



Data @ UCI

.....

# Datathon @ UCI

Resource Sheet

*\*Remember to check out our Winter Workshop Series!!*

.....

## Table of Contents

|   |           |
|---|-----------|
| <b>Datathon @ UCI</b>                     | <b>1</b>  |
| <b>Data @ UCI Workshops</b>               | <b>2</b>  |
| Winter Workshop Series                    | 2         |
| Data Project Bootcamp                     | 2         |
| R Visualization                           | 2         |
| Intro to Python for Data Analysis         | 2         |
| Beginner SQL                              | 2         |
| <b>Data Projects</b>                      | <b>2</b>  |
| Tips for a Good Data Project              | 2         |
| Project Examples                          | 2         |
| Software & Applications                   | 3         |
| <b>Exploratory Data Analysis (EDA)</b>    | <b>3</b>  |
| Pandas: A Python Library                  | 3         |
| SQL                                       | 3         |
| R   | 5         |
| Time Series Analysis                      | 5         |
| <b>Data Visualization</b>                 | <b>7</b>  |
| Plotly: Interactive Graphs!               | 7         |
| Seaborn & Matplotlib                      | 7         |
| ggplot2: A R Library                      | 7         |
| <b>Statistical Tests &amp; Models</b>     | <b>8</b>  |
| Hypothesis Testing (T-Test, Chi-Square)   | 8         |
| ANOVA                                     | 8         |
| Regression Models (Linear, Poisson, etc.) | 9         |
| Model Validation                          | 10        |
| <b>Machine Learning</b>                   | <b>11</b> |
| Common Terminology                        | 11        |
| ML Workflow                               | 13        |
| Model Dangers                             | 13        |
| ML Validation & Optimization              | 13        |
| Sample Code                               | 14        |
| Statistical Models                        | 14        |
| Statistical Tests                         | 14        |
| ML Models                                 | 14        |
| <b>Misc. Resources</b>                    | <b>15</b> |
| <b>Resource Submission</b>                | <b>15</b> |

---

# Data @ UCI Workshops

## Winter Workshop Series

[Deepnote Notebook for all 4 workshops](#)

[Workshop 1 Slides](#)

[Workshop 2 Slides](#)

[Workshop 3 Slides](#)

[Workshop 4 Slides](#)

## Data Project Bootcamp

[Recording](#) | [Google Slides](#)

## R Visualization

[Recording](#) | [Google Slides](#) | [Workbook](#)

## Intro to Python for Data Analysis

Recording: [2023](#), [2022](#) | [Google Slides](#) | [Setup Instructions](#) | [Workbook](#)

## Beginner SQL

***Notice: Made an announcement about this a while back but the Q2 solution was wrong during the workshop. So, we fixed the solution + more! I also added additional SQL examples to the workbook for you to reference!***

[Google Slides](#) | [Setup Instructions](#) | [Workbook](#)

# Data Projects

## Tips For Working on a Collaborative Data Project

We highly recommend using a cloud based data science environment so it's easier to share information and work amongst your team!

- <https://deepnote.com/>
  - Most similar to a Jupyter Notebook you run locally
- <https://hex.tech/>
- <https://count.co/>
  - Provides a miro/figma-like canvas

## Tips for a Good Data Project

**Data @ UCI - Data Project Bootcamp** (Check out **Data @ UCI Workshops**)

- Components to a Data Project
- Titanic Dataset (w/ a spin)
- Machine Learning Tips
- [Slidego](#): Free Google Slide Templates

**StrataScratch:**

- [What makes a Good Data Project](#)
- [Guide to Jump Start Data Project](#)

Medium:

- [7 Tips for a Successful Data Project in Industry](#)

“Becoming a Data Head” by Alex J. Gutman and Jordan Goldmeier

## Project Examples

### Rice Datathon

- Website: <https://datathon.rice.edu/>
- 2022 Winners: <https://d2k.rice.edu/news/4th-annual-rice-datathon-winner-announced>

### UCI Machine Learning Hackathon

- [Four Impactful Projects at 2022 ML Hackathon](#)

### Something I found online 😊

- <https://www.linkedin.com/pulse/analyzing-linkedin-content-practical-application-data-acheampong/>

### William Hou

- [Pandemic of Lies – Detecting Misinformation on Twitter](#) (UCI ML Hackathon ‘21)
- [Telemedicine Trends](#) (Rice Datathon ‘21)

## Software & Applications

### Team Collaboration

- Jupyter Notebook: a Shareable Coding Workbook (Installs w/ Anaconda!)
- [DeepNote](#): Jupyter Notebook but on the cloud ☁
- [GitHub](#): Developer Platform
  - [Git It? How to use Git and Github](#)
  - [Git QuickStart](#)
  - [Git Guide \(no deep shit\)](#)
  - [Learn Git Branching](#)
  - [Hack at UCI Git/Github Workshop](#)

### Software

- [Anaconda](#): the world's most popular data science platform (highly recommend)
- [Alteryx Designer](#): low code analytical platform (good for large datasets)

# Exploratory Data Analysis (EDA)

**Exploratory Data Analysis** is the process of analyzing and summarizing datasets in order to extract insights and patterns. It is usually the first step in the data analysis process and is used to gain a preliminary understanding of the data before going to more advanced analysis (e.g. machine learning)

Some examples of EDA include (but not limited to):

- Data cleaning and manipulation
- Summary Statistics
- Data Visualization
- and more!

