How to Approach a Machine Learning Project

Let's explore a step-by-step process for approaching ML problems in Real life:

1. Understanding the Business Requirements and the Nature of the Available Data.
2. Classify the problem as Supervised/Unsupervised as well as Regression/Classification (in advance).
3. Download, Clean & Explore data and Create New Features (if required) that may improve models.
4. Create Training/Validation/Test sets of data and prepare the data for training ML models.
5. Create a quick & easy baseline model to evaluate and benchmark future models.
6. Select a modeling strategy, train a model, and tune hyperparameters to achieve optimal fit.
7. Experiment and combine results from multiple strategies to get a better result.
8. Interpret models, study individual predictions, and present your findings to the stakeholders.

Step 1 - Understand Business Requirements & Nature of Data

Most machine learning models are trained to serve a real-world use case. It's important to understand the business requirements, modeling objectives and the nature of the data available before you start building a machine learning model.

Understanding the Big Picture

The first step in any machine learning problem is to read the given documentation, talk to various stakeholders and identify the following:

1. What business problem are you trying to solve with machine learning?
2. Why do we want to solve this problem, and how will it benefit the business?
3. How is this problem currently addressed without using machine learning tools?
4. Can we see some sample rows from the dataset? Do they accurately represent the entire dataset?
5. Where is the data stored, and how will you access it?
6. How much historical data do we have, and how was it collected?
7. What information/features does the historical data include? Does it have the past values of the feature for what we're trying to predict?
8. Are there any known issues with the data, like mistakes in data entry, missing information, or differences in units?
9. Who will use the results of the model, and how does it fit into other business processes?
10. Are there any privacy or security considerations with the data, and how will they be addressed?
11. ...

Gather as much information about the problem as possible, so that you're clear understanding of the objective and feasibility of the project.

Step 2 - Classify the problem as supervised/unsupervised & regression/classification

If you're new to Data Science, make sure to visit (this) to upgrade your skills.

And if you're already an expert, make sure not to overlook enhancing your skills by checking out this (Machine Learning in 10 minutes Blog)

Loss Functions and Evaluation Metrics

Once you've figured out the type of problem you're solving, it's crucial to choose an appropriate evaluation metric. Also, depending on the kind of model you train, your model will also use a loss/cost function to optimize during the training process.

- **Evaluation metrics** - used by industry experts to evaluate the machine learning model
- **Loss functions** - used by computers to fine-tune/optimize the ML model during the training process

They are often the same (e.g. RMSE for regression problems), but they can be different for instance, Classification problems could use Cross Entropy for the loss function and Accuracy for the evaluation metric.

Check out this article for a comprehensive overview of common loss functions and evaluation metrics: (11 Evaluation Metrics Data Scientists Should Be Familiar with Blog)

Step 3 - Download, clean & explore the data and create new features

Downloading Data

There may be different sources to get the data:

- CSV files
- SQL databases
- Raw File URLs
- Kaggle datasets
- Google Drive
- Dropbox
- etc.

Identify the right tool/library to get the data.

Exploratory Data Analysis and Visualization

Objectives of exploratory data analysis:

- Study the patterns/distributions/spread of individual columns (uniform, normal, exponential)
- Identify any unusual or incorrect data points(e.g. missing/incorrect values)
- Explore how the target column relates to other columns, (linear, non-linear etc.)
- Gather valuable insights about both the problem you're solving and the dataset itself
- Basically, come up with ideas for Preprocessing and Feature Engineering of data

Feature Engineering

Feature engineer is the process of creating new features (columns) by transforming/combining existing features or by incorporating data from external sources.

For example, here are some features that can be extracted from the "Date" column:

1. Day of week
2. Day or month
3. Month
4. Year
5. Weekend/Weekday
6. Month/Quarter End

Using date information, we can also create new current columns like:

1. Weather on each day
2. Whether the date was a public holiday
3. Whether the store was running a promotion on that day
4. Social media trends or online activities on specific dates
5. Seasonal factors influencing sales or activities

These additional columns can provide a more comprehensive understanding of the context surrounding the date information.

Step 4 - Create a training/validation/test split and prepare the data for training

Train/Test/Validation Split

In data analysis, we split our dataset into three parts for effective model training and evaluation. The training set, typically 70-75% of the data, train the model. The test set (15-20%) checks how well the model generalizes to new, unseen data. Validation set (10-15%) helps fine-tune the model's parameters without influencing the test set, ensuring unbiased evaluation.

