WGU C964

Task 2

COMPUTER SCIENCE CAPSTONE

Ethan Harris

Student ID # 009582334

12/17/2023

Table of Contents

Letter of Transmittal

Tsunekazu Ishihara, CEO
Pokemon Tournament Training Club

Dear Mr. Ishihara,

I am delighted to present a comprehensive proposal for implementing our groundbreaking data product, Pokemon CHAMPS, aimed at addressing key challenges within the Pokemon Tournament Training Club. This proposal highlights the numerous benefits this product offers to our customers and the organization.

Trainers currently face difficulties optimizing deck composition and strategy for tournaments due to the lack of a systematic approach. Pokemon CHAMPS resolves this by providing predictive insights into deck success potential, enabling informed decision-making through prescriptive recommendations for strategic card additions.

At the core of Pokemon CHAMPS is the sophisticated machine learning model, C.H.A.M.P.S., analyzing a 12-year dataset of historical tournament data. Trainers can easily input card details into the user-friendly app interface, generating success predictions. Our project methodology includes meticulous data preprocessing, model training, and testing, with continuous learning ensured through regular updates based on new tournament data. We estimate an annual funding requirement of $10,000 for development, hosting, and ongoing updates.

Pokemon CHAMPS fosters community collaboration within the Pokemon Tournament Training Club. Ethical considerations are prioritized, with strict privacy measures and legal compliance in place. Our experienced team, versed in machine learning, data analysis, and application development, ensures the successful implementation of Pokemon CHAMPS. Our previous work includes a machine learning solution proposing a new algorithm for a popular trading platform in late-2023.

In conclusion, Pokemon CHAMPS aligns with our mission to empower the community and elevate performance in Pokemon tournaments. We eagerly anticipate further discussions on this proposal, exploring the significant potential Pokemon CHAMPS brings to our organization.

Thank you for your time and consideration.

A.1. Summary of Problem

Trainers face challenges in optimizing their decks for tournaments due to the absence of a systematic approach. This lack of a structured method hinders their ability to strategically compose and refine their decks, impacting their overall performance in Pokemon tournaments. The problem lies in the complexity of decision-making regarding deck composition and strategy, which can be overwhelming without a systematic and data-driven tool like Pokemon CHAMPS.

A.2. Description of Benefit

Pokemon CHAMPS empowers trainers with predictive insights, guiding informed decisions for optimal deck compositions. Utilizing a sophisticated machine learning model and a comprehensive historical dataset, the system prescribes specific card additions, enhancing trainers' success potential in tournaments.

A.3. Outline of the Data Product

Pokemon CHAMPS is a user-friendly app powered by the C.H.A.M.P.S. machine learning model. Trainers input their card details to receive success predictions based on a 12-year historical tournament dataset. The system offers prescriptive recommendations for strategic card additions, enabling trainers to optimize deck composition for increased success in Pokemon tournaments.

A.4. Description of the Data to be Used

The data used for Pokemon CHAMPS includes a comprehensive historical dataset spanning the past 12 years of Pokemon tournaments. This dataset encompasses detailed information on various card types, attributes, and the corresponding outcomes of past tournaments. It forms the foundation for the C.H.A.M.P.S. machine learning model to discern patterns and relationships between cards and tournament success.

A.5. Objectives & Hypotheses

The objective of the Pokemon CHAMPS project is to enhance decision-making for trainers in the Pokemon Tournament Training Club by utilizing the C.H.A.M.P.S. machine learning model and analyzing a 12-year historical dataset of tournament data. The goal is to predict the success potential of individual cards and provide trainers with actionable insights and prescriptive recommendations for strategic card additions. The hypothesis underlying this project asserts that the integration of Pokemon CHAMPS will significantly improve overall trainer performance in Pokemon tournaments. The predictive capabilities of the C.H.A.M.P.S. model, informed by historical tournament data, are expected to guide trainers towards more informed deck composition decisions, with prescriptive recommendations leading to strategic enhancements and increased success in upcoming tournaments.

A.6. Project Methodology

The project methodology begins with the loading and random sampling of a tournament dataset to expedite test-train split execution. Following user input for selecting a card, the dataset is augmented with card occurrence counts, and a binary column is created to signify card success based on a specified threshold. The dataset is then split into training and testing sets for the initialization and training of a RandomForestClassifier. Model evaluation includes assessing accuracy using Mean Squared Error and Root Mean Squared Error. For individual card predictions, the user inputs a card name, and the model predicts its success probability against an opponent.

A.7. Funding Requirements

The estimated annual funding requirement for the Pokemon CHAMPS project is $10,000. This funding is allocated to cover various aspects of the project, including development, hosting, and ongoing updates. These financial resources are crucial to ensure the successful implementation and continued improvement of the machine learning model, app interface, and associated infrastructure. The funding will support the project's goal of providing trainers within the Pokemon Tournament Training Club with a reliable and effective tool for optimizing their deck compositions and enhancing their success in tournaments.

A.8. Stakeholder Impact

Pokemon CHAMPS significantly benefits trainers within the Tournament Training Club by offering predictive insights and strategic recommendations. The project encourages community collaboration and aligns with the Club's mission, positively impacting both individual trainers and the overall success of the Club in Pokemon tournaments. Senior management and executives overseeing the Club may observe increased participation and improved performance as trainers engage with the CHAMPS system.

A.9. Ethical Considerations

Ethical considerations for Pokemon CHAMPS prioritize strict privacy measures and legal compliance. User consent is obtained, and regular audits detect and mitigate unauthorized access risks. Secure backups and data integrity measures are in place to prevent loss or corruption. Security protocols, including firewalls and intrusion detection systems, are consistently updated for optimal data security. The project aims to minimize risks, ensure data integrity, and foster user trust in handling large datasets.

A.10. Relevant Expertise

Our relevant expertise for the Pokemon CHAMPS project includes proficiency in machine learning, data analysis, and application development. Specifically, our team has previously worked on a machine learning solution proposing a new algorithm for a popular trading platform in late-2023. This experience demonstrates our capability to leverage machine learning techniques to solve complex problems and implement innovative solutions. Our expertise positions us well to successfully execute the Pokemon CHAMPS project and deliver impactful results for the Pokemon Tournament Training Club.

B. Executive Summary

B.1. Decision Support Problem

The Pokemon CHAMPS project addresses a critical decision support problem within the Pokemon Tournament Training Club. Trainers currently lack a systematic approach for optimizing deck compositions and strategies, hindering their performance in tournaments. Pokemon CHAMPS aims to provide predictive insights and prescriptive recommendations, leveraging a sophisticated machine learning model (C.H.A.M.P.S.) trained on a 12-year dataset of historical tournament data.

