MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

NATIONAL TECHNICAL UNIVERSITY
"KHARKIV POLYTECHNIC INSTITUTE"

Svitlana Gavrylenko

MACHINE LEARNING

The workshops guide for the students of
123 – "Computer Engineering"
for full-time and distance education

Approved by
the university editorial and publishing council,
protocol No. 1 of February 15, 2024, p.178.

Kharkiv
NTU "KhPi"
2024

UDK 004.85

Reviewer:

Ihor YAKOVENKO, Dr. Ph. and Math. Sciences, professor of the Information Systems department, National Technical University "Kharkiv polytechnic institute".

Svitlana Gavrylenko
Machine learning: The study guide. – Kh.: NTU «KhPI», 2024, 26 p.

The workshops guide contains the necessary material for performing workshops: options for tasks, examples of program texts and report.

Intended for students of computer majors at higher educational institutions

Table of Contents

Introduction

Machine learning (ML) is a type of artificial intelligence (AI) that allows computers to learn without being explicitly programmed. It's like training a child to recognize a dog from pictures without explicitly telling them what features define a dog. Instead, you show them many pictures of dogs, and they eventually learn to identify dogs on their own.

Similarly, in machine learning, algorithms are trained on large amounts of data, and they learn to identify patterns and relationships in that data. These patterns can then be used to make predictions or decisions on new data. Here are some examples of how machine learning is used in our everyday lives:

– Recommender systems: When you see recommendations for products or movies you might like on Amazon or Netflix, that's thanks to machine learning algorithms that have analyzed your past purchases or viewing habits.

– Spam filtering: Machine learning algorithms can identify spam emails with high accuracy, keeping your inbox clean.

– Fraud detection: Banks and credit card companies use machine learning to detect fraudulent transactions in real-time.

– Image recognition: Facial recognition software, self-driving cars, and medical imaging analysis all rely on machine learning to make sense of visual data.

– Natural language processing: Machine learning powers virtual assistants like Siri and Alexa, as well as chatbots and automatic translation tools.

Besides, Machine learning (ML) plays a crucial role in modern intrusion detection systems (IDS) by enabling them to analyze large amounts of data and identify patterns indicative of malicious activity. Traditional IDS relied on signature-based detection, which involved matching network traffic against a database of known attack signatures. However, this approach was often ineffective against zero-day attacks and other novel threats.

Here's how machine learning empowers IDS:

1. Anomaly Detection:

ML algorithms can analyze network traffic, system logs, and other data sources to identify deviations from normal behavior. This allows them to detect even unknown threats that haven't been seen before.

2. Classification:

ML algorithms can be trained on labeled datasets of intrusion and normal activity. This enables them to classify new data points as either malicious or benign with high accuracy.

3. Predictive Analytics:

ML models can be used to predict future attacks based on historical data and current trends. This allows IDS to take proactive measures to prevent security breaches.

Specific ML Techniques used in IDS Supervised Learning, Unsupervised Learning, Reinforcement Learning. Supervised Learning Algorithms like decision trees, support vector machines, and neural networks are trained on data that has been labeled with the desired output. For example, an algorithm might be trained on a dataset of labeled images, where each image is labeled with the object it contains (e.g., cat, dog, car). Once trained, the algorithm can then be used to predict the labels of new, unseen images.

Unsupervised Learning Techniques like clustering and anomaly detection algorithms can identify patterns and outliers in data without relying on pre-labeled examples. For example, an unsupervised learning algorithm might be used to cluster customer data into different groups based on their purchase history

Reinforcement Learning approach is still in its early stages for IDS, but it has the potential to train IDS to adapt to new threats and make automated decisions in real-time.

Benefits of using ML in IDS:

– Improved accuracy. ML can detect a wider range of threats, including zero-day attacks, compared to traditional signature-based methods.

– Reduced false positives. ML algorithms can better distinguish between normal and malicious activity, leading to fewer false alarms.

– Scalability. ML models can handle large amounts of data efficiently, making them suitable for complex network environments.

– Adaptability. ML algorithms can learn and adapt to new threats over time, improving the overall security posture.

Challenges of using ML in IDS:

– Data quality. The effectiveness of ML models depends heavily on the quality and quantity of training data.

– False positives. Even with advanced algorithms, some false positives are inevitable, which can waste security resources.

– Evasion techniques. Attackers can develop techniques to bypass ML-based IDS, requiring constant adaptation and improvement of the models.

