Linkedin

JOP Analysis

focuses on analyzing the global job market using LinkedIn job postings. We wanted to turn raw job listing data into useful insights for understanding hiring trends, in-demand skills, and market patterns. We combined Python for data cleaning, feature engineering, and machine learning with Power BI for an interactive dashboard.

let's Start

Problem Statement

1. Job markets change rapidly and are highly competitive
2. huge amounts of job data, but it’s messy and unstructured
3. Difficult for job seekers to know which skills to focus on
4. Companies struggle to quickly identify hiring trends

Goal of the Project

Turn messy LinkedIn job data into clean, structured insights 1

Identify hiring trends, top skills, and in demand industries 2

Understand job market patterns across locations and time 3

Present insights in an interactive dashboard for easy exploration 4

Data Overview

Dataset

31,475

LinkedIn job postings

Columns

- job title – Title of the posted job
- company name – Company offering the job
- location – City, region, and country
- hiring status – Posting status (early applicant, etc.)
- date – Posting date

Columns

- seniority level – Required experience level
- job function – Main role function (IT, Sales, etc.)
- employment type – Full-time, part-time, contract, etc.
- industry – Industry category of the job
- city – Extracted city name
- country – Extracted country name

Data Cleaning Problem

1. Web scraped data had unusual issues

2. Emojis, random spaces, and formatting errors

Solved using NLP text cleaning techniques

Feature Engineering

Columns Created

- City & Country: Extracted from location text to enable geographic analysis.
- Job Title Category: Grouped similar job titles into broader categories for trend detection.

Columns Created

- Posting Weekday: Day of the week each job was posted, to identify posting patterns.
- Country Job Density: Number of postings per country to measure hiring concentration.

Columns Created

- Seniority Code: Numeric code representing seniority levels, useful for modeling.
- Month & Day of Year: Extracted from posting date for seasonal and time-based analysis.

Machine Learning Models

- 60.69% MeanShift
- 60.43% Kmean
- 59.84% AgglomerativeClustering
- -1.00 DBSCAN

Model Comparison

Before PCA

```
KMeans Silhouette Score: 0.4196
==================================================
AgglomerativeClustering Silhouette Score: 0.3237
==================================================
DBSCAN: Only one cluster found. Silhouette Score skipped.
==================================================
MeanShift: Only one cluster found. Silhouette Score skipped.
==================================================
GaussianMixture Silhouette Score: 0.4013
==================================================
```

