
# Job Recommendation Engine for Seek

Biying Chen

2025-03-19

# Contents

# 1. Background

Australia’s online recruitment platform in thriving[1].

- Market size: 0.9 billion AUD
- Annual growth rate: 8.4%

**Seek** is one of the biggest online job marketplaces, aiming to connect more people to relevant employment.

To realize this ambition, Seek is building a job recommendation engine to increase the view and application rates of the jobs[2].

This presentation will introduce the design, develop of a job recommendation engine.


*Reference:*

1. *https://www.ibisworld.com/australia/industry/online-recruitment-services/4049/*
2. *https://talent.seek.com.au/products/jobads*

# 2. Data Preprocessing and Analysis

# 2.1.1 Raw Data Preprossing: Job Advertisement Overview

This dataset is the job description for jobs posted on Seek.

It contains 50k records in json format. **No duplication** but **has missing data** for some fields.

**Data Drop logic**

- Row removal:
  - Jobs (372 records) with **non-English** description are removed to avoid negative effect during text analysis.
- Column removal:
  - Column *location*, *suburb*, *area* are dropped **after address logic**.
  - *bullet 1, bullet 2, bullet 3* are dropped due to **too many missing data** and no way to fill in blanks.

# 2.1.1 Raw Data Preprossing: Job Advertisement Columns

Cleaned and Processed table:

| Column | Data Type | Comment | Action |
|---|---|---|---|
| **id** | String | PK for each job | |
| **cleaned_title** | String | Job name | Remove meaningless words from title |
| **abstract_content** | String | Description | Combine abstract and content, remove meaningless words |
| **classification** | String | Industry | |
| **sub_classification** | String | Sub -Industry | |
| **work_type** | string | Work types | |

## 2.1.1 Raw Data Preprossing: Job Advertisement Columns

To continue:

| Column | Data Type | Comment | Action |
|---|---|---|---|
| **latitude** | float (64) | | Obtain coordinates and full addresses from 3rd API based on location info |
| **longitude** | float (64) | | |
| **region_code** | integer(64) | A flag to distinguish Australian and non-Australian address<br>*(0 - AU, 1- NZ, 999-Others)* | |
| **country** | string | Country name | |
| **state** | string | State name for Australian address | |
| **salary_unit** | String | Salary unit(hour, week, month, year) | Extract from additionalSalaryText |
| **salary_value** | Float(64) | | |

# 2.1.1 Raw Data Preprossing: Parsing locations in Job Advertisement

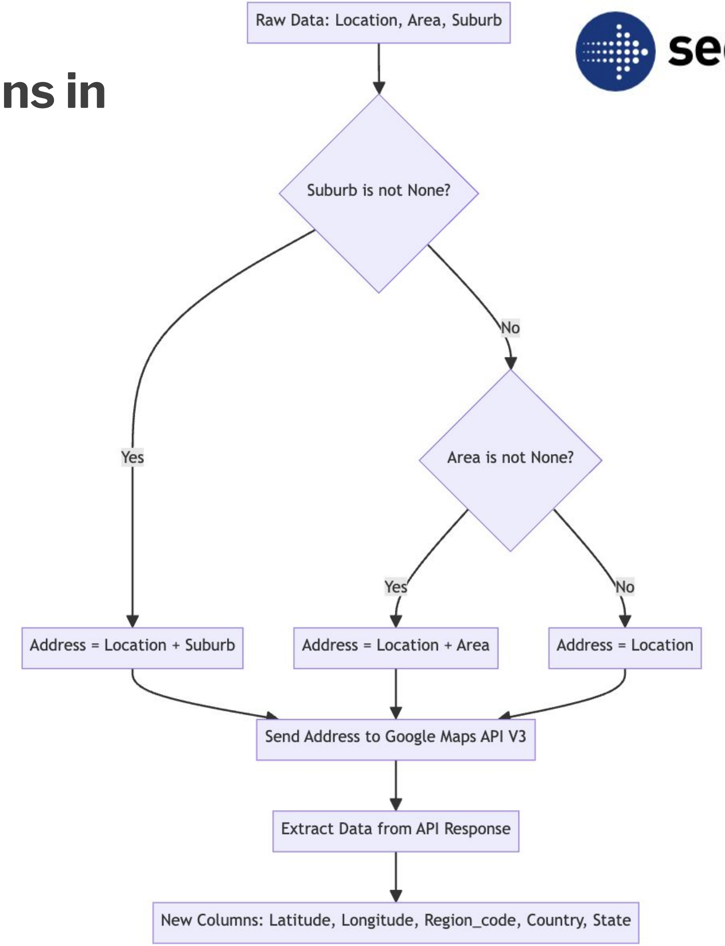
The logic of new geographical columns:

Location is always not None.

```
If suburb is not None,
      address = concat(location, suburb)
else if area is not None,
      address = concat(location, area)
else address = location
```

Address is then put into the Google Map API to generate the new columns:

- Latitude
- Longtitude
- Region_code
- Country
- State



## 2.1.1 Raw Data Preprossing: Parsing Salary in Job Advertisement

Parsing the *additionalSalaryText* in the original dataset.

Apply **Regex** rules to extract **salary unit** and **salary amount**.
Available salary unit: Hourly, Daily, Weekly, Monthly, Annually

For a salary range, use the average of the range.

Leave a placeholder 'NA' in case the salary is not available.

| additionalSalaryText | Salary Unit | Salary Amount |
|---|---|---|
| **$140k + Car Park - Call James Calleja** | Annually | 140000 |
| **$110k - $120k p.a. + Numerous Perks!** | Annually | 115000 |
| **$30 - $34.99 per hour** | Hourly | 32.50 |
| **Base + Super + Uncapped Commission** | NA | 0 |

## 2.1.2 Raw Data Preprossing: Job Event Overview and Columns

This dataset is the event job triggered every time resume (candidate) view or apply for a job. It originally contain 4.3M records in csv format. After **deduplication**, it keeps 1.4M records

Cleaned and Processed table:

| Column | Data Type | Comment | Action |
|---|---|---|---|
| **event_datetime** | String | Timestamp for each log | No need |
| **resume_id** | String | Identify who triggered the log | |
| **job_id** | String | Join key for Job Ads Table | |
| **event_platform** | String | ios, Andriod, web | |
| **kind** | String | V - View, A - Apply | |

## 2.1.3 Raw Data Prepropssing: Join two datasets

The new dataset combines Job Event with Job Advertisement, **joined** through “**id**” from Job Advertisement and “**job_id**” from Job Event, with **1.4M records** in total.

