**Menyetel hiperparameter dengan Azure Databricks**

Di Azure Databricks, Anda dapat mengotomatisasi proses penyetelan hiperparameter untuk mengidentifikasi model terbaik dengan lebih mudah.

**Tujuan pembelajaran**

Setelah menyelesaikan modul ini, Anda akan dapat:

* + Memahami penyetelan hiperparameter dan perannya dalam pembelajaran mesin.
  + Pelajari cara menggunakan dua alat sumber terbuka - MLflow otomatis dan Hyperopt - untuk mengotomatisasi proses pemilihan model dan penyetelan hiperparameter.

1. **Pengantar**

Hiperparameter adalah parameter yang ditentukan sebelum pelatihan model yang dapat memengaruhi performa model. Ada berbagai hiperparameter yang tersedia untuk disempurnakan, tergantung pada algoritma yang digunakan untuk melatih model, yang dapat dilakukan melalui proses yang disebut **penyetelan hiperparameter.**

Dalam modul ini, Anda akan mempelajari cara menggunakan Azure Databricks dengan MLflow untuk melakukan penyetelan hiperparameter dan pemilihan model.

1. **Tujuan pembelajaran**

Setelah menyelesaikan modul ini, Anda akan mampu:

* Memahami penyetelan hiperparameter dan perannya dalam pembelajaran mesin.
* Pelajari cara menggunakan dua alat sumber terbuka - MLflow otomatis dan Hyperopt - untuk mengotomatisasi proses pemilihan model dan penyetelan hiperparameter.

1. **Memahami penyetelan hiperparameter**

Membangun solusi pembelajaran mesin melibatkan pengujian berbagai model yang berbeda. Mari kita jelajahi dua konsep yang dapat membantu kita menemukan model optimal:

* Penyetelan hiperparameter
* Validasi silang

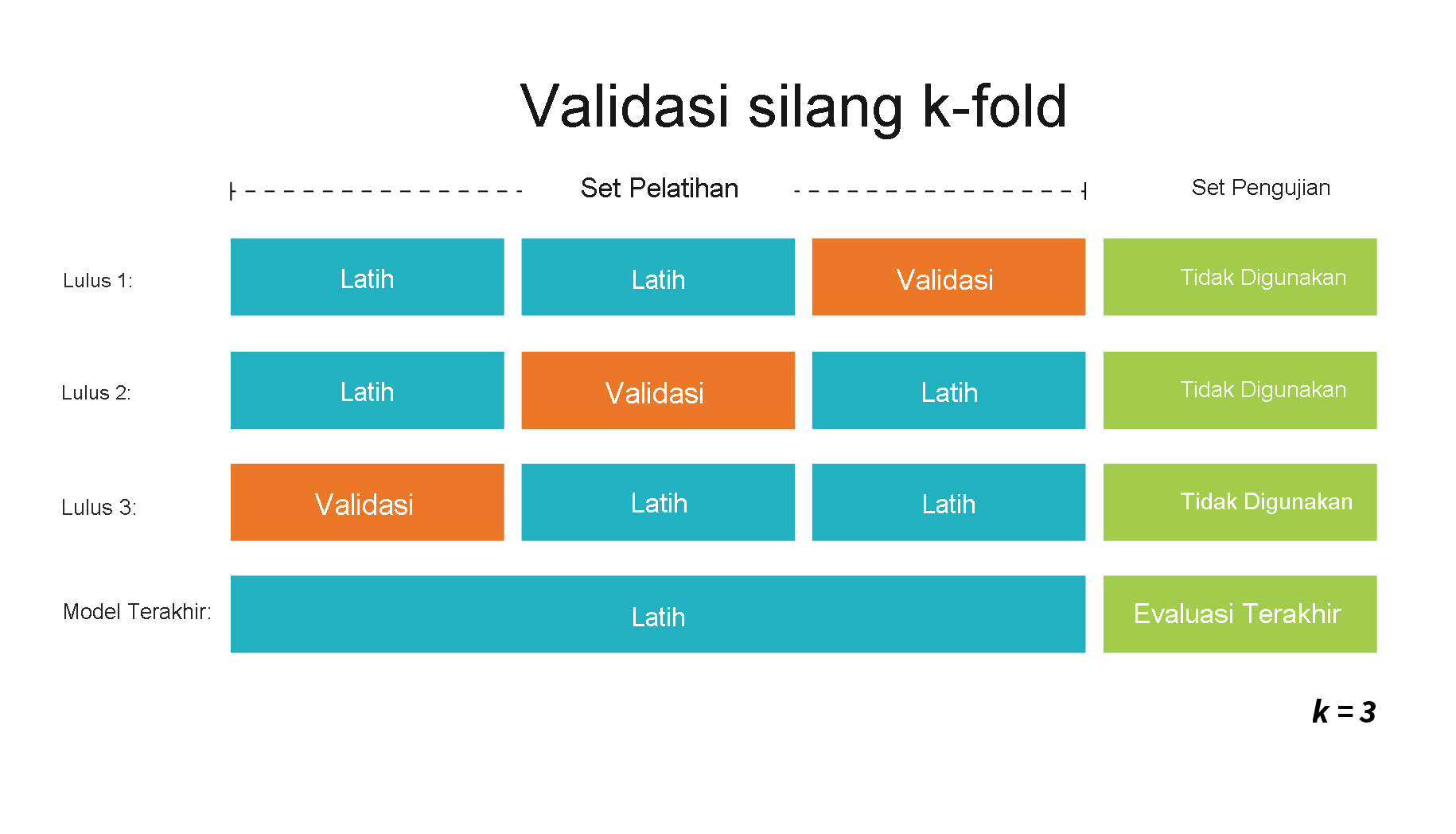
1. **Penyetelan hiperparameter**

**Hiperparameter** adalah parameter yang digunakan dalam algoritma pembelajaran mesin yang ditetapkan sebelum proses pembelajaran dimulai. Dengan kata lain, algoritma pembelajaran mesin tidak dapat mempelajari hiperparameter dari data itu sendiri. Hiperparameter diuji dan divalidasi dengan melatih beberapa model. Hiperparameter umum meliputi jumlah perulangan dan kompleksitas model. Penyetelan **hiperparameter** adalah proses memilih hiperparameter yang memiliki hasil terbaik pada fungsi kerugian kita, atau cara kita menghukum algoritma karena salah.

1. **Validasi silang**

Ketika Anda melatih dan mengevaluasi model pada data yang sama, hal ini dapat menyebabkan **overfitting**. Overfitting adalah saat model memiliki performa yang baik pada data yang telah dilihatnya tetapi gagal memprediksi apa pun yang berguna pada data yang belum dilihatnya. Untuk menghindari overfitting, Anda dapat menggunakan pemisahan pelatihan/pengujian di mana himpunan data dibagi antara set pelatihan yang digunakan untuk melatih model dan satu set pengujian untuk mengevaluasi performa model pada data yang tidak terlihat.

Jika Anda melatih berbagai model yang berbeda dengan hiperparameter yang berbeda, kemudian mengevaluasi performa mereka pada set pengujian, Anda masih akan berisiko mengalami overfitting karena Anda dapat memilih hiperparameter yang kebetulan memiliki performa terbaik pada data yang Anda miliki di himpunan data Anda. Untuk mengatasi overfitting saat menggunakan hyperparameters, Anda dapat menggunakan subset k dari kumpulan pelatihan Anda untuk melatih model, proses yang disebut **k-fold cross-validation**. Sebuah model kemudian dilatih pada lipatan k-1 dari data pelatihan dan lipatan terakhir digunakan untuk mengevaluasi performanya.



Dalam Azure Databricks, terdapat dua pendekatan untuk menyetel hiperparameter, yang akan dibahas di unit berikutnya:

* Pelacakan MLflow otomatis.
* Penyetelan hiperparameter dengan Hyperopt.

