MACHINE LEARNING: 50 STARTUPS

Step 1: Look at the big picture and frame the problem

1. Define the objective in business terms

   Our solution is going to serve as a building block for JPM-Finance to easier evaluate future newly started companies (also known as startups) using data and linear regression to predict if the said startup is worth investing in.

2. How shall your solution be used?

   As an analyst,
   I want to understand, explore, prepare data, and do a linear regression analysis,
   So that JPM_Finance can evaluate new startups' profit based on features.

   As a financial advisor,
   I want to analyze data from different startups,
   So that I can understand their financial characteristics and identify potential investment opportunities for our clients.

4. How should you frame this problem (supervised/unsupervised, online/offline, etc.)?

   Upon examining our dataset, we've concluded that this problem should be framed as a supervised regression task. Additionally, since our dataset is not continuously streaming but rather fixed, we are approaching it using offline learning.

5. How should performance be measured?

There are several possibilities, such as RMSE, MAE and Confusion Matrix.

We decided to exclude the Confusion Matrix, since it shows the number of True Positives, True Negatives, False Positives, and False Negatives. It provides a detailed breakdown of the model's performance, which is especially useful in binary classification problems.

RMSE is a commonly used measure for regression problems that gives a higher penalty to large errors. It does so by squaring the residuals (the differences between the actual and predicted values), averaging them, and then taking the square root. RMSE might have more and greater outliers, which won't be as helpful.

By the process of elimination, MAE might be the best fit.

In summary, we chose the evaluation metric based on the specific requirements and characteristics of our problem. We could say if the goal is to understand the average magnitude of errors, MAE could be a good choice. If we needed to take account for the impact of large errors, RMSE might be more appropriate. For classification problems, the confusion matrix provides a detailed overview of the model's performance across different classes.
Hence why we settled on MAE.

MACHINE LEARNING: 50 STARTUPS

6. Is the performance measure aligned with the business objective?

MAE (Mean Absolute Error) is a useful metric for this scenario, as it excels in Robustness to outliers, compared to a performance measure like RMSE. It's also more easily interpretable, as it represents the absolute difference between the predicted and the actual values, which makes for an easier understanding. MAE also provides meaningful measurements of the average prediction errors, so users have a fair idea of how far off the model is from actual values. MAE is suitable when the magnitude of errors is more important than the specific direction.

For example, in a regression problem where you'll be predicting house prices, you might be interested in knowing, on average, how far off your predictions are, without considering whether you overestimated or underestimated.

8. What are comparable problems? Can you reuse experience?

We have used and changed some code snippets we have been taught in class and used for some in class exercises.

11. List the assumptions you (or others have made so far)

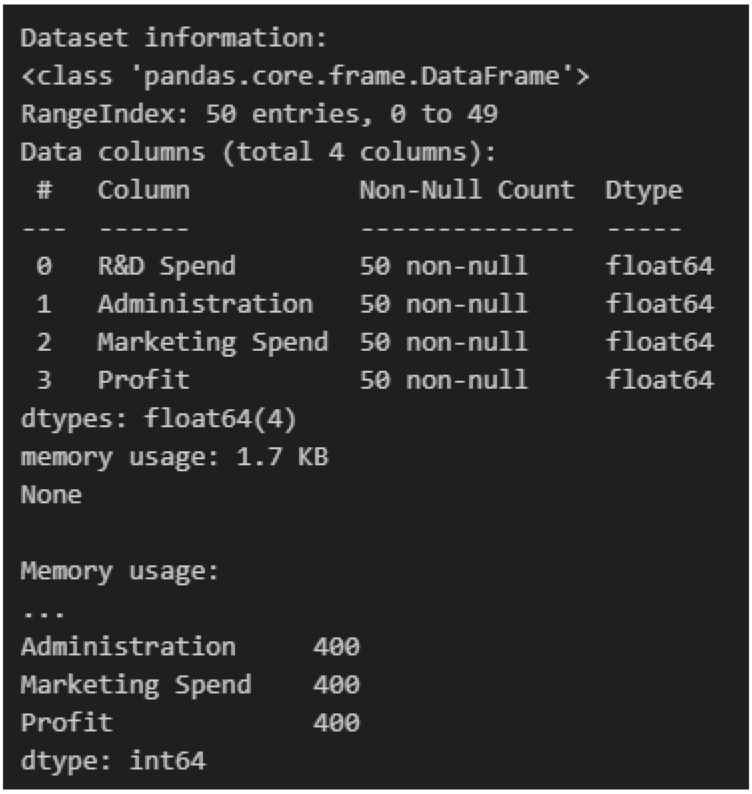
Step 2. Get the data

3. Check out how much space it requires

Code:

```
print("\nDataset information:")
print(dataset.info())
# Check memory usage of the dataset
print("\nMemory usage:")
print(dataset.memory_usage(deep=True))
```

