* Course Overview
* Course Introduction
  + Data Science: set of fundamental principles that guide the extraction of knowledge of data
  + Kaggle: data science competition
* Target Audience
* Course Prerequisite
* Data Science Project Cycle O…
  + Extract data
  + Organize data
  + Analyze and create models
  + Present
* Why Python for Data Science
  + Easy and intuitive
  + Tools and libraries
  + Active community
  + Scalability and fast
  + Production python based application stack
* Course Outline
  + Set up environment
  + Python distributions
  + Jupyter notebook
  + Data science project template
  + Versioning
  + Extracting data
    - Databases
    - Apis
    - Web scraping
    - Titanic dataset
    - Database connectors
    - Requests
    - Beautiful soup
  + Basic exploratory data analysis
  + Numpy
  + Panda
  + Advanced exploratory data analysis
  + Data munging(identifying issues)
  + Feature engineering
  + Visualization
  + Matplotlib
  + Machine learning
  + Build and evaluate models
  + Kaggle submission
  + Scikit-learn
  + Model tuning
  + Model persistence
  + Machine learning API
  + Pickle library
  + Flask library
* Summary
* Introduction
* Overview
  + Python distributions
  + Jupyter notebook
* Python Distributions for Data ..
  + Option 1
    - Base python
    - Then install packages one by one
  + Option 2
    - Specialized python distributions
    - Comes with preinstalled and optimized python packages
  + Python distributions for Data Science
    - Anaconda
    - Enthought canopy
* Python 3.x vs Python 2.x
  + Python 3.x
    - Clean & faster
    - Future
  + Python 2.x
    - Stable third-party packages
    - Better community support
    - Backward compatibility
* Demo: Installing Anaconda D…
  + Download installer at anaconda.com
  + Open terminal
    - Type ‘python --version’
    - Type ‘pip list’ to see install packages
    - Type ‘conda list’ to packages in anaconda distribution
* Jupyter Notebook
  + Formerly know as IPython notebook
  + Combine code block, human-friendly text, images, videos in a single document
  + Run in web browsers
  + Support different kernels
  + Viewed with nbviewer(also in github)
  + Export to various formats such as pdf
* Demo: Setting up Jupyter Not..
  + Use terminal
    - Make folder
    - Navigate to folder
    - Type ‘jupyter notebook ‘ to launch jupyter note book server on your local machine
* Demo: Jupyter Notebook - Ba…
  + In terminal type ‘jupyter notebook’ to launch
  + In the browser it will open jupyter notebook
  + Click ‘new’
    - Then select the kernel
  + Jupyter notebook is made up of cells
    - These cells can contain different types of items
    - Will be treated as code
  + On each cell you can click run
    - Or ‘CTRL + Enter’ for windows
    - Or ‘Command + Enter’ for Mac
  + The number next to cell is the execution number of the cell
  + “Shift + Enter” will execute shell and create new cell below it
  + Markdowns
    - Use to create formatted text in web browsers
    - Click on ‘Cell’ -> ‘Cell Type’ -> ‘Markdown’
    - Or use the short cut ‘Esc + M’
    - The brackets next to the cell will be blank
    - Ex) # My First Notebook
  + “Esc + L” to show line numbers
  + ! to run shell commands in jupyter
    - !python --version
  + Will automatically save after a while
    - “CTRL + S” to manually save
* Demo: Jupyter Notebook -Ma..
  + Magic functions start with % for single line, %% for multiple lines
  + %matplotlib inline
    - Commonly used data visualization
  + %time x = range(10000)
  + %lsmagic
    - List magic functions avaliable
* Data Science Project Template
  + Structing your project in a consistent fashion
  + Consistency
  + Standardization
  + Collaboration
  + Reusability
  + Code Quality
  + Cookiecutter data science - Driven data
    - <https://github.com/drivendata/cookiecutter-data-science>
* Demo: Setting up Cookiecutte…
  + Install the template
    - Type ‘pip install cookiecutter’
    - Or if you have conda ‘conda install cookiecutter’
  + Use cookiecutter to start a new project
    - Type ‘cookiecutter <https://github.com/drivendata/cookiecutter-data-science>’
    - Give it a name
* Versioning for Data Science Pr..
  + Common repository
  + Track changes
  + Suitable for individuals too
  + Versioning Systems
    - Git
    - Github
    - Bitbucket
* Demo: Add Project To Git
  + ‘git init’: initialize empty git repository
  + ‘git add .’ take all files in folder
  + ‘git commit’: create check point
  + ‘git log’: to see information
* Summary
* Introduction
* Overview
  + Databases
  + Api
  + Web scraping
  + Libraries
    - Request
    - BeautifulSoup
  + Query languages
* Extracting Data from Databases
  + SQLite
  + MySQL
  + SQL Server
  + Steps
    - Import package
    - Connect top the database
    - Create the cursor
    - Execute query
    - Fetch results
    - Close connection
* Demo: Extracting Data from D…
  + Ex)
    - import sqlite3
    - #create the database if not avaliable
    - connection = sqlite3.connect(“classroomDB.db”)
    - connetion.close()
    - connection = sqlite3.connect(“classroomDB.db”)
    - cursor = connection.cursor()
    - create\_table = “””
    - CREATE TABLE classroom {
    - student\_id INTEGER PRIMARY KEY,
    - name VARCHAR(20),
    - ……
    - ); “”””
    - cursor.execute(create\_table)
    - connection.commit()
    - connection.close()
  + ex) insert data
    - classroom\_data = [ (1, “Raj”, “M”, 70, 84, 92),
    - (2, “Poonam”, “F”, 87, 69, 93),
    - …….
    - connection = sqlite3.connect(“classroomDB.db”)
    - cursor = connection.cursor()
    - for student in classroom\_data:
    - insert\_statement = “”” INSERT INTO classroom
    - (student\_id, name, gender, physics\_marks, chemistry\_marks, mathematics\_marks) VALUES ({0}, “{1}”, “{2}”, {3}, {4}, {5}); “””.format(student[0], student[1], student[2], student[3], student[4], student[5])
    - cursor.execute(insert\_statement)
    - connection.commit()
    - connection.close()
  + extract data
    - connection = sqlite3.connect(“classroomDB.db”)
    - cursor = connection.cursor()
    - query = “ SELECT \* FROM classroom”
    - cursor.execute(query)
    - result = cursor.fetchall()
    - for row in result:
    - print(row)
    - connection.close()
  + MySQL example
    - !conda install -y -q pymsql
    - import pymsql
  + connect to database
    - cnx = {‘host’: ‘[server url]’,
    - ‘username’: ‘[username]’,
    - ‘password’: ‘[password]’,
    - ‘db’: ‘[name of database]’}
    - connection = pymysql.connect(cnx[‘host’], cnx[‘username’], cnx[‘password’], cnx[‘db’] )
    - connection.close()
* Extracting Data Through APIs