

# FEATURE SELECTION AND FEATURE ENGINEERING

---

TIM AJAR KECERDASAN ARTIFISIAL

# AGENDA

---

INTRODUCTION

FEATURE SELECTION

FEATURE ENGINEERING





# INTRODUCTION

---

Feature selection and feature engineering is an important component of your machine learning pipeline.

Effective feature selection and engineering can significantly impact the performance of a machine learning model by improving predictive accuracy, reducing overfitting, and enhancing interpretability.

# WHY TO LOWER THE AMOUNT OF INPUT FEATURE

- Reduce *Multicollinearity*
  - *Multicollinearity* (also *collinearity*) is a phenomenon observed with features in a dataset where one predictor feature in a regression model can be linearly predicted from the other's features with a substantial degree of accuracy.
- Reducing the time
  - will allow us to run more variations of the models leading to quicker and better results

# WHY TO LOWER THE AMOUNT OF INPUT FEATURE

- easier to explain
  - When the number of features goes up, the explainability of the model goes down.
  - Reducing the amount of input features also makes it easier to visualize the data when reduced to low dimensions
- the *curse of dimensionality*
  - As the number of dimensions increases, the possible configurations increase exponentially, and the number of configurations covered by an observation decreases.
  - As you have more features to describe your target, you might be able to describe the data more precisely, but your model will not generalize with new data points – your model will overfit the data



***ONE IMPORTANT REASON TO DROP  
FEATURES IS THE HIGH  
CORRELATION AND REDUNDANCY  
BETWEEN INPUT VARIABLES OR THE  
IRRELEVANCY OF CERTAIN FEATURES***

# CASE STUDY: ZESTIMATES

There is a real estate site in the US that allows real estate agents and homeowners to list homes for rent or for sale. Zillow is famous, among other things, for its Zestimate. The Zestimate is an estimated price using machine learning. It is the price that Zillow estimates a home will sell for if it was put on the market today. The Zestimates are constantly updated and recalculated.

- How does Zillow come up with this number

<https://www.kaggle.com/c/zillow-prize-1>

# LIST OF POTENTIAL INPUT VARIABLE FOR OUR MACHINE LEARNING MODEL

- Square footage
- Number of bedrooms
- Number of bathrooms
- Mortgage interest rates
- Year
- Property taxes
- House color
- Zip code
- Comparable sales
- Tax assessment





**WHAT DO YOU THINK ABOUT THE  
HOUSE COLOR?**

---

# FEATURES THAT CAN BE DROPPED WITHOUT IMPACTING THE MODEL'S PRECISION

- Redundant:
  - This is a feature that is highly correlated to other input features and therefore does not add much new information to the signal.
- Irrelevant:
  - This is a feature that has a low correlation with the target feature and for that reason provides more noise than signal.

# FEATURE SELECTION

- Feature selection refers to the process of selecting subset of relevant features from the original features set while removing irrelevant or redundant features.
- Various techniques, such as filter methods, wrapper methods, and embedded methods, are used to determine feature importance
- Feature selection is important as it reduces dimensionality, improves model interpretability, speeds up training, helps alleviate the curse of dimensionality, prevent overfitting, and enhances generalization.

# EXPLORATORY DATA ANALYSIS

- exploratory data analysis can be a good way to get an intuitive understanding and to obtain insights into the dataset we are working with.
- Three approaches that are commonly used :
  - Feature importance
  - Univariate selection
  - Correlation matrix with heatmap



