Football Dataset
Random Forest + Neural Networks

Patrícia Kovaleski

Elementos de Processamento Digital de Sinais - 2018/2

The Dataset

- 30 matches
  - 5 championships
  - 15 stadiums
  - 4 commentators
- 20 features
  - Audio
  - Image
  - Video

Figure 1: Frames extracted from the dataset videos. Source: [1]

Pre-Processing

---

1. Covariance

   - em_mcs_energy
   - em_mcs_energy_diff
   - em_mcs_energy_diff_ascending

   =

   - em_cs_energy
   - em_cs_energy_diff
   - em_cs_energy_diff_ascending

- 16 features remain

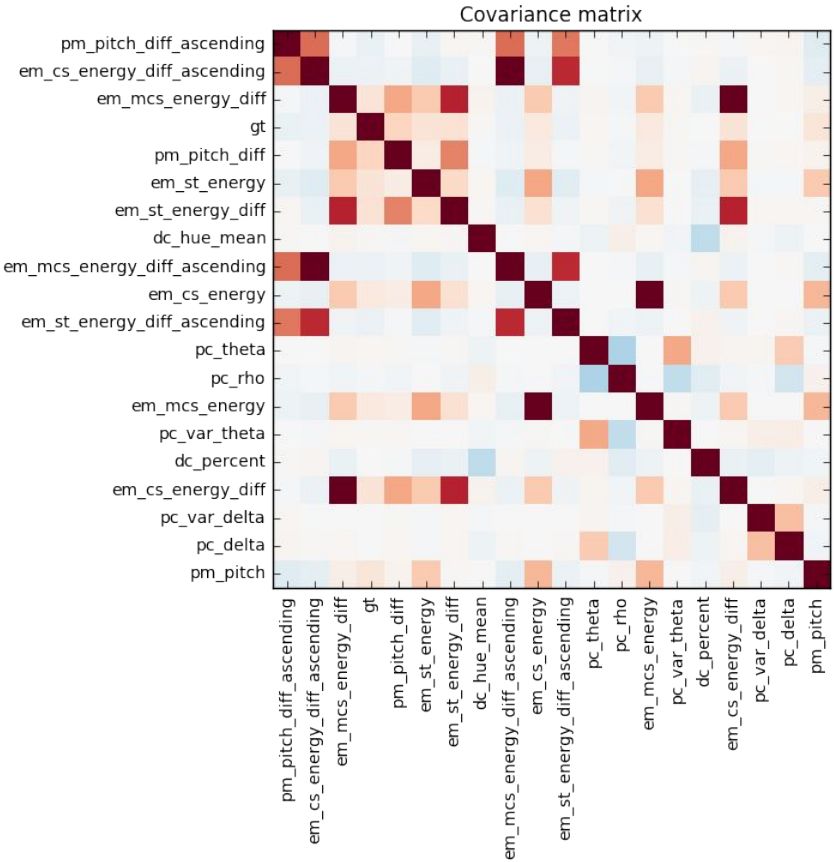


Figure 2: Covariance matrix for the dataset features

Pre-Processing

2. Ballance Data
    a. ~ 3.5min of "good moments" in a 90min match.
    b. Solution: Downsampling the majority class

3. Normalization
    a. Remove mean and standart deviation

Random Forests

- Ensemble of decisions trees
- Selects a random subset of features
- Accurate and stable
- Avoid overfitting

