GA Data Science Project

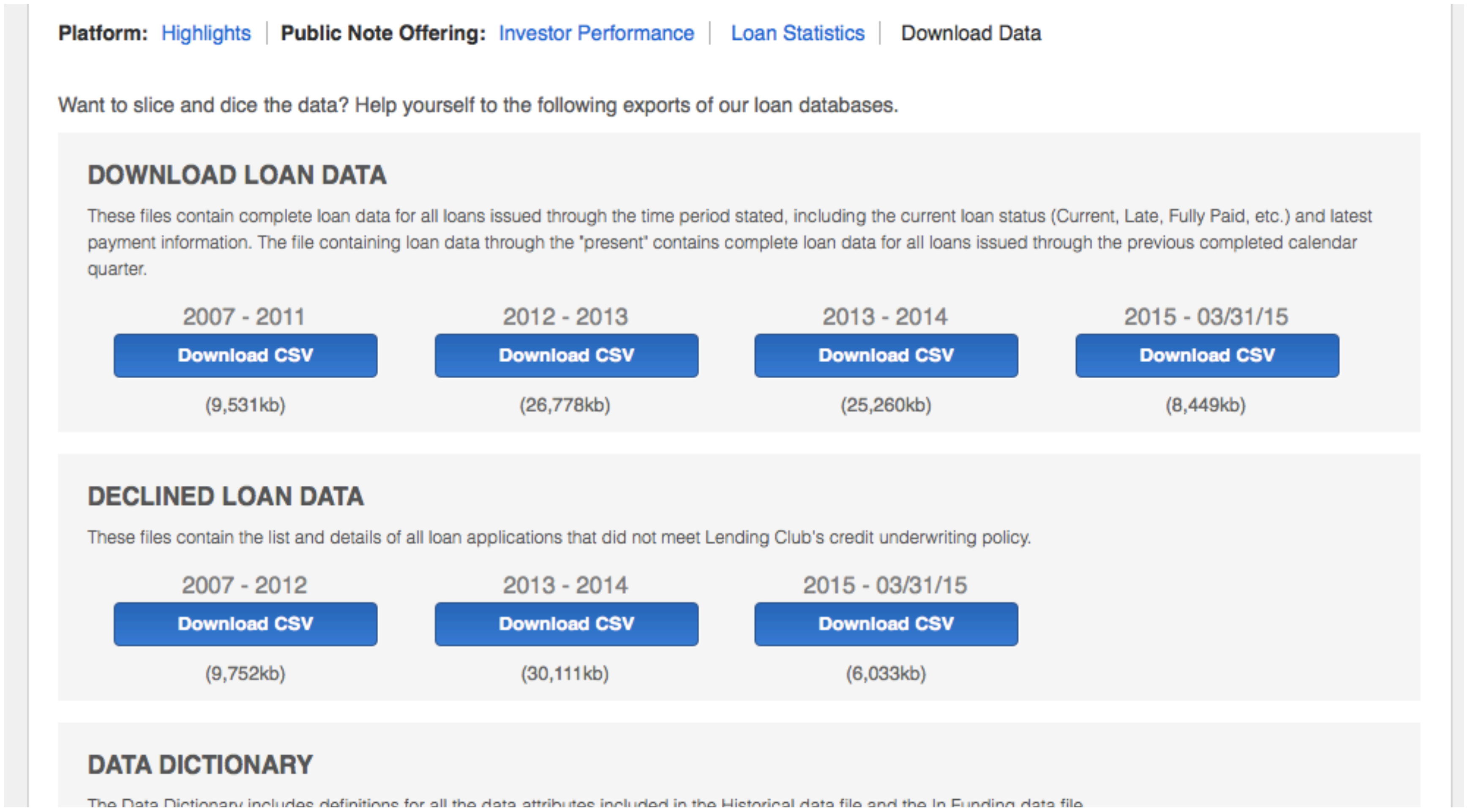
Understanding “Lending Club” loan data performance.

Lending Club Background

Lending Club is a "Market Place Lender" which allows an individual to apply for a personal loan to be filled by the market, as opposed to a bank or other lender.   They have grown in success and to date have closed over $3Bn in loans.

