INTRODUCTION TO MACHINE LEARNING

CPOSC

CENTRAL PENNSYLVANIA OPEN SOURCE CONFERENCE

Sponsored by LISTRAK®

About Me…

Development Environment

Supervised Learning

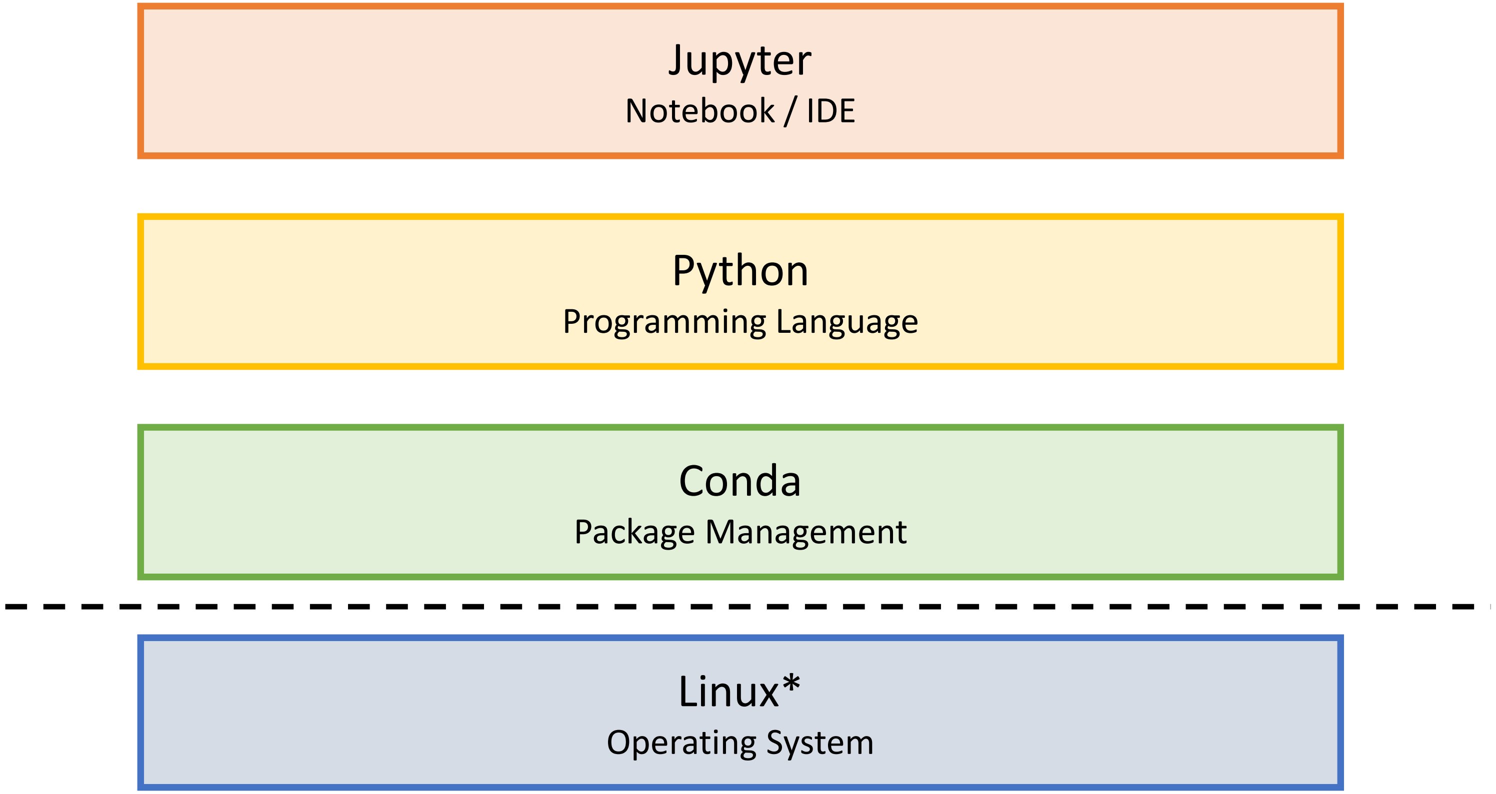
Unsupervised Learning

Development Environment

Supervised Learning

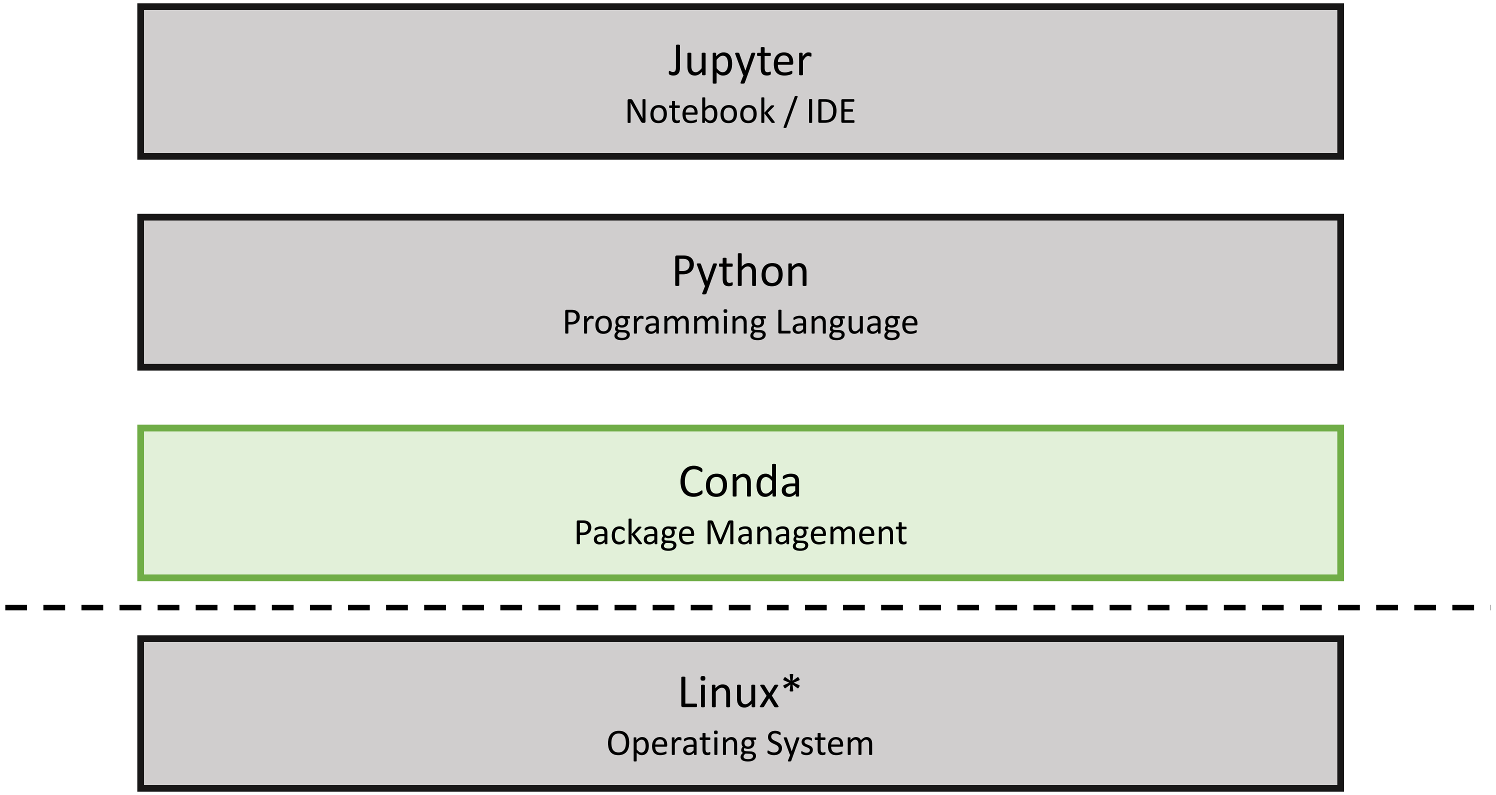
Unsupervised Learning

Development Environment