B.2. Customer Description

Pokemon CHAMPS customers are members of the Pokemon Tournament Training Club, a community focused on improving gameplay. These trainers vary in experience levels, from beginners to seasoned players, united by a common goal of enhancing their performance in tournaments. The tool caters to this diverse community, offering a sophisticated yet user-friendly solution for informed decision-making and strategic deck optimization.

B.3. Gaps in Existing Data Products

The existing gap lies in the absence of a comprehensive, data-driven tool for trainers to optimize deck compositions. Traditional approaches lack predictive capabilities and fail to provide prescriptive insights based on historical tournament data. Pokemon CHAMPS bridges this gap by introducing a sophisticated machine learning model for strategic decision-making.

B.4. Data Availability

The Pokemon CHAMPS data product requires diverse datasets for its lifecycle. Historical tournament data covers the past 12 years, detailing tournament dates, locations, participants, decks, and outcomes. A comprehensive card dataset includes names, types, attributes, and rarity levels. Success metrics, like all-time scores, define card performance.

B.5. Methodology

The methodology involves random sampling, user input augmentation, and detailed data preprocessing. Feature engineering, model training with a RandomForestClassifier, and accuracy assessment through Mean Squared Error form integral parts of the design and development process.

B.6. Deliverables

Key deliverables include a user-friendly app interface, a dynamic machine learning model (C.H.A.M.P.S.), and prescriptive recommendations for trainers. Regular updates based on new tournament data ensure the model's relevance and accuracy.

B.7. Implementation Plan

Implementation involves continuous model refinement, user engagement, and updates. Anticipated outcomes include enhanced decision-making for trainers, improved overall performance in tournaments, and increased community collaboration within the Club.

B.8. Validation Methodology

Validation and verification are achieved through accuracy assessment, including Mean Squared Error and Root Mean Squared Error. The developed data product is validated against trainer needs, ensuring alignment with predictive success probabilities.

B.9. Programming Environment

The project utilizes Python programming, with associated costs covering development, hosting, and ongoing updates estimated at $10,000 annually. Human resources include a skilled team experienced in machine learning, data analysis, and application development.

B.10. Projected Timeline

1. **Data Preprocessing:**
   - 1.1. Start/End: 12/18/23 – 12/22/23
   - 1.2. Duration: 1 week
   - 1.3. Dependencies: Completion of dataset acquisition
   - 1.4. Resources: Data scientists, preprocessing tools
2. **Random Sampling and User Input Augmentation:**
   - 2.1. Start/End: 12/26/23 – 12/29/23
   - 2.2. Duration: 1 week
   - 2.3. Dependencies: Completion of data preprocessing
   - 2.4. Resources: Data scientists, developers
3. **Model Design:**
   - 3.1. Start/End: 1/1/24 – 2/2/24
   - 3.2. Duration: 4 weeks
   - 3.3. Dependencies: Completion of random sampling and user input augmentation
   - 3.4. Resources: Data scientist, ML experts
4. **Model Training:**
   - 4.1. Start/End: 2/5/24 – 2/16/24
   - 4.2. Duration: 2 weeks
   - 4.3. Dependencies: Completion of model design
   - 4.4. Resources: Data scientist, ML experts
5. **App Development**
   - 5.1. Start/End: 2/19/24 – 3/8/24

5.2. Duration: 3 weeks

5.3. Dependencies: Completion of model training

5.4. Resources: Developers, UI/UX designers

6. **Validation & Verification**

6.1. Start/End: 3/11/24 – 3/22/24

6.2. Duration: 2 weeks

6.3. Dependencies: Completion of model training

6.4. Resources: QA team, data scientists

7. **Deployment & Launch:**

7.1. Start/End: 3/25/24 – 3/29/24

7.2. Duration: 1 week

7.3. Dependencies: Completion of validation and verification

7.4. Resources: Development team, deployment specialists

C. Data Product

C.1. Descriptive Method

The product utilizes visualizations to understand Pokemon card tournament data. Specifically, two visualizations—a pie chart showing the top 10 cards by occurrences in winning decks and a bar chart illustrating the average deck cost per country—offer insights into card popularity and cost distribution.

C.2. Nondescriptive Method

The product employs a RandomForestClassifier to predict a Pokemon card's success based on historical tournament data. Features related to the card's name are used for training and evaluation, providing a predictive measure of the card's future performance. Users can assess their cards' potential success using historical data and the model's predictions.

C.3. Collected Dataset

The dataset, sourced from the 'tournaments.csv' file, contains information pertinent to Pokemon card tournaments. It encompasses details like card names, tournament countries, card prices, and an all-time score indicating card performance. The dataset is utilized for visualizations, card sampling, and training a machine learning model to predict card success based on historical tournament data. The dataset can be found here on Kaggle: https://www.kaggle.com/datasets/enriccogemha/pokemon-tcg-all-tournaments-decks-2011-2023

C.4. Decision Support Functionality

The product employs a RandomForestClassifier to support decision-making in the Pokemon Tournament Training Club. It begins with data visualizations, showcasing the top 10 cards and average deck costs per country. To expedite processing, a random sample of the dataset is used. Users interact by selecting a card, initiating the training of the RandomForestClassifier. The model predicts the card's success against opponents, outputting a percentage likelihood. The decision is then based on a predefined threshold, simplifying the process for users in optimizing deck strategies for tournaments.

C.5. Dataset Wrangling

The product efficiently handles dataset featurization, parsing, cleaning, and wrangling. It reads a Pokemon tournament dataset, visualizes key information, addresses missing or problematic data, and streamlines processing using a random sample. The user selects a card for detailed analysis, triggering dynamic modifications to the dataset. This involves creating a 'success' binary column based on a defined 'all_time_score' threshold. OneHotEncoder encodes features, MinMaxScaler normalizes data, and the dataset is split for training and testing using train_test_split. The RandomForestClassifier is trained, and accuracy is assessed through Mean Squared Error. The code provides a systematic approach to dataset preparation for Pokemon tournament analytics.

