Universitatea „Petrol – Gaze” din Ploiești

Calitate vinuri

Realizat de: Coșereanu Emanuel, INFO an 3, grupa 40322

Cuprins

1. Alegerea bazei de date și a modului de lucru 2
2. Variabile, parametrii, prelucrarea datelor 3
3. Tabele de centralizare a rezultatelor. Grafice 3
4. Cod sursă arhitectură RNA5r 5
5. Cod sursă arhitectură RNA10a 9
6. Concluzii 13

Lista de figuri

Fig. 1 - Grafic vinuri roșii folosind RNA5r la a doua rulare a rețelei 4
Fig. 2 - Grafic vinuri albe folosind RNA10a la a doua rulare a rețelei 4

Lista de tabele

Tabel 1 - Rezultate experimentale pentru RNA regresie – **vin roșu** 3
Tabel 2 - Rezultate experimentale pentru RNA regresie – **vin alb** 4

1. Alegerea bazei de date și a modului de lucru

Baza de date pe care am ales-o se numește „Wine Quality”. Las link-ul aici. În cadrul acestei baze de date se găsesc două fișiere, unul pentru vinuri albe, unul pentru vinuri roșii. Fiecare fișier are următorii parametrii pe care îi voi lăsa mai jos. Lângă aceștia voi lasă unitatea de măsură a lor, respectiv mulțimea de valori pentru valorile nominale.

Nume parametru	Unitatea de măsură	Interval valori pentru vinuri roșii	Interval valori pentru vinuri albe
1. Fixed acidity	g/L	4.6-15.9	3.8-14.2
2. Volatile acidity	g/L	0.12-1.58	0.08-1.1
3. Citric acid	g/L	0-1	0-1.66
4. Residual sugar	g/L	0.9-15.5	0.6-65.8
5. Chlorides	g/L	0.012-0.611	0.009-0.346
6. Free sulfur dioxide	mg/L	1-72	2-289
7. Total sulfur dioxide	mg/L	6-289	9-440
8. Density	g/mL	0.99007-1.00369	0.98711-1.03898
9. PH	-	2.74-4.01	2.72-3.82
10. Sulphates	g/L	0.33-2	0.22-1.08
11. Alcohol	% vol	8.4-14.9	8-14.2
12. Quality	-	0-10	0-10

Baza de date a vinurilor roșii numără un total de **1599 de înregistrări**, în schimb ce baza de date pentru vinuri albe însumează un total de **4898 de înregistrări**.

Tipul de model ales a fost de **regresie**. Aceasta are pe ultimul strat un singur neuron ce ne returnează o valoare numerică (motiv pentru care este și de regresie), în cazul nostru calitatea vinului.

Pentru fiecare RNA s-a utilizat *k-fold cross validation*, deoarece această metodă oferă cea mai complexă modalitate de împărțite a datelor în seturi de antrenare și de testare, împărțind datele în k seturi si luând pe rând 1 set pentru testare și k-1 seturi pentru antrenare.

În cadrul proiectului s-a urmărit **ca fiecare arhitectură** să aibă între 2 și 3 straturi ascunse cu cel puțin 10 neuroni fiecare, 100 de epoci și 4 k-fold-uri.

2. Variabile, parametrii, prelucrarea datelor

Parametrii statistici folosiți pentru evaluarea modelului au fost RMSE (Root Mean Squared Error) și Loss (eroarea pe setul de date de antrenare).

Variabila dependentă din cadrul fiecărei rețele neuronale este calitatea vinului (quality), iar variabilele independente sunt: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol.

În cadrul ambelor baze de date nu s-au găsit valori lipsă, respectiv nu au existat dificultăți sau provocări în cadrul prelucrării acesteia.

3. Tabele de centralizare a rezultatelor. Grafice

Nume Rețea	Stratul ascuns H1	Stratul ascuns H2	Stratul ascuns H3	Epoci	Early Stopping (ma)	RMSE	K-Folds	Eroare/Loss
RNA1r	25	10	-	100	44,8	**0.69493**	5	0.4855
RNA2r	25	10	-	200	76	0.71676	5	0.4853
RNA3r	15	20	10	150	24	0.70598	5	0.5219
RNA4r	15	20	10	100	41,28	0.70544	7	0.4905
RNA5r	30	35	30	100	26,3	0.69705	10	**0.4703**
RNA6r	25	35	-	500	27,25	0.72994	4	0.5438
RNA7r	15	20	-	250	37,67	0.70399	6	0.4847
RNA8r	35	40	-	300	23	0.70918	8	0.5118
RNA9r	25	30	25	1000	34	0.69587	5	0.4878
RNA10r	25	30	25	1000	25	0.70403	10	0.5070
RNA5r	*30*	*35*	*30*	*100*	*27,5*	*0.70735*	*10*	***0.4480***

