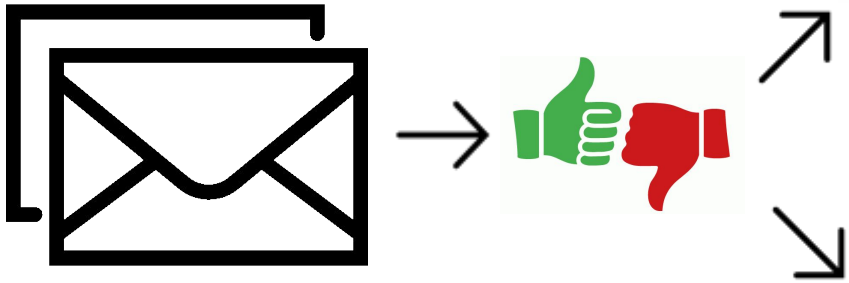
W207 Final Project
Random Acts Of Pizza

Fall 2016 | Sannat & Rubin

Problem Statement

Reddit
Message
Predict



Pizza /No-Pizza

Approach

- prepare the data
- create a classifier list
- visualizations
- accuracy report
- innovate

lather-rinse-repeat

Approach

prepare the data

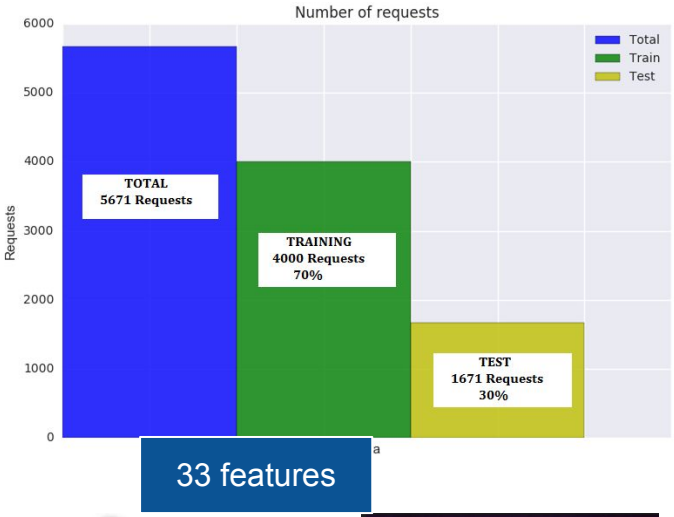
- ➔ **data exploration**
- ➔ feature engineering
- ➔ topic modeling

