

Business Intelligence & Machine Learning

By Regeneration & Piraeus Bank



The Essentials of Part I



OLTP DATABASE
MIGRATION.



DATA WAREHOUSE
DESIGN.



ETL PROCESS
IMPLEMENTATION.



POWER BI
INSIGHTS.

```
Date_Key INT NOT NULL,  
Invoice_Date DATE NOT NULL,  
Quantity SMALLINT NOT NULL,  
Price FLOAT NOT NULL,  
Total FLOAT NOT NULL,  
Extended_Price_Amount FLOAT NOT NULL,  
Discount_Amount FLOAT DEFAULT 0 NOT NULL);
```

```
DECLARE @CurrentDate DATETIME = '2005-01-01' --Starting value of Date Range  
DECLARE @EndDate DATETIME = '2005-01-31' --End Value of Date Range
```

SQL Scripts

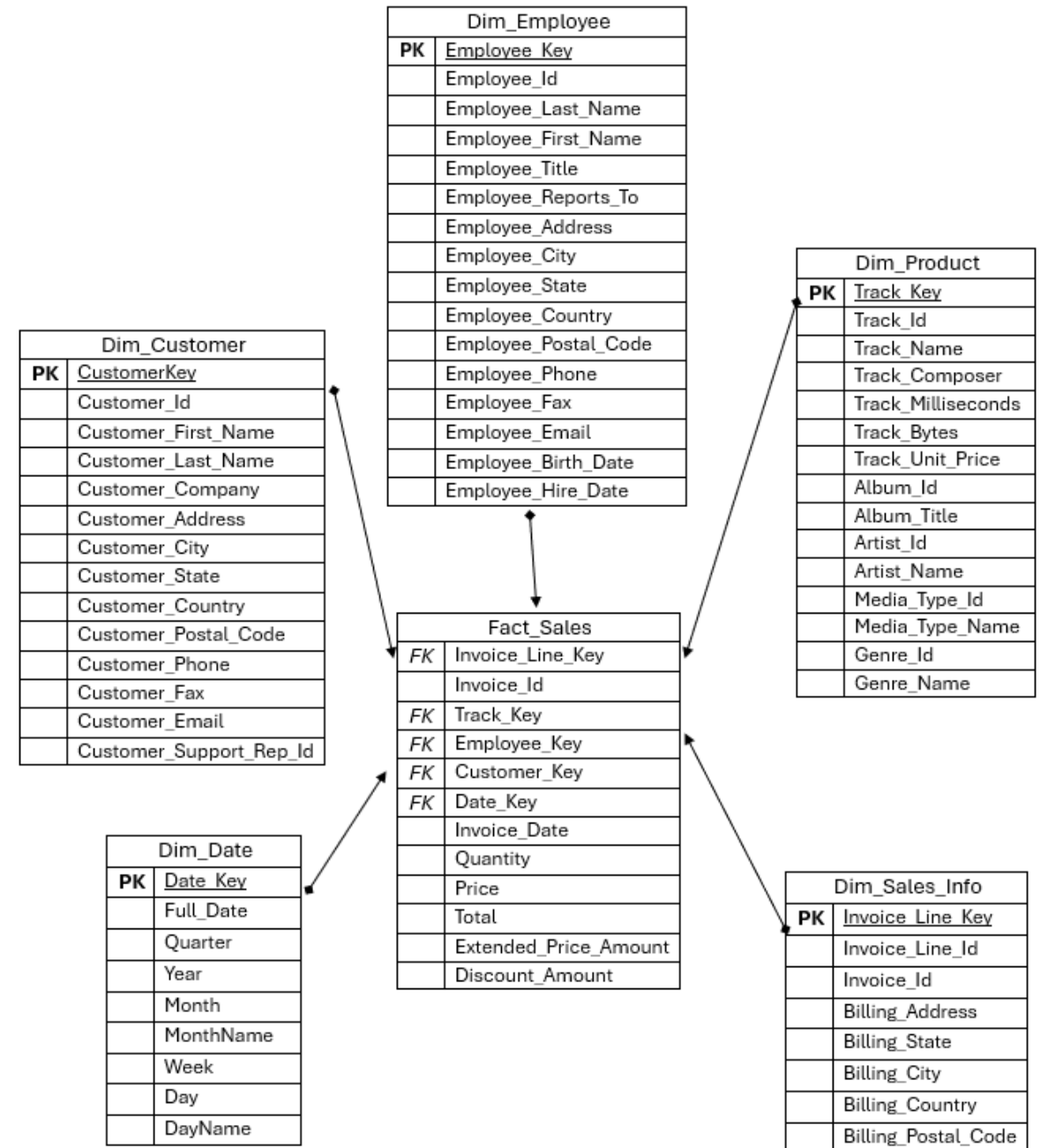
```
WHILE @CurrentDate <= @EndDate  
BEGIN
```

- To Develop a Staging Area from the OLTP Database.
- To Design the ERD Schemas and define the final Star Schema of the Data Warehouse.
- To Establish the connections to the main Fact Table.
- To Extract, Transform, Load data from the Staging Area into the Data Warehouse.
- To Ensure efficient data transfer and query performance.

```
INSERT INTO StagingArea (Date_Key, Invoice_Date, Quantity, Price, Total, Extended_Price_Amount, Discount_Amount)  
VALUES (  
    CONVERT(INT, FORMAT(@CurrentDate, 'yyyymmdd')), -- Datekey  
    @CurrentDate, -- FullDate  
    YEAR(@CurrentDate), -- Year  
    DATEPART(QUARTER, @CurrentDate), -- Quarter  
    MONTH(@CurrentDate), -- Month  
    DATENAME(MONTH, @CurrentDate), -- MonthName  
    DATEPART(WEEK, @CurrentDate), -- Week  
    DAY(@CurrentDate), -- Day  
    DATENAME(WEEKDAY, @CurrentDate) -- DayName  
);
```

