Project Report

Project Name: Branch / Location Recommendation System: Optimal Branch / Location Selection with Machine Learning

Student Name&Surname: Ege HARPUTLU

General Summarized Info Of My Term Project:

Branch / Location Recommendation System: Optimal Branch / Location Selection with Machine Learning which predicts potential revenue for alternative locations and recommend the most suitable locations for investors, institutions, or franchise companies for the coffee or restaurant businesses.

Features:

- Population_Density
- Competitor_Count
- Rent_Price
- Daily_Foot_Traffic
- Average_Income
- Distance_To_Center
- University_Count
- Location (categoric value)
- Estimated_Revenue (output / result feature, I mean this may also considered as a feature)

Label: Estimated_Revenue (continues value)

1. INTRODUCTION:

This project (Branch / Location Selection with Machine Learning) aims to design a machine learning model that assists businesses, particularly in the hospitality and retail industries (restaurants, cafes, etc.), to identify the most profitable franchrise locations for their new

branches. By analyzing and predicting revenue potential, this project enables business owners to make informed and data-driven decisions when expanding their operations. Factors such as population density, competitor count, rental costs, daily foot traffic, average income levels, and proximity to city centers, and more are considered to evaluate a location's business potential by predicting the revenue by these features written and more.

In today's competitive market, selecting the right location is critical for success. A poor choice of location can lead to significant revenue losses and operational inefficiencies, while a strategically chosen location can maximize profitability and enhance brand visibility. By leveraging machine learning, this project aims to streamline the decision-making process, reduce reliance on intuition, and minimize risks associated with expansion. The ability to estimate revenue accurately for different locations empowers businesses to optimize their resources and focus on high-potential areas.

This project implements and evaluates two machine learning algorithms (**Linear Regression** and **Random Forest Regressor**) to predict the revenue potential of various locations for cafes, restaurants, etc. The comparison of these models will be based on evaluation metrics such as **Mean Absolute Error (MAE)**, **Mean Squared Error (MSE)**, **Root**

Mean Squared Error (RMSE), and **R^2 Score**. By analyzing the performance of these algorithms, the project will determine the most reliable model for predicting location-specific revenue. Also the combination of these two models can be used if both are very successful and perform well. For an example, if both perform very well, we can produce an output (prediction as result) by taking the mean of these two models . Or accordingly, other taking mean methods can be applied.

Additionally, this study highlights the transformative role of machine learning in location-based decision-making and its capability to provide actionable insights for strategic business growth, especailly for cafes and restaurants.

2. PROJECT PURPOSE:

The purpose of this project is to create a predictive model that can estimate the revenue potential of various locations based on key features such as population density, rental costs, daily foot traffic, competitor presence, average income of the environment population, distancity to the center of the city, etc. This has several practical applications across various sectors, particularly for franchise businesses, restaurant owners, and real estate strategists or investors about venue operations or other venue operators. The applications are like below:

1. Optimal Location Selection: Businesses can utilize the model to evaluate potential locations and identify those with the highest revenue potential. This ensures data-driven decisions that minimize risks and maximize profitability during branch expansion. Also with this, the investors who want to expand their businesses can make more analytic decisions. Otherwise, they can make decision with their feelings, etc. But at this point, the success and performance of model is very critical.

2. Investment Decisions: Franchise owners and investors can use the system to prioritize high-potential locations for investment, ensuring better returns and reduced risks.

3. Revenue Forecasting: By accurately estimating revenue, business owners can develop better financial forecasts, allocate resources efficiently, and plan their operations effectively.

4. Strategic Market Positioning: The model can provide insights into market competition by analyzing competitor density in a given area, helping businesses strategically position their branches to gain a competitive advantage.

5. Urban Development and Commercial Planning: Policymakers and urban developers can leverage these insights to identify areas with high commercial demand, enabling balanced urban development and infrastructure planning.

This project employs machine learning techniques to understand and quantify the relationships between location-specific features and estimated revenue. By comparing the performance of Linear Regression and Random Forest, it is aimed to identify the most effective model for accurate revenue predictions. This comparative analysis will also highlight the practical utility of machine learning in optimizing location selection, showcasing its ability to transform decision-making processes in the business and commercial sectors.

Also, even if the desired performance and success I expect are not achieved in this project, the outcomes (conclusion) will still be documented and evaluated as a conclusion. With this way, the reasons for the lack of success can be identified, providing valuable experience and knowledge for future projects. So, also, this can be considred as a purpose of the project.

3. MACHINE LEARNING ALGORITHM SELECTION:

This project is a supervised learning problem, specifically a regression problem. Here's why, detailly:

A. Supervised Learning

- Definition: In supervised learning, the model is trained on a labeled dataset where the input features are used to predict an output label (target variable).

  For this project:

- Features: Location-specific data such as population density, rental costs, daily foot traffic, competitor count, average income, distancity to center, number of university if exists around, and location, etc.

- Target Variable (OUTPUT): Estimated revenue (a continuous numeric value) for the areas with spesific location-based local features.

- Training Process: The dataset consists of examples where both the features and the target revenue are known. The model learns the relationship between these features and the target, and also correlations between each feature and target if needed, during training. Once trained, it can make predictions for new, unseen locations based on the learned patterns. For example after the training process, the model can pedict the potential revenue of a different venue (e.g. restaurant) with different values of same features. So when a new column considered without a target value (revenue), the model can produce a prediction for the target (revenue) field.

B. Regression Problem

- Definition: Regression is a type of supervised learning where the goal is to predict a continuous value, rather than a discrete class label (as in classification problems).

- Relevance to This Project:

    - The output variable in this case is estimated revenue of a business, which is a continuous value (e.g., $100,000, $150,000, etc.).

    - Predicting a continuous target variable fits the definition of a regression problem, making regression algorithms the appropriate choice.

C. Why Regression?

1. Continuous Output:

    - The goal is to predict revenue, which is a continuous numeric variable. Regression algorithms are specifically designed for these tasks.

2. Feature Relationships:

- Many features (like population density, rental costs, and foot traffic) may have linear or non-linear relationships with revenue target, which regression algorithms can effectively model.

3. Versatility of Algorithms:

    - Regression techniques allow us to handle both simple linear relationships and more complex, non-linear interactions between features and the target. So, there are spesific algorithms that may be used for each task. I can decide the algorithm based on what I want and what is okay for my project task.

D. Why NOT Classification?

While classification algorithms are powerful for tasks where the output (label) is categorical (e.g., predicting the target variable as "high-revenue", “mid-revenue” or "low-revenue"), but this project requires a **continuous numerical prediction** for revenue, such as $120,000 or $350,000.

Classification could have been also used / selected if the problem was framed as determining revenue ranges (e.g., "low", "medium", "high"), but this approach would result in a loss of precision. Also maybe the target customers who use this project would not satisfied with categorical labels as classified without exact analytical numbers and this may cause low information about the revenue prediciton and this may increase the satisfaction of the customers of this project.