classifier list

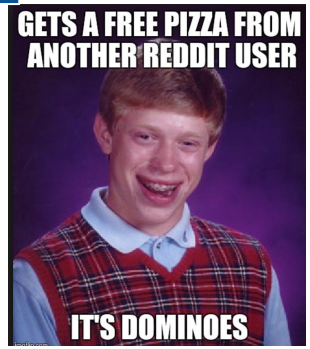
visualizations

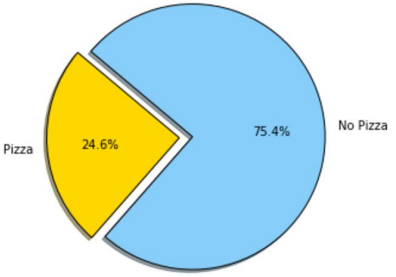
accuracy reports

innovation

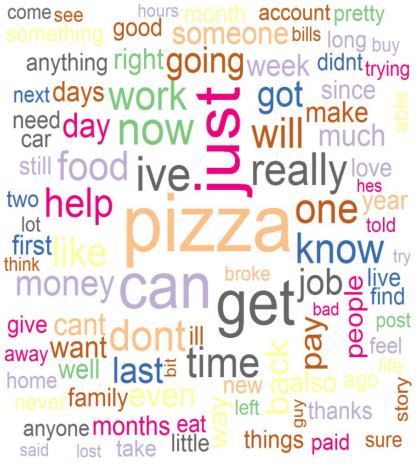


33 features





Odds: 1/3



Approach

prepare the data

- ➔ data exploration
- ➔ **feature engineering**
- ➔ topic modeling

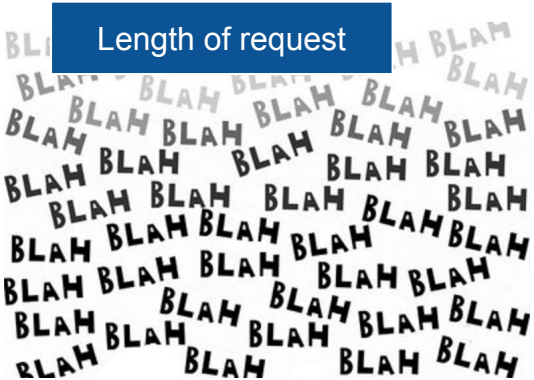
classifier list

visualizations

accuracy reports

innovation

Length of request



Evidence of need

Spelling errors in request

Time of request

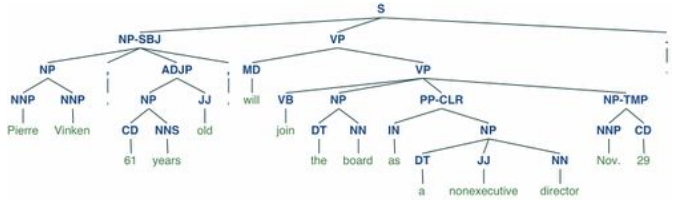
Approach

prepare the data

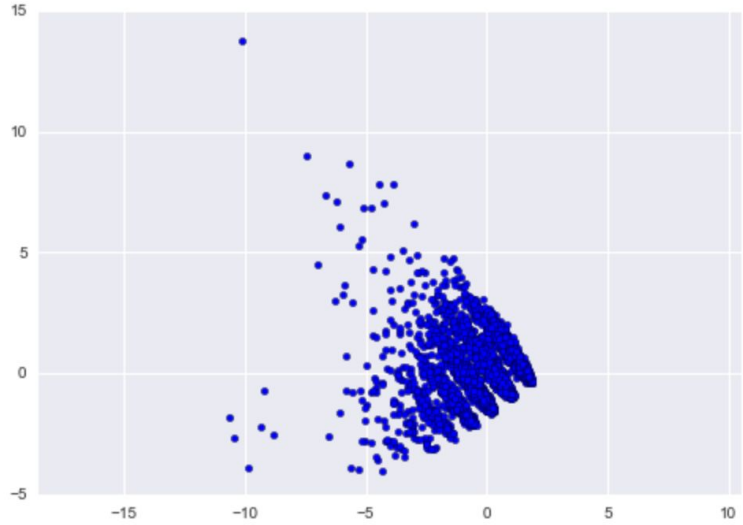
- ➔ data exploration
- ➔ feature engineering
- ➔ **topic modeling (LDA)**

classifier list
visualizations
accuracy reports
innovation

1.Using NLTK POS tagger, get nouns & adjectives for all of the requests



2. Using PCA, project TOP Nouns & Adjectives to 2d space and check for visually separable clusters



Approach

prepare the data

- ➔ data exploration
- ➔ feature engineering
- ➔ **topic modeling (LDA)**

classifier list

visualizations

accuracy reports

innovation

3. Using LDA get word distributions for the five topics

TOPIC 1

TOPIC 2

TOPIC 3

TOPIC 4

TOPIC 5

4. For each request get Word frequencies for the five topics

5. Standardize topics' word frequencies for each request using standard and robust scaler

Approach

prepare the data

- ➔ data exploration
- ➔ feature engineering
- ➔ **topic modeling (Stanford narratives)**

classifier list
visualizations
accuracy reports
innovation

1. Get lexicon for five narratives in Stanford Paper

```
money = ["money", "now", "broke", "week", "until", "time",
         "last", "day", "when", "today", "tonight", "paid", "next",
         "first", "night", "after", "tomorrow", "month", "while",
         "account", "before", "long", "Friday", "rent", "buy",
         "bank", "still", "bills", "ago", "cash", "due",
         "soon", "past", "never", "paycheck", "check", "spent",
         "years", "poor", "till", "yesterday", "morning", "dollars",
         "financial", "hour", "bill", "evening", "credit",
         "budget", "loan", "bucks", "deposit", "dollar", "current",
         "payed"]
job =["work", "job", "paycheck", "unemployment", "interview",
         "fired", "employment", "hired", "hire"]
student = ["college", "student", "school", "roommate",
         "studying", "university", "finals", "semester",
         "class", "study", "project", "dorm", "tuition"]
family =["family", "mom", "wife", "parents", "mother", "husband",
          "dad", "son", "daughter", "father", "parent",
          "mum"]
craving = ["friend", "girlfriend", "craving", "birthday",
         "boyfriend", "celebrate", "party", "game", "games",
         "movie", "date", "drunk", "beer", "celebrating", "invited",
         "drinks", "crave", "wasted", "invite"]
```