C.6. Algorithmic Method

The product incorporates essential methods and algorithms for streamlined data exploration and preparation. It employs random sampling enhances efficiency for subsequent processes, particularly beneficial for extensive datasets, this one has over 2.9 million rows! Furthermore, it engages in feature engineering by creating a binary 'success' column based on a specified threshold (this can be changed by the user), indicating card success. The application of One-Hot Encoding, Min-Max Scaling, and a RandomForestClassifier for training and prediction underscores the utilization of machine learning. The model's evaluation through Mean Squared

Error and accuracy metrics provides valuable performance insights. Ultimately, the code delivers a robust framework for data analysis and decision support, predicting card success and offering a probability estimate for user-selected cards in tournament scenarios

C.7. Data Visualization

The product uses Matplotlib and Seaborn libraries to create attractive visualizations for data exploration. The first visualization is a Pie Chart displaying the top 10 cards by occurrences, offering a quick overview of prevalent cards. The second visualization, a Correlation Heatmap in the form of a Bar Plot, illustrates the average deck cost per country, revealing cost distribution across regions. The third visualization features a stack plot which is used to depict the number of tournaments held by each country, providing a clear global tournament distribution. These visualizations facilitate a concise exploration of the dataset, aiding in the identification of patterns and insights for further analysis.

C.8. Interactivity

The product incorporates an interactive query mechanism by prompting the user to select a card through numerical input. This allows users to dynamically choose a card of interest, enhancing the code's interactivity. The selected card is then used to query and analyze relevant information from the dataset, such as the card's success probability and predicted outcomes. The interactive queries make the code versatile and user-friendly, enabling users to explore specific scenarios and obtain tailored insights based on their input.

C.9. Machine Learning Method Implementation

The data product demonstrates a comprehensive implementation of machine-learning methodologies for predictive analysis and classification tasks. Beginning with data loading and visualization using Pandas, Matplotlib, and Seaborn, the code explores key insights through visualizations like pie charts and bar plots. It incorporates data preprocessing techniques, including numeric data cleaning and random sampling, to enhance efficiency during test-train splits. User interaction is facilitated by displaying randomly selected card names and prompting the user's card selection. Feature engineering involves one-hot encoding and min-max scaling to prepare categorical features for machine learning. The RandomForestClassifier from scikit-learn is employed for classification, with model training and evaluation metrics such as Mean Squared

Error and Root Mean Squared Error. Predictions are made for both a test set and a user-selected card, and a threshold-based approach determines the success message, providing insights into the card's potential performance. The implementation concludes by persisting the trained model using joblib for potential future use without retraining, showcasing a holistic machine-learning pipeline.

C.10. Accuracy Evaluation

The code evaluates model accuracy using Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) on the test data. It employs scikit-learn's `accuracy_score` for an overall accuracy measure. These functionalities offer a straightforward assessment of the model's accuracy and its practical implications for individual cards.

C.11. Security Features

Security could be enhanced by implementing data encryption, access controls, secure input handling, and comprehensive logging. Conducting security audits, penetration testing, and addressing model deployment security concerns are essential.

C.12. Monitoring and Maintenance

Integrate Splunk for logging, Prometheus for performance monitoring, and Docker for system availability to ensure monitoring. Employ Git for version control and seamless integration into new systems.

C.13. User-Friendliness

Using Jupyter Notebooks and matplotlib the user is able to interact with the data and visualizations provided through the code. The code allows the user to interact with the data via a prompt which allows them to select one of 10 randomly selected cards for the sake of demoing functionality of the product. This layout allows a seamless and attractive user experience.

D. Product Documentation

D.1. Business Vision

The Pokemon Card-based Helper for Assessing Match Performance System (C.H.A.M.P.S.) and the Pokemon Tournament Training Club revolved around leveraging data analytics and machine learning to gain insights into the tournament data. By analyzing the dataset of tournament results and card attributes, the project was aiming to provide valuable visualizations and predictive models. The project could potentially cater to the gaming or trading card industry, providing valuable insights for players and enthusiasts. The ultimate vision involved creating a platform that could empower a user with data-driven decision-making in the context of their next game.

D.2. Raw & Clean Data

Data found here: https://www.kaggle.com/datasets/enriccogemha/pokemon-tcg-all-tournaments-decks-2011-2023

D.3. Hypothesis Assessment

The Pokemon CHAMPS project aimed to help decision-making for trainers in the Pokemon Tournament Training Club through the application of the C.H.A.M.P.S. machine learning model. The underlying hypothesis asserts that the integration of Pokemon CHAMPS will enhance trainer performance in Pokemon tournaments. This hypothesis anticipates that the model's predictive capabilities, informed by historical tournament data, will guide trainers toward more informed deck composition decisions, and that the prescriptive recommendations will lead to strategic enhancements, ultimately resulting in increased success in upcoming tournaments. The long-term success of the project could be assessed through rigorous evaluations of the model's performance, its impact on trainer decision improvement, the practicality of provided insights, long-term effectiveness, and user feedback, collectively determining the project's alignment with its objectives and the validation of its underlying hypothesis. For the sake of the project at hand, I believe that the project hypothesis was met using this machine learning algorithm and the dataset provided.

