Data Science Handbook

HANDBOOK V.01

The friendly handbook for your journey into data science

By Randy Lao

Contents

Introduction

The Secret to your Success is Commitment

I'm going to put it out there: If you're an aspiring data scientist, you're definitely going to get stuck in your initial learning journey. **You'll have setbacks**, you'll get confused with new topics, and you'll have some hard time applying what you learned. And it's okay. It's part of the process.

But if you're committed to take full responsibility for your actions, to put in the work, to collaborate/share with others, to learn from your mistakes, and to not quit, then no matter what you do, you're definitely on your way to success.

Everyone has been through it. And if you ask any great person how they made it, they will let you know that it's all about their commitment to the process. **They never give up.**

So when you're going through this course keep your commitment in mind. Start taking action today, no matter how big or small. Because a small commitment will lead to action. And any ounce of action will bring you closer to your goals.

You'll never be 100% ready. So do it anyways.

You don't have to wait 6 months to do a data science project. You don't have to spend 3 months learning the theory of machine learning before doing a kaggle competition.
You don't have to know ALL the visualization libraries/tools out there to start an analysis.

There's a common mistake where some people may wait till they feel "ready" before they start. The truth is: it'll never happen. **So instead of "learning" and "waiting", start doing.** You're going to learn so much more about the theory and application when you are in the process of doing the projects.

It may feel like you don't know much now, but that's all of us. We always have so much more to learn and so much more to do. **And if you get stuck, that's totally okay**. Look up the solutions on Google, Quora, or Stack-overflow and get it. Rinse and repeat.

Make those amazing projects, tackle that kaggle competition, and make that killer app. And then share it with us. Share your story, your experience, and your learning so that we can all grow together. Because that's what the data community is about.

Don’t Fall Into this Trap

But...here is something nobody talks about. A big trap in data science education is:

- learning data science libraries before learning coding basics
- learning ML algorithms before learning how to preprocess your data
- learning deep learning before machine learning
- learning data viz before understanding the basics of statistical inference

This is where **most people may spend months** trying to read a book, finish a MOOC or learn a topic on YouTube with little retention of the applied topic. Anytime you're learning a new concept, you **can't neglect the fundamentals** and always keep the big picture in mind because:

- You have to know coding basics before you can even debug the implementations of your DS/ML libraries
- You have to know how to preprocess your data before applying machine learning methods accurately
- You have to know statistical inference before you make sense of your visualization.

Remember to don't just jump into a course without taking the time to ask yourself "why" and "how" is this being used (even this course). It's only when you start asking yourself questions about the problem where you begin to connect the dots. And by having a good foundation of the basics, those dots that you connect with new concepts will be retained a lot longer.

Focus on the Foundations

Finally, if you're new to data science or machine learning, **it may be tempting to learn everything as quick as possible.**

I highly recommend for you to take the time to learn the basics of each concept before you start implementing:

1. Summarizing descriptive statistics of your data,
2. Handling different types of variables (categorical, ordinal, numerical)
3. Visualizing trends and finding correlations

4. Preprocessing techniques
5. Train/Test Split & Cross Validation
6. Types of evaluation metrics

It's way better to understand one thing really well and move forward slowly than to barely understand something and move forward quickly. Everything is built upon each other.

If you don't have a solid understanding of the foundations: Data cleaning, pre-processing, and visualization **then you're not ready** to move forward to the more advanced topics of feature engineering, hyper parameter tuning, and model selection.

Start from the basics and implement new concepts as you go. Learn the concept really well and then start implementing it. **Focus on why it works and when it's used.** Repeat this over and over. Before you know it, you've probably built a whole end-to-end project that you should be really proud of.

The Statistics You Need to Know

I would say the **20%** of statistics that accounts for **80%** of all data science problems can be captured by:

1. **Statistical Models** Linear regression, logistic regression, one multi-purpose ML model like xgboost, and one unsupervised learning algorithm, probably k-means
2. **Basic feature engineering:**
    a. Avoiding colinearity
    b. Generating simple transformations of numerical variables
    c. Handling categorical variables (normally one-hot encoding works just fine)
    d. Understanding which variables can actually be used in prediction based on timing/business constraints
3. **Evaluating quality of fit**
    a. Working on a train/test split dataset
    b. R^2 for regression and accuracy/specificity/recall for classification should handle most cases.

Some people may argue that this list is not entirely correct, but in a general sense this is basically the core statistical concepts you should know. Be familiar with this and you'll have a solid grasp on the field. **The goal here is to be able to explain what you have done and justify how you did it.**

Data Science 101

The 2 Main Data Scientist Roles

Product Data Scientist - They focus on the product. These data scientists are more tech-savvy and are working on building a product for the end user. You'll need to be more versed in development, coding, implementation, and ETL.

Business Data Scientist - They focus on the insights. These data scientists are more business-savvy and are focused on the decisions your team make. You'll need to be more versed in communication, business understanding, marketing, forecasting, and profitability analysis.

All Data Scientists - At the end of the day, each of these two roles will have one main thing in common: **they are both problem solvers.** Your goal here is to take data and solve a problem out of it, using whatever tools and skills you need. You need to be able to perform an end-to-end project and solve real problems. This includes performing statistical analysis, communicating with stakeholders, designing pipelines, creating machine learning models, and etc.

Data Analytics VS Data Science

I'll summarize it in two sentences:

- **Data analytics** is about using data to explain the **past**.
- **Data science** is about using data to explain the **future**.

The primary responsibilities for Data Scientists are:

- Use quantitative tools to uncover opportunities, set team goals and work with cross-functional partners to guide the product roadmap
- Explore, analyze and aggregate large data sets to provide actionable information, and create intuitive visualizations to convey those results to a broad audience
- Design informative experiments considering statistical significance, sources of bias, target populations and potential for positive results
- Collaborate with Engineers on logging, product health monitoring and experiment design/analysis

Data, Analytics role is NOT characterized by the following:

- Working in a centralized service organization, consulting and responding to ad-hoc requests. Data Scientists in the Analytics org are embedded in product teams and expected to be involved in driving both the high-level direction and the day-to-day progress of the team.
- Specializing in ETL, pipelining, and other data infrastructure tasks. While it is common for Data Scientists to build and maintain some pipelines, the Data Engineering team focuses more heavily on this.
- Taking an academic approach to Facebook data and publishing white papers. That activity typically resides in the Core Data Science team.
- Building, training and deploying machine learning models to production. Although a few Analytics roles will involve modeling, most production models are handled by the Engineering team.