Since predicting an exact revenue value is critical for decision-making in this task, regression is the more logical and effective choice. So, I did not prefer to choose classiffication.

E. Type of Regression Algorithms I Could Use

Several regression algorithms would be applied to this problem, each with its own strengths and weaknesses:

1. Linear Regression:

- Assumes a linear relationship between features and the target variable.

- Advantages:
  - Simple and interpretable.
  - Useful when relationships between features and target variable are linear.

- Limitations:
  - Struggles with non-linear relationships and outliers.

2. **Random Forest:**

   - An ensemble method that combines multiple decision trees to improve prediction accuracy.

   - Advantages:
     - Handles non-linear relationships well.
     - Reduces overfitting, well.
     - Robust to missing or noisy, and also works well with complex data.

   - Limitations:
     - Less interpretable compared to linear regression and single decision tree algorithms.
     - Computationally intensive for large datasets.

3. **Decision Trees:**

   - Splits data into branches based on feature values to make predictions.

   - Advantages:
     - Captures non-linear relationships and mid-complex data.
     - Easy to interpret with smaller datasets.

   - Limitations:
     - Prone to overfitting unless properly pruned.
     - Performance may drop compared to ensemble methods like Random Forest.
     - Generally works well with small or mid datasets.

4. **Support Vector Machines (for Regression):**

   - Uses support vectors to find a hyperplane that minimizes prediction error.

   - Advantages:

- Effective for high-dimensional data.
- Can model non-linear relationships with kernel functions.

- Limitations:
  - Computationally expensive for large datasets.
  - Requires careful parameter tuning.
  - If the hyperplane is not placed in the correct area, this may cause wrong predictions and also overfitting or even underfitting.

F. Selected Algorithms for This Project

In this project, I selected two algorithms for comparison and to use:

1. Linear Regression:

- Chosen for its simplicity and interpretability.

- Helps me understand the basic relationships between features and target variable.

- I know it has critical limitiations like it assumes a linear relationship which not might always hold true in real-world scenarios but it's worth it for its simplicity and interpretability. Also I want to highlight that the features in my project are generally and approxiemtly have LINEAR or LOW-LINEAR relationships with the target variable (revenue) until a saturation point, at least. For example, population density, one of the feautures in my dataset: A more densely populated area generally indicates a higher customer potential, which can directly increase revenue and this may exhibit a linear relationship, at least approximetly. This also applies to most of the features like average income, daily foot traffic, etc. Most of the features are much low-linear relationships with the target variable as they are like linear until / to a one point (saturation point), but after that point, they may lose their linear relaitonships. To give an example: Think about one of the features "competitor count": As competition (competitor count) increases, revenue may decrease, but beyond a certain point (when competition becomes excessive), revenue can drop / decrease significantly and this may destroy the lineary, at all. And this type of relationship typically exhibits a non-linear trend after a point. But since most of the features have like middle or low linear relationships with the target variable, generally and approixemtly at least until a saturation point, it's worth to use this algorithm when consider

the advantages of linear regression. But some of the features don't have any linear relationships, for instance: "location". **In this project how this explained low-linearity and saturation points affects the performance of linear regression algorithm and the compariosn between random forest algorithm will be observed. And also it will be observed it is really worth or not worth to use this algorithm because of its simplicity and interpretebility, in low or non linear relaitonship-datasets**

- Also one of important critical limitiations of Linear Regrssion algorithm is that it is sensitive to outliers but when I analyse my dataset used in this project, I could not find any critical outliers. So this limitiation is not a big deal for my project, for now, at least.

2. **Random Forest (Regression):**

   - Chosen for its ability to handle non-linear relationships, complex data, and capture complex interactions between features. Actually, this algorithm helps me with the limitiations of linear regression algorithm.

By comparing these two algorithms, I aim to determine which method provides the most accurate predictions. Evaluation metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R^2 Score will be used to measure the effectiveness of each approach. This comparison will provide valuable insights into the capabilities of different regression algorithms in location-based revenue prediction tasks.

4. EVALUATION METRICS:

Since this project involves a regression problem, the performance of the machine learning models will be evaluated using the following metrics:

1. Mean Absolute Error (MAE)

- Definition: The average of the absolute differences between the predicted and actual values. It measures the average magnitude of errors in a set of predictions, without considering their direction.

- Interpretation:

- Provides a straightforward measure of how far the model's predictions deviate from the actual values on average.

- MAE is less sensitive to large outliers compared to other metrics like MSE or RMSE.

2. Mean Squared Error (MSE)

- Definition: The average of the squared differences between the predicted and actual values. It penalizes larger errors more than smaller ones because of the squaring term.

- Interpretation:

    - MSE emphasizes large errors due to squaring, making it useful for highlighting models that fail to predict well for certain observations.

    - While useful for optimization during training, the squared nature makes it harder to interpret directly in the original units of the target variable.

3. Root Mean Squared Error (RMSE)

- Definition: The square root of the MSE, providing an error measurement in the same units as the target variable. It is often preferred for interpretability when comparing errors in real-world terms.

- Interpretation:

    - RMSE gives more weight to large errors (like MSE) but makes the result interpretable by converting it back to the original scale of the target variable.

    - A smaller RMSE indicates better model performance, but its sensitivity to outliers should be considered.

4. R-squared (R^2)

- Definition: Also known as the coefficient of determination, R^2 measures how well the independent variables explain the variance in the target variable. It quantifies the proportion of variance in the target that is predictable from the features.

- Interpretation:
    - It provides a useful summary and very understanding of how well the features collectively explain the target variable. So it very easy to apply and have an idea about the performance of the model.
    - A value of 1 means perfect prediction, while 0 means terrible prediciton.

Why These Metrics in This Project?

1. MAE: Provides an intuitive sense of the average prediction error in absolute terms, making it easy to interpret.
2. MSE and RMSE: Penalize larger errors more, which is useful for identifying underperforming models.
3. R^2: Offers a clear measure and understanding of the overall explanatory power of the model, helping us understand how much of the target variable's variance is captured by the features. It is very easy to understand and to have an idea about the general performance at the first look.

By using a combination of these metrics, we can thoroughly evaluate the models' accuracy, robustness, and predictive reliability. This ensures a well-rounded analysis of the models' performance in predicting location-based revenue.

5. CHALLENGES IN THIS PROJECT:

This project presents several unique challenges due to the complexity of the relationships between features and their impact on revenue. These challenges must be addressed to improve the model accuracy and reliability:

1. Non-linearity

- Challenge:
    - The relationship between some of features (e.g., population density, rental costs, competitor count) and the revenue potential is often non-complete-

linear and even it is considered as a approximetly low or middle linear, it lose its approximet-linear after a limit saturation point. For instance, while an increase in daily foot traffic might initially boost revenue, there may be a saturation point where further increases yield diminishing returns. So this shows that some of the features can be considered as they have a linear relationshps with the target variable but even this may, they can lose their linerity after a point. So there is no continous and certain linearity in the dataset.

- Also, some of fetaures like locations directly don't have any linear realitonships with the target variable.

