



МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ  
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»  
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ  
Кафедра Інформаційної Безпеки

## Хмарні технології обробки даних

### Лабораторна робота №5

### Елементи машинного навчання у AWS SageMaker

**SageMaker** – сервіс машинного навчання у хмарі AWS, який містить чимало вже реалізованих алгоритмів. Окрім наявних користувач може додати ті, які реалізує самостійно (принцип BYOM – Bring Your Own Model). Непогане оглядове відео:

[https://www.youtube.com/watch?v=Qv\\_Tr\\_BCFCQ](https://www.youtube.com/watch?v=Qv_Tr_BCFCQ)

Перевірів:

\_\_\_\_\_

Виконав:

студент III курсу

групи ФБ-01

Сахній Н.Р.

Київ 2023

ФБ-01 Сахній Назар

## Завдання:

1. Оберемо наступний датасет, що має назву “**Covertypes Data Set**” із репозиторію <https://archive.ics.uci.edu/ml/index.php>:

UCI

Machine Learning Repository

Center for Machine Learning and Intelligent Systems

[About](#) [Citation Policy](#) [Donate a Data Set](#) [Contact](#)

[Repository](#) [Web](#)


[View ALL Data Sets](#)

Check out the [beta version](#) of the new UCI Machine Learning Repository we are currently testing! [Contact us](#) if you have any issues, questions, or concerns. [Click here to try out](#) the new site.

### Covertypes Data Set

Download: [Data Folder](#), [Data Set Description](#)

Abstract: Forest CoverType dataset



Data Set Characteristics:	Multivariate	Number of Instances:	581012	Area:	Life
Attribute Characteristics:	Categorical, Integer	Number of Attributes:	54	Date Donated	1998-08-01
Associated Tasks:	Classification	Missing Values?	No	Number of Web Hits:	450238

2. Вивчимо його особливості, переглянувши наступний опис датасету:

Predicting forest cover type from cartographic variables only (no remotely sensed data). The actual forest cover type for a given observation (30 x 30 meter cell) was determined from US Forest Service (USFS) Region 2 Resource Information System (RIS) data. Independent variables were derived from data originally obtained from US Geological Survey (USGS) and USFS data. Data is in raw form (not scaled) and contains binary (0 or 1) columns of data for qualitative independent variables (wilderness areas and soil types).

This study area includes four wilderness areas located in the Roosevelt National Forest of northern Colorado. These areas represent forests with minimal human-caused disturbances, so that existing forest cover types are more a result of ecological processes rather than forest management practices.

Some background information for these four wilderness areas: Neota (area 2) probably has the highest mean elevational value of the 4 wilderness areas. Rawah (area 1) and Comanche Peak (area 3) would have a lower mean elevational value, while Cache la Poudre (area 4) would have the lowest mean elevational value.

As for primary major tree species in these areas, Neota would have spruce/fir (type 1), while Rawah and Comanche Peak would probably have lodgepole pine (type 2) as their primary species, followed by spruce/fir and aspen (type 5). Cache la Poudre would tend to have Ponderosa pine (type 3), Douglas-fir (type 6), and cottonwood/willow (type 4).

The Rawah and Comanche Peak areas would tend to be more typical of the overall dataset than either the Neota or Cache la Poudre, due to their assortment of tree species and range of predictive variable values (elevation, etc.) Cache la Poudre would probably be more unique than the others, due to its relatively low elevation range and species composition.

Name	Data Type	Measurement	Description
Elevation	quantitative	meters	Elevation in meters
Aspect	quantitative	azimuth	Aspect in degrees azimuth
Slope	quantitative	degrees	Slope in degrees
Horizontal_Distance_To_Hydrology	quantitative	meters	Horz Dist to nearest surface water features
Vertical_Distance_To_Hydrology	quantitative	meters	Vert Dist to nearest surface water features
Horizontal_Distance_To_Roadways	quantitative	meters	Horz Dist to nearest roadway
Hillshade_9am	quantitative	0 to 255 index	Hillshade index at 9am, summer solstice
Hillshade_Noon	quantitative	0 to 255 index	Hillshade index at noon, summer solstice
Hillshade_3pm	quantitative	0 to 255 index	Hillshade index at 3pm, summer solstice
Horizontal_Distance_To_Fire_Points	quantitative	meters	Horz Dist to nearest wildfire ignition points
Cover_Type (7 types)	integer	1 to 7	Forest Cover Type designation

