Project 5: Machine Learning

John Wesley Mathis

Dr. Anthony Choi

June 19, 2024

ECE/SSE 591, Summer 2024

Table of Contents:

[Deliverable Table 6](#__RefHeading___Toc2569_331549091)

[1. Introduction 7](#__RefHeading___Toc2573_331549091)

[2. Predicting Ice Cream Sales Linear Regression 8](#__RefHeading___Toc2577_331549091)

[3. Fruit Classification 8](#__RefHeading___Toc7733_1546921247)

[4. Pokemon Classification 8](#__RefHeading___Toc7735_1546921247)

[5. Conclusion 9](#__RefHeading___Toc2609_331549091)

[References 10](#__RefHeading___Toc3692_331549091)

Table of Figures

[Figure 1: Importing Libraries and Customizing Matplotlib 8](#Figure!40|sequence)

[Figure 2: SIR Model Code 9](#Figure!43|sequence)

[Figure 3: SIR Model Dynamics Line Plot Code 9](#Figure!0|sequence)

[Figure 4: SIR Model Dynamics Line Plot 10](#Figure!1|sequence)

[Figure 5: Peak Infection Day Annotation Code 10](#Figure!2|sequence)

[Figure 6: Peak Infection Day Annotation Plot 11](#Figure!3|sequence)

[Figure 7: Monte Carlo Simulation Code 12](#Figure!44|sequence)

[Figure 8: Monte Carlo Simulation Results Code 12](#Figure!4|sequence)

[Figure 9: Monte Carlo Simulation Results Scatter Plot 13](#Figure!5|sequence)

[Figure 10: Final Infected Individuals Histogram Code 13](#Figure!6|sequence)

[Figure 11: Final Infected Individuals Histogram Plot 14](#Figure!7|sequence)

[Figure 12: 3D Plot of SIR Model Code 14](#Figure!8|sequence)

[Figure 13: 3D Plot of SIR Model 15](#Figure!9|sequence)

[Figure 14: Contour Plot of Final Infected Individuals Code 16](#Figure!45|sequence)

[Figure 15: Contour Plot of Final Infected Individuals 16](#Figure!10|sequence)

[Figure 16: Interactive SIR Model Dynamics Code 17](#Figure!11|sequence)

[Figure 17: Interactive SIR Model Dynamics Plot and Layout 18](#Figure!12|sequence)

[Figure 18: Trajectory Plot Code 19](#Figure!13|sequence)

[Figure 19: Trajectory Plot (a) 19](#Figure!49|sequence)

[Figure 20: Trajectory Plot (b) 20](#Figure!14|sequence)

[Figure 21: Movie Analysis Code (a) 21](#Figure!46|sequence)

[Figure 22: Movie Analysis Code (b) 22](#Figure!47|sequence)

[Figure 23: Movie Analysis Code (c) 22](#Figure!48|sequence)

[Figure 24: Distribution of IMDB Ratings Code 23](#Figure!15|sequence)

[Figure 25: Distribution of IMDB Ratings Histogram Plot 23](#Figure!16|sequence)

[Figure 26: Distribution of Gross Earnings Code 24](#Figure!17|sequence)

[Figure 27: Distribution of Gross Earnings 24](#Figure!18|sequence)

[Figure 28: Distribution of Movie Runtimes Code 25](#Figure!19|sequence)

[Figure 29: Distribution of Movie Runtimes 25](#Figure!20|sequence)

[Figure 30: Number of Movies Released per Year Code 26](#Figure!21|sequence)

[Figure 31: Number of Movies Released per Year Plot 26](#Figure!22|sequence)

[Figure 32: Average IMDB Rating by Genre Code 27](#Figure!23|sequence)

[Figure 33: Average IMDB Rating by Genre Plot 27](#Figure!24|sequence)

[Figure 34: Scatter Plot of Gross Earnings vs IMDB Rating Code 28](#Figure!25|sequence)

[Figure 35: Scatter Plot of Gross Earnings vs IMDB Rating 28](#Figure!26|sequence)

[Figure 36: Visualizing Top Directors Code 29](#Figure!27|sequence)

[Figure 37: Visualizing Top Directors Pie Chart 29](#Figure!28|sequence)

[Figure 38: Box Plot of IMDB Ratings by Genre Code 30](#Figure!29|sequence)

[Figure 39: Box Plot of IMDB Ratings by Genre 30](#Figure!30|sequence)