D.4. Data Exploration

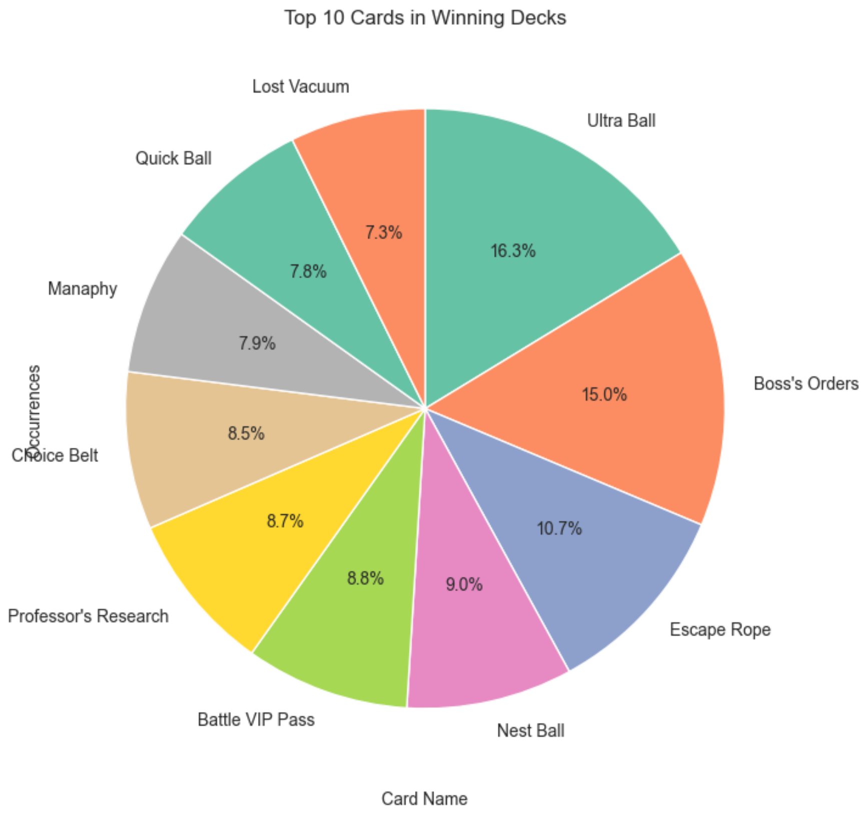


Figure D.4.1 – Top 10 Cards in Winning Decks

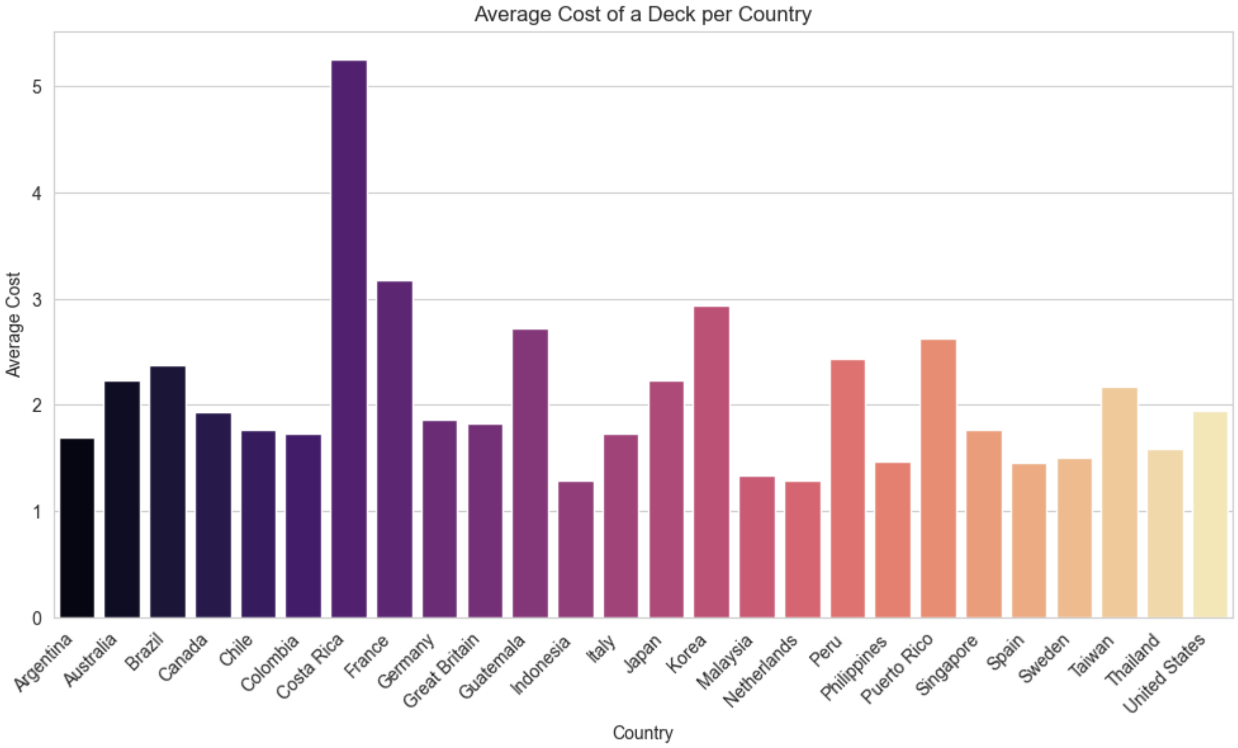


Figure D.4.2 – Average Cost of a Deck Per Country

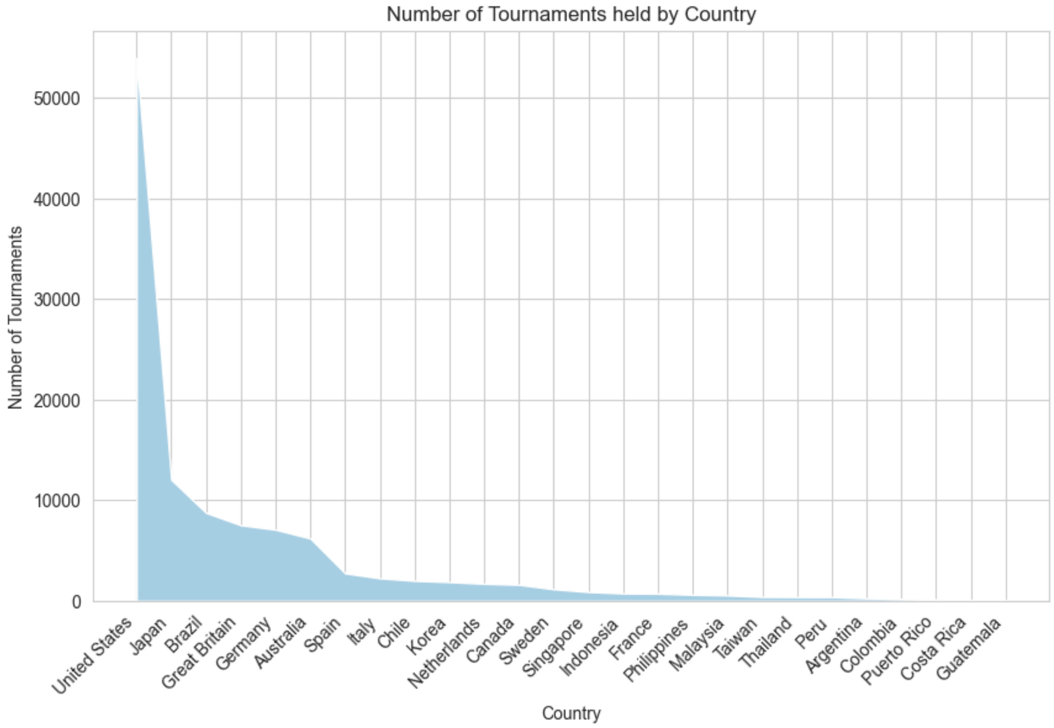


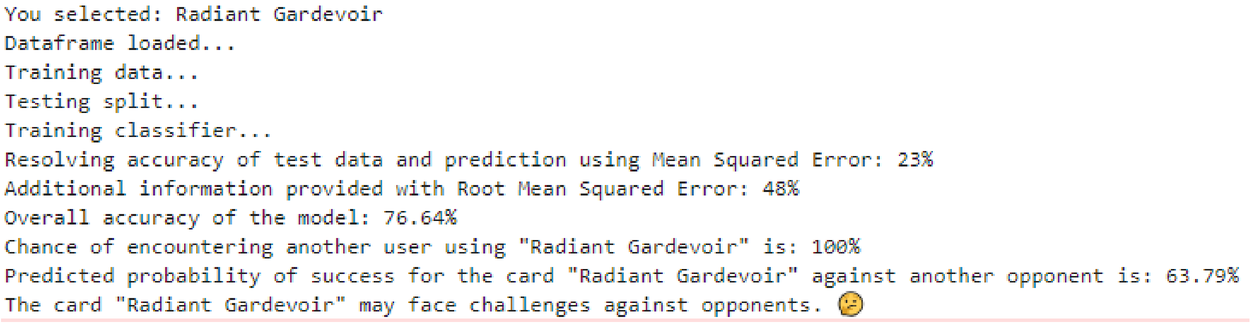
Figure D.4.3 – Number of Tournaments Held by Country

D.5. Accuracy Assessment

The accuracy assessment involves evaluating the C.H.A.M.P.S. machine learning model's ability to predict card success in Pokemon tournaments using metrics like mean squared error and root mean squared error. The assessment also includes measuring the model's overall predictive accuracy and its capacity to differentiate between successful and less successful card choices.

D.6. Results & Optimizations

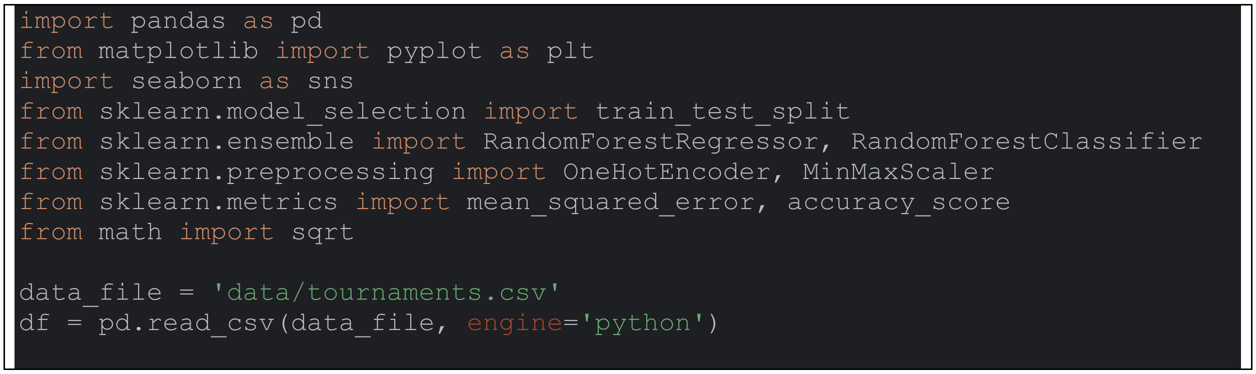