## Pandas: A Python Library

**Pandas** is a data manipulation and analysis library for Python. The library allows users to import data from sources such as SQL Tables, CSV files, and more into manipulable dataframe and series objects. To learn more: GeeksforGeeks: [Intro to Pandas](#), StrataScratch: [How to Import Pandas](#)

- [Pandas Documentation](#)
- **Data @ UCI** – *Intro to Python for Data Analysis* (Check out **Data @ UCI Workshops**)
  - Importing Data
  - Basic Pandas Functions
  - Data Cleaning & Data Manipulation
  - Aggregation and Transformation (.groupby(), .merge())
- **Kaggle**:
  - [Pandas: Solve short hands-on challenges to perfect your data manipulation skills](#)

- [Data Cleaning](#)
- **DataCamp**
  - [Python Pandas Tutorial: Ultimate Guide for Beginners](#)
  - [Search-Resources](#) (type in Pandas)

## SQL

**SQL** (Structured Query language) is a programming language used to manage and manipulate relational databases. SQL allows users to perform various operations stored in a DB. Depending on the data sources (.csv file, SQL DB, NoSQL DB, SQLite DB), you might use no SQL at all in your projects! But I will just provide some resources here for those who are interested 😊

### MISC Resources

- [All SQL Functions & Clauses](#) (remember that there are different types of SQL!)
- Check out the chart below!

### Data @ UCI - Beginner SQL (Check out **Data @ UCI Workshops**)

- SELECT, FROM, WHERE, INNER JOIN (using WHERE)
- Handling Null Values
- LIMIT BY, ORDER BY, Comparison Operators
- Subqueries

### StrataScratch

- [SQL Articles](#)
- [SQL Cheat Sheet for Technical Interview](#)

### SQLAlchemy - Querying SQL databases using Python

- [ODBC Driver for SQL Server](#)
- [SQLAlchemy Code](#)
  - Use {SQL Server} or {ODBC Driver 17 for SQL Server} for **Driver**
  - Otherwise, filter the data using SQL and place it in a csv for analysis

### Advanced SQL Chart:

|                       |   |   |
|-----------------------|---|---|
| <a href="#">Joins</a> | <a href="#">LEFT/RIGHT JOIN</a>   | Returns all data from the left/right table and only matching rows from the adjacent table (null otherwise). |
|                       | <a href="#">[INNER] JOIN</a>  | Outputs only matching rows found in both tables (based on indicator)  |
| Time/Date Functions   | <a href="#">EXTRACT()</a>   | EXTRACT(YEAR from ...)<br>● For only PostgreSQL & MySQL   |
|                       | <a href="#">DATEPART()</a>  | DATEPART(yy, order_date)<br>● MSSQL   |
|                       | <a href="#">CAST()</a>  | Convert a value (of any type) into a specified datatype<br>Can used for ALL SQL types (Not just SQL Server) |
|                       | <a href="#">DATEDIFF()</a>  | ● MSSQL   |
| Window Functions      | <p><code>window_function ( expression ) OVER (</code><br/> <code>[PARTITION BY list_of_columns]</code><br/> <code>[ORDER BY list_of_columns]</code><br/> <code>[ROW or RANGE clause])</code><br/> <b>PARTITION BY</b> - Defines a window frame (window function executed on each partition separately)<br/> <b>ORDER BY</b> - Specify the order in which the window function will be executed<br/> <b>ROW/RANGE</b> - Defines a window frame (define a start and end rows of the window frame)</p> <ul style="list-style-type: none"> <li>- <b>ROW</b>: doesn't look at the current row's value. Defines window frame by number of rows before/after current row (lowkey cool 😊)</li> <li>- <b>RANGE</b>: Specifying the rows in relation to the current row's value</li> </ul> <p><i>Find the 3-month rolling average of total revenue from purchases given a table with users, their purchase amount, and date purchased (in PostgreSQL)</i></p> <pre>SELECT t.month,        monthly_revenue,        AVG(t.monthly_revenue) OVER(</pre> |   |