Tabel 1 *- Rezultate experimentale pentru RNA regresie –* ***vin roșu***

Nume Rețea	Stratul ascuns H1	Stratul ascuns H2	Stratul ascuns H3	Epoci	Early Stopping (ma)	RMSE	K-Folds	Eroare/Loss
RNA1a	25	10	-	100	24,2	0.81711	5	0.5940
RNA2a	25	10	-	200	33,4	0.80046	5	0.6104
RNA3a	15	20	10	150	23	0.80074	5	0.6374
RNA4a	15	20	10	100	23,71	0.80803	7	**0.5848**

RNA5a	30	35	30	100	17,9	0.82468	10	0.6202
RNA6a	25	35	-	500	22	0.79238	4	**0.5848**
RNA7a	15	20	-	250	27,17	0.81499	6	0.6076
RNA8a	35	40	-	300	16,5	0.84611	8	0.6577
RNA9a	25	30	25	1000	15,4	0.80469	5	0.7168
RNA10a	25	30	25	1000	21,1	**0.78056**	10	0.5957
RNA10a	*25*	*30*	*25*	*1000*	*15,6*	*0.80863*	*10*	*0.6460*

Tabel 2** - Rezultate experimentale pentru RNA regresie – **vin alb

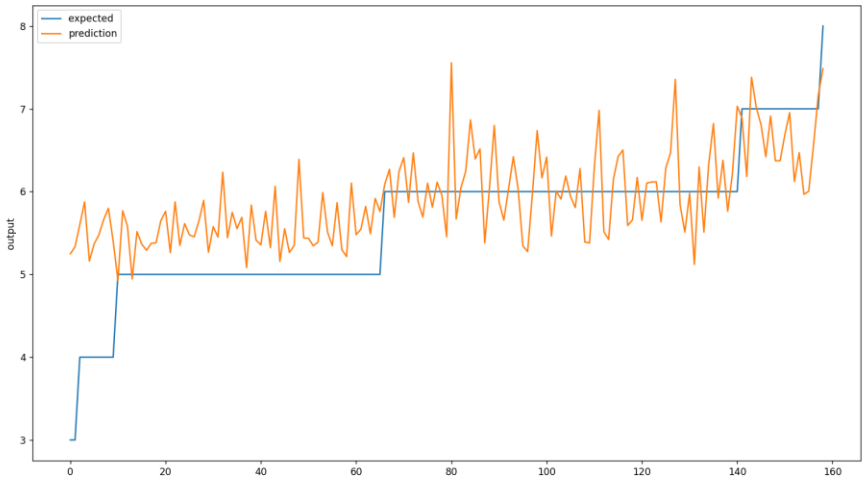

***Fig. 1** - Grafic vinuri roșii folosind RNA5r la a doua rulare a rețelei*

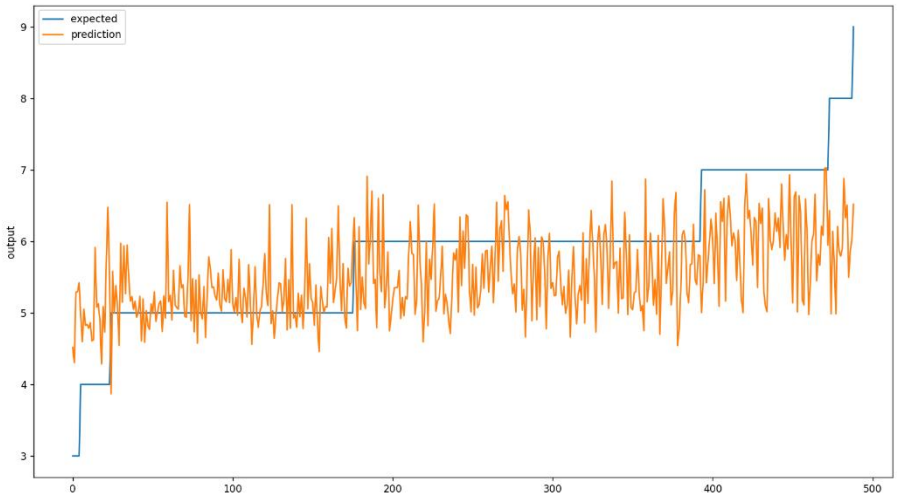

***Fig. 2** - Grafic vinuri albe folosind RNA10a la a doua rulare a rețelei*

4. Cod sursă arhitectură RNA5r

Mai jos va fii lăsat codul rețelei neuronale 5 a vinurilor roșii folosită la generarea graficului.

```
import pandas as pd
import os
import numpy as np
from sklearn import metrics
from keras import layers
from scipy.stats import zscore
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint
from sklearn.model_selection import KFold
from keras.models import Sequential
from keras.layers import Dense, Activation
import matplotlib.pyplot as plt


path = "./data/"
filename_read = os.path.join(path,"winequality-red.csv")
filename_write = os.path.join(path,"winequality-red-out-of-
sample.csv")
df = pd.read_csv(filename_read, sep =";") #nu avem valori lipsa,
daca aveam, puneam al doilea parametru - na_values=['NA','?']
# punem separatorul ; deoarece asa sunt separate datele in
fisier


# Shuffle
```

```
np.random.seed(42)
df = df.reindex(np.random.permutation(df.index))
df.reset_index(inplace=True, drop=True)

#nu avem string uri si nici valori nule
# Preprocess
# def missing_median(df, name):
#     med = df[name].median()
#     df[name] = df[name].fillna(med)
# cars = df['car name']
# df.drop('car name',axis=1,inplace=True)
# missing_median(df, 'horsepower')

dataset=df.values
