**Capstone planning**

You’ve got an idea for a project, that’s great! However, you’re not sure where to get started. This is typically the most difficult and daunting part of the entire project, but don’t worry, we’ve got you covered. Below we have taken a checklist inspired from the [Hands-On Machine Learning with Scikit-Learn & TensorFlow book by Aurélien Geron](https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/). This checklist will guide you through a data science project.

Here are the main steps:

1. Frame the problem and look at the big picture.

2. Get the data.

3. Explore the data to gain insights.

4. Prepare the data.

5. Data story telling

6. Explore different models and short-list the best ones.

7. Fine-tune your models.

8. Present your solution.

**Note**

The most important thing you must do at the start of any project is to plan. **You must resist the urge to jump straight into code**. Each section of the checklist will ask you questions about your project- these questions are designed in a way to help you six-fold:

* Concrete ideas by identifying areas that need developing
* Breaks down project into manageable chunks of work
* Systematic order of this plan reduces the chance of getting lost
* Reduces ambiguity especially if working in a team
* Gives supervisors an overview on the project
* Documents decisions made during a project- this will make it very easy to build your presentation and a blog

Before we go any further, it’s imperative you treat this as a living and breathing document. Things change, new things come to light, assumptions are wrong- this is perfectly normal and expected. You must document your findings and thoughts throughout the project- always keep this checklist open when working.

**Frame the Problem and Look at the Big Picture**

1. Define the objective of the project.

* What’s the background?
* What are you trying to do/solve/show?
* Why are you trying to do this?

2. How will your solution be used?

* Will you be telling a story with data?
* Will you have user interaction?
* Will there be a demo?

3. What are the current solutions/workarounds (if any)?

* Has this problem existed before?
* Has it already been solved?
* Add resources to this section of similar projects
* What inspiration can you gather from these solutions?

4. How should you frame this problem?

* Supervised?
* Unsupervised?
* Data storytelling?

5. How should performance be measured?

* Will you use metrics?
* Are there any existing benchmarks
* What does success look like?

6. Is the performance measure aligned with the objective?

* Why did you choose these metrics/success criteria?

7. What would be the minimum performance needed to reach the objective?

* What’s the most basic version of your project look like?
* Is there minimum metric score you want to reach?
* Is there a definitive question that needs answering?

8. Is human expertise available?

* Do you have domain expertise?
* Do you know anyone who does?

9. What are the rough steps to approach this project?

* List out basic components you will need to gather or build

10. List the assumptions you (or others) have made so far.

* Ensure assumptions are verified where possible

11. Is everyone on the same page with definitions and objectives?

**Get the Data**

1. List the data you need

* What data is needed to fairly represent the problem you are facing?

2. Find and document where you can get that data.

* List the resources required to gather the data
* Check section 3 of Framing a Problem, what sources did others use?

3. Check how much space it will take.

* Can your computing resource cope with the amount of data needed?

4. Check legal obligations and get authorization if necessary.

* Are there any term and conditions you must adhere by? (Kaggle etc)

5. Get access authorizations.

* Does access to data require sign up and authentication?

6. Get the data.

* Download data directly
* Build functions to access API’s

7. Convert the data to a format you can easily manipulate.

* Does your data need concatenated?
* Does it need to be in a tabular format?

8. Ensure sensitive information is deleted or protected (e.g., anonymized).

9. If using machine learning- sample a test set, put it aside and only use it at the end of the project.

* What size does your test need to be to ensure reliable results?
* Does the test set fairly represent the real-world application?

**Explore the Data**

1. Create a copy of the data for exploration (sampling it down to a manageable size if necessary).

2. Create a notebook to keep a record of your data exploration.

3. Study each attribute and its characteristics:

* Name Type (categorical, int/float, bounded/unbounded, text, structured, etc.)
* Percentage of missing values
* Noisiness and type of noise (stochastic, outliers, rounding errors, etc.)
* Type of distribution (Gaussian, uniform, logarithmic, etc.)