- So, to sum-up, I can say that one the challenges for my project is that most of features lose their linear relaitonships and some of fetaures do not have any linear relationships or low/middle-linear relationships. But linear regression works well with the linear relationship-datasets.

- Solution?

    - Simple linear models may struggle to capture such non-linear patterns. Other models like Random Forest Regressor can handle non-linearity more effectively, as they split the data into smaller subsets to capture complex relationships. So random forest algorithm can be condireded as a alternative method when linear patterns disappeared mostly.

2. Multicollinearity

- Challenge:

    - Some features might be highly correlated with each other. For example:
        - University count and daily foot traffic may be strongly linked, as areas with higher university count around are likely to see higher foot traffic.
        - This can distort the coefficients in linear models and reduce their interpretability.
- Solution:

    - Techniques like correlation analysis can identify highly correlated features. If multicollinearity is present, methods such as regularization (e.g., Ridge or Lasso Regression) can be applied to stabilize the model by penalizing large coefficients or maybe the can be combined. **But I pereferd to handle with this challenge by removing the unnecessary features highly correlated**

with each other like university count (this feature is higly correlated with the daily foot traffic feature).

3. Outliers and Missing Data

- Challenge:
    - Outliers in the data can significantly impact the accuracy of regression models, especially Linear Regression. For example:
        - A location with abnormally high rental costs or revenue in an otherwise low-revenue area could skew predictions.

NOTE: When I analyse my dataset used in this project, I could not find any critical outliers or missing data. But even, when another arranging or additions applied to the dataset, it may cause outliers or missings in the future and that's why I wanted to explain this section.

- Solution if I faced any outliers or missing data:
    - Outlier detection techniques such as IQR (Interquartile Range) can help identify extreme values.
    - Depending on the context, outliers can be removed or transformed to reduce their impact on the model.
    - To handle the missing data, I can replace the missing data with the mean or mod of that column. Especially for missing categorical variables, mod can be used for the missing place.

4. Geographical Differences

- Challenge:
    - Location plays a crucial role in determining revenue potential, but it is often difficult to encode geographical data effectively. For instance:
        - Suburban, Urban, and Rural areas may have vastly different revenue potentials, and encoding these as simple categorical variables may not capture the full complexity of geographical differences.
- Solution:

- Advanced encoding techniques (here, One-Hot Encoding) can be applied to location data. The aim is to convert the categorical values into numerical values. So the algorithms can be implemented easier.

5. Data Sparsity and Generalizability

- Challenge:
  - The dataset may not fully represent the diversity of real-world locations, leading to poor generalization for unseen data. For example:
    - If the dataset contains mostly suburban areas, the model may perform poorly for urban or rural locations.
- Solution:
  - Ensuring a balanced dataset that includes a wide variety of location types is crucial.
  - Techniques such as data augmentation or obtaining more representative data can address this issue.

6. Feature Engineering Complexity

- Challenge:
  - Some features may not adequately represent their real-world impact on revenue. For instance:
    - "Competitor count" alone might not fully describe competition without considering the size or quality of the competitors. So here, competitor count feature can affect the output missingly which is very ciritcal for the all project.
- Solution:
  - Creating new features, such as competitor density or competitor quality metrics, can provide a deeper understanding of the competitive landscape.

7. Interpretability

- Challenge:
  - Random Forest is less interpretable and business stakeholders often prefer interpretable models to understand the reasoning behind recommendations.

- Solution:

    - I used two different models as Random Forest and Linear Regression. So, if linear regression model achives the similar performance and accuracy with the random forest, the stakeholders can interprete by linear regression.

    - Also well-prepared reports, documentations, and presentations can solve this problem for stakeholders.

Addressing These Challenges

By systematically addressing these challenges, this project aims to build a robust and reliable recommendation system. Incorporating advanced modeling techniques, effective feature engineering, and thorough data preprocessing will enhance the model's predictive power while maintaining interpretability for stakeholders. These efforts will ensure accurate, actionable, and data-driven location recommendations for businesses.

6. FEATURE ENGINEERING:

The performance of regression models in this project largely depends on how effectively the features are selected, transformed, and engineered and the success is mostly depends on the dataset preproccesed. For this project, feature engineering plays a crucial role in capturing meaningful relationships between location-specific factors and revenue potential. The following approaches can enhance the predictive capability of the models:

1. Interaction Terms

- What?
    - Interaction terms involve combining two or more features to capture the joint effect they may have on the target variable.

- Relevance:
    - In this project, the interaction between features, such as daily foot traffic and population density, may provide more information about customer volume than the features independently.

- Example:
    - Create a new feature: Foot Traffic x Population Density
    - This interaction can reflect how customer potential scales in dense, high-traffic areas.

2. Handling Categorical Features

- What?
    - Categorical features like Location (e.g., Urban, Suburban, Rural) need to be converted into numerical formats to be usable in algorithm models implemented as code.

- Relevance:
    - A location can significantly influence revenue potential, but treating it as text data won't be meaningful for the model. Encoding techniques like One-Hot-Encoding can transform this into a usable format for the models.

- Techniques as solution:
    - One-Hot Encoding: Create binary columns for each category (e.g., Urban = 1, Suburban = 0. If both are 0, then it is Rural) **(I prefered to use this method in my project to handle categorical variables because it very simple to apply and also very easy to read and understand)**.

    - Ordinal Encoding: Assign ordinal values to categories if there is an inherent order (e.g., Rural = 0, Suburban = 1, Urban = 2, ….).

3. Feature Scaling

- What?
    - Scaling ensures that all features are in a similar range, which is especially important for models like Linear Regression, Support Vector Machines, etc. that are sensitive to feature magnitudes. This is very important to make the model understands the all variables in the same unit.

- Relevance:
    - In this project, features like Population Density (e.g., thousands per square km) and University Count (small integers), etc. operate on vastly different scales. Without scaling, features with larger values can dominate the model.

- Techniques:
    - Standardization: Scales data to have a mean of 0 and a standard deviation of 1.

    - Normalization: Scales data to fit within a range, typically [0, 1] (**I prefered to use normalization to handle scaling in my project because it is more simple and interpretable and with this, it is easy to read and understand the normalized dataset**).

4. Creating New Features

- What?
    - Deriving additional features from existing ones can capture hidden relationships and improve model performance.

- Examples for This Project:
    - Competitor Density: Divide the number of competitors by the population density to understand how crowded the competition is in a given area.

5. Addressing Multicollinearity

- What?
    - Highly correlated features can distort the model's coefficients and reduce interpretability in linear models.

- Relevance:
    - For example, University Count and Daily Foot Traffic are likely correlated. Keeping both as separate features might not add value.

- Solution:
    - Calculate a correlation matrix to identify highly correlated features. If two features are strongly correlated ($r > 0.8$), consider dropping one or combining them.
    - **I prefered to remove the feature of university count because it is too highly correlated with the fetaure of daily foot traffic.**

6. Dropping Irrelevant Features

- What?
    - Some features may not contribute significantly to the target variable and could may noisy.