2. For each request get word frequencies for the five narratives

3. Standardize narratives' word frequencies for each request using standard and robust Scaler

Approach

prepare the data
classifier list
visualizations
accuracy reports
innovation

```
# Ada Boost Classifier
def adaBoostClassifier(x_train, y_train, x_test, y_test, print_str):
    global fu
    global ada
    function_start_time=time.time()
    ada_clf =
        Decisi                                                    0,
        algori
    )
    ada_clf.fi
    y_pred_ada
    accuracy = accuracy_score(y_test, y_pred_ada)*100.0

    # add cla                                    assifierInfo_list
    classifierInfo_list.append( classifierInfo( ada_clf, accuracy) )

    print 'Acc                                    ;(accuracy)
    printElapsedTime("AdaBoostClassifier")
```

LogisticRegression

RandomForestClassifier GridSearchCV

ExtraTreesClassifier GridSearchCV

VotingClassifier

SVC

GradientBoostingClassifier

KNeighborsClassifier GridSearchCV

AdaBoostClassifier

BaggingClassifier GridSearchCV

Approach

prepare the data
classifier list
visualizations
accuracy reports
innovation

- classification plot
  - PCA to project features onto 2 dims
- topic clusters plot for top 200 nouns
  - PCA to project features onto 2 dims

Approach

prepare the data
classifier list
visualizations
accuracy reports
innovation

- aggregation of results from processing the data sets through the classifier list
- creates benchmark values

Approach

prepare the data
classifier list
visualizations
accuracy reports
innovation

- 2 data sets required a repeatable process
- this lead to accuracy reporting
- the classifier list suggested ensembling (2 ways)
- PCA plots suggested GMM Clusters
- CNN: why not?

Implement

- split the data
- apply the classifiers
- ROC Curve
  - Receiver Operating Characteristic
- visualization
  - using PCA for data generation
- accuracy report

Implement

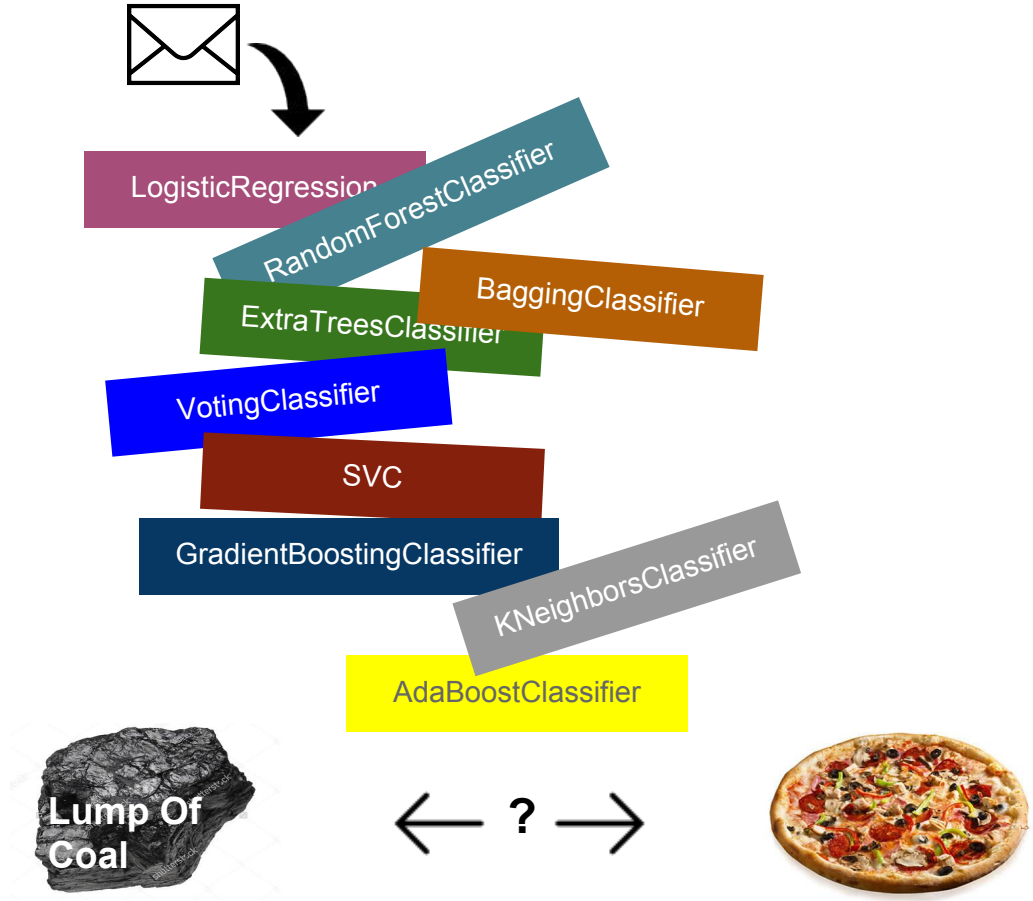
split the data
apply classifiers
ROC Curve
visualization
accuracy report

```
np.random.seed(0)
def randomizeDataArrays():
    # create randomized arrays for test, dev and train
    X = RAOP_updated_stanford.values
    X_sd = RAOP_updated_stanford.values

    global raop_test, raop_train
    global raop_test_sd, raop_train_sd
    global raop_test_labels,raop_train_labels
    global raop_test_labels_sd,raop_train_labels_sd
    global Xtr_s, Xte_s, Xtr_r, Xte_r
    global Xtr_s_sd, Xte_s_sd, Xtr_r_sd, Xte_r_sd

    raop_test = X[4040:]
    raop_train = X[:4040]
    raop_test_sd = X_sd[4040:]
    raop_train_sd = X_sd[:4040]

    raop_test_labels = np.array(raop_test[:,18], dtype=int)
    raop_train_labels = np.array(raop_train[:,18], dtype=int)
    raop_test_labels_sd = np.array(raop_test_sd[:,18], dtype=int)
    raop_train_labels_sd = np.array(raop_train_sd[:,18], dtype=int)

    raop_test = np.delete(raop_test , [18], axis=1)
    raop_train = np.delete(raop_train, [18], axis=1)
    raop_test_sd = np.delete(raop_test_sd , [18], axis=1)
    raop_train_sd = np.delete(raop_train_sd, [18], axis=1)

    # Scale data
    # Standard Scaler
    standard_scaler = preprocessing.StandardScaler()
```