Except for the columns from both tables, additional columns are added to enrich applicant profile.

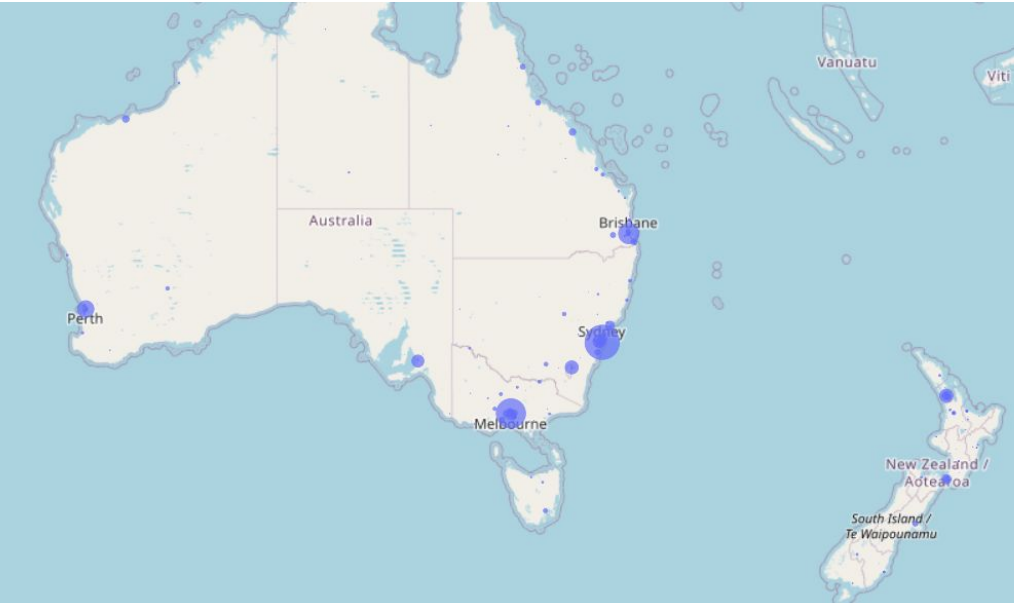
| Additional Column | Data Type | Comment | Action |
|---|---|---|---|
| **centroid_longitude** | Float(64) | Assume location of each applicant | Calculate the averaged coordinates for all jobs each candidate interacted with, based on training set |
| **centroid_latitude** | Float(64) | | |
| **farthest_distance_to_center_km** | Float(64) | Farthest distance for all jobs viewed/applied by each applicant | Calculate the maximum distance between each candidate and jobs |
| **shortest_distance_to_center_km** | Float(64) | Shortest distance for all jobs viewed/applied by each applicant | Calculate the minimum distance between each candidate and jobs |

# 2.1.3 Raw Data Preprossing: Join two datasets

To continue:

| Column | Data Type | Comment | Action |
|---|---|---|---|
| **average_distance_to_center_km** | Float(64) | Averaged distance for all jobs viewed/applied by each applicant | Calculate the average distance between each candidate and jobs |
| **title_keywords** | String | Keywords for titles | Get keywords for each job based on classification based TF-IDFmatrixes |
| **abstract_content_keywords** | String | Keywords for abstract_content | Get keywords for each job based on classification based TF-IDFmatrixes |

# 2.2.1 EDA: Job Density Map across Australia and New Zealand

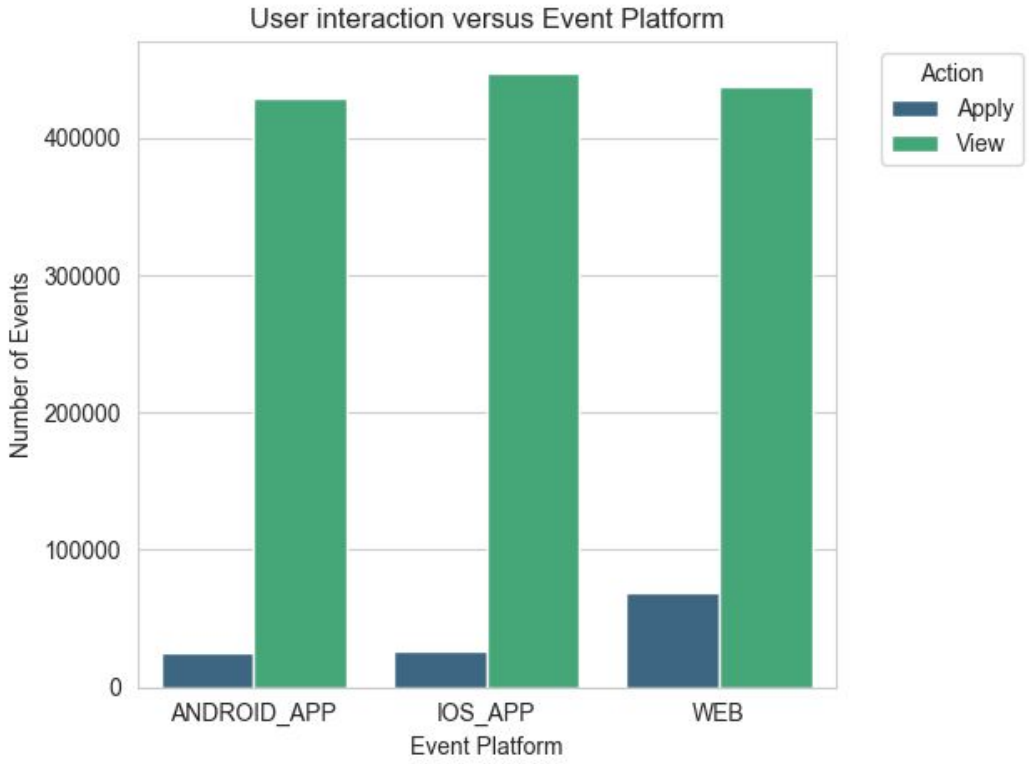



Most job opportunities are from the **capital city** of each state.

