Establishing Python Environments

Suman Samui

Initial Steps

- Anaconda Installation → Done

- Jupyter Notebooks.

- How to test your Anaconda installation?

Project Specific Python Environment

- It is important for the reproducibility of your work (project).
- You should share the project environment along with source codes.

Project Specific Python Environment

Jack

Jack's project

Jack_main.py

Olivia

- **Matplotlib 1.5.3**
- **Numpy1.12**
- **Keras 2.3**
- **Pandas 1.0.3**

Jack_python_env

- **Matplotlib 1.2.6**
- **Numpy1.1**
- **Keras 1.8**
- **Pandas 1.0.3**

Create, save, and share portable Python environments

- Python's *virtualenv, pipenv, pyenv etc.*

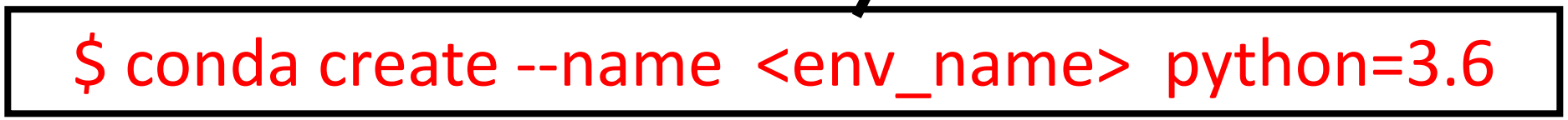
- *conda* is a *general-purpose* package management system, and not just a *Python* package management system

Steps to follow:

1. **New Project, New Directory**

Your Choice

2. **New Project, New Environment**



Steps to follow (contd…)

3. **Activating/Deactivating the Environment**

```
$ activate <env_name>
```

4. Installing Packages

```
[<env_name>] $ conda install <package_name>
```

5. **Deactivating the Environment**

```
$ deactivate <env_name>
```

Additional commands:

- **Saving the environment:**

```
[<env_name>] $ conda env export > environment.yaml
```

- **Reusing the environment (when shared with others):**

```
$ conda env create -f environment.yaml
```