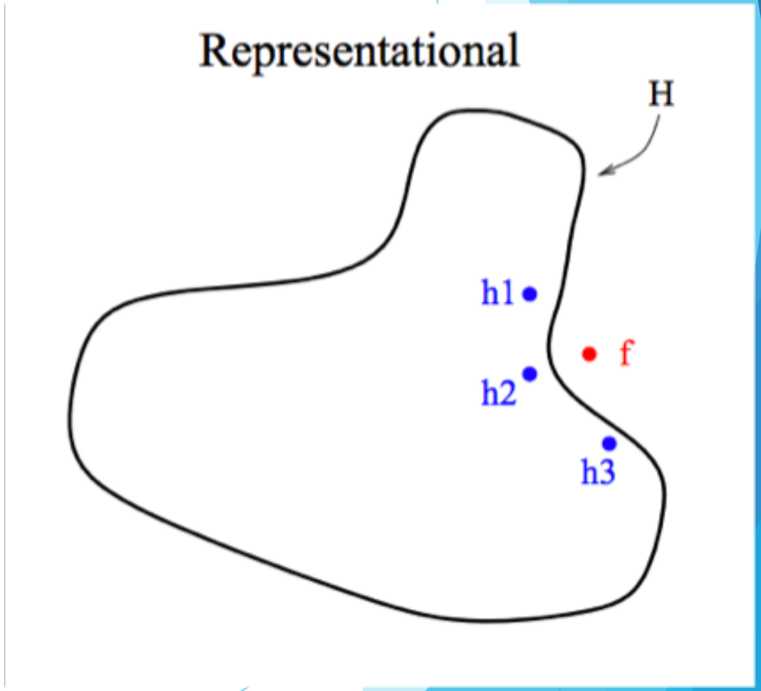
Computational

- May algorithms work by performing local search that can get stuck in local optima.
  - Example: Optimal training of Neural Networks and decision trees are both considered NP-hard
- It is sometimes computationally difficult for a learning algorithm to find the best hypothesis
- An ensemble constructed from running local search from many different starting points for an unknown function may be better than individual classifiers.



Representational

- In most applications of machine learning, the true function f cannot be represented by any of the hypotheses in H
- Weighted sums of the hypothesis may expand the space of representable functions.



Kaggle Ensembling Guide

Introduction/Motivation

- This website gathers all the major ensembling techniques used on Kaggle as well as their actual results during past competitions
- This will be a useful exercise in understanding ensemble learning techniques for future competitions and during my thesis.

Creating Ensembles from Submission Files

- On Kaggle many competitions require submission files which usually contain the ID’s and predictions of the test set after performing modeling
- This section details some methods for ensembling or combining these model predictions
    - Voting Ensembles
    - Correlation
    - Weighted Majority Vote
    - Averaging
    - Rank Averaging

Voting Ensembles

- Simple Majority Vote Ensemble
  - Class with the most votes wins
  - Perhaps the simplest form of Ensembling
- For Simplicity all samples in this section will use a Test Set of 10 samples:
  - 1111111111 (Ground truth or 100 % Accuracy)

Initial Models

1111110100 = 70 % accuracy
1111011100 = 70 % accuracy
1011111100 = 70 % accuracy

Result after Majority Vote

1111111100 = 80 % accuracy

Correlation

- One Kaggler calculated the Pearson Correlation for all model submission files and gathered the few well-performing models that were less correlated
  - This lead to a 50 spot increase on the leaderboard for the forest cover type prediction competition
- Uncorrelated submissions do better when ensembled

Stronger Correlated Models	Weaker Un-Correlated Models
1111111100 = 80 % accuracy	1111111100 = 80 % accuracy
1111111100 = 80 % accuracy	0111011101 = 70 % accuracy
1011111100 = 70 % accuracy	1000101111 = 60 % accuracy
Result after Majority Vote	**Result after Majority Vote**
1111111100 = 80 % accuracy	1111111101 = 90 % accuracy

Weighted Majority Vote

- Weighting the voting ensemble by better model or models gives the stronger models more pull in a vote.
  - Example: You have 5 models, the best model gets 3 votes while the other 4 models get 1 vote each
- The only way for weaker models to overrule the best model in this case is for them to all agree that the best model is wrong
- This can help fix some mistakes that the dominant model makes

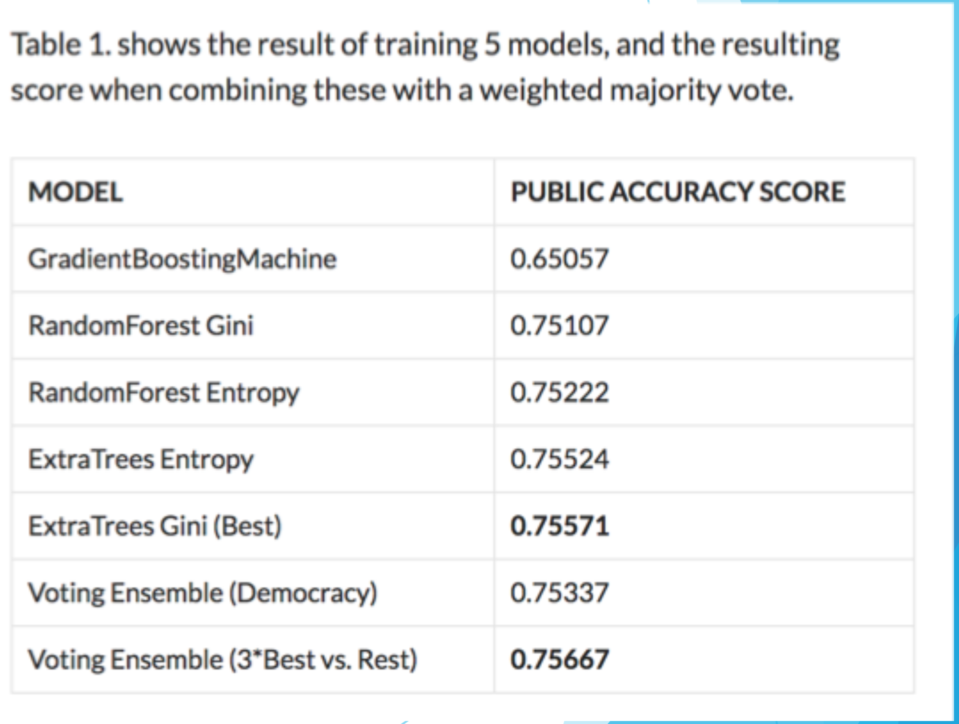


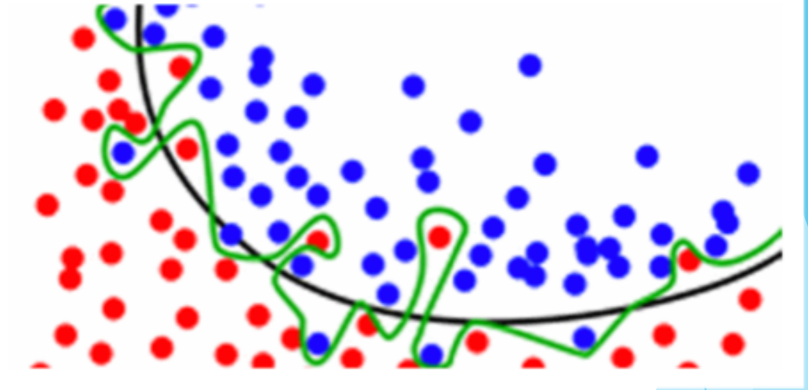
Table 1. shows the result of training 5 models, and the resulting score when combining these with a weighted majority vote.

