Python for Building & Tuning ML Models

DataWeek, Day #3
August 2025

Notebook: bit.ly/3HGKBDB

Sponsored by CUNY 2X Tech, NYC Tech Talent Pipeline, & Hunter College Computer Science

While waiting for the workshop to begin...

- Today, we're building models using Python and Scikit-Learn and visualizations using Seaborn. Open in Colab, CoCalc (or your favorite IDE):

  today's notebook

- Today's datasets:
  - Titanic & Tips (seaborn)
  - Federal Reserve Economic Data (FRED): cookie pricing (linked in notebook)
  - NIST Handwritten Digits (sklearn)

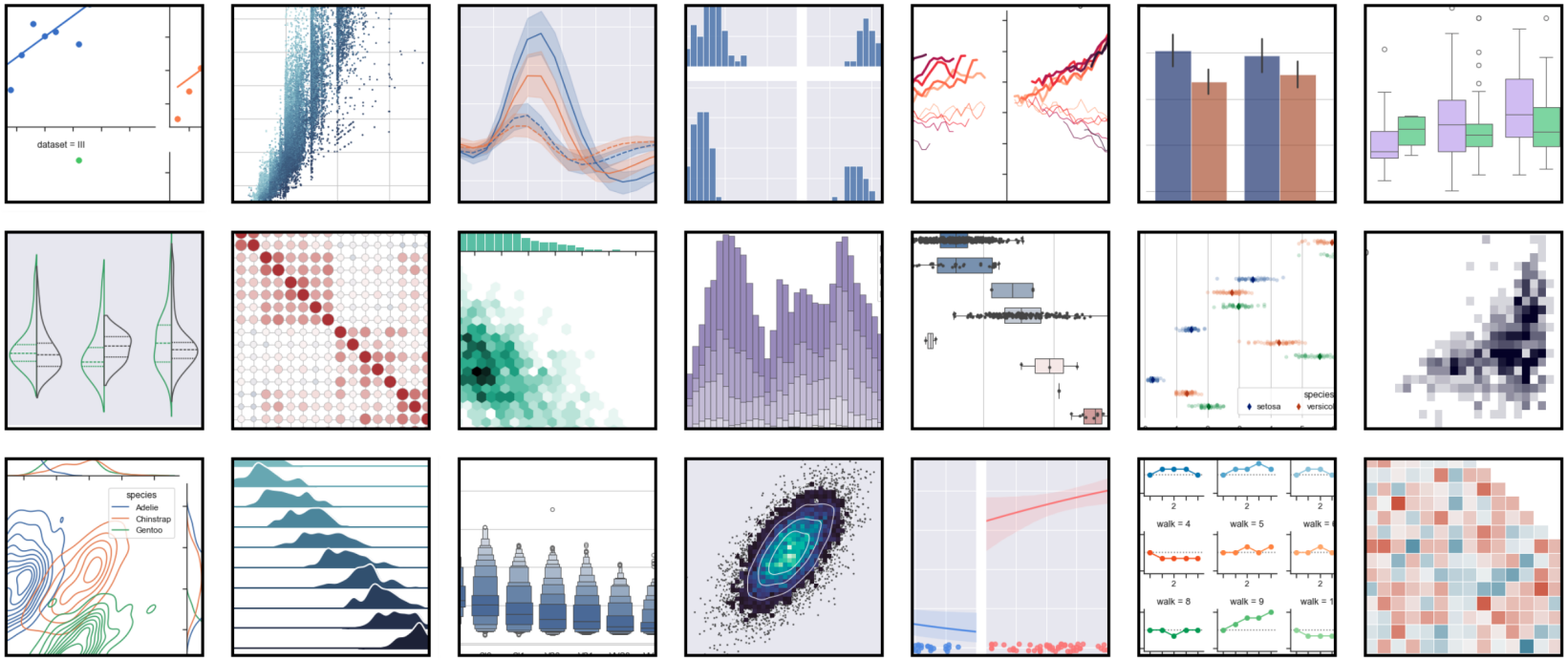
Outline

- Morning:
  - Seaborn
  - Models & Machine Learning
  - Regression
- Afternoon:
  - Training Models
  - Scikit-Learn
  - Classifiers
- Wrap Up

Outline

- Morning:
  - Seaborn
  - Models & Machine Learning
  - Regression
- Afternoon:
  - Training Models
  - Scikit-Learn
  - Classifiers
- Wrap Up

Visualization: `seaborn` & `matplotlib`



seaborn gallery

- Quick overview of visualizations using seaborn which is built on top of matplotlib.
- Notebook: What can you say about the data:

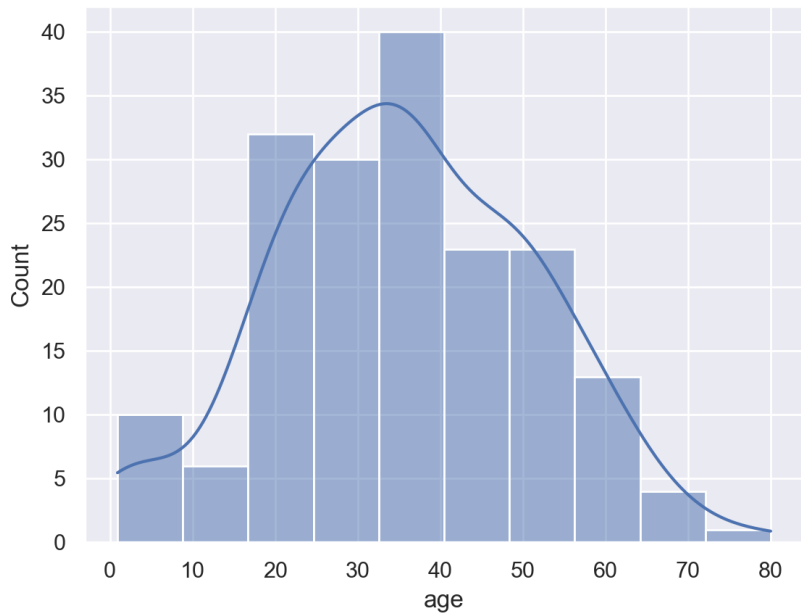
```
ti = sns.load_dataset('titanic').dropna().reset_index(drop=True)
```

- Notebook: We'll use different vis functions to explore the data.

Visualizing Quantitative Data: Histograms

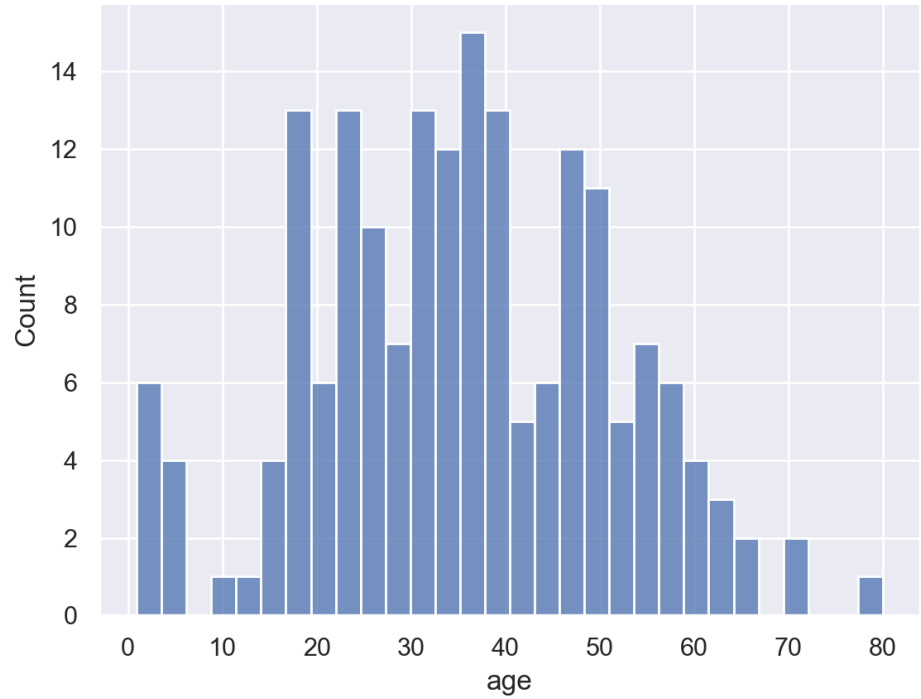
- Let's start by building a histogram with the function `sns.histplot()`:

```
sns.histplot(data = ti['age'], kde=True)
plt.show()
```



- `kde` parameter controls if the "kernel density estimate" is visible.

Visualizing Quantitative Data: Histograms



- Can also how the data is divided (as well as shading, weights, and standard plt parameters):

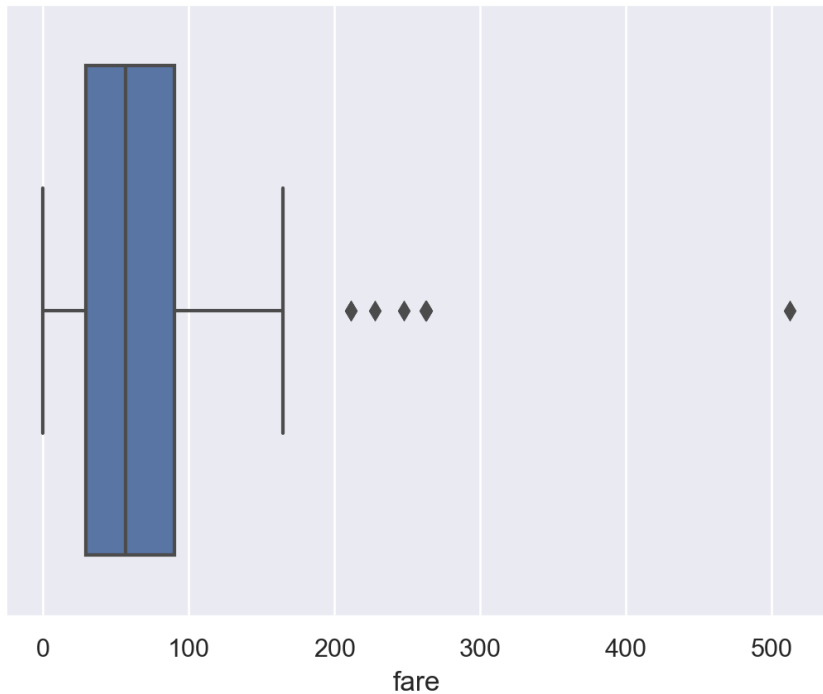
```
sns.histplot(data = ti['age'], bins=30)
plt.show()
```

- Notebook: What can we find out about who was aboard?

