Отчет по лабораторной работе №2
по курсу «Машинное обучение»

Студент группы 8О-308 Баженова Надежда, № по списку 1.

Контакты: nad110508@gmail.com
Работа выполнена: 12.04.19

Ахмед Самир Халид
Отчет сдан: 12.04.19

1. Постановка задачи

Ваша задача познакомиться с платформой Azure Machine Learning, реализовывав полный цикл разработки решения задачи машинного обучения, использовав три различных алгоритма, реализованные на этой платформе.

2. Требования

Уникальность решения

Обоснованность выбора той или иной операции

В отчете должны быть указаны алгоритмы, которые применялись, результаты применения этих алгоритмов, а также скрины некоторых этапов обработки данных

3.Описание проделанной работы

С платформой Azure я работала впервые. Познакомившись с данной студией и прочитав руководство по созданию своей машины[1], я перешла к созданию своей модели. В качестве данных для создания эксперимента я использовала датасет fire_forest.

Моя машина решает задачу классификации, поставленную в нулевой лабораторной:

Классифицировать пожары на две группы: 0- происходящие с янв.-авг., 1 - с сен – дек. Данная классификация может пригодиться, например, при составлении сметы на требуемое оборудование и технику на следующий год.

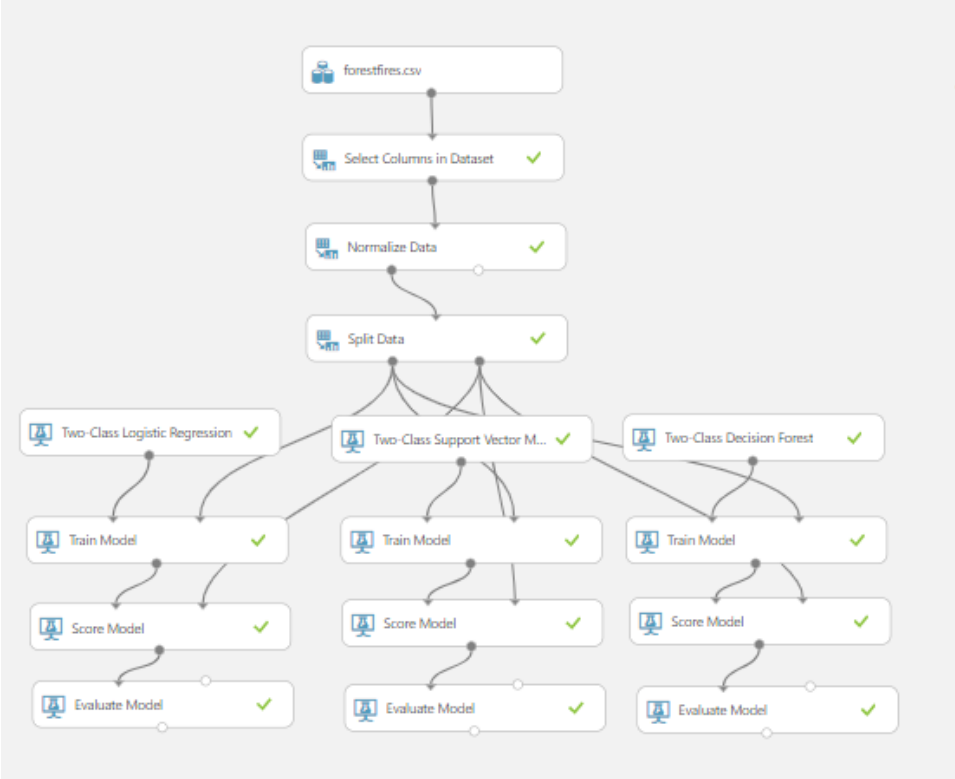
Для решения данной задачи мне пришлось модифицировать датасет, добавив дополнительный бинарный признак, разделяющий все данные на два класса. Класс 0 - пожары, произошедшие с янв.-авг., класс 1 - с сен – дек.