|             |   |   |
|-------------|---|---|
|             | ORDER BY t.month ROWS BETWEEN 2 PRECEDING<br>AND CURRENT ROW) AS avg_revenue<br>FROM<br>(SELECT to_char(created_at::date, 'YYYY-MM') AS MONTH, # <i>to_char in PostgreSQL</i><br>sum(purchase_amt) AS monthly_revenue<br>FROM amazon_purchases<br>WHERE purchase_amt>0<br>GROUP BY to_char(created_at::date, 'YYYY-MM')<br>ORDER BY to_char(created_at::date, 'YYYY-MM')) t |   |
|             | <b>ROW</b>  | <b>UNBOUNDED PRECEDING</b> – all the rows before the current row and including the current row<br><b>UNBOUNDED FOLLOWING</b> – all the rows after the current row and including the current row<br><b>N PRECEDING</b> – defined number of rows before the current row and including the current row<br><b>M FOLLOWING</b> – defined number of rows after the current row and including the current row<br><b>CURRENT ROW</b> – only the current row |
|             | <b>RANK()</b>   | <ul style="list-style-type: none"> <li>Ranks rows &amp; skip ties (makes them same rank. Skip number of ranks in line w/ # of ties)</li> <li>E.g. 1, 2, 2, 4</li> </ul>   |
|             | <b>ROW_NUMBER()</b>   | <ul style="list-style-type: none"> <li>Allocates row number to rows</li> <li>May be used for ranking in combination with ORDER BY()</li> </ul>  |
|             | <b>LEAD(), LAG()</b>  | <ul style="list-style-type: none"> <li><b>LEAD()</b> returns value from the <b>following</b> row</li> <li><b>LAG()</b> returns value from the <b>previous</b> row</li> <li>Ordering really <u>matters</u></li> </ul>  |
|             | <b>Aggregates (COUNT(), SUM(), etc.)</b>  | <a href="#">Ultimate Guide to Window Functions</a>  |
| <b>MISC</b> | <b>WHERE vs HAVING</b>  | <b>WHERE</b> – filter specific rows<br><b>HAVING</b> – filter specific groups based on AGGREGATION in data  |
|             | <b>CTEs</b>   | Temporary result set returned by a query (different from subquery is that it can be named and reference itself)   |
|             | <b>CTE vs Subquery</b>  | <ul style="list-style-type: none"> <li><b>CTEs</b> are reusable, more readable, and recursive</li> <li><b>Subqueries</b> can be used in the WHERE clause, act as a column w/ a single value, used as a correlated subquery</li> </ul>   |
|             | <b>UNION (vs UNION ALL)</b>   | <ul style="list-style-type: none"> <li><b>UNION</b> returns only the unique values from the output of two/more queries (must have equal # of columns, same data types)</li> <li><b>UNION ALL</b> is similar to <b>UNION</b> but returns duplicates</li> </ul>   |

| Query Optimization  | Query Simplification   |
|---|--|
| <ul style="list-style-type: none"> <li>Replace complex subqueries with CTEs</li> <li>Find a better SQL feature(s) to accomplish the same in fewer lines</li> <li>Use more readable syntax</li> <li>Format, index, and comment</li> <li>Joins take time</li> </ul> | <ul style="list-style-type: none"> <li>Utilize Keys (since they are already loaded)</li> <li>Range Queries benefit from B+ Trees &amp; Clustered indexes</li> <li>Equality Queries benefit from Hashes (and from clustered when duplicates)</li> <li>Like expressions are bad!!</li> </ul> |

## R

R is a programming language that's mostly used for statistical computing and analysis. If you are in mathematics, statistics, or data science, you probably have heard of R.

**R for Data Science (tidyverse, ggplot2, etc.):** <https://r4ds.had.co.nz/>

*Will be very helpful if you never used R before* ↑

**SoCal R User Group:** <https://socalr.org/> (they have a YouTube Channel 😊)

**DataCamp:** <https://www.datacamp.com/search-resources> (You can search up R here)

### Time Series Analysis

Time Series Analysis is the statistical technique used to analyze and predict data that varies over time. It involves identifying patterns and trends in the data using that information to make predictions and forecasts. More information about Time Series Analysis is [here](#) (Tableau) and [here](#) (Kaggle)

Think about using a **Paired T-Test** and working with **Longitudinal Data** 🕒

# Data Visualization

## Plotly: Interactive Graphs!

**Plotly** is a data visualization library that allows you to create **interactive** graphs, charts, and even dashboards in Python, R, and other coding languages (though we will only share Python resources here). It provides a wide range of chart types including scatter plots, line charts, heatmaps, **ACTUAL MAPS**, and more.

### **Plotly Website**

- [Python Documentation](#)
- [Making an Interactive Map \(Choropleth Mapbox\)](#)

### **Medium**

- [Simple Plotly Tutorial. Creating Beautiful Animated Maps](#)

### **Kaggle**

- [Plotly Tutorial for Beginners](#)

**GeoJson Mapper:** <https://geojson.io/>

## Seaborn & Matplotlib

**Matplotlib** is a Python library used to create different kinds of charts. You can generate bar graphs, histograms, scatterplots, etc.

[Matplotlib Documentation](#)

### **Kaggle**

- [Data Visualization using Matplotlib](#)

### **Analytics Vidhya**

- [Introduction to Matplotlib using Python for Beginners](#)

### **DataCamp**

- [Matplotlib Tutorial in Python](#)
- [Matplotlib Cheat Sheet](#)

**Seaborn** is another data visualization library that's built on top of Matplotlib. Compared to Matplotlib, it's generally considered easier to use, have more attractive visualization styles, less customization, etc. Pick which library is right for you.

[Seaborn Documentation](#)

### **Kaggle**

- [Data Visualization Tutorial](#)

### **DataCamp**

- [Seaborn Python Tutorial](#)
- [Seaborn Cheat Sheet](#)

## ggplot2: A R Library

**ggplot2** is an R library for creating different data visualizations. It allows for the creation of complex and customizable plots by combining different layers (filters, data manipulation, etc.), and provides a wide range of customization options and built-in plot types.

**Data @ UCI – R for Data Visualization** (Check out **Data @ UCI Workshops**)

- Bar Chart, Pie Chart
- Histograms, Scatter Plots

[ggplot2 Documentation](#)

### **R for Data Science**

- [Exploratory Data Analysis](#)

### **DataCamp**

- [ggplot2 Cheat Sheet](#)

# Statistical Tests & Models

*\*I recommend taking the STATS 110 series!*

There is a lot to cover when it comes to statistics. And the resources here may not cover everything (or anything) you need for your data project. So take the time to determine for yourself what kinds of testing and models you will need for your own data project.