Visualizing Quantitative Data: Box Plots

- Box plots show where most of the data is.
- Usually use 25th and 75th percentiles of the box with "whiskers" capturing rest of data (modulo outliers).

```
sns.boxplot(x='fare', data=ti)
plt.show()
```



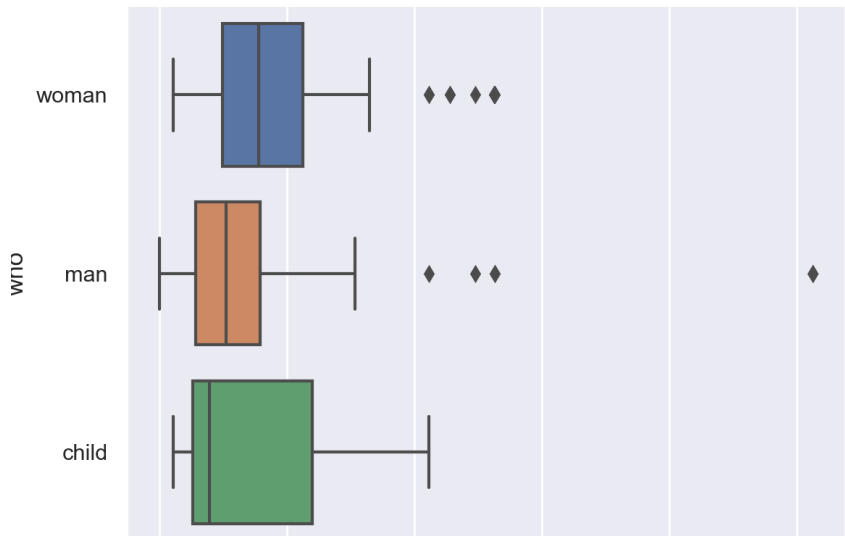
Visualizing Quantitative Data: Box Plots

- Inter-Quartile Range (IQR) used to decide outliers. The difference between the 75th percentile of the data and the 25th percentile:

```
lower, upper = np.percentile(ti['fare'], [25, 75])
iqr = upper - lower
iqr
```

- Can display box plots for different categories:

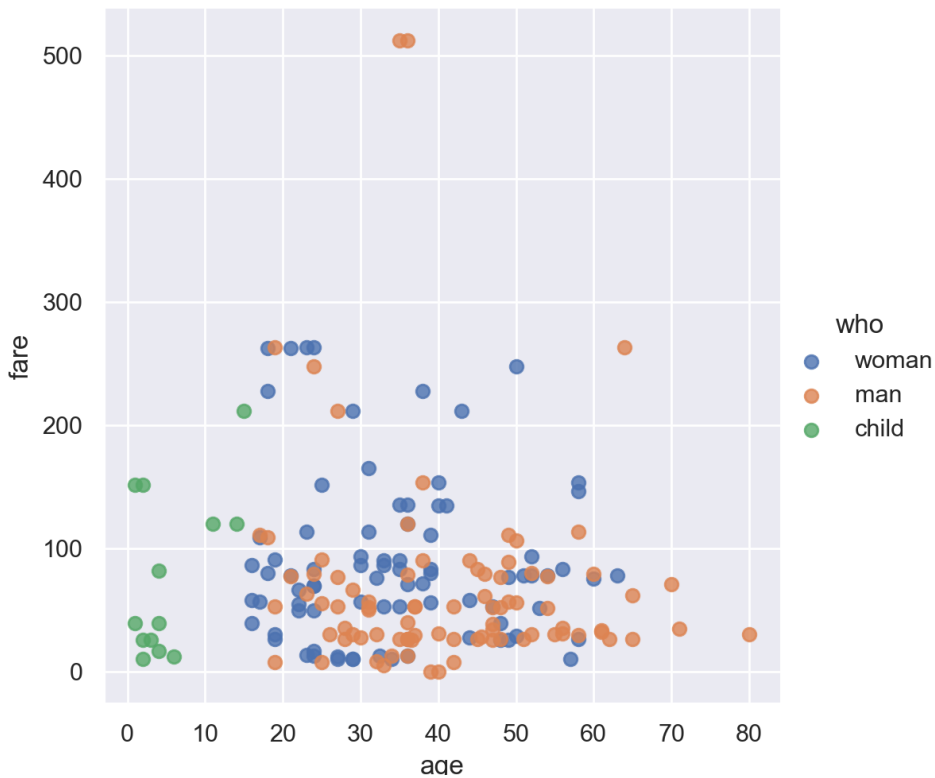
```
sns.boxplot(x='fare', y='who', data=ti)
plt.show()
```



Visualizing Quantitative Data: Scatter Plots

- Scatter plots are used to compare two quantitative variables.
- Will use seaborn's `lmplot()` which also allows regression lines (more on these later) in the plots:

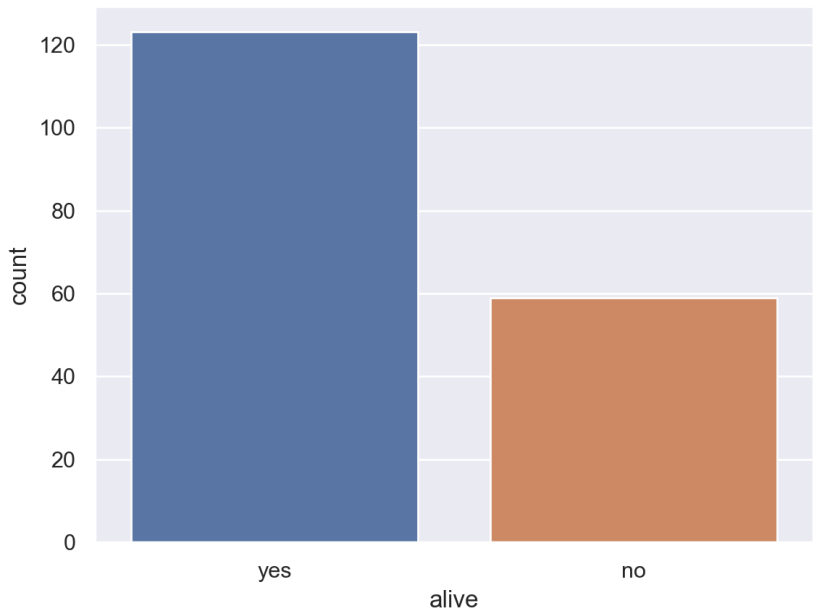
```
sns.lmplot(x='age', y='fare', hue='who', data=ti, fit_reg=False)
plt.show()
```



Visualizing Qualitative Data

- For qualitative or categorical data, we most often use bar charts and dot charts (described in text).
- Counting the number of survivors:

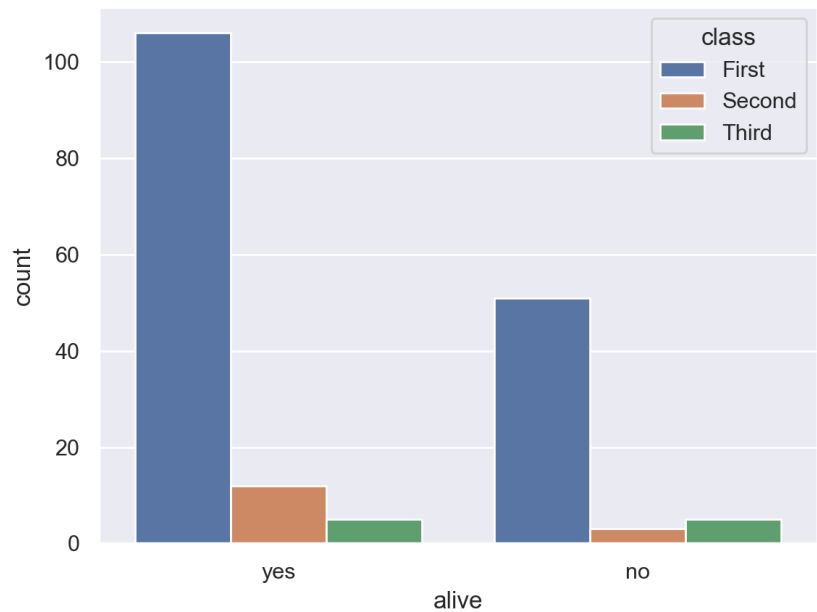
```
sns.countplot(x='alive', data=ti)
plt.show()
```



Visualizing Qualitative Data

- For qualitative or categorical data, we most often use bar charts and dot charts (described in text).
- Counting the number of survivors, refining by class:

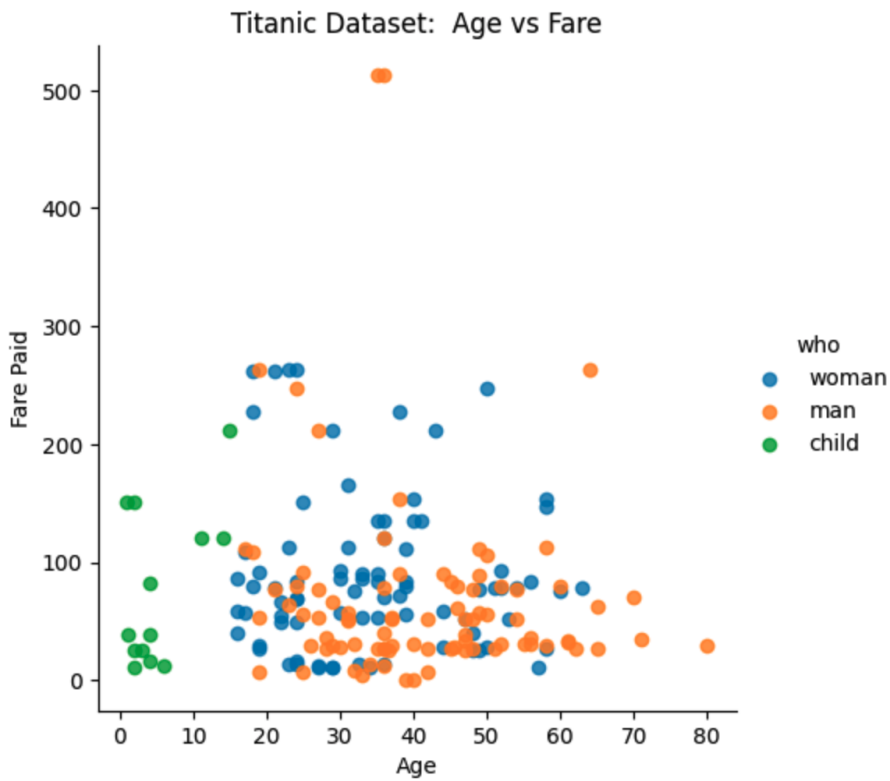
```
sns.countplot(x='alive',hue='class', data=ti)
```