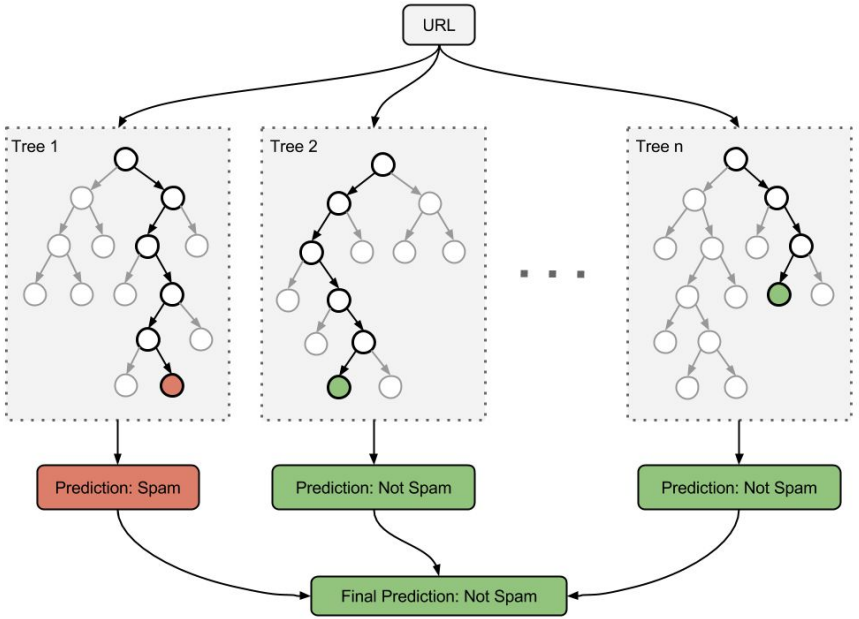


Figure 3: Random forest graph scheme. Source [2]

First Attempts

- More trees, still the same threshold:
  - ~.90 / ~.75
- Speedup of 30% in (*)

# trees	Precision	Recall
10	0.898	0.732
50	0.901	0.760
100	0.905	0.771
150	**0.908**	**0.775**

Table 1: Results of precision and recall for random forests with different number of trees.

Depth	Precision	Recall
10 (*)	0.885	**0.793**
50	**0.900**	0.754
100	0.899	0.750
150	0.897	0.759

Table 2: Results of precision and recall for random forests with different depths.