– Overall, machine learning has revolutionized the field of intrusion detection by providing powerful tools for identifying and preventing cyberattacks. While challenges remain, the ongoing advancements in ML research promise even more effective and adaptable IDS solutions in the future.

Thus, Machine learning is a powerful tool that can be used to solve a wide range of problems. It is important to use machine learning responsibly and ethically, and to be aware of the potential biases that can be encoded in data and algorithms.

This manual contains 8 practical tasks covering the main sections of the taught course "Machine Learning"

1. WORKSHOP №1. WORKING WITH DATAFRAME. THE SIMPLEST OPERATIONS

The purpose of the work: gaining skills in working with DataFrame.

Progress:

To prepare working with DataFrame you should use materials from lectures.

It is recommended to perform laboratory work in the Jupyter are hosted by Colab. https://colab.research.google.com/

1. Download the **1_workshop_DataSet.xlsx** file to DataFrame using read_excel() from pandas.

When loading, take into account the fact that it is necessary to form the correct Header so that the names of the features correspond to the features themselves and index_col. Print DataFrame.

2. Calculate the Total score for every subject as a sum of Test1 and Test 2 and add two new columns to DataFrame: Total score for subject 1 and Total score for subject 2. Print DataFrame.

3 Used below ASSESSMENT AND GRADING add two new columns to

Table 1 – Assessment and grading

ASSESSMENT AND GRADING			
Ranges of points corresponding to grades	**Total score (points) for all types of learning activities**	**ECTS grading scale**	**The national grading scale**
	90-100	A	excellent
	82-89	B	good
	74-81	C	
	64-73	D	satisfactory
	60-63	E	
	35-59	FX	Unsatisfactory (with the exam retake option)
	0-34	F	Unsatisfactory (with mandatory repetition of the course)

DataFrame for every subject: ECTS grading scale, The national grading scale. Print DataFrame DataFrame (Tabl.1).

Use df.info() to be sure that all columns have datatype you need.

4. Create a new_ DataFrame with columns: Last name, first name, Group, The national grading scale for subject 1 and subject 2. The national grading scale you can find in Tabl. 2. Enter new name for two subjects and rename the columns: subject 1 and subject 2 . Print DataFrame. Save the new_ DataFrame as a file the **2_workshop_DataSet.xlsx** file. You will use this file as input data for Workshop#2.

5. Using below Table recode the student mark. Print DataFrame. Save the new_ DataFrame as a file the Lab_2_DataSet.xlsx file.

Table 2 – The national grading scale

The national grading scale	Marks
excellent	5
good	4
satisfactory	3
Unsatisfactory (with the exam retake option)	2
Unsatisfactory (with mandatory repetition of the course)	1

5. Apply unique() method to each column of the new_ DataFrame to *identify unique values.* Save results to different lists. Be careful with list names, they have to be defined so that it's clear what they contain.

6. Apply value_counts() method to each column of the new_ DataFrame to count *unique values*. Save results to dataframe, specify columns names to make clear what it consists of.

7. Calculate how many students with only excellent and good marks are in *all groups*. Use df.groupby() method (https://realpython.com/pandas-groupby/) and, for example, .count() or size() method. Save results to DataFrame, pay attention to the name of columns.

8. Calculate how many students with only Unsatisfactory marks are *separate in every Group*. Use df.groupby() method (https://realpython.com/pandas-groupby/) and, for example, .count() or size() method. Save results to DataFrame..

9. Calculate how many students got in specific mark positions (excellent , good, etc). Use df.groupby() method (https://habr.com/ru/post/501214/ or https://realpython.com/pandas-groupby/) and, for example, .count() or size() method. Save results to DataFrame.

10. Find group without any Unsatisfactory marks using groupby() and count(). Save result to dataframe.

11. * **Additional task**. Add extra columns with Start Time and Finish Time of the both tests if format: 2023-12-04 00:00:00 . Calculate the average time of both tests of each student (period between Start Time and Finish Date). Be careful when using arithmetic operations to datetime64[ns] objects (dt.days should help you to realize it). Save results to the DataFrame, sort it and find 5 students with the shortest average time of test and 5 students with the shortest one.

Reporting

The report is a **power point presentation** with slides that should contain SLIDE NAME, maybe little description and charts with particular results (no more than 10 slides). Also, please, add a file with the program code in *.ipynb format. Do not forget to sign the presentation on first slide ☺. **The file name must contain your last name and workshop number.**

2. WORKSHOP №2. WORKING WITH DATAFRAME. THE SIMPLEST VISUALIZATIONS

The purpose of the work: gaining skills in different visualization approaches.

