FLIGHT DELAY PREDICTION

Stanford University

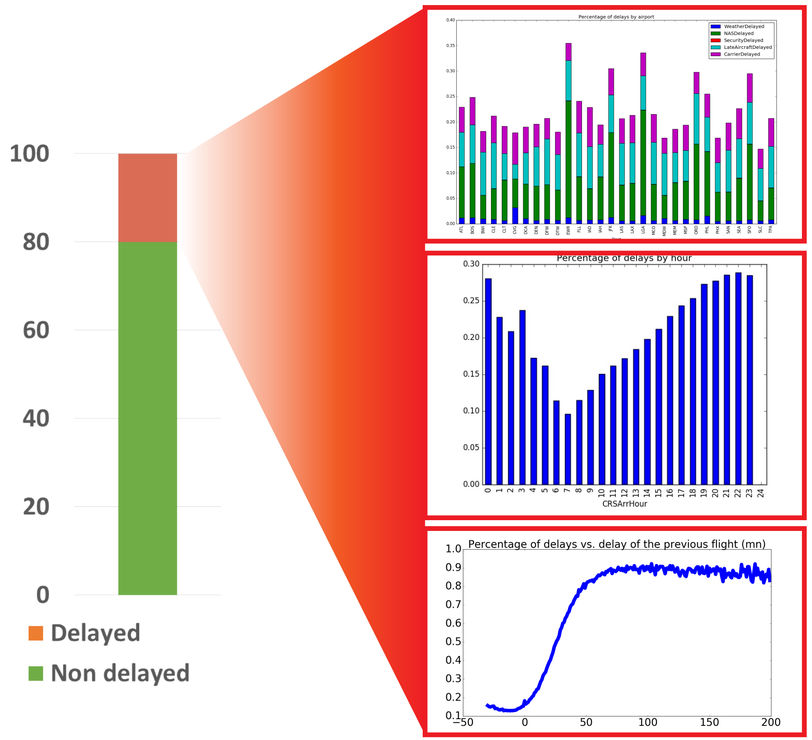
PROJECT PRESENTATION

- **DATASETS:** our main dataset includes every domestic flight from 2001 to 2008, with scheduled times, actual times, airports (depature/arrival), delay (if any) and cause of delay, plane identifier

	Cause of delay	Datasets we used
Delayed Flight	Weather	▪ Historic weather data at every airport
	National Airspace system	▪ Holidays / major travel days data
	Carrier	▪ Airlines statistics computed from the flights dataset / FAA
	Security	▪ Passengers numbers per airport (congestion) ▪ Terrorist attacks around the world with US fatalities
	Aircraft arriving late	▪ Type of plane and characteristics (seats, engines) ▪ Previous flight data

- **GOAL :** to predict if a flight will be delayed at arrival or not
- **CHALLENGES :**
  - hard to train a model to capture the delay of previous flights
  - imbalanced dataset between delayed and non delayed examples

EXPLORATORY DATA ANALYSIS



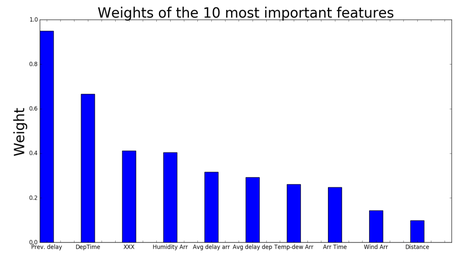
FEATURES

- **FEATURE ENGINEERING**

Description of some features we added	W	NAS	C	S	AAL
▪ **attack_killed_americans**: 1 if a terrorist attack killed americans somewhere in the world in the past 2 weeks	W	NAS	C	S	AAL
▪ **attack_homeground**: 1 if a major attack took place on the US soil in the past 2 weeks	W	NAS	C	S	AAL
▪ **is_holiday**: 1 if the flight took place on a holiday or 2 days before or after	W	NAS	C	S	AAL
▪ **previous_flight_delay**: the value of the arrival delay of the plane at its previous flight	W	NAS	C	S	AAL
▪ **avg_arr_delay_at_dep_airport**: the average arrival delay at the departure airport in the hour preceding the scheduled departure time	W	NAS	C	S	AAL
▪ **Departure/arrival airports**, **airplane make and model** and other textual data (encoded into integers)	W	NAS	C	S	AAL