Sydney has the most job opportunities, followed by Melbourne and Brisbane.

The amount of jobs from other cities are even fewer than that from NZ cities.

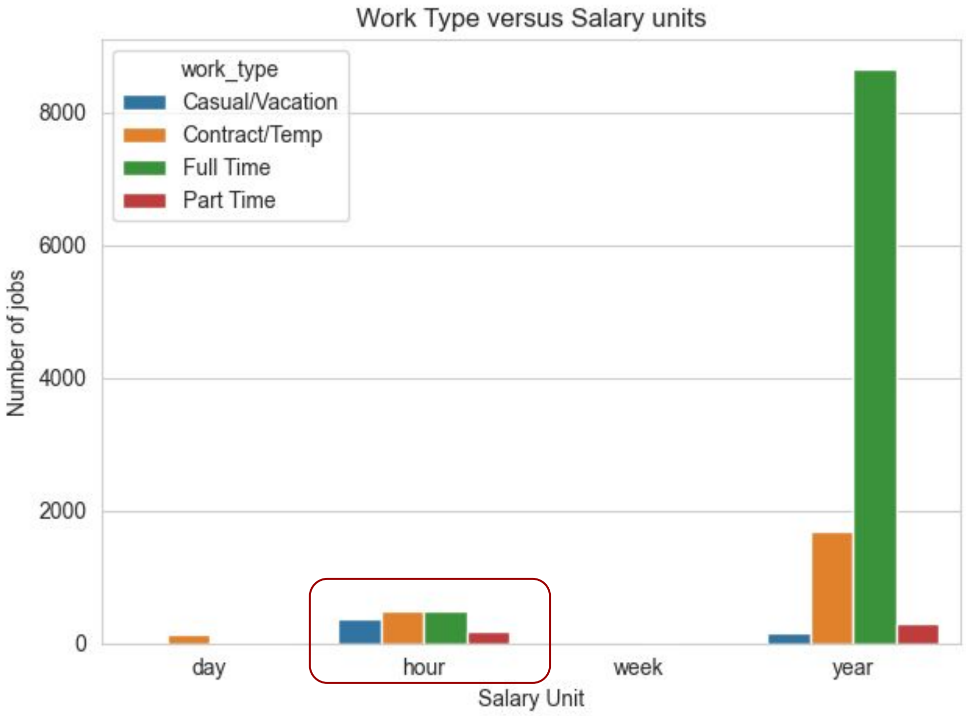
# 2.2.2 EDA: User Interaction vs Platform




Almost no difference between Andriod and iOS.

The conversion rate (VIEW to APPLY) from mobile platforms (Android/iOS) is 50% less than that from the web.

# 2.2.3 EDA: Work Type vs Salary Unit

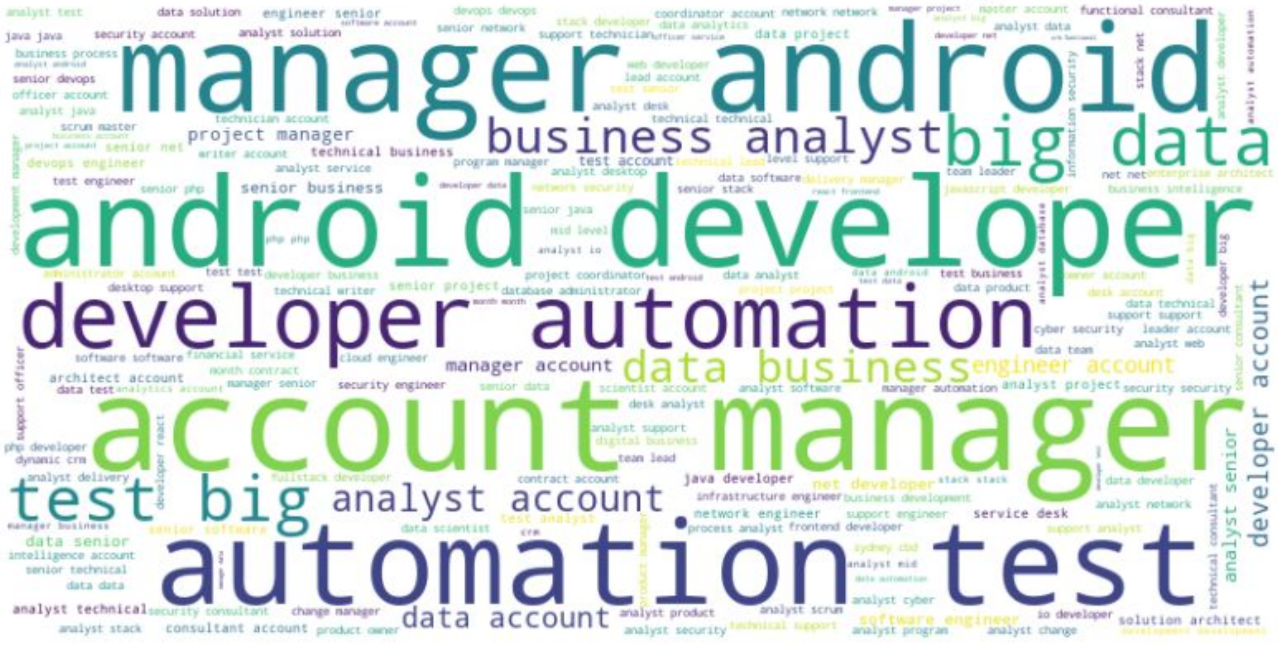



Most full-time jobs are annual salary, which makes sense.

