Perencanaan untuk sukses MLOps

# AWS Bimbingan Preskriptif

# AWS Bimbingan Preskriptif: Perencanaan untuk sukses MLOps

# Table of Contents

# Perencanaan untuk sukses MLOps

Bruno Klein, Amazon Web Services ()AWS

Desember 2021 (riwayat dokumen)

Menerapkan solusi pembelajaran mesin (ML) dalam produksi memperkenalkan banyak tantangan yang tidak muncul dalam proyek pengembangan perangkat lunak standar. Solusi ML lebih kompleks dan lebih sulit untuk dilakukan dengan benar sejak awal. Mereka juga ada di lingkungan yang biasanya tidak stabil, di mana distribusi data menyimpang secara signifikan dari waktu ke waktu karena berbagai alasan yang diharapkan dan tidak terduga.

Masalah ini semakin diperparah oleh fakta bahwa banyak praktisi ML tidak berasal dari latar belakang rekayasa perangkat lunak, jadi mereka mungkin tidak terbiasa dengan praktik terbaik industri ini, seperti menulis kode yang dapat diuji, memodulasi komponen, dan menggunakan kontrol versi secara efektif. Tantangan-tantangan ini menciptakan utang teknis, dan solusi menjadi lebih kompleks dan sulit dipertahankan dari waktu ke waktu, didukung oleh efek peracikan, untuk tim ML.

Panduan ini menyebutkan praktik terbaik operasi (MLOps) ML yang membantu mengurangi tantangan ini dalam proyek dan beban kerja ML.

Karena MLOps merupakan perhatian lintas sektoral, masalah ini tidak hanya memengaruhi proses penerapan dan pemantauan, tetapi seluruh siklus hidup model. Dalam panduan ini, praktik MLOps terbaik diatur ke dalam empat bidang utama:

- Data
- Pelatihan
- Deployment
- Pemantauan

## Hasil bisnis yang ditargetkan

Menerapkan model ML dalam produksi adalah tugas yang membutuhkan upaya berkelanjutan dan tim yang berdedikasi untuk mempertahankan sumber daya ini sepanjang masa hidup mereka (dalam beberapa kasus, bahkan bertahun-tahun). Model ML dapat membuka nilai yang cukup besar dari data bisnis, tetapi mereka memiliki biaya tinggi. Untuk meminimalkan biaya, perusahaan harus mengikuti praktik yang baik dalam pengembangan perangkat lunak dan ilmu data. Mereka harus

menyadari nuansa sistem ML, seperti data drift, yang membuat model tampil tak terduga setelah beberapa saat. Dengan menyadari kekhawatiran ini, perusahaan dapat memenuhi tujuan bisnis mereka dengan aman dan dengan kelincahan dalam jangka pendek dan jangka panjang.

Ada beberapa jenis model ML, dan industri yang mereka targetkan memiliki berbagai jenis tugas dan masalah bisnis, jadi Anda perlu mempertimbangkan serangkaian masalah yang berbeda untuk setiap model dan industri. Praktik yang tercantum dalam panduan ini tidak spesifik untuk model atau bisnis, tetapi berlaku untuk serangkaian model dan industri yang luas untuk meningkatkan waktu penyebaran, menghasilkan produktivitas yang lebih tinggi, dan membangun tata kelola dan keamanan yang lebih kuat.

Menempatkan model ke dalam produksi adalah tugas multi-disiplin yang membutuhkan ilmuwan data, insinyur pembelajaran mesin, insinyur data, dan insinyur perangkat lunak. Saat Anda membangun tim ML Anda, kami sarankan Anda menargetkan keterampilan dan latar belakang ini.

# Data

DevOps adalah praktik rekayasa perangkat lunak yang berhubungan dengan operasionalisasi perangkat lunak. Elemen umum adalah kode DevOps yang dikendalikan versi, integrasi berkelanjutan dan pipeline pengiriman berkelanjutan (CI/CD), pengujian unit, dan pembuatan dan penerapan kode yang dapat direproduksi, yang semuanya melibatkan kode. Model ML adalah produk kode dan data, sehingga data harus memenuhi standar yang sama dengan kode. MLOps harus menjawab pertanyaan terkait data seperti bagaimana menjaga kualitas data, cara mengidentifikasi kasus tepi dalam data, cara mengamankan data, dan cara membuat data lebih dapat dipelihara.

Topik



# Pelabelan

## Berikan instruksi pelabelan yang jelas

Kumpulan data mungkin menyertakan sampel ambigu yang menghasilkan pelabelan yang tidak konsisten di seluruh kumpulan data. Misalnya, pertimbangkan tugas memberi label pada gambar yang berisi kucing. Beberapa sampel mungkin hanya berisi sekilas hewan. Haruskah mereka ditandai dengan label positif atau negatif? Jenis masalah ini dapat diselesaikan dengan memberikan instruksi yang jelas dan obyektif kepada pelabel.

## Gunakan suara mayoritas

Sekarang pertimbangkan masalah pelabelan speech-to-text kumpulan data yang berisi audio berisik dengan kata-kata yang secara fonetis mirip atau identik dengan yang lain, seperti know and go, shoe and two, cry and high, atau right and write. Dalam hal ini, pelabel mungkin memberi label sampel ini secara tidak konsisten.

Untuk mempertahankan tingkat kebenaran yang tinggi dalam pelabelan, pendekatan umum adalah dengan menggunakan pemungutan suara mayoritas, di mana sampel data yang sama diberikan kepada banyak pekerja dan hasilnya dikumpulkan. Metode ini dan variasinya yang lebih canggih

dijelaskan dalam posting blog Gunakan kebijaksanaan orang banyak dengan Amazon SageMaker AI Ground Truth untuk membubuhi keterangan data lebih akurat di blog AWS Machine Learning.

# Pemisahan dan kebocoran data

Kebocoran data terjadi ketika model Anda mendapatkan data selama inferensi—saat model dalam produksi dan menerima permintaan prediksi—yang seharusnya tidak memiliki akses ke, seperti sampel data yang digunakan untuk pelatihan, atau informasi yang tidak akan tersedia saat model digunakan dalam produksi.

Jika model Anda diuji secara tidak sengaja pada data pelatihan, kebocoran data dapat menyebabkan overfitting. Overfitting berarti model Anda tidak menggeneralisasi dengan baik ke data yang tidak terlihat. Bagian ini memberikan praktik terbaik untuk menghindari kebocoran data dan overfitting.

## Membagi data Anda menjadi setidaknya tiga set

Salah satu sumber kebocoran data yang umum adalah membagi (memisahkan) data Anda secara tidak benar selama pelatihan. Misalnya, ilmuwan data mungkin secara sadar atau tidak sadar melatih model pada data yang digunakan untuk pengujian. Dalam situasi seperti itu, Anda mungkin mengamati metrik keberhasilan yang sangat tinggi yang disebabkan oleh overfitting. Untuk mengatasi masalah ini, Anda harus membagi data menjadi setidaknya tiga set:`training`,`validation`, dan`testing`.

Dengan memisahkan data Anda dengan cara ini, Anda dapat menggunakan `validation` set untuk memilih dan menyetel parameter yang Anda gunakan untuk mengontrol proses pembelajaran (hyperparameters). Ketika Anda telah mencapai hasil yang diinginkan atau mencapai dataran tinggi peningkatan, lakukan evaluasi di lokasi syuting. `testing` Metrik kinerja untuk `testing` set harus serupa dengan metrik untuk set lainnya. Ini menunjukkan tidak ada ketidakcocokan distribusi antara set, dan model Anda diharapkan dapat digeneralisasi dengan baik dalam produksi.

## Gunakan algoritma split bertingkat

Saat Anda membagi data menjadi`training`,`validation`, dan `testing` untuk kumpulan data kecil, atau saat Anda bekerja dengan data yang sangat tidak seimbang, pastikan untuk menggunakan algoritma split bertingkat. Stratifikasi menjamin bahwa setiap split berisi kira-kira jumlah atau distribusi kelas yang sama untuk setiap split. Pustaka scikit-learn SM sudah mengimplementasikan stratifikasi, begitu juga Apache Spark.

Untuk ukuran sampel, pastikan bahwa set validasi dan pengujian memiliki data yang cukup untuk evaluasi, sehingga Anda dapat mencapai kesimpulan yang signifikan secara statistik. Misalnya, ukuran split umum untuk kumpulan data yang relatif kecil (kurang dari 1 juta sampel) adalah 70%, 15%, dan 15%, untuk`training`,, dan. `validation` `testing` Untuk kumpulan data yang sangat besar (lebih dari 1 juta sampel), Anda dapat menggunakan 90%, 5%, dan 5%, untuk memaksimalkan data pelatihan yang tersedia.

Dalam beberapa kasus penggunaan, akan berguna untuk membagi data menjadi set tambahan, karena data produksi mungkin mengalami perubahan radikal dan tiba-tiba dalam distribusi selama periode pengumpulannya. Misalnya, pertimbangkan proses pengumpulan data untuk membangun model peramalan permintaan untuk item toko kelontong. Jika tim ilmu data mengumpulkan `training` data selama 2019 dan `testing` data dari Januari 2020 hingga Maret 2020, sebuah model mungkin akan mendapat skor bagus di `testing` lokasi syuting. Namun, ketika model tersebut digunakan dalam produksi, pola konsumen untuk barang-barang tertentu akan berubah secara signifikan karena pandemi COVID-19, dan model tersebut akan menghasilkan hasil yang buruk. Dalam skenario ini, masuk akal untuk menambahkan set lain (misalnya,`recent_testing`) sebagai perlindungan tambahan untuk persetujuan model. Penambahan ini dapat mencegah Anda menyetujui model untuk produksi yang akan langsung berkinerja buruk karena ketidakcocokan distribusi.

Dalam beberapa kasus, Anda mungkin ingin membuat tambahan `validation` atau `testing` set yang menyertakan jenis sampel tertentu, seperti data yang terkait dengan populasi minoritas. Sampel data ini penting untuk dilakukan dengan benar tetapi mungkin tidak terwakili dengan baik dalam kumpulan data keseluruhan. Subset data ini disebut irisan.

Pertimbangkan kasus model ML untuk analisis kredit yang dilatih pada data untuk seluruh negara, dan seimbang untuk memperhitungkan secara merata seluruh domain variabel target. Selain itu, pertimbangkan bahwa model ini mungkin memiliki `City` fitur. Jika bank yang menggunakan model ini memperluas bisnisnya ke kota tertentu, mungkin tertarik dengan kinerja model untuk wilayah itu. Jadi, jalur persetujuan seharusnya tidak hanya menilai kualitas model berdasarkan data pengujian untuk seluruh negara, tetapi juga harus mengevaluasi data pengujian untuk potongan kota tertentu.

Ketika ilmuwan data bekerja pada model baru, mereka dapat dengan mudah menilai kemampuan model dan memperhitungkan kasus tepi dengan mengintegrasikan irisan yang kurang terwakili dalam tahap validasi model.

## Pertimbangkan sampel duplikat saat melakukan pemisahan acak

Sumber kebocoran lain yang kurang umum adalah dalam kumpulan data yang mungkin mengandung terlalu banyak sampel duplikat. Dalam kasus ini, bahkan jika Anda membagi data menjadi subset,

subset yang berbeda mungkin memiliki sampel yang sama. Bergantung pada jumlah duplikat, overfitting mungkin disalahartikan sebagai generalisasi.

### Pertimbangkan fitur yang mungkin tidak tersedia saat menerima kesimpulan dalam produksi

Kebocoran data juga terjadi ketika model dilatih dengan fitur yang tidak tersedia dalam produksi, pada saat kesimpulan dipanggil. Karena model sering dibangun berdasarkan data historis, data ini mungkin diperkaya dengan kolom atau nilai tambahan yang tidak ada di beberapa titik waktu. Pertimbangkan kasus model persetujuan kredit yang memiliki fitur yang melacak berapa banyak pinjaman yang telah dibuat pelanggan dengan bank dalam enam bulan terakhir. Ada risiko kebocoran data jika model ini digunakan dan digunakan untuk persetujuan kredit bagi pelanggan baru yang tidak memiliki riwayat enam bulan dengan bank.

Amazon SageMaker AI Feature Store membantu memecahkan masalah ini. Anda dapat menguji model Anda lebih akurat dengan menggunakan kueri perjalanan waktu, yang dapat Anda gunakan untuk melihat data pada titik waktu tertentu.

## Toko fitur

Menggunakan SageMaker AI Feature Store meningkatkan produktivitas tim, karena memisahkan batas komponen (misalnya, penyimpanan versus penggunaan). Ini juga menyediakan fitur yang dapat digunakan kembali di berbagai tim ilmu data dalam organisasi Anda.

### Gunakan kueri perjalanan waktu

Kemampuan perjalanan waktu di Feature Store membantu mereproduksi pembuatan model dan mendukung praktik tata kelola yang lebih kuat. Ini dapat berguna ketika sebuah organisasi ingin menilai garis keturunan data, mirip dengan bagaimana alat kontrol versi seperti Git menilai kode. Pertanyaan perjalanan waktu juga membantu organisasi menyediakan data yang akurat untuk pemeriksaan kepatuhan. Untuk informasi selengkapnya, lihat Memahami kemampuan utama Amazon SageMaker AI Feature Store di blog AWS Machine Learning.

### Gunakan IAM role

Feature Store juga membantu meningkatkan keamanan tanpa memengaruhi produktivitas dan inovasi tim. Anda dapat menggunakan peran AWS Identity and Access Management (IAM) untuk memberikan atau membatasi akses terperinci ke fitur tertentu untuk pengguna atau grup tertentu.