Conda

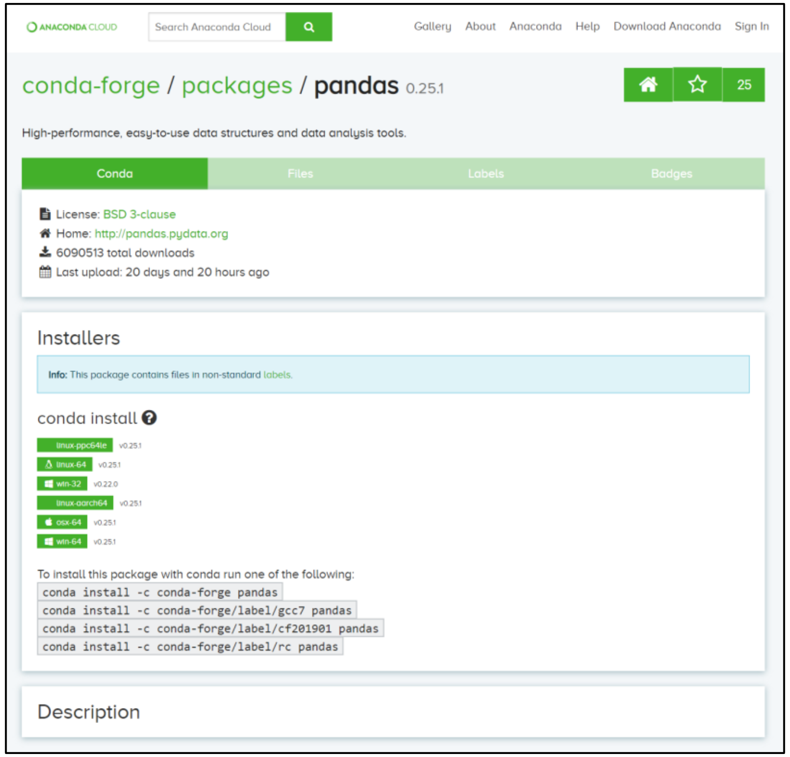
https://conda.io/
https://github.com/conda



Conda

https://conda.io/
https://github.com/conda

- Package Management
  - https://anaconda.org/
  - “You can search and download popular Python and R packages and notebooks to jumpstart your data science work.”




```
$ conda install pandas
```

