

MDC Prediction Model (MDC P.M.) Documentation



Development Supervisors

JoAnna Kelly, Rosario Fiallos, Alex Garcia

Intern Development Team

Daniel Jalali, Taha Muhammad, Valerie Bénédict, Alejandro Morales, Kenechukwu Nneji

Date of Creation

05/14/2024

Project Goals

Problem

The public data on MDC's website about Corrections reports (<https://www.miamidade.gov/global/corrections/home.page>) is currently static or semi-dynamic.

The report that is posted there is for public consumption and on purpose static or semi-dynamic. It is not intended to provide the ability to anticipate trends.

The Data Science and Business Intelligence group at Miami-Dade County, currently comprises working with 46 departments, including Aviation, Solid Waste, Police, Corrections and Rehabilitation, Port of Miami, Transit, and others. Each department utilizes one or more systems to manage its operations and business processes. Additionally, there are several Enterprise systems, such as Finance and Human Resources, that are used by all departments.

Our primary responsibility involves extracting data from operational and transactional systems, transforming the data, and providing analytics, dashboards, and reports to facilitate data-driven educated decision-making for the various departments.

In addition to traditional data analysis, we leverage Generative AI technology based on ChatGPT. This allows us to input data like departmental procedures into the model, enabling the business to pose inquiries related to the documents and receive informative responses.

Furthermore, our team has delved into Machine Learning, conducting a Proof of Concept (POC) project aimed at identifying instances of Corrections and Rehabilitation Inmate Violence. This initiative demonstrates our commitment to utilizing advanced technologies to address specific challenges within the county's departments.

However, the Corrections and Rehabilitation Department did not have a way to predict the future rate of inmate violence and the rate of inmates coming into the system.

Solution:

It is the goal, purpose, or objective of the Data Science and Business Intelligence group to develop analytics, dashboards, and reports to facilitate data-driven decision-making.

Our approach involves several key steps:

- **Data Collection and Cleaning:** Aggregating and preprocessing the available data to ensure accuracy and consistency.
- **Exploratory Data Analysis (EDA):** Analyzing the data to identify patterns, correlations, and anomalies.
- **Model Development:** Using machine learning techniques to create predictive models that forecast future trends.
- **Validation and Testing:** Evaluating the models to ensure they are reliable and accurate.
- **Implementation:** Integrating the prediction model into the existing system to provide real-time forecasts.
- **Visualization:** Develop user-friendly dashboards to display predictions and insights for easy interpretation by stakeholders.

By implementing this Prediction Model System, we aim to help Corrections management anticipate and prevent jail population overcrowding by predicting the number of inmates that will be in jail.

Commented [1]: Add the other points of interest that you are predicting

References & Links

Overall Files (Sharepoint):

[Sprinterns 2024 - Sprinterns Data Analytics - All Documents \(sharepoint.com\)](#)

Presentation (Powerpoint):

[Prediction Model Presentation.pptx \(sharepoint.com\)](#)

Task Management (Planner):

[Data Analytics Team - Model Prediction System - Planner \(office.com\)](#)

Source Control (Azure DevOps):

[Projects - Home \(azure.com\)](#)

Guide on Git with Miami Dade:

[ITD-Agile - Miami-Dade County - All Documents \(sharepoint.com\)](#)

IDE (Pycharm):

<https://www.jetbrains.com/pycharm/download/?section=windows#section=windows>

Flowchart Creator (Visio):

[Flow Chart Design.vsd \(sharepoint.com\)](#)

PDF Files for Analysis:

\\ecsf3\ITD3\ITDDData\Application Development\Corrections and Rehabilitation\Inmate Data Warehouse\Cognos Data\IDW Daily Statistics Reports

Power BI Group Files:

[Power BI \(powerbigov.us\)](#)

Management & Collaboration Outline

Meetings & Communication

Stand-up meetings with our mentor, Rosario, at 9:15 am

Additional meetings will be performed through Teams at times after daily 9:15 AM meetings

