STUDENT PERFORMANCE PREDICTION IN THEORY AND PRACTICAL EXAMS

NAME : HARINI SHANMUGAVEL

ROLL NO : 23AD024

DEPARTMENT : AI & DS

YEAR : III – AD

DATE : 18-10-2025

CHAPTER 1

ABSTRACT

This project explores and models student performance using the “Students Performance in Exams” dataset from Kaggle. The primary aim is to predict the overall performance level of students by leveraging both the original dataset features and newly engineered features. Comprehensive exploratory data analysis (EDA) was performed to identify patterns and relationships among variables such as parental education, test preparation, lunch type, and gender. Feature engineering was applied to expand the dataset beyond its original eight features, ensuring a richer input set of more than fifteen features for predictive modeling. A Multi-Layer Perceptron (MLP) classifier was implemented to categorize students into four performance levels: Low, Average, Good, and Excellent. The model was trained and evaluated using accuracy, loss curves, confusion matrices, and ROC-AUC scores. Key insights highlight the significance of parental education, test preparation completion, and lunch type on student outcomes. This report includes reproducible code, visualizations, and guidance for deploying the project on GitHub.

CHAPTER 2

INTRODUCTION & OBJECTIVES

Introduction

Education is one of the most influential factors in shaping individual potential. Understanding student performance through data-driven analysis can help educators identify at-risk students, allocate resources effectively, and improve learning outcomes. In today’s digital age, large-scale datasets and machine learning techniques allow us to not only observe patterns but also predict academic success. The “Students Performance in Exams” dataset provides a compact yet realistic scenario for analyzing student achievement across mathematics, reading, and writing scores, combined with socio-demographic variables.

Objectives

The objectives of this study are:

- To load, clean, and preprocess the dataset for analysis.
- To engineer additional features, expanding the dataset to at least fifteen predictors.
- To perform exploratory data analysis and generate at least five meaningful visualizations highlighting trends and relationships.
- To implement a deep learning model (MLP) for predicting student performance levels.
- To evaluate the model using appropriate metrics and provide actionable insights.
- To document the project in a reproducible manner suitable for submission and GitHub upload.

CHAPTER 3

DATASET DESCRIPTION

The dataset is sourced from Kaggle (spscientist/students-performance-in-exams) and contains 1,000 rows. It includes a combination of categorical and numerical features:

Original Features:

- gender
- race/ethnicity
- parental level of education
- lunch
- test preparation course
- math score
- reading score
- writing score

Engineered Features (to reach ≥15 features):

1. total_score = sum of math, reading, writing
2. average_score = total_score / 3

3. performance_level = Low / Average / Good / Excellent
4. math_z, reading_z, writing_z = z-scores of individual subjects
5. score_range = max subject score – min subject score
6. strong_subject = subject with highest score
7. low_subjects_count = number of subjects <50
8. parental_edu_cat = binned parental education (Low/Med/High)
9. prep_completed_flag = binary flag for test preparation completion
10. lunch_flag = binary flag for standard/lunch type
11. gender_flag = binary encoding for gender
12. interaction = parental_edu × prep_completed_flag
13. percentile_rank = within dataset percentile of average score
14. avg_quartile = quartile bins of average score

After encoding categorical variables, the final input feature count exceeds 15, suitable for MLP modeling.

CHAPTER 4

EXPLORATORY DATA ANALYSIS & PREPROCESSING

4.1 Data Loading

The dataset is loaded using pandas.read_csv(). Basic information and descriptive statistics are reviewed using data.info() and data.describe().