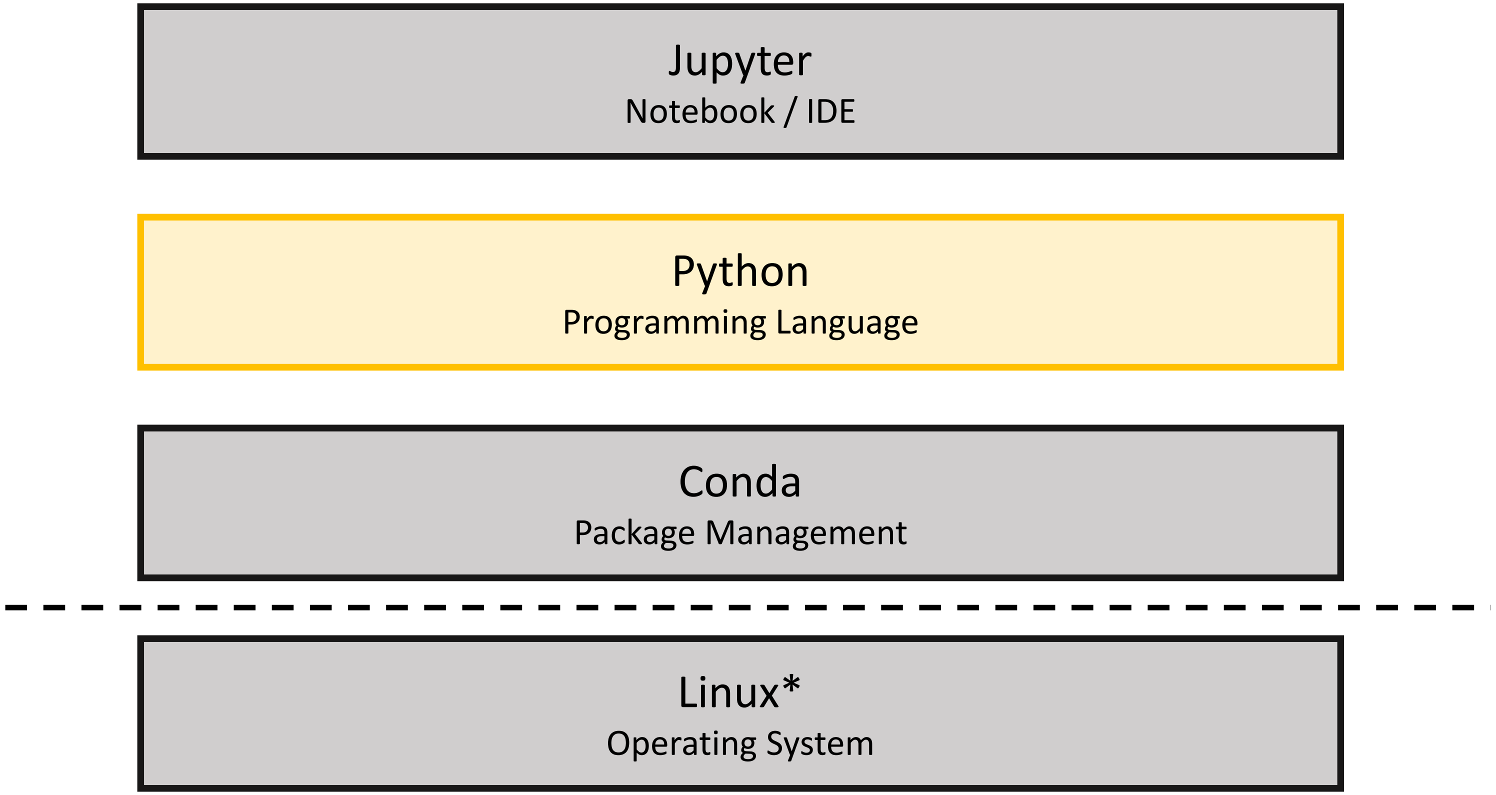
Conda

https://conda.io/
https://github.com/conda

```
$ cd ~
$ wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
$ sh Miniconda3-latest-Linux-x86_64.sh
```

Python

https://www.python.org/
https://github.com/python



Python

https://www.python.org/
https://github.com/python

TensorFlow

SciPy

scikit learn

NumPy

PYTORCH

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

Keras

matplotlib

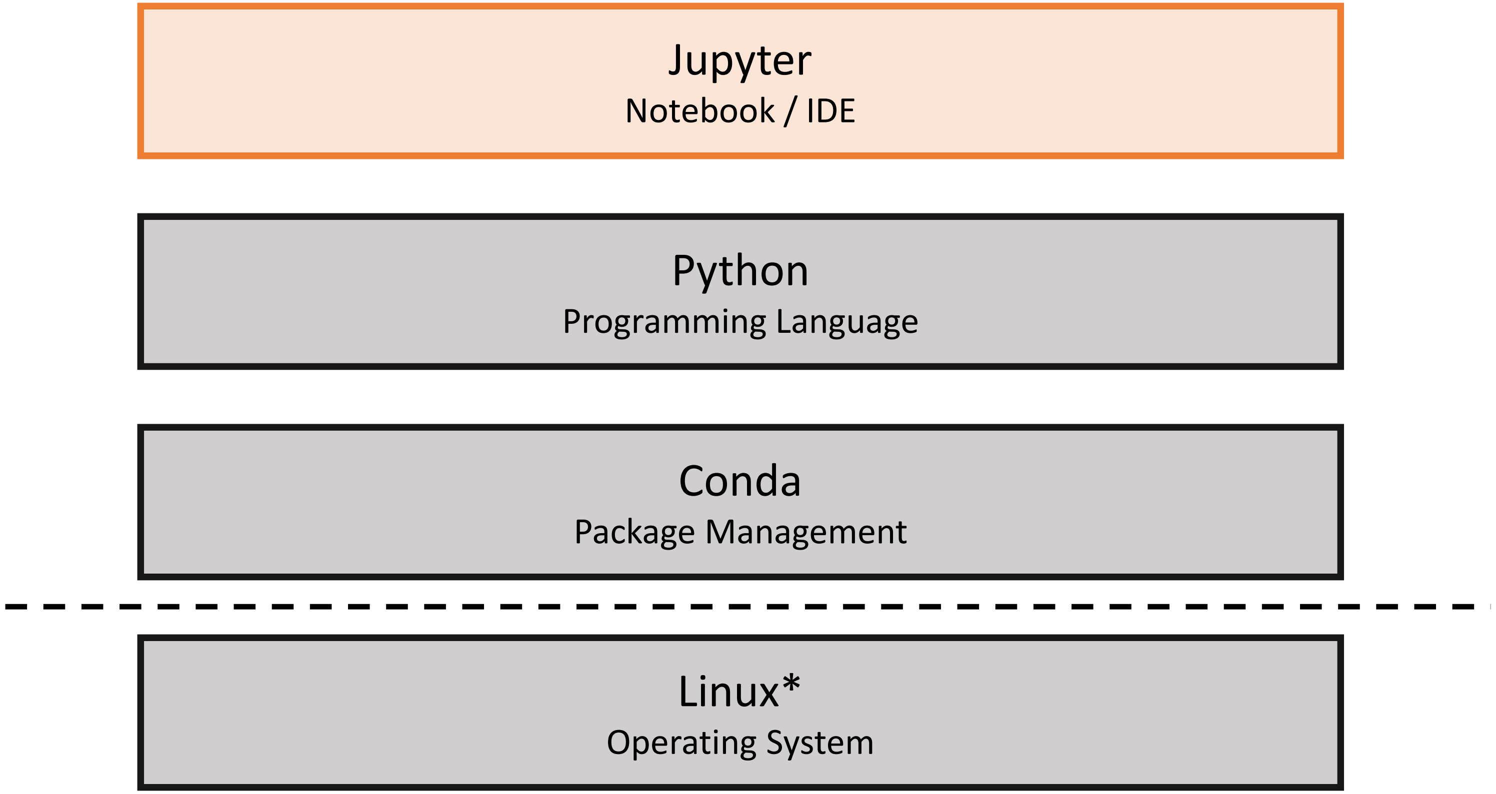
Python

https://www.python.org/
https://github.com/python

```
$ conda install python
```