1. **MLflow otomatis untuk penyetelan model**

Untuk memilih model terbaik yang dilatih selama penyetelan hiperparameter, Anda ingin membandingkan semua model dengan mengevaluasi metrik mereka. Salah satu pendekatan umum dan sederhana untuk melacak pelatihan model di Azure Databricks adalah menggunakan platform sumber terbuka **MLflow**.

1. **Menggunakan MLflow otomatis**

Saat Anda melatih beberapa model dengan penyetelan hiperparameter, Anda ingin menghindari kebutuhan untuk melakukan panggilan API eksplisit untuk mencatat semua informasi yang diperlukan tentang berbagai model tersebut ke MLflow. Untuk membuat pelacakan penyetelan hiperparameter lebih mudah, **Databricks Runtime for Machine Learning (** <https://docs.databricks.com/runtime/mlruntime.html> **)** juga mendukung Pelacakan MLflow otomatis. Ketika Anda menggunakan MLflow otomatis untuk penyetelan model, nilai hiperparameter dan metrik evaluasi secara otomatis dicatat dalam MLflow dan hierarki akan dibuat untuk berbagai eksekusi yang mewakili berbagai model yang Anda latih.

Untuk menggunakan pelacakan MLflow otomatis, Anda harus melakukan hal berikut:

* Gunakan buku catatan Python untuk meng-hosting kode Anda.
* Lampirkan buku catatan ke kluster dengan Databricks Runtime atau Databricks Runtime for Machine Learning.
* Siapkan penyetelan hiperparameter dengan **CrossValidator** atau **TrainValidationSplit**.

MLflow akan secara otomatis membuat eksekusi utama atau induk yang berisi informasi untuk metode yang Anda pilih: **CrossValidator** atau **TrainValidationSplit**. MLflow juga akan membuat eksekusi anak yang bersarang di bawah eksekusi utama atau induk. Setiap anak yang dijalankan akan mewakili model terlatih dan Anda dapat melihat nilai hiperparameter mana yang digunakan serta metrik evaluasi yang dihasilkan.

1. **Menjalankan kode penyetelan**

Ketika Anda ingin menjalankan kode yang akan melatih beberapa model dengan pengaturan hiperparameter yang berbeda, Anda dapat menelusuri langkah-langkah berikut:

* Cantumkan hiperparameter yang tersedia untuk algoritma tertentu.
* Siapkan ruang pencarian dan metode pengambilan sampel.
* Jalankan kode dengan MLflow otomatis, menggunakan CrossValidator atau TrainValidationSplit.

1. **Mencantumkan hiperparameter yang tersedia**

Anda dapat menjelajahi hiperparameter dari algoritma pembelajaran mesin tertentu menggunakan metode **.explainParams()** pada model. Misalnya, jika kita ingin melatih model regresi linier **lr**, kita dapat menggunakan perintah berikut untuk melihat hiperparameter yang tersedia:



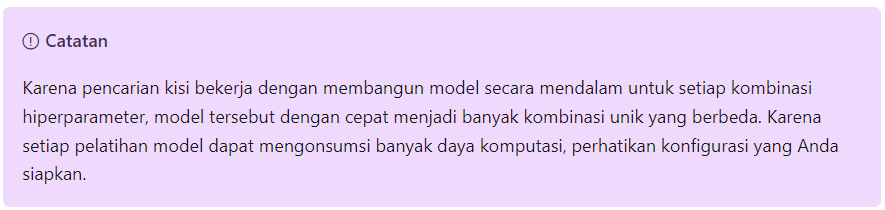
Metode **.explainParams()** akan menampilkan daftar hiperparameter yang dapat Anda pilih, termasuk nama hiperparameter, deskripsi, dan nilai default. Tiga dari hiperparamater yang tersedia untuk model regresi linier adalah:

* maxIter: jumlah maksimum perulangan (>= 0). (default: 100)
* fitIntercept: memasang syarat pencegatan atau tidak. (default: True)
* standardization: menstandarkan fitur pelatihan sebelum memasang model atau tidak. (default: True)