Source Control

Source control will be done through Azure DevOps

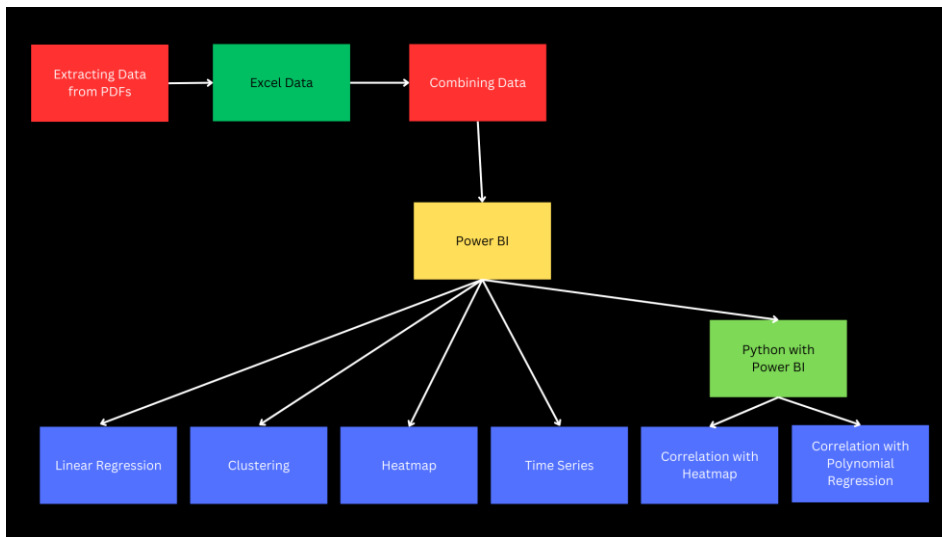
Task Delegation & Management

Task delegation will be done through Microsoft Planner

- The whole team worked on extracting the data
- The whole team worked on putting the data together
- Alejandro Morales worked on the time series predicting model
- Muhammed Taha worked on a predictive heatmap model
- Kenechukwu Nneji worked on a predictive model using polynomial regression and correlation
- Valerie Benedit worked on clustering and linear regression predictive models
- Daniel Jalali worked on a correlation heatmap predictive model

Product Outline

Structure Flowchart



Text-Based Showcase

- Python > Excel Data > Prediction via Python > PowerBi

Current Tech Stack

Programming Language:

- Python

Programming Modules:

- Tabula
- Matplotlib
- Seaborn
- Pandas

Package Management:

- Pip3

Recommended IDE:

- Pycharm

Source Control

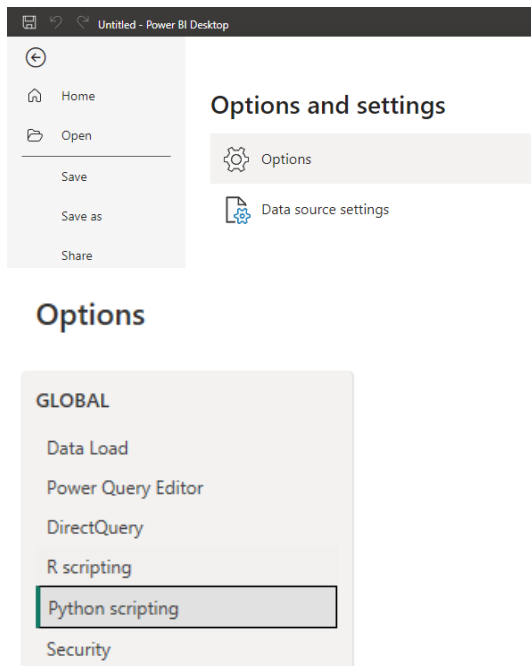
- Azure Devops (Git)

Guide on how to Install Python with PowerBI



1. Install Python on Your PC

- Visit the official Python website (<https://www.python.org/downloads/>) and download the latest version compatible with your operating system.
- Follow the installation instructions for your specific Windows, macOS, or Linux system.



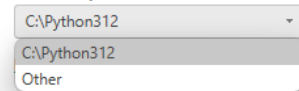
2. Power BI Detection

- In Power BI Desktop, go to **File > Options and settings > Options**.
- Select **Python scripting**.

Python script options

To choose a home directory for Python, select a detected Python installation from the drop-down list, or select Other and browse to the location you want.

Detected Python home directories:



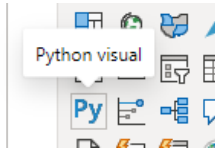
C:\Python312

C:\Python312

Other

3. Automatic & Manual Detection

- If Python is installed and your system path is set correctly, Power BI should detect it.
- If not detected, click **other** and navigate to your Python installation directory (e.g., [C:\Python310](#)). Click **OK** to save the settings.



