Attendance checking

An end-to-end ML project

Mauricio A. Álvarez

Foundations of Machine Learning
The University of Manchester

Overview

- This week's session will show an example of an end-to-end project in ML. The focus is on supervised learning.
- The lecture provides a general overview and the Jupyter Notebook will provide a hands-on exercise.
- We will describe the typical process involved in providing an ML solution.
- We will use *scikit-learn* (`https://scikit-learn.org/stable/`) to illustrate several of the steps involved.

Why *scikit-learn*?

- scikit-learn is a machine learning library for Python.

- It supports supervised and unsupervised learning.

- It provides several utilities for data preprocessing and model selection and evaluation.

- More importantly: it is open source (BSD licence).

Machine learning project checklist

1. Frame the problem and look at the big picture.
2. Get the data.
3. Explore the data to get insights.
4. Prepare the data to better expose the underlying data patterns.
5. Explore many different models and shortlist the best ones.
6. Fine-tune your models and combine them into a soluttion.
7. Present your solution.

Contents

Frame the problem and look at the big picture

Get the data

Explore the data

Prepare the data

Shortlist models and then fine-tune them

Present your solution

Set the problem in context (I)

- ❑ How will your solution be used?
- ❑ What are the current solutions to the problem (if any)?
- ❑ Is it a regression or a classification problem?
- ❑ How is performance measured?

Set the problem in context (II)

- What would be the minimum performance necessary?
- Are there comparable problems?
- Is human expertise available?

Model performance assessment

- In ML, we use several metrics to assess the performance of a model.

- Performance measures commonly used in regression are
  - root mean squared error (RMSE).
  - mean absolute error (MAE).

- For classification, metrics of performance include:
  - confusion matrix.
  - precision/recall.
  - accuracy.

Metrics in regression

- We have a predictive model $f(\mathbf{x}, \mathbf{w})$ and we want to assess its performance over a dataset of instances $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$.

- The root mean squared error is defined as

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} [y_i - f(\mathbf{x}_i, \mathbf{w})]^2}.$$

- The mean absolute error is defined as

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} \left| y_i - f(\mathbf{x}_i, \mathbf{w}) \right|.$$

Metrics in classification: confussion matrix

- We have a predictive model $f(\mathbf{x}, \mathbf{w})$ and we want to assess its performance over a dataset of instances $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$.

- Binary classification problem: spam detection. There are two classes, spam and not-spam.

- The confusion matrix is a table that summarizes how successful the classification model is at predicting examples belonging to various classes

	spam (predicted)	not-spam (predicted)
spam (actual)	23 (TP)	1 (FN)
not-spam (actual)	12 (FP)	556 (TN)

whereTP stands for **true positive**, TN stands for **true negative**, FP stands for **false positive** and TN stands for **true negative**.

Metrics in classification: precision/recall (I)

- **Precision** is the ratio of correct positive predictions to the overall number of positive predictions

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}.$$

- **Recall** is the ratio of correct positive predictions to the overall number of positive examples in the dataset

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}.$$

Metrics in classification: precision/recall (II)

- Say you look for the terms “pattern recognition and machine learning” in the University’s Library website.

- Precision is the fraction of relevant documents in the list of returned documents (“how valid the results are?”)

- Recall is the fraction of relevant documents returned to the total number of relevant documents that could have been returned (“how complete the results are?”)

- In the spam example, we would like high precision, this is we would like the FP to be low meaning having few genuine messages predicted as spam.

- We are probably Ok with a low recall, some spam messages sent to Inbox because they are predicted as not-spam.

Metrics in classification: accuracy

- **Accuracy** is the ratio of examples corrected classified over the total number of examples classified

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}.$$

- Accuracy is a useful metric when errors predicting all classes are equally important.

- In the spam example, FP are worse that FN.

- Accuracy can be misleading in imbalance classification problems, e.g. you can have a high a TP value and a low TN value, and your accuracy could still be high.

