

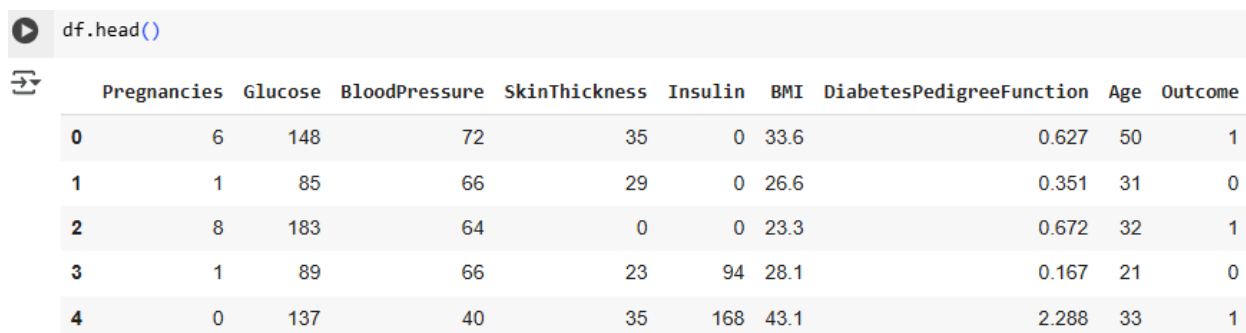
# Обучение модели для определения риска возникновения диабета с визуализацией

Студент Карась Д.Д.

Группа: КББО-01-23

## 1. Постановка задачи и исходные данные

Цель: Прогнозирование наличия диабета у пациентов на основе датасета с различными медицинскими показателями. Датасет `diabetes.csv` содержит 768 записей с 9 признаками (рис. 1):



	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Рисунок 1 – первые пять записей датасета

- Pregnancies (количество беременностей)
- Glucose (уровень глюкозы)
- BloodPressure (артериальное давление)
- SkinThickness (толщина кожи)
- Insulin (уровень инсулина)
- BMI (индекс массы тела)
- DiabetesPedigreeFunction (генетическая предрасположенность)
- Age (возраст)

Целевая переменная: Outcome (0 — отсутствие диабета, 1 — наличие).

Ограничения:

- Несбалансированность данных (около 65% пациентов без диабета).
- Наличие нулевых значений в признаках (например, Glucose = 0).

## 2. Обучение модели

Алгоритм: Random Forest.

Причины выбора:

- Устойчивость к переобучению
- Возможность выделения важных признаков
- Высокая интерпретируемость (визуализация деревьев)

Параметры модели:

- random\_state = 0
- Дефолтные настройки (критерий Джини, максимальная глубина деревьев не ограничена)

## 3. Решение задачи

- Загрузка данных:

```
df = pd.read_csv("diabetes.csv")
```

- Разделение данных:

- 70% — обучение, 30% — тестирование (test\_size = 0.3).
- Фиксация случайности (random\_state = 1).

- Обучение модели:

```
clf = RandomForestClassifier(random_state = 0)
```

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

- Оценка точности:

- Метрика Accuracy: 76-79% (зависит от разбиения данных)

- Визуализации:

- Деревья решений: Построение первых 5 деревьев ансамбля для анализа логики принятия решений (рис. 2):

