Starbucks Capstone Challenge

- Domain Background:
  The main focus of the project lies in the area of Customer Behavior Analysis. The data provided was simulated by Starbucks upon the usage data of their mobile app. It was simplified to one product and three offer categories, which is considered representative for the rest of the business workflow. The goal is to develop a model that can generalize the data good enough in order to apply it on daily basis decisions. To achieve this, a model should be able to detect which offer would be the best option for which group of customers.
  For Starbucks with their ca. 35,000 locations in 84 countries around the world it would be a good possibility and well invested money to create ML-models for their customer behavior analysis. For myself it is not only an exciting and challenging area to improve my ML-skills but also a warm memory of my very first employer back in 2011, Starbucks. I think it's very symbolic that I'm starting my new career path in ML by learning from the data provided from the company that gave me my first job.
- Problem Statement:
  With the data provided it is our goal to predict how people make purchasing decisions and how those decisions are influenced by promotional offers.
- Datasets and inputs:
  There are three json-files provided by Starbucks:
  - *profile.json*: Rewards program users (17000 users x 5 fields)
    - gender: (categorical) M, F, O, or null
    - age: (numeric) missing value encoded as 118
    - id: (string/hash)
    - became_member_on: (date) format YYYYMMDD
    - income: (numeric)
  - *portfolio.json:* Offers sent during 30-day test period
    (10 offers x 6 fields)
    - reward: (numeric) money awarded for the amount spent
    - channels: (list) web, email, mobile, social
    - difficulty: (numeric) money required to be spent to receive reward
    - duration: (numeric) time for offer to be open, in days
    - offer_type: (string) bogo, discount, informational
    - id: (string/hash)
  - *transcript.json:* Event log (306648 events x 4 fields)

    - person: (string/hash)
    - event: (string) offer received, offer viewed, transaction, offer completed
    - value: (dictionary) different values depending on event type:
        - offer id: (string/hash) not associated with any "transaction"
        - amount: (numeric) money spent in "transaction"
        - reward: (numeric) money gained from "offer completed"
    - time: (numeric) hours after start of test

- Solution Statement:
  In order to determine how customers make purchasing decisions and how those decisions are influenced by promotional offers I'd like to build a regressor model which would take data about a customer as an input and would give predictions about how many offers would be accepted/completed by the customer. The regressor should give the output for all provided offer categories simultaneously.
  I tend to use the *Gradient Boosting Regressor* and to compare it to a simple linear regression as well as to other ensemble methods commonly used for regression tasks such as *AdaBoost* and *Random Forest*. Because of the nature of the chosen algorithms it is not necessary to use additional computational resources: The algorithms are quite fast even on a single CPU, therefore the project can be developed locally. It is possible, however, to use AWS such as Sagemaker Notebook for Data Wrangling and for developing the model and Sagemaker Endpoints to host the final solution for the usage. For the purpose of this project I would perform all steps needed for the development locally on my device.
- Benchmark Models: Linear Regression, AdaBoost, Random Forest
- Evaluation Metrics: As we deal with a regression task I will use *RMSE* and R^2 to evaluate the model performance.
- Project Design: The Project will be developed using *jupyter notebook* with the following order of steps to complete:
    - Load the data
    - Perform EDA
    - Clean the data, perform feature engineering
    - Train the model
    - Evaluate the model performance

- Perform a Hyperparameter Search
- Train and evaluate the best estimator
- Compare the performance to the benchmark models
- Analyse whether or not the results meet expectation and are applicable.