Familiarize yourself with terms such as **Two Way Table** (you learn this in Stats 111), **Type I Error**, etc.

The **CODE** for these tests and models are in the **Machine Learning** section.

| <a href="#">Statistical Tests</a>       | <b>Statistical Models</b>   |
|---|---|
| Test specific hypotheses about the data | Predicting future outcomes, understanding the relationship between variables, or testing hypotheses |

## Hypothesis Testing (T-Test, Chi-Square)

| <a href="#">T-Test</a><br><i>A way to test hypotheses (w/out the population standard deviation)</i> |  |
|---|--|
| <a href="#">One Sample T-Test</a>   | Tests the mean of a single group against a known mean (e.x. Expected average, if the mean is more than the majority, etc.)   |
| <a href="#">Two Sample T-test</a>   | <b>Unequal Variance (Welch's T-test):</b> # of samples in each group are different, variance of two also different.<br><br><b>Equal Variance (Student's T-test):</b> when the # of samples in each group is the same or the variance of two datasets is similar. |
| <a href="#">Paired T-test</a>   | Samples consist of matched pairs of similar units or when there are cases of repeated measures   |

| <a href="#">Chi-Square Test</a>  |  |
|--|--|
| <u>One Sample: Goodness of Fit</u>   | <u>Two Sample: Test of Independence</u>                    |
| <i>Determines whether distribution of groups (e.g. flavors of candy, races, etc.) are the same or not.</i><br><br><i>Or check if these distributions/proportions certain thresholds 😊</i>        | <i>Determines whether two variables are related or not</i> |
| Apparently, the chi-square test can be used for model validation?<br>I am guessing it will be similar to what <b>ANOVA</b> does.<br><br>More info about the Chi-Square Test <a href="#">here</a> |  |

## ANOVA

| <a href="#">ANOVA</a><br><i>A great way to compare two or more groups/models</i>                     |  |
|--|--|
| <a href="#">One Way</a>  | <a href="#">Two Way</a>  |
| Basically the same as One Sample T-Test<br><i>We have the grades for four (K=4) exams. Each exam</i> | <b>Two</b> categorical explanatory variables and a <b>single</b> quantitative response variable<br><b>Example 1:</b> |



|  |  |
|--|--|
| <p>has 5 grade observations (<math>n_k = 5</math> for <math>k=1,2,3,4</math>)<br/>We would like to test if the average grade for each exam is the same</p> | <p>Factor A: (A1, A2, A3)<br/>Factor B: (wood, gas)<br/>Response: time until fire completely out</p> <p><b>Example 2:</b><br/>Is music treatment the main factor affecting performance? Do groups subjected to different music differ significantly in their test performance?</p> |
| <p>More info about ANOVA and other related methods <a href="#">here</a> (ANOVA, MANOVA, etc.) and <a href="#">here</a>.</p>                                |  |

### Regression Models (Linear, Poisson, etc.)

| Risk Difference                               | Risk Ratio  | Odds Ratio   |
|---|---|--|
| $RD = p_1 - p_0$<br>Absolute Effect of X on Y | $RR = \frac{p_1}{p_0}$<br>$p_1$ likely over $p_0$ | $OR = \frac{p_1/(1-p_1)}{p_0/(1-p_0)}$ (relative effect)<br>Likelihood that event will happen over not happening. Strictly Non-Negative<br>When $p_0$ and $p_1$ are close to 0, $OR \sim RR$ |

| Regression Model Types  |   |
|---|---|
| <a href="#">Linear</a>  | <p>Used to measure a <b>numeric response</b> variable from one or more independent explanatory variable (either numeric or categorical)</p> <ul style="list-style-type: none"> <li>For categorical variables, would need to quantify it somehow<br/>house price = 172893+241582*pool (pool = 0 is NO pool, pool = 1 is pool)</li> </ul>   |
| <p><b>Poisson, Logistic, and Multinomial</b> regression models are <b>General Linear Models (GLMs)</b><br/><b>General Linear Models</b> change the response values so that it can have <i>appropriate bounded outcomes</i>.<br/>So instead of getting “# of green people” (direct number), you get “probability of being green” (probability)<br/>Or something like that.</p> |   |
| <a href="#">Logistic</a>  | <p>Model that can be used to <b>classify data</b> into categories.<br/>Predicts the <b>probability that an observation falls into a particular class/group</b> based on their features</p> $\text{logit}(\hat{\mu}) = \log\left(\frac{\hat{\mu}}{1-\hat{\mu}}\right) = \hat{\beta}_0 + \hat{\beta}_1 I(\text{group B}) + \hat{\beta}_2 I(\text{group C})$ <p><math>e^{\hat{\beta}_1}</math> = estimated relative change in odds for Y=1 comparing group B to base group (group B vs group A) (aka change in odds ratio)<br/> <math>e^{\hat{\beta}_1 - \hat{\beta}_2}</math> = estimated relative change in odds for Y=1 comparing group B to group C (aka change in odds ratio)</p> $P(Y=1) = \mu = g^{-1}(g(\mu)) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$ $P(Y=0) = 1 - \mu = \frac{1}{1 + e^{\beta_0 + \beta_1 X}}$ <p><math>\beta_1 &gt; 0</math> – probability of success (Y=1) increases w/ X<br/> <math>\beta_1 &lt; 0</math> – probability of success (Y=1) decreases w/ X (X <math>\uparrow</math> → odds <math>\downarrow</math> → prob <math>\downarrow</math>)</p> <p>Odds of Y = 1 is <math>e^{\beta_0 + \beta_1 X}</math></p> |
| <a href="#">Poisson</a>   | <p>Model calculates the <b>rate of an event happening over a period of time</b></p> <ol style="list-style-type: none"> <li>Number of vehicles passing on a toll road per weekend</li> <li>Number of requests to a cloud server per day</li> </ol>   |