Adding Labels & Titles

- Adding titles and labels is very useful for multiple analysis and sharing.
- Seaborn is built on top of the powerful `matplotlib.pyplot` (`plt`).
- Can style your figures with `plt` commands:

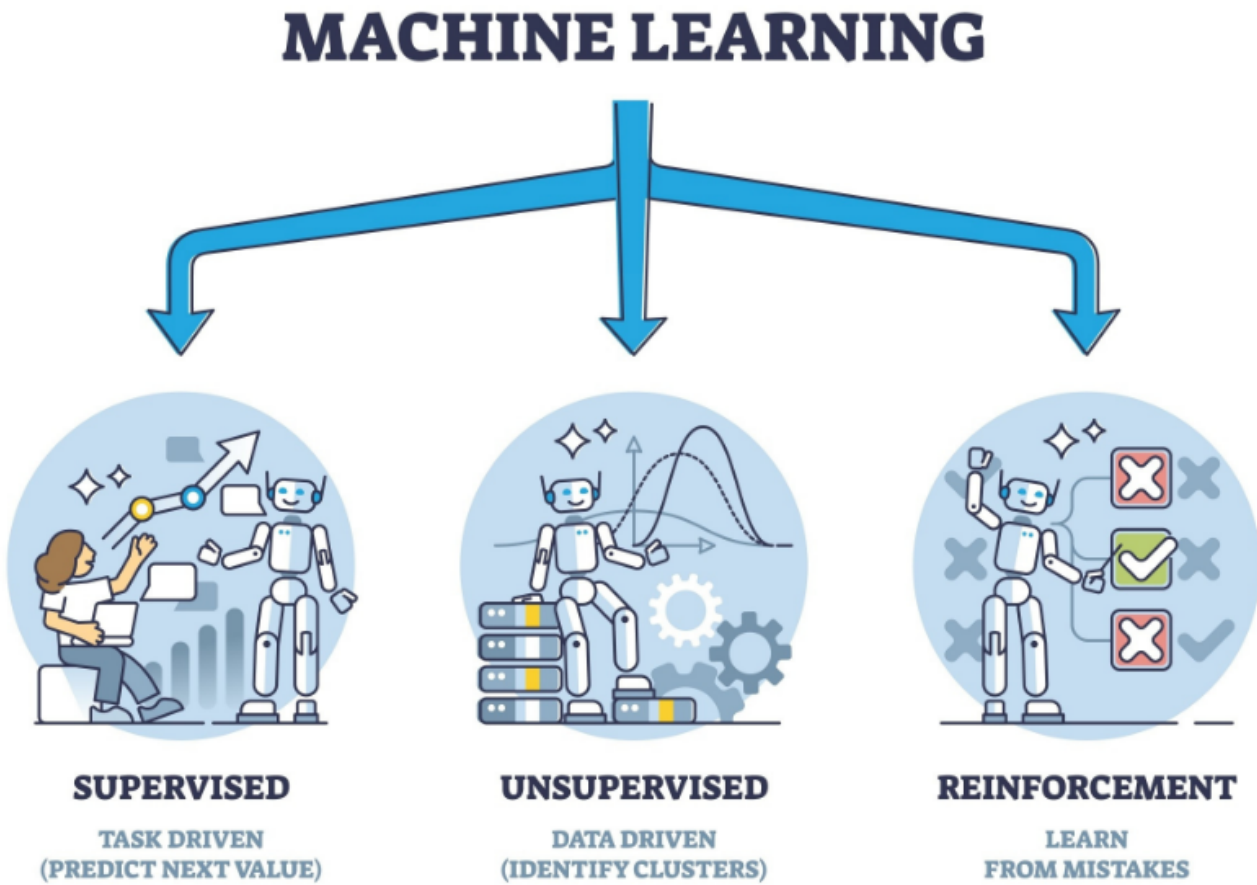
```
sns.lmplot(x='age',y='fare',hue='who',data=ti,fit_reg=False)
plt.title('Titanic Dataset:  Age vs Fare')
plt.xlabel('Age')
plt.ylabel('Fare Paid')
```



Outline

- Morning:
  - Seaborn
  - Models & Machine Learning
  - Regression
- Afternoon:
  - Training Models
  - Scikit-Learn
  - Classifiers
- Wrap Up

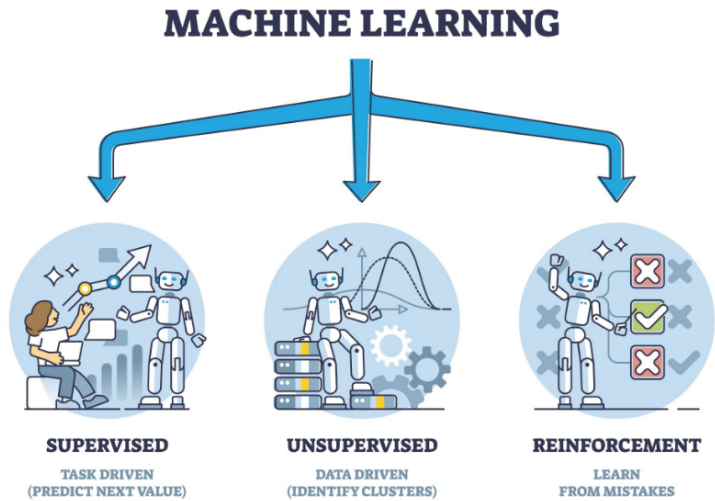
Machine Learning



ntiva.com

- Machine learning: computers "learn", without explicitly coding every step.
- The goal is to find patterns and make inferences that were not seen before.

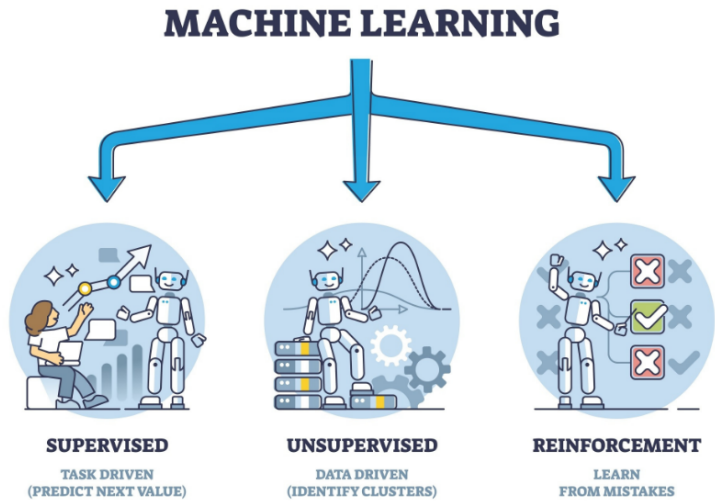
Machine Learning



ntiva.com

- Supervised learning: Labeled datasets (e.g. correct answer known) "supervise" the algorithms into classifying data or predicting outcomes accurately.
- Unsupervised learning: Analyze or cluster unlabeled datasets. Used to discover hidden patterns in data without human supervision.
- Reinforcement learning: Operates on unlabeled data and interactions that reinforce/penalize to build better models.

Machine Learning



ntiva.com

- **TODAY:** Supervised learning: Labeled datasets (e.g. correct answer known) "supervise" the algorithms into classifying data or predicting outcomes accurately.
- Unsupervised learning: Analyze or cluster unlabeled datasets. Used to discover hidden patterns in data without human supervision.
- Reinforcement learning: Operates on unlabeled data and interactions that reinforce/penalize to build better models.

Modeling & Estimation

Essentially, all models are wrong, but some are useful.
— George Box, Statistician (1919-2013)

- A model is an idealized representation of a system.
- Examples: weather forecasts make predictions that are often correct but sometimes not.
- Today:
  - Introduce some basic models.
  - Introduce loss minimization to evaluate & tune models.
  - Introduce supervised learning techniques to build models to classify data.

Restaurant Tips

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
...	...	...	...	...	...	...	...
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

Open today's notebook:

- What's in the dataset?
- Can you predict tips?

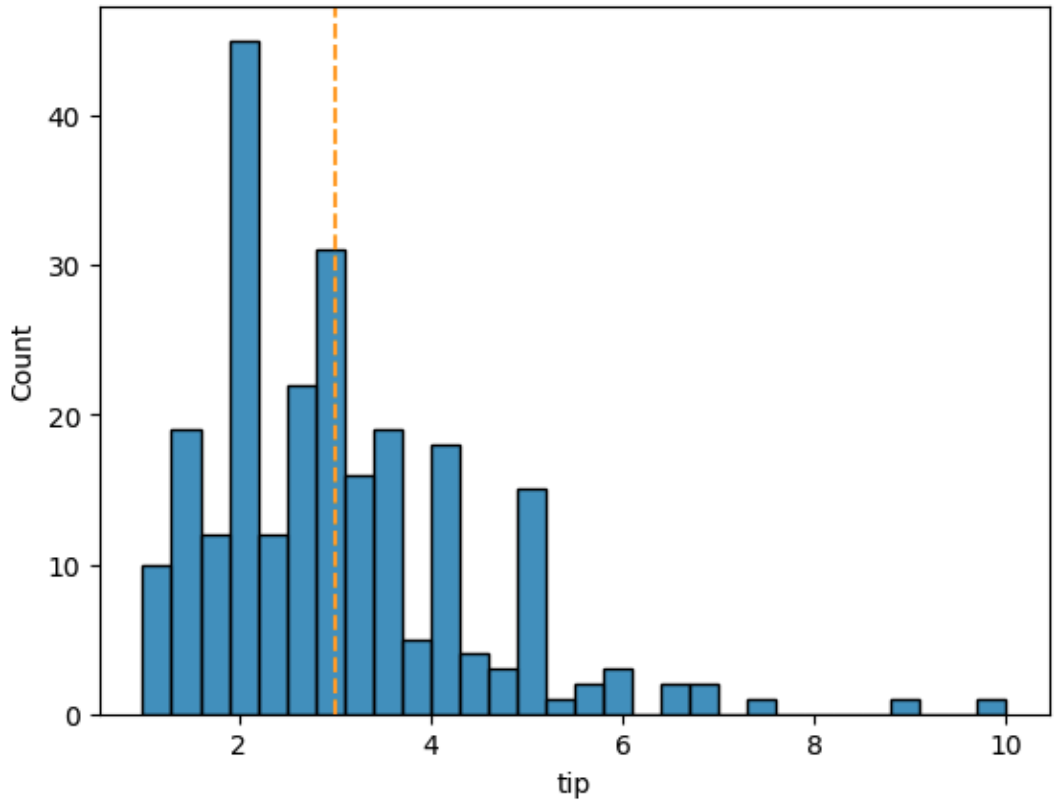
Constant Model

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
...	...	...	...	...	...	...	...
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