Contents

Frame the problem and look at the big picture

Get the data

Explore the data

Prepare the data

Shortlist models and then fine-tune them

Present your solution

Be aware of (I)

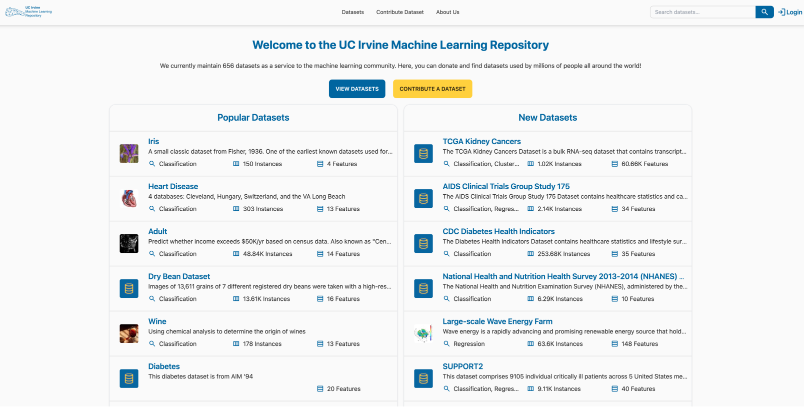
- what data do you need? is it enough?
- legal obligations regarding the data. Who owns the data? Do you have to sign an NDA to work with the data? Do you require a legal agreement?
- For example, since 2018 the UK introduced the Data Protection Act 2018, that implements the General Data Protection Regulation (GDPR) under EU law.

Be aware of (II)

- removing sensitive information (e.g. anonymisation)
- institutions usually have in place a data ethics advisory committees.
- check the size and type of the data (e.g time series? images? spatio-temporal?)
- sample a test set and do not look at it (e.g. avoid data snooping).

Where do we get data from?

UC Irvine Machine Learning Repository



Welcome to the UC Irvine Machine Learning Repository

We currently maintain 656 datasets as a service to the machine learning community. Here, you can donate and find datasets used by millions of people all around the world!

Popular Datasets

Iris
A small classic dataset from Fisher, 1936. One of the earliest known datasets used for...
Classification 150 Instances 4 Features

Heart Disease
4 databases: Cleveland, Hungary, Switzerland, and the VA Long Beach
Classification 303 Instances 13 Features

Adult
Predict whether income exceeds $50K/yr based on census data. Also known as "Cen...
Classification 48.84K Instances 14 Features

Dry Bean Dataset
Images of 13,611 grains of 7 different registered dry beans were taken with a high-res...
Classification 13.61K Instances 16 Features

Wine
Using chemical analysis to determine the origin of wines
Classification 178 Instances 13 Features

Diabetes
This diabetes dataset is from AIM '94
20 Features

New Datasets

TCGA Kidney Cancers
The TCGA Kidney Cancers Dataset is a bulk RNA-seq dataset that contains transcript...
Classification, Cluster... 1.02K Instances 60.66K Features

AIDS Clinical Trials Group Study 175
The AIDS Clinical Trials Group Study 175 Dataset contains healthcare statistics and ca...
Classification, Regres... 2.14K Instances 34 Features

CDC Diabetes Health Indicators
The Diabetes Health Indicators Dataset contains healthcare statistics and lifestyle sur...
Classification 253.68K Instances 35 Features

National Health and Nutrition Health Survey 2013-2014 (NHANES) ...
The National Health and Nutrition Examination Survey (NHANES), administered by the...
Classification 6.29K Instances 10 Features

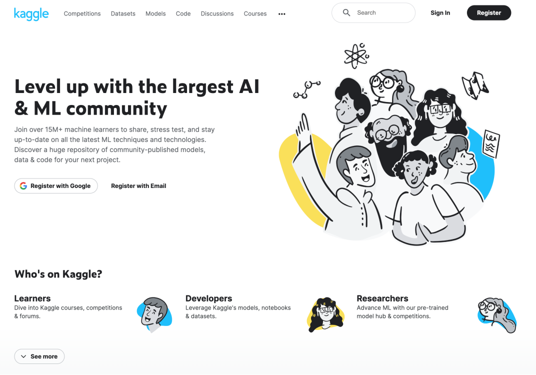
Large-scale Wave Energy Farm
Wave energy is a rapidly advancing and promising renewable energy source that hold...
Regression 63.6K Instances 148 Features

SUPPORT2
This dataset comprises 9105 individual critically ill patients across 5 United States me...
Classification, Regres... 9.11K Instances 40 Features

https://archive.ics.uci.edu/

Where do we get data from?