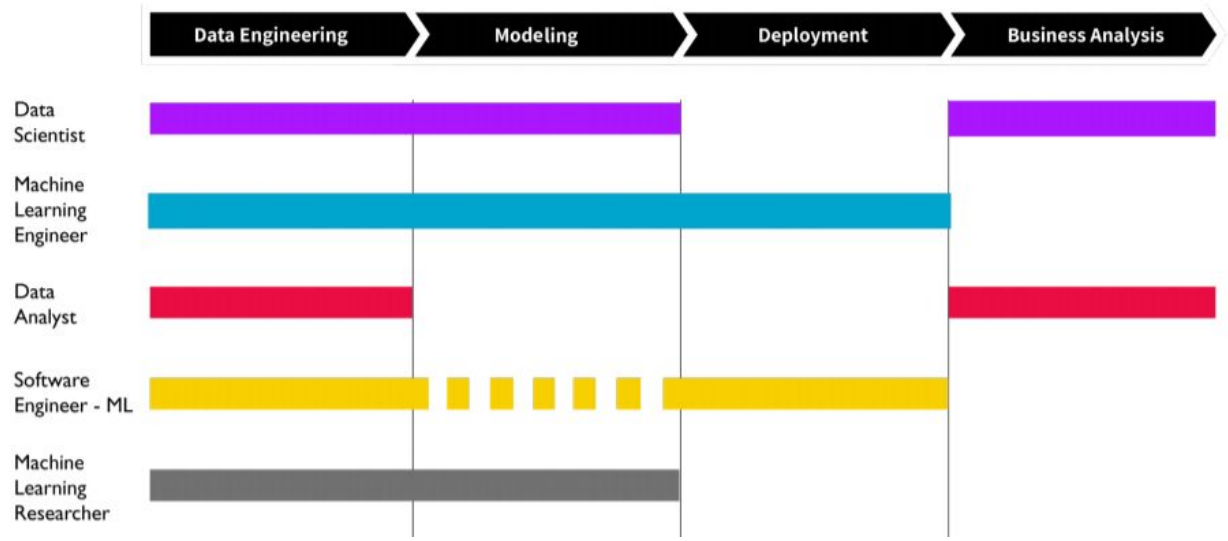
Summary

SQL is all about data.
Statistics is all about inference.
Data Visualization is all about insight.
Machine Learning is all about prediction.
Communication is all about decision making.
*Data science is **all the above.***

The Data Science Team

Ideal data scientist is not an expert in software development and databases. You just need familiarity with it. In a big team in industry its the job of data engineers to get you the data, who are more specialized in databases. Then you also have system admins and database admins maintaining that side of things. Software developers work with devops to integrate your algorithms into production. Statistics is probably the #1 thing you should know as a data scientist and is the expertise you are bringing to the table which shouldn't really expect any other IT staff to know.

The Data Science Team Chart



I. Data Scientist

The data scientist will have a mix of software engineering, modeling, and business analytics. These people will do a similar role as data analysts, but are a step above them because of their knowledge in machine learning. A good understanding of statistics, machine learning, analytics, and communication is key for this role.

Tools Used:

- Scikit-Learn, TensorFlow, Pytorch, NumPy, and etc. for modeling
- SQL for querying data
- Python or R for business analysis and programming

- Tableau or Excel for data visualization and reporting
- Git, Jupyter Notebook, and Command Line for version control and collaboration

II. Data Analyst

The data analysts will usually create reports for their team and present their findings. They should have a good understanding on how to query from a database and know how to monitor/track different metrics and KPIs. Communication is definitely key for this role.

Tools Used:

- SQL for querying data
- Python or R for business analysis and programming
- Tableau or Excel for data visualization and reporting

III. Machine Learning Engineer

Machine Learning Engineers normally focus on the implementation and deployment of the models. They will need a good understanding on optimization, as well as software engineering. They are less focused on the insights you get from your models, but more on the productionalization and application of it.

Tools Used:

- Scikit-Learn, TensorFlow, Pytorch, NumPy, and etc. for modeling
- SQL for querying data
- Python, Java, C++, and etc. for deployment and OOP
- AWS, GCP, Azure, etc. for cloud technologies
- Git, Jupyter Notebook, and Command Line for version control and collaboration
- JIRA for issue tracking product

IV. Software Engineer - ML

Software Engineers bring in a lot of programming and algorithmic coding to the table. They focus more on the underlying code and are most likely all technical. They don't really work on the models perse but are involved with the deployment and integration of it into their product. These engineers focus on applications usually powered by machine learning or deep learning.

Tools Used:

- Scikit-Learn, TensorFlow, Pytorch, NumPy, and etc. for modeling
- SQL for querying data
- Python, Java, C++, and etc. for deployment and OOP
- AWS, GCP, Azure, etc. for cloud technologies
- Git, Jupyter Notebook, and Command Line for version control and collaboration
- JIRA for issue tracking product

V. Machine Learning Researcher

Machine Learning Researchers have a heavy focus on algorithms, experimentation, optimization, and sometimes deployment. These people have a particular focus on their domain, such as NLP, speech recognition, and computer vision.

Tools Used:

- Scikit-Learn, TensorFlow, Pytorch, NumPy, and etc. for modeling
- SQL for querying data
- Git, Jupyter Notebook, and Command Line for version control and collaboration
- Papers, Conferences, Reddit, Twitter, and etc. for research and updates

The Data Science Pipeline

OSEMN Pipeline

Understanding the typical work flow on how the data science pipeline works is a crucial step towards business understanding and problem solving. If you are intimidated about how the data science pipeline works, say no more. I'll make this simple for you. I found a very simple acronym from Hilary Mason and Chris Wiggins that you can use throughout your data science pipeline. That is O.S.E.M.N.

OSEMN Pipeline

O — Obtaining our data
S — Scrubbing / Cleaning our data
E — Exploring / Visualizing our data will allow us to find patterns and trends
M — Modeling our data will give us our predictive power as a wizard
N — Interpreting our data

Business Question

So before we even begin the OSEMN pipeline, the most crucial and important step that we must take into consideration is understanding what problem we're trying to solve. Let's say this again. Before we even begin doing anything with "Data Science", we must first take into consideration what problem we're trying to solve. If you have a small problem you want to solve, then at most you'll get a small solution. If you have a BIG problem to solve, then you'll have the possibility of a BIG solution.

Ask yourself:

- How can we translate data into dollars?
- What impact do I want to make with this data?
- What business value does our model bring to the table?
- What will save us lots of money?
- What can be done to make our business run more efficiently?

Knowing this fundamental concept will bring you far and lead you to greater steps in being successful towards being a Data Scientist. No matter how well your model predicts, no matter how much data you acquire, and no matter how OSEMN your pipeline

is... your solution or actionable insight will only be as good as the problem you set for yourself.

“Good data science is more about the questions you pose of the data rather than data munging and analysis” — Riley Newman

O — Obtaining our data

You cannot do anything as a data scientist without even having any data. As a rule of thumb, there are some things you must take into consideration when obtaining your data. You must identify all of your available datasets (which can be from the internet or external/internal databases). You must extract the data into a usable format (.csv, json, xml, etc..)

Skills Required:

- Database Management: MySQL, PostgresSQL,MongoDB
- Querying Relational Databases
- Retrieving Unstructured Data: text, videos, audio files, documents
- Distributed Storage: Hadoops, Apache Spark/Flink

S — Scrubbing / Cleaning our data

This phase of the pipeline should require the most time and effort. Because the results and output of your machine learning model is only as good as what you put into it. Basically, garbage in garbage out. Your objective here is to examine the data, understand every feature you’re working with, identify errors, missing values, and corrupt records, clean the data, and replace and/or fill missing values.

Skills Required:

Scripting language: Python, R, SAS
Data Wrangling Tools: Python Pandas, R
Distributed Processing: Hadoop, Map Reduce / Spark

E — Exploring / Visualizing our data

Now during the exploration phase, we try to understand what patterns and values our data has. We’ll be using different types of visualizations and statistical testing

to back up our findings. This is where we will be able to derive hidden meanings behind our data through various graphs and analysis. Go out and explore!

Your objective here is to find patterns in your data through visualizations and charts and to extract features by using statistics to identify and test significant variables.

Skills Required:

- Python: Numpy, Matplotlib, Pandas, Scipy
- R: GGplot2, Dplyr
- Inferential statistics
- Experimental Design
- Data Visualization

M — Modeling our data

Models are general rules in a statistical sense.Think of a machine learning model as tools in your toolbox. You will have access to many algorithms and use them to accomplish different business goals. The better features you use the better your predictive power will be. After cleaning your data and finding what features are most important, using your model as a predictive tool will only enhance your business decision making.

Your objective here is to perform in-depth analytics by creating predictive models. Machine learning algorithms may be better at predicting, detecting, and processing patterns than you. But they can't reason. And that's where you com in!

