# 1 Introduction

## Group Members

Julia Reisler, Zafir Abou-Zamzam, and Michelle Zhao

## Team Name

flickr.com/photos/yisongyue

## Division of Labor

* Julia: Wrote helper functions to calculate classification error, write predictions to a file, and use term frequency-inverse document frequency to transform the feature data. Implemented a variety of models, including logistic regression, naïve Bayes, linear classifier with stochastic gradient descent training, Bernoulli naïve Bayes, linear support vector machine, ridge classification, and K neighbors classification. Built off of Michelle and Zafir’s neural networks to create a deeper network using a hyperbolic tangent as the activation function. Developed different methods of blending the models together to use predictions from subsets of different models. Split the data into training and validation sets and used these to optimize model parameters.
* Zafir:
* Michelle:

**TODO: insert what Michelle and Zafir did**

# 2 Overview

* Models and Techniques tried
  + Models: logistic regression, multinomial naïve Bayes, linear classifier with stochastic gradient descent training, Bernoulli naïve Bayes, linear support vector machine, ridge classification, K neighbors classification, neural network.
  + Techniques: TF-IDF, blending multiple models.
* Work Timeline
  + Weekend before competition: met to discuss general methods and approaches to the problem.
  + Week of competition: tested and submitted models individually in addition to meeting 3 times to work together on models.

# 3 Approach

* Data processing and manipulation
  + We used the scikit-learn TfidfVectorizer to convert the features into TF-IDF features. To do this, we turned each data point into a string using the words present in the data point, and then used the TfidfVectorizer method. TF-IDF is a way of weighting words based on how important they are in a document and set of documents. The weight for a word increases as the prevalence of the word in a document increases, but decreases as the prevalence of the word in the set of documents increases. We noticed that using TF-IDF generally decreased the validation error from our models. For example, with basic linear regression, the error validation error decreases from around 14.725% without TF-IDF to about 14.425%.
* Details of models and techniques
  + Details of each model: In the end, our best score resulted from a blending of a logistic model, linear classifier with stochastic gradient descent training, multinomial naïve Bayes, and a neural network. We used scikit-learn’s implementation from the first three and used keras for the neural network. After doing research about the best models to use with bag of words data, we decided on these four. Logistic regression relates the probability of an input belonging to a particular group, is relatively simple, and can work well on sparse matrixes. Assuming that the data is nearly linearly separable, we thought that a linear classifier would work well due to its simplicity. Naïve Bayes assumes each feature is completely independent of other features and uses maximum likelihood estimates to classify data. **TODO: INSERT REASON FOR USING NEURAL NETWORK AND DISADVANTAGES OF THESE MODELS**.
  + Blending: Noting that each model uses a different method to classify data differently, we thought that blending the models would reduce overfitting from each model by forming a consensus through different parameters used to classify the data. We also used scikit-learn’s notions of confidence intervals for the linear and logistic models. Noticing that when these two models had high confidence in their classification, the classification error was only around 5%, we decided to only use all four models to classify in cases when the confidence scores for both logistic and linear regression had a magnitude less than 1. In those cases, we took a majority vote for the classification, giving the logistic model two votes since it was the best model.
  + Testing: To test our data, we split the given set into 80% training and 20% validation. While this was done to prevent overfitting, we chose parameters that optimized the classification on the validation set, which actually caused overfitting to the validation set.

# 4 Model Selection

* Validation and Test: To test our data, we split the given set into 80% training and 20% validation. While this was done to prevent overfitting, we chose parameters that optimized the classification on the validation set, which actually caused overfitting to the validation set. However, ideally, the best score on the validation set would have helped us figure out which model would perform best on the test set.
* Scoring: We scored our models based on classification error from the validation set. On a particular partition of the dataset, the logistic regression scored 14.775% classification error, the linear regression scored 15.425%, naïve Bayes scored 16.925%, the neural network scored 15.85%, and the blended model scored 14.25%.

**TODO: WRITE CONCLUSION AND INCLUDE GRAPHS THAT SHOW THE PERFORMANCE OF MODELS**