Jupyter

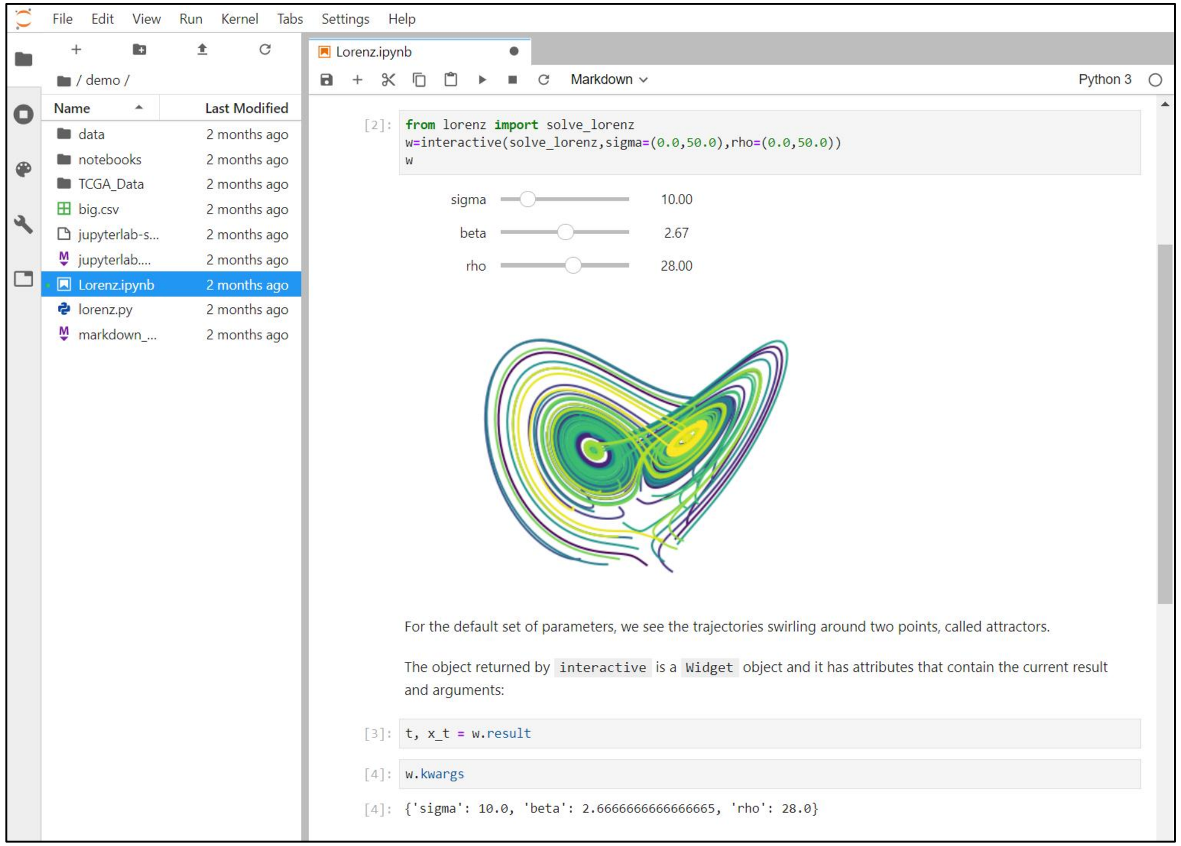
https://jupyter.org/
https://github.com/jupyter



Jupyter

https://jupyter.org/
https://github.com/jupyter

- Web-based development environment
- Notebooks
    - Live code cells
    - Visualizations
    - Text (Markdown)



Jupyter

https://jupyter.org/
https://github.com/jupyter

```
$ conda install jupyterlab
$ jupyter lab
```

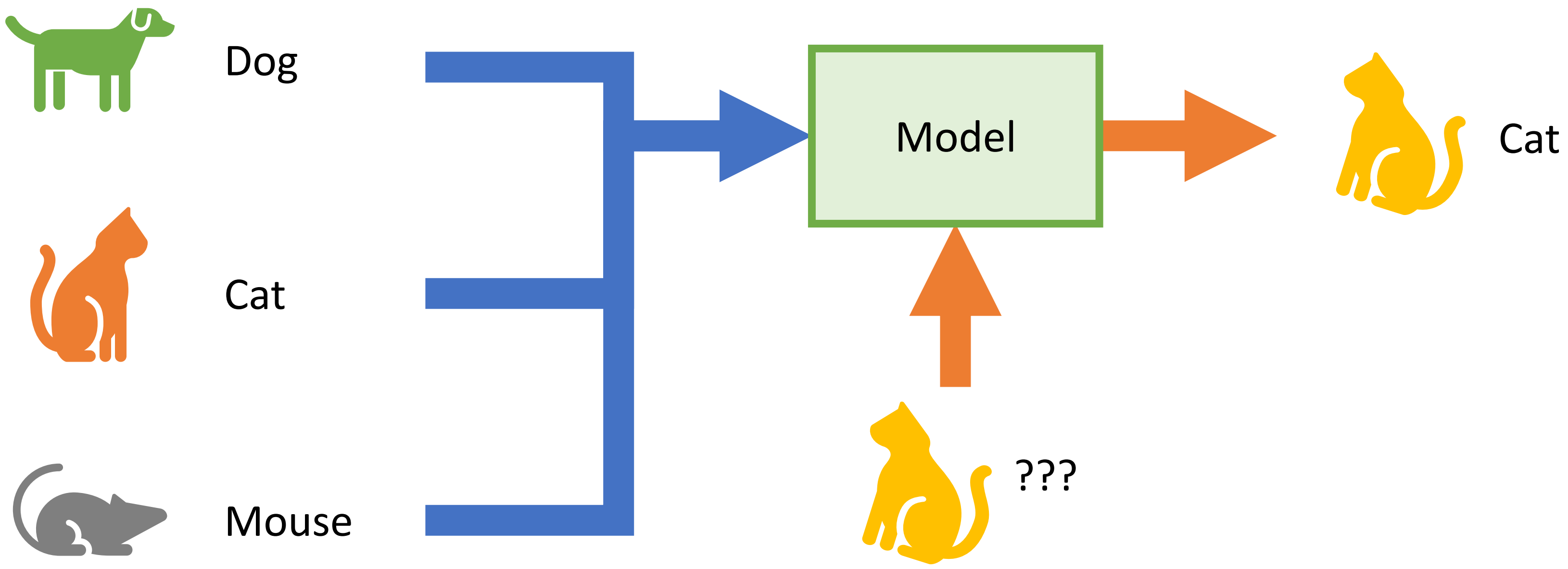
Development Environment

Supervised Learning

Unsupervised Learning

Supervised Learning

- Train a model using labeled data



Supervised Learning

- Classification
  - Predicts a class [dog, cat, mouse]
- Binary classification
  - Predicts a binary class [true, false]
- Regression
  - Predict a continuous value

Binary classification

Regression

Heart Disease Example

https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

- UCI Machine Learning Repository
  - https://archive.ics.uci.edu/ml/datasets/Heart+Disease
- 303 labeled patients
  - 13 features (age, cholesterol, blood pressure, etc.)
  - Is heart disease present? (yes/no)
- Can we predict the presence of heart disease?
  - Binary classification

Heart Disease Example

https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

- **age**: age in years
- **sex**: sex
  - 1 = male; 0 = female
- **cp**: chest pain type
  - 1 = typical angina; 2 = atypical angina; 3 = non-anginal pain; 4 = asymptomatic
- **trestbps**: resting blood pressure in mm Hg
- **chol**: serum cholesterol in mg/dl
- **fbs**: fasting blood sugar > 120 mg/dl
  - 1 = true; 0 = false
- **restecg**: resting electrocardiographic results

Heart Disease Example

https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

- **thalach**: maximum heart rate achieved
- **exang**: exercise induced angina
  - 1 = yes; 0 = no
- **oldpeak:** ST depression induced by exercise relative to rest
- **slope:** the slope of the peak exercise ST segment
  - 1 = upsloping; 2 = flat; 3 = downsloping
- **ca**: number of major vessels (0-3) colored by flourosopy
- **thal:**
  - 3 = normal; 6 = fixed defect; 7 = reversable defect
- **num:** diagnosis of heart disease (angiographic disease status)
  - 0 = absence; 1, 2, 3, 4 = presence