[Figure 40: Pairplot of Ratings, Gross Earnings, and Runtime Code 31](#Figure!31|sequence)

[Figure 41: Pairplot of Ratings, Gross Earnings, and Runtime 32](#Figure!50|sequence)

[Figure 42: Genre Popularity over Time with Area Plot Code 33](#Figure!32|sequence)

[Figure 43: Genre Popularity over Time with Area Plot 33](#Figure!33|sequence)

[Figure 44: Average IMDB Rating by Genre with Error Bars Code 34](#Figure!34|sequence)

[Figure 45: Average IMDB Rating by Genre with Error Bars 34](#Figure!35|sequence)

[Figure 46: Importing and Cleaning COVID-19 Dataset Code 35](#Figure!51|sequence)

[Figure 47: Importing and Cleaning COVID-19 Dataset Output 35](#Figure!52|sequence)

[Figure 48: Visualizing COVID-19 Cases in the US with Basemap Code 36](#Figure!36|sequence)

[Figure 49: Visualizing COVID-19 Cases in the US with Basemap 36](#Figure!37|sequence)

[Figure 50: Subplots of Average Number of Confirmed COVID-19 Cases and Deaths by State Code 37](#Figure!42|sequence)

[Figure 51: Subplots of Average Number of Confirmed COVID-19 Cases and Deaths by State 38](#Figure!41|sequence)

[Figure 52: Importing Pokemon dataset code 39](#Figure!53|sequence)

[Figure 53: Output of Pokemon dataset 40](#Figure!54|sequence)

[Figure 54: Violin Plot with Pokemon Data Code 40](#Figure!38|sequence)

[Figure 55: Violin Plot with Pokemon Data 41](#Figure!39|sequence)

# Deliverable Table

The purpose of this table is to provide a complete view of the concepts covered in chapter 5 of *"Python Data Science Handbook"* (VanderPlas, 2016) and provide a general page location for where the topic was demonstrated.

|  |  |
| --- | --- |
| Deliverables | Location |
| What is Machine Learning? |  |
| Introducing Scikit-Learn |  |
| Hyperparameters and Model Validation |  |
| Feature Engineering |  |
| In-Depth: Decision Trees and Random Forests |  |

Additionally, here is a link to my GitHub were the datasets and the Jupyter Notebook for the project can be downloaded: https://github.com/jwmathis/SSE591\_Project5. In order to run the file, Python and other dependencies must be installed.

# 1. Introduction

This report aims to demonstrate my proficiency in using machine learning to build mathematical models to understand data as covered in Chapter 5 of the “Python Data Science Handbook” by Jake VanderPlas (2016). This report attempts to illustrate the core concepts and functionalities of lby implementing the concepts into practical examples. The code presented in this report was developed using Visual Studio Code with Jupyter Notebook extensions. I will provide detailed explanations, highlighting key features and operations that make machine learning an essential tool for data analysis.

# 2. Predicting Ice Cream Sales Linear Regression

My first example I demonstrate how to perform a simple linear regression using scikit-learn, from data I generated to model evaluation and visualization. For this I decided to model ice cream sales and correlate it to the temperature. To generate the data I use *np.random.seed(42)* to generate random numbers and ensure that the same numbers are generated every time. I use *np.random.normal(30, 10, 100)* to generate temperature values with a mean of 30C and standard deviation of 10C. To better simulate ice cream sales, I use a linear function of temperature with added random noise. To begin the machine learning, I first define my features matrix and target vector by reshaping the temperature array to a 2D array. Then I import *train\_test\_split*  to split the data into training and testing sets. Afterwards I follow the procedure of training the model as outlined in chapter 5. I choose my model, in this case *LinearRegression,* I instantiate the model, then I fit the model to the training data, *Xtrain and ytrain.* Lastly, I use the trained model to predict ice cream sales on the test data. Next I evaluate the model by importing the metrics *mean\_squared\_error t*o computethe mean squared error between actual values and predicted sales and  *r2\_score* to compute the R-squared value to have some indication of the proportion of the variance in the dependent variable that is predictable form the independent variable. Lastly, I visualize the results using Matplotlib’s *plt.scatter* and *plt.plot* to plot the actual test data points and the regression line based on the model’s predictions. From this example I was able to walk through the process of generating synthetic data, splitting the data into training and testing sets, training a linear regression model, and evaluating and visualizing the performance and results.

