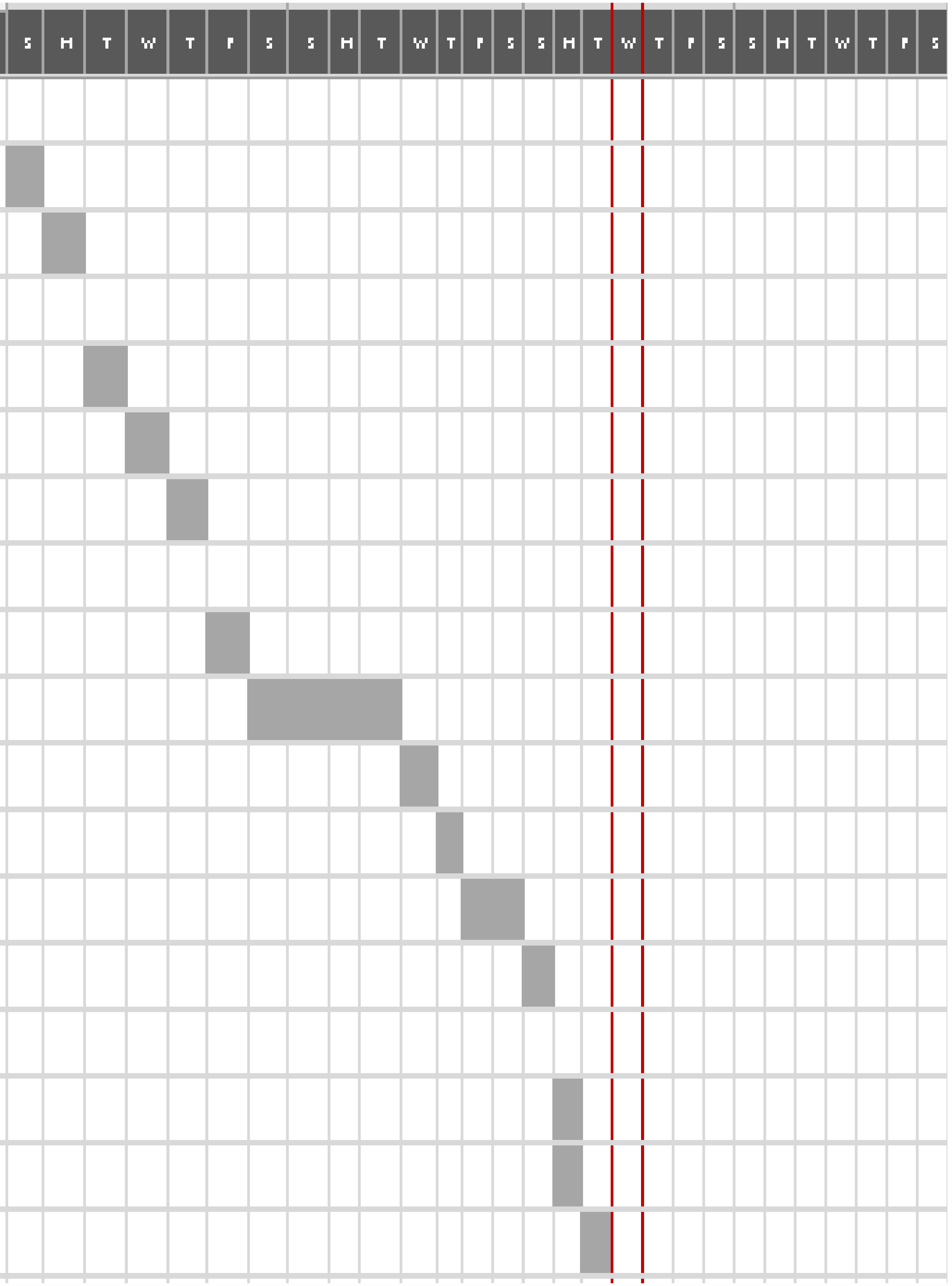
Capstone Project
Data Science Bootcamp

Presented by:
Bytes Matter Group

TASK	ASSIGNED TO	PROGRESS	START	END
Data Selection				
Select multiple datasets	Team Members	100%	8/21/22	8/21/22
Choose dataset	Team Members	100%	8/22/22	8/22/22
Data Preprocessing				
Rename, removal of columns	Team Members	100%	8/23/22	8/23/22
EDA	Team Members	100%	8/24/22	8/24/22
Outliers identification and handlings	Team Members	100%	8/25/22	8/25/22
Data Modeling				
Target and features identification	Team Members	100%	8/26/22	8/26/22
Build models	Team Members	100%	8/27/22	8/30/22
Model evaluation	Team Members	100%	8/31/22	8/31/22
Best model election	Team Members	100%	9/1/22	9/1/22
Optemization	Team Members	100%	9/2/22	9/3/22
Pipelines	Team Members	100%	9/4/22	9/4/22
Finalization				
Power BI dashboard	Team Members	100%	9/5/22	9/5/22
Web blog	Team Members	100%	9/5/22	9/5/22
Presentation	Team Members	100%	9/6/22	9/6/22



Saudi Vision 2030

Education Systeem

The Saudi Arabia Vision 2030 aims to increase the efficiency of the educational sector and bring it to the best possible level in the offered programs of education level. It relies on the latest and modern education strategies and raises the learners' to think and analyze.

Human Capability Development Program

Human Capability Development Program

The Program aims to prepare citizens for the job market and to be able to compete globally. It will do this through developing basic and future skills, developing knowledge and values that enhance the 21st century and global citizenship skills.

Key Objectives

02 Project indicators and Objectives:

Education Systeem

OBJECTIVE **01**

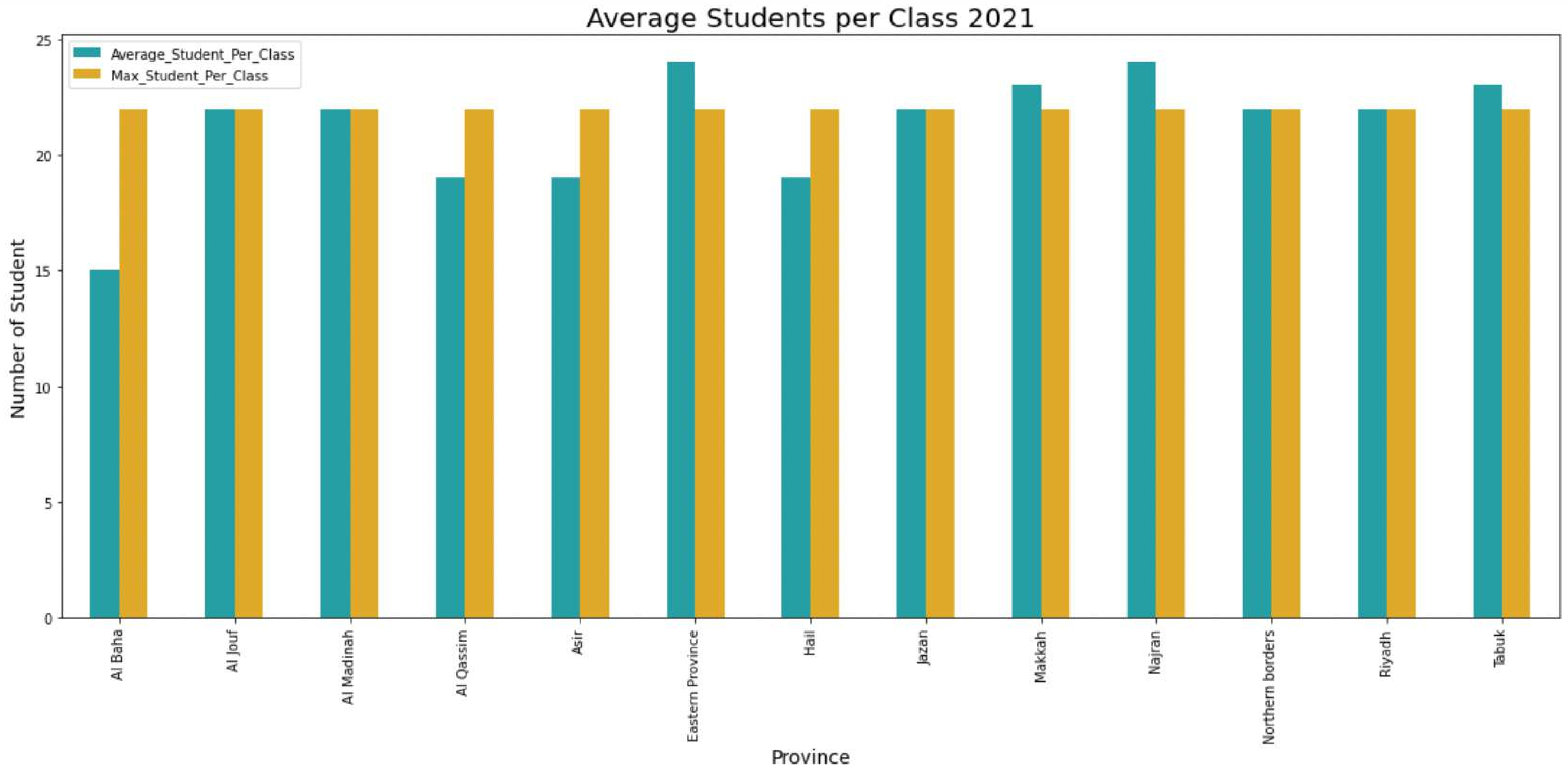
Kindergarten enrollment increase to 40% in 2025

OBJECTIVE **02**

The number of students does not exceed 22 students per class

02 Students per Class

Challenges



Exploratory Data Analysis

3.1 The DataSet

DATASET: Students in Public Schools 2014-2021
COLUMNS: 21
ROWS: 45537

Indicators:

- New Students Column
- New Saudi Students Column
- Total Teachers Column
- Total Saudi Teachers Column
- Total AdmInistrators Column
- Total Saudi AdmInistrators Column
- School Type Column
- Total Servants Column
- Total Workers Column

- Year Column
- Province Column
- Education Department Column
- Education Office Column
- Authority Column
- Educational Level Column
- School Type Column
- School Gender Column
- School System Column
- Total Classes Column
- Total Students Column
- Total Saudi Students Column

3.2 Exploratory Data Analysis

3.2.1 Packages

```
# EDA
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px

# visualizations
import seaborn as sns
import numpy as np
from plotly import graph_objs as go

# colors
pallete4 = ["#61B8A9","#67606F","#438B7E","#B0A48F"]
```

3.2 Exploratory Data Analysis

```
# print nuber of rows and coloumns in the dataset
student.shape
```

```
(45536, 21)
```

```
#checking the null in the dataset
student.isna().sum()
```

```
Year (Gregorian)              0
Year (Hijri)                  0
Province                      0
Education Department          0
Education_Office              0
Authority                     0
Educational_Level             0
School_Type                   0
School_Gender                 0
School_System                 0
Total_Classes                 0
Total_Students                0
Total_Saudi_Students          0
New_Students                  0
New_Saudi_Students            0
Total_Teachers                0
Total_Saudi_Teachers          0
Total_AdmInistrators          0
Total_Saudi_AdmInistrators    0
Total_Servants                0
Total_Workers                 0
dtype: int64
```

```
# print data type for the colomns
student.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45536 entries, 0 to 45535
Data columns (total 21 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Year (Gregorian)            45536 non-null  object
 1   Year (Hijri)                45536 non-null  object
 2   Province                    45536 non-null  object
 3   Education Department        45536 non-null  object
 4   Education_Office            45536 non-null  object
 5   Authority                   45536 non-null  object
 6   Educational_Level           45536 non-null  object
 7   School_Type                 45536 non-null  object
 8   School_Gender               45536 non-null  object
 9   School_System               45536 non-null  object
 10  Total_Classes               45536 non-null  int64
 11  Total_Students              45536 non-null  int64
 12  Total_Saudi_Students        45536 non-null  int64
 13  New_Students                45536 non-null  int64
 14  New_Saudi_Students          45536 non-null  int64
 15  Total_Teachers              45536 non-null  int64
 16  Total_Saudi_Teachers        45536 non-null  int64
 17  Total_AdmInistrators        45536 non-null  int64
 18  Total_Saudi_AdmInistrators  45536 non-null  int64
 19  Total_Servants              45536 non-null  int64
 20  Total_Workers               45536 non-null  int64
dtypes: int64(11), object(10)
memory usage: 7.3+ MB
```

3.2 Exploratory Data Analysis

```
# print calculating some statistical data like percentile, mean and std of the numerical
student.describe()
```

	Total_Classes	Total_Students	Total_Saudi_Students	New_Students	New_Saudi_Students	Total_Teachers	Total_Saudi_Teachers	Total_AdmInistrators	Total_Saudi_AdmInistrators	Total_Servants	Total_Workers	Students_Per_Class
count	45150.000000	45150.000000	45150.000000	45150.000000	45150.000000	45150.000000	45150.000000	45150.000000	45150.000000	45150.000000	45150.000000	45150.000000
mean	50.331561	1074.873798	891.178494	267.819070	222.077298	93.669568	87.002215	18.855858	18.603942	0.962924	1.911561	16.191650
std	109.334510	2803.360778	2252.267942	660.841948	536.005995	202.857613	196.763013	62.122301	62.035477	4.104096	6.423898	14.902291
min	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	4.000000	40.000000	32.000000	7.000000	5.000000	4.000000	3.000000	0.000000	0.000000	0.000000	0.000000	8.000000
50%	12.000000	173.000000	145.000000	37.000000	30.000000	19.000000	16.000000	2.000000	2.000000	0.000000	0.000000	16.000000
75%	43.000000	825.000000	736.000000	199.000000	178.000000	87.000000	75.000000	11.000000	11.000000	0.000000	1.000000	22.000000
max	1726.000000	37657.000000	32512.000000	10738.000000	9578.000000	3006.000000	2399.000000	1152.000000	1152.000000	150.000000	118.000000	1232.000000

