

# Machine Learning Engineer Nanodegree

## Capstone Proposal

---

Niall O'Hara  
April 16th, 2018

## Proposal

---

### Domain Background

I am the product lead at [ChangeX.org](https://changex.org), a platform that helps people start projects that transform communities across the world. We do this by providing funding, know-how and social connections to local leaders who apply to start. This year our small team will deal with over 10,000 project applicants. One of the most difficult decisions we make on a daily is selecting which applicants should be approved based on how likely they are to get their projects active and how much of an impact they will have locally. We are currently putting in place data collection measures to start augmenting this decision making process with machine learning and predictive algorithms.

It is for this reason that I have chosen the [\*'Donors Choose – Application Screening'\*](#)<sup>[1]</sup> project on Kaggle for my final ML capstone project. My intention is to learn as much as I can from this project before implementing it at practice at Changex.org when we have sufficient data collected later this year. Founded in 2000 by a high school teacher in the Bronx, [DonorsChoose.org](https://donorschoose.org) empowers public school teachers from across the country to request much-needed materials and experiences for their students. At any given time, there are thousands of classroom requests that can be brought to life with a gift of any amount.

Each year DonorsChoose.org receive hundreds of thousands of applications that they have to decide to approve or not approve manually. This project will focus on building a ML model that will help automate this process to allow their staff to focus on higher value work helping teachers realise their projects. This is a binary classification problem, I will outline the proposed solution to this problem in the coming sections.

## **Problem Statement**

Similar to our experience at ChangeX, DonorsChoose.org receives hundreds of thousands of project proposals each year for classroom projects in need of funding. Right now, large numbers of volunteers are needed to manually screen each submission before it's approved to be posted on the DonorsChoose.org website.

Next year, DonorsChoose.org expects to receive close to 500,000 project proposals. As a result, there are three main problems they need to solve [1].

1. How to scale current manual processes and resources to screen 500,000 projects so that they can be posted as quickly and as efficiently as possible
2. How to increase the consistency of project vetting across different volunteers to improve the experience for teachers
3. How to focus volunteer time on the applications that need the most assistance

The goal of this project is to predict whether or not a DonorsChoose.org project proposal submitted by a teacher will be approved, using the text of project descriptions as well as additional metadata about the project, teacher, and school. DonorsChoose.org can then use this information to identify projects most likely to need further review before approval.

With an algorithm to pre-screen applications, DonorsChoose.org will be able auto-approve some applications quickly so that volunteers can spend their time on more nuanced and detailed project vetting processes, including doing more to help teachers develop projects that qualify for specific funding opportunities.

## **Datasets and Inputs**

The project dataset contains information from teachers' project applications to DonorsChoose.org including teacher attributes, school attributes, and the project proposals including application essays. My objective as outlined above is to use a combination of this data to predict whether or not a DonorsChoose.org project proposal submitted by a teacher will be approved.

## Project File Descriptions

- **train.csv** - the training set [**182,080 Rows**]
- **test.csv** - the test set [**78,036 Rows**]
- **resources.csv** - resources requested by each proposal [**1,048,856 Rows**]

## Data Fields (Test & Train CSVs)

- **id** - unique id of the project application
- **teacher\_id** - id of the teacher submitting the application
- **teacher\_prefix** - title of the teacher's name (Ms., Mr., etc.)
- **school\_state** - US state of the teacher's school
- **project\_submitted\_datetime** - application submission timestamp
- **project\_grade\_category** - school grade levels (PreK-2, 3-5, 6-8, and 9-12)
- **project\_subject\_categories** - category of the project (e.g., "Music & The Arts")
- **project\_subject\_subcategories** - sub-category of the project (e.g., "Visual Arts")
- **project\_title** - title of the project
- **project\_essay\_1** - first essay\*
- **project\_essay\_2** - second essay\*
- **project\_essay\_3** - third essay\*
- **project\_essay\_4** - fourth essay\*
- **project\_resource\_summary** - summary of the resources needed for the project
- **teacher\_number\_of\_previously\_posted\_projects** - number of previously posted applications by the submitting teacher
- **project\_is\_approved** - whether DonorsChoose proposal was accepted (0="rejected", 1="accepted"); train.csv only

\* **Note:** Prior to May 17, 2016, the prompts for the essays were as follows:

- **project\_essay\_1:** "Introduce us to your classroom"
- **project\_essay\_2:** "Tell us more about your students"
- **project\_essay\_3:** "Describe how your students will use the materials you're requesting"
- **project\_essay\_4:** "Close by sharing why your project will make a difference"

Starting on May 17, 2016, the number of essays was reduced from 4 to 2, and the prompts for the first 2 essays were changed to the following:

**project\_essay\_1:** "Describe your students: What makes your students special? Specific details about their background, your neighborhood, and your school are all helpful."