Lending Club has made loan and performance information freely available on their website.   The data is available on the Lending Club website. I manually downloaded the CSV files and stored them locally.   I am using only one subset of the total lending club data.

Lending Club (Continued)



Platform: Highlights | **Public Note Offering:** Investor Performance | Loan Statistics | Download Data

Want to slice and dice the data? Help yourself to the following exports of our loan databases.

DOWNLOAD LOAN DATA

These files contain complete loan data for all loans issued through the time period stated, including the current loan status (Current, Late, Fully Paid, etc.) and latest payment information. The file containing loan data through the 'present' contains complete loan data for all loans issued through the previous completed calendar quarter.

2007 - 2011	2012 - 2013	2013 - 2014	2015 - 03/31/15
Download CSV	Download CSV	Download CSV	Download CSV
(9,531kb)	(26,778kb)	(25,260kb)	(8,449kb)

DECLINED LOAN DATA

These files contain the list and details of all loan applications that did not meet Lending Club's credit underwriting policy.

2007 - 2012	2013 - 2014	2015 - 03/31/15
Download CSV	Download CSV	Download CSV
(9,752kb)	(30,111kb)	(6,033kb)

DATA DICTIONARY

The Data Dictionary includes definitions for all the data attributes included in the Historical data file and the In Funding data file

Objectives

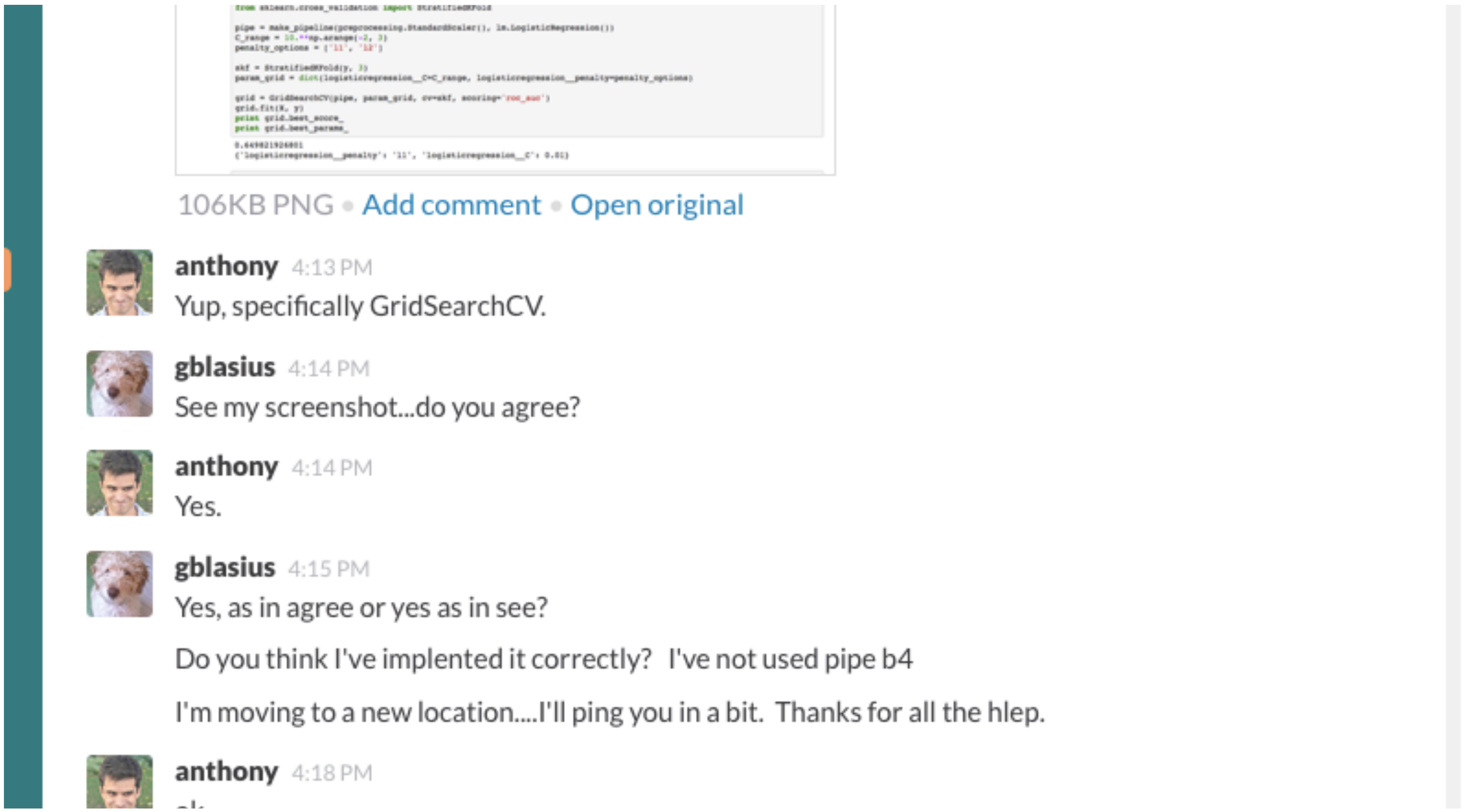
My objectives in this project are threefold:

i) Learn how to view, clean and manipulate datasets in Python;

ii) Develop better intuition for fitting estimators to the data;

iii) Start to develop a better way to predict creditworthiness of borrowers beyond the current credit metrics (eg, FICO). For example, can we parse their loan purpose comments to develop a predictor of default?

Anthony (& Josh) were a great help!! (one page out of hundreds).



```
from sklearn.cross_validation import StratifiedKFold

pipe = make_pipeline(preprocessing.StandardScaler(), lm.LogisticRegression())
C_range = 10.**np.arange(-2, 3)
penalty_options = ['l1', 'l2']

skf = StratifiedKFold(y, 3)
param_grid = dict(logisticregression__C=C_range, logisticregression__penalty=penalty_options)

grid = GridSearchCV(pipe, param_grid, cv=skf, scoring='roc_auc')
grid.fit(X, y)
print grid.best_score_
print grid.best_params_

0.649821926801
{'logisticregression__penalty': 'l1', 'logisticregression__C': 0.01}
```

106KB PNG • Add comment • Open original

anthony 4:13 PM
Yup, specifically GridSearchCV.

gblasius 4:14 PM
See my screenshot...do you agree?

anthony 4:14 PM
Yes.

gblasius 4:15 PM
Yes, as in agree or yes as in see?

Do you think I've implented it correctly? I've not used pipe b4

I'm moving to a new location....I'll ping you in a bit. Thanks for all the hlep.