This three-way split ensures our model is both trained well and performs reliably on real-world scenarios.

Input and Target columns

Also identify input and target columns. We needn't use all the available columns, we can start out with just a small subset.

Note: We can't use the no. of customers as an input, because this information isn't available beforehand.

Imputation, Scaling and Encode

In data preparation, we often encounter missing values, differing scales, and categorical variables.

- Imputation is the process of filling in missing data, ensuring a complete dataset.
- Scaling involves adjusting numerical values to a common scale, preventing one variable from dominating others.
- Encoding tackles categorical data, transforming it into numerical format for machine learning algorithms to comprehend

algorithms to comprehend.

Note: We can apply a different imputation strategy to different columns depending on their distributions (e.g. mean for normally distribute and median for exponentially distributed).

And also encode categorical columns as one-hot vectors.

Explore the (scikit-learn preprocessing module))

Step 5 - Create quick & easy baseline models to benchmark future models

A quick baseline model helps establish the minimum score any ML model you train should achieve.

Fixed/Random Guess

Let's define a model that always returns the mean value of Sales as the prediction.

def return_mean(inputs): return np.full(len(inputs), merged_df.Sales.mean())

Let's evaluate this to using the RMSE score.

from sklearn.metrics import mean_squared_error

Baseline ML model

Let's train a simple `LinearRegression` model, with no customization.

`model.fit` uses the following workflow for training the model (From Basic Machine Learning to Deep Learning in 5 Minutes):

1. We initialize a model with random parameters (weights & biases).
2. We pass some inputs into the model to obtain predictions.
3. We compare the model's predictions with the actual targets using the loss function.
4. We use an optimization technique (like least squares, gradient descent etc.) to reduce the loss by adjusting the weights & biases of the model
5. We repeat steps 1 to 4 till the predictions from the model are good enough.

Note: A simple linear regression model isn't much better than our fixed baseline model which always predicts the mean.

Based on the above baselines, we now know that any model we train should have ideally have a RMSE score lower than $2800. This baseline can also be conveyed to other stakeholders to get a sense of whether the range of loss makes sense.

Step 6 - Pick a strategy, train a model & tune hyperparameters

Systematically Exploring Modeling Strategies

Scikit-learn offers the following cheatsheet to decide which model to pick.

Here's the general strategy to follow:

- Find out which models are applicable to the problem you're solving.
- Train a basic version for each type of model that's applicable
- Identify the modeling approaches that work well and tune their hypeparameters
- Use a worksheet to keep track of your experiments and results.

Let's define a function `try_model`, which takes a model, then performs training and evaluation.

Linear Models

Read about (linear models)

from sklearn.linear_model import LinearRegression, Ridge(L2 Regularization), Lasso(L1 Regularization), ElasticNet, SGDRegressor

Tree Based Models

- (Decision Trees)
- (Random Forests and Gradient Boosting)

Other Supervised Learning Models

Checkout (Spervised Learning Models)

Try some other supervised learning algorithms and see if you can get a better result.

Unsupervised Learning Techniques

Checkout (Spervised Learning Models)

Step 7 - Experiment and combine results from multiple strategies

In general, the following strategies can be used to improve the performance of a model:

- Gather more data. A greater amount of data can let you learn more relationships and generalize the model better.
- Include more features. The more relevant the features for predicting the target, the better the model gets.
- Tune the hyperparameters of the model. Increase the capacity of the model while ensuring that it doesn't overfit.
- Look at the specific examples where the model make incorrect or bad predictions and gather some insights
- Try strategies like grid search for hyperparameter optimization and K-fold cross validation
- Combine results from different types of models (ensembling), or train another model using their results.

Hyperparameter Optimization & Grid Search

You can hypertune parameters manually, our use an automated tuning strategy like random search or Grid search. Follow this tutorial for hyperparameter tuning using (Grid search)

K-Fold Cross Validation

- Apply K fold cross validation for the random forest regressor trained earlier.

Here's what (K-fold cross validation) looks like visually

Follow this tutorial to apply ([K-fold cross validation]
(https://machinelearningmastery.com/repeated-k-fold-cross-validation-with-python/))

Ensembling and Stacking

Ensembling refers to combining the results of multiple models. Here's what (ensembling) looks like visually

Stacking is a more advanced version of ensembling, where we train another model using the results