- **FEATURE ANALYSIS**

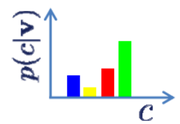
  - Weights for the logistic regression on normalized features
  -Several features of force memory are present

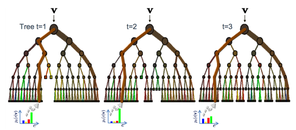


ALGORITHMS

- **RANDOM FOREST**

$$p(c|v) = \frac{1}{T}\sum_{t=1}^{T} p_t(c|v)$$





- **GAUSSIAN NAIVE BAYES**

  - Model:

$$p(x_1,\ldots,x_n|y) = \prod_{j=1}^{n} p(x_j|y)$$

$$p(y=1|x) = \frac{\prod_j p(x_j|y=1)p(y=1)}{\prod_j p(x_j|y=0)p(y=0) + \prod_j p(x_j|y=1)p(y=1)}$$

  - Training:

$$\phi_{j|y=1} = \frac{\sum_{i=1}^{m} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\}}{\sum_{i=1}^{m} 1\{y^{(i)} = 1\}} \qquad \phi_y = \frac{\sum_{i=1}^{m} 1\{y^{(i)} = 1\}}{m}$$

- **LOGISTIC REGRESSION**

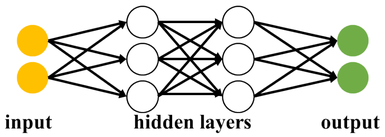
  - Model:

$$p(y=1|x;\theta) = \frac{1}{1+e^{-\theta^T x}}$$

  - Training:

$$\theta_j := \theta_j + \alpha(y^{(i)} - h_\theta(x^{(i)}))x_j^{(i)}$$

- **NEURAL NETWORK**

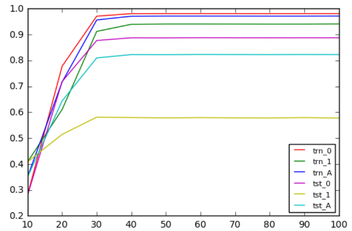


RESULTS

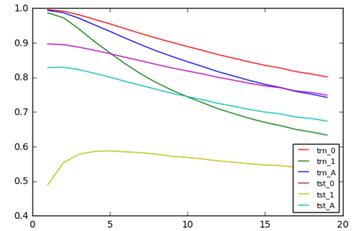
	PRECISION	RECALL	F1 -SCORE
Baseline	77%	100%	87%
	0%	0%	0%
	59%	77%	67%
Random Forest	86%	91%	89%
	64%	53%	58%
	81%	82%	82%
Gaussian Naive Bayes	82%	88%	85%
	47%	36%	40%
	74%	76%	75%
Logistic Regression	85%	90%	88%
	60%	48%	53%
	79%	81%	80%
Neural Network	93%	73%	82%

«Not delayed»
«Delayed»
Total

- **OPTIMIZATION OF THE RANDOM FOREST**

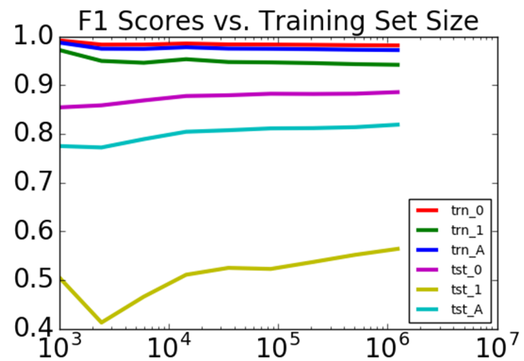


F1-score vs max_depth



F1-score vs min_samples

ANALYSIS



- **NEXT STEPS:**
  - Improve the Neural Network (better shape)
  - Increase size of the sample of training examples with data from other years
  - Add new features to capture non linearity (we still have no overfitting)