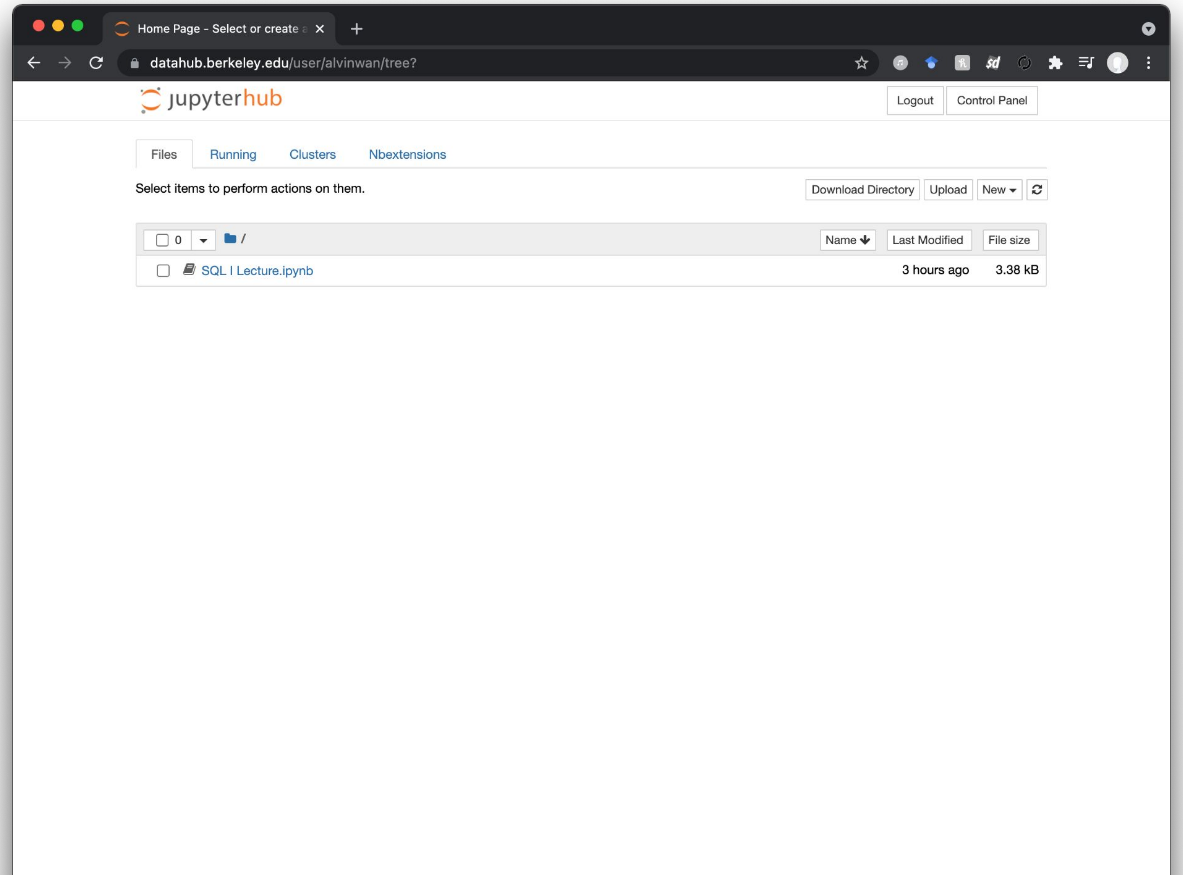
LECTURE 16

Numpy Coding a Linear Model

Building your first linear model

Data 100/Data 200, Fall 2021 @ UC Berkeley
Fernando Pérez and Alvin Wan
(content by Alvin Wan, John DeNero, Joseph E. Gonzalez, Josh Hug)

data100.datahub.berkeley.edu



```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Load Dataset

Fit OLS

Fit Biased OLS

Load Dataset

Fit OLS

Fit Biased OLS

Solving OLS

$$\mathcal{L} = \mathbb{Y} - \hat{\mathbb{Y}}$$

$$\mathcal{L} = (\mathbb{Y} - \hat{\mathbb{Y}})^2$$

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(\mathbb{Y}_i - \hat{\mathbb{Y}}_i)^2$$

$$\hat{\mathbb{Y}} = \mathbb{X}\theta$$

$$\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\mathbb{Y}_i - (\mathbb{X}\theta)_i)^2$$

$$\hat{\theta} = \text{argmin}_{\theta}\mathcal{L}(\theta)$$

$$\hat{\theta} = (\mathbb{X}^T\mathbb{X})^{-1}\mathbb{X}^T\mathbb{Y}$$

Predict with OLS

$$\hat{\mathbb{Y}} = \mathbb{X}\hat{\boldsymbol{\theta}}$$

Analyze Model Performance