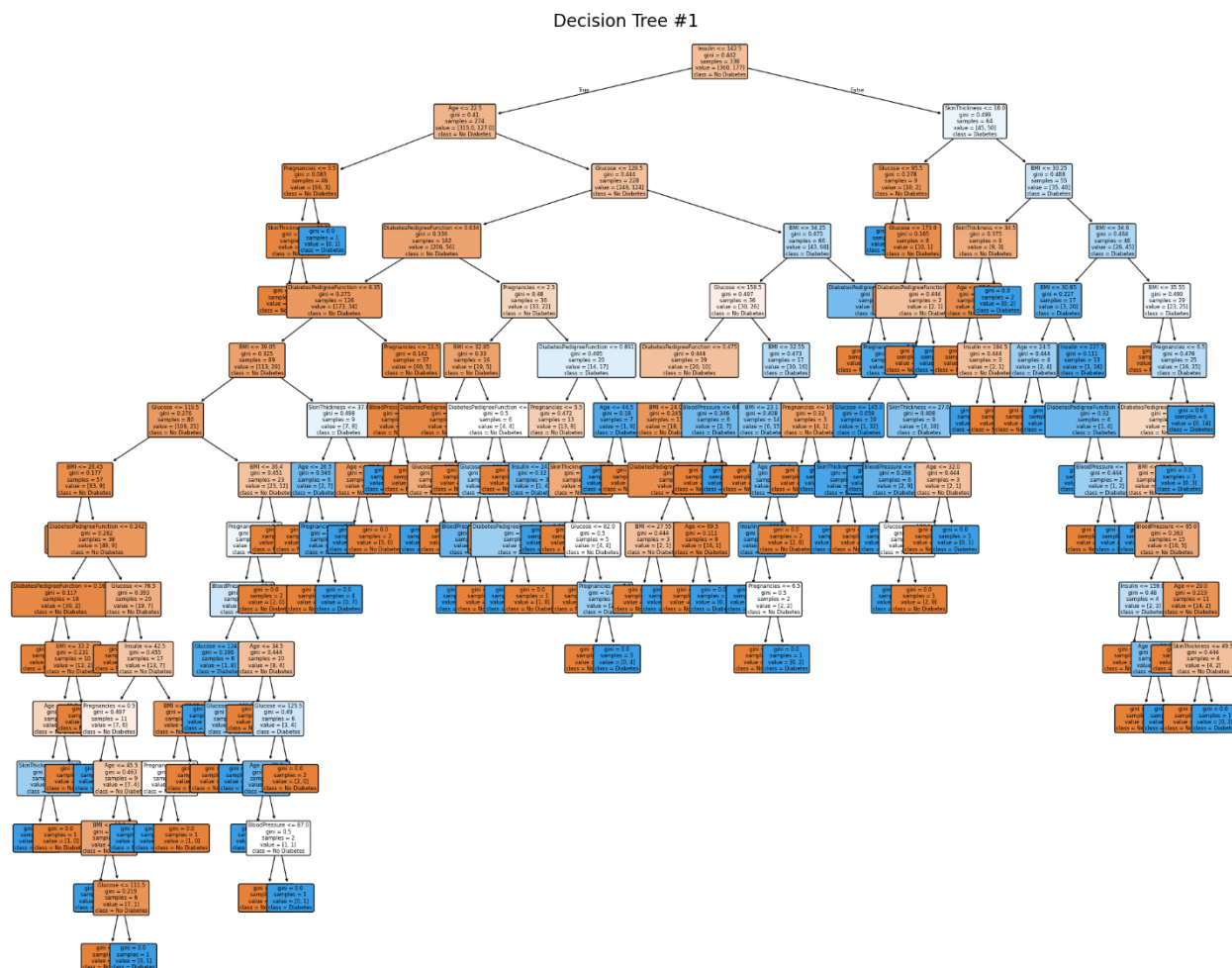


Рисунок 2 – первое дерево решений

- Матрица ошибок:

```
sns.heatmap(cm, annot = True, fmt = 'd', cmap = 'Blues')
```

Показывает количество True Positive, False Positive, True Negative, False Negative (рис. 3):