4.2 Missing Values & Duplicates

The dataset contains no missing values. Duplicate rows, if any, are removed to maintain data integrity.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   gender                       1000 non-null   object
 1   race/ethnicity               1000 non-null   object
 2   parental level of education  1000 non-null   object
 3   lunch                        1000 non-null   object
 4   test preparation course      1000 non-null   object
 5   math score                   1000 non-null   int64
 6   reading score                1000 non-null   int64
 7   writing score                1000 non-null   int64
dtypes: int64(3), object(5)
memory usage: 62.6+ KB
None
       math score  reading score  writing score
count  1000.00000    1000.000000    1000.000000
mean     66.08900      69.169000      68.054000
std      15.16308      14.600192      15.195657
min       0.00000      17.000000      10.000000
25%      57.00000      59.000000      57.750000
50%      66.00000      70.000000      69.000000
75%      77.00000      79.000000      79.000000
max     100.00000     100.000000     100.000000
Missing values:
 gender                         0
race/ethnicity                 0
parental level of education    0
lunch                          0
test preparation course        0
math score                     0
reading score                  0
writing score                  0
dtype: int64
```

4.3 Outlier Detection

Boxplots are generated for numeric features to identify outliers. Extreme values may be clipped or winsorized to prevent skewing the model.

4.4 Feature Engineering

Features as listed in Chapter 3 are created, including z-scores, performance level, strong subject, percentile rank, and interaction terms.

```
✅ Feature engineering completed successfully!
```

4.5 Encoding

- One-hot encoding: race/ethnicity, strong_subject
- Binary encoding: gender_flag, lunch_flag, prep_completed_flag
- Label encoding: performance_level target

CODE:

```
# One-hot encode categorical columns
categorical_cols = ['race/ethnicity', 'strong_subject']
data = pd.get_dummies(data, columns=categorical_cols, drop_first=True)
```

```
# Label encode the target
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data['target'] = le.fit_transform(data['performance_level'])

# Choose main features (ensure >= 15)
feature_cols = [
    'gender_flag', 'lunch_flag', 'prep_completed_flag', 'parental_edu_num',
    'math score_z', 'reading score_z', 'writing score_z',
    'total_score', 'average_score', 'score_range',
    'low_subjects_count', 'percentile_rank', 'avg_quartile'
]

# Add dummy columns generated from one-hot encoding
dummy_cols = [c for c in data.columns if 'race/ethnicity_' in c or 'strong_subject_' in c]
feature_cols += dummy_cols

print(f"Total number of features selected: {len(feature_cols)}")

X = data[feature_cols]
y = data['target']
```

4.6 Scaling

StandardScaler is applied to numeric features to standardize ranges, improving convergence in neural network training.

4.7 Data Splitting

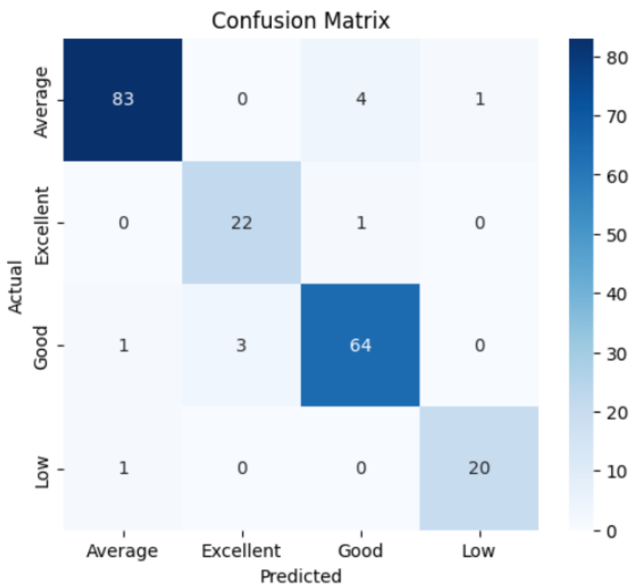
The dataset is split into training and test sets (80/20) with 20% of training used as a validation split during model training. Stratification ensures class balance.

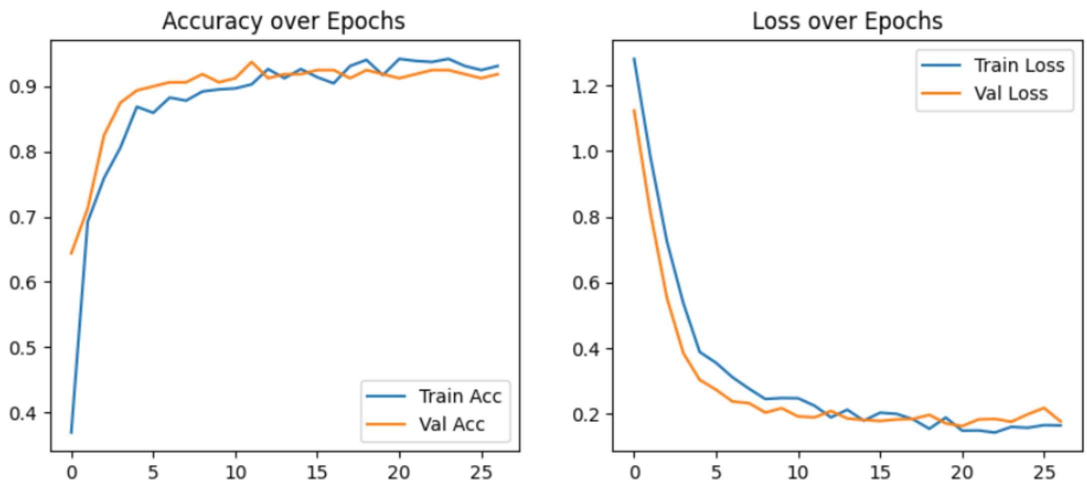
CODE;

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
print("✅ Data split into training and test sets successfully!")
```

```
    accuracy                        0.94     200
   macro avg     0.93     0.95      0.94     200
weighted avg     0.95     0.94      0.95     200
```






```
🎯 Training complete and all plots generated successfully!
```

