GMU Fall 2013, CS 780
Class Project

Partly Sunny with Chance of HashTags

a Kaggle Competition, Sponsored by CrowdFlower

Presented by

Talha Oz
Venkat Tadakamalla

Agenda

1. Problem Description
2. Dataset
3. Goal
4. Tools and Software Packages Used
5. Processing Steps
6. Results

1. Problem Description

Partly Sunny with Chance of HashTags

a Kaggle Competition sponsored by CrowdFlower Open Data Library

Training Data: Approx. **78K** weather related tweets labeled by multiple people, crowd sourced !

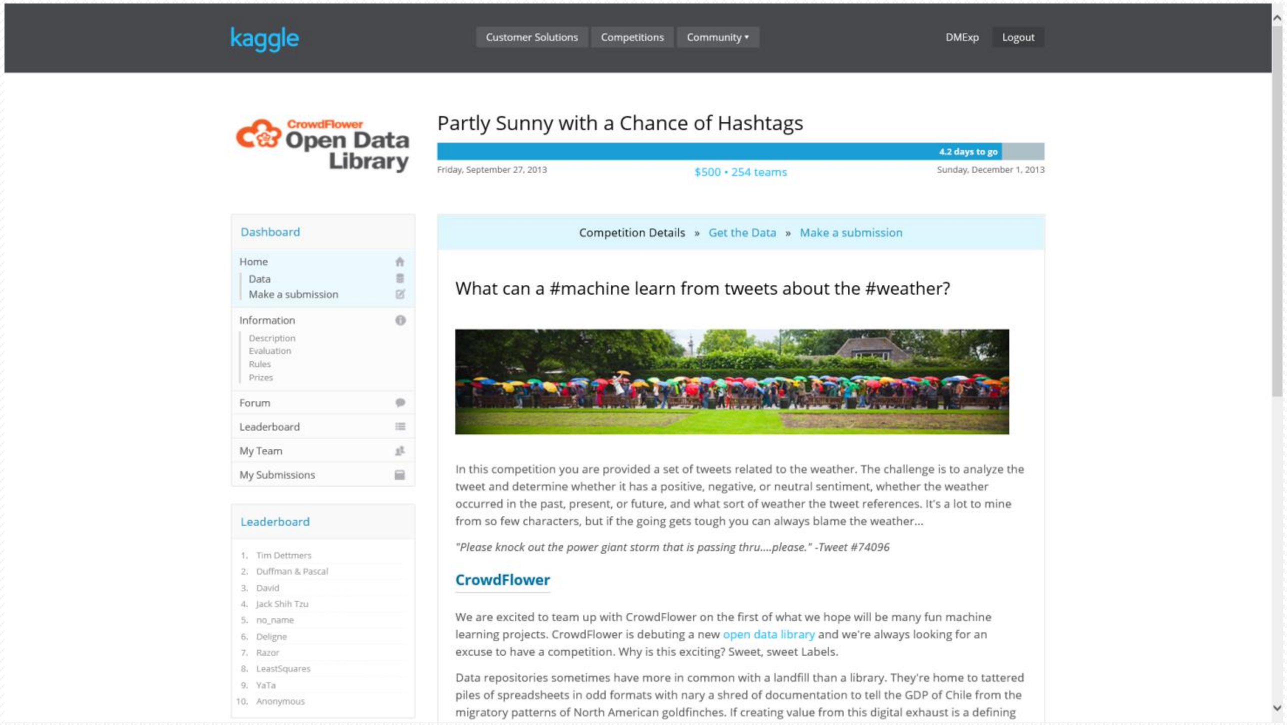
Test Data: Approx. **42K**. Predict **24 classes** in 3 different categories.

- whether it has a positive, negative, or neutral sentiment
- whether the weather occurred in the past, present, or future
- what sort of weather the tweet references.

Let us look at some more details…
from *Kaggle* website

1a. Problem Description, contd.

