Introduction

We decided to create a project that aims to explore the intersection of data science and pop culture by using machine learning to predict the outcomes of randomized battles against an average opponent. Our predictive model analyzes various superheroes' attributes—such as strength, speed, intelligence, weaknesses and special abilities—to determine the likely winner in combat scenarios. By selecting and preprocessing data on a diverse roster of heroes, we trained a logistic regression model to make informed predictions about who would be victorious.

To enhance our findings and encourage more data exploration, we deployed our model within an interactive web application, allowing users to input different scenarios and visualize the predicted outcomes. We also developed two Tableau dashboards to provide an insightful comparison of superhero statistics from the machine learning model and from supplemental hero data to offer a dynamic platform for users to engage with. Our ultimate goal was to create a robust machine learning model that not only predicts the victors in these battles but also provides a deeper understanding of the attributes that contribute to a superhero's potential success.

Data Cleaning

For our project, we started with three datasets including: a dataset containing randomized statistics for superheroes, a dataset that contained basic hero information, and a dataset that contained information about what heroes had what powers. All three required thorough cleaning and preprocessing to ensure their suitability for model training and data visualization in Tableau.

Initial Dataset Inspection

We began by inspecting the dataset for null values and missing entries. This involved checking each column for completeness. Initially, we identified a column labeled “Comic Universe” in the dataset we intended to use for machine learning. This column contained randomized entries that did not accurately reflect the actual universe of each superhero. To address this we made lists of the characters and their correct universe and then used a .apply to create a new column with the correct values.

```
#making list of character names to make it eaiser
marvel_characters = ['Iron Man','Spider-Man','Thor','Captain America']
dc_characters = ['Wonder Woman','Flash','Batman','Superman']
```

```
# Creating a NEW universe column based on the condition for the correct character universe
battle_info2['Universe'] = battle_info2['Character'].apply(lambda x: 'Marvel' if x in marvel_characters else 'DC Comics')
battle_info2.head(25)
```

Cleaning Superhero Information

Next, we turned our attention to a separate dataset containing essential information about the superheroes. Upon inspection, we found that while there were no initial null values, some columns had entries filled with hyphens, which represented missing data. We identified the columns affected—specifically, eye color, hair color, race, and skin color. We replaced the hyphens with NaN (Not a Number) values to standardize our representation of missing data.

```
# Replace hyphens with NaN for race now
hero_info['Race'].replace('-', np.nan, inplace=True)
hero_info.head(15)
```

This process then revealed a significant issue: dropping the rows containing NaN values led to a substantial loss of data, resulting in the removal of 90% of the dataset. To mitigate this loss, we decided to drop the columns with the most missing data—eye color, hair color, race, and skin color—in an effort to keep as much data as possible.

```
#dropping all the NaN values in the copy of the data frame
hero_info_clean1 = hero_info_clean1.dropna()
hero_info_clean1.head(15)
```

Generalizing Superhero Powers

To enhance the hero powers dataset's usability, we also grouped hero powers into more general categories. This restructuring involved creating broader columns for physical attributes, energy-related abilities, elemental powers, magic/cosmic powers, and several other categories. This consolidation helped reduce the dimensionality of the dataset, making it easier to analyze and interpret.

```
# Group the columns into lists as per the categories you defined
physical_attributes = ['Agility', 'Super Strength', 'Dexterity', 'Stamina', 'Reflexes',
                       'Peak Human Condition', 'Enhanced Senses', 'Enhanced Hearing',
                       'Enhanced Sight', 'Enhanced Smell', 'Enhanced Touch', 'Super Speed', 'Jump']

energy_related = ['Lantern Power Ring', 'Energy Absorption', 'Energy Armor', 'Energy Blasts',
                  'Energy Beams', 'Energy Constructs', 'Energy Manipulation', 'Energy Resistance']

elemental_powers = ['Fire Control', 'Water Control', 'Wind Control', 'Terrakinesis',
                    'Weather Control', 'Cryokinesis', 'Heat Resistance', 'Heat Generation',
                    'Element Control', 'Elemental Transmogrification', 'Seismic Power']

magic_cosmic = ['Magic', 'Darkforce Manipulation', 'Power Cosmic', 'Odin Force', 'Phoenix Force',
                'Symbiote Costume']
```