Chinook Data Warehouse

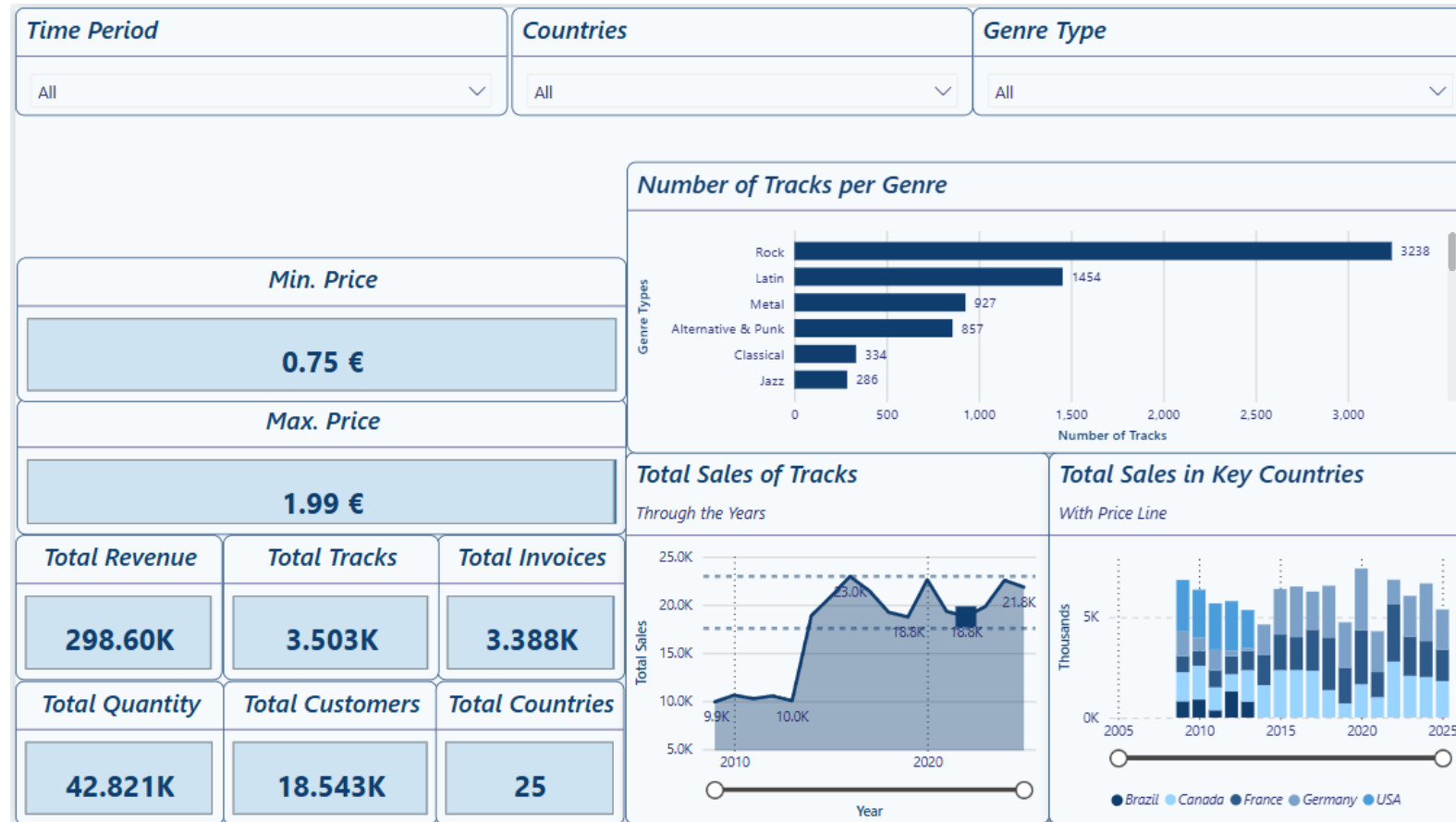
- Star Schema
- 5 Dimension Tables
- Main Focus : Fact_Sales
- Connections with primary keys
- 1 to N Cardinality



PowerBI Insights

Comments:

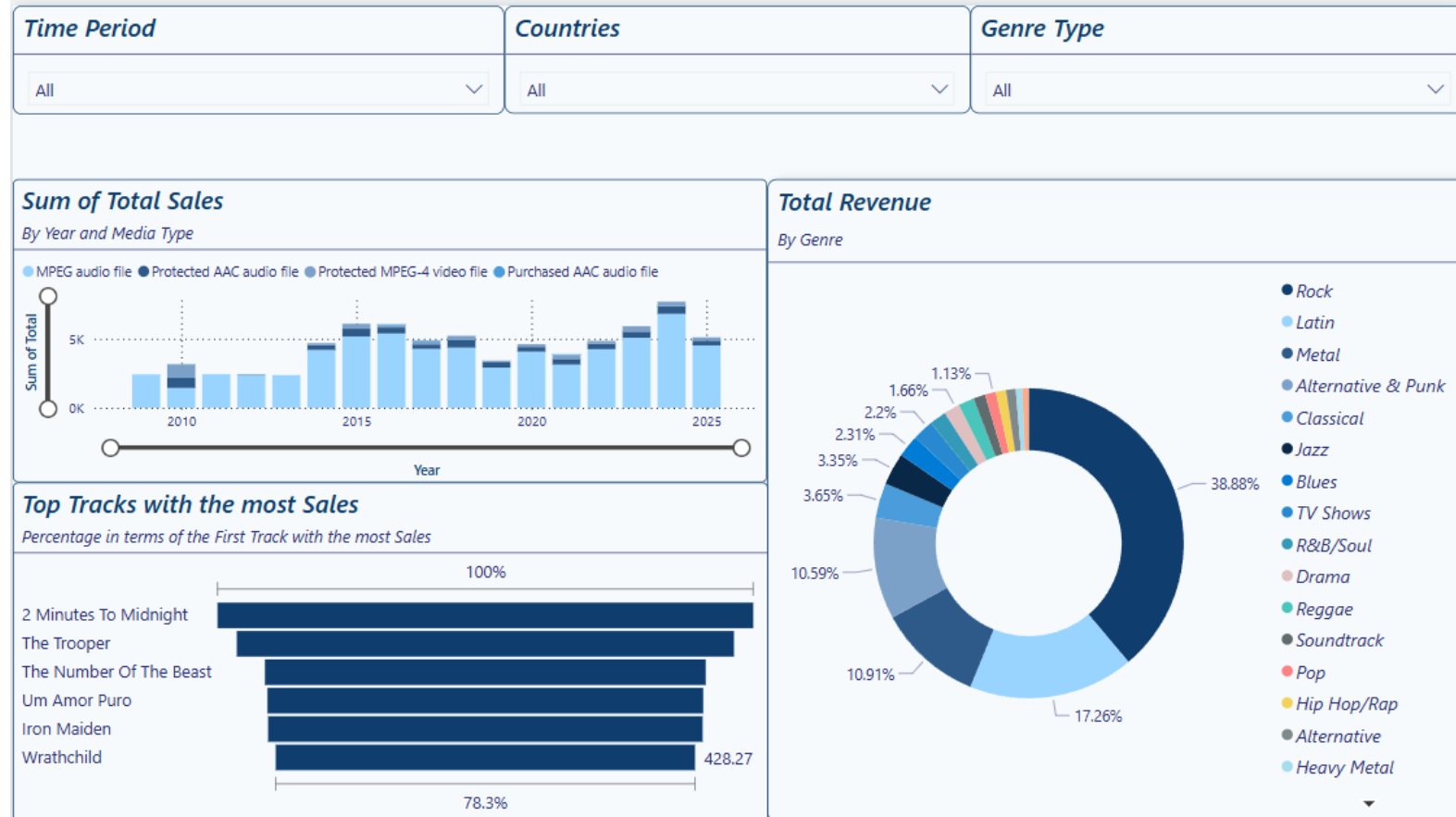
- Dashboards for a comprehensive analysis
- Slicer usage for customized results
- Cards to portray key analytics
- Variety on Visual Reports



PowerBI Insights

Comments:

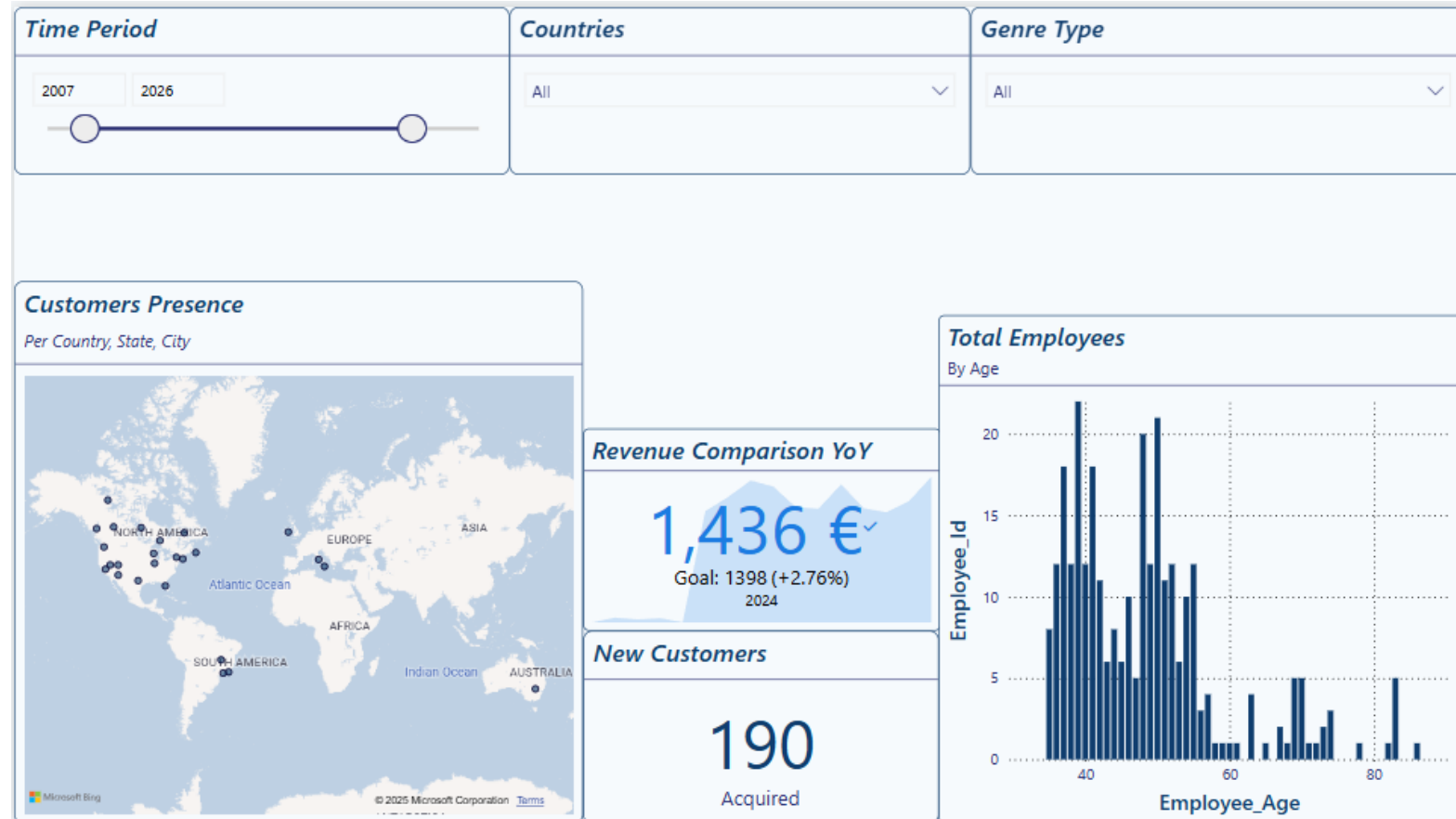
- Dashboards for a comprehensive analysis
- Slicer usage for customized results
- Cards to portray key analytics
- Variety on Visual Reports



