Experimentaci?n MNIST

January 10, 2019

Todo el proceso de experimentar con el dataset segment

```
In [1]: from demo_utils.demo0 import Demo0
        from demo_utils.demo3 import Demo3
        from sklearn.model_selection import GridSearchCV
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.linear_model import LogisticRegression
        from sklearn.svm import LinearSVC
        from demo_utils.general import get_data
        import warnings
        d = Demo0()
        testing_dataset = 'mnist'
        run_mode = False
```

```
In [2]: warnings.filterwarnings('ignore')
        # warnings.filterwarnings('once')
        #warnings.filterwarnings(action='once')
```

1 Con los modelos simples

Ver cómo se comportan los modelos simples con este dataset

Habría que encontrar los mejores hiper-parámetros para cada uno de ellos

Los híper-parámetros que tienen son:

Decision Tree: - max_depth - min_samples_split - min_samples_leaf - min_weight_fraction_leaf - max_leaf_nodes - min_impurity_decrease

Logit: - C

Linear SVM: - C

1.0.1 HíperParámetros con DecisionTree

```
In [4]: if run_mode:
            tuned_parameters = [{
                'max_depth': [10, 100, 1000, 10000, 100000, None],
                'min_samples_split': [2, 3, 4, 5, 6, 7, 8, 9],
                'min_samples_leaf': [1, 2, 3, 4],
                'min_weight_fraction_leaf': [.0, .1, .3],
                'max_leaf_nodes': [10, 50, 100, 1000, None],
```

```
        'min_impurity_decrease': [.0, .2, .6],
    }]
    clf = GridSearchCV(DecisionTreeClassifier(), tuned_parameters, cv=10)
    data = get_data(testing_dataset, n_ins=2000)
    data_train = data['data_train']
    data_test = data['data_test']
    target_train = data['target_train']
    target_test = data['target_test']

    clf.fit(data_train, target_train)
    dt_best_params = clf.best_params_
    print('DecisionTree best params.')
    print(dt_best_params)
```

- **max_depth**: 1000
- **max_leaf_nodes**: 100
- **min_impurity_decrease**: 0.0
- **min_samples_leaf**: 1
- **min_samples_split**: 2
- **min_weight_fraction_leaf**: 0.0

1.0.2 HíperParámetros con Logit

```
In [5]: if run_mode:
            tuned_parameters = [{
                'C': [0.001, .001, .01, .1, 1, 10, 100],
            }]
            clf = GridSearchCV(LogisticRegression(multi_class='multinomial', solver='lbfgs'),
                               tuned_parameters, cv=10, iid=False)
            data = get_data(testing_dataset, n_ins=2000)
            data_train = data['data_train']
            data_test = data['data_test']
            target_train = data['target_train']
            target_test = data['target_test']

            clf.fit(data_train, target_train)
            print('LogisticRegression best params.')
            logit_best_params = clf.best_params_
            print(logit_best_params)
```

- **C**: 0.01

(Tener en cuenta que da un convergence warning, que aquí se ignora)

1.0.3 HíperParámetros con LinearSVC

```
In [6]: if run_mode:
            tuned_parameters = [{
                'C': [.000001, .00001, .0001, .001, .01, .1, 1],
```

```
            }]
            clf = GridSearchCV(LinearSVC(), tuned_parameters, cv=10, iid=False)
            data = get_data(testing_dataset, n_ins=2000)
            data_train = data['data_train']
            data_test = data['data_test']
            target_train = data['target_train']
            target_test = data['target_test']

            clf.fit(data_train, target_train)
            print('LinearSVC best params.')
            linear_svc_best_params = clf.best_params_
            print(linear_svc_best_params)
```

- **C**: 0.001 (oscila un poco)

```
In [7]: if not run_mode:
            # Para no tener que ejecutar el CV otra vez
            dt_best_params = {
                'max_depth': 1000,
                'max_leaf_nodes': 100,
                'min_impurity_decrease': 0.0,
                'min_samples_leaf': 1,
                'min_samples_split': 2,
                'min_weight_fraction_leaf': 0.0
            }
            logit_best_params = {'C': 0.01}
            linear_svc_best_params = {'C': 0.001}
```

