**scikit\_learn\_cheatsheet**

Contents

[Comparing Multiple Prediction Algorithms using a Workflow 1](#_Toc6198749)

[Grid Search to to find Optimal Hyperparameterization 1](#_Toc6198750)

[Logistic Regression precision, recall, confusion matrix, F1 score 1](#_Toc6198751)

# Comparing Multiple Prediction Algorithms using a Workflow

This is a workflow which measures the performance of multiple algorithms on a set of data, to try to guide the selection of algorithm for a problem.

<https://github.com/LahiruTjay/Machine-Learning-With-Python/blob/master/Machine%20Learning%20Workflow%20on%20Diabetes%20Data.ipynb>

# Grid Search to to find Optimal Hyperparameterization

Example at <https://github.com/savitech/data-science/blob/master/eta/svm_regression_UC2.py>

# Logistic Regression precision, recall, confusion matrix, F1 score

import numpy as np

import pandas as pd

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.linear\_model.logistic import LogisticRegression

from sklearn.cross\_validation import train\_test\_split, cross\_val\_score

from sklearn.metrics import confusion\_matrix

df = pd.read\_csv('data/sms.csv')

X\_train\_raw, X\_test\_raw, y\_train, y\_test = train\_test\_split(df['message'],

df['label'])

vectorizer = TfidfVectorizer()

X\_train = vectorizer.fit\_transform(X\_train\_raw)

X\_test = vectorizer.transform(X\_test\_raw)

classifier = LogisticRegression()

classifier.fit(X\_train, y\_train)

predictions = classifier.predict(X\_test)

# Precision and recall

precisions = cross\_val\_score(classifier, X\_train, y\_train, cv=5, scoring='precision')

print 'Precision', np.mean(precisions)

recalls = cross\_val\_score(classifier, X\_train, y\_train, cv=5, scoring='recall')

print 'Recall', np.mean(recalls)

# Confusion matrix

confusion\_matrix = confusion\_matrix(y\_test, predictions)

print confusion\_matrix

# F1 score

f1s = cross\_val\_score(classifier, X\_train, y\_train, cv=5, scoring='f1')

print "F1 score", np.mean(f1s)