Handling Negative Values

During our cleaning process, we encountered negative values in the height and weight columns in the hero information dataset, which are not feasible for real-world data. Heroes don't generally have negative heights or weights. To address this issue, we calculated the mean and median of the valid entries in those columns. Instead of dropping the rows with negative values, we replaced them with the median values. This approach allowed us to retain more data while ensuring the accuracy of the height and weight attributes.

```
# Calculate mean and median of the Weight column (excluding -99)
mean_weight = merged_df[merged_df['Weight'] != -99]['Weight'].mean()
median_weight = merged_df[merged_df['Weight'] != -99]['Weight'].median()

# Print the results
print(f"Mean Weight (excluding -99): {mean_weight}")
print(f"Median Weight (excluding -99): {median_weight}")
```

Merging Datasets for Tableau Integration

After completing the cleaning process, we merged the superhero information dataset with the superhero powers dataset based on the heroes' names. This merger created a comprehensive dataset that combined both descriptive information and powers, making it easier to analyze and visualize in Tableau. By merging these datasets, we were able to present unified insights and comparisons across multiple attributes in a clear, intuitive manner.

```
# Rename the 'hero_names' column in df to 'name' to match with df1 for merging
df = df.rename(columns={'hero_names': 'name'})

# Perform a merge on the 'name' column
merged_df = pd.merge(df1, df, on='name', how='inner')  # 'inner' keeps only matching rows

# Verify the merged DataFrame
merged_df.info()
```

Machine Learning

Before jumping into model training, we ensured we had all the required imports for libraries like pandas, scikit-learn, and pickle to handle the data processing, model training, and saving. After the initial cleaning we decided to drop the “Universe” column as we determined it wouldn’t add much value to the predictions.

Next, we moved to one of the most important steps in the preprocessing phase: handling categorical data. We one-hot encoded categorical columns such as special abilities, character, and weaknesses to transform them into numerical format, making them suitable for machine learning algorithms. For numerical data, including speed, strength, and intelligence, we used a StandardScaler to normalize the values, ensuring that features were on the same scale.We initially attempted to use a preprocessing pipeline to automate the transformation process. However, we faced several roadblocks, so instead of initially struggling with the pipeline we decided to encode everything manually. While this worked, it increased the workload later, as we had to manage each step manually when working on the web app.

Model Selection and Training

Once the data was preprocessed, we trained the model using the "battle outcome" as the target variable. We set up a function to easily run different models, which streamlined the testing of various machine learning algorithms.

We tested several models, including:

- Random Forest
- Logistic Regression
- Support Vector Classifier (SVC)
- Decision Tree
- Extra Trees
- AdaBoost
- Gradient Boost

After comparing the performance metrics of these models, Logistic Regression proved to be the best fit for our dataset. It provided consistent results and good interpretability for predicting battle outcomes.

	Model	Accuracy	Precision	Recall	F1 Score	AUC	Confusion Matrix
0	Random Forest	0.761905	0.593750	0.463415	0.520548	0.815053	[[372, 52], [88, 76]]
1	Logistic Regression	0.799320	0.669118	0.554878	0.606667	0.848143	[[379, 45], [73, 91]]
2	Support Vector Classifier	0.785714	0.682692	0.432927	0.529851	0.821136	[[391, 33], [93, 71]]
3	Decision Tree	0.741497	0.537975	0.518293	0.527950	0.679497	[[351, 73], [79, 85]]
4	Extra Trees	0.760204	0.592000	0.451220	0.512111	0.778863	[[373, 51], [90, 74]]
5	AdaBoost	0.795918	0.664179	0.542683	0.597315	0.845332	[[379, 45], [75, 89]]
6	Gradient Boosting	0.797619	0.671756	0.536585	0.596610	0.841312	[[381, 43], [76, 88]]

Model Saving with Pickle

After selecting Logistic Regression as our final model, we retrained it using all the available data to ensure it captured as much information as possible. We then saved the model using pickle so it could be reused in our web application. However, this is where we encountered some unexpected issues. Despite the model saving successfully, the file size of the pickle file was much smaller than anticipated, raising concerns about its completeness and functionality.