|                               |   |
|-------------------------------|---|
|                               | <p>3. Number of cases of a disease over a year time<br/> 4. Number of credit cards someone owns<br/> 5. Number of smartphones a household owns</p> $P(Y = k) = \frac{e^{-\mu} \mu^k}{k!} \text{ for } k = 0, 1, 2, \dots$ $E(Y) = \mu = A\lambda, \text{ var}(Y) = \mu$ <p><math>\mu</math> = expected value of Y (positive) (# of cloud server request in a day)<br/> Y = count of events that came about a rate of <math>\lambda</math> per unit-time of exposure<br/> <math>\lambda</math> = rate parameter<br/> A = exposure period<br/> E.g. <math>\lambda = 10</math> events/day. A = 2 days<br/> <math>E(Y) = 20</math></p> <p>E.g. If we have rate of <math>\lambda = 0.05</math> event per subject. Let A be 1000 subjects. Then<br/> <math>\mu = 1000 \times 0.05 = 50</math> events per 1000 people/subjects<br/> Crime rate <math>\rightarrow</math> Crime event counts</p> |
|                               | <p>Can be model with <b>an offset term</b> to balance out any differences in exposure periods<br/> e.g. We have two observations. One is a 24 hour period. Another is a 12 hour period.<br/> By adjusting the exposure period with the offset period, we can just compare rates per unit time between observations</p> <p><math>Y_i</math> - Poisson where <math>\lambda_i</math> - rate, <math>A_i</math> - exposure period<br/> A = 1, 2, 5, 7 days<br/> <math>\log(\mu_i) = \log(A_i \lambda_i) = \log(A_i) + \log(\lambda_i)</math><br/> <math>= \log(A_i) + \beta_0 + \beta_1 X_{i1} + \dots + \beta_p X_{ip}</math><br/> <math>\log(A_i) = 1 \times \log(A_i)</math> - offset term</p>  |
| <b>Multinomial (Logistic)</b> | <p>Used when response Y can be one of several (J&gt;2) categories</p> <ul style="list-style-type: none"> <li>Blood Type (A, B, AB, O)</li> <li>Automobile Choice (Ford, Chevy, Dodge)</li> <li>Political Party Affiliation</li> </ul> <p>Used to find the probability of an observation being in one of the 3+ categories based on one or more independent variables.</p> $\eta_{ij} = \log\left(\frac{P(Y_i=j X_i)}{P(Y_i=1 X_i)}\right) = \beta_{0j} + \beta_{1j} \text{Hinc}_i + \beta_{2j} \text{Psize}_i \text{ for } j = 2, 3, 4 \text{ (1 is base group)}$   |

## Model Validation

| Model Validation (w/ Models & Tests!) |  |
|---------------------------------------|--|
| <b>ANOVA</b>                          | You can use ANOVA to compare nested models (full model vs reduced model) and determine what coefficients are not necessary.  |
| <b><u>ROC Curve</u></b>               | <p>Used to check the performance of a <b>binary classifier</b> (e.g. logistic regression)</p> <p>In particular, the ROC curve is composed by plotting a model's</p> <ul style="list-style-type: none"> <li><b>True-Positive Rate</b> (TPR) (probability that a positive sample is correctly predicted in the positive class) versus</li> <li><b>False-Positive Rate</b> (FPR) (probability that a negative sample is incorrectly predicted in the positive class)</li> </ul> <p>Check out the link on the left for more information!</p> |
| <b>RSME</b>                           | <a href="https://www.statology.org/what-is-a-good-rmse/">https://www.statology.org/what-is-a-good-rmse/</a>  |
| <b>Deviance</b>                       | $D = -2 \log(L_0/L_1) = -2[\log(L_0) - \log(L_1)]$ $= -2[\log(L_0) - \log(L_S)] - (-2[\log(L_1) - \log(L_S)])$ <p><math>L_1</math> - maximized value of likelihood under full model</p>  |

|                                       |   |
|---------------------------------------|---|
|                                       | $L_0$ - maximized value of likelihood under reduced model<br>D has appropriate chi-squared distribution w/ $p - k$ degrees of freedom   |
| <a href="#">Likelihood Ratio Test</a> | Calculates the <b>deviance</b> between two nested models and compares them to see which one is better (this was covered in Stats 111 😊)<br><br>Essentially, we have this <b>“perfect” model</b> that fits our data 100%. We will compare this “perfect model” between two models (one has more coefficients than the other one) and calculate their <b>deviances</b> (aka the difference between the model and our “perfect model”). Then we <b>compare the deviances</b> to see which one is better! |
| <a href="#">AIC</a>                   | Similar to calculating Deviance but penalizes you if your model is more complicated. Its penalization is similar to what the adjusted $R^2$ does.<br>$2k - 2\log(L_M)$ $k$ - # of parameters to be estimated by model<br>$L_M$ - maximized likelihood<br>AIC good when comparing models (lower AIC → better)  |