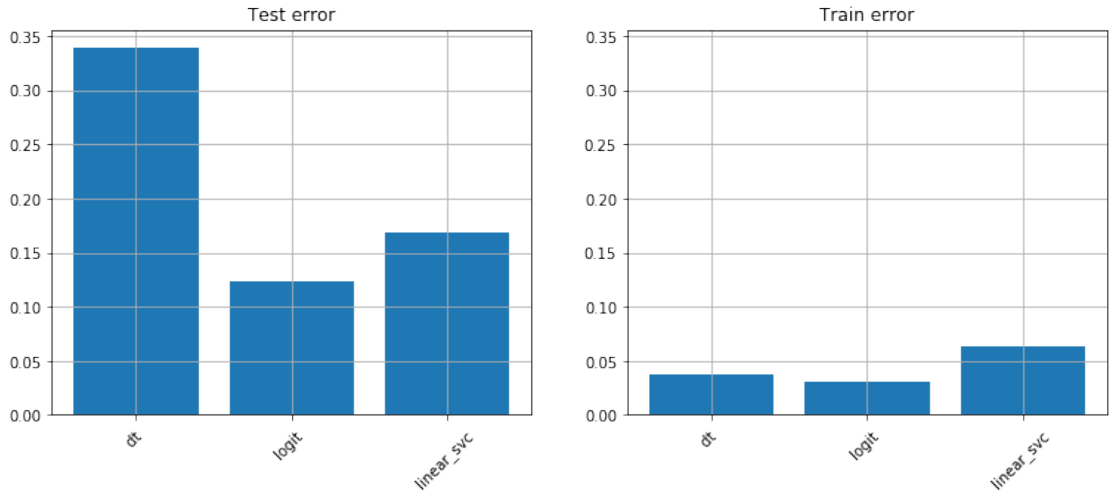
```
In [8]: data = {
            'dts_name': testing_dataset,
            'dts_size': 1000,
            'features_range': (30, 100),
            'rbfsampler_gamma': 0.2,
            'nystroem_gamma': 0.2,
            'hparams': {
                'dt': dt_best_params,
                'logit': logit_best_params,
                'linearsvc': linear_svc_best_params,
            },
            'models': [
                {'model_name': 'dt',
                 'sampler_name': 'identity',
                 'box_type': 'none',
                 'n_estim': None,
                 'pca': False,
                 'pca_first': False
                },
                {'model_name': 'logit',
```

```
            'sampler_name': 'identity',
            'box_type': 'none',
            'n_estim': None,
            'pca': False,
            'pca_first': False
           },
           {'model_name': 'linear_svc',
            'sampler_name': 'identity',
            'box_type': 'none',
            'n_estim': None,
            'pca': False,
            'pca_first': False
           }
       ]
   }
```

```
In [9]: d.non_interactive(**data)
```

2 Demo genérica

- Dataset: **mnist**
- Size: **1000**



No obtienen muy buenos resultados. A lo mejor con PCA mejora...

2.0.1 Añadiendo PCA

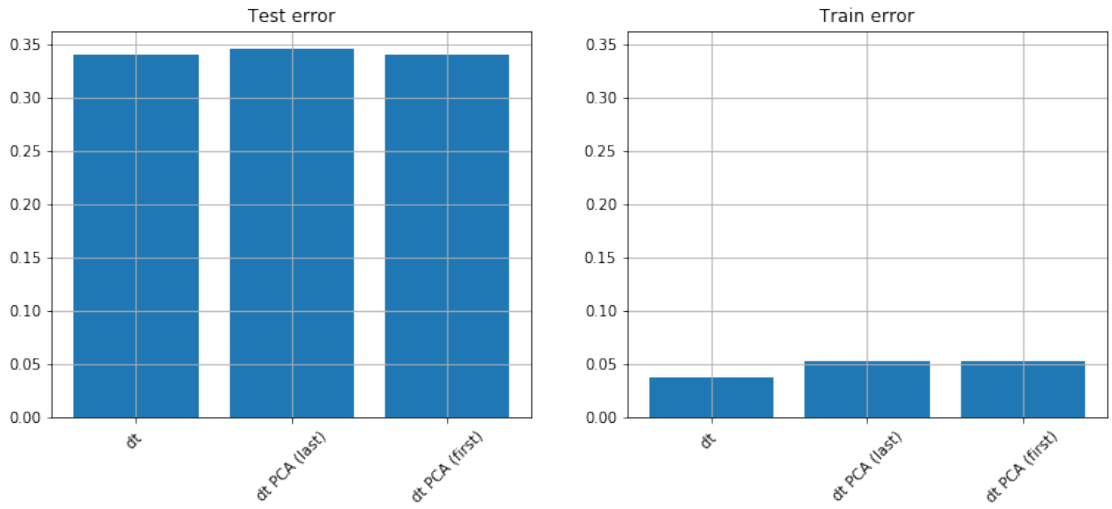
```
In [10]: data = {
             'dts_name': testing_dataset,
             'dts_size': 1000,
             'features_range': (30, 100),
```

```
            'rbfsampler_gamma': 0.2,
            'nystroem_gamma': 0.2,
            'hparams': {
                'dt': dt_best_params,
                'logit': logit_best_params,
                'linearsvc': linear_svc_best_params,
            },
            'models': [
                {'model_name': 'dt',
                 'sampler_name': 'identity',
                 'box_type': 'none',
                 'n_estim': None,
                 'pca': False,
                 'pca_first': False
                },
                {'model_name': 'dt',
                 'sampler_name': 'identity',
                 'box_type': 'none',
                 'n_estim': None,
                 'pca': True,
                 'pca_first': False
                },
                {'model_name': 'dt',
                 'sampler_name': 'identity',
                 'box_type': 'none',
                 'n_estim': None,
                 'pca': True,
                 'pca_first': True
                }
            ]
        }
```

```
In [11]: d.non_interactive(**data)
```