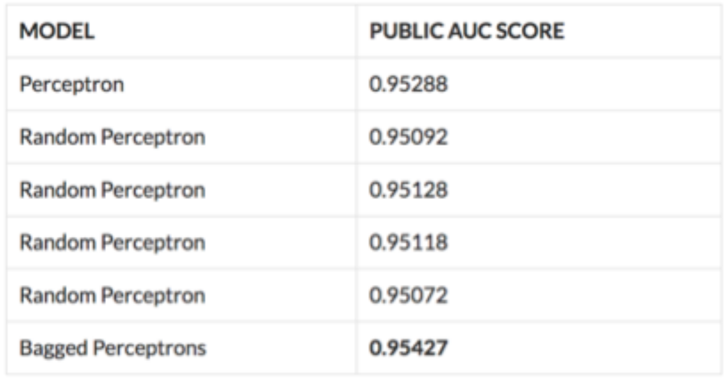
MODEL	PUBLIC ACCURACY SCORE
GradientBoostingMachine	0.65057
RandomForest Gini	0.75107
RandomForest Entropy	0.75222
ExtraTrees Entropy	0.75524
ExtraTrees Gini (Best)	**0.75571**
Voting Ensemble (Democracy)	0.75337
Voting Ensemble (3*Best vs. Rest)	**0.75667**

Averaging

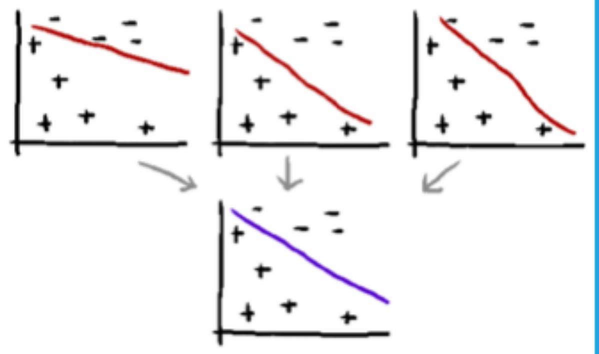
- Averaging (Taking the mean of individual predictions in Regression Problems)
  - Averaging predictions of each model often reduces overfitting
  - This is also known as bagging submissions
- The goal of most machine learning problems is to generalize to unseen data and averaging helps smooth out the decision boundary.

Don't overfit



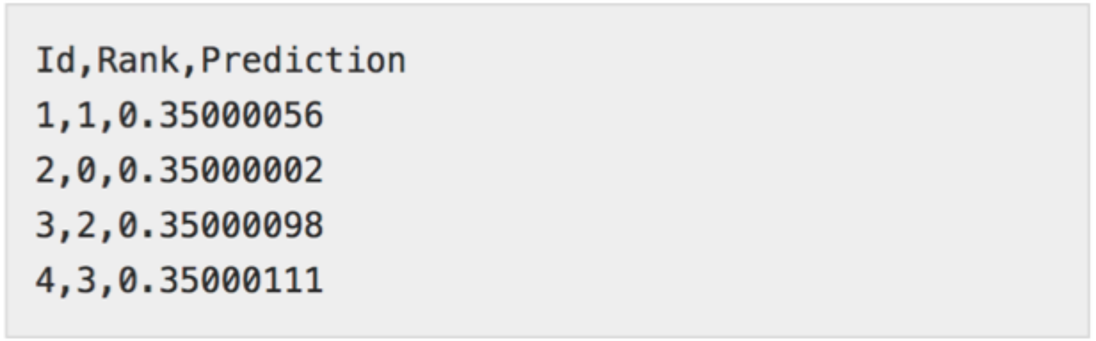


MODEL	PUBLIC AUC SCORE
Perceptron	0.95288
Random Perceptron	0.95092
Random Perceptron	0.95128
Random Perceptron	0.95118
Random Perceptron	0.95072
Bagged Perceptrons	**0.95427**

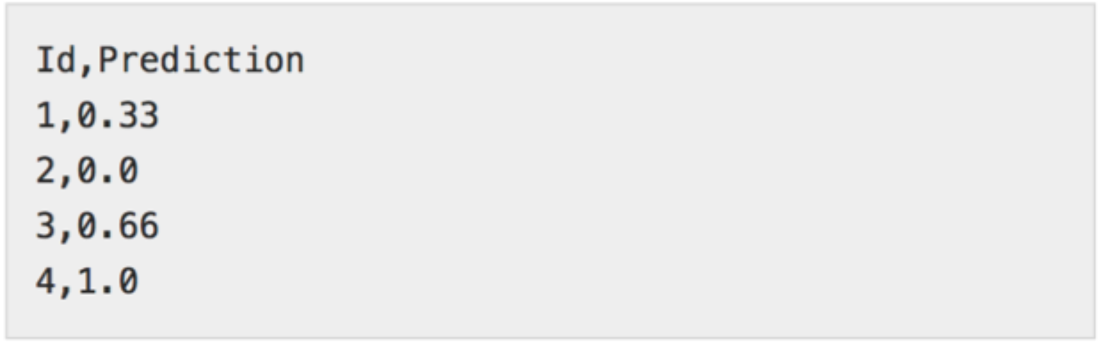


Rank Averaging

- When averaging outputs from multiple models, sometimes not all predictors are perfectly calibrated (on the same scale)
  - One solution is to turn the predictions into ranks and then normalize the averaged ranks between 0 and 1 to get an even distribution in predictions.
- Rank Averaging does well on threshold metrics like AUC and search-engine quality metrics
- Steps:
  - Rank all values in each model
  - Average the ranks between models
  - Normalize ranks between 0 and 1



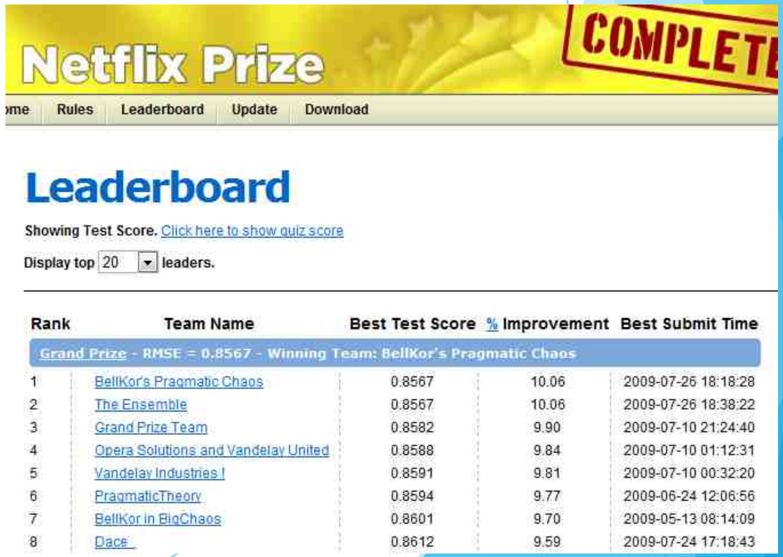
```
Id,Rank,Prediction
1,1,0.35000056
2,0,0.35000002
3,2,0.35000098
4,3,0.35000111
```



```
Id,Prediction
1,0.33
2,0.0
3,0.66
4,1.0
```

Stacked Generalization & Blending

- Stacking Blending have lead to a lot of the large performance improvements in modeling as of recently.
- The Netflix data science competition helped contribute to what we know as modern ensemble creation.
  - Stacked Generalization
  - Blending
  - Stacking with Logistic Regression
  - Stacking with non-linear algorithms
  - Feature Weighted linear stacking
  - Stacking classifiers with regressors and vice versa
  - Stacking unsupervised learned features



Leaderboard

Showing Test Score. Click here to show quiz score

Display top 20 leaders.