## Machine Learning

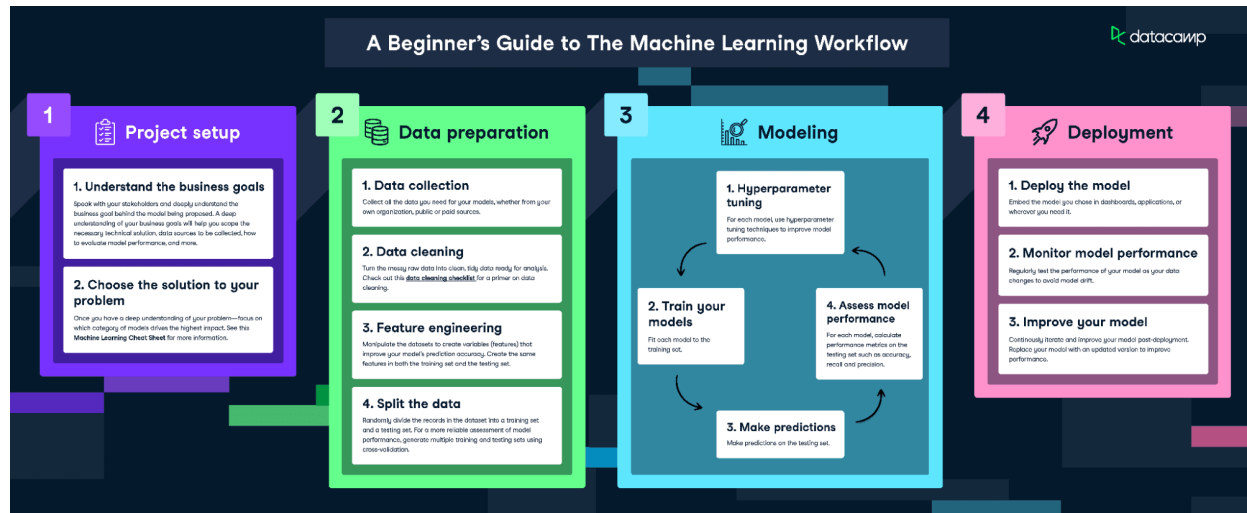
| Difference between ML and AI   | Difference between ML and Statistics  |
|--|---|
| <p><b>Artificial Intelligence (AI)</b> is a <i>“a huge set of tools for making computers behave intelligently”</i> and in an automated fashion. This includes voice assistants, recommendation systems, and self-driving cars.</p> <p><b>Machine Learning (ML)</b> is the <i>“field of study that gives computers the ability to learn without being explicitly programmed.”</i> The lion’s share of ML involves computers learning patterns from existing data and applying it to new data in the form of making predictions, such as predicting whether an email is spam or not, whether a customer will churn or not, and diagnosing a particular piece of medical imaging.</p> <p><b>Source:</b><br/> <a href="https://www.datacamp.com/blog/the-difference-between-ai-and-machine-learning">https://www.datacamp.com/blog/the-difference-between-ai-and-machine-learning</a> </p> | <a href="https://towardsdatascience.com/the-actual-difference-between-statistics-and-machine-learning-64b49f07ea3">https://towardsdatascience.com/the-actual-difference-between-statistics-and-machine-learning-64b49f07ea3</a> |

### Common Terminology

|   |   |
|---|---|
| <b>Machine Learning</b>                             | <a href="https://www.datacamp.com/blog/what-is-machine-learning">https://www.datacamp.com/blog/what-is-machine-learning</a>   |
| <a href="#">Supervised vs Unsupervised Learning</a> | <p><b>Supervised Learning</b> – Type of Machine Learning where the model uses <b>labeled</b> datasets. By “labeled”, it means that the data is already tagged with the right answer</p> <ul style="list-style-type: none"> <li>Regression (Linear, Poisson, Logistics, etc.)</li> <li>Classification (K-Nearest Neighbor, Random Forest, Decision Trees)</li> </ul> <p><b>Unsupervised Learning</b> – Type of machine learning that uses <b>unlabeled</b> datasets. Such learning can uncover unknown patterns in data <b>without</b> human supervision</p> <ul style="list-style-type: none"> <li>K-Means Clustering</li> <li>Dimensionality Reduction (PCA)</li> </ul> <p>Check out the link on the left to learn more about supervised vs unsupervised learning!</p> |

