Data Prep for Machine Learning in Python

CFI

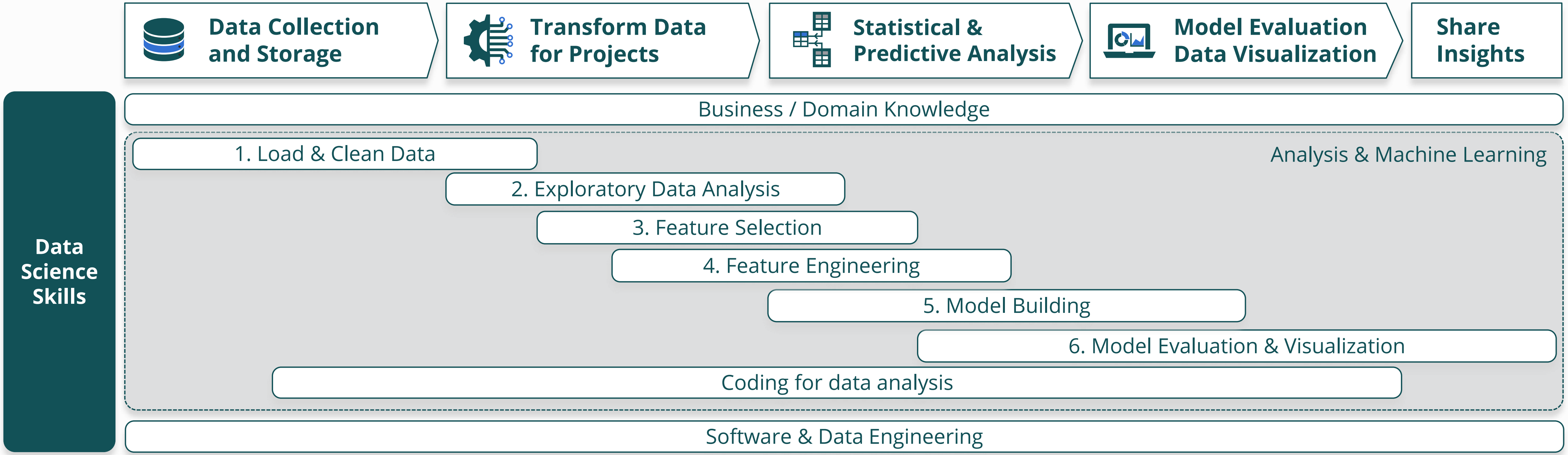
Pre-Requisite Knowledge

Pre-Requisite Knowledge

To get the best out of this course, we recommend that students have completed the following courses before starting.

- Data Science Fundamentals
- Statistics Fundamentals
- Python Fundamentals
- Regression Analysis – Fundamentals & Practical Applications
- Classification – Fundamentals & Practical Applications

Data Science Skills & The Machine Learning Process



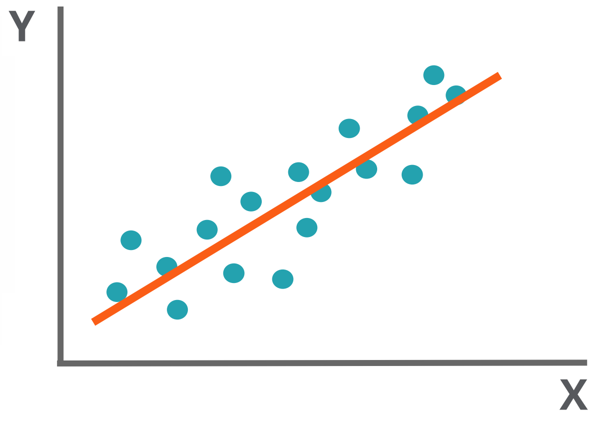
Types of Machine Learning

Supervised Machine Learning

- More common in business to answer **pre-defined questions**.
- **Predict a target variable** based on input data.
- Once model is trained on example data, predictions can be made on new data.
- Ensemble models are **combinations of other models**.

Input Data (Features)			Target Data
Income	Credit Score	Age	Default Loan
$56k	755	43	No
$38k	682	22	Yes
$120,000	731	38	No
$65,000	595	54	Yes
$52,00	784	68	No

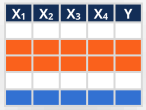
Classification Problems
Which one? What category? True or false?



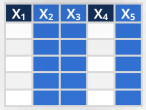
Regression Problems
How much? How many?