x=dataset[:, 0:11]
y=dataset[:, 11]

# Cross-Validate
kf = KFold(10)

oos_y = []
oos_pred = []
fold = 0
for train, test in kf.split(x):
    fold+=1
    print("Fold #{}".format(fold))
```

```
    x_train = x[train]
    y_train = y[train]
    x_test = x[test]
    y_test = y[test]

    model = Sequential()
    model.add(layers.Input(shape=(x_train.shape[1],)))
    model.add(layers.Dense(30, activation='relu'))         #
Stratul ascuns 1
    model.add(layers.Dense(35, activation='relu'))         #
Stratul ascuns 2
    model.add(layers.Dense(30, activation='relu'))         #
Stratul ascuns 3
    model.add(layers.Dense(1))
    model.compile(loss='mean_squared_error', optimizer='adam')

    monitor = EarlyStopping(monitor='val_loss', min_delta=1e-3,
patience=5, verbose=1, mode='auto')

model.fit(x_train,y_train,validation_data=(x_test,y_test),callba
cks=[monitor],verbose=1,epochs=100)

    pred = model.predict(x_test)

    oos_y.append(y_test)
    oos_pred.append(pred)

    # Measure this fold's RMSE
    score = np.sqrt(metrics.mean_squared_error(pred,y_test))
```

```
    print("Fold score (RMSE): {}".format(score))

# Build the oos prediction list and calculate the error.
oos_y = np.concatenate(oos_y)
oos_pred = np.concatenate(oos_pred)
score = np.sqrt(metrics.mean_squared_error(oos_pred,oos_y))
print("Final, out of sample score (RMSE): {}".format(score))

# Write the cross-validated prediction
oos_y = pd.DataFrame(oos_y)
oos_pred = pd.DataFrame(oos_pred)
oosDF = pd.concat( [df, oos_y, oos_pred],axis=1 )
oosDF.to_csv(filename_write,index=False)

print(oosDF)

# Regression chart.
def chart_regression(pred, y, sort=True):
    t = pd.DataFrame({'pred': pred, 'y': y.flatten()})
    if sort:
        t.sort_values(by=['y'], inplace=True)
    plt.plot(t['y'].tolist(), label='expected')
    plt.plot(t['pred'].tolist(), label='prediction')
    plt.ylabel('output')
    plt.legend()
    plt.show()
```

```
# Plot the chart
chart_regression(pred.flatten(),y_test)


# RNA5r - rularea a doua (cu generare de graphic)
```