|  |   |
|--|---|
|  | <b>StrataScratch:</b><br><a href="https://www.stratascratch.com/blog/supervised-vs-unsupervised-learning/">https://www.stratascratch.com/blog/supervised-vs-unsupervised-learning/</a>  |
| <b><u>Deep Learning</u></b>                | <p>Type of machine learning that involves training artificial neural networks to learn from large amounts of data. It resembles human brains.</p> <ul style="list-style-type: none"> <li>• Computer Vision</li> <li>• Generative AI (ChatGPT)</li> <li>• CNNs, RNNs</li> </ul> <p><b>DataCamp:</b><br/> <a href="https://www.datacamp.com/tutorial/demystifying-mathematics-concepts-deep-learning">https://www.datacamp.com/tutorial/demystifying-mathematics-concepts-deep-learning</a></p>   |
| <b><u>Reinforcement Learning</u></b>       | <p>Type of machine learning where an agent learns through interaction with its environment to make decisions that maximize reward.</p> <p>It's <b>different from Supervised Learning</b> as we don't tell the agent what to do. But it's also <b>different from Unsupervised Learning</b> because we are maximizing reward, not correctness.</p> <p><b>DataCamp:</b><br/> <a href="https://www.datacamp.com/tutorial/introduction-reinforcement-learning">https://www.datacamp.com/tutorial/introduction-reinforcement-learning</a></p>   |
| <b>Bias Variance Tradeoff</b>              | <a href="https://mlu-explain.github.io/bias-variance/">https://mlu-explain.github.io/bias-variance/</a>   |
| <b>Regularization</b>                      | <a href="https://www.dataquest.io/blog/regularization-in-machine-learning/">https://www.dataquest.io/blog/regularization-in-machine-learning/</a><br><a href="https://www.instagram.com/p/Cpe9meOjRZJ/?igshid=YmMyMTA2M2Y=">https://www.instagram.com/p/Cpe9meOjRZJ/?igshid=YmMyMTA2M2Y=</a>  |
| <b><u>Dimensionality Reduction</u></b>     | <p>Technique used in ML to reduce the number of features (e.g. columns in a table) in a dataset while retaining most important information. We map the <b>original "high dimensional"</b> dataset into a <b>"lower-dimensional"</b> dataset.</p> <p>The reason for Dimensionality Reduction is to reduce the computational complexity of ML algorithms and improve accuracy by removing irrelevant information.</p>   |
| <b><u>Principal Component Analysis</u></b> | <p>A type of <b>dimensionality reduction</b> where we transform the original dataset into a "new coordinate system" (like with vectors), in which the axes are the principal components of the data.</p> <p>It's like setting "Color" to the y axis, "Size" to the x-axis, and the line on the graph represents "Color" <b>and</b> "Size"!</p>  |
| <b><u>Entropy</u></b>                      | <p>It's the quantitative value of uncertainty. It's mostly related to <b>Decision Trees</b> and <b>Random Forests</b> because the goal of these models is to <b>reduce Entropy</b>.</p>   |
| <b><u>Boosting</u></b>                     | <p>Method that trains a sequence of "weak" models, where each sequentially compensates for the weakness of the preceding models</p>   |
| <b>Clustering</b>                          | <p>Type of unsupervised machine learning where the goal is to group similar objects/data objects into clusters, based on some kind of similarity. There are different kinds of clustering (density-based, hierarchical, partitioning, grid-based)</p> <p><b>DataCamp:</b><br/> <a href="https://www.datacamp.com/blog/clustering-in-machine-learning-5-essential-clustering-algorithms">https://www.datacamp.com/blog/clustering-in-machine-learning-5-essential-clustering-algorithms</a></p> <p><b>GeeksforGeeks:</b> <a href="https://www.geeksforgeeks.org/clustering-in-machine-learning/">https://www.geeksforgeeks.org/clustering-in-machine-learning/</a></p> <p><b>StrataScratch:</b><br/> <a href="https://www.stratascratch.com/blog/machine-learning-algorithms-explained-clustering/">https://www.stratascratch.com/blog/machine-learning-algorithms-explained-clustering/</a></p> |

## ML Workflow



Source: <https://www.datacamp.com/blog/a-beginner-s-guide-to-the-machine-learning-workflow>

## Model Dangers

|                     |  |
|---------------------|--|
| <b>Overfitting</b>  | <p>When the ML model gives accurate predictions for training data but not for new data (it's like a very saturated model, an old dog (can learn no new tricks), etc..).</p> <p>Some ways to prevent overfitting is cross validation, regularization, adding penalties to more complex models (AIC), etc.</p> |
| <b>Underfitting</b> | <p>When the ML model cannot determine the true relationship in the underlying data</p>   |

## ML Validation & Optimization

|                              |   |
|------------------------------|---|
| <b>Cross-Validation</b>      | <p>Assess the performance of a ML model by dividing the available data into multiple subsets where each fold is used for testing the model while the remaining is used to train the model.</p> <p>Most common CV is <b>k-fold cross-validation</b> where the data is divided into k-equally sized subsets</p> <p>Source &amp; Code: <a href="https://scikit-learn.org/stable/modules/cross_validation.html">https://scikit-learn.org/stable/modules/cross_validation.html</a></p> |
| <b>Hyperparameter Tuning</b> | <p>Check the link on the left by DataCamp for a <b>good explanation and code</b>.<br/><b>Grid Search vs Random Search</b></p>   |
| <b>Gradient Descent</b>      | <p><a href="https://www.datacamp.com/tutorial/tutorial-gradient-descent">https://www.datacamp.com/tutorial/tutorial-gradient-descent</a></p>  |

|  |   |
|--|---|
| <a href="https://www.datacamp.com/tutorial/precision-recall-curve-tutorial">Precision Recall</a> | <a href="https://www.datacamp.com/tutorial/precision-recall-curve-tutorial">https://www.datacamp.com/tutorial/precision-recall-curve-tutorial</a> |
| <b>Confusion Matrix</b>  | <a href="https://www.geeksforgeeks.org/confusion-matrix-machine-learning/#">https://www.geeksforgeeks.org/confusion-matrix-machine-learning/#</a> |
| You can also use the validation models and techniques in the Stats section too.                  |   |