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace	0.8612	9.59	2009-07-24 17:18:43

Stacked Generalization

- Also referred to as stacking, Introduced by Wolpert in 1992 paper “Stacked Generalization”
- A Stacked Model trains base classifiers then uses another classifier to combine their predictions to reduce generalization error
  - First stage predictions are used as features for the stacker model (meta features)
- *“stacked generalization is a means of non-linearly combining generalizers to make a new generalizer, to try to optimally integrate what each of the original generalizers has to say about the learning set. The more each generalizer has to say (which isn’t duplicated in what the other generalizer’s have to say), the better the resultant stacked generalization. ” Wolpert (1992) Stacked Generalization*

Blending

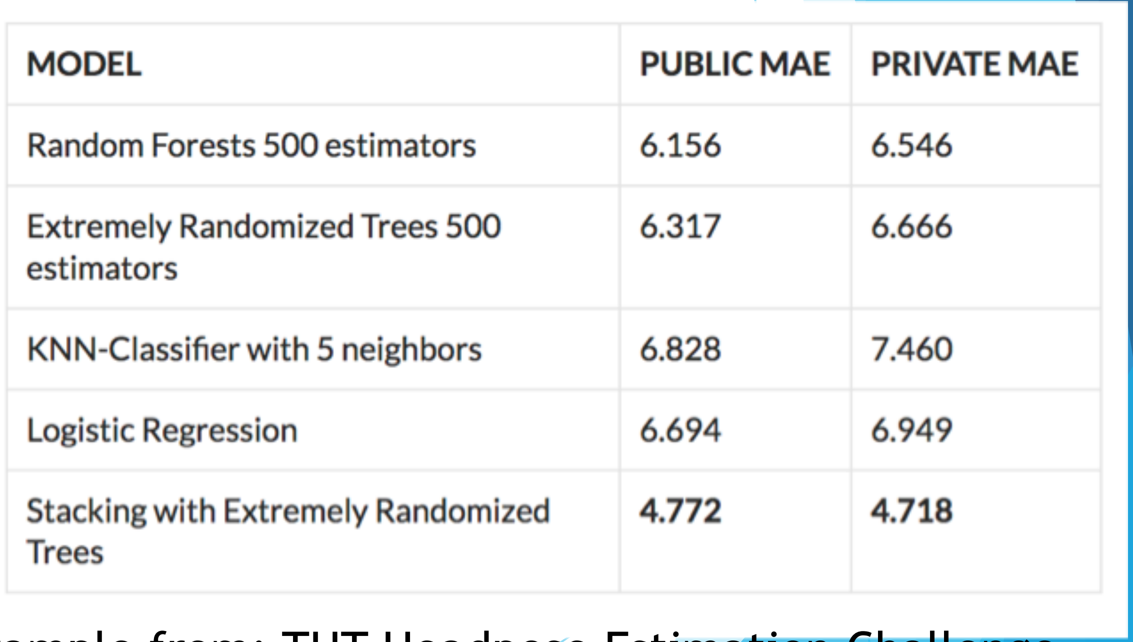
- Blending is similar to stacked generalization and is a term sometimes used interchangeably with stacking and blending
- The difference is the train set is typically divided into a training set and validation set (10 % of the data)
  - The Stacker Model trains on only the holdout/validation set.

Stacking with Logistic Regression

- One of the most traditional ways of stacking
- Use the base models predictions as features (meta-features) for a Logistic Regression Model

Stacking with non-linear algorithms

- Popular non-linear algorithms for stacking:
  - GBM (Gradient Boosting Machines)
  - KNN (K-Nearest Neighbors)
  - NN (Neural Networks)
  - RF (Random Forest)
  - ET (Extremely randomized trees)
- Stacking with non-linear algorithms gives good performance improvements and finds useful interactions between the original features as well as meta model features

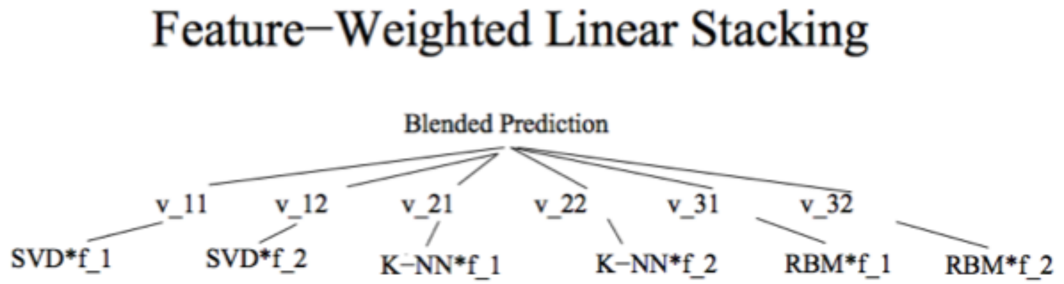


MODEL	PUBLIC MAE	PRIVATE MAE
Random Forests 500 estimators	6.156	6.546
Extremely Randomized Trees 500 estimators	6.317	6.666
KNN-Classifier with 5 neighbors	6.828	7.460
Logistic Regression	6.694	6.949
Stacking with Extremely Randomized Trees	**4.772**	**4.718**

Example from: TUT Headpose Estimation Challenge

Feature-Weighted Linear Stacking

- Feature-Weighted Linear Stacking is a more complicated version of stacking
  - Helped win the Netflix Recommender System Competition
- FWLS Weighs the models but also the meta features.
- “FLWS combines model predictions linearly using coefficients that are themselves linear functions of meta-features.”
- Goal – Learn which base model’s meta features is the best predictor for samples with a certain feature value