**project\_essay\_2:** "About your project: How will these materials make a difference in your students' learning and improve their school lives?"

For all projects with **project\_submitted\_datetime** of 2016-05-17 and later, the values of **project\_essay\_3** and **project\_essay\_4** will be NaN.

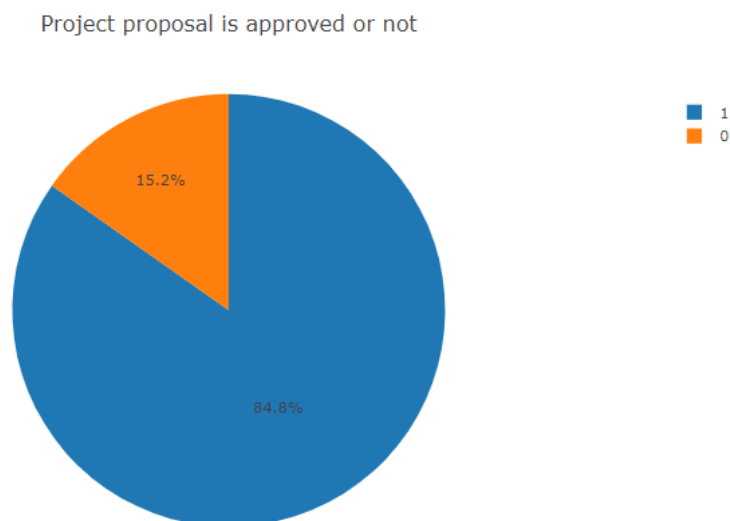
## Data Resource.csv

Proposals also include resources requested. Each project may include multiple requested resources. Each row in **resources.csv** corresponds to a resource, so multiple rows may tie to the same project by id.

- **id** - unique id of the project application; joins with test.csv. and train.csv on id
- **description** - description of the resource requested
- **quantity** - quantity of resource requested
- **price** - price of resource requested

## Target Class Distribution

The training data is highly imbalanced with approximately 85 % of the projects being approved while only 15 % project were not approved. Majority imbalanced class is positive. With this in mind, an evaluation metric of 'Area under ROC curve' is deemed suitable for a problem of this nature.



## Validation Data Set

This project is part of a Kaggle competition. As such the test set provided does not contain the correct labels. Due to this fact I will likely either:

- Hold back 20% of my training data as a separate validation data set
- Use k-fold or shuffle\_split cross validation on my training data set

## **Solution Statement**

*(approx. 1 paragraph)*

This is a binary classification problem. The inputs include categorical, numerical and text features submitted in teacher's applications and the goal is to predict whether the application should be approved or not. As much of the rich data in this project is in the form of long form text statements I will be using natural language processing and TF-IDF vectorization to process & extract useful features.

For benchmarking, comparison and learning purposes I propose utilising three different classification algorithms with varying levels of complexity:

- Logistic Regression
- Random Forest
- Multi Channel Neural Network

Finally, I will select the best model for this problem before fine tuning it's hyperparameters to get the best score. My goal is to understand how basic methods like Logistic Regression work in practice while obtaining a useful comparison of performance to more advanced methods utilising neural networks.











## **Benchmark Model**

The accuracy of the Machine Learning model derived as part of this project will be evaluated on area under the ROC curve between the predicted probability and the observed target. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The true-positive rate is also known as sensitivity, recall or probability of detection in machine learning. The false-positive rate is also known as the fall-out or probability of false alarm and can be calculated as  $(1 - \text{specificity})$ . It can also be thought of as a plot of the Power as a function of the Type I Error of the decision rule (when the performance is calculated from just a sample of the population, it can be thought of as estimators of these quantities) [2].

Benchmark ROC scores of solutions to this problem are outlined in the competitive [leader board on Kaggle](#). A score above a value of 0.76 would put the ML algorithm performance in the top 50% of public results. A score above a value of 0.82 would put the ML algorithm in the top 30% of public results. A score greater than 0.83240 would be the best performing ML algorithm for the problem.

I will also try and form an early benchmark in my project workflow by utilising a simple logistic regression model with stripped back features.