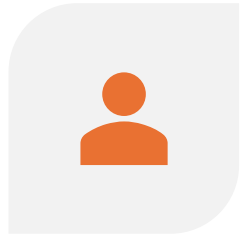
PowerBI Insights

Comments:

- Dashboards for a comprehensive analysis
- Slicer usage for customized results
- Cards to portray key analytics
- Variety on Visual Reports



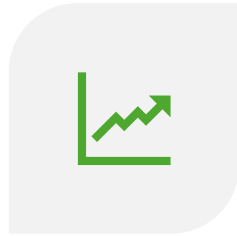
The Essentials of Part II



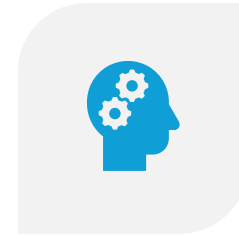
**EMPLOYEE SALES
PERFORMANCE
CLASSIFICATION**



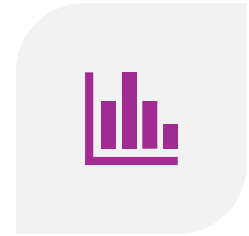
**PREPROCESSING
AND FEATURE
ENGINEERING**



**PERFORMANCE
LABELING &
ENCODING**



**MACHINE
LEARNING MODEL
PIPELINE**



**EVALUATION
INSIGHTS AND
ANALYSIS**


```

66 # Define performance labels based on quantiles of 'TotalRevenue'
67 quantile_labels = ['Low Performer', 'Average Performer', 'High Performer']
68 df_final['Performance_Label'] = pd.qcut(df_final['TotalRevenue'], q=3, labels=quantile_labels, duplicates='drop')
69
70
71 # Create a dictionary to map labels to desired numbers
72 label_mapping = {'Low Performer': 0, 'Average Performer': 1, 'High Performer': 2}
73
74 # Map labels to numbers
75 df_final['Performance_Label_Encoded'] = df_final['Performance_Label'].map(label_mapping)
76
77 # Display the first few rows of the filtered DataFrame
78 print(df_final.head())
79
80 # Define X (features) and y (target)
81 X = df_final[['TotalInvoices', 'AvgRevenue', 'AnnualRevenue']]
82 y = df_final['Performance_Label_Encoded']
83
84 # Normalize the features using StandardScaler
85 scaler = StandardScaler()
86 X_scaled = scaler.fit_transform(X)
87
88 # Stratified K-Fold Cross-Validation with GridSearchCV
89 models = {...}
90
91 skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
92
93 for model_name, (model, param_grid) in models.items():
94     # Train-Test-Validation Split (80%-10%-10%)
95     X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, stratify=y, random_state=42)
96     X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.5, stratify=y_train, random_state=42)
97
98     # Train and Evaluate Models and perform confusion matrix
99     best_estimator = GridSearchCV(model, param_grid, cv=skf, scoring='accuracy')
100     best_estimator.fit(X_train, y_train)
101
102     # Plot the distribution of performance labels
103     plt.figure(figsize=(8, 6))
104     value_counts.plot(kind='bar')
105     plt.title('Distribution of Performance Label Encoded')
106     plt.xlabel('Performance Label Encoded')
107     plt.ylabel('Count')
108     plt.xticks(rotation=0)
109     plt.show()
110
111 # Train and Evaluate Models and perform confusion matrix
112 for model_name, model in best_estimators.items():
113     # Train the model
114     model.fit(X_train, y_train)
115
116     # Predict on the test set
117     y_pred = model.predict(X_test)
118
119     # Calculate the confusion matrix
120     cm = confusion_matrix(y_test, y_pred)
121
122     # Print the confusion matrix
123     print(f'Confusion Matrix for {model_name}:')
124     print(cm)