Misalnya, kebijakan berikut membatasi akses ke fitur sensitif di Feature Store.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket--usw2-az1--x-s3/12345678910/
sagemaker/us-east-2/offline-store/doctor-appointments"
        }
    ]
}
```

Untuk informasi selengkapnya tentang keamanan data dan enkripsi menggunakan Feature Store, lihat Keamanan dan kontrol akses dalam dokumentasi SageMaker AI.

## Gunakan pengujian unit

Ketika ilmuwan data membuat model berdasarkan beberapa data, mereka sering membuat asumsi tentang distribusi data, atau mereka melakukan analisis menyeluruh untuk sepenuhnya memahami properti data. Ketika model-model ini digunakan, mereka akhirnya menjadi basi. Ketika dataset menjadi usang, ilmuwan data, insinyur ML, dan (dalam beberapa kasus) sistem otomatis melatih kembali model dengan data baru yang diambil dari toko online atau offline.

Namun, distribusi data baru ini mungkin telah berubah, yang dapat mempengaruhi kinerja algoritma saat ini. Cara otomatis untuk memeriksa jenis masalah ini adalah dengan meminjam konsep pengujian unit dari rekayasa perangkat lunak. Hal-hal umum yang harus diuji termasuk persentase nilai yang hilang, kardinalitas variabel kategoris, dan apakah kolom bernilai riil mematuhi beberapa distribusi yang diharapkan dengan menggunakan kerangka kerja seperti statistik uji hipotesis (t-test). Anda mungkin juga ingin memvalidasi skema data, untuk memastikannya tidak berubah dan tidak akan menghasilkan fitur input yang tidak valid secara diam-diam.

Pengujian unit memerlukan pemahaman data dan domainnya sehingga Anda dapat merencanakan pernyataan yang tepat untuk dilakukan sebagai bagian dari proyek ML. Untuk informasi selengkapnya, lihat Menguji kualitas data dalam skala besar dengan PyDeequ di blog AWS Big Data.

# Pelatihan

MLOps berkaitan dengan operasionalisasi siklus hidup ML. Oleh karena itu, harus memfasilitasi pekerjaan ilmuwan data dan insinyur data untuk menciptakan model pragmatis yang mencapai kebutuhan bisnis dan bekerja dengan baik dalam jangka panjang, tanpa menimbulkan hutang teknis.

Ikuti praktik terbaik di bagian ini untuk membantu mengatasi tantangan pelatihan model.

Topik



## Buat model dasar

Ketika praktisi menghadapi masalah bisnis dengan solusi ML, biasanya kecenderungan pertama mereka adalah menggunakan algoritme. state-of-the-art Praktik ini berisiko, karena kemungkinan state-of-the-art algoritme belum teruji waktu. Selain itu, state-of-the-art algoritme seringkali lebih kompleks dan tidak dipahami dengan baik, sehingga mungkin hanya menghasilkan perbaikan marjinal daripada model alternatif yang lebih sederhana. Praktik yang lebih baik adalah membuat model dasar yang relatif cepat untuk divalidasi dan diterapkan, dan dapat memperoleh kepercayaan dari pemangku kepentingan proyek.

Saat Anda membuat baseline, sebaiknya Anda mengevaluasi kinerja metriknya bila memungkinkan. Bandingkan kinerja model dasar dengan sistem otomatis atau manual lainnya untuk menjamin keberhasilannya dan untuk memastikan bahwa implementasi model atau proyek dapat disampaikan dalam jangka menengah dan panjang.

Model dasar harus divalidasi lebih lanjut dengan insinyur ML untuk mengkonfirmasi bahwa model dapat memberikan persyaratan non-fungsional yang telah ditetapkan untuk proyek, seperti waktu inferensi, seberapa sering data diharapkan untuk mengubah distribusi, jika model dapat dengan mudah dilatih ulang dalam kasus ini, dan bagaimana itu akan digunakan, yang akan mempengaruhi biaya solusi. Dapatkan sudut pandang multi-disiplin tentang pertanyaan-pertanyaan ini untuk meningkatkan kemungkinan Anda mengembangkan model yang sukses dan berjalan lama.

Ilmuwan data mungkin cenderung menambahkan fitur sebanyak mungkin ke model dasar. Meskipun ini meningkatkan kemampuan model untuk memprediksi hasil yang diinginkan, beberapa fitur ini mungkin hanya menghasilkan peningkatan metrik tambahan. Banyak fitur, terutama yang sangat berkorelasi, mungkin berlebihan. Menambahkan terlalu banyak fitur meningkatkan biaya, karena membutuhkan lebih banyak sumber daya komputasi dan penyetelan. Terlalu banyak fitur juga mempengaruhi day-to-day operasi untuk model, karena penyimpangan data menjadi lebih mungkin atau terjadi lebih cepat.

Pertimbangkan model di mana dua fitur input sangat berkorelasi, tetapi hanya satu fitur yang memiliki kausalitas. Misalnya, model yang memprediksi apakah pinjaman akan gagal bayar mungkin memiliki fitur input seperti usia pelanggan dan pendapatan, yang mungkin sangat berkorelasi, tetapi hanya pendapatan yang harus digunakan untuk memberi atau menolak pinjaman. Model yang dilatih pada kedua fitur ini mungkin mengandalkan fitur yang tidak memiliki kausalitas, seperti usia, untuk menghasilkan output prediksi. Jika, setelah pergi ke produksi, model menerima permintaan inferensi untuk pelanggan yang lebih tua atau lebih muda dari usia rata-rata yang termasuk dalam set pelatihan, itu mungkin mulai berkinerja buruk.

Selain itu, setiap fitur individu berpotensi mengalami pergeseran distribusi saat dalam produksi dan menyebabkan model berperilaku tidak terduga. Untuk alasan ini, semakin banyak fitur yang dimiliki model, semakin rapuh sehubungan dengan penyimpangan dan kebuntuan.

Ilmuwan data harus menggunakan ukuran korelasi dan nilai Shapley untuk mengukur fitur mana yang menambah nilai prediksi yang cukup dan harus disimpan. Memiliki model kompleks seperti itu meningkatkan kemungkinan loop umpan balik, di mana model mengubah lingkungan tempat model itu dimodelkan. Contohnya adalah sistem rekomendasi di mana perilaku konsumen dapat berubah karena rekomendasi model. Loop umpan balik yang bekerja di seluruh model kurang umum. Misalnya, pertimbangkan sistem rekomendasi yang merekomendasikan film, dan sistem lain yang merekomendasikan buku. Jika kedua model menargetkan set konsumen yang sama, mereka akan saling mempengaruhi.

Untuk setiap model yang Anda kembangkan, pertimbangkan faktor mana yang mungkin berkontribusi pada dinamika ini, sehingga Anda tahu metrik mana yang harus dipantau dalam produksi.

## Gunakan pendekatan data-sentris dan analisis kesalahan

Jika Anda menggunakan model sederhana, tim ML Anda dapat fokus pada peningkatan data itu sendiri, dan mengambil pendekatan data-sentris alih-alih pendekatan model-sentris. Jika proyek Anda menggunakan data tidak terstruktur, seperti gambar, teks, audio, dan format lain yang dapat dinilai oleh manusia (dibandingkan dengan data terstruktur, yang mungkin lebih sulit untuk dipetakan

ke label secara efisien), praktik yang baik untuk mendapatkan kinerja model yang lebih baik adalah melakukan analisis kesalahan.

Analisis kesalahan melibatkan evaluasi model pada set validasi dan memeriksa kesalahan yang paling umum. Ini membantu mengidentifikasi kelompok potensial dari sampel data serupa yang model mungkin mengalami kesulitan untuk diperbaiki. Untuk melakukan analisis kesalahan, Anda dapat membuat daftar kesimpulan yang memiliki kesalahan prediksi lebih tinggi, atau kesalahan peringkat di mana sampel dari satu kelas diprediksi berasal dari kelas lain, misalnya.

## Arsitek model Anda untuk iterasi cepat

Ketika ilmuwan data mengikuti praktik terbaik, mereka dapat bereksperimen dengan algoritme baru atau mencampur dan mencocokkan fitur yang berbeda dengan mudah dan cepat selama pembuktian konsep atau bahkan pelatihan ulang. Eksperimen ini berkontribusi pada keberhasilan dalam produksi. Praktik yang baik adalah membangun model dasar, menggunakan algoritme yang sedikit lebih kompleks dan menambahkan fitur baru secara berulang sambil memantau kinerja pada set pelatihan dan validasi untuk membandingkan perilaku aktual dengan perilaku yang diharapkan. Kerangka pelatihan ini dapat memberikan keseimbangan optimal dalam kekuatan prediksi dan membantu menjaga model sesederhana mungkin dengan jejak utang teknis yang lebih kecil.

Untuk iterasi cepat, ilmuwan data harus menukar implementasi model yang berbeda untuk menentukan model terbaik untuk digunakan untuk data tertentu. Jika Anda memiliki tim besar, tenggat waktu yang singkat, dan logistik terkait manajemen proyek lainnya, iterasi cepat bisa sulit tanpa metode yang ada.

Dalam rekayasa perangkat lunak, prinsip substitusi Liskov adalah mekanisme untuk merancang interaksi antar komponen perangkat lunak. Prinsip ini menyatakan bahwa Anda harus dapat mengganti satu implementasi antarmuka dengan implementasi lain tanpa merusak aplikasi klien atau implementasinya. Saat Anda menulis kode pelatihan untuk sistem ML Anda, Anda dapat menggunakan prinsip ini untuk menetapkan batasan dan merangkum kode, sehingga Anda dapat mengganti algoritme dengan mudah, dan mencoba algoritme baru dengan lebih efektif.

Misalnya, dalam kode berikut, Anda dapat menambahkan eksperimen baru hanya dengan menambahkan implementasi kelas baru.

```
from abc import ABC, abstractmethod

from pandas import DataFrame
```

```
class ExperimentRunner(object):

    def __init__(self, *experiments):
        self.experiments = experiments

    def run(self, df: DataFrame) -> None:
        for experiment in self.experiments:
            result = experiment.run(df)
            print(f'Experiment "{experiment.name}" gave result {result}')


class Experiment(ABC):

    @abstractmethod
    def run(self, df: DataFrame) -> float:
        pass

    @property
    @abstractmethod
    def name(self) -> str:
        pass


class Experiment1(Experiment):

    def run(self, df: DataFrame) -> float:
        print('performing experiment 1')
        return 0

    def name(self) -> str:
        return 'experiment 1'


class Experiment2(Experiment):

    def run(self, df: DataFrame) -> float:
        print('performing experiment 2')
        return 0

    def name(self) -> str:
        return 'experiment 2'
```

```
class Experiment3(Experiment):

    def run(self, df: DataFrame) -> float:
        print('performing experiment 3')
        return 0

    def name(self) -> str:
        return 'experiment 3'


