img source : http://epistasislab.github.io/tpot/

Data Project - Finding The Best Employee: Using TPOT

Edit article
View stats

Ranggasena Tranggana
Digital Business Development | Data Enthusiast
2 articles

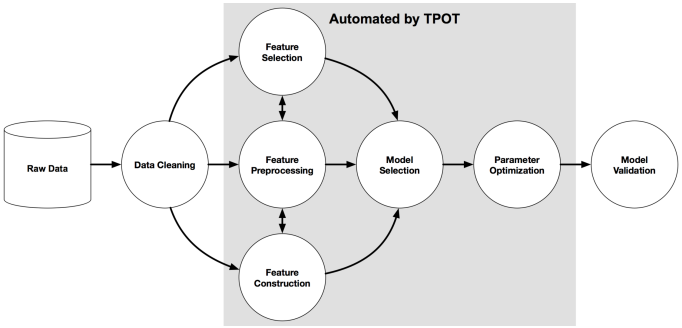
January 25, 2023

People say curiosity is a mind in search of knowledge, and I find myself always curious. The last time I wrote an article about my previous project in Data Science Bootcamp, and I am going to use others library to predict label and compare it with previous project (see in this link).

TPOT, yes it is. My curiosity about Automated Machine Learning (AutoML) is getting bigger, so I searched everything about AutoML, and finally, I try to use TPOT as my assistant (they say it on their website).

For those who don't understand AutoML libraries are software tools that automate the process of building machine learning models, from data pre-processing to model selection and tuning. These libraries can save data scientists and engineers time and effort when developing and deploying machine learning models by automating many of the tedious and time-consuming tasks involved. The one of popular AutoML libraries is TPOT. TPOT is a

Python Automated Machine Learning tool that optimizes machine learning pipelines using genetic programming.



An example machine learning pipeline

As the picture above, TPOT can handle after we clean the data. Despite TPOT can handle feature selection, I am going to use X_train data that the features have been selected.

Using TPOT

```
from tpot import TPOTClassifier

pipeline_optimizer = TPOTClassifier()

pipeline_optimizer = TPOTClassifier(generations=5, population_size=20, cv=5,
                                    random_state=42, verbosity=2)

pipeline_optimizer.fit(X_train, y_train)
```

Using TPOT Code

Of course you should install TPOT first, click in this **link**. After that we import TPOTClassifier, we use 5 generations which means 5 iterations and population_size 20 which means TPOT will evaluate 100 pipeline configurations before finishing. Literally the more you put number in generation / population_size the more chance model will be good but it will take a long time to learn.

Let's Predict Data 2022

```
X_new = data_2022.drop(columns=['is_promoted'])
```

```
y_proba = pipeline_optimizer.predict(X_new)
```

```
y_proba
```

```
array([0, 0, 0, ..., 0, 0, 0])
```

We predict data X_new that from data_2022 features. For those who don't know where's from the data you can see my previous project through this **link**.

```
data_2022['tpot'] = y_proba
```

```
data_2022.sample(5)
```

	previous_year_rating	awards_won	avg_training_score	is_promoted	tpot
9976	-0.264757	-0.150860	-0.790909	0	0
12737	-0.264757	-0.150860	-0.264285	0	0
13094	-0.264757	6.628672	-0.189053	0	0
9456	1.389284	-0.150860	0.713730	0	0
2254	-0.264757	-0.150860	0.187107	0	0

The result from predict we added to data_2022 in 'tpot' column. We will compare the result from previous project which was we used XGBClassifier without SMOTE in 'is_promoted' column.

```
data_2022.is_promoted.value_counts()
```

```
0    13835
1      231
Name: is_promoted, dtype: int64
```

```
round(231/data_2022.is_promoted.count()*100,2)
```

```
1.64
```

Prediction using XGBoost in previous project

Using XGBoost we got 1.64% from employees (data_2022) is considered got promoted.

```
data_2022.tpot.value_counts()
```

```
0    13826
1      240
Name: tpot, dtype: int64
```

```
round(240/data_2022.tpot.count()*100,2)
```

```
1.71
```

Prediction using TPOT

When we used TPOT, we got 1.71% from employees (data_2022) is considered got promoted. That means TPOT pipeline predicted 1 more than manual 'XGBoost' and has difference only 9 employees, but we will check who is predicted get promoted from 'XGBoost' but don't get it when using TPOT and vice versa.

```
data_2022[((data_2022.is_promoted == 1) &
           (data_2022.tpot == 0))]['is_promoted'].count()
```

```
13
```

```
data_2022[((data_2022.is_promoted == 0) &
           (data_2022.tpot == 1))]['tpot'].count()
```

```
22
```

The result is TPOT is more predict '1' that predicted '0' from manual XGBoost. But if we compared 1.64% vs 1.71% is only 0.07% different. It's not late to mention that the accuracy test in TPOT with data test (data_2022) is 91%.

Conclusion

TPOT will help you build your model without hesitation without having to try one algorithm and then hyperparameters for which you have to read the documentation first. But people says there's no free lunch, which is you need more resources to run TPOT because TPOT needs to try many pipelines to find the best one for you.

Thank you.