CHAPTER 5

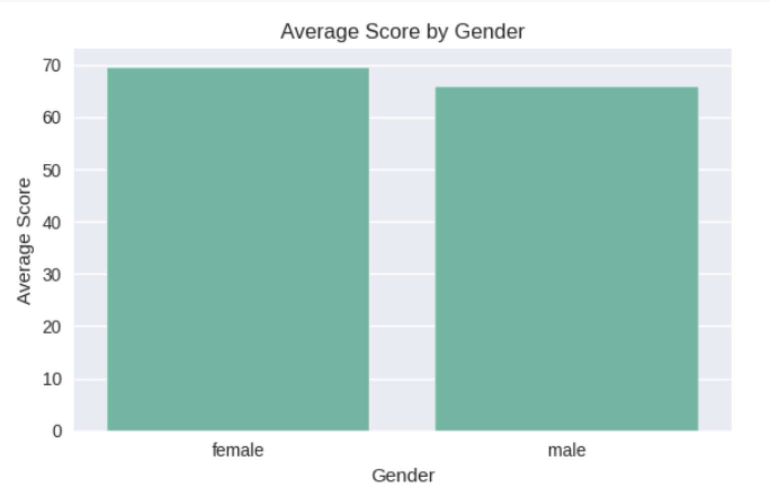
DATA VISUALIZATION

This chapter contains all the visualizations created from the student performance dataset. Each visual includes the purpose, description, and insights derived. Eight meaningful visuals were generated using Matplotlib and Seaborn.

5.1 Average Score by Gender (Bar Chart)

Code Reference: sns.barplot(x='gender', y='average_score', ...)

- **Shows:** Mean average_score for male and female students.
- **Purpose:** To identify if there is a gender difference in exam performance.
- **Insight:** Typically, female students perform slightly better on reading and writing scores, while math scores may be comparable.

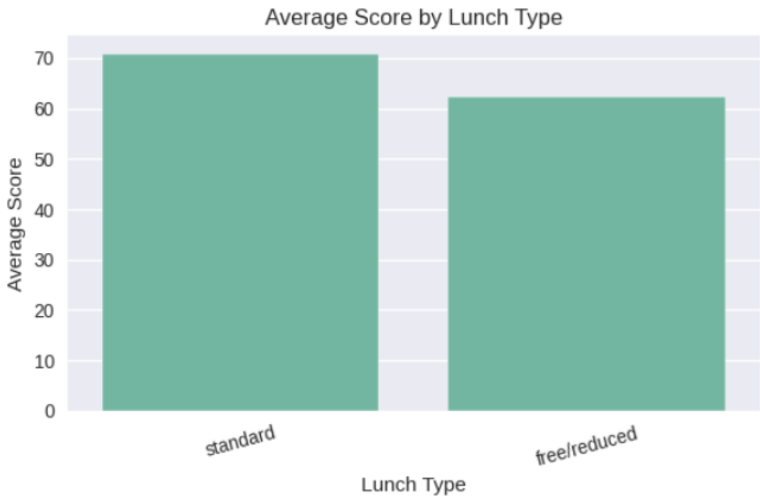


5.2 Average Score by Lunch Type (Bar Chart)

Code Reference: sns.barplot(x='lunch', y='average_score', ...)