Progress:

To prepare working with DataFrames you should use materials from lectures and finish workshop #1. For visualization you can use different data visualization libraries for Python: matplotlib, seaborn, pandas, maybe another one - up to you ☺

Creating the plots, use theirs different parameters : Font Properties, Width, Color, Size , Grid Lines, ColorMap by the way you want to. Create Multiple Plots or 3D-Plot.

1. Download the 2_workshop_DataSet.xlsx to DataFrame using read_excel() from pandas. File 2_workshop_DataSet.xlsx is the result of creating a new DataFrame in Workshop#1 and save it (see step 4 # in Workshop 1). Pay attention to data types you got.

2 Apply unique() method to each column of the new_ DataFrame to *identify unique values.* (step 5 in workshop #1). Save results to different lists. Choose DataFrames with meaningful content and create any chart you want to present results (ex. bar chart). Pay attention to sorting of data on the charts (from max to min values, by alphabetical order etc).

3 Calculate how many students with only excellent and good marks are in *all groups*. (step 7 in workshop#1). Build simple charts to present a result with percentage of marks. It can be bar chart, or pie chart, or both of them.

4. Calculate how many students with only Unsatisfactory marks are *separate in every Group*. Use (step8 in workshop #1). Visualize it by the way you want to.

5. Calculate how many students got in specific mark positions (excellent , good, etc). (step 9 in workshop #1). Created stacked bar chart (1 column with values of).

6. Calculate how many students with unique name in *each group* (for example ‘Дмитро’ has the same mark. Save results to DataFrame and created stacked bar chart with number of marks in each groups. (1 column with values of).

Reporting

The report is a **power point presentation** with slides that should contain SLIDE NAME, maybe little description and charts with particular results (no more than 10 slides). Also, please, add a file with the program code in *.ipynb format. Do not forget to sign the presentation on first slide ☺. **The file name must contain your last name and workshop number.**

3. WORKSHOP №3. DATA PRE-PROCESSING

The purpose of the work: gaining skills in different Data pre-processing function.

Progress:

To prepare working with DataFrames you should use materials from lectures and finish workshop #2. For data pre-processing you can use different data pre-processing libraries for Python: Pandas, NumPy, StandardScaler, LabelEncoder, OneHotEncode, train_test_split, one - up to you ☺

1. Import libs and upload any DataFrame from Kaggle, Internet or Generate a random DataFrame for classification problem with your own setting (use function *make_classification*()).
2. Print a concise summary of a DataFrame (use function *df.info(), where df* is a name of DataFrame).
3. Estimation the number of missing values. Create a Series that displays the total count of missing values per column. Create any count plot.
4. Remove the columns with more than 23% of gaps or fill any missing data. Revisit the DataFrame to check result (use function *df.dropna*() and *df.fillna*())
5. Handle conflicting cases in DataFrame.
6. Remove unnecessary or duplicated features (use *df.duplicated*()). Justify the decision to remove the features
7. Convert categorical string features to numeral values (use differences Label Encoding functions).
8. Remove anomaly data(use functions: *Isolation Forest, Minimum Covariance Determinant, Local Outlier Factor, One-Class SVM* or another one - up to you).
9. Create Correlations heatmap. **Additional step**. Correlations feature selection. Drop data with correlation more than 95%.

10. Create a Train and a Test Split of DataFrame.
11. Perform Data Normalization.
12. **Additional step**. Investigate techniques to Handle Imbalanced Data.

Reporting

The report is a **power point presentation** with slides that should contain SLIDE NAME, maybe little description and charts with particular results (no more than 10 slides). Also, please, add a file with the program code in **.ipynb* format and *input data (*.cvs* or **. xlsx* format). Do not forget *to sign* the presentation on first slide ☺. **The file name must contain your last name and workshop number.**

4. WORKSHOP №4. DATA CLASSIFICATION

The purpose of the work: gaining skills in different classification method.

Progress:

To prepare working with DataFrames you should use materials from lectures #5,6 and finish workshop #3. For **Data classification** you can use the **Sklearn** libraries, which contains a range of useful algorithms that can easily be implemented and setting for the purposes of classification and other machine learning tasks You can also use different libraries for Python up to you. ☺

1. Load the DataFrame, which you used in the workshop #3 or Generate a random DataFrame for classification problem with your own setting (use function *make_classification*()).
2. Generate descriptive statistics. Group DataFrame using a Series of columns up to you. Create two count plots up to you.
3. Choose a Classification method (*K-Nearest Neighbors, Support Vector Machines, Decision Tree Classifiers, Naive Bayes, Linear Discriminant Analysis, Logistic Regression and so on*) and perform necessary data-preparation for this model. Justify the need for choosing preprocessing steps.
4. Separate the dataset into feature columns and target column. Make their visualization. Perform data preprocessing. Create the testing and training splits (*import train_test_split*).
5. Perform an Exploratory Data Analysis (*import pyplot, import seaborn*).
6. Analyze the model settings. Create classification models **with different settings**. For each model with different settings:

– Estimate the model on Testing dataset.

