EMERGING TECH CITIES

Top Cities Where New Tech Professionals Can Thrive

Mak

&

Calvin

Overview

Business Problem: Many entry-level tech professional & entry-level startup groups are relocating to cities with growing tech industries. However, the availability of job opportunities and financial sustainability in these cities remain key concerns.

Primary Stakeholders:

- **Tech Professionals:** Benefit from insights into cities with promising career prospects and manageable living costs.
- **Tech Companies & Startups:** Gain access to data on cities with a growing talent pool and favorable business conditions.

Objective: This project aims to build classification models to help new tech professionals identify cities that:

- **Have a strong and expanding tech sector.**
- **Provide ample entry-level job opportunities.**
- **Are financially sustainable based on entry-level salaries and living costs.**

Data Understanding

Dataset & Features:

We chose multiple data sets to help us calculate two major indices:

- **Tech-Friendliness Index:** (Employment_per_1k_jobs * 0.25) + (Location_Quotient * 0.20) + (Average_7_Year_Growth_Rate * 0.20) + (Entry_Level_Tech_Wage * 0.15) + (Number_of_Tech_Events * 0.20) + (Layoffs_per_Capita * 0.10)
- **Financial Comfort Index:** Entry Level Tech Wages / ((Cost of Living Index * 100)+ Total Taxes of the State)

Target Variable (Classification Categories): What we are trying to predict, by using categorical variable that describes different combinations of ***"Tech Friendly"*** and ***"Cost-Effective"*** are five distinct categories. Locations that are:

- **Mid Tech Friendly, Mid Cost-Effective**
- **Not Tech Friendly, Cost-Effective**
- **Not Tech Friendly, Expensive**
- **Tech Friendly, Cost-Effective**
- **Tech Friendly, Expensive**

Challenges: Mention any challenges with the data (e.g., missing values, imbalanced classes, etc.)

Data Processing

For Data Sets:

- **Data Cleansing**: Standardized city names for consistency across datasets.
- **Database Usage**: In-memory SQLite database to merge data efficiently.
- **SQL Joins**: Merged the data using SQL joins based on city names.

For the Machine Learning:

- **Data Cleaning:** Missing values were handled, ensuring that only complete data for the relevant indices were used.
- **Threshold-Based Classification:** Cities were classified into categories based on Tech_Friendly_Index and Financial_Comfort_Index using percentile-based thresholds.
- **Label Encoding:** Class labels were converted into numeric labels for use in machine learning models.
- **Standardization:** The features were standardized to ensure equal scaling for the model.

Modeling Goal

Determining Business Recommendations: From all the analysis and modeling that has been done, with we determined two major models to be ones the three major stakeholders, First Entry Tech individuals, Startup Companies, Economic Developers should look after. The two different models:

- **Decision Tree**
- **Extra Tree Classifier**

General Limitations: Some models may have barriers when doing predictions, such as:

- **Small Dataset**: All models are based on a small sample size, reducing generalizability to a wider range of cities.
- **Imbalance in Classes**: Some categories have very few examples, leading to class imbalance and potential bias.
- **Bias in Categories**: Underrepresented categories can skew predictions for some cities.

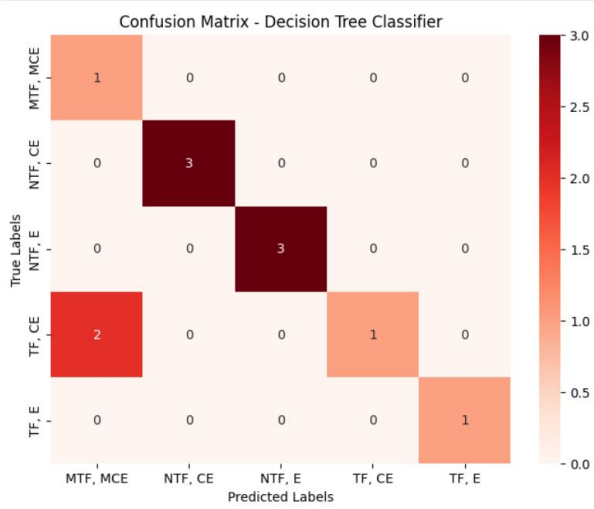
1. Tech Professionals (Decision Tree)

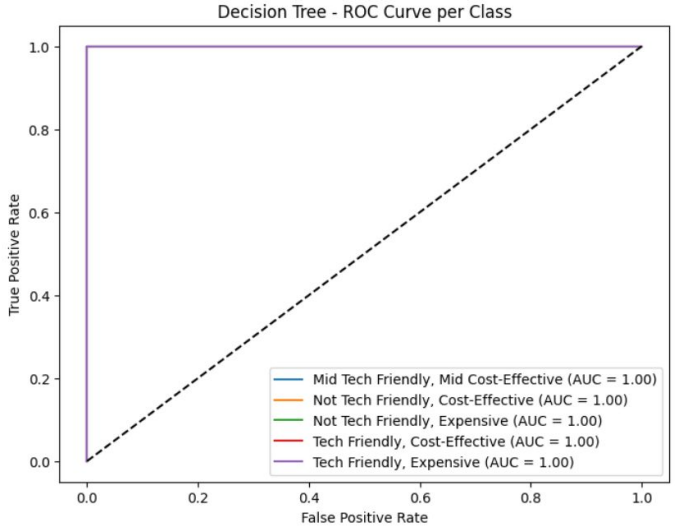
Reason:

- The **Decision Tree** model provides the best balance of **high accuracy (0.9091)** and performance across categories.
- Ideal for tech professionals looking to understand which cities are both tech-friendly and offer manageable living costs.
- **Good recall and precision for key categorie**s, making it useful for tech professionals when evaluating potential relocation options.