Scatter plot Y vs. $\hat{Y}$

Histogram Y - $\hat{Y}$

Compute $L(\boldsymbol{\theta})$

Plot Predictions

Load Dataset

Fit OLS

Fit Biased OLS

$$\hat{\mathbb{Y}} = \mathbb{X}\boldsymbol{\theta} + \boldsymbol{b}$$

Solving OLS

$$\hat{\boldsymbol{\theta}} = (\mathbb{X}^T\mathbb{X})^{-1}\mathbb{X}^T\mathbb{Y}$$

$$\phi \Big\downarrow$$

$$\hat{\boldsymbol{\theta}} = (\phi(\mathbb{X})^T\phi(\mathbb{X}))^{-1}\phi(\mathbb{X})^T\mathbb{Y}$$

Predict with OLS

$$\hat{\mathbb{Y}} = \mathbb{X}\hat{\boldsymbol{\theta}}$$

$$\phi \Big\downarrow$$

$$\hat{\mathbb{Y}} = \phi(\mathbb{X})\hat{\boldsymbol{\theta}}$$

Analyze Model Performance

Scatter plot Y vs. $\hat{Y}$

Histogram $Y - \hat{Y}$

Compute $L(\boldsymbol{\theta})$

Plot Predictions

	MSE
OLS	346
Biased OLS	187

Analyze Model Performance

Scatter plot Y vs. $\hat{Y}$

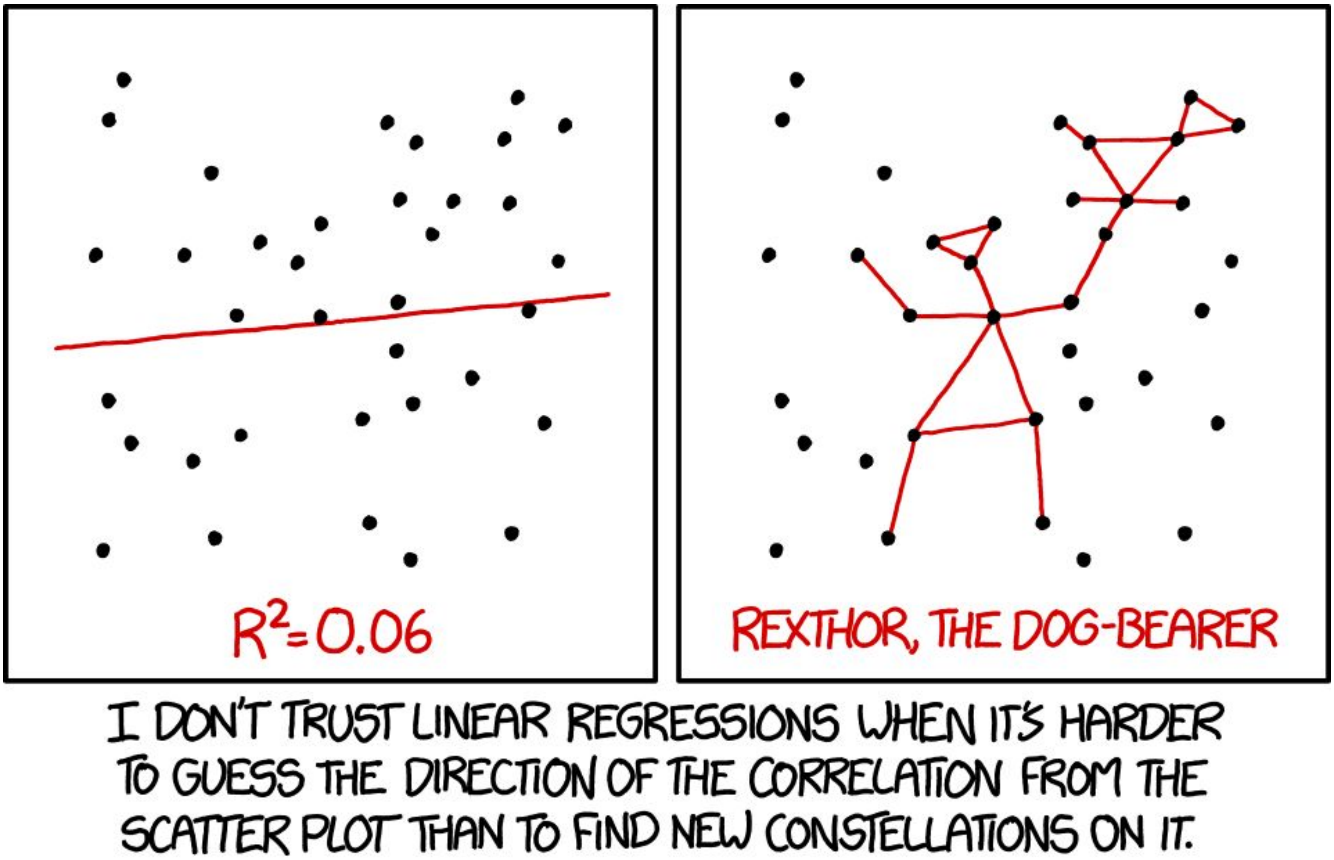
Histogram Y - $\hat{Y}$

Compute $L(\boldsymbol{\theta})$

Plot Predictions

PRACTICAL TIP

`numpy.linalg.solve` is more efficient at solving linear equations.