Testing the Model

Before deploying the model, we tested the model in a jupyter notebook by loading the pickle file to ensure it was production-ready. We defined a make_predictions function, which allowed us to make predictions by passing in the relevant attributes of the superheroes.This is where we should have taken the time to figure out the preprocessing pipeline to make our function cleaning and easier to work with. After running tests with two characters, we confirmed that the model was functioning correctly in a controlled environment.

```
character = "Batman"
strength = 6
speed = 3
intelligence = 4
specialAbilities = "Telekinesis"
weaknesses = "Kryptonite"


makePredictions(character,strength,speed,intelligence,specialAbilities,weaknesses)
```

```
0.9994903057353131
```

```
character = "Iron Man"
strength = 1
speed = 1
intelligence = 1
specialAbilities = "Telekinesis"
weaknesses = "Kryptonite"


makePredictions(character,strength,speed,intelligence,specialAbilities,weaknesses)
```

```
0.19327923645219897
```

Challenges with Deployment

After successful testing in our local environment, we began integrating the model into our web app. This is where we encountered significant hurdles. When attempting to load and use the model within the web app, we received errors indicating that the pickle files could not be found or loaded correctly. Upon further investigation, it appeared that the model files were either not saving correctly or were incomplete, as their file sizes were unusually small.

We are continuing to debug the issue, but these challenges have slowed down the deployment of our web app. The next steps involve ensuring the pickle files save correctly and troubleshooting why they are not loading as expected in the web environment.

Dashboard Design Concepts

For our dashboards, we created one for each of our chosen datasets: the superhero dataset and the battle dataset used for the machine learning model.

The first dashboard focuses on the battle dataset, where randomized qualities such as strength, speed, and intelligence were assigned to superheroes. This dataset formed the basis for training our machine learning model to predict battle outcomes. The dashboard provides an overview of how these randomized qualities are distributed across different superheroes, giving users insights into the traits that were key to determining whether a superhero would win a battle. This distribution-based visualization allows for an easy exploration of how various attributes, like intelligence or strength, played a role in the predictions.

The second dashboard focuses on the superhero dataset and provides insights into the real attributes of over 600 superheroes. This dashboard allows users to explore distributions related to height, weight, alignment, and publisher. It also displays the number of powers each superhero has, offering a breakdown of power categories and showing how many heroes possess each type of power. Filters for gender and alignment further enable users to dive into specific groups and compare characteristics. The dashboard presents these distributions in an accessible way, giving a comprehensive view of how superhero traits and powers are spread across different categories.

Analysis

The analysis of the randomized battle dataset and the real-world superhero dataset reveals key insights about the factors influencing superhero success and the distribution of superhero traits.

In the battle dataset, the results indicate that the characteristics assigned to each superhero played a critical role in their number of wins. The dataset did a good job of maintaining balance, with most superheroes achieving a relatively even number of wins based on their assigned traits. The analysis shows that super strength was consistently the most beneficial attribute for securing wins, while kryptonite as a weakness significantly reduced a superhero's chances of victory. Interestingly, despite weaknesses being included, they generally had less impact on the outcomes, with kryptonite being the notable exception. Special abilities like telekinesis and flight appeared to reduce the number of wins, while invisibility and Iron Man presented an anomaly, leading to skewed results that warrant further investigation. These findings suggest that, while weaknesses do play a role, it is the positive traits—particularly super strength—that are most predictive of success in the model's battle outcomes.

In the superhero dataset, several important distributions were observed. Height and weight distributions show slight right skews, which is understandable given that a few superheroes, such as the Hulk, are exceptionally large. The most common height for superheroes was around 182 cm (5'9"), and the most common weight was around 81 kg (170 lbs). This suggests that most superheroes fall within a fairly average range for physical stature, while a small subset exhibits extreme values.