Limitations/Barriers:

- **Overfitting**
- **Limited Data**
- **Bias in Categories**






```
===== Decision Tree =====
Accuracy: 0.9091
Classification Report:
```

	precision	recall	f1-score	support
Mid Tech Friendly, Mid Cost-Effective	0.50	1.00	0.67	1
Not Tech Friendly, Cost-Effective	1.00	1.00	1.00	3
Not Tech Friendly, Expensive	1.00	1.00	1.00	3
Tech Friendly, Cost-Effective	1.00	0.67	0.80	3
Tech Friendly, Expensive	1.00	1.00	1.00	1
accuracy			0.91	11
macro avg	0.90	0.93	0.89	11
weighted avg	0.95	0.91	0.92	11

2. Tech Startups (Extra Trees Classifier)

Reason:

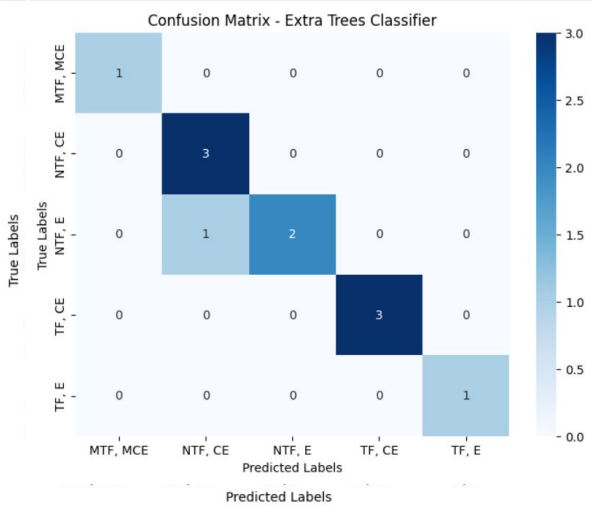
- The **Extra Trees Classifier** also achieves an **accuracy of 0.9091**, but it excels with even more consistent performance across all categories,
- Key category it **persists in is, "Not Tech Friendly, Cost Effective" cities,** which are most relevant to tech companies looking for both a place to expand tech talent and manageable operational costs.
- **Higher macro average precision (0.95)** makes it a better choice for identifying cities with the ideal mix of tech industry potential and financial sustainability.

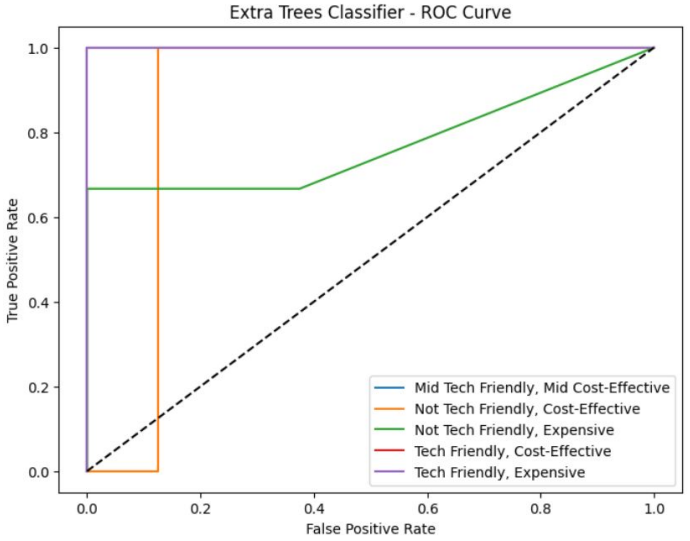
Limitations/Barriers:

- **Overfitting**
- **Limited Data**
- **Bias in Categories**

```
===== Extra Trees Classifier =====
Accuracy: 0.9091
Classification Report:
```

	precision	recall	f1-score	support
Mid Tech Friendly, Mid Cost-Effective	1.00	1.00	1.00	1
Not Tech Friendly, Cost-Effective	0.75	1.00	0.86	3
Not Tech Friendly, Expensive	1.00	0.67	0.80	3
Tech Friendly, Cost-Effective	1.00	1.00	1.00	3
Tech Friendly, Expensive	1.00	1.00	1.00	1
accuracy			0.91	11
macro avg	0.95	0.93	0.93	11
weighted avg	0.93	0.91	0.91	11





How we dealt with the Limitations (Next Steps)

1. Small Dataset

- **Increase Dataset Size**: Seek additional data sources or generate synthetic data.
- **Cross-Validation**: Use k-fold cross-validation to assess model performance.
- **Data Augmentation**: Generate variations of existing data if possible.

2. Class Imbalance

- **Resampling**: Apply oversampling (SMOTE) or undersampling to balance classes.
- **Adjust Class Weights**: Modify model algorithms to prioritize minority classes.
- **Alternative Metrics**: Use F1-score or weighted precision/recall instead of accuracy.

3. Overfitting

- **Regularization**: Apply L2 regularization or limit tree depth to reduce overfitting.
- **Pruning**: Prune tree models to avoid overly complex decision boundaries.
- **Ensemble Methods**: Use Bagging or Boosting to combine model predictions and reduce overfitting.

THANK YOU

FOR YOUR

COOPERATION

Any Questions???