Kaggle specific requirements: No external info limits for this competition



kaggle · Customer Solutions · Competitions · Community ▾ · DMExp · Logout

CrowdFlower Open Data Library

Partly Sunny with a Chance of Hashtags

4.2 days to go

Friday, September 27, 2013 · $500 • 254 teams · Sunday, December 1, 2013

Dashboard

- Home
  - Data
  - Make a submission
- Information
  - Description
  - Evaluation
  - Rules
  - Prizes
- Forum
- Leaderboard
- My Team
- My Submissions

Leaderboard

1. Tim Dettmers
2. Duffman & Pascal
3. David
4. Jack Shih Tzu
5. no_name
6. Deligne
7. Razor
8. LeastSquares
9. YaTa
10. Anonymous

Competition Details » Get the Data » Make a submission

What can a #machine learn from tweets about the #weather?

In this competition you are provided a set of tweets related to the weather. The challenge is to analyze the tweet and determine whether it has a positive, negative, or neutral sentiment, whether the weather occurred in the past, present, or future, and what sort of weather the tweet references. It's a lot to mine from so few characters, but if the going gets tough you can always blame the weather...

"Please knock out the power giant storm that is passing thru....please." -Tweet #74096

CrowdFlower

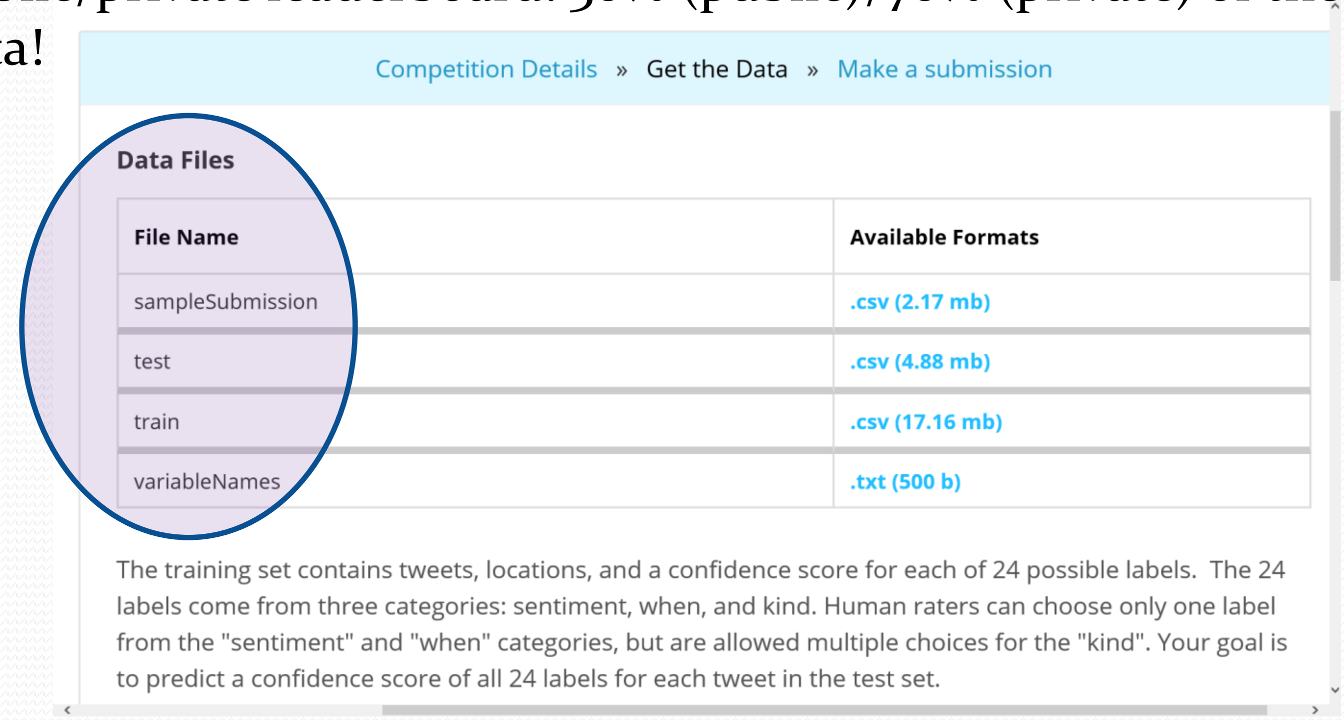
We are excited to team up with CrowdFlower on the first of what we hope will be many fun machine learning projects. CrowdFlower is debuting a new open data library and we're always looking for an excuse to have a competition. Why is this exciting? Sweet, sweet Labels.