Machine Learning Analogy

Think of Machine learning algorithms as us (students). Just like students, all of the algorithms learn differently. Each has their own sets of qualities and way of learning. Some algorithms learn faster, while others learn slower. Some algorithms are lazy learners (i.e. KNN) while others are eager learners. Some algorithms learn from parametric data (i.e. Linear Regression) while others learn from non-parametric data (i.e. Decision Trees). Just like students, some algorithms perform better for a certain problem, whereas others may perform better on another, i.e. linearly separable vs non-linearly separable problems. Just like students, these algorithms learn through various patterns and relationships from the data. That's why it's important to perform EDA and visualizations because if you don't see a pattern, you're model won't as well. Just like

students, if you give an algorithm garbage information to learn, then it won't perform as good. That's why it's important to choose your features correctly and that each has some relationship to the problem. So remember, choose the algorithm that is the most appropriate for your problem. Because there is no "best" learner but there is always the "right" learner.

"Machines can predict the future, as long as the future doesn't look too different from the past."

Skills Required:

- Machine Learning: Supervised/Unsupervised algorithms
- Evaluation methods: MSE, Precision/Recall, ROC/AUC
- Machine Learning Libraries: Python (Sci-kit Learn) / R (CARET)
- Linear algebra & Multivariate Calculus

N — Interpreting our data

The most important step in the pipeline is to understand and learn how to explain your findings through communication. Telling the story is key, don't underestimate it. It's about connecting with people, persuading them, and helping them. The art of understanding your audience and connecting with them is one of the best part of data storytelling.

The power of emotion

Emotion plays a big role in data storytelling. People aren't going to magically understand your findings. The best way to make an impact is telling your story through emotion. We as humans are naturally influenced by emotions. If you can tap into your audiences' emotions, then you my friend, are in control. When you're presenting your data, keep in mind the power of psychology. The art of understanding your audience and connecting with them is one of the best parts of data storytelling.

The objective you should have for yourself is to be able to identify business insights, relate back to the business problem, visualize your findings accordingly, keeping it simple and priority driven, and to tell a clear and actionable story.

Skills Required:

- Business Domain Knowledge
- Data Visualization Tools: Tableau, D3.JS, Matplotlib, GGplot, Seaborn

- Communication: Presenting/Speaking & Reporting/Writing

Conclusion

Data is not about statistics, machine learning, visualization, or wrangling. **Data is about understanding.** Understanding the problem and how you can solve it using data with whatever tools or techniques you choose. Understand your problem. Understand your data. **And the rest will follow.**

Most of the problems you will face are, in fact, engineering problems. Even with all the resources of a great machine learning god, most of the impact will come from great features, not great machine learning algorithms.

So, the basic approach is:

- Make sure your pipeline is solid end to end
  - Start with a reasonable objective
  - Understand your data intuitively
- Make sure that your pipeline stays solid

Machine Learning 101

What is Machine Learning?

Machine Learning involves a computer to recognize patterns by examples, rather than programming it with specific rules. These patterns are found within Data.

Machine = Your machine or computer
Learning = Finding patterns in data

Machine Learning is about: Creating algorithms (a set of rules) that learns from complex functions (patterns) from data to make predictions on it.

In short, Machines can predict the future, as long as the future doesn't look too different from the past.

Essentially, it can be summarized in 3 Steps:
1. It takes some data
2. Finds pattern from the data
3. Predicts new pattern from the data

Applications of Machine Learning

Before we get started, here is a quick overview of what machine learning is capable of:

- **Healthcare**: Predicting patient diagnostics for doctors to review
- **Social Network**: Predicting certain match preferences on a dating website for better compatibility
- **Finance**: Predicting fraudulent activity on a credit card
- **E-commerce:** Predicting customer churn
- **Biology**: Finding patterns in gene mutations that could represent cancer

How Do Machines Learn?

To keep things simple, just know that **machines "learn" by finding patterns in similar data.** Think of data as information you acquire from the world. The more data given to a machine, the "smarter" it gets.

But not all data are the same. Imagine you're a pirate and your life mission was to find the buried treasure somewhere on the island. In order to find the treasure, you're going to need sufficient amount of information. Like data, this information can either lead you to the right direction or the wrong direction. **The better the information/data that is obtained, the more uncertainty is reduced, and vice versa.** So it's important to keep in mind the type of data you're giving to your machine to learn.

Nonetheless, after a sufficient amount of data is given, then the machine can make predictions. **Machines can predict the future, as long as the future doesn't look too different from the past.**

Machine "learns" really by using old data to get information about what's the most likelihood that will happen. If the old data looks a lot like the new data, then the things you can say about the old data will probably be relevant to the new data. It's like looking back to look forward.

Types of Machine Learning

There are three main categories of machine learning:

1. Supervised learning: The machine learns from labeled data. Normally, the data is labeled by humans.
2. Unsupervised learning: The machine learns from unlabeled data. Meaning, there is no "right" answer given to the machine to learn, but the machine must hopefully find patterns from the data to come up with an answer.
3. Reinforcement learning: The machine learns through a reward-based system.

Supervised Machine Learning

Supervised learning is the most common and studied type of learning because it is easier to train a machine to learn with labeled data than with unlabeled data. Depending on what you want to predict, supervised learning can be used to solve two types of problems: regression or classification.

Regression

If you want to predict continuous values, such as trying to predict the cost of a house or the weather outside in degrees, you would use regression. This type of problem doesn't have a specific value constraint because the value could be any number with no limits.

Classification

If you want to predict discrete values, such as classifying something into categories, you would use classification. A problem like, "Will he make this purchase" will have an answer that falls into two specific categories: yes or no. This is also called a binary classification problem.

Unsupervised Machine Learning

Since there is no labeled data for machines to learn from, the goal for unsupervised machine learning is to detect patterns in the data and to group them. Unsupervised learning are machines trying to learn "on their own"without help. Imagine someone throwing you piles of data and says "Here you go boy, find some patterns and group them out for me. Thanks and have fun."

Depending on what you want to group together, unsupervised learning can group data together by: clustering or association.

Clustering Problem

Unsupervised learning tries to solve this problem by looking for similarities in the data. If there is a common cluster or group, the algorithm would then categorize them in a certain form. An example of this could be trying to group customers based on past buying behavior.

Association Problem

Unsupervised learning tries to solve this problem by trying to understand the rules and meaning behind different groups. Finding a relationship between customer purchases is a common example of an association problem. Stores may want to know what type of products were purchased together and could possibly use this information to organize the placement of these products for easier access. One store found out that there was a strong association between customers buying beer and diapers. They deduced from this statement that males who had gone out to buy diapers for their babies also tend to buy beer as well.

Reinforcement Machine Learning

This type of machine learning requires the use of a reward/penalty system. The goal is to reward the machine when it learns correctly and to penalize the machine when it learns incorrectly.

Reinforcement Machine Learning is a subset of Artificial Intelligence. With the wide range of possible answers from the data, the process of this type of learning is an iterative step. It continuously learns.

Examples of Reinforcement Learning:

- Training a machine to learn how to play (Chess, Go)
- Training a machine how to learn and play Super Mario by itself
- Self-driving cars

Machine Learning Algorithms

With big loads of data everywhere, the power in deriving meaning behind all of it relies on the use of Machine Learning. **Machine learning algorithms are used to learn the structure of the data.** Is there a structure in the data? Can we learn the structure from the data? And if we can, we can then use it for prediction , description, compression, and etc.

Each machine learning algorithm/model has its own strengths and weaknesses. A problem that resides in the machine learning domain is the concept of understanding the details in the algorithms being used and its prediction accuracy. Some models are easier to interpret/understand but lack prediction power. Whereas other models may have really accurate predictions but lack interpretability.

