**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**НАВЧАЛЬНО-НАУКОВИЙ КОМПЛЕКС**
**«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»**
**НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ**
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»**
**КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ**

# Лабораторна робота №3
## з курсу: «Аналіз економічних та фінансових ризиків»

Виконали:
Байбара Ангеліна
Костенко Максим
Калініченко Назар
Ципаренко Ілля

Прийняла:
Гуськова В.Г.

Київ-2022 р.

Базуючись на лабораторній роботі 1 та лабораторній роботі 2 для обраних початкових даних побудувати прогноз (за **одним** із наведених підходів) при розбитті вибірки на навчальну та перевірочну для співвідношеннь 50/50, 60/40, 70/30, 80/20 та 90/10.

Заміна середнім значенням та експоненційне зглажування.

Прогнозування регресійною моделю.

Програма реалізована на Python.

Пропуски

Таблиця результатів

| r2 | dw | Se2 | mse | mae | mape | theil | |
|---|---|---|---|---|---|---|---|
| | | | | | | | **50/50** |
| 0.4062735 | 1.1319398 | 27291977 | 1186.6077 | 20.261453 | 7.1215678 | 0.06424901 | Без заповнення пропусків |
| 0.4863236 | 1.1664948 | 39273243 | 1240.3513 | 20.860344 | 7.0693135 | 0.089848404 | Із заповненням пропусків |
| 0.3759309 | 1.1498327 | 31806735 | 1272.2694 | 20.153169 | 7.0399882 | 0.064455189 01003 | Без фільтрації |
| 0.1765133 | 0.6203582 | 12575214 | 546.74842 | 13.822399 | 0.3731905 | 0.087574840 3884 | З фільтрацією |
| | | | | | | | 60/40 |
| 0.3938066 | 1.1260611 | 21567255 | 1172.1334 | 20.185592 | 7.0205056 | 0.082341047 | Без заповнення пропусків |
| 0.490557832 | 1.179184012 | 31936793.54 | 1260.779 | 20.864205 | 6.9298743 | 0.073866941 | Із заповненням пропусків |
| 0.4002555 | 1.1444149 | 24453126 | 1222.6563 | 20.401716 | 7.1945932 | 0.067520268 | фільтрації |
| 0.1682215 | 0.615388 | 9946715.5 | 540.58237 | 13.712345 | 0.3704417 | 0.066996776 | фільтрацією |
| | | | | | | | 70/30 |
| 0.4009141 | 1.1266019 | 16186511 | 1172.8506 | 20.421096 | 7.107729 | 0.066954402 | Без заповнення пропусків |
| 0.4805217 | 1.1818496 | 24126641 | 1269.9569 | 20.963002 | 7.1450711 | 0.060784749 | Із заповненням пропусків |
| 0.3872536 | 1.16205 | 19019608 | 1267.8893 | 20.600378 | 7.6863136 | 0.050616471 | фільтрації |
| 0.1653843 | 0.6224595 | 7727026.7 | 559.8889 | 13.857726 | 0.3722068 | 0.045616716 | фільтрацією |
| | | | | | | | 80/20 |
| 0.4035334 | 1.116884 | 10460540 | 1137.0153 | 20.061986 | 7.1950467 | 0.039435863 | Без заповнення пропусків |
| 0.4578803 | 1.1724376 | 16455228 | 1299.1653 | 21.055643 | 7.5183609 | 0.035312534 | Із заповненням пропусків |
| 0.3890894 | 1.1698186 | 12842690 | 1284.269 | 20.830372 | 7.8800973 | 0.031680179 | фільтрації |
| 0.1593171 | 0.6059561 | 4981180.3 | 541.43265 | 13.70167 | 0.3643254 | 0.030466984 | фільтрацією |
| | | | | | | | 90/10 |
| 0.3586613 | 1.131696 | 5731447.6 | 1245.9669 | 20.205628 | 6.967652 | 0.017612533 544997 | Без заповнення пропусків |
| 0.4283132 | 1.1570218 | 8369386.3 | 1321.5516 | 21.080936 | 6.584692 | 0.017680178 7360411 | Із заповненням пропусків |
| 0.3461852 | 1.1768774 | 6586719.5 | 1317.3439 | 20.989418 | 8.0499906 | 0.017666984 1895704 | Без фільтрації |
| 0.1751971 | 0.5901442 | 2442446 | 530.96651 | 13.53085458 | 0.3516719 | 0.017622109 | фільтрацією |

# Лістинг

```
import pandas as pd
import sys
import os
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from sklearn import metrics
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeClassifier
from statsmodels.stats.stattools import durbin_watson
#from aif360.metrics import ClassificationMetric
from statsmodels.tsa.api import ExponentialSmoothing, SimpleExpSmoothing, Holt
```

```
from sklearn.model_selection import train_test_split

df = pd.read_csv("Data/travel_insurance.csv")
df = df[:50000]
df.head()
df["Distribution Channel"] = np.where(df["Distribution Channel"] == "Offline", 0., 1.)
df["Agency Type"] = np.where(df["Agency Type"] == "Travel Agency", 0., 1.)
df["Claim"] = np.where(df["Claim"] == "No", 0., 1.)
df['Age'] = df['Age'].astype(float)
df.head()

#df_final = pd.DataFrame(columns=['Partition','R2','DW','E2','MSE','MAE','MAPE','Theil'])

X_NetSales = df.loc[:,df.columns.drop(['NetSales'])]
y_NetSales = df.NetSales
X_NetSales.head()
for i in range(5,10):
    print(i/10)
```

```
    X_train, X_test, y_train, y_test = train_test_split(X_NetSales, y_NetSales, test_size=1-i/10,
            random_state=42)
    regressor = LinearRegression()
    regressor.fit(X_train, y_train) #training the algorithm
    #To retrieve the intercept:
    print(regressor.intercept_)

    #df_final = df_final.append([[i/10,i/10,i/10,i/10,i/10,i/10,i/10,i/10]], ignore_index = True)

    #For retrieving the slope:
    print(regressor.coef_)
    y_pred = regressor.predict(X_test)
    y_test.to_numpy()
    df_res = pd.DataFrame({'Actual': y_test.to_numpy().flatten(), 'Predicted': y_pred.flatten()})
    df_res



    #     df_final = df_final.append([[i/10,metrics.r2_score(y_test,
            y_pred) ,durbin_watson(y_test),np.sum(np.square(y_pred -y_test)),
    #                 metrics.mean_squared_error(y_test, y_pred), metrics.mean_absolute_error(y_test,
            y_pred),metrics.mean_absolute_percentage_error(y_test, y_pred),0]],
            columns=['Partition','R2','DW','SE2','MSE','MAE','MAPE','Theil'])
    print('R2:', metrics.r2_score(y_test, y_pred))
    print('DW:', durbin_watson(y_test))
    print('sum e 2:',np.sum(np.square(y_pred -y_test)))
    print('MSE:', metrics.mean_squared_error(y_test, y_pred))
    print('MAE:', metrics.mean_absolute_error(y_test, y_pred))

    print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
    print('MAPE:', metrics.mean_absolute_percentage_error(y_test, y_pred))



df_row
```

```
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
print('MAPE:', metrics.mean_absolute_percentage_error(y_test, y_pred))
print('R2:', metrics.r2_score(y_test, y_pred))
print('DW:', durbin_watson(y_test))
print('sum e 2:',np.sum(np.square(y_pred -y_test)))

#print('sum e 2:',ClassificationMetric(y_pred -y_test))
```