xkcd.com/1725/

LECTURE 16

Scikit-Learn Coding a Linear Model

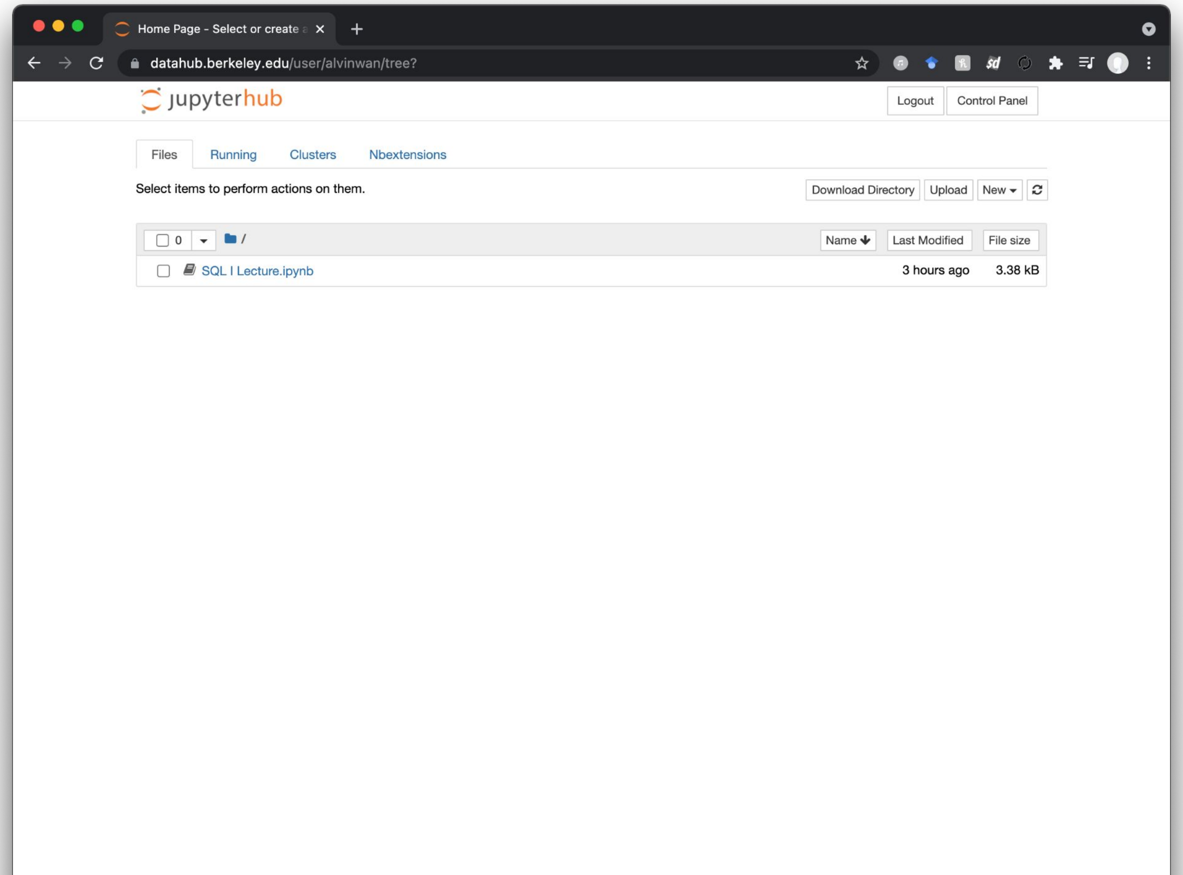
Build a linear model much faster, with scikit-learn abstractions

Data 100/Data 200, Fall 2021 @ UC Berkeley

Fernando Pérez and Alvin Wan

(content by Alvin Wan, John DeNero, Joseph E. Gonzalez, Josh Hug)

data100.datahub.berkeley.edu



Load Dataset

Fit OLS

Fit Biased OLS

Load Dataset

Fit OLS

Fit Biased OLS

PRACTICAL TIP

`sklearn` models always follow 3 steps:

1. Instantiate
   `model = SuperCoolModel()`
2. Fit
   `model.fit(X, Y)`
3. Predict
   `model.predict(X)`

Load Dataset

Fit OLS

Fit Biased OLS

PRACTICAL TIP

`sklearn` models always follow 3 steps:

1. Instantiate

   `model = SuperCoolModel()`

2. Fit

   `model.fit(X, Y)`

3. Predict

   `model.predict(X)`

	MSE	MAE	RMSE
OLS	346	14.0	18.6
Biased OLS	187	9.9	13.7

TAKEAWAY

Learn **scikit-learn** to vastly **simplify and speed up** your data science and machine learning workflow.