[1] https://docs.microsoft.com/ru-ru/azure/machine-learning/studio/create-experiment

Код программы:

```
import pandas as pd
if __name__ == '__main__':
    data = pd.read_csv('forestfires.csv')
    month_name = {month: i//6 for i , month in
                  enumerate(['jan', 'feb', 'mar', 'apr', 'may', 'jun', 'jul',
'aug', 'sep', 'oct', 'nov', 'dec'])}
    print(month_name)
    data['bin_month'] = data['month'].map(month_name)
    data.to_csv('reload_forest_fire.csv')
```

Подготовив датасет, я создала схему эксперимента, используя предоставляемые модули



Основные этапы :

1. **Создание модели**

   - Получение данных (загрузила свой датасет)
   - Подготовка данных (Я нормализовала данные для увеличения точности прогнозирования)
   - Определение признаков (Я исключила из рассмотрения признаки Column 0, определяющий порядковый номер записи (не несет никакой информации), month, определяющий месяц произошедшего события, (он вызывает переобучение модели, в случае использовании признака по переданным данным модель со 100% точностью определяет отношение к заданному классу)

2.

- Обучение моделиВыбор и применение алгоритма

Для решения задачи классификации я выбрала 3 алгоритма обучения:

- Двух классовая логистическая регрессия (Two-Class Logistic Regression)
- Двух классовые деревья решений (Two-Class Decision Forest)
- Двух классовая машина опорных векторов (Two-Class Support Vector Machine)

Рассмотрим подробнее каждый из них.

Логистическая регрессия

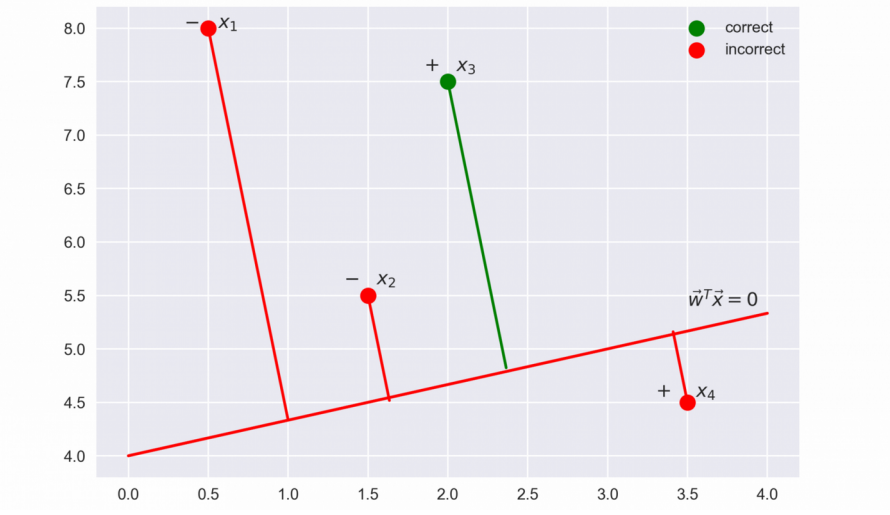
Логистическая регрессия позволяет решить задачу классификации, а также предсказать вероятность правильного определения класса.

Основная идея: Основная идея логистической регрессии заключается в том, что пространство исходных значений может быть разделено линейной границей (т.е. прямой) на две соответствующих классам области.

Для обучения модели задается некоторая функция (отступ классификации) определяющая ошибается модель в классификации или нет

 если отступ большой (по модулю) и положительный, это значит, что метка класса поставлена правильно, а объект находится далеко от разделяющей гиперплоскости (такой объект классифицируется уверенно). На рисунке – x3.
 если отступ большой (по модулю) и отрицательный, значит метка класса поставлена неправильно, а объект находится далеко от разделяющей гиперплоскости (скорее всего такой объект – аномалия, например, его метка в обучающей выборке поставлена неправильно). На рисунке – x1.

 если отступ малый (по модулю), то объект находится близко к разделяющей гиперплоскости, а знак отступа определяет, правильно ли объект классифицирован. На рисунке – x2 и x4.



Далее используется принцип максимального правдоподобия для минимизации числа ошибок классификации.