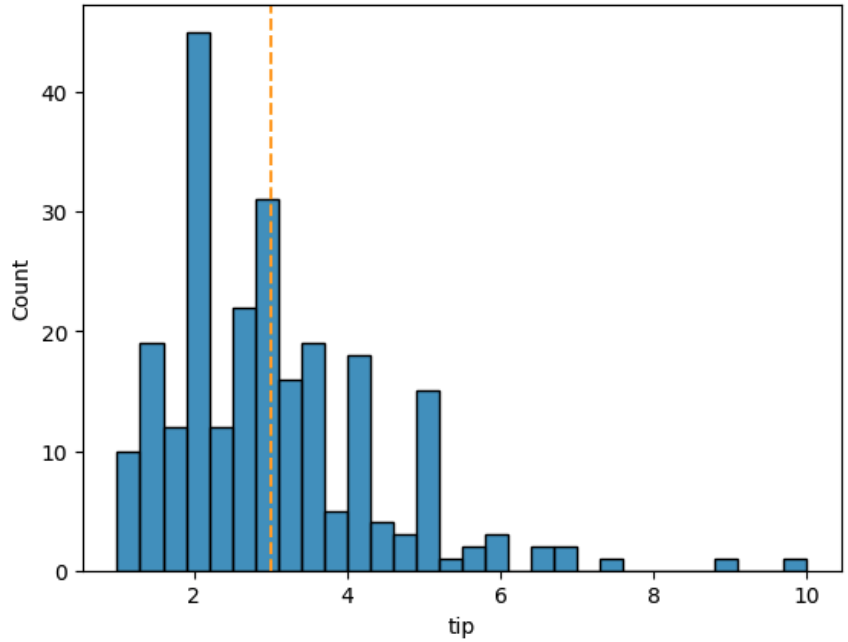
- Our first model: no matter what the input, we predict a tip of \$3: $\Theta = 3$.
- Called a constant model since it doesn't change.
  - Pros: Very easy to compute!
  - Cons: Doesn't do well for very large parties.
- Can you build a better constant model?
  - Is $\theta = 2$ or $\theta = 5$ better?
- Need to agree on what *better* means– for ML, we use *loss functions.*

Loss Functions



- To quantify how good an estimate is, we will use loss functions.
- A loss function
  - takes in an estimate θ and the points in our dataset, $y_1, y_2, \ldots, y_n$, and
  - outputs a single number, the loss, that measures how well θ fits the data.
- The choice of loss function affects the downstream analysis.

Popular Loss Functions



- To compare different size samples, return the average:
  - Mean Absolute Error: Use average of absolute value of the differences:

$$f(\theta, y_1, y_2, \ldots, y_n) = \frac{1}{n}(|y_1 - \theta| + |y_2 - \theta| + \ldots + |y_n - \theta|)$$

  - Mean Squared Error: Use average of the square the differences:

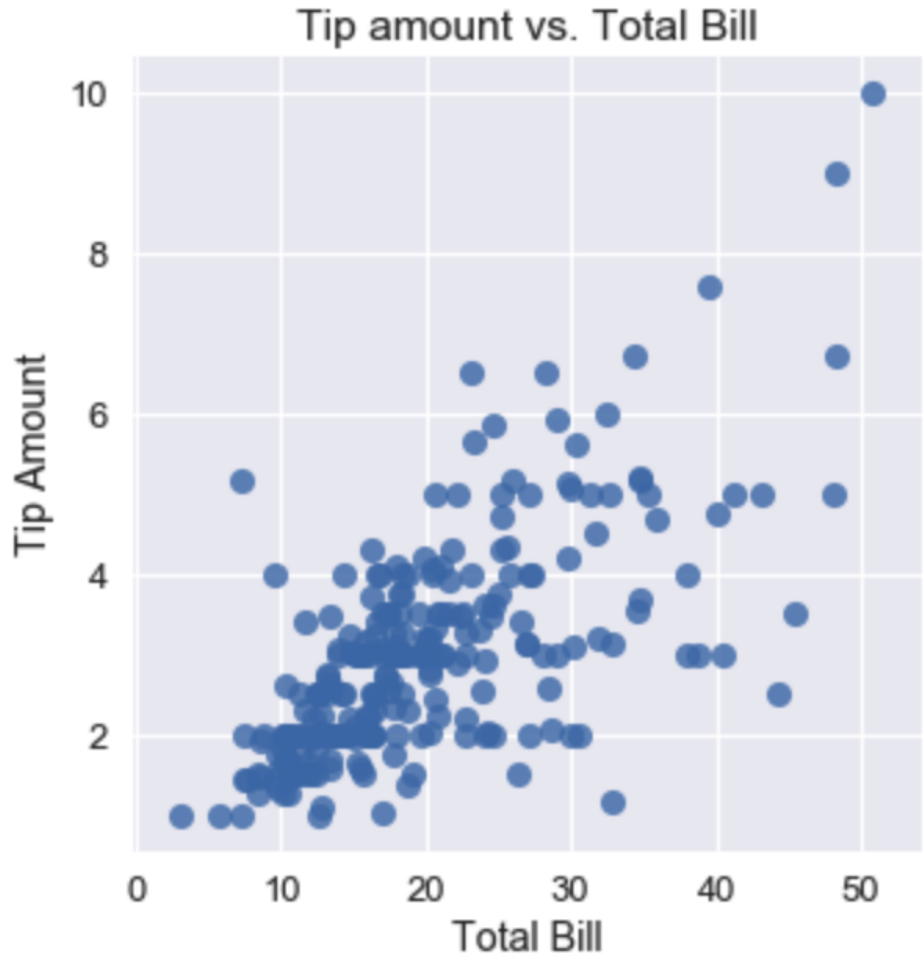
$$f(\theta, y_1, y_2, \ldots, y_n) = \frac{1}{n}((y_1 - \theta)^2 + (y_2 - \theta)^2 + \ldots + (y_n - \theta)^2)$$

- Notebook: Try the sklearn function, so, don't need to compute by hand!

Outline

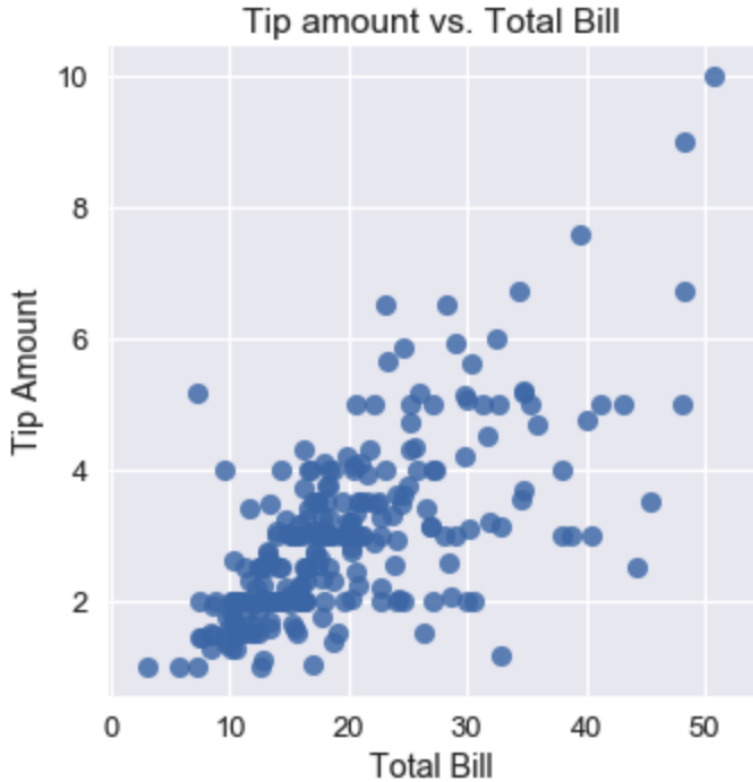
- Morning:
  - Seaborn
  - Models & Machine Learning
  - Regression
- Afternoon:
  - Training Models
  - Scikit-Learn
  - Classifiers
- Wrap Up

Restaurant Tips



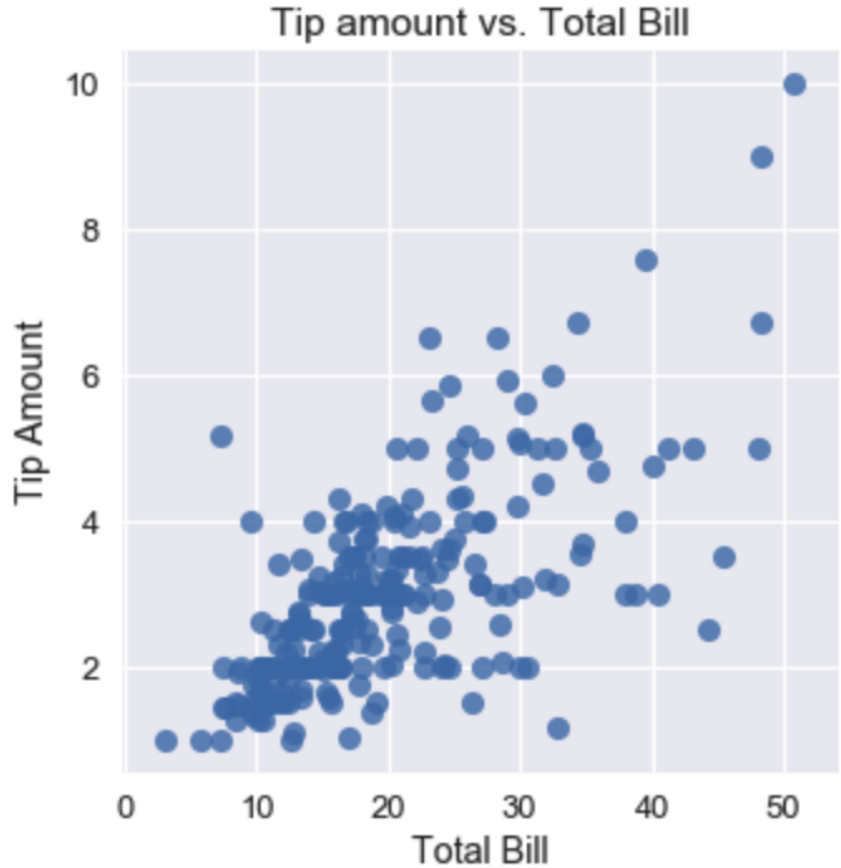

```
sns.lmplot(x='total_bill', y='tip', data=tips, fit_reg=False)
plt.title('Tip amount vs. Total Bill')
plt.xlabel('Total Bill')
plt.ylabel('Tip Amount');
```

Regression



- A regression analysis determines the relationships between a dependent variable (often called **Y**) and an independent variable(s) (often called **X**).
- The results are often used for prediction.
- It is a simple example of supervised machine learning.