Output:



```
Dataset information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 4 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   R&D Spend        50 non-null     float64
 1   Administration   50 non-null     float64
 2   Marketing Spend  50 non-null     float64
 3   Profit           50 non-null     float64
dtypes: float64(4)
memory usage: 1.7 KB
None

Memory usage:
...
Administration     400
Marketing Spend    400
Profit             400
dtype: int64
```

5. Get access authorizations

   Action: *That is, give yourself access to e.g. your Google drive. Otherwise, this may be a tedious and bureaucratic procedure that must be considered in the planning.*

6. Create a workspace (with enough storage space)

   Action: *That is establishing a notebook at your PC (e.g. Jupyter) or in the cloud (e.g. Colab) for the program and data set.*

7. Get the data

   Action: *Establish a data fetch routine in your notebook. You may get a suggestion from the Moodle Machine Learning room. Code is to be established in your notebook.*

8. Convert the data to a format you can easily manipulate (without changing the data itself)

   Action: *Common conversions between datatypes needed as you go. Code is to be established in your notebook. Goes along with your coding.*

10. Check the size and type of data (time series, sample, geographical, etc.)

    Action: *Code is to be established in your notebook like section 'Take a Quick Look at the Data Structure' in notebook for chapter 2.*

MACHINE LEARNING: 50 STARTUPS

11. Create a test set, put it aside, and never look at it (no data snooping!)

    A problem here is that the data set is small -> Use stratification.
    Action: Make stratified test and learning sets (1 fold). Code is to be established in your notebook. Find your stuff in section 'Create a Test Set' in notebook for chapter 2.

Step 3. Explore the data

1. Create a copy of the data for exploration (sampling it down to a manageable size if necessary)

   Action: Easy action, no sampling needed.

2. Create a notebook to keep a record of your data exploration

   Action: You have probably already done that. A copy and paste and changes of the chapter 2 housing notebook is not illegal.

MACHINE LEARNING: 50 STARTUPS

3. Study each attribute and its characteristics:

- *Name*
- *Type (categorical, int/float, bounded/unbounded, text, structured, etc.)*
- *% of missing values*
- *Usefulness for the task*
- *Type of distribution (Gaussian, uniform, logarithmic, etc.). Check with the selected model if OK.*
- *Do a histogram for each attribute.*

Action: *Get inspired by the notebook for chapter 2. Code is to be established. Find your stuff in section 'Take a Quick Look at the Data Structure' in notebook for chapter 2.*

4. For supervised learning tasks, identify the target attribute(s); i.e. the label(s)

The target attribute in this case would be "Profit", since this is the attribute we are trying to predict for JPM. To predict the target attribute "Profit", we must use the features "R&D Spending", "Administration" and "Marketing Spend".

6. Study the correlations between attributes

Action: *Also make a scatter matrix plot together with the correlation results. Code is to be established in your notebook like section 'Looking for Correlations' in notebook for chapter 2.*

8. Experiment with attribute combinations

Action: *Code is to be established in your notebook like in section 'Experimenting with Attribute Combinations' in notebook for chapter 2.*

9. Identify a new promising attribute you may want to apply, if any

Looking at the Correlation Matrix we made using the new attributes created in step 3.8, we can see that the attribute "R&D Spend" has a very strong and positive correlation to our target profit, which in this case is "Profit". This means that the attribute "R&D Spend" can be a useful attribute for when we must train our model, as there is a good relationship between the two.

Step 4. Prepare the data

2. Feature selection (optional)

We've opted to remove the attributes "RD_to_Marketing" and "Admin_to_Marketing". This decision was influenced by the weak Correlation Coefficients (0.196444 & -0.238996, respectively) between these attributes and our target variable "Profit".