Unsupervised Machine Learning

- No specific question in mind
- Point us in the right direction



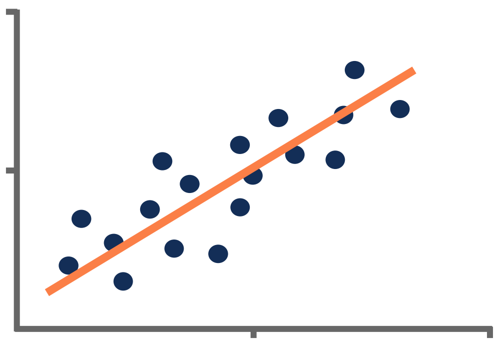
Clustering Problems



Variable Reduction

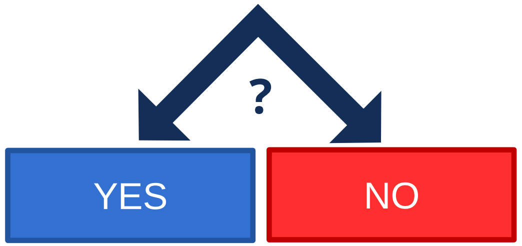
Model Evaluation

Evaluating Regression



```
R-squared:                     0.883
Adj. R-squared:                0.873
F-statistic:                   90.22
Prob (F-statistic):         8.02e-56
Log-Likelihood:              -1332.3
AIC:                           2689.
BIC:                           2724.
```

Evaluating Classification

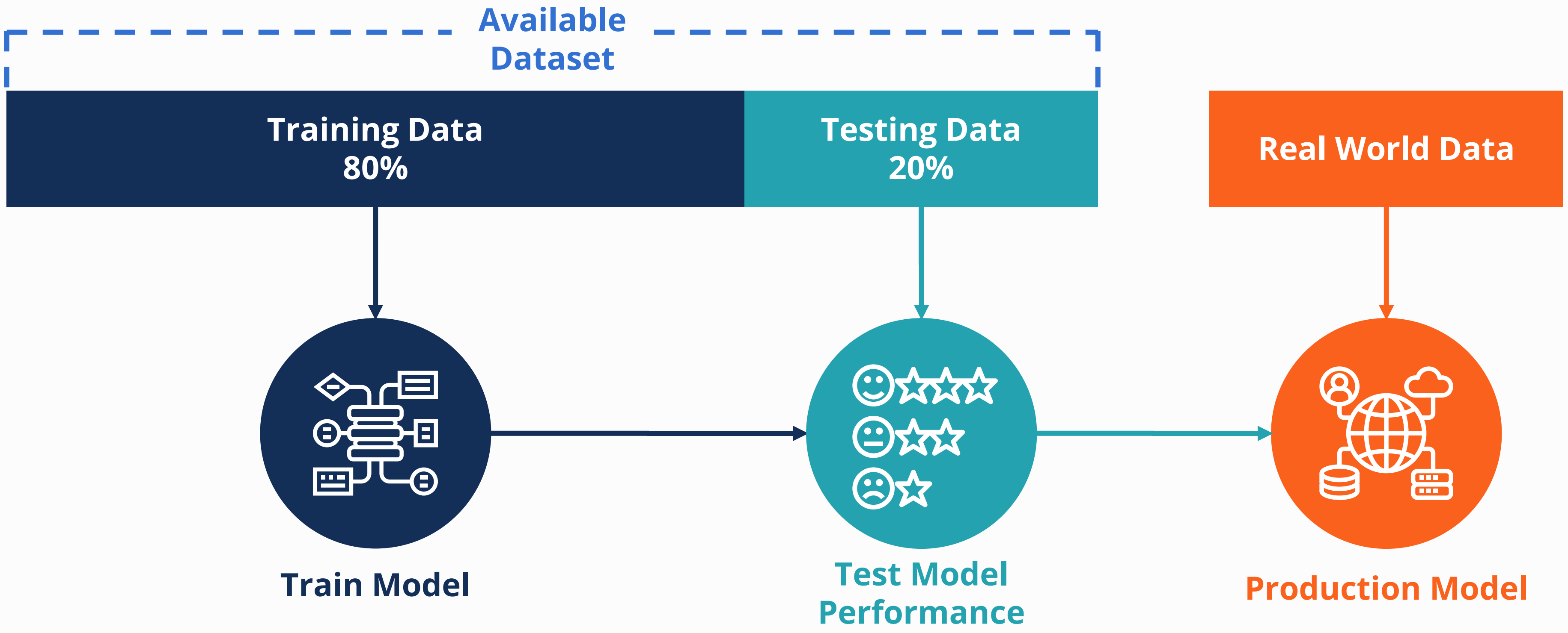


Prediction

		(0)	(1)
Actual	(0)	True Negative	False Positive
	(1)	False Negative	True Positive

Training & Testing

Models need to be tested before we use them to predict real-world outcomes.



Tests must be carried out on new data that the model has **never seen before**.