Kaggle



Level up with the largest AI & ML community

Join over 15M+ machine learners to share, stress test, and stay up-to-date on all the latest ML techniques and technologies. Discover a huge repository of community-published models, data & code for your next project.

Who's on Kaggle?

Learners
Dive into Kaggle courses, competitions & forums.

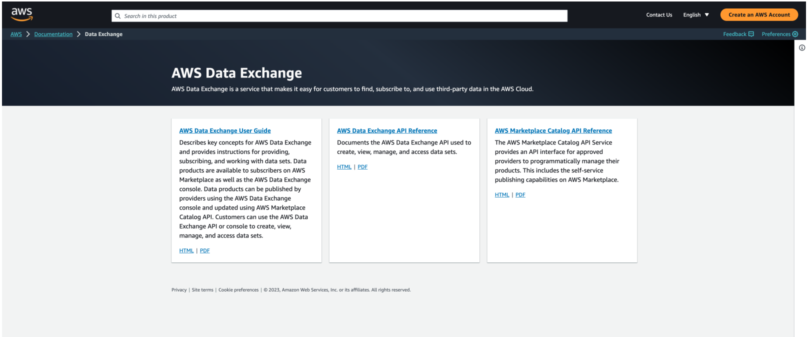
Developers
Leverage Kaggle's models, notebooks & datasets.

Researchers
Advance ML with our pre-trained model hub & competitions.

https://www.kaggle.com/

Where do we get data from?

Amazon's AWS datasets



AWS Data Exchange

AWS Data Exchange is a service that makes it easy for customers to find, subscribe to, and use third-party data in the AWS Cloud.

AWS Data Exchange User Guide

Describes key concepts for AWS Data Exchange and provides instructions for providing, subscribing, and working with data sets. Data products are available to subscribers on AWS Marketplace as well as the AWS Data Exchange console. Data products can be published by providers using the AWS Data Exchange console and updated using AWS Marketplace Catalog API. Customers can use the AWS Data Exchange API or console to create, view, manage, and access data sets.

HTML | PDF

AWS Data Exchange API Reference

Documents the AWS Data Exchange API used to create, view, manage, and access data sets.

HTML | PDF

AWS Marketplace Catalog API Reference

The AWS Marketplace Catalog API Service provides an API interface for approved providers to programmatically manage their products. This includes the self-service publishing capabilities on AWS Marketplace.

HTML | PDF

Privacy | Site terms | Cookie preferences | © 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

https://docs.aws.amazon.com/data-exchange/

Where do we get data from?

- Meta portals listing open data repositories.
- Data portals
- Open data monitor
- Quandl

Three sets: training, validation and test sets

- In the previous session, we mentioned that we use a dataset to train an ML model.
- Actually, when we get the data, we should partition the data intro three sets
  - training set.
  - validation set.
  - test set.
- The partition of the dataset into these three sets is done randomly.
- The training set is usually the biggest one.
- The validation and test sets have roughly the same amount of samples.

Three sets: what for?

- The training set is used to fit the model using the objective function.
- The validation set is used to choose the best predictive model among a set of candidates.
- The test set is used to assess the final performance of the model before shipping the model to production.
- You can use the training and validation sets as you prefer.
- As we said before, never look at your test set when designing your model, only until you have decided what model to use based on the validation set.

Three sets: large datasets

- When you have a large dataset, you can have 70% data for training, 15% for validation and 15% for testing.