if __name__ == '__main__':
    runner = ExperimentRunner(*[
        Experiment1(),
        Experiment2(),
        Experiment3()
    ])
    df = ...
    runner.run(df)
```

## Lacak eksperimen ML Anda

Ketika Anda bekerja dengan sejumlah besar eksperimen, penting untuk mengukur apakah perbaikan yang Anda amati adalah produk dari perubahan atau peluang yang diterapkan. Anda dapat menggunakan Amazon SageMaker AI Experiments untuk membuat eksperimen dengan mudah dan mengaitkan metadata dengannya untuk pelacakan, perbandingan, dan evaluasi.

Mengurangi keacakan proses pembuatan model berguna untuk debugging, pemecahan masalah, dan meningkatkan tata kelola, karena Anda dapat memprediksi inferensi model keluaran dengan lebih pasti, dengan kode dan data yang sama.

Seringkali tidak mungkin untuk membuat kode pelatihan sepenuhnya dapat direproduksi, karena inisialisasi bobot acak, sinkronisitas komputasi paralel, kompleksitas GPU bagian dalam, dan faktor non-deterministik serupa. Namun, mengatur benih acak dengan benar, untuk memastikan bahwa setiap latihan dimulai dari titik yang sama dan berperilaku serupa, secara signifikan meningkatkan prediktabilitas hasil.

## Memecahkan masalah pekerjaan pelatihan

Dalam beberapa kasus, mungkin sulit bagi ilmuwan data untuk menyesuaikan bahkan model dasar yang sangat sederhana. Dalam hal ini, mereka mungkin memutuskan bahwa mereka memerlukan

algoritme yang dapat lebih sesuai dengan fungsi kompleks. Tes yang baik adalah menggunakan garis dasar dari bagian yang sangat kecil dari kumpulan data (misalnya, sekitar 10 sampel) untuk memastikan bahwa algoritme sesuai dengan sampel ini. Ini membantu mengesampingkan masalah data atau kode.

Alat bermanfaat lainnya untuk men-debug skenario kompleks adalah Amazon SageMaker AI Debugger, yang dapat menangkap masalah yang terkait dengan kebenaran algoritmik dan infrastruktur, seperti penggunaan komputasi yang optimal.

# Deployment

Dalam rekayasa perangkat lunak, menempatkan kode dalam produksi memerlukan uji tuntas, karena kode mungkin berperilaku tidak terduga, perilaku pengguna yang tidak terduga dapat merusak perangkat lunak, dan kasus tepi yang tidak terduga dapat ditemukan. Insinyur dan DevOps insinyur perangkat lunak biasanya menggunakan pengujian unit dan strategi rollback untuk mengurangi risiko ini. Dengan ML, menempatkan model dalam produksi membutuhkan lebih banyak perencanaan, karena lingkungan nyata diperkirakan akan melayang, dan pada banyak kesempatan, model divalidasi pada metrik yang merupakan proxy untuk metrik bisnis nyata yang mereka coba tingkatkan.

Ikuti praktik terbaik di bagian ini untuk membantu mengatasi tantangan ini.

Topik

- Otomatiskan siklus penerapan
- Pilih strategi penerapan
- Pertimbangkan persyaratan inferensi Anda

## Otomatiskan siklus penerapan

Proses pelatihan dan penerapan harus sepenuhnya otomatis untuk mencegah kesalahan manusia dan untuk memastikan bahwa pemeriksaan build dijalankan secara konsisten. Pengguna tidak boleh memiliki izin akses tulis ke lingkungan produksi.

Amazon SageMaker AI Pipelines dan AWS CodePipelinemembantu membuat CI/CD pipelines for ML projects. One of the advantages of using a CI/CD pipeline adalah bahwa semua kode yang digunakan untuk menyerap data, melatih model, dan melakukan pemantauan dapat dikontrol versi dengan menggunakan alat seperti Git. Terkadang Anda harus melatih ulang model dengan menggunakan algoritma dan hyperparameter yang sama, tetapi data yang berbeda. Satu-satunya cara untuk memverifikasi bahwa Anda menggunakan versi algoritma yang benar adalah dengan menggunakan kontrol sumber dan tag. Anda dapat menggunakan templat proyek default yang disediakan oleh SageMaker AI sebagai titik awal untuk MLOps latihan Anda.

Saat Anda membuat pipeline CI/CD untuk menerapkan model Anda, pastikan untuk menandai artefak build Anda dengan pengenal build, versi kode atau komit, dan versi data. Praktik ini membantu Anda memecahkan masalah penerapan apa pun. Penandaan juga terkadang diperlukan untuk model yang membuat prediksi di bidang yang sangat diatur. Kemampuan untuk bekerja mundur

dan mengidentifikasi data, kode, pembuatan, pemeriksaan, dan persetujuan yang tepat yang terkait dengan model ML dapat membantu meningkatkan tata kelola secara signifikan.

Bagian dari tugas pipa CI/CD adalah melakukan tes pada apa yang sedang dibangun. Meskipun pengujian unit data diharapkan terjadi sebelum data dicerna oleh feature store, pipeline masih bertanggung jawab untuk melakukan pengujian pada input dan output model tertentu dan untuk memeriksa metrik kunci. Salah satu contoh pemeriksaan tersebut adalah untuk memvalidasi model baru pada set validasi tetap dan untuk mengkonfirmasi bahwa kinerjanya mirip dengan model sebelumnya dengan menggunakan ambang batas yang ditetapkan. Jika kinerja secara signifikan lebih rendah dari yang diharapkan, build akan gagal dan model tidak boleh diproduksi.

Penggunaan pipa CI/CD yang ekstensif juga mendukung permintaan tarik, yang membantu mencegah kesalahan manusia. Saat Anda menggunakan permintaan tarik, setiap perubahan kode harus ditinjau dan disetujui oleh setidaknya satu anggota tim lainnya sebelum dapat diproduksi. Permintaan tarik juga berguna untuk mengidentifikasi kode yang tidak mematuhi aturan bisnis dan untuk menyebarkan pengetahuan dalam tim.

## Pilih strategi penerapan

MLOps strategi penyebaran termasuk blue/green, canary, shadow, and A/B pengujian.

### Biru/hijau

Blue/green deployments are very common in software development. In this mode, two systems are kept running during development: blue is the old environment (in this case, the model that is being replaced) and green is the newly released model that is going to production. Changes can easily be rolled back with minimum downtime, because the old system is kept alive. For more in-depth information about blue/greenpenerapan dalam konteks SageMaker, lihat posting blog Menyebarkan dan memantau titik akhir Amazon SageMaker AI dengan aman dengan AWS CodePipeline dan di blog AWS CodeDeploy Machine Learning AWS .

### Kenari

Penerapan Canary mirip dengan blue/green deployments in that both keep two models running together. However, in canary deployments, the new model is rolled out to users incrementally, until all traffic eventually shifts over to the new model. As in blue/green penerapan, risiko dikurangi karena model baru (dan berpotensi rusak) dipantau secara ketat selama peluncuran awal, dan dapat

dibatalkan jika terjadi masalah. Di SageMaker AI, Anda dapat menentukan distribusi lalu lintas awal dengan menggunakan InitialVariantWeightAPI.

## Bayangan

Anda dapat menggunakan penerapan bayangan untuk membawa model ke produksi dengan aman. Dalam mode ini, model baru bekerja bersama model lama atau proses bisnis, dan melakukan kesimpulan tanpa mempengaruhi keputusan apa pun. Mode ini dapat berguna sebagai pemeriksaan akhir atau eksperimen kesetiaan yang lebih tinggi sebelum Anda mempromosikan model ke produksi.

Mode bayangan berguna saat Anda tidak memerlukan umpan balik inferensi pengguna. Anda dapat menilai kualitas prediksi dengan melakukan analisis kesalahan dan membandingkan model baru dengan model lama, dan Anda dapat memantau distribusi output untuk memverifikasi bahwa itu seperti yang diharapkan. Untuk melihat cara melakukan penyebaran bayangan dengan SageMaker AI, lihat posting blog Menerapkan model shadow HTML di Amazon SageMaker AI di blog AWS Machine Learning.

## Pengujian A/B

Ketika praktisi ML mengembangkan model di lingkungan mereka, metrik yang mereka optimalkan seringkali merupakan proxy untuk metrik bisnis yang benar-benar penting. Hal ini membuat sulit untuk mengetahui dengan pasti apakah model baru benar-benar akan meningkatkan hasil bisnis, seperti pendapatan dan rasio klik-tayang, dan mengurangi jumlah keluhan pengguna.

Pertimbangkan kasus situs web e-commerce di mana tujuan bisnisnya adalah menjual produk sebanyak mungkin. Tim peninjau tahu bahwa penjualan dan kepuasan pelanggan berkorelasi langsung dengan ulasan yang informatif dan akurat. Seorang anggota tim mungkin mengusulkan algoritma peringkat tinjauan baru untuk meningkatkan penjualan. Dengan menggunakan pengujian A/B, mereka dapat meluncurkan algoritme lama dan baru ke grup pengguna yang berbeda tetapi serupa, dan memantau hasilnya untuk melihat apakah pengguna yang menerima prediksi dari model yang lebih baru lebih cenderung melakukan pembelian.

Pengujian A/B juga membantu mengukur dampak bisnis dari model staleness dan drift. Tim dapat menempatkan model baru dalam produksi dengan beberapa pengulangan, melakukan pengujian A/B dengan masing-masing model, dan membuat bagan usia versus kinerja. Ini akan membantu tim memahami volatilitas drift data dalam data produksi mereka.

Untuk informasi lebih lanjut tentang cara melakukan pengujian A/B dengan SageMaker AI, lihat posting blog model A/B Testing ML dalam produksi menggunakan Amazon SageMaker AI di blog AWS Machine Learning.

# Pertimbangkan persyaratan inferensi Anda

Dengan SageMaker AI, Anda dapat memilih infrastruktur yang mendasarinya untuk menerapkan model Anda dengan berbagai cara. Kemampuan pemanggilan inferensi ini mendukung kasus penggunaan dan profil biaya yang berbeda. Opsi Anda mencakup inferensi waktu nyata, inferensi asinkron, dan transformasi batch, seperti yang dibahas di bagian berikut.

## Inferensi waktu nyata

Inferensi waktu nyata sangat ideal untuk beban kerja inferensi di mana Anda memiliki persyaratan real-time, interaktif, latensi rendah. Anda dapat menerapkan model Anda ke layanan hosting SageMaker AI dan mendapatkan titik akhir yang dapat digunakan untuk inferensi. Titik akhir ini dikelola sepenuhnya, mendukung penskalaan otomatis (lihat Model SageMaker AI Amazon secara otomatis), dan dapat digunakan di beberapa Availability Zone.

Jika Anda memiliki model pembelajaran mendalam yang dibuat dengan Apache MXNet,, atau PyTorch TensorFlow, Anda juga dapat menggunakan Amazon SageMaker AI Elastic Inference (EI). Dengan EI, Anda dapat melampirkan fraksional GPUs ke instans SageMaker AI apa pun untuk mempercepat inferensi. Anda dapat memilih instance klien untuk menjalankan aplikasi Anda dan melampirkan akselerator EI untuk menggunakan jumlah akselerasi GPU yang benar untuk kebutuhan inferensi Anda.

Pilihan lain adalah menggunakan titik akhir multi-model, yang memberikan solusi yang dapat diskalakan dan hemat biaya untuk menerapkan sejumlah besar model. Titik akhir ini menggunakan wadah penyajian bersama yang diaktifkan untuk meng-host beberapa model. Titik akhir multi-model mengurangi biaya hosting dengan meningkatkan pemanfaatan titik akhir dibandingkan dengan menggunakan titik akhir model tunggal. Mereka juga mengurangi overhead penerapan, karena SageMaker AI mengelola pemuatan model dalam memori dan menskalakannya berdasarkan pola lalu lintas.

Untuk praktik terbaik tambahan untuk menerapkan model ML di SageMaker AI, lihat Praktik terbaik penerapan dalam dokumentasi SageMaker AI.

## Inferensi asinkron

Amazon SageMaker AI Asynchronous Inference adalah kemampuan dalam SageMaker AI yang mengantri permintaan masuk dan memprosesnya secara asinkron. Opsi ini sangat ideal untuk permintaan dengan ukuran muatan besar hingga 1 GB, waktu pemrosesan yang lama, dan

persyaratan latensi mendekati waktu nyata. Inferensi asinkron memungkinkan Anda menghemat biaya dengan secara otomatis menskalakan jumlah instans ke nol saat tidak ada permintaan untuk diproses, jadi Anda hanya membayar saat titik akhir memproses permintaan.

## Transformasi Batch

Gunakan transformasi batch saat Anda ingin melakukan hal berikut:

- Preprocess dataset untuk menghilangkan noise atau bias yang mengganggu pelatihan atau inferensi dari dataset Anda.
- Dapatkan kesimpulan dari kumpulan data besar.
- Jalankan inferensi saat Anda tidak membutuhkan titik akhir yang persisten.
- Kaitkan catatan input dengan kesimpulan untuk membantu interpretasi hasil.

# Pemantauan

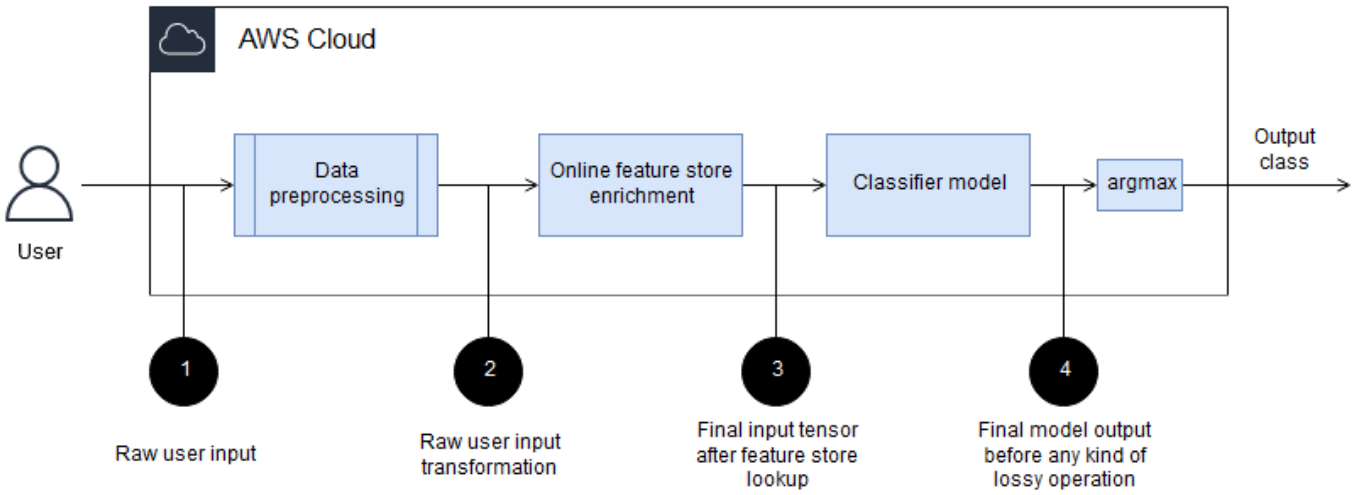
Ketika model sudah dalam produksi dan memberikan nilai bisnis, jalankan pemeriksaan berkelanjutan untuk mengidentifikasi kapan model harus dilatih ulang atau diambil tindakan.

Tim pemantauan Anda harus berperilaku proaktif, tidak reaktif, untuk lebih memahami perilaku data lingkungan, dan untuk mengidentifikasi frekuensi, kecepatan, dan tiba-tiba penyimpangan data. Tim harus mengidentifikasi kasus tepi baru dalam data yang mungkin kurang terwakili dalam set pelatihan, set validasi, dan irisan kasus tepi lainnya. Mereka harus menyimpan metrik kualitas layanan (QoS), menggunakan alarm untuk segera mengambil tindakan ketika masalah muncul, dan menentukan strategi untuk menelan dan mengubah kumpulan data saat ini. Praktik ini dimulai dengan mencatat permintaan dan tanggapan untuk model, untuk memberikan referensi untuk pemecahan masalah atau wawasan tambahan.

Idealnya, transformasi data harus dicatat dalam beberapa tahap kunci selama pemrosesan:

- Sebelum segala jenis preprocessing
- Setelah segala jenis pengayaan feature store
- Setelah semua tahapan utama dari sebuah model
- Sebelum segala jenis fungsi lossy pada output model, seperti `argmax`

Diagram berikut menggambarkan tahapan ini.



Anda dapat menggunakan SageMaker AI Model Monitor untuk secara otomatis menangkap data input dan output dan menyimpannya di Amazon Simple Storage Service (Amazon S3). Anda dapat menerapkan jenis logging perantara lainnya dengan menambahkan log ke wadah penyajian khusus.

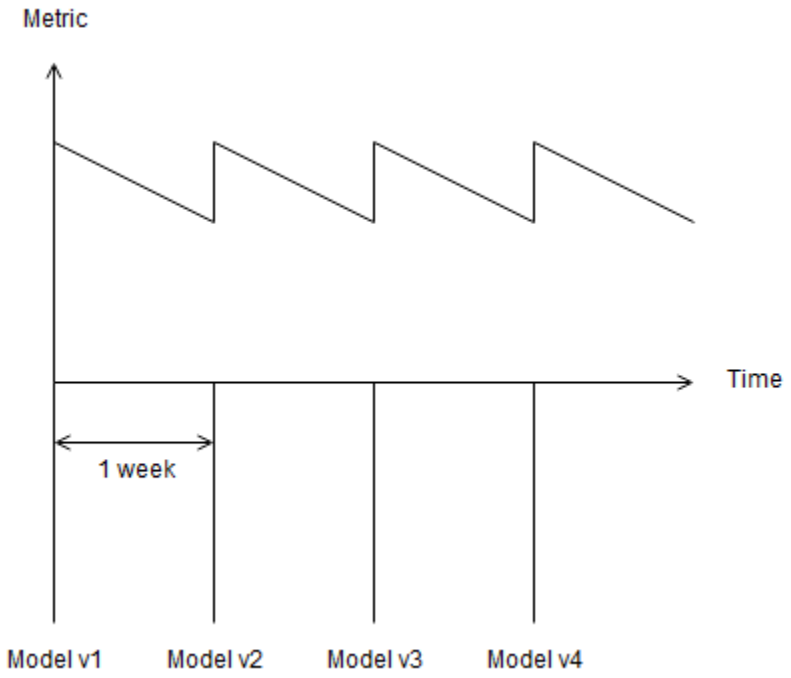
Setelah Anda mencatat data dari model, Anda dapat memantau penyimpangan distribusi. Dalam beberapa kasus, Anda bisa mendapatkan kebenaran dasar (data yang diberi label dengan benar) segera setelah inferensi. Contoh umum dari ini adalah model yang memprediksi iklan yang paling relevan untuk ditampilkan kepada pengguna. Segera setelah pengguna meninggalkan halaman, Anda dapat menentukan apakah mereka mengklik iklan. Jika pengguna telah mengklik iklan, Anda dapat mencatat informasi tersebut. Dalam contoh sederhana ini, Anda dapat dengan mudah mengukur seberapa sukses model Anda dengan menggunakan metrik, seperti akurasi atau F1, yang dapat diukur baik dalam pelatihan maupun dalam penerapan. Untuk informasi selengkapnya tentang skenario ini di mana Anda telah memberi label data, lihat Memantau kualitas model dalam dokumentasi SageMaker AI. Namun, skenario sederhana ini jarang terjadi, karena model sering dirancang untuk mengoptimalkan metrik yang nyaman secara matematis yang hanya proxy untuk hasil bisnis yang sebenarnya. Dalam kasus seperti itu, praktik terbaik adalah memantau hasil bisnis ketika model digunakan dalam produksi.

Pertimbangkan kasus model peringkat ulasan. Jika hasil bisnis yang ditentukan dari model ML adalah untuk menampilkan ulasan yang paling relevan dan berguna di bagian atas halaman web, Anda dapat mengukur keberhasilan model dengan menambahkan tombol seperti “Apakah ini membantu?” untuk setiap review. Mengukur rasio klik-tayang tombol ini bisa menjadi ukuran hasil bisnis yang membantu Anda mengukur seberapa baik kinerja model Anda dalam produksi.

Untuk memantau penyimpangan label input atau output di SageMaker AI, Anda dapat menggunakan kemampuan kualitas data SageMaker AI Model Monitor, yang memantau input dan output. Anda juga dapat menerapkan logika Anda sendiri untuk SageMaker AI Model Monitor dengan membuat wadah khusus.