## FEATURE IMPORTANCE



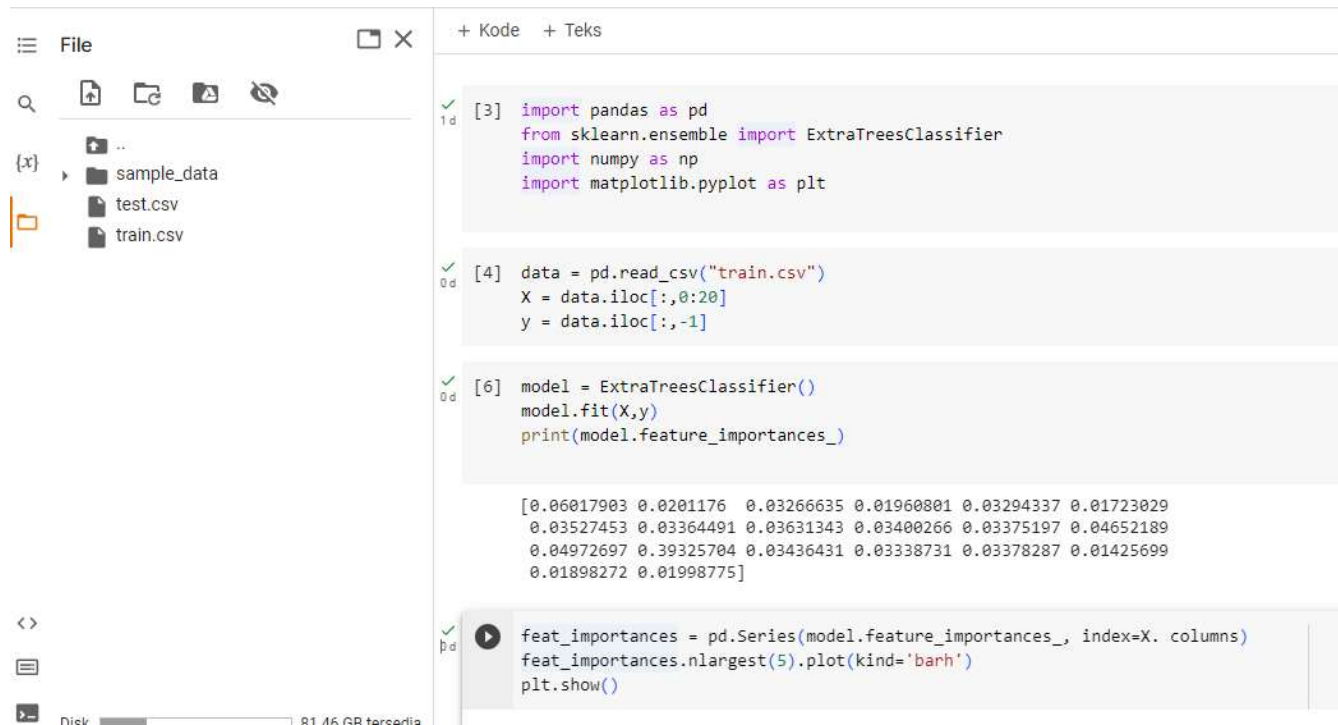
# FEATURE IMPORTANCE

- The importance of each feature of a dataset can be established by using this method.
- Feature importance provides a score for each feature in a dataset. A higher score means the feature has more importance or relevancy in relation to the output feature.
- Feature importance is normally an inbuilt class that comes with Tree-Based Classifiers.
- In the following example, we use the Extra Tree Classifier to determine the top five features in a dataset