```

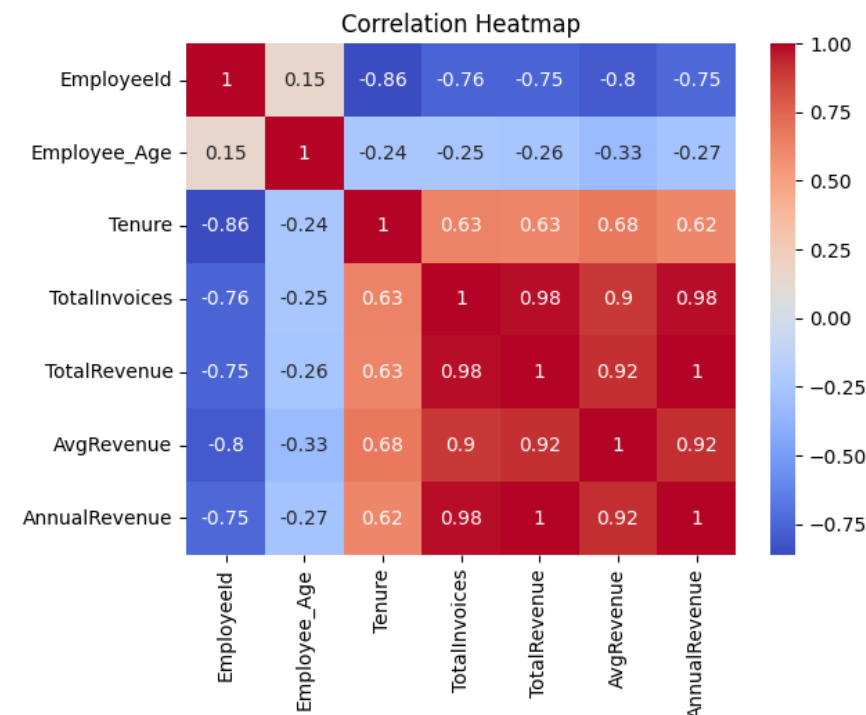
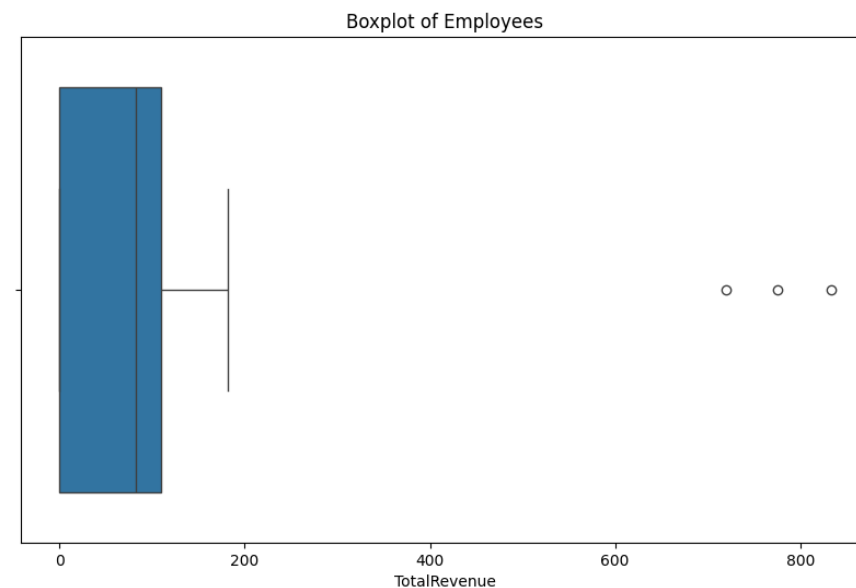
Python Scripts

- To Calculate Employee Tenure, Clean Nulls and Remove Outliers.
- To Create Correlation Plots and define the final features.
- To Establish Performance Labels and Encode them.
- To Set Hyperparameters for Optimization, Standardize features and Split the Data Set.
- To Train Models, Evaluate Metrics & Predictions and Generate insightful plots

Pre-Processing & Feature Engineering

Comments:

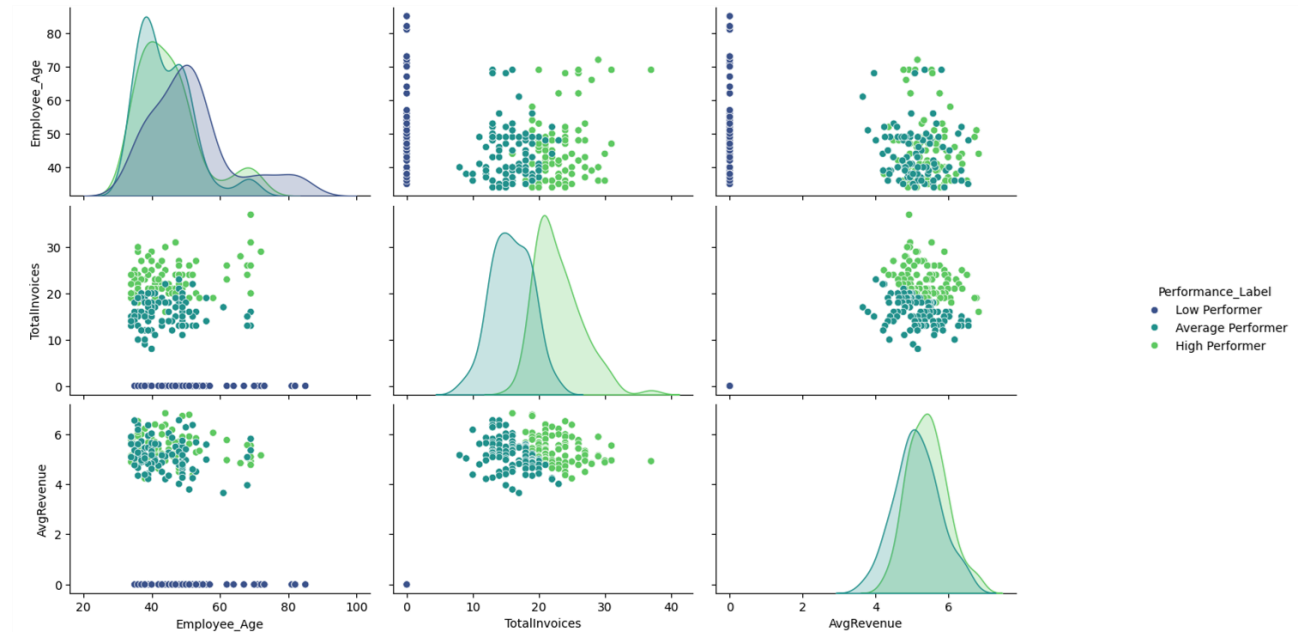
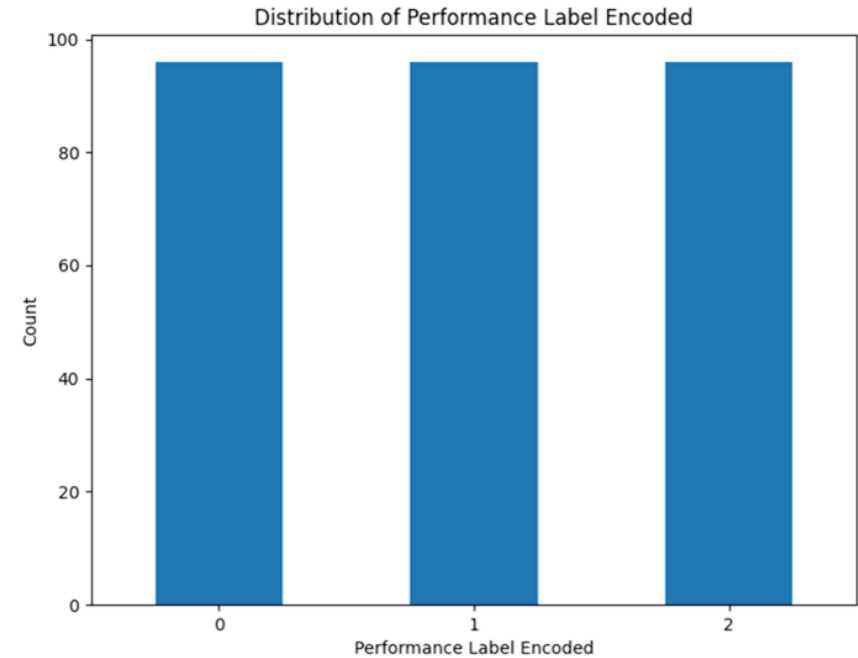
- Sample Cleanup with NA filling and Feature Calculation such as “Tenure” as well as “Annual Revenue”
- Outlier Removal such as salesmen with “Total Revenue” more than 600 and employees who are not salesmen
- Feature Selection based on High Correlation to the “Total Revenue” such as “Total Invoices”, “Annual Revenue”, “Average Revenue”



Label Performance & Encoding

Comments:

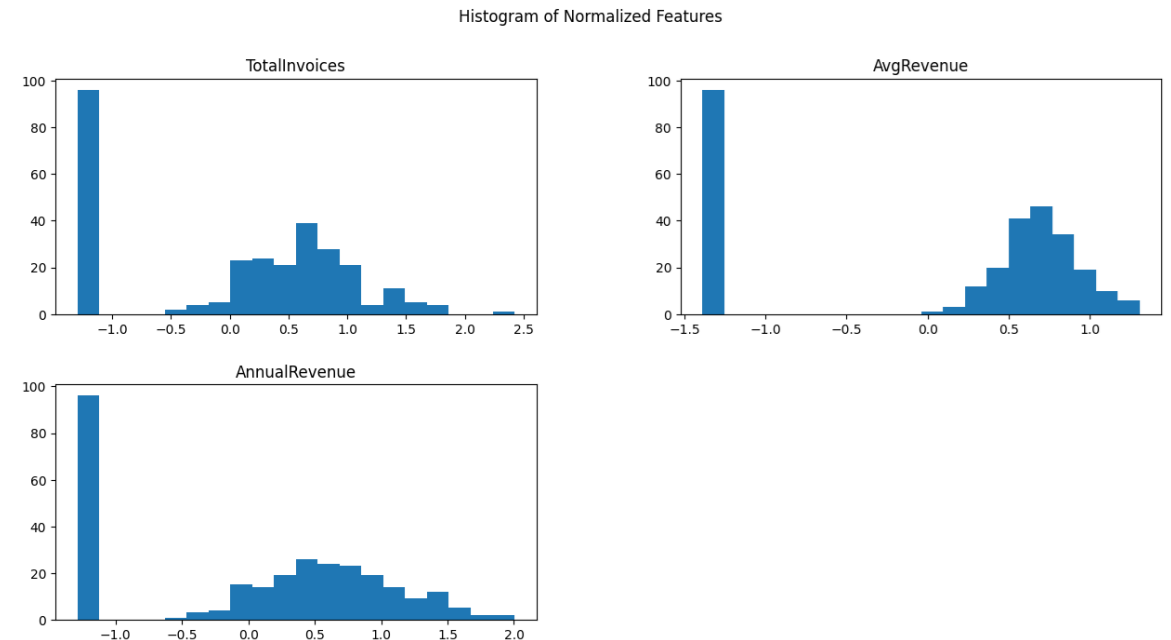
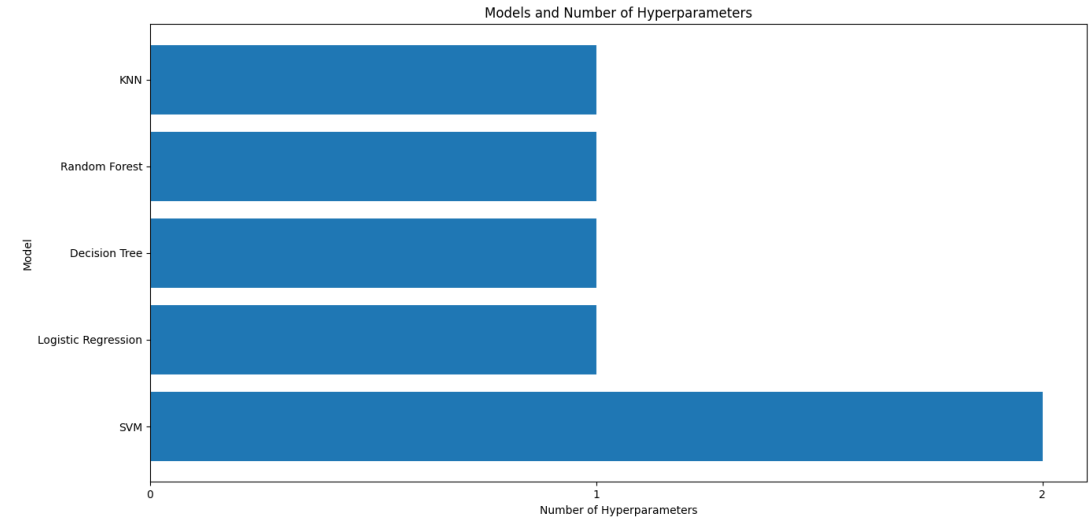
- Labeling based on Performance using Quantiles (High, Average, Low)
- Label Encoding manually with Pandas and Label Mapping



