# Лабораторная работа 6

## Анализ Malware атак моделями машинного обучения

Дан датасет Malware атак

https://github.com/saurabh48782/Malware_Classification/blob/master/Malware_Detection.ipynb

Разработать и протестировать модели машинного обучения для классификации атак вредоносного ПО (Malware).
Провести сравнительный анализ различных алгоритмов.

**Этапы выполнения**

**1. Подготовка данных**

1. **Загрузка данных**
   - Скачать датасет с GitHub и загрузить в среду разработки (Jupyter Notebook, Google Colab, PyCharm).
   - Использовать библиотеки pandas и numpy для работы с данными.
2. **Анализ данных**
   - Определить целевую переменную (метка Malware/Legitimate).
   - Проверить баланс классов (value_counts()).
   - Изучить распределение признаков (describe(), info()).
   - Проверить наличие пропущенных значений (df.isnull().sum()).
3. **Предобработка данных**
   - Заполнить или удалить пропущенные значения.
   - Кодировать категориальные признаки (LabelEncoder, OneHotEncoder).
   - Масштабировать числовые признаки (StandardScaler, MinMaxScaler).
   - Разделить данные на обучающую и тестовую выборки (train_test_split).

**2. Обучение моделей машинного обучения**

Обучить и протестировать следующие модели:

1. **Наивный Байесовский классификатор (Naïve Bayes)**
   - GaussianNB или MultinomialNB из sklearn.naive_bayes.
2. **Логистическая регрессия (Logistic Regression)**
   - LogisticRegression из sklearn.linear_model.
3. **Метод опорных векторов (Support Vector Machine, SVM)**
   - SVC из sklearn.svm с разными ядрами (linear, rbf).
4. **Метод k-ближайших соседей (k-Nearest Neighbors, k-NN)**
   - KNeighborsClassifier из sklearn.neighbors.
   - Оптимизация k с кросс-валидацией.
5. **Дерево решений (Decision Tree)**
   - DecisionTreeClassifier из sklearn.tree.
   - Настроить max_depth, criterion.
6. **Случайный лес (Random Forest)**
   - RandomForestClassifier из sklearn.ensemble.
   - Оптимизация n_estimators, max_depth.
7. **Градиентный бустинг (XGBoost)**
   - XGBClassifier из xgboost.

    - Оптимизация learning_rate, n_estimators, max_depth.
8. **CatBoost**
    - CatBoostClassifier из catboost.
    - Оптимизация iterations, depth, learning_rate.
9. **AdaBoost**
    - AdaBoostClassifier из sklearn.ensemble.
    - Подбор n_estimators, learning_rate.

**3. Оценка моделей**

1. **Метрики качества**
    - accuracy
    - precision
    - recall
    - F1-score
    - ROC-AUC
2. **Кросс-валидация**
    - KFold или StratifiedKFold для проверки устойчивости моделей.
3. **Матрица ошибок (Confusion Matrix)**
    - confusion_matrix для анализа ошибок классификации.

**4. Визуализация результатов**

1. **Графики:**
    - Сравнение accuracy моделей.
    - ROC-кривые для нескольких моделей.
    - Матрицы ошибок лучших моделей.

**5. Анализ и выводы**

1. Сравнить модели:
    - Какая модель показала наилучший результат?
    - Время обучения моделей.
    - Какие модели лучше справляются с данной задачей?
    - Влияют ли признаки на качество классификации?
2. Сделать вывод о применимости машинного обучения для выявления Malware-атак.