Дерево принятия решений

Дерево принятия решений (также может называться деревом классификации или регрессионным деревом) — средство поддержки принятия решений, использующееся в машинном обучении, анализе данных и статистике. Структура дерева представляет собой «листья» и «ветки». На рёбрах («ветках») дерева решения записаны атрибуты, от которых зависит целевая функция, в «листьях» записаны значения целевой функции, а в остальных узлах — атрибуты, по которым различаются случаи. Чтобы классифицировать новый случай, надо спуститься по дереву до листа и выдать соответствующее значение.

Для построения дерева на каждом внутреннем узле необходимо найти такое условие (проверку), которое бы разбивало множество, ассоциированное с этим узлом на подмножества. В качестве такой проверки должен быть выбран один из атрибутов. Общее правило для выбора атрибута можно сформулировать следующим образом: выбранный атрибут должен разбить множество так, чтобы получаемые в итоге подмножества состояли из объектов, принадлежащих к одному классу, или были максимально приближены к этому, т.е. количество объектов из других классов ("примесей") в каждом из этих множеств было как можно меньше.

Преимущества:

- быстрый процесс обучения;
- генерация правил в областях, где эксперту трудно формализовать свои знания;
- извлечение правил на естественном языке;
- интуитивно понятная классификационная модель;
- высокая точность прогноза, сопоставимая с другими методами (статистика, нейронные сети);
- построение непараметрических моделей.

Недостатки:

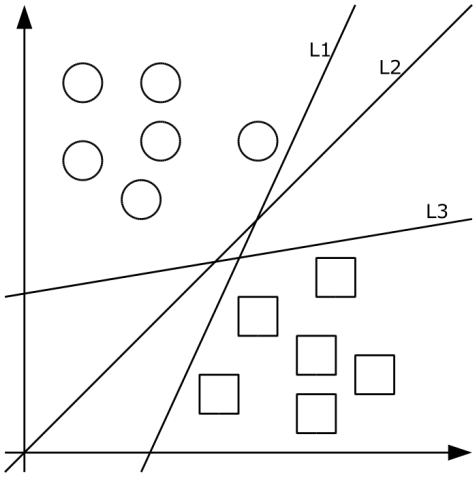
- При использование маленького количества деревьев модель работает медленно.
- Ветвистость деревьев

Решение проблем: регулирование высоты и количества листьев.

Метод опорных векторов

Часто в алгоритмах машинного обучения возникает необходимость классифицировать данные. Каждый объект данных представляется как вектор (точка) в n-мерном пространстве Каждая из этих точек принадлежит только одному из двух классов. Вопрос состоит в том, можно ли разделить точки гиперплоскостью размерности $(n-1)$. Это — типичный случай линейной разделимости. Искомых гиперплоскостей может быть много, поэтому полагают, что максимизация зазора между классами способствует более уверенной классификации. То есть, можно ли найти такую гиперплоскость, чтобы расстояние от неё до ближайшей точки было максимальным. Это эквивалентно тому, что сумма расстояний до гиперплоскости от двух ближайших к ней точек, лежащих по разные стороны от неё, максимально. Если такая гиперплоскость существует, она называется оптимальной разде-

ляющей гиперплоскостью, а соответствующий ей линейный классификатор называется оптимально разделяющим классификатором.



3. **Оценка и тестирование модели**

Для тестирования модели я разбила свои данные в соотношении 80/20, отправив 80% на обучение и 20% на тестирование.

Для оценки модели я использовала модуль **Evaluate Model**

Он анализирует натренированную модель и показывает графики и различные показатели, позволяющие оценить ее качество.

Рассмотрим их более подробно.[2]

TP (*True Positives*) – верно классифицированные положительные примеры (так называемые истинно положительные случаи);

TN (*True Negatives*) – верно классифицированные отрицательные примеры (истинно отрицательные случаи);

FN (*False Negatives*) – положительные примеры, классифицированные как отрицательные (ошибка I рода). Это так называемый "ложный пропуск" – когда интересующее нас событие ошибочно не обнаруживается (ложно отрицательные примеры);

FP (*False Positives*) – отрицательные примеры, классифицированные как положительные (ошибка II рода); Это ложное обнаружение, т.к. при отсутствии события ошибочно выносится решение о его присутствии (ложно положительные случаи).

