Supervised : basics

The objective of those exercises is to practice the basics of supervised machine learning.

1 Fundamentals

1. Preparing the dataset
   (a) Get ready to use the same dataset as for the previous class.
   (b) Create a new dataset containing only numerical columns: budget, popularity, revenue, runtime, vote_average, vote_count. Clean them by removing rows with NaN.
   (c) From the release_date, extract 2 columns: year, and month. You can use `to_datetime` (using option: `errors=...` to get rid of errors). Add those columns to the numerical dataset
   (d) Transform the first column, "adult", into a boolean variable, whose value can be 0 or 1. You can use pandas `get_dummies` or do it manually (or with sklearn `OneHotEncoder`, less convenient). Add to the numerical dataset
   (e) Let's say that we want to predict the column *popularity*, a score given by the platform the data was extracted from. Remove this column from the table, and keep it in a separate list. The order of elements in the list allows to match them with variables
   (f) Split your dataset into a train and test set. You can do it manually or use sklearn `train_test_split` function. Keep for instance 1/3 as test set.
2. First predictions : linear
   (a) Train a linear regression with sklearn. You can use the `LinearRegression` class, and method `fit`.
   (b) Compute the scores we have seen, using corresponding functions in sklearn and the `predict` method of the linearRegression class. Do it first by using the train set, and then using the test set. Compare the difference.
   (c) To get a more intuitive idea of the performance, plot the relation between the target variable and the prediction (e.g., seaborn scatterplot, x=target variable, y=your prediction). With a perfect prediction, you should observe a diagonal line.
   (d) Check the coefficients `coef_` and the intercept (β_0), `intercept_`. Discuss with your peers about their interpretation. What about their magnitude? Sign ?
3. Questioning our results
   (a) Remove anomalous rows with zeros in Budget, re-run your train and check the differences. Can we say that a lower/larger RMSE/R2 is a proof of being worst/better?
   (b) Standardize your data using `StandardScaler` and compare the results. Compare the coefficients, some should have changed a lot, other not so much.
   (c) Re-run the last experiment 10 times, each time with a different split. Print or plot the variation in scores (MSE, R2)... There should be relatively strong differences: remember that comparing a single run can be biased...
4. Decision tree

(a) Train a Decision tree (`DecisionTreeRegressor`) with sklearn, using default parameters, using un-normalized data (normalization is useless with trees, and makes interpretation harder).

(b) Compute the scores, comparing between evaluation on training and on testing sets, and with the linear approach.

(c) The default parameters make the tree overfit. Play with the parameters `max_depth` , `min_samples_leaf` , `max_leaf_nodes` , to try to limit the overfit (you should be able to improve over the linear regression)

(d) Train a tree small enough to be visualized, and plot it. You can use the built-in tool following the documentation `https://scikit-learn.org/stable/modules/tree.html#tree`. The `graphviz` method usually gives the nicest results.

(e) Try to interpret the tree. Discuss with your peers.

2 Going Further

(a) Let's say that we would now like to predict if a movie will be profitable or not. Compute a new column defined as revenue - budget, and transform into a boolean value, True if positive, False if negative. Be careful with missing values encoded as zero...

(b) Train a Logistic Regression and a Decision Tree Classifier.

(c) We want to explore the robustness of those predictions. Doing it manually or using sklearn tools `https://scikit-learn.org/stable/modules/classes.html#module-sklearn.model_selection`, generate several train/test set pairs, and compare (rather for the regression task than the classification one) the robustness of scores and of fitted models (coefficients, ploted trees)