## Прогнозування типу лісового покриття лише за картографічними змінними

3. Вирішимо двома способами задачу класифікації/кластеризації:

- За допомогою локальної реалізації на власних ресурсах ↓

```
column_names = ['Elevation', 'Aspect', 'Slope', 'Horizontal_Distance_To_Hydrology',  
                'Vertical_Distance_To_Hydrology', 'Horizontal_Distance_To_Roadways', 'Hillshade_9am',  
                'Hillshade_Noon', 'Hillshade_3pm', 'Horizontal_Distance_To_Fire_Points', 'Cover_Type']  
df = pd.read_csv('covtype.data', names=column_names)
```

```
df.head()
```

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_N
0	2596	51	3	258	0	510	221	
1	2590	56	2	212	-6	390	220	
2	2804	139	9	268	65	3180	234	
3	2785	155	18	242	118	3090	238	
4	2595	45	2	153	-1	391	220	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4811 entries, 0 to 4810
Data columns (total 11 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Elevation                            4811 non-null   int64
1   Aspect                              4811 non-null   int64
2   Slope                               4811 non-null   int64
3   Horizontal_Distance_To_Hydrology     4811 non-null   int64
4   Vertical_Distance_To_Hydrology       4811 non-null   int64
5   Horizontal_Distance_To_Roadways      4811 non-null   int64
6   Hillshade_9am                       4811 non-null   int64
7   Hillshade_Noon                      4811 non-null   int64
8   Hillshade_3pm                      4811 non-null   int64
9   Horizontal_Distance_To_Fire_Points   4811 non-null   int64
10  Cover_Type                          4811 non-null   int64
dtypes: int64(11)
memory usage: 413.6 KB
```

```
df.describe()
```

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_Noon	Hillshade_3pm	Horizontal_Distance_To_Fire_Points	Cover_Type
count	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000
mean	2598.034296	155.386822	17.529620	184.315527	45.828726	1744.472875	212.445438	215.742673	130.802120	1531.437331	3.623779
std	390.955420	109.387211	9.409759	160.360352	56.161330	1585.123497	33.267918	25.277312	50.596236	1338.250688	1.635815
min	1863.000000	0.000000	0.000000	0.000000	-134.000000	30.000000	0.000000	99.000000	0.000000	30.000000	1.000000
25%	2219.000000	66.000000	10.000000	42.000000	3.000000	618.000000	196.000000	203.000000	97.000000	602.000000	2.000000
50%	2697.000000	122.000000	16.000000	150.000000	26.000000	1168.000000	221.000000	221.000000	135.000000	1130.000000	4.000000
75%	2913.000000	261.000000	25.000000	277.000000	74.000000	2291.000000	237.000000	234.000000	167.000000	2034.500000	5.000000
max	3442.000000	360.000000	52.000000	997.000000	554.000000	6890.000000	254.000000	254.000000	248.000000	6853.000000	7.000000

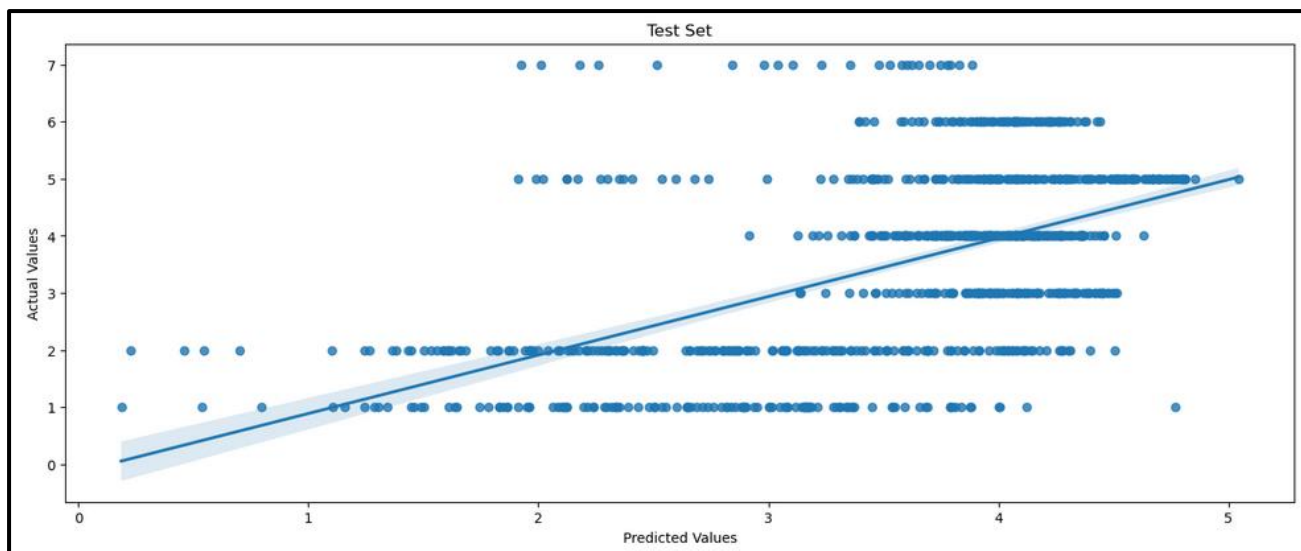
```
print("Testing Results")
print(f"R^2 Score: {score_scaled}")
y_pred = linreg.predict(X_test_scaled)