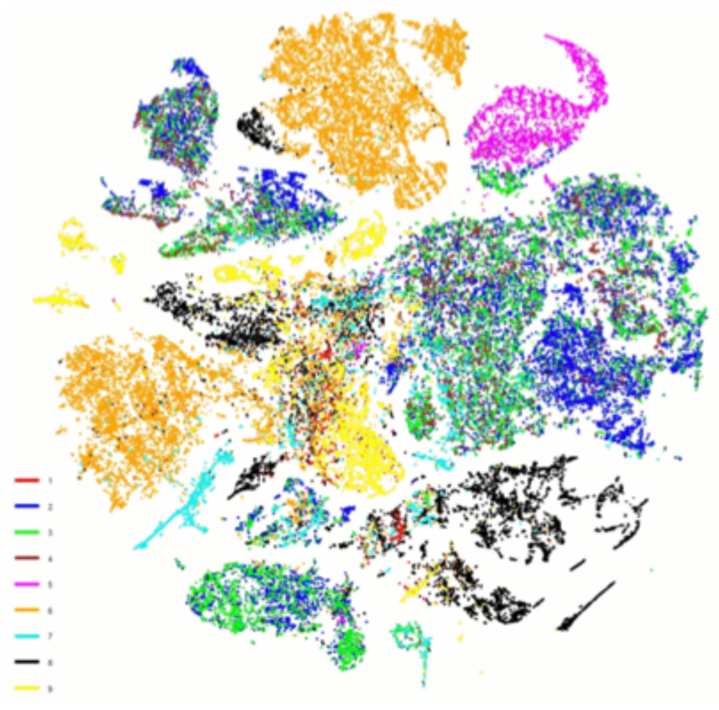


Stacking Classifiers with regressors and vice versa

- With stacking we can use classifiers for regression problems and regressors for classification problems.
- Example: Classifier for regression problem
  - We would like to predict cost of a flight based on descriptive features.
- Use bins to group y-label (target) into evenly spaced classes.
  - Everything Under 10,000 Yen is class 1
  - Everything between 10,000 yen and 50,000 yen is class 2
  - Everything over 50,000 yen is Class 3
- Use classification algorithm to return predicted probabilities for each class:
  - Results can be used as meta-features in stacking to help a regressor make better predictions.

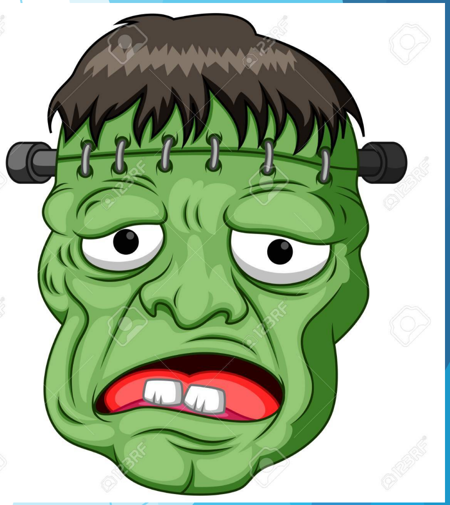
Stacking Unsupervised Learned Features

- You can also stack with unsupervised learning techniques
- Methods like K-means and T-SNE are popular for this
- Reduce the dataset to 2 or 3 dimensions and stack with a non-linear stacker
  - Use t-SNE output vectors and boost them with another model



Conclusion

- Why create these “Frankenstein Ensembles?”
  - You can win Kaggle competitions with ensembling
  - 1 % increase in accuracy can have a big impact on real world problems (1 % increase in profit is large when talking about big companies)
  - Can beat a lot of state-of-the-art academic benchmarks



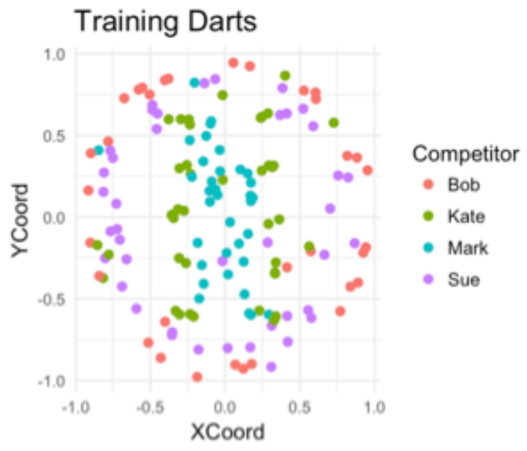
A Step by Step Guide to Model Stacking

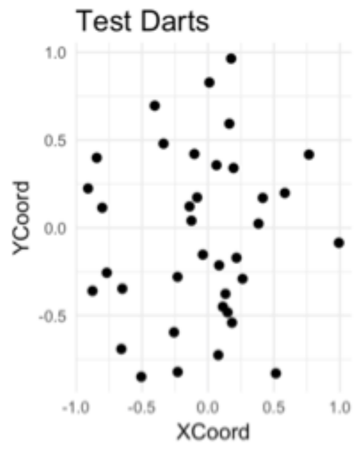
Introduction

- Stacking- “(Also called meta ensembling) is a model ensembling technique used to combine information from multiple predictive models to generate a new model.“
- Stacking is one of the techniques considered very important for competitive machine Learning and very powerful compared to individual models by themselves
- Stacking is most effective when base models are significantly different
- This post by Kaggle is a Tutorial on Model stacking for beginners by Kaggle Master Ben Gorman.

Motivation

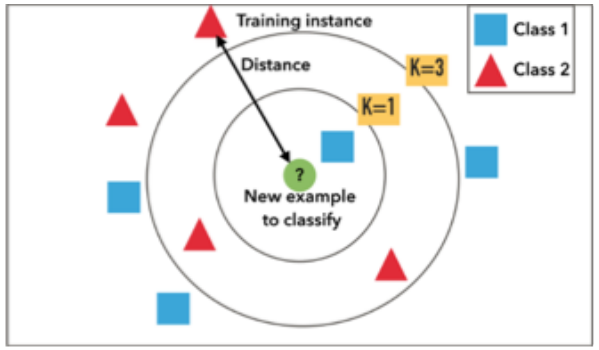
- Problem – Four people throw 187 darts at a board.
  - 150 darts – We get to see where they each persons landed (Training set)
  - 37 darts – Can only see where the dart landed (Test Set)
  - Objective – Predict who threw each dart based on where they landed





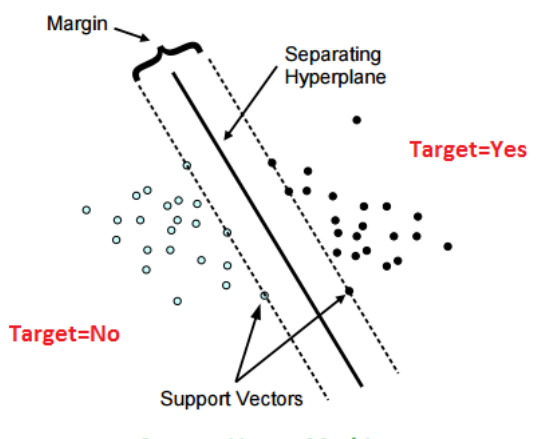
K-Nearest Neighbors (Base Model1)

- Lets try solving the darts problem with K-Nearest Neighbors first
- Hyperparameter Tuning - To select the best value of K, 5-fold Cross Validation combined with grid search is used with a grid space of K = (1,2, ... 30)
  - CV result - K = 1 had the best performance with 67 % accuracy.
- After the model is trained, predictions on the test set achieved around a 70 % classification accuracy.
  - Not too bad.