First Attempts

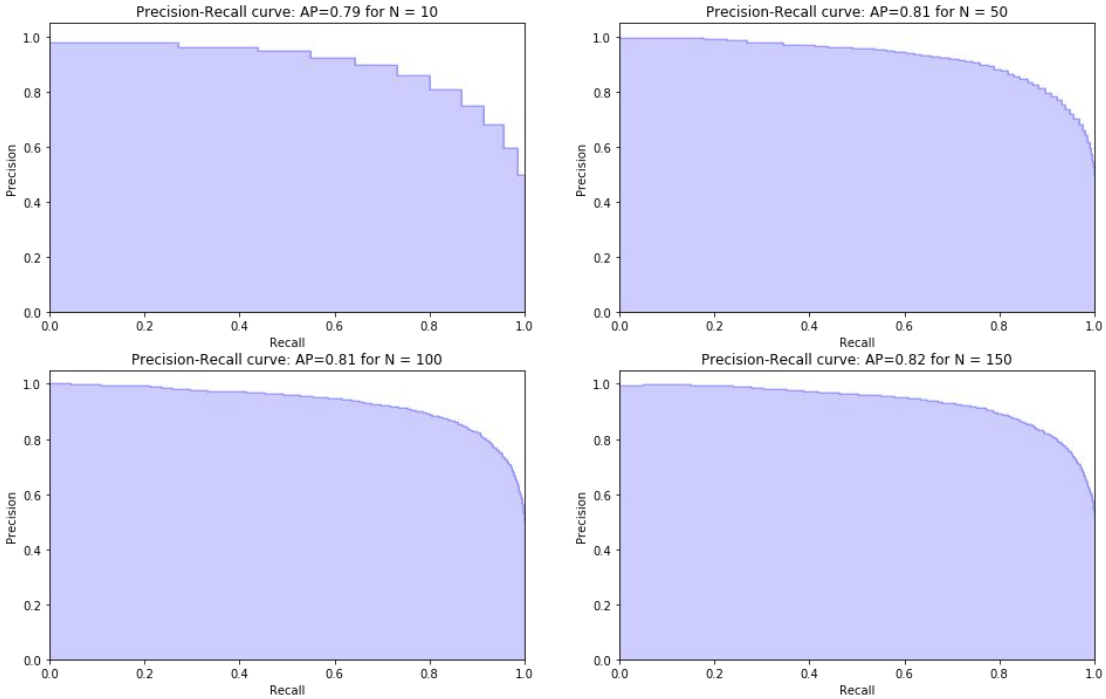
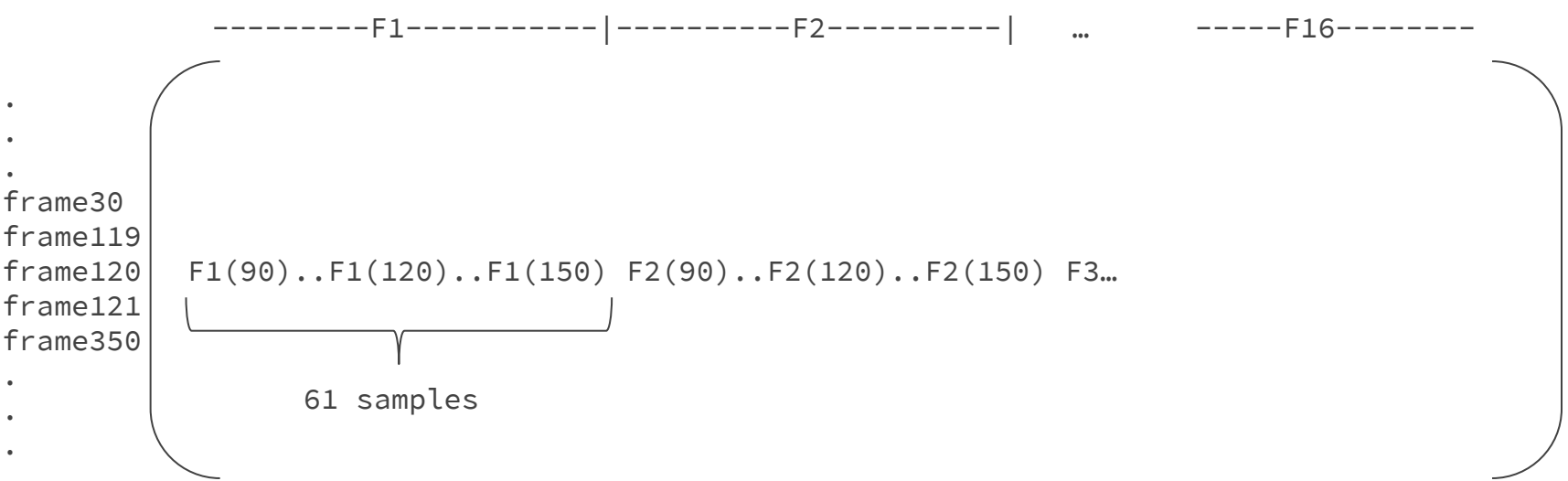


Figure 4: Precision-Recall curve for different # of trees (N).

Temporal Data Representation



Temporal Data Representation

Skip	1		2		4		8	
# trees	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
10	0.929	0.891	0.964	0.805	0.971	0.743	0.974	0.646
50	0.924	0.933	0.964	0.876	0.970	0.807	0.979	0.758
100	0.924	0.936	0.967	0.885	0.973	0.828	0.979	0.751
150	0.925	0.936	0.967	0.890	**0.973**	**0.830**	0.979	0.751

Table 3: Results of precision and recall for random forests with different number of trees and skips between features in the temporal representation.

Neural Networks

- Multilayer Perceptron
- Hyperparameters grows with dept
- Needs normalization
- Many manual parameters to set
  - # Layers and # neurons
  - Layers activation functions
  - Learning rate
  - Initialization