LECTURE 16

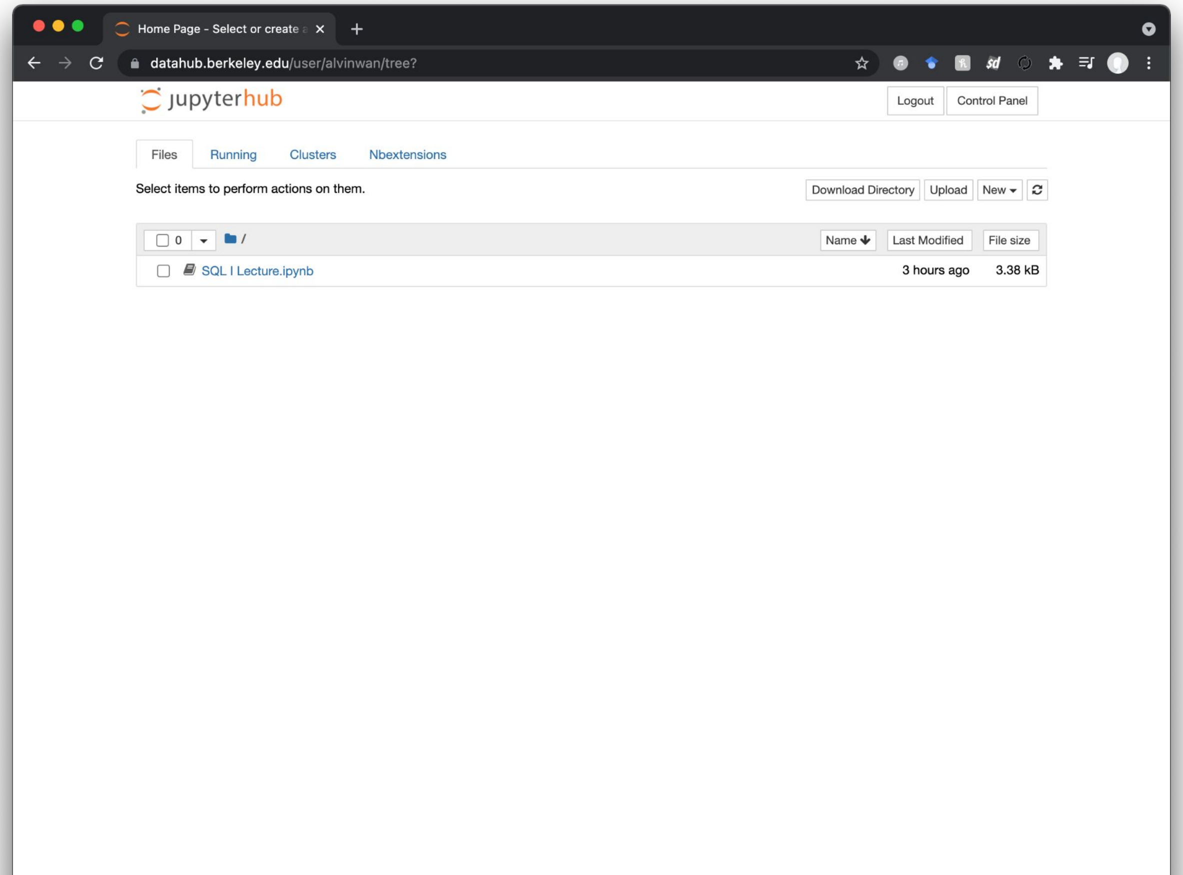
Benefit #1 Modeling Non-Linearities

Use feature engineering to model non-linear relationships, with a linear model

Data 100/Data 200, Fall 2021 @ UC Berkeley

Fernando Pérez and Alvin Wan
(content by Alvin Wan, John DeNero, Joseph E. Gonzalez, Josh Hug)

data100.datahub.berkeley.edu



$$\hat{y} = \sum_{j=1}^{d} \overbrace{x_j}^{\text{d Features}} \underbrace{\theta_j}_{\text{Parameters}}$$

$$\phi : \mathbb{R}^d \to \mathbb{R}^p$$

$$\hat{y} = \sum_{j=1}^{p} \overbrace{\underbrace{\phi}_{\substack{\text{(phi)ture function} \\ \textbf{- Prof. Gonzalez, Sp21}}}(x)_j}^{\text{p Features}} \, \underbrace{\theta_j}_{\text{Parameters}}$$

Load Dataset

Fit Biased OLS

Polynomial Features

Sinusoidal Features

Load Dataset

Fit Biased OLS

Polynomial Features

Sinusoidal Features

	MSE	MAE	RMSE
OLS	346	14.0	18.6
Biased OLS	187	9.9	13.7
Biased OLS + Location	167	9.0	12.9

Load Dataset

Fit Biased OLS

Polynomial Features

Sinusoidal Features

	MSE	MAE	RMSE
OLS	346	14.0	18.6
Biased OLS	187	9.9	13.7
Biased OLS + Location	167	9.0	12.9
Biased OLS + Location + Poly	141	8.5	11.9

Load Dataset

Fit Biased OLS

Polynomial Features

Sinusoidal Features

	MSE	MAE	RMSE
OLS	346	14.0	18.6
Biased OLS	187	9.9	13.7
Biased OLS + Location	167	9.0	12.9
Biased OLS + Location + Poly	141	8.5	11.9
Biased OLS + Location + Poly + Sin	132	8.3	11.5

TAKEAWAY

Use feature functions to model non-linear relationships, with a linear model.