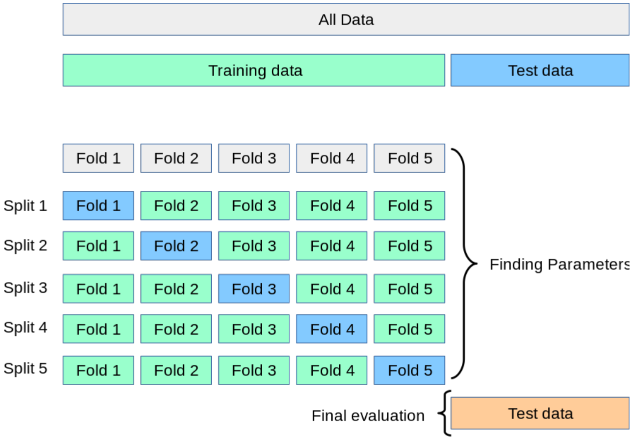
- Nowdays, you can have a dataset with millions of data instances. In those cases, you might want to try 95% for training, 2.5% for validation and 2.5% for testing.

Three sets: smaller datasets

- If you data is small, say in the hundreds or just a few thousands, you want to use as many training instances as possible.

- We can use a strategy known as *cross-validation* to make better use of the training/validation sets.

k-fold cross-validation



All Data

Training data | Test data

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Split 1	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Split 2	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Split 3	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Split 4	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Split 5	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5

Finding Parameters

Final evaluation | Test data

The extreme case is when $k = N$, the number of folds is equal to the number of instances in the training/validation set: leave-one-out (LOO) cross-validation.

Contents

Frame the problem and look at the big picture

Get the data

Explore the data

Prepare the data

Shortlist models and then fine-tune them

Present your solution

To take into account (I)

- ❑ Create a copy of the data for exploration.
- ❑ If you have a large dataset, you can sample a fraction of the data for exploring it.
- ❑ Create a Jupyter Notebook so that you keep track of your data exploration process.
- ❑ Study each feature (inputs and outputs) and its characteristics including
  - name
  - type (continuous, bounded/unbounded, categorical, etc.)
  - percentage of missing values
  - noisiness and type of noise (e.g. outliers)
  - Is this feature useful for the task?
  - type of distribution (Gaussian, Gamma, logarithmic, etc.)

To take into account (II)

- Identify the target atributte (for supervised learning).
- Visualise the data using histograms, scatterplots, maps, etc.
- Study correlations between attributes.
- Identify what transformations you might be able to apply.
- Is there any additional data that you might find useful? Can you get it?

Predicting bike rentals

By Sarah Stierch - Own work, CC BY 4.0, https://commons.wikimedia.org/w/index.php?curid=16075615

Dataset

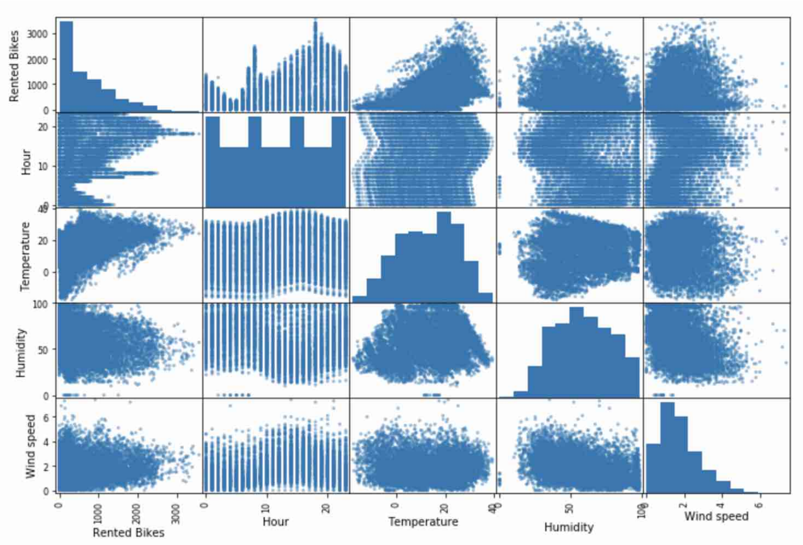
- The feature vector **x** includes the following features: hour, temperature, humidity, wind speed, visibility, Dew point temperature, solar radiation, rainfall, snowfall, seasons, holiday, functioning day.