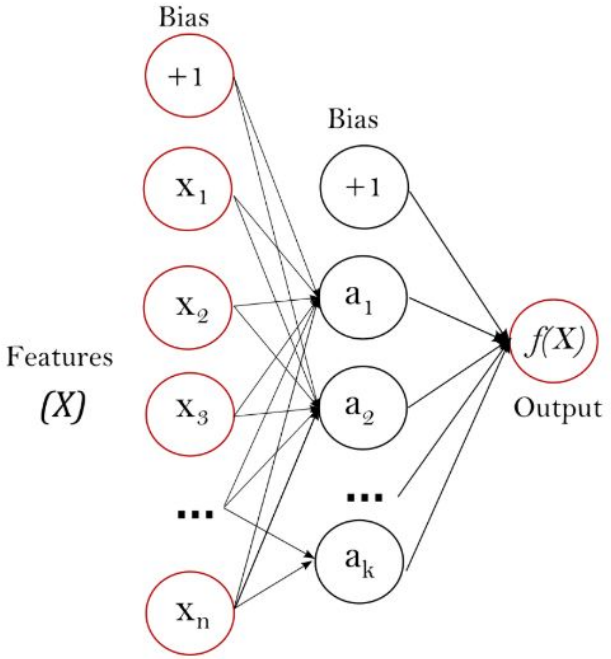


Figure 5: One hidden layer MLP. Source [3]

MLP architectures

Type 1:

```
nn = Classifier(
  layers=[
    Layer('Tanh', units=10),
    Layer('Tanh', units=10),
    Layer('Softmax')],
  learning_rate=0.01,
  n_iter=25,
  valid_set=(X_val,y_val))
```

Type 2:

```
nn = Classifier(
  layers=[
    Layer('Tanh', units=10),
    Layer('Tanh', units=10),
    Layer('Tanh', units=10),
    Layer('Softmax')],
  learning_rate=0.01,
  n_iter=25,
  valid_set=(X_val,y_val))
```

Type 3:

```
nn = Classifier(
  layers=[
    Layer('Tanh', units=10),
    Layer('Linear',units=10),
    Layer('Softmax')],
  learning_rate=0.01,
  n_iter=25,
  valid_set=(X_val,y_val))
```

MLP architectures

Type 1:

```
nn = Classifier(
    layers=[
        Layer('Tanh', units=10),
        Layer('Tanh', units=10),
        Layer('Softmax')],
    learning_rate=0.01,
    n_iter=25,
    valid_set=(X_val,y_val))
```

Precision: 0.820 Recall: 0.804

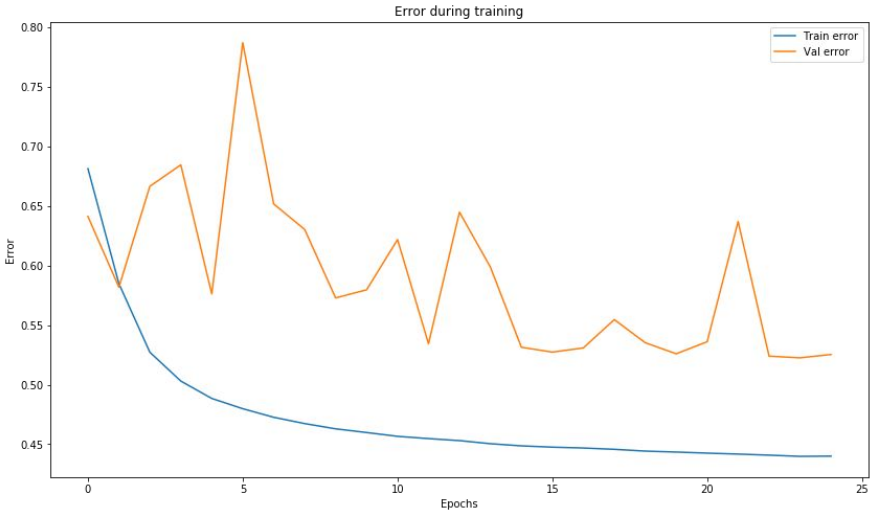


Figure 6: Training and validation error during the MLP type 1 training.

