Tugas Matematika 3

Random Search

Mata Kuliah:
Matematika 3

Oleh:

Robi'atul Adawiyah	**3323600041**
Najiyah Al Mujahidah	**3323600044**
Azalia Fitriana Bagardini	**3323600051**
Adriyans Jusa Hutapea	**3323600052**
Moch. Ariel Sulton	**3323600054**
Wahyu Ikbal Maulana	**3323600056**
Jogi Fergio Schumacher	**3323600060**

Program Studi D4 Sains Data Terapan
Departemen Teknik Informatika dan Komputer
Politeknik Elektronika Negeri Surabaya

2024

A. Pengertian Random Search

Random search adalah teknik optimisasi yang menggunakan pendekatan acak untuk menemukan solusi optimal untuk suatu masalah. Teknik ini melibatkan pemilihan acak dari titik-titik dalam ruang pencarian, dan kemudian evaluasi dari fungsi tujuan pada titik-titik tersebut. Solusi terbaik yang ditemukan selama pencarian acak menjadi solusi optimal.

B. Kelebihan Random Search

- Mudah diimplementasikan
- Tidak memerlukan pengetahuan sebelumnya tentang ruang pencarian
- Berlaku untuk berbagai jenis masalah
- Efisien untuk ruang pencarian yang kompleks
- Sangat baik dalam menghindari minimum lokal

C. Kelemahan Random Search

- Tidak menjamin menemukan solusi optimal global
- Performa bergantung pada ukuran ruang pencarian dan distribusi titik-titik acak
- Dapat memakan waktu jika ruang pencarian sangat besar
- Tidak efektif untuk masalah dengan banyak minimum lokal

D. Contoh Soal Random Search

Misalnya, Anda ingin menemukan nilai maksimum dari fungsi f(x) = -x^2 + 4x - 3 di interval [0, 3]. Untuk menggunakan random search, Anda akan memilih secara acak nilai-nilai x dalam interval [0, 3] dan mengevaluasi f(x) pada setiap nilai yang dipilih. Nilai x yang menghasilkan f(x) terbesar adalah solusi optimal.

E. Penyelesaian Manual Contoh Soal Random Search

Berikut langkah-langkah untuk menyelesaikan contoh soal random search secara manual:

1. Pilih nilai x secara acak dalam interval [0, 3]. Misalnya, x1 = 1, x2 = 2, x3 = 0.5, x4 = 2.5.
2. Hitung nilai f(x) untuk setiap nilai x yang dipilih. f(x1) = 0, f(x2) = 1, f(x3) = -1.25, f(x4) = 1.25.
3. Nilai x yang menghasilkan f(x) terbesar adalah solusi optimal. Dalam contoh ini, x2 = 2 adalah solusi optimal karena f(x2) = 1 adalah nilai f(x) terbesar.

F. Penyelesaian Menggunakan Program Python Hyperparameter sklearn random search

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

wine_df = pd.read_csv("red-wine-quality-cortez-et-al-2009/winequality-red.csv")
from sklearn.model_selection import train_test_split

X = wine_df.drop(['quality', 'good_wine'], axis='columns')
y = wine_df['good_wine']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
random_state=42)
X_train.shape, X_test.shape, y_train.shape, y_test.shape

from lightgbm import LGBMClassifier
model = LGBMClassifier(random_state=0)
model.fit(X_train, y_train)
from sklearn.metrics import roc_auc_score

y_predictions = model.predict_proba(X_test)[:,1]
roc_auc_score(y_test, y_predictions)
```

```
0.9350225653852429
```

```
from sklearn.model_selection import RandomizedSearchCV
from sklearn.utils.fixes import loguniform

parameters_distributions = {'learning_rate': loguniform(1e-3, 1e-1),
                            'num_leaves': list(range(1, 50, 5)),
                            'min_child_samples': list(range(1, 50, 5)),
                            'subsample': [0.05, 1.0],
                            'colsample_bytree': [0.1, 1.0]}

random_search = RandomizedSearchCV(model, parameters_distributions,
random_state=0, n_iter=30, n_jobs=-1)
random_search.fit(X_train, y_train)
```

```
RandomizedSearchCV(cv=None, error_score=nan,
                   estimator=LGBMClassifier(boosting_type='gbdt',
                                            class_weight=None,
                                            colsample_bytree=1.0,
                                            importance_type='split',
                                            learning_rate=0.1, max_depth=-1,
                                            min_child_samples=20,
                                            min_child_weight=0.001,
                                            min_split_gain=0.0,
                                            n_estimators=100, n_jobs=-1,
                                            num_leaves=31, objective=None,
                                            random_state=0, reg_alpha=0.0,
                                            reg_lambda=0.0, sile...
                   iid='deprecated', n_iter=30, n_jobs=-1,
                   param_distributions={'colsample_bytree': [0.1, 1.0],
```

```
                                    'learning_rate': <scipy.stats._distn_infrastructure.rv_frozen object at
0x7fcf74240278>,
                                    'min_child_samples': [1, 6, 11, 16, 21,
                                                          26, 31, 36, 41,
                                                          46],
                                    'num_leaves': [1, 6, 11, 16, 21, 26, 31,
                                                   36, 41, 46],
                                    'subsample': [0.05, 1.0]},
                   pre_dispatch='2*n_jobs', random_state=0, refit=True,
                   return_train_score=False, scoring=None, verbose=0)
```

```
best_model_rs = random_search.best_estimator_

y_predictions = best_model_rs.predict_proba(X_test)[:,1]
roc_auc_score(y_test, y_predictions)
```

```
0.9401881355010603
```