- encapsulated in re-runnable function
- randomly shuffles the data
- 70/30 train/test ratio
- creates both data sets

Implement

split the data
apply classifiers 1
ROC Curve
visualization
accuracy report



Tim Althoff, Cristian Danescu-Niculescu-Mizil, Dan Jurafsky. *How to Ask for a Favor: A Case Study on the Success of Altruistic Requests*, Proceedings of ICWSM, 2014

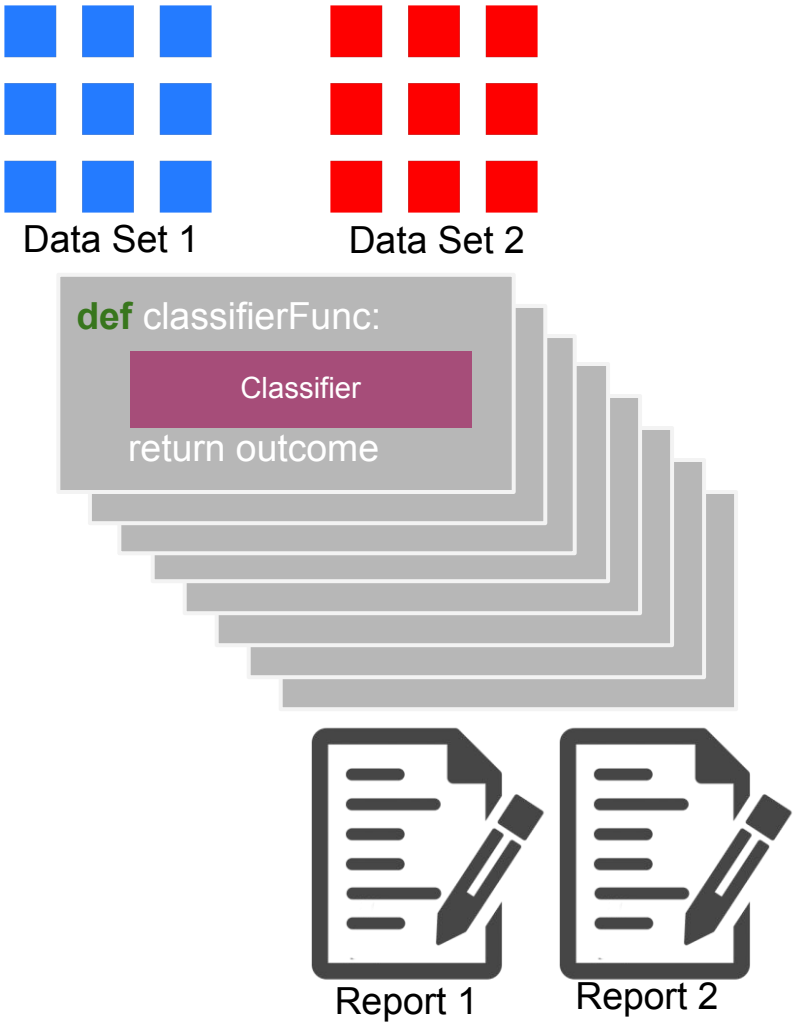
Implement

split the data

apply classifiers 2

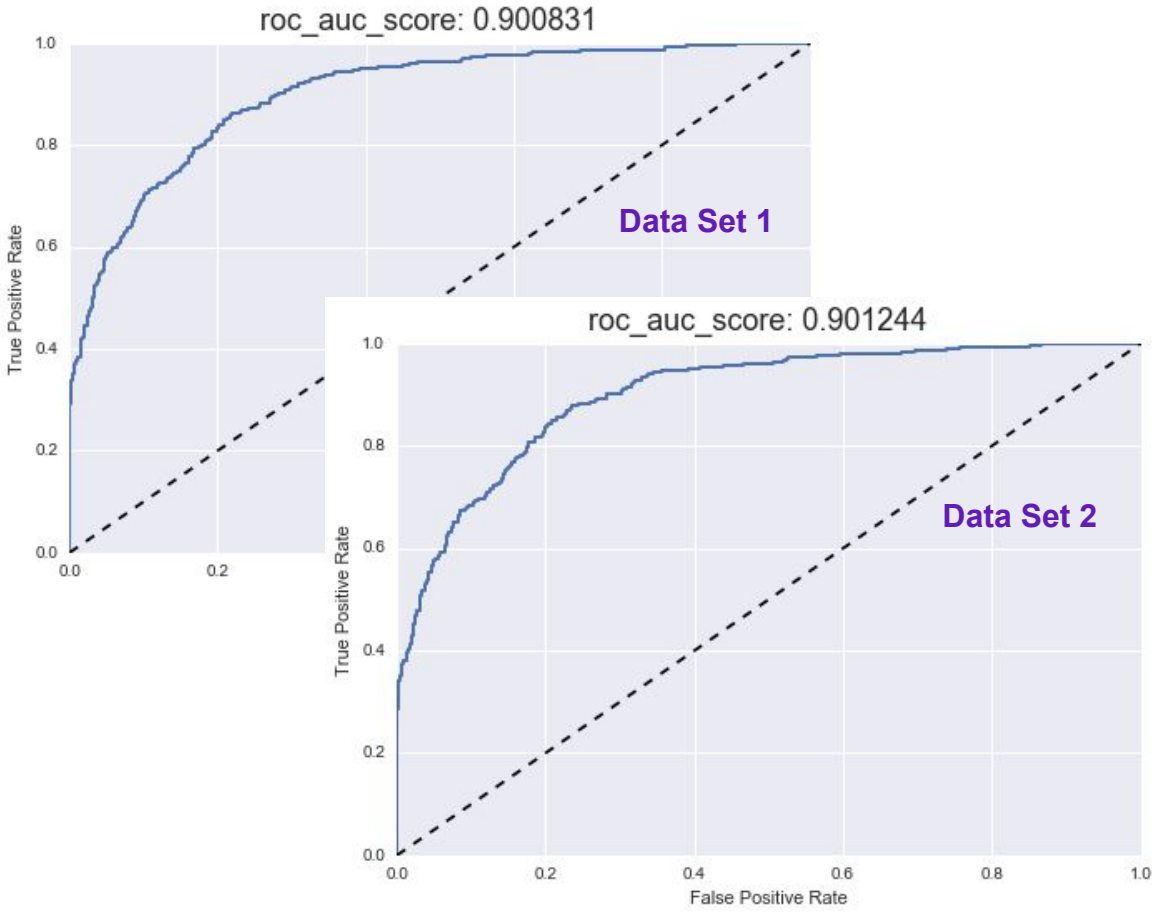
ROC Curve visualization

accuracy report



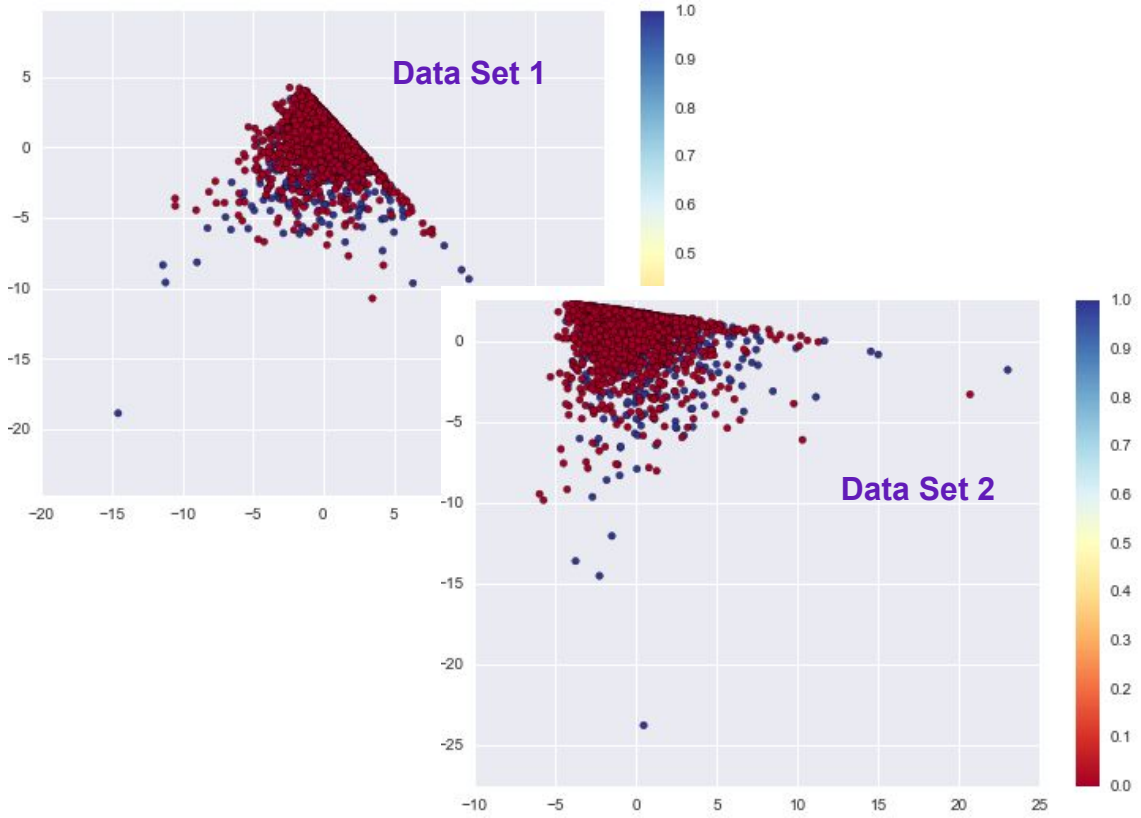
Implement

split the data
apply classifiers
ROC Curve
visualization
accuracy report