# FEATURE IMPORTANCE

- Download train.csv (<https://bit.ly/3ELORg2>)
- Open google colab

# FEATURE IMPORTANCE



The screenshot displays a Jupyter Notebook environment. On the left, a file explorer shows a directory named 'sample\_data' containing 'test.csv' and 'train.csv'. The main area shows three code cells. The first cell imports necessary libraries: pandas, sklearn.ensemble.ExtraTreesClassifier, numpy, and matplotlib.pyplot. The second cell loads the 'train.csv' file, splits it into features (X) and target (y), and uses the first 20 rows for X. The third cell creates an ExtraTreesClassifier, fits it to the data, and prints the feature importances. Below the code, the output shows a list of 10 feature importance values. The final cell is partially visible, showing the creation of a Series from the importances and a bar plot.

```
[3] import pandas as pd
from sklearn.ensemble import ExtraTreesClassifier
import numpy as np
import matplotlib.pyplot as plt

[4] data = pd.read_csv("train.csv")
X = data.iloc[:,0:20]
y = data.iloc[:,-1]

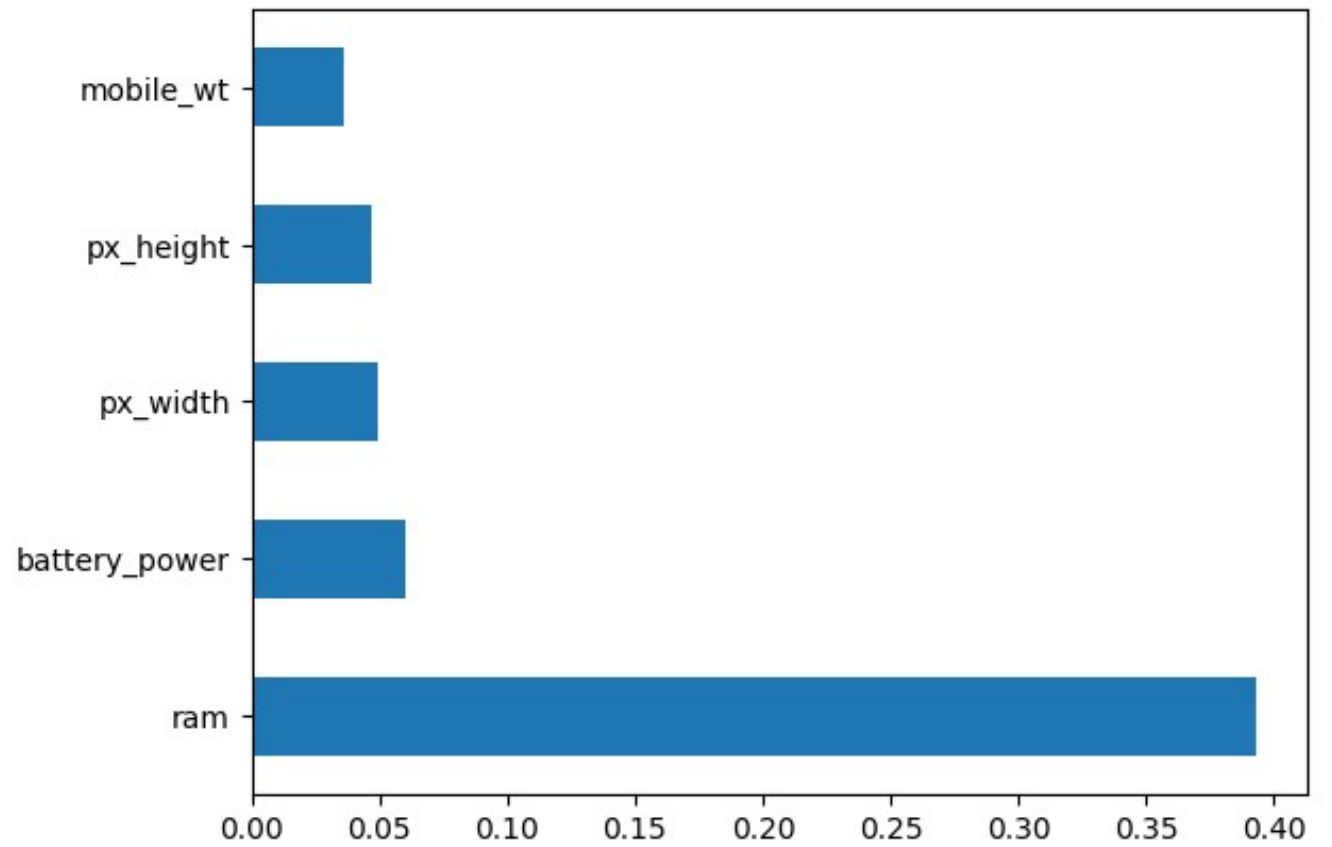
[6] model = ExtraTreesClassifier()
model.fit(X,y)
print(model.feature_importances_)

[0.06017903 0.0201176  0.03266635 0.01960801 0.03294337 0.01723029
 0.03527453 0.03364491 0.03631343 0.03400266 0.03375197 0.04652189
 0.04972697 0.39325704 0.03436431 0.03338731 0.03378287 0.01425699
 0.01898272 0.01998775]

feat_importances = pd.Series(model.feature_importances_, index=X.columns)
feat_importances.nlargest(5).plot(kind='barh')
plt.show()
```



# OUTPUT





## UNIVARIATE SELECTION



# UNIVARIATE FEATURE SELECTION

- Univariate feature selection evaluates each feature independently, considering its relationship with the target variable to identify the most relevant features.
- Univariate feature selection is computationally efficient, interpretable, and particularly useful for identifying highly informative single features. It is also less susceptible to overfitting compared to multivariate approaches.
- Statistical tests can be used to determine which features have the strongest correlation to the output variable.
- The scikit-learn library has a class called SelectKBest that provides a set of statistical tests to select the K "best" features in a dataset.
- The following is an example that uses the chi-squared ( $\chi^2$ ) statistical test for non-negative features to select the five best features in an input dataset

# UNIVARIATE SELECTION

+ Kode + Teks

## Univariate selection

```
✓ [11] from sklearn.feature_selection import SelectKBest  
0d      from sklearn.feature_selection import chi2
```

```
✓ [12] bestfeatures = SelectKBest(score_func=chi2, k=5)  
0d      fit = bestfeatures.fit(X,y)  
      dfscores = pd.DataFrame(fit.scores_)
```

```
✓ [13] dfcolumns = pd.DataFrame(X.columns)  
0d      scores = pd.concat([dfcolumns,dfscores],axis=1)  
      scores.columns = ['specs','score']  
      print(scores.nlargest(5,'score')) #print the 5 best features
```

# OUTPUT

	specs	score
13	ram	931267.519053
11	px_height	17363.569536
0	battery_power	14129.866576
12	px_width	9810.586750
8	mobile_wt	95.972863

---



# CORRELATION MATRIX WITH HEATMAP



# CORRELATION MATRIX WITH HEATMAP

- A correlation exists between two features when there is a relationship between the different values of the features
- If a feature changes consistently in relation to another feature, these features are said to be highly correlated.
- Correlation can be positive (an increase in one value of a feature increases the value of the target variable) or negative (an increase in one value of a feature decreases the value of the target variable).