The results of all of these steps resulted in a product that provided an appropriate output which gives the user an idea of how successful the chosen card could be against an opponent if it’s included in their deck. Below is an example of how the output provides insight:



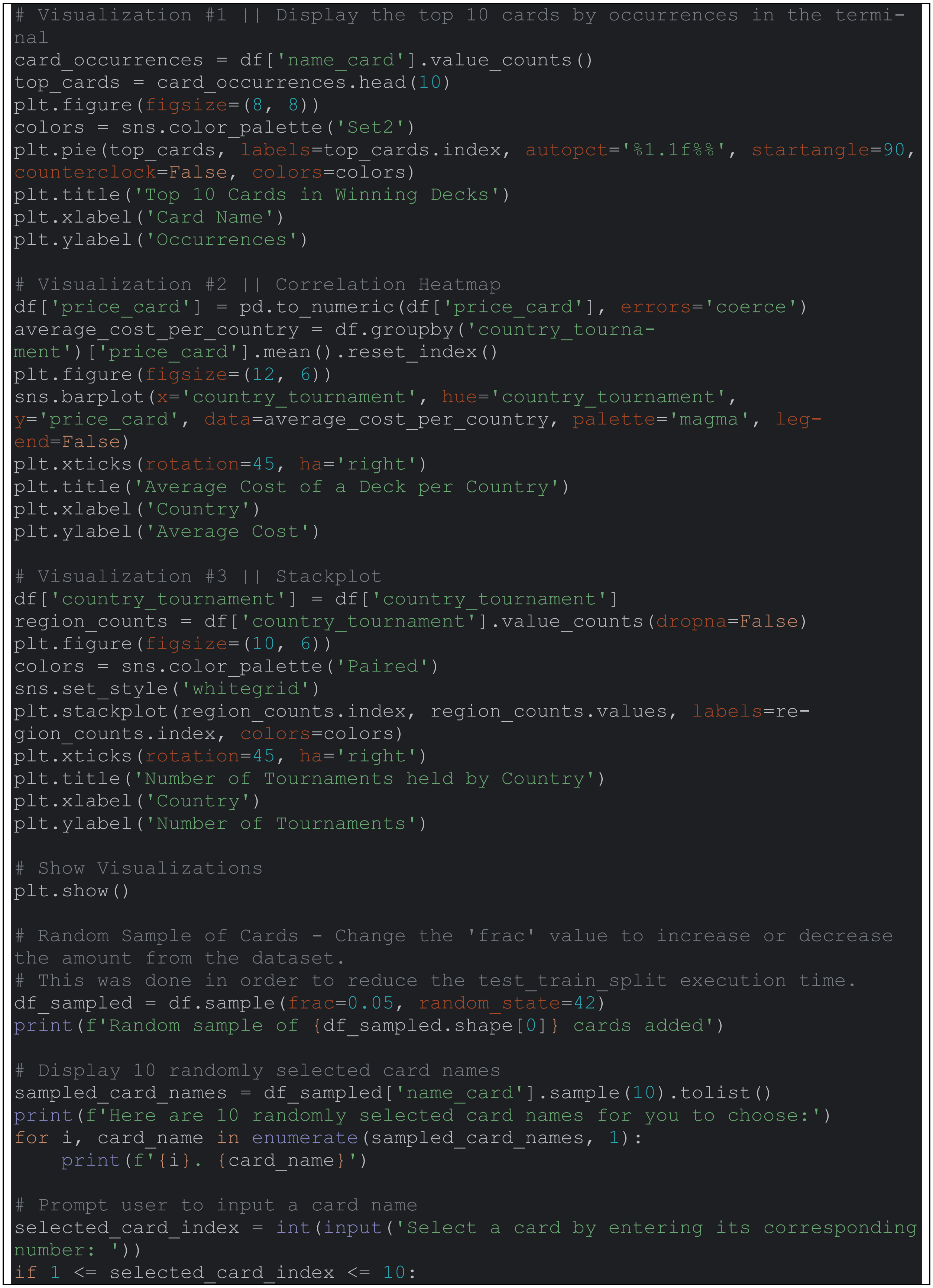
```
You selected: Radiant Gardevoir
Dataframe loaded...
Training data...
Testing split...
Training classifier...
Resolving accuracy of test data and prediction using Mean Squared Error: 23%
Additional information provided with Root Mean Squared Error: 48%
Overall accuracy of the model: 76.64%
Chance of encountering another user using "Radiant Gardevoir" is: 100%
Predicted probability of success for the card "Radiant Gardevoir" against another opponent is: 63.79%
The card "Radiant Gardevoir" may face challenges against opponents. 😕
```

