Note

Your Task: Use data from three Kaggle datasets to achieve a score that would rank in the top 75th percentile.

Datasets

- Regression - Allstate Claims Severity
- Logistic Regression - Springleaf Marketing

Deliverables

- 3 screenshots of your Kaggle submissions with scores in the top 75th percentile on the LB (leaderboard)
- File(s) of your code (Students may use any python based machine learning API's for the tasks and are not limited to `sci-kit learn`.)

Part A

1. Download the dataset from https://www.kaggle.com/c/allstate-claims-severity/data
2. Load the train and test datasets into `pandas` dataframes
3. Figure out which variables are categorical and which are numeric. Linear models only accept numeric variables. You can either drop the categorical columns OR turn them into numeric variables (one-hot encoding, perturbed rate-by-level, label encoding, BLUP, etc). Many times these columns are very useful and dropping them will hurt your model.
4. Feed your data to `sci-kit learn`'s Linear Regression algorithm. Sample code below:

```
from sklearn import linear_model
import numpy as np

# LOAD DATA HERE

# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training sets
# you will need to split the data into two data
# frames one with only the Y column and one with all Xs
regr.fit(X_train, y_train)

# Evaluation metrics on test data
# The coefficients
print('Coefficients: \n', regr.coef_)
# The mean squared error
print("Mean squared error: %.2f"
      % np.mean((regr.predict(X_train) - y_train) ** 2))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % regr.score(X_train, y_train))
```

5. Run predictions on test set and create a submission csv file to submit to Kaggle.

```
predictions = regr.predict(X_test)
# CRATE YOUR SUBMISSION FILE HERE
```

Part B

1. Download the dataset from https://www.kaggle.com/c/springleaf-marketing-response/data - note this data set is much larger than the others. Will have a longer processing time.
2. Load the train and test datasets into pandas dataframes
3. Figure out which variables are categorical and which are numeric. Linear models only accept numeric variables. You can either drop the categorical columns OR turn them into numeric variables (one-hot encoding, perturbed rate-by-level, label encoding, BLUP, etc). Many times these columns are very useful and dropping them will hurt your model.
4. Feed your data to `sci-kit learn`'s Logistic Regression algorithm. Sample code below:

```
from sklearn import linear_model

# LOAD DATA HERE

#create model object with one hyperparamater set
logreg = linear_model.LogisticRegression(C=1e5)

#Train the model
logreg.fit(X_train, Y_train)
```

5. Run predictions on test set and create a submission csv file to submit to Kaggle.

```
predictions = logreg.predict(X_test)
# CRATE YOUR SUBMISSION FILE HERE
```

Turn in your work as two notebooks A10a_Gwid.ipynb and A10b_Gwid.ipynb. So if your GWID is G19860011 then you should name the files as A10a_ G19860011.ipynb and A10b_ G19860011.ipynb.

Grading Rubric

	Unsatisfactory	Satisfactory	Good	Excellent
Delivery	Completed less than 70% of the requirements. Not delivered on time or not in correct format (Blackboard or git)	Completed between 70-80% of the requirements. Delivered on time, and in correct format (Blackboard or git)	Completed between 80-90% of the requirements. Delivered on time, and in correct format (Blackboard or git)	Completed between 90-100% of the requirements. Delivered on time, and in correct format (Blackboard or git)
Coding Standards	No name, date, or assignment title included Poor use of white space (indentation, blank lines). Disorganized and messy Poor use of variables(many global variables, ambiguous naming).	Includes name, date, and assignment title. White space makes program fairly easy to read. Organized work. Good use of variables (few global variables, unambiguous naming).	Includes name, date, and assignment title. Good use of white space. Organized work. Good use of variables (no global variables, unambiguous naming)	Includes name, date, and assignment title. Excellent use of white space. Creatively organized work. Excellent use of variables (no global variables, unambiguous naming).
Document ation	No documentation included.	Basic documentation has been completed including descriptions of all variables. Purpose is noted for each function.	Clearly documented including descriptions of all variables. Specific purpose is noted for each function and control structure.	Clearly and effectively documented including descriptions of all variables. Specific purpose is noted for each function, control structure, input requirements, and output results.
Runtime	Does not execute due to errors. User prompts are misleading or non-existent. No testing has been completed.	Executes without errors(if applicable). User prompts contain little information, poor design(if applicable). Some testing has been completed.	Executes without errors User prompts are understandable, minimum use of symbols or spacing in output(if applicable). Thorough testing has been completed	Executes without errors excellent. user prompts, good use of symbols, spacing in output(if applicable). Thorough and organized testing has been completed and output from test cases is included.
Efficiency	A difficult and inefficient solution.	A logical solution that is easy to follow but it is not the most efficient.	Solution is efficient and easy to follow (i.e. no confusing tricks).	Solution is efficient, easy to understand, and maintain.