Data repositories sometimes have more in common with a landfill than a library. They're home to tattered piles of spreadsheets in odd formats with nary a shred of documentation to tell the GDP of Chile from the migratory patterns of North American goldfinches. If creating value from this digital exhaust is a defining

Started: **2:43 pm, Friday 27 September 2013 UTC**
Ends: **11:59 pm, Sunday 1 December 2013 UTC (65 total days)**

1b. Problem Description, contd.

- Compete with data miners all around the world (259 teams, 300 players)
- Daily upload limit for each team (5)
- Public/private leaderboard: 30% (public)/70% (private) of the test data!



Data Files

File Name	Available Formats
sampleSubmission	.csv (2.17 mb)
test	.csv (4.88 mb)
train	.csv (17.16 mb)
variableNames	.txt (500 b)

The training set contains tweets, locations, and a confidence score for each of 24 possible labels. The 24 labels come from three categories: sentiment, when, and kind. Human raters can choose only one label from the "sentiment" and "when" categories, but are allowed multiple choices for the "kind". Your goal is to predict a confidence score of all 24 labels for each tweet in the test set.

1c. Problem Description, contd.

- 3 categories, **24 classes**:

Sentiment (5)	When (4)	Kind (15)
s1,"I can't tell"	w1,"current (same day) weather"	k1,"clouds"
s2,"Negative"	w2,"future (forecast)"	k2,"cold"
s3,"Neutral / author is just sharing information"	w3,"I can't tell"	k3,"dry"
s4,"Positive"	w4,"past weather"	k4,"hot"
s5,"Tweet not related to weather condition"		k5,"humid"
		k6,"hurricane"
		k7,"I can't tell"
		k8,"ice"
		k9,"other"
		k10,"rain"
		k11,"snow"
		k12,"storms"
		k13,"sun"
		k14,"tornado"
		k15,"wind"

1d. Problem Description, contd.

- Each tweet is labeled by multiple people
- Graders have reliability scores but we do not know them, instead:
- For When and Sentiment categories we have a probability distribution that sums up to one. Kind values can be more than one !

id	tweet	state	location
10	Rather be storm chasing.	district of columbia	Washington, DC
11	#WEATHER: 1:54 pm : 61.0F. Feels 60F. 29.98% Humidity. 6.9MPH East Wind.	michigan	St Louis, MI, USA

Y: S, W, K
(continuous variables)

s1	s2	s3	s4	s5
0	0	0.6	0	0.4
0	0	1	0	0

w1	w2	w3	w4
0	0	1	0
1	0	0	0

k1	k2	k3	k4	k5	k6	k7	k8	k9	k10	k11	k12	k13	k14	k15
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0.203	0.176	0	0.376	0	0.421	0	0.176	0	0	0	0	0	0.579

✓ This is a multi-class continuous value problem !

2. Dataset

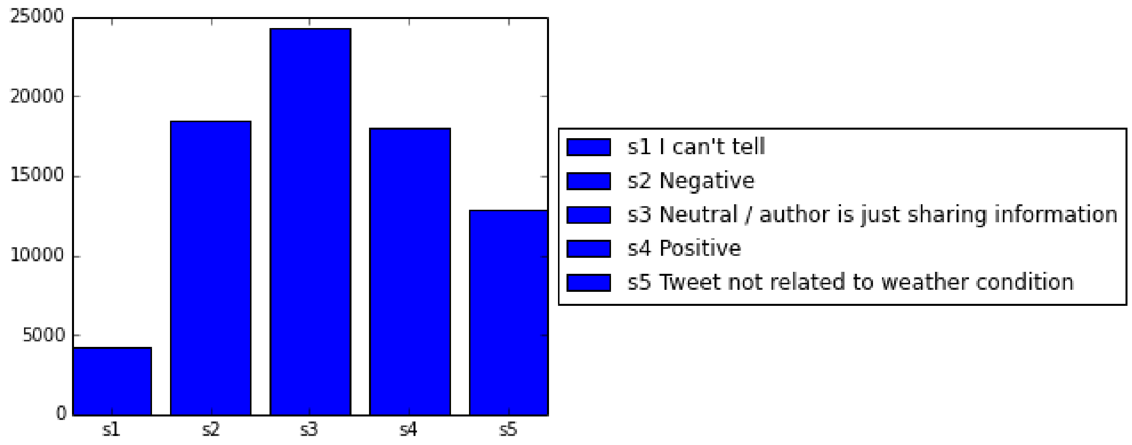
- Training Records (tweets) = 77,946
- Test Records (tweets) = 42,147
- Class Variables →