- Relevance:
    - Features like "Distance to Center" might not always have a significant impact on revenue, depending on the business context and usage area of the project and models. Or maybe the lcoation is too far to center but there is a university nearby, so here he feature of ditance to center can lose its affect to the target variable.

- Solution:
    - Perform feature importance analysis (e.g., using Random Forest) to identify and drop critical low-impact features.

Why Is Feature Engineering Critical?

Feature engineering allows to:

1. Extract more meaningful patterns from the data.

2. Handle non-linear relationships that simple models like Linear Regression might miss.

3. Improve model performance by reducing noise and focusing on high-impact features.

So, the well-prepared dataset is very critical to perform the machine learning models because the dataset direclt affects the result and accuracy of the models.

By carefully engineering features, we ensure that the models not only fit the data well but also generalize effectively to new, unseen locations. This process is crucial for building a recommendation system that provides actionable and accurate predictions.

7. DATASET:

Here is the dataset used in the Branch / Location Recommendation System project, which consists of various features to predict the estimated revenue of venues like cafes or restaurants. Below is a description of the dataset, including a detailed explanation of its columns.

Columns and Their Descriptions:

1. **Population_Density (per sq km):**

    - Description: Represents the population density of the area where the venue is located.
    - Type: Numerical (integer).
    - Unit: People per square kilometer.
    - Importance: Areas with higher population density typically attract more foot traffic, potentially leading to higher revenue.

2. **Competitor_Count (count):**

    - Description: Indicates the number of competitors (e.g., other cafes, restaurants) in the vicinity.
    - Type: Numerical (integer).
    - Unit: Count (number of competitors).
    - Importance: More competitors might reduce revenue due to increased competition, though high competition may also indicate a commercially active area.

3. **Rent_Price (monthly, local currency):**

    - Description: The monthly rent price of the venue.
    - Type: Numerical (integer / double).
    - Unit: Local currency.
    - Importance: Higher rents may correlate with premium areas but may also affect profitability negatively.

4. **Daily_Foot_Traffic (people):**

    - Description: The number of people passing by the venue daily.
    - Type: Numerical (integer).
    - Unit: People (number of people).
    - Importance: High daily foot traffic can lead to higher customer conversion rates, thus increasing revenue.

5. **Average_Income (monthly, local currency):**

    - Description: The average income level of people living in the area.
    - Type: Numerical (integer).
    - Unit: Local currency.

- Importance: Higher average income levels often correlate with higher spending power, potentially leading to increased revenue.

6. **Distance_To_Center (minutes):**

   - Description: The walking distance (in minutes) from the venue to the city center.
   - Type: Numerical (integer).
   - Unit: Minutes.
   - Importance: Proximity to the city center is often an indicator of accessibility and foot traffic potential.

7. **Location:**

   - Description: Categorical variable indicating the type of area where the venue is located (e.g., 'Urban', 'Suburban', or 'Rural').
   - Type: Categorical.
   - Importance: Different location types may appeal to different demographics and impact revenue differently. This feature is encoded using one-hot encoding for machine learning purposes.

8. **University_Count (REMOVED FROM THE DATA-SET):**

   - Description: The number of universities if exists in the location explicited.
   - Type: Numerical (integer).
   - Unit: Count (number of universities)
   - Importance: **This was excluded from the dataset because its effect on the target variable (Estimated_Revenue) was already captured by other features, particularly Daily_Foot_Traffic.** Including this feature might have introduced multicollinearity, which can negatively impact the performance and interpretability of some machine learning models, such as Linear Regression. So, to sum-up, I realized and decided that this feature dependent to the Daily_Foot_Traffic feature so I decided to remove this feature from the data-set in the phase of dataset preprocessing to enhance the performance of the models.

9. **Estimated_Revenue (yearly, local currency) (OUTPUT FEATURE, LABEL)**

   - Description: The predicted yearly revenue of the venue. This is a output result feature.
   - Type: Numerical (continuous).
   - Unit: Local currency.

- Importance: This is the target variable (result) for the project, representing the revenue potential of a venue.

Sample Data (First 10 Rows):

A total of 2000 data-points (records) is used as dataset but in the report, just 10 example is given. %30 of the dataset is used for test and %70 of the dataset is used for training.

Population_Density	Competitor_Count	Rent_Price	Daily_Foot_Traffic	Average_Income	Distance_To_Center	University_Count	Location	Estimated_Revenue
7770	18	6539	808	115435	31	0	Suburban	80390
1360	6	10637	587	63919	52	1	Rural	193401
5890	16	5199	916	55247	19	0	Urban	53088
5691	19	13115	867	114758	58	0	Rural	113196
6234	3	9445	970	111747	47	0	Suburban	180883
6765	4	12375	457	82752	16	4	Urban	65479
966	6	3557	574	76573	53	2	Urban	29963
4926	12	6592	877	79101	5	0	Suburban	65123
6078	14	1098	555	46646	35	2	Suburban	135911

Additional Notes:

- This dataset is usable for regression problem tasks based on the explained features as the target variable (Estimated_Revenue) is continuous.
- Features like Population_Density, Daily_Foot_Traffic, and Average_Income are expected to have **approximately** linear relationships with the target variable, at least until / to a saturation point.
- Features like Competitor_Count and Rent_Price may exhibit **low**-linear patterns that can be effectively captured by models like Random Forest.
- The linearity between the features and target variable will be observed in this project. If they have really like approximetly linear relationships, then linear regression model is expected to be performed well.

- Since the dataset has generally non-linear or low-linear relationships, especially **AFTER a saturation point** between the features and target variable, the Random Forest algorithm model is expected to be performed better than Linear Regression.

- Location column will be encoded to a numerical value by One-Hot method.

- All numerical columns except encoded Location column will be normalized to scale the whole data-set in the like same unit. With One-Hot method, the Location column will be already normalized, automatically, so no need to normalize it again.

This dataset has been / will be carefully preprocessed, including handling missing values, encoding categorical features, and normalizing numerical features where necessary. It serves as the foundation for building and evaluating machine learning models in this project.

8. ALGORITHMS USED:

As explained in the section of 3. Machine Learning Algorithm Selection / F. Selected Algorithms for This Project; I prefered to use 2 algorithms:

1. Linear Regression: A basic regression algorithm that models the relationship between the target and features by fitting a linear equation. **This is chosen for its simplicity.**

2. Random Forest (for regression): An ensemble learning method that combines the predictions of multiple decision trees, improving prediction accuracy by reducing overfitting. **Chosen for its power of handling with complex and non-linear datasets and resistance to the overfitting.**

NOTE: For the details about why I selected these two algorithms, please take a look at the section of **3. Machine Learning Algorithm Selection / F. Selected Algorithms for This Project**, in this report.