3 Demo genérica

- Dataset: **mnist**
- Size: **1000**



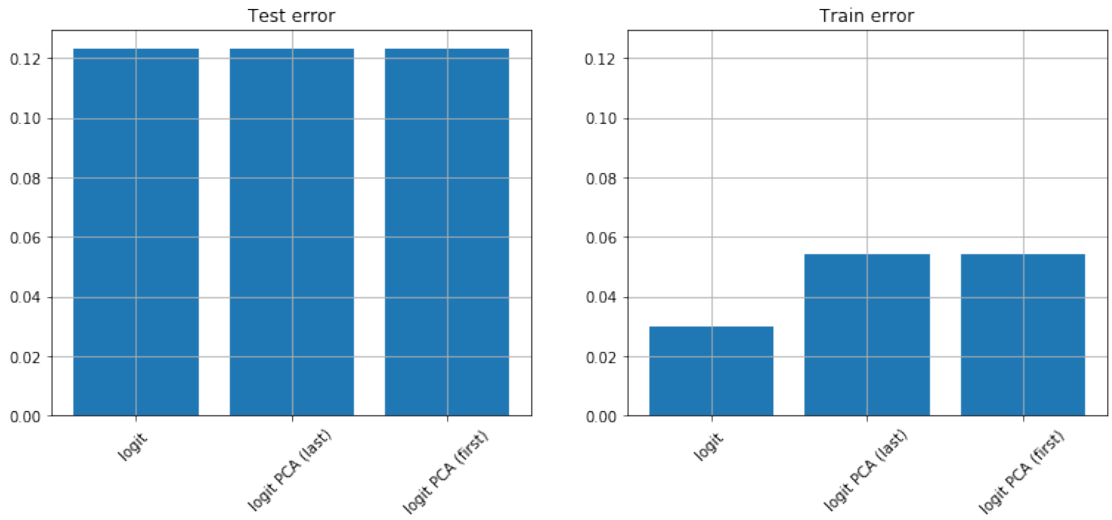

```
In [12]: data = {
             'dts_name': testing_dataset,
             'dts_size': 1000,
             'features_range': (30, 100),
             'rbfsampler_gamma': 0.2,
             'nystroem_gamma': 0.2,
             'hparams': {
                 'dt': dt_best_params,
                 'logit': logit_best_params,
                 'linearsvc': linear_svc_best_params,
             },
             'models': [
                 {'model_name': 'logit',
                  'sampler_name': 'identity',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 },
                 {'model_name': 'logit',
                  'sampler_name': 'identity',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': True,
                  'pca_first': False
                 },
                 {'model_name': 'logit',
                  'sampler_name': 'identity',
                  'box_type': 'none',
                  'n_estim': None,
```

```
                'pca': True,
                'pca_first': True
               }
           ]
       }
```

```
In [13]: d.non_interactive(**data)
```

4 Demo genérica

- Dataset: **mnist**
- Size: **1000**



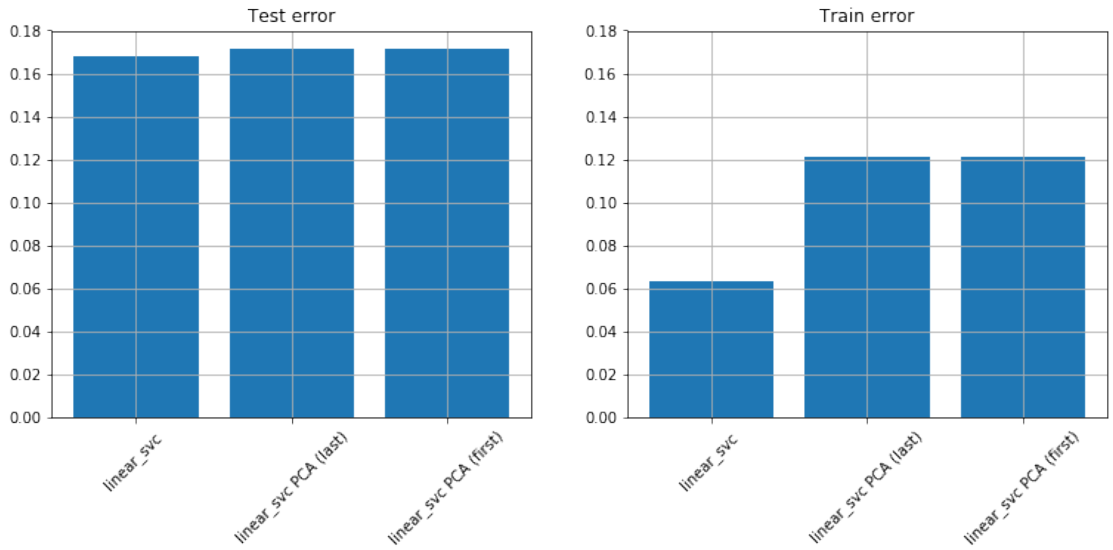

```
In [14]: data = {
             'dts_name': testing_dataset,
             'dts_size': 1000,
             'features_range': (30, 100),
             'rbfsampler_gamma': 0.2,
             'nystroem_gamma': 0.2,
             'hparams': {
                 'dt': dt_best_params,
                 'logit': logit_best_params,
                 'linearsvc': linear_svc_best_params,
             },
             'models': [
                 {'model_name': 'linear_svc',
                  'sampler_name': 'identity',
                  'box_type': 'none',
                  'n_estim': None,
```

```
                'pca': False,
                'pca_first': False
               },
               {'model_name': 'linear_svc',
                'sampler_name': 'identity',
                'box_type': 'none',
                'n_estim': None,
                'pca': True,
                'pca_first': False
               },
               {'model_name': 'linear_svc',
                'sampler_name': 'identity',
                'box_type': 'none',
                'n_estim': None,
                'pca': True,
                'pca_first': True
               }
           ]
       }
```

```
In [15]: d.non_interactive(**data)
```

