Energy Market - Data Analysis

André Cruz
Edgar Carneiro
Xavier Fontes

Agents and Distributed Artificial Intelligence - AIAD
Faculty of Engineering
University of Porto
2018

Problem Description

The problem consists in a **classification** task, aiming to **identify a Consumer's class** (Normal, Lazy, or Ecological) from its **behaviour** (represented by the last N contracts that Consumer has accepted).

Each contract is represented by three features: price (per energy unit), distance (between Consumer and Broker), and percentage of renewable energy that the Broker receives.

We had initially planned to additionally predict the energy market's performance from its start settings, but this route proved difficult for acquiring meaningful amounts of data, as one run represented only one learning instance, and each run took 5 to 7 minutes to complete. In this presentation we discuss only the first prediction scenario.

Experiments

- In our experiments, we found that using 5 contracts to predict a Consumer's class performed well.
- Input: feature vector of size 3*5 = 15.
    - Independent variables: the price, distance, and percentage of renewable energy of each of the 5 contracts.
- Output (dependent variable): the Consumer's category.

- We performed experiments with different techniques for generating training data, different prediction models, and different hyperparameters for each model.
- Performance reported on a holdout set, with 30% of the data.

Dataset Generation

- For each Consumer, we have a list of all the contracts it signed during the simulation.
- We used two techniques to generate a train dataset from the gathered data:
    a. Splitting the list in sublists of 5 contracts each, and generating one training instance from each sublist.
    b. Generating a learning instance from a rolling window of 5 contiguous contracts.

Contract 1	Contract 2	Contract 3	Contract 4	...

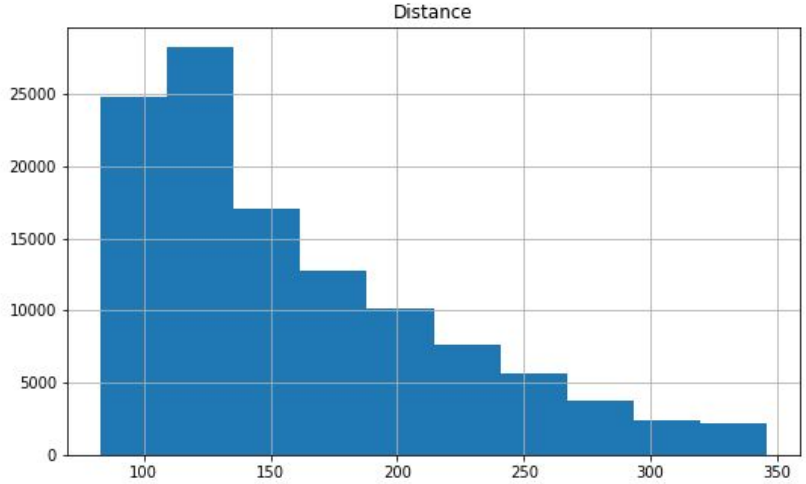
a.

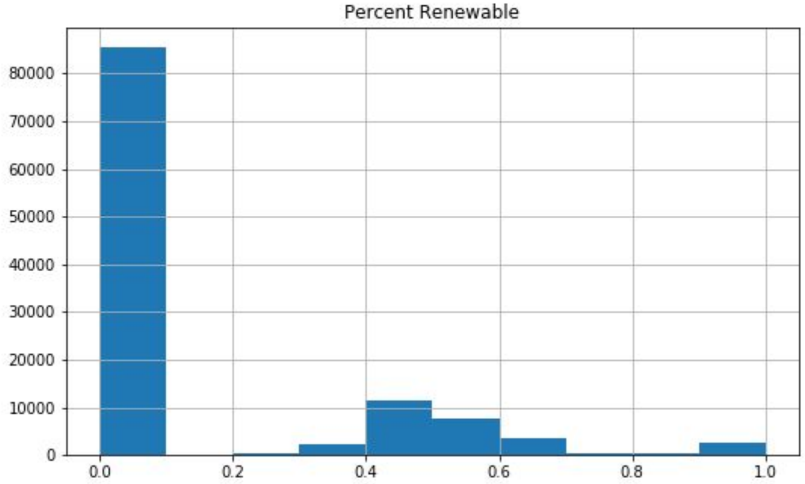
Contract 1	Contract 2
Contract 3	Contract 4

b.

Contract 1	Contract 2
Contract 2	Contract 3
Contract 3	Contract 4

Data Analysis

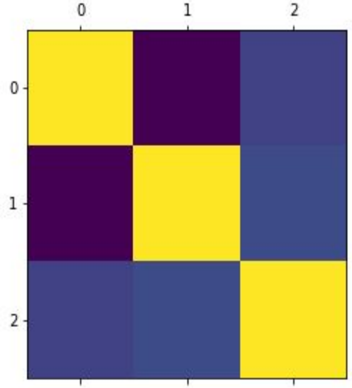




	Price per Unit	Percent Renewable	Distance
count	114700.000000	114700.000000	114700.000000
mean	14.804603	0.141561	160.185618
std	1.685296	0.257165	60.036497
min	9.000000	0.000000	82.873400
25%	14.000000	0.000000	111.865990
50%	15.000000	0.000000	141.449630
75%	16.000000	0.342051	195.064090
max	23.000000	1.000000	345.974000

Data Analysis

- Even class distribution among the three classes
    - A baseline system would achieve at max 35,9% accuracy
- Negative correlation between Ecology and Price (-0.28)
    - Due to our pricing for renewable sources being lower (based on real data).



Class	Freq.	Percent.
RegularConsumer	8243	35,9%
EcologicalConsumer	7802	34,0%
LazyConsumer	6895	30,1%

Correlation Matrix

Var.	Price	Eco	Dist
Price	1.0	-0.28	-0.02
Ecology	-0.28	1.0	0.01
Distance	-0.02	0.01	1.0

Data Analysis with RapidMiner

- We aggregate data from 10 separate runs of our SAJaS simulation, using DataRecorder to store the information of the contracts.
- We experiment with several classification algorithms: **Random Forest**, **Gradient Tree Boosting**, **Naive Bayes**, **Deep Learning**.
- We test the models with both methods of generating training data (even split or rolling window, as previously discussed).

Data Analysis with RapidMiner - Results

a. Even-split data

	Acc. on Train(%)	Acc. on Test(%)
Gradient Boosting	**100.00**	48.50
Random Forest	**89.49**	40.92
Naive Bayes	**36.97**	**36.86**
Deep Learning	**44.99**	35.75

b. Rolling window data

	Acc. on Train(%)	Acc. on Test(%)
Gradient Boosting	99.94	**85.82**
Random Forest	75.32	**50.53**
Naive Bayes	36.27	36.06
Deep Learning	39.20	**36.29**

Conclusions

- The data extracted features meaningful information for the prediction task.
- Each Consumer class has distinguishable behaviours.
- Random Forest and Gradient Boosted Tree models performed the best.
- Model could be useful for the Broker agents to use in contract negotiations, as they'd better understand the needs of each specific Consumer.
- RapidMiner is useful for fast prototyping, but does not provide the same in-depth usability of other tools (*e.g.* scikit-learn).
    - We experienced x2 speed-up when using python with scikit-learn (vs Rapid Miner).

Future Work

- Experiment different configurations and with even more data because the Deep Learning option of RapidMiner gave disappointing results.
- Try different experiments using data collected from the simulations run. Initially we thought about predicting the average consumer satisfaction using the parameters of the simulation but several problems with respect to gathering the data made it infeasible in the time we had.
- It would be interesting to understand if these models could be supplied to the simulation itself in order to have the Brokers understand the needs of each of its Consumers (by predicting their category).