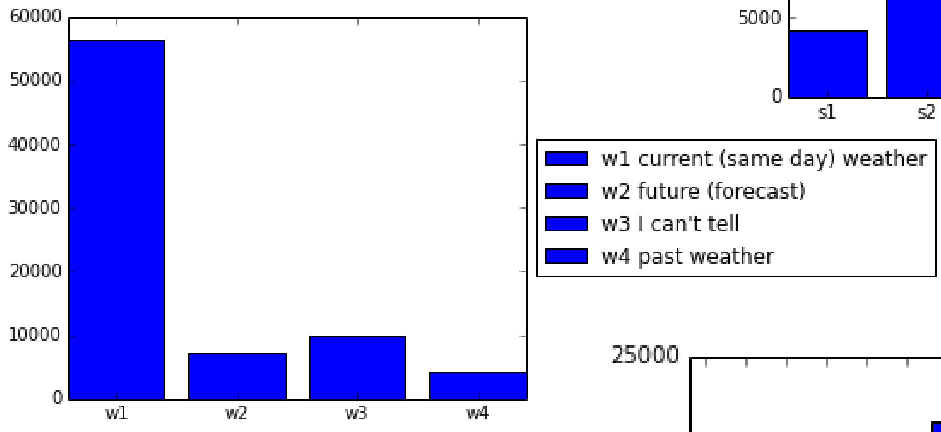
Sentiment	When	Kind
s1,"I can't tell"	w1,"current (same day) weather"	k1,"clouds"
s2,"Negative"	w2,"future (forecast)"	k2,"cold"
s3,"Neutral / author is just sharing information"	w3,"I can't tell"	k3,"dry"
s4,"Positive"	w4,"past weather"	k4,"hot"
s5,"Tweet not related to weather condition"		k5,"humid"
		k6,"hurricane"
		k7,"I can't tell"
		k8,"ice"
		k9,"other"
		k10,"rain"
		k11,"snow"
		k12,"storms"
		k13,"sun"
		k14,"tornado"
		k15,"wind"

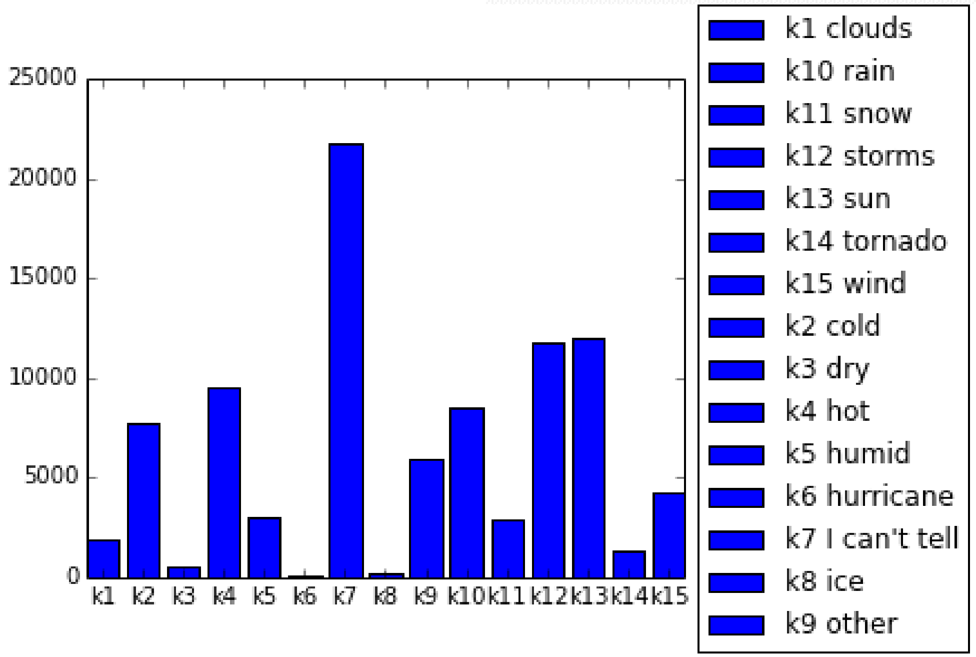
- Sample tweets:

```
"1","Jazz for a Rainy Afternoon: {link}","oklahoma","Oklahoma", //id, tweet and location
"0","0","1","0","0", //sentiment
"0.8","0","0.2","0", //when
"0","0","0","0","0","0","0","0","0","1","0","0","0","0","0" //kind
"30","Today was windy AF, I don't wanna go meet up w/ @mention and get my sis from school
:/","arizona","Phoenix (Prescott)",
"0","0.613","0.387","0","0", //sentiment
"0.793","0","0","0.207", //when
"0","0","0","0","0","0","0","0","0","0","0","0","0","0","1" //kind
```