# 3. Fruit Classification

For my next example, I demonstrate how to process categorical data by making a simple fruit classification program. To begin, I constructed a dataFrame called *fruit\_df* containing columns called *Weight, Color, Size, and Fruit* containing the data of information that I defined in a dictionary. This example served as my means of better understanding how to convert and process data to be used for machine learning models. I used *Labelencoder* to encode the features of the fruits from categorical variables into numerical labels. Initially, I used *le = LabelEncoder()*  for all the features. Unfortunately, this resulted in errors as I quickly learn that LabelEncoder() stores only the most recent information. So, when I would try to decode the information, I would get incorrect results. To fix this, I created three separate variables to encode the features. Using the *fit\_transform* attribute I transformed the columns of Color, Size, and Fruit. To ensure the DataFrame was properly encoded, I printed the decoded versions to the screen using the attribute *inverse\_transform.* This allowed me to verify the correctness of the encoding by inversely transforming the numerical labels back to their original values. Now that the data is processed and in a format that can be used for machine learning, I begin the process of training a model as outline in chapter 5. I define my features matrix and target vector; split the data into training and testing sets. For this example, I instantiate a decision tree classifier and fit the training data to this model and predict on the test data. For this example, I use *accuracy\_score*  to evaluate the performance of the model and I use Seaborn’s *pairplot* to visualize the relationships in *fruit\_df*. Each feature (*Weight, Color, Size)* is plotted, with points colored by *Fruit* category. This example was simple but it allowed me begin understanding the process how to handle categorical data and build a simple machine learning model.

# 4. Pokemon Classification

For the next example, I attempt to classify Pokemon based off their legendary status. First I load the Pokemon data obtained from Kaggle and display the first few rows. Then I begin to define my features matrix using the columns *HP, Attack, Defense, Sp. Atk, Sp. Def, and Generation* and the target vector as the *Legendary* column and split the data using *train\_test\_split.* Then I train a *RandomForestClassifier* model and make predictions on the test data. For visualization, I extract a single decision tree from the *RandomForestClassifier* model using the *estimators\_[]* attribute. Then I used *plot\_tree* to plot the decision tree with features and class names. In order to properly evaluate the model, I import the metrics *accuracy­\_score* to calculate the proportion of correct predictions *and classification\_report* to provide a more detailed analysis of the performance of the model. Then I visualized the model using a *confusion\_matrix and ConfusionMatrixDisplay.*

# 5. Predicting Animal Species

For my last project I tried to make a comprehensive example using immage classification and clustering tasks. First I import the necessary libraries such as *numpy, matplotlib.pyplot* and necessary packages such as *imread and resize.* With proper libraries imported, I then load my images folder, *animals-10* and get the class names from the folder names. Because I was using a MacBook when coding this section a macOS system file was present so in order to deal with this being included as a class name, I remove it from the list if it was present. Next I use a nested for loop if conditional statemtns to read images and their corresponding labels into lists. Additionally, the images are converted to 3-channel images. Once the images and labels lists are completed, the lists are then converted into NumPy arrays. The images array is defined as the features matrix and the labels array is defined as the target vector. For machine learning, the features array is typically in the format of [n\_samples, n\_features].

# 5. Conclusion

This report documents my journey in learning Matplotlib. Matplotlib is a versatile plotting library in Python. I explored concepts that include fundamental plotting techniques essential for visualizing data and customizing the titles, labels, legends and colors of charts and graphs to enhance clarity and aesthetics. Key concepts covered include creating various types of plots such as line plots, bar charts, scatter plots, histograms, and more. By using real data and previous projects, I was able to explore how to go about creating graphs and charts.

# References

1. Harshit Shankhdhar. (2021). IMDB Dataset of Top 1000 Movies and TV Shows. Retrieved from https://www.kaggle.com/datasets/harshitshankhdhar/IMDB-dataset-of-top-1000-movies-and-tv-shows
2. Johns Hopkins University Center for Systems Science and Engineering (2024). COVID-19 Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University. GitHub. Retrieved Jun 14, 2024, from https://github.com/CSSEGISandData/COVID-19/tree/master/csse\_covid\_19\_data/csse\_covid\_19\_daily\_reports\_us
3. Terminus7. (2024). Pokemon Challenge dataset. Kaggle. Retrieved June 14, 2024, from https://www.kaggle.com/datasets/terminus7/pokemon-challenge?select=pokemon.csv
4. VanderPlas, J. (*2016*).  *Python Data Science Handbook*. O’Reilly Media. Retrieved from https://jakevdp.github.io/PythonDataScienceHandbook/index.html