Memantau data yang diterima model baik dalam waktu pengembangan maupun dalam runtime sangat penting. Insinyur harus memantau data tidak hanya untuk perubahan skema tetapi juga untuk ketidakcocokan distribusi. Mendeteksi perubahan skema lebih mudah dan dapat diimplementasikan dengan seperangkat aturan, tetapi ketidakcocokan distribusi seringkali lebih rumit, terutama karena mengharuskan Anda untuk menentukan ambang batas untuk mengukur kapan harus menaikkan alarm. Dalam kasus di mana distribusi yang dipantau diketahui, seringkali cara termudah adalah dengan memantau parameter distribusi. Dalam kasus distribusi normal, itu akan menjadi mean dan standar deviasi. Metrik kunci lainnya, seperti persentase nilai yang hilang, nilai maksimum, dan nilai minimum, juga berguna.

Anda juga dapat membuat pekerjaan pemantauan berkelanjutan yang mengambil sampel data pelatihan dan data inferensi dan membandingkan distribusinya. Anda dapat membuat pekerjaan ini untuk input model dan output model, dan memplot data terhadap waktu untuk memvisualisasikan penyimpangan mendadak atau bertahap. Ini diilustrasikan dalam bagan berikut.




Untuk lebih memahami profil drift data, seperti seberapa sering distribusi data berubah secara signifikan, pada tingkat berapa, atau seberapa tiba-tiba, kami menyarankan Anda untuk terus menerapkan versi model baru dan memantau kinerjanya. Misalnya, jika tim Anda menerapkan model baru setiap minggu dan mengamati bahwa kinerja model meningkat secara signifikan setiap saat, mereka dapat menentukan bahwa mereka harus memberikan model baru dalam waktu minimal kurang dari seminggu.

# Langkah dan sumber daya selanjutnya

Panduan ini memandu Anda melalui beberapa pertimbangan saat merencanakan siklus hidup model pembelajaran mesin yang ingin Anda bawa ke produksi. Ini membahas tantangan dan praktik terbaik di empat bidang — data, pelatihan, penyebaran, dan pemantauan — dan mencakup sumber daya tambahan yang relevan.

AWS Menyediakan Well-Architected Framework, yang membantu arsitek cloud membangun infrastruktur yang aman, berkinerja tinggi, tangguh, dan efisien untuk berbagai aplikasi, beban kerja, dan domain teknologi. Untuk bacaan tambahan, lihat Machine Learning Lens yang ditawarkan oleh AWS Well-Architected.

## Sumber daya

Dokumentasi Amazon SageMaker AI

- Toko Fitur Amazon SageMaker AI
- Fitur Keamanan Toko dan kontrol akses
- Nilai Shapley
- Amazon SageMaker AI Debugger
- Saluran Pipa Amazon SageMaker AI
- Templat proyek default Amazon SageMaker AI
- SageMaker Inferensi waktu nyata AI
- Secara otomatis menskalakan model Amazon SageMaker AI
- Inferensi asinkron Amazon SageMaker AI
- SageMaker Monitor Model AI

AWS alat pengembang

- AWS CodePipeline

AWS posting blog

- Memahami kemampuan utama Amazon SageMaker AI Feature Store

- Menguji kualitas data dalam skala dengan PyDeequ
- Eksperimen SageMaker AI Amazon
- Menyebarkan dan memantau SageMaker titik akhir Amazon dengan aman dengan dan CodePipeline AWS CodeDeploy
- Terapkan model bayangan bayangan di Amazon AI SageMaker
- A/B Menguji model ML dalam produksi menggunakan Amazon AI SageMaker

# Riwayat dokumen

Tabel berikut menjelaskan perubahan signifikan pada panduan ini. Jika Anda ingin diberi tahu tentang pembaruan masa depan, Anda dapat berlangganan umpan RSS.

| Perubahan | Deskripsi | Tanggal |
| --- | --- | --- |
| Publikasi awal | — | Desember 20, 2021 |

# AWS Glosarium Panduan Preskriptif

Berikut ini adalah istilah yang umum digunakan dalam strategi, panduan, dan pola yang disediakan oleh Panduan AWS Preskriptif. Untuk menyarankan entri, silakan gunakan tautan Berikan umpan balik di akhir glosarium.

## Nomor

7 Rs

Tujuh strategi migrasi umum untuk memindahkan aplikasi ke cloud. Strategi ini dibangun di atas 5 Rs yang diidentifikasi Gartner pada tahun 2011 dan terdiri dari yang berikut:

- Refactor/Re-Architect — Memindahkan aplikasi dan memodifikasi arsitekturnya dengan memanfaatkan sepenuhnya fitur cloud-native untuk meningkatkan kelincahan, kinerja, dan skalabilitas. Ini biasanya melibatkan porting sistem operasi dan database. Contoh: Migrasikan database Oracle lokal Anda ke Amazon Aurora PostgreSQL Compatible Edition.
- Replatform (angkat dan bentuk ulang) — Pindahkan aplikasi ke cloud, dan perkenalkan beberapa tingkat pengoptimalan untuk memanfaatkan kemampuan cloud. Contoh: Migrasikan database Oracle lokal Anda ke Amazon Relational Database Service (Amazon RDS) untuk Oracle di. AWS Cloud
- Pembelian kembali (drop and shop) - Beralih ke produk yang berbeda, biasanya dengan beralih dari lisensi tradisional ke model SaaS. Contoh: Migrasikan sistem manajemen hubungan pelanggan (CRM) Anda ke Salesforce.com.
- Rehost (lift dan shift) — Pindahkan aplikasi ke cloud tanpa membuat perubahan apa pun untuk memanfaatkan kemampuan cloud. Contoh: Migrasikan database Oracle lokal Anda ke Oracle pada instans EC2 di. AWS Cloud
- Relokasi (hypervisor-level lift and shift) — Pindahkan infrastruktur ke cloud tanpa membeli perangkat keras baru, menulis ulang aplikasi, atau memodifikasi operasi yang ada. Anda memigrasikan server dari platform lokal ke layanan cloud untuk platform yang sama. Contoh: Migrasikan Microsoft Hyper-V aplikasi ke AWS.
- Pertahankan (kunjungi kembali) - Simpan aplikasi di lingkungan sumber Anda. Ini mungkin termasuk aplikasi yang memerlukan refactoring besar, dan Anda ingin menunda pekerjaan itu sampai nanti, dan aplikasi lama yang ingin Anda pertahankan, karena tidak ada pembenaran bisnis untuk memigrasikannya.

- Pensiun — Menonaktifkan atau menghapus aplikasi yang tidak lagi diperlukan di lingkungan sumber Anda.

# A

ABAC

Lihat kontrol akses berbasis atribut.

layanan abstrak

Lihat layanan terkelola.

ASAM

Lihat atomisitas, konsistensi, isolasi, daya tahan.

migrasi aktif-aktif

Metode migrasi database di mana database sumber dan target tetap sinkron (dengan menggunakan alat replikasi dua arah atau operasi penulisan ganda), dan kedua database menangani transaksi dari menghubungkan aplikasi selama migrasi. Metode ini mendukung migrasi dalam batch kecil yang terkontrol alih-alih memerlukan pemotongan satu kali. Ini lebih fleksibel tetapi membutuhkan lebih banyak pekerjaan daripada migrasi aktif-pasif.

migrasi aktif-pasif

Metode migrasi database di mana database sumber dan target disimpan dalam sinkron, tetapi hanya database sumber yang menangani transaksi dari menghubungkan aplikasi sementara data direplikasi ke database target. Basis data target tidak menerima transaksi apa pun selama migrasi.

fungsi agregat

Fungsi SQL yang beroperasi pada sekelompok baris dan menghitung nilai pengembalian tunggal untuk grup. Contoh fungsi agregat meliputi `SUM` dan`MAX`.

AI

Lihat kecerdasan buatan.

AIOps

Lihat operasi kecerdasan buatan.

anonimisasi

Proses menghapus informasi pribadi secara permanen dalam kumpulan data. Anonimisasi dapat membantu melindungi privasi pribadi. Data anonim tidak lagi dianggap sebagai data pribadi.

anti-pola

Solusi yang sering digunakan untuk masalah berulang di mana solusinya kontra-produktif, tidak efektif, atau kurang efektif daripada alternatif.

kontrol aplikasi

Pendekatan keamanan yang memungkinkan penggunaan hanya aplikasi yang disetujui untuk membantu melindungi sistem dari malware.

portofolio aplikasi

Kumpulan informasi rinci tentang setiap aplikasi yang digunakan oleh organisasi, termasuk biaya untuk membangun dan memelihara aplikasi, dan nilai bisnisnya. Informasi ini adalah kunci untuk penemuan portofolio dan proses analisis dan membantu mengidentifikasi dan memprioritaskan aplikasi yang akan dimigrasi, dimodernisasi, dan dioptimalkan.

kecerdasan buatan (AI)

Bidang ilmu komputer yang didedikasikan untuk menggunakan teknologi komputasi untuk melakukan fungsi kognitif yang biasanya terkait dengan manusia, seperti belajar, memecahkan masalah, dan mengenali pola. Untuk informasi lebih lanjut, lihat Apa itu Kecerdasan Buatan?

operasi kecerdasan buatan (AIOps)

Proses menggunakan teknik pembelajaran mesin untuk memecahkan masalah operasional, mengurangi insiden operasional dan intervensi manusia, dan meningkatkan kualitas layanan. Untuk informasi selengkapnya tentang cara AIOps digunakan dalam strategi AWS migrasi, lihat panduan integrasi operasi.

enkripsi asimetris

Algoritma enkripsi yang menggunakan sepasang kunci, kunci publik untuk enkripsi dan kunci pribadi untuk dekripsi. Anda dapat berbagi kunci publik karena tidak digunakan untuk dekripsi, tetapi akses ke kunci pribadi harus sangat dibatasi.

atomisitas, konsistensi, isolasi, daya tahan (ACID)

Satu set properti perangkat lunak yang menjamin validitas data dan keandalan operasional database, bahkan dalam kasus kesalahan, kegagalan daya, atau masalah lainnya.

kontrol akses berbasis atribut (ABAC)

Praktik membuat izin berbutir halus berdasarkan atribut pengguna, seperti departemen, peran pekerjaan, dan nama tim. Untuk informasi selengkapnya, lihat ABAC untuk AWS dokumentasi AWS Identity and Access Management (IAM).

sumber data otoritatif

Lokasi di mana Anda menyimpan versi utama data, yang dianggap sebagai sumber informasi yang paling dapat diandalkan. Anda dapat menyalin data dari sumber data otoritatif ke lokasi lain untuk tujuan memproses atau memodifikasi data, seperti menganonimkan, menyunting, atau membuat nama samaran.

Zona Ketersediaan

Lokasi berbeda di dalam Wilayah AWS yang terisolasi dari kegagalan di Availability Zone lainnya dan menyediakan konektivitas jaringan latensi rendah yang murah ke Availability Zone lainnya di Wilayah yang sama.

AWS Kerangka Adopsi Cloud (AWS CAF)

Kerangka pedoman dan praktik terbaik AWS untuk membantu organisasi mengembangkan rencana yang efisien dan efektif untuk bergerak dengan sukses ke cloud. AWS CAF mengatur panduan ke dalam enam area fokus yang disebut perspektif: bisnis, orang, tata kelola, platform, keamanan, dan operasi. Perspektif bisnis, orang, dan tata kelola fokus pada keterampilan dan proses bisnis; perspektif platform, keamanan, dan operasi fokus pada keterampilan dan proses teknis. Misalnya, perspektif masyarakat menargetkan pemangku kepentingan yang menangani sumber daya manusia (SDM), fungsi kepegawaian, dan manajemen orang. Untuk perspektif ini, AWS CAF memberikan panduan untuk pengembangan, pelatihan, dan komunikasi orang untuk membantu mempersiapkan organisasi untuk adopsi cloud yang sukses. Untuk informasi lebih lanjut, lihat situs web AWS CAF dan whitepaper AWS CAF.

AWS Kerangka Kualifikasi Beban Kerja (AWS WQF)

Alat yang mengevaluasi beban kerja migrasi database, merekomendasikan strategi migrasi, dan memberikan perkiraan kerja. AWS WQF disertakan dengan AWS Schema Conversion Tool ()AWS SCT. Ini menganalisis skema database dan objek kode, kode aplikasi, dependensi, dan karakteristik kinerja, dan memberikan laporan penilaian.

# B

bot buruk

Bot yang dimaksudkan untuk mengganggu atau menyebabkan kerugian bagi individu atau organisasi.

BCP

Lihat perencanaan kontinuitas bisnis.

grafik perilaku

Pandangan interaktif yang terpadu tentang perilaku dan interaksi sumber daya dari waktu ke waktu. Anda dapat menggunakan grafik perilaku dengan Amazon Detective untuk memeriksa upaya logon yang gagal, panggilan API yang mencurigakan, dan tindakan serupa. Untuk informasi selengkapnya, lihat Data dalam grafik perilaku di dokumentasi Detektif.

sistem big-endian

Sistem yang menyimpan byte paling signifikan terlebih dahulu. Lihat juga endianness.

klasifikasi biner

Sebuah proses yang memprediksi hasil biner (salah satu dari dua kelas yang mungkin). Misalnya, model ML Anda mungkin perlu memprediksi masalah seperti “Apakah email ini spam atau bukan spam?” atau “Apakah produk ini buku atau mobil?”

filter mekar

Struktur data probabilistik dan efisien memori yang digunakan untuk menguji apakah suatu elemen adalah anggota dari suatu himpunan.

deployment biru/hijau

Strategi penyebaran tempat Anda membuat dua lingkungan yang terpisah namun identik. Anda menjalankan versi aplikasi saat ini di satu lingkungan (biru) dan versi aplikasi baru di lingkungan lain (hijau). Strategi ini membantu Anda dengan cepat memutar kembali dengan dampak minimal.

bot

Aplikasi perangkat lunak yang menjalankan tugas otomatis melalui internet dan mensimulasikan aktivitas atau interaksi manusia. Beberapa bot berguna atau bermanfaat, seperti perayap web yang mengindeks informasi di internet. Beberapa bot lain, yang dikenal sebagai bot buruk, dimaksudkan untuk mengganggu atau membahayakan individu atau organisasi.

botnet

Jaringan bot yang terinfeksi oleh malware dan berada di bawah kendali satu pihak, yang dikenal sebagai bot herder atau operator bot. Botnet adalah mekanisme paling terkenal untuk skala bot dan dampaknya.

cabang

Area berisi repositori kode. Cabang pertama yang dibuat dalam repositori adalah cabang utama. Anda dapat membuat cabang baru dari cabang yang ada, dan Anda kemudian dapat mengembangkan fitur atau memperbaiki bug di cabang baru. Cabang yang Anda buat untuk membangun fitur biasanya disebut sebagai cabang fitur. Saat fitur siap dirilis, Anda menggabungkan cabang fitur kembali ke cabang utama. Untuk informasi selengkapnya, lihat Tentang cabang (GitHub dokumentasi).

akses break-glass

Dalam keadaan luar biasa dan melalui proses yang disetujui, cara cepat bagi pengguna untuk mendapatkan akses ke Akun AWS yang biasanya tidak memiliki izin untuk mengaksesnya. Untuk informasi lebih lanjut, lihat indikator Implementasikan prosedur break-glass dalam panduan Well-Architected AWS .

strategi brownfield

Infrastruktur yang ada di lingkungan Anda. Saat mengadopsi strategi brownfield untuk arsitektur sistem, Anda merancang arsitektur di sekitar kendala sistem dan infrastruktur saat ini. Jika Anda memperluas infrastruktur yang ada, Anda dapat memadukan strategi brownfield dan greenfield.

cache penyangga

Area memori tempat data yang paling sering diakses disimpan.

kemampuan bisnis

Apa yang dilakukan bisnis untuk menghasilkan nilai (misalnya, penjualan, layanan pelanggan, atau pemasaran). Arsitektur layanan mikro dan keputusan pengembangan dapat didorong oleh kemampuan bisnis. Untuk informasi selengkapnya, lihat bagian Terorganisir di sekitar kemampuan bisnis dari Menjalankan layanan mikro kontainer di whitepaper. AWS

perencanaan kelangsungan bisnis (BCP)

Rencana yang membahas dampak potensial dari peristiwa yang mengganggu, seperti migrasi skala besar, pada operasi dan memungkinkan bisnis untuk melanjutkan operasi dengan cepat.

# C

KAFE

Lihat Kerangka Adopsi AWS Cloud.

penyebaran kenari

Rilis versi yang lambat dan bertahap untuk pengguna akhir. Ketika Anda yakin, Anda menyebarkan versi baru dan mengganti versi saat ini secara keseluruhan.

CCoE

Lihat Cloud Center of Excellence.

CDC

Lihat mengubah pengambilan data.

ubah pengambilan data (CDC)

Proses melacak perubahan ke sumber data, seperti tabel database, dan merekam metadata tentang perubahan tersebut. Anda dapat menggunakan CDC untuk berbagai tujuan, seperti mengaudit atau mereplikasi perubahan dalam sistem target untuk mempertahankan sinkronisasi.

rekayasa kekacauan

Sengaja memperkenalkan kegagalan atau peristiwa yang mengganggu untuk menguji ketahanan sistem. Anda dapat menggunakan AWS Fault Injection Service (AWS FIS) untuk melakukan eksperimen yang menekankan AWS beban kerja Anda dan mengevaluasi responsnya.

CI/CD

Lihat integrasi berkelanjutan dan pengiriman berkelanjutan.

klasifikasi

Proses kategorisasi yang membantu menghasilkan prediksi. Model ML untuk masalah klasifikasi memprediksi nilai diskrit. Nilai diskrit selalu berbeda satu sama lain. Misalnya, model mungkin perlu mengevaluasi apakah ada mobil dalam gambar atau tidak.

Enkripsi sisi klien

Enkripsi data secara lokal, sebelum target Layanan AWS menerimanya.

Pusat Keunggulan Cloud (CCoE)