Future optimizations would include additional predictive models that give a user a suggested card to add to their deck. Additionally, the project could also provide a space where a user could input a card and the algorithm could tell the user what deck-type that card appears in the most.

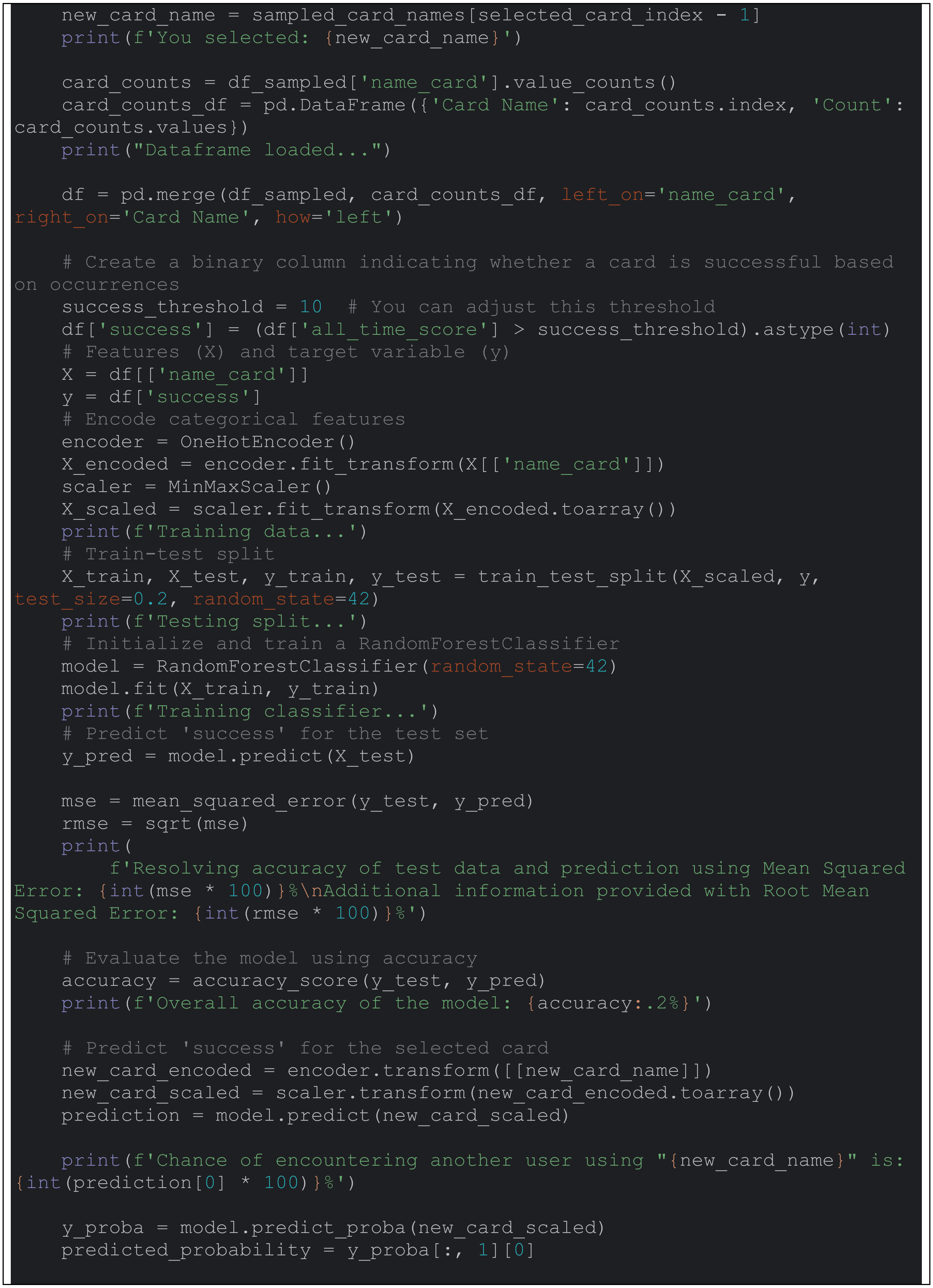
D.7. Source Code



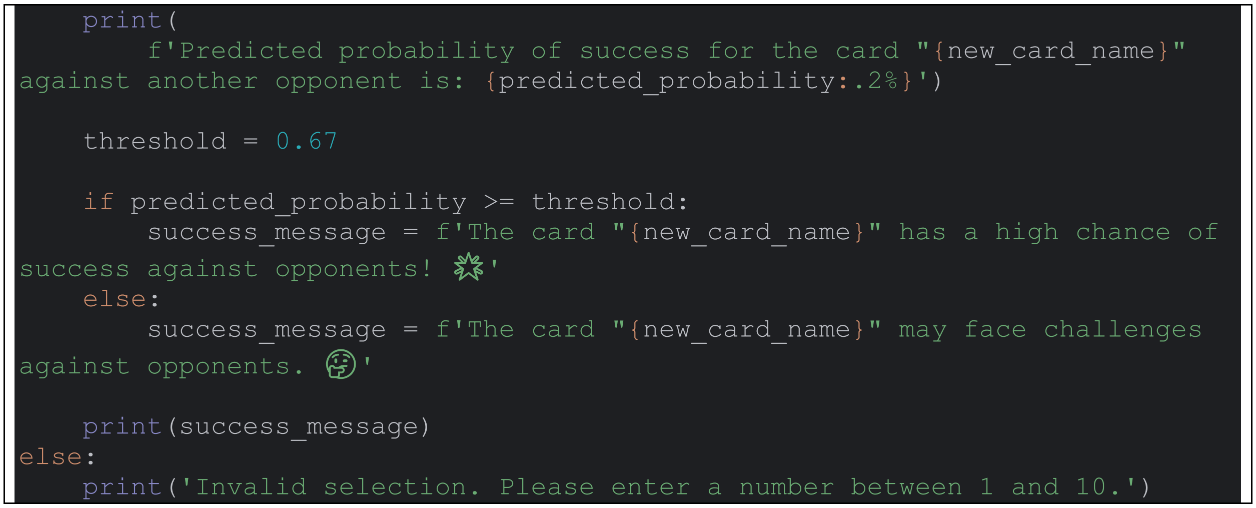
```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor, RandomForestClassifier
from sklearn.preprocessing import OneHotEncoder, MinMaxScaler
from sklearn.metrics import mean_squared_error, accuracy_score
from math import sqrt

data_file = 'data/tournaments.csv'
df = pd.read_csv(data_file, engine='python')
```



```python
# Visualization #1 || Display the top 10 cards by occurrences in the terminal
card_occurrences = df['name_card'].value_counts()
top_cards = card_occurrences.head(10)
plt.figure(figsize=(8, 8))
colors = sns.color_palette('Set2')
plt.pie(top_cards, labels=top_cards.index, autopct='%1.1f%%', startangle=90, counterclock=False, colors=colors)
plt.title('Top 10 Cards in Winning Decks')
plt.xlabel('Card Name')
plt.ylabel('Occurrences')

# Visualization #2 || Correlation Heatmap
df['price_card'] = pd.to_numeric(df['price_card'], errors='coerce')
average_cost_per_country = df.groupby('country_tournament')['price_card'].mean().reset_index()
plt.figure(figsize=(12, 6))
sns.barplot(x='country_tournament', hue='country_tournament', y='price_card', data=average_cost_per_country, palette='magma', legend=False)
plt.xticks(rotation=45, ha='right')
plt.title('Average Cost of a Deck per Country')
plt.xlabel('Country')
plt.ylabel('Average Cost')

# Visualization #3 || Stackplot
df['country_tournament'] = df['country_tournament']
region_counts = df['country_tournament'].value_counts(dropna=False)
plt.figure(figsize=(10, 6))
colors = sns.color_palette('Paired')
sns.set_style('whitegrid')
plt.stackplot(region_counts.index, region_counts.values, labels=region_counts.index, colors=colors)
plt.xticks(rotation=45, ha='right')
plt.title('Number of Tournaments held by Country')
plt.xlabel('Country')
plt.ylabel('Number of Tournaments')

# Show Visualizations
plt.show()

# Random Sample of Cards - Change the 'frac' value to increase or decrease the amount from the dataset.
# This was done in order to reduce the test_train_split execution time.
df_sampled = df.sample(frac=0.05, random_state=42)
print(f'Random sample of {df_sampled.shape[0]} cards added')

# Display 10 randomly selected card names
sampled_card_names = df_sampled['name_card'].sample(10).tolist()
print(f'Here are 10 randomly selected card names for you to choose:')
for i, card_name in enumerate(sampled_card_names, 1):
    print(f'{i}. {card_name}')

# Prompt user to input a card name
selected_card_index = int(input('Select a card by entering its corresponding number: '))
if 1 <= selected_card_index <= 10:
```



```
    new_card_name = sampled_card_names[selected_card_index - 1]
    print(f'You selected: {new_card_name}')

    card_counts = df_sampled['name_card'].value_counts()
    card_counts_df = pd.DataFrame({'Card Name': card_counts.index, 'Count':