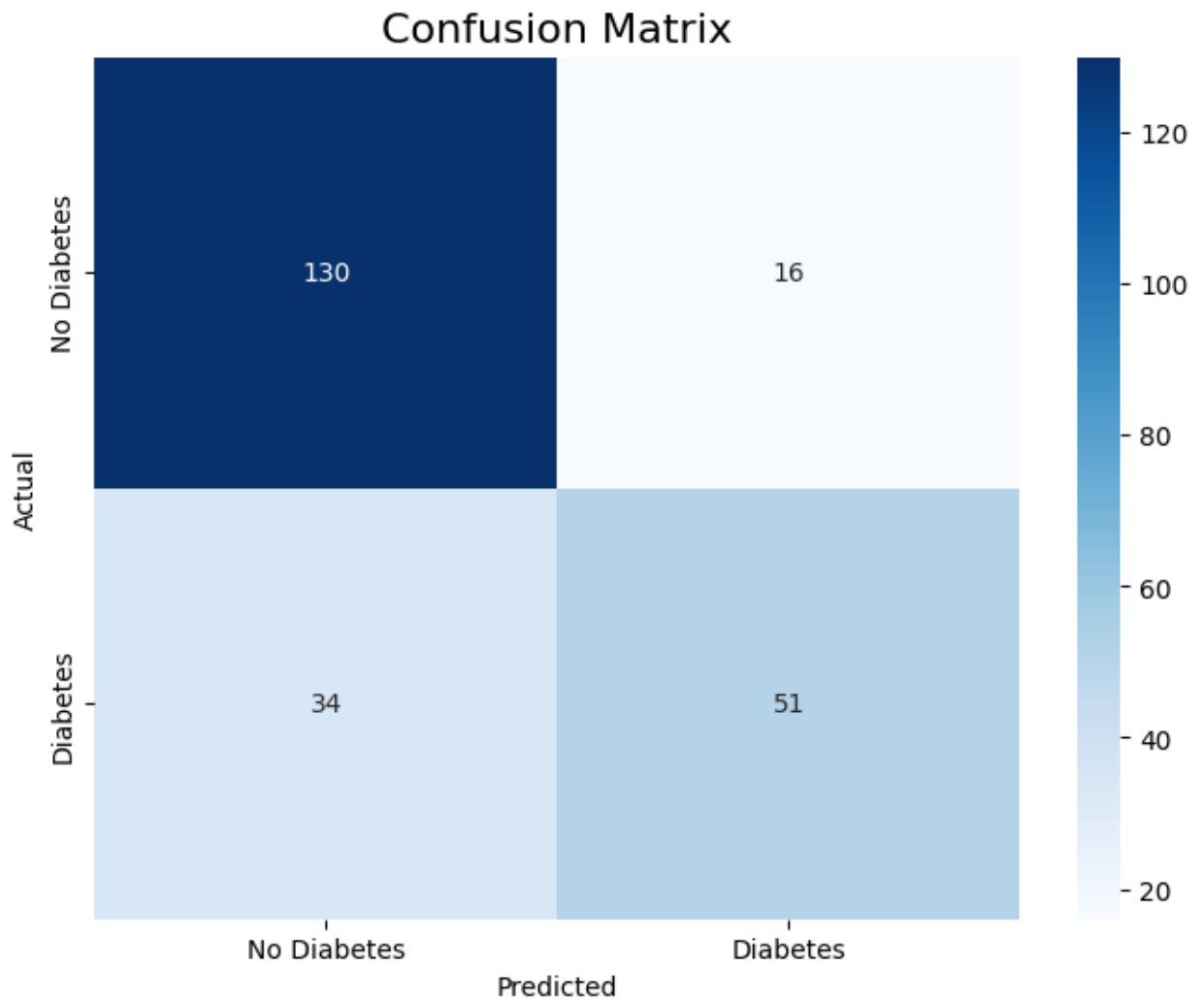


Рисунок 3 – матрица ошибок

- Важность признаков (рис. 4):

```
feature_importances.plot(kind = 'barh', color = 'skyblue')
```