Table 1.1. Public Leader board- Donors Choose Application Screening (Kaggle.com)

#	Δ1w	Team Name	Kernel	Team Members	Score ?	Entries	Last
1	—	Kagemusha			0.83240	149	3d
2	—	Ahmed Alesh			0.82885	107	2d
3	▲2	ml_test			0.82699	52	4d
4	—	Quan Nguyen			0.82683	120	19h
5	new	User of Kaggle			0.82683	5	2d
6	▲3	Patrick DeKelly			0.82660	120	20h
7	▲22	Vikas Pandey			0.82635	7	2d
8	▼1	digitalspecialists			0.82634	111	1d
9	▼6	TetyanaYatsenko			0.82632	151	2d
10	▼4	AlejandroCoronado			0.82632	59	1d

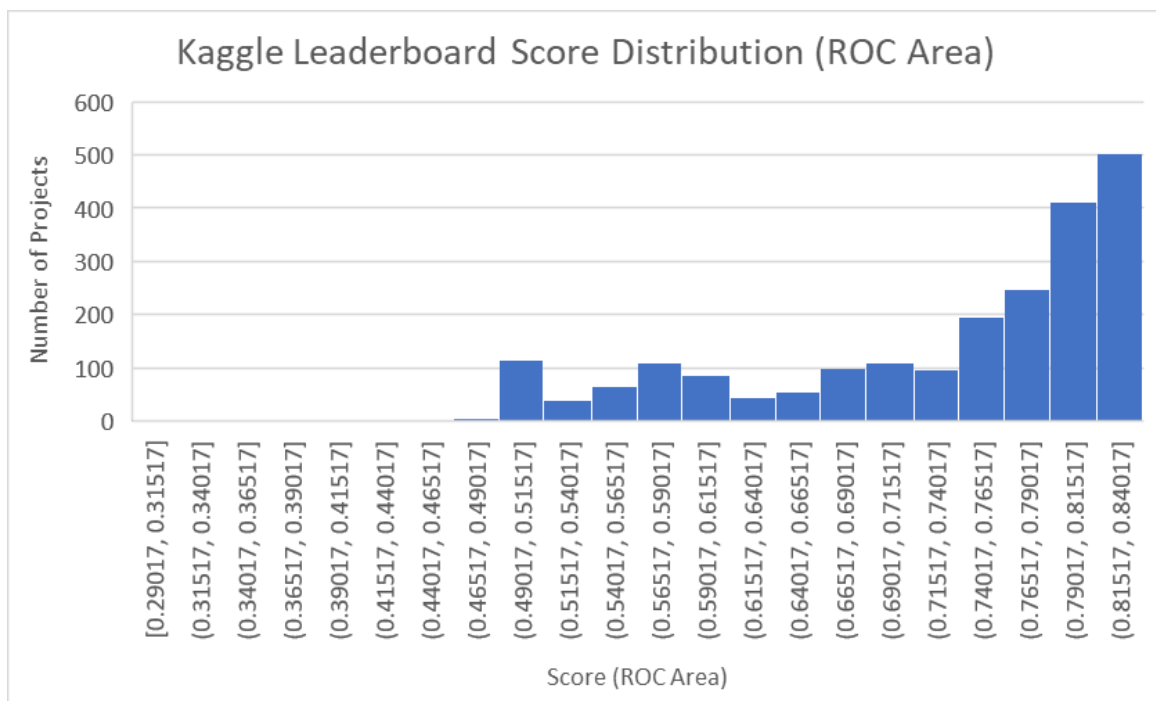


Figure 1.2. Distribution of Kaggle leader board scores (Donors Choose)