4. Handle text and categorical attributes using “import OneHotEncoder”

We have decided to drop the “State” attribute per the case description, as both JP and Mike have found out from previous studies that having the “State” attribute in the dataset isn't as important, and that doing calculation with the “OneHotEncoder” would be superfluous. In the code, this is done in cell x.

5. Feature scaling

- *Standardize or normalize features, if necessary.*

Action: Include the StandardScaler in the pipeline and observe if it makes any difference when applying the LinearRegression algorithm. Code is to be established in your notebook e.g. find out how to apply the make_pipeline as done in section ‘Training and Evaluating on the Training Set’ in the notebook for chapter 2.

Step 5. Select and train a model

1. Train many quick-and-dirty models from different categories (e.g., linear, naive Bayes, SVM, Random Forest, neural net, etc.) using standard default parameters

Action: In this project only try the LinearRegression algorithm using standard parameters. Code is to be established in your notebook as done for linear regression in section ‘Training and Evaluating on the Training Set’ in the notebook for chapter 2.

MACHINE LEARNING: 50 STARTUPS

2. Measure and compare the performance

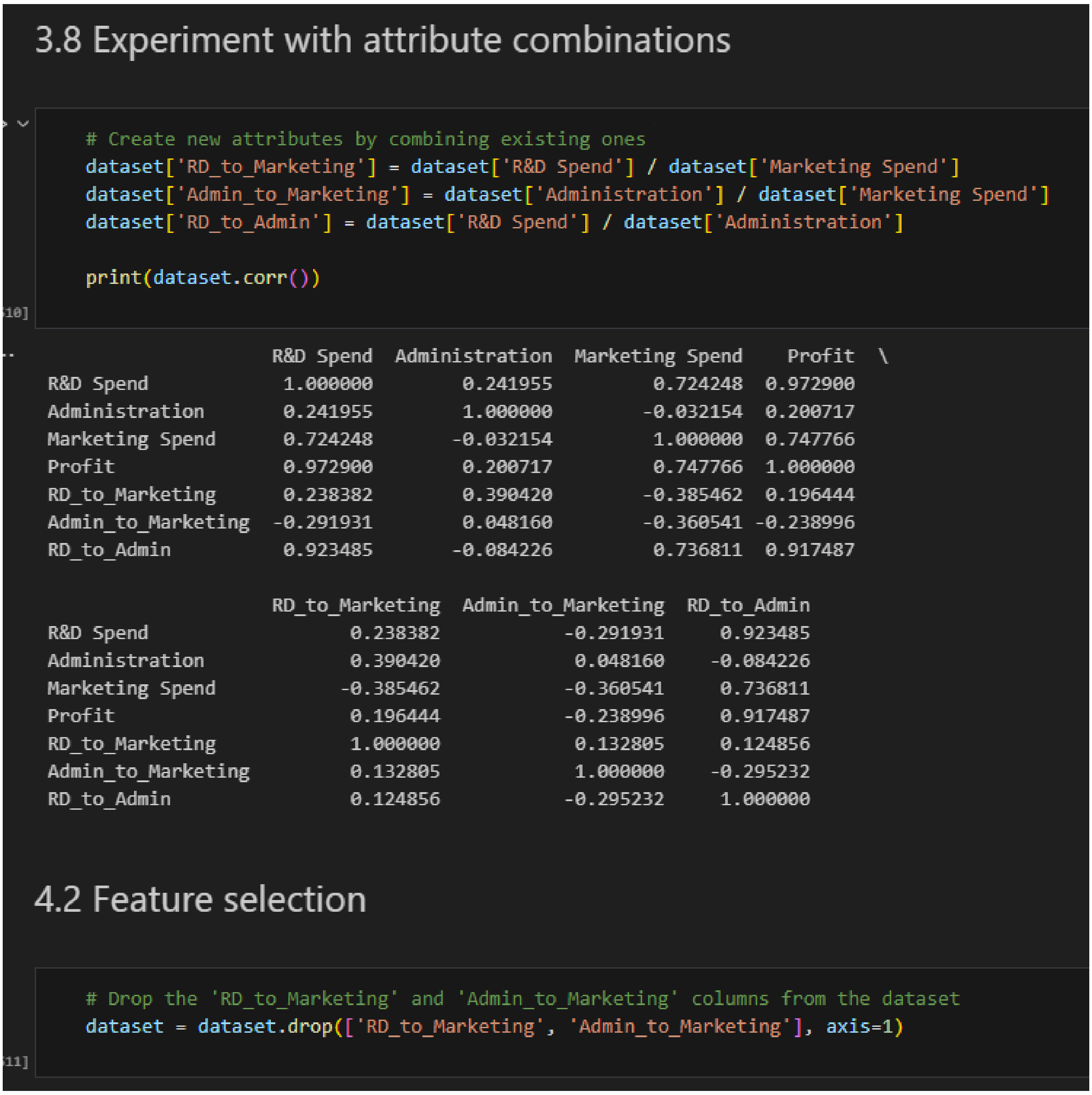
   For each model, compute the mean and the root mean square of the performance measure on a manually selected subset (5-10 data) of the training data.