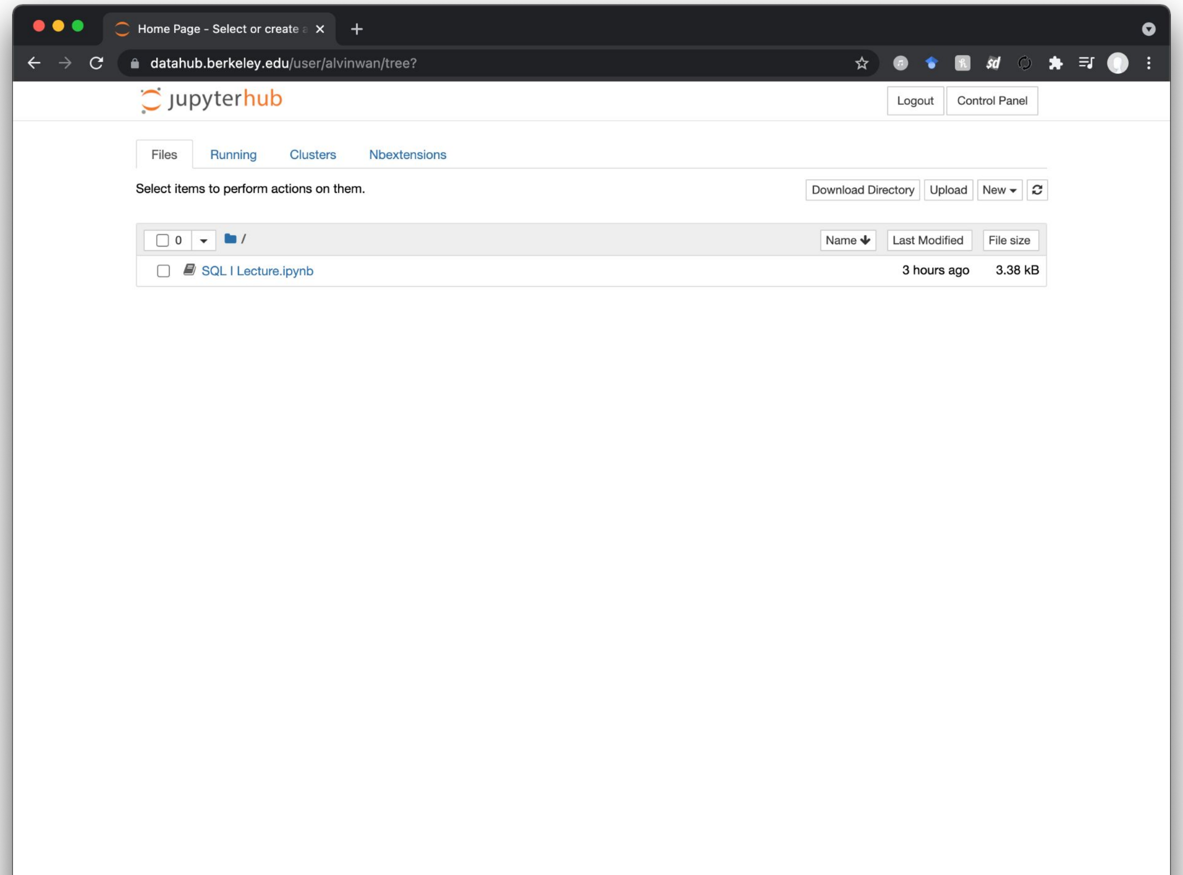
LECTURE 16

Imputing Data with Scikit-Learn

Learn how to impute data quickly with scikit-learn

Data 100/Data 200, Fall 2021 @ UC Berkeley
Fernando Pérez and Alvin Wan
(content by Alvin Wan, John DeNero, Joseph E. Gonzalez, Josh Hug)

data100.datahub.berkeley.edu



Load Dataset

Track Models

Imputing Data

Stable Feature Functions

Scikit-learn Model Imputer

Load Dataset

Track Models

Imputing Data

Stable Feature Functions

Scikit-learn Model Imputer

Load Dataset

Track Models

Imputing Data

Stable Feature Functions

Scikit-learn Model Imputer

Load Dataset

Track Models

Imputing Data

Stable Feature Functions

Scikit-learn Model Imputer

Load Dataset
Track Models
Imputing Data
Stable Feature Functions
Scikit-learn Model Imputer

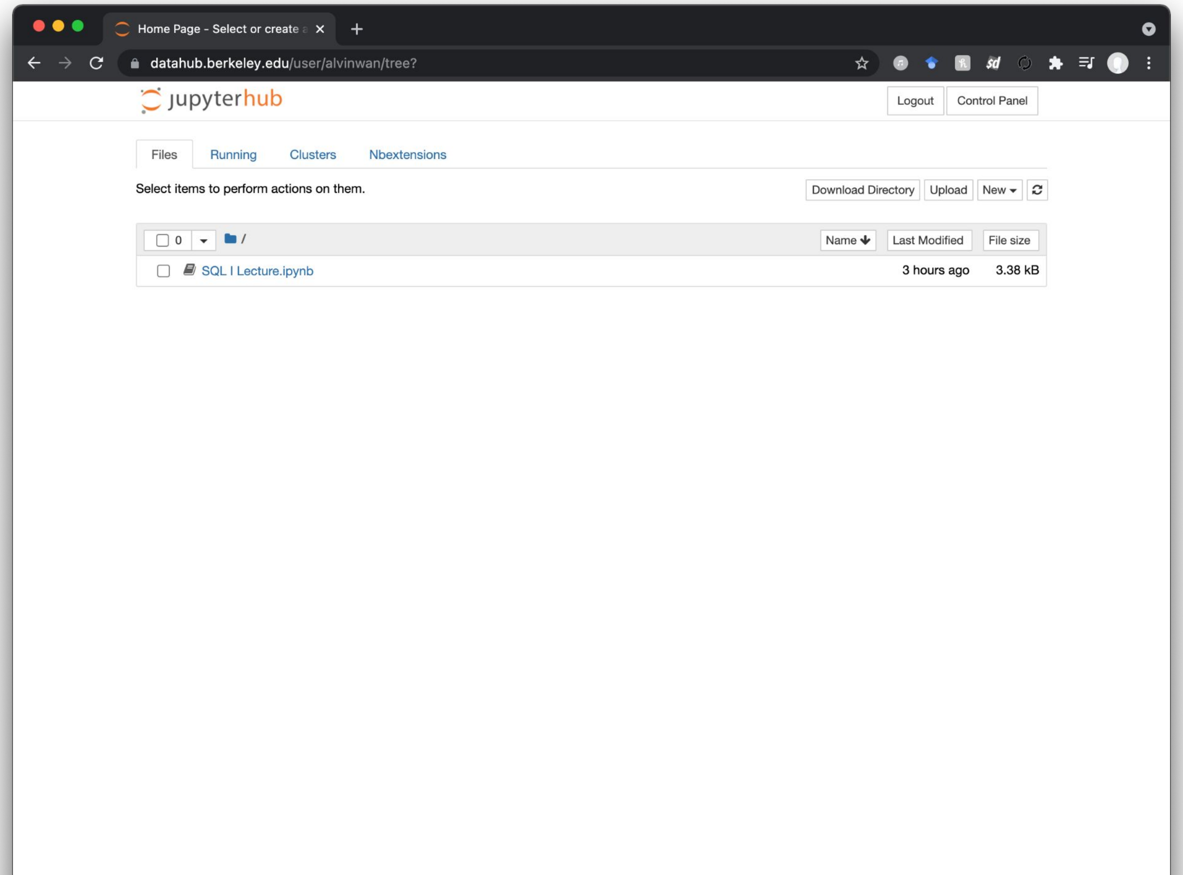
LECTURE 16

