**Introduction to Machine Learning – Final Project**

Team 34 – Michaela Segall (313562191), Avi Sananes (208057497)

**Executive Summary**

**Data Exploration**

To really understand the data is probably the most important, difficult and exhausting part of a Data Science project. At first, we really wanted to have a better understating and to gain a domain knowledge of the data we are facing, and which features it contains.

In order to do so, we used several data exploration techniques. At first, we wanted to take a glimpse at the train data, so we used the head() function to see how the values in our data look like. At this point we already understood that some serious topics that should be addressed at the pre-processing stage, for example the fact that the duration columns have the word “minutes” in it, and that some of the values are missing.

We noticed that our train data has 10479 rows, which gives us plenty of room for outliers detection and removing, and for separating the dataset to train and validation sets in a later stage. In addition, we saw that we have 22 features (not including the “purchase” column), which we might want to reduce when testing our models.

Afterwards, we used both boxplot and histogram to see the data distribution. We noticed that due to high variation, some of the columns are not really fit the boxplot visualization, but it did help us to realize that some of the columns, like the region column, have obvious outliers. With the histogram visualization, we were able to see that most of the columns are either normally or log-normally distributed. This assumption will serve us for detecting and removing outliers.

Eventually, we used heatmap to plot the feature correlation. We found out that some features, like BoundRates and ExitRates, are strongly correlated, and we might want to consider reducing features with this kind of strong correlation. In addition, we can see that some features are more correlated to “purchase” than other – like PageValues and the “D” column.

Once we get to the dimensional reduction part, we will take these conclusions into consideration.

**Pre-processing**

The first part of the pre-processing stage is to normalize the data and modify it in a way that would allow us to use it in machine learning models. At the exploration part of the project, we've reviewed the data manually and looked how the values look like. We understand that we need all the values to be numeric and this required modification.

As the modifications can have several methods of implementation (for example: categorical data can have different encoders), we've created a function that receives a dataframe and which columns should go through which preprocessing. We've determined which processing is required through manually looking at the table and the distinct values of each feature. The preprocessing contains:

* Converting boolean data to 0,1
* Converting month names to numbers
* Extract numbers from string with words = "23.5 minutes"
* Browser data - looking carefully on the values, there is a pattern detected - "<browser\_name>\_<version>" where the version can be in different patterns and styles. We extract the browser name and later encode it, and the browser version we take only the "major" and ignore the "minor" version part.
* Categorical data – we decided to encode categorical data with DUMMY, and to create a vector of options for each categorical feature.

As we finished setting our data with numerical values, we were able to move on to dealing with missing values. During the data exploration process, we noticed that all the columns in our train data have missing values (NaNs). With the knowledge we gained during the exploration, we decided to use different approaches for each column:

* Column D – since most of that data in this column is empty, we decided to remove it completely and to not take it into account.
* Page durations – first, we wanted to make sure to avoid conflicts between columns. We filled the page duration columns with zeros wherever the total duration equals zero, wherever the number of visited page equals zero and vice versa.
* Column A – we noticed that the column values are between 1-15, so we filled the missing values with zeros.
* Column C – since it has very little amount of missing data, we filled it with the most common value.
* At this point, we used KNN to fill any remaining missing values in all the other columns, except the total duration.
* At last, we filled the total duration with the sum of the other page duration fields.

Afterwards, we used z-score test with 5.5 standard deviations to remove outliers. We didn’t want to remove more than 10% of our train data, so we used 5.5 instead of the default 3 STDs.

Now that we have our data standardized and filled, we wanted to test it on a model. We ran a Logistic Regression model for the first time on our data, and the results were rather surprising.

The Logistic Regression model got a ±0.9 AUC score on our validation data, which was 20% of our training data, with MSE of just 0.106. We ran the same model on the original test data and got pretty much the same results. The fact that we got a similar result for both the train and validation sets is reducing the probability of our model being overfitted, but we will take some safety measures such as dimensional reduction and K-fold cross validations.