anthony 4:18 PM

Steps in my journey

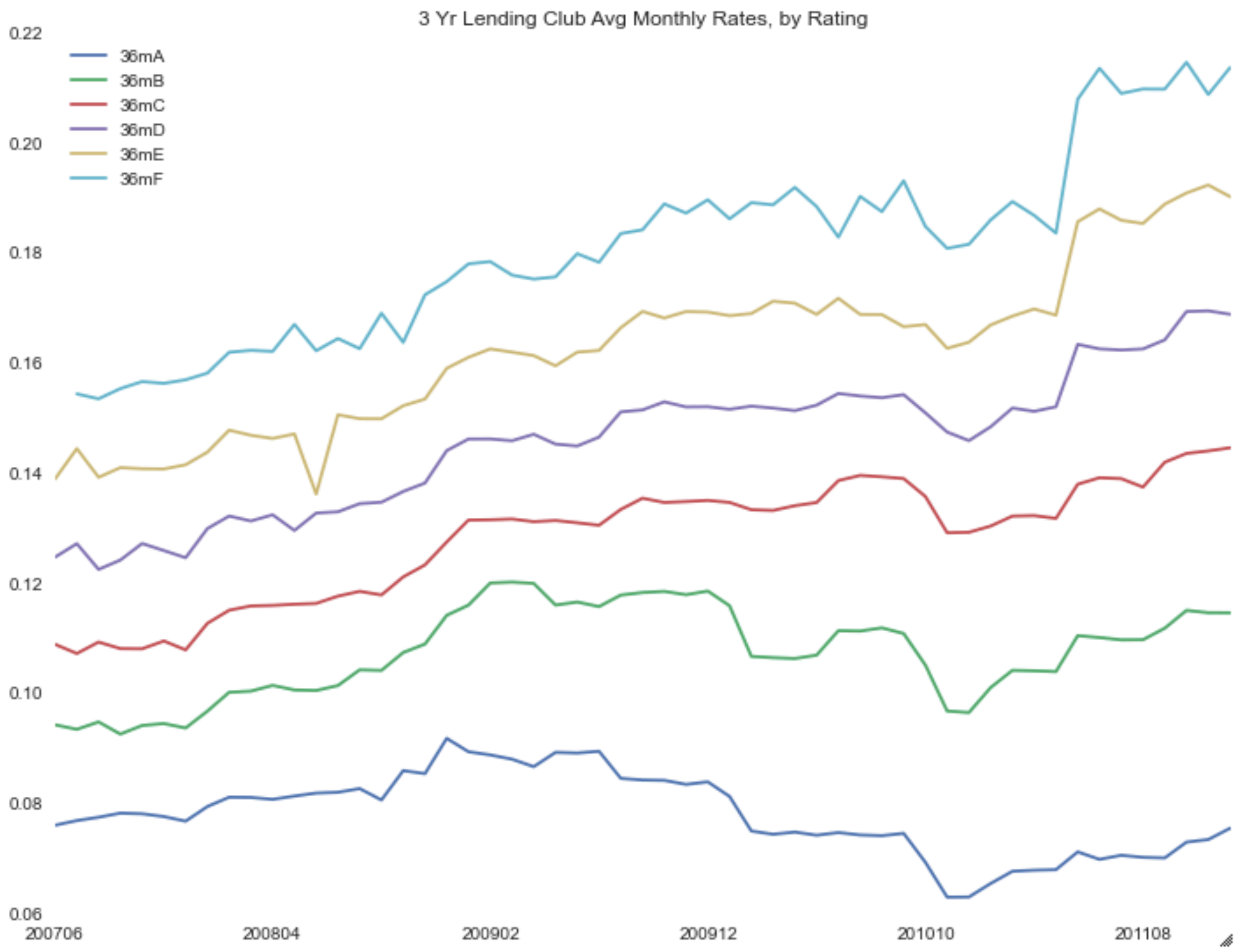
- 1) Read in data, and clean;
- 2) Visualize data to gain an intuition;
- 3) Create calculated features;
- 4) Try to fit estimators to data and evaluate

1) As predicted took up much of my time. There are still lots of things I'd like to do with the data. I only started work on the estimators this week. Lots more to do!