5 Demo genérica

- Dataset: **mnist**
- Size: **1000**



PCA no tiene ningún efecto positivo con este problema. No lo vamos a usar

5.1 Sampler con los modelos simples

```
In [16]: feature_range = (30, 800)
```

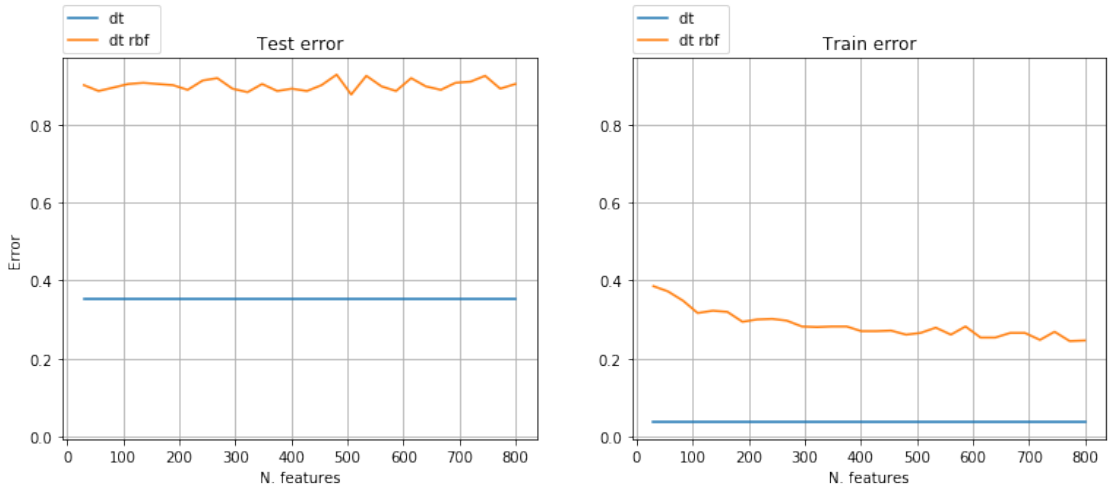
DT

```
In [17]: data = {
             'dts_name': testing_dataset,
             'dts_size': 1000,
             'features_range': feature_range,
             'rbfsampler_gamma': 0.2,
             'nystroem_gamma': 0.2,
             'hparams': {
                 'dt': dt_best_params,
                 'logit': logit_best_params,
                 'linearsvc': linear_svc_best_params,
             },
             'models': [
                 {'model_name': 'dt',
                  'sampler_name': 'identity',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 },
                 {'model_name': 'dt',
                  'sampler_name': 'rbf',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 }
             ]
         }
```

```
In [18]: d.non_interactive(**data)
```

6 Demo genérica

- Dataset: **mnist**
- Size: **1000**



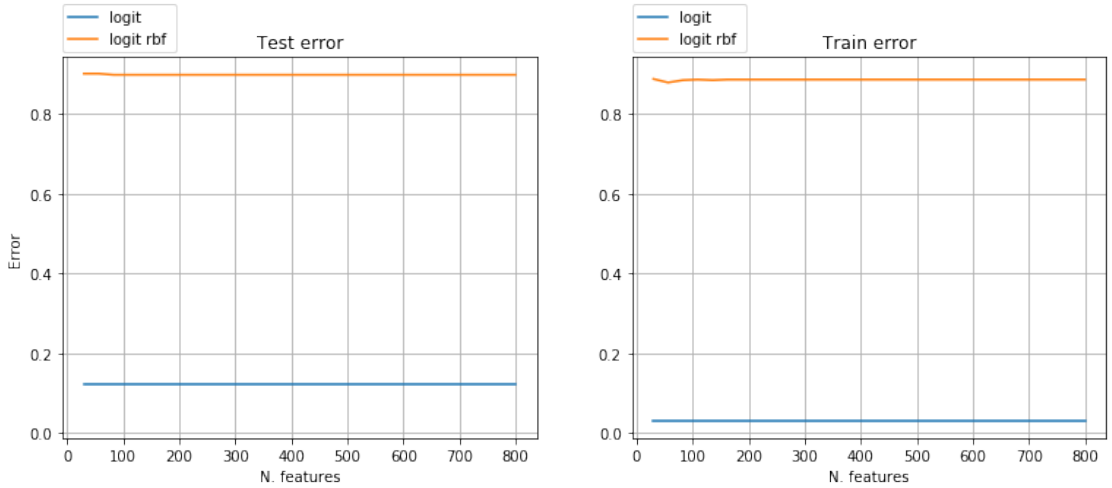
Logit

```
In [19]: data = {
             'dts_name': testing_dataset,
             'dts_size': 1000,
             'features_range': feature_range,
             'rbfsampler_gamma': 0.2,
             'nystroem_gamma': 0.2,
             'hparams': {
                 'dt': dt_best_params,
                 'logit': logit_best_params,
                 'linearsvc': linear_svc_best_params,
             },
             'models': [
                 {'model_name': 'logit',
                  'sampler_name': 'identity',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 },
                 {'model_name': 'logit',
                  'sampler_name': 'rbf',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 }
             ]
         }
```

```
In [20]: d.non_interactive(**data)
```