3.2 Exploratory Data Analysis

```
#returning unique values for Authority column
student['Authority'].value_counts()
```

```
Public               36245
Private               6806
International         2382
Royal Commission       103
Name: Authority, dtype: int64
```

```
#returning the unique values for Educational_Level column
student['Educational_Level'].value_counts()
```

```
High            14863
Elementary      14816
Middle          12503
Kindergarten     3354
Name: Educational_Level, dtype: int64
```

```
#returning the unique values for School_Type column
student['School_Type'].value_counts()
```

```
Day                      21141
Religious (Koranic)       8920
Special education         8453
Adult Education           7022
Name: School_Type, dtype: int64
```

```
#returning the unique values for School_System column
student['School_System'].value_counts()
```

```
General system       40449
curriculum system     5087
Name: School_System, dtype: int64
```

```
#returning the unique values for School_Gender column
student['School_Gender'].value_counts()
```

```
Boys     22999
Girls    22537
Name: School_Gender, dtype: int64
```

```
#returning the unique values for Province column
student['Province'].value_counts()
```

```
Riyadh              9177
Makkah              8254
Asir                4995
Eastern Province    4636
Al Qassim           3482
Al Madinah          3030
Jazan               2889
Hail                2118
Al Baha             1755
Tabuk               1675
Al Jouf             1359
Najran              1145
Northern borders    1021
Name: Province, dtype: int64
```

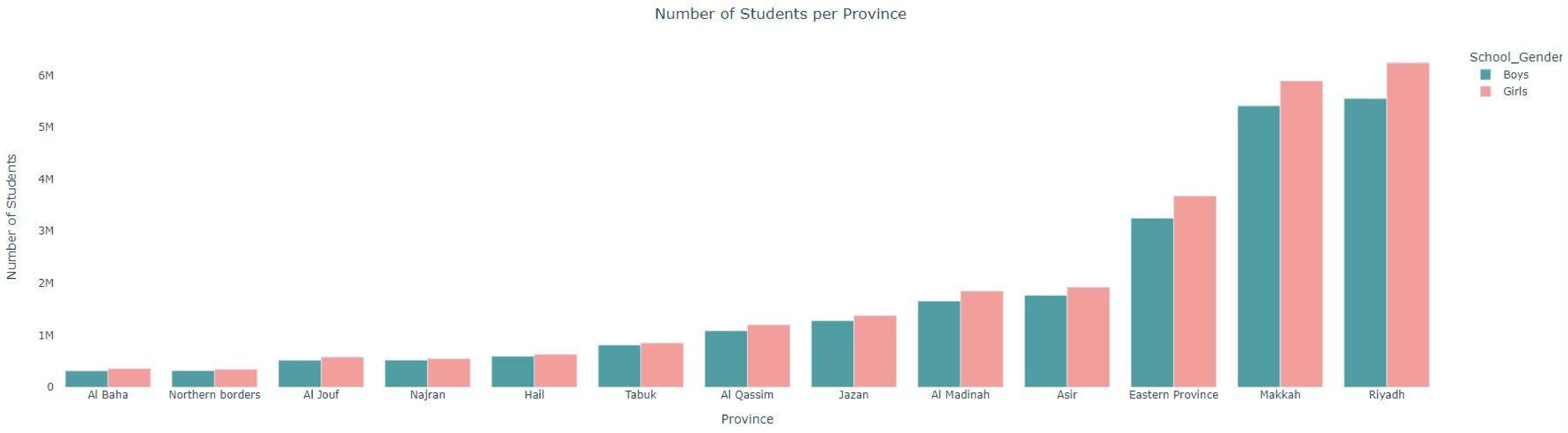
3.2 Exploratory Data Analysis

3.2.2 Preprocessing

```
# Rename Columns:
education_system.rename(columns = {'Year (Gregorian)':'Year'}, inplace = True)

# Drop the irrelevant  Columns That Aren't Useful:
education_system = education_system.drop(['Education Department', 'Education_Office', 'Year (Hijri)' ,
                                          'Total_Saudi_Students','New_Saudi_Students', 'Total_Saudi_Teachers',
                                          'Total_Saudi_AdmInistrators' , 'Total_Servants', 'Total_Workers' ],
                                         axis=1) #We dropped nine columns

# Change Columns Types
education_system['Year'] = education_system['Year'].astype(object)
```

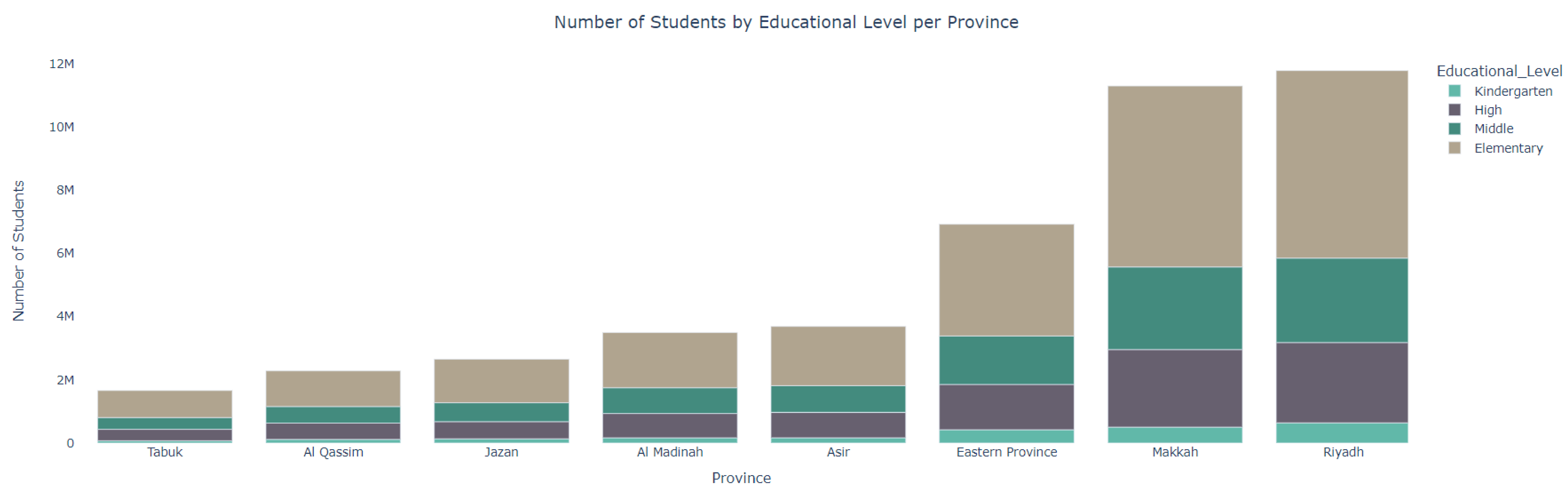
3.2.3 Visualization

1. Number of Students by Gender Per Province



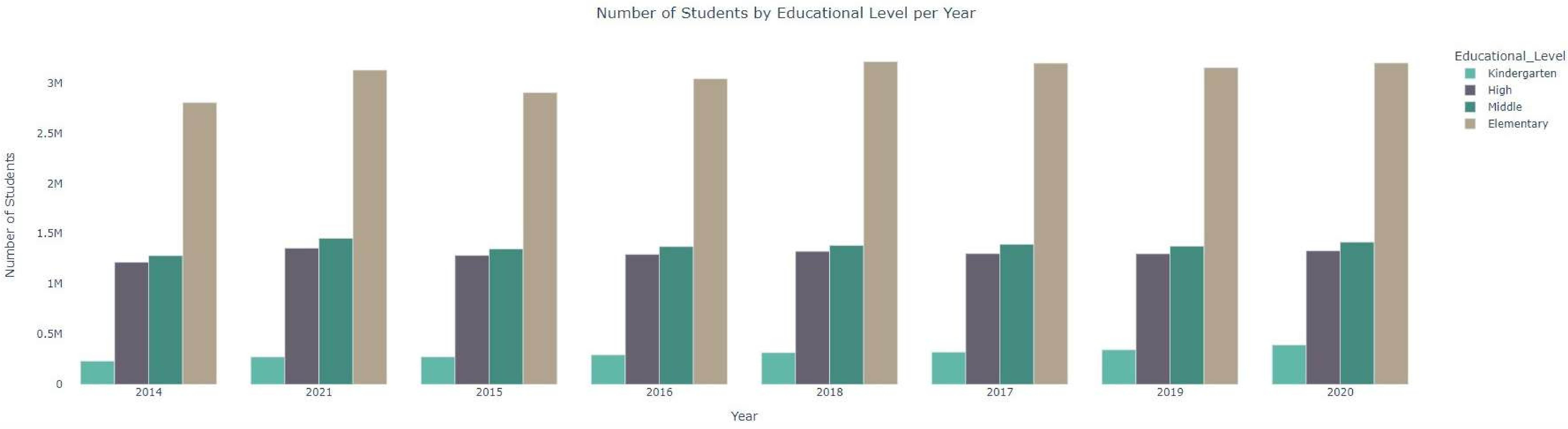
3.2.3 Visualization

2. Number of Students by Educational Level per Province



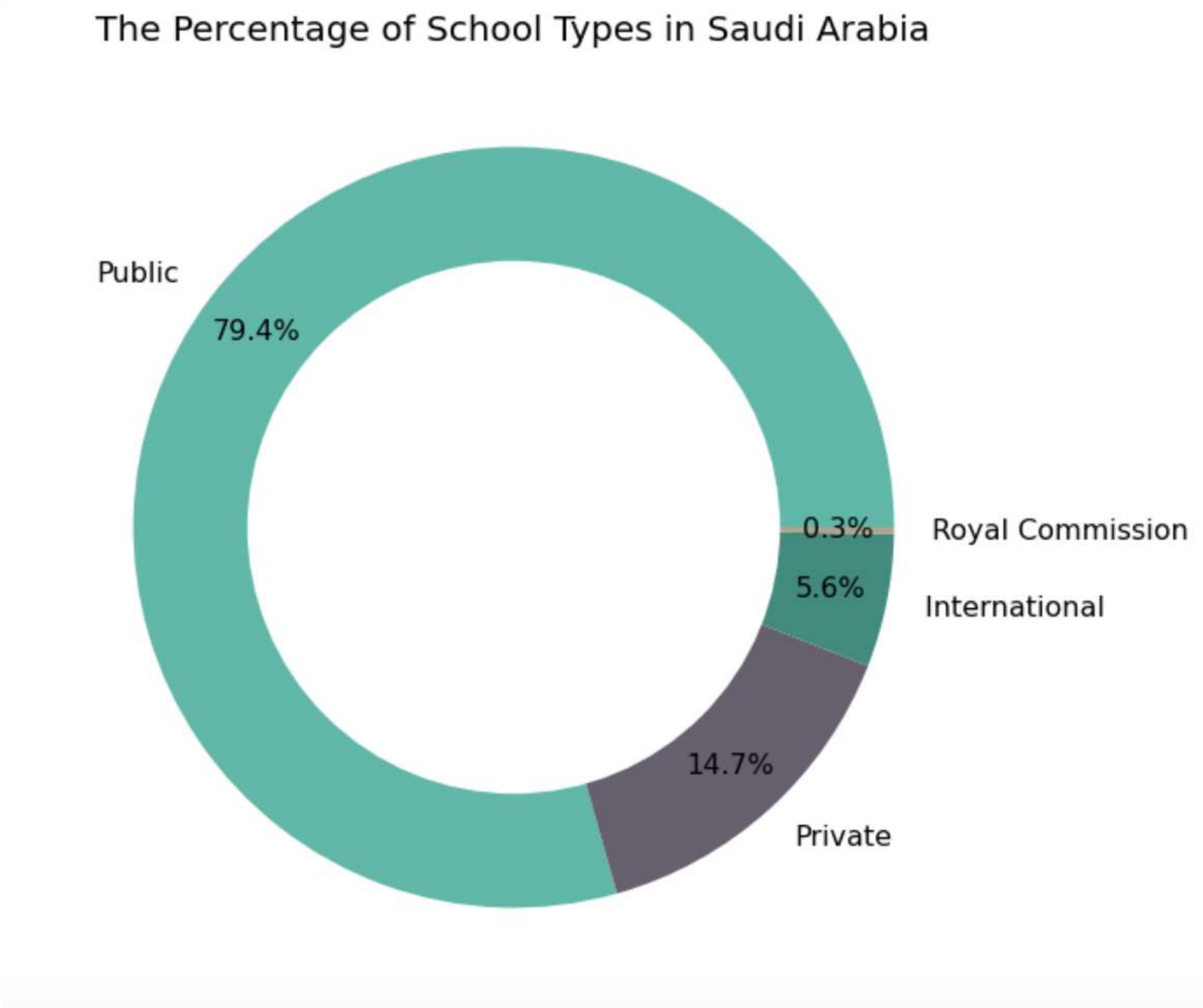
3.2.3 Visualization

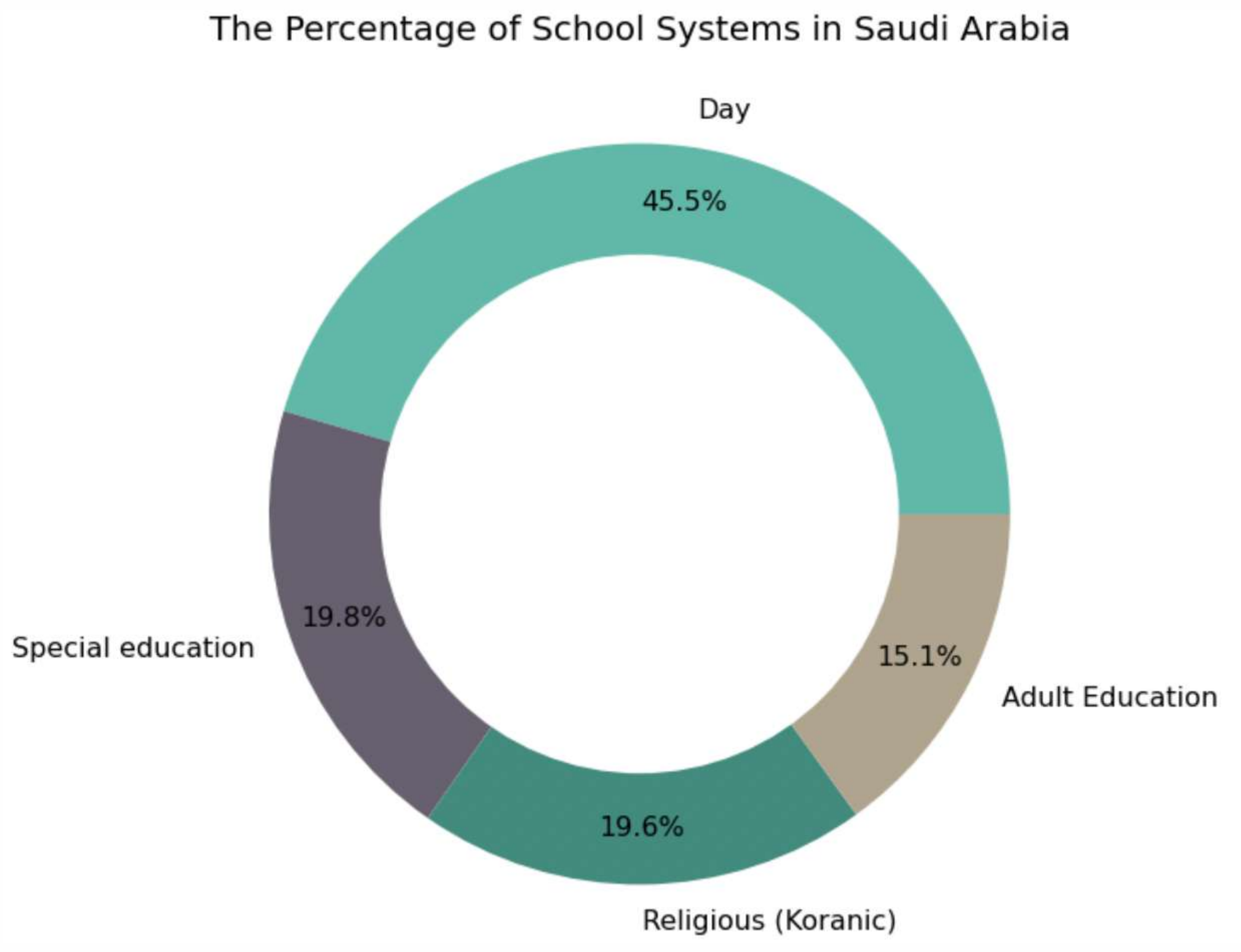
3. Number of Students by Educational Level per Year