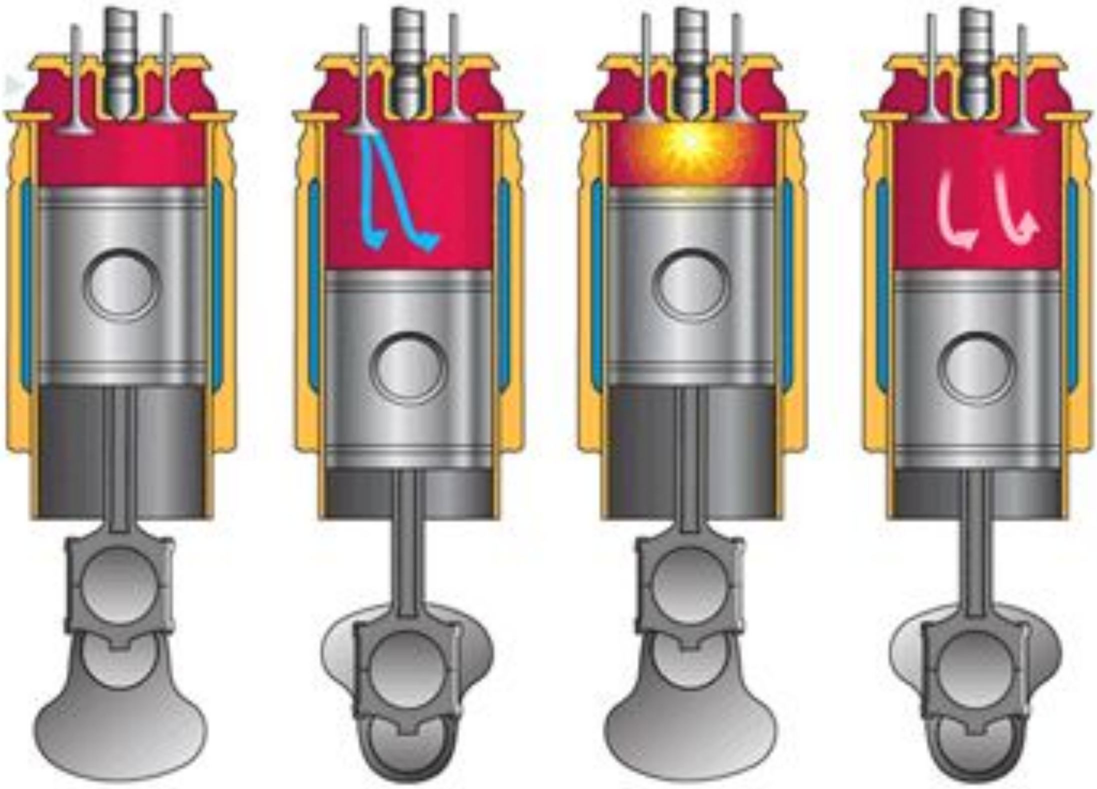
Benefit #2 Applying Domain Knowledge

Leverage insight into the problem to improve model performance

Data 100/Data 200, Fall 2021 @ UC Berkeley
Fernando Pérez and Alvin Wan
(content by Alvin Wan, John DeNero, Joseph E. Gonzalez, Josh Hug)

data100.datahub.berkeley.edu





Displacement Features

Polynomial Features

Sinusoidal Features

Displacement Features

Polynomial Features

Sinusoidal Features

Displacement Features

Polynomial Features

Sinusoidal Features

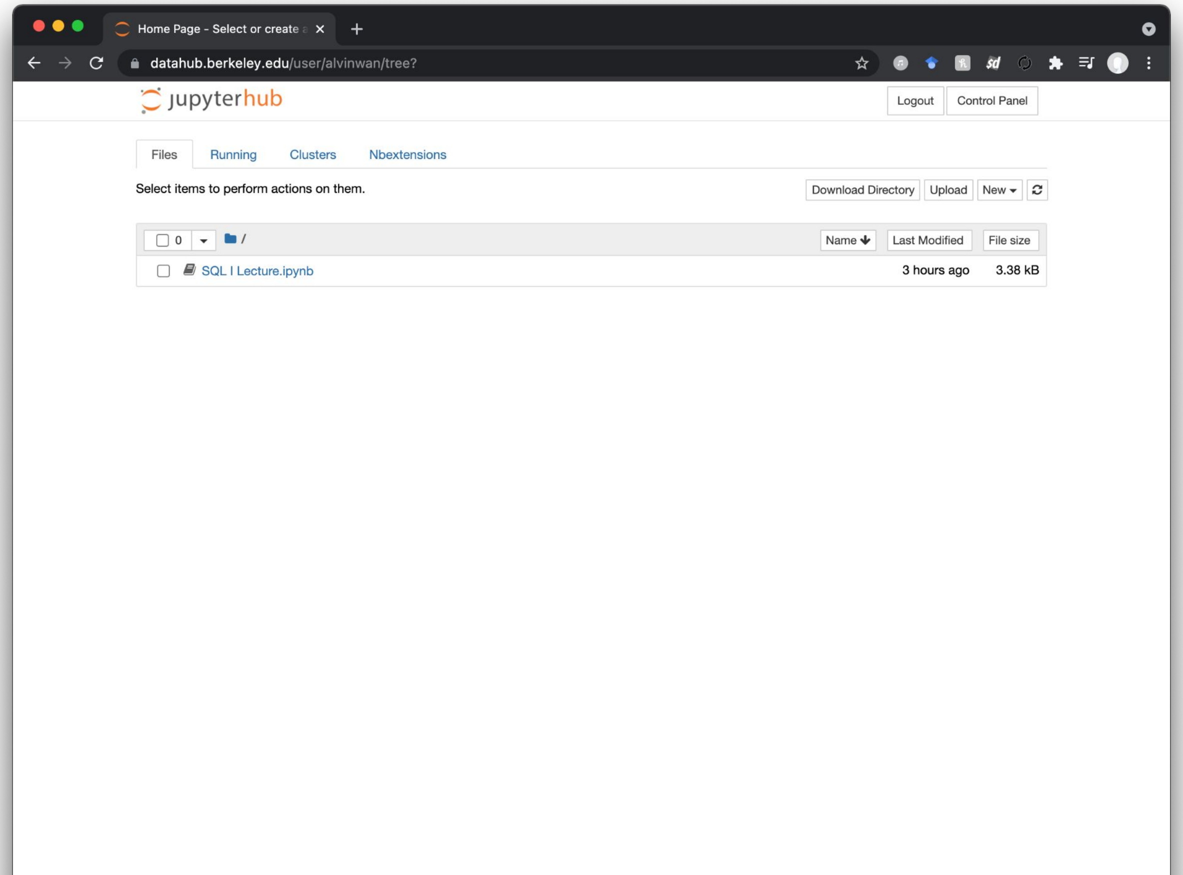
LECTURE 16

Benefit #3 Encoding Non-Numeric, Categorical Data

Dealing with data that is not quantitative continuous

Data 100/Data 200, Fall 2021 @ UC Berkeley
Fernando Pérez and Alvin Wan
(content by Alvin Wan, John DeNero, Joseph E. Gonzalez, Josh Hug)

data100.datahub.berkeley.edu