4. For supervised learning tasks, identify the target attribute(s).

5. Visualize the data.

6. Study the correlations between attributes.

7. Identify the promising transformations you may want to apply.

8. Document what you have learned

**Prepare the Data**

1. Data cleaning:

* Drop their rows (or columns).
* Replace in missing values (e.g., with zero, mean, median)
* Fix or remove outliers (optional).

2. Feature scaling: standardize or normalize features.

3. Feature selection: Drop the attributes that provide no useful information for the task.

4. Feature engineering, where appropriate:

* Discretize continuous features. Decompose features (e.g., categorical, date/time, etc.).
* Add promising transformations of features (e.g., log(x), sqrt(x), x^2, etc.).
* Aggregate features into promising new features.

**Data storytelling (Data Analysis)**

A study found 63% of audiences can remember a story, but only 5% could remember a statistic. Don’t throw numbers at people, it’ll go in one ear and out the other.

1. Design a story arc

* What are the main points you want to get across
* Give context to each point you make

1. Drilling down when necessary

* If you find something interesting, spend time finding out the reasons behind it.

1. Highlight areas of interest

* One focal point per graph
* Use colour rather than shapes to emphasise points ([colourblind friendly colours](https://davidmathlogic.com/colorblind/#%23D81B60-%231E88E5-%23FFC107-%23004D40))
* Use appropriate plots and annotations

**Short-List Promising Models** (Machine learning only)

Notes: If the data is huge, you may want to sample smaller training sets so you can train many different models in a reasonable time. Once again, try to automate these steps as much as possible.

1. Train many quick and dirty models from different categories (e.g., linear, naive Bayes, SVM, Random Forests, neural net, etc.) using standard parameters.

2. Measure and compare their performance. For each model, use N-fold cross-validation and compute the mean and standard deviation of the performance measure on the N folds.

3. Analyse the most significant variables for each algorithm.

4. Analyse the types of errors the models make.

* What data would a human have used to avoid these errors?

5. Have a quick round of feature selection and engineering.

6. Have one or two more quick iterations of the five previous steps.

7. Short-list the top three to five most promising models

**Fine-Tune the System** (Machine learning only)

Notes: You will want to use as much data as possible for this step, especially as you move toward the end of fine-tuning. As always automate what you can.

1. Fine-tune the hyperparameters using cross-validation.

* Treat your data transformation choices as hyperparameters, especially when you are not sure about them (e.g., should I replace missing values with zero or with the median value? Or just drop the rows?).
* Unless there are very few hyperparameter values to explore, prefer random search over grid search.

2. Try Ensemble methods. Combining your best models will often perform better than running them individually.

3. Once you are confident about your final model, measure its performance on the test set to estimate the generalization error.

**Present Your Solution**

1. Document what you have done.

* Go through this document again in more detail

2. Build your narrative:

Glue the following points together with a narrative:

* Situation (Frame the Problem and Look at the Big Picture)
* Task (Frame the Problem and Look at the Big Picture)
* Actions (Get, explore, prepare, model data/storytelling)
* Result (demo, comparing against benchmarks, interesting findings)

3. Support your narrative with

* Visualisations e.g., Graphs relating to results/important features/storytelling points
* Explanations why decisions/steps were taken in relation to the overall objective

4. Add future proofing:

* Identify limitations to your project
* How would you improve your project if you had more time

5. Summary

* Summarise the objective, solution and results/key findings in 2 lines

**Presentation tips**

* You have people's attention for the first 10 seconds, anything after that is a gift- Make sure you get the main points across as soon as possible
* Be concise, when you waffle you lose peoples interest
* Don’t cram information/words on slides, be greedy with the number of slides you use
* Record yourself presenting and watch it back- analyse how much you use filler words, where points are unclear, where you forget your lines.
* Take a breath and talk slowly