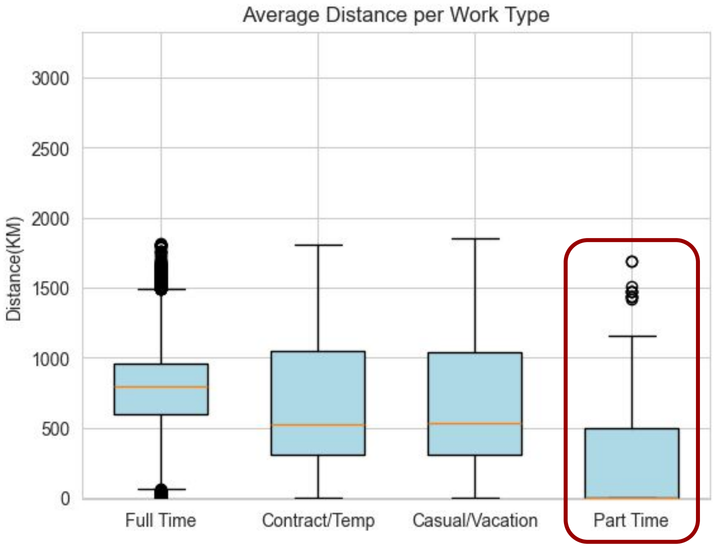
However some full-time jobs are hourly salary, which might be errors in the raw data.

```
{'additionalSalaryText':
      'Up to $55 per hour',
'classification':
      {'name': 'Healthcare & Medical'},
'subClassification':
      {'name': 'Medical Imaging'},
'location':
      {'name':'Rockhampton'},
'workType':
      {'name': 'Full Time'}
}
```

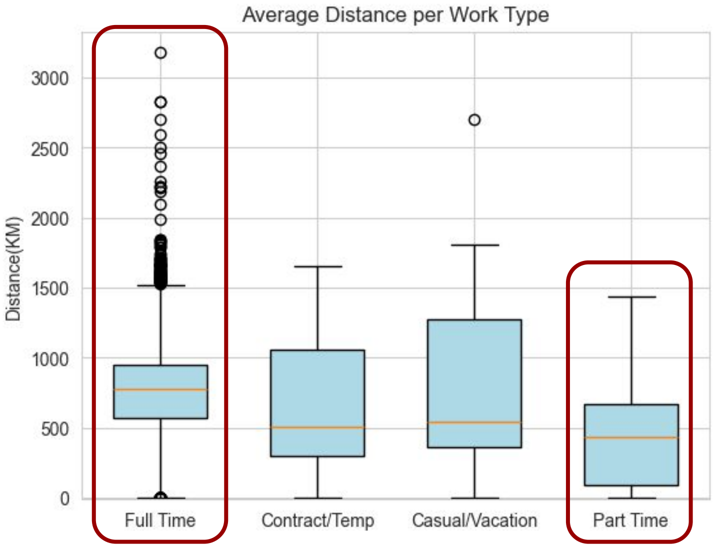
# 2.2.4 EDA: Word Cloud for ICT job titles



# 2.2.5 EDA: Avg Travel Distance vs Work Type



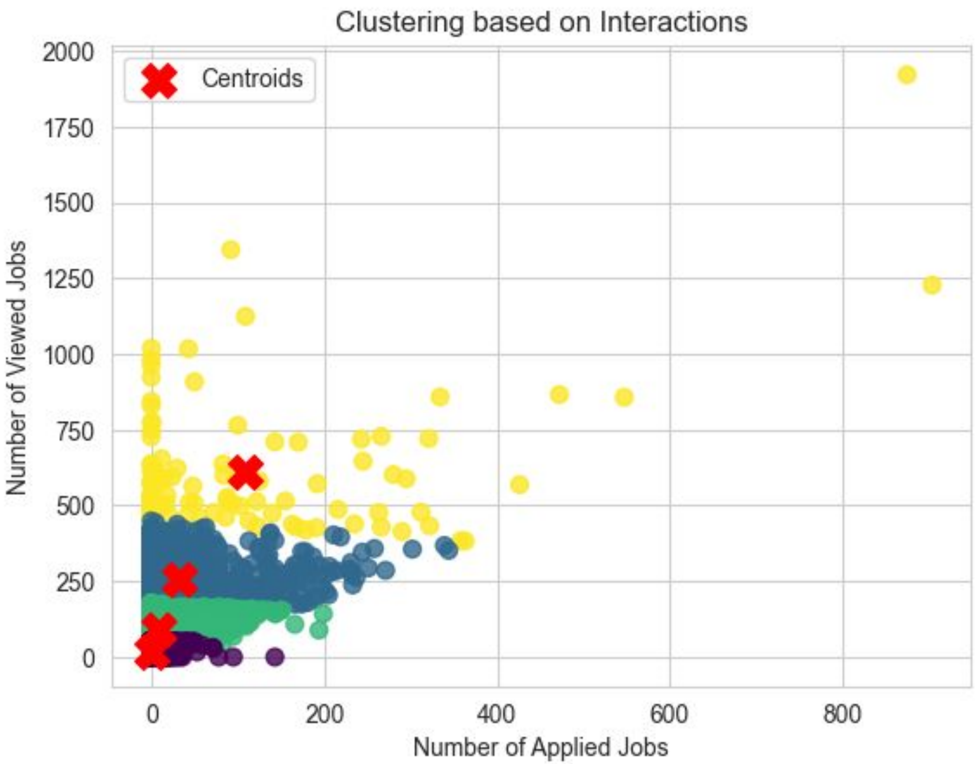

View




Apply

No significant difference of the avg. accepted distance across all work types for APPLY.

# 2.2.6 EDA: Applicant Interaction Clustering




Applicants are clustered into **4 groups**.

Group 1: **Explorer**. Casually browsing, or just keeping an eye on the market.

