Machine Learning Project Checklist

A modest checklist to guide your **Machine Learning (ML)** projects.

Adapted from the book: "Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow", A. Géron (2019).

1 Frame the Problem

Challenge

1. Provide the context of the problem.
2. Define the objective in business terms.

Define baselines

1. How is the **problem** currently resolved and what are their **gains** and **losses**?

Solution Planning

1. What kind of ML algorithms to use (**supervised/unsupervised**, **online/offline**, etc)?
2. How should **performance** be measured?
3. What would be the **minimum performance** needed to reach the business objective?
4. What are the available data sources?
5. List the assumptions for the problem.
6. How will the **project deliverable** be?
   - Exploratory Data Analysis, ML model, ...

2 Get Data

1. List the data you need (and how much).
2. Check **legal obligations**, and get authorization if necessary.
3. Create a workspace with enough disk space.
4. **Get the data.**
5. Take a quick look at the **data structure**.
6. Convert the data to a format you can easily work (without changing the data itself).
7. Ensure **sensitive information is deleted** or protected (e.g., anonymized).
8. **Sample a test set**, put it aside, and **never look at it** (no data snooping!).
   - There are situations in which the data should be cleared first (*e.g.*, drop instances with missing values).

3 Data Cleaning

Note: You may prefer to reverse steps 3 and 4.

1. Convert types accordingly (*e.g.*, string to datetime, string to numeric, ...)
2. **Fix** or **remove** outliers and duplicated instances (optional).
3. **Fill in** missing values (with zero, median, ...) or **drop their** instances or attributes.

4 Explore the Data

Notes:

- Try to get insights from a **field expert** first.
- You may need **to** clean the data again during this step.

1. Create a **copy of the data** for exploration (sampling it down if necessary).
2. Study each attribute and its characteristics:
   - Name
   - Type (categorical, int/float, (un)bounded, text, structured, etc)
   - Noisiness and type of noise
   - Type of distribution (uniform, log., etc.)
3. For supervised learning tasks, identify the **target attribute(s)**.
4. **Formulate and validate** business hypotheses.
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.

5 Prepare the Data (Preprocessing)

Notes:

- Work on **copies** of the data.
- You may need **to clean the data** again.

1. **Feature selection** (optional)
2. **Feature engineering**, where appropriate:
   - **Discretize** continuous features.
   - **Encode categorical features.**
   - Add promising **transformations of features** (*e.g.*, log(x), sqrt(x), x^2, etc.).
   - **Aggregate features** into promising new features
3. **Feature scaling:**
   - Standardize or normalize features.

6 Train ML Algorithms

Note: If the **data is huge**, you may want to **sample smaller training sets** so you can train many different models in a **reasonable time** (be aware that this may **penalize** some models).

Shortlist Promising Models

1. Train many quick-and-dirty models from different categories.
2. Measure and compare their performance on the **training set**:
   - For each model, use N-fold cross-validation and compute the mean and standard deviation of the N folds.
3. Analyze the **most significant variables** for each algorithm.
4. Analyze the **types of errors** the models make.
5. Perform **one or two more quick iterations** of the previous steps.
6. Shortlist the **top three to five most promising models**, preferring models that make different types of errors.

Fine-Tune the System

1. **Fine-tune** the **hyperparameters** using **cross-validation:**
   - Treat your **data transformation** choices as **hyperparameters** when you are not sure about them.
   - Use **grid search** only if there are **very few hyperparameter values** to explore, otherwise **prefer random search**.
   - If training is very long, you may prefer a **Bayesian optimization approach**.
2. Try **Ensemble methods**. Combining your **best models** will often report **better performance** than running them individually.

7 Evaluation

Technical Evaluation

1. Measure the **performance** of the **selected models** on the **test set** to estimate the **generalization error**.
- Compare your **results** with the baselines.

Evaluation in Business Terms

1. What does your **(technical) performance** reflect in business terms?
   - How much will the **system's performance financially impact** the business?
   - Will your solution **save money and/or time**?
2. **How beneficial** is your solution (in business terms) compared to the baselines?

8 Present your Solution

1. Highlight the **big picture** first.
2. Explain why **your solution** achieves the **business objective**.
3. Present **interesting points** you noticed:
   - Considered **hypotheses and top 5 insights**.
   - Describe what worked and what did not.
   - **Assumptions** and your **system's limitations**.
   - **Technical** and **business results**.
   - Main **learned lessons** and next steps.

9 Launch

1. Get your solution ready for production.
   - Plug into production data, write tests, ...
2. **Deploy** your **solution**.
3. Write **monitoring code** to check your **system's live performance** at regular intervals and **trigger alerts** when it drops.
4. **Retrain your models** on a regular basis on **fresh data** (automate as much as possible).

Created by

Prof. Samuel Botter Martins

samucoding.com

@samucoding

@iamsamucoding

youtube.com/@xavecoding

samucoding