10. Практическое занятие: Определение типов и классификация текста. Машинное обучение и нейронные сети

Цель занятия

Изучить архитектуру систем классификации текста, пройти путь от подготовки признаков до обучения классической модели машинного обучения и нейронной сети, а также научиться оценивать их точность.

Теоретический минимум

Классификация текста — это задача отнесения документа к одной или нескольким категориям (например, "Спам" / "Не спам", "Спорт" / "Политика").

1. **Классическое ML:** Текст переводится в векторы (TF-IDF), затем используется алгоритм (например, Наивный Байес, Random Forest или SVM).
2. **Нейронные сети:** Используют эмбеддинги (векторные представления слов) и слои (Dense, CNN или RNN), которые самостоятельно выявляют сложные закономерности в тексте.

Задание 1. Подготовка данных для классификации

Для обучения модели нам нужен размеченный датасет. Мы создадим синтетический набор данных из двух категорий: "Технологии" и "Кулинария".

Инструкция:

Подготовьте данные и разделите их на обучающую (train) и тестовую (test) выборки.

Python

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

data = [
    ("Новый смартфон оснащен мощным процессором", "tech"),
    ("Программирование на Python это весело", "tech"),
    ("Облачные вычисления меняют мир", "tech"),
    ("Рецепт вкусного борща с чесноком", "food"),
    ("Как приготовить пасту карбонара", "food"),
    ("Специи придают блюду неповторимый аромат", "food")
]

texts, labels = zip(*data)

# Разделение 80% на обучение и 20% на проверку
X_train, X_test, y_train, y_test = train_test_split(texts, labels,
test_size=0.2, random_state=42)
```

Задание 2. Классическое машинное обучение (Наивный Байес)

Наивный Байесовский классификатор — это стандарт индустрии для быстрой классификации текстов.

Инструкция:

Обучите модель с использованием конвейера (Pipeline), который включает векторизацию и сам классификатор.

Python
```
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report

# Создание и обучение модели
text_clf = Pipeline([
    ('tfidf', TfidfVectorizer()),
    ('clf', MultinomialNB()),
])

text_clf.fit(X_train, y_train)

# Предсказание
predictions = text_clf.predict(X_test)
print(classification_report(y_test, predictions))
```

Задание 3. Основы нейронных сетей для текста

В нейронных сетях текст представляется как последовательность чисел. Мы будем использовать полносвязную сеть (MLP) для классификации.

Инструкция:

Изучите структуру простейшей нейронной сети в Keras/TensorFlow (концептуальный пример).

Python
```
import tensorflow as tf
from tensorflow.keras import layers

# Пример архитектуры нейросети для текста
model = tf.keras.Sequential([
    layers.Dense(64, activation='relu', input_shape=(500,)), # Вход - вектор словаря
    layers.Dropout(0.2), # Защита от переобучения
    layers.Dense(32, activation='relu'),
    layers.Dense(2, activation='softmax') # 2 выхода (вероятности категорий)
])
```

```
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy',
metrics=['accuracy'])
# model.summary()
```

Задание 4. Оценка качества модели (Metrics)

Для оценки классификации используются метрики:

- **Accuracy:** общая доля правильных ответов.
- **Precision (Точность):** доля истинно положительных среди всех предсказанных как положительные.
- **Recall (Полнота):** доля найденных объектов класса среди всех реальных объектов этого класса.

Инструкция:

Постройте матрицу ошибок (Confusion Matrix) для вашей модели из Задания 2.

Python

```
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

cm = confusion_matrix(y_test, predictions)
sns.heatmap(cm, annot=True, fmt='d', xticklabels=text_clf.classes_,
yticklabels=text_clf.classes_)
plt.ylabel('Реальность')
plt.xlabel('Предсказание')
plt.show()
```

Задание 5. Эксперимент: Проверка на новых данных

Проверьте, как модель справится с фразой, которую она никогда не видела, но которая относится к одной из тем.

Инструкция:

Введите фразу "Искусственный интеллект готовит еду".

1. Какую категорию выберет модель?
2. Почему возникла сложность (проблема пересечения тем)?

Контрольные вопросы

1. В чем преимущество нейронных сетей перед классическим ML при работе с очень большими объемами данных?

2. Почему важно делить данные на обучающую и тестовую выборки?
3. Что такое "переобучение" (overfitting) и как его заметить по графику точности?

Итог работы

Студенты научились строить классификаторы текста, понимать разницу между классическими и нейросетевыми подходами, а также интерпретировать метрики качества. Это финальная ступень в обработке естественного языка, позволяющая создавать работающие приложения — от фильтров спама до систем автоматической сортировки документов.