This section will not go into detail on what goes inside each algorithm, but will cover the high-level overview of what each machine learning algorithm does.

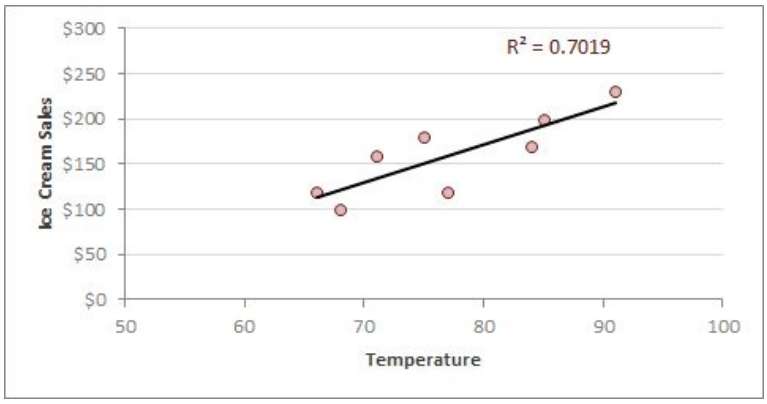
Let's talk about the 7 important machine learning algorithms:

1. Linear Regression
2. Logistic Regression
3. K-Nearest Neighbors (KNN)
4. Support Vector Machines (SVM)
5. Decision Tree
6. Random Forest
7. Gradient Boosting Machine

Linear Regression

This is the go to method for regression problems. The linear regression algorithm is used to see a relationship between the predictor variables and explanatory variable. This relationship is either a positive, negative, or neutral change between the variables. In its simplest form, it attempts to fit a straight **line** to your training data. This **line** can then be used as reference to predict future data.

Imagine you're an **ice cream man.** Your intuition from previous sales was that you found yourself selling more ice cream when it was hotter outside. Now you want to know how much to sell your ice cream at a certain temperature. We can use linear regression to predict just that! The linear regression algorithm is represented as a formula: **y = mx + b.** Where "y" is your dependent variable (ice cream sales) and "x" is your independent variable (temperature).



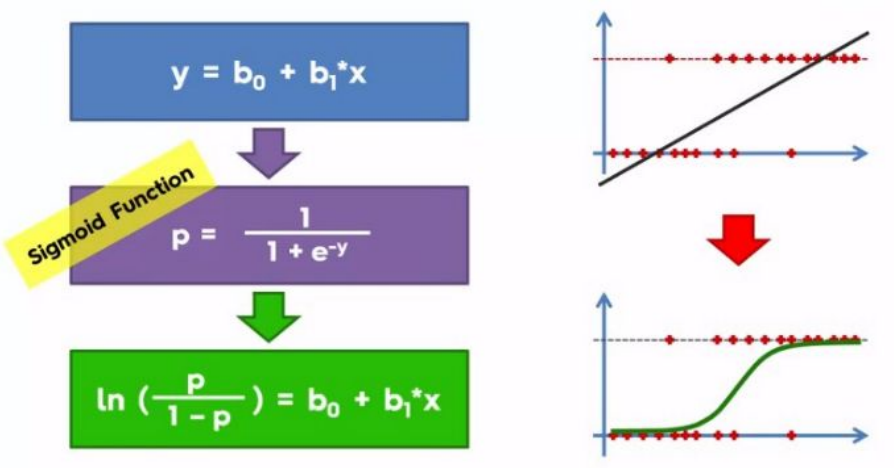
Example: If the temperature is about **75** degrees outside, you would then sell the ice cream at about **$150.** This shows that as temperature increases, so does ice cream sales.

Strengths: Linear regression is very fast to implement, easy to understand, and is less prone to overfitting. It's a great go-to algorithm for using it as your first model and works really on linear relationships

Weaknesses: Linear regression performs poorly when there are non-linear relationships. It is hard to be used on complex data sets.

Logistic Regression

This is the go to method for **classification** method and is commonly used for **interpretability**. It is commonly used to predict the **probability** of an event occurring. Logistic regression is an algorithm borrowed from statistics and uses a **logic/sigmoid function** (Purple Formula) to transform its output, making it either 0 or 1.



Example: Imagine you're a banker and you wanted a machine learning algorithm to predict the probability of a person paying you back the money. They will either: pay you back or not pay you back. Since a probability is within the range of (0–1), using a linear regression algorithm wouldn't make sense here because the line would extend pass 0 and 1. You can't have a probability that's negative or above 1.

Strengths: Similar to its **sister**, linear regression, logistic regression is easy to interpret and is less prone to overfitting. It's fast to implement as well and has surprisingly great accuracy for its simple design.

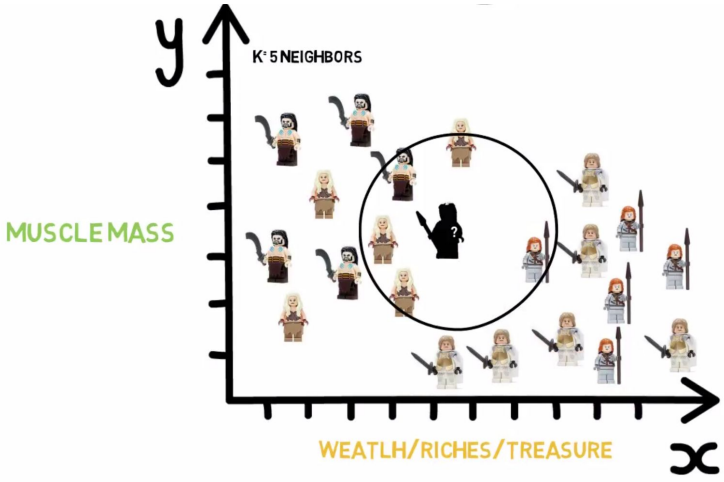
Weaknesses: This algorithm performs poorly where there are multiple or non-linear decision boundaries or capturing complex relationships.

K-Nearest Neighbors

K-Nearest Neighbors algorithm is one of the **simplest classification** techniques. It's an algorithm that classifies objects based on its closest training example in its featured space. The **K** in **K-**Nearest Neighbors refers to the number of nearest neighbors the model will be used for its prediction.

How it Works:

1. Assign K a value (preferably a small odd number)
2. Find closest number of K points
3. Assign the new point from the majority of classes



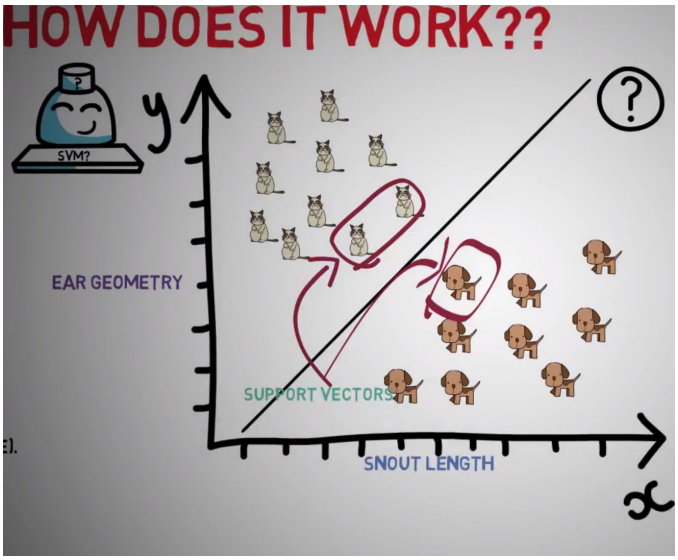
Example: Looking at our unknown person in the graph, where would you classify him under as: **Dothraki** or **Westerosian**? We'll be assigning **K=5**. So in this case, we'll look at the 5 closest neighbors to our unassigned person and assign him to the majority class. If you picked **Dothraki**, then you are correct! Out of the **5** neighbors, 4 of them were Dothrakis and 1 was Westerosian.