# CORRELATION IS A CONTINUOUS VALUE BETWEEN -1 AND 1:


- If the correlation between two variables is 1, there is a perfect direct correlation.
- If the correlation between two features is -1, a perfect inverse correlation exists.
- If the correlation is 0 between two features, there is no correlation between the two features.



# CORRELATION MATRIX WITH HEATMAP

Correlation matrix with heatmap

✓  
0d [14] `import seaborn as sns`

✓  
2d  `correlation_matrix = data.corr()  
top_corr_features = correlation_matrix.index  
plt.figure(figsize=(20,20))  
#plot heat map  
g=sns.heatmap(data[top_corr_features].corr(),annot=True,cmap="RdYlGn")`

# OUTPUT



# FEATURE ENGINEERING

- Feature engineering involves creating new features or transforming existing ones to enhance the performance of machine learning models.
- Techniques like polynomial expansion, binning, log transformations, one-hot encoding, and feature scaling are commonly used in feature engineering to capture non-linear relationships, handle missing data, and normalize features.
- Feature engineering is crucial as it allows the extraction of meaningful information from raw data, improves model performance, enhances interpretability, and caters to specific domain knowledge requirements

# FEATURE ENGINEERING

- According to a recent survey performed by the folks at Forbes, data scientists spend around 80% of their time on data preparation

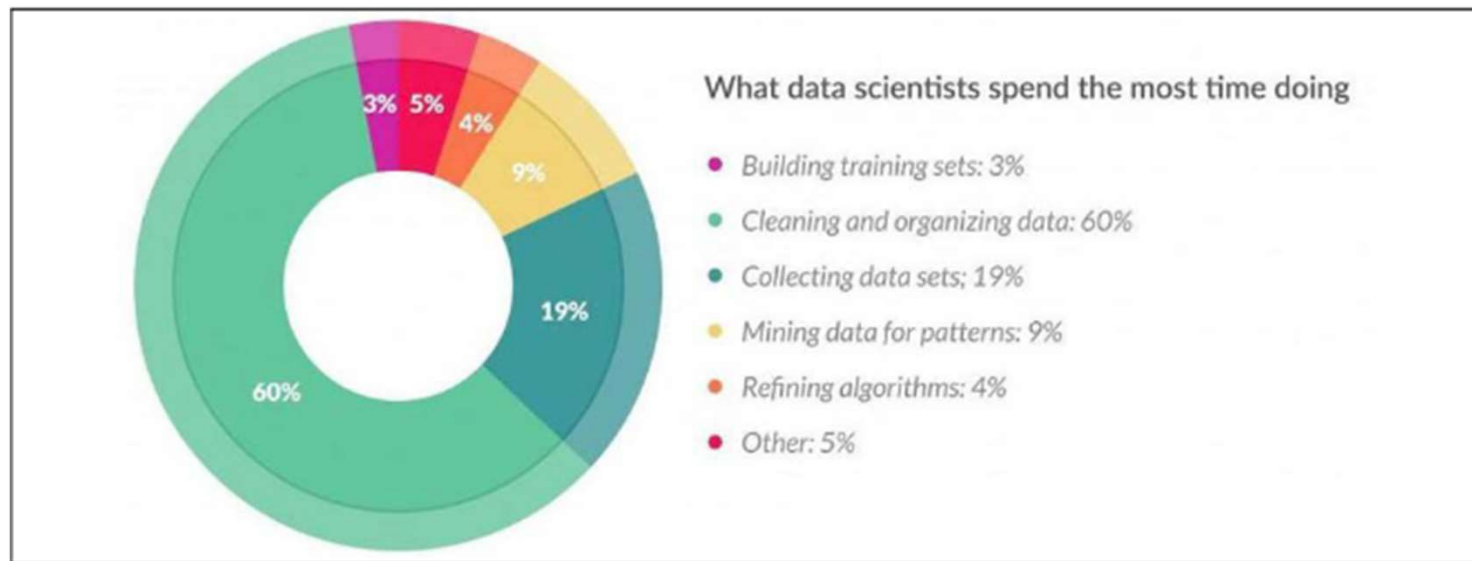


Figure 4: Breakdown of time spent by data scientists (source: Forbes)

# FEATURE ENGINEERING

There are also some generic data science techniques that can be applied in data preparation and feature engineering:

- Imputation
- Outlier management
- One-hot encoding
- Log transform
- Scaling
- Date manipulation

# IMPUTATION

- There might be many reasons why values are missing, such as inconsistent datasets, clerical error, and privacy issue.
- Having missing values can affect the performance of a model, and in some cases, it can bring things to a screeching halt since some algorithms do not take kindly to missing values.
- There are techniques to handle missing values, includes:
  - Removing the row with missing values
  - Numerical imputation
  - Categorical imputation



**THANK YOU**