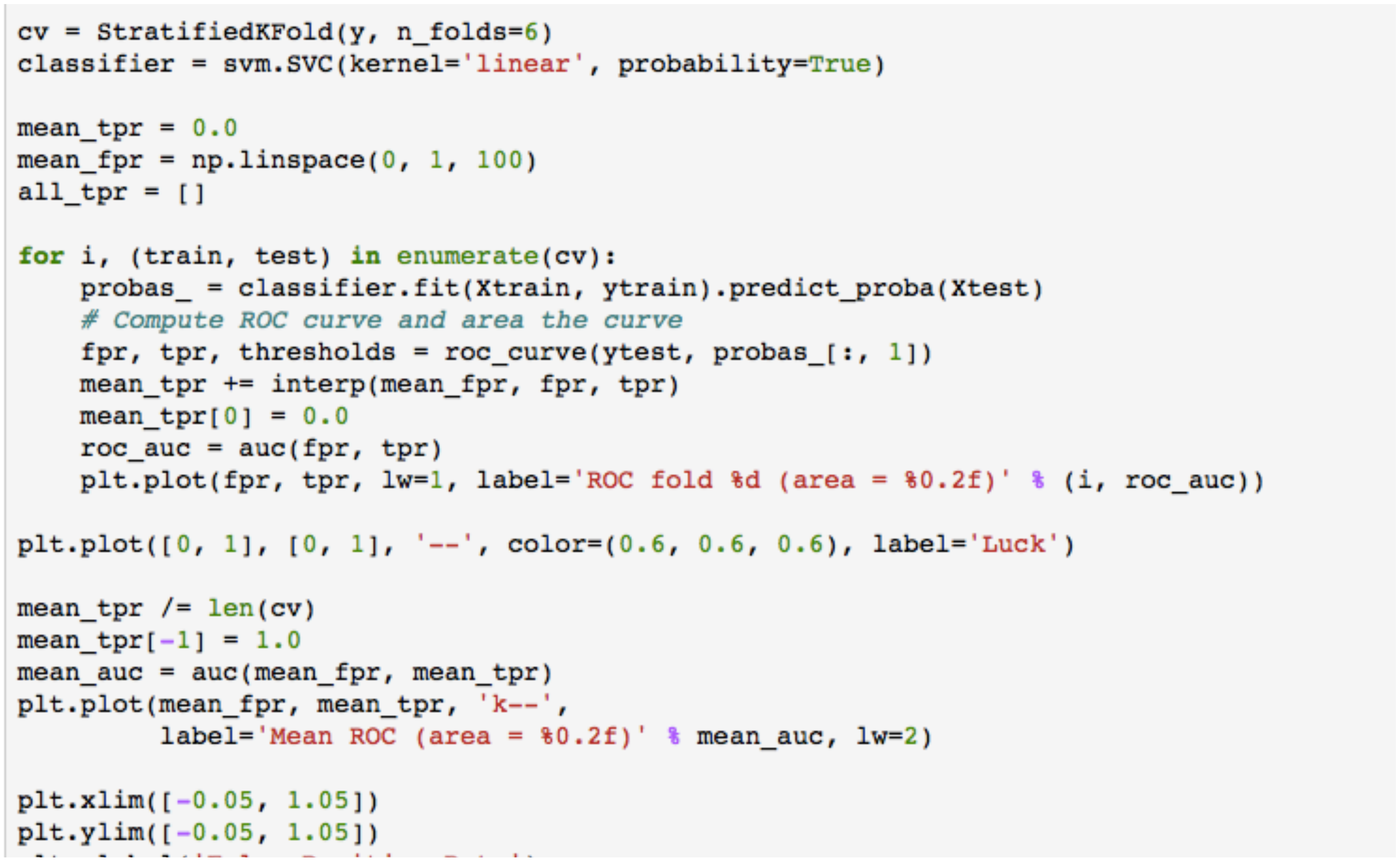
Read in data and clean.

- The .csv file had some imperfections (such as unexpected blank lines in the middle of the dataset), which consumed a lot of time.
- All NaNs are not created equal; some needed to be eliminated, others had meaning. For example, “Months since last delinquency” feature was left empty if borrower was not ever delinquent.

Visualize Data
(Lending Club Loan Rates by Borrower Rating)