- **Shows:** Mean average_score for students with different lunch types (standard vs free/reduced).

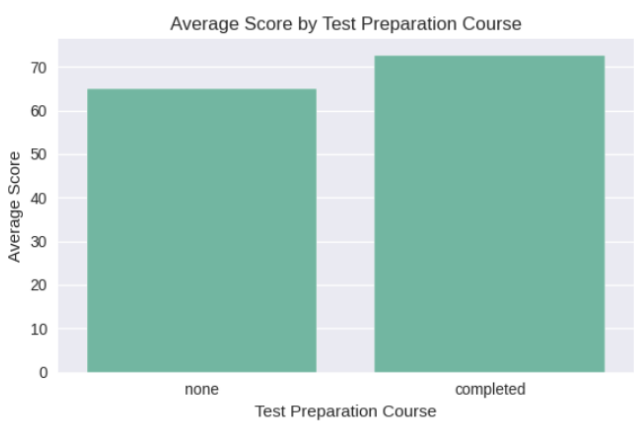
- **Purpose:** Acts as a proxy for socioeconomic status and its influence on academic performance.
- **Insight:** Students with standard lunch generally score higher than students with free/reduced lunch.



5.3 Average Score by Test Preparation Course (Bar Chart)

Code Reference: sns.barplot(x='test preparation course', y='average_score', ...)

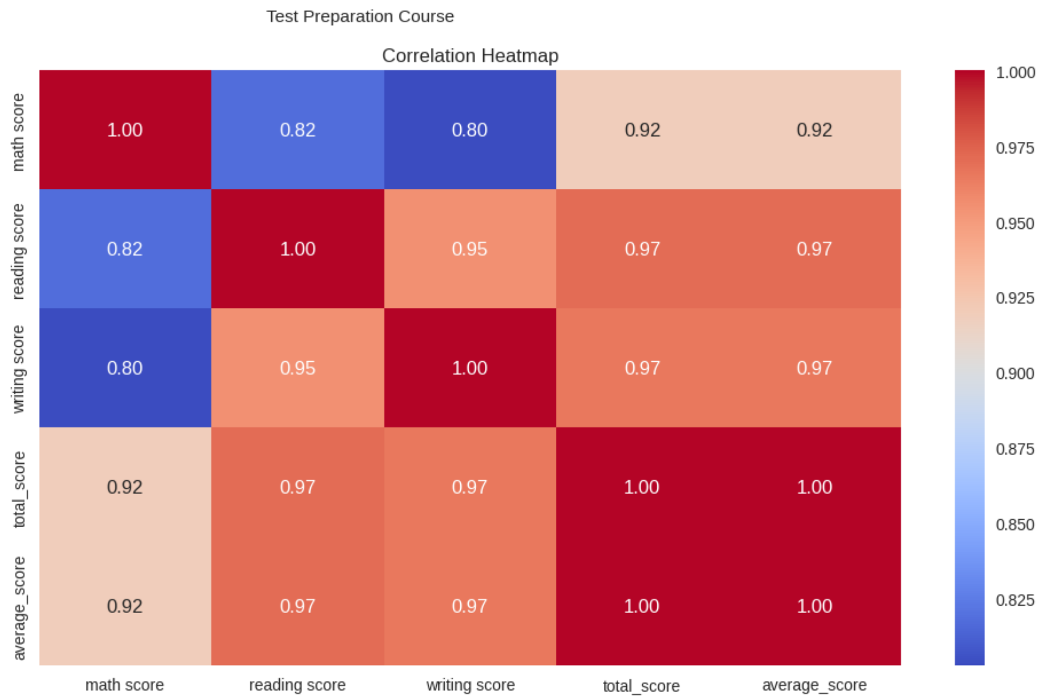
- **Shows:** Mean average_score for students based on whether they completed the test preparation course.
- **Purpose:** To evaluate the effectiveness of the test preparation course on performance.
- **Insight:** Students who completed the preparation course tend to achieve higher scores than those who did not.