Model Comparison

After PCA

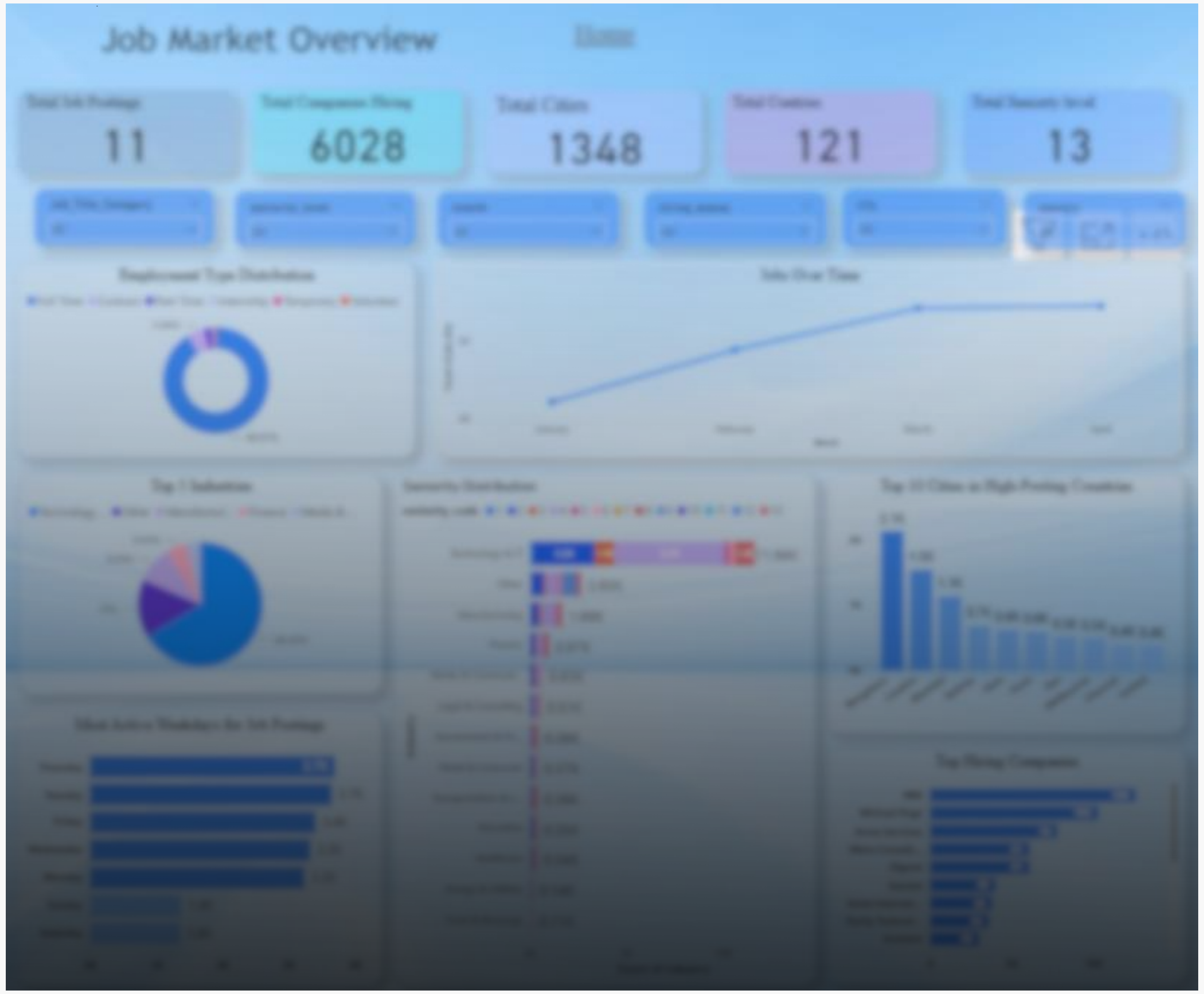
```
KMeans Silhouette Score: 0.4310
==================================================
AgglomerativeClustering Silhouette Score: 0.3843
==================================================
DBSCAN: Only one cluster found.
==================================================
MeanShift Silhouette Score: 0.6069
==================================================
GaussianMixture Silhouette Score: 0.2700
==================================================
```

Model Comparison

After Hyber Parameter Tuning

```
=== KMeans ===
Best Params: {'n_init': 10, 'n_clusters': 2, 'max_iter': 200, 'init': 'random'}
Best Silhouette Score: 0.6043

=== AgglomerativeClustering ===
Best Params: {'n_clusters': 2, 'linkage': 'ward'}
Best Silhouette Score: 0.5984

=== DBSCAN ===
Best Params: None
Best Silhouette Score: -1.0000

=== MeanShift ===
Best Params: {'bandwidth': None}
Best Silhouette Score: 0.6069

=== GaussianMixture ===
Best Params: {'n_components': 2, 'covariance_type': 'spherical'}
Best Silhouette Score: 0.5644

=== Final Best Results ===
KMeans: Score=0.6043, Params={'n_init': 10, 'n_clusters': 2, 'max_iter': 200, 'init': 'random'}
AgglomerativeClustering: Score=0.5984, Params={'n_clusters': 2, 'linkage': 'ward'}
DBSCAN: Score=-1.0000, Params=None
MeanShift: Score=0.6069, Params={'bandwidth': None}
GaussianMixture: Score=0.5644, Params={'n_components': 2, 'covariance_type': 'spherical'}
```

Explore Our Dashboard



Thank You