Strengths: This algorithm is good for large data, it learns well on complex patterns, it can detect linear or non-linear distributed data, and is robust to noisy training data.

Weaknesses: It's hard to find the right K-value, bad for higher dimensional data, and requires a lot of computation when fitting larger amounts of features. It's expensive and slow.

Support Vector Machine

If you want to compare **extreme** values in your dataset for classification problems, **SVM** algorithm is the way to go. It draws a decision boundary, also known as a **hyperplane that** best segregates the two classes from one another. Or you can think of it as an algorithm that looks for some pattern in the data points and find a best line that can separate the pattern(s).



S — Support refers to the extreme values/points in your dataset.

V — Vector refers to the values/points in dataset / feature space.

M — Machine refers to the machine learning algorithm that focuses on the support vectors to classify groups of data. This algorithm literally only focuses on the extreme points and ignores the rest of the data.

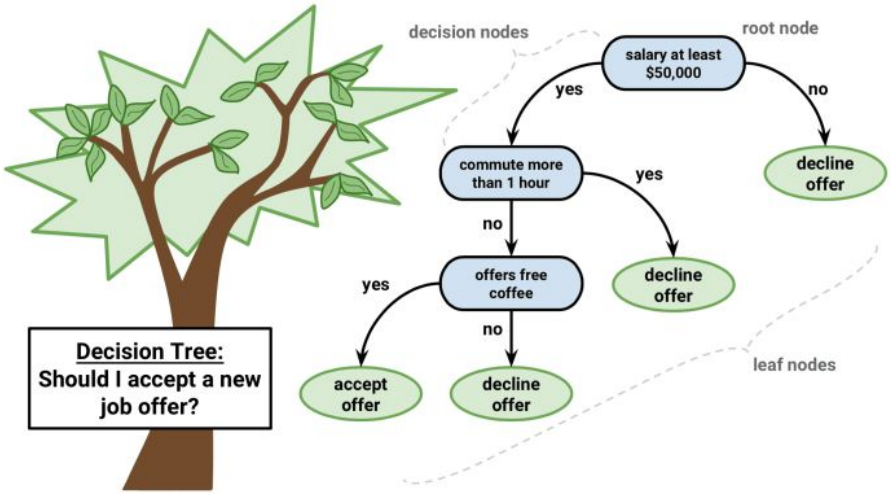
Example: This algorithm only focuses on the extreme values (**support vectors**) to create this decision line, which are the **two cats** and **one dog** circled in the graph.

Strengths: This model is good for classification problems and can model very complex dimensions. It's also good to model non-linear relationships.

Weaknesses: It's hard to interpret and requires a lot of memory and processing power. It also does not provide probability estimations and is sensitive to outliers.

Decision Tree

A decision tree is made up of **nodes**. Each node represents a question about the data. And the branches from each node represents the possible answers. Visually, this algorithm is very easy to understand. With every decision tree, there is always a **root node** and this represents the top most question. The order of importance for each feature is represented in a top-down approach of the nodes. The higher the node the more important its property/feature.

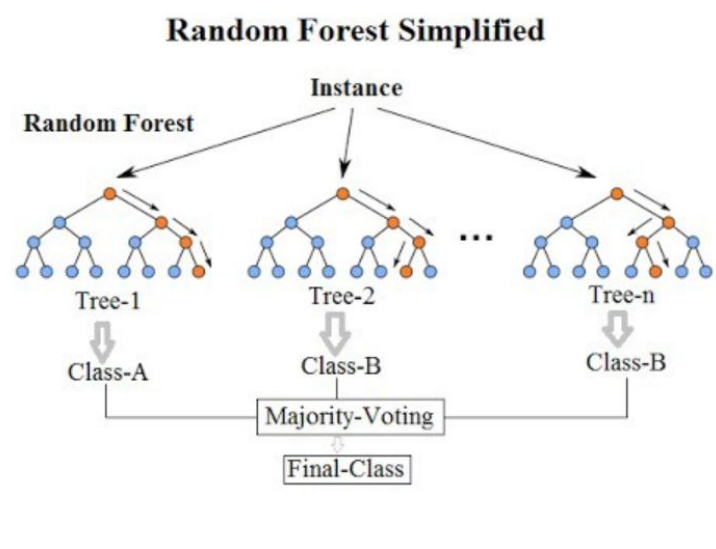


Strengths: The decision tree is a very easy to **understand** and visualize. It's fast to learn, robust to outliers, and can work on non-linear relationships. This model is commonly used to understand what features are being used, such as medical diagnosis and credit risk analysis. It also has a built in feature selection.

Weaknesses: The biggest drawback of a single decision tree is that it loses its predictive power from not collecting other overlapping features. A downside to decision trees is the possibility of building a complex tree which do not generalize well to future data. hard to interpret, duplication is possible. Decision trees can be unstable because it naturally has low variance of features.

Random Forest

One of the most used and powerful supervised machine learning algorithms for **prediction accuracy**. Think of this algorithm as a bunch of decision trees, instead of one single tree like the Decision Tree algorithm. This grouping of algorithms, in this case decision trees, is called an **Ensemble Method**. It's accurate performance is generated by averaging the many decision trees together. Random Forest is naturally hard to interpret because of the various combinations of decision trees it uses. But if you want a model that is a **predictive powerhouse,** use this algorithm!

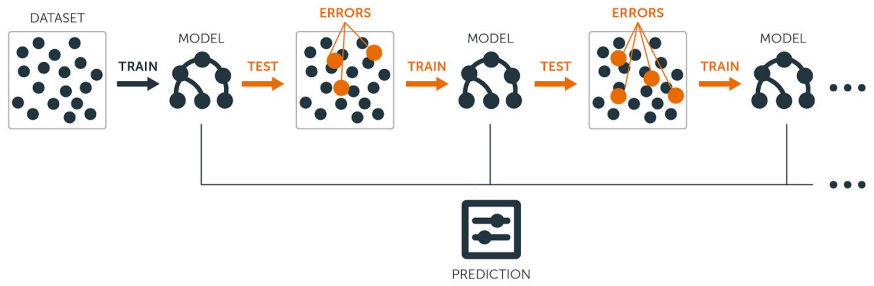


Strengths: Random Forest is known for its great **accuracy**. It has automatic feature selection, which identifies what features are most important. It can handle missing data and imbalanced classes and generalizes well.

Weaknesses: A drawback to random forest is that you have very little control on what goes inside this algorithm. It's hard to interpret and won't perform well if given a set of bad features.

Gradient Boosting Machine

Gradient Boost Machine is another type of **Ensemble Method,** where it aggregates many models together and takes a consensus of their predictions. A simple key concept to remember about this algorithm is this: **it trains a weak model into a stronger model.** Try not to over-complicate things here.



How It Works: You're first model is considered the "weak" model. You train it and find out what errors your first model produced. The second tree would then use these errors from the first tree to re-calibrate its algorithm and emphasis more priority to the errors. The third tree would repeat this process and see what errors the second tree made. And this would just repeat. **Essentially, this model is building a team of models that works together to solve their weaknesses.**

Strengths: Gradient Boosting Machines are very good to use for predictions. It's one of the best off-the-shelf algorithm used for great accuracy with decent run time and memory usage.

Weaknesses: Like other ensemble methods, GBM lacks interpretability. It is not well suited for handling large dimensions/features because it could take a lot of training process and computation. So a balance of computational cost and accuracy is a concern.