Standardization, Model Selection & Training

Comments:

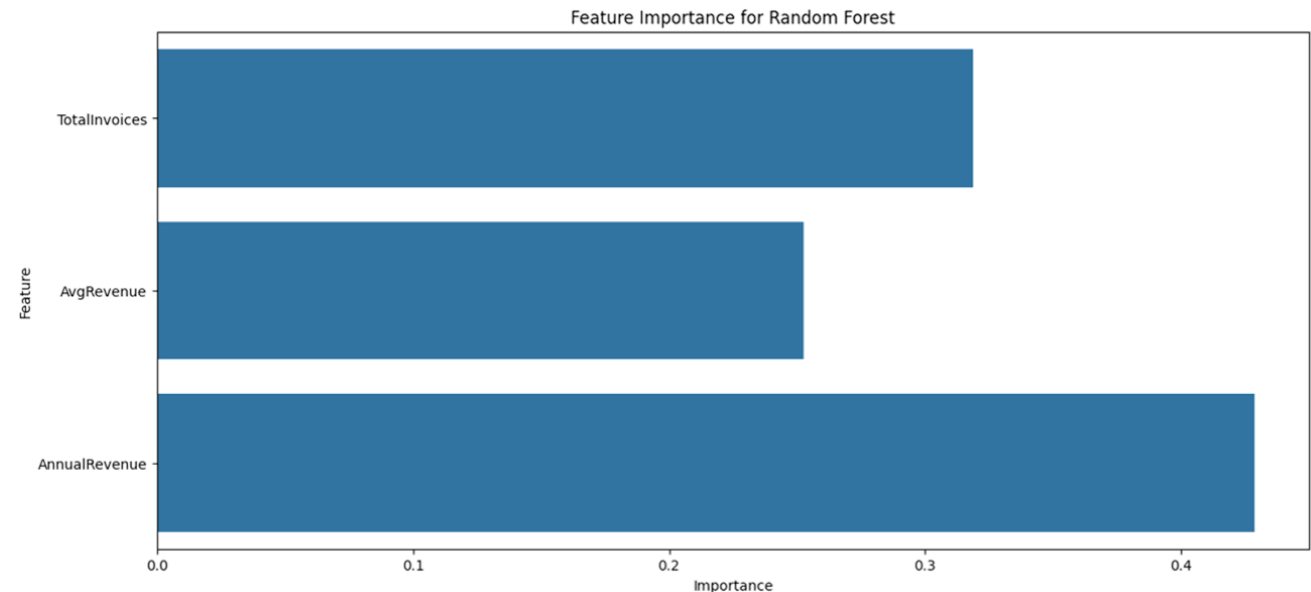
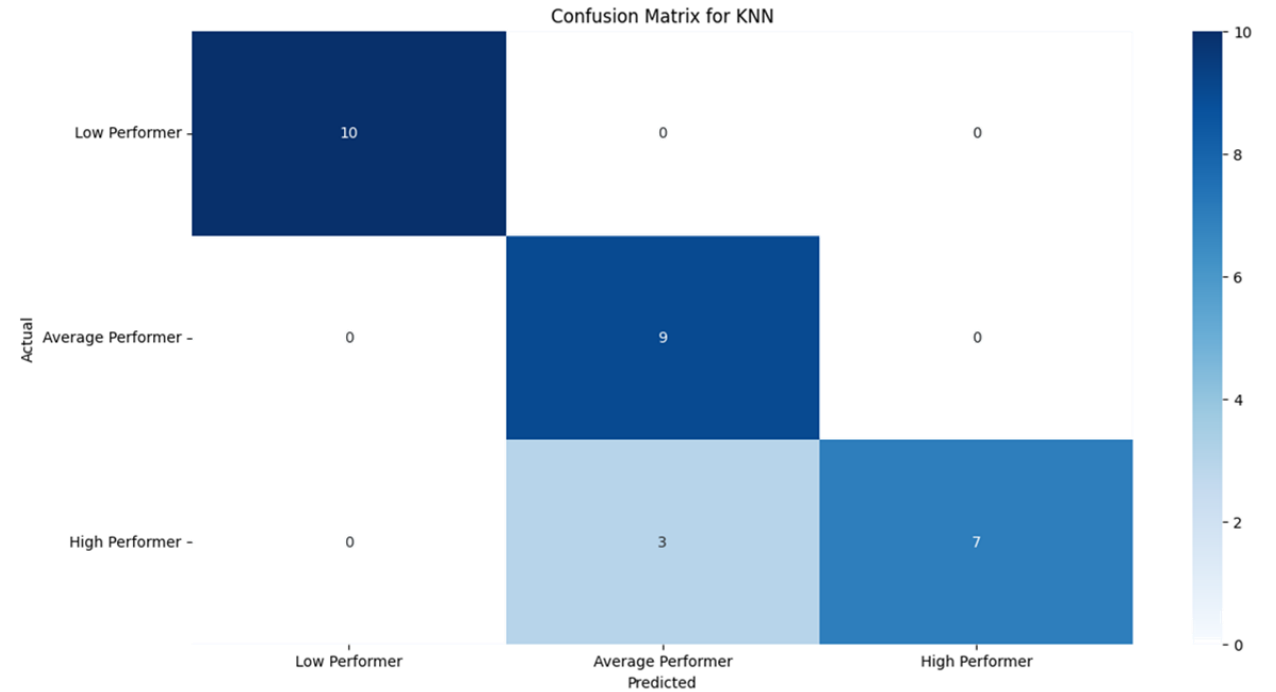
- Feature Scaling of the Training Data with Standard Scaler
- Classification Model Selection for the training
- Set of Hyperparameters per Model using Stratified K-fold Cross Validation with GridSearch