7 Demo genérica

- Dataset: **mnist**
- Size: **1000**



LinearSVC

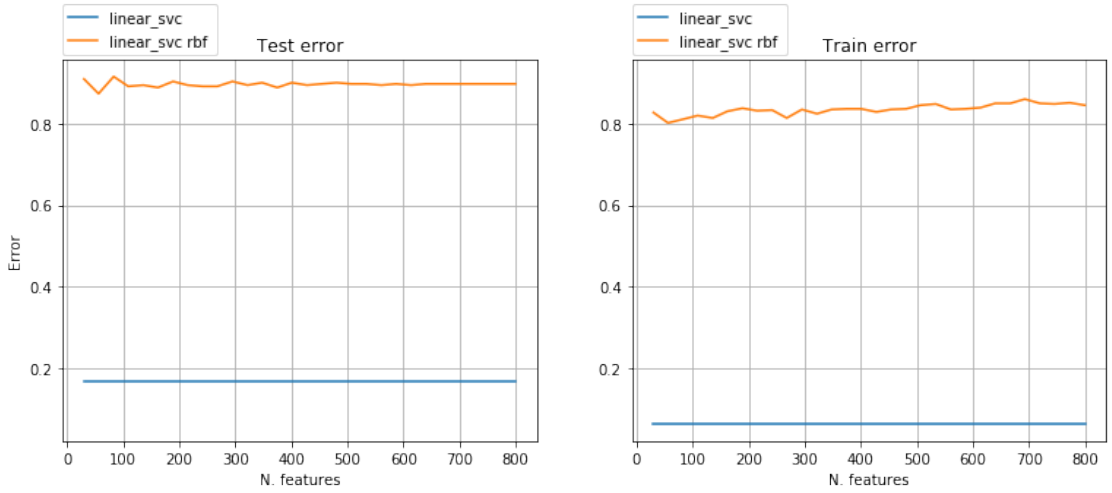
```
In [21]: data = {
             'dts_name': testing_dataset,
             'dts_size': 1000,
             'features_range': feature_range,
             'rbfsampler_gamma': 0.2,
             'nystroem_gamma': 0.2,
             'hparams': {
                 'dt': dt_best_params,
                 'logit': logit_best_params,
                 'linearsvc': linear_svc_best_params,
             },
             'models': [
                 {'model_name': 'linear_svc',
                  'sampler_name': 'identity',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 },
                 {'model_name': 'linear_svc',
                  'sampler_name': 'rbf',
```

```
                'box_type': 'none',
                'n_estim': None,
                'pca': False,
                'pca_first': False
               }
           ]
       }
```

```
In [22]: d.non_interactive(**data)
```

8 Demo genérica

- Dataset: **mnist**
- Size: **1000**



Los resultados son desastrosos. Muy malos, y no mejoran con la cantidad de features
A lo mejor el problema está en la gamma que hemos usado por defecto.
Vamos a usar diversas gamas

```
In [23]: d3 = Demo3()
```

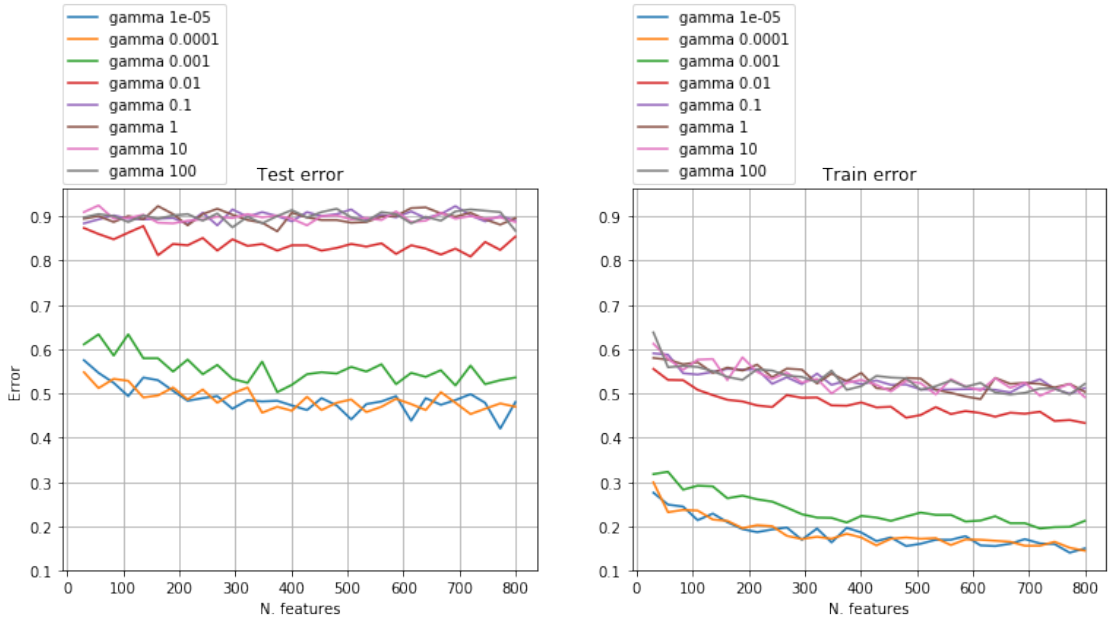
```
In [24]: data = {
             'dts_name': testing_dataset,
             'model_data':
                 {'model_name': 'dt',
                  'sampler_name': 'rbf',
                  'pca_bool': False,
                  'n_estim': None,
                  'box_type': 'none'
                 },
             'hparams': {
```

```
            'dt': dt_best_params,
            'logit': logit_best_params,
            'linearsvc': linear_svc_best_params,
        },
        'features_range': (30, 800)
    }
```

```
In [25]: d3.non_interactive(**data)
```