**Dimensional reduction**

One of the measures that we can take to reduce overfitting is dimensional reduction. We previously noticed that we got many features, with some correlations between them. This led us to try and use to approaches - "logical" reduction and a "computed" one:

* Logical reduction:
  + High correlation - features that are very similar can be redundant and we may prefer taking only one of them instead of two. We saw that BoundRates and ExitRates are strongly correlated and decided to remove the BoundRates feature.
  + Logical dependency - all the duration columns are summed into a "total duration" column. Therefore, we can either use only the total, or its three separate components and then ignore the "total" column as redundant. We decided to use only the total duration column.
* Computational methods - we would test two different methods for dimension reduction and measure their impact on predicting with linear regression as a test case. The two models we'll tests:
  + Forward selection - compare each number of features to the one before it and pick the best number of features subset.
  + PCA - variance of 0.95 was chosen as a const for not losing too much of the data. Different consts would yield different results and it is configurable in change this assumption changes.

The performance evaluation will be measured using Logistic Regression model, as a simple model that will be easy to predict with and measure the errors. We weren't sure what should be done first - model selection or dimension reduction - so we followed the order of the exercise, assuming linear regression will reflect impact on performance of other models as well.

The metric for evaluation was chosen to be RMSE after researching what metric is more often used for this use case. There is no consensus and different metrics will yield different results.

The approach that yielded the best result is the Forward Selection, which chose the following features for the model:

'num\_of\_product\_pages', 'total\_duration', 'ExitRates', 'PageValues', 'Month', 'user\_type', 'Weekend', 'C']

To that we added the “id” and “purchase” columns and moved on to testing the models.

**Model implementations and estimations**

Now that we dealt with all the necessary perquisites, we can now run some models on our data and test their predictions:

* Logistic Regression:
  + First, we set the validation sample as 20% of our data.
  + Before running the model, we used the “score” metric to search for the best “C” hyperparameter. The result we got is 0.754.
  + We ran the model with the best “C” value and “l2” penalty. We used 5-fold cross validation to calculate the result.
  + We received a mean AUC score of 0.88, which is slightly lower from the first time we run it, but the risk of overfitting is reduced.
  + We looked at the feature importance stats to see which features impacted the model the most:
    - The most impactful feature was “Weekend” – our intuition to this behavior is that people tend to spend more money during their free time. Our assumption is backed by [an article by Gallup](https://news.gallup.com/poll/123839/consumers-spend-more-weekends-payday-weeks.aspx), that shows that Americans spend 18% more money during weekends.
    - User type is also a strong feature. The reason for it might be that returning customers tend to buy more products than new customers. [This Brilliance article on eCommerce statistics](https://www.barilliance.com/new-vs-returning/#:~:text=Returning%20visitors%20will%20complete%20a,that%20you%20can%20do%20that.) shows that returning visitors will complete a purchase 75% more than a new visitor.
* KNN:
  + We used the same train and validation split we used for the Logistic Regression.
  + To find the best K value to use, we plotted for each K its MSE result. The K value that yielded the best MSE result is 19.
  + We ran the KNN model with 19 neighbors, and got an AUC score of 0.76, which is worse than the Logistic Regression model

**Predictions**

**Summary**

* To explore the data, we used some basic DataFrame functions to see its values, and histogram and box plots to see the columns distribution.
* We also used a correlation matrix that helped us find features that are strongly correlated, and we can reduce in later stage.
* We normalized the data to make it numeric and removed unnecessary characters.
* Afterwards, we used several methods to fill in all missing values, including KNN for most of the fields.
* We assumed that most of data is normally distributed and used z-score test to remove outliers.
* After examining two dimensional reduction method, we decided to use forward selection that yielded better results.
* The Logistic Regression model yielded an AUC score of 0.88, which is considered good, meaning it predicts almost all the results successfully.

**Appendix**

**Division of responsibility**

* Data exploration - Together
* Data standardization - Michaela
* Data completion – Michaela
* Removing outliers - Avi
* Dimension reduction – Michaela
* Evaluating basic models and hyperparameters – Avi
* Evaluating complex models and hyperparameters – Avi
* Estimation utility – Michaela
* Final pipeline code - Michaela

**Interesting plots**

???