4. Create a Python Visual

- In your Power BI report, switch to the visualization pane.
- Locate the **Python** visual icon, which resembles a **PY** symbol.
- Drag and drop the Python visual onto your report canvas.

Enable script visuals



You need to enable script visuals to begin creating Python script. Script visuals can execute script code that may contain security or privacy risks.

Enable

Cancel

5. Enable Script Visuals

- Before adding Python code to the visual, ensure script visuals are enabled:
 - You will see a popup to "Enable script visuals".
 - Click Enable to enable python code to be integrated with the visual.

6. Write Python Code for the Visual (using pandas)

- Once the Python visual is added, navigate to the bottom panel of Power BI. The Python script editor will appear there.
- **Import pandas:** You'll likely need the `pandas` library for data manipulation within your Python code.
- **Access Data from Power BI:** Write code to access your imported data from Power BI. The specific variable name might differ based on your data import process.

During our project, we faced a variety of challenges in creating our prediction model, most of which are outlined below:

1. Data Gathering With Python & Excel

Our initial attempt to gather data involved extracting information from PDFs of reports from the last few years and importing it into Excel. However, we encountered several challenges:

We found that the graphs within the PDFs were not displayed correctly, and in some cases, they were not displayed at all. This disruption affected the structure and integrity of the data. Additionally, the import process often resulted in disorganized data, requiring extensive manual correction and reformatting to ensure accuracy.

These issues highlighted the need for a more robust method of data extraction and processing to maintain data quality and reliability.

[illegible]

Possible Solution:

1. Use the current data we are able to obtain from the PDFs to make predictions
2. (If Approved) Obtain a separate server under MDC that is isolated with 1-2 months of data from SQL

2. Figuring Out Source Control

We initially planned to use Git for source control with a private repository on GitHub since we were all familiar with it. However, due to organizational restrictions requiring us to use internal tools, we had to switch to Azure DevOps with Git for managing our source control. This transition required us to learn a new system and adapt our workflow accordingly.

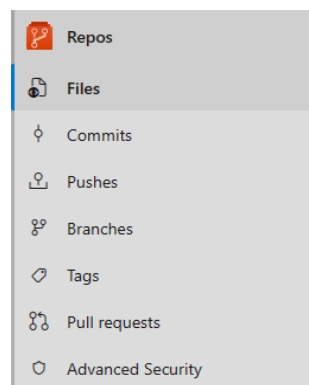


Possible Solution:

1. Conduct training sessions on Azure DevOps to familiarize the team with the new tool.
2. Establish a clear workflow for using Git within Azure DevOps, including guidelines for committing and merging code to avoid conflicts and ensure smooth collaboration.

3. Using and Learning DevOps

Adopting DevOps practices was a new challenge for many team members. We needed to understand its principles and tools to streamline our development process. This required additional training and the setup of a system where pull requests must be reviewed and accepted by other team members to maintain code quality and project stability.



Possible Solution:

1. Schedule regular training sessions to cover DevOps fundamentals and best practices.
2. Implement a code review process to ensure that all pull requests are thoroughly checked before being merged.

4. Installing PyCharm and Packages for Other Users

To maintain consistency in our development environment, we decided to use PyCharm as our Integrated Development Environment (IDE). An IDE is a software application that combines common developer tools like code editing, debugging, and project management into a single interface. This simplifies the development process and helps developers work more efficiently.

However, not all team members had experience with PyCharm, and some faced issues with setting up Python and its package manager, pip3. We had to provide support and troubleshooting to ensure everyone could work efficiently.

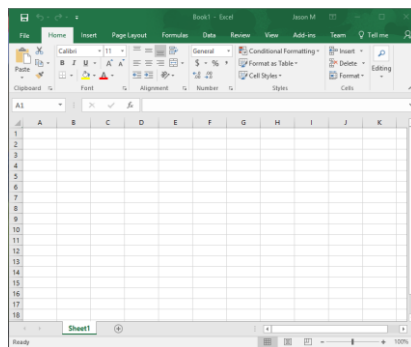


Possible Solution:

1. Create a detailed installation guide for PyCharm and necessary packages.
2. Offer one-on-one support sessions to assist team members with setup and troubleshooting.

5. Spreading Data Analysis Between Team Members

Analyzing the large volume of PDF files required distributing the workload among team members. We designated specific months and years to each person to streamline the process. This approach ensured that everyone had a manageable portion of data to analyze, leading to more efficient and thorough data processing.



