Machine Learning Project Checklist

Adapted from Appendix A of the book "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron.

The main steps

1. Get the data.
2. Frame the problem and look at the big picture.
3. Explore the data to gain insights.
4. Prepare the data to better expose the underlying data patterns to machine learning algorithms.
5. Explore many different models and shortlist the best ones.
6. Fine-tune your models and select the best one. If needed, combine multiple models. **(You can well ignore the latter point for the projects for this lecture)**
7. Present your solution.

⚠ This is a guideline. You don't have to follow it slavishly! Feel free to adapt this checklist to your needs. You don't need to follow all the steps of this checklist to get a decent grade. It is more to be seen as a list of what you COULD do.

Get the Data

1. Get (download) the data.
2. Convert the data to a format you can easily manipulate (without changing the data itself), e.g. a Pandas DataFrame.
3. Sample a test set, put it aside, and never look at it (no data snooping!)

Frame the Problem and Look at the Big Picture

1. Define the objective.
2. How should you frame this problem (supervised/unsupervised, regression/classification, etc)
3. How should performance be measured?
4. What would be the minimum performance needed to reach the objective?
5. How would you solve the problem manually?

Explore the Data

1. Create a copy of the data for exploration (sampling it down to a manageable size if necessary).
2. Recommended: Create a Jupyter notebook to keep a record of your data exploration.
3. Study each attribute and its characteristics:
    - Name
    - Type (categorical, int/float, bounded/unbounded, text, etc.)
    - % of missing values
    - Noisiness and type of noise (e.g. stochastic, outliers, rounding errors, etc.)
    - Usefulness for the task
    - Type of distribution (Gaussian, uniform, logarithmic, etc.) **[Feel free to omit this for the projects for this lecture]**
4. For supervised learning tasks, identify the target attribute(s).
5. Visualize the data.

6. Study the correlations between attributes and in particular between the "feature attributes" and the target attribute.
7. Study how you would solve the problem manually.
8. Identify promising transformations (feature engineering) you may want to apply.
9. Document what you have learned

Prepare the Data

1. Clean the data:
    - Fix or remove outliers (optional).
    - Fill in missing values (e.g., with zero, mean, median...) or drop their rows (or columns).
2. Perform feature selection (optional): Drop the attributes that provide no useful information for the task.
3. Perform feature engineering, **where appropriate**. For instance:
    - 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.
4. Perform feature scaling: Standardize or normalize features.

Shortlist Promising Models

1. Train many quick-and-dirty models from different categories (e.g., linear, naive Bayes, kNN, SVM, random forest, 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. Analyze the most significant variables for each algorithm (e.g., the features with the largest coefficients in a linear regression model).
4. Analyze the types of errors the models make: What data would a human have used to avoid these errors?
5. Perform a quick round of feature selection and engineering.
6. Perform one or two more quick iterations of the five previous steps.
7. Shortlist the top three to five most promising models, preferring models that make different types of errors.

Note: If the dataset 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 penalizes complex models such as large neural nets or random forests).

Fine-Tune the System

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., if you're not sure whether to replace missing values with zeros or with the median value, or to 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 produce better performance than running them individually. **You can ignore this step for the projects in this lecture.**
3. Once you are confident about your final model, measure its performance on the test set to estimate the

generalization error.

⚠ Don’t tweak your model after measuring the generalization error: you would just start overfitting the test set.

Present Your Solution

1. Document what you have done.
2. Create a nice presentation:
    - Make sure you highlight the big picture first.
3. Explain why (or why not) your solution achieves the objective.
4. Don’t forget to present interesting points you noticed along the way:
    - Describe what worked and what did not.
5. Ensure your key findings are communicated through beautiful visualizations or easy-to-remember statements (e.g., “the median income is the number-one predictor of housing prices”).