Linear Regression

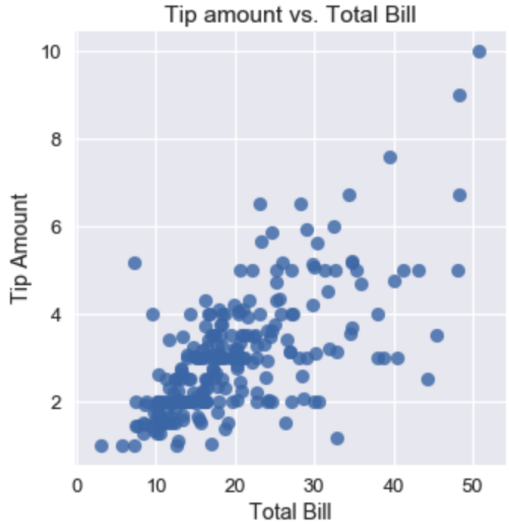


- Goal: find a linear relationship between the independent and dependent variables:

$$y = mx + b$$

that minimizes the loss function & we can use to make predictions.

Linear Models

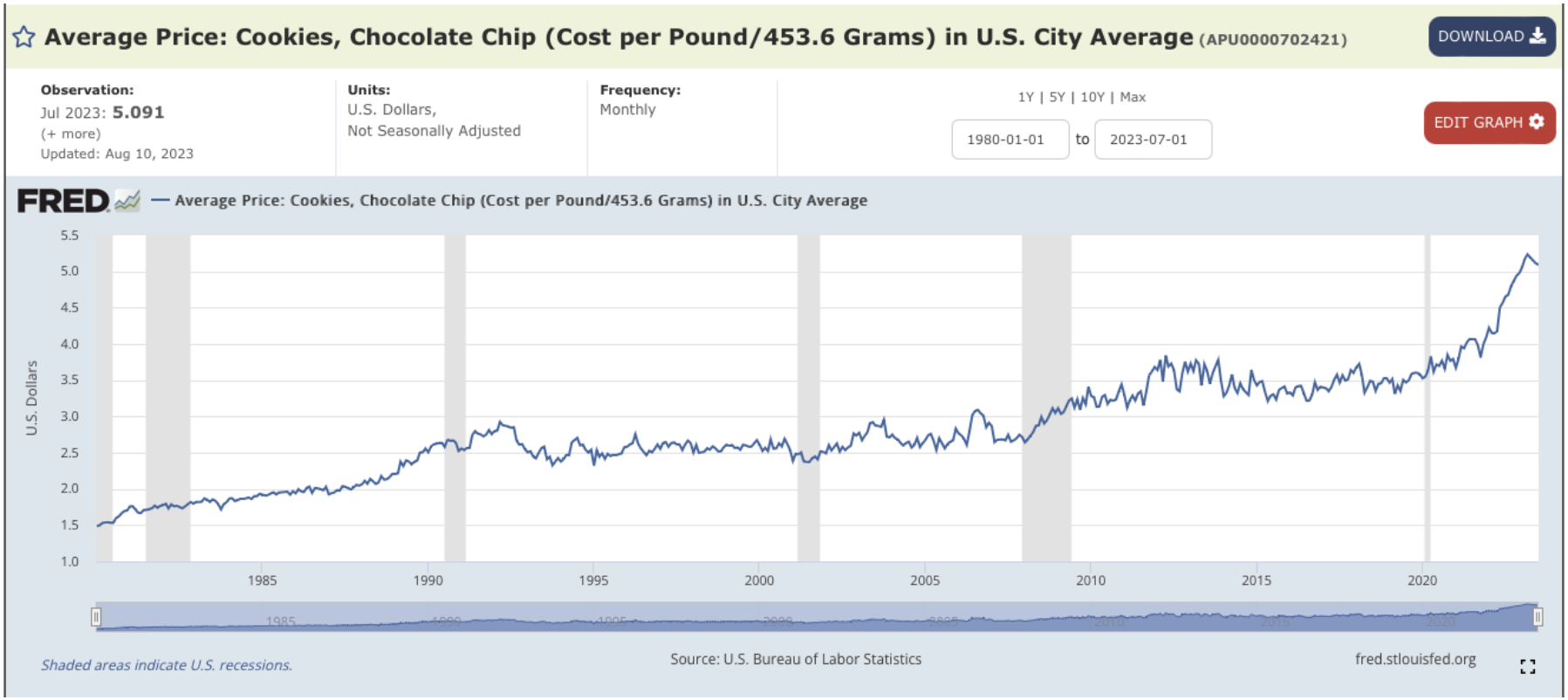


- Goal: construct models to make predictions.
- First: focus on models which seek to explain outputs (dependent variable) via a linear function of the inputs (independent variable).
- Example: If the bill is \$20, what will the tip be?

  Are there values for `m` & `b` such that: `tip` $=$ `m`$\cdot$`bill` $+$ `b`?

  One model: `m`$=15\%$ and `b` $=0$: `tip` $=$ `0.15`$\cdot$`bill` $+$ `0`
  Another model: `m`$=10\%$ and `b` $=2$: `tip` $=$ `0.10`$\cdot$`bill` $+$ `2`
  Third model: `m`$=-2\%$ and `b` $=100$: `tip` $=$ `-0.02`$\cdot$`bill` $+$ `100`
- `bill` is the independent variable, and `tip` is the dependent variable.
- Notebook: find the best line to fit the data.

Case Study: Federal Reserve Pricing Data



Notebook: Work through the case study on food pricing & unemployment rates.

Outline

- Morning:
  - Seaborn
  - Models & Machine Learning
  - Regression
- Afternoon:
  - Training Models
  - Scikit-Learn
  - Classifiers
- Wrap Up

Training Models

- Example: build a model to classify apples by variety.
- Training:
  - Want to use as much training data as possible, but
  - Also want to test competing models to choose best.

Training Models

- Train model(s) to classify apples by variety:
  - Want to use as much training data as possible, but
  - Also want to test competing models to choose best.
- Solution:
  - Randomly split the data into train and test sets.
  - Hide the test set and only use the train set to fit the model.
  - Once done training, test the models predictive power on the test set.

Outline

- Morning:
  - Seaborn
  - Models & Machine Learning
  - Regression
- Afternoon:
  - Training Models
  - Scikit-Learn
  - Classifiers
- Wrap Up

Scikit-learn: Linear Models

scikit-learn.org

- Toolkit for predictive data analysis.
- Built on NumPy, SciPy, and matplotlib.
- Open source, commercially usable (BSD license).
- Easy 'building blocks' for analysis, many tutorials & recipes.

Building Models: General Recipe

From sklearn linear regression example:

```
X = tips[['total_bill']]
y = tips['tip']
from sklearn.model_selection import train_test_split
```

scikit-learn.org

1. Split your data into pieces for training & testing.

   ```
   X_train,X_test,y_train,y_test =
   train_test_split(X,y,test_size=0.33, random_state=42)
   ```

2. Select a model.

   ```
   reg = linear_model.LinearRegression()
   ```

3. Select a loss function. (For linear regression, almost always use RMSE.)

4. Minimize the loss function, using training data.

   ```
   reg.fit(X_train,y_train)
   ```

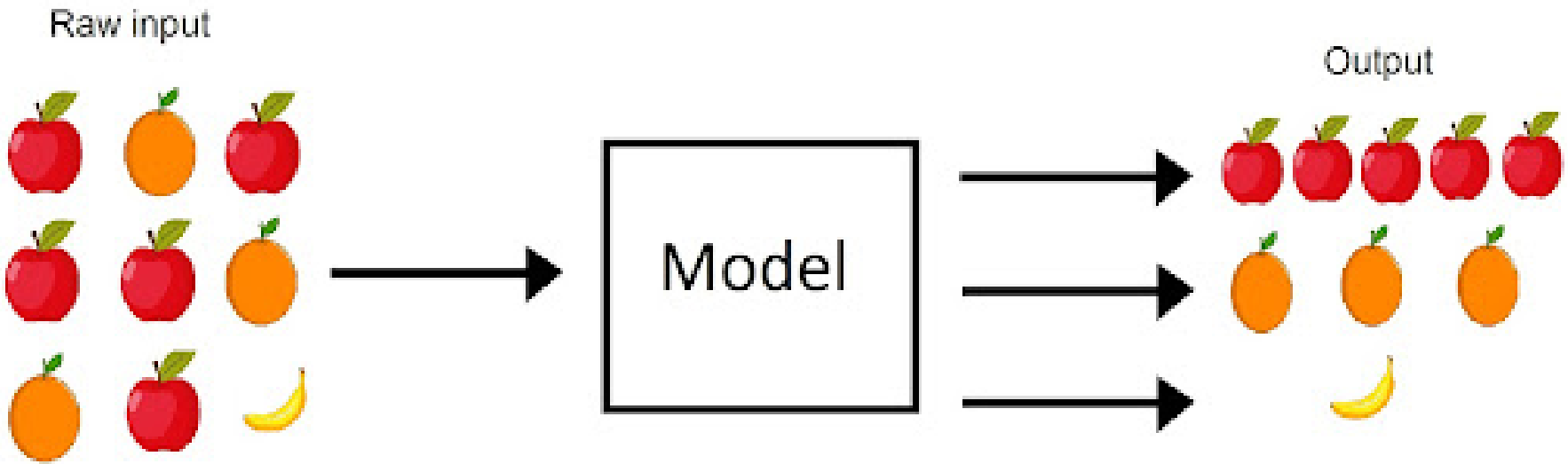
5. Validate your model with reserved test data. Compute & assess results:
   ```
   y_pred = reg.predict(X_test)
   ```

Notebook: practice recipe on regression example.