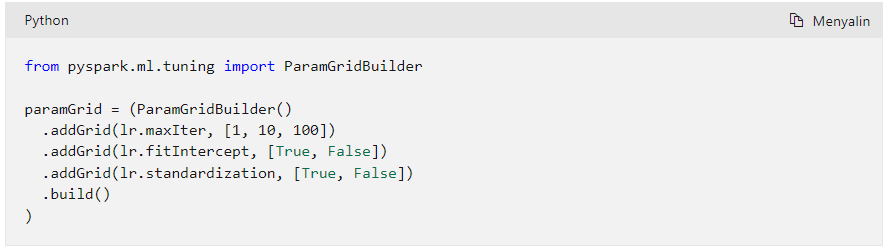
1. **Menyiapkan ruang pencarian dan metode pengambilan sampel**

Setelah memilih hiperparameter, Anda dapat menggunakan ParamGridBuilder() untuk menentukan ParamGridBuilder(). Ruang pencarian adalah rentang nilai hiperparameter yang ingin Anda coba. Anda kemudian dapat menentukan bagaimana Anda ingin memilih nilai dari ruang pencarian itu untuk melatih model individual yang dikenal sebagai **metode pengambilan sampel.** Metode pengambilan sampel yang paling terus terang dikenal sebagai **pengambilan sampel kisi**. Metode pengambilan sampel kisi mencoba semua kemungkinan kombinasi nilai untuk hiperparameter yang tercantum.

Secara default, model individu akan dilatih secara seri. Melatih model dengan nilai hiperparameter yang berbeda secara paralel dapat dilakukan. Anda dapat menemukan informasi selengkapnya tentang menyiapkan kisi parameter dalam **dokumentasi di sini. :** <https://spark.apache.org/docs/latest/ml-tuning.html>



Jika kita melanjutkan contoh dengan model regresi linier **lr**, kode berikut menunjukkan cara menyiapkan pencarian kisi untuk mencoba semua kemungkinan kombinasi parameter:

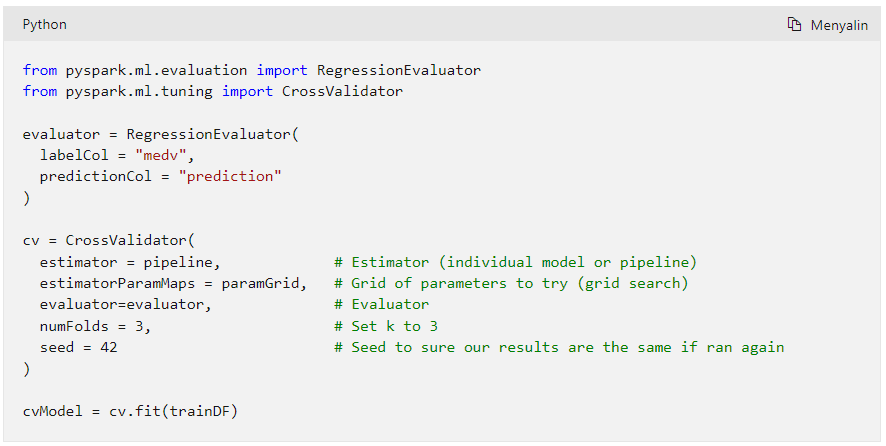


1. **Jalankan kode dan panggil MLflow otomatis**

Untuk menguji bagaimana performa model dan untuk menghasilkan metrik evaluasi, Anda dapat menggunakan himpunan data pengujian. Jika ingin melatih beberapa model pada himpunan data pelatihan yang sama dan himpunan data pengujian yang sama, Anda dapat menggunakan metode **TrainValidationSplit** untuk menjalankan kode, membangun model, dan mencatatnya secara otomatis dengan MLflow.

Jika ingin mengambil langkah-langkah ekstra untuk mencegah overfitting, Anda dapat menggunakan metode **CrossValidator** untuk melatih model dengan himpunan data pelatihan yang berbeda untuk setiap model dan himpunan data pengujian yang berbeda untuk menghitung metrik evaluasi.