Basic Dataset Terminology

A few key terms to get you started…

UNIQUE ID | FEATURE | FEATURE | FEATURE | TARGET

Application ID	Age (18-90)	Credit Rating	Income	Credit Approved
1	25	697	25,000	YES
2	15	527	13,000	NO
3	19	658	23,000	YES
4	65	738	49,000	YES
5	72	538	32,000	NO
6	26	243	9,000	NO
7	186	999	25,000	NO

Feature: Used as inputs to calculations in models or machine learning algorithms.

Target: The variable of interest, that we are trying to predict, estimate or model.

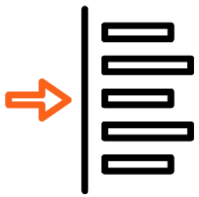
Unique ID: Uniquely identifies each row.

Row: Each row represents a **single observation.**

ROW / OBSERVATION

Descriptive Statistics

Descriptive statistics broadly describe data through single values. These are used to describe the measures of central tendency (how close to the center is data dispersed), measures of dispersion (how far data is dispersed), and the shape of the distribution (how is the data distribution shaped).

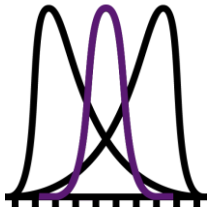


Measures of Central Tendency

- Mean
- Median
- Mode

Measures of Dispersion

- Range
- Variance
- Standard deviation



Shapes of Distribution

- Skewness
- Kurtosis

A Recap of Python Packages

Basic Operations	Data Manipulation	Data Visualization	Machine Learning
Basic Python	Pandas	Seaborn	SKLearn
Functions & Methods Strings & Numbers Lists, Tuples, Sets & Dictionaries Conditional Statements Loops	Dataframes Series	Detailed visuals Advanced statistical plots	Easy to use machine learning packages
	NumPy	Matplotlib	SciPy
	Arrays	Simple + basic plots Multiple plotting	Mathematical algorithms and functions for complex data manipulation for machine learning

Loading & Cleaning Data

Learning Objectives – Loading & Cleaning Data

Learn the basics of loading data from an external source to your Jupyter Notebook

Learn the techniques to filter and slice through your imported data

Validate which parts of your data make sense for analysis

Practice various ways of cleaning data to fit your use case

Identify and fix the errors that hinder your dataset quality

Use imputation to modify data that makes more sense to the context at hand

Data Types

Understanding the **data types** helps us **understand limitations and challenges** we may encounter.

Continuous

Continuous features allow us to **measure amounts** or points along a scale.

Can be measured on a scale or timeline.

1.7

Might include numbers, or a timeline of dates.

Categorical

Categorical features tell us **which bucket** a data point falls into.

Unordered Categorical Features

- Unemployed
- Part Time
- Full Time
- Student

No specified order

Ordinal Categorical Features

1. Small
2. Medium
3. Large
4. X Large

A logical order exists

Binary Features

- 1 True
- 0 False

Two buckets

Common Data Types in DataFrame Columns

We will generally be working with data in a pandas Data frame. Here's a reminder of data types.

Pandas Data Type	Data Type Description	Example
object / category	String or mix of numeric and string Use category when cardinality is low	"a line of text", "20,000 sqft", "category 1", "category 2"
int64	Whole numbers that can be stored exactly	1,3,5, etc.
float64	Numbers that have decimals and cannot be stored exactly	1.2341242….., 3.435436241…, 5.321321….etc.
bool	True or False values	Values that state "True" or "False"
datetime	Values that contain the date and time	2018-11-15, 2020-12-24 00:30:00

Imputation

Imputation allows us to **replace missing or null values with an estimated value**, improving the quality of our data.

Example Data

Occupation	Income ($ 000's)
Engineering	100
Business	70
Research	80
Engineering	110
Business	90
Engineering	**105**
Research	**80**

