Intelligent Systems Project One

The Front Line

By: Kate Anglin, Jacob Baker, Nick Watts

What is the prediction problem?

- The problem
  - People that don't have a credit history may experience trouble getting a loan. Consumer finance providers cannot accurately predict someone's repayment capabilities without credit history. This is a financial inclusion problem.
  - Can we predict who will default on a loan? (prediction problem)
  - There are only two possible outcomes: the person will default or the person will not default. This is a classification problem. (type of machine learning task)
- The solution
  - We have a lot of data from people that have and have not defaulted on loans.
  - Use that data to create a model capable of predicting who might default on a loan.
  - If someone without credit history wants a loan, this model can be used to predict if they will default.

For the uninformed: Defaulting on a loan means missing a payments on the loan. (This is not someone a lender would want to lend to)

Dataset

- Total size of files: 26.77 gigabytes
- Total training files: 33
- We chose to analyze one: train_person_1.csv
  - Train_person_1.csv contained personal information such as marriage status, date of birth, years employed, employment industry, gender, etc.
  - Each training instance had an index for the case_id. The case_id was the person applying for a loan. If there were multiple instances of a case_id, that meant that they have applied for multiple loans. We decided to go with the first loan for every case_id since our computers could not handle this much data.
- Number of training instances (m):
  - With multiple loan instances: 2,973,992
  - Reduced to one loan instance: 1,526,660
- Number of features: 37
- Range: There are two classes:
  - People that have not defaulted on a loan: 0
  - People that have defaulted on a loan: 1
    - Count of 0: 1478664, which is 96.86%
    - Count of 1: 47994, which is 3.14%

Dataset Pruning

- This was stratified to 1% of those values, then that was used as the 80:20 split for dev.csv test.csv.
  - 15,268 instances.
- For dev.csv
  - Count of 0: 11828, which is 96.86%
  - Count of 1: 384, which is 3.14%
- For test.csv
  - Count of 0: 2957, which is 96.86%
  - Count of 1: 96, which is 3.14%
- We also cut 33 features down to 16 for run time issues.

The Algorithms in Training

We used 5 types of algorithms:

- Logistic Regression
- Multinomial Naive Bayes
- Decision Tree
- Random Forest
- Neural Networks

Logistic Regression With Max Iteration

Accuracy Scores:

Fold 1: 96.85%

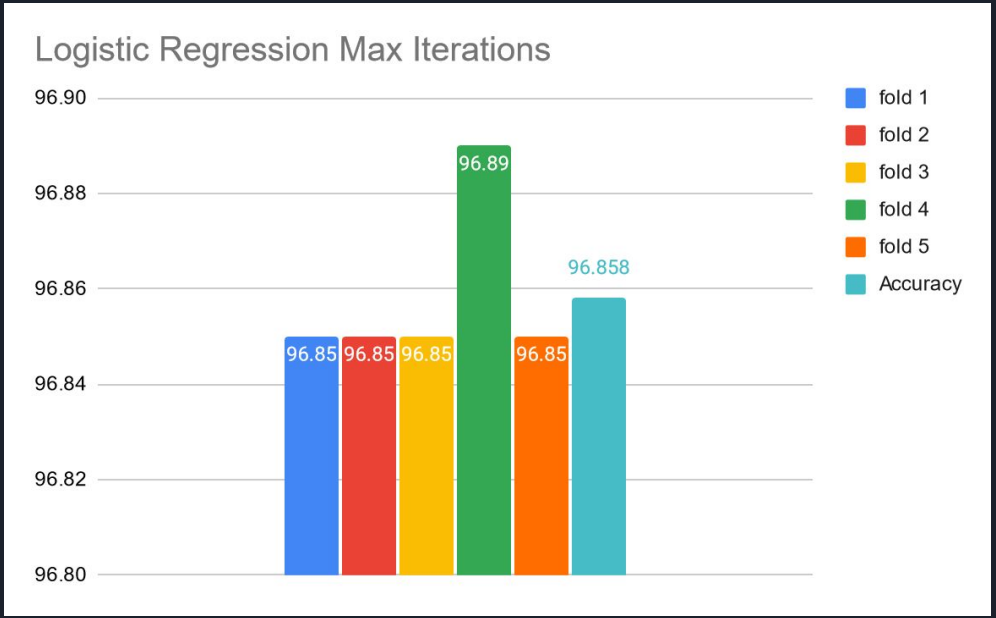
Fold 2: 96.85%

Fold 3: 96.85%

Fold 4: 96.89%

Fold 5: 96.85%

Accuracy Score over all: 96.858%



MultiNomial Naive Bayes With Alpha Of 1.0

Accuracy Scores:

Fold 1: 96.56%

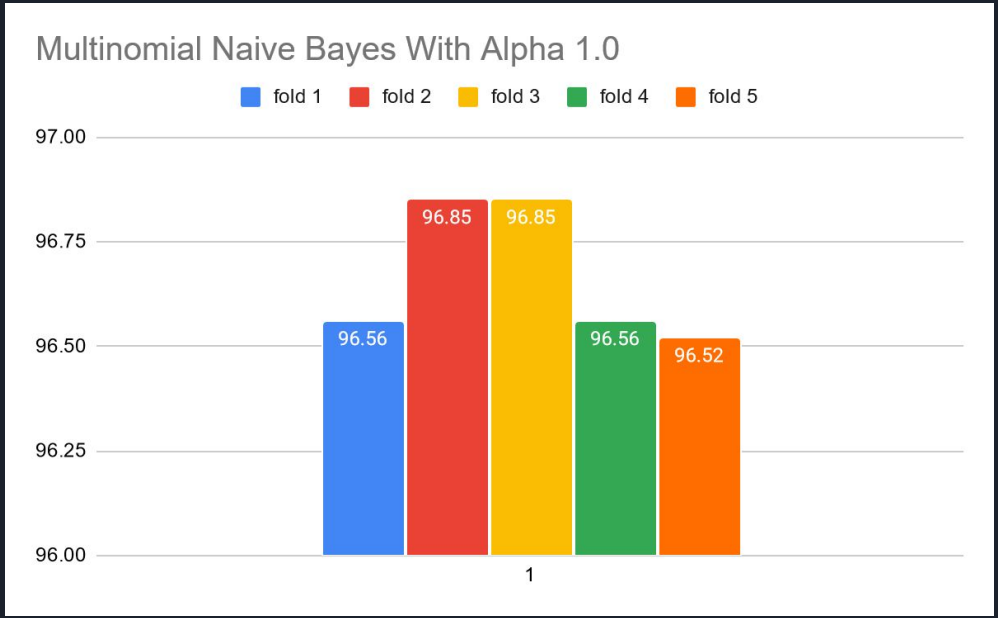
Fold 2: 96.85%

Fold 3: 96.85%

Fold 4: 96.56%

Fold 5: 96.52%

Accuracy Score over all: 96.667% ± 0.0015%



Decision Tree with Depth of 70

Accuracy Scores:

Fold 1: 94.23%

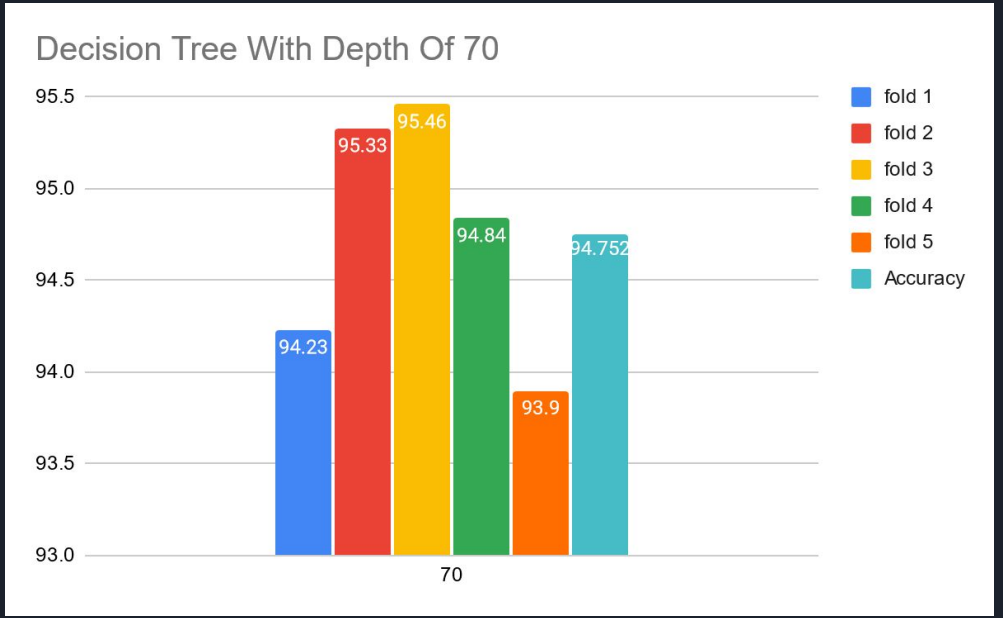
Fold 2: 95.42%

Fold 3: 95.42%

Fold 4: 95.09%

Fold 5: 94.51%

Accuracy Score over all: 94.934%



Random Forest With 100 Trees

Accuracy Scores:

Fold 1: 96.73%

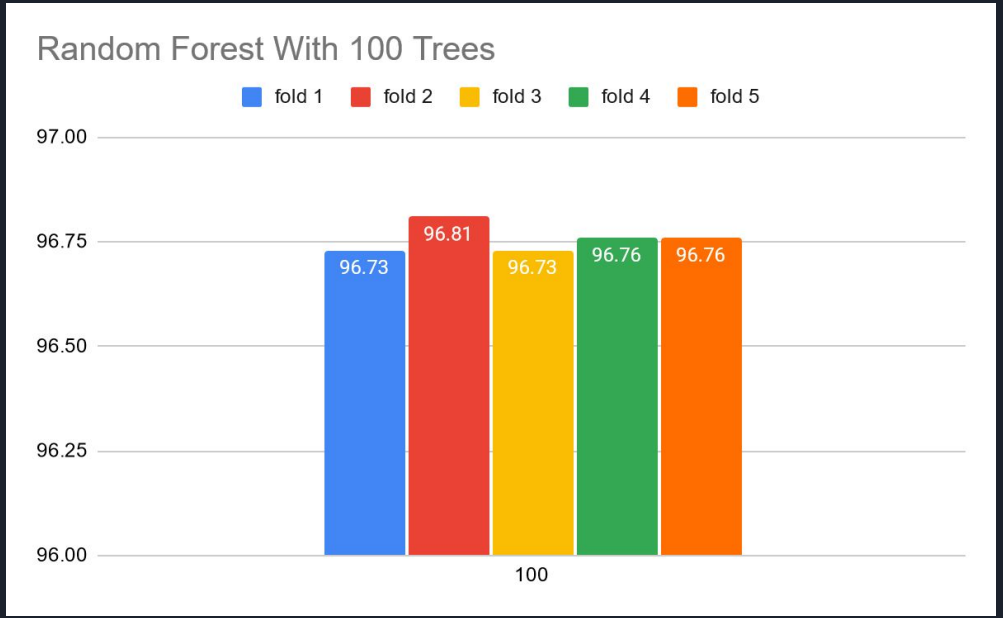
Fold 2: 96.81%

Fold 3: 96.73%

Fold 4: 96.76%

Fold 5: 96.76%

Accuracy Score over all: 96.758%



Neural Network With 100 Hidden Layers

Accuracy Scores:

Fold 1: 96.15%

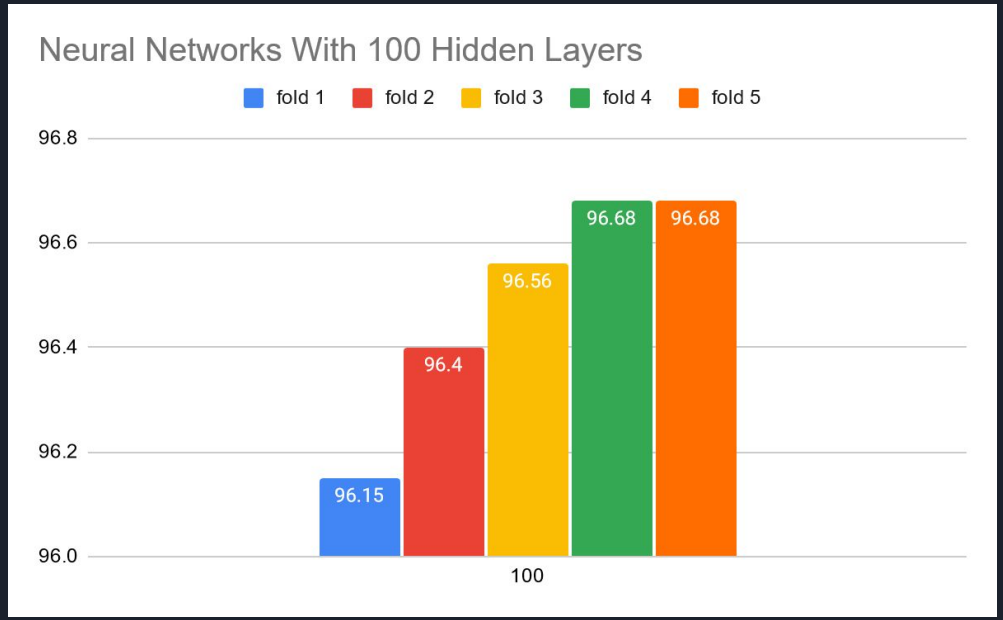
Fold 2: 96.40%

Fold 3: 96.56%

Fold 4: 96.68%

Fold 5: 96.68%

Accuracy Score over all: 96.494%

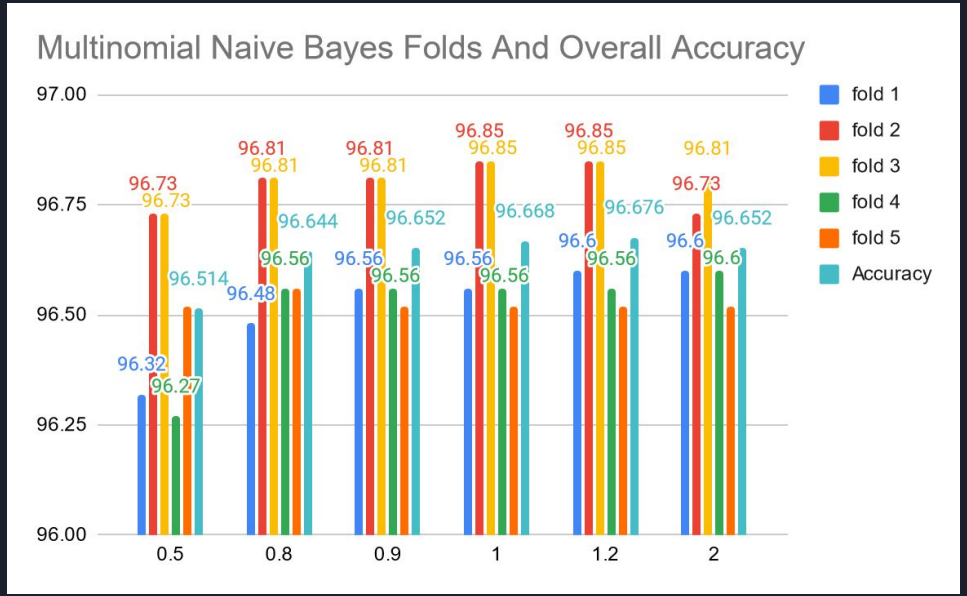


Configurations (Logistic Regression)

- 5 iterations
- 25 iterations
- 50 iterations
- 100 iterations
- 1000 iterations

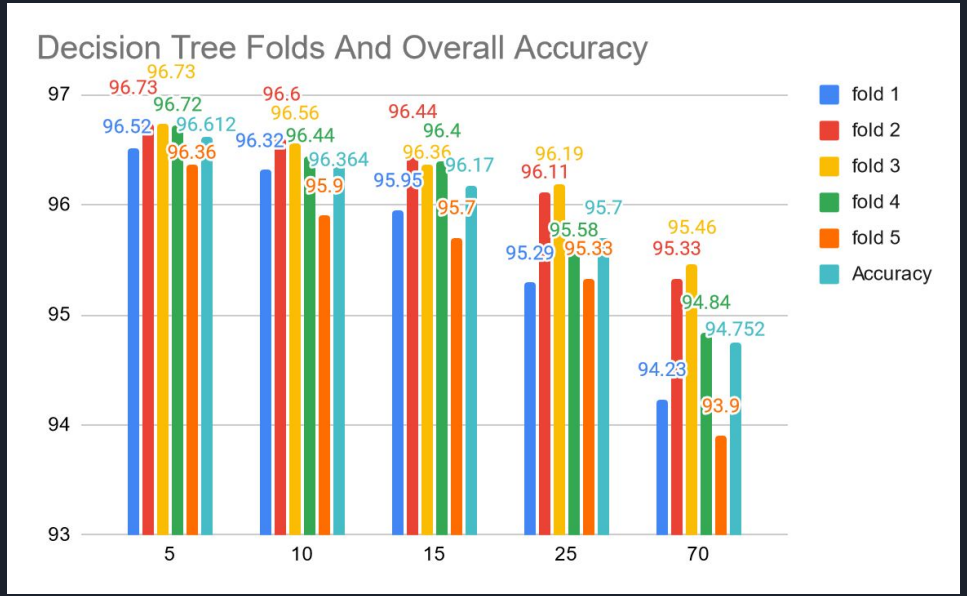
Configurations (Multinomial Naive Bayes)