Untuk nmembangun model untuk model regresi linier **lr** yang digunakan dalam contoh di atas, Anda dapat membuat **RegressionEvaluator()** untuk mengevaluasi eksperimen pencarian kisi, yang akan membantu memutuskan model mana yang terbaik. Pengaturan untuk eksperimen penyetelan hiperparameter dapat diatur menggunakan metode **CrossValidator()** seperti yang dilakukan pada contoh di bawah ini.



Setelah semua model dilatih, Anda bisa mendapatkan model terbaik dengan kode berikut:



Atau, Anda dapat melihat semua model yang Anda latih melalui UI MLflow. Harap diingat bahwa akan ada eksekusi induk untuk eksperimen yang selesai dan eksekusi anak untuk setiap model individu yang telah dilatih.

1. **Penyetelan hiperparameter dengan Hyperopt**

Alat sumber terbuka lain yang memungkinkan Anda untuk mengotomatisasi proses penyetelan hiperparameter dan pemilihan model adalah Hyperopt. Hyperopt mudah digunakan, tetapi menggunakannya secara efisien membutuhkan ketelitian. Keuntungan utama menggunakan Hyperopt adalah Hyperopt bersifat fleksibel dan dapat mengoptimalkan model Python dengan hiperparameter.

1. **Menggunakan Hyperopt**

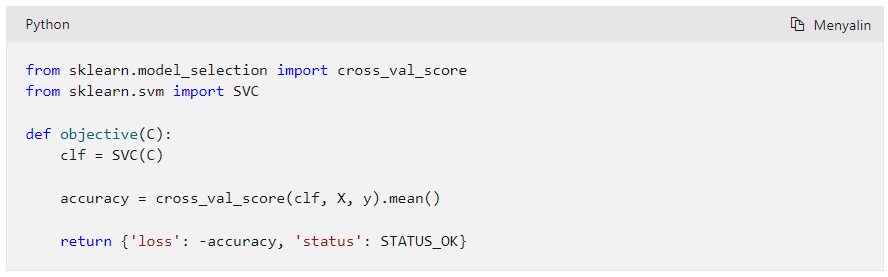
Hyperopt sudah diinstal jika Anda membuat komputasi dengan Databricks Runtime ML. Untuk menggunakannya saat melatih model Python, Anda harus mengikuti langkah-langkah dasar ini:

1. Tentukan fungsi objektif untuk diminimalkan.
2. Tentukan ruang pencarian hiperparameter.
3. Tentukan algoritma pencarian.
4. Jalankan fungsi fmin() Hyperopt.
5. **Menentukan fungsi objektif untuk diminimalkan**

Fungsi objektif mewakili tujuan utama melatih beberapa model melalui penyetelan hiperparameter. Sering kali, tujuannya adalah untuk meminimalkan pelatihan atau hilangnya validasi.

Saat menentukan fungsi, Anda dapat menggunakan metrik evaluasi apa pun yang dapat dihitung dengan algoritma yang Anda pilih. Misalnya, jika kita menggunakan **pengklasifikasi mesin vektor dukungan dari pustaka scikit-learn**, ( <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html> ) Anda dapat memvariasikan nilai untuk parameter regularisasi . Tujuannya adalah untuk memiliki model dengan akurasi tertinggi. Karena Hyperopt menginginkan fungsi yang perlu diminimalkan, Anda dapat menentukan fungsi objektif sebagai akurasi negatif sehingga skor yang lebih rendah sebenarnya berarti akurasi yang lebih tinggi.

Dalam contoh berikut, parameter regularisasi **c** didefinisikan sebagai input, model pengklasifikasi mesin vektor dukungan dilatih, akurasi dihitung, dan fungsi objektif didefinisikan sebagai akurasi negatif, yang merupakan nilai yang akan diminimalkan oleh Hyperopt.