Column Mean = 90

Average Income Calculated by Occupation

Occupation	Average Income
Engineering	**105**
Business	**80**
Research	**80**

Exploratory Data Analysis

Learning Objectives – Exploratory Data Analysis

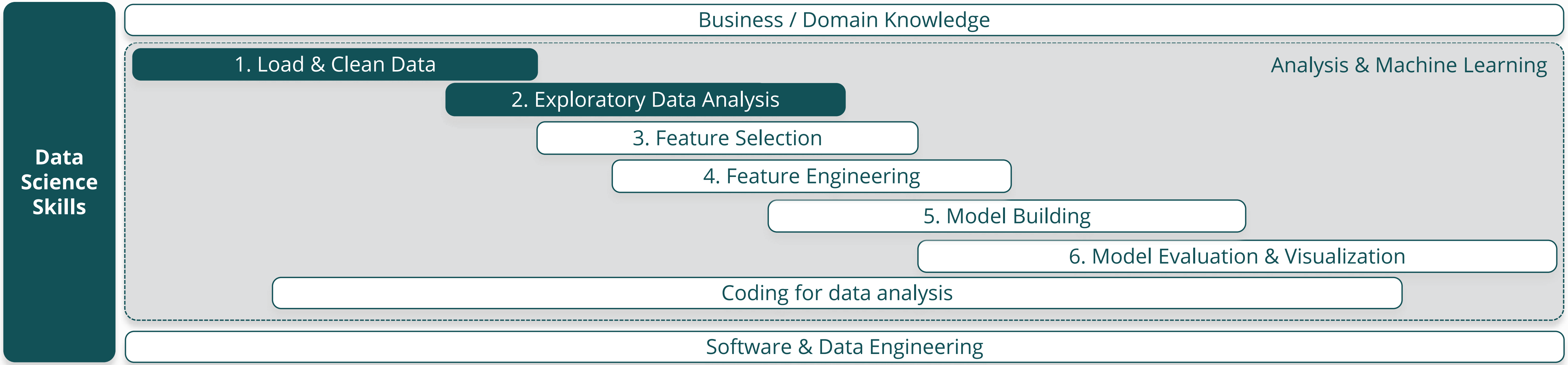
Understand and analyze the statistics of each feature

Learn how to plot basic visuals for single numeric or categorical features

Learn how to analyze relationships between categorical and continuous variables

Learn how to plot basic visuals for multiple numeric or categorical features

Exploratory Data Analysis (EDA)



Exploratory Data Analysis is about analyzing your data to discover trends, patterns and validating your assumptions.

Univariate Analysis

Multivariate Analysis

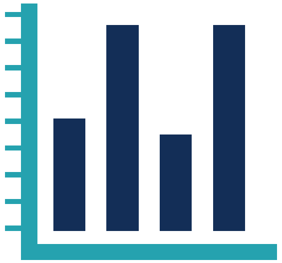
Univariate EDA

Univariate EDA helps us to understand each feature in our dataset. We can analyse things such as the shape, statistics, and trends in our dataset to give further context and insights when building our data science projects.

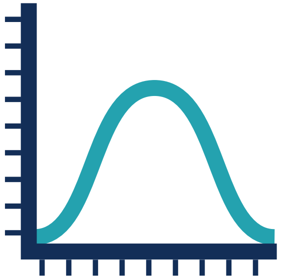
Descriptive Stats

Statistic	Value
Mean	5.334
Std dev.	1.234

Bar Chart / Histogram

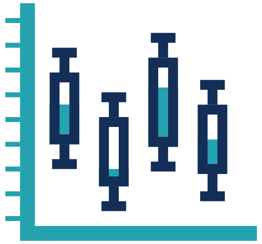


Density Plot



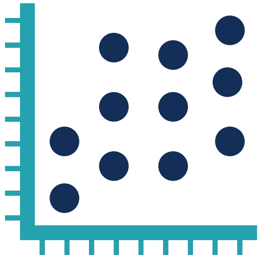
Bar Chart

Box Plot



Pie Chart

Scatter Plot

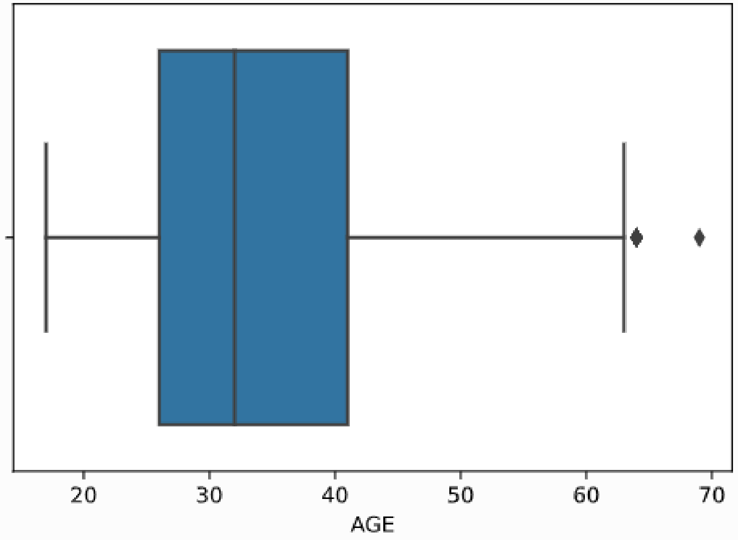


Descriptive Statistics

- Summary statistics for continuous variables are an essential part of exploratory data analysis
- Typically when we talk about summary statistic we are referring to
  - **Maximum -** Largest value
  - **Minimum -** Smallest Value
  - **Range –** Difference between the maximum and the minimum
  - **Mean -** Average Value
  - **Median -** Middle Value
  - **Standard Deviation -** How spread out is the data
  - **Inter Quartile Range (IQR) –** Range of the middle 50% of the data