The distribution of powers across superheroes revealed that the majority possess between one and three powers, with enhanced physical traits being the most common, followed by enhanced survival and travel-related powers. This supports the expectation that many superheroes rely on physical prowess and abilities to survive and thrive in their respective narratives. Additionally, the breakdown of alignment shows that more superheroes are good-aligned than bad, which is consistent with the nature of superhero storytelling, where the focus is typically on heroic figures.

Finally, the analysis of publishers confirms that Marvel and DC dominate the superhero landscape, being the largest contributors to the roster of heroes included in the dataset. This highlights the significant role these two companies play in shaping the popular understanding of superheroes.

Bias and limitations

One major bias in our model is that we don't know the true stats of the "average opponent" that each superhero is being compared against. Since we are predicting outcomes based on a simplified dataset, the average opponent's characteristics are assumed, which might not reflect real-world scenarios or battles in comics accurately. This can lead to skewed predictions that might favor certain heroes over others. There is also the issue that the heroes in this dataset do not generally have the special abilities or weaknesses making it hard to truly see if they would win in a battle.

Another bias stems from the limited scope of our dataset. With only eight characters, four special abilities, and four weaknesses, the diversity of inputs is quite constrained. While this provides a good starting point for showcasing the model, it does not allow for the complexity or variability that real superhero matchups might involve. This narrow scope also limits the model’s capacity to handle a wider

range of potential user inputs. A more comprehensive dataset with more characters, powers, and weaknesses would greatly enhance the model's accuracy and robustness.

One of the major technical limitations we faced during this project was related to deploying the model in our web application. Despite successfully training and testing the model locally, we encountered significant issues when trying to load the pickle files in the web app environment. This prevented us from fully deploying the model to allow real-time predictions via the web app. Debugging this issue has been challenging, but with further investigation, we believe this can be resolved. Also, the use of manual preprocessing introduced complexity, which led to increased chances for errors and extra work in transforming the data. A more automated preprocessing pipeline would streamline this process in future iterations.

Conclusions

This project explored how superhero traits influence battle outcomes, with super strength proving to be the most impactful attribute in wins, while kryptonite consistently lowered chances of success. Other weaknesses had little effect, with some anomalies, like Iron Man with invisibility, requiring further investigation.

In the broader superhero dataset, most heroes had 1-3 powers, with enhanced physical abilities being the most common. The distributions of height and weight showed slight right skews, and good-aligned heroes outnumbered bad ones. Marvel and DC dominated the publisher landscape, reflecting their major influence on superhero narratives.

While we encountered challenges in deploying our machine learning model, this project demonstrated the potential of predictive modeling in understanding superhero success. Our dashboards provided dynamic visual insights and laid a foundation for expanding the model with more diverse data in the future.

Bias/Limitations

While we didn't have many issues with the data there are some common limitations that can occur when doing an ETL pipeline. Some of those limitations include data quality issues, data loss, and processing time. When creating the pipeline it's important to make sure that you are using quality data. Data that is inaccurate, incomplete, or inconsistent can lead to errors in the transform stage and can cause issues with the analysis and results. Data loss is another very important limitation. There is risk of corruption or losing data at any point in the ETL process. To help mitigate data loss issues in our project we made a few copies of dataframes to ensure that we had back ups just in case something was deleted or changed. Processing time is another limitation that could cause issues when creating an ETL pipeline. If you are trying to process large amounts of data it could slow down processing times and can impact the efficiency of the project. This could then impact how long it takes to make data-driven decisions with the pipeline.

Conclusions/Reflection

It's a wrap! As a data analyst, transforming and manipulating data can provide a better understanding of the factors that contribute to the success or failure of a project in this case the crowdfunding data provided just that. The opportunity to derive insights that can help future campaigners optimize their chances of success. This project was a great test of our skills in creating an ETL pipeline and how we may be able to use it in the future.

References

Xpert by edX Boot Camps LLC and is intended for educational purposes only.

Panda web inquiry: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html

Classroom exercises from Boot Camp Consortium East Coast DATA-PT-EAST-APRIL-041524

Postgresql Inquiry:
https://www.postgresql.org/docs/

Regex Inquiry
https://www.rexegg.com/regex-quickstart.php