Model Evaluation, Metrics & Confusion Matrix (Part I)

Comments:

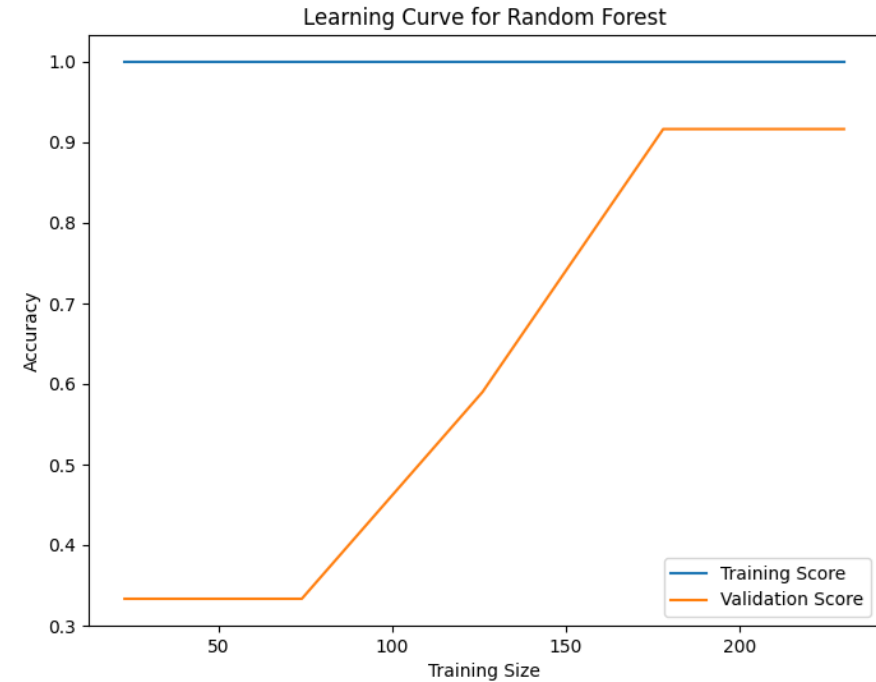
- Strong results based on the classification report per Model
- Metrics achieve over 96% score on Average



Model Evaluation, Metrics & Confusion Matrix (Part II)

Comments:

- Strong results based on the classification report per Model
- Metrics achieve over 96% score on Average
- Difference between Metrics when we removed all NA values reducing however the Data Set size



Evaluating Decision Tree...

Accuracy: 0.966
Precision: 0.969
Recall: 0.966
F1 Score: 0.966

Classification Report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	10
1	0.90	1.00	0.95	9
2	1.00	0.90	0.95	10
accuracy			0.97	29
macro avg	0.97	0.97	0.96	29
weighted avg	0.97	0.97	0.97	29

VS

Evaluating Decision Tree...

Accuracy: 0.850
Precision: 0.863
Recall: 0.850
F1 Score: 0.853

Classification Report:

	precision	recall	f1-score	support
0	1.00	0.83	0.91	6
1	0.75	0.86	0.80	7
2	0.86	0.86	0.86	7
accuracy			0.85	20
macro avg	0.87	0.85	0.86	20
weighted avg	0.86	0.85	0.85	20

Project Highlights

Learning Outcomes

- ❑ End-to-end process of transforming raw data into insights.
- ❑ Implementing modern techniques to solve real problems
- ❑ Getting Familiar with the Tech Stack

Tech Stack

- ❑ GitHub
- ❑ MS SQL Server
- ❑ MS Power BI
- ❑ Python

Challenges Faced

- ❑ Schema Set up for DW
- ❑ Feature Engineering
- ❑ Learning a new coding language
- ❑ The correct order of steps for a ML project

Thank you for your attention.

From Our Team:

Alexia Kalliani

Giorgos Petrakis

Konstantinos Gkaravelos

Nikos Antoniou