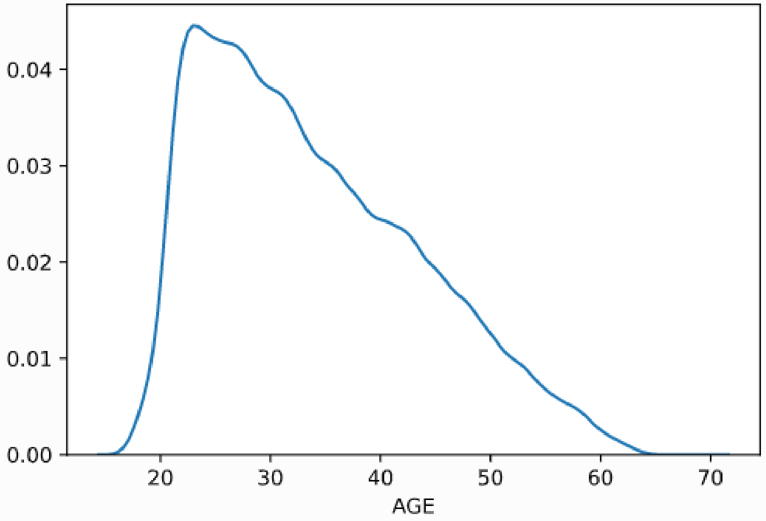
- Summary statistics help us understand our data, identify obvious problems, identify questions we need to answer or focus areas.
  - How old is the average person in our data set?
  - What was the amount of the largest loan?
  - What does it mean if someone has an age of -1?

Boxplots and Distributions



Boxplots

- Provide a visualization of the summary statistics
- Minimum, maximum, Outliers, IQR and Median
- Allow us to confirm assumptions about skewness and extreme values
  - Age has some positive outliers

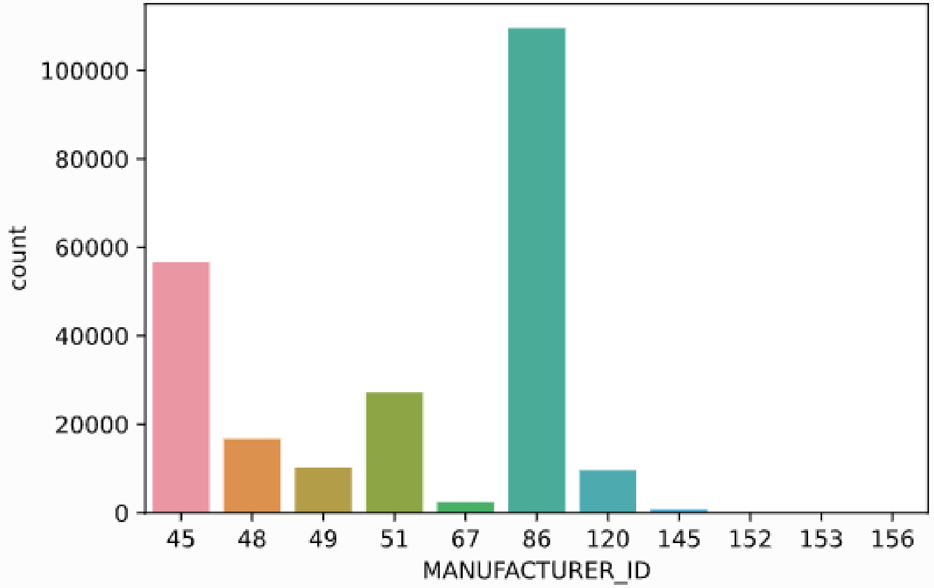


Distributions

- Show us the underlying frequency distribution of a variable
- The shape of the distribution gives us information about the data
  - Is it normally distributed?
  - Is it skewed?
  - How many peaks?

Bar Charts

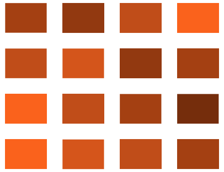
- Very useful for visualizing categorical variables
- Show us the frequency of data for each category
- Essential part of EDA, we need to get an idea of common and uncommon values for our features



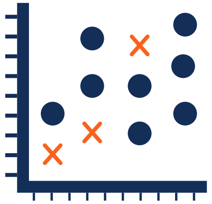
Multivariate EDA

Multiple Variable EDA allows us to **analyse several variables together**.

