Customer CASE STUDY

databricks

データをリアルタイムに収集・加工・分析し、全てのユーザーに、より早くより魅力的なマッチング体験を。

RECRUIT 株式会社リクルート

プロダクト統括本部
プロダクト開発統括室
データ推進室
川合 真大 氏

プロダクト統括本部
プロダクト開発統括室
データ推進室
田村 優友 氏

人や企業に対して最適なマッチングの場を提供。

リクルートホールディングスには、「マッチング&ソリューションSBU」、「HRテクノロジーSBU」、「人材派遣SBU」3つの事業ドメインがある。そのうち「マッチング&ソリューションSBU」を統括するリクルートには、住宅、旅行、飲食などの「販促領域」があり、代表的なサービスは『SUUMO』、『ゼクシィ』、『じゃらん』、『カーセンサー』などがある。また、ユーザーの求職活動や企業の採用活動を支援する「人材領域」において、『リクナビ』や『リクナビNEXT』、『RECRUIT AGENT』などのサービスを展開している。ビジネスモデルとしては、様々な人や企業に対して、最適なマッチングを行う場を提供することで、その対価を得ている。

データ推進室では、複数展開している各事業の戦略に基づいたデータ施策を推進する為、販促・人材領域といったビジネス領域ごとに構成される縦の組織と、データサイエンス・データエンジニアリングなど専門機能毎にまとまった横の組織の組み合わせで構成されている。こちらをベースにビジネス価値の創出とガバナンスの強化に取り組んでいる。

リアルタイムレコメンドの実現でより価値の高いユーザー体験を。

様々なプロジェクトを支援する中で、データ推進室が提供する価値を高めるには、「より早く、より魅力的なマッチングをユーザーに提供すること」が必要であると感じていた。当時、既にいくつかのサービスにおいて、個別にデータ分析基盤は構築されており、それにもとづいたレコメンデーション(ユーザーへのコンテンツ推奨)自体は提供されていた。しかし、中にはデータ鮮度の低い特徴量を活用したレコメンデーションモデルも存在していた。具体的には、ユーザー行動データは前日までのものであったり、新規ユーザーに関しては、最適なレコメンデーションが提供されていなかった。目指す理想像としては、ユーザーの興味や関心がリアルタイムに反映され、新規・既存に問わず、全てのユーザーに最適なレコメンデーションを提供することである。いわゆるリアルタイムレコメンドの実現だ。

例えば、多種多様な動画コンテンツを配信するサービスがあったと仮定して、あるユーザーが1日前に"犬"に関する動画を視聴、そして"料理"に関する動画を5分前に視聴したとする。リアルタイムレコメンドが実装されていない場合、"犬"に関するレコメンドのみが提供され、当該ユーザーの直近の興味・関心を活かすことができない。裏を返せば、リアルタイムレコメンドによって、ユーザーへ"犬"と"料理"の双方に関するレコメンドをすることが可能となり、より価値の高いユーザー体験の提供が実現する。

このリアルタイムレコメンドの実装には、データをリアルタイムに「収集・加工・分析」することが可能な「ストリーム基盤の構築」が必要であった。今回のプロジェクトには、3つの要件が存在していた。1つ目は、初回のリリースまでの納期は3ヶ月であったこと。2つ目は、今後、他サービスへの横展開を予定していること。3つ目は、開発、および運用保守要員は限られていること。つまり、限られた人員で構築・運用ができる拡張性/柔軟性の高いストリーミング基盤を、短納期に構築する必要があったわけである。そして、ここから3つのシステム要件が導き出された。

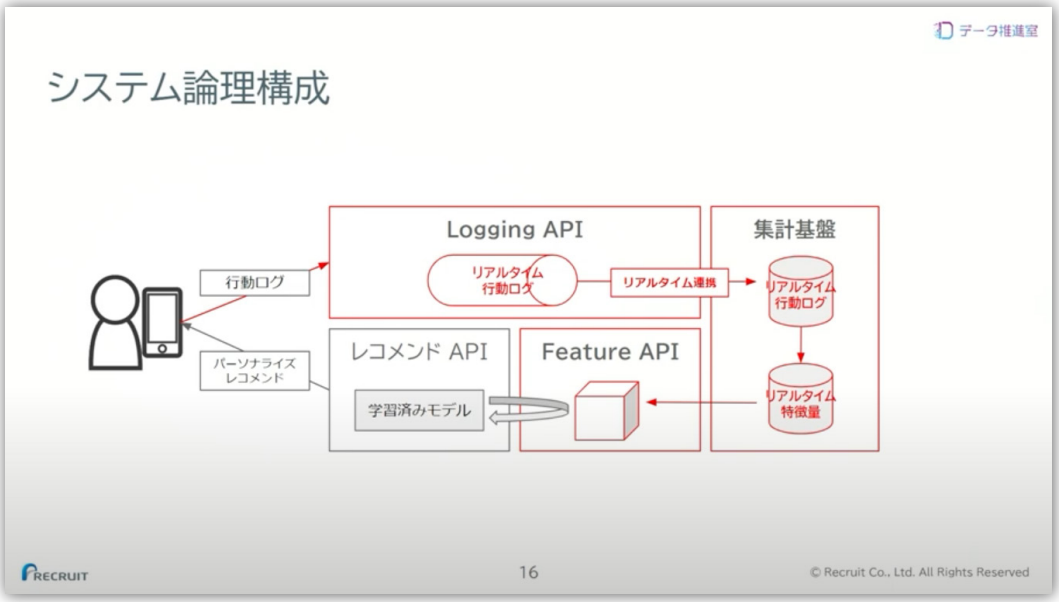
1つ目は「開発容易性/効率性」。決められたスケジュール内で開発を終えられることに加え、データサイエンティストによるリリース後の高頻度な仮説検証や機能エンハンスが着手可能であることである。2つ目は、「各サービス特性に対応できる柔軟性」。サービスによって収集・加工・分析するデータの種類やデータ量が異なる為、様々なデータフォーマットや大規模なデータ処理にも対応できる柔軟性が求められる。3つ目は、「最小限工数での運用/保守性」。システムの拡張/拡大に比例して人員を拡大することはできないため、少人数での運用/保守を継続できることが必須となる。