– Make a classification report *(import ConfusionMatrixDisplay, roc_curve, auc, classification_report,accuracy_score, confusion_matrix*).Visualize a confusion matrix. Plot ROC curve.

7. **Additional step**. Compare the quality of the model at different settings. Present the result of comparing as a barplot. Print the value of the criteria upper the every barplot.

Reporting

The report is a **power point presentation** with slides that should contain SLIDE NAME, maybe little description and charts with particular results (no more than 10 slides). Also, please, add a file with the program code in *.ipynb format and *input data* (cvs or xlsx format) . Do not forget to sign the presentation on first slide. The file name must contain your last name and workshop number. ☺

5. WORKSHOP №5. ENSEMBLE CLASSIFIER. BEGGING. RANDOM FOREST CLASSIFIER

The purpose of the work: gaining skills in **Random Forest** classification method.

Progress:

To prepare working with DataFrames you should use materials from lectures 8 and finish workshop #4. For **Data classification** you can use the **Sklearn** libraries, which contain a range of useful algorithms that can easily be implemented and setting for the purposes of classification and other machine learning tasks *(import RandomForestClassifier, import accuracy_score, confusion_matrix, precision_score, recall_score, ConfusionMatrixDisplay).* You can also use different libraries for Python up to you. ☺

8. Load the DataFrame, that you used in the workshop #3, 4 or generate a random DataFrame for classification problem with your own setting (use function *make_classification*()).
9. If necessary, make pre-processing data.
10. Separate the dataset into feature columns and target column. Create the testing and training splits (*import train_test_split*).
11. Fit a **Random Forest** model with *default setting*. Estimate the model on Validation (Testing) dataset. Make a classification report *(import ConfusionMatrixDisplay, roc_curve, auc, classification_report,accuracy_score, confusion_matrix*).Visualize a confusion matrix and ROC curve.
12. Investigate the dependence of the RF classification quality on the number of trees in the forest (**n_estimators)**. Make a result visualization.
13. Investigate the dependence of the RF classification quality on the type of **criterion***("gini", "entropy", "log_loss").* Make a result visualization.

14. Explore the dependence of the RF classification quality on the number of depths of the tree (**max_depth)**. Make a result visualization.

15. Investigate the dependence of the RF classification quality on the minimum number of samples required to split an internal node (**min_samples_split)**. Make result visualization.

16. Investigate the dependence of the RF classification quality on the minimum number of samples required for a leaf node (**min_samples_leaf)**. Make a result visualization

17. Investigate the dependence of the RF classification quality on the number of features to consider when looking for the best split (**max_features).** Make a result visualization.

18. **Additional step**. Create new Begging Ensemble model, based on another base Classifier (KNN, SVM and etc). Make a classification report. Present the result of comparing with Random Forest model. Visualize it by the way you want to.

Reporting

The report is a **power point presentation** with slides that should contain SLIDE NAME, maybe little description and charts with particular results (no more than 10 slides). Also, please, add a file with the program code in *.ipynb format and *input data* (cvs or xlsx format) . Do not forget to sign the presentation on first slide. **The file name must contain your last name and workshop number**. ☺

6. WORKSHOP №6. ENSEMBLE CLASSIFIER. BOOSTING

The purpose of the work: gaining skills in **Boosting** classification methods.

Progress:

To prepare working with DataFrames you should use materials from lectures 8,9 and finish workshop #5. For **Data classification** you can use the **Sklearn** libraries, which contain a range of useful algorithms that can easily be implemented and setting for the purposes of classification and other machine learning tasks *(import* `AdaBoostClassifier`, *import accuracy_score, confusion_matrix, precision_score, recall_score, ConfusionMatrixDisplay).* You can also use different libraries for Python up to you. ☺