5. Cod sursă arhitectură RNA10a

Mai jos va fii lăsat codul rețelei neuronale 10 a vinurilor albe folosită la generarea graficului.

```
import pandas as pd
import os
import numpy as np
from sklearn import metrics
from keras import layers
from scipy.stats import zscore
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint
from sklearn.model_selection import KFold
from keras.models import Sequential
from keras.layers import Dense, Activation
import matplotlib.pyplot as plt

path = "./data/"
filename_read = os.path.join(path,"winequality-white.csv")
```

```
filename_write   =   os.path.join(path,"winequality-white-out-of-sample.csv")
df = pd.read_csv(filename_read, sep =";") #nu avem valori lipsa, daca aveam, puneam al doilea parametru - na_values=['NA','?']
# punem separatorul ; deoarece asa sunt separate datele in fisier

# Shuffle
np.random.seed(42)
df = df.reindex(np.random.permutation(df.index))
df.reset_index(inplace=True, drop=True)

#nu avem string uri si nici valori nule
# Preprocess
# def missing_median(df, name):
#     med = df[name].median()
#     df[name] = df[name].fillna(med)
# cars = df['car name']
# df.drop('car name',axis=1,inplace=True)
# missing_median(df, 'horsepower')

dataset=df.values
x=dataset[:, 0:11]
y=dataset[:, 11]

# Cross-Validate
kf = KFold(10)

oos_y = []
```

```
oos_pred = []
fold = 0
for train, test in kf.split(x):
    fold+=1
    print("Fold #{}".format(fold))

    x_train = x[train]
    y_train = y[train]
    x_test = x[test]
    y_test = y[test]

    model = Sequential()
    model.add(layers.Input(shape=(x_train.shape[1],)))
    model.add(layers.Dense(25, activation='relu'))              #
Stratul ascuns 1
    model.add(layers.Dense(30, activation='relu'))              #
Stratul ascuns 2
    model.add(layers.Dense(25, activation='relu'))              #
Stratul ascuns 3
    model.add(layers.Dense(1))
    model.compile(loss='mean_squared_error', optimizer='adam')

    monitor = EarlyStopping(monitor='val_loss', min_delta=1e-3,
patience=5, verbose=1, mode='auto')

model.fit(x_train,y_train,validation_data=(x_test,y_test),callba
cks=[monitor],verbose=1,epochs=1000)

    pred = model.predict(x_test)
```

```
    oos_y.append(y_test)
    oos_pred.append(pred)

    # Measure this fold's RMSE
    score = np.sqrt(metrics.mean_squared_error(pred,y_test))
    print("Fold score (RMSE): {}".format(score))

# Build the oos prediction list and calculate the error.
oos_y = np.concatenate(oos_y)
oos_pred = np.concatenate(oos_pred)
score = np.sqrt(metrics.mean_squared_error(oos_pred,oos_y))
print("Final, out of sample score (RMSE): {}".format(score))

# Write the cross-validated prediction
oos_y = pd.DataFrame(oos_y)
oos_pred = pd.DataFrame(oos_pred)
oosDF = pd.concat( [df, oos_y, oos_pred],axis=1 )
oosDF.to_csv(filename_write,index=False)

print(oosDF)

def chart_regression(pred, y, sort=True):
    t = pd.DataFrame({'pred': pred, 'y': y.flatten()})
    if sort:
        t.sort_values(by=['y'], inplace=True)
    plt.plot(t['y'].tolist(), label='expected')
```

```
    plt.plot(t['pred'].tolist(), label='prediction')
    plt.ylabel('output')
    plt.legend()
    plt.show()


# Plot the chart
chart_regression(pred.flatten(),y_test)


# RNA10a – folosit la a doua rulare (generare grafic)
```

6. Concluzii

Vinuri roșii

Pentru setul de date al vinurilor roșii am considerat că cea mai potrivită arhitectură este **RNA1r**, deoarece este arhitectura care a obținut cel mai mic *RMSE*, și unul dintre cele mai mici *Loss-uri*, în condițiile în care are doar 2 straturi ascunse, și acelea cu puțini neuroni (25, respectiv 10), are cel mai mic număr de epoci alături de RNA4r și RNA5r, și al doilea cel mai mic număr de K-Folds, și anume 5.

Cel mai mic Loss l-a înregistrat RNA5r, dar aceasta are 3 straturi și s-au folosit 10 K-Folds, motiv pentru care nu mi se pare eficientă.

Vinuri albe

Pentru setul de date al vinurilor albe am considerat că cea mai potrivită arhitectură este **RNA6a**, deoarece a obținut cel mai mic *Loss,* alături de RNA4a (0.5848), cu toate acestea cea de a doua a obținut un RMSE mai mare. În comparație cu RNA4a, RNA6a are doar 2 straturi ascunse (cu 25, respectiv 35 de neuroni), cu doar 4 K-Folds, dar cu 500 de epoci.

Cel mai mic RMSE l-a înregistrat RNA10a, dar aceasta este și cea mai complexă și plină arhitectură, având 3 straturi ascunde (2 cu 25 de neuroni și unul cu 30 de neuroni), 1000 de epoci și 10 K-Folds.