2a. Dataset, contd.







3. Goal

- Minimize the root mean square error !

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(p_i - a_i)^2}{n}}$$

where:

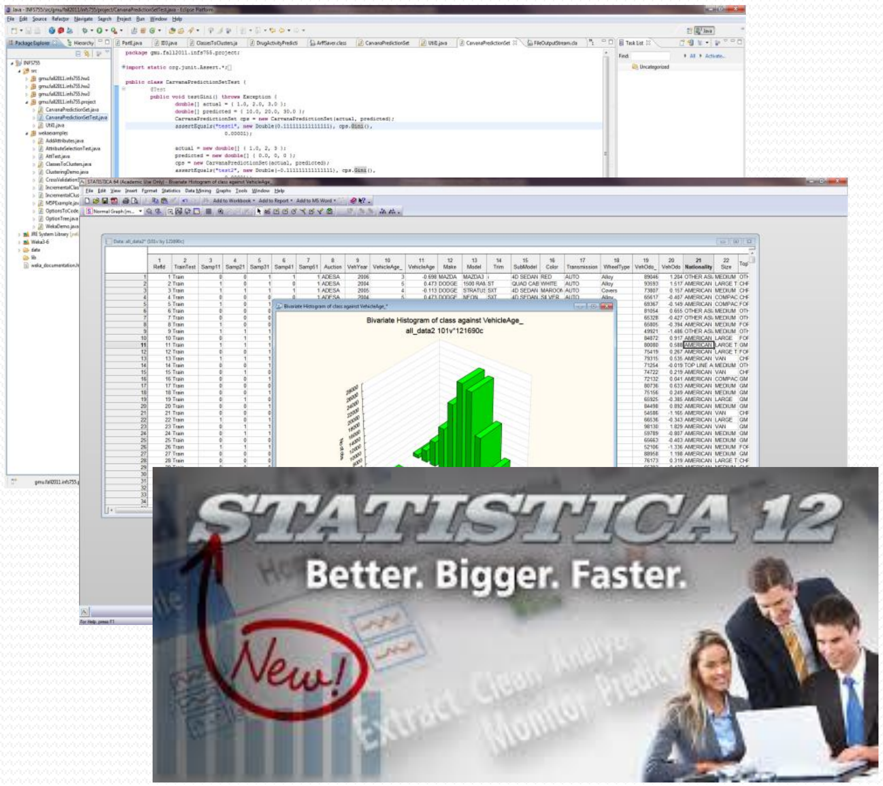
- n is 24 times the total number of tweets
- p_i is the predicted confidence rating for a given label
- a_i is the actual confidence rating for a given label

- Python code snippet:

```
np.sqrt(np.sum(np.array(pred-y_true)**2)/(y_true.shape[0]*float(y_true.shape[1])))
```

4. Tools and Software Packages Used

- Python
  - Numpy, Scipy, Pandas
  - Scikit-learn, NLTK
- Java
  - Pre and Post Processing
  - Explored MULAN, an open-source Java library based on WEKA for multi-class multi-label predictions. Found to work only for categorical values so discarded the tool.
- Statsoft' s *STATISTICA*
  - model building and predictions