3.2.3 Visualization

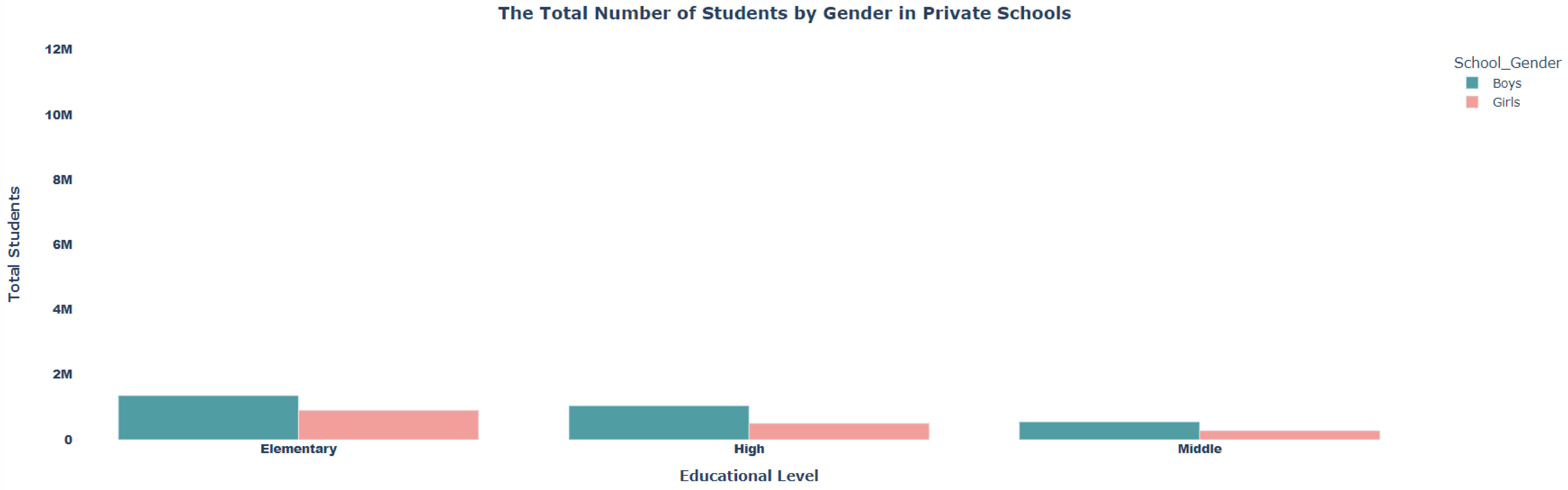
4. The Percentage of School Types & Systems in Saudi Arabia





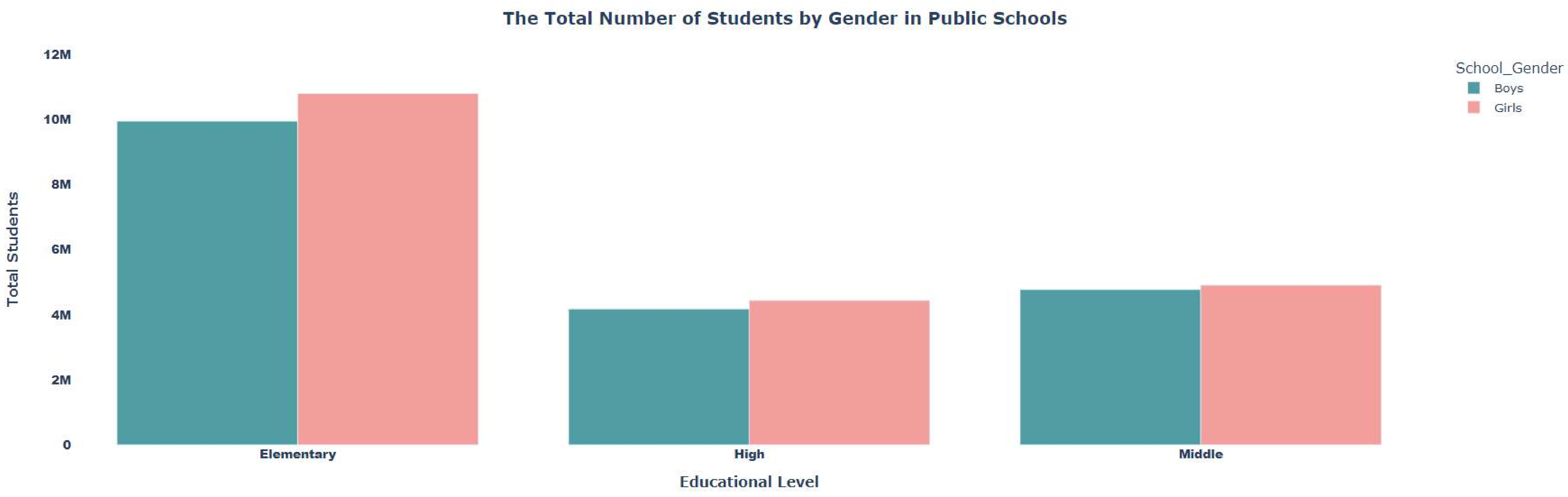
3.2.3 Visualization

5. The Total Number of Students By Gender In Private Schools



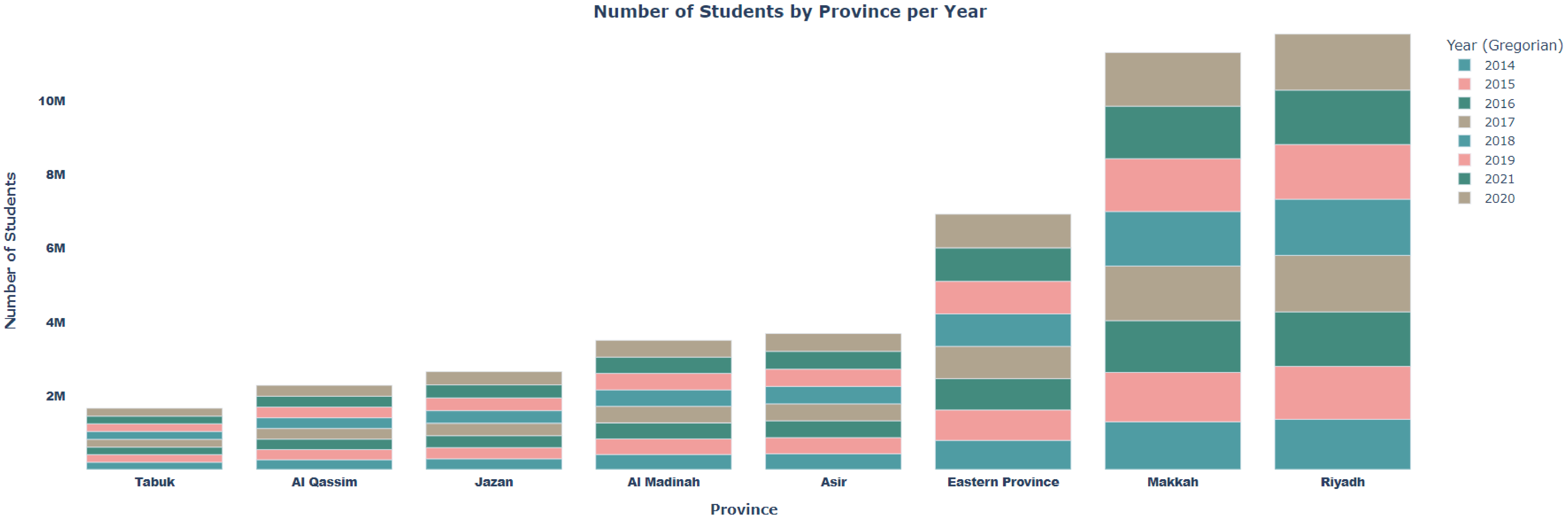
3.2.3 Visualization

6. The Total Number of Students By Gender In Public Schools



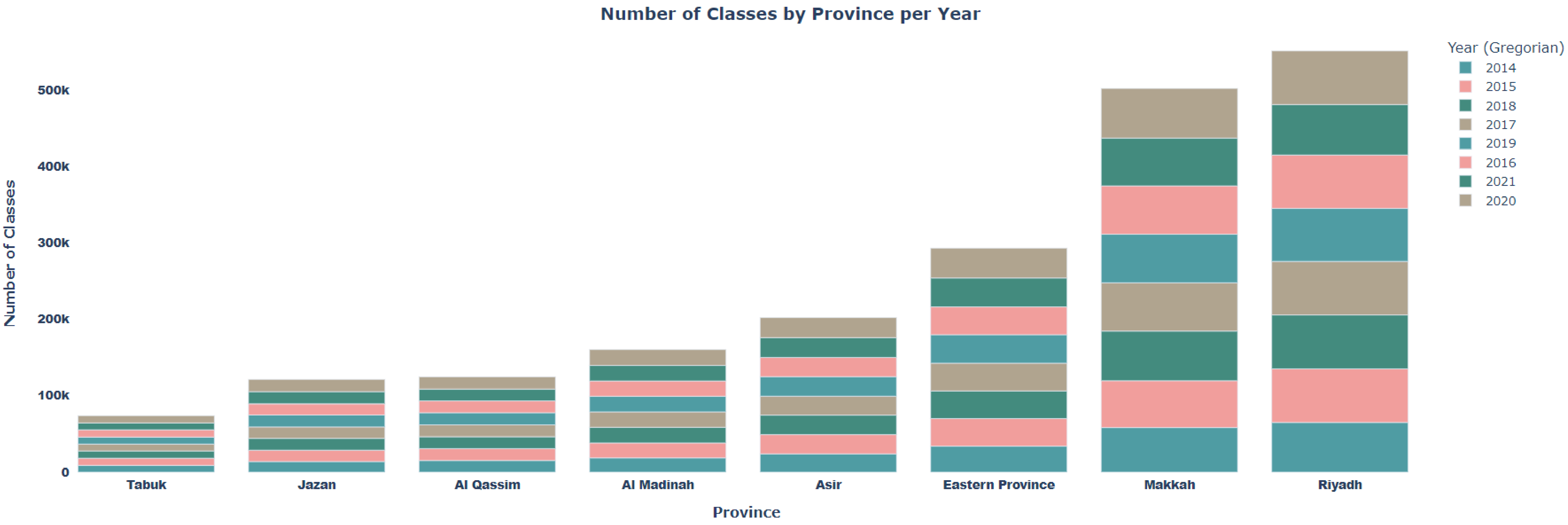
3.2.3 Visualization

7. Top 8 Province with Total Students by Year



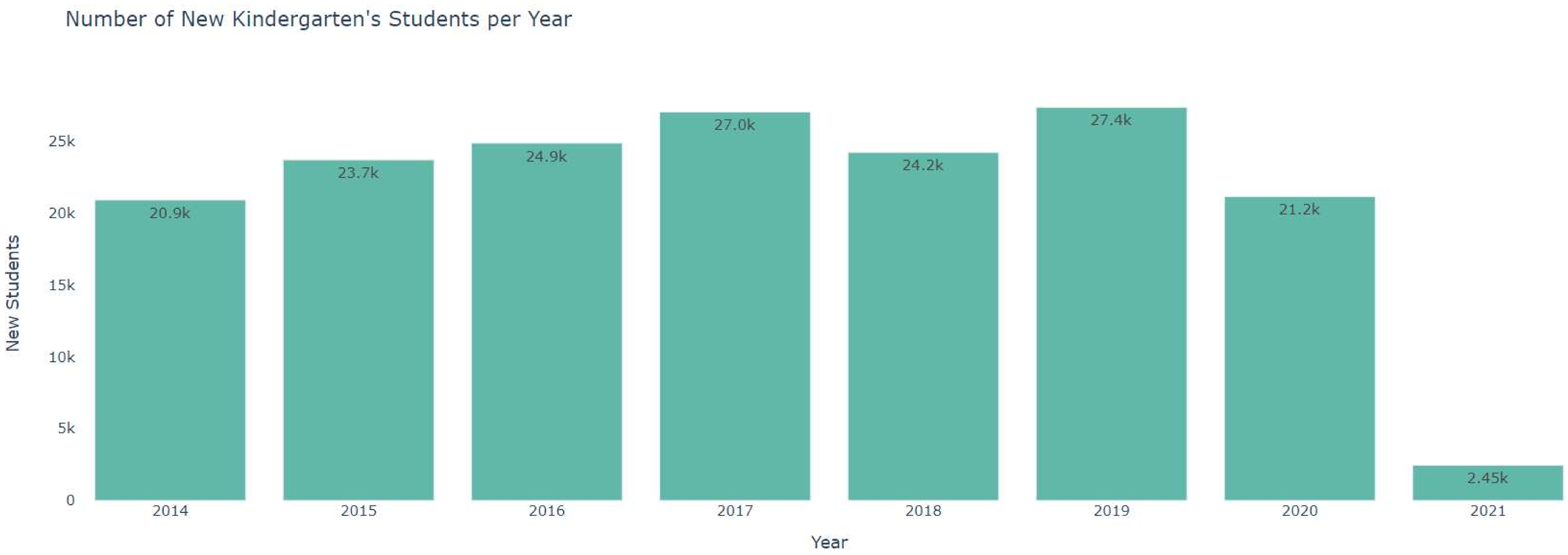
3.2.3 Visualization

8. Top 8 Province with Total Classes by Year



3.2.3 Visualization

9. Number of New Kindergarten's Students per Year



Build Machine Learning Models

4 Machine Learning Models

4.1 Packages

```python
# for building the models (Linear Regression) & (Polynomial Regression)
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures

# for the train split
from sklearn.model_selection import train_test_split

# for the errors (Example Cost Functions for Regression) :
# used to quantify and minimize the errors in regression models
from sklearn.metrics import mean_absolute_error # MAE
from sklearn.metrics import mean_squared_error # MSE
from sklearn.metrics import r2_score

from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import plot_confusion_matrix
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
from sklearn.preprocessing import scale
from sklearn import model_selection
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.neural_network import MLPRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn import neighbors
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBRegressor
from sklearn import metrics
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import StandardScaler
from sklearn import preprocessing
```