Outline

- Morning:
  - Seaborn
  - Models & Machine Learning
  - Regression
- Afternoon:
  - Training Models
  - Scikit-Learn
  - Classifiers
- Wrap Up

Classifers



engo645-assignments.readthedocs.io

Goal: train a model to classify inputs into different categories.

Spam Classifier

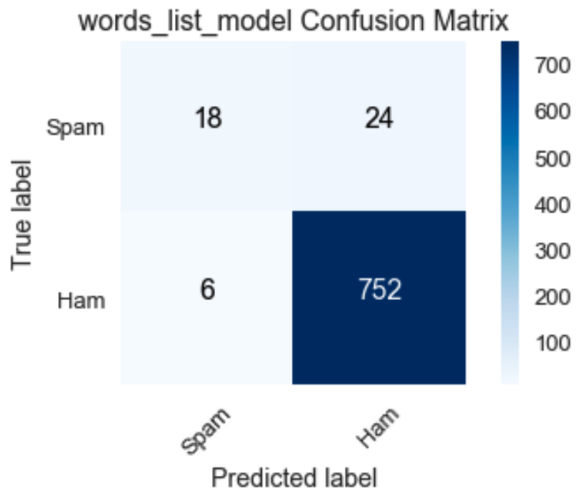
```
1.3. Format
-----------

The files contain one message per line. Each line is composed by two columns: one with label (ham
or spam) and other with the raw text. Here are some examples:

ham   What you doing?how are you?
ham   Ok lar... Joking wif u oni...
ham   dun say so early hor... U c already then say...
ham   MY NO. IN LUTON 0125698789 RING ME IF UR AROUND! H*
ham   Siva is in hostel aha:-.
ham   Cos i was out shopping wif darren jus now n i called him 2 ask wat present he wan lor. Then
he started guessing who i was wif n he finally guessed darren lor.
spam   FreeMsg: Txt: CALL to No: 86888 & claim your reward of 3 hours talk time to use from your
phone now! ubscribe6GBP/ mnth inc 3hrs 16 stop?txtStop
spam   Sunshine Quiz! Win a super Sony DVD recorder if you canname the capital of Australia? Text
MQUIZ to 82277. B
spam   URGENT! Your Mobile No 07808726822 was awarded a L2,000 Bonus Caller Prize on 02/09/03! This
is our 2nd attempt to contact YOU! Call 0871-872-9758 BOX95QU

Note: messages are not chronologically sorted.
```

UCI ML Data Archives

- Build a classifer to identify spam email vs. non-spam (aka ham).
- Many approaches to this classic problem.
- Measure solutions on how often the model predicts correctly and incorrectly, often represented as a confusion matrix (next slide).

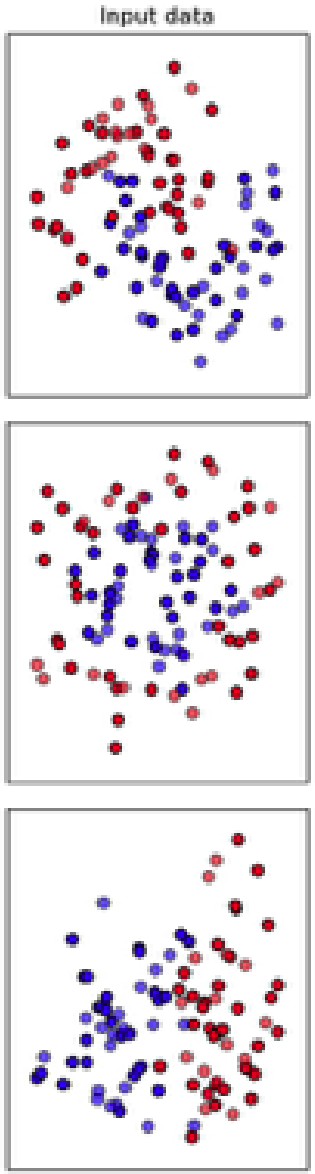
Confusion Matrices

True Positive	False Negative
False Positive	True Negative



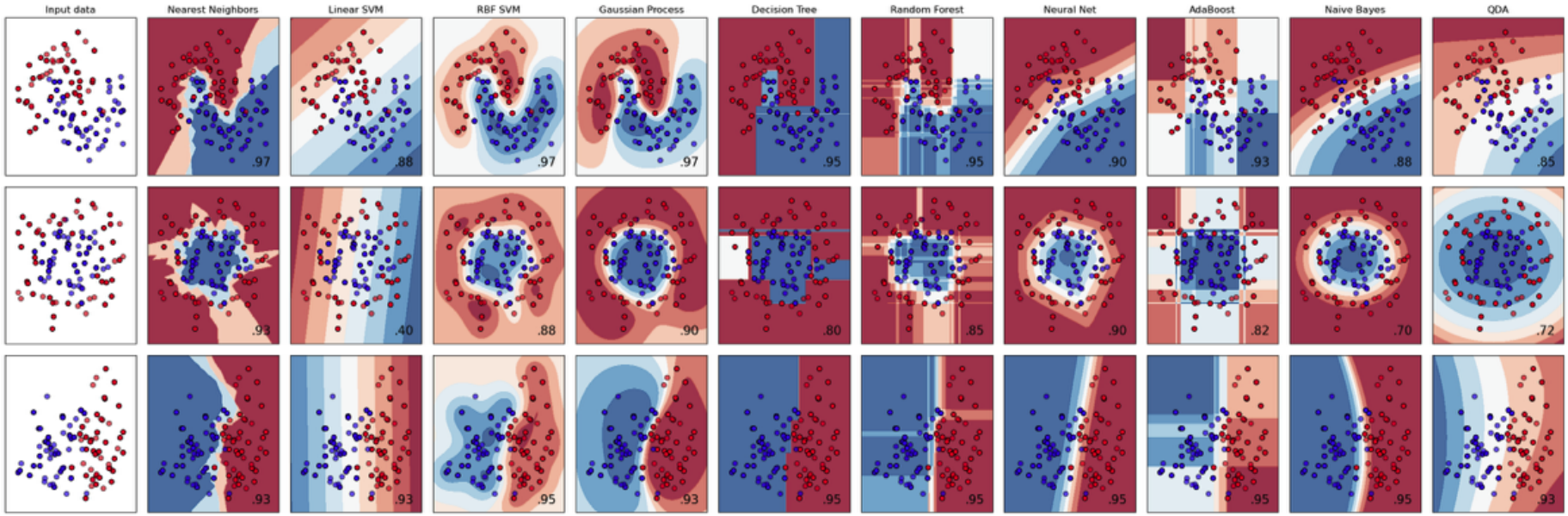
- Standard metrics: measure how often the model predicts correctly and incorrectly.
- Many are combination of 4 basic measures:
  - True Positive: correctly labeled with positive class.
  - False Negative: belongs to positive class but mislabeled as negative.
  - False Positive: belongs to negative class but mislabeled as positive.
  - True Negative: correctly labeled with negative class.
- Often organized in a confusion matrix: a heatmap of model predictions vs. actual labels.
- See notebook for implementation details.

Classification Survey



- Data from sklearn: Classifier Comparison.
- Goal: classify red versus blue points.
- Training points in solid colors.
- Testing points semi-transparent.
- Many different approaches. We will survey:
  - Discriminative Classification:
    - ★ Support Vector Machines: separates (transformed) space with a line/plane/etc.
  - Generative Classification:
    - ★ Naive Bayes Models: quick, few parameters.
  - Ensemble Methods:
    - ★ Random Forests: uses multiple models (decision trees) to build the classifer.

Classifier Survey



- sklearn: Classifier Comparison: many built-in models you can use, that all follow the same general recipe to use.
- Most of today's code demos are from Data Science Handbook (linked in notebook).

MNIST Handwritten Digits

A selection from the 64-dimensional digits dataset

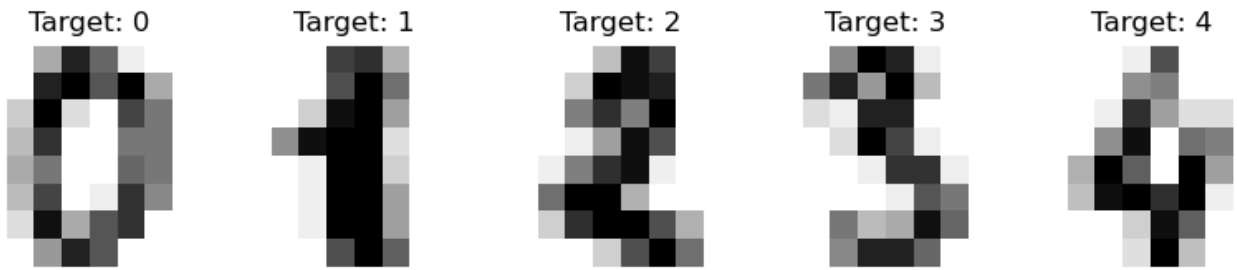
MNIST Handwritten Digits

A selection from the 64-dimensional digits dataset

- MNIST (Modified National Institute of Standards and Technology) Dataset: built into seaborn
- The dataset has 1797 scans of hand-written digits.
- Each entry has the digit represented (target) as well as the 64 values representing the gray scale for the 8 x 8 image.