Tim multi-disiplin yang mendorong upaya adopsi cloud di seluruh organisasi, termasuk mengembangkan praktik terbaik cloud, memobilisasi sumber daya, menetapkan jadwal migrasi, dan memimpin organisasi melalui transformasi skala besar. Untuk informasi selengkapnya, lihat posting CCo E di Blog Strategi AWS Cloud Perusahaan.

komputasi cloud

Teknologi cloud yang biasanya digunakan untuk penyimpanan data jarak jauh dan manajemen perangkat IoT. Cloud computing umumnya terhubung ke teknologi edge computing.

model operasi cloud

Dalam organisasi TI, model operasi yang digunakan untuk membangun, mematangkan, dan mengoptimalkan satu atau lebih lingkungan cloud. Untuk informasi selengkapnya, lihat Membangun Model Operasi Cloud Anda.

tahap adopsi cloud

Empat fase yang biasanya dilalui organisasi ketika mereka bermigrasi ke AWS Cloud:

- Proyek — Menjalankan beberapa proyek terkait cloud untuk bukti konsep dan tujuan pembelajaran
- Foundation — Melakukan investasi dasar untuk meningkatkan adopsi cloud Anda (misalnya, membuat landing zone, mendefinisikan CCo E, membuat model operasi)
- Migrasi — Migrasi aplikasi individual
- Re-invention — Mengoptimalkan produk dan layanan, dan berinovasi di cloud

Tahapan ini didefinisikan oleh Stephen Orban dalam posting blog The Journey Toward Cloud-First & the Stages of Adoption di blog Strategi Perusahaan. AWS Cloud Untuk informasi tentang bagaimana kaitannya dengan strategi AWS migrasi, lihat panduan kesiapan migrasi.

CMDB

Lihat database manajemen konfigurasi.

repositori kode

Lokasi di mana kode sumber dan aset lainnya, seperti dokumentasi, sampel, dan skrip, disimpan dan diperbarui melalui proses kontrol versi. Repositori cloud umum termasuk GitHub atau. Bitbucket Cloud Setiap versi kode disebut cabang. Dalam struktur layanan mikro, setiap repositori

dikhususkan untuk satu bagian fungsionalitas. Pipa CI/CD tunggal dapat menggunakan beberapa repositori.

cache dingin

Cache buffer yang kosong, tidak terisi dengan baik, atau berisi data basi atau tidak relevan. Ini mempengaruhi kinerja karena instance database harus membaca dari memori utama atau disk, yang lebih lambat daripada membaca dari cache buffer.

data dingin

Data yang jarang diakses dan biasanya historis. Saat menanyakan jenis data ini, kueri lambat biasanya dapat diterima. Memindahkan data ini ke tingkat penyimpanan atau kelas yang berkinerja lebih rendah dan lebih murah dapat mengurangi biaya.

visi komputer (CV)

Bidang AI yang menggunakan pembelajaran mesin untuk menganalisis dan mengekstrak informasi dari format visual seperti gambar dan video digital. Misalnya, Amazon SageMaker AI menyediakan algoritma pemrosesan gambar untuk CV.

konfigurasi drift

Untuk beban kerja, konfigurasi berubah dari status yang diharapkan. Ini dapat menyebabkan beban kerja menjadi tidak patuh, dan biasanya bertahap dan tidak disengaja.

database manajemen konfigurasi (CMDB)

Repositori yang menyimpan dan mengelola informasi tentang database dan lingkungan TI, termasuk komponen perangkat keras dan perangkat lunak dan konfigurasinya. Anda biasanya menggunakan data dari CMDB dalam penemuan portofolio dan tahap analisis migrasi.

paket kesesuaian

Kumpulan AWS Config aturan dan tindakan remediasi yang dapat Anda kumpulkan untuk menyesuaikan kepatuhan dan pemeriksaan keamanan Anda. Anda dapat menerapkan paket kesesuaian sebagai entitas tunggal di Akun AWS dan Wilayah, atau di seluruh organisasi, dengan menggunakan templat YAMM. Untuk informasi selengkapnya, lihat Paket kesesuaian dalam dokumentasi. AWS Config

integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD)

Proses mengotomatiskan sumber, membangun, menguji, pementasan, dan tahap produksi dari proses rilis perangkat lunak. CI/CD biasanya digambarkan sebagai pipa. CI/CD dapat membantu

Anda mengotomatiskan proses, meningkatkan produktivitas, meningkatkan kualitas kode, dan memberikan lebih cepat. Untuk informasi lebih lanjut, lihat Manfaat pengiriman berkelanjutan. CD juga dapat berarti penerapan berkelanjutan. Untuk informasi selengkapnya, lihat Continuous Delivery vs Continuous Deployment.

CV

Lihat visi komputer.

# D

data saat istirahat

Data yang stasioner di jaringan Anda, seperti data yang ada di penyimpanan.

klasifikasi data

Proses untuk mengidentifikasi dan mengkategorikan data dalam jaringan Anda berdasarkan kekritisan dan sensitivitasnya. Ini adalah komponen penting dari setiap strategi manajemen risiko keamanan siber karena membantu Anda menentukan perlindungan dan kontrol retensi yang tepat untuk data. Klasifikasi data adalah komponen pilar keamanan dalam AWS Well-Architected Framework. Untuk informasi selengkapnya, lihat Klasifikasi data.

penyimpangan data

Variasi yang berarti antara data produksi dan data yang digunakan untuk melatih model ML, atau perubahan yang berarti dalam data input dari waktu ke waktu. Penyimpangan data dapat mengurangi kualitas, akurasi, dan keadilan keseluruhan dalam prediksi model ML.

data dalam transit

Data yang aktif bergerak melalui jaringan Anda, seperti antara sumber daya jaringan.

jala data

Kerangka arsitektur yang menyediakan kepemilikan data terdistribusi dan terdesentralisasi dengan manajemen dan tata kelola terpusat.

minimalisasi data

Prinsip pengumpulan dan pemrosesan hanya data yang sangat diperlukan. Mempraktikkan minimalisasi data di dalamnya AWS Cloud dapat mengurangi risiko privasi, biaya, dan jejak karbon analitik Anda.

perimeter data

Satu set pagar pembatas pencegahan di AWS lingkungan Anda yang membantu memastikan bahwa hanya identitas tepercaya yang mengakses sumber daya tepercaya dari jaringan yang diharapkan. Untuk informasi selengkapnya, lihat Membangun perimeter data pada AWS.

prapemrosesan data

Untuk mengubah data mentah menjadi format yang mudah diuraikan oleh model ML Anda. Preprocessing data dapat berarti menghapus kolom atau baris tertentu dan menangani nilai yang hilang, tidak konsisten, atau duplikat.

asal data

Proses melacak asal dan riwayat data sepanjang siklus hidupnya, seperti bagaimana data dihasilkan, ditransmisikan, dan disimpan.

subjek data

Individu yang datanya dikumpulkan dan diproses.

gudang data

Sistem manajemen data yang mendukung intelijen bisnis, seperti analitik. Gudang data biasanya berisi sejumlah besar data historis, dan biasanya digunakan untuk kueri dan analisis.

bahasa definisi database (DDL)

Pernyataan atau perintah untuk membuat atau memodifikasi struktur tabel dan objek dalam database.

bahasa manipulasi basis data (DHTML)

Pernyataan atau perintah untuk memodifikasi (memasukkan, memperbarui, dan menghapus) informasi dalam database.

DDL

Lihat bahasa definisi database.

ansambel yang dalam

Untuk menggabungkan beberapa model pembelajaran mendalam untuk prediksi. Anda dapat menggunakan ansambel dalam untuk mendapatkan prediksi yang lebih akurat atau untuk memperkirakan ketidakpastian dalam prediksi.

pembelajaran mendalam

Subbidang ML yang menggunakan beberapa lapisan jaringan saraf tiruan untuk mengidentifikasi pemetaan antara data input dan variabel target yang diinginkan.

defense-in-depth

Pendekatan keamanan informasi di mana serangkaian mekanisme dan kontrol keamanan dilapisi dengan cermat di seluruh jaringan komputer untuk melindungi kerahasiaan, integritas, dan ketersediaan jaringan dan data di dalamnya. Saat Anda mengadopsi strategi ini AWS, Anda menambahkan beberapa kontrol pada lapisan AWS Organizations struktur yang berbeda untuk membantu mengamankan sumber daya. Misalnya, defense-in-depth pendekatan mungkin menggabungkan otentikasi multi-faktor, segmentasi jaringan, dan enkripsi.

administrator yang didelegasikan

Di AWS Organizations, layanan yang kompatibel dapat mendaftarkan akun AWS anggota untuk mengelola akun organisasi dan mengelola izin untuk layanan tersebut. Akun ini disebut administrator yang didelegasikan untuk layanan itu. Untuk informasi selengkapnya dan daftar layanan yang kompatibel, lihat Layanan yang berfungsi dengan AWS Organizations AWS Organizations dokumentasi.

deployment

Proses pembuatan aplikasi, fitur baru, atau perbaikan kode tersedia di lingkungan target. Deployment melibatkan penerapan perubahan dalam basis kode dan kemudian membangun dan menjalankan basis kode itu di lingkungan aplikasi.

lingkungan pengembangan

Lihat lingkungan.

kontrol detektif

Kontrol keamanan yang dirancang untuk mendeteksi, mencatat, dan memperingatkan setelah suatu peristiwa terjadi. Kontrol ini adalah garis pertahanan kedua, memperingatkan Anda tentang peristiwa keamanan yang melewati kontrol pencegahan yang ada. Untuk informasi selengkapnya, lihat Kontrol Detektif dalam Menerapkan kontrol keamanan pada. AWS

pemetaan aliran nilai pengembangan (DVSM)

Sebuah proses yang digunakan untuk mengidentifikasi dan memprioritaskan kendala yang mempengaruhi kecepatan dan kualitas dalam siklus hidup pengembangan perangkat lunak. DVSM memperluas proses pemetaan aliran nilai yang awalnya dirancang untuk praktik

manufaktur ramping. Ini berfokus pada langkah-langkah dan tim yang diperlukan untuk menciptakan dan memindahkan nilai melalui proses pengembangan perangkat lunak.

kembar digital

Representasi virtual dari sistem dunia nyata, seperti bangunan, pabrik, peralatan industri, atau jalur produksi. Kembar digital mendukung pemeliharaan prediktif, pemantauan jarak jauh, dan optimalisasi produksi.

tabel dimensi

Dalam skema bintang, tabel yang lebih kecil yang berisi atribut data tentang data kuantitatif dalam tabel fakta. Atribut tabel dimensi biasanya bidang teks atau angka diskrit yang berperilaku seperti teks. Atribut ini biasanya digunakan untuk pembatasan kueri, pemfilteran, dan pelabelan set hasil.

musibah

Peristiwa yang mencegah beban kerja atau sistem memenuhi tujuan bisnisnya di lokasi utama yang digunakan. Peristiwa ini dapat berupa bencana alam, kegagalan teknis, atau akibat dari tindakan manusia, seperti kesalahan konfigurasi yang tidak disengaja atau serangan malware.

pemulihan bencana (DR)

Strategi dan proses yang Anda gunakan untuk meminimalkan downtime dan kehilangan data yang disebabkan oleh bencana. Untuk informasi selengkapnya, lihat Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML~

Lihat bahasa manipulasi basis data.

desain berbasis domain

Pendekatan untuk mengembangkan sistem perangkat lunak yang kompleks dengan menghubungkan komponennya ke domain yang berkembang, atau tujuan bisnis inti, yang dilayani oleh setiap komponen. Konsep ini diperkenalkan oleh Eric Evans dalam bukunya, Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston: Addison-Wesley Professional, 2003). Untuk informasi tentang cara menggunakan desain berbasis domain dengan pola gambar pencekik, lihat Memodernisasi layanan web Microsoft ASP.NET (ASMX) lama secara bertahap menggunakan container dan Amazon API Gateway.

DR

Lihat pemulihan bencana.

deteksi drift

Melacak penyimpangan dari konfigurasi dasar. Misalnya, Anda dapat menggunakan AWS CloudFormation untuk mendeteksi penyimpangan dalam sumber daya sistem, atau Anda dapat menggunakannya AWS Control Tower untuk mendeteksi perubahan di landing zone yang mungkin memengaruhi kepatuhan terhadap persyaratan tata kelola.

DVSM

Lihat pemetaan aliran nilai pengembangan.

# E

EDA

Lihat analisis data eksplorasi.

EDI

Lihat pertukaran data elektronik.

komputasi tepi

Teknologi yang meningkatkan daya komputasi untuk perangkat pintar di tepi jaringan IoT. Jika dibandingkan dengan komputasi awan, komputasi tepi dapat mengurangi latensi komunikasi dan meningkatkan waktu respons.

pertukaran data elektronik (EDI)

Pertukaran otomatis dokumen bisnis antar organisasi. Untuk informasi selengkapnya, lihat Apa itu Pertukaran Data Elektronik.

enkripsi

Proses komputasi yang mengubah data plaintext, yang dapat dibaca manusia, menjadi ciphertext.

kunci enkripsi

String kriptografi dari bit acak yang dihasilkan oleh algoritma enkripsi. Panjang kunci dapat bervariasi, dan setiap kunci dirancang agar tidak dapat diprediksi dan unik.

endianness

Urutan byte disimpan dalam memori komputer. Sistem big-endian menyimpan byte paling signifikan terlebih dahulu. Sistem little-endian menyimpan byte paling tidak signifikan terlebih dahulu.

titik akhir

Lihat titik akhir layanan.

layanan endpoint

Layanan yang dapat Anda host di cloud pribadi virtual (VPC) untuk dibagikan dengan pengguna lain. Anda dapat membuat layanan endpoint dengan AWS PrivateLink dan memberikan izin kepada prinsipal lain Akun AWS atau ke AWS Identity and Access Management (IAM). Akun atau prinsipal ini dapat terhubung ke layanan endpoint Anda secara pribadi dengan membuat titik akhir VPC antarmuka. Untuk informasi selengkapnya, lihat Membuat layanan titik akhir di dokumentasi Amazon Virtual Private Cloud (Amazon VPC).

perencanaan sumber daya perusahaan (ERP)

Sistem yang mengotomatiskan dan mengelola proses bisnis utama (seperti akuntansi, MES, dan manajemen proyek) untuk suatu perusahaan.

enkripsi amplop

Proses mengenkripsi kunci enkripsi dengan kunci enkripsi lain. Untuk informasi selengkapnya, lihat Enkripsi amplop dalam dokumentasi AWS Key Management Service (AWS KMS).

lingkungan

Sebuah contoh dari aplikasi yang sedang berjalan. Berikut ini adalah jenis lingkungan yang umum dalam komputasi awan:

- Development Environment — Sebuah contoh dari aplikasi yang berjalan yang hanya tersedia untuk tim inti yang bertanggung jawab untuk memelihara aplikasi. Lingkungan pengembangan digunakan untuk menguji perubahan sebelum mempromosikannya ke lingkungan atas. Jenis lingkungan ini kadang-kadang disebut sebagai lingkungan pengujian.
- lingkungan yang lebih rendah — Semua lingkungan pengembangan untuk aplikasi, seperti yang digunakan untuk build awal dan pengujian.
- lingkungan produksi — Sebuah contoh dari aplikasi yang berjalan yang dapat diakses oleh pengguna akhir. Dalam sebuah CI/CD pipeline, lingkungan produksi adalah lingkungan penyebaran terakhir.
- lingkungan atas — Semua lingkungan yang dapat diakses oleh pengguna selain tim pengembangan inti. Ini dapat mencakup lingkungan produksi, lingkungan praproduksi, dan lingkungan untuk pengujian penerimaan pengguna.

epik

Dalam metodologi tangkas, kategori fungsional yang membantu mengatur dan memprioritaskan pekerjaan Anda. Epik memberikan deskripsi tingkat tinggi tentang persyaratan dan tugas implementasi. Misalnya, epos keamanan AWS CAF mencakup manajemen identitas dan akses, kontrol detektif, keamanan infrastruktur, perlindungan data, dan respons insiden. Untuk informasi selengkapnya tentang epos dalam strategi AWS migrasi, lihat panduan implementasi program.

ERP

Lihat perencanaan sumber daya perusahaan.

analisis data eksplorasi (EDA)

Proses menganalisis dataset untuk memahami karakteristik utamanya. Anda mengumpulkan atau mengumpulkan data dan kemudian melakukan penyelidikan awal untuk menemukan pola, mendeteksi anomali, dan memeriksa asumsi. EDA dilakukan dengan menghitung statistik ringkasan dan membuat visualisasi data.

# F

tabel fakta

Tabel tengah dalam skema bintang. Ini menyimpan data kuantitatif tentang operasi bisnis. Biasanya, tabel fakta berisi dua jenis kolom: kolom yang berisi ukuran dan yang berisi kunci asing ke tabel dimensi.

gagal cepat

Filosofi yang menggunakan pengujian yang sering dan bertahap untuk mengurangi siklus hidup pengembangan. Ini adalah bagian penting dari pendekatan tangkas.

batas isolasi kesalahan

Dalam AWS Cloud, batas seperti Availability Zone, Wilayah AWS, control plane, atau data plane yang membatasi efek kegagalan dan membantu meningkatkan ketahanan beban kerja. Untuk informasi selengkapnya, lihat Batas Isolasi AWS Kesalahan.

cabang fitur

Lihat cabang.

fitur