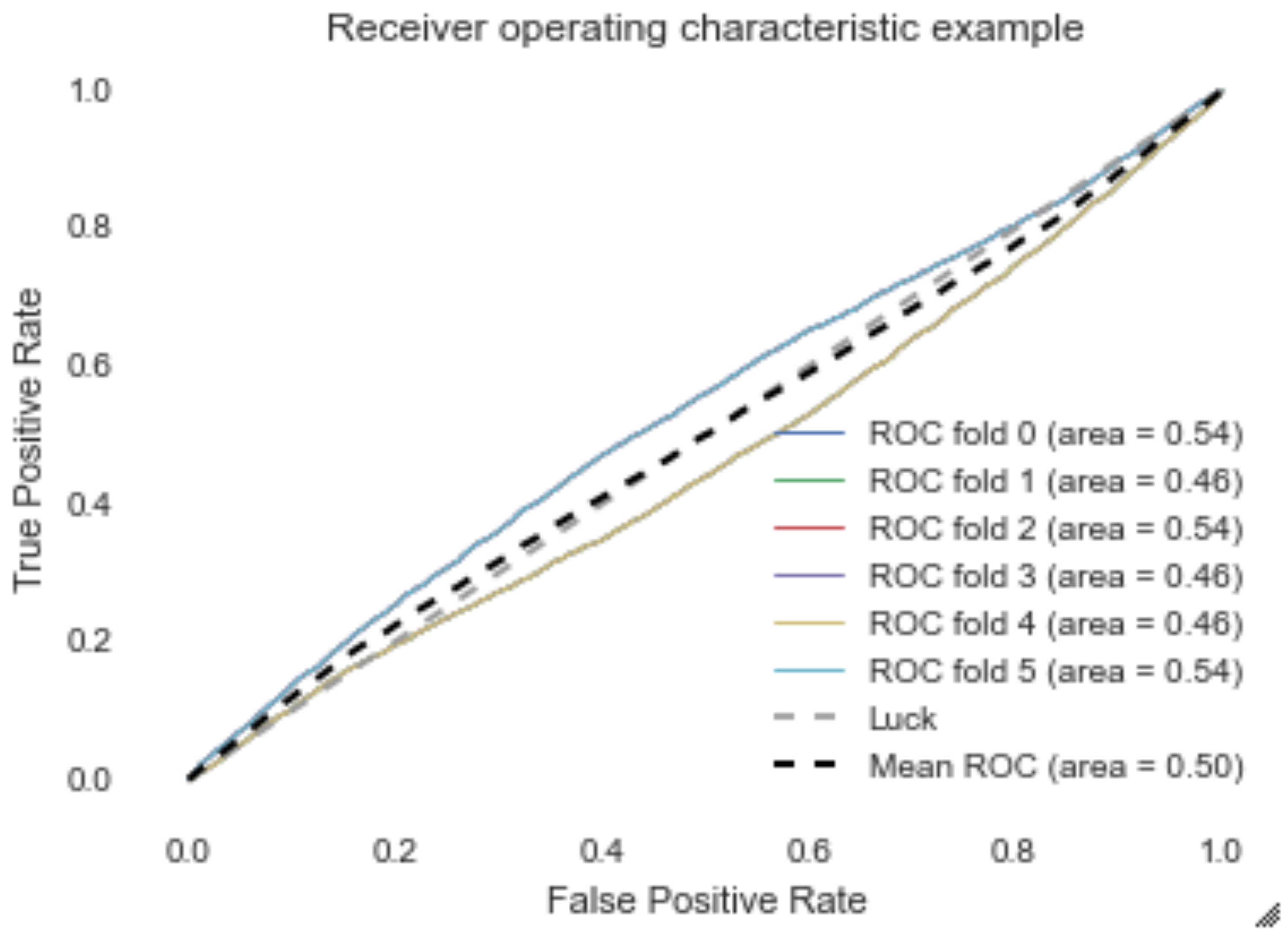


Learn how to fit and tune estimators!



```
cv = StratifiedKFold(y, n_folds=6)
classifier = svm.SVC(kernel='linear', probability=True)

mean_tpr = 0.0
mean_fpr = np.linspace(0, 1, 100)
all_tpr = []

for i, (train, test) in enumerate(cv):
    probas_ = classifier.fit(Xtrain, ytrain).predict_proba(Xtest)
    # Compute ROC curve and area the curve
    fpr, tpr, thresholds = roc_curve(ytest, probas_[:, 1])
    mean_tpr += interp(mean_fpr, fpr, tpr)
    mean_tpr[0] = 0.0
    roc_auc = auc(fpr, tpr)
    plt.plot(fpr, tpr, lw=1, label='ROC fold %d (area = %0.2f)' % (i, roc_auc))

plt.plot([0, 1], [0, 1], '--', color=(0.6, 0.6, 0.6), label='Luck')

mean_tpr /= len(cv)
mean_tpr[-1] = 1.0
mean_auc = auc(mean_fpr, mean_tpr)
plt.plot(mean_fpr, mean_tpr, 'k--',
         label='Mean ROC (area = %0.2f)' % mean_auc, lw=2)

plt.xlim([-0.05, 1.05])
plt.ylim([-0.05, 1.05])
```

But lots more needs to be done…(oops).



Conclusions

- I underestimated the amount of work needed to clean data, even though it was readily sourced.
- Some of this included thinking about how to structure/use data….I'm still learning more about this.
- Lots more work needs to be done on estimators!!