## Sample Code

### Statistical Models

|  |   |
|--|---|
| <b>Scikit-Learn (Python Library)</b><br><a href="#">Linear</a>   <a href="#">Logistic</a>   <a href="#">Poisson</a>   <a href="#">Multinomial</a> (multi_class = 'multinomial')  | <b>Statsmodels (Python Library)</b><br><a href="#">Linear</a>   <a href="#">Logistic</a>   <a href="#">Poisson</a>   <a href="#">Multinomial</a><br><i>You can just use the glm() function to make all of these models on your own!</i> |
| <p style="text-align: center;"><b>Stats (R Library)</b><br/><a href="#">GLM</a></p> <div> <div> <b>Linear:</b><br/> <a href="#">LM()</a><br/>           model = lm(Y~equation, data = table)         </div> <div> <b>Logistic:</b><br/>           model = glm(Y~equation,<br/>           family=binomial(link="logit"),data= table)         </div> </div> <p style="text-align: center;"><b>Poisson:</b></p> <p> <b>w/out offset:</b> model = glm(Y~equation, family=poisson, data=table)<br/> <b>w/offset:</b> glm(Y~equation + offset(log(variable)), family=poisson, data=table)         </p> <p style="text-align: center;"><b>nnet (R Library)</b><br/> <a href="#">multinom</a> (Multinomial)<br/>           model = multinom(Y ~ equation, data = table)         </p> |   |

### Statistical Tests

|   |  |  |
|---|--|--|
| Python Libraries: <a href="#">SciPy</a> , <a href="#">Statsmodels</a><br>R Libraries: <a href="#">Stats</a>   |  |  |
| <b>T-Test</b>   | <b>Chi-Square Test</b>   | <b>ANOVA</b>   |
| <b>DataCamp</b><br><a href="#">Python</a>   <a href="#">R</a>   | <b>Scipy (Python Library)</b><br><a href="#">One Way</a>   <a href="#">Two Way</a><br><br><b>Stats (R Library)</b><br><a href="#">chisq.test()</a> | <b>Statsmodels (Python Library)</b><br><a href="#">ANOVA</a>   <a href="#">Statsology Link</a><br><br><b>Stats (R Library)</b><br><a href="#">AOV (One &amp; Two Way)</a>   <a href="#">ANOVA (Model Comparison)</a><br><a href="#">AOV vs ANOVA</a> |
| <b>Statistic Model Validation Tests</b>   |  |  |
| <b>Scikit-Learn (Python Library)</b><br><a href="#">ROC Curve</a><br><br><b>Statsmodels (Python Library)</b><br><a href="#">Likelihood Ratio Test</a>   <a href="#">AIC</a> |  |  |
| <b>Stats (R Library)</b><br><a href="#">AIC</a>   | <b>lntest (R Library)</b><br><a href="#">Likelihood Ratio Test</a>   | <b>pROC (R Library)</b><br><a href="#">ROC Curve</a>   |

### ML Models

Scikit-Learn Cheat Sheet (DataCamp): [here](#)

|                               |                               |                                    |                                    |
|-------------------------------|-------------------------------|------------------------------------|------------------------------------|
| <b>Trees</b>                  |                               | <b>Clustering</b>                  |                                    |
| <a href="#">Decision Tree</a> | <a href="#">Random Forest</a> | <a href="#">K Nearest Neighbor</a> | <a href="#">K-Means Clustering</a> |

## Neural Networks

### **RNN: Recurrent Neural Networks**

- Often used for sequential data such as audio, video (etc. where current context depends on past history)
- Has a notion of memory for accurate predictions (in a video, the **previous** frame relates more to the **current** frame than the **first** frame of the video)

### **CNN: Convolutional Neural Networks**

- Commonly used for image and video processing. It analyzes an image/video in small pieces and combine those pieces later to form a complete understanding of the media
- Analyzes the image through a set of filters that capture different features/aspects of the image

### **LSTM: Long Short Term Memory**

- Type of RNN that have “cells” that store information for long periods of time and gates that control the flow of information into and out of those cells.
- Compared to regular RNNs, can hold more long-term information/dependencies

...and more (SNNs, etc.)!

Some places to find **sample code** for these models:


- Scikit-Learn Documentation
- DataCamp Articles
- GitHub (e.g. [Handson](#), [Nyandwi](#), [Donmartin](#))
- Kaggle
- GeeksforGeeks
- [MadeWithML](#)
- StrataScratch
- LinkedIn (ppl post resources)

## Misc. Resources

### DataCamp

- [Machine Learning Cheat Sheet](#)
- [Data Science Glossary](#)

### StrataScratch:

- [Machine Learning Algorithms: Regression](#)
- [Machine Learning Algorithms You Should Know](#)
- [Top 18 Python Libraries a Data Scientist Should Know](#)
- [4 Data Collection Libraries in Python](#) (Good for when getting your own data!)
-  Working with APIs in Python [For Your Data Science Project]

[Neural Net \(Instagram\)](#) (recommend )

### Kaggle:

- [Intro to Machine Learning Course](#)
- [Intermediate Machine Learning Course](#)
- [Geospatial Analysis: GeoPandas](#)

 How AI, Like ChatGPT, \*Really\* Learns

## Resource Submission

Do you have a great resource that you would like to share? Submit the resource here so it can be placed in next year's resource sheet!: <https://forms.gle/45J4WGoE69DdmJWt8>

.....