Accuracy - измеряет качество модели классификации как отношение истинных результатов к общему количеству случаев.

Precision - это соотношение истинных результатов ко всем положительным результатам.

Recall - это доля всех правильных результатов, возвращаемых моделью.

---

[2] https://basegroup.ru/community/articles/logistic

https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/evaluate-model#bkmk_classification

F-score рассчитывается как средневзвешенное значение точности и повторного вызова между 0 и 1, где идеальное значение F-оценки равно 1.

Чувствительность (Sensitivity) – это и есть доля истинно положительных случаев:

Специфичность (Specificity) – доля истинно отрицательных случаев, которые были правильно идентифицированы моделью:

Заметим, что FPR=100-Sp.

Модель с высокой чувствительностью часто дает истинный результат при наличии положительного исхода (обнаруживает положительные примеры). Наоборот, модель с высокой специфичностью чаще дает истинный результат при наличии отрицательного исхода (обнаруживает отрицательные примеры). Если рассуждать в терминах медицины – задачи диагностики заболевания, где модель классификации пациентов на больных и здоровых называется диагностическим тестом, то получится следующее:

Чувствительный диагностический тест проявляется в гипердиагностике – максимальном предотвращении пропуска больных;

Специфичный диагностический тест диагностирует только доподлинно больных. Это важно в случае, когда, например, лечение больного связано с серьезными побочными эффектами и гипердиагностика пациентов не желательна.

ROC-кривая получается следующим образом:

1. Для каждого значения порога отсечения, которое меняется от 0 до 1 с шагом dx (например, 0.01) рассчитываются значения чувствительности Se и специфичности Sp. В качестве альтернативы порогом может являться каждое последующее значение примера в выборке.
2. Строится график зависимости: по оси Y откладывается чувствительность Se, по оси X – 100%–Sp (сто процентов минус специфичность), или, что то же самое, FPR – доля ложно положительных случаев.

Для идеального классификатора график ROC-кривой проходит через верхний левый угол, где доля истинно положительных случаев составляет 100% или 1.0 (идеальная чувствительность), а доля ложно положительных примеров равна нулю. Поэтому чем ближе кривая к верхнему левому углу, тем выше предсказательная способность модели. Наоборот, чем меньше изгиб кривой и чем ближе она расположена к диагональной прямой, тем менее эффективна модель. Диагональная линия соответствует "бесполезному" классификатору, т.е. полной неразличимости двух классов.

Численный показатель площади под кривой называется AUC (Area Under Curve).

С большими допущениями можно считать, что чем больше показатель AUC, тем лучшей прогностической силой обладает модель. Однако следует знать, что:

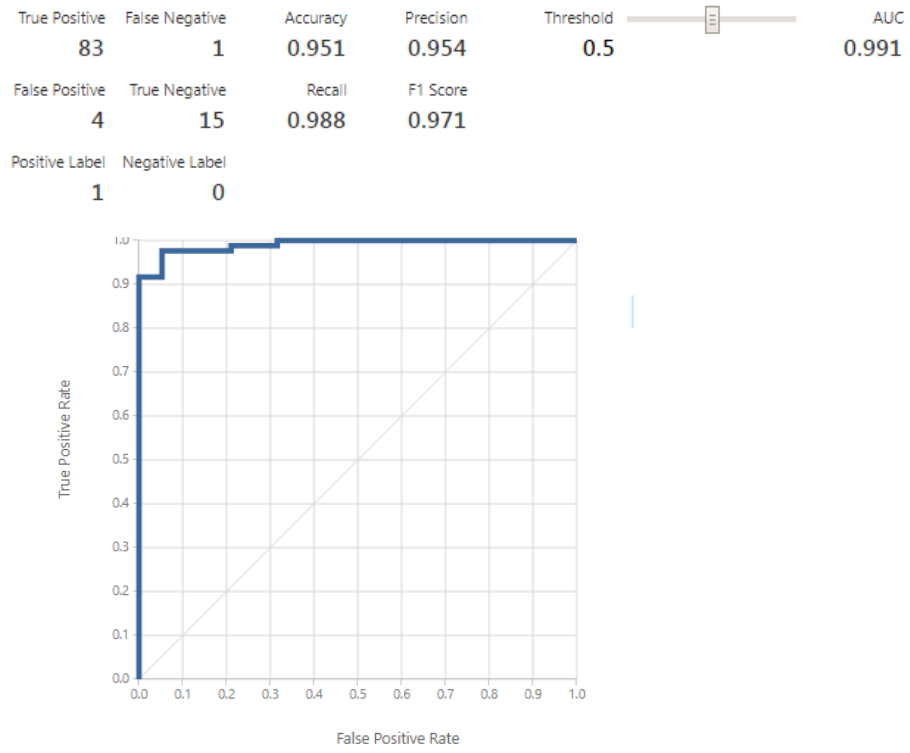
- показатель AUC предназначен скорее для сравнительного анализа нескольких моделей;
- AUC не содержит никакой информации о чувствительности и специфичности модели.