Data Frame

https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

```
>>> heart = pd.read_csv('data/processed.cleveland.data', names=names, dtype=dtype, na_values='?')
>>> heart.dropna(inplace=True)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	num
0	63	1	1	145	233	1	2	150	0	2.3	3	0	6	0
1	67	1	4	160	286	0	2	108	1	1.5	2	3	3	2
2	67	1	4	120	229	0	2	129	1	2.6	2	2	7	1
3	37	1	3	130	250	0	0	187	0	3.5	3	0	3	0
4	41	0	2	130	204	0	2	172	0	1.4	1	0	3	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
297	57	0	4	140	241	0	0	123	1	0.2	2	0	7	1
298	45	1	1	110	264	0	0	132	0	1.2	2	0	7	1
299	68	1	4	144	193	1	0	141	0	3.4	2	2	7	2
300	57	1	4	130	131	0	0	115	1	1.2	2	1	7	3
301	57	0	2	130	236	0	2	174	0	0.0	2	1	3	1

297 rows × 14 columns

One Hot Encoding

https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

```
>>> heart.cp
```

	cp
0	1
1	4
2	4
3	3
4	2
...	...
297	4
298	1
299	4
300	4
301	2

297 rows × 1 columns

```
>>> pd.get_dummies(heart.cp, prefix='cp')
```

	cp_1	cp_2	cp_3	cp_4
0	1	0	0	0
1	0	0	0	1
2	0	0	0	1
3	0	0	1	0
4	0	1	0	0
...	...	...	...	...
297	0	0	0	1
298	1	0	0	0
299	0	0	0	1
300	0	0	0	1
301	0	1	0	0

297 rows × 4 columns

Training Set / Test Set

https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

```
>>> x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
```

	age	sex	trestbps	...	thal_3	thal_6	thal_7
278	57	1	154	...	1	0	0
259	57	1	124	...	0	0	1
7	57	0	120	...	1	0	0
186	42	1	120	...	0	0	1
172	59	0	174	...	1	0	0
...	...	...	...	...	...	...	...
107	57	1	128	...	0	0	1
83	68	1	180	...	0	0	1
17	54	1	140	...	1	0	0
233	74	0	120	...	1	0	0
99	48	1	122	...	1	0	0

222 rows × 20 columns

x_train

	y
278	True
259	True
7	False
186	False
172	True
...	...
107	True
83	True
17	False
233	False
99	False

222 rows × 1 columns

y_train

	age	sex	trestbps	...	thal_3	thal_6	thal_7
81	53	0	130	...	1	0	0
126	56	0	200	...	0	0	1
106	59	1	140	...	0	0	1
279	58	0	130	...	1	0	0
268	40	1	152	...	0	0	1
...	...	...	...	...	...	...	...
264	61	1	138	...	1	0	0
153	55	1	160	...	0	0	1
221	54	0	108	...	1	0	0
5	56	1	120	...	1	0	0
250	57	1	110	...	0	1	0

75 rows × 20 columns

x_test

	y
81	False
126	True
106	True
279	False
268	True
...	...
264	True
153	True
221	False
5	False
250	False

75 rows × 1 columns

y_test

Random Forest

https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

```
>>> rf = RandomForestClassifier(n_estimators=500)
>>> rf.fit(x_train, y_train)
```

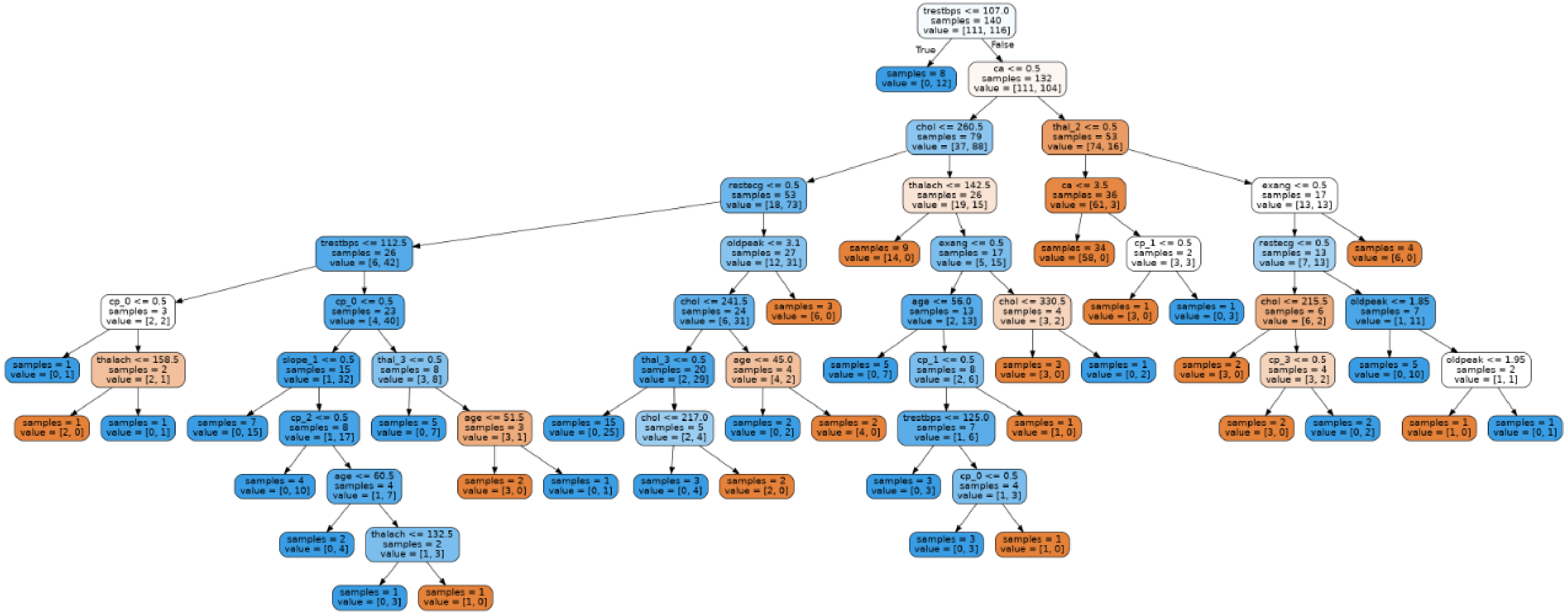
	age	sex	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	ca	cp_1	cp_2	cp_3	cp_4	slope_1	slope_2	slope_3	thal_3	thal_6	thal_7
278	57	1	154	232	0	2	164	0	0.0	1	0	1	0	0	1	0	0	1	0	0
259	57	1	124	261	0	0	141	0	0.3	0	0	1	0	0	1	0	0	0	0	1
7	57	0	120	354	0	0	163	1	0.6	0	0	0	0	1	1	0	0	1	0	0
186	42	1	120	240	1	0	194	0	0.8	0	0	0	1	0	0	0	1	0	0	1
172	59	0	174	249	0	0	143	1	0.0	0	0	0	0	1	0	1	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
107	57	1	128	229	0	2	150	0	0.4	1	0	0	1	0	0	1	0	0	0	1
83	68	1	180	274	1	2	150	1	1.6	0	0	0	1	0	0	1	0	0	0	1
17	54	1	140	239	0	0	160	0	1.2	0	0	0	0	1	1	0	0	1	0	0
233	74	0	120	269	0	2	121	1	0.2	1	0	1	0	0	1	0	0	1	0	0
99	48	1	122	222	0	2	186	0	0.0	0	0	0	0	1	1	0	0	1	0	0

222 rows × 20 columns

Random Forest

https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