Group 2: **Opportunist**. Actively looking for the right opportunity but are cautious.

Group 3: **Power Seeker**. Highly motivated, aggressively pursuing new opportunities.

Group 4: Probably **bot**. Unreasonable amount of views and applies.

# 3. Use Cases

# 3.1 Boosting Engagement & Conversions with Dual-Layer AI Predictions

## Problem Statement

- **Low Email Effectiveness**
  - Subscription emails fail to re-engage inactive users, resulting in declining view rates.
  - Example: Only 15% of users open non-personalized job alerts.
- **Inefficient Recommendations**
  - Users click emailed job links but rarely apply due to mismatch.
  - Example: 70% of clicks do not convert to applications.
- **Retention Risks**
  - Declining user activity harms long-term growth and revenue.

## Assumption

- The jobs that applicants view or apply for are typically concentrated around their geographical location. So, an applicant's location is determined by calculating the **geographic centre** (centroid) of all the **job locations they have interacted with**, whether viewed or applied.

# 3.1 Boosting Engagement & Conversions with Dual-Layer AI Predictions

**Objective**

1. **Boost User Re-engagement**:
    - Predict users' likelihood to **view jobs** and trigger personalized emails.
    - Target: Increase email open rates by 30%.
2. **Drive Meaningful Conversions**:
    - Predict jobs users are most likely to **apply for** and prioritize them post-click.
    - Target: Improve apply rates by 25%.
3. **Enhance Platform Value**:
    - Create a seamless journey from email → view → application.
    - Align with business success metrics: User retention, employer satisfaction, and revenue growth.

# 3.1 Boosting Engagement & Conversions with Dual-Layer AI Predictions

**Methodology**

**Part 1: Build Models**

Raw data: User event data, Job Advertisement data.

Potential models: Random Forrest, XGBoost, Neural Network, etc.

Layer 1: View Prediction Model

- Goal: Identify jobs that users likely to view.
- Action: Trigger personalized email alerts for high-probability users.

Layer 2: Apply Prediction Model

- Goal: Prioritize jobs a user is most likely to apply to.
- Action: Dynamically reorder recommendations post-click.

# 3.1 Boosting Engagement & Conversions with Dual-Layer AI Predictions

**Methodology**

**Part 2: Set up pipeline**

Goal: Build pipeline on  Google Cloud Platform to streamline job recommendation, scheduling daily run.

Action: Obtain new jobs that not seen by applicant, make predictions and send emails to applicants.

**Part 3: Implement A/B Testing**

Goal: Compare view-to-apply conversion rate for dual-layer performance with the legacy systems.

Action: Randomly divide applicants into 2 groups and route applicants to new pipeline or legacy system for a month

# 3.1 Boosting Engagement & Conversions with Dual-Layer AI Predictions

## Key Challenges

- **Data Sparsity**: Predicting behavior for inactive users with **limited interaction history**.
- **Model Drift**: Adapting to shifting candidate preferences (e.g., post-pandemic trends).

## Outcomes

- **+35% Email Open Rate**
- **+25% Apply Rate**: Jobs ranked by apply probability outperformed legacy rankings.
- **+15% User Retention**: Re-engaged inactive users with tailored emails.
- **Revenue Impact**: $2M+ annualized revenue from increased successful placements.

## Future Work

- Enrich data by adding applicant and employer demographics, accurate job salary info, etc.
- Scale up by applying GPU.
- Create real-time pipeline.

# 3.2 AI-Driven Job Ad Performance Prediction (Idea)

**Problem Statement**

Employers are struggling to create high-performing job ads:

- **Market Complexity**: Rapid shifts in candidate preferences (e.g., WFH).
- **Platform Impact**: Poor ad performance harms SEEK's marketplace health by reducing hirer retention and candidate satisfaction.

**Objective**

Empower employers to create high-performing job ads using data-driven insights:

- **Predict Engagement**: Build ML models to forecast click-through rates and conversion rates.
- **Deliver Actionable Insights in time**: Provide real-time recommendations to optimize ad content.
- **Leverage SEEK's Data**: Utilize historical ad interactions across industries and regions.

# 3.2 AI-Driven Job Ad Performance Prediction (Idea)

## Methodology

**Part 1: NLP-Driven Content Analysis**

- Technique: Fine-tune **BERT** to extract semantic patterns.
- Output: Key phrases (e.g., "WFH") linked to high click-through rate.

**Part 2: Predictive Modeling**

- Model: **XGBoost, Neural Network, Gradient-boosted trees** for CTR prediction.
- Validation: Time-based splits and SHAP for interpretability.

**Part 3: A/B Testing & Iteration**

- Design: Compare AI recommendations vs. legacy tools.
- Dashboard: Real-time "what-if" scenarios for hirers.

**Part 4: Deployment & Monitoring**

- Infrastructure: GCP Vertex AI, DataFlow, REST APIs.
- Monitoring: Track model drift and retrain monthly.

# 3.2 AI-Driven Job Ad Performance Prediction (Idea)

## Key Challenges

- **Complex Data**: Unstructured text and noisy employer metadata.
- **Real-Time Demands**: Low-latency requirements for dashboard interactions.
- **Model Drift**: Adapting to shifting candidate preferences (e.g., post-pandemic trends).

## Outcomes

- **20% Average CTR Increase**: For employers adopting AI recommendations.
- **15% Faster Hiring Cycles**: Reduced time-to-fill for high-priority roles.
- **10% Revenue Growth**: From increased ad spend by satisfied hirers.
- **Platform Differentiation**: SEEK positioned as a leader in AI-driven recruitment tools.

# 4. AI solutions

Use case 1: Boosting Engagement & Conversions with Dual-Layer AI Predictions

# 4.1 Split Training and Validation Data

Training and Validation Data is generated based on **Joined Data**.

Based on our 2-step prediction, 2 sets of Training and Validation Data is created. Only **applicants with sufficient records** (more than 100 event logs) will be selected for training and validation.