Support Vector Machine (Base Model 2)

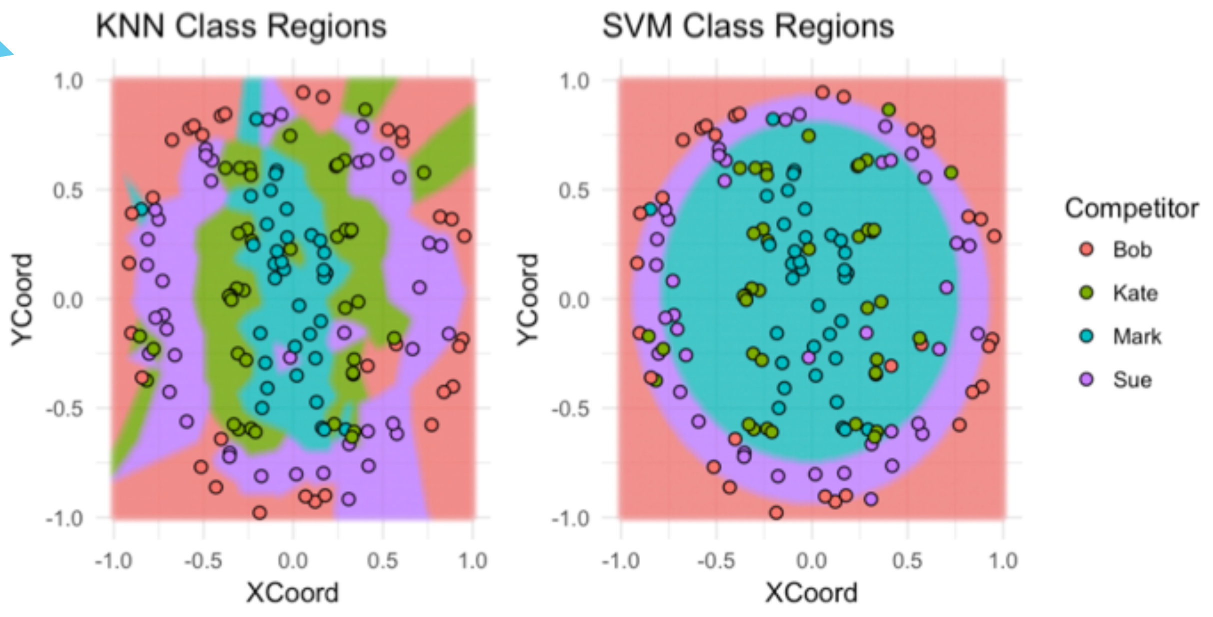
- Lets see if Support Vector Machine can perform any better.
- One feature titled 'DistFromCenter' was added to make the data more linearly separable for SVM.
  - Measures distance each point lies from the center or bullseye
- Two hyperparameters to be tuned:
  - Type
    - L2-regularized L2-loss support vector classification (dual)
    - L2-regularized L2-loss support vector classification (primal)
    - L2-regularized L1-loss support vector classification (dual)
    - support vector classification by Crammer and Singer
    - L1-regularized L2-loss support vector classification
  - Cost
    - (.01, .1, 1, 10, 100, 1000, 2000)
- The best hyperparameters determined by CV were type = 4 and cost = 1000
  - 61 % CV Classification Accuracy
  - 78 % Classification Accuracy on Test Set



Support Vector Machine

Stacking (Meta Ensembling)

- The classifications of Bob, Kate, Mark, and Sue's throws in the previous KNN and SVM models.
  - SVM does well at separating Bob's and Sue's throws but poor at separating Kates and Mark's Throws.
  - The opposite seems to be true for the KNN model

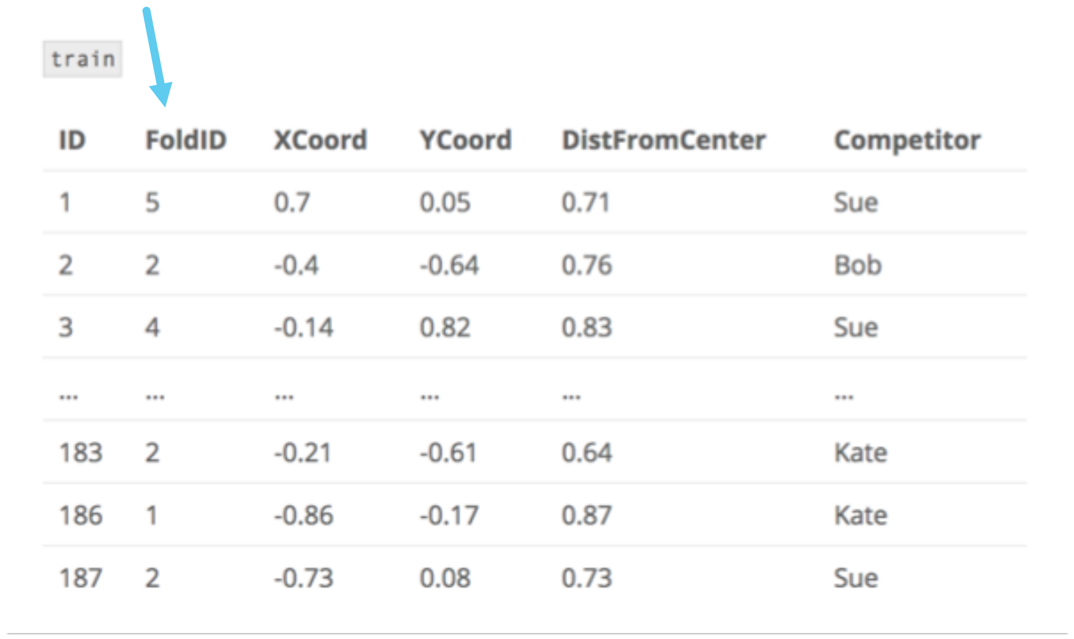


70 % Accuracy

78 % Accuracy

How to Implement Stacking?

- Step 1. Partition the training data into five test folds
  - Notice the new FoldID field

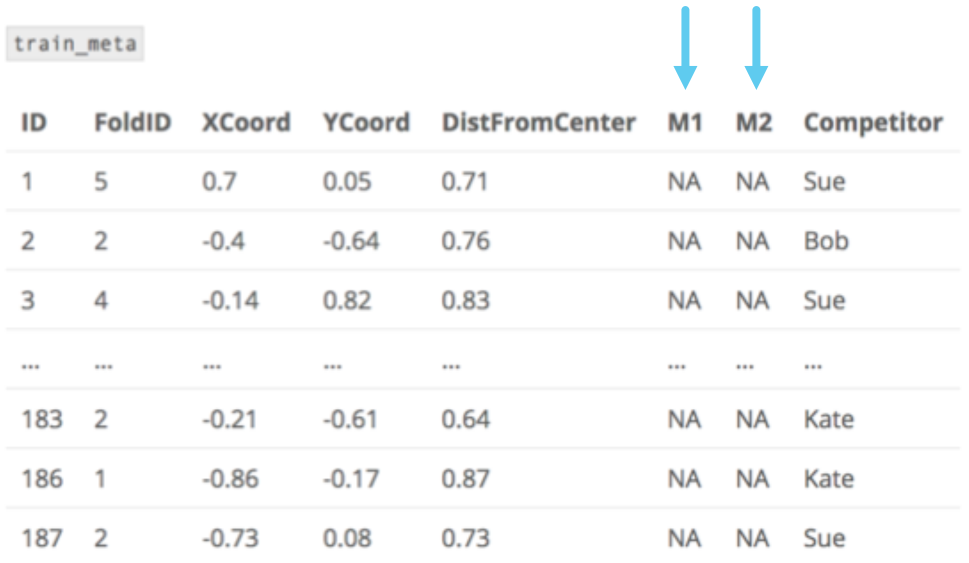


train

ID	FoldID	XCoord	YCoord	DistFromCenter	Competitor
1	5	0.7	0.05	0.71	Sue
2	2	-0.4	-0.64	0.76	Bob
3	4	-0.14	0.82	0.83	Sue
...	...	...	...	...	...
183	2	-0.21	-0.61	0.64	Kate
186	1	-0.86	-0.17	0.87	Kate
187	2	-0.73	0.08	0.73	Sue