Your Data Science Portfolio

How to Craft Your Projects

If you don't have experience in data science, you have to start doing some personal projects. My advice is for you to actually choose a dataset that you find interesting. The quality of work that you do will show during your analysis and documentation.

One thing to always keep in mind is to know your "why". **When an interviewer asks about a project** that you've worked on recently, this is what they want to hear:
1. How you started the project
2. Why you thought it was important
3. Your process
4. Your result
5. What you learned

If you can tell a story about your dataset, emphasize why your analysis is so interesting/compelling, and let them know about all the hard work you've done and learned from, ***then that's a worthy project****. Put in the extra effort to document your work, structure your story, and write about the interesting things you've found in your analysis. Make sure you give it your all in whatever work you do.* ***It'll pay off.***

Last but not least, share it. Put it on GitHub or Kaggle. Get feedback from others. Improve your project over time. And most definitely: have fun!

The Projects You Should Do For Jobs

Your personal projects are the next best thing to experience that you can show employers about your data science skill set. But there are good and bad projects to be mindful of. Definitely do not put the Titanic dataset or IRIS dataset as your project because everybody has done them and it’s too saturated. When designing your project, you should always ask yourself “How do I set myself apart?”

Here’s a few things to keep in mind (what impresses me?):

- **When you scrape your own data.** it requires more engineering effort, and it tells how hard someone did on their project.
- **When you do advance feature engineering** (cluster analysis to include in feature engineering) or (PCA for dim reduction) or just being creative with adding in new features and see if there is added lift to your analysis.
- **When you productionalize it.** (flask wrapper on it and puts it on your website) It shows you're doing that extra step and someone can see this project from the beginning to the end
- **When you do descriptive analysis.** Such as cleaning your data, how do you handle null values? Do you remove it? Do you impute it with the mean? These decisions have a large impact on your analysis. Talking about correlation between variables and if there are predictive power.

The 6 Types of Projects

1. Doing a multiple linear regression OR logistic regression (both linear models, easy to explain in business context).

- These models are easy to explain because they are linear, meaning the feature importance is directly relatable to the outcome variable.

2. Some form of classification problem

- evaluation metrics (accuracy, precision, recall, f1 score)
- each individual problem has a particular error that's more weighted
- naive bayes, KNN, decision trees, random forest, or SVM

3. Regression problem

- Evaluation is importance (MAE, MSE, RMLSE, RMSE)
- random forest regressor, lasso/ridge regression

4. Time Series Analysis

- good for finance or any field using time series analysis
- rolling averages, ARIMA, SARIMA
- good understanding of how data works over time (this is fundamentally diff than other dataset works)

5. Clustering project (unsupervised learning)

- KMeans, Hierarchical clustering, DBSCAN
- Group or cohort analysis
- can be used to create features in your data set (feature engineering by the cluster grouping)

6. Natural Language Processing

- Sentiment Analysis (analyzing the sentiment of a sentence to determine positive or negative reviews)

- Text Preprocessing (using TFIDF, CountVectorizer, Stemming, Stop Word Removal)
- Topic Modeling with LDA

The Importance of Kaggle

If you're concerned about not having enough practice in honing your DataScience and MachineLearning skills then here are some things you can do on Kaggle. I highly recommend this site to everybody. Whether you're a beginner or expert, this site has a bunch of helpful resources for you to learn and get new ideas from:

1. **Kaggle Learn -** https://lnkd.in/gghHUH2
   Kaggle has their own education track where you can start learning the foundations. The course includes:
   - Machine Learning
   - Python
   - Data Visualization
   - SQL
   - R
   - Deep Learning

   2. Kaggle Kernels - https://lnkd.in/e_VcNpk
   There's no doubt that one of Kaggle's biggest features is its access to user's Kernels/Notebooks. You literally have a person's documentation and code right in front of you. My advice is for you to choose a problem you want to learn (classification, regression, EDA, etc..) and find a top-voted Kernel for you to study off from.

   3. Kaggle Competitions - https://lnkd.in/epb9c8N
   This is a must for those that are really interested in ML. Kaggle has many competitions for you to choose from and it's great practice for trying out different models & improve your scores. My tip for you is to look at the Kenerls section in the competition you're interested in. Many people upload their kernels with base models you can use and optimize off from.

   4. Datasets - https://lnkd.in/gZjpxtX
   There are thousands of datasets you can download here all from different domains!

How to Build Your Resume

Remember, **your resume is only good for one thing**: to just get a follow up call. Interviewers only spend about **7-10 seconds** reviewing your resume before putting you into their denied / approved pile. So that's why it is so important to craft a great resume. It's all about show and tell. **You're basically marketing and selling yourself on paper.**

Professional Summary (3-5 lines)

What professional role do you want to achieve?
Who are you?
What main skills do you want them to know about you?

Education & Certifications

Put this in order of importance
Include relevant certifications to the role
Make sure to include the year of graduation / enrollment

Management & Technical Skills

Bullet point keywords/skills related to the position
Include relevant tools and technologies

Professional Experience (3-5 lines)

Must be in descending chronological order
Include Company Name, Location, Job Title, and Date
List your responsibility, description, and what you've achieved. Be specific

Personal Projects

Include relevant technical projects that demonstrate your capabilities
Coding projects, DS/ML Projects, or competitions

Awards / Volunteer Work / Publications

Highlight any achievements worth noting

How to Create Your LinkedIn Profile

Recruiters are heavily using LinkedIn for their job recruitment strategy and knowing how to optimize your profile to beat the ranking algorithm is key. Use this template to write out your profile description:

About

I am a *<Your Role / Position>* with a passion for *<Field of Interest>*. I've been using *<Tool>* for *<Amount>* of years and am interested in creating positive business impact with my skills.

Key Competencies

- Leadership and Management
- Project Management
- Stakeholder Management
- Strategic Operations
- Business Development
- Program Management

Key Technologies and Methodologies

- Programming: Python, R, Java
- Database: SQL,Postgres, NoSQL, MySQL
- Statistics and Probability
- Machine Learning: Regression, Classification, Cluster Analysis
- Agile
- Waterfall

Key Achievements

- Delivered business plans supporting $5m revenue generation
- Managed budgets of up to $25m
- Managed $10m in capital expenditure
- Achieved cost reduction in excess of 40%

Contact

Please contact me at *<Your Email>* to find out more on how I can best help you or your organization.

Interview Prep and Templates

How to Follow Up After an Interview

It is crucial to always follow up after an interview. Not everybody does it, and this simple task will do wonders. And why should you do it... **Because you want to be remembered.**

There's two rules you should follow on how to do this:

1. **3-8 hours:** Send a "thank you" email within 3-8 hours of the interview.
2. **10 days:** Send a "follow up" email within 10 days of the interview

Thank You Email Template

"Dear <Interviewer Name>,

Thank you very much for taking the time to see me today. It was great to meet you and I really enjoyed talking with you today. I'm definitely interested in this role.

<Insert line reminding them about something positive in the interview OR something you wish you had mentioned>

<Insert line reminding them about something positive in the interview OR something you wish you had mentioned>

<Insert line reminding them about something positive in the interview OR something you wish you had mentioned>

I believe I am a good candidate for this role because <insert 3 reasons>.

Hope you have a great rest of your day and I'm looking forward to hearing from you.

Best,
<Your Name

Follow Up Email Template

"Dear <Interviewer Name>,

I had an interview with you on <interview date> and really enjoyed meeting with you. I'm still very interested in this role of <job title> and was wondering if you had any feedback or updates for me at this time. Thank you.