```
>>> rf = RandomForestClassifier(n_estimators=500)
>>> rf.fit(x_train, y_train)
```



Random Forest

https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

```
>>> rf = RandomForestClassifier(n_estimators=500)
>>> rf.fit(x_train, y_train)
```

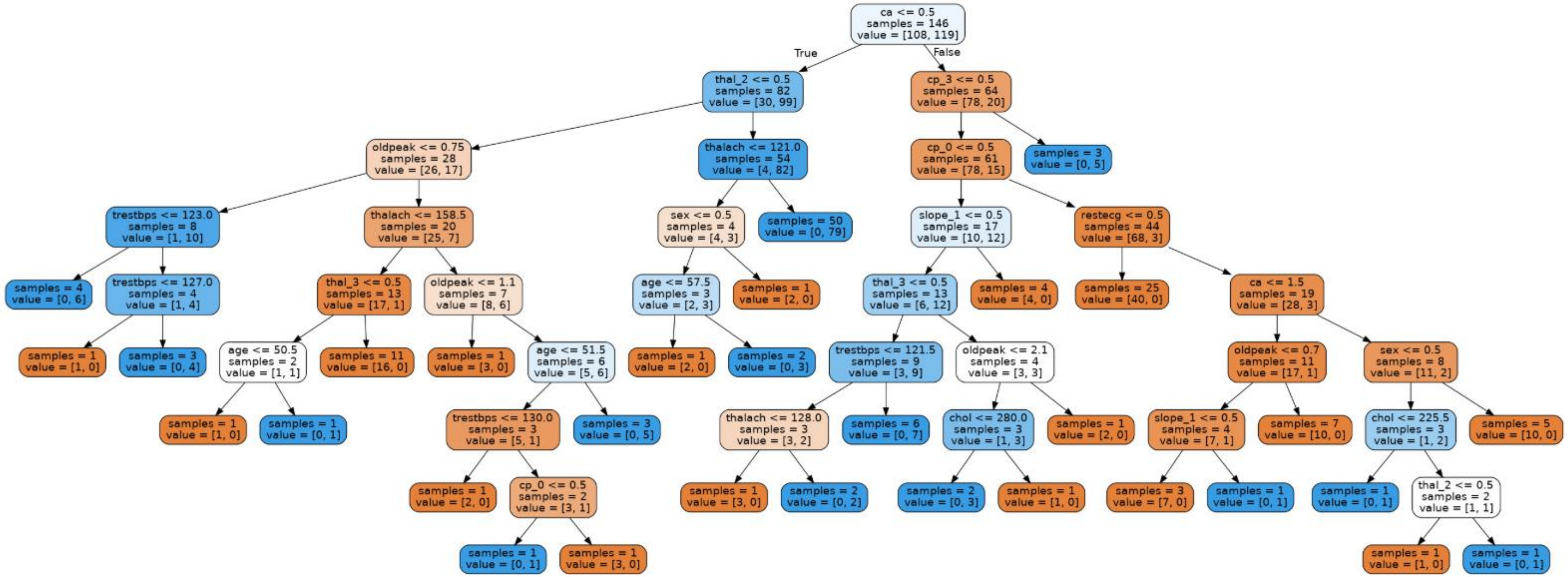
	age	sex	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	ca	cp_1	cp_2	cp_3	cp_4	slope_1	slope_2	slope_3	thal_3	thal_6	thal_7
278	57	1	154	232	0	2	164	0	0.0	1	0	1	0	0	1	0	0	1	0	0
259	57	1	124	261	0	0	141	0	0.3	0	0	1	0	0	1	0	0	0	0	1
7	57	0	120	354	0	0	163	1	0.6	0	0	0	0	1	1	0	0	1	0	0
186	42	1	120	240	1	0	194	0	0.8	0	0	0	1	0	0	0	1	0	0	1
172	59	0	174	249	0	0	143	1	0.0	0	0	0	0	1	0	1	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
107	57	1	128	229	0	2	150	0	0.4	1	0	0	1	0	0	1	0	0	0	1
83	68	1	180	274	1	2	150	1	1.6	0	0	0	1	0	0	1	0	0	0	1
17	54	1	140	239	0	0	160	0	1.2	0	0	0	0	1	1	0	0	1	0	0
233	74	0	120	269	0	2	121	1	0.2	1	0	1	0	0	1	0	0	1	0	0
99	48	1	122	222	0	2	186	0	0.0	0	0	0	0	1	1	0	0	1	0	0

222 rows × 20 columns

Random Forest

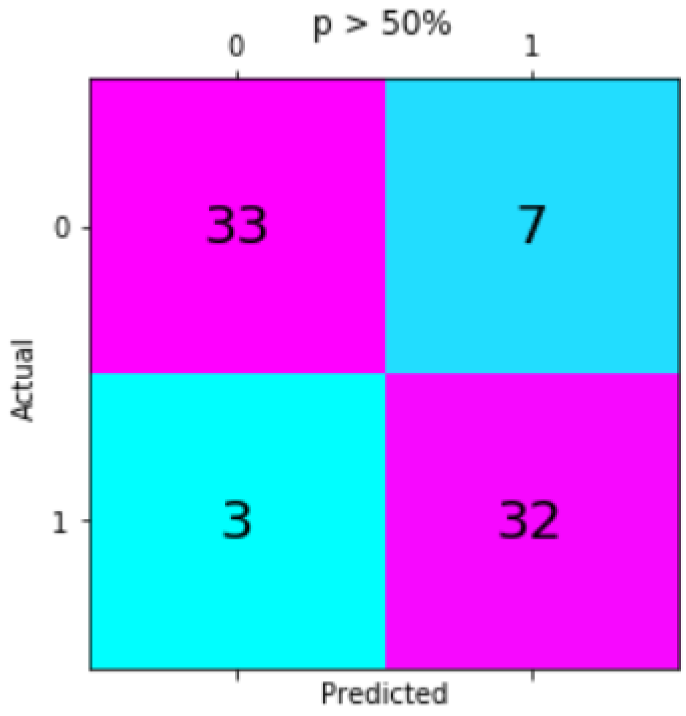
https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