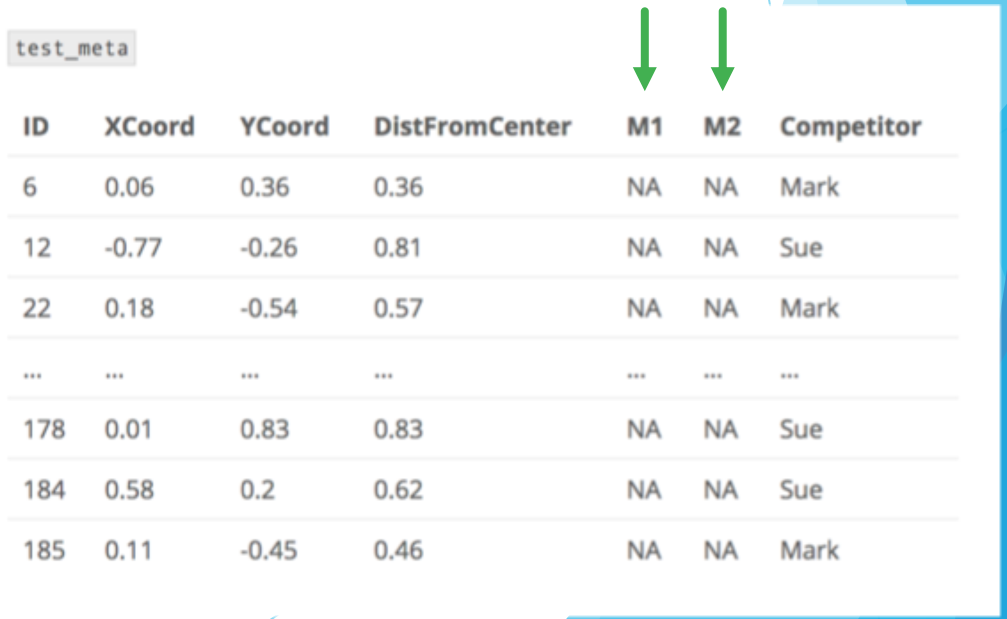
How to Implement Stacking?

- Step 2. Create a dataset titled train_meta with the same row id's and fold ids as the training dataset, with empty columns M1 and M2
  - Also do the same for test_meta



train_meta

ID	FoldID	XCoord	YCoord	DistFromCenter	M1	M2	Competitor
1	5	0.7	0.05	0.71	NA	NA	Sue
2	2	-0.4	-0.64	0.76	NA	NA	Bob
3	4	-0.14	0.82	0.83	NA	NA	Sue
...	...	...	...	...	...	...	...
183	2	-0.21	-0.61	0.64	NA	NA	Kate
186	1	-0.86	-0.17	0.87	NA	NA	Kate
187	2	-0.73	0.08	0.73	NA	NA	Sue

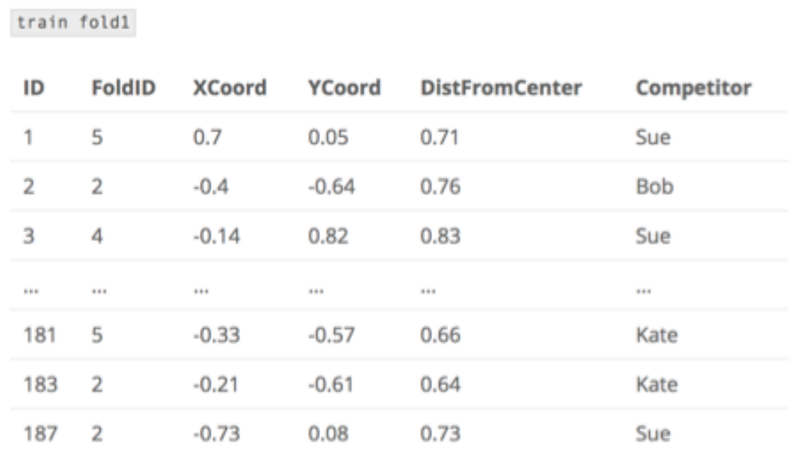


test_meta

ID	XCoord	YCoord	DistFromCenter	M1	M2	Competitor
6	0.06	0.36	0.36	NA	NA	Mark
12	-0.77	-0.26	0.81	NA	NA	Sue
22	0.18	-0.54	0.57	NA	NA	Mark
...	...	...	...	...	...	...
178	0.01	0.83	0.83	NA	NA	Sue
184	0.58	0.2	0.62	NA	NA	Sue
185	0.11	-0.45	0.46	NA	NA	Mark

How to Implement Stacking?

- Step 3.1 – For Each Test Fold {1,2,3,4,5} combine the other four folds to be used as a training fold
  - (Fold 2,3,4,5 in this case)

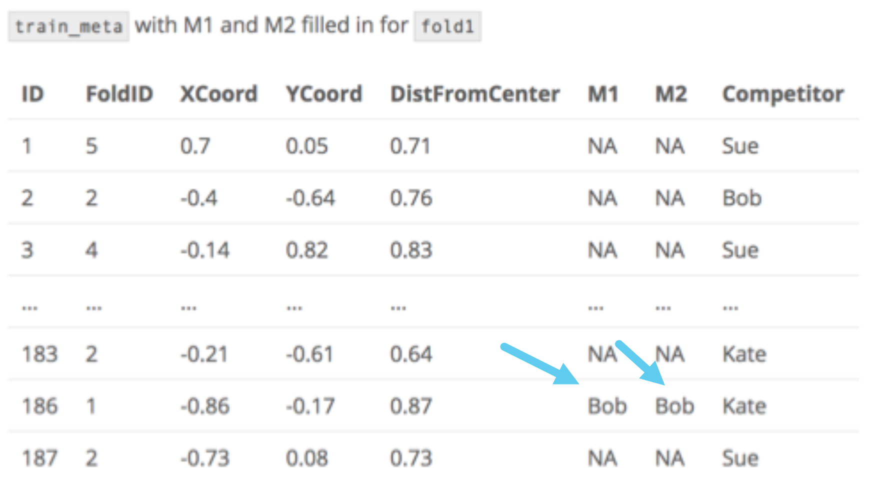


`train_fold1`

ID	FoldID	XCoord	YCoord	DistFromCenter	Competitor
1	5	0.7	0.05	0.71	Sue
2	2	-0.4	-0.64	0.76	Bob
3	4	-0.14	0.82	0.83	Sue
...	...	...	...	...	...
181	5	-0.33	-0.57	0.66	Kate
183	2	-0.21	-0.61	0.64	Kate
187	2	-0.73	0.08	0.73	Sue

- Step 3.2 – For each base model fit the base model to the training fold and make predictions on the test fold.
  - **K-NN - M1**
    - k=1
  - **SVM - M2**
    - type=1, cost = 1000