Implement

split the data
apply classifiers
ROC Curve
visualization
accuracy report



Visualization using PCA for the generated data with Unsupervised topic modeling (Data Set 1) and with added narratives (Data Set 2)

Implement

split the data
apply classifiers
ROC Curve
visualization
accuracy report

Data Set 1

Classifier	Note	Accuracy
AdaBoostClassifier		85.530%
GradientBoostingClassifier		85.469%
BaggingClassifier	n_estimators=1000, max_samples=150	85.285%
RandomForestClassifier	n_estimators=1000, max_leaf_nodes=32, criterion='gini'	85.101%
LogisticRegression		84.488%
SVC		84.120%
ExtraTreesClassifier	n_ ma crit	
KNeighborsClassifier	n_	
LogisticRegression	Ro	
Baseline KNN	me	

Data Set 2

Classifier	Note	Accuracy
GradientBoostingClassifier		86.021%
AdaBoostClassifier		85.592%
BaggingClassifier	n_estimators=500, max_samples=200	85.469%
RandomForestClassifier	n_estimators=3000, max_leaf_nodes=40, criterion='gini'	84.917%
LogisticRegression		84.672%
SVC		84.181%
ExtraTreesClassifier	n_estimators=2000, max_leaf_nodes=40, criterion='gini'	82.955%
KNeighborsClassifier	n_neighbors=2	81.790%
LogisticRegression	Robust Scaling	74.617%
Baseline KNN	message length only	64.700%

Innovate

- Convolutional Neural Net (CNN)
- ensembling
  - using pre-fitted classifiers
- ensembling
  - using non-fitted classiffiers
- GMM Clustering