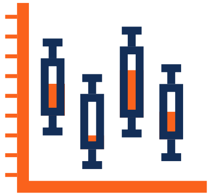
Correlation Matrix



Scatter Plot



Categorical Box Plot

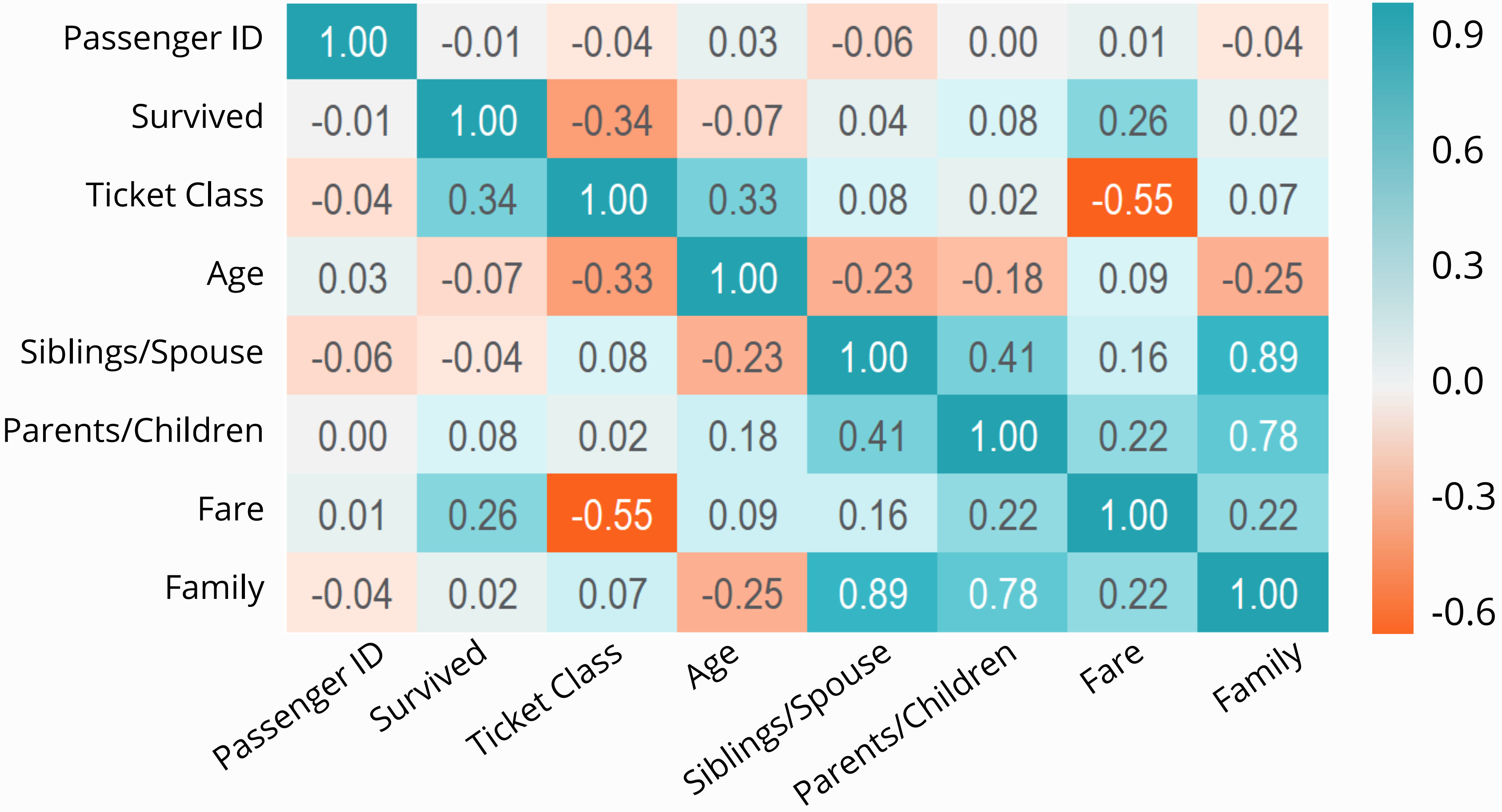


Categorical Bar Chart

We can gain further context and insights from our data by learning from various relationships of numeric and categorical features.