The Code I Wrote by Python to Implement Algorithms:

I explained my code with steps and comments:

#Step 1: Load the Data and Required Libraries !!! !!! !!!

```
#Import required libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from google.colab import files

#Load the dataset
print("Please upload a file with CSV format:")
uploaded = files.upload()
file_name = list(uploaded.keys())[0]
dataset = pd.read_csv(file_name)

#Display the first 10 rows of the dataset to understand its structure
print("First 10 rows of the original dataset:")
print(dataset.head(10))

print("\nGeneral Info about the dataset:")
print(dataset.info())
```

#Step 2: Data Preprocessing !!! !!! !!!

```
#Remove Unnecessary Features
#Drop the 'University_Count' feature as it overlaps with other features like 'Daily_Foot_Traffic' ('University_Count' was removed because its effect on revenue is already represented by 'Daily_Foot_Traffic'.)
```

```
dataset = dataset.drop(columns=["University_Count"])

#Handle missing data if exists
print("Checking for missing values:")
print(dataset.isnull().sum())

#Fill numeric columns with the mean
numeric_columns = dataset.select_dtypes(include=["float64", "int64"]).columns
dataset[numeric_columns] = dataset[numeric_columns].fillna(dataset[numeric_columns].mean())

#Fill categorical columns with the mode. Because for categoric values, mod matters!
categorical_columns = dataset.select_dtypes(include=["object"]).columns
for col in categorical_columns:
    dataset[col] = dataset[col].fillna(dataset[col].mode()[0])

print("\nMissing values after handling:")
print(dataset.isnull().sum())

#One-hot encoding for the Location column.
dataset = pd.get_dummies(dataset, columns=["Location"], drop_first=True)

#Split Features and Target
#Split the dataset into features (X) and target variable (y)
X = dataset.drop(columns=["Estimated_Revenue"])  #Features
y = dataset["Estimated_Revenue"]  #Target

X_original = X.copy() #Maybe it will be needed to compare with original ones later, so I copied X
```

```
#Normalize the features using Min-Max Scaling (scales features to a range of [0, 1])
numeric_columns = X.select_dtypes(include=["float64", "int64"]).columns
categorical_columns = X.select_dtypes(include=["uint8"]).columns

#Apply Min-Max Scaling only to numerical columns because one-hot encoded feature (location) is already normalized automatically and defaultly!
scaler = MinMaxScaler()
X_normalized_numeric = pd.DataFrame(scaler.fit_transform(X[numeric_columns]), columns=numeric_columns)

#Combine normalized numerical columns and untouched categorical columns
X_normalized = pd.concat([X_normalized_numeric, X[categorical_columns].reset_index(drop=True)], axis=1)

#Display the first few rows of the normalized dataset
print("\nFirst 10 rows of the normalized dataset:")
print(X_normalized.head(10))

#THIS IS JUST TO SEE THE CORREALTIONS. WITH THIS, I WILL MAKE A DECISON WHICH FEATURES CAN BE COMBNED OR REMOVED.
#BUT THIS CORELLATION ANALYSIS CODE LINES JUST BELOW DID NOT WORK LIKE I WANT, SO I COMMENTED ALL OF THEM AND MADE PERSONAL AND MANUEL DECISIONS ABOUT WICH FEATURES CAN BE COMBNED OR REMOVED.
#Coreallation analysis and remove the low corellations
#correlation_matrix = X_normalized.corr()
#target_correlation = correlation_matrix.iloc[:, -1]  #Target is the last column!
#ow_corr_features = target_correlation[abs(target_correlation) < 0.1].index
#rint("Column with low-corelations with the target variable:")
#print(low_corr_features)
```

```
#Remove the low-corealitons
#X_normalized = X_normalized.drop(columns=low_corr_features)
#print("\nRemoved low-corelations. Updated normalized data-set:")
#print(X_normalized.head(10))


#Outlier Analysis and Removal by IQR Method
def remove_outliers_iqr(df, columns):

    for col in columns:
        Q1 = df[col].quantile(0.25)  #First quartile
        Q3 = df[col].quantile(0.75)  #Third quartile
        IQR = Q3 - Q1                #Interquartile Range

        #Define outlier boundaries
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        #Filter the data to remove outliers
        df = df[(df[col] >= lower_bound) & (df[col] <= upper_bound)]
    return df

#Identify numeric columns for outlier analysis
numeric_columns = X.select_dtypes(include=["float64", "int64"]).columns

#Remove outliers
print(f"\nDataset shape before removing outliers: {dataset.shape}")
dataset = remove_outliers_iqr(dataset, numeric_columns)
```

```
print(f"Dataset shape after removing outliers: {dataset.shape}")

#Display the first 10 rows of the cleaned from outliers dataset
print("\nFirst 10 rows of the dataset after removing outliers:")
print(dataset.head(10))


#Split the normalized data into training (70%) and testing (30%) sets
X_train, X_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.3, random_state=42)

#Display the shape of the training and testing sets
print(f"\nTraining Set Shape: {X_train.shape}")
print(f"Testing Set Shape: {X_test.shape}")
```

#Step 3: Implementing Linear Regression !!! !!! !!!

```
#Initialize and train the Linear Regression model
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)

#Make predictions on the test set
y_pred_lin = lin_reg.predict(X_test)

#Evaluate the Linear Regression model
mae_lin = mean_absolute_error(y_test, y_pred_lin)
mse_lin = mean_squared_error(y_test, y_pred_lin)
r2_lin = r2_score(y_test, y_pred_lin)
```

```
#Display performance metrics
print("\nLinear Regression Performance:")
print(f"Mean Absolute Error (MAE): {mae_lin:.2f}")
print(f"Mean Squared Error (MSE): {mse_lin:.2f}")
print(f"Root Mean Squared Error (RMSE): {np.sqrt(mse_lin):.2f}")
print(f"R² Score: {r2_lin:.2f}")
```

#Step 4: Implementing Random Forest for Regression !!! !!! !!!

```
#Initialize and train the Random Forest Regressor
rf_reg = RandomForestRegressor(n_estimators=100, random_state=42)
rf_reg.fit(X_train, y_train)

#Make predictions on the test set
y_pred_rf = rf_reg.predict(X_test)

#Evaluate the Random Forest Regressor model
mae_rf = mean_absolute_error(y_test, y_pred_rf)
mse_rf = mean_squared_error(y_test, y_pred_rf)
r2_rf = r2_score(y_test, y_pred_rf)

#Display performance metrics
print("\nRandom Forest Regressor Performance:")
print(f"Mean Absolute Error (MAE): {mae_rf:.2f}")
print(f"Mean Squared Error (MSE): {mse_rf:.2f}")
print(f"Root Mean Squared Error (RMSE): {np.sqrt(mse_rf):.2f}")
print(f"R² Score: {r2_rf:.2f}")
```

#Step 5: Compare and Interpret Results !!! !!! !!!