Possible Solution:

- Develop a clear plan for dividing the data analysis tasks among team members.
- Use collaborative tools to track progress and ensure consistency in data analysis.

6. Switching Focus on Data Collection Due to Security Risks

After initial analysis, we found that the data collected from PDFs was insufficient for accurate predictions. Additionally, using sensitive data posed security risks. To address these issues, we decided to switch to using public data from other state government police department sites and reports from the UK, which provided more comprehensive and secure datasets.



Possible Solution:

- Identify and compile a list of reliable public data sources.
- Develop a protocol for securely handling and processing public data.

7. Switching Data Collection to Public Florida County Data

The data we initially obtained from PDFs proved insufficient and presented security concerns. We then explored public data from other state government police department sites. While the UK data offered a more comprehensive dataset on offenses by different police forces, it required significant adaptation for the MDC context.

To address this challenge, we've shifted our focus to collecting public data from Florida County Detention Facilities (specifically the one on average inmate population). This data will likely include details on crime rates, offense types, and demographics, offering a closer resemblance to the data relevant to MDC.



Possible Solution:

- Identify and compile a list of reliable public data sources from Florida counties.
- Develop a protocol for securely handling and processing this public data.
- Refine the prediction model to utilize the Florida county data effectively, focusing on relevant metrics like crime types, demographics, and arrest rates.

Sample Tables & Data

In this section, we'll delve into the heart of our data analysis by showcasing the tables we constructed from the extracted data.

Each table is specifically designed to shed light on factors that might influence future inmate populations within the Miami-Dade system.

We'll provide insights into the logic and formulas employed to create these tables, along with relevant code snippets (where applicable) to illustrate the data manipulation process.

Time Series

Multiplicative

Level : $L_t = \alpha [Y_t / S_{t-1}] + (1 - \alpha) [L_{t-1} + S_{t-1}]$

Where

Y_t = Time series

L_t = Level (Intercept)

T_t = Slope / Trend

S_t = Seasonality

H = Period

Trend: $T_t = \beta [L_t - L_{t-1}] + (1 - \beta) T_{t-1}$

Seasonality : $S_t = \gamma [Y_t] / [L_{t-1} + T_{t-1}] + (1 - \gamma) S_{t-1}$

Forecast: $F_{t+h} = [L_t + T_t * h] * S_{t(\text{respective period})}$

A time series is a collection of data points measured at specific points in time. The order of the data points matters because it shows how the value changes over time. This is important for understanding trends and making predictions..

How to Setup The Chart

- You would need to use a line chart from Power BI for this model. This chart includes a forecasting function that allows the model to predict future values based on plotted data.
- With the line chart selected, plot all the available data points of the variable that is to be explored.
- On the line chart analysis tool, the forecasting should be turned on, and the setting adjusted based on the data that is to be predicted. Some of the settings that can be adjusted, are the length of the prediction, the seasonality, and the confidence interval.

Linear and Polynomial Regression

Simple
Linear
Regression

$$y = b_0 + b_1x_1$$

Multiple
Linear
Regression

$$y = b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

Polynomial
Linear
Regression

$$y = b_0 + b_1x_1 + b_2x_1^2 + \dots + b_nx_1^n$$

- In regards to polynomial regression, one must analyze how the points are distributed to determine up to which power best fits the data. Referencing typical polynomial graphs would be helpful.

How to Setup The Chart

One may use the DAX functionality or python to implement either Linear Regression or Polynomial Regression.

DAX:

*Note: This code was used specifically with the data given to us. Adapt as needed.

Step 1:

Create a separate LinestX table with the variables (the column names) you would like to analyze

Step 2:

Code the formula into a DAX expression

```
Pretrial Linear Regression =  
VAR _linest = LINESTX(Sheet2, [Total PercentPretrial], [Date].[Year])  
VAR _slope = SELECTCOLUMNS(_linest, [slope1])  
VAR intercept = SELECTCOLUMNS(_linest, [Intercept])  
RETURN  
intercept + _slope * 'Yearly prediction'[Yearly prediction Value]
```

Clustering

Clustering in Power BI allows you to uncover hidden patterns and relationships within your data by grouping similar data points together.

How to Setup The Chart

Create a scatter plot:

- In the "Visualizations" pane on the right, find the "Scatter" chart icon.
- Drag and drop the two numerical columns you want to analyze onto the "X" and "Y" axes of the scatter plot, respectively.
- You should now see a scatter plot with your data points.