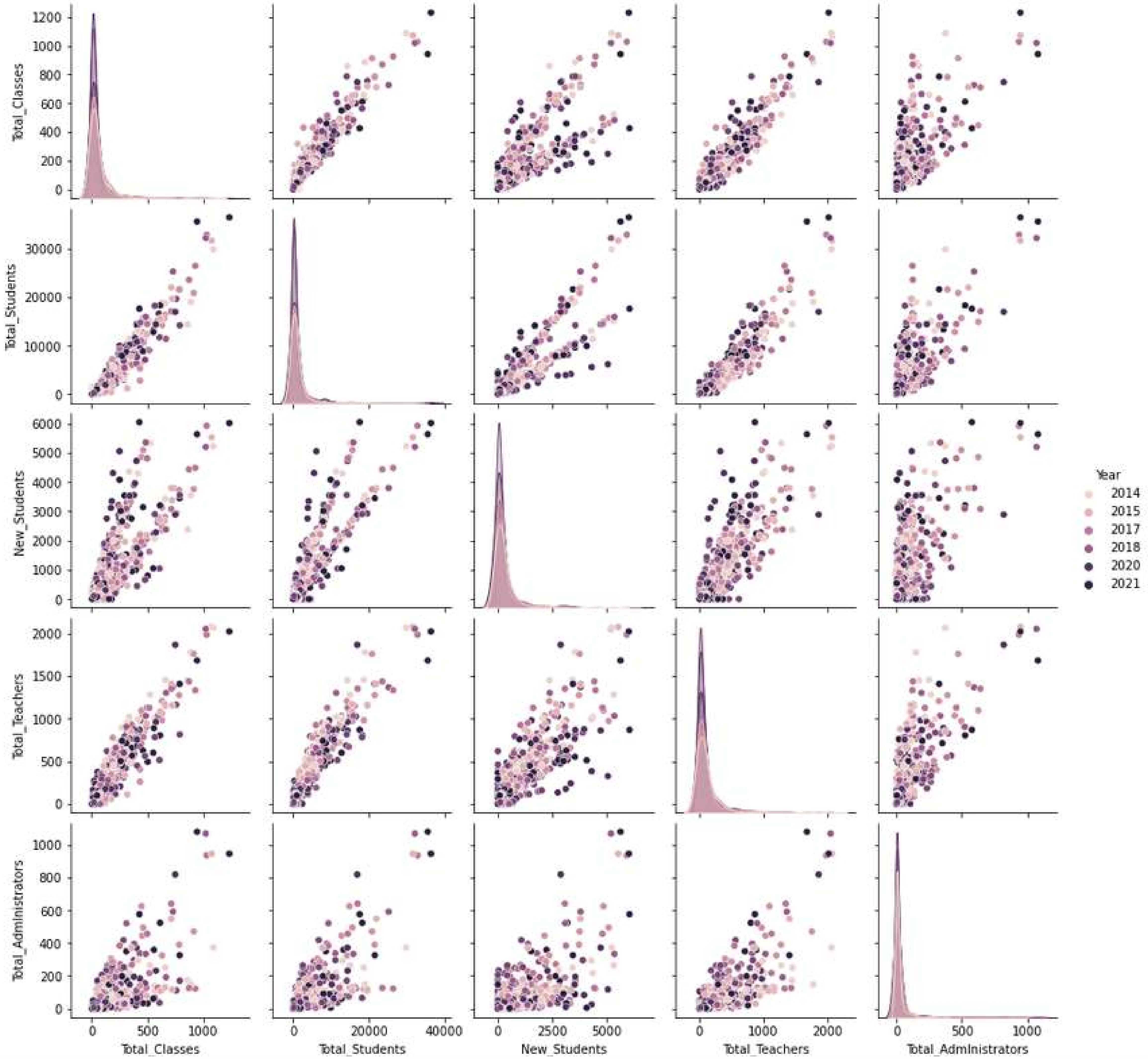
```python
# pipeline
from sklearn.pipeline import Pipeline
from sklearn.pipeline import make_pipeline
from sklearn.compose import ColumnTransformer
from sklearn.impute import SimpleImputer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.metrics import accuracy_score
from imblearn.over_sampling import SMOTE
```

4 Machine Learning Models

4.2 Preprocessing

```
# Rename Columns:
education_system.rename(columns = {'Year (Gregorian)':'Year'}, inplace = True)

# Drop the irrelevant  Columns That Aren’t Useful:
education_system = education_system.drop(['Education Department', 'Education_Office', 'Year (Hijri)' ,
                                          'Total_Saudi_Students','New_Saudi_Students', 'Total_Saudi_Teachers',
                                          'Total_Saudi_AdmInistrators' , 'Total_Servants', 'Total_Workers' ],
                                         axis=1) #We dropped nine columns

# Change Columns Types
education_system['Year'] = education_system['Year'].astype(object)
```

4.3 Correlation



4 Machine Learning Models

4.4 Outliers Identification

- Skewness

```
# the skewness value should be within the range of -1 to 1 for a normal distribution,
# any major changes from this value may indicate the presence of outliers.

print('skewness value of Total Students: ',education_system['Total_Students'].skew())
print('skewness value of Total Classes: ',education_system['Total_Classes'].skew())
```

```
skewness value of Total Students:  5.79556672864755
skewness value of Total Classes:  4.946723587913649
```

4 Machine Learning Models

4.4 Outliers Identification

- Interquartile Range

```
# ----------- (1) Finding the Outliers
def find_outliers_IQR(df):
    q1=df.quantile(0.25)
    q3=df.quantile(0.75)
    IQR=q3-q1
    outliers = df[((df<(q1-1.5*IQR)) | (df>(q3+1.5*IQR)))]
    return outliers
```

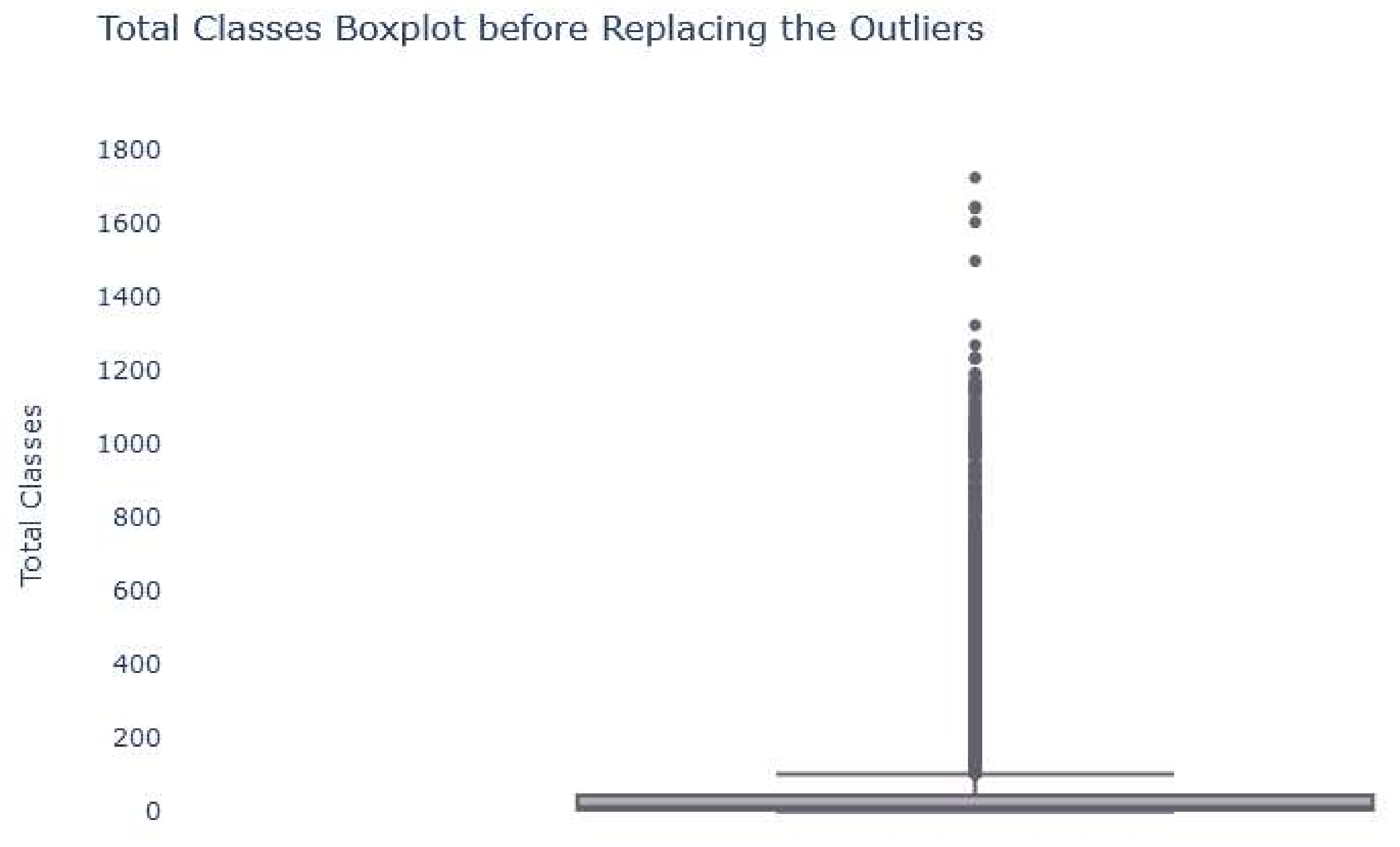
- Print min and max values

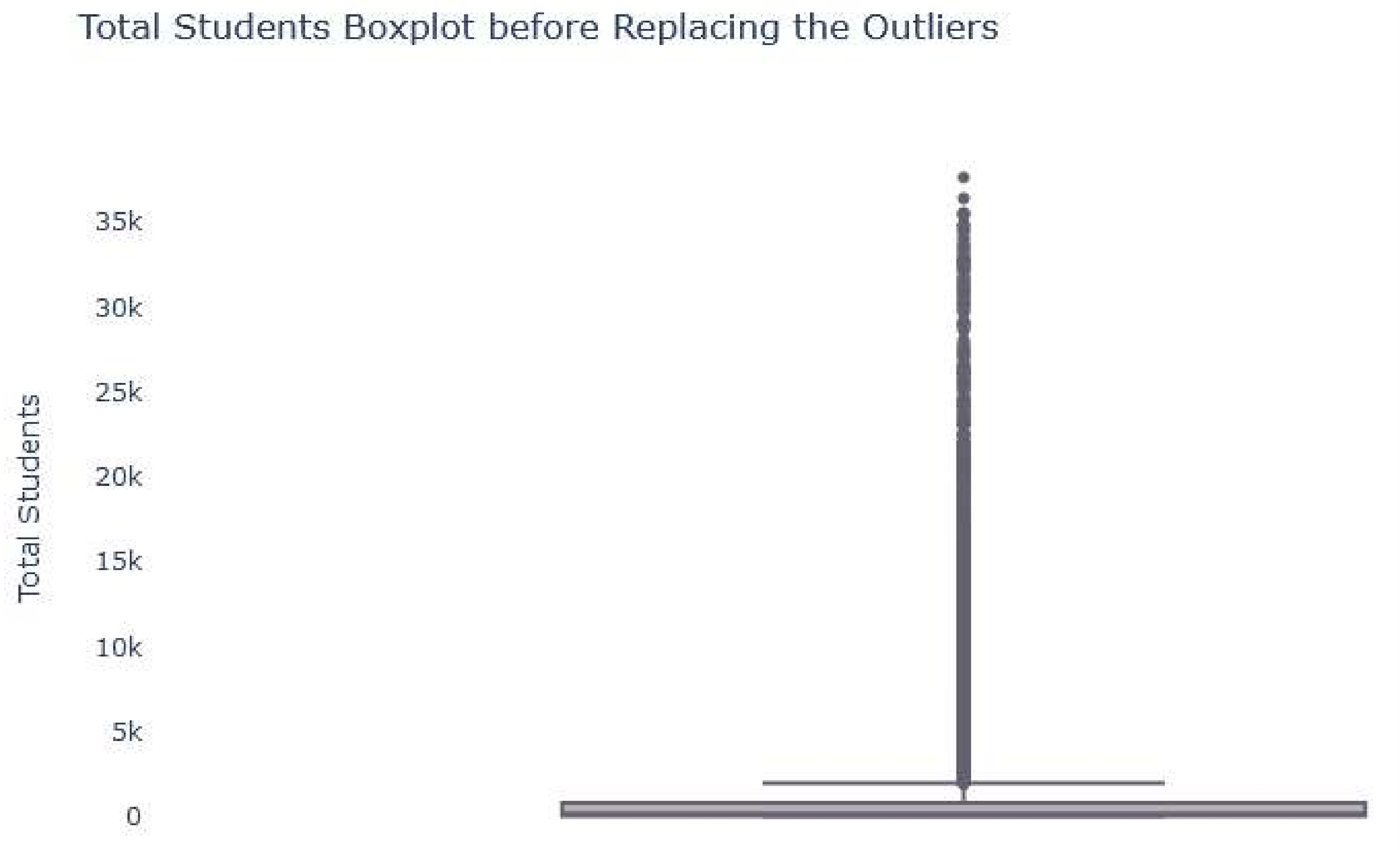
```
number of outliers in Total_Classes :
number of outliers:  5848
max outlier value:1726
min outlier value: 102

number of outliers in Total_Students :
number of outliers:  5656
max outlier value:37657
min outlier value: 1982
```