# Compute the RMSE of our predictions
rmse = np.sqrt(metrics.mean_squared_error(y_test, y_pred))
print(f"Test RSME: {rmse}")
```

```
Testing Results
R^2 Score: 0.282653748542975
Test RSME: 1.3859322190102281
```

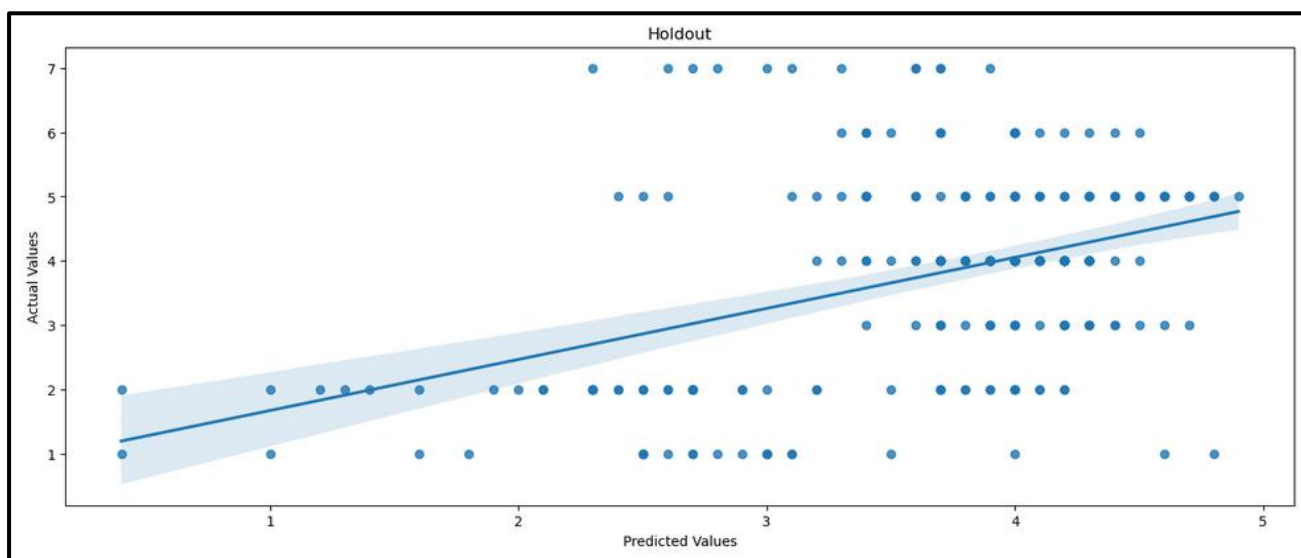
Test DataSet

	actual Cover_Type	predicted Cover_Type	rings Cover_Type
3692	4	4.0	4
3998	4	3.3	4
1553	1	3.3	1
4345	4	3.6	4
4268	5	3.9	5



Holdout DataSet

	actual Cover_Type	predicted Cover_Type	Cover_Type diff
4651	4	4.1	4
4122	4	3.9	4
2376	6	4.0	6
1468	2	4.0	2
4415	4	4.0	4



- За допомогою можливостей AWS SageMaker ↓

Jupyter Machine learning in AWS SageMaker Last Checkpoint: 30 хвилин тому (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted | conda\_tensorflow2\_p310 \$

Run Code nbdiff