これらの要件を満たすストリーミング基盤を構築した。(図1)はシステムの論理構成図である。基盤は、3つのコンポーネントで構成されている。1つ目は、サービスから行動ログを取得するLogging APIである。2つ目は、行動ログを集計し特徴量を生成する集計基盤である。3つ目は、集計基盤から収集した最新の特徴量をレコメンドAPIへ提供するFeature APIである。

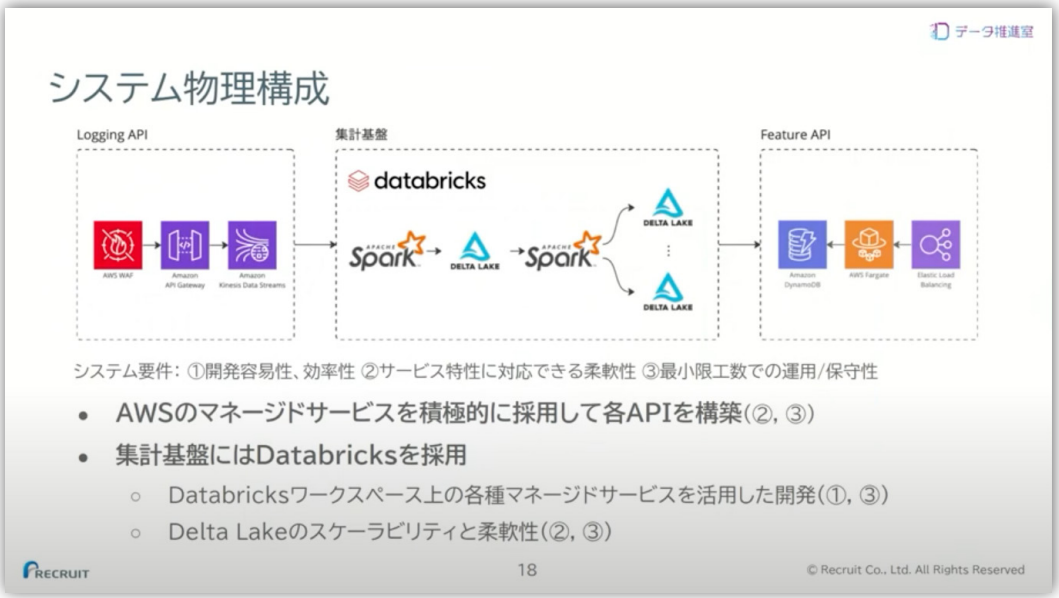
(図2)は、システムの物理構成図である。Logging APIとFeature APIは、Amazon API GatewayやAmazon Kinesis、Amazon DynamoDBなどのマネージドサービスを活用して構築した。実装コストや運用コストの削減を重視した。例えば、Logging APIは、テラフォームとOpenAPIのリソース定義から作成したので、サーバーロジックの設定は不要であった。Logging APIの要件がシンプルであったため、このような構成が可能となった。一方で、Feature APIは、レスポンス対応の99パーセンタイルを50ms(ミリセカンド)以内に収めることを目指したため、ロジックの最適化やネットワーク遅延の最小化のため、Elastic Load BalancingとAWS Fargateで構築した。中央の集計基盤に関しては、データブリックスのレイクハウス・プラットフォーム上に構築しており、Apache SparkとDelta Lakeの集計ジョブが稼働している。

データブリックスの採用には、3つのポイントがあった。1つ目は、「Notebookを用いた開発の敷居の低さ」である。Spark 初学者でも3ヶ月以内に開発可能な基盤であり、コードの実行が容易にでき、結果やデータ流量を視覚的に確認できるところが気に入っている。2つ目は、「Delta Lakeと統合されたプラットフォーム」であること。集計基盤のストレージにDelta Lakeを採用しており、追加の設定やデータフォマットの変換などが不要であった。3つ目は、「開発に必要なマネージド機能を備えたワークスペース」を有していることである。クラスターの起動やコピー、マシーンタイプの設定などのリソース管理は、

(図1)



(図2)



GUIから容易に実行でき、ストリーミング処理とバッチ処理の双方に利用可能なワークフロー機能も提供している為、集計ロジックの開発に集中することができた。

3ヶ月でのリリース目標を達成。今後はQA環境の整備を目指す。

当初目標として3ヶ月での初回リリースは無事に達成し、リリース後も安定稼働中である。また、他サービス領域での横展開も1ヶ月程度で実施できる目処も立っている状況である。ほぼ全てのリソースをInfrastructure as a Codeとして管理しており、ログ仕様やクラスタースレッドなどのサービス特性によって変わる箇所は、変数化してあるので、その部分だけ対応可能である。今後の展望としては、まず「Unity Catalog」の導入を予定している。これにより、Databricks Workspaceを超えたデータ共有が可能となり、本番相当のデータで検証を実施できるQA環境の整備が実現する。次に、「CI/CDの改善」である。開発効率の改善やダウンタイムを最小化するための仕組み改善に取り組む予定である。そして、より効果の高いレコメンドを実現するために、特徴量の変更や追加を継続的に実施していく予定だ。データ推進室では、これらのプロジェクトを共に進めるメンバーを募集している。詳細は、是非こちらのQRコードを確認して欲しい。