LendUp Data Challenge Solution

J. Mark Ettinger
jettinger35@gmail.com

January 16, 2016

1 Histogram and summary statistics

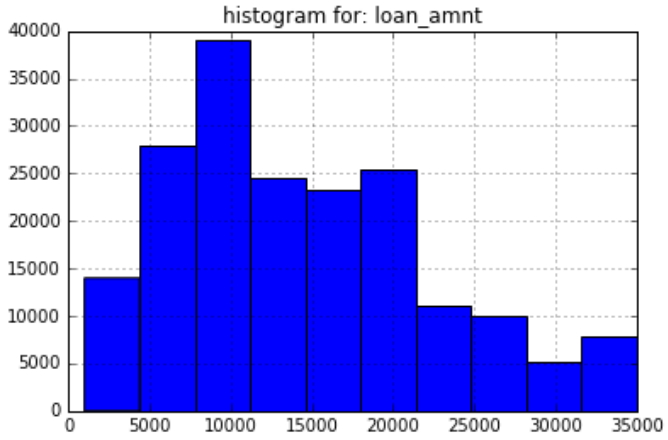

Figure 1: Sample histogram

```
count    188123.000000
mean      14354.545962
std        8115.066458
min        1000.000000
25%        8000.000000
50%       12175.000000
75%       20000.000000
```

```
max       35000.000000
Name: loan_amnt, dtype: float64
```

2 Best parameters for Question #2

I used logistic regression with l1 and l2 regularization, for each case trying the scikit-learn default values of 10 values for C. Using cross validation, l2 regularization with $C = 10^4$ produced the maximal CV accuracy = 0.99941.

3 Code for Question #1

The arguments for my function are a dataframe and a column name rather than a list of numerical values. Hopefully this is acceptable.

```
import pandas as pd
import matplotlib.pyplot as plt

def summarizeAndHistogram(dataframe, columnName = 'loan_amnt'):
    print dataframe[columnName].describe()

    dataframe[columnName].hist()
    plt.title("histogram for: " + columnName)
    plt.show()

dataFile = "/Users/mettinger/Data/lendUp/loanData.csv"
df = pd.read_csv(dataFile)
summarizeAndHistogram(df)
```

4 Code for Question #2

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import Imputer
from sklearn.linear_model import LogisticRegressionCV
```

```
#%% READ RAW DATA

dataFile = "/Users/mettinger/Data/lendUp/loanData.csv"
df = pd.read_csv(dataFile)

#%% PREPROCESS THE DATA

# drop some unhelpful features
df1 = df.dropna(subset = ['loan_status'])
dropColumns = ['id','member_id','url','desc','policy_code']
df2 = df1.drop(dropColumns, axis = 1)

# drop categorical columns for the moment and return to them later
dtypes = df2.dtypes
categoricalCols = [i for i in df2.columns if dtypes[i] == 'object']
categoricalCols.remove('loan_status')
df2 = df2.drop(categoricalCols, axis = 1)

# drop features with "too much" missing data
naDropTreshold = 90000
df3 = df2.dropna(axis=1, thresh=naDropTreshold)

# prepare data for scikit-learn models
y = [1 if i == "Fully Paid" else 0 for i in df3['loan_status'].values]
X = df3.ix[:, df3.columns != 'loan_status'].values

# impute values for missing data
imp = Imputer(strategy='mean', axis=0)
X = imp.fit_transform(X)

#%%

# fit and assess two logistic regression models, using l1 and l2 regularization,
#    using CV to find good hyperparameters
model_l1 = LogisticRegressionCV(penalty='l1', solver='liblinear')
model_l1.fit(X,y)
```

```
scores = np.mean(model_l1.scores_[1], axis=0)
score_best_l1 = max(scores)
C_best_l1 = model_l1.Cs_[np.argmax(scores)]

model_l2 = LogisticRegressionCV(penalty='l2')
model_l2.fit(X,y)

scores = np.mean(model_l2.scores_[1], axis=0)
score_best_l2 = max(scores)
C_best_l2 = model_l2.Cs_[np.argmax(scores)]

if score_best_l2 > score_best_l1:
    print "l2 regularization is better."
    print "best regularization coeff: " + str(C_best_l2)
else:
    print "l1 regularization is better."
    print "best regularization coeff: " + str(C_best_l1)
```

5 Code for Question #3

```
#%% DOWNLOAD AND PREPROCESS DATA

import pandas as pd
import sys
import requests, zipfile, StringIO

#%% get command line arguments: zipfile URL and number of samples

if len(sys.argv) < 2:
    print "data set URL required...exiting...."
    sys.exit()
else:
    zipFileURL = sys.argv[1]
    numSample = int(sys.argv[2])

#%% download the data
```

```
r = requests.get(zipFileURL)
z = zipfile.ZipFile(StringIO.StringIO(r.content))

#%% read the data

csvFile = zipFileURL.split('/')[-1][0:-4]
df = pd.read_csv(z.open(csvFile), skiprows=1)

#%% reduce the data

dfSmall = df.sample(numSample)[['id','loan_status']]

#%% save the reduced data

sampleFile = 'random.csv'
dfSmall.to_csv(sampleFile, index = False)

#%% print the size of the file

print "Number of random samples in " + sampleFile + ": " + str(numSample)
```