**Final Project (Due Dec 18, report in PDF/Word and Python code in .py file):**

**Part I: Counting the genres of highly rated movies/shows, using a web API**

1. Read in the file "ratings.csv" and get the "tconst" values for the top 1000 rated movies (or any collection of 1000 movies in this file all with rating 10.0)
2. Request an API key at the following website (a free account allows you to access their data 1000 times per day): <http://www.omdbapi.com/>
3. Use this API to download the JSON files for your 1000 movies based on their imdb ID and store this in a list of dictionaries (some these dictionaries will just contain an error message because IMDB could not find the movie you requested, so ignore these and only keep the dictionaries that have some actual movie data in them)
4. Count the number of movies in your list for each of the possible values for "Genre" (some movies have multiple genres, if one is tagged as, say, "Drama, Comedy" then I want you to count it as both a Comedy and a Drama, then produce a bar plot showing the number for each genre (for example, it might show 350 Drama, 200 Romance, 500 Comedy, 400 NaN, etc.---it may add up to more than 1000 due to movies tagged as multiple genres, but also you removed the JSON requests with error responses so it may add up to less than 1000).

**Part II: Predicting movie genre from movie descriptions, using text mining on clean data**

1. Read in the file "imdb.csv" as a Pandas data frame, and create a new column in this data frame consisting of the first word appearing in the Genre (so if the Genre for a movie is listed as "Action,Adventure,Sci-Fi" then your new column should have the value "Action")
2. Following the text classification example on p386 of Python Handbook, use multinomial Naive Bayes "MultinonialNB" with the "TfidfVectorizer" to transform the text in the "Description" column of your data frame into a feature matrix and use your single-genre column as the target array.  In the example on p386 there is a training data set and a test data set, but for your movie data you have the target variable (the single-genre) for every movie so similar to the Example on p351, for your movie data I want you to use cross-validation to automatically split your data into train and test and to average the accuracy score obtained over these splits.

**Part III: Revenue, runtime, and rating**

1. Continue with the Pandas data frame you read in from "imdb.csv", and create a new column that is the second Genre listed for each movie, if there are at least two genres listed, and if there is only one genre listed then just use that genre (for example, if a movie has Genre "Action,Adventure,Comedy" then you should have a column with value "Action" from before and now a new column with value "Adventure", whereas if a movie has Genre "Action" then both your old and new columns should have value "Action").
2. Produce a pivot table showing the average Revenue for each value of first genre and second genre (so it should list the average revenue for Action-Action movies, and Action-Adventure movies, and ... etc.)
3. Produce a scatter plot showing how Revenue is related to Rating, and include a linear regression best-fit-line in this plot (there's a command to do this automatically, you can use that).
4. Use cross validation to get an average accuracy for predicting the first genre of movies, as you did in Part II, except now use Gaussian Naive Bayes "GaussianNB" instead of multinomial Naive Bayes, and use Revenue, Runtime, and Rating for your predictors instead of text mining the movie descriptions.