12 July 2021 Report

Brad Burkman

9 July 2021

Contents

1 Activities this Week

-

2 Scikit-Learn Code

2.1 Average Precision

There's something called *average precision*, but it's just Precision with `weighted=macro`, which is the average of the precision of random samples, I think. It's definitely not the balanced precision I'm thinking of.

In `_ranking.py`, the comments define it as

```
average_precision_score : Area under the precision-recall curve.
```

2.2 Defining Your Own Metric

In `_scorer.py`:

```
sklearn.metrics.make_scorer(
    score_func,
    greater_is_better=True,
    needs_proba=False,
    needs_threshold=False, **kwargs)
```

2.3 "Support"

From imblearn.metrics:

"The support is the number of occurrences of each class in `y_true`."

2.4 Definitions of Metrics Functions

Accuracy, precision, and recall are defined in `sklearn/metrics/_classification.py`.

When you import it from `metrics`, it looks in `_classification.py`.

2.5 Adding New Metrics

How do they do it in Imbalanced-Learn?

2.6 Implementation of *Accuracy* in base.py

A stackoverflow site implied that changing this metric from *accuracy* to *recall* would change how the model works. In fact, it only changes how the model reports its score, not how it finds its prediction.

In `sklearn/base.py`, in `class ClassifierMixin`,
`"""Mixin class for all classifiers in scikit-learn."""`
Here's where we can switch the metric.
In the function
`def score(self, X, y, sample_weight=None):`

```
from .metrics import accuracy_score
return accuracy_score(y, self.predict(X), sample_weight=sample_weight)
```

I added a print statement in this function, and it appeared exactly once when I ran each classifier, so it's only calling that for the final report. I want to find where it calculates the loss function in each iteration of the model.

2.7 Implementation of Penalty in RandomForestClassifier

In `metrics`, `__init__.py` imports `RandomForestClassifier` from `_forest.py`

In `_forest.py`, the class `class RandomForestClassifier(ForestClassifier)`

The actual splits in the tree are done by DecisionTreeClassifier(), which is in `sklearn/tree/_classes.py`.

There is a `class_weights` parameter in DecisionTreeClassifier(). I need to see some examples to figure out what it does. I tried, but didn't get anything interesting.

The important parameter may be `criterion`, which has two options, gini and entropy.

```
criterion : {"gini", "entropy"}, default="gini"
The function to measure the quality of a split.
Supported criteria are "gini" for the Gini impurity
        and "entropy" for the information gain.
```

3 Implementing Different Metrics in Perceptron

`sklearn -> linear_model -> __init__.py -> _perceptron.py`

Perceptron is just a particular implementation of `BaseSGDClassifier`.

`sklearn -> linear_model -> stochastic_gradient.py -> class BaseSGDClassifier`

The loss function for Perceptron is called Hinge with argument `(Hinge, 0,0)`. Hinge is defined in a cpython function that is already compiled, `_sgd_fast.cpython-38-darwin.so`, so I can't see how it works.

4 Class_Weight

Made these changes in `Crash_Data_06_10_2021_Attempt.ipynb`.

There's a file `test_class_weight.py` that illustrates what `class_weights` does.

Many models have a `class_weight` parameter, some don't.

4.1 Models and `class_weight = "balanced"` Parameter

In the table below,

- `cw` tells whether the model has a `class_weight` parameter.
- PB tells whether using the `class_weight` parameter gives a significant performance boost.
- bf1 is the balanced f1 score.
- Two bf1 scores indicates without → with `class_weight = "balanced"`
- MLPClassifier gets this good result with these parameters:
  `MLPClassifier(alpha=1e-05, hidden_layer_sizes=(5, 2), random_state=1, solver=’lbfgs’)`

Type	cw	Model	PB	bf1	Comments
Ensemble	No	AdaBoostClassifier		37%	
	No	BaggingClassifier		48%	
	Yes	ExtraTreesClassifier	Yes	5 → 15%	
	No	GradientBoostingClassifier		51%	
	Yes	RandomForestClassifier	Yes	nan → 8%	
		StackingClassifier			Stacks several classifiers together. Not its own classifier.
		VotingClassifier			Same
Linear	Yes	LogisticRegression	Yes	47 → 89%	
	Yes	Perceptron	Yes	80 → 88%	
	Yes	RidgeClassifier	YES	nan → 89%	
	Yes	RidgeClassifierCV	YES	nan → 89%	
	Yes	SGDClassifier	YES	37 → 90%	
Naive Bayes	No	GaussianNB		66%	
Neighbors	No	KNeighborsClassifier		8%	
	No	KNeighborsClassifier(n_neighbors=3)		6%	
	No	RadiusNeighborsClassifier			Error: No neighbors found within radius. Perhaps not applicable for binary?
Neural Network	No	MLPClassifier		63%	
SVM	Yes	LinearSVC	YES	34 → 86%	
	Yes	NuSVC			"Specified nu is infeasible."
	Yes	SVC	Yes	nan → 72%	
Tree	Yes	DecisionTreeClassifier	NO	57 → 48%	
	Yes	ExtraTreeClassifier	NO	31 → 27%	