```
import warnings
warnings.filterwarnings("ignore")
import boto3
from sagemaker import get_execution_role
import sys, os
import pandas as pd
import numpy as np
import sagemaker.amazon.common as smac
import io
```

Імпортуємо необхідні бібліотеки.

```

from pathlib import Path
import shutil
from sklearn.model_selection import train_test_split
from io import StringIO
import json
from sklearn import metrics
from sagemaker.amazon.amazon_estimator import get_image_uri
import sagemaker
from sklearn.preprocessing import StandardScaler
import seaborn as sns
import time
import matplotlib.pyplot as plt
%matplotlib inline

```

```

filename = 'covtype_recordio_train.data'
bucket = 'fightingmyself'
raw_prefix = 'raw'
dataset_name = 'covtype.data'
data_dir='dataset'
train_prefix = 'train'
output_prefix = 'output'
train_path = f'{train_prefix}/{filename}'
s3_train_data = f's3://{bucket}/{train_path}'
output_location = f's3://{bucket}/{output_prefix}'

```

Вкажемо параметри взаємодії із S3.

```

print(s3_train_data)
print(output_location)

s3://fightingmyself/train/covtype_recordio_train.data
s3://fightingmyself/output

```

Перевіримо шляхи до даних та виходів моделі.

```

%env DATA_DIR=$data_dir
%env S3_DATA_BUCKET_NAME = $bucket/$raw_prefix
%env DATASET_NAME = $dataset_name
%env TRAINING_PATH = $bucket/$train_prefix

```

Налаштуємо також змінні оточення для роботи з AWS CLI.

```

env: DATA_DIR=dataset
env: S3_DATA_BUCKET_NAME=fightingmyself/raw
env: DATASET_NAME=covtype.data
env: TRAINING_PATH=fightingmyself/train

```

```

!aws s3 cp s3://$S3_DATA_BUCKET_NAME/$DATASET_NAME ./DATA_DIR/

download: s3://fightingmyself/raw/covtype.data to dataset/covtype.data

```

Заберемо сирі дані з S3.

```

column_names = ['Elevation', 'Aspect', 'Slope', 'Horizontal_Distance_To_Hydrology',
                'Vertical_Distance_To_Hydrology', 'Horizontal_Distance_To_Roadways', 'Hillshade_9am',
                'Hillshade_Noon', 'Hillshade_3pm', 'Horizontal_Distance_To_Fire_Points', 'Cover_Type']
df = pd.read_csv('./dataset/covtype.data', names=column_names)
df.head()

```

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_Noon
0	2596	51	3	258	0	510	221	234
1	2590	56	2	212	-6	390	220	234
2	2804	139	9	268	65	3180	234	234
3	2785	155	18	242	118	3090	238	234
4	2595	45	2	153	-1	391	220	234