```
>>> rf = RandomForestClassifier(n_estimators=500)
>>> rf.fit(x_train, y_train)
```



Confusion Matrix

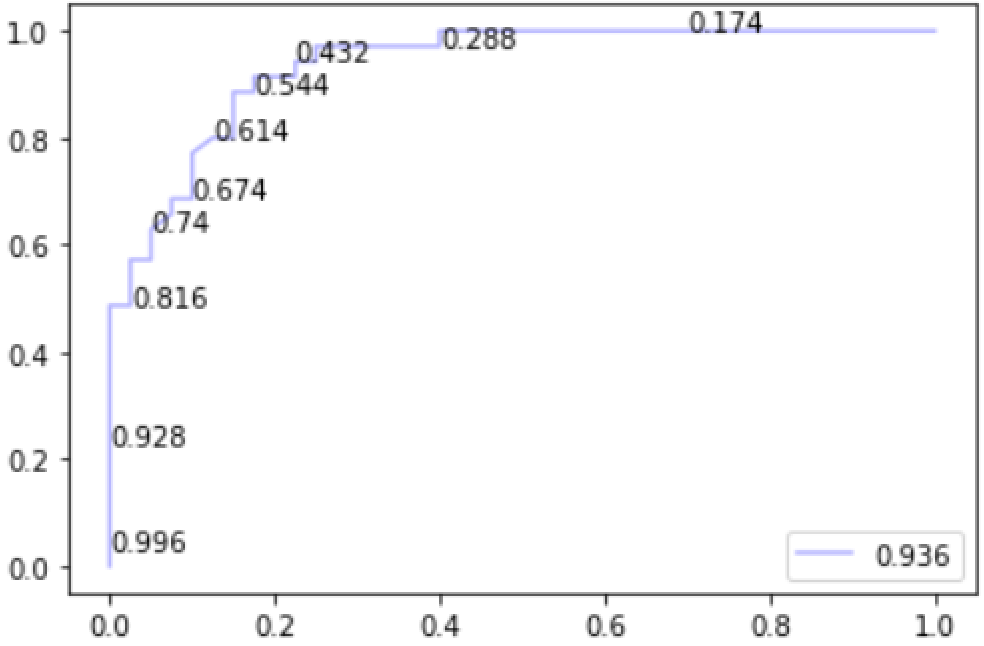
https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb



ROC Curve

https://archive.ics.uci.edu/ml/datasets/Heart+Disease
https://github.com/sranck/cposc2019/heart.ipynb

- Receiver Operating Characteristic
- True Positive Rate (y-axis) vs False Positive Rate (x-axis)



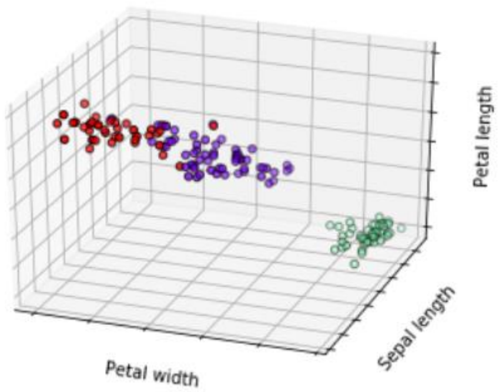
Development Environment

Supervised Learning

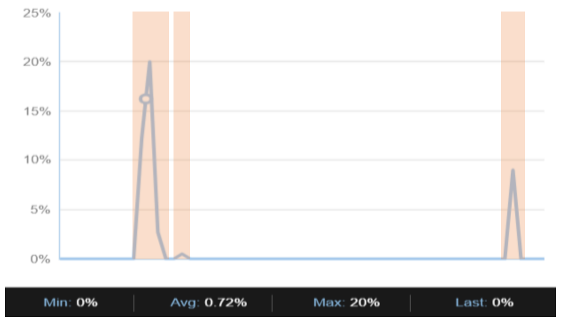
Unsupervised Learning

Unsupervised Learning

- Find patterns in unlabeled data
- Clustering
  - Arrange objects into groups
- Anomaly detection
  - Identify rare events / outliers



Clustering



Anomaly detection

US Senate Example

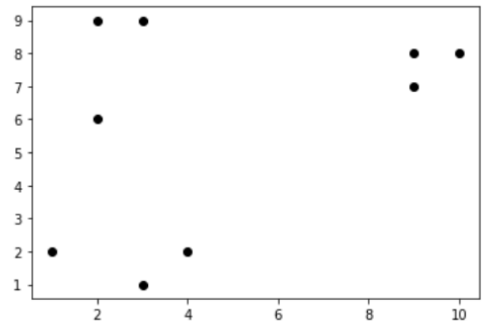
https://voteview.com/data
https://github.com/sranck/cposc2019/senate.ipynb

- Voteview
  - https://voteview.com/data
  - https://voteview.com/static/data/out/votes/S114_votes.csv
  - https://voteview.com/static/data/out/members/S114_members.csv
- Voting record for the 114th Congress (Senate only)
  - 100 senators
  - 502 votes
  - 50,200 voting records
- What happens if we cluster senators by their voting record?

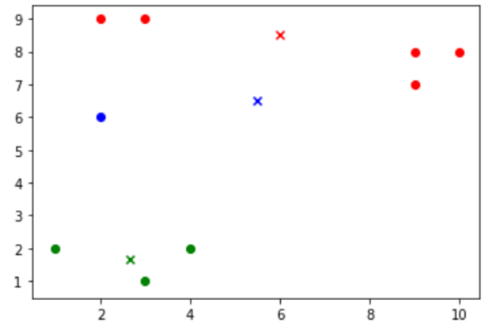
K-means Clustering

https://voteview.com/data
https://github.com/sranck/cposc2019/senate.ipynb

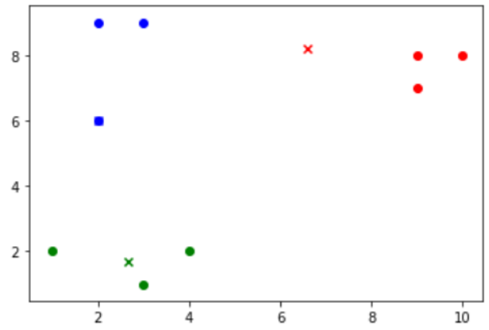
- Groups n points into k clusters



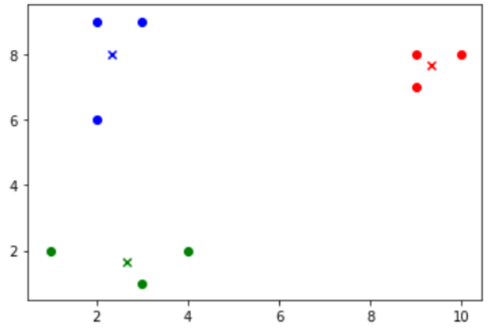
Points



Iteration 1



Iteration 2



Iteration 3

Data Frame

https://voteview.com/data
https://github.com/sranck/cposc2019/senate.ipynb