- Step 3.3 - Store these predictions in train_meta to be used as features for stacking

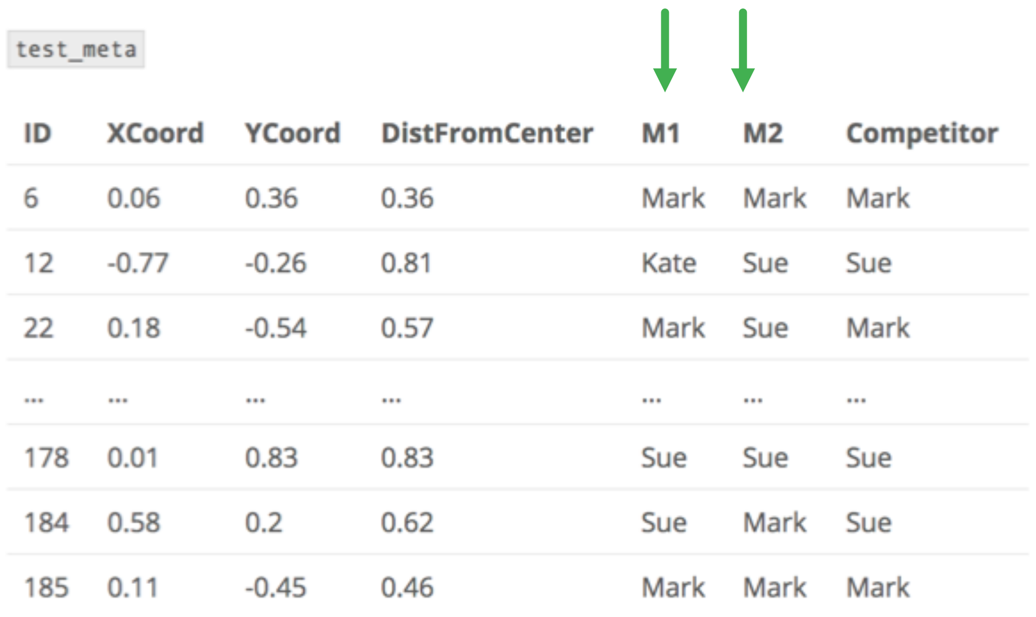


`train_meta` with M1 and M2 filled in for `fold1`

ID	FoldID	XCoord	YCoord	DistFromCenter	M1	M2	Competitor
1	5	0.7	0.05	0.71	NA	NA	Sue
2	2	-0.4	-0.64	0.76	NA	NA	Bob
3	4	-0.14	0.82	0.83	NA	NA	Sue
...	...	...	...	...	...	...	...
183	2	-0.21	-0.61	0.64	NA	NA	Kate
186	1	-0.86	-0.17	0.87	Bob	Bob	Kate
187	2	-0.73	0.08	0.73	NA	NA	Sue

How to Implement Stacking

- Step 4- Fit each base model to the full training set and make predictions on the test set. Store these in the test set (test_meta)

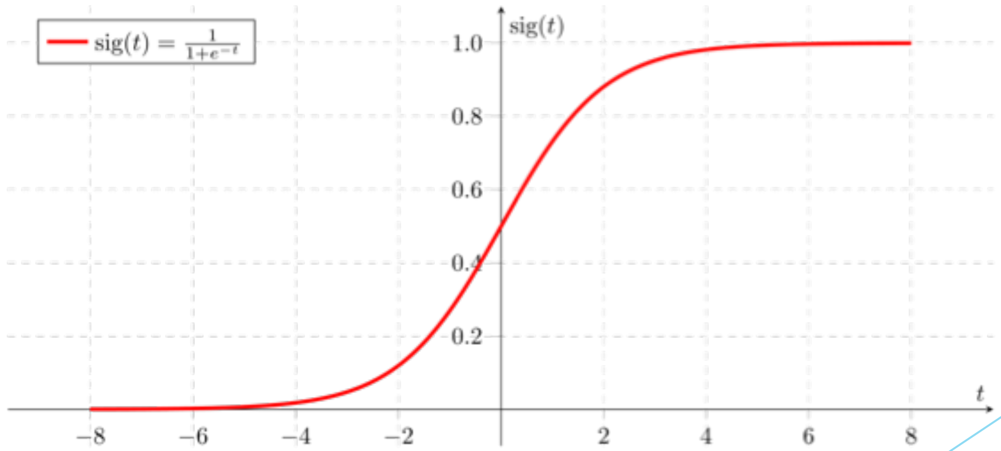


test_meta

ID	XCoord	YCoord	DistFromCenter	M1	M2	Competitor
6	0.06	0.36	0.36	Mark	Mark	Mark
12	-0.77	-0.26	0.81	Kate	Sue	Sue
22	0.18	-0.54	0.57	Mark	Sue	Mark
...	...	...	...	...	...	...
178	0.01	0.83	0.83	Sue	Sue	Sue
184	0.58	0.2	0.62	Sue	Mark	Sue
185	0.11	-0.45	0.46	Mark	Mark	Mark

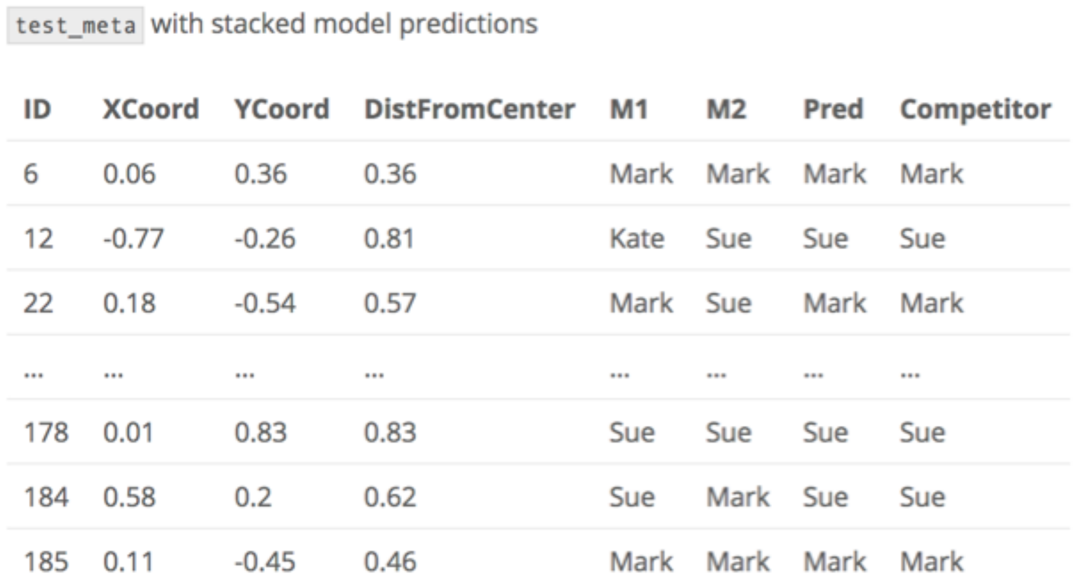
Stacked Model Hyperparameter Tuning

- Step 5- Fit a new model (S – the stacking model) to train_meta using M1 and M2 as features.
  - In this case Logistic Regression
- Optional – Include any newly engineered features