```
df.shape
```

```
(4811, 11)
```

```
df.describe()
```

Оскільки датасет уже на інстансі, то з ним можна працювати, як і на локальних ресурсах.

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_Noon
count	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000	4811.000000
mean	2598.034296	155.386822	17.529620	184.315527	45.828726	1744.472875	212.4454	212.4454
std	390.955420	109.387211	9.409759	160.360352	56.161330	1585.123497	33.2679	33.2679
min	1863.000000	0.000000	0.000000	0.000000	-134.000000	30.000000	0.0000	0.0000
25%	2219.000000	66.000000	10.000000	42.000000	3.000000	618.000000	196.0000	196.0000
50%	2697.000000	122.000000	16.000000	150.000000	26.000000	1168.000000	221.0000	221.0000
75%	2913.000000	261.000000	25.000000	277.000000	74.000000	2291.000000	237.0000	237.0000
max	3442.000000	360.000000	52.000000	997.000000	554.000000	6890.000000	254.0000	254.0000



```
numeric_features = list(df.select_dtypes([np.number]).columns)
X = df[numeric_features].copy()
X.drop(columns=['Cover_Type'], axis=1, inplace=True)
y = df['Cover_Type']
```

```
# Validation set - 5%
X_train, X_holdout, y_train, y_holdout = train_test_split(X, y, test_size=0.05)
```

```
X_train.shape
```

```
(4570, 10)
```

```
X_holdout.shape
```

```
(241, 10)
```

Розділимо датасет на вибірки – на тестову частину залишимо 5% даних (4570 + 241 = 4811).

```
buf = io.BytesIO()
smac.write_numpy_to_dense_tensor(buf, np.array(X_train).astype('float32'), np.array(y_train).astype('float32'))
buf.seek(0)
boto3.resource('s3').Bucket(bucket).Object(f'{train_path}').upload_fileobj(buf)
```

```
container = get_image_uri(boto3.Session().region_name, 'linear-learner')
sess = sagemaker.Session()
role = get_execution_role()
linear = sagemaker.estimator.Estimator(
    container,
    role,
    instance_count=1,
    instance_type='ml.c5.xlarge',
    output_path=output_location,
    sagemaker_session=sess,
)
linear.set_hyperparameters(
    feature_dim=10,
    epochs=10,
    num_models=16,
    loss='absolute_loss',
    predictor_type='regressor',
    mini_batch_size=16,
    normalize_data=True,
    normalize_label=False
)
linear.fit({'train': s3_train_data}, job_name=f"job-covtype-{int(time.time())}")
```

Тепер створимо сесію SageMaker та отримаємо IAM роль. Далі через Estimator API виберемо інстанс, який відповідає задачі та датасету. У секції з гіперпараметрами зробимо лише нормалізацію даних.

```
WARNING:sagemaker.deprecations:The method get_image_uri has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
```

```
INFO:sagemaker.image_uris:Same images used for training and inference. Defaulting to image scope: inference.
```

```
INFO:sagemaker.image_uris:Ignoring unnecessary instance type: None.
INFO:sagemaker:Creating training-job with name: job-covtype-1684931036
```

```
2023-05-24 12:23:56 Starting - Starting the training job...
2023-05-24 12:24:10 Starting - Preparing the instances for training...
2023-05-24 12:25:01 Downloading - Downloading input data...
2023-05-24 12:25:26 Training - Downloading the training image.....
2023-05-24 12:26:22 Training - Training image download completed. Training in progress.Docker entrypoint called with argument
(s): train
Running default environment configuration script
[05/24/2023 12:26:28 INFO 139955250411328] Reading default configuration from /opt/amazon/lib/python3.7/site-packages/algorit
hm/resources/default-input.json: {'mini_batch_size': '1000', 'epochs': '15', 'feature_dim': 'auto', 'use_bias': 'true', 'bina
ry_classifier_model_selection_criteria': 'accuracy', 'f_beta': '1.0', 'target_recall': '0.8', 'target_precision': '0.8', 'num
_models': 'auto', 'num_calibration_samples': '1000000', 'init_method': 'uniform', 'init_scale': '0.07', 'init_sigma': '0.0
1', 'init_bias': '0.0', 'optimizer': 'auto', 'loss': 'auto', 'margin': '1.0', 'quantile': '0.5', 'loss_insensitivity': '0.0
1', 'huber_delta': '1.0', 'num_classes': '1', 'accuracy_top_k': '3', 'wd': 'auto', 'l1': 'auto', 'momentum': 'auto', 'learnin
```