card_counts.values})
    print("Dataframe loaded...")

    df = pd.merge(df_sampled, card_counts_df, left_on='name_card',
right_on='Card Name', how='left')

    # Create a binary column indicating whether a card is successful based
on occurrences
    success_threshold = 10  # You can adjust this threshold
    df['success'] = (df['all_time_score'] > success_threshold).astype(int)
    # Features (X) and target variable (y)
    X = df[['name_card']]
    y = df['success']
    # Encode categorical features
    encoder = OneHotEncoder()
    X_encoded = encoder.fit_transform(X[['name_card']])
    scaler = MinMaxScaler()
    X_scaled = scaler.fit_transform(X_encoded.toarray())
    print(f'Training data...')
    # Train-test split
    X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.2, random_state=42)
    print(f'Testing split...')
    # Initialize and train a RandomForestClassifier
    model = RandomForestClassifier(random_state=42)
    model.fit(X_train, y_train)
    print(f'Training classifier...')
    # Predict 'success' for the test set
    y_pred = model.predict(X_test)

    mse = mean_squared_error(y_test, y_pred)
    rmse = sqrt(mse)
    print(
        f'Resolving accuracy of test data and prediction using Mean Squared
Error: {int(mse * 100)}%\nAdditional information provided with Root Mean
Squared Error: {int(rmse * 100)}%')

    # Evaluate the model using accuracy
    accuracy = accuracy_score(y_test, y_pred)
    print(f'Overall accuracy of the model: {accuracy:.2%}')

    # Predict 'success' for the selected card
    new_card_encoded = encoder.transform([[new_card_name]])
    new_card_scaled = scaler.transform(new_card_encoded.toarray())
    prediction = model.predict(new_card_scaled)

    print(f'Chance of encountering another user using "{new_card_name}" is:
{int(prediction[0] * 100)}%')

    y_proba = model.predict_proba(new_card_scaled)
    predicted_probability = y_proba[:, 1][0]
```



```
    print(
        f'Predicted probability of success for the card "{new_card_name}"
against another opponent is: {predicted_probability:.2%}')

    threshold = 0.67

    if predicted_probability >= threshold:
        success_message = f'The card "{new_card_name}" has a high chance of
success against opponents! 🌟'
    else:
        success_message = f'The card "{new_card_name}" may face challenges
against opponents. 🤔'

    print(success_message)
else:
    print('Invalid selection. Please enter a number between 1 and 10.')
```

D.8. Quick Start Guide

Step 1: Click the following link: https://mybinder.org/v2/gh/ethanmaxwellharris/C964_Capstone/HEAD

Step 2: Within the folder structure on the left-hand side double-click: **pokechamps.ipynb**

Step 3: Review the source code & visualizations

Step 4: Run the interactive cell by clicking on "Run" at the toolbar on top, then selecting "Run All Cells"

Step 5: Follow the prompt and enter a number between 1 and 10.

ALTERNATIVE QUICK START GUIDE:

Note: mybinder.org has been finicky due to the large dataset – if that's happening follow these alternative steps to host the Jupyter Lab locally.

