# Sieci neuronowe i sztuczna inteligencja – laboratorium 1

**10.03.2023**

**Monika Błyszcz, 236623**

## Zad 1.

W zadaniu można pominąć poniższe kolumny, bo nie zawierają danych numerycznych, istotnych do analizy statystycznej jak:

2. condition (categorical): name of condition
3. review (text): patient review
5. date (date): date of review entry

Kod implementujący:

```
import pandas
import numpy
from pandas.plotting import scatter_matrix
import matplotlib.pyplot as plt
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC

#Load dataset
dataset = pandas.read_table("./drugsComTest_raw.tsv")
dataset.drop('lp', inplace=True, axis=1)
dataset.drop('review', inplace=True, axis=1)
dataset.drop('date', inplace=True, axis=1)
dataset.drop('condition', inplace=True, axis=1)

#Enkoder
drugs = dataset.groupby("drugName")["drugName"].count().keys().to_numpy()
new_column = []
for row in dataset.loc[:, "drugName"]:
    new_column.append(numpy.where(drugs == row)[0][0])
dataset.insert(1, "drugNumber", new_column, allow_duplicates=True)
dataset.drop('drugName', inplace=True, axis=1)

#Shape
print(dataset.shape)

#Head
print(dataset.head(30))

#Descriptions
print(dataset.describe())

#Class distribution
print(dataset.groupby('drugNumber').size())

#Box and whisher plots
```

```python
dataset.plot(kind='box', subplots=True, layout=(2,2), sharex=False,
sharey=False)
plt.show()

#Histograms
dataset.hist()
plt.show()

#Scatter plot matrix
scatter_matrix(dataset)
plt.show()

#Split-out validation dataset
array =dataset.values
X = array[:, 1:2]
Y = array[:, 0]
validation_size =0.20
seed = 7
X_train, X_validation, Y_train, Y_validation =
model_selection.train_test_split(X, Y, test_size=validation_size,
random_state=seed)
scoring = 'accuracy'

#Spot Check Algorithms
models =[]
models.append(('LR', LogisticRegression(solver='liblinear',
multi_class='ovr')))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC(gamma='auto')))

#Evaluate each model in turn
results = []
names = []
for name, model in models:
    kfold = model_selection.KFold(n_splits=10, random_state=seed,
shuffle=True)
    cv_results = model_selection.cross_val_score(model, X_train, Y_train,
cv=kfold, scoring=scoring)
    results.append(cv_results)
    names.append(name)
    msg ="%s: %f (%f)" % (name, cv_results.mean(),cv_results.std())
    print(msg)

#Compare Algorithms
fig =plt.figure()
fig.suptitle('Algorithm Comparison')
ax =fig.add_subplot(111)
plt.boxplot(results)
ax.set_xticklabels(names)
plt.show()

#Make predictions on validation dataset
knn = KNeighborsClassifier()
knn.fit(X_train, Y_train)
predictions = knn.predict(X_validation)
print(accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation,predictions))
print(classification_report(Y_validation, predictions))
```

## Zad 2.

Należy użyć komendy pandas.read_table(). Wycinek kodu poniżej:

```
#Load dataset
dataset = pandas.read_table("./drugsComTest_raw.tsv")
```

## Zad 3.

Wynik dokładności wybranych algorytmów przedstawiono poniżej.

```
LR: 0.026365 (0.001919)
LDA: 0.025970 (0.001713)
KNN: 0.004324 (0.002524)
CART: 0.026806 (0.001789)
NB: 0.000372 (0.000259)
SVM: 0.027132 (0.001553)
```

Największy wynik dokładności uzyskała Maszyna Wektorów Nośnych(SVM).