```
{
  "sum": 5.728244781494141,
  "count": 10,
  "min": 0.1506805419921875,
  "max": 1.1210441589355469,
  "update.time": {
    "sum": 17860.005617141724,
    "count": 10,
    "min": 1649.388313293457,
    "max": 1986.2239360809326,
    "finalize.time": {
      "sum": 276.8056392669678,
      "count": 1,
      "min": 276.8056392669678,
      "max": 276.8056392669678,
      "setuptime": {
        "sum": 1.8775463104248047,
        "count": 1,
        "min": 1.8775463104248047,
        "max": 1.8775463104248047,
        "totaltime": {
          "sum": 18980.578660964966,
          "count": 1,
          "min": 18980.578660964966,
          "max": 18980.578660964966
        }
      }
    }
  }
}
```

```
2023-05-24 12:27:08 Uploading - Uploading generated training model
2023-05-24 12:27:08 Completed - Training job completed
Training seconds: 127
Billable seconds: 127
```

Модель вже навчена і для її використання потрібно менше ресурсів.

```
linear_predictor = linear.deploy(initial_instance_count=1, instance_type='ml.c5.xlarge', endpoint_name="covtype-endpoint")
```

```
INFO:sagemaker:Creating model with name: linear-learner-2023-05-24-12-30-4
INFO:sagemaker:Creating endpoint-config with name covtype-endpoint
INFO:sagemaker:Creating endpoint with name covtype-endpoint
```

Створюємо endpoint моделі, який забезпечує доступ до неї.

```
-----!
```

```
X_holdout.shape
```

```
(241, 10)
```

```
from sagemaker.predictor import csv_serializer, json_deserializer
linear_predictor.serializer = csv_serializer
linear_predictor.deserializer = json_deserializer
```

```
result = linear_predictor.predict(X_holdout.values)
```

Використаємо метод `predict` для `linear_predictor` **endpoint**, щоб отримати вихід моделі для тестового датасету.

```
WARNING:sagemaker.deprecations:The csv_serializer has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
WARNING:sagemaker.deprecations:The json_deserializer has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
```

```
predictions = [ x['score'] for x in result["predictions"]]
print(f"RSME: {np.sqrt(metrics.mean_squared_error(y_holdout.values, predictions))}")
```

```
RSME: 1.4639308754951537
```

Заберемо з отриманого json прогнози значення та розрахуємо RMSE помилку.

```
print("Holdout DataSet")
prediction_df = pd.DataFrame({'actual covtype': y_holdout.values, 'predicted covtype': predictions})
prediction_df['covtype diff'] = prediction_df['actual covtype']
prediction_df['predicted covtype']
prediction_df.head()
```

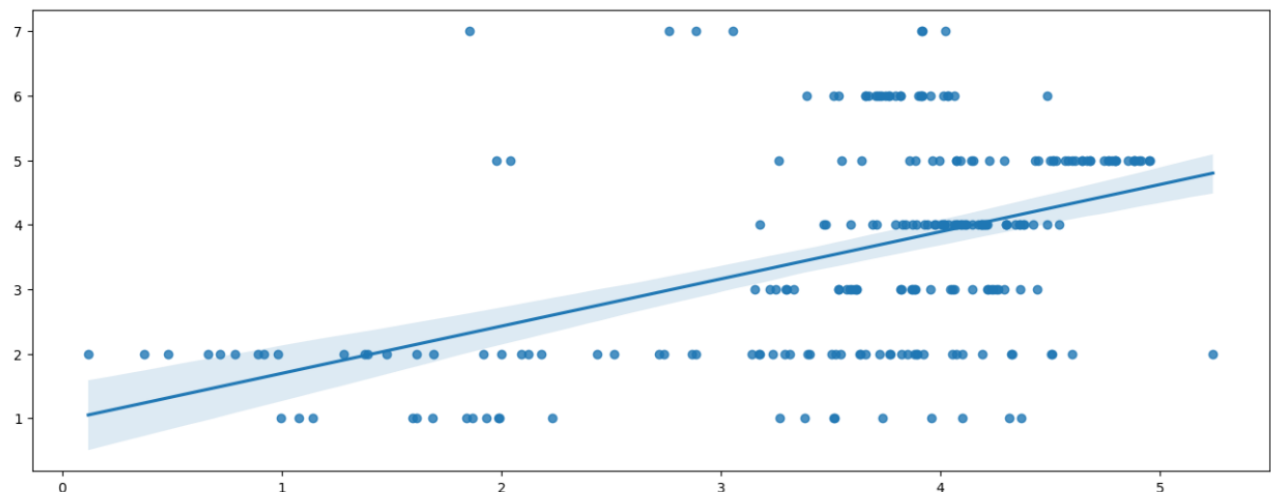
```
Holdout DataSet
```

	actual covtype	predicted covtype	covtype diff
0	4	4.361879	4
1	1	4.100094	1
2	5	4.088346	5
3	2	4.070676	2
4	6	3.815552	6

Отримані результати є цілком співставними із тими, що були отримані під час локальної реалізації.