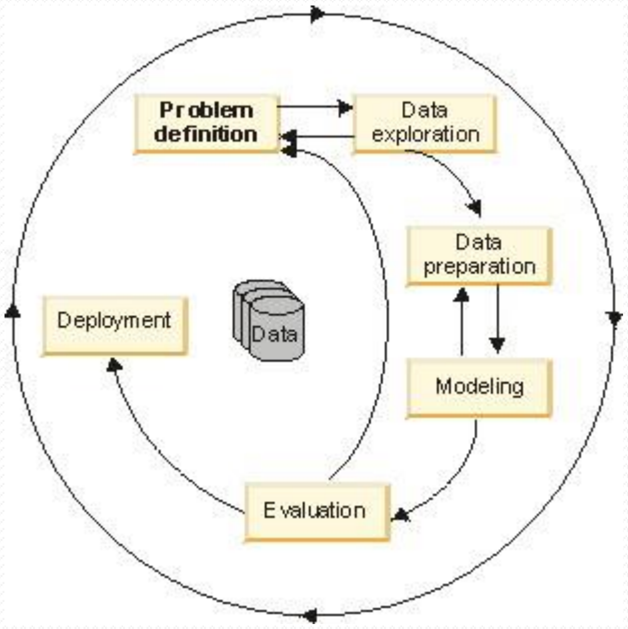


```
package gmu.fall2011.infs755.project;

import static org.junit.Assert.*;

public class CarvanaPredictionSetTest {
        @Test
        public void testGini() throws Exception {
                double[] actual = { 1.0, 2.0, 3.0 };
                double[] predicted = { 10.0, 20.0, 30.0 };
                CarvanaPredictionSet cps = new CarvanaPredictionSet(actual, predicted);
                assertEquals("test1", new Double(0.1111111111111111), cps.Gini(),
                                0.00001);

                actual = new double[] { 1.0, 2, 3 };
                predicted = new double[] { 0.0, 0, 0 };
                cps = new CarvanaPredictionSet(actual, predicted);
                assertEquals("test2", new Double(-0.1111111111111111), cps.Gini(),
```

STATISTICA 64 (Academic Use Only) - Bivariate Histogram of class against VehicleAge_
Data: all_data2* (101v by 121690c)
Bivariate Histogram of class against VehicleAge_
all_data2 101v*121690c
STATISTICA 12
Better. Bigger. Faster.
New!
Extract Clean Analyze Monitor Predict

5. Processing Steps

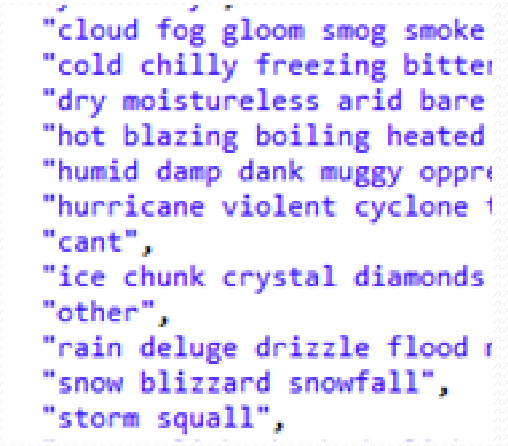
a) Pre-process data
b) Build models
c) Evaluate models
d) Predict and Deploy
(upload to Kaggle site)



5a. Pre-process Data

- Feature Extraction/Creation from raw tweets
  - Raw words, Porter/Lancaster/Snowball stemmers, wordnet lemmatizer (NLTK)
  - Stop words removal (NLTK, scikit-learn)
  - Emoticons (happy/sad)
  - Unigrams, bigrams, together (scikit-learn)

- *STATISTICA*: for some selective runs, added additional features (Label_1 to Label_24) which are computed in case a synonym for any of the class label is found. Improved the RMSE by about 10%.



```
"cloud fog gloom smog smoke
"cold chilly freezing bitte
"dry moistureless arid bare
"hot blazing boiling heated
"humid damp dank muggy oppr
"hurricane violent cyclone
"cant",
"ice chunk crystal diamonds
"other",
"rain deluge drizzle flood
"snow blizzard snowfall",
"storm squall",
```

5a. Pre-process Data, contd.