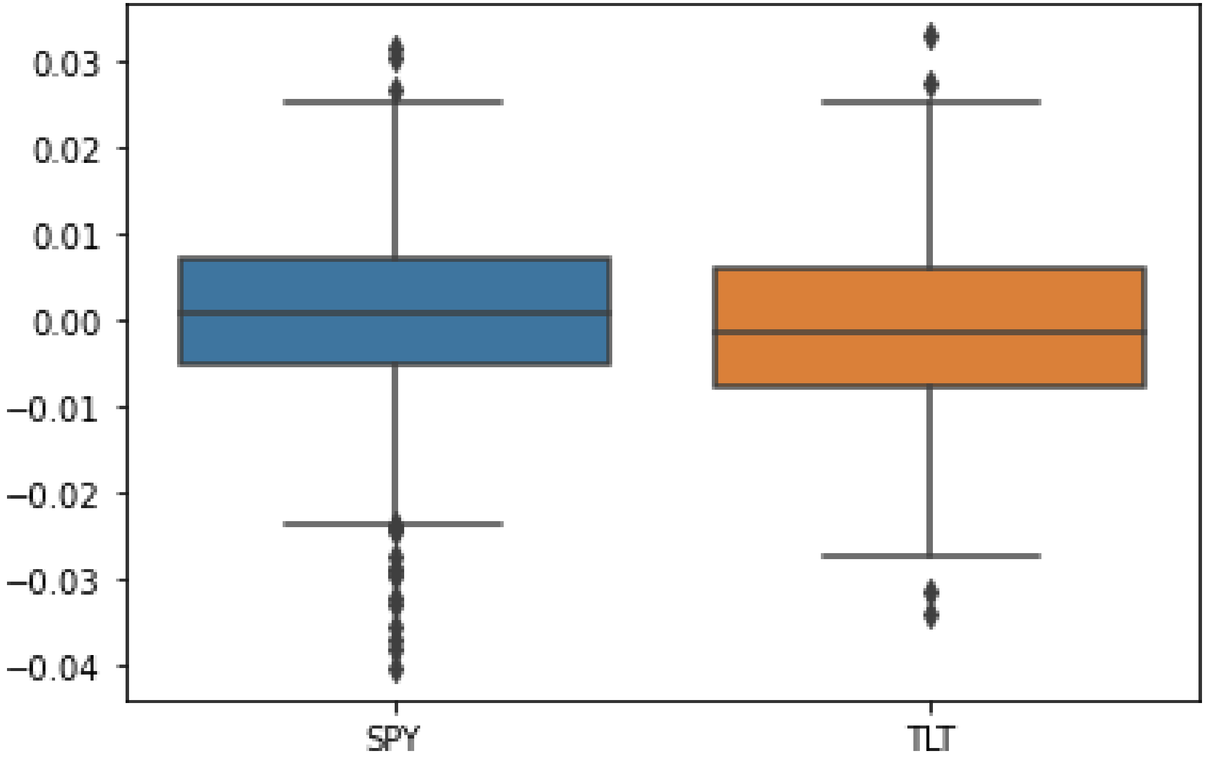
Correlation Matrix Recap

Correlation Matrix helps us see the strength and direction of relationships between features.



Box Plots Across Categories

Box plots by category allows us to compare the distributions of multiple features.



IQR definitions to add here.

Feature Engineering

Learning Objectives – Feature Engineering

Learn how to modify your data to create new variables that can help improve model performance

Learn how to transform categorical variables using One Hot Encoding

Learn how to distinguish and modify outlier data

Apply functions to transform skewed data

Apply binning functions to reduce the amount of noise in our dataset

Apply scaling functions to properly transform our data to a state that is more accurate for our models

Feature Engineering

Data Science Skills

Business / Domain Knowledge

Analysis & Machine Learning

1. Load & Clean Data
2. Exploratory Data Analysis
3. Feature Selection
4. Feature Engineering
5. Model Building
6. Model Evaluation & Visualization