   Action: *Only do for our linear regression model. Code is to be established in your notebook. Find your stuff in section 'Training and Evaluating on the Training Set' in the notebook for chapter 2.*

3. Analyze the most significant variables for each algorithm

   Action: *Only do it for our linear regression model. Code is to be established in your notebook apply a similar evaluation as done in section 'Analyze the Best Models and Their Errors' in the notebook for chapter 2.*

5. Perform a quick round of feature selection and engineering



3.8 Experiment with attribute combinations

```
# Create new attributes by combining existing ones
dataset['RD_to_Marketing'] = dataset['R&D Spend'] / dataset['Marketing Spend']
dataset['Admin_to_Marketing'] = dataset['Administration'] / dataset['Marketing Spend']
dataset['RD_to_Admin'] = dataset['R&D Spend'] / dataset['Administration']

print(dataset.corr())
```

```
                    R&D Spend  Administration  Marketing Spend    Profit  \
R&D Spend            1.000000        0.241955         0.724248  0.972900
Administration       0.241955        1.000000        -0.032154  0.200717
Marketing Spend      0.724248       -0.032154         1.000000  0.747766
Profit               0.972900        0.200717         0.747766  1.000000
RD_to_Marketing      0.238382        0.390420        -0.385462  0.196444
Admin_to_Marketing  -0.291931        0.048160        -0.360541 -0.238996
RD_to_Admin          0.923485       -0.084226         0.736811  0.917487

                    RD_to_Marketing  Admin_to_Marketing  RD_to_Admin
R&D Spend                  0.238382           -0.291931     0.923485
Administration             0.390420            0.048160    -0.084226
Marketing Spend           -0.385462           -0.360541     0.736811
Profit                     0.196444           -0.238996     0.917487
RD_to_Marketing            1.000000            0.132805     0.124856
Admin_to_Marketing         0.132805            1.000000    -0.295232
RD_to_Admin                0.124856           -0.295232     1.000000
```

4.2 Feature selection

```
# Drop the 'RD_to_Marketing' and 'Admin_to_Marketing' columns from the dataset
dataset = dataset.drop(['RD_to_Marketing', 'Admin_to_Marketing'], axis=1)
```

On this screenshot, some combined attributes have been created to see if there were any that could fit better together than on their own. We then found out that two of them had no correlation and could therefore be deleted. However, we chose to keep 'rd_to_admin' as it shows a promising correlation, but it hasn't changed our target attribute since the correlation is still lower.

MACHINE LEARNING: 50 STARTUPS

Step 6: Fine tune and test the model

1. Fine-tune the hyperparameters using cross-validation

- *Treat your data transformation choices as hyperparameters, especially when you are not sure about them (e.g., if you're not sure whether to replace missing values with zeros or with the median value, or to just drop the rows).*
- *Unless there are very few hyperparameter values to explore, prefer random search over grid search. If training is very long, you may prefer a Bayesian optimization approach (e.g., using Gaussian process priors, as described by Jasper Snoek et al.).*

Action: Try grid search on the LinearRegression algorithm. Identify the relevant (hyper-) parameters and include them in the search. Code is to be established in your notebook that is doing a grid search like what is done in section 'Fine-Tune Your Model' in notebook for chapter 2.

3. Once you are confident about your final model, measure its performance on the test set to estimate the generalization error. This is important

Action: Present your model with the set of model parameters that gives the best performance. Also present the set of hyper parameters that leads to this performance. Code is to be established in your notebook. To be documented – e.g. in your notebook.