MLP architectures

Type 2:

```
nn = Classifier(
    layers=[
        Layer("Tanh", units=10),
        Layer("Tanh", units=10),
        Layer("Tanh", units=10),
        Layer("Softmax")],
    learning_rate=0.01,
    n_iter=25,
    valid_set=(X_val,y_val))
```

Precision: 0.839 Recall: 0.763

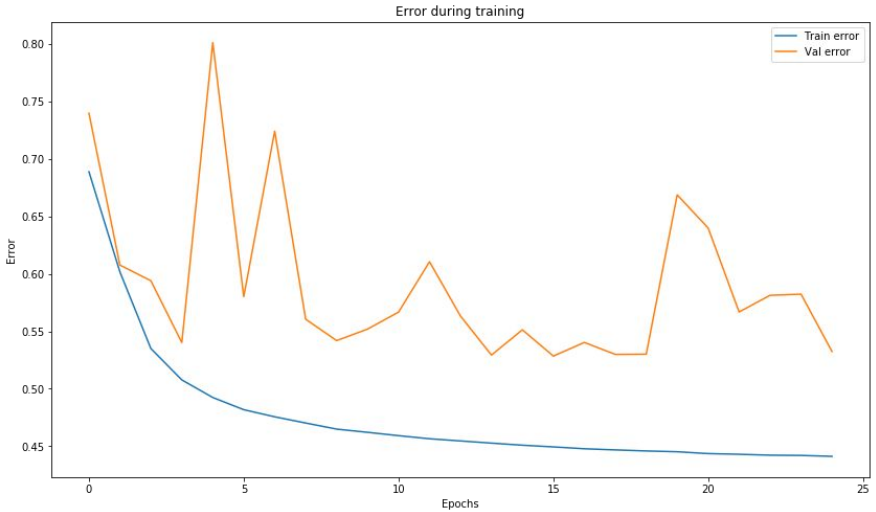


Figure 7: Training and validation error during the MLP type 2 training.

MLP architectures

Type 3:

```
nn = Classifier(
    layers=[
        Layer("Tanh", units=10),
        Layer("Linear", units=10),
        Layer("Softmax")],
    learning_rate=0.01,
    n_iter=25,
    valid_set=(X_val,y_val))
```

Precision: 0.896 Recall: 0.878

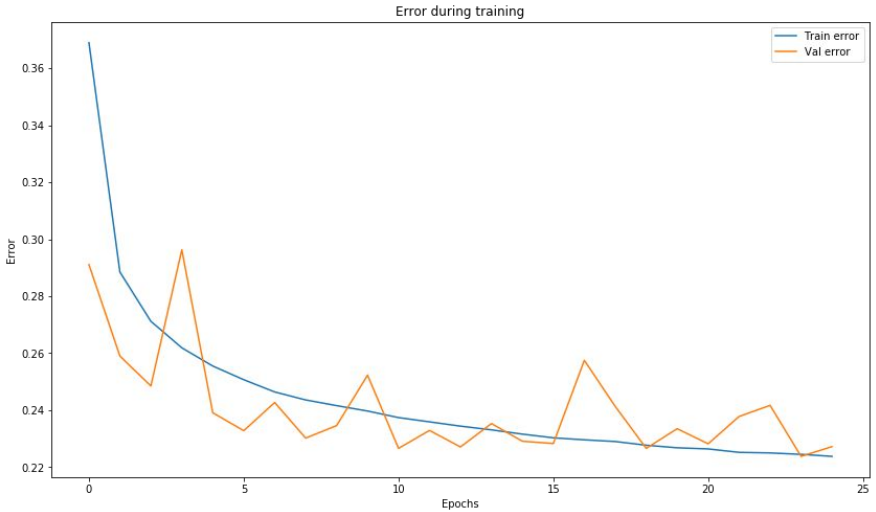


Figure 8: Training and validation error during the MLP type 3 training.

Temporal Data Representation

- Using skip = 8

# Layers	Precision	Recall
2 tanh	0.902	**0.949**
3 tanh	**0.912**	0.867
2 tanh + 1 linear	0.896	0.878

Table 4: Results of precision and recall for different MLP types.