- The variables hour, temperature, humidity, wind speed, visibility, Dew point temperature, solar radiation, rainfall, snowfall can be considered as continuous.

- The variables seasons, holiday, and functioning day are categorical variables.

- The output variable y is the number of bikes rented.

Scatter plot

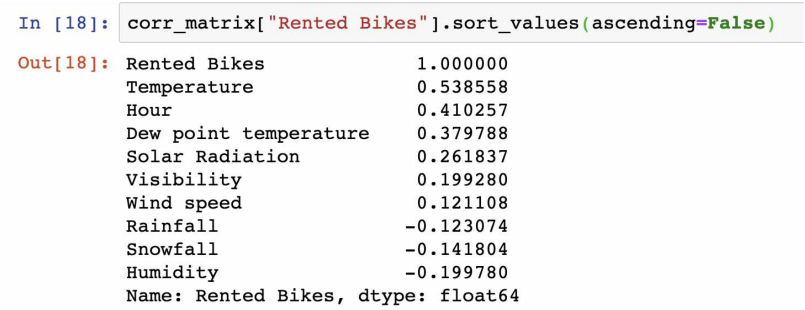


Study correlations between attributes

The correlation coefficient between two RVs X and Y is given as

$$\rho_{X,Y} = \frac{E\{(X-\mu_X)(Y-\mu_Y)\}}{\sigma_X \sigma_Y} = \frac{\sigma_{X,Y}}{\sigma_X \sigma_Y},$$

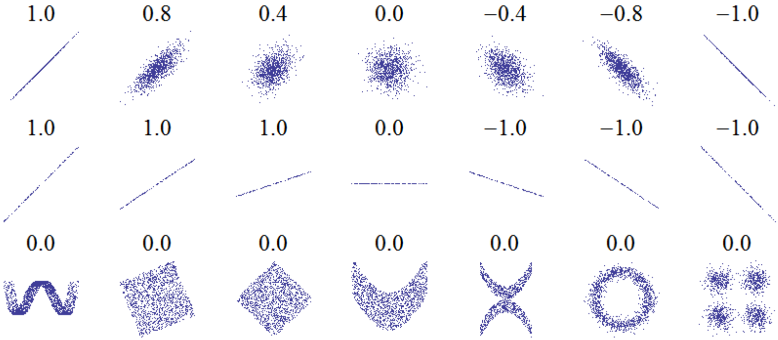
where $-1 < \rho_{X,Y} < 1$ and $\sigma_{X,Y}$ is known as the covariance between X and Y.



```
In [18]: corr_matrix["Rented Bikes"].sort_values(ascending=False)
```

```
Out[18]: Rented Bikes              1.000000
         Temperature               0.538558
         Hour                      0.410257
         Dew point temperature     0.379788
         Solar Radiation           0.261837
         Visibility                0.199280
         Wind speed                0.121108
         Rainfall                 -0.123074
         Snowfall                 -0.141804
         Humidity                 -0.199780
         Name: Rented Bikes, dtype: float64
```

Correlation and dependence



By DenisBoigelot, original uploader was Imagecreator - Own work, original uploader was Imagecreator, CC0,

https://commons.wikimedia.org/w/index.php?curid=15165296

Contents

Frame the problem and look at the big picture

Get the data

Explore the data

Prepare the data

Shortlist models and then fine-tune them

Present your solution

Data cleaning (I)

- Remove the outliers in your data (optional).

- Handle the missing values
  - filling them in using the mean, the median, or any other value.
  - drop the feature if most of the instances have a missing value.
  - drop the instance if you have several instances with missing values.
  - you can add an additional feature indicating whether the instance has a missing feature or not, and then use a value of 0 for the missing feature.

Data cleaning (II)

- Most ML methods require features that are numbers rather than categories (usually appearing as text).

- Also, if the feature is categorical, it is useful to use a different representation.

- In the previous bike rentals example, there were three categorical features
  - season that can take four categories.
  - holiday and functioning day, each taking two categories.