Best,
<Your Name>

How to Write a Cover Letter

You don't have to write a cover letter for every job you apply for. I'd recommend to only use it for a specific company where you have a specific interest in working for. Nonetheless, it's a great way to stand out from the crowd.

Cover Letter Template

"Dear <Name>,

I am writing in regards to your job opening of <Position Title>. I found the position from <Source> and I am really interested in the role. As a person with great experience in <Job Title>, I am highly skilled in <Hard Skills>.

The reason why I would love to join <Company Name> is because <2-3 Reasons>. I also hold a <Degree, Cert> and can perform any tasks related to <JD Responsibilities>.
From my career history, I have been exposed to many challenges and have grown to demonstrate high levels of responsibilities from my previous company. <List 2-3 achievements>

It'll be a pleasure to have the opportunity to meet you for an interview. I look forward to your response and hope you have a great day.

Best,
<Your Name>

The Behavioral Interview

How to Answer Any Behavioral Questions

The behavioral questions that come along in the interview process is designed to filter out candidates based upon their qualities as a person. It's a filter mechanism to see what type of person you are, what are your values, what is your work ethic, and what are your goals for even applying for this position.

You should answer these behavioral questions based upon your own personal and professional experience. Keep it honest and don't ever lie about your answers.There are no 'correct' answers to these questions and your main goal here is to tell **your story**.

To answer these questions, master the CARL method.

CARL Method

C — Context: What role were you in and define the task you went through.
A — Action: What did you do and how did you do it
R — Result: What was the outcome
L — Learning: What key lessons did you take away from that experience

Most Common Type of Behavioral Questions

There are 5 main types of interview questions you should be aware of and are commonly asked in many interviews. Make sure you have a complete story for each and you're super comfortable in answering these questions:

1. **Tell me about a time where you had to deal with a difficult teammate or client and how did you handle it?**
    a. This question is designed to gauge how you work under pressure. There will always be challenges in the workplace and how you respond to these situations will determine how you will fit into their company.
2. **Give me an example of a time when you had to challenge a management decision.**
    a. There will always be someone over you that will manage your work and monitor how you're doing. Nobody is always right and this question is

geared towards your beliefs. They want to know how assertive you are and whether you are passive or aggressive in this situation.

3. **Tell me a time when you demonstrated ownership**.
    a. This question is about being independent. They want somebody who owns their work/project and is self motivated. Make sure you elaborate about why you demonstrated ownership and how it fits into the big picture.
4. **Tell me about a time when you had to quickly adjust your work priorities to meet changing demands.**
    a. This question is about how you manage priorities and distribute your tasks properly. The work environment will sometimes be hectic and understanding how to adapt to the situation is a skill any employer will need.
5. **Have you ever failed in a task or project? If so, tell me about it.**
    a. This is not a trick question. Because everybody has failed and nobody is perfect. Choose a story that aligns with the position you applied for and turn a negative into a positive. What they really care about is the lesson you got out of that experience.
6. **Tell me a little about yourself.**
    a. Your goal here is to explain the What, Who, and Why you are a great candidate for this role. You should keep it short and only talk about the most relevant attributes for the role and include any necessary tools that they are currently using in their company.
7. **What is your biggest weakness?**
    a. This is also not a trick question. Just state your weakness and clearly explain how you overcame it. If you haven't fixed it, then elaborate on how you're still working on it. Make sure to make it clear that it's more of a challenge you're overcoming instead of a flat out weakness.
8. **Why did you leave your last job?**
    a. Be honest and don't trash talk your previous boss or employer. It can be any legitimate reason, such as you're not growing anymore in that role, you moved, or you found different interests in career.
9. **What is your salary expectation?**
    a. The way to approach this is to ask them what their budget is. Every new role in a company has an allocated budget range that they can afford to hire. You can also give you a range based on the market value for that position from prior research. Don't give a single number, but a RANGE.
10. **Where do you see yourself in 5 years?**

a. Talk more about your personal and professional career growth, goals, and ambitions. This is your chance to let them know about your sincere interest in the company and why you chose to work with them in the first place.

The Technical Interview

How to Prepare for the Technical Round

The first technical interview will be 2-3 parts (for a total of 30-45 minutes):

1 – Analytical (10-20 minutes)
1 – Technical (10-20 minutes)

Please be able to talk through how you would set up an experiment, how you would define success [which metrics would you look at, how would you use those metrics], and talk about the pros/cons of different experimental setups.

Analytical Portion (10-20 minutes)

The analytical case study will focus on questions to gauge your product sense.

For the analytical questions, the interviewer is trying to understand how you solve business questions and problems, as well as how creative and articulate you are at thinking through these problems while solving them. It's not about arriving at the perfect, or correct answer, but how you engage with the problem.

Spend some thinking about the company's product less as a user and more as someone who is tasked with improving or developing these products.

Put yourself in the shoes of the product team who built the product / feature:

- Why do you think they made certain decisions about how it works?
- What could be done to improve the product?
- What kind of metrics you would want to consider when solving for questions around health, growth, or the engagement of a product?
- How would you measure the success of different parts of the product?

- What metrics would you assess when trying to solve business problems related to our products?
- How would you tell if a product is performing well or not?
- How would you set up an experiment to evaluate any new products or improvements?

Technical Portion (10-20 minutes)

You will be given 1-2 data processing questions during this portion. This will be a hands-on technical investigation of data problems. We are looking not only for coding skills, but also for the ability to design an operational approach to figure out a concrete answer to a specific question using data.

Sample Question:

Given timestamps of logins, figure out how many people on Facebook were active all seven days of the week on a mobile phone.

How do you determine what product in Facebook was used most by the non-employee users for the last quarter?

While the actual data presented in the technical question will vary based on the question being asked, you can expect to see event-level data, dimension-level data, or both. This data set is designed for interview purposes and is not representative of the large data sets we work with at FB.

Mock Data And Questions

Here's a couple of examples that don't pertain specifically to Facebook but are representative of the kind of data and format with which you'll be presented:

Event-level data: an attendance log for every student in a school district

date | student_id | attendance

Dimension-level data: a summary table with demographics for each student in the district

student_id | school_id | grade_level | date_of_birth | hometown

Using this data, you can answer questions like the following:

- What was the overall attendance rate for the school district yesterday?
- Which grade level currently has the most students in this school district?
- Which school had the highest attendance rate? The lowest?
- You will be expected to write code that would answer the data processing questions given based on a schema or set of schemas that will be provided to you. Whether you choose SQL, Python, or R, please follow standard coding style and best practices for easy readability.

If using SQL, you can expect to be assessed on some subset of the following:

- Write a query or set of queries to derive insights based on the given log(s) or schema(s)
- Work with aggregate functions
- Utilize different types of Joins (IE: Left, Inner, Outer, etc.)
- Utilize Union and Union All.
- Work with concepts including Distinct, Random Sampling, De-Duplication, Optimization.
- Apply the results of your analysis to make product decisions or suggestions.

If using Python:

- Always check for edge cases and corner cases when you are coding and try to provide a bug free solution.
- Be prepared to improve your solution when asked.
- Make sure to practice the core CS concepts like arrays, linked list, stacks, queues, hash maps, binary trees and graphs, searching and sorting, recursion and parity.

If using R:

- Be thoughtful about implementation efficiency.
- Be familiar with apply functions family, dplyr package or an equivalent.
- While statistical packages can be called directly in R, please also expect some general programming that needs to be implemented, such as data frame manipulation and control flow.

The On-Site Interviews

How to Prepare for the On-Site Interviews

The on-site interviews are a bit more intensive and requires you to meet up with the company's team in person. It is normally the final part of the whole interview process and they use this time to screen out their last bit of candidates. You want to make sure that you leave a good impression and are prepared for everything they put on you.