```
plt.figure(figsize=(16, 6))
sns.regplot(np.array(predictions), np.array(y_holdout.values))
```

```
<AxesSubplot:>
```



/\* Проте перед тим як працювати із **Jupyter Notebook** необхідно було виконати деякі кроки підготування середовища для AWS SageMaker \*/

## 1. Створити S3-бакет:

### General configuration

Bucket name

Bucket name must be globally unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region

EU (Frankfurt) eu-central-1

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Choose bucket

2. Створити папки:

Amazon S3 > Buckets > fightingmyself

fightingmyself

Objects | Properties | Permissions | Metrics | Management | Access Points

Objects (3)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Refresh

Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder

Upload

Find objects by prefix

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	output/	Folder	-	-	-
<input type="checkbox"/>	raw/	Folder	-	-	-
<input type="checkbox"/>	train/	Folder	-	-	-

3. Додати датасет до папки **/raw**:

Files and folders (1 Total, 195.3 KB)

Remove

Add files

Add folder

All files and folders in this table will be uploaded.

Find by name

	Name	Folder	Type	Size
<input type="checkbox"/>	covtype.data	-	-	195.3 KB

Destination

Destination

s3://fightingmyself/raw/

4. Створити IAM-роль:

Create an IAM role

Passing an IAM role gives Amazon SageMaker permission to perform actions in other AWS services on your behalf. Creating a role here will grant permissions described by the [AmazonSageMakerFullAccess](#) IAM policy to the role you create.

The IAM role you create will provide access to:



✓ S3 buckets you specify - *optional*

☐ Any S3 bucket  
Allow users that have access to your notebook instance access to any bucket and its contents in your account.

☒ Specific S3 buckets

Comma delimited. ARNs, "\*" and "/" are not supported.

☐ None

---

✓ Any S3 bucket with "sagemaker" in the name

✓ Any S3 object with "sagemaker" in the name

✓ Any S3 object with the tag "sagemaker" and value "true" [See Object tagging](#)

✓ S3 bucket with a Bucket Policy allowing access to SageMaker [See S3 bucket policies](#)

Cancel [Create role](#)

## 5. Додати доменне ім'я:

**Domain name**

Name  