4 Machine Learning Models

4.4 Outliers Identification

- Visualization
  - Box Plot





4 Machine Learning Models

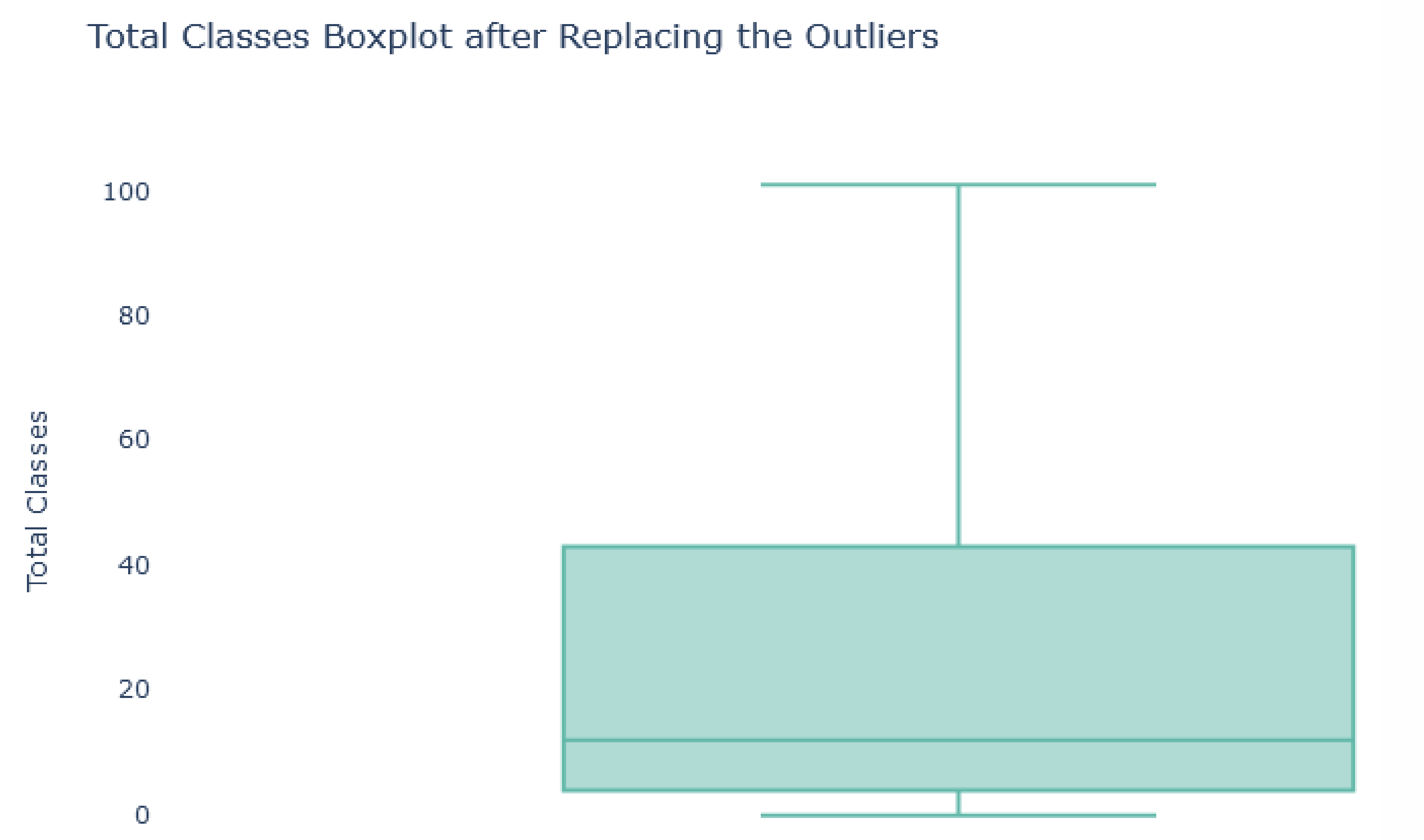
4.4 Outliers Treatment

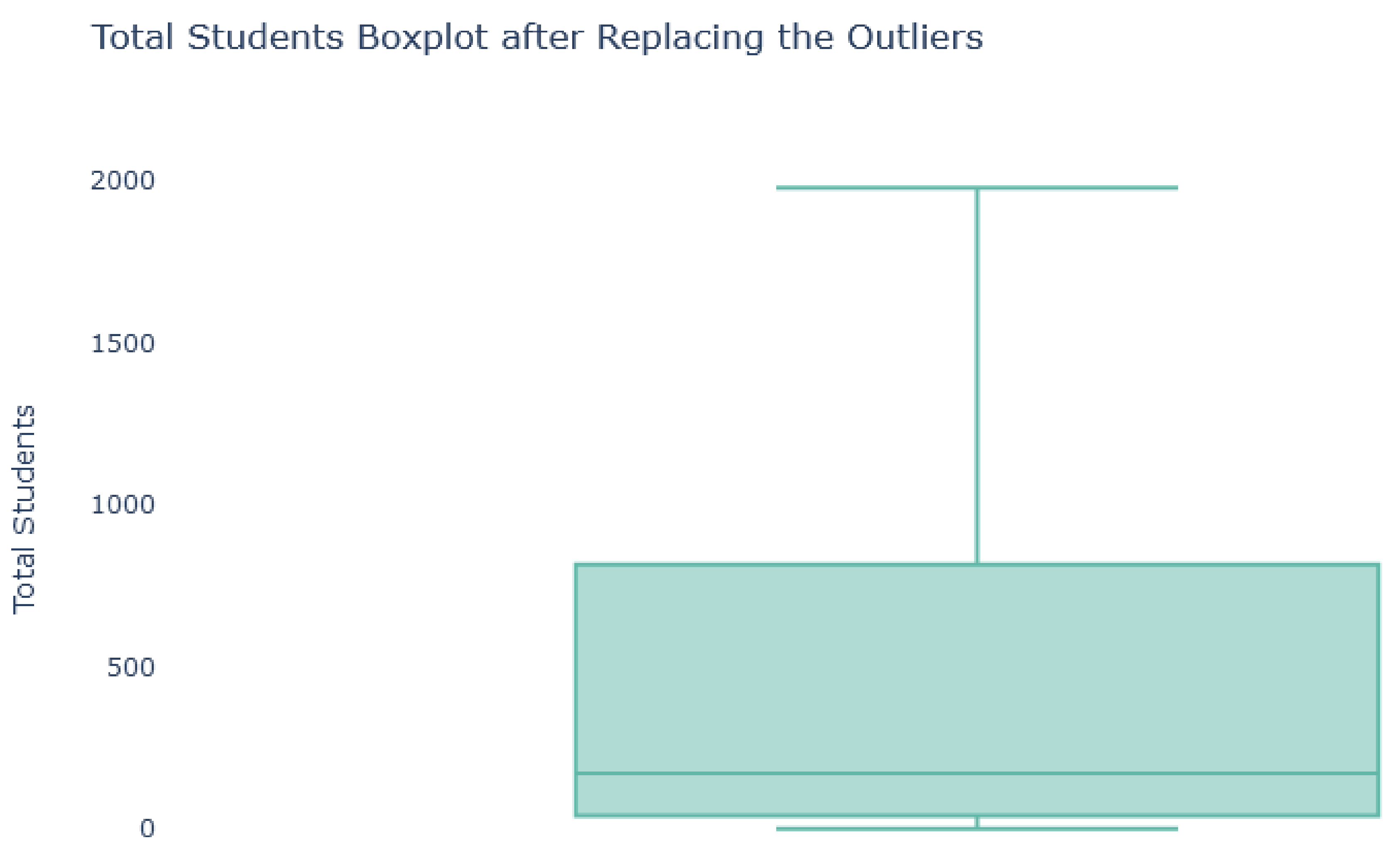
- imputation technique

```
def impute_outliers_IQR(df):
    q1=df.quantile(0.25)
    q3=df.quantile(0.75)
    IQR=q3-q1
    upper = df[~(df>(q3+1.5*IQR))].max()
    lower = df[~(df<(q1-1.5*IQR))].min()
    df = np.where(df > upper,
       df.mean(),
       np.where(
           df < lower,
           df.mean(),
           df
           )
       )
    return df
```

4 Machine Learning Models

4.4 Outliers Treatment





4 Machine Learning Models

4.5 Building Our Regression Models

4.5.1 Find the Target

```
# add coluomn to calculate avarage of student per class
education_system_df["Avg_Student_Per_Class"] = (round(
    education_system_df["Total_Students"] / education_system_df["Total_Classes"]))

education_system_df
```

_Type_Religious (Koranic)	School_Type_Special education	School_Gender_Boys	School_Gender_Girls	School_System_General system	School_System_curriculum system	Avg_Student_Per_Class
0	0	1	0	1	0	4.0
1	0	1	0	1	0	7.0
0	1	1	0	1	0	2.0
0	0	1	0	1	0	9.0
0	1	1	0	1	0	2.0
...	...	...	...	...	...	...
0	0	0	1	1	0	NaN
0	0	0	1	0	1	23.0
1	0	0	1	1	0	22.0
0	0	0	1	1	0	24.0
0	0	0	1	1	0	14.0

4 Machine Learning Models

4.5 Building Our Regression Models

4.5.1 Clean the Target

```
#change infinty to null
education_system_df["Avg_Student_Per_Class"].replace([np.inf, -np.inf], np.nan, inplace=True)
```

```
#after replaced calculate nulls in students Per Class
education_system_df["Avg_Student_Per_Class"].isna().sum()
```

```
386
```

```
education_system_df = education_system_df.dropna()
```

```
#after replaced calculate nulls in students Per Class
education_system_df["Avg_Student_Per_Class"].isna().sum()
```

```
0
```

4 Machine Learning Models

4.5 Building Our Regression Models

4.5.2 Feature Engineering

- Categorical Features to Dummy/Indicators

```
# create dummies
education_system_df = pd.get_dummies(education_system, columns=['Province','Authority','Educational_Level',
                                                                'School_Type', 'School_Gender','School_System'])
```

```
education_system_df.head()
```

Province_Al Baha	Province_Al Jouf	Province_Al Madinah	Province_Al Qassim	...	Educational_Level_Kindergarten	Educational_Level_Middle	School_Type_Adult Education	School_Type_Day	Scho
1	0	0	0	...	0	0	1	0	
1	0	0	0	...	0	0	0	0	
1	0	0	0	...	0	0	0	0	
1	0	0	0	...	0	0	0	1	
1	0	0	0	...	0	0	0	0	

4 Machine Learning Models

4.5 Building Our Regression Models

4.5.3 Split Data

```
# select target from our data by avarage student per class
Target_var = "Avg_Student_Per_Class"
Target = education_system_df[Target_var]
Features = education_system_df.drop(Target_var, axis=1)
```

```
# assgin the feature set and the target for both the train and the test
X_train, X_test, y_train, y_test = train_test_split(Features, Target, test_size=0.2, random_state=55)
```

```
X_train.shape
```

```
(31605, 35)
```

```
y_train.shape
```

```
(31605,)
```

4.5.4 Calculate the Baseline

```
Baseline
MSE: 209
MAE: 8
RMSE: 14
```

4 Machine Learning Models

4.5 Building Our Regression Models

4.5.5 Regression Models

Define

```
# create differnt models array for test and select the best model
models = []
models.append(('KNN', KNeighborsRegressor()))
models.append(('MLP',MLPRegressor()))
models.append(('CART', DecisionTreeRegressor()))
models.append(('RF', RandomForestRegressor()))
models.append(("LR", LinearRegression()))
```

Fit

```
# create model score array to store the retrive score from for loop
model_scores =[]
# create for loop toe apply the models and calculate the cost function for each model
for name, model in models:
    # first step: fitting the X_train and y_tain in the model
        model.fit(X_train, y_train)
```

Predict

```
# second step: apply the prediction on X_test
y_pred = model.predict(X_test)
```

4 Machine Learning Models

4.5 Building Our Regression Models

4.5.6 Regression Models Evaluation

```
Model Name: KNN
MSE: 97
MAE: 2
RMSE: 10
The score : 50.55
* ** ** ** ** ** ** ** ** ** ** ** ** ** ** *

Model Name: MLP
MSE: 33
MAE: 3
RMSE: 6
The score : 83.19
* ** ** ** ** ** ** ** ** ** ** ** ** ** ** *

Model Name: RF
MSE: 14
MAE: 0
RMSE: 4
The score : 92.82
* ** ** ** ** ** ** ** ** ** ** ** ** ** ** *
```