9 Diferencias entre los valores de gamma

- Model: **dt**
- Sampler: **rbf**
- Bagging: **none**
- N. estim.: **None**
- PCA: **False**



Destaca **gamma** = 0.00001
Todavía es un resultado muy malo, pero ya es mejor que aleatorio

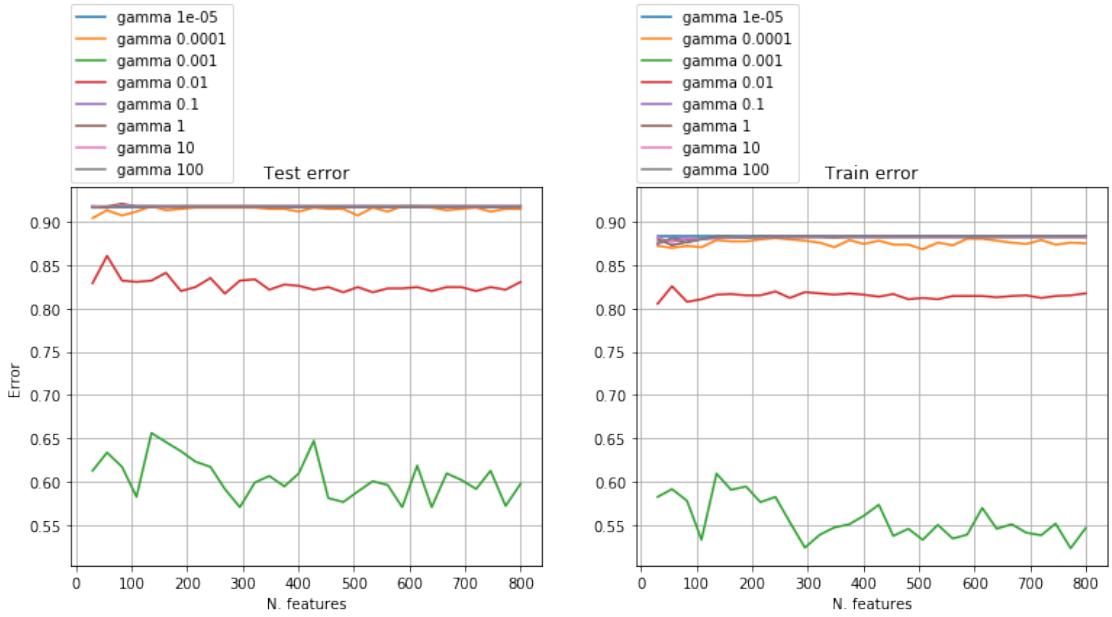
```
In [26]: data = {
            'dts_name': testing_dataset,
            'model_data':
                {'model_name': 'logit',
                 'sampler_name': 'rbf',
                 'pca_bool': False,
                 'n_estim': None,
```

```
            'box_type': 'none'
        },
    'hparams': {
        'dt': dt_best_params,
        'logit': logit_best_params,
        'linearsvc': linear_svc_best_params,
    },
    'features_range': (30, 800)
}
```

```
In [27]: d3.non_interactive(**data)
```

10 Diferencias entre los valores de gamma

- Model: **logit**
- Sampler: **rbf**
- Bagging: **none**
- N. estim.: **None**
- PCA: **False**



Destaca **gamma** = 0.001
Todavía es un resultado muy malo, pero ya es mejor que aleatorio