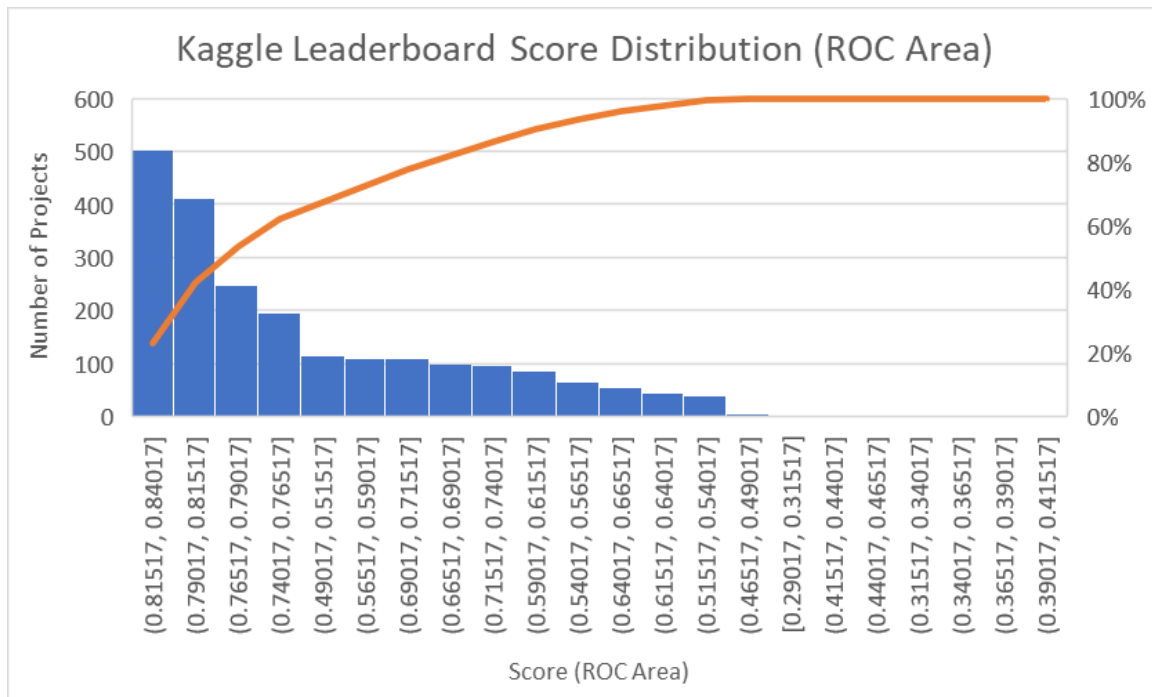


Figure 1.3. Cumulative Distribution of Kaggle leader board scores (Donors Choose)

## Evaluation Metrics

As identified in the benchmark section, the accuracy of the Machine Learning model derived as part of this project will be evaluated on area under the ROC curve between the predicted probability and the observed target. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The true-positive rate is also known as sensitivity, recall or probability of detection in machine learning. The false-positive rate is also known as the fall-out or probability of false alarm and can be calculated as  $(1 - \text{specificity})$ . It can also be thought of as a plot of the Power as a function of the Type I Error of the decision rule (when the performance is calculated from just a sample of the population, it can be thought of as estimators of these quantities) [2].

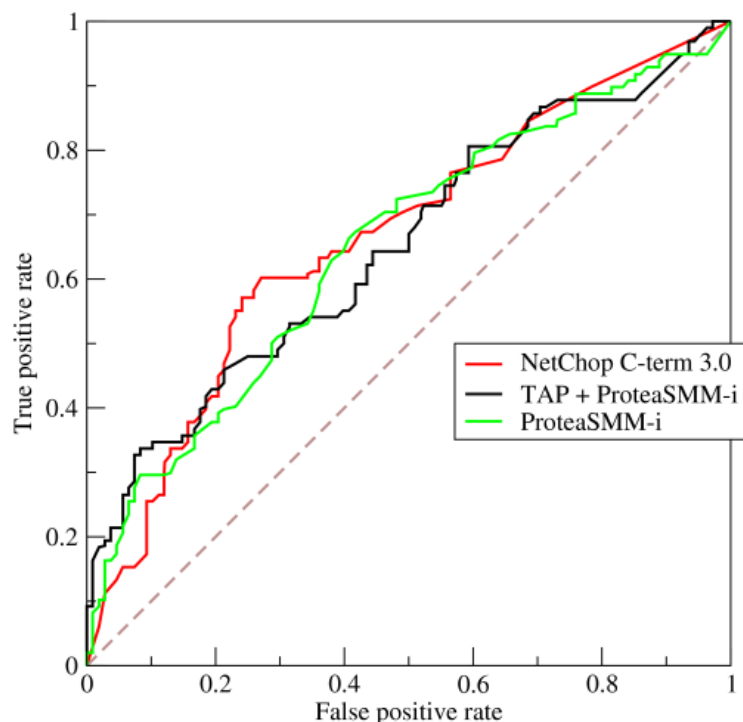


Figure 1.4. Example ROC Curve - Three predictors of peptide cleaving in the proteasome. [1]

In a binary classification like this one, the outcomes are labelled either as positive (p) or negative (n). There are four possible outcomes from a binary classifier. If the outcome from a prediction is p and the actual value is also p, then it is called a true positive (TP); however if the actual value is n then it is said to be a false positive (FP). Conversely, a true negative (TN) has occurred when both the prediction outcome and the actual value are n, and false negative (FN) is when the prediction outcome is n while the actual value is p.

