МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«МИРЭА – Российский технологический университет»

Институт кибербезопасности и цифровых технологий

Кафедра КБ-4 «Интеллектуальные системы информационной безопасности»

Отчёт по практической работе № 5

По дисциплине

«Анализ защищенности систем искусственного интеллекта»

Выполнил:

ББМО–02–22

Шмарковский М. Б.

Проверил:

Спирин А. А.

Москва, 2024

Практическая работа №5

Шмарковский МБ ББМО-**02-22**

Ход работы

In [1]:

```
#Импортируем необходимые библиотеки
import numpy as np
import matplotlib.pyplot as plt
from skimage.color import gray2rgb, rgb2gray, label2rgb
```

In [2]:

```
from sklearn.datasets import fetch_openml
mnist = fetch_openml('mnist_784')
#Тут сделаем каждое изображение цветным, чтобы LIME_image работал правильно
X_vec = np.stack([gray2rgb(iimg) for iimg in mnist.data.values.astype(np.uint8).reshape(
(-1, 28, 28))],0).astype(np.uint8)
y_vec = mnist.target.astype(np.uint8)
```

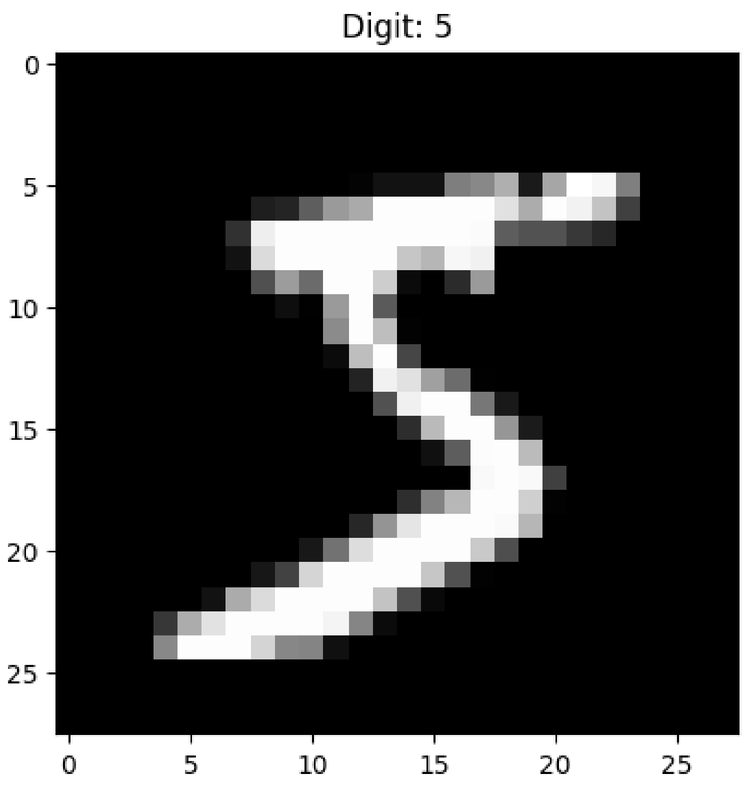
```
/usr/local/lib/python3.10/dist-packages/sklearn/datasets/_openml.py:968: FutureWarning: T
he default value of `parser` will change from `'liac-arff'` to `'auto'` in 1.4. You can s
et `parser='auto'` to silence this warning. Therefore, an `ImportError` will be raised fr
om 1.4 if the dataset is dense and pandas is not installed. Note that the pandas parser m
ay return different data types. See the Notes Section in fetch_openml's API doc for detai
ls.
  warn(
```

In [3]:

```
%matplotlib inline
fig, ax1 = plt.subplots(1,1)
ax1.imshow(X_vec[0], interpolation = 'none')
ax1.set_title('Digit: {}'.format(y_vec[0]))
```