1. Load the **DataFrame UNSW_NB15_training-set.csv**.
2. Make pre-processing data. Investigate techniques to Handle Imbalanced Data (see methods: **SMOTEENN,** SVMSMOTE, BorderlineSMOTE, ADASYN, SMOTE, KMeansSMOTE). Perform data balancing.
3. Separate the dataset into feature columns and target column. Create the testing and training splits (*import train_test_split*).
4. Fit a *AdaBoostClassifier* model with *default setting*. Estimate the model on Validation (Testing) dataset. Make a classification report *(import ConfusionMatrixDisplay, roc_curve, auc, classification_report,accuracy_score, confusion_matrix*).Visualize a confusion matrix and ROC curve.
5. Fit a *GradientBoostingClassifier()*model with *default setting*. Estimate the model on Validation (Testing) dataset. Make a classification report *(import ConfusionMatrixDisplay, roc_curve, auc, classification_report,accuracy_score, confusion_matrix*).Visualize a confusion matrix and ROC curve.
6. Fit a *HistGradientBoostingClassifier* model with *default setting*. Estimate the model on Validation (Testing) dataset. Make a classification report *(import ConfusionMatrixDisplay, roc_curve, auc,*

classification_report,accuracy_score, confusion_matrix).Visualize a confusion matrix and ROC curve.

7. Compare the quality of three models. Make a result visualization.

The report is a **power point presentation** with slides that should contain SLIDE NAME, maybe little description and charts with particular results (no more than 10 slides). Also, please, add a file with the program code in *.ipynb format and *input data* (cvs or xlsx format) . Do not forget to sign the presentation on first slide. **The file name must contain your last name and workshop number**. ☺

7. WORKSHOP №7. BUILDING A PREDICTION MODEL IN PYTHON

The purpose of the work: gaining skills in **Building a Prediction Model** .

Progress:

To prepare working with DataFrames you should use materials from lectures 10 and finish workshop #6. For **Building a Prediction Model** you can use the **Sklearn** libraries, which contain a range of useful algorithms *(import* LinearRegression, SVR, RandomForestRegressor, GradientBoostingRegresso and others. Import the metrics for evaluating models*).* You can also use different libraries for Python up to you. ☺

1. Load a DataFrame for prediction problem or generate a random DataFrame for regression problem with your own setting (use *sklearn.datasets.make_regression()* function).

2. Make a Data manipulationцу .

2.1. Print 0 to 10 rows and 0 to 5 columns from dataset.

2.2. Analyze DataFrame. Remove some columns up to you (use *drop ()* function).

2.2. Check the data types of the columns.

3. Make Data Visualization. Create a simple line chart.

4. Make pre-processing data (see workshop #3).

– Remove the columns with more than 20% of gaps or fill any missing data.

– Remove unnecessary or duplicated features (use *df.duplicated*()). Justify the decision to remove the features.

– Convert categorical data to numeral values.

– Drop data with correlation more than 90%.

– Separate the dataset into feature columns and target column.

– Create Training and Testing Data.

– Perform Data Normalization.

5. Fitting **Prediction Model (** up to you) with your own setting. Make result visualization up to you (for example: pie plot, scatter plot, histogram, bar plot) .

6. Estimate the model performance. Use Mean absolute error (MAE),Root mean squared error (RMSE),Relative absolute error (RAE), Relative squared error (RSE),the coefficient of determination, often called R^2.

7. Make visualization for the model performance.

Reporting.

The report is a **power point presentation** with slides that should contain SLIDE NAME, maybe little description and charts with particular results (no more than 10 slides). Also, please, add a file with the program code in *.ipynb format and *input data* (cvs or xlsx format) . Do not forget to sign the presentation on first slide. **The file name must contain your last name and workshop number**. ☺

8. WORKSHOP №8. CLUSTERING METHODS

The purpose of the work: gaining skills in **Clustering methods**.

Progress:

To prepare working with DataFrames you should use materials from lectures 11 and finish workshop #6. For **Data Clustering** you can use the **Sklearn** libraries, which contain a range of useful algorithms that can easily be implemented and setting for the purposes of classification and other machine learning tasks *(import DBSCAN, import KMeans, import accuracy_score).* You can also use different libraries for Python up to you. ☺

1. Load the **DataFrame UNSW_NB15_training-set.csv** or generate a random DataFrame for clustering problem with your own setting (use method make_blobs ()). / Analyze the Dataset. Describe the features of Dataset (sense of each feature)

2. Make pre-processing data (see workshop #3).

– Remove the columns with more than 20% of gaps or fill any missing data.

– Remove unnecessary or duplicated features (use df.duplicated()). Justify the decision to remove the features.

– Convert categorical data to numeral values.

– Drop data with correlation more than 90%.

– Separate the dataset into feature columns and target column

– Perform Data Normalization.