#Compare these two models. The comparison helps identify the most suitable model for this task.

```
print("\nModel Comparison:")
print(f"Linear Regression R² Score: {r2_lin:.2f}")
print(f"Random Forest R² Score: {r2_rf:.2f}")

print("\n**********************************************************************
**********************************\n")

results = pd.DataFrame({
    "Model": ["Linear Regression", "Random Forest Regressor"],
    "MAE": [mae_lin, mae_rf],
    "MSE": [mse_lin, mse_rf],
    "RMSE": [np.sqrt(mse_lin), np.sqrt(mse_rf)],
    "R² Score": [r2_lin, r2_rf]
})

print("\nModel Performance Comparison:\n")
print(results)
```

9. RESULT COMPARISON:

After running the code, I got the following results at the end of the code output:

```
Model Performance Comparison:

                     Model           MAE           MSE          RMSE  R² Score
0        Linear Regression  39593.075447  2.245764e+09  47389.489142  0.038811
1  Random Forest Regressor   6505.100583  1.285085e+08  11336.161094  0.944998
```

NOTE: You can find the total complete output of my code in the attached file to this report.

ANALYSIS:

1. **Mean Absolute Error (MAE):**

- Observation: The Random Forest Regressor achieved a significantly lower MAE compared to Linear Regression.
- Interpretation: A lower MAE indicates that the Random Forest model has much smaller average prediction errors. This means that it is more accurate in predicting the estimated revenue of venues.

2. **Mean Squared Error (MSE):**

- Observation: The MSE for Random Forest model is considerably smaller than that of Linear Regression.
- Interpretation: MSE emphasizes larger errors due to its squaring effect. The significantly lower MSE of the Random Forest model highlights its ability to handle complex relationships and reduce large prediction errors compared to Linear Regression.

3. **Root Mean Squared Error (RMSE):**

- Observation: The RMSE of Random Forest model is much lower than that of Linear Regression.
- Interpretation: RMSE provides an error metric in the same units as the target variable (revenue). A lower RMSE for Random Forest indicates that its predictions are more closely aligned with actual revenue values, making it the better-performing model in this regard.

4. **R^2 Score:**

- Observation: The R^2 score for Random Forest Regressor is 0.94, whereas it is only 0.04 for Linear Regression.
- Interpretation: The R^2 score reflects how well the model explains the variance in the target variable (1 means very well). The Random Forest model explains 94% of the variance in the estimated revenue, demonstrating its strong predictive power. In contrast, Linear Regression struggles significantly with only 4% variance explained, indicating poor model performance. So according to R^2 Score results, Random Forest is much more better than Linear Regression.

To conclude these metrics:

Based on the evaluation metrics, the **Random Forest Regressor significantly outperforms** Linear Regression in predicting the estimated revenue of venues. Its ability to handle non-linear relationships, and complex feature interactions makes it the superior choice for this project. **While Linear Regression serves as a good baseline model due to its simplicity and interpretability, it lacks the predictive power required for this complex dataset.**

For future implementations, Random Forest or similar ensemble models (e.g., Gradient Boosting Machines) should be prioritized for location-based revenue prediction tasks to ensure high accuracy and reliability for the similar data-sets to this dataset. **To implement Linear Regression algorithm, it will be better to choose and use a dataset with more linear relaitonships between the features and target vairables.**

10. CONCLUSION:

This project demonstrated the development and evaluation of a Branch/Location Recommendation System using two machine learning models: Linear Regression and Random Forest Regressor. The goal was to predict potential yearly revenue for venues (cafes, restaurants, etc.) based on several location-specific features such as population density, daily foot traffic, rent price, and more. The project successfully highlights the practical applications of machine learning in business decision-making and the importance of selecting the right model and dataset preprocessing techniques. Also this project highlights the importance of the dataset to the accuracy and performance of predicting.

Key Insights:

1. Performance:

Why Random Forest Performed Better?

- Captures Non-Linear Relationships: Unlike Linear Regression, which assumes linear relationships between features and the target variable, Random Forest can handle non-linear interactions effectively. Features like Competitor Count and Rent Price, which exhibit non-linear relationships with revenue, are better modeled by Random Forest.
- Handles Feature Interactions: Random Forest can capture complex feature interactions, such as how Population Density and Daily Foot Traffic collectively impact revenue.

Why Linear Regression Underperformed?

- Linear Assumption: Linear Regression assumes a linear relationship between features and the target variable. However, many features in this dataset, such as Competitor Count and Distance to Center, exhibit non-linear relationships, leading to poor performance. Also, the other features' linear-relaitonships with the target variable is linear until a saturation point. After the saturation point, linearity is directly destroyed. This also may cause the linear regression model to be underperformed.
- Feature Complexity: Categorical feature Location, may not be fully represented by the linear model's assumptions even it is converted to the numerical value.
- ***INFERENCE:*** **Addressing this could involve refining the dataset to include more linearly related features or implementing more suitable algorithms such as Ridge or Lasso Regression, which are better equipped to handle such data.**

2. Dataset's Impact on Performance

- Feature Count and Data Points: The dataset contained 8 features after preprocessing (excluding the removed University_Count) and 2000 data points. This provided sufficient diversity for training and testing the models. However, certain limitations were observed:

- Some features, like Location, required careful encoding (one-hot encoding was applied).

- Relationships between features and the target variable (Estimated_Revenue) varied in linearity (they have different linearity or non-linearity relationships), which impacted the model performances differently.

- For a complex prediction, the number of data-points (record) and the number of feature are less and some of expanding and additions can be applied. But also the size of dataset looks well for the Random Forest model to perform.

- Quality of Data: Despite its strengths, the dataset could benefit from:

  - More balanced representation of different location types (e.g., urban, suburban, rural).

  - Additional features to capture complex business dynamics, such as Competitor Quality or Marketing Spend.

3. My Observations, Interpretations, and Guesses About Overfitting and Underfitting

- Linear Regression:
  - Underfitting **may** occurred because the model assumes linear relationships, which were not consistent across all features. For example, Competitor_Count and Rent_Price showed non-linear relationships.

  - Its inability to model these complexities led to poor performance, with an R^2 score of just 0.04, indicating that it explained only 4% of the variance in the target variable.

- Random Forest:
  - Random Forest **might** avoided overfitting by leveraging its ensemble nature and effectively capturing non-linear relationships. With an R^2 score of 0.94, it demonstrated its capability to explain 94% of the variance.

  - The model performed well across the dataset without overfitting, likely due to the sufficient size of the dataset.

Recommendations for Improvement:

1. Data Augmentation:
    - Add features such as Competitor Quality, Marketing Spend, or Accessibility Index to capture the full context of revenue drivers.
    - Include time-based data (e.g., monthly revenue trends) to account for seasonality.
2. Advanced Feature Engineering:
    - Create interaction terms (e.g., Population_Density x Daily_Foot_Traffic) to better represent joint effects.
    - Explore polynomial features to capture non-linear trends explicitly.
3. Balanced Data:
    - Ensure equal representation of urban, suburban, and rural areas to improve model generalization.
4. Linear Relationships:
    - If Linear Regression model will be continued to be implemented, then the dataset should have more linearty between the features and the target varaible.