Out[3]:

```
Text(0.5, 1.0, 'Digit: 5')
```



Настройка **Pipeline**

In [4]:

```
#Здесь мы создаем Pipeline для обработки изображений, где в основном мы сглаживаем изобра
жение до одномерных векторов, а затем используем классификатор RandomForest.
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import Normalizer

class PipeStep(object):
    """
    Wrapper for turning functions into pipeline transforms (no-fitting)
    """
    def __init__(self, step_func):
        self._step_func=step_func
    def fit(self,*args):
        return self
    def transform(self,X):
        return self._step_func(X)


makegray_step = PipeStep(lambda img_list: [rgb2gray(img) for img in img_list])
flatten_step = PipeStep(lambda img_list: [img.ravel() for img in img_list])

simple_rf_pipeline = Pipeline([
    ('Make Gray', makegray_step),
    ('Flatten Image', flatten_step),
    #('Normalize', Normalizer()),
    #('PCA', PCA(16)),
    ('RF', RandomForestClassifier())
                              ])
```

In [5]:

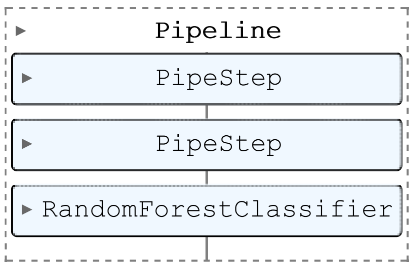
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X_vec, y_vec,
                                                    train_size=0.55)
```

In [6]:

```
simple_rf_pipeline.fit(X_train, y_train)
```

Out[6]:



In [7]:

```
!pip install lime
```

```
Collecting lime
  Downloading lime-0.2.0.1.tar.gz (275 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 275.7/275.7 kB 2.1 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (fro
m lime) (3.7.1)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from lim
e) (1.23.5)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from lim
e) (1.11.4)
```

```
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from lime
) (4.66.1)
Requirement already satisfied: scikit-learn>=0.18 in /usr/local/lib/python3.10/dist-packa
ges (from lime) (1.2.2)
Requirement already satisfied: scikit-image>=0.12 in /usr/local/lib/python3.10/dist-packa
ges (from lime) (0.19.3)
Requirement already satisfied: networkx>=2.2 in /usr/local/lib/python3.10/dist-packages (
from scikit-image>=0.12->lime) (3.2.1)
Requirement already satisfied: pillow!=7.1.0,!=7.1.1,!=8.3.0,>=6.1.0 in /usr/local/lib/py
thon3.10/dist-packages (from scikit-image>=0.12->lime) (9.4.0)
Requirement already satisfied: imageio>=2.4.1 in /usr/local/lib/python3.10/dist-packages
(from scikit-image>=0.12->lime) (2.31.6)
Requirement already satisfied: tifffile>=2019.7.26 in /usr/local/lib/python3.10/dist-pack
ages (from scikit-image>=0.12->lime) (2023.12.9)
Requirement already satisfied: PyWavelets>=1.1.1 in /usr/local/lib/python3.10/dist-packag
es (from scikit-image>=0.12->lime) (1.5.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages
(from scikit-image>=0.12->lime) (23.2)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (
from scikit-learn>=0.18->lime) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-pac
kages (from scikit-learn>=0.18->lime) (3.2.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-package
s (from matplotlib->lime) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (f
rom matplotlib->lime) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packag
es (from matplotlib->lime) (4.47.2)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packag
es (from matplotlib->lime) (1.4.5)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-package
s (from matplotlib->lime) (3.1.1)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-pac
kages (from matplotlib->lime) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from
python-dateutil>=2.7->matplotlib->lime) (1.16.0)
Building wheels for collected packages: lime
  Building wheel for lime (setup.py) ... done
  Created wheel for lime: filename=lime-0.2.0.1-py3-none-any.whl size=283835 sha256=41d21
3d767c90dbe636bfd25d5b6f852fab8ec9edba4fdeb7a356339dbe5a6bd
  Stored in directory: /root/.cache/pip/wheels/fd/a2/af/9ac0a1a85a27f314a06b39e1f492bee15
47d52549a4606ed89
Successfully built lime
Installing collected packages: lime
Successfully installed lime-0.2.0.1
```

In [8]:

```
%load_ext autoreload
%autoreload 2
import os,sys
try:
    import lime
except:
    sys.path.append(os.path.join('..', '..')) # add the current directory
    import lime
```

In [9]:

```
from lime import lime_image
from lime.wrappers.scikit_image import SegmentationAlgorithm
explainer = lime_image.LimeImageExplainer(verbose = False)
segmenter = SegmentationAlgorithm('quickshift', kernel_size=1, max_dist=200, ratio=0.2)
```

In [10]:

```
%%time
explanation = explainer.explain_instance(X_test[0],
                                         classifier_fn = simple_rf_pipeline.predict_prob
a,
                                         top_labels=10, hide_color=0, num_samples=10000
, segmentation_fn=segmenter)
```

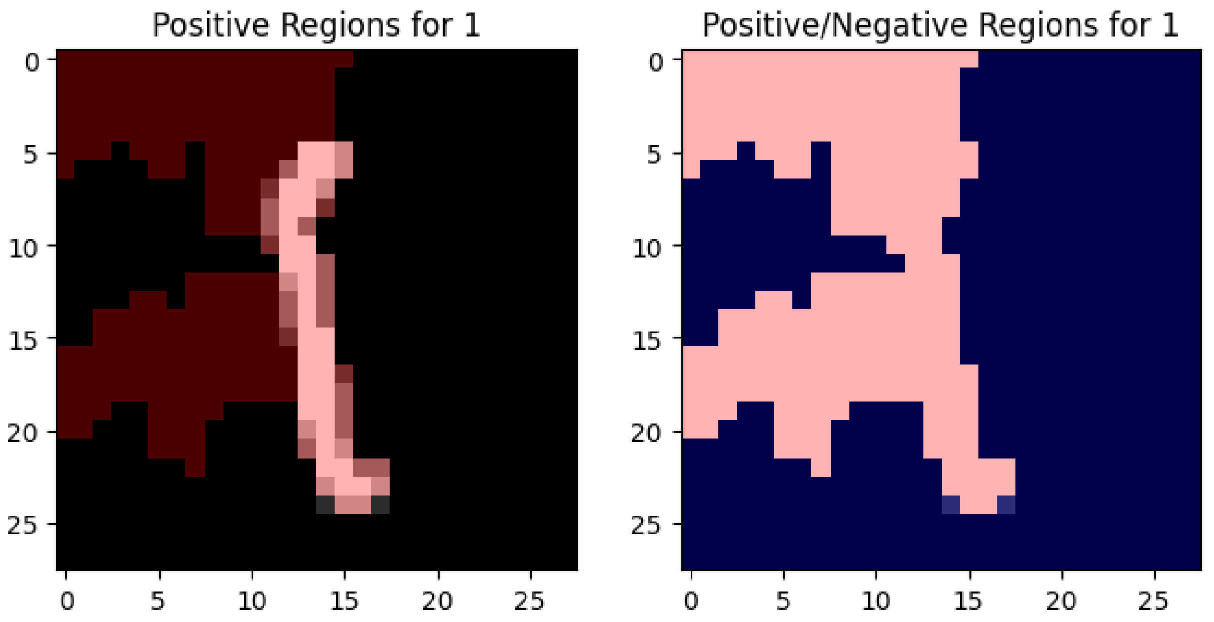
```
CPU times: user 10.7 s, sys: 194 ms, total: 10.9 s
Wall time: 15.1 s
```

In [12]:

```
temp, mask = explanation.get_image_and_mask(y_test.iloc[0], positive_only=True, num_featu
res=10, hide_rest=False, min_weight = 0.01)
fig, (ax1, ax2) = plt.subplots(1,2, figsize = (8, 4))
ax1.imshow(label2rgb(mask,temp, bg_label = 0), interpolation = 'nearest')
ax1.set_title('Positive Regions for {}'.format(y_test.iloc[0]))
temp, mask = explanation.get_image_and_mask(y_test.iloc[0], positive_only=False, num_feat
ures=10, hide_rest=False, min_weight = 0.01)
ax2.imshow(label2rgb(3-mask,temp, bg_label = 0), interpolation = 'nearest')
ax2.set_title('Positive/Negative Regions for {}'.format(y_test.iloc[0]))
```

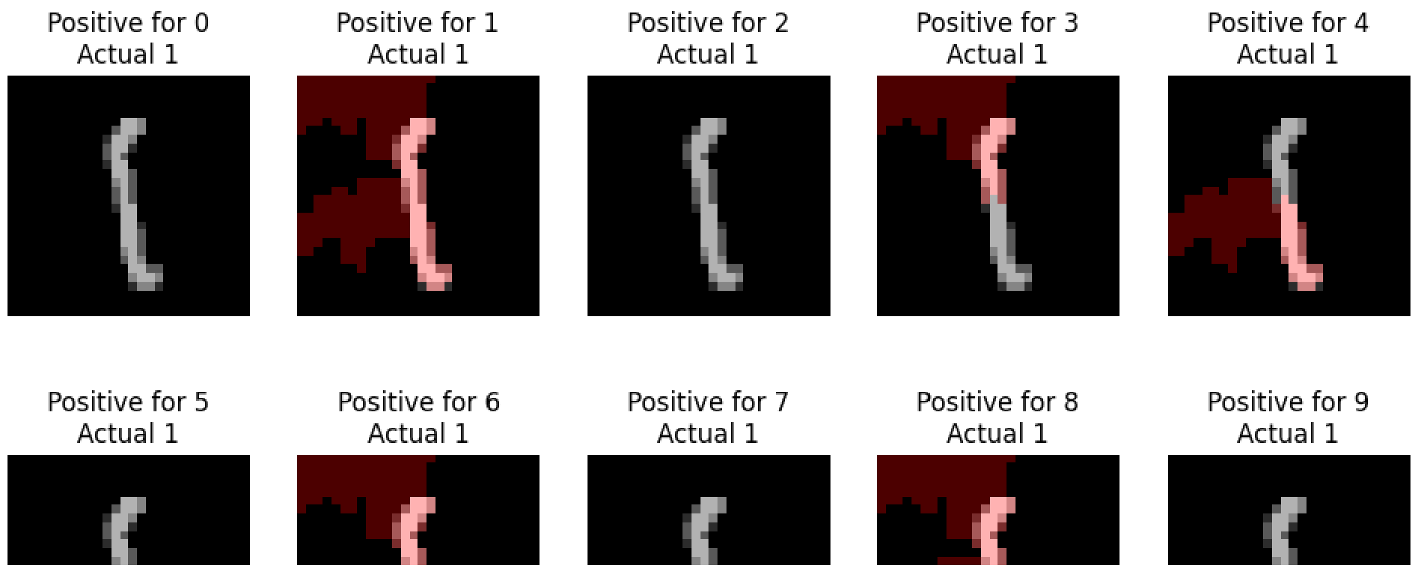
Out[12]:

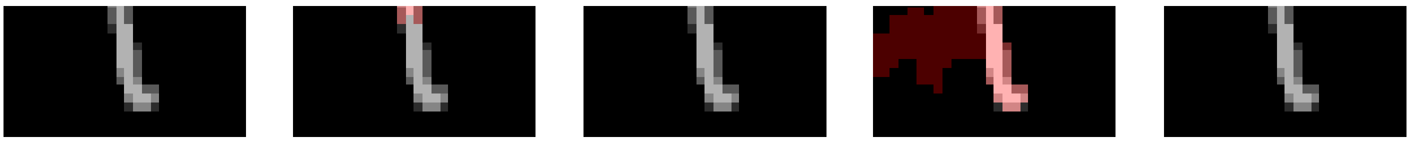
```
Text(0.5, 1.0, 'Positive/Negative Regions for 1')
```



In [17]:

```
#Теперь для каждого класса
fig, m_axs = plt.subplots(2,5, figsize = (12,6))
for i, c_ax in enumerate(m_axs.flatten()):
    temp, mask = explanation.get_image_and_mask(i, positive_only=True, num_features=1000
, hide_rest=False, min_weight = 0.01 )
    c_ax.imshow(label2rgb(mask,X_test[0], bg_label = 0), interpolation = 'nearest')
    c_ax.set_title('Positive for {}\nActual {}'.format(i, y_test.iloc[0]))
    c_ax.axis('off')
```





Поиск объяснения

Проверим, можем ли мы найти объяснение классификации, в которой алгоритм ошибся?

In [18]:

```
pipe_pred_test = simple_rf_pipeline.predict(X_test)
wrong_idx = np.random.choice(np.where(pipe_pred_test!=y_test)[0])
print('Using #{} where the label was {} and the pipeline predicted {}'.format(wrong_idx,
y_test[wrong_idx], pipe_pred_test[wrong_idx]))
```