## 4.1 Split Training and Validation Data

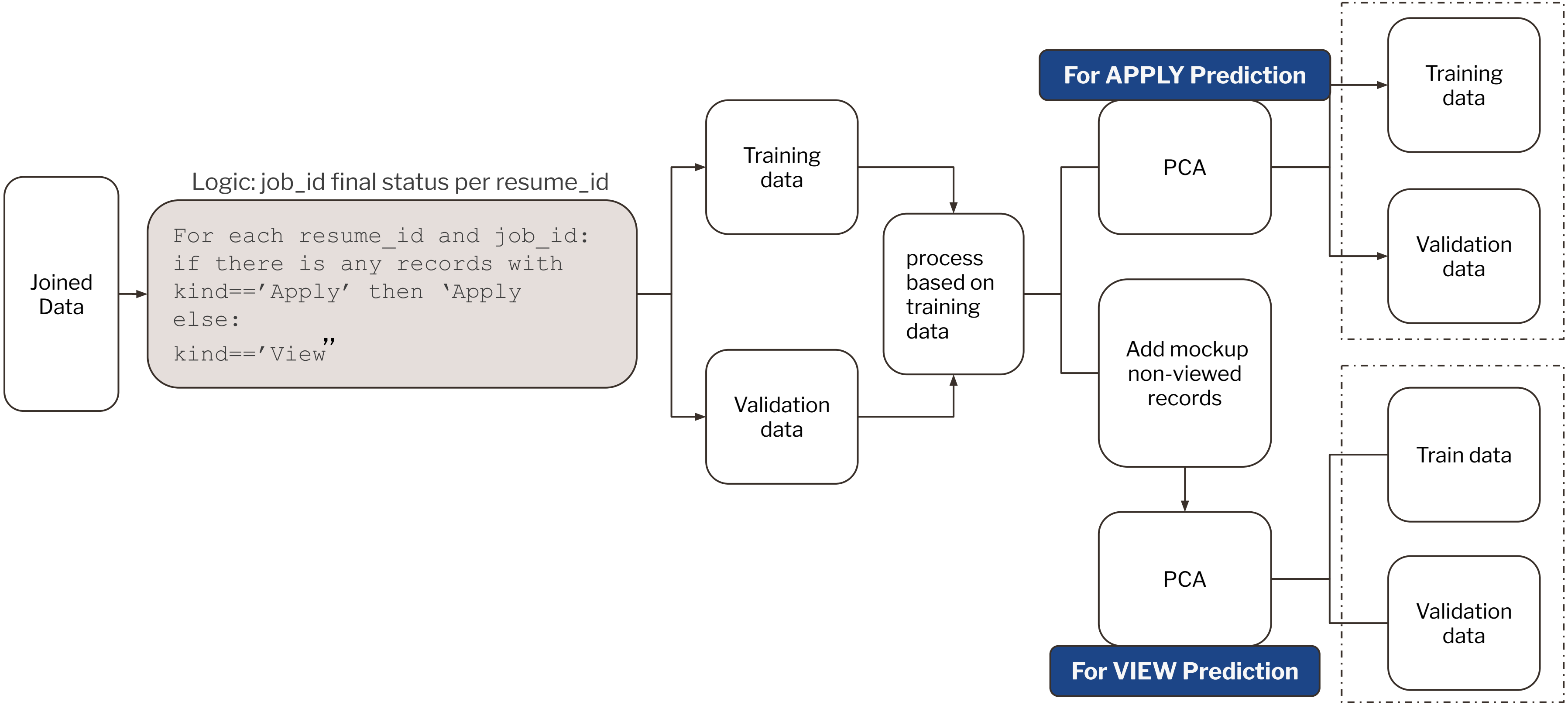
**Data for VIEW Prediction**

- Split the training-validation per resume_id to make sure all valid applicants can be found in the dataset
- Mockup records that are not viewed by the user, based on jobs that viewed or applied by other applicants.

**Data for APPLY Prediction**

- Split the training-validation per resume_id to make sure all valid applicants can be found in the dataset.

# 4.1 Split Training and Validation Data



# 4.2 Transforming Training and Validation Data

| Column | Data Type | Transform |
|---|---|---|
| **distance** | float(64) | StandardScaler |
| **farthest_distance_to_center_km** | float(64) | |
| **shortest_distance_to_center_km** | float(64) | |
| **average_distance_to_center_km** | float(64) | |
| **salary_value** | float(64) | |

# 4.2 Transforming Training and Validation Data

| Column | Data Type | Transform |
|---|---|---|
| **event_platform** | String | OneHotEncoder |
| **work_type** | String | |
| **region_code** | String | |
| **salary_unit** | String | |

## 4.2 Transforming Training and Validation Data

| Column | Data Type | Transform |
| --- | --- | --- |
| **classification** | String | OrdinalEncoder |
| **sub_classification** | String | |
| **resume_id_cat** | String | |
| **title_keywords** | String | TfidfVectorizer |
| **abstract_content_keywords** | String | |