Categorical Data

Text as Bag-of-Words

Text as N-Grams

Uh Oh

Encoding Categorical Data?

state
NY
WA
CA

?

state
0
1
2

INTUITION

Given k categories, represent each category as a unit vector an axis-aligned vector, in k-dimensional space.

One-Hot (Dummy) Encoding

state
NY
WA
CA

AK	…	CA	…	NY	…	WA	…	WY
0	…	0	…	1	…	0	…	0
0	…	0	…	0	…	1	…	0
0	…	1	…	0	…	0	…	0

One-Hot (Dummy) Encoding

pandas

`pd.get_dummies`

sklearn

`sklearn.preprocessing.OneHotEncoder`

TAKEAWAY

For categorical data, use the **one-hot encoding**.

Pro tip: In particular, use **`sklearn.preprocessing.OneHotEncoder`** instead of panda's **`pd.get_dummies`**.

Categorical Data

Text as Bag-of-Words

Text as N-Grams

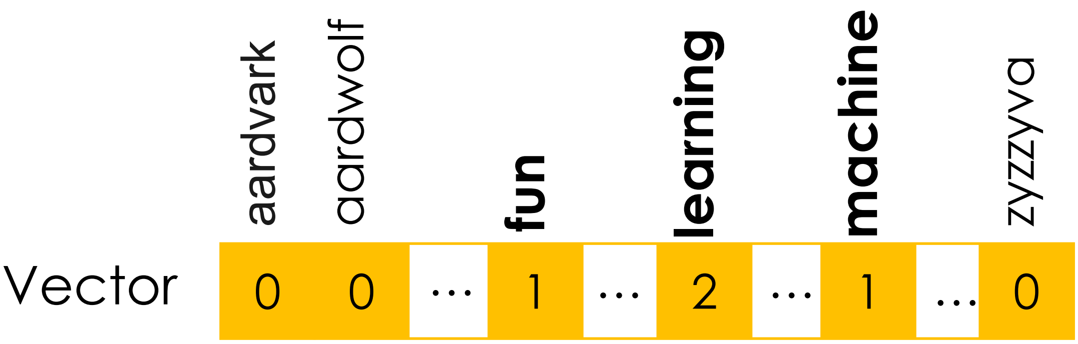
Uh Oh

INTUITION

Generalize one-hot encoding to sequences of "categories". Add 1 for each occurrence of a word.