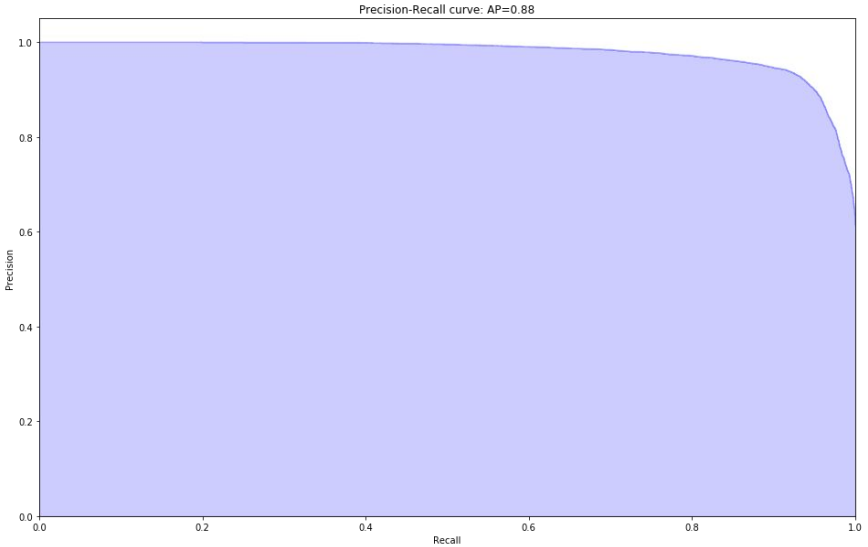


Figure 9: Precision-Recall curve for MLP type 1 on new data representation.

Temporal Data Representation

- Use lower learning rates values
- Train for more epochs
- Evaluate others layers types (Rectifier, Sigmoid)
- Try different initializations

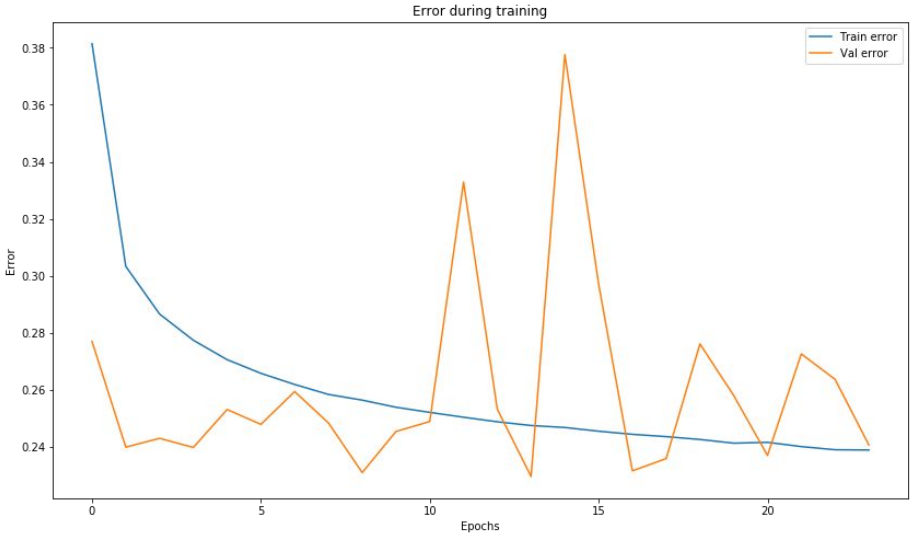


Figure 10: Training and validation error during the MLP type 1 training on new data representation.

Random Forest vs. Neural Network

- Computational complexity
- Possible improvements
- Applications

	Static data		Temporal Data	
	Random Forest	**Neural Networks**	**Random Forest**	**Neural Networks**
Precision	**0.908**	0.896	**0.973**	0.902
Recall	0.775	**0.878**	0.830	**0.949**

Table 5: Best results of precision and recall for each case.

Bibliography

[1] Vasconcelos, Luiz. Sumarização automática em melhores momentos de transmissões televisivas de futebol. 2011

[2] https://adpearance.com/blog/automated-inbound-link-analysis

[3] http://scikit-learn.org/stable/modules/neural_networks_supervised.html