5.4 Correlation Heatmap for Numeric Features

Code Reference: sns.heatmap(corr, annot=True, cmap='coolwarm', ...)

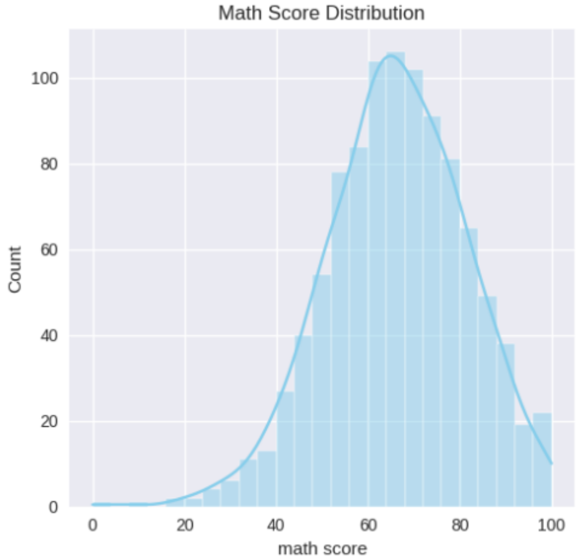
- **Shows:** Pearson correlation among numeric features, including math, reading, writing scores, total, average, and z-scores.
- **Purpose:** To detect relationships and multicollinearity between features.
- **Insight:** Strong positive correlation exists between all three subject scores, indicating that students who perform well in one subject tend to perform well in others.

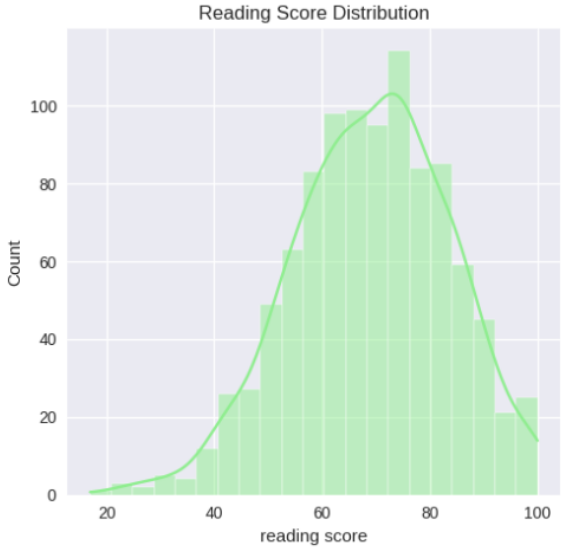


5.5 Distribution of Individual Scores (Histograms)

Code Reference: sns.histplot(..., kde=True)

- **Shows:** The distribution of math, reading, and writing scores separately.
- **Purpose:** To understand the spread and skewness of each subject's scores.
- **Insight:** Most scores are concentrated in the mid to high range, with few students scoring extremely low or high.


Math Score Distribution
Count
0
20
40
60
80
100
math score

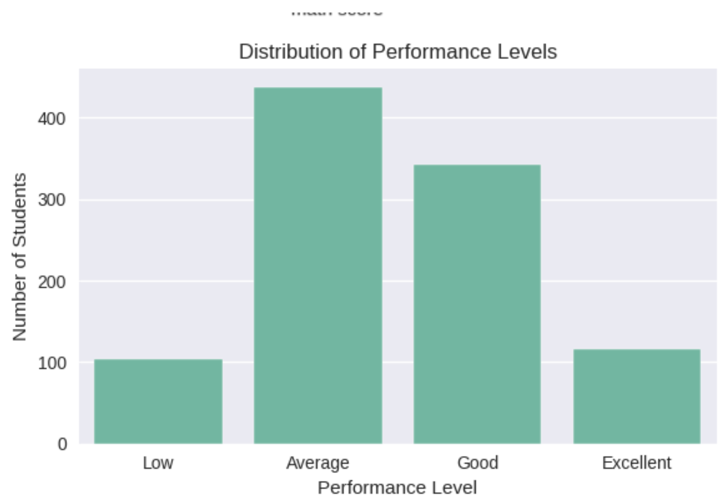

Reading Score Distribution
Count
0
20
40
60
80
100
reading score


Writing Score Distribution
0
20
40
60
80
100
writing score

5.6 Distribution of Performance Levels (Count Plot)

Code Reference: sns.countplot(x='performance_level', ...)