# 4.3 Model Performance Evaluation

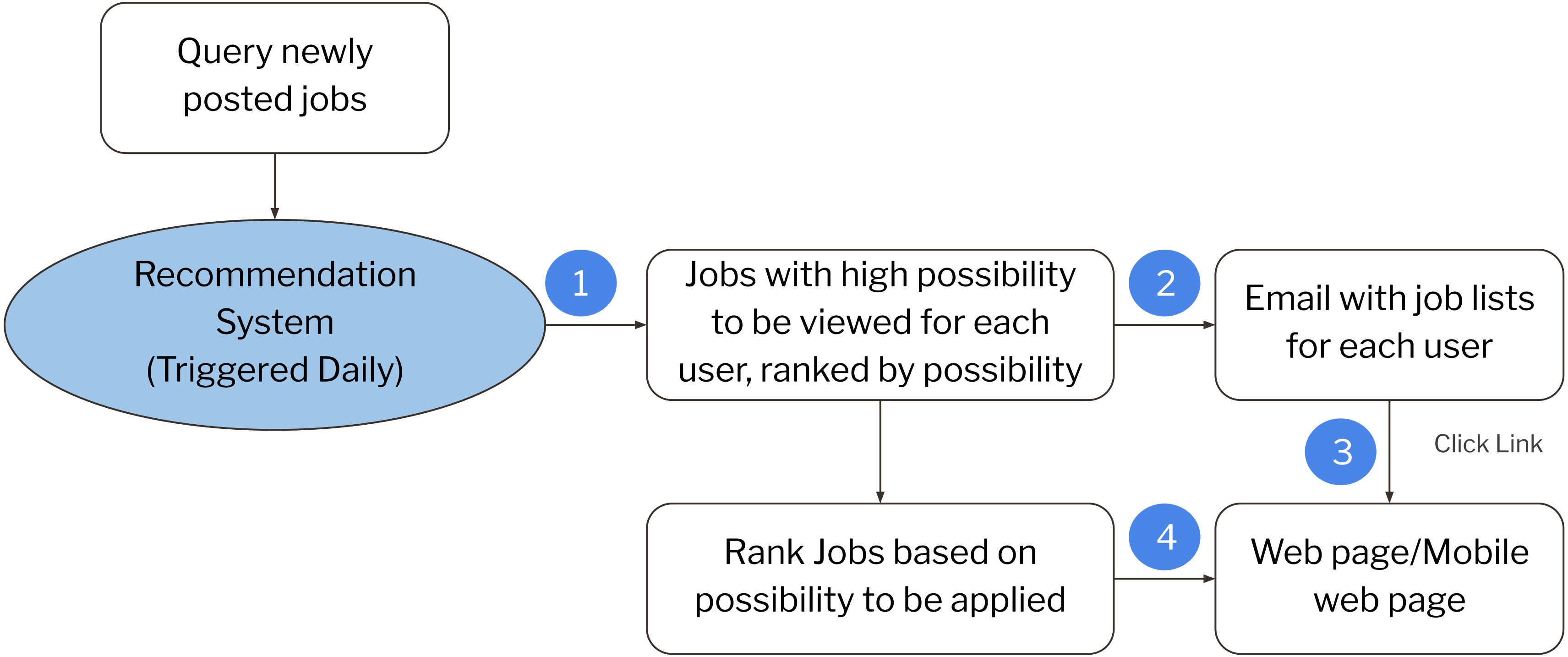
VIEW model: Based on 500 applicants, due to doubled data volume and limited memory.

| View Prediction | Accuracy | F1 | Precision | Recall |
|---|---|---|---|---|
| Random Forest 👌 | 59.2% | 0.7 | 68.7% | 72% |
| Neural Network | 67.5% | 0.8 | 67.5% | 100% |

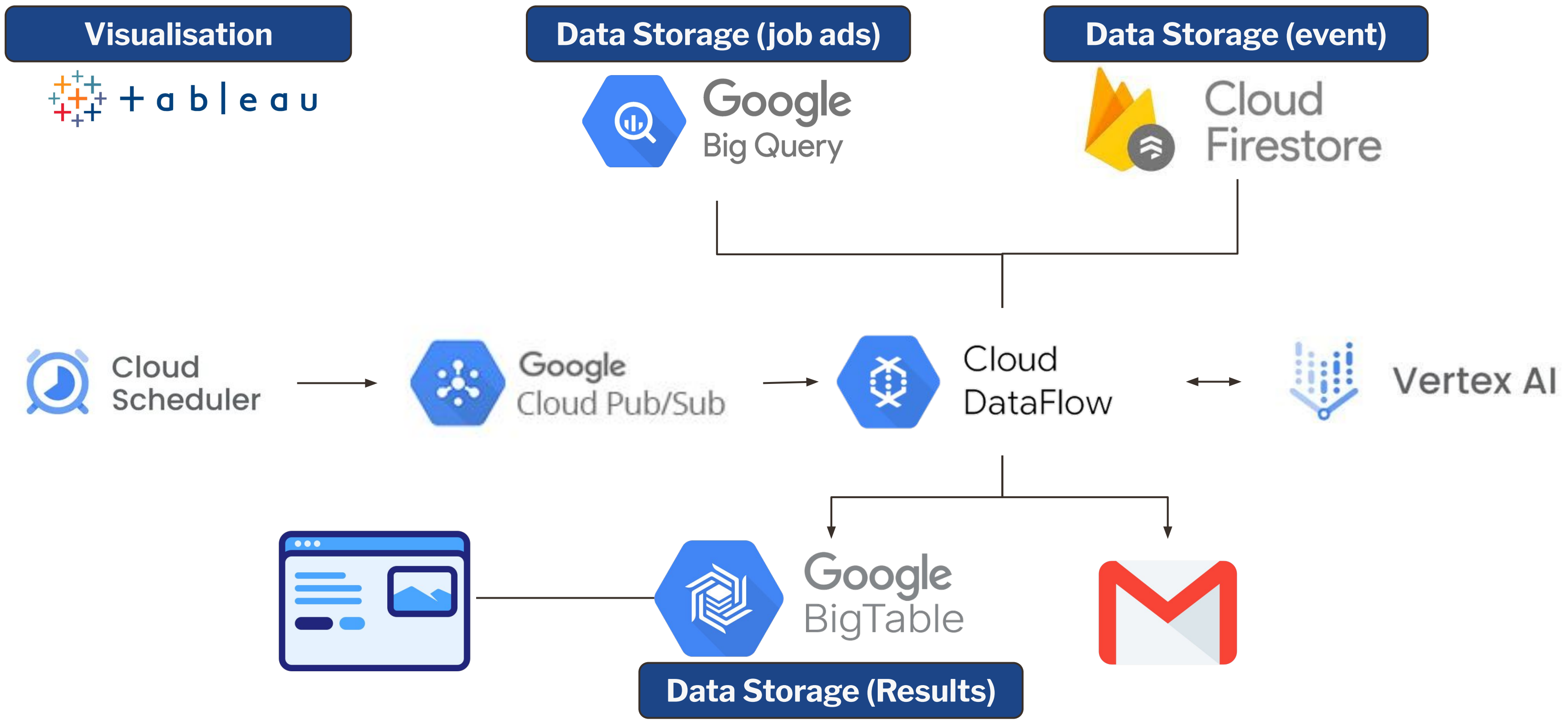
APPLY model: Based on all applicants.

