logistic_regression

January 30, 2019

In [1]:
```
import numpy
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split
X = numpy.loadtxt("./data/Train/X_train.txt")
y = numpy.loadtxt("./data/Train/y_train.txt")
```

In [3]:
```
X, y = shuffle(X, y)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
```

In [3]:
```
from models import logistic_regression
from sklearn.metrics import accuracy_score
ls = [5, 10, 50, 100, 561]
report = {}
```

In [4]:
```
from sklearn.feature_selection import SelectKBest, mutual_info_classif
report['mutual_info_classif'] = []
for l in ls:
    transformer = SelectKBest(mutual_info_classif, k=l)
    X_train_new = transformer.fit_transform(X_train, y_train)
    logistic_regression_model = logistic_regression.without_penalty(X_train_new, y_train
    y_pred = logistic_regression_model.predict(transformer.transform(X_test))
    score = accuracy_score(y_test, y_pred)
    data={
        'l': l,
        'score': score
    }
    report['mutual_info_classif'].append(data)
```

```
/home/mahdi/.local/share/virtualenvs/machine-learning-final-project-uk2p9d2v/lib/python3.6/site-
  "number of iterations.", ConvergenceWarning)
/home/mahdi/.local/share/virtualenvs/machine-learning-final-project-uk2p9d2v/lib/python3.6/site-
  "number of iterations.", ConvergenceWarning)
/home/mahdi/.local/share/virtualenvs/machine-learning-final-project-uk2p9d2v/lib/python3.6/site-
  "number of iterations.", ConvergenceWarning)
```

In [6]:
```
report['mutual_info_classif']
```

```
Out[6]: [{'l': 5, 'score': 0.592020592020592},
         {'l': 10, 'score': 0.6377091377091377},
         {'l': 50, 'score': 0.7451737451737451},
         {'l': 100, 'score': 0.9343629343629344},
         {'l': 561, 'score': 0.9646074646074646}]
```

```
In [6]: from sklearn.feature_selection import SelectFromModel
        clf = logistic_regression.penalty_l1(X_train, y_train, 0.01)
        report['sfm_lrl1_0.01'] = []
        for l in ls:
            transformer = SelectFromModel(clf, prefit=True, max_features=l)
            X_train_new = transformer.transform(X_train)
            logistic_regression_model = logistic_regression.without_penalty(X_train_new, y_train
            y_pred = logistic_regression_model.predict(transformer.transform(X_test))
            score = accuracy_score(y_test, y_pred)
            data={
                'l': l,
                'score': score
            }
            report['sfm_lrl1_0.01'].append(data)
```

```
/home/mahdi/.local/share/virtualenvs/machine-learning-final-project-uk2p9d2v/lib/python3.6/site-
  "the coef_ did not converge", ConvergenceWarning)
/home/mahdi/.local/share/virtualenvs/machine-learning-final-project-uk2p9d2v/lib/python3.6/site-
  "number of iterations.", ConvergenceWarning)
```

```
In [8]: report['sfm_lrl1_0.01']
```

```
Out[8]: [{'l': 5, 'score': 0.7606177606177607},
         {'l': 10, 'score': 0.859073359073359},
         {'l': 50, 'score': 0.954954954954955},
         {'l': 100, 'score': 0.965894658944659},
         {'l': 561, 'score': 0.9646074646074646}]
```

```
In [9]: clf = logistic_regression.penalty_l2(X_train, y_train, 0.01)
        report['sfm_lrl2_0.01'] = []
        for l in ls:
            transformer = SelectFromModel(clf, prefit=True, max_features=l)
            X_train_new = transformer.transform(X_train)
            logistic_regression_model = logistic_regression.without_penalty(X_train_new, y_train
            y_pred = logistic_regression_model.predict(transformer.transform(X_test))
            score = accuracy_score(y_test, y_pred)
            data={
                'l': l,
                'score': score
            }
            report['sfm_lrl2_0.01'].append(data)
```

```
In [14]: report['sfm_lrl2_0.01']

Out[14]: [{'l': 5, 'score': 0.7297297297297297},
          {'l': 10, 'score': 0.8384813384813384},
          {'l': 50, 'score': 0.9536679536679536},
          {'l': 100, 'score': 0.9678249678249679},
          {'l': 561, 'score': 0.963963963963964}]

In [15]: report

Out[15]: {'mutual_info_classif': [{'l': 5, 'score': 0.592020592020592},
           {'l': 10, 'score': 0.6377091377091377},
           {'l': 50, 'score': 0.7451737451737451},
           {'l': 100, 'score': 0.9343629343629344},
           {'l': 561, 'score': 0.9646074646074646}],
          'sfm_lrl1_0.01': [{'l': 5, 'score': 0.7606177606177607},
           {'l': 10, 'score': 0.859073359073359},
           {'l': 50, 'score': 0.954954954954955},
           {'l': 100, 'score': 0.9658944658944659},
           {'l': 561, 'score': 0.9646074646074646}],
          'sfm_lrl2_0.01': [{'l': 5, 'score': 0.7297297297297297},
           {'l': 10, 'score': 0.8384813384813384},
           {'l': 50, 'score': 0.9536679536679536},
           {'l': 100, 'score': 0.9678249678249679},
           {'l': 561, 'score': 0.963963963963964}]}
```