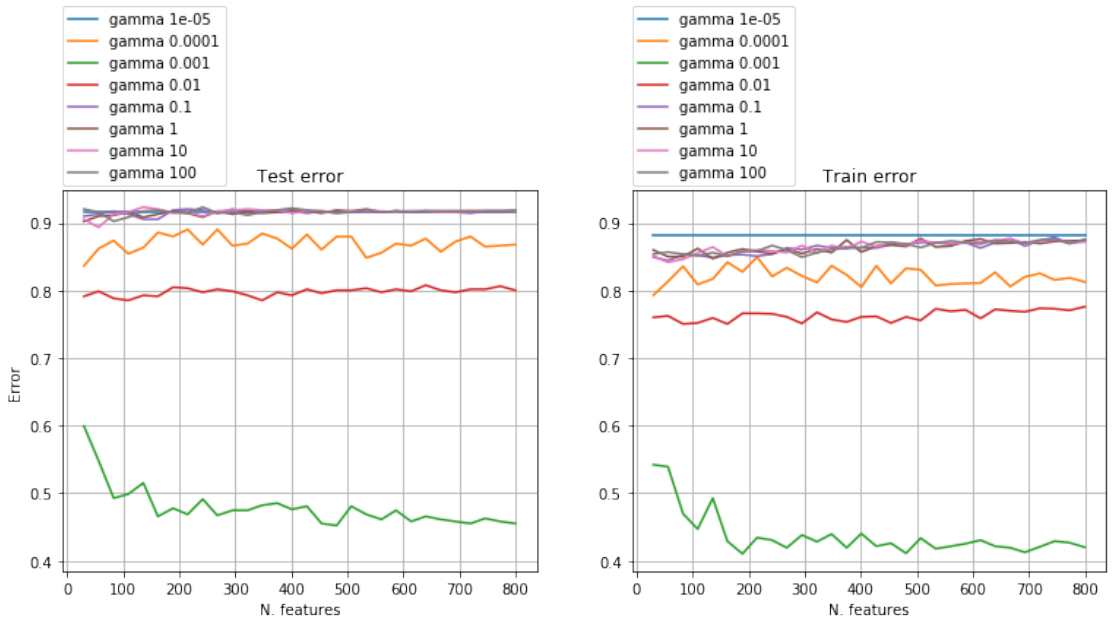
```
In [28]: data = {
            'dts_name': testing_dataset,
            'model_data':
                {'model_name': 'linear_svc',
                 'sampler_name': 'rbf',
```

```
                'pca_bool': False,
                'n_estim': None,
                'box_type': 'none'
            },
        'hparams': {
            'dt': dt_best_params,
            'logit': logit_best_params,
            'linearsvc': linear_svc_best_params,
        },
        'features_range': (30, 800)
    }
```

In [29]: `d3.non_interactive(**data)`

11 Diferencias entre los valores de gamma

- Model: **linear_svc**
- Sampler: **rbf**
- Bagging: **none**
- N. estim.: **None**
- PCA: **False**



Destaca **gamma** = 0.001

Todavía es un resultado muy malo, pero ya es mejor que aleatorio

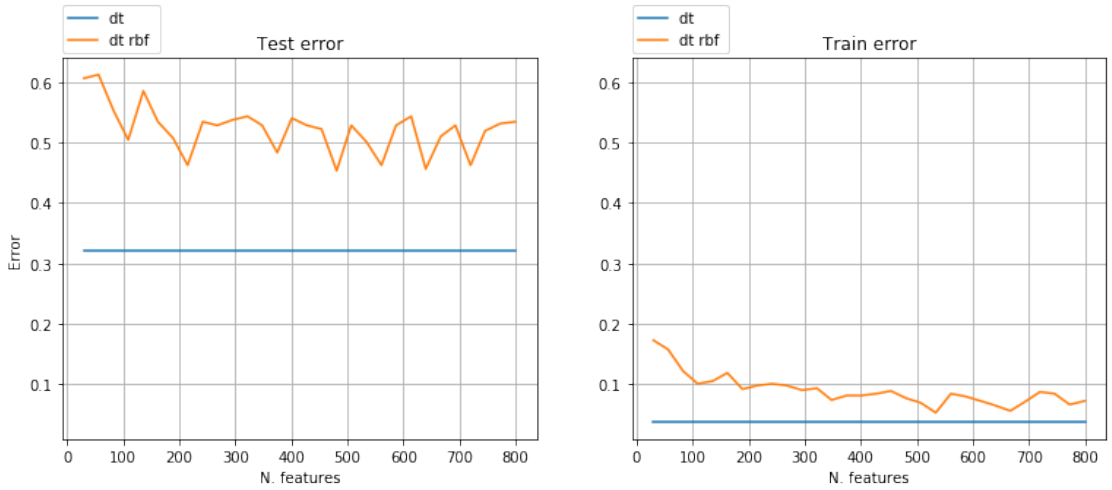
Parece que destacan algunos valores de gamma. £Qué tal si usamos esos valores en las ejecuciones anteriores?

```
In [30]: data = {
             'dts_name': testing_dataset,
             'dts_size': 1000,
             'features_range': feature_range,
             'rbfsampler_gamma': 0.00001,
             'nystroem_gamma': 0.2,
             'hparams': {
                 'dt': dt_best_params,
                 'logit': logit_best_params,
                 'linearsvc': linear_svc_best_params,
             },
             'models': [
                 {'model_name': 'dt',
                  'sampler_name': 'identity',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 },
                 {'model_name': 'dt',
                  'sampler_name': 'rbf',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 }
             ]
         }

In [31]: d.non_interactive(**data)
```

