

Вариант 1 ($29 \% 7 + 1 = 2$)

Признак 2

$X \in N(3, 10)$

$e \in N(0, 0.3)$

Признак	1	2	3	4	5	6	7
Формула	$X^2 + e$	$\sin(X/2) + e$	$\cos(2x) + e$	$X - 3 + e$	$-X + e$	$ X + e$	$(X^3)/4 + e$

Выполнение работы.

Была написана программа, генерирующая тренировочный и тестовый датасет. Код программы находится в файле generateDataset.py.

```
x = np.random.normal(3.0, 10.0, 3000).reshape(-1,1) # генерируем x
e = np.random.normal(0, 0.3, 3000).reshape(-1,1) # генерируем ошибку

y_1 = x**2 + e
y_2 = np.sin(x/2) + e
y_3 = np.cos(2*x) + e
y_4 = x - 3 + e
y_5 = -x + e
y_6 = abs(x) + e
y_7 = (x**3)/4 + e

np.savetxt('train_dataset.csv', np.hstack((y_1, y_3, y_4, y_5, y_6, y_7,
y_2))) # записываем в файл
```

Была создана модель, согласно схеме.



Рисунок 1 – Схема модели

Модель:

```

inp = Input(shape=(6,))

# encoder
dense_encoder = Dense(64, activation='relu', name='dense_encoder')(inp)
dense_encoder = Dense(32, activation='relu')(dense_encoder)
dense_encoder = Dense(4, activation='relu')(dense_encoder)

# regression
dense_1 = Dense(4, activation='relu')(dense_encoder)
dense_2 = Dense(64, activation='relu')(dense_1)
out_reg = Dense(1, name='out_reg')(dense_2)

# decoder
dense_decoder = Dense(32, activation='relu',
name='dense_decoder_1')(dense_encoder)
dense_decoder = Dense(64, activation='relu',
name='dense_decoder_2')(dense_decoder)
dense_decoder = Dense(6, activation='relu',
name='dense_decoder_out')(dense_decoder)

# configuration
model = Model(inputs=inp, outputs=[out_reg, dense_decoder])
model.compile(optimizer='rmsprop', loss='mse', metrics=['mae'])
model.fit(train_data, [train_targets, train_data], epochs=200, batch_size=10)

```

Результаты для каждой модели запишем в файлы. Так же сохраним каждую из 3х моделей.

```

# encoder model
ecoder_model = Model(inputs=inp, outputs=dense_encoder)
encoder_pred = np.asarray(ecoder_model.predict(test_data))

```

```
# encoder model
ecoder_model = Model(inputs=inp, outputs=dense_ecoder)
ecoder_pred = np.asarray(ecoder_model.predict(test_data))
np.savetxt('outOfModels/data_encoder_model.csv', ecoder_pred)
ecoder_model.save('models/ecoder_model.h5')

# regression model
reg_model = Model(inputs=inp, outputs=out_reg)
reg_pred = np.asarray(reg_model.predict(test_data))
np.savetxt('outOfModels/data_reg_model.csv', np.hstack((test_targets,
reg_pred)))
ecoder_model.save('models/reg_model.h5')

# decoder model
decoder = Model(inp, dense_decoder)
decoder_pred = np.asarray(decoder.predict(test_data))
np.savetxt('outOfModels/data_decoder_model.csv', decoder_pred)
ecoder_model.save('models/decoder_model.h5')
```

Результаты запусков показали, что модели работают плохо. В подавляющем большинстве случаев предсказания модели имеют большую погрешность. Предполагаю, что это связано с обучением модели, так как 3 части большой модели мешают друг другу обучаться. При обучении частей по отдельности, каждая часть показывает хорошие результаты.