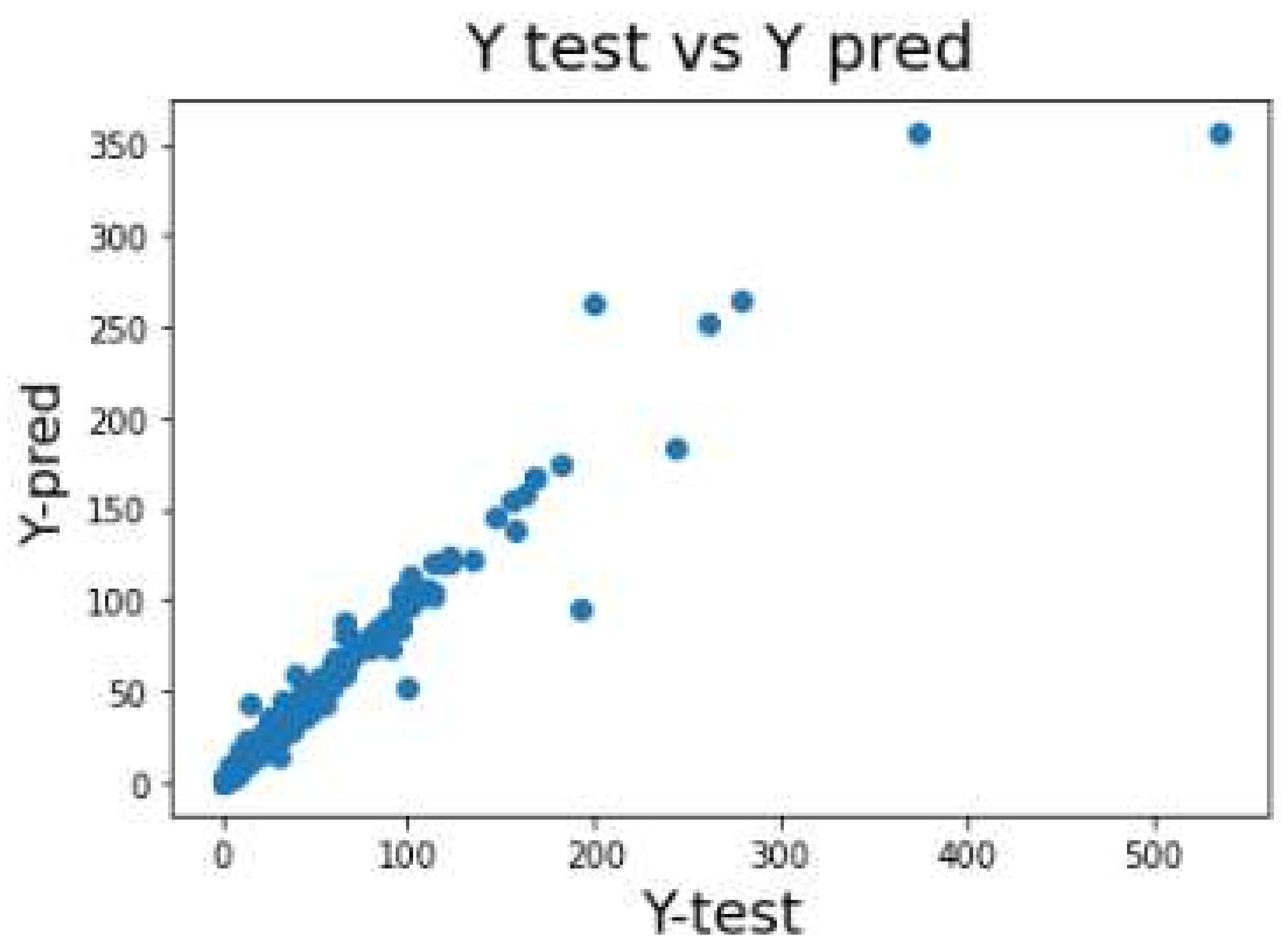
```
Model Name: CART
MSE: 12
MAE: 0
RMSE: 4
The score : 93.69
* ** ** ** ** ** ** ** ** ** ** ** ** ** ** *

Model Name: LR
MSE: 117
MAE: 4
RMSE: 11
The score : 40.19
* ** ** ** ** ** ** ** ** ** ** ** ** ** ** *
```

4 Machine Learning Models

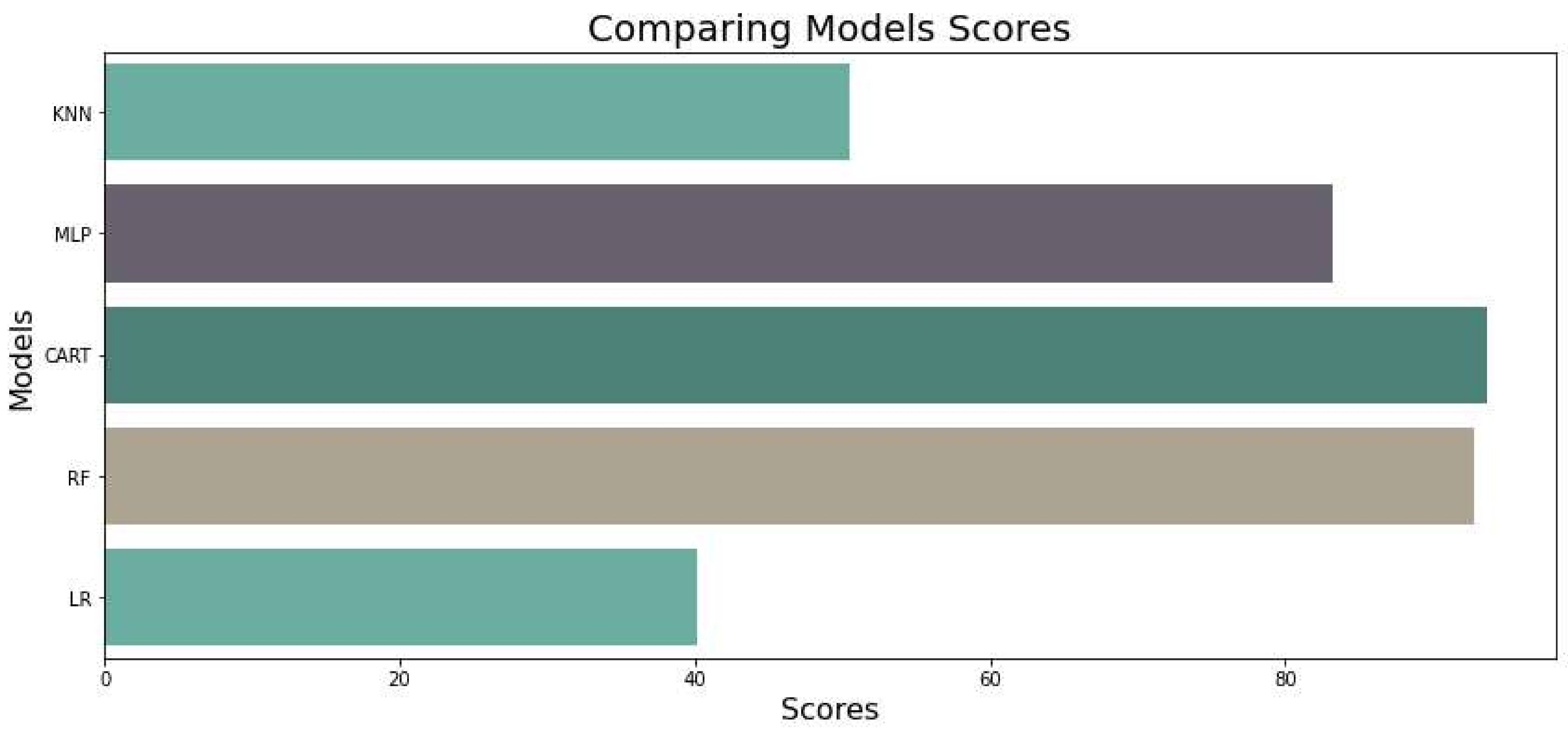
4.5 Building Our Regression Models

4.5.7 Regression Models Evaluation

```
Model Name: CART
MSE: 12
MAE: 0
RMSE: 4
The score : 93.69
* ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** *
```



4.5.8 Model Selection



4 Machine Learning Models

4.5 Building Our Regression Models

4.5.9 Regression Models Optimization

Model Tuning

CART model tuning

```
In [50]: cart_params = {"max_depth": [2,3,4,5,6,8,10,20,30,50, 100, 500, 1000,5000,10000],
                        "min_samples_split": [2,5,10,20,30,50,100,500,1000,5000,10000]}

In [51]: cart_model = DecisionTreeRegressor()

In [52]: cart_cv_model = GridSearchCV(cart_model, cart_params, cv = 10).fit(X_train, y_train)

In [53]: cart_cv_model.best_params_
Out[53]: {'max_depth': 30, 'min_samples_split': 2}

In [54]: cart_tuned = DecisionTreeRegressor(**cart_cv_model.best_params_).fit(X_train, y_train)

In [55]: y_pred = cart_tuned.predict(X_test)
         cart_tuned_score = np.sqrt(mean_squared_error(y_test, y_pred))
         cart_tuned_score
Out[55]: 2.6877145540048564

In [62]: print('The score :', round(r2_score(y_true=y_test, y_pred=y_pred)*100,2),'%')
         The score : 96.31 %
```

4 Machine Learning Models

4.5 Building Our Regression Models

4.5.10 Regression Models Pipeline

```
X_train_n = X_train.select_dtypes(exclude=["category", "object"])
X_test_n = X_test.select_dtypes(exclude=["category", "object"])

# Create a Pipeline for our model
pipe = make_pipeline(
    # scale columns
    StandardScaler(),
    # apply the model
    DecisionTreeRegressor() #our best model
)

pipe.fit(X_train_n,y_train)
print(round(pipe.score(X_test_n, y_test)*100,2),'%')
```

```
96.52 %
```

4 Machine Learning Models

4.6 Building Our Classification Models

4.6.1 Feature Engineering

```
education_system_df['Avg_Student_Per_Class']=np.where(education_system_df['Avg_Student_Per_Class'] > 22,1, 0)
```

```
education_system_df
```

_Type_Religious (Koranic)	School_Type_Special education	School_Gender_Boys	School_Gender_Girls	School_System_General system	School_System_curriculum system	Avg_Student_Per_Class
0	0	1	0	1	0	1
1	0	1	0	1	0	1
0	1	1	0	1	0	1
0	0	1	0	1	0	1
0	1	1	0	1	0	1
...	...	...	...	...	...	...
1	0	0	1	0	1	0
0	0	0	1	0	1	0
1	0	0	1	1	0	1
0	0	0	1	1	0	0
0	0	0	1	1	0	1

4 Machine Learning Models

4.6 Building Our Classification Models

4.6.2 Split Data

```
X = education_system_df.drop('Avg_Student_Per_Class', axis=1)
y = education_system_df['Avg_Student_Per_Class']

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=42)

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.fit_transform(X_test)

print("Shape of training set:", X_train.shape)
print("Shape of test set:", X_test.shape)
```

```
Shape of training set: (31605, 35)
Shape of test set: (13545, 35)
```

4.6.3 Calculate the Baseline

```
1    0.828948
0    0.171052
Name: Avg_Student_Per_Class, dtype: float64
```

4 Machine Learning Models

4.6 Building Our Classification Models

4.6.4 Other Classification Models

Define

```
classic_models = []
classic_models.append(('KNN', KNeighborsClassifier(n_neighbors=5)))
classic_models.append(('LOGISTIC',LogisticRegression(solver='lbfgs', max_iter=500)))
classic_models.append(('CART', DecisionTreeRegressor()))
classic_models.append(('RF',RandomForestClassifier(n_estimators=100)))
classic_models.append(("NAVIE",GaussianNB()))
```

Fit

```
model_accurecy =[]
for name, model in classic_models:
        model.fit(X_train, y_train)
```

Predict

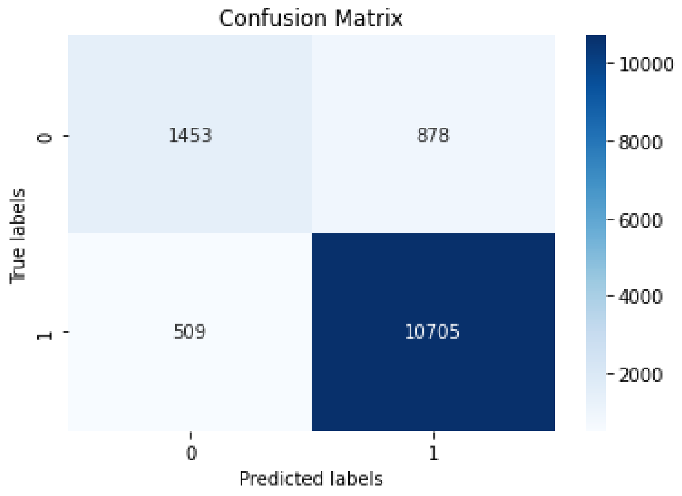
```
y_pred = model.predict(X_test)
```

4 Machine Learning Models

4.6 Building Our Classification Models

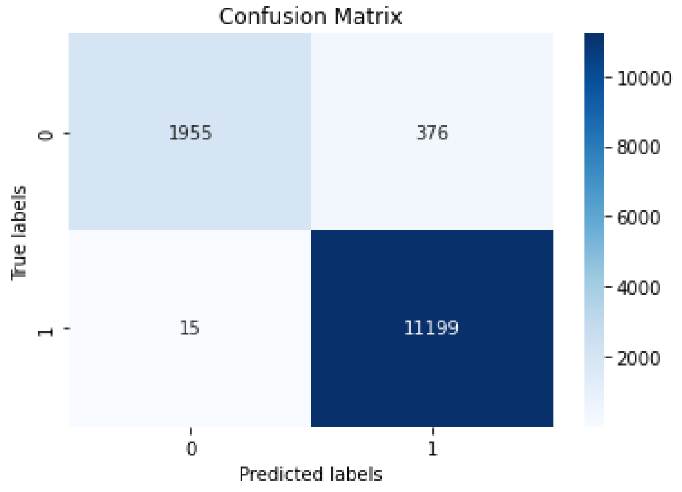
4.6.5 Classification Models Evaluation

```
Model Name: KNN
Confusion Matrix:
 [[ 1453   878]
 [  509 10705]]
Accurecy: 89.76
              precision    recall  f1-score   support

           0       0.74      0.62      0.68      2331
           1       0.92      0.95      0.94     11214

    accuracy                           0.90     13545
   macro avg       0.83      0.79      0.81     13545
weighted avg       0.89      0.90      0.89     13545
```



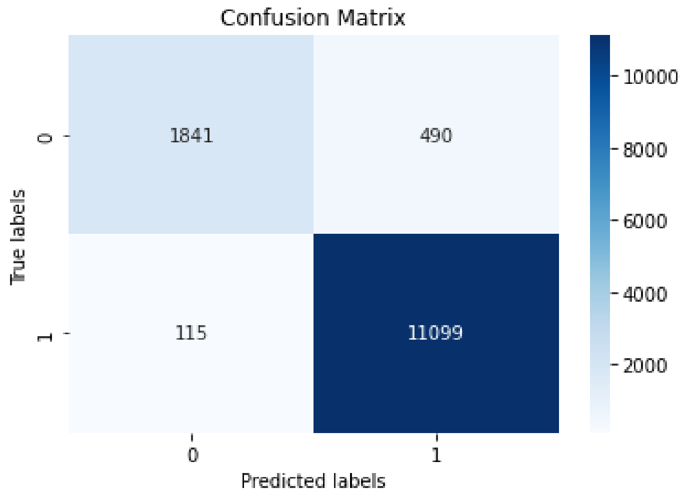

```
* ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** *
```

```
Model Name: LOGISTIC
Confusion Matrix:
 [[ 1955   376]
 [   15 11199]]
Accurecy: 97.11
              precision    recall  f1-score   support

           0       0.99      0.84      0.91      2331
           1       0.97      1.00      0.98     11214

    accuracy                           0.97     13545
   macro avg       0.98      0.92      0.95     13545
weighted avg       0.97      0.97      0.97     13545
```