- Feature Selection/Reduction
  - Stacking state and location information to tweets
  - TFIDF vectorizer (max_features, smoothing, min_df, max_df), count vectorizer
  - LSA (truncated SVD, work with sparse matrix, on sample vec)
  - PCA (dense only, on covariance matrix)
  - SelectKBest(chi2, class/feature dependency)

- *STATISTICA*: Utilized the best feature selection option for each individual classes. Used top 100 of such features for each class.

5b. Build Models

Used several algorithms

- Regressors (continuous, clip [0,1])
  - Ridge Regressor (sklearn)
  - SGD (Stochastic Gradient Descent) Regressor (sklearn)
- Classifiers
  - Predict_proba
  - MultinomialNB (cs780 hw2, sklearn)
  - KNN
- Ensemble methods, Regression (Using *STATISTICA**)
  - Boosted Trees
  - Random Forests

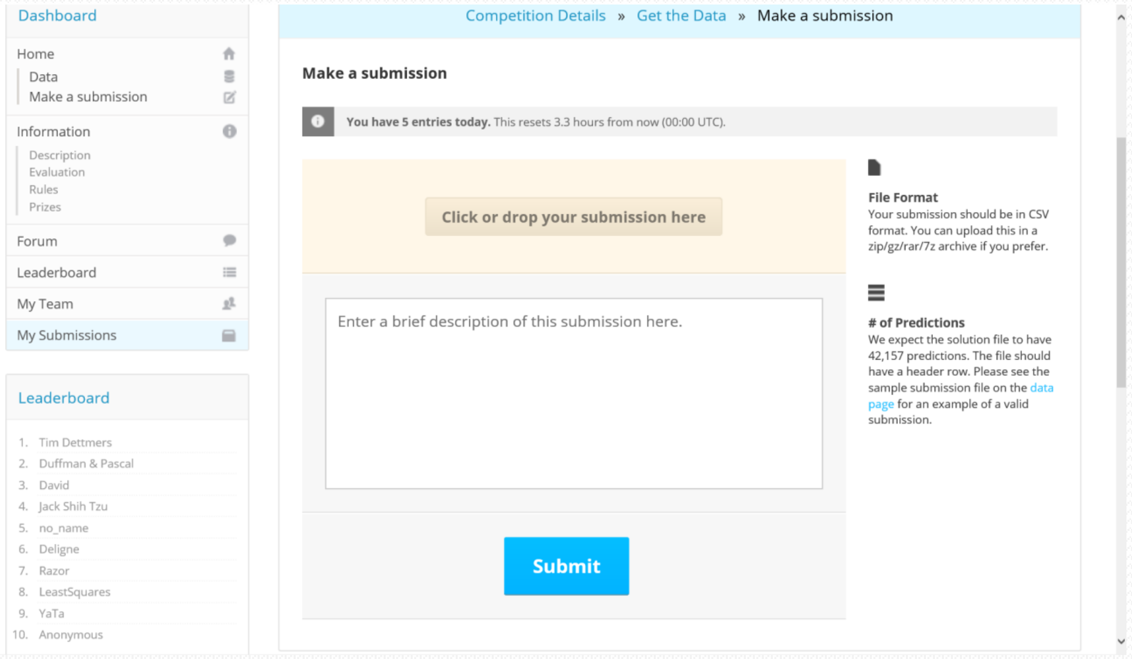
** Noticed the following limitations (may or may not be accurate as we have a very limited time to explore the tool): Required the input data to be in sparse format; No multi-class support for many models. Typical test/ train file with about top 4000 features was approx. 1GB. Typical execution on a 64 bit AMD 2.5 GHz Quad Core PC with 8GB of RAM took about an hour (Execution of each model consisted of 24 individual executions, one for each class, manual assembling of results into CSV file).*

5c. Evaluate Models

- Cross Validation (everything in CV loop)
- Optimizing predictor variables (Grid Search)
- Optimizing token vectorizer variables

5d. Predict & Deploy

- With the best models figured out in the previous step, run the model on the `Test Set` to get the predictions.
- Upload the predictions to `Kaggle` web site.