```
>>> votes = pd.read_csv('data/s114_votes.csv')
>>> votes = pd.DataFrame({
...     'senator_id': votes.icpsr.astype('str’),
...     'vote_id': votes.rollnumber.astype('str’),
...     'vote_cast': votes.cast_code.astype('category’)
... })
```

	congress	chamber	rollnumber	icpsr	cast_code	prob
0	114	Senate	1	14009	6	100.0
1	114	Senate	1	14226	6	100.0
2	114	Senate	1	14307	1	90.5
3	114	Senate	1	14435	1	99.8
4	114	Senate	1	14440	1	91.1
...	...	...	...	...	...	...
50195	114	Senate	502	49308	6	34.1
50196	114	Senate	502	49700	6	8.9
50197	114	Senate	502	49703	1	69.3
50198	114	Senate	502	49706	1	91.8
50199	114	Senate	502	94659	6	4.5

50200 rows × 6 columns

	senator_id	vote_id	vote_cast
0	14009	1	6
1	14226	1	6
2	14307	1	1
3	14435	1	1
4	14440	1	1
...	...	...	...
50195	49308	502	6
50196	49700	502	6
50197	49703	502	1
50198	49706	502	1
50199	94659	502	6

50200 rows × 3 columns

Co-occurrence Matrix

```
>>> votes.set_index(['vote_id', 'vote_cast'], inplace=True)
>>> senator_pairs = votes.join(votes, lsuffix='_a', rsuffix='_b')
>>> senator_pairs = senator_pairs.groupby(['senator_id_a', 'senator_id_b']).n.count()
>>> matrix = senator_pairs.pivot(index='senator_id_a', columns='senator_id_b')
```

	senator_id	vote_id	vote_cast
0	14009	1	6
1	14226	1	6
2	14307	1	1
3	14435	1	1
4	14440	1	1
...	...	...	...
50195	49308	502	6
50196	49700	502	6
50197	49703	502	1
50198	49706	502	1
50199	94659	502	6

50200 rows × 3 columns

senator_id_b	14009	14226	14307	14435	14440	...	49308	49700	49703	49706	94659
senator_id_a											
14009	502	434	206	181	213	...	226	408	396	418	394
14226	434	502	181	168	189	...	207	417	378	432	417
14307	206	181	502	461	446	...	454	154	296	176	143
14435	181	168	461	502	431	...	447	141	272	160	138
14440	213	189	446	431	502	...	430	158	296	181	161
...	...	...	...	...	...	...	...	...	...	...	...
49308	226	207	454	447	430	...	502	159	299	195	156
49700	408	417	154	141	158	...	159	502	339	414	439
49703	396	378	296	272	296	...	299	339	502	353	335
49706	418	432	176	160	181	...	195	414	353	502	402
94659	394	417	143	138	161	...	156	439	335	402	502

100 rows × 100 columns

K-means Clustering

https://voteview.com/data
https://github.com/sranck/cposc2019/senate.ipynb

```
>>> km = KMeans(n_clusters=2).fit(matrix)
>>> clusters = pd.DataFrame({ 'cluster': km.labels_ }, matrix.index.rename('senator_id'))
```

senator_id	party	name
14307	D	LEAHY, Patrick Joseph
14435	D	MARKEY, Edward John
14440	D	MIKULSKI, Barbara Ann
14651	D	NELSON, Clarence William (Bill)
14858	D	SCHUMER, Charles Ellis (Chuck)
...	...	...
41303	D	HEITKAMP, Mary Kathryn (Heidi)
41305	D	KAINE, Timothy Michael (Tim)
41308	D	BOOKER, Cory Anthony
49300	D	FEINSTEIN, Dianne
49308	D	MURRAY, Patty

46 rows × 2 columns

```
>>> senators[clusters.cluster == 0]
```

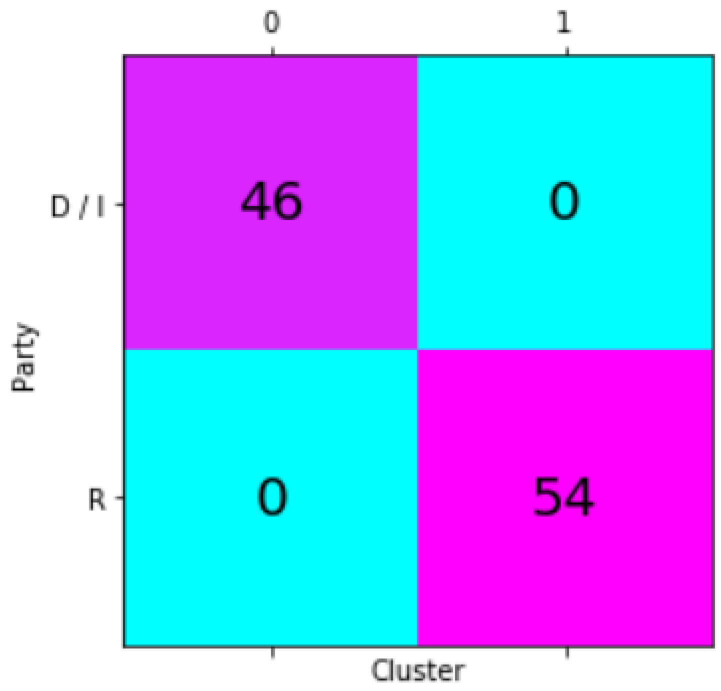
senator_id	party	name
14009	R	COCHRAN, William Thad
14226	R	GRASSLEY, Charles Ernest
14503	R	HATCH, Orrin Grant
14806	R	COATS, Daniel Ray
14852	R	ROBERTS, Charles Patrick (Pat)
...	...	...
41505	R	ROUNDS, Marion Michael (Mike)
49700	R	SESSIONS, Jefferson Beauregard III (Jeff)
49703	R	COLLINS, Susan Margaret
49706	R	ENZI, Michael B.
94659	R	SHELBY, Richard C.

54 rows × 2 columns

```
>>> senators[clusters.cluster == 1]
```

Confusion Matrix

https://voteview.com/data
https://github.com/sranck/cposc2019/senate.ipynb



QUESTIONS?

MORE INFO: https://github.com/sranck/cposc2019/

THANK YOU!