3. Choose the optimal number of clusters. The elbow method is use (import cluster as sk_cluster, import distance as sci_distance).Create a plot: Dependence the Number of clusters on Average within-cluster sum of squares. Define an optimal number of clusters *k*.

4. Fitting KMeans model with different number of clusters *k*. For each cluster, count the members. Make result visualization up to you (for example: pie plot, scatter plot, histogram, bar plot) .

5. Fitting DBSCAN model with different setting. Make result visualization up to you.

Use techniques: Silhouette_scor, Calinski_harabasz score, Davies_bouldin_score, Dendrogram, Bayesian information criterion select optimal number of clusters (import silhouette_score, calinski_harabasz_scor, davies_bouldin_score,scipy.cluster.hierarchy, GaussianMixture). Justify your decision. Make visualization for each method.

Reporting.

The report is a **power point presentation** with slides that should contain SLIDE NAME, maybe little description and charts with particular results (no more than 10 slides). Also, please, add a file with the program code in *.ipynb format and *input data* (cvs or xlsx format) . Do not forget to sign the presentation on first slide. **The file name must contain your last name and workshop number**. ☺

9 EXAMPLE OF REPORT

NATIONAL TECHNICAL UNIVERSITY
«KHARKIV POLYTECHNIC INSTITUTE»
Department of Computer Engineering and Programming

MACHINE LEARNING
Workshop 4

St. Group KH-M924 i.e
First Name, Second Name
Data: March 17, 2024

1. Load the DataFrame

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_csv("train.csv")
df.head(100)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
95	96	0	3	Shorney, Mr. Charles Joseph	male	NaN	0	0	374910	8.0500	NaN	S
96	97	0	1	Goldschmidt, Mr. George B	male	71.0	0	0	PC 17754	34.6542	A5	C
97	98	1	1	Greenfield, Mr. William Bertram	male	23.0	0	1	PC 17759	63.3583	D10 D12	C
98	99	1	2	Doling, Mrs. John T (Ada Julia Bone)	female	34.0	0	1	231919	23.0000	NaN	S
99	100	0	2	Kantor, Mr. Sinai	male	34.0	1	0	244367	26.0000	NaN	S

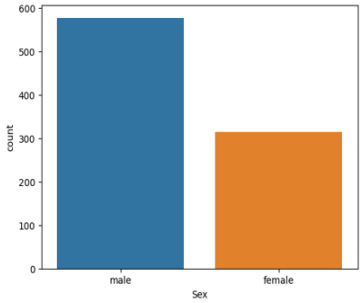
100 rows × 12 columns

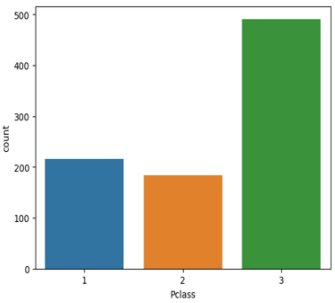
2. Generate descriptive statistics. Group DataFrame using a Series of columns up to you. Create two count plots up to you.

```
# Generate descriptive statistics
print(df.describe())

# Group DataFrame using a Series of columns (e.g., 'Sex', 'Pclass')
grouped_df = df.groupby(['Sex', 'Pclass'])

# Create two count plots
sns.countplot(x='Sex', data=df)
plt.show()

sns.countplot(x='Pclass', data=df)
plt.show()
```

	PassengerId	Survived	Pclass	Age	SibSp
count	891.000000	891.000000	891.000000	714.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008
std	257.353842	0.486592	0.836071	14.526497	1.102743
min	1.000000	0.000000	1.000000	0.420000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000
50%	446.000000	0.000000	3.000000	28.000000	0.000000
75%	668.500000	1.000000	3.000000	38.000000	1.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000

	Parch	Fare
count	891.000000	891.000000
mean	0.381594	32.204208
std	0.806057	49.693429
min	0.000000	0.000000
25%	0.000000	7.910400
50%	0.000000	14.454200
75%	0.000000	31.000000
max	6.000000	512.329200