Let us define an experiment from P positive instances and N negative instances for some condition. The four outcomes can be formulated in a 2x2 contingency table or confusion matrix, as follows:

		True condition			
		Condition positive	Condition negative	Prevalence = $\frac{\sum \text{Condition positive}}{\sum \text{Total population}}$	Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$
Predicted condition	Predicted condition positive	<b>True positive,</b> Power	<b>False positive,</b> Type I error	Positive predictive value (PPV), Precision = $\frac{\sum \text{True positive}}{\sum \text{Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\sum \text{False positive}}{\sum \text{Predicted condition positive}}$
	Predicted condition negative	<b>False negative,</b> Type II error	<b>True negative</b>	False omission rate (FOR) = $\frac{\sum \text{False negative}}{\sum \text{Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\sum \text{True negative}}{\sum \text{Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection = $\frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\sum \text{False positive}}{\sum \text{Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR-}}$  F <sub>1</sub> score = $\frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$
		False negative rate (FNR), Miss rate = $\frac{\sum \text{False negative}}{\sum \text{Condition positive}}$	True negative rate (TNR), Specificity (SPC) = $\frac{\sum \text{True negative}}{\sum \text{Condition negative}}$	Negative likelihood ratio (LR-) = $\frac{\text{FNR}}{\text{TNR}}$	

Figure 1.5. Calculations to determine a confusion matrix or 2x2 contingency table [2]



## Project Design

Having carried out an intensive data exploration and visualisation exercise, the early parts of the project will focus on processing the project input data into formats that are suitable for modelling. We will begin by joining the data from resources.csv and train.csv. We will clean the data by removing outliers or data that could negatively affect our analysis. From the data description given it appears that the lack of quality data available for Essays 3 and 4 could render them unnecessary for this project.

We will then split the input data into categorical, numerical and text columns. This will allow us to pre-process the data effectively and also feed the input data into our training algorithms effectively also. Once segmented we will undertake a process of encoding and scaling the categorical and numerical features in the data.

One of the main elements of the data pre-processing will involve processing the text bodies of the teacher applications using TFIDF vectorization. In information retrieval, TFIDF, short for term frequency-inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus [3,4]. We use TFIDF to determine weighting factors whose values increase proportionally to the number of times a word appears in the teacher essays while offsetting by the frequency of the word in the corpus, which will help to adjust for the fact that some words appear more frequently in general.

As the problem being solved is one of binary classification we will train our classification models using Logistic Regression, Random Forest Classifier and Multi Channel Neural Networks. Our intention is to explore simpler models (Logistic Regression) alongside a more complex Neural Network solution to compare performance in terms of speed, complexity and train/test times.

A multi-channel is being utilised to facilitate the input of 3 distinct data types into the final classification decision (categorical, numerical, text). The final layers of the neural network will merge these separate input channels to give a final predictive output layer.

The final steps of the project will involve evaluating the performance of the trained models on the held back test data. The models performance will be evaluated by establishing the area under a ROC curve.

Once the best performing algorithm is established a final round of hyper parameter tuning will be carried out to obtain the best results possible.

## **My project solution will be organised around the following workflow:**

1. Importing Libraries, Modules & Packages
2. Data Exploration & Visualisation
3. Loading & Pre-processing Data:
  - a. Joining disparate data sources (resources.csv + train.csv)
  - b. Removing unwanted data (e.g. Essays 3 & 4)
  - c. Splitting data into Categorical, Numerical & Text Columns
  - d. Encoding/Factorizing & Scaling (Numerical & Categorical Features)
  - e. Text Processing (TF-IDF vectorization, Snowbell Stemmer & Stacking)
4. Model Training: Will explore the following as part of this project
  - a. Option 1: Logistic Regression
  - b. Option 2: Random Forest
  - c. Option 3: Multi Channel Neural Network (Merged Layers)
  - d. Option 4: XgBoost
  - e. Option 5: Ligth GBM
5. Results:
  - a. Predictive Estimations (Test Features)
  - b. Model Scores: Area under ROC curve

## **References**

- [1] "Kaggle Competition: Donors Choose Application Screening", 17/04/18, <https://www.kaggle.com/c/donorschoose-application-screening>
- [2] "Receiver operating characteristic", 16/04/18, [https://en.wikipedia.org/wiki/Receiver\\_operating\\_characteristic](https://en.wikipedia.org/wiki/Receiver_operating_characteristic),
- [3] "tf-idf", 16/04/18, <https://en.wikipedia.org/wiki/Tf%E2%80%93idf>
- [4] Breitingner, Corinna; Gipp, Bela; Langer, Stefan (2015-07-26). "Research-paper recommender systems: a literature survey". International Journal on Digital Libraries. 17 (4): 305–338. doi:10.1007/s00799-015-0156-0. ISSN 1432-5012.