Data input yang Anda gunakan untuk membuat prediksi. Misalnya, dalam konteks manufaktur, fitur bisa berupa gambar yang diambil secara berkala dari lini manufaktur.

pentingnya fitur

Seberapa signifikan fitur untuk prediksi model. Ini biasanya dinyatakan sebagai skor numerik yang dapat dihitung melalui berbagai teknik, seperti Shapley Additive Explanations (SHAP) dan gradien terintegrasi. Untuk informasi lebih lanjut, lihat Interpretabilitas model pembelajaran mesin dengan. AWS

transformasi fitur

Untuk mengoptimalkan data untuk proses ML, termasuk memperkaya data dengan sumber tambahan, menskalakan nilai, atau mengekstrak beberapa set informasi dari satu bidang data. Hal ini memungkinkan model ML untuk mendapatkan keuntungan dari data. Misalnya, jika Anda memecah tanggal “2021-05-27 00:15:37" menjadi “2021", “Mei”, “Kamis”, dan “15", Anda dapat membantu algoritme pembelajaran mempelajari pola bernuansa yang terkait dengan komponen data yang berbeda.

beberapa tembakan mendorong

Menyediakan LLM dengan sejumlah kecil contoh yang menunjukkan tugas dan output yang diinginkan sebelum memintanya untuk melakukan tugas serupa. Teknik ini adalah aplikasi pembelajaran dalam konteks, di mana model belajar dari contoh (bidikan) yang tertanam dalam petunjuk. Beberapa bidikan dapat efektif untuk tugas-tugas yang memerlukan pemformatan, penalaran, atau pengetahuan domain tertentu. Lihat juga bidikan nol.

FGAC

Lihat kontrol akses berbutir halus.

kontrol akses berbutir halus (FGAC)

Penggunaan beberapa kondisi untuk mengizinkan atau menolak permintaan akses.

migrasi flash-cut

Metode migrasi database yang menggunakan replikasi data berkelanjutan melalui pengambilan data perubahan untuk memigrasikan data dalam waktu sesingkat mungkin, alih-alih menggunakan pendekatan bertahap. Tujuannya adalah untuk menjaga downtime seminimal mungkin.

FM

Lihat model pondasi.

model pondasi (FM)

Jaringan saraf pembelajaran mendalam yang besar yang telah melatih kumpulan data besar-besaran data umum dan tidak berlabel. FMs mampu melakukan berbagai tugas umum, seperti memahami bahasa, menghasilkan teks dan gambar, dan berbicara dalam bahasa alami. Untuk informasi selengkapnya, lihat Apa itu Model Foundation.

# G

AI generatif

Subset model AI yang telah dilatih pada sejumlah besar data dan yang dapat menggunakan prompt teks sederhana untuk membuat konten dan artefak baru, seperti gambar, video, teks, dan audio. Untuk informasi lebih lanjut, lihat Apa itu AI Generatif.

pemblokiran geografis

Lihat pembatasan geografis.

pembatasan geografis (pemblokiran geografis)

Di Amazon CloudFront, opsi untuk mencegah pengguna di negara tertentu mengakses distribusi konten. Anda dapat menggunakan daftar izinkan atau daftar blokir untuk menentukan negara yang disetujui dan dilarang. Untuk informasi selengkapnya, lihat Membatasi distribusi geografis konten Anda dalam dokumentasi. CloudFront

Alur kerja Gitflow

Pendekatan di mana lingkungan bawah dan atas menggunakan cabang yang berbeda dalam repositori kode sumber. Alur kerja Gitflow dianggap warisan, dan alur kerja berbasis batang adalah pendekatan modern yang lebih disukai.

gambar emas

Sebuah snapshot dari sistem atau perangkat lunak yang digunakan sebagai template untuk menyebarkan instance baru dari sistem atau perangkat lunak itu. Misalnya, di bidang manufaktur, gambar emas dapat digunakan untuk menyediakan perangkat lunak pada beberapa perangkat dan membantu meningkatkan kecepatan, skalabilitas, dan produktivitas dalam operasi manufaktur perangkat.

strategi greenfield

Tidak adanya infrastruktur yang ada di lingkungan baru. Saat mengadopsi strategi greenfield untuk arsitektur sistem, Anda dapat memilih semua teknologi baru tanpa batasan kompatibilitas dengan infrastruktur yang ada, juga dikenal sebagai brownfield. Jika Anda memperluas infrastruktur yang ada, Anda dapat memadukan strategi brownfield dan greenfield.

pagar pembatas

Aturan tingkat tinggi yang membantu mengatur sumber daya, kebijakan, dan kepatuhan di seluruh unit organisasi ()OUs. Pagar pembatas preventif menegakkan kebijakan untuk memastikan keselarasan dengan standar kepatuhan. Mereka diimplementasikan dengan menggunakan kebijakan kontrol layanan dan batas izin IAM. Detective guardrails mendeteksi pelanggaran kebijakan dan masalah kepatuhan, dan menghasilkan peringatan untuk remediasi. Mereka diimplementasikan dengan menggunakan AWS Config, AWS Security Hub CSPM, Amazon GuardDuty AWS Trusted Advisor, Amazon Inspector, dan pemeriksaan khusus AWS Lambda .

# H

HA

Lihat ketersediaan tinggi.

migrasi database heterogen

Memigrasi database sumber Anda ke database target yang menggunakan mesin database yang berbeda (misalnya, Oracle ke Amazon Aurora). Migrasi heterogen biasanya merupakan bagian dari upaya arsitektur ulang, dan mengubah skema dapat menjadi tugas yang kompleks. AWS menyediakan AWS SCT yang membantu dengan konversi skema.

ketersediaan tinggi (HA)

Kemampuan beban kerja untuk beroperasi terus menerus, tanpa intervensi, jika terjadi tantangan atau bencana. Sistem HA dirancang untuk gagal secara otomatis, secara konsisten memberikan kinerja berkualitas tinggi, dan menangani beban dan kegagalan yang berbeda dengan dampak kinerja minimal.

modernisasi sejarawan

Pendekatan yang digunakan untuk memodernisasi dan meningkatkan sistem teknologi operasional (OT) untuk melayani kebutuhan industri manufaktur dengan lebih baik. Sejarawan

adalah jenis database yang digunakan untuk mengumpulkan dan menyimpan data dari berbagai sumber di pabrik.

data penahanan

Sebagian dari data historis berlabel yang ditahan dari kumpulan data yang digunakan untuk melatih model pembelajaran mesin. Anda dapat menggunakan data penahanan untuk mengevaluasi kinerja model dengan membandingkan prediksi model dengan data penahanan.

migrasi database homogen

Memigrasi database sumber Anda ke database target yang berbagi mesin database yang sama (misalnya, Microsoft SQL Server ke Amazon RDS for SQL Server). Migrasi homogen biasanya merupakan bagian dari upaya rehosting atau replatforming. Anda dapat menggunakan utilitas database asli untuk memigrasi skema.

data panas

Data yang sering diakses, seperti data real-time atau data translasi terbaru. Data ini biasanya memerlukan tingkat atau kelas penyimpanan berkinerja tinggi untuk memberikan respons kueri yang cepat.

perbaikan terbaru

Perbaikan mendesak untuk masalah kritis dalam lingkungan produksi. Karena urgensinya, perbaikan terbaru biasanya dibuat di luar alur kerja DevOps rilis biasa.

periode hypercare

Segera setelah cutover, periode waktu ketika tim migrasi mengelola dan memantau aplikasi yang dimigrasi di cloud untuk mengatasi masalah apa pun. Biasanya, periode ini panjangnya 1-4 hari. Pada akhir periode hypercare, tim migrasi biasanya mentransfer tanggung jawab untuk aplikasi ke tim operasi cloud.

## I

IAc

Lihat infrastruktur sebagai kode.

kebijakan berbasis identitas

Kebijakan yang dilampirkan pada satu atau beberapa prinsip IAM yang mendefinisikan izin mereka dalam lingkungan. AWS Cloud

aplikasi idle

Aplikasi yang memiliki penggunaan CPU dan memori rata-rata antara 5 dan 20 persen selama periode 90 hari. Dalam proyek migrasi, adalah umum untuk menghentikan aplikasi ini atau mempertahankannya di tempat.

IIoT

Lihat Internet of Things industri.

infrastruktur yang tidak dapat diubah

Model yang menyebarkan infrastruktur baru untuk beban kerja produksi alih-alih memperbarui, menambal, atau memodifikasi infrastruktur yang ada. Infrastruktur yang tidak dapat diubah secara inheren lebih konsisten, andal, dan dapat diprediksi daripada infrastruktur yang dapat berubah. Untuk informasi selengkapnya, lihat praktik terbaik Deploy using immutable infrastructure di AWS Well-Architected Framework.

masuk (masuknya) VPC

Dalam arsitektur AWS multi-akun, VPC yang menerima, memeriksa, dan merutekan koneksi jaringan dari luar aplikasi. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan inbound, outbound, dan inspeksi VPCs untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

migrasi inkremental

Strategi cutover di mana Anda memigrasikan aplikasi Anda dalam bagian-bagian kecil alih-alih melakukan satu cutover penuh. Misalnya, Anda mungkin hanya memindahkan beberapa layanan mikro atau pengguna ke sistem baru pada awalnya. Setelah Anda memverifikasi bahwa semuanya berfungsi dengan baik, Anda dapat secara bertahap memindahkan layanan mikro atau pengguna tambahan hingga Anda dapat menonaktifkan sistem lama Anda. Strategi ini mengurangi risiko yang terkait dengan migrasi besar.

Industri 4.0

Sebuah istilah yang diperkenalkan oleh Klaus Schwab pada tahun 2016 untuk merujuk pada modernisasi proses manufaktur melalui kemajuan dalam konektivitas, data real-time, otomatisasi, analitik, dan AI/ML.

infrastruktur

Semua sumber daya dan aset yang terkandung dalam lingkungan aplikasi.

infrastruktur sebagai kode (IAc)

Proses penyediaan dan pengelolaan infrastruktur aplikasi melalui satu set file konfigurasi. IAc dirancang untuk membantu Anda memusatkan manajemen infrastruktur, menstandarisasi sumber daya, dan menskalakan dengan cepat sehingga lingkungan baru dapat diulang, andal, dan konsisten.

Internet of Things industri (IIoT)

Penggunaan sensor dan perangkat yang terhubung ke internet di sektor industri, seperti manufaktur, energi, otomotif, perawatan kesehatan, ilmu kehidupan, dan pertanian. Untuk informasi lebih lanjut, lihat Membangun strategi transformasi digital Internet of Things (IIoT) industri.

inspeksi VPC

Dalam arsitektur AWS multi-akun, VPC terpusat yang mengelola inspeksi lalu lintas jaringan antara VPCs (dalam yang sama atau berbeda Wilayah AWS), internet, dan jaringan lokal. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan inbound, outbound, dan inspeksi VPCs untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

Internet of Things (IoT)

Jaringan objek fisik yang terhubung dengan sensor atau prosesor tertanam yang berkomunikasi dengan perangkat dan sistem lain melalui internet atau melalui jaringan komunikasi lokal. Untuk informasi selengkapnya, lihat Apa itu IoT?

interpretabilitas

Karakteristik model pembelajaran mesin yang menggambarkan sejauh mana manusia dapat memahami bagaimana prediksi model bergantung pada inputnya. Untuk informasi lebih lanjut, lihat Interpretabilitas model pembelajaran mesin dengan. AWS

IoT

Lihat Internet of Things.

Perpustakaan informasi TI (ITIL)

Serangkaian praktik terbaik untuk memberikan layanan TI dan menyelaraskan layanan ini dengan persyaratan bisnis. ITIL menyediakan dasar untuk ITSM.

Manajemen layanan TI (ITSM)

Kegiatan yang terkait dengan merancang, menerapkan, mengelola, dan mendukung layanan TI untuk suatu organisasi. Untuk informasi tentang mengintegrasikan operasi cloud dengan alat ITSM, lihat panduan integrasi operasi.

ITIL

Lihat perpustakaan informasi TI.

ITSM

Lihat manajemen layanan TI.

# L

kontrol akses berbasis label (LBAC)

Implementasi kontrol akses wajib (MAC) di mana pengguna dan data itu sendiri masing-masing secara eksplisit diberi nilai label keamanan. Persimpangan antara label keamanan pengguna dan label keamanan data menentukan baris dan kolom mana yang dapat dilihat oleh pengguna.

landing zone

Landing zone adalah AWS lingkungan multi-akun yang dirancang dengan baik yang dapat diskalakan dan aman. Ini adalah titik awal dari mana organisasi Anda dapat dengan cepat meluncurkan dan menyebarkan beban kerja dan aplikasi dengan percaya diri dalam lingkungan keamanan dan infrastruktur mereka. Untuk informasi selengkapnya tentang zona pendaratan, lihat Menyiapkan lingkungan multi-akun AWS yang aman dan dapat diskalakan.

model bahasa besar (LLM)

Model AI pembelajaran mendalam yang dilatih sebelumnya pada sejumlah besar data. LLM dapat melakukan beberapa tugas, seperti menjawab pertanyaan, meringkas dokumen, menerjemahkan teks ke dalam bahasa lain, dan menyelesaikan kalimat. Untuk informasi lebih lanjut, lihat Apa itu LLMs.

migrasi besar

Migrasi 300 atau lebih server.

LBAC

Lihat kontrol akses berbasis label.

hak istimewa paling sedikit

Praktik keamanan terbaik untuk memberikan izin minimum yang diperlukan untuk melakukan tugas. Untuk informasi selengkapnya, lihat Menerapkan izin hak istimewa terkecil dalam dokumentasi IAM.

angkat dan geser

Lihat 7 Rs.

sistem endian kecil

Sebuah sistem yang menyimpan byte paling tidak signifikan terlebih dahulu. Lihat juga endianness.

LLM

Lihat model bahasa besar.

lingkungan yang lebih rendah

Lihat lingkungan.

# M

pembelajaran mesin (ML)

Jenis kecerdasan buatan yang menggunakan algoritma dan teknik untuk pengenalan dan pembelajaran pola. ML menganalisis dan belajar dari data yang direkam, seperti data Internet of Things (IoT), untuk menghasilkan model statistik berdasarkan pola. Untuk informasi selengkapnya, lihat Machine Learning.

cabang utama

Lihat cabang.

malware

Perangkat lunak yang dirancang untuk membahayakan keamanan atau privasi komputer. Malware dapat mengganggu sistem komputer, membocorkan informasi sensitif, atau mendapatkan akses yang tidak sah. Contoh malware termasuk virus, worm, ransomware, Trojan horse, spyware, dan keyloggers.

layanan terkelola

Layanan AWS yang AWS mengoperasikan lapisan infrastruktur, sistem operasi, dan platform, dan Anda mengakses titik akhir untuk menyimpan dan mengambil data. Amazon Simple Storage Service (Amazon S3) dan Amazon DynamoDB adalah contoh layanan terkelola. Ini juga dikenal sebagai layanan abstrak.

sistem eksekusi manufaktur (MES)

Sistem perangkat lunak untuk melacak, memantau, mendokumentasikan, dan mengendalikan proses produksi yang mengubah bahan baku menjadi produk jadi di lantai toko.

PETA

Lihat Program Percepatan Migrasi.

mekanisme

Proses lengkap di mana Anda membuat alat, mendorong adopsi alat, dan kemudian memeriksa hasilnya untuk melakukan penyesuaian. Mekanisme adalah siklus yang memperkuat dan meningkatkan dirinya sendiri saat beroperasi. Untuk informasi lebih lanjut, lihat Membangun mekanisme di AWS Well-Architected Framework.

akun anggota

Semua Akun AWS selain akun manajemen yang merupakan bagian dari organisasi di AWS Organizations. Akun dapat menjadi anggota dari hanya satu organisasi pada suatu waktu.

MES

Lihat sistem eksekusi manufaktur.

Transportasi Telemetri Antrian Pesan (MQTT)

Protokol komunikasi ringan machine-to-machine (M2M), berdasarkan pola terbitkan/berlangganan, untuk perangkat IoT yang dibatasi sumber daya.

layanan mikro

Layanan kecil dan independen yang berkomunikasi dengan jelas APIs dan biasanya dimiliki oleh tim kecil yang mandiri. Misalnya, sistem asuransi mungkin mencakup layanan mikro yang memetakan kemampuan bisnis, seperti penjualan atau pemasaran, atau subdomain, seperti pembelian, klaim, atau analitik. Manfaat layanan mikro termasuk kelincahan, penskalaan yang fleksibel, penyebaran yang mudah, kode yang dapat digunakan kembali, dan ketahanan. Untuk

informasi selengkapnya, lihat Mengintegrasikan layanan mikro dengan menggunakan layanan tanpa AWS server.

arsitektur microservices

Pendekatan untuk membangun aplikasi dengan komponen independen yang menjalankan setiap proses aplikasi sebagai layanan mikro. Layanan mikro ini berkomunikasi melalui antarmuka yang terdefinisi dengan baik dengan menggunakan ringan. APIs Setiap layanan mikro dalam arsitektur ini dapat diperbarui, digunakan, dan diskalakan untuk memenuhi permintaan fungsi tertentu dari suatu aplikasi. Untuk informasi selengkapnya, lihat Menerapkan layanan mikro di AWS.

Program Percepatan Migrasi (MAP)