Positive Label – пожар произошел с сентября по декабрь

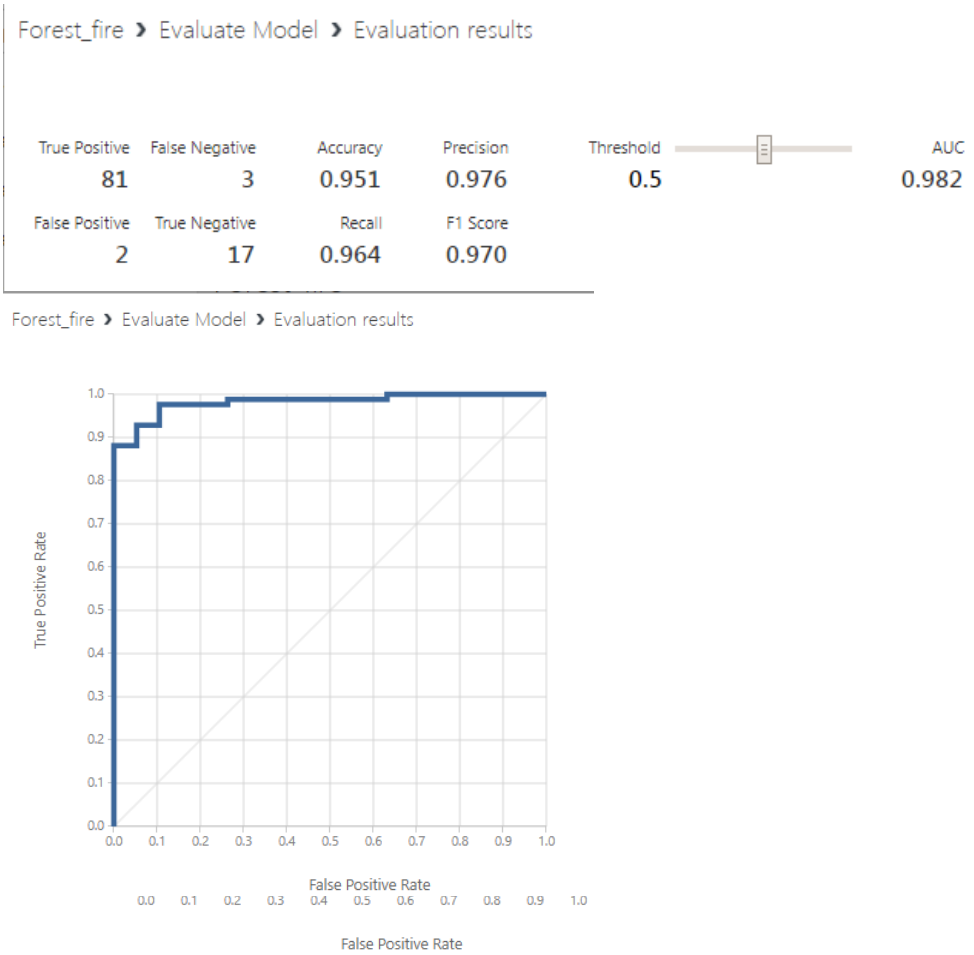
Negative Label – пожар произошел с января по август

Посмотрим на показатели каждого из алгоритмов.