- 0.5 alpha
- 0.8 alpha
- 0.9 alpha
- 1.0 alpha
- 1.2 alpha
- 2.0 alpha



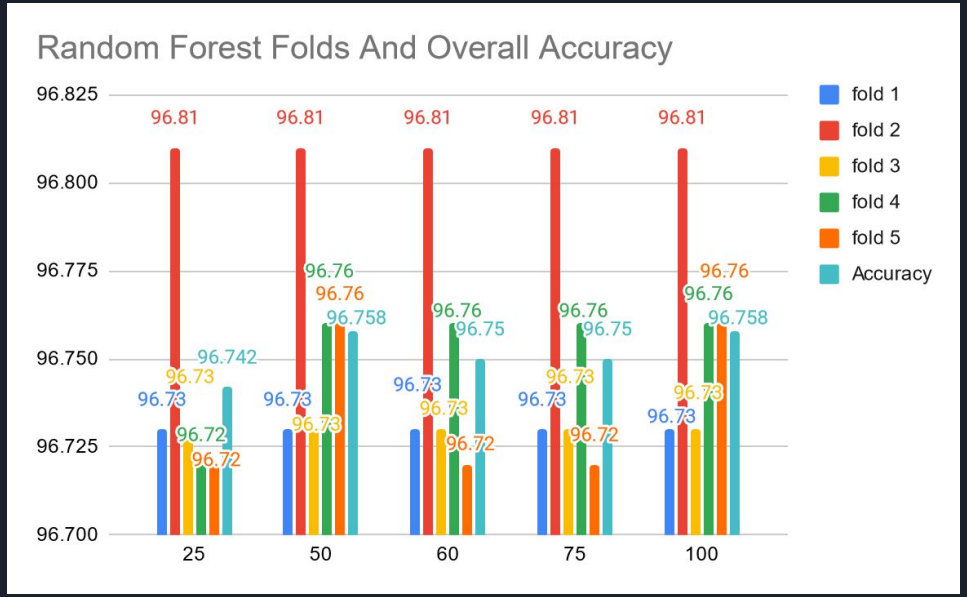
Configurations (Decision Tree)

- Depth 5
- Depth 10
- Depth 15
- Depth 25
- Depth 70



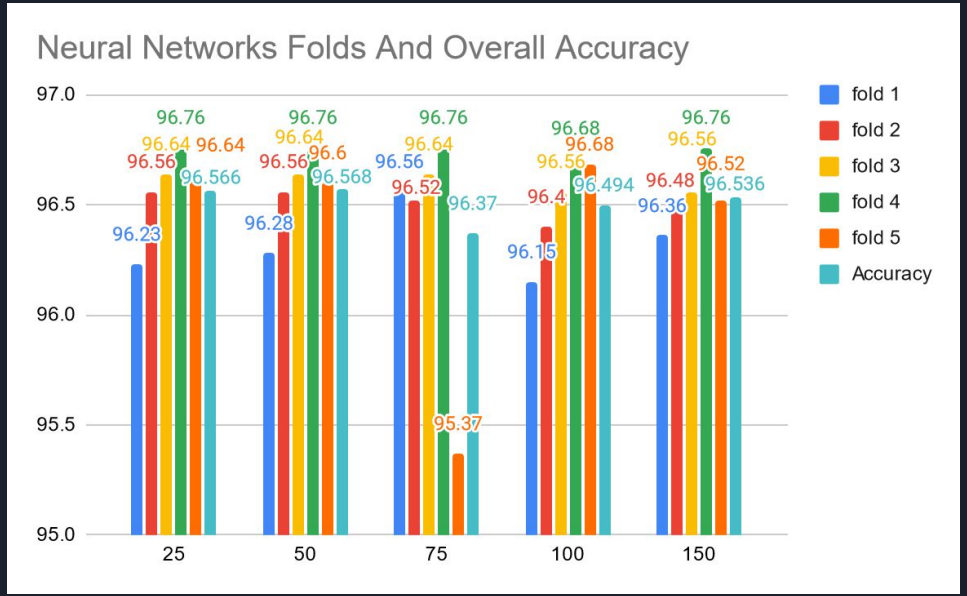
Configurations (Random Forest)