Access Clustering options: (This might vary slightly depending on your Power BI version)

- Right-click on the scatter plot itself.
- Look for an option named "**Find Clusters**" or similar wording.

Analyze the results:

- Power BI will add a new column to your data table representing the cluster assignment for each data point.
- You can visualize the clusters by assigning different colors to each cluster in the formatting options of your scatter plot. This will help you see how the data points are grouped based on the clustering analysis.
- You can use the Azure Machine Learning feature with clustering to provide future predictions within the clustering groups created.

Correlation

Correlation analysis in Power BI is vital because it reveals how variables move together. It helps you understand if changes in one variable are linked to changes in another. This can be crucial for predicting trends, segmenting your data, and uncovering hidden relationships within your dataset.

How to Setup The Chart

Earlier in this document, we discussed how to create visuals using Python in PowerBI. You can create correlations via python, which are particularly useful for visually representing correlations between variables in your data set.

Relevant Python Code:

Note: This code was specifically used with the data given to us. One must replace "correlation_matrix_males" with the relevant table or matrix for the heatmap you would like to create.

Python Code with PowerBI

```
# Create heatmaps
plt.figure(figsize=(16, 6), facecolor='lightgrey')

# Heatmap for males
plt.subplot(1, 2, 1)
sns.heatmap(correlation_matrix_males, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap for Males')
plt.tight_layout()
plt.show()
```


Heatmap

Heatmaps are like colorful thermometers for correlations. They use color intensity to reveal strong positive (red) or negative (blue) relationships between variables in your data. This makes it much easier to spot patterns and trends compared to raw numbers in a table. They're a great way to quickly grasp the big picture of how things are connected in your data.

How to Setup The Chart

Earlier in this document, we discussed how to create visuals using Python in PowerBI. You can create heat maps via python, which are particularly useful for visually representing correlations between variables in your data set.

Relevant Python Code:

Note: This code was specifically used with the data given to us. One must replace "correlation_matrix_males" with the relevant table or matrix for the heatmap you would like to create.

Python Code with PowerBI

```
# Create heatmaps
plt.figure(figsize=(16, 6), facecolor='lightgrey')

# Heatmap for males
plt.subplot(1, 2, 1)
sns.heatmap(correlation_matrix_males, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap for Males')
plt.tight_layout()
plt.show()
```

Correlation with Heatmap

Heatmaps are like colorful thermometers for correlations. They use color intensity to reveal strong positive (red) or negative (blue) relationships between variables in your data. This makes it much easier to spot patterns and trends compared to raw numbers in a table. They're a great way to quickly grasp the big picture of how things are connected in your data.

How to Setup The Chart

Earlier in this document, we discussed how to create visuals using Python in PowerBI. You can create heat maps via python, which are particularly useful for visually representing correlations between variables in your data set.

Python Code with PowerBI

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Assuming Power BI provides the dataset as 'dataset'
df = dataset

# Select relevant columns for males and females
male_columns = df[['Adult Males', 'Adult Males Felonies Sentenced (365 days, more)', 'Adult Males Felonies Sentenced (364 days, less)']]
female_columns = df[['Adult Females', 'Adult Females Felonies Sentenced (365 days, more)', 'Adult Females Felonies Sentenced (364 days, less)']]

# Drop duplicated rows
male_columns = male_columns.drop_duplicates()
female_columns = female_columns.drop_duplicates()

# Calculate the correlation matrices
correlation_matrix_males = male_columns.corr()
correlation_matrix_females = female_columns.corr()

# Create heatmaps
plt.figure(figsize=(16, 6), facecolor='lightgrey')

# Heatmap for males
plt.subplot(1, 2, 1)
sns.heatmap(correlation_matrix_males, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap for Males')

# Heatmap for females
plt.subplot(1, 2, 2)
```

```
#sns.heatmap(correlation_matrix_females, annot=True, cmap='coolwarm', linewidths=0.5, square=True,  
cbar_kws={'shrink': .5}, fmt=".2f", vmin=-1, vmax=1, annot_kws={"color": "white"})  
sns.heatmap(correlation_matrix_females, annot=True, cmap='coolwarm', linewidths=0.5)  
plt.title('Correlation Heatmap for Females')  
  
plt.tight_layout()  
plt.show()
```