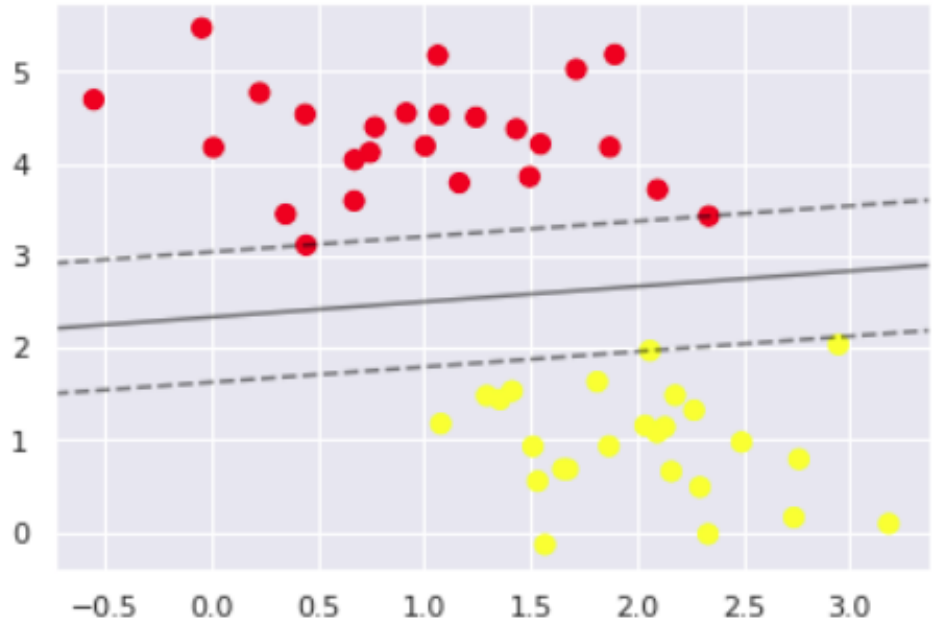
MNIST Handwritten Digits

- Each entry has the digit represented (target) as well as the 64 values representing the gray scale for the 8 x 8 image.
- The first 5 entries are:



- The pixels of the image are numbers between 0 and 15, representing how dark that region is.
- The 64 numbers for the first entry are:
[0. 0. 5. 13. 9. 1. 0. 0. 0. 0. 13. 15. 10. 15. 5. 0. 0. 3. 15. 2. 0. 11. 8. 0. 0. 4. 12. 0. 0. 8. 8. 0. 0. 5. 8. 0. 0. 9. 8. 0. 0. 4. 11. 0. 1. 12. 7. 0. 0. 2. 14. 5. 10. 12. 0. 0. 0. 0. 6. 13. 10. 0. 0. 0.]

Support Vector Machines

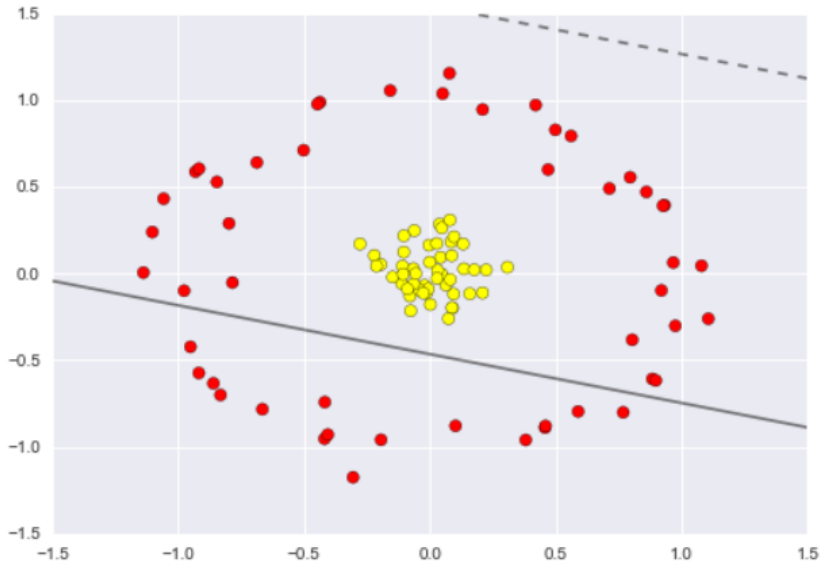


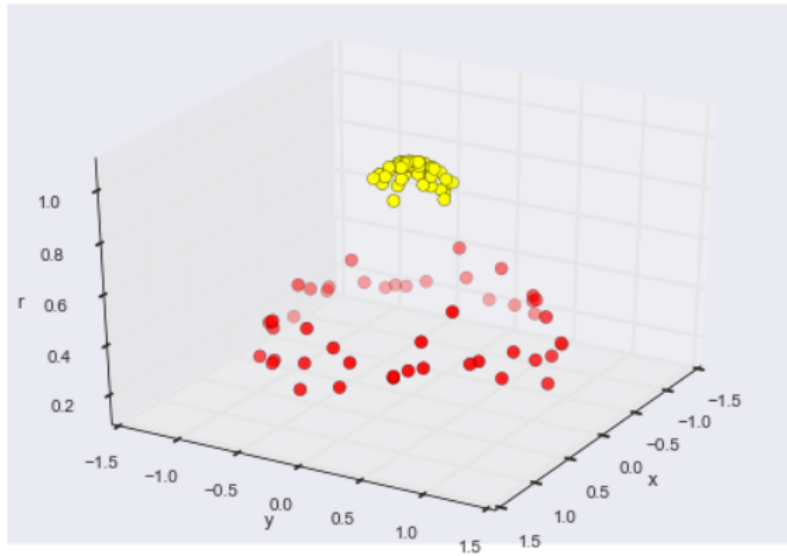
Karpathy's SVM Demo

- Example of discriminative classification.
- Seeks a line/plane/manifold to divide space.
- Employs "kernel trick" to transform space to make division easier.

Classification: Transforming the Input Space

(From Python DS Handbook SVM)



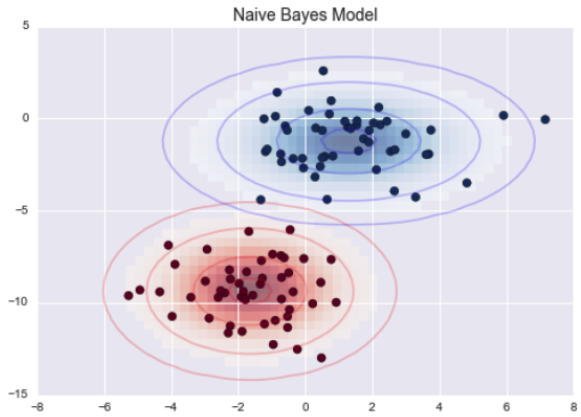


- Transform the data to be separate the sets by line/plane:

```
r = np.exp(-(X ** 2).sum(1))
```

- Can employ a "kernel trick" to be able to query the transformed space without having the huge cost of transforming every point.
- Notebook: SVM on digits & Handbook example.

Naive Bayes Models



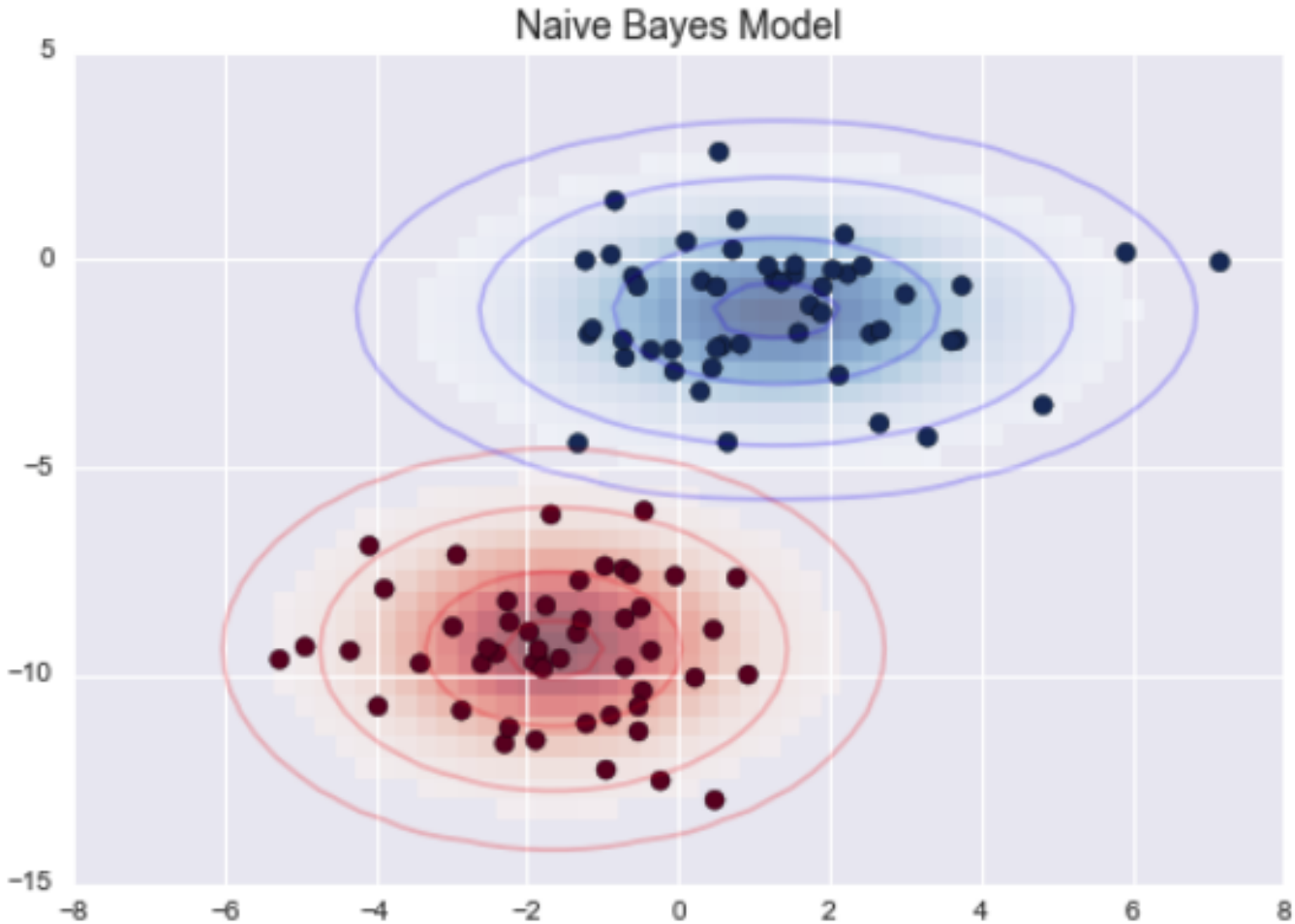
Handbook: Naive Bayes

- Thinks of the data as being generated by a random processes.
- And uses Bayes Theorem: $P(L|features) = \frac{P(features|L)P(L)}{P(features)}$.
- To decide between two labels, use the ratio of posterior probabilities:

$$\frac{P(L1|features)}{P(L2|features)} = \frac{P(features|L1)}{P(features|L2)P(L1)P(L2)}$$