- 25 Trees
- 50 Trees
- 60 Trees
- 75 Trees
- 100 Trees



Configurations (Neural Networks)

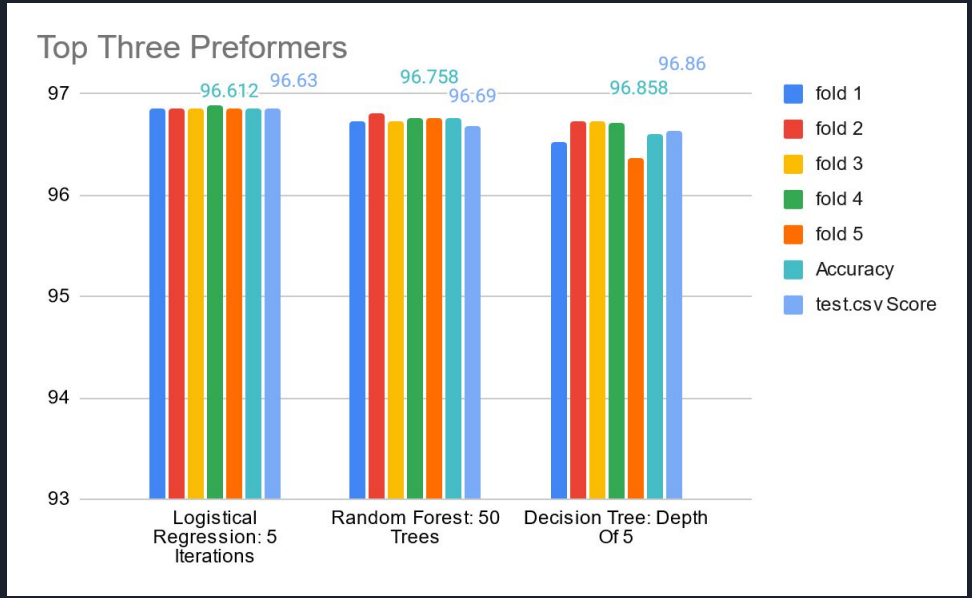
- 25 Hidden Layers
- 50 Hidden Layers
- 75 Hidden Layers
- 100 Hidden Layers
- 150 Hidden Layers



Top Three Best Performers and Results

Test.csv Scores

- Logistic Regression: 96.86%
- Random Forest: 96.69%
- Decision Tree: 96.63%



Top Performer

Logistic Regression with 5 iterations has a 96.86% accuracy on test.csv

Success:

- All of the iterations that we tested all created the same accuracy, meaning this is a positive result indicating that your logistic regression model is performing well on the dataset.
- The dataset is well-structured, or the model has converged to a stable solution.

Clever Tricks

- Removing features - getting rid of features that disrupted the data.
- Data reduction - decreasing the size of the data set
- MinMax Scaler - linearly scales outliers down

Why did it work so well?

- Removing features -
  - Not enough information
  - Skewing the data in the wrong direction
  - Decreasing run time
- Data Reduction -
  - Satisfied random sampling to pick the train data set
  - Decreasing run time
- MinMax Scaler -
  - Made the data's outliers less of an influence

Bonus: Kaggle Competition

Home Credit - Credit Risk Model Stability

Create a model measured against feature stability over time

Overview Data Code Models Discussion Leaderboard Rules Team Submissions

Leaderboard

Raw Data Refresh

the front line

Public Private

This leaderboard is calculated with approximately 30% of the test data. The final results will be based on the other 70%, so the final standings may be different.

Prize Contenders

#	Team	Members	Score	Entries	Last	Join
1389	The Front Line		0.000	7	7m	

Thank you

The Front Line
By: Kate Anglin, Jacob Baker, Nick Watts