```
Using #11507 where the label was 5 and the pipeline predicted 0
```

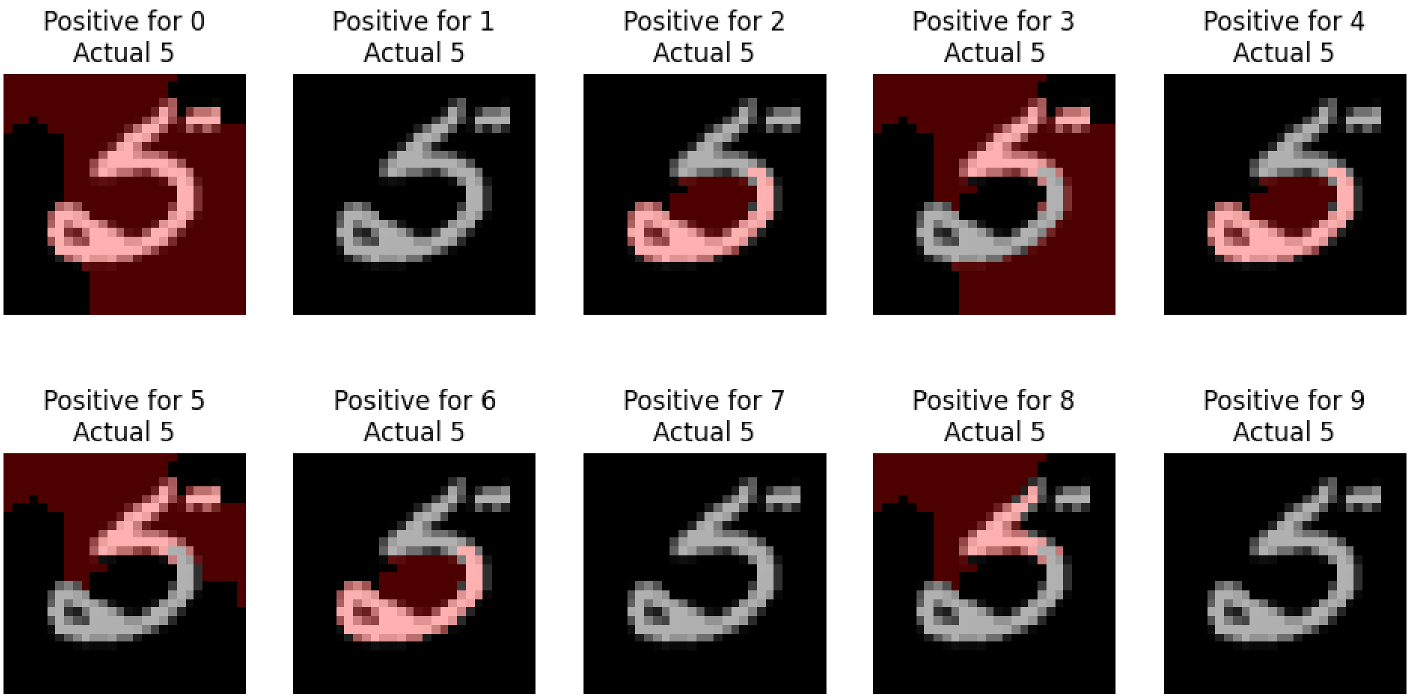
In [19]:

```
%%time
explanation = explainer.explain_instance(X_test[wrong_idx],
                                         classifier_fn = simple_rf_pipeline.predict_prob
a,
                                         top_labels=10, hide_color=0, num_samples=10000
, segmentation_fn=segmenter)
```

```
CPU times: user 8.98 s, sys: 178 ms, total: 9.16 s
Wall time: 9.41 s
```

In [20]:

```
fig, m_axs = plt.subplots(2,5, figsize = (12,6))
for i, c_ax in enumerate(m_axs.flatten()):
    temp, mask = explanation.get_image_and_mask(i, positive_only=True, num_features=10,
hide_rest=False, min_weight = 0.01 )
    c_ax.imshow(label2rgb(mask,temp, bg_label = 0), interpolation = 'nearest')
    c_ax.set_title('Positive for {}\nActual {}'.format(i, y_test[wrong_idx]))
    c_ax.axis('off')
```


Positive for 0
Actual 5
Positive for 1
Actual 5
Positive for 2
Actual 5
Positive for 3
Actual 5
Positive for 4
Actual 5
Positive for 5
Actual 5
Positive for 6
Actual 5
Positive for 7
Actual 5
Positive for 8
Actual 5
Positive for 9
Actual 5