Innovate

convolutional net
ensembling, fitted ensembling
gaussian mixture

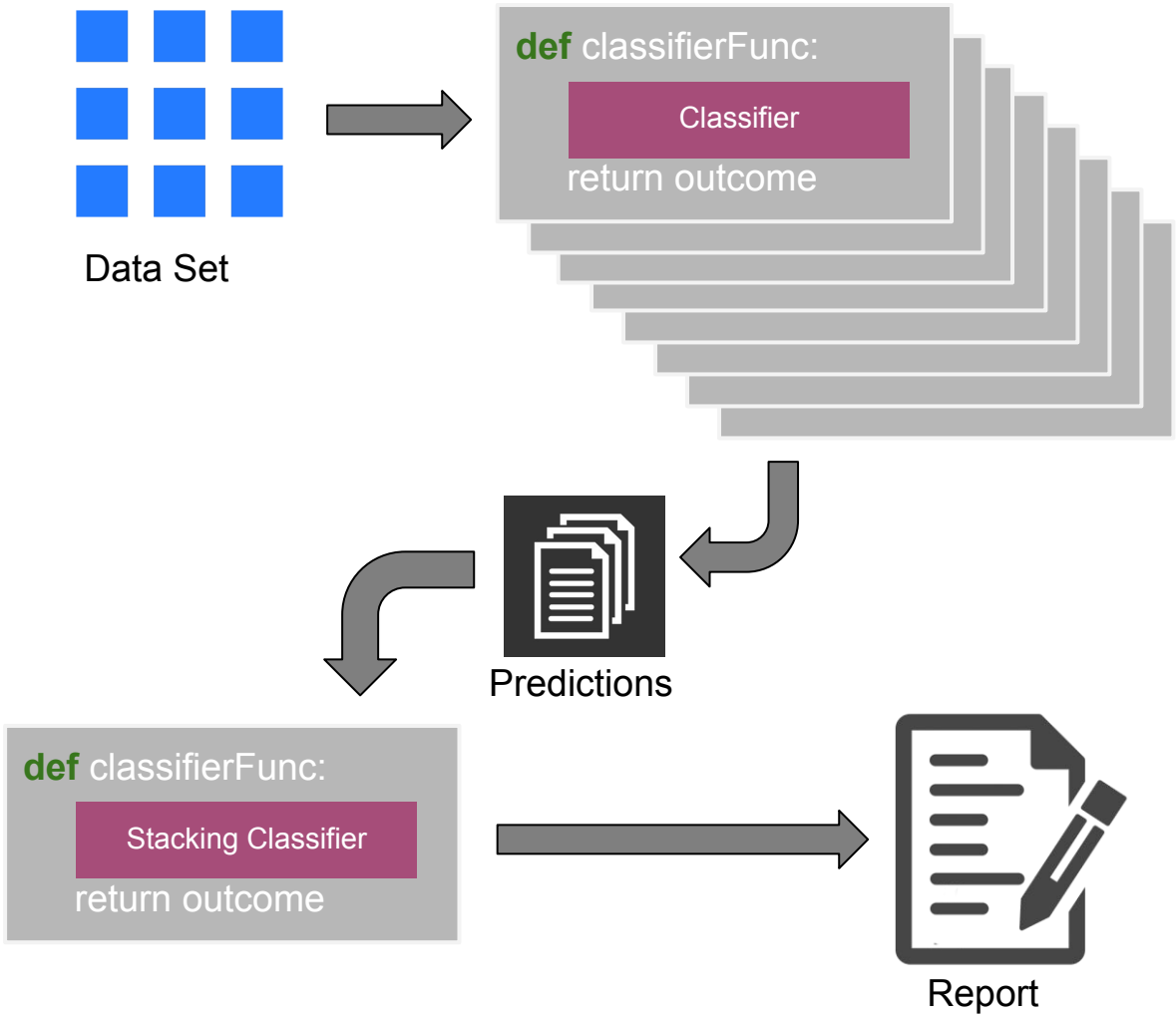
```
# immature CNN attempt, limited to 5 epochs
cnnProcess(Xtr_s, raop_train_labels, Xte_s, raop_test_labels)
```

```
1) accuracy = 53.0963%
2) accuracy = 53.0963%
3) accuracy = 53.0963%
4) accuracy = 53.0963%
5) accuracy = 53.0963%
```

Barking up the right tree, but underestimated the effort

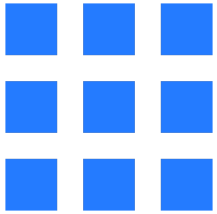
Innovate

convolutional net
ensembling,
fitted
ensembling
gaussian mixture



Innovate

convolutional net
ensembling, fitted
ensembling
gaussian mixture



Data Set

```
base_models = [
    RandomForestClassifier( ),
    RandomForestClassifier( ),
    ExtraTreesClassifier( ),
    ExtraTreesClassifier( ),
    GradientBoostingClassifier( )
]
For m in base_models:
    m.fit( x_data, y_data )
    m.calculate_accuracy( )
```