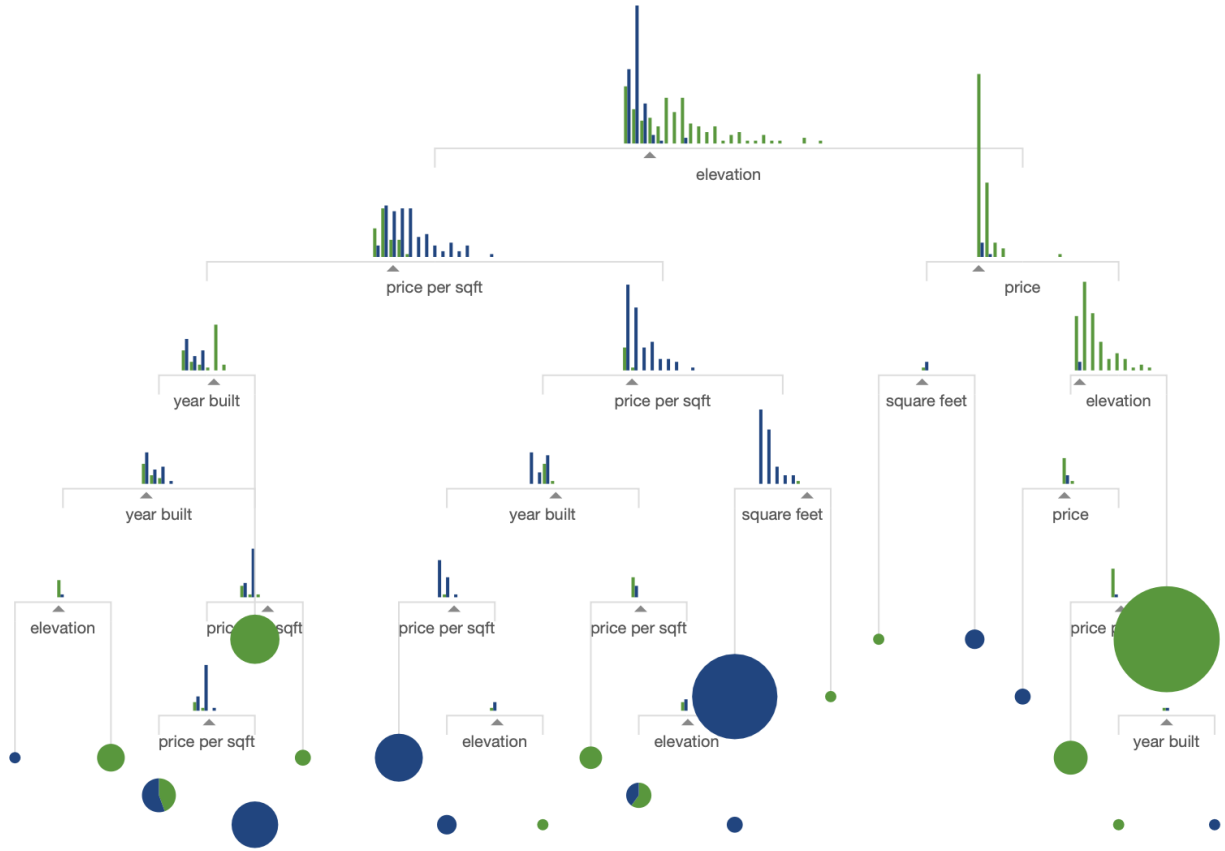
- "Naive" since we make simplifying, naive assumptions about the data.

Naive Bayes Models



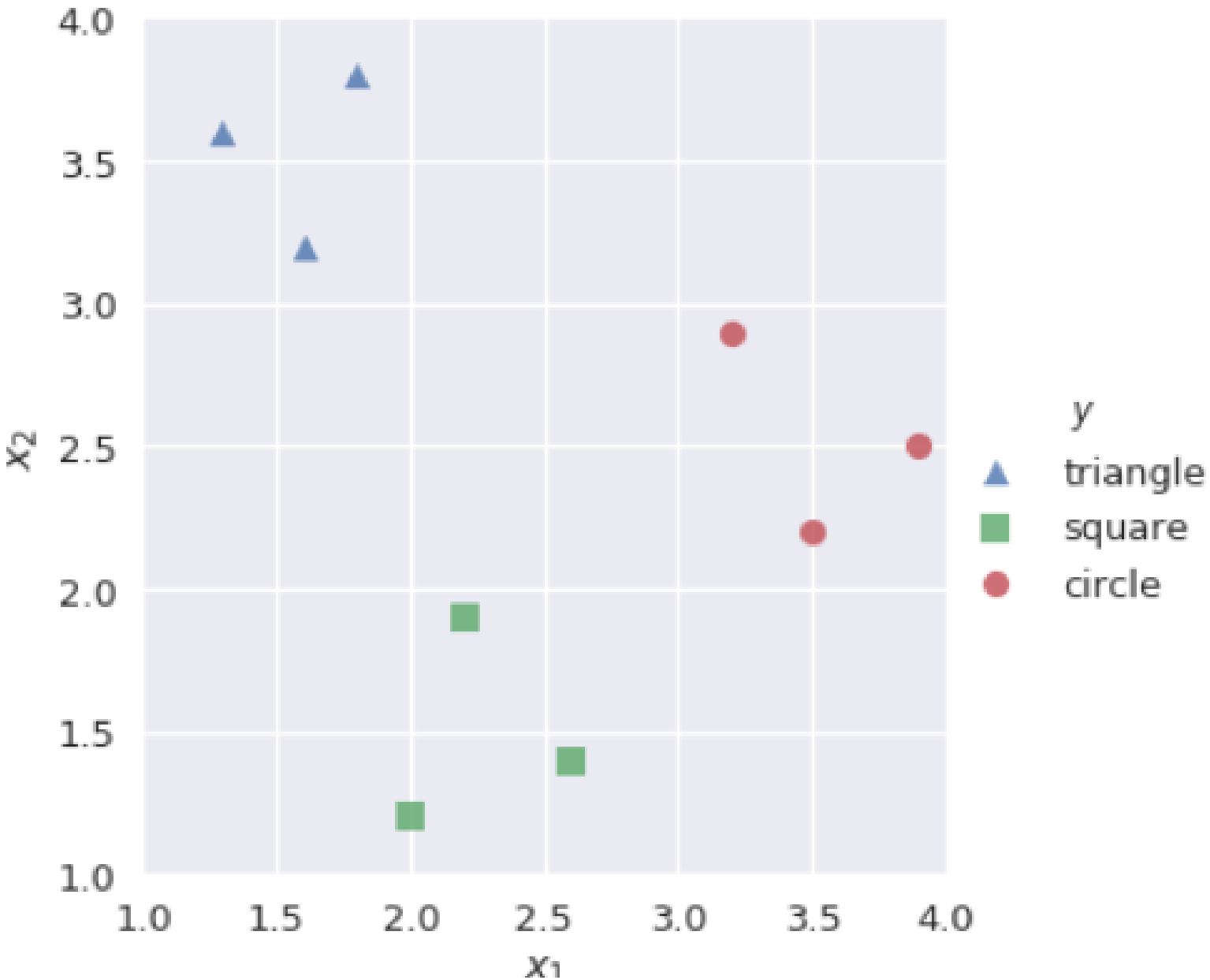
Notebook: work through Handbook: Naive Bayes

Random Forests

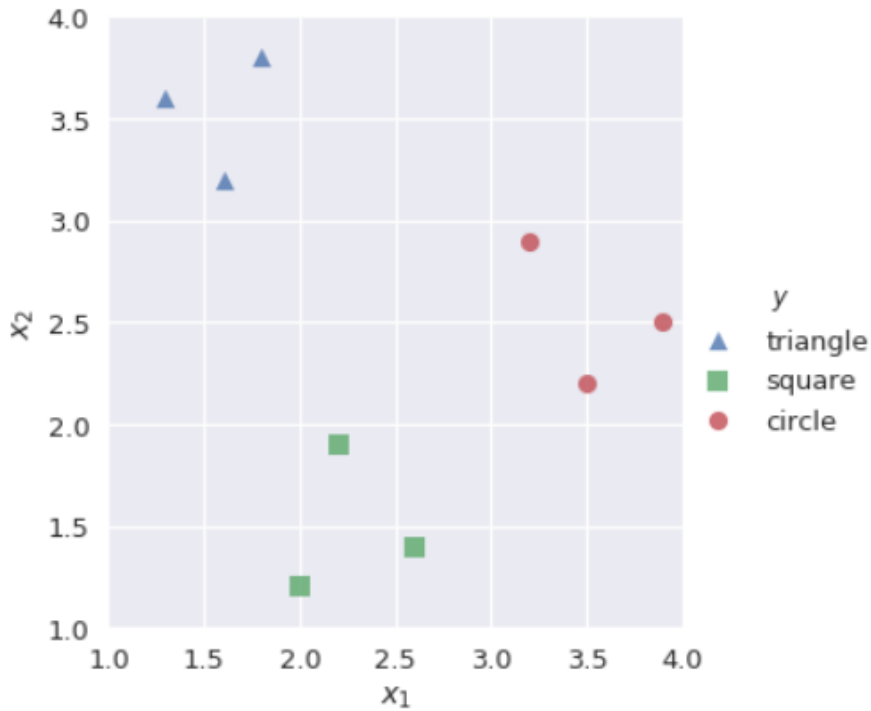


- Intuition: see r2d3's gorgeous decision trees.
- A random forest is an ensemble of decision trees: while any one tree might not be the best classifier, many combined together are.
- Notebook: use random forests to classify.

Extending to Multiclass Classification



Extending to Multiclass Classification



- Can leverage our binary classifiers to distinguish multiple classes.
- Idea: build classifiers that separate each class from all the rest.
- Called One-vs-Rest (OvR) or One-vs-All (OvA).
- Multiclass option is built in to most classifer software.

Outline

- Morning:
  - Seaborn
  - Models & Machine Learning
  - Regression
- Afternoon:
  - Training Models
  - Scikit-Learn
  - Classifiers
- Wrap Up

Wrap Up

- What we did...
  - Used Seaborn to visualize.
  - Introduced Models & Learning
  - Analyzed FRED Price Data
  - Survey of popular classifiers in sklearn
- Tomorrow:
  - PyTorch: building neural nets
  - A/B Testing
  - Time Series Analysis