| Apply Prediction | Accuracy | F1 | Precision | Recall |
|---|---|---|---|---|
| Random Forest 👌 | 89.6% | 0.6 | 56.2% | 63% |
| Neural Network | 87.9% | 0.004 | 88.4% | 0.4% |

# 4.4 Business Workflow



# 4.5 Pipeline


Visualisation
tableau
Data Storage (job ads)
Google Big Query
Data Storage (event)
Cloud Firestore
Cloud Scheduler
Google Cloud Pub/Sub
Cloud DataFlow
Vertex AI
Google BigTable
Data Storage (Results)

# 5. Business Value

# 5.1 Dual-Layer AI: Driving Platform Growth Through Smarter Engagement

- **Increase User Engagement**: Re-engage inactive users with hyper-targeted job alerts.
- **Boost Conversions**: Turn clicks into applications with AI-ranked recommendations.
- **Improve Marketplace Health**: Satisfy candidates (better jobs) and hirers (faster hires).

| Metric | Before | After | Impact |
|---|---|---|---|
| Email Open Rate | 15% | **35%** | 2.3x more users re-engaged |
| Click-to-Apply Rate | 20% | **45%** | 125% increase in conversions |
| Hirer Retention | 75% | **88%** | 13% improvement |
| Annual Revenue Growth | 5% | **7%** | Increased ad spend & fees |

## 5.2 Increase User Engagement

**Impact**

- **Personalized Job Alerts**: Users receive emails for roles matching their preferences (e.g., remote work, salary, work type).
- **Higher-Quality Matches**: AI-ranked jobs reduce "search fatigue" – users apply to 2x more relevant roles.
- **Faster Hiring**: Average time-to-application drops from 7 days to **3 days**.

# 5.3 Boost Conversion

**Impact**

- **Higher-Quality Applicants**: AI prioritizes jobs for users most likely to apply, reducing unqualified applicants.
- **Cost-Per-Hire Reduction**: From $3,000 to $1800 due to faster role fulfillment
- **Competitive Edge**: Hirers using AI tools renew subscriptions at **2x the rate** of non-users.

# 5.4 Improve Marketplace Health

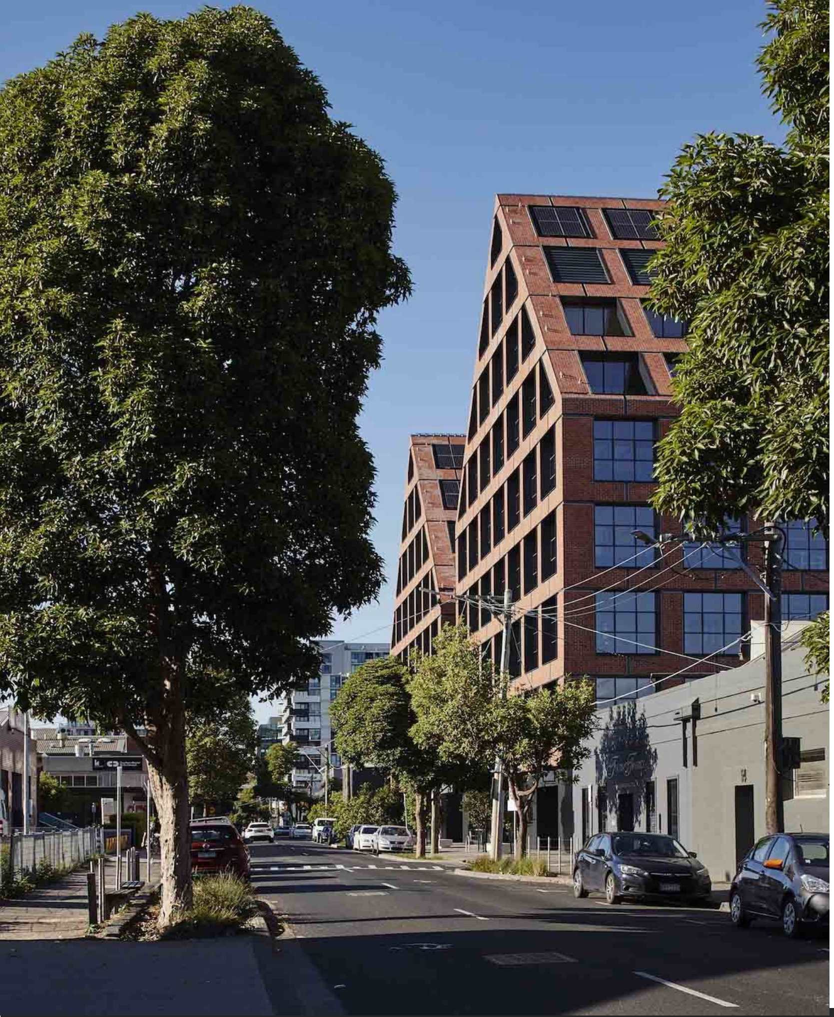
## Impact

- **User Retention**: 15% increase in monthly active users (MAU).
- **Hirer Loyalty**: 20% rise in premium subscription upgrades.
- **Data Flywheel**: More user/hirer activity → richer data → better AI predictions.
- **Market Positioning**: Differentiates platform as AI-driven and candidate/hirer-centric.

## Future Opportunies

- Expand to SMS/push notifications (+30% engagement potential).
- Monetize insights (e.g., "Top 5 Jobs You'll Apply To" premium reports).


# Q&A Time



# Thank you!

# Appendix:  GitHub Repo

https://github.com/annabellachen/seek_project.git