AWS Program yang menyediakan dukungan konsultasi, pelatihan, dan layanan untuk membantu organisasi membangun fondasi operasional yang kuat untuk pindah ke cloud, dan untuk membantu mengimbangi biaya awal migrasi. MAP mencakup metodologi migrasi untuk mengeksekusi migrasi lama dengan cara metodis dan seperangkat alat untuk mengotomatisasi dan mempercepat skenario migrasi umum.

migrasi dalam skala

Proses memindahkan sebagian besar portofolio aplikasi ke cloud dalam gelombang, dengan lebih banyak aplikasi bergerak pada tingkat yang lebih cepat di setiap gelombang. Fase ini menggunakan praktik dan pelajaran terbaik dari fase sebelumnya untuk mengimplementasikan pabrik migrasi tim, alat, dan proses untuk merampingkan migrasi beban kerja melalui otomatisasi dan pengiriman tangkas. Ini adalah fase ketiga dari strategi AWS migrasi.

pabrik migrasi

Tim lintas fungsi yang merampingkan migrasi beban kerja melalui pendekatan otomatis dan gesit. Tim pabrik migrasi biasanya mencakup operasi, analis dan pemilik bisnis, insinyur migrasi, pengembang, dan DevOps profesional yang bekerja di sprint. Antara 20 dan 50 persen portofolio aplikasi perusahaan terdiri dari pola berulang yang dapat dioptimalkan dengan pendekatan pabrik. Untuk informasi selengkapnya, lihat diskusi tentang pabrik migrasi dan panduan Pabrik Migrasi Cloud di kumpulan konten ini.

metadata migrasi

Informasi tentang aplikasi dan server yang diperlukan untuk menyelesaikan migrasi. Setiap pola migrasi memerlukan satu set metadata migrasi yang berbeda. Contoh metadata migrasi termasuk subnet target, grup keamanan, dan akun. AWS

pola migrasi

Tugas migrasi berulang yang merinci strategi migrasi, tujuan migrasi, dan aplikasi atau layanan migrasi yang digunakan. Contoh: Rehost migrasi ke Amazon EC2 AWS dengan Layanan Migrasi Aplikasi.

Penilaian Portofolio Migrasi (MPA)

Alat online yang menyediakan informasi untuk memvalidasi kasus bisnis untuk bermigrasi ke. AWS Cloud MPA menyediakan penilaian portofolio terperinci (ukuran kanan server, harga, perbandingan TCO, analisis biaya migrasi) serta perencanaan migrasi (analisis data aplikasi dan pengumpulan data, pengelompokan aplikasi, prioritas migrasi, dan perencanaan gelombang). Alat MPA (memerlukan login) tersedia gratis untuk semua AWS konsultan dan konsultan APN Partner.

Penilaian Kesiapan Migrasi (MRA)

Proses mendapatkan wawasan tentang status kesiapan cloud organisasi, mengidentifikasi kekuatan dan kelemahan, dan membangun rencana aksi untuk menutup kesenjangan yang diidentifikasi, menggunakan CAF. AWS Untuk informasi selengkapnya, lihat panduan kesiapan migrasi. MRA adalah tahap pertama dari strategi AWS migrasi.

strategi migrasi

Pendekatan yang digunakan untuk memigrasikan beban kerja ke. AWS Cloud Untuk informasi lebih lanjut, lihat entri 7 Rs di glosarium ini dan lihat Memobilisasi organisasi Anda untuk mempercepat migrasi skala besar.

ML

Lihat pembelajaran mesin.

modernisasi

Mengubah aplikasi usang (warisan atau monolitik) dan infrastrukturnya menjadi sistem yang gesit, elastis, dan sangat tersedia di cloud untuk mengurangi biaya, mendapatkan efisiensi, dan memanfaatkan inovasi. Untuk informasi selengkapnya, lihat Strategi untuk memodernisasi aplikasi di. AWS Cloud

penilaian kesiapan modernisasi

Evaluasi yang membantu menentukan kesiapan modernisasi aplikasi organisasi; mengidentifikasi manfaat, risiko, dan dependensi; dan menentukan seberapa baik organisasi dapat mendukung keadaan masa depan aplikasi tersebut. Hasil penilaian adalah cetak biru arsitektur target, peta

jalan yang merinci fase pengembangan dan tonggak untuk proses modernisasi, dan rencana aksi untuk mengatasi kesenjangan yang diidentifikasi. Untuk informasi lebih lanjut, lihat Mengevaluasi kesiapan modernisasi untuk aplikasi di. AWS Cloud

aplikasi monolitik (monolit)

Aplikasi yang berjalan sebagai layanan tunggal dengan proses yang digabungkan secara ketat. Aplikasi monolitik memiliki beberapa kelemahan. Jika satu fitur aplikasi mengalami lonjakan permintaan, seluruh arsitektur harus diskalakan. Menambahkan atau meningkatkan fitur aplikasi monolitik juga menjadi lebih kompleks ketika basis kode tumbuh. Untuk mengatasi masalah ini, Anda dapat menggunakan arsitektur microservices. Untuk informasi lebih lanjut, lihat Mengurai monolit menjadi layanan mikro.

MPA

Lihat Penilaian Portofolio Migrasi.

MQTT

Lihat Transportasi Telemetri Antrian Pesan.

klasifikasi multiclass

Sebuah proses yang membantu menghasilkan prediksi untuk beberapa kelas (memprediksi satu dari lebih dari dua hasil). Misalnya, model ML mungkin bertanya "Apakah produk ini buku, mobil, atau telepon?" atau "Kategori produk mana yang paling menarik bagi pelanggan ini?"

infrastruktur yang bisa berubah

Model yang memperbarui dan memodifikasi infrastruktur yang ada untuk beban kerja produksi. Untuk meningkatkan konsistensi, keandalan, dan prediktabilitas, AWS Well-Architected Framework merekomendasikan penggunaan infrastruktur yang tidak dapat diubah sebagai praktik terbaik.

# O

OAC

Lihat kontrol akses asal.

OAI

Lihat identitas akses asal.

OCM

Lihat manajemen perubahan organisasi.

migrasi offline

Metode migrasi di mana beban kerja sumber diturunkan selama proses migrasi. Metode ini melibatkan waktu henti yang diperpanjang dan biasanya digunakan untuk beban kerja kecil dan tidak kritis.

OI

Lihat integrasi operasi.

OLA

Lihat perjanjian tingkat operasional.

migrasi online

Metode migrasi di mana beban kerja sumber disalin ke sistem target tanpa diambil offline. Aplikasi yang terhubung ke beban kerja dapat terus berfungsi selama migrasi. Metode ini melibatkan waktu henti nol hingga minimal dan biasanya digunakan untuk beban kerja produksi yang kritis.

OPC-UA

Lihat Komunikasi Proses Terbuka - Arsitektur Terpadu.

Komunikasi Proses Terbuka - Arsitektur Terpadu (OPC-UA)

Protokol komunikasi machine-to-machine (M2M) untuk otomasi industri. OPC-UA menyediakan standar interoperabilitas dengan enkripsi data, otentikasi, dan skema otorisasi.

perjanjian tingkat operasional (OLA)

Perjanjian yang menjelaskan apa yang dijanjikan kelompok TI fungsional untuk diberikan satu sama lain, untuk mendukung perjanjian tingkat layanan (SLA).

Tinjauan Kesiapan Operasional (ORR)

Daftar pertanyaan dan praktik terbaik terkait yang membantu Anda memahami, mengevaluasi, mencegah, atau mengurangi ruang lingkup insiden dan kemungkinan kegagalan. Untuk informasi lebih lanjut, lihat Ulasan Kesiapan Operasional (ORR) dalam Kerangka Kerja Well-Architected AWS .

teknologi operasional (OT)

Sistem perangkat keras dan perangkat lunak yang bekerja dengan lingkungan fisik untuk mengendalikan operasi industri, peralatan, dan infrastruktur. Di bidang manufaktur, integrasi sistem OT dan teknologi informasi (TI) adalah fokus utama untuk transformasi Industri 4.0.

integrasi operasi (OI)

Proses modernisasi operasi di cloud, yang melibatkan perencanaan kesiapan, otomatisasi, dan integrasi. Untuk informasi selengkapnya, lihat panduan integrasi operasi.

jejak organisasi

Jejak yang dibuat oleh AWS CloudTrail itu mencatat semua peristiwa untuk semua Akun AWS dalam organisasi di AWS Organizations. Jejak ini dibuat di setiap Akun AWS bagian organisasi dan melacak aktivitas di setiap akun. Untuk informasi selengkapnya, lihat Membuat jejak untuk organisasi dalam CloudTrail dokumentasi.

manajemen perubahan organisasi (OCM)

Kerangka kerja untuk mengelola transformasi bisnis utama yang mengganggu dari perspektif orang, budaya, dan kepemimpinan. OCM membantu organisasi mempersiapkan, dan transisi ke, sistem dan strategi baru dengan mempercepat adopsi perubahan, mengatasi masalah transisi, dan mendorong perubahan budaya dan organisasi. Dalam strategi AWS migrasi, kerangka kerja ini disebut percepatan orang, karena kecepatan perubahan yang diperlukan dalam proyek adopsi cloud. Untuk informasi lebih lanjut, lihat panduan OCM.

kontrol akses asal (OAC)

Di CloudFront, opsi yang disempurnakan untuk membatasi akses untuk mengamankan konten Amazon Simple Storage Service (Amazon S3) Anda. OAC mendukung semua bucket S3 di semua Wilayah AWS, enkripsi sisi server dengan AWS KMS (SSE-KMS), dan dinamis dan permintaan ke bucket S3. `PUT` `DELETE`

identitas akses asal (OAI)

Di CloudFront, opsi untuk membatasi akses untuk mengamankan konten Amazon S3 Anda. Saat Anda menggunakan OAI, CloudFront buat prinsipal yang dapat diautentikasi oleh Amazon S3. Prinsipal yang diautentikasi dapat mengakses konten dalam bucket S3 hanya melalui distribusi tertentu. CloudFront Lihat juga OAC, yang menyediakan kontrol akses yang lebih terperinci dan ditingkatkan.

ORR

Lihat tinjauan kesiapan operasional.

OT

Lihat teknologi operasional.

keluar (jalan keluar) VPC

Dalam arsitektur AWS multi-akun, VPC yang menangani koneksi jaringan yang dimulai dari dalam aplikasi. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan inbound, outbound, dan inspeksi VPCs untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

# P

batas izin

Kebijakan manajemen IAM yang dilampirkan pada prinsipal IAM untuk menetapkan izin maksimum yang dapat dimiliki pengguna atau peran. Untuk informasi selengkapnya, lihat Batas izin dalam dokumentasi IAM.

Informasi Identifikasi Pribadi (PII)

Informasi yang, jika dilihat secara langsung atau dipasangkan dengan data terkait lainnya, dapat digunakan untuk menyimpulkan identitas individu secara wajar. Contoh PII termasuk nama, alamat, dan informasi kontak.

PII

Lihat informasi yang dapat diidentifikasi secara pribadi.

buku pedoman

Serangkaian langkah yang telah ditentukan sebelumnya yang menangkap pekerjaan yang terkait dengan migrasi, seperti mengirimkan fungsi operasi inti di cloud. Buku pedoman dapat berupa skrip, runbook otomatis, atau ringkasan proses atau langkah-langkah yang diperlukan untuk mengoperasikan lingkungan modern Anda.

PLC

Lihat pengontrol logika yang dapat diprogram.

PLM

Lihat manajemen siklus hidup produk.

kebijakan

Objek yang dapat menentukan izin (lihat kebijakan berbasis identitas), menentukan kondisi akses (lihat kebijakan berbasis sumber daya), atau menentukan izin maksimum untuk semua akun di organisasi (lihat kebijakan kontrol layanan). AWS Organizations

ketekunan poliglot

Secara independen memilih teknologi penyimpanan data microservice berdasarkan pola akses data dan persyaratan lainnya. Jika layanan mikro Anda memiliki teknologi penyimpanan data yang sama, mereka dapat menghadapi tantangan implementasi atau mengalami kinerja yang buruk. Layanan mikro lebih mudah diimplementasikan dan mencapai kinerja dan skalabilitas yang lebih baik jika mereka menggunakan penyimpanan data yang paling sesuai dengan kebutuhan mereka.

penilaian portofolio

Proses menemukan, menganalisis, dan memprioritaskan portofolio aplikasi untuk merencanakan migrasi. Untuk informasi selengkapnya, lihat Mengevaluasi kesiapan migrasi.

predikat

Kondisi kueri yang mengembalikan `true` atau`false`, biasanya terletak di `WHERE` klausa.

predikat pushdown

Teknik pengoptimalan kueri database yang menyaring data dalam kueri sebelum transfer. Ini mengurangi jumlah data yang harus diambil dan diproses dari database relasional, dan meningkatkan kinerja kueri.

kontrol preventif

Kontrol keamanan yang dirancang untuk mencegah suatu peristiwa terjadi. Kontrol ini adalah garis pertahanan pertama untuk membantu mencegah akses tidak sah atau perubahan yang tidak diinginkan ke jaringan Anda. Untuk informasi selengkapnya, lihat Kontrol pencegahan dalam Menerapkan kontrol keamanan pada. AWS

principal

Entitas AWS yang dapat melakukan tindakan dan mengakses sumber daya. Entitas ini biasanya merupakan pengguna root untuk Akun AWS, peran IAM, atau pengguna. Untuk informasi selengkapnya, lihat Prinsip dalam istilah dan konsep Peran dalam dokumentasi IAM.

privasi berdasarkan desain

Pendekatan rekayasa sistem yang memperhitungkan privasi melalui seluruh proses pengembangan.

zona yang dihosting pribadi

Container yang menyimpan informasi tentang bagaimana Anda ingin Amazon Route 53 merespons kueri DNS untuk domain dan subdomainnya dalam satu atau lebih. VPCs Untuk informasi selengkapnya, lihat Bekerja dengan zona yang dihosting pribadi di dokumentasi Route 53.

kontrol proaktif

Kontrol keamanan yang dirancang untuk mencegah penyebaran sumber daya yang tidak sesuai. Kontrol ini memindai sumber daya sebelum disediakan. Jika sumber daya tidak sesuai dengan kontrol, maka itu tidak disediakan. Untuk informasi selengkapnya, lihat panduan referensi Kontrol dalam AWS Control Tower dokumentasi dan lihat Kontrol proaktif dalam Menerapkan kontrol keamanan pada AWS.

manajemen siklus hidup produk (PLM)

Manajemen data dan proses untuk suatu produk di seluruh siklus hidupnya, mulai dari desain, pengembangan, dan peluncuran, melalui pertumbuhan dan kematangan, hingga penurunan dan penghapusan.

lingkungan produksi

Lihat lingkungan.

pengontrol logika yang dapat diprogram (PLC)

Di bidang manufaktur, komputer yang sangat andal dan mudah beradaptasi yang memantau mesin dan mengotomatiskan proses manufaktur.

rantai cepat

Menggunakan output dari satu prompt LLM sebagai input untuk prompt berikutnya untuk menghasilkan respons yang lebih baik. Teknik ini digunakan untuk memecah tugas yang kompleks menjadi subtugas, atau untuk secara iteratif memperbaiki atau memperluas respons awal. Ini membantu meningkatkan akurasi dan relevansi respons model dan memungkinkan hasil yang lebih terperinci dan dipersonalisasi.

pseudonimisasi

Proses penggantian pengidentifikasi pribadi dalam kumpulan data dengan nilai placeholder. Pseudonimisasi dapat membantu melindungi privasi pribadi. Data pseudonim masih dianggap sebagai data pribadi.

publish/subscribe (pub/sub)

Pola yang memungkinkan komunikasi asinkron antara layanan mikro untuk meningkatkan skalabilitas dan daya tanggap. Misalnya, dalam MES berbasis layanan mikro, layanan mikro dapat mempublikasikan pesan peristiwa ke saluran yang dapat berlangganan layanan mikro lainnya. Sistem dapat menambahkan layanan mikro baru tanpa mengubah layanan penerbitan.

# Q

rencana kueri

Serangkaian langkah, seperti instruksi, yang digunakan untuk mengakses data dalam sistem database relasional SQL.

regresi rencana kueri

Ketika pengoptimal layanan database memilih rencana yang kurang optimal daripada sebelum perubahan yang diberikan ke lingkungan database. Hal ini dapat disebabkan oleh perubahan statistik, kendala, pengaturan lingkungan, pengikatan parameter kueri, dan pembaruan ke mesin database.

# R

Matriks RACI

Lihat bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI).

LAP

Lihat Retrieval Augmented Generation.

ransomware

Perangkat lunak berbahaya yang dirancang untuk memblokir akses ke sistem komputer atau data sampai pembayaran dilakukan.

Matriks RASCI

Lihat bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI).

RCAC

Lihat kontrol akses baris dan kolom.

replika baca

Salinan database yang digunakan untuk tujuan read-only. Anda dapat merutekan kueri ke replika baca untuk mengurangi beban pada database utama Anda.

arsitek ulang

Lihat 7 Rs.

tujuan titik pemulihan (RPO)

Jumlah waktu maksimum yang dapat diterima sejak titik pemulihan data terakhir. Ini menentukan apa yang dianggap sebagai kehilangan data yang dapat diterima antara titik pemulihan terakhir dan gangguan layanan.

tujuan waktu pemulihan (RTO)

Penundaan maksimum yang dapat diterima antara gangguan layanan dan pemulihan layanan.

refactor

Lihat 7 Rs.

Region

Kumpulan AWS sumber daya di wilayah geografis. Masing-masing Wilayah AWS terisolasi dan independen dari yang lain untuk memberikan toleransi kesalahan, stabilitas, dan ketahanan. Untuk informasi selengkapnya, lihat Menentukan Wilayah AWS akun yang dapat digunakan.

regresi