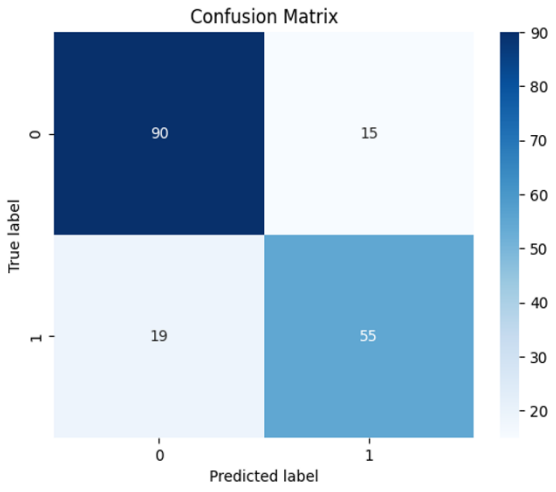


3. Choose a Classification method (K-Nearest Neighbors, Support Vector Machines, Decision Tree Classifiers, Naive Bayes, Linear Discriminant Analysis, Logistic Regression and so on) and perform necessary data-preparation for this model. Justify the need for choosing preprocessing steps.

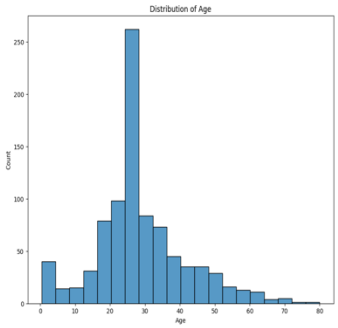
```
[ ] from sklearn.model_selection import train_test_split
    from sklearn.preprocessing import StandardScaler
    from sklearn.linear_model import LogisticRegression

# drop unnecessary columns
df = df.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1)

# fill missing values with median or mode
df['Age'] = df['Age'].fillna(df['Age'].median())
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])

# convert categorical variables to numeric
df['Sex'] = df['Sex'].map({'male': 0, 'female': 1})
df = pd.get_dummies(df, columns=['Embarked'], prefix='Embarked')

# split data into training and testing sets
X = df.drop('Survived', axis=1)
y = df['Survived']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# standardize features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# fit logistic regression model
lr = LogisticRegression()
lr.fit(X_train, y_train)

# evaluate model
train_acc = lr.score(X_train, y_train)
test_acc = lr.score(X_test, y_test)
print(f'Training accuracy: {train_acc:.4f}')
print(f'Testing accuracy: {test_acc:.4f}')
```

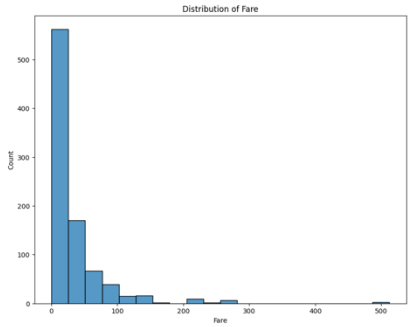
```
from sklearn.metrics import confusion_matrix

# predict on test set
y_pred = lr.predict(X_test)

# calculate confusion matrix
cm = confusion_matrix(y_test, y_pred)

# plot confusion matrix
sns.heatmap(cm, annot=True, fmt='g', cmap='Blues')
plt.xlabel('Predicted label')
plt.ylabel('True label')
plt.title('Confusion Matrix')
plt.show()
```

```
Training accuracy: 0.8006
Testing accuracy: 0.8101
```



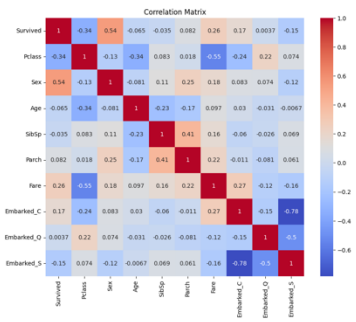
4. Separate the dataset into feature columns and target column. Create the testing and training splits (import train_test_split).

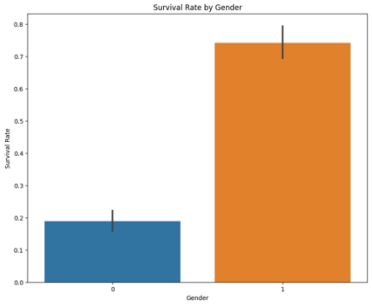


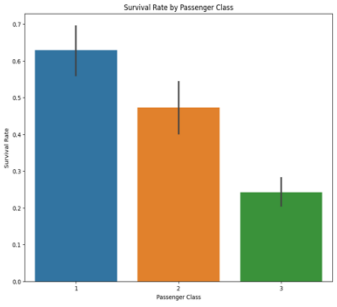


Saparate them and count

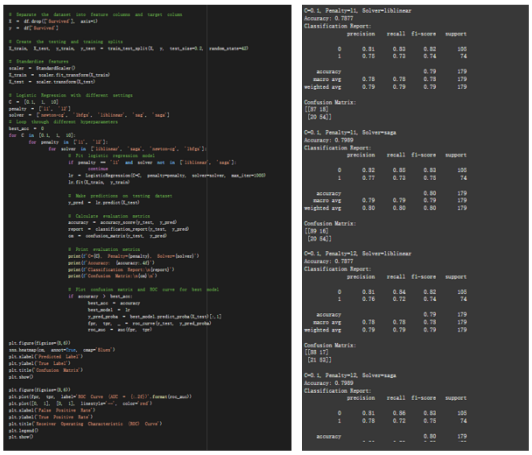
4. Perform data preprocessing. Create the testing and training splits (import train_test_split).