Domain name should be unique across the AWS account.

---

**User profile**

Name

The name can have up to 63 characters. Valid characters: A-Z, a-z, 0-9, and - (hyphen)

Execution role  
The default execution role for both users and spaces in the domain. The execution role must have the [AmazonSageMakerFullAccess](#) policy attached.

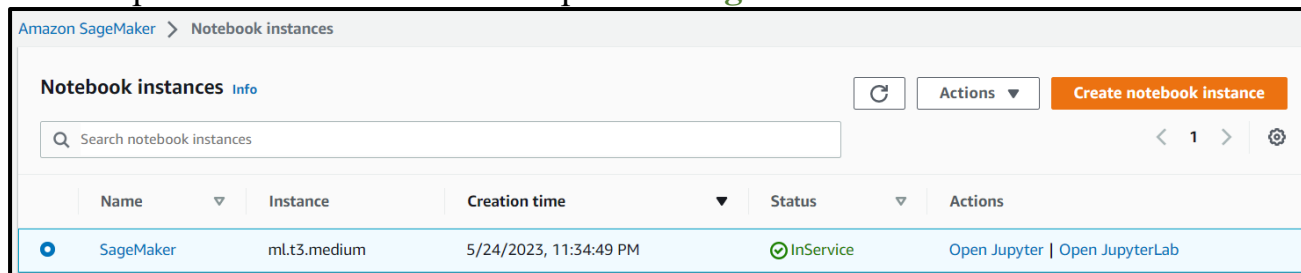
✓ **Success! You created an IAM role.**

[AmazonSageMaker-ExecutionRole-20230524T233166](#)

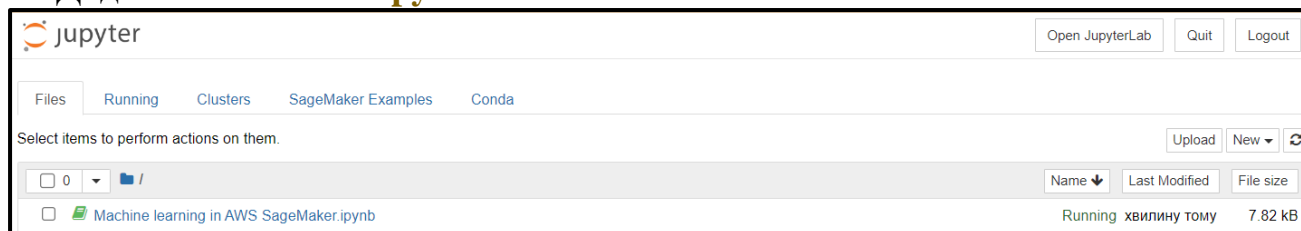
[Create role using the role creation wizard](#)

Cancel [Submit](#)

## 6. Створити блокнотний екземпляр **AWS SageMaker**:



## 7. Додати той самий **Jupyter Notebook**:



## Висновки:

Якщо чесно виконання лабораторної роботи забрало у мене дуже багато часу, а також трохи коштів. Хоча ні, щодо останнього, то я домовився із підтримкою AWS, що вони умовно не будуть знімати кошти за перевикористані ресурси **SageMaker**. Але питання у тому, чому ж я вийшов за ліміти **Free Tier**.

По-перше, у мене виникнули серйозні проблеми під час навчання моделі, які я, ну ніяк, не міг вирішити. Я писав усім кому можна, навіть у службу підтримки, але так і не зміг знайти відповіді на моє питання: “Чому на акаунті, наприклад мого одnogрупника, усе чітко працює, а на моєму при такому самому регіоні, датасеті та аналогічних налаштуваннях середовища – ні?”. Але вже думаю, що нехай так і буде, спробую змінити регіон та проробити усе спочатку. І щось я знову натикаюсь на проблеми, уже на цей раз проблема із копіюванням датасету із бакета на інстанс. Пробую ще у двох інших регіонах... Безрезультатно! Хоча як виявилось пізніше, при наступних взаємодіях із інстансом у нових регіонах я забував створити IAM-роль під новий бакет ::)

По-друге, коли я створив інстанси для машинного навчання та невдало з ними попрацював, то я забув їх видалити по завершенню “роботи” з ними -- Повертаюся через три тижні за роботу, і виявляється години простою інстансу у регіоні **eu-north-1 (Stockholm)** не безкоштовні. І мені накрутило, аж на 15\$, що на секунду дорівнює чверті звичайної студентської стипендії. Але, на щастя, служба підтримки мала співчуття до мене, та надала додаткові кредити на цей місяць для покриття незапланованих витрат. God save AWS Support Team!