Data Science Notes PDF

Data science notes summarise the skills used to turn raw data into decisions: Python and SQL for data access, statistics for reasoning, cleaning and visualisation for understanding, machine learning for prediction, and evaluation and communication for trust. Use them as a quick revision sheet before exams, projects or interviews.

Key takeaways

- The workflow runs from question to data, cleaning, exploration, modelling, evaluation and communication.
- Most project time goes into data preparation, not model training.
- Always split data into training and test sets before judging a model.
- Choose metrics that match the business cost of errors.
- A simple model you can explain often beats a complex one nobody trusts.

What is data science revision notes?

Data science is the practice of using programming, statistics and domain knowledge to collect, clean, analyse and model data so that organisations can answer questions, predict outcomes and support decisions.

What does a data science workflow look like?

A project usually follows seven steps. **Define the question** in a sentence with a measurable outcome. **Collect data** from databases, files or APIs. **Clean** it by fixing types, missing values and duplicates. **Explore** with summaries and charts. **Model** with a suitable algorithm. **Evaluate** on data the model has not seen. **Communicate** the result and its limits in plain language. Notes should follow this order because it is also the order of most interviews.

Which tools should the notes cover?

Python is the common language, with **pandas** for tables, **NumPy** for arrays, **Matplotlib or Seaborn** for charts and **scikit-learn** for classic machine learning. SQL is needed to pull data. Jupyter notebooks help you explore, while scripts and Git help you repeat and share work. Keep one line per tool saying what it is for and one example you have actually run.

What statistics must I revise?

Revise mean, median, mode, variance, standard deviation, quartiles, correlation, probability, the normal distribution, sampling, confidence intervals and hypothesis tests. For each, write what it

measures and a case where it misleads. For example, the mean describes a skewed income column poorly, and correlation does not show cause.

How do I handle messy data?

List your cleaning checks as a routine:

1. Check shape, types and a sample of rows.
2. Count missing values per column and decide to drop, fill or flag.
3. Remove exact duplicates and check key columns for repeats.
4. Fix inconsistent labels, such as "Delhi" and "delhi".
5. Look for impossible values, such as a negative age.
6. Document each change so the work can be repeated.

How are machine learning problems grouped?

Problem type	Goal	Example algorithms
Regression	Predict a number	Linear regression, random forest
Classification	Predict a class	Logistic regression, decision tree
Clustering	Find groups without labels	K-means
Dimensionality reduction	Compress features	PCA
Write one sentence on when each is the right choice and one on how it fails.		

How do I judge a model fairly?

Split data into **train and test** sets, or use cross-validation. For classification, read accuracy together with precision, recall and the confusion matrix, since accuracy hides problems when one class is rare. For regression, use MAE or RMSE. Watch for **overfitting**, where training scores are high and test scores are low, and for **data leakage**, where information from the future slips into the features.

What small code patterns are worth memorising?

```
import statistics as st
marks = [62, 71, 55, 90, 71, 68]
print(round(st.mean(marks), 2), st.median(marks), st.mode(marks))
```

```
train_test_split -> fit on train -> predict on test -> score
```

Know these shapes well enough to write them without searching.

How do I explain results to non-technical people?

Lead with the answer, show one clear chart, state the main limit and finish with a recommended action. Avoid jargon unless you define it. Practise saying your project in one minute.

How should I organise revision across a week?

Give each day one theme so nothing is skipped. Day one covers Python and pandas, day two statistics and probability, day three cleaning and visualisation, day four supervised learning, day five unsupervised learning and evaluation, day six one small coding exercise from start to finish, and day seven a rest plus a quick self-test. At the end of each day write three lines: what you learned, what confused you and what you will try next. Review the confusing items first on the following day, because spaced repetition of weak points gives the best return on limited time.

What do examiners and interviewers usually ask from these notes?

Expect definitions with an example, such as "explain bias and variance", comparisons such as "classification versus regression", and process questions such as "how do you handle missing data?". A good habit is to answer in three parts: the idea in one sentence, a small example from your own work and one limitation. This structure keeps answers short, clear and honest, and it works for oral and written tests alike.

Which mistakes should these notes warn about?

Four warnings deserve a place in the notes. First, never tune a model on the test set, because the final score then stops being an honest estimate. Second, do not drop missing values blindly, since the absence of a value can itself carry meaning. Third, do not read a high correlation as proof of cause. Fourth, do not chase a tiny score gain at the cost of a model nobody can explain or maintain.

Frequently asked questions

What should data science notes include?

They should cover the workflow, Python and SQL basics, statistics, data cleaning, machine learning types, evaluation metrics and how to communicate results clearly.

How long does it take to revise data science?

A focused revision of the core topics takes a few weeks of regular study, but the time depends on your background and goals.

Which language is used most in data science?

Python is the most common language, supported by libraries such as pandas, NumPy and scikit-learn, with SQL for databases.

Do I need deep maths for data science?

You need solid basics in statistics, probability and some linear algebra, but you can learn them step by step alongside practice.

Will these notes get me a job?

No. Notes support learning, but hiring depends on skills shown through projects, interviews and the job market at the time.

Sources and further reading

- Python documentation
- pandas documentation
- scikit-learn user guide

About this PDF

These PDFs are study material for learning and revision. They are not a certificate, a degree or a promise of a job, and technology changes, so check official documentation for the latest details. Spot an error? Write to info@uncodemy.com and the trainer team will review it.

More free resources and courses: uncodemy.com | info@uncodemy.com | +91 98183 66550