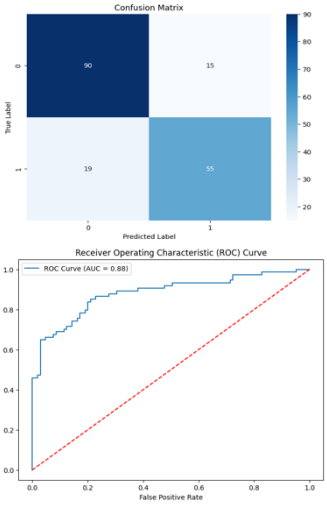




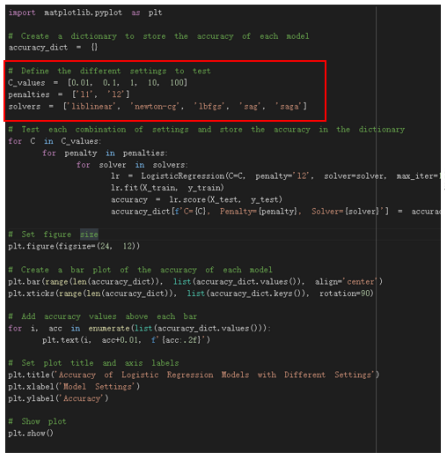


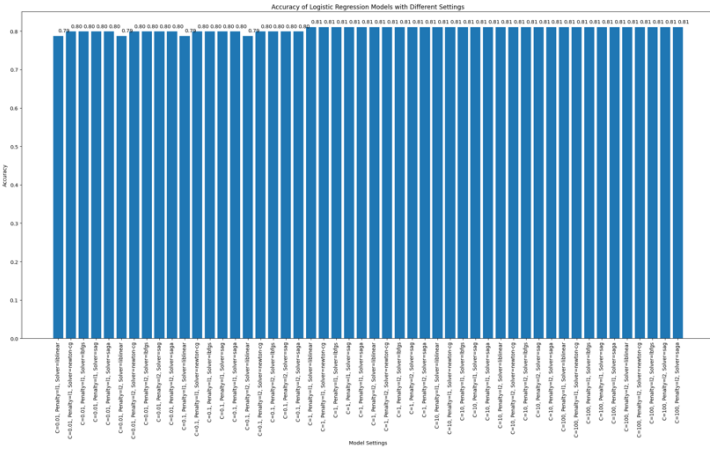
6.Analyze the model settings. Create classification models with different settings. For each model with different settings:
(1)Estimate the model on Testing dataset.
(2)Make a classification report (import ConfusionMatrixDisplay, roc_curve, auc, classification_report,accuracy_score, confusion_matrix)Visualize a confusion matrix. Plot ROC curve.





7. Additional step. Compare the quality of the model at different settings. Present the result of comparing as a barplot. Print the value of the criteria upper the every barplot.





Make some setting

Referenses

1. Halterman R.L. Fundamentals of Python Programming. – Southern Adventist University. 2019, 658 p. https://folk.ntnu.no/sverrsti/INGG1001-H2019/pythonbook_20191015.pdf

2. C. Müller and Sarah Guido. Introduction to Machine Learning with Python, USA, 2019, 368 p. https://www.nrigroupindia.com/e-book/Introduction%20to%20Machine%20Learning%20with%20Python%20(%20PDFDrive.com%20)-min.pdf

3. Alex Smola and S.V.N. Vishwanathan, Introduction to Machine Learning, Cambridge university press, 2018, 225 p. https://alex.smola.org/drafts/thebook.pdf

4. Shai Shalev-Shwartz and Shai Ben-David, Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press, 2014, 449 p. https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf

Svitlana GAVRYLENKO

The workshops guide to the performance of Calculation assignment instructions for the students of 123 – "Computer Engineering" for full-time and distance education

The work was recommended for publication by N.I. Zapolovskiy

In the author's edition

Independent electronic publication