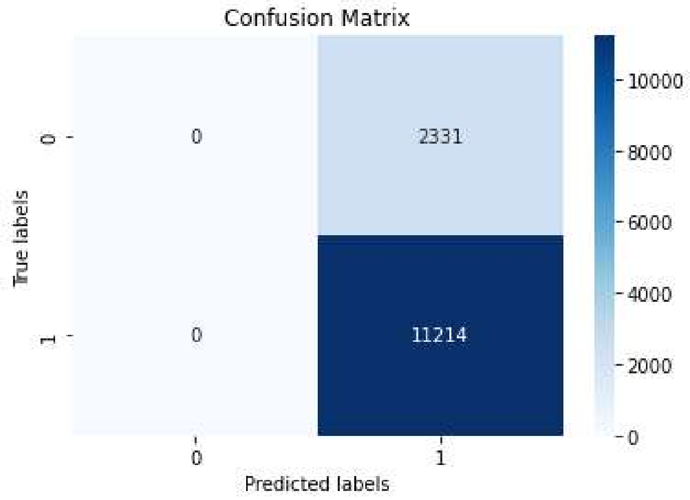

```
* ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** *
```

```
Model Name: RF
Confusion Matrix:
 [[ 1841   490]
 [  115 11099]]
Accurecy: 95.53
              precision    recall  f1-score   support

           0       0.94      0.79      0.86      2331
           1       0.96      0.99      0.97     11214

    accuracy                           0.96     13545
   macro avg       0.95      0.89      0.92     13545
weighted avg       0.95      0.96      0.95     13545
```



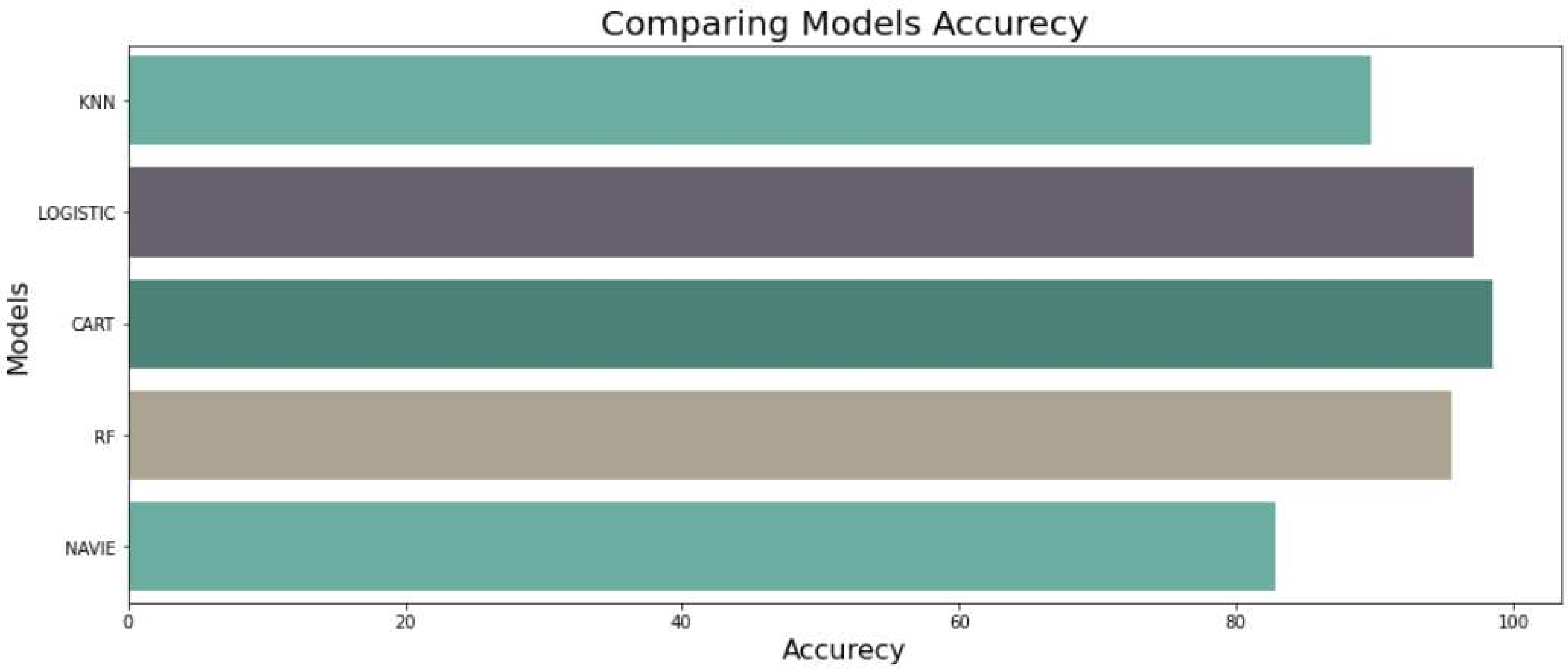

```
* ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** *
```

```
Model Name: NAVIE
Confusion Matrix:
 [[    0  2331]
 [    0 11214]]
Accurecy: 82.79
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      2331
           1       0.83      1.00      0.91     11214

    accuracy                           0.83     13545
   macro avg       0.41      0.50      0.45     13545
weighted avg       0.69      0.83      0.75     13545
```




```
* ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** *
```

4.6.7 Model Selection

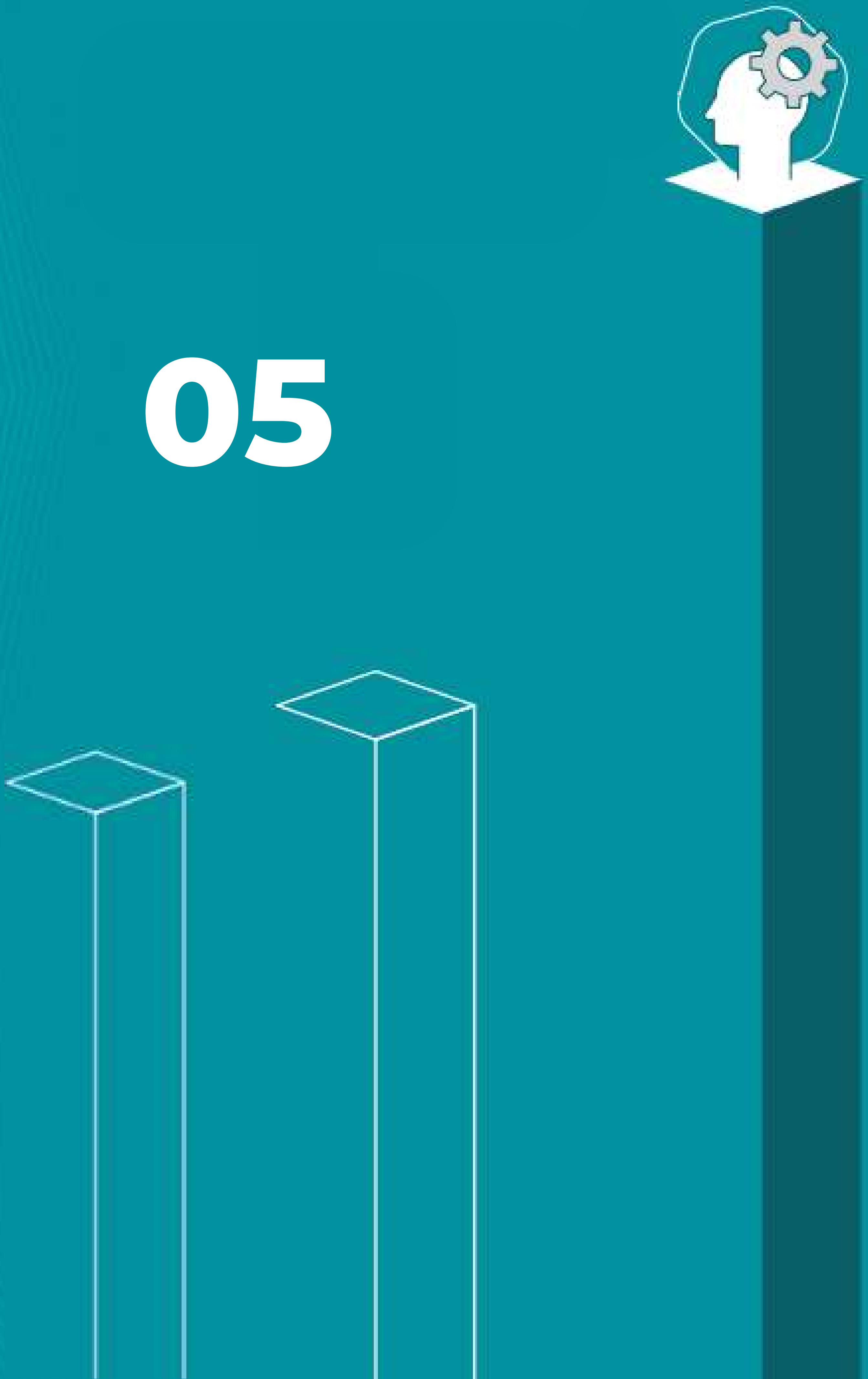


4 Machine Learning Models

4.6 Building Our Classification Models

4.6.9 Classification Models Pipeline

```
X_train_n = X_train
X_test_n = X_test

# Create a Pipeline for our model
pipe = make_pipeline(
    # 1st step handle missing values
    SimpleImputer(), # Impute missing values
    # scale columns
    StandardScaler(),
    # apply the model
    DecisionTreeRegressor()
)

pipe.fit(X_train_n,y_train)
round(pipe.score(X_test_n, y_test)*100,2)
```