Coding for data analysis

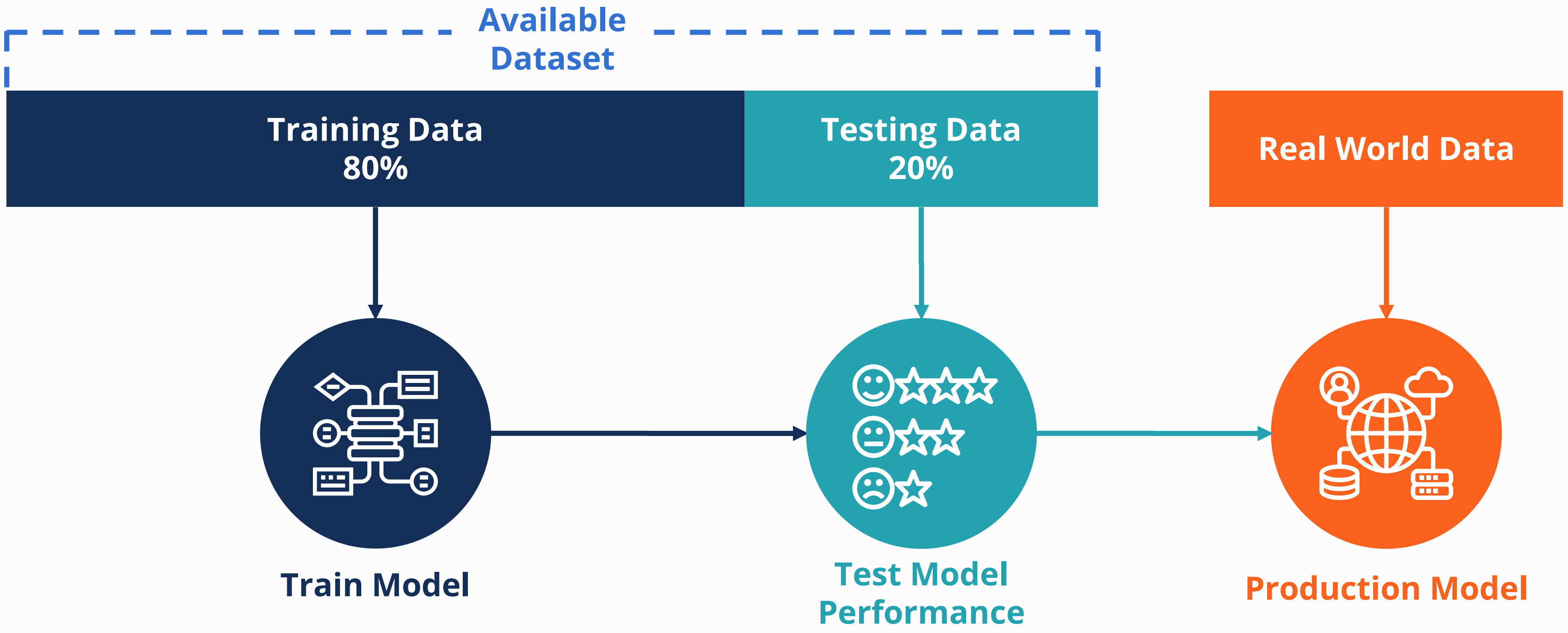
Software & Data Engineering

Feature Engineering is about modifying our data to make it **more optimal for our analysis**.

Encoding | **Binning** | **Outliers** | **Scaling** | **Transforms** | **Other**

Training & Testing

Models need to be tested before we use them to predict real-world outcomes.



Training Vs Testing

Feature Engineering **must be applied on training and testing datasets in the same way**.

Suppose we take the log of all values.

Training

Size	→	LN(Size)
1		0.000000
2		0.693147
3		1.098612
2		0.693147
3		1.098612

Testing

Size	→	LN(Size)
1		0.000000
3		1.098612

This causes no issue, since **every transformation is independent**.

Suppose now we subtract the mean from all values.

Training

Mean = 2.2

Size	→	Size – Mean(Size)
1		-1.2
2		-0.2
3		0.8
2		-0.2
3		0.8

Testing

Size	→	Size – Mean(Size)
1		
3		

For the transformation to be the same, **we rely on data from the training set**.

Training Vs Testing

The processes of learning from the Training data is called Fitting.

The process of applying the changes is known as Transforming.

1. Load & Clean Data

2. Exploratory Data Analysis

Dataset Split into Training & Testing

3. **Fit** Feature Engineering on Training Data

3. **Transform** Training Data

3. **Transform** Testing Data

4. **Fit** Model to Training Data

4. **Predict** on Testing Data

5. Model Evaluation

Fitting & Transforming in SKLearn

Looking at the documentation for SKLearn's OneHotEncoder for example shows us the available methods.

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

Methods

`fit`(X[, y])	Fit OneHotEncoder to X.
`fit_transform`(X[, y])	Fit OneHotEncoder to X, then transform X.
`get_feature_names`([input_features])	DEPRECATED: get_feature_names is deprecated in 1.0 and will be removed in 1.2.
`get_feature_names_out`([input_features])	Get output feature names for transformation.
`get_params`([deep])	Get parameters for this estimator.
`inverse_transform`(X)	Convert the data back to the original representation.
`set_params`(**params)	Set the parameters of this estimator.
`transform`(X)	Transform X using one-hot encoding.

The **fit()** method fits the OneHotEncoder method to a set of data.

The **transform()** method modifies the data according to the fitted transformation.

The **fit_transform()** method does both at the same time, typically for use on the training data set.

ML Algorithms – Dataset Considerations

Feature engineering makes use of several techniques such **as encoding, binning, outliers, scaling** etc.

A common question is **when exactly should each of these techniques be applied**.

There is no golden rule, however, we can **learn some characteristics of model types** that help us understand whether a technique might be appropriate.

Ultimately, our goal is always the same, **to improve model performance according to some evaluation metric**, so that should be the primary factor in determining what techniques we use.

ML Algorithms – Dataset Considerations Pt 1

This guide represents the general, simple implementation of each model. It should be used as a starting point only.

	Encoding	Binning	Outliers	Scaling	Feature distribution
Linear Regression	O.h.e works if: - Regularized model is used (common) - OR Fit intercept = false - OR one dummy is dropped	Loss of information leads to reduction in noise. May reduce overfitting and improve model performance. Caution.	Sensitive	Helps interpretation. May improve performance.	No specific assumptions. Normality assumption is for errors!
Logistic Regression	O.h.e increases dimensionality, which may reduce model performance		Highly sensitive	Essential	No specific assumptions
KNN	O.h.e increases dimensionality, which may reduce model performance		Can be sensitive with low value of k	Essential	Modifying the distribution may improve