Step 1: Install Python

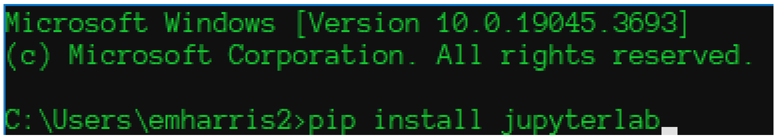
- Ensure that Python is installed on your machine. You can download the latest version of Python from https://www.python.org/downloads/.

Step 2: Install Jupyter Lab

- Open a command prompt or terminal on your machine.

- Run the following command to install Jupyter Lab using Python's package manager, pip:

```
pip install jupyterlab
```



```
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\Users\emharris2>pip install jupyterlab
```

Step 3: Download the Jupyter Lab File

- Enter the following GitHub into your browser: https://github.com/ethanmaxwellharris/C964_Capstone

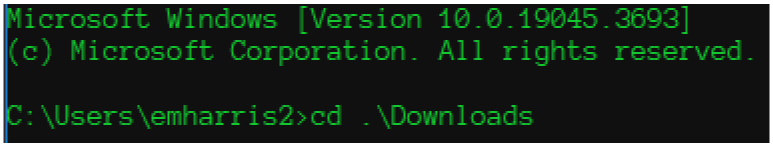
- Download the project by clicking the green “<> Code” droptown, then ‘Download ZIP’ to your desired location on your local machine.

Step 4: Open Jupyter Lab

- Open a command prompt or terminal.

- Navigate to the directory where you saved the Jupyter Lab files using the `cd` command. For example:

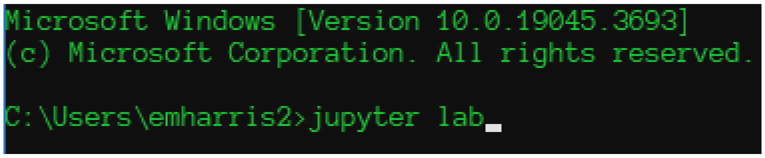
```
cd .\Downloads
```



```
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\Users\emharris2>cd .\Downloads
```

- Run the following command to start the Jupyter Lab server:

```
jupyter lab
```



```
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\Users\emharris2>jupyter lab_
```

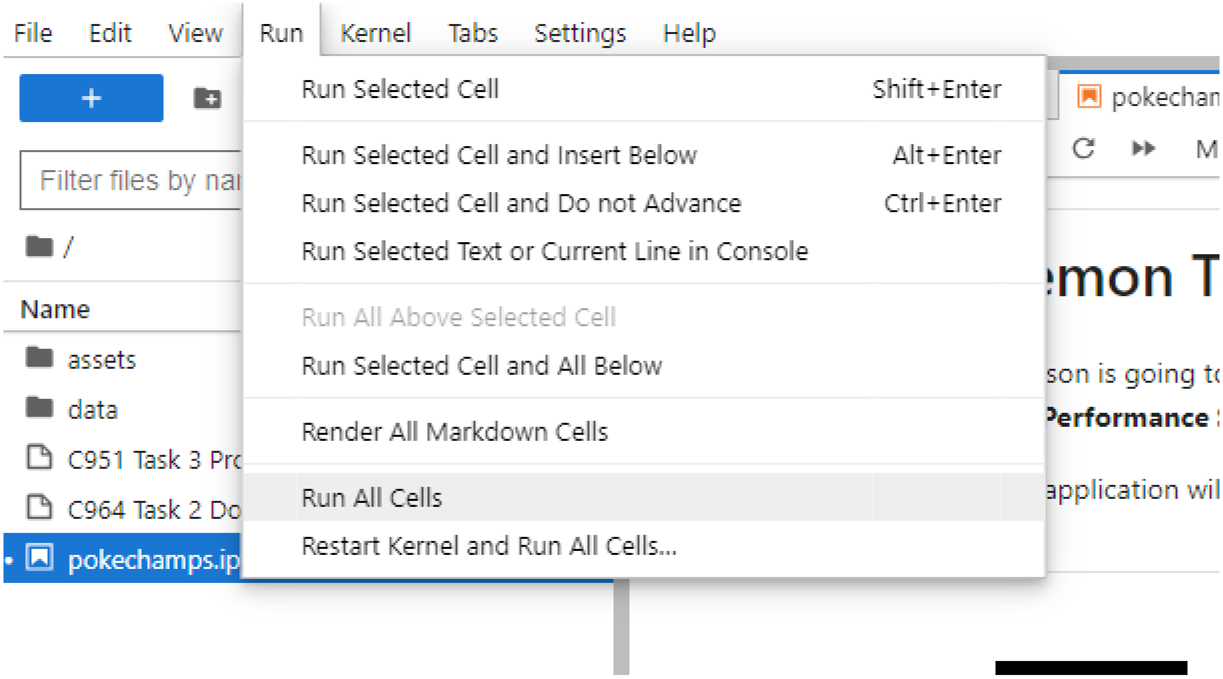
- Your default web browser will open, displaying the Jupyter Lab dashboard.

Step 5: Access and Run the Lab

- In the Jupyter Lab dashboard, you'll see a list of files in the current directory. Click on the Jupyter Lab file you downloaded to open it. Hint: It's titled 'pokechamps.ipynb'

- The Jupyter Lab interface will open in a new browser tab. You can navigate through the notebook cells and run code cells by clicking the "Run" button or using keyboard shortcuts.

- Navigate through each cell to view the visualizations and when ready click the 'Run' dropdown on the topmost toolbar, then 'Run All Cells'


File
Edit
View
Run
Kernel
Tabs
Settings
Help
+
Filter files by nai
/
Name
assets
data
C951 Task 3 Prc
C964 Task 2 Do
pokechamps.ip
Run Selected Cell Shift+Enter
Run Selected Cell and Insert Below Alt+Enter
Run Selected Cell and Do not Advance Ctrl+Enter
Run Selected Text or Current Line in Console
Run All Above Selected Cell
Run Selected Cell and All Below
Render All Markdown Cells
Run All Cells
Restart Kernel and Run All Cells...
pokechan
M
mon T
son is going tc
Performance
application wil

Step 6: Interact with PokeChamps

- Enter 1 through 10 for your desired card and see what the prediction outputs.

That's it! You've successfully downloaded, installed, and opened the Jupyter Lab on your local machine.

Thank you for taking the time to read through the documentation and more importantly thank you for the time you’ve put into reviewing my Capstone!