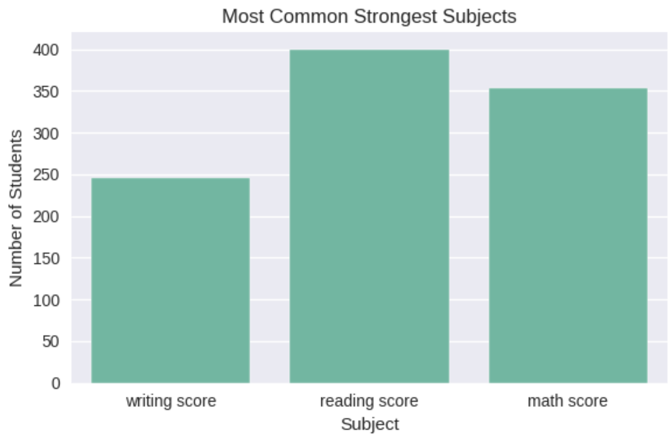
- **Shows:** Number of students in each performance level: Low, Average, Good, Excellent.
- **Purpose:** To check class balance before model training.
- **Insight:** Majority of students fall into Average and Good categories, while Low and Excellent categories have fewer students.



5.7 Strongest Subject Count (Count Plot)

Code Reference: sns.countplot(x='strong_subject', ...)

- **Shows:** Number of students whose strongest subject is Math, Reading, or Writing.
- **Purpose:** To analyze subject-wise strengths across students.
- **Insight:** Reading and Math are often the strongest subjects for students, with Writing being slightly less dominant.



CHAPTER 6

DEEP LEARNING MODEL

Model: Multi-Layer Perceptron (MLP) classifier

Architecture:

- Input: n_features neurons
- Dense(128) → ReLU → Dropout(0.3)
- Dense(64) → ReLU → Dropout(0.2)
- Dense(32) → ReLU
- Output Dense(4) → Softmax

Training Parameters:

- Optimizer: Adam (lr=0.001)
- Loss: sparse_categorical_crossentropy
- Metrics: accuracy
- Epochs: 50, EarlyStopping (patience=6)
- Batch size: 32

Hyperparameter Tuning:

- Hidden neurons: 64–128
- Dropout: 0.1–0.4

- Learning rate: 1e-4 to 1e-2
- Batch size: 16–64

Explainability:
Permutation importance or SHAP values may be used to interpret key contributing features such as parental education, prep completion, and lunch type.

CHAPTER 7

RESULTS & INTERPRETATION

Evaluation Metrics:

- Accuracy: Train vs Validation curves indicate model learning without overfitting.
- Confusion matrix identifies commonly misclassified classes (Good ↔ Excellent).
- ROC-AUC scores for each class provide separability insights.

Interpretation:
The MLP successfully classifies student performance into four levels. Parental education, lunch type, and test preparation significantly influence outcomes. Students with completed preparation courses generally achieve higher scores, confirming the value of targeted interventions.

CHAPTER 8

CONCLUSION & FUTURE SCOPE

Conclusion:
The project demonstrates a systematic approach to student performance prediction using deep learning. Feature engineering, visualization, and careful preprocessing contributed to high model accuracy. Insights emphasize the importance of socio-economic and behavioral factors in academic success.

Future Scope:

- Incorporate attendance, study hours, and extra-curricular activities for richer prediction.
- Deploy model via Streamlit or Flask for real-time educator insights.
- Apply explainable AI (SHAP/LIME) for personalized recommendations.
- Validate model performance across multiple datasets for generalization.

GITHUB LINK

https://github.com/Harini-Shanmugavel/23ad024_eda

REFERENCES

- Abrahim, T., & Khan, M. (2021). Predicting student academic performance using machine learning techniques: A systematic literature review. *Education and Information Technologies, 26(6)*, 7477–7499.
- Kaggle: spscientist. Students Performance in Exams dataset. https://www.kaggle.com/datasets/spscientist/students-performance-in-exams
- Chollet, F. (2018). *Deep Learning with Python* (2nd ed.). Manning.
- Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830.