Bag of Words Encoding

Learning about machine learning is fun.



Bag of Words Issues

Word order lost

Be aware: Acceptable for some tasks

Potentially really long vector

Solution: Use sparse encoding

Unseen words at test time

Hack: Drop unseen words

Categorical Data

Text as Bag-of-Words

Text as N-Grams

Uh Oh

Order Matters

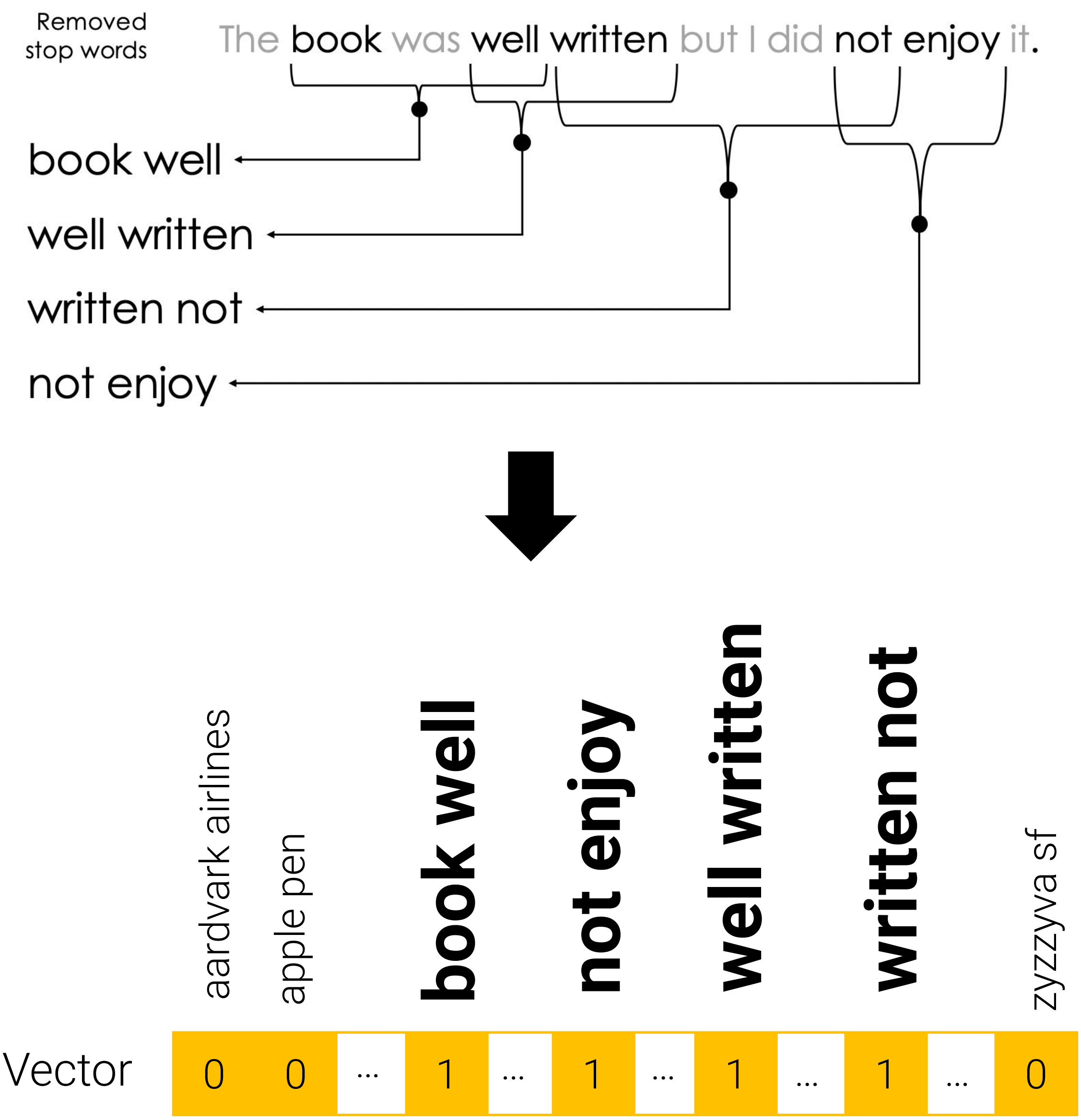
*The book was **not** well written but I did enjoy it.*

*The book was well written but I did **not** enjoy it.*

INTUITION

Capture bags of sequences-of-words, instead, to preserve order information.

2-Gram Encoding



Categorical Data
Text as Bag-of-Words
Text as N-Grams
Uh Oh