1. **Menentukan ruang pencarian hiperparameter**

Saat menyetel hiperparameter, Anda perlu menentukan ruang pencarian. Jika Anda ingin memanfaatkan pendekatan Bayesian Hyperopt untuk pengambilan sampel, ada serangkaian ekspresi yang dapat Anda gunakan untuk menentukan ruang pencarian yang kompatibel dengan pendekatan Hyperopt untuk pengambilan sampel.

Beberapa contoh ekspresi yang digunakan untuk menentukan ruang pencarian adalah:

* **hp.choice(label, options)**: Menampilkan salah satu **options** yang Anda cantumkan.
* **hp.randint(label, upper)**: Menampilkan bilangan bulat acak dalam rentang [0, lebih tinggi].
* **hp.uniform(label, low, high**): Menampilkan nilai secara seragam antara **low** dan **high**.
* **hp.normal(label, mu, sigma)**: Menampilkan nilai riil yang biasanya didistribusikan dengan mean mu dan simpangan baku **sigma**.

Untuk daftar lengkap ekspresi, lihat **dokumentasi Hyperopt**. ( <https://github.com/hyperopt/hyperopt/wiki/FMin#21-parameter-expressions> )

1. **Memilih algoritma pencarian**

Terdapat dua pilihan utama dalam bagaimana Hyperopt akan mengambil sampel melalui ruang pencarian:

* **hyperopt.tpe.suggest**: Tree of Parzen Estimators (TPE), pendekatan Bayesian, yang secara berulang dan adaptif memilih pengaturan hiperparameter baru untuk dijelajahi berdasarkan hasil sebelumnya.
* **hyperopt.rand.suggest**: Pencarian acak, pendekatan non-adaptif yang mengambil sampel melalui ruang pencarian.

1. **Menjalankan fungsi fmin() Hyperopt**

Terakhir, untuk menjalankan eksekusi Hyperopt, Anda dapat menggunakan fungsi fmin(). Fungsi fmin() ini mengambil argumen berikut:

* **fn**: Fungsi objektif.
* **space**: Ruang pencarian.
* **algo**: Algoritma pencarian yang Anda inginkan untuk digunakan oleh Hyperopt.
* **max\_evals:** Jumlah maksimum model untuk dilatih.
* **max\_queue\_len:** Jumlah pengaturan hiperparameter yang dihasilkan sebelumnya. Hal ini dapat menghemat waktu saat menggunakan algoritma TPE.
* **trials**: Objek **SparkTrials** atau **Trials**. **SparkTrials** digunakan untuk algoritma mesin tunggal seperti scikit-learn. **Trials** digunakan untuk algoritma pelatihan terdistribusi seperti metode MLlib atau Horovod. Saat menggunakan **SparkTrials** atau Horovod, pelacakan MLflow otomatis diaktifkan dan hiperparameter dan metrik evaluasi secara otomatis dicatat dalam MLflow.

Untuk informasi selengkapnya tentang cara mengonfigurasi **fmin()** dan **SparkTrials**, baca tentang konsep Hyperopt **fmin().**

**5. Latihan**

Sekarang, ini adalah kesempatan Anda untuk menggunakan Azure Databricks untuk menyetel hiperparameter.

Dalam latihan ini, Anda akan:

* Jelajahi penyetelan hiperparameter MLflow otomatis.
* Jelajahi Hyperopt untuk penyetelan hiperparameter.

1. **Instruksi**

Ikuti petunjuk berikut untuk menyelesaikan latihan:

1. Buka petunjuk latihan di <https://aka.ms/mslearn-dp090.> \ <https://microsoftlearning.github.io/dp-090-databricks-ml/>
2. Selesaikan latihan **Penyetelan Hiperparameter dengan Azure Databricks**.

**6. Ringkasan**

Dalam modul ini, Anda mempelajari cara melakukan penyetelan hiperparameter di Azure Databricks.

Sekarang setelah Anda menyelesaikan modul ini, Anda dapat:

* Jelaskan penyetelan hiperparameter.
* Gunakan MLflow otomatis untuk penyetelan hiperparameter.
* Gunakan Hyperopt untuk penyetelan hiperparameter.