12 Demo genérica

- Dataset: **mnist**
- Size: **1000**



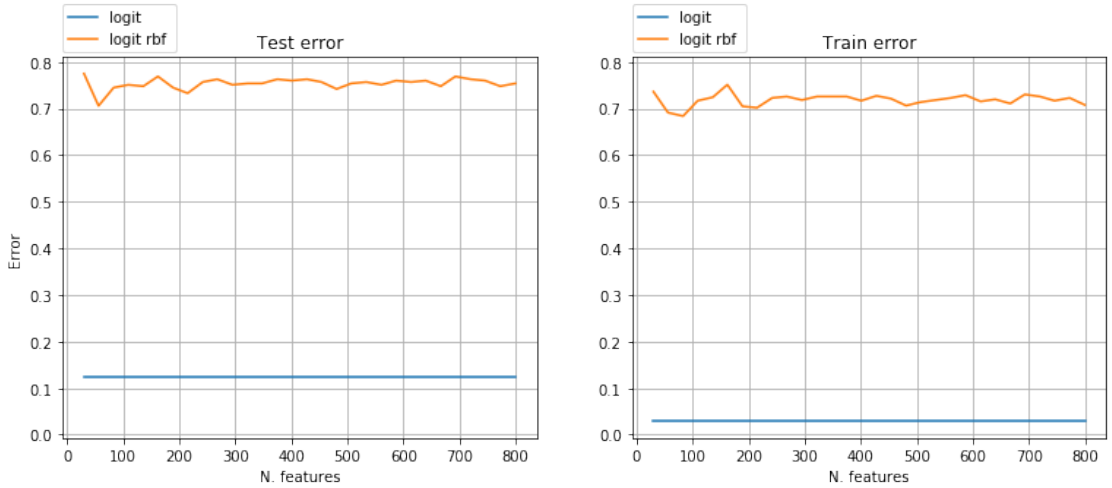
Lo mismo que hemos visto antes. Parece que aumentando la cantidad de features podría mejorar

```
In [32]: data = {
             'dts_name': testing_dataset,
             'dts_size': 1000,
             'features_range': feature_range,
             'rbfsampler_gamma': 0.001,
             'nystroem_gamma': 0.2,
             'hparams': {
                 'dt': dt_best_params,
                 'logit': logit_best_params,
                 'linearsvc': linear_svc_best_params,
             },
             'models': [
                 {'model_name': 'logit',
                  'sampler_name': 'identity',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 },
                 {'model_name': 'logit',
                  'sampler_name': 'rbf',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 }
             ]
         }
```

```
In [33]: d.non_interactive(**data)
```

13 Demo genérica

- Dataset: **mnist**
- Size: **1000**



No tiene nada que ver con lo que hemos visto antes. £Cómo es posible?

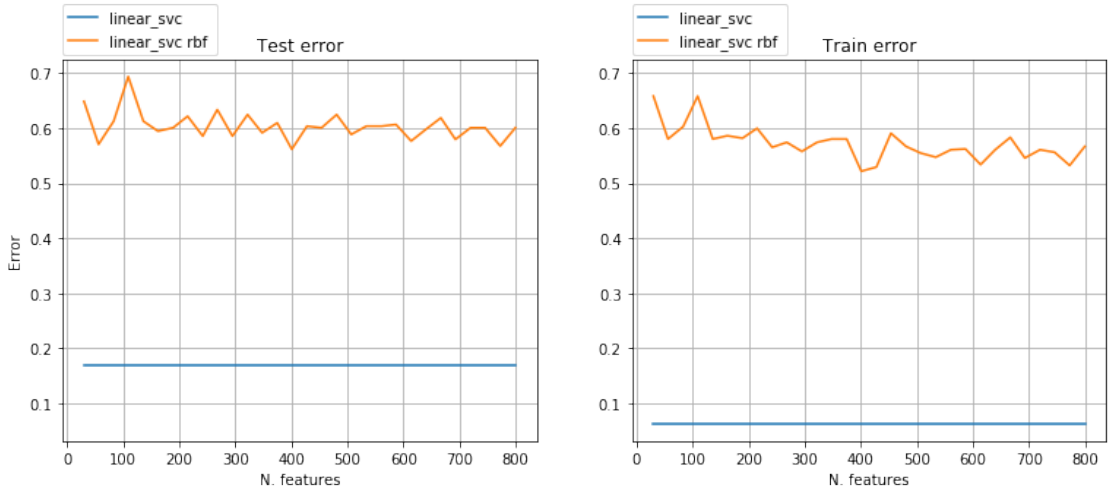
```
In [34]: data = {
             'dts_name': testing_dataset,
             'dts_size': 1000,
             'features_range': feature_range,
             'rbfsampler_gamma': 0.001,
             'nystroem_gamma': 0.2,
             'hparams': {
                 'dt': dt_best_params,
                 'logit': logit_best_params,
                 'linearsvc': linear_svc_best_params,
             },
             'models': [
                 {'model_name': 'linear_svc',
                  'sampler_name': 'identity',
                  'box_type': 'none',
                  'n_estim': None,
                  'pca': False,
                  'pca_first': False
                 },
                 {'model_name': 'linear_svc',
                  'sampler_name': 'rbf',
                  'box_type': 'none',
```

```
                'n_estim': None,
                'pca': False,
                'pca_first': False
               }
        ]
    }
```

```
In [35]: d.non_interactive(**data)
```

14 Demo genérica

- Dataset: **mnist**
- Size: **1000**



Creo que no terminan de cuadrar las dos gráficas