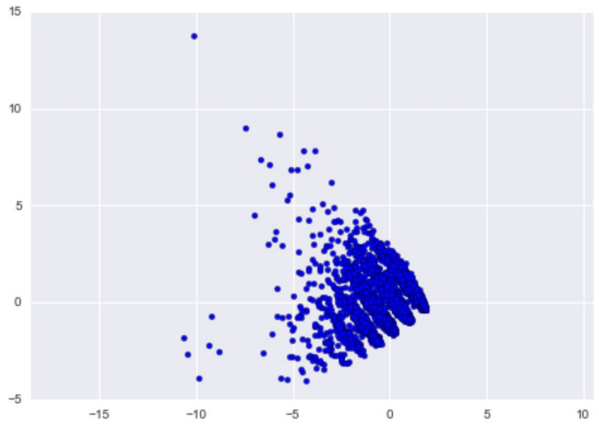
Predictions

```
def classifierFunc:
    Stacking Classifier
    return outcome
```

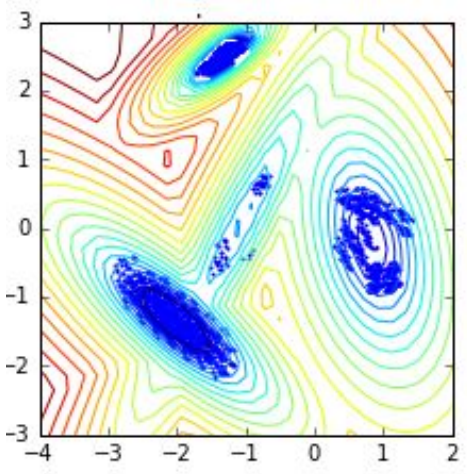
Report

Innovate

convolutional net ensembling, fitted ensembling **gaussian mixture**



this reminded us of



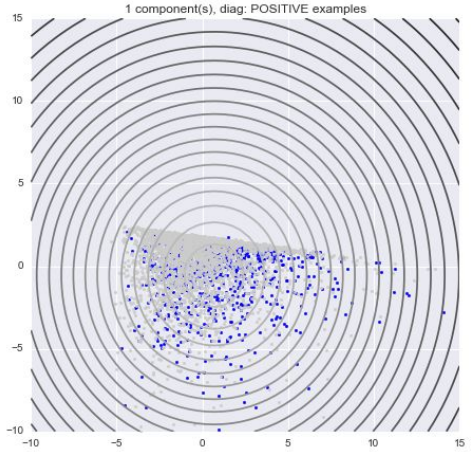
Process:

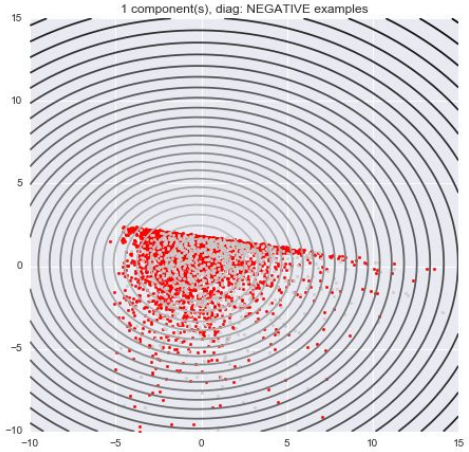
1. project features to 2d with PCA
2. brute-force covariance matrix types and n_component values
3. Compare maximum probability for each element with label and compute accuracy
4. compile report

Innovate

convolutional net ensembling, fitted ensembling **gaussian mixture**

Best Probability	Covariance Matrix Type	Number of Components	Accuracy
negative	diag	1	70.39%
negative	full	1	69.71%
negative	tied	1	69.71%
negative	spherical	1	69.16%
negative	tied	2	66.34%
negative	tied	3	65.36%
negative	full	3	62.60%
negative	full	2	61.13%
negative	spherical	3	60.94%
negative	diag	2	60.58%
negative	spherical	2	60.09%





Conclusion

Best Result:
GradientBoostingClassifier, Accuracy **86.021**%

Best Ensembling Result:
Accuracy **85.469**%

Baseline (message length only):
Accuracy **64.700**%

Gaussian Mixture Model Clustering:
Accuracy **70.390**%

CNN Result:
Accuracy **53.096**% (lower than baseline!)