So to highlight some interference,

1. Linear Regression Performance: The Linear Regression model underperformed (R^2 = 0.04), indicating limited linear relationships in the dataset. Future work could refine (morel-linear) features or use Ridge/Lasso Regression for better results. **According to the output results, Linear Regression is not a very correct choice for this task.**
2. Dataset Diversity: Expanding the dataset with data from diverse cities, industries, and regions could **improve generalization** and real-world applicability.
3. Feature Expansion: **Incorporating new features** like "Competitor Quality" or "Marketing Spend" and reconsidering removed features like "University Count" through feature engineering could enhance predictive accuracy.

CONCLUSION:

The Random Forest Regressor emerged as the superior model for this task, significantly outperforming Linear Regression. Its ability to handle non-linear relationships and complex feature interactions made it the best choice for location-based revenue prediction, for this current-task. However, further refinements to the dataset and additional feature engineering could unlock even greater accuracy and reliability.

The Linear Regression resulted in poor performance in this project, contrary to my initial expectations. I assumed that Linear Regression would perform reasonably well even with moderately linear relationships in the dataset. However, the results revealed that Linear Regression struggles when the features exhibit partial linearity with the target variable, especially when this linearity breaks down beyond a saturation point. This finding highlights an important limitation of Linear Regression: it requires strong and consistent linear relationships between the features and the target variable to perform well. In datasets where feature relationships are only linear within certain ranges or exhibit non-linear behavior overall, Linear Regression fails to capture the complexity, resulting in suboptimal performance. This was a key factor contributing to its low R^2 score and overall poor accuracy in this project.

This study also underscores the importance of understanding the nature of feature relationships in the dataset before selecting a model. It also reinforces the value of advanced algorithms like Random Forest, which can handle both linear and non-linear relationships effectively, providing significantly better results in scenarios like this.

11. MY EXPECTATIONS VS WHAT I FACED?

At the start of this project, I expected Linear Regression to perform reasonably well, given its effectiveness in datasets with linear relationships. I believed that the features in the dataset I used, such as Population Density, Daily Foot Traffic, and Average Income, exhibited at least low to moderate linear relationships with the target variable (Estimated Revenue). Even, I believed that since most of my features are low or mid-linear with the target variable, at least until a saturation point, Linear regression would perform well. However, the results were surprising—Linear Regression significantly underperformed, achieving an R^2 score of just 0.04. Upon reflection, this outcome can be attributed to several factors:

1. Non-linear Relationships: Although some features have linear relationships with revenue, these relationships weaken or disappear beyond certain thresholds (e.g., Competitor Count or Daily Foot Traffic reaching saturation points). Linear Regression struggles in such scenarios, which likely led to its poor performance.

2. Feature Complexity: Features like Location and Competitor Count introduce non-linear and categorical complexities that Linear Regression cannot effectively handle without advanced feature engineering.

3. Saturation Points: Even there are linear relationships, their saturation point is very ciritcal because after a saturation point, the linearity can be lost totally and this affects the linear regression to perform poorly. **I was not expecting this, but this happened and I've learned that even there are linear relationships until a saturation point, the saturation points are very critical for Linear Regression. Because the linearty is lost after that point and this directly affects the model's performance.**

For Random Forest Regressor, my expectations were relatively high due to its known ability to capture non-linear relationships and interactions between features and also it can handle complex dataset. The model not only met but exceeded my expectations with an R^2 score of 0.94. This success can be attributed to:

1. Handling Non-linearity: Random Forest excels at capturing non-linear patterns, which are prevalent in this dataset. Features like Competitor Count and Rent Price, which exhibit complex, non-linear relationships with revenue, were effectively modeled.

2. Robustness to Feature Interactions: The algorithm could account for interactions between features, such as how Population Density and Daily Foot Traffic jointly influence revenue.

3. Reduced Overfitting: Despite its complexity, Random Forest remained robust and avoided overfitting due to its ensemble nature, even with the relatively limited size of the dataset.

So,

I initially underestimated the complexity and the linearity of the dataset and overestimated the ability of Linear Regression to capture nuanced relationships. While Random Forest delivered excellent results, the gap between the two models highlights the importance of understanding feature relationships, choosing algorithms accordingly, and preparing the dataset very well. This experience reinforced the value

of model experimentation and the need for iterative dataset refinement to achieve optimal results in machine learning projects.

12. FUTURE WORK:

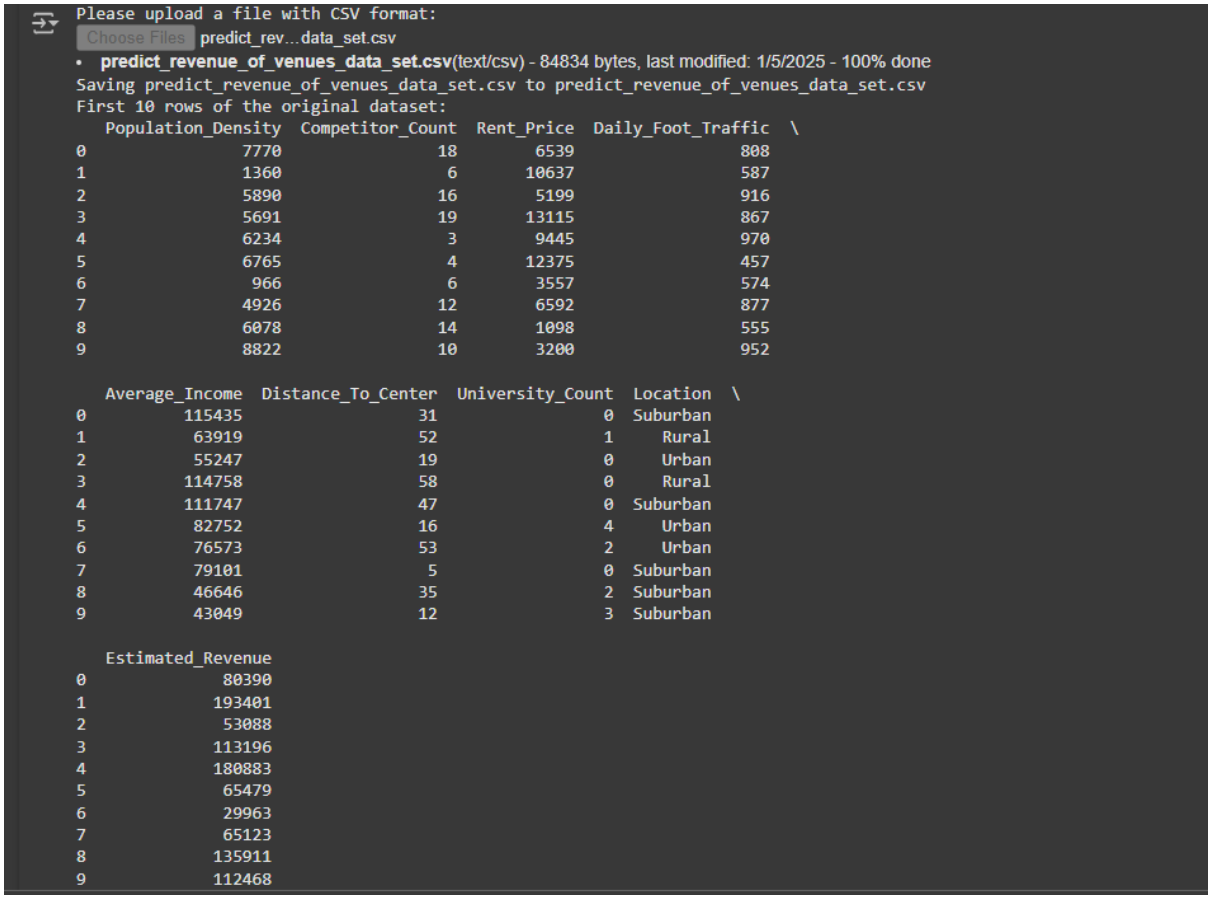
For future work, several enhancements can be considered to improve the performance and applicability of the project:

1. Feature Engineering: Introduce more advanced features, such as competitor quality, marketing expenditure, or customer demographics, to better capture real-world business dynamics.
2. Advanced Models: Explore other ensemble methods like Gradient Boosting Machines or XGBoost to achieve even higher accuracy and robustness.
3. Dataset Expansion: Increase the dataset size by incorporating data from more diverse locations and industries to improve generalizability and reduce bias.
4. Time-Series Analysis: Integrate temporal data to analyze seasonal trends and predict revenue fluctuations over time.
5. Visualization Tools: Develop an interactive dashboard to visualize model predictions and provide actionable insights for business stakeholders.

These steps would enhance both the predictive accuracy and practical usability of the system for real-world decision-making.

ATTACHMENT:

You can find the complete total output of my code in this attachment:



```
Please upload a file with CSV format:
Choose Files  predict_rev...data_set.csv
• predict_revenue_of_venues_data_set.csv(text/csv) - 84834 bytes, last modified: 1/5/2025 - 100% done
Saving predict_revenue_of_venues_data_set.csv to predict_revenue_of_venues_data_set.csv
First 10 rows of the original dataset:
   Population_Density  Competitor_Count  Rent_Price  Daily_Foot_Traffic  \
0                7770                18        6539                 808
1                1360                 6       10637                 587
2                5890                16        5199                 916
3                5691                19       13115                 867
4                6234                 3        9445                 970
5                6765                 4       12375                 457
6                 966                 6        3557                 574
7                4926                12        6592                 877
8                6078                14        1098                 555
9                8822                10        3200                 952

   Average_Income  Distance_To_Center  University_Count  Location  \
0          115435                  31                 0  Suburban
1           63919                  52                 1     Rural
2           55247                  19                 0     Urban
3          114758                  58                 0     Rural
4          111747                  47                 0  Suburban
5           82752                  16                 4     Urban
6           76573                  53                 2     Urban
7           79101                   5                 0  Suburban
8           46646                  35                 2  Suburban
9           43049                  12                 3  Suburban

   Estimated_Revenue
0             80390
1            193401
2             53088
3            113196
4            180883
5             65479
6             29963
7             65123
8            135911
9            112468
```

```
General Info about the dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 9 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Population_Density  2000 non-null   int64
 1   Competitor_Count    2000 non-null   int64
 2   Rent_Price          2000 non-null   int64
 3   Daily_Foot_Traffic  2000 non-null   int64
 4   Average_Income      2000 non-null   int64
 5   Distance_To_Center  2000 non-null   int64
 6   University_Count    2000 non-null   int64
 7   Location            2000 non-null   object
 8   Estimated_Revenue   2000 non-null   int64
dtypes: int64(8), object(1)
memory usage: 140.8+ KB
None
Checking for missing values:
Population_Density    0
Competitor_Count      0
Rent_Price            0
Daily_Foot_Traffic    0
Average_Income        0
Distance_To_Center    0
Location              0
Estimated_Revenue     0
dtype: int64

Missing values after handling:
Population_Density    0
Competitor_Count      0
Rent_Price            0
Daily_Foot_Traffic    0
Average_Income        0
Distance_To_Center    0
Location              0
Estimated_Revenue     0
dtype: int64
```

```
First 10 rows of the normalized dataset:
   Population_Density  Competitor_Count  Rent_Price  Daily_Foot_Traffic  \
0            0.740632          0.857143    0.394285            0.786561
1            0.115144          0.285714    0.681280            0.568182
2            0.557182          0.761905    0.300441            0.893281
3            0.537763          0.904762    0.854822            0.844862
4            0.590749          0.142857    0.597801            0.946640
5            0.642564          0.190476    0.802997            0.439723
6            0.076698          0.285714    0.185447            0.555336
7            0.463115          0.571429    0.397997            0.854743
8            0.575527          0.666667    0.013236            0.536561
9            0.843286          0.476190    0.160445            0.928854

   Average_Income  Distance_To_Center
0        0.929755            0.492063
1        0.446676            0.825397
2        0.365357            0.301587
3        0.923407            0.920635
4        0.895172            0.746032
5        0.623278            0.253968
6        0.565336            0.841270
7        0.589042            0.079365
8        0.284703            0.555556
9        0.250973            0.190476

Dataset shape before removing outliers: (2000, 9)
Dataset shape after removing outliers: (2000, 9)

First 10 rows of the dataset after removing outliers:
   Population_Density  Competitor_Count  Rent_Price  Daily_Foot_Traffic  \
0                7770                18        6539                 808
1                1360                 6       10637                 587
2                5890                16        5199                 916
3                5691                19       13115                 867
4                6234                 3        9445                 970
5                6765                 4       12375                 457
6                 966                 6        3557                 574
7                4926                12        6592                 877
8                6078                14        1098                 555
9                8822                10        3200                 952
```

	Average_Income	Distance_To_Center	Estimated_Revenue	Location_Suburban	Location_Urban
0	115435	31	80390	True	False
1	63919	52	193401	False	False
2	55247	19	53088	False	True
3	114758	58	113196	False	False
4	111747	47	180883	True	False
5	82752	16	65479	False	True
6	76573	53	29963	False	True
7	79101	5	65123	True	False
8	46646	35	135911	True	False
9	43049	12	112468	True	False

```
Training Set Shape: (1400, 6)
Testing Set Shape: (600, 6)

Linear Regression Performance:
Mean Absolute Error (MAE): 39593.08
Mean Squared Error (MSE): 2245763681.13
Root Mean Squared Error (RMSE): 47389.49
R² Score: 0.04

Random Forest Regressor Performance:
Mean Absolute Error (MAE): 6505.10
Mean Squared Error (MSE): 128508548.34
Root Mean Squared Error (RMSE): 11336.16
R² Score: 0.94

Model Comparison:
Linear Regression R² Score: 0.04
Random Forest R² Score: 0.94

**********************************************************************************************


Model Performance Comparison:

                   Model           MAE           MSE          RMSE  R² Score
0      Linear Regression  39593.075447  2.245764e+09  47389.489142  0.038811
1  Random Forest Regressor   6505.100583  1.285085e+08  11336.161094  0.944998
```