It's normally broken down like this:

1 – Behavioral (10-20 minutes)
1 – Technical Code (30-60 minutes)
1 – Technical Whiteboard (30-60 minutes)
1 – Technical Stats/Prob Questions (30-60 minutes)
1 – Business Use Case (30-45 minutes)
1 – [optional] Project Presentation (30-60 minutes)

You may not get asked all of these questions I've listed here because it really depends on the company. But at a high level, you can expect a couple or even all of those questions.

These on-site interviews may take up to a couple of hours and essentially they want to know if you're a good fit for the company.

Resource for Technical Code / Whiteboard Questions: *HERE*
Resource for Stats/Prob Questions: *HERE*
Resource for Business Use Case Questions: HERE

Interview Tips

Last Minute Advice

1. During the video conference interview, you will need access to a computer. You will log in to www.coderpad.com - basically like a chat room, so the person on the other side can look at your code. (You will receive additional details about this when the interview is confirmed).
2. I strongly encourage you to have a full understanding of being able to analyze data sets through a scripting language of your choice (SQL, R, or Python). If you do NOT know the basics of one of these languages, you should not be moving forward with scheduling an interview.
3. Normally you can answer the questions in the language that you feel most comfortable with, although choosing a language that is going to assist in getting to an optimal solution in the most speedy and efficient manner is key.
4. It's important that you think out loud/ provide a narrative as you go through the problem so the interviewer has insight into your thought process.
5. The interviewer will prompt you if you are heading in the wrong direction. Your ability to pivot your answer is a positive signal.
6. Feel free to ask clarifying questions during the interview as well.

What Hiring Managers Want to Know About You

This is what hiring managers are looking for in all of their candidates. Please do your due diligence to research about their company and role so you can have a higher rate of success. They are looking for candidates that exhibit passion, curiosity, and creativity. **At the end of the day, it's either they like you OR don't.**

This is what they are looking for:

1. *Are you prepared for the interview and can answer the questions without complete struggle*
2. *Are you able to demonstrate and showcase your experience well*
3. *How knowledgeable are you about the role you've applied for*
4. *How knowledgeable are you about the company*
5. *Do you know the company's core values and culture*

6. *Are you realistic about the salary and compensations*
7. *Are you honest and transparent*
8. *Are you professional and likeable*

Questions You Should Ask Your Interviewer

Remember, the interview isn't just about you. You're also giving an interview to them as well. It's your responsibility to ask any questions to make sure this role is the right fit for you. This is a win-win situation and both of you are trying to get to know each other.

Here are some questions that you should definitely ask to make sure that they are the right fit and it also shows how serious you are about the company:

1. *Who does the role report to?*
2. *If I were to get hired right now, what are the things that I would need to do to be a 'success' in this role?*
3. *How long has this job been vacant?*
4. *What main project(s) is/are your team currently working on?*
5. *How long do you expect the interview process and decision-making to take?*
6. *How many other candidates are you presenting for this role?*
7. *How is the work culture like here?*
8. *Is the company also considering internal candidates?*
1. *You said the salary is 'competitive'. Can you define what the numeric range is for the position?*
2. *Is somebody currently still in this role?*

Resources

Datasets

I highly advise you to download a dataset from one of these sources and start doing a person project of your own. Get the data, clean it, do exploratory data analysis, apply some models, and interpret your results.

- Kaggle - Kaggle has come up with a platform, where people can donate datasets and other community members can vote and run Kernel / scripts on them.
- World Bank – The open data from the World bank. The platform provides several tools like Open Data Catalog, world development indices, education indices etc.
- Five Thirty Eight Datasets – Here is a link to datasets used by Five Thirty Eight in their stories. Each dataset includes the data, a dictionary explaining the data and the link to the story carried out by Five Thirty Eight.
- Amazon Web Services (AWS) datasets –Amazon provides a few big datasets, which can be used on their platform or on your local computers.
- Google datasets – Google provides a few datasets as part of its Big Query tool. This includes baby names, data from GitHub public repositories, all stories & comments from Hacker News etc.
- data.gov – This is the home of the U.S. Government's open data. The site contains more than 190,000 data points at time of publishing. These datasets vary from data about climate, education, energy, Finance and many more areas.
- data.gov.in – This is the home of the Indian Government's open data. Find data by various industries, climate, health care etc.
- https://data.gov.uk/ - Find data published by the United Kingdom central government, local authorities and public bodies to help you build products and services
- https://data.europa.eu/euodp/en/home - The European Union Open Data Portal (EU ODP) gives you access to open data published by EU institutions and bodies.

Facebook Groups

One of the best ways to learn data science is being involved with the community. You'll be surrounded by like minded people and learn so much from others in these groups. Take the initiative to start a conversion, ask questions, and share your knowledge with the community. Here are my favorites:

- Beginning Data Science, Analytics, Machine Learning, Data Mining, R, Python
- Data Science With R
- Data Science and Predictive Analytics News
- Data Science, Deep Learning, and Machine Learning with Python
- Data Science With Python
- Data Science World
- Data Science Community

Blogs

Here are some of my favorite blogs you can follow and learn from.

- kdnuggets.com
- datasciencecentral.com
- blog.kaggle.com
- dataquest.io-bd888fd8ff
- r-bloggers.com
- simplystatistics.org
- andrewgelman.com
- blog.echen.me
- Medium.com-e88f044ea0 (Airbnb blog)
- Data.quora.com

Job Portals

Here are my recommended job portals that has good call-back rates:

- Glassdoor.com
- LinkedIn.com/jobs
- ZipRecruiter.com
- Monster.com
- Indeed.com
- AngelList.com (good for startups)
- UpWork.com (good for freelance)

Recommended Courses

DataCamp

I actually used DataCamp when I first started learning data science and I can honestly say that it has grown quite a bit since then. The community there is great, the UI is superb, and the way their courses are structured are super friendly. I'd recommend DataCamp for super beginners in the field of data science, those who are completely new and have no idea on how to approach these topics. This is because DataCamp was meant to guide you and provide you a high level overview of how you can understand these data science concepts and tools. Don't use DataCamp and expect to be a master of your craft. Use it as a guide to get your hands dirty and have a sense of what you can do with these tools/techniques. After that I'd recommend taking more in-depth courses via Udemy or Coursera to enhance your data science chops.

Summary: Use DataCamp if you're a beginner and get comfortable understanding the high-level overview of the tools/techniques. Don't use it for expert level understanding.

Visit Website - HERE

Udemy

I started using Udemy half way during my journey in data science and it definitely enhanced my learning experience. The best thing about it is that there's so much content on that site and at a cheap price. I'd recommend going to Udemy once you have some clarity on how data science and machine learning is used. It's a game changer for those that are already familiar with some concepts of data science and machine learning. On the other hand, since Udemy has so many courses that doesn't mean all of them are good. There's only a few courses that I recommend and you should take

Summary: Use Udemy if you're already familiar with data science concepts and are not a total beginner. The courses here should definitely push you towards an intermediate level in data science.

Visit Website - HERE

Coursera

Coursera is known for their specialization courses and is completely jam packed with quality content. The downside of Coursera compared to Udemy and DataCamp is that some of these courses require a lot more hours to complete. I'd recommend this site to anybody who has the time and wants to learn a concept from the ground up. Completing the courses here should get you to an intermediate or advanced level of a concept. One thing I really like about Coursera is that it is free. If you decide to complete the certifications then you do have to pay for it.

Summary: Use Coursera if you have the time to complete the course. It's worth the investment and it should push you towards an intermediate or advanced level in data science. It's also a great way to include certifications on your resume and LinkedIn profile.

Visit Website - HERE