Teknik ML yang memprediksi nilai numerik. Misalnya, untuk memecahkan masalah “Berapa harga rumah ini akan dijual?” Model ML dapat menggunakan model regresi linier untuk memprediksi harga jual rumah berdasarkan fakta yang diketahui tentang rumah (misalnya, luas persegi).

rehost

Lihat 7 Rs.

melepaskan

Dalam proses penyebaran, tindakan mempromosikan perubahan pada lingkungan produksi.

memindahkan

Lihat 7 Rs.

memplatform ulang

Lihat 7 Rs.

pembelian kembali

Lihat 7 Rs.

ketahanan

Kemampuan aplikasi untuk melawan atau pulih dari gangguan. Ketersediaan tinggi dan pemulihan bencana adalah pertimbangan umum ketika merencanakan ketahanan di. AWS Cloud Untuk informasi lebih lanjut, lihat AWS Cloud Ketahanan.

kebijakan berbasis sumber daya

Kebijakan yang dilampirkan ke sumber daya, seperti bucket Amazon S3, titik akhir, atau kunci enkripsi. Jenis kebijakan ini menentukan prinsipal mana yang diizinkan mengakses, tindakan yang didukung, dan kondisi lain yang harus dipenuhi.

matriks yang bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI)

Matriks yang mendefinisikan peran dan tanggung jawab untuk semua pihak yang terlibat dalam kegiatan migrasi dan operasi cloud. Nama matriks berasal dari jenis tanggung jawab yang didefinisikan dalam matriks: bertanggung jawab (R), akuntabel (A), dikonsultasikan (C), dan diinformasikan (I). Tipe dukungan (S) adalah opsional. Jika Anda menyertakan dukungan, matriks disebut matriks RASCI, dan jika Anda mengecualikannya, itu disebut matriks RACI.

kontrol responsif

Kontrol keamanan yang dirancang untuk mendorong remediasi efek samping atau penyimpangan dari garis dasar keamanan Anda. Untuk informasi selengkapnya, lihat Kontrol responsif dalam Menerapkan kontrol keamanan pada AWS.

melestarikan

Lihat 7 Rs.

pensiun

Lihat 7 Rs.

Retrieval Augmented Generation (RAG)

Teknologi AI generatif di mana LLM merujuk sumber data otoritatif yang berada di luar sumber data pelatihannya sebelum menghasilkan respons. Misalnya, model RAG mungkin melakukan

pencarian semantik dari basis pengetahuan organisasi atau data kustom. Untuk informasi lebih lanjut, lihat Apa itu RAG.

rotasi

Proses memperbarui rahasia secara berkala untuk membuatnya lebih sulit bagi penyerang untuk mengakses kredensil.

kontrol akses baris dan kolom (RCAC)

Penggunaan ekspresi SQL dasar dan fleksibel yang telah menetapkan aturan akses. RCAC terdiri dari izin baris dan topeng kolom.

RPO

Lihat tujuan titik pemulihan.

RTO

Lihat tujuan waktu pemulihan.

buku runbook

Satu set prosedur manual atau otomatis yang diperlukan untuk melakukan tugas tertentu. Ini biasanya dibangun untuk merampingkan operasi berulang atau prosedur dengan tingkat kesalahan yang tinggi.

# D

SAML 2.0

Standar terbuka yang digunakan oleh banyak penyedia identitas (IdPs). Fitur ini memungkinkan sistem masuk tunggal gabungan (SSO), sehingga pengguna dapat masuk ke Konsol Manajemen AWS atau memanggil operasi AWS API tanpa Anda harus membuat pengguna di IAM untuk semua orang di organisasi Anda. Untuk informasi lebih lanjut tentang federasi berbasis SAMP 2.0, lihat Tentang federasi berbasis SAMP 2.0 dalam dokumentasi IAM.

SCADA

Lihat kontrol pengawasan dan akuisisi data.

SCP

Lihat kebijakan kontrol layanan.

Rahasia

Dalam AWS Secrets Manager, informasi rahasia atau terbatas, seperti kata sandi atau kredensil pengguna, yang Anda simpan dalam bentuk terenkripsi. Ini terdiri dari nilai rahasia dan metadata-nya. Nilai rahasia dapat berupa biner, string tunggal, atau beberapa string. Untuk informasi selengkapnya, lihat Apa yang ada di rahasia Secrets Manager? dalam dokumentasi Secrets Manager.

keamanan dengan desain

Pendekatan rekayasa sistem yang memperhitungkan keamanan melalui seluruh proses pengembangan.

kontrol keamanan

Pagar pembatas teknis atau administratif yang mencegah, mendeteksi, atau mengurangi kemampuan pelaku ancaman untuk mengeksploitasi kerentanan keamanan. Ada empat jenis kontrol keamanan utama: preventif, detektif, responsif, dan proaktif.

pengerasan keamanan

Proses mengurangi permukaan serangan untuk membuatnya lebih tahan terhadap serangan. Ini dapat mencakup tindakan seperti menghapus sumber daya yang tidak lagi diperlukan, menerapkan praktik keamanan terbaik untuk memberikan hak istimewa paling sedikit, atau menonaktifkan fitur yang tidak perlu dalam file konfigurasi.

sistem informasi keamanan dan manajemen acara (SIEM)

Alat dan layanan yang menggabungkan sistem manajemen informasi keamanan (SIM) dan manajemen acara keamanan (SEM). Sistem SIEM mengumpulkan, memantau, dan menganalisis data dari server, jaringan, perangkat, dan sumber lain untuk mendeteksi ancaman dan pelanggaran keamanan, dan untuk menghasilkan peringatan.

otomatisasi respons keamanan

Tindakan yang telah ditentukan dan diprogram yang dirancang untuk secara otomatis merespons atau memulihkan peristiwa keamanan. Otomatisasi ini berfungsi sebagai kontrol keamanan detektif atau responsif yang membantu Anda menerapkan praktik terbaik AWS keamanan. Contoh tindakan respons otomatis termasuk memodifikasi grup keamanan VPC, menambal instans Amazon EC2, atau memutar kredensil.

enkripsi sisi server

Enkripsi data di tujuannya, oleh Layanan AWS yang menerimanya.

kebijakan kontrol layanan (SCP)

Kebijakan yang menyediakan kontrol terpusat atas izin untuk semua akun di organisasi. AWS Organizations SCPs menentukan pagar pembatas atau menetapkan batasan pada tindakan yang dapat didelegasikan oleh administrator kepada pengguna atau peran. Anda dapat menggunakan SCPs daftar izin atau daftar penolakan, untuk menentukan layanan atau tindakan mana yang diizinkan atau dilarang. Untuk informasi selengkapnya, lihat Kebijakan kontrol layanan dalam AWS Organizations dokumentasi.

titik akhir layanan

URL titik masuk untuk file Layanan AWS. Anda dapat menggunakan endpoint untuk terhubung secara terprogram ke layanan target. Untuk informasi selengkapnya, lihat Layanan AWS titik akhir di Referensi Umum AWS.

perjanjian tingkat layanan (SLA)

Perjanjian yang menjelaskan apa yang dijanjikan tim TI untuk diberikan kepada pelanggan mereka, seperti waktu kerja dan kinerja layanan.

indikator tingkat layanan (SLI)

Pengukuran aspek kinerja layanan, seperti tingkat kesalahan, ketersediaan, atau throughputnya.

tujuan tingkat layanan (SLO)

Metrik target yang mewakili kesehatan layanan, yang diukur dengan indikator tingkat layanan.

model tanggung jawab bersama

Model yang menjelaskan tanggung jawab yang Anda bagikan AWS untuk keamanan dan kepatuhan cloud. AWS bertanggung jawab atas keamanan cloud, sedangkan Anda bertanggung jawab atas keamanan di cloud. Untuk informasi selengkapnya, lihat Model tanggung jawab bersama.

SIEM

Lihat informasi keamanan dan sistem manajemen acara.

titik kegagalan tunggal (SPOF)

Kegagalan dalam satu komponen penting dari aplikasi yang dapat mengganggu sistem.

SLA

Lihat perjanjian tingkat layanan.

SLI

Lihat indikator tingkat layanan.

SLO

Lihat tujuan tingkat layanan.

split-and-seed model

Pola untuk menskalakan dan mempercepat proyek modernisasi. Ketika fitur baru dan rilis produk didefinisikan, tim inti berpisah untuk membuat tim produk baru. Ini membantu meningkatkan kemampuan dan layanan organisasi Anda, meningkatkan produktivitas pengembang, dan mendukung inovasi yang cepat. Untuk informasi lebih lanjut, lihat Pendekatan bertahap untuk memodernisasi aplikasi di. AWS Cloud

SPOF

Lihat satu titik kegagalan.

skema bintang

Struktur organisasi database yang menggunakan satu tabel fakta besar untuk menyimpan data transaksional atau terukur dan menggunakan satu atau lebih tabel dimensi yang lebih kecil untuk menyimpan atribut data. Struktur ini dirancang untuk digunakan dalam gudang data atau untuk tujuan intelijen bisnis.

pola ara pencekik

Pendekatan untuk memodernisasi sistem monolitik dengan menulis ulang secara bertahap dan mengganti fungsionalitas sistem sampai sistem warisan dapat dinonaktifkan. Pola ini menggunakan analogi pohon ara yang tumbuh menjadi pohon yang sudah mapan dan akhirnya mengatasi dan menggantikan inangnya. Pola ini diperkenalkan oleh Martin Fowler sebagai cara untuk mengelola risiko saat menulis ulang sistem monolitik. Untuk contoh cara menerapkan pola ini, lihat Memodernisasi layanan web Microsoft ASP.NET (ASMX) lama secara bertahap menggunakan container dan Amazon API Gateway.

subnet

Rentang alamat IP dalam VPC Anda. Subnet harus berada di Availability Zone tunggal.

kontrol pengawasan dan akuisisi data (SCADA)

Di bidang manufaktur, sistem yang menggunakan perangkat keras dan perangkat lunak untuk memantau aset fisik dan operasi produksi.

enkripsi simetris

Algoritma enkripsi yang menggunakan kunci yang sama untuk mengenkripsi dan mendekripsi data.

pengujian sintetis

Menguji sistem dengan cara yang mensimulasikan interaksi pengguna untuk mendeteksi potensi masalah atau untuk memantau kinerja. Anda dapat menggunakan Amazon CloudWatch Synthetics untuk membuat tes ini.

sistem prompt

Teknik untuk memberikan konteks, instruksi, atau pedoman ke LLM untuk mengarahkan perilakunya. Permintaan sistem membantu mengatur konteks dan menetapkan aturan untuk interaksi dengan pengguna.

# T

tag

Pasangan nilai kunci yang bertindak sebagai metadata untuk mengatur sumber daya Anda. AWS Tanda membantu Anda mengelola, mengidentifikasi, mengatur, dan memfilter sumber daya. Untuk informasi selengkapnya, lihat Menandai AWS sumber daya Anda.

variabel target

Nilai yang Anda coba prediksi dalam ML yang diawasi. Ini juga disebut sebagai variabel hasil. Misalnya, dalam pengaturan manufaktur, variabel target bisa menjadi cacat produk.

daftar tugas

Alat yang digunakan untuk melacak kemajuan melalui runbook. Daftar tugas berisi ikhtisar runbook dan daftar tugas umum yang harus diselesaikan. Untuk setiap tugas umum, itu termasuk perkiraan jumlah waktu yang dibutuhkan, pemilik, dan kemajuan.

lingkungan uji

Lihat lingkungan.

pelatihan

Untuk menyediakan data bagi model ML Anda untuk dipelajari. Data pelatihan harus berisi jawaban yang benar. Algoritma pembelajaran menemukan pola dalam data pelatihan yang

memetakan atribut data input ke target (jawaban yang ingin Anda prediksi). Ini menghasilkan model ML yang menangkap pola-pola ini. Anda kemudian dapat menggunakan model ML untuk membuat prediksi pada data baru yang Anda tidak tahu targetnya.

gerbang transit

Hub transit jaringan yang dapat Anda gunakan untuk menghubungkan jaringan Anda VPCs dan lokal. Untuk informasi selengkapnya, lihat Apa itu gateway transit dalam AWS Transit Gateway dokumentasi.

alur kerja berbasis batang

Pendekatan di mana pengembang membangun dan menguji fitur secara lokal di cabang fitur dan kemudian menggabungkan perubahan tersebut ke cabang utama. Cabang utama kemudian dibangun untuk pengembangan, praproduksi, dan lingkungan produksi, secara berurutan.

akses tepercaya

Memberikan izin ke layanan yang Anda tentukan untuk melakukan tugas di organisasi Anda di dalam AWS Organizations dan di akunnya atas nama Anda. Layanan tepercaya menciptakan peran terkait layanan di setiap akun, ketika peran itu diperlukan, untuk melakukan tugas manajemen untuk Anda. Untuk informasi selengkapnya, lihat Menggunakan AWS Organizations dengan AWS layanan lain dalam AWS Organizations dokumentasi.

penyetelan

Untuk mengubah aspek proses pelatihan Anda untuk meningkatkan akurasi model ML. Misalnya, Anda dapat melatih model ML dengan membuat set pelabelan, menambahkan label, dan kemudian mengulangi langkah-langkah ini beberapa kali di bawah pengaturan yang berbeda untuk mengoptimalkan model.

tim dua pizza

Sebuah DevOps tim kecil yang bisa Anda beri makan dengan dua pizza. Ukuran tim dua pizza memastikan peluang terbaik untuk berkolaborasi dalam pengembangan perangkat lunak.

# U

waswas

Sebuah konsep yang mengacu pada informasi yang tidak tepat, tidak lengkap, atau tidak diketahui yang dapat merusak keandalan model ML prediktif. Ada dua jenis ketidakpastian: ketidakpastian epistemik disebabkan oleh data yang terbatas dan tidak lengkap, sedangkan

ketidakpastian aleatorik disebabkan oleh kebisingan dan keacakan yang melekat dalam data. Untuk informasi lebih lanjut, lihat panduan Mengukur ketidakpastian dalam sistem pembelajaran mendalam.

tugas yang tidak terdiferensiasi

Juga dikenal sebagai angkat berat, pekerjaan yang diperlukan untuk membuat dan mengoperasikan aplikasi tetapi itu tidak memberikan nilai langsung kepada pengguna akhir atau memberikan keunggulan kompetitif. Contoh tugas yang tidak terdiferensiasi termasuk pengadaan, pemeliharaan, dan perencanaan kapasitas.

lingkungan atas

Lihat lingkungan.

# V

menyedot debu

Operasi pemeliharaan database yang melibatkan pembersihan setelah pembaruan tambahan untuk merebut kembali penyimpanan dan meningkatkan kinerja.

kendali versi

Proses dan alat yang melacak perubahan, seperti perubahan kode sumber dalam repositori.

Peering VPC

Koneksi antara dua VPCs yang memungkinkan Anda untuk merutekan lalu lintas dengan menggunakan alamat IP pribadi. Untuk informasi selengkapnya, lihat Apa itu peering VPC di dokumentasi VPC Amazon.

kerentanan

Kelemahan perangkat lunak atau perangkat keras yang membahayakan keamanan sistem.

# W

cache hangat

Cache buffer yang berisi data terkini dan relevan yang sering diakses. Instance database dapat membaca dari cache buffer, yang lebih cepat daripada membaca dari memori utama atau disk.

data hangat

Data yang jarang diakses. Saat menanyakan jenis data ini, kueri yang cukup lambat biasanya dapat diterima.

fungsi jendela

Fungsi SQL yang melakukan perhitungan pada sekelompok baris yang berhubungan dengan catatan saat ini. Fungsi jendela berguna untuk memproses tugas, seperti menghitung rata-rata bergerak atau mengakses nilai baris berdasarkan posisi relatif dari baris saat ini.

beban kerja

Kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi yang dihadapi pelanggan atau proses backend.

aliran kerja

Grup fungsional dalam proyek migrasi yang bertanggung jawab atas serangkaian tugas tertentu. Setiap alur kerja independen tetapi mendukung alur kerja lain dalam proyek. Misalnya, alur kerja portofolio bertanggung jawab untuk memprioritaskan aplikasi, perencanaan gelombang, dan mengumpulkan metadata migrasi. Alur kerja portofolio mengirimkan aset ini ke alur kerja migrasi, yang kemudian memigrasikan server dan aplikasi.

CACING

Lihat menulis sekali, baca banyak.

WQF

Lihat AWS Kerangka Kualifikasi Beban Kerja.

tulis sekali, baca banyak (WORM)

Model penyimpanan yang menulis data satu kali dan mencegah data dihapus atau dimodifikasi. Pengguna yang berwenang dapat membaca data sebanyak yang diperlukan, tetapi mereka tidak dapat mengubahnya. Infrastruktur penyimpanan data ini dianggap tidak dapat diubah.

# Z

eksploitasi zero-day

Serangan, biasanya malware, yang memanfaatkan kerentanan zero-day.

kerentanan zero-day

Cacat atau kerentanan yang tak tanggung-tanggung dalam sistem produksi. Aktor ancaman dapat menggunakan jenis kerentanan ini untuk menyerang sistem. Pengembang sering menyadari kerentanan sebagai akibat dari serangan tersebut.

bisikan zero-shot

Memberikan LLM dengan instruksi untuk melakukan tugas tetapi tidak ada contoh (tembakan) yang dapat membantu membimbingnya. LLM harus menggunakan pengetahuan pra-terlatih untuk menangani tugas. Efektivitas bidikan nol tergantung pada kompleksitas tugas dan kualitas prompt. Lihat juga beberapa bidikan yang diminta.

aplikasi zombie

Aplikasi yang memiliki CPU rata-rata dan penggunaan memori di bawah 5 persen. Dalam proyek migrasi, adalah umum untuk menghentikan aplikasi ini.

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.