1. **Логистическая регрессия**



2. Метод опорных векторов



Деревья решений

Как уже сказано выше увеличение количества деревьев дает учеличение скорости обучания. Сначала я попробовала обучить модель на 5 деревьях, что дало результат на картинке ниже. При увеличении количества до 30 модель улучшилась, что показывает roc-кривая, и время выполнения обучения сократилось.

Forest_fire › Evaluate Model › Evaluation results

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
81	3	0.951	0.976	0.5	0.984

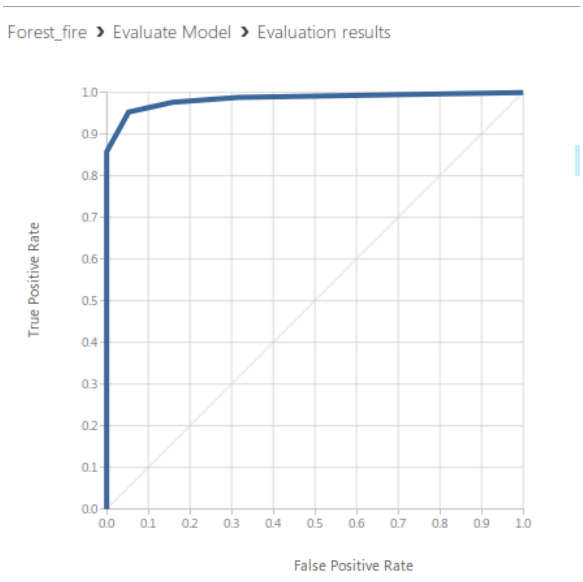
False Positive	True Negative	Recall	F1 Score
2	17	0.964	0.970

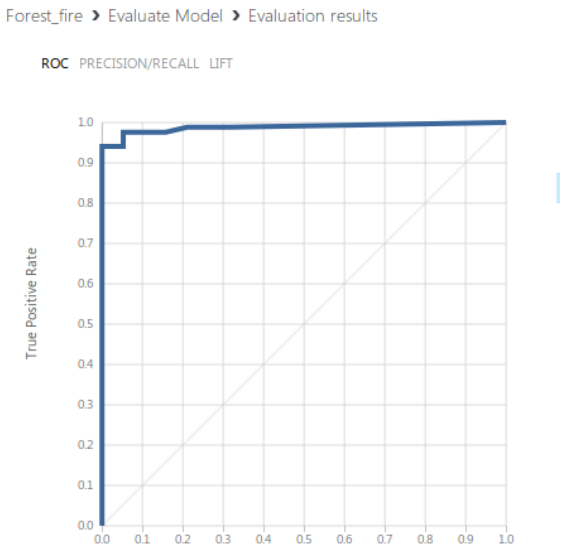
Positive Label	Negative Label
1	0

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
82	2	0.971	0.988	0.5	0.988

False Positive	True Negative	Recall	F1 Score
1	18	0.976	0.982

Positive Label	Negative Label
1	0





4.Вывод

Инструменты, предоставляемые студией Azure, упрощают процесс построения и тренировки модели. Визуализация данного процесса в виде схемы проста для понимания и доступна обычному пользователю. Однако процесс выбора алгоритма тренировки и настраивание параметров не прост и предоставляется самому пользователю. Во многих задачах сложно угадать наиболее подходящий алгоритм тренировки и даже хорошие специалисты подбирают оптимальные алгоритмы путем перебора разных вариантов. Для успешного решения задачи нужно не только уметь пользоваться реализованными блоками, но и разбираться в алгоритмах, входящих в их основу. Данных алгоритмов много, и в основном они основаны на глубоком математическом анализе и теории вероятности. Мне было интересно разбираться, в используемых мной методах обучения.