Review of Week 2

Held on Monday – Week3

Elaina Hyde

2.1.1, 2.1.2, 2.1.3 Repos

- Pecha Kucha Intro
- Import pandas as pd, pd.read_csv, file.head(), file.tail(), file.shape, file.columns, file.info(), file['column'].describe(), file.mean(), indexers (.ix, .iloc, .loc), pd.DataFrame, mydata.dtypes, file.column.unique(), file.sort_values
- Import matplotlib (.hist)
- Hercules jobs talk
- Read_csv
- Import string, string.ascii_upercase, string.ascii_lowercase
- Subset=file[[col for col in file.columns if conditions]]

2.2.1,2.2.2,2.2.3,2.2.4 repos

- EDA = exploratory data analysis
- Data types = float, int, bool, datetime64, timedelta, category, object
- Pandas .groupby (group.mean, group.max), .apply(np.mean, axis=0), .map(my_function), .value_counts (counts null values), .dtypes, pd.Series,
- Creating a dictionary:
  - 1) data=dict(A=np.random.rand(3), B=1, C='foo'…)
  - 2) df=pd.DataFrame(data)
- Type of the columns = data.get_dtype_counts().astype(list)
- Data cleaning, find a null value with .isnull()
- Visualization: matplotlib (.plot), sns (.pairplot, .corr(), .heatmap)
- Intro SQL, noSQL

Examples of mean for vector, list, array, dataframe, sql

- A list, array (choose an axis), or vector:
- l = [15, 18, 2, 36, 12, 78, 5, 6, 9]
- import numpy as np
- print np.mean(l)

- DataFrame
- df[“column“].mean()

- Sql_query= ‘‘‘**SELECT AVG**(sqft), **MIN**(price), **MAX**(price) **FROM** houses_pandas **WHERE** bdrms = 2;‘‘‘

2.3.1

- SQLite Command line: sqlite3 test1.sqlite, .help, .databases, .tables, .schema, CREATE TABLE, .exit
- DB Browser for SQLite
- Postgres SQL: \q:, \c_database_:, \d_table_:, \dt *.*, \l:, \dn:, etc.
- Interacting with sqlite from Python
- Import sqlite3, conn=sqlite3.connect(sqlite_db), c=conn.cursor(), c.execute('Query String'), conn.commit(), results.fetchall()
- Pandas Connector, .read_csv, data.to_sql()
- SQL Queries: SELECT, FROM, WHERE, AVG, MIN, MAX, MEDIAN, MODE, SUM, COUNT

2.4.1,2.4.2 repos

- import pandas as pd
- from pandas.io import sql
- cars = pd.read_csv('data/csv/car-names.csv', encoding = 'utf-8')
- .connect, .to_sql, .read_sql, .execute
- SQL join types: inner, left, right, full
- Concatenate vector1 with vector2 → np.concatenate
- Concatenation using pandas dataframes → pd.concat([df1,df2], axis=0)
- SQL-style joins in pandas → pd.merge()
- Joins concat, pd.database
- Concats of arrays in python
- Concat sql

2.5.1

- Connecting to a remote database in python:
  - from sqlalchemy import create_engine
  - import psycopg2
  - import pandas as pd
  - conn = psycopg2.connect(conn_str)
- SQL Strings: ORDER BY, Alias AS, LIKE, DISTINCT, GROUP BY, HAVING (aggregate functions), CASE (if/then logic), EXTRACT (dates)
- SQL_STRING = '''SELECT "OrderID" AS "revenue"FROM order_details;'''
- df = pd.read_sql(SQL_STRING, con=conn)

Subquery:
Once the inner query runs, the outer query will run using the results from the inner query as its underlying table:

- A subquery is a SQL query within a query. Subqueries are nested queries that provide data to the enclosing query. Subqueries can return individual values or a list of records. Subqueries must be enclosed with parenthesis.
- A subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved. Subqueries can be used with the SELECT, INSERT, UPDATE, and DELETE statements along with the operators like =, <, >, >=, <=, IN, BETWEEN etc.

Using subqueries to aggregate in multiple stages

- What if you wanted to figure out how many incidents get reported on each day of the week? Better yet, what if you wanted to know how many incidenct happen, on average, on a Friday in December? In January? There are two steps to this process: counting the number of incidents each day (inner query), then determining the monthly average (outer query):

```
SELECT LEFT(sub.date, 2) AS cleaned_month,
        sub.day_of_week,
        AVG(sub.incidents) AS
average_incidents
FROM (
        SELECT day_of_week,
                date,
                COUNT(incidnt_num) AS
incidents       FROM
tutorial.sf_crime_incidents_2014_01
        GROUP BY 1,2
         ) sub
GROUP BY 1,2
ORDER BY 1,2
```

Consider the CUSTOMERS table having the following records:

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  35 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

```
SQL> SELECT * FROM CUSTOMERS
WHERE ID IN (SELECT ID FROM
CUSTOMERS WHERE SALARY > 4500) ;
```

```
+----+----------+-----+---------+----------+
| ID | NAME     | AGE | ADDRESS | SALARY   |
+----+----------+-----+---------+----------+
|  4 | Chaitali |  25 | Mumbai  |  6500.00 |
|  5 | Hardik   |  27 | Bhopal  |  8500.00 |
|  7 | Muffy    |  24 | Indore  | 10000.00 |
+----+----------+-----+---------+----------+
```

Query String examples

- Using SQL String Functions to Clean Data
- https://community.modeanalytics.com/sql/tutorial/sql-string-functions-for-cleaning/
- Subqueries
- http://www.w3resource.com/sql/subqueries/understanding-sql-subqueries.php

Free Data

Data.gov http://data.gov The US Government pledged last year to make all government data available freely online. This site is the first stage and acts as a portal to all sorts of amazing information on everything from climate to crime.

US Census Bureau http://www.census.gov/data.html A wealth of information on the lives of US citizens covering population data, geographic data and education.

Socrata is another interesting place to explore government-related data, with some visualisation tools built-in.

European Union Open Data Portal http://open-data.europa.eu/en/data/ As the above, but based on data from European Union institutions.

Data.gov.uk http://data.gov.uk/ Data from the UK Government, including the British National Bibliography – metadata on all UK books and publications since 1950.

Free Data (2)

Canada Open Data is a pilot project with many government and geospatial datasets.

Datacatalogs.org offers open government data from US, EU, Canada, CKAN, and more.

The CIA World Factbook https://www.cia.gov/library/publications/the-world-factbook/ Information on history, population, economy, government, infrastructure and military of 267 countries.

Free Data (3)

Healthdata.gov https://www.healthdata.gov/ 125 years of US healthcare data including claim-level Medicare data, epidemiology and population statistics.

NHS Health and Social Care Information Centre http://www.hscic.gov.uk/home Health data sets from the UK National Health Service.

UNICEF offers statistics on the situation of women and children worldwide.

World Health Organization offers world hunger, health, and disease statistics.

Amazon Web Services public datasets http://aws.amazon.com/datasets Huge resource of public data, including the 1000 Genome Project, an attempt to build the most comprehensive database of human genetic information and NASA ’s database of satellite imagery of Earth.

More Free Data

Google GOOGL +% Trends
Google Finance
Google Books
National Climatic Data Center
DBPedia
New York Times NYT -1.03%
Freebase
Million Song Data Set
UCI Machine Learning Repository
Financial Data Finder
Pew Research Center
The BROAD Institute

Facebook FB -0.27% Graph
Face.com
UCLA
Data Market
Google Public data explorer
Junar is a data scraping service
Buzzdata
Gapminder