Data cleaning (III)

- For example, the feature season takes values `autumn`, `winter`, `spring` and `summer`.
- The way to handle this feature is to use a representation known as *one-hot encoding* to obtain a higher-dimensional binary representation for each value,

$$\begin{aligned}\texttt{autumn} &= [1,0,0,0]\\ \texttt{winter} &= [0,1,0,0]\\ \texttt{spring} &= [0,0,1,0]\\ \texttt{summer} &= [0,0,0,1]\end{aligned}$$

- The values of the feature season do not have a natural order, therefore one should not map these values to numbers like 1 for `autumn`, 2 for `winter`, 3 for `spring` and 4 for `summer`.
- The ML method will try to find regularities within these ordered values even though they do not exist.

Feature selection and feature engineering

- You have the option to remove features that are uninformative.

- You have the option to discretise a continuous feature (e.g. binning).

- You can also create new features from the ones you have available.

- For example, instead of using feature x, you can use $\log(x)$, $\sqrt{x}$, x^2, etc.

Feature scaling (I)

- Several ML methods do not perform well when the input features have very different scales.

- In the rental bikes example, the variable humidity is in the range 0 to 100, whereas wind speed is in the range 0 to 8.

- Two ways to get all features to have the same scale are *normalisation* (or min-max scaling) and *standardisation* (or z-score normalisation).

Feature scaling (II)

- In normalisation, we map the range of values that a feature takes to the range $[-1, 1]$ or $[0, 1]$.

- The normalisation formula is given as

$$\bar{x}_j = \frac{x_j - \min x_j}{\max x_j - \min x_j},$$

  where $\min x_j$ and $\max x_j$ are the minimum and maximum values for the feature in the training set.

- In standardisation, the features are scaled so that they have mean zero and standard deviation equal to one,

$$\hat{x}_j = \frac{x_j - \mu_j}{\sigma_j},$$

  where μ_j and σ_j are the mean and standard deviation of the feature x_j.

Feature scaling (III)

- Which scaling to use?

- If the dataset is not big and there is time, one can try both and see which one performs better in the validation set.

- If there is no time, as a rule of thumb:
  - unsupervised learning algorithms usually benefit more from standardisation than normalisation.
  - standardisation is preferred if the feature has already a distribution close to a Gaussian.
  - if the feature has outliers standardisation is preferred since normalisation will squeeze all the other values into a small range.
  - normalisation is preferred in all the other cases.

Contents

Frame the problem and look at the big picture

Get the data

Explore the data

Prepare the data

Shortlist models and then fine-tune them

Present your solution

Shortlist promising models

- If the data is big, use a smaller subset of the data to try different models in a reasonable time.
- Try different models from several categories (e.g. linear, non-linear, probabilistic, non-probabilistic).
- This can be easily done with scikit-learn.
- Measure and compare the performance of the ML models you used.
  - make sure you are using the exact same data in the training and the validation sets.
  - when using cross-validation, compute the mean and standard deviation of the performance measure over the k-folds for each model.
- shortlist two to three models that look promising.

Fine-tune the system (I)

- It is a good idea to use as much data as you can at this stage.
- Fine-tune the hyperparameters of your model using cross-validation.
  - data transformations are hyperparameters, e.g. should I input missing data with zero or with the mean?
  - use random search or grid search to explore hyperparameters
  - if the training takes long, use Auto-ML to fine-tune the hyperparameters.

Fine-tune the system (II)

- Once you are confident about your final model, usually the one that performs better on average on the validation data:
  - merge the training and the validation datasets
  - fit the model again using the set of hyperparameters you found before.

- Finally, use the test data to assess the generalisation error of your ML system.
  - Do not try to change your model based on the performance on the test data, you might start overfitting your model to the data.

Contents

Frame the problem and look at the big picture

Get the data

Explore the data

Prepare the data

Shortlist models and then fine-tune them

Present your solution

Present your solution

- Document what you have done.
- Create a nice presentation.
- Explain why your solution achieves the business objective.
- Bring up interesting points you noticed in the process.