Dashboard
Home
Data
Make a submission
Information
Description
Evaluation
Rules
Prizes
Forum
Leaderboard
My Team
My Submissions
Competition Details » Get the Data » Make a submission
Click or drop your submission here
Enter a brief description of this submission here.
Submit

Make a submission

You have 5 entries today. This resets 3.3 hours from now (00:00 UTC).

File Format
Your submission should be in CSV format. You can upload this in a zip/gz/rar/7z archive if you prefer.

# of Predictions
We expect the solution file to have 42,157 predictions. The file should have a header row. Please see the sample submission file on the data page for an example of a valid submission.

Leaderboard

1. Tim Dettmers
2. Duffman & Pascal
3. David
4. Jack Shih Tzu
5. no_name
6. Deligne
7. Razor
8. LeastSquares
9. YaTa
10. Anonymous

6. Results

- Huge part of the effort was pre-processing and trying various different models to see which model gives the best RMSE.

- Used Python and data mining libraries.

- Used Custom Java programs and `STATISTICA`
  - Developed custom Java programs for pre and post processing.
  - Normalized predicted class labels `STATISTICA` to sum to a desired value , e.g., 1.0 for sentiment and when labels.

- Built several models.
  - Models based on Ridge Regressor have performed extremely well.

- Uploaded many predictions onto *Kaggle* website, about 35 (1% of all participant uploads).

6. Results, contd.

Used several algorithms – Benchmark(all zeros: RMSE ~ 0.31597)

- Regressors (continuous, clip [0,1])
  - Ridge Regressor (sklearn) (**RMSE ~ 0.1568** ← best of all models)
  - SGD (Stochastic Gradient Descent) Regressor (sklearn)
- Classifiers (nominal, RMSE ~ 0.26)
  - Predict_proba
  - MultinomialNB (cs780 hw2, sklearn)
  - KNN
- Ensemble methods, Regression (Using *STATISTICA*)
  - Boosted Trees (RMSE ~ 0.22)
  - Random Forests (RMSE ~ 0.21 and ~ 0.19**)

*** With new additional features introduced based on the synonyms for the class labels*

6. Results, contd.

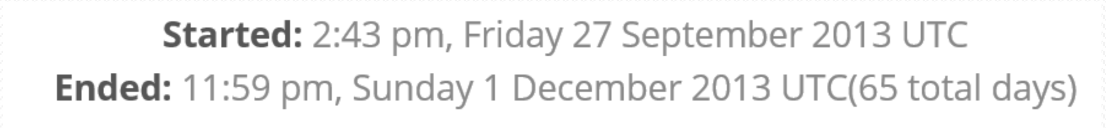
What we learned from the *Kaggle* online discussion forum:

- Simple stacking: feed the output of a first level model to use it as features to the 2nd level model

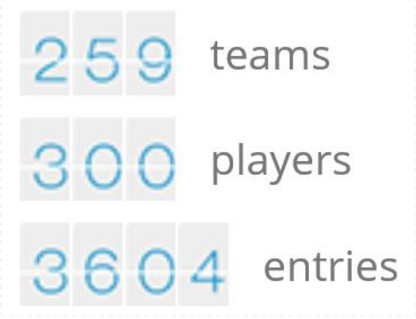
- Add more features: POS tagging, sentiment dictionary, etc. (1st ranked fellow had ~1.9M features)

6. Results, contd.

- Competition started about two months ago. Some teams had 100+ submissions.



Started: 2:43 pm, Friday 27 September 2013 UTC
Ended: 11:59 pm, Sunday 1 December 2013 UTC(65 total days)



259 teams
300 players
3604 entries

- Final Numbers from *Kaggle* website:

- We entered just two weeks ago and are glad to report that we made it to the top 25% of the leader board by the time the competition finished on Dec. 1st 2013.

TOP 25%
62nd/259

References

- *NLTK (Natural Language Toolkit), http://nltk.org/*
- *Scikit-learn Machine Learning in Python, http://scikit-learn.org/stable/*
- *Pandas Python Data Analysis Library, http://pandas.pydata.org/*
- *StatSoft's STATISTICA 12 (64-bit edition) Academic License, http://www.statsoft.com/*

Thank you!

&

Questions?