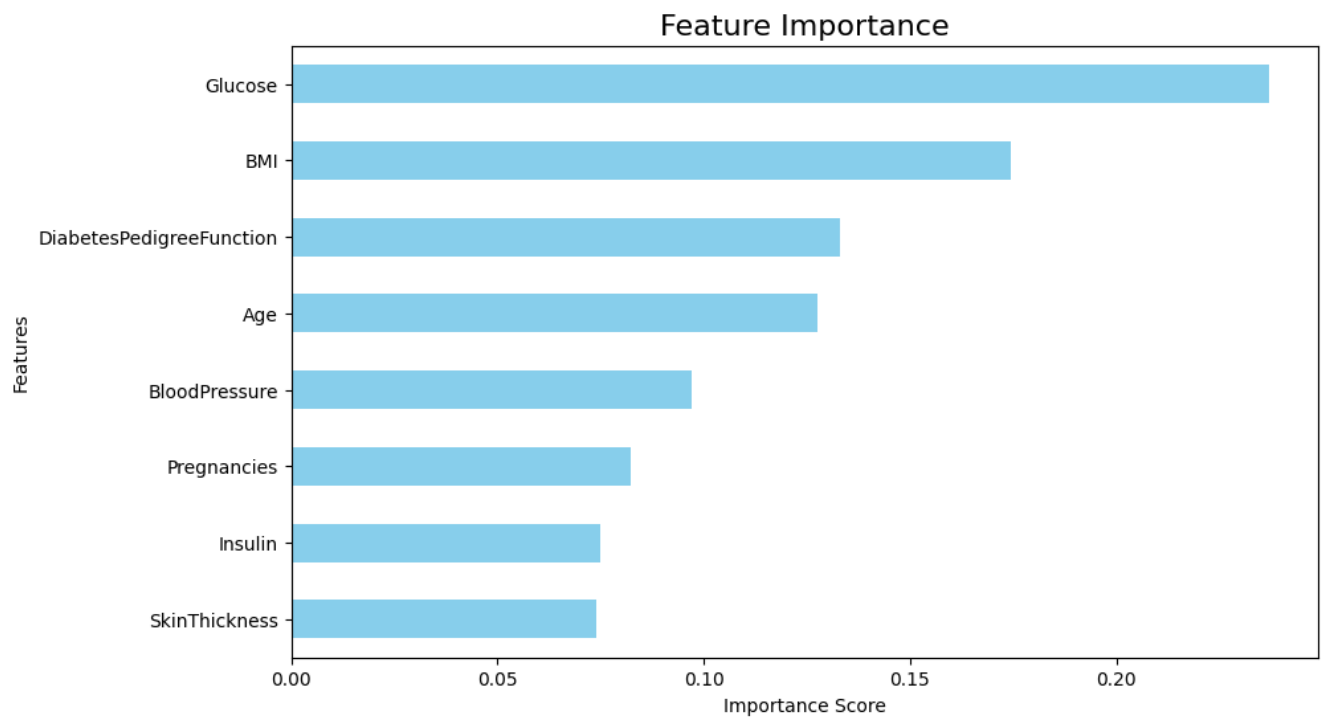


Рисунок 4 – график определения важности отдельных признаков

- Наиболее значимые признаки: Glucose, BMI, Age.

Скриншоты кода и вывода точности работы модели (рис. 5-6):

```

import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.tree import plot_tree

# Загрузка данных
df = pd.read_csv("diabetes.csv")
y = df.Outcome.values
X = df.drop(['Outcome'], axis=1)

# Разделение данных
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)

# Обучение модели
clf = RandomForestClassifier(random_state=0)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

# Оценка точности
result = accuracy_score(y_test, y_pred)
print("Accuracy:", result)

# Визуализация деревьев
for index in range(0, 5):
    fig = plt.figure(figsize=(25, 20))
    plot_tree(clf.estimators_[index],
              filled=True,
              rounded=True,
              feature_names=X.columns,
              class_names=['No Diabetes', 'Diabetes'],
              fontsize=6)
    plt.title(f"Decision Tree #{index+1}", fontsize=20)
    plt.show()

```

Рисунок 5 – первая часть кода

```

# Матрица ошибок
plt.figure(figsize=(8, 6))
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=['No Diabetes', 'Diabetes'],
            yticklabels=['No Diabetes', 'Diabetes'])
plt.title('Confusion Matrix', fontsize=16)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()

# Важность признаков
plt.figure(figsize=(10, 6))
feature_importances = pd.Series(clf.feature_importances_, index=X.columns)
feature_importances.nlargest(8).sort_values().plot(kind='barh', color='skyblue')
plt.title('Feature Importance', fontsize=16)
plt.xlabel('Importance Score')
plt.ylabel('Features')
plt.show()

```

Accuracy: 0.7835497835497836

Рисунок 6 – конец кода и вывод точности (Accuracy)

#### 4. Анализ работы модели

- Точность модели: 78-79% — неплохой результат, но есть потенциал для улучшения (например, балансировка данных или автоматический подбор гиперпараметров).
- Интерпретация матрицы ошибок: ложные отрицательные (30): модель пропускает пациентов с диабетом, что критично для медицинской диагностики. Ложные положительные (15): ложная тревога, менее критична.
- Важные признаки:
  - Уровень глюкозы (Glucose) — ключевой индикатор.
  - Индекс массы тела (BMI) и возраст (Age) — второстепенные факторы.

Итог: код решает задачу классификации с неплохой точностью, а добавленные визуализации помогают анализировать ошибки и значимость признаков. Для медицинского применения требуется доработка модели для снижения ложноотрицательных прогнозов.