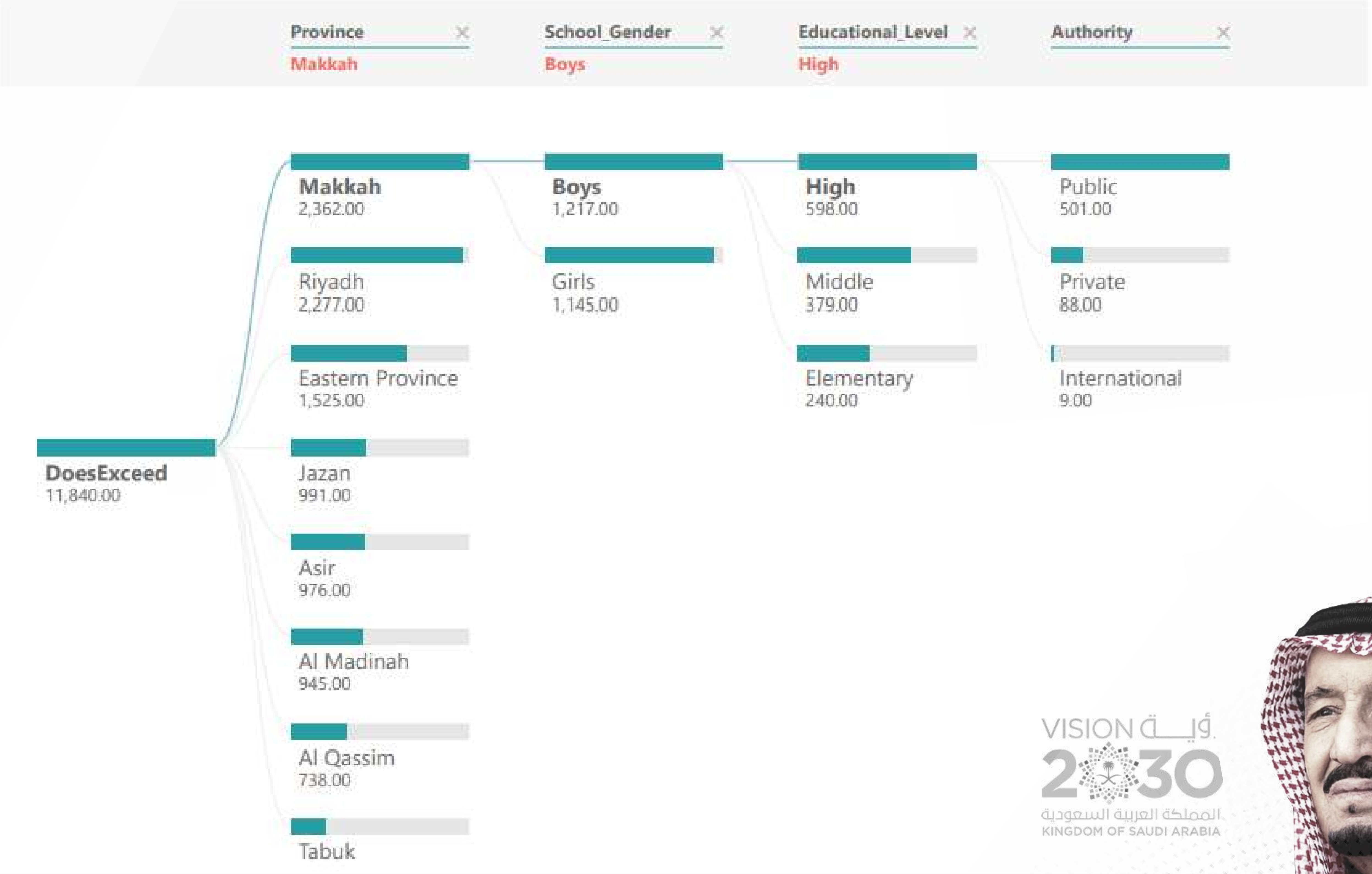


05

Power BI Dashboard

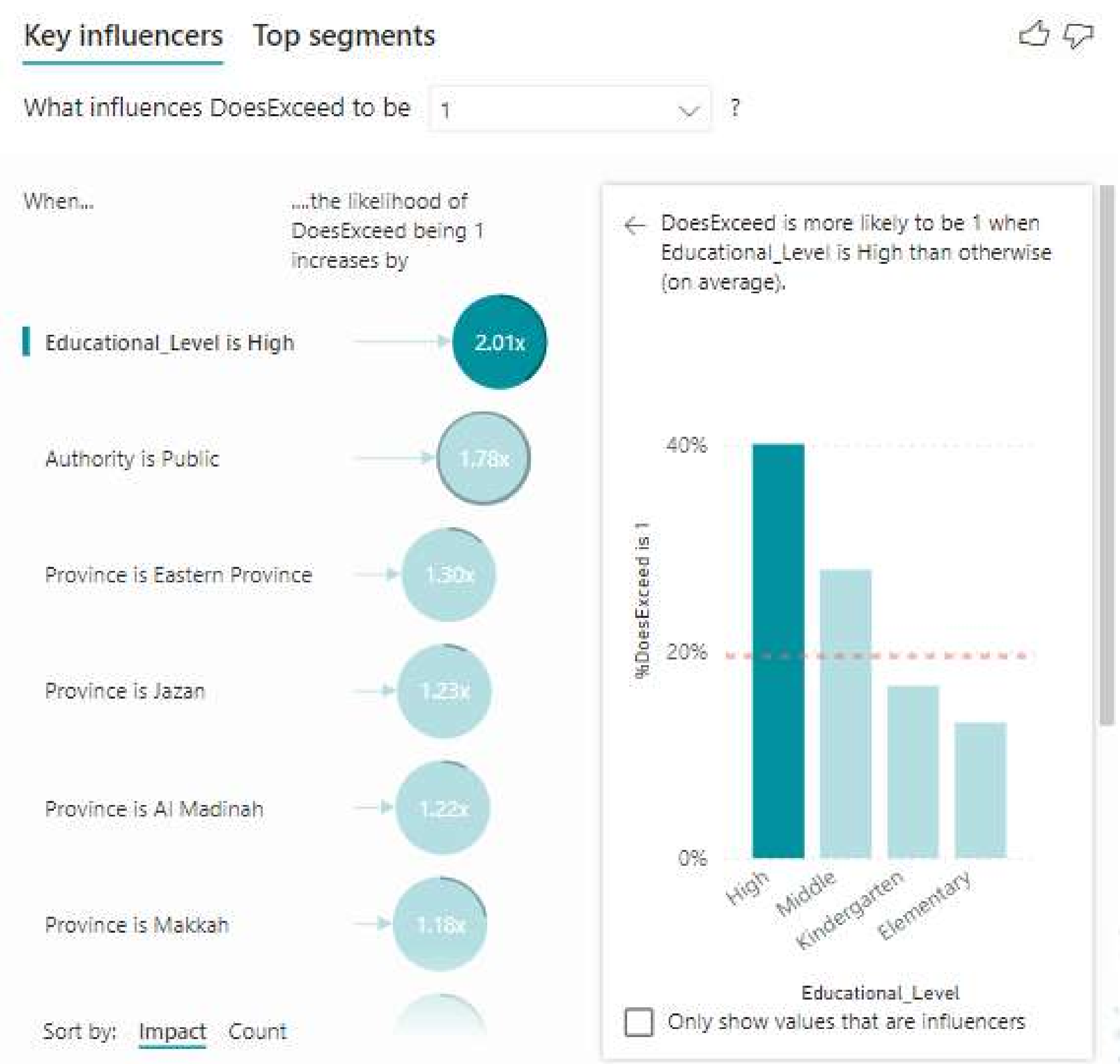
Average Students per Class Analysis

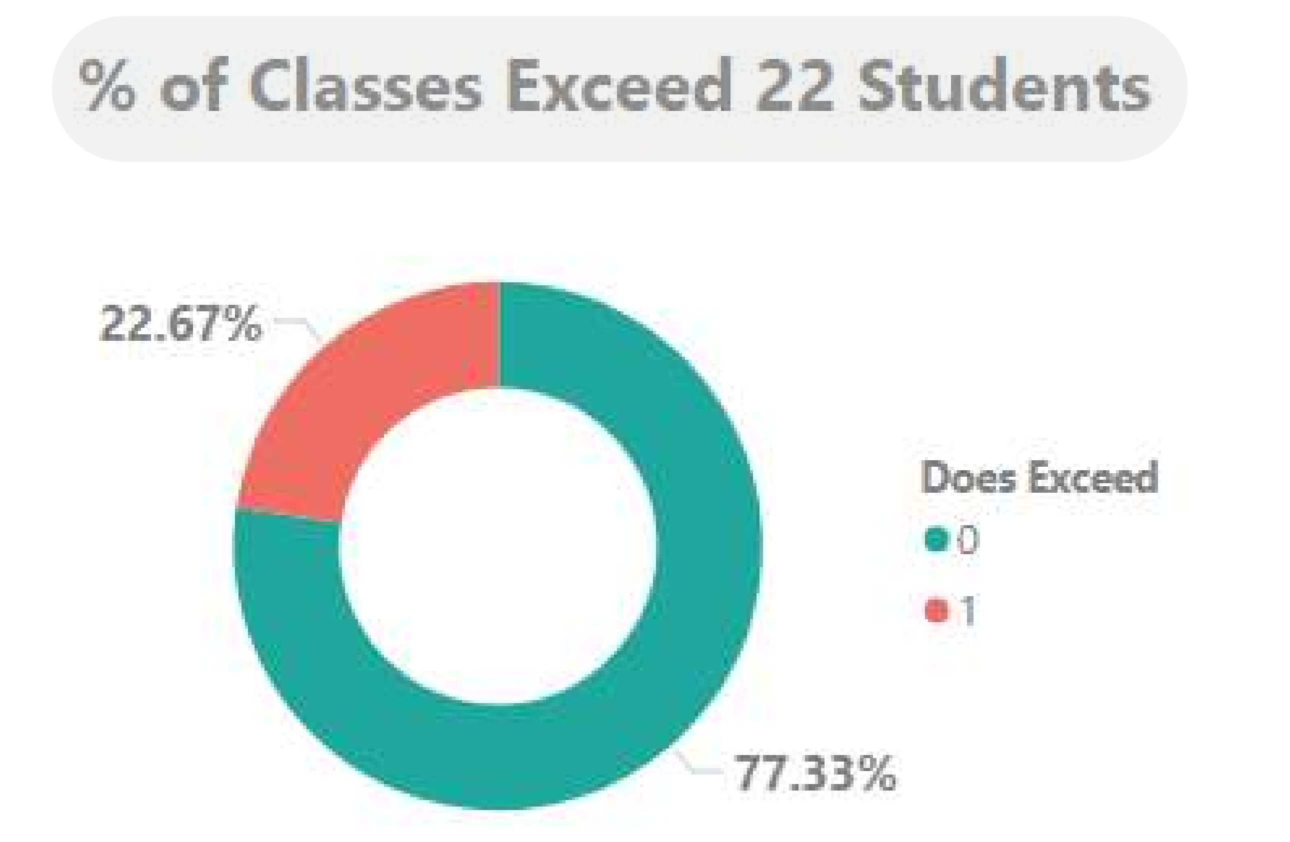
Decomposition



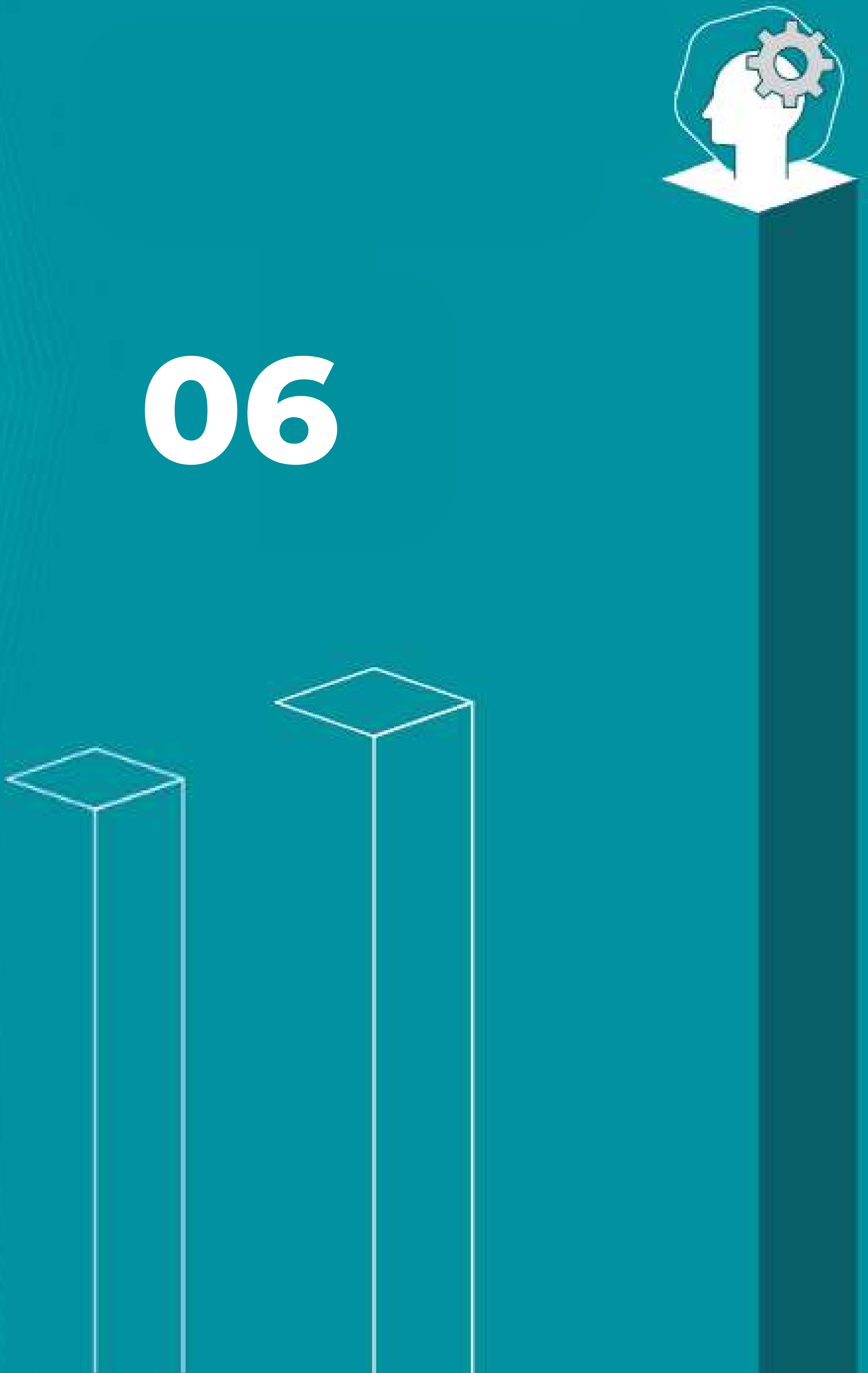
Average Students per Class Analysis

What Influences The Average to Increases





06

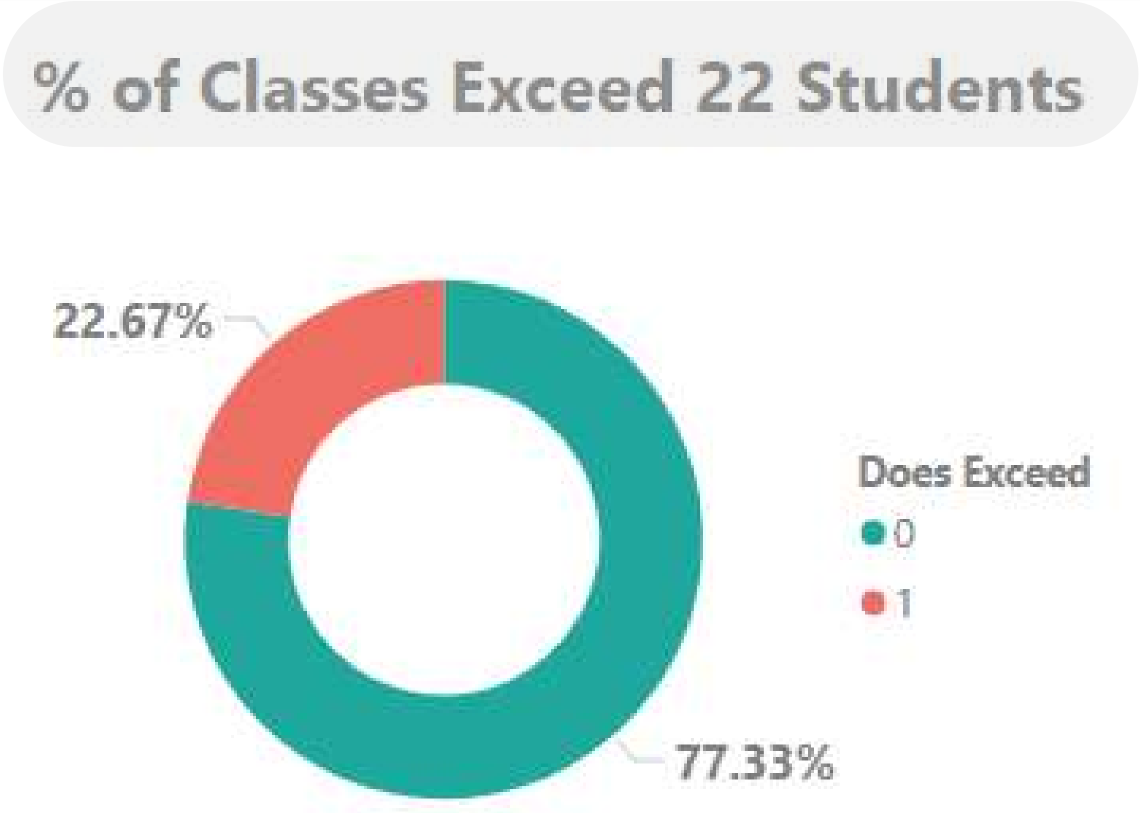


Findings and Recommendations

6 Findings and Recommendations

6.1 Findings

- 77.33% less than or equal to the targeted number (22)
- 22.7% more than the targeted number (22)



We found out that (by 96% of certainty) we can reach the targeted number of student per class, namely (22) by the end of 2025

6 Findings and Recommendations

6.2 Recommendations

Reduce the percentage of schools with average number of students above 22

- From 22.7% to 15% by the end of 2023
- From 15% to 7.5 % by the end of 2024
- From 7.5 % to 0% by the end of 2025

VISION رؤية 2030
المملكة العربية السعودية
KINGDOM OF SAUDI ARABIA

5 Reference

1- Saudi Open Data Portal
https://data.gov.sa/Data/en/dataset/2014-2021/resource/03cced36-a608-49ba-85ad-3c8e8d9a4984

2- Vision Realization Programs
https://www.vision2030.gov.sa/ar/

3- Ministry of Education
https://moe.gov.sa/ar/knowledgecenter/dataandstats/Pages/educationindicators.aspx

4- General Authority for Statistics
https://database.stats.gov.sa/beta/dashboard/indicator/410

5- Trading Economics - Saudi Arabia - School Enrollment, Preprimary (% Gross)
https://tradingeconomics.com/saudi-arabia/school-enrollment-preprimary-percent-gross-wb-data.html

THANK YOU!

Have a nice day