Stacked Model Hyperparameter Tuning

- Step 6: Use the stacked model (S) to make predictions on test_meta


`test_meta` with stacked model predictions

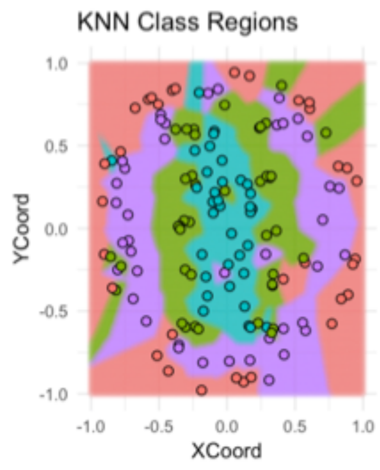
ID	XCoord	YCoord	DistFromCenter	M1	M2	Pred	Competitor
6	0.06	0.36	0.36	Mark	Mark	Mark	Mark
12	-0.77	-0.26	0.81	Kate	Sue	Sue	Sue
22	0.18	-0.54	0.57	Mark	Sue	Mark	Mark
...	...	...	...	...	...	...	...
178	0.01	0.83	0.83	Sue	Sue	Sue	Sue
184	0.58	0.2	0.62	Sue	Mark	Sue	Sue
185	0.11	-0.45	0.46	Mark	Mark	Mark	Mark

Key Points

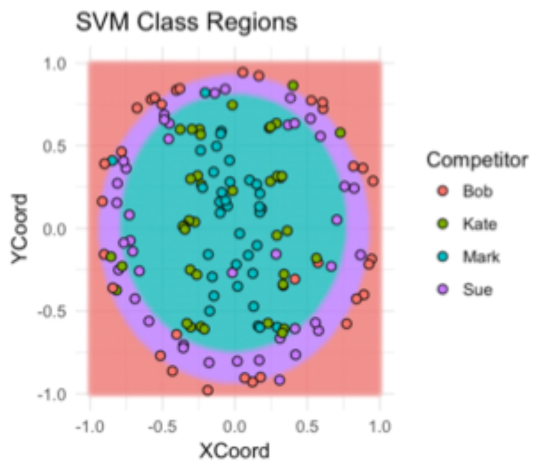
- Predictions of the base models are being used as features (meta features) for the stacked model
- This allows the stacked model to know where each base model performs well or poorly.

Stacked Logistic Regression

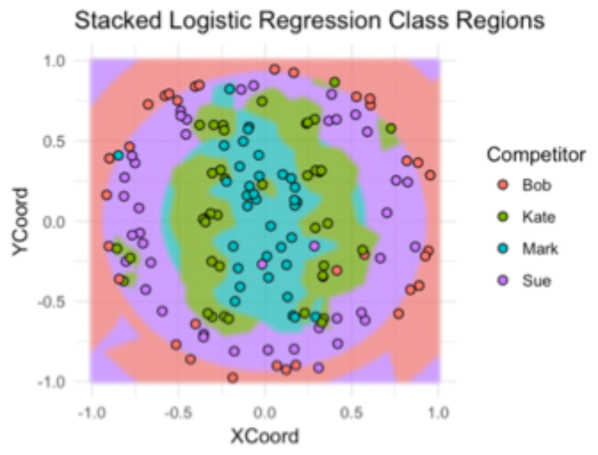
- The Logistic Regression Stacked Model retains the best aspects of each base model which is why it performs better than either base model by itself.
  - 75 % CV Accuracy (Training Set)
  - 86 % Test Accuracy



70 % Accuracy

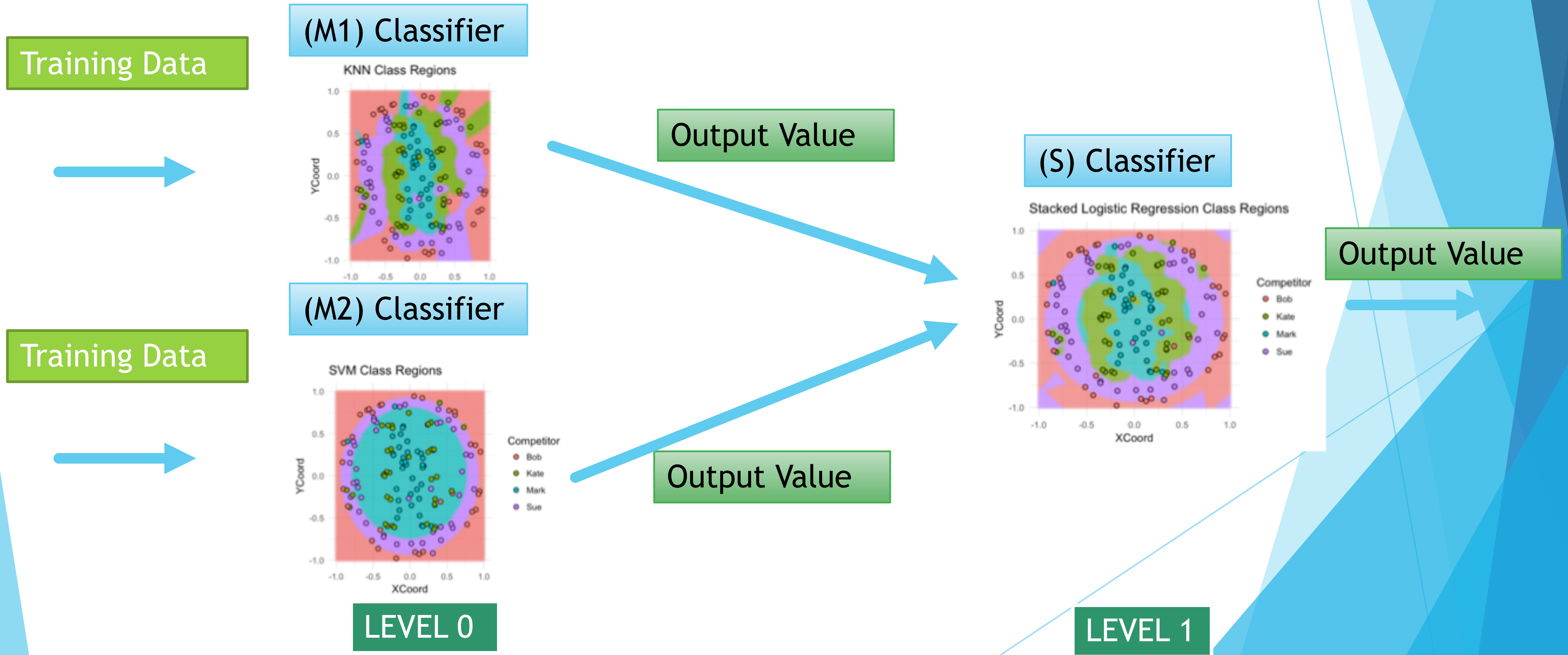


78 % Accuracy



86 % Accuracy

Stacking Diagram



Conclusion

- Stacking is mainly used on Machine Learning competitions on Kaggle
- It achieves very small gains with a lot of added complexity which make it unsuitable for most businesses.
  - However, small gains in competitive machine learning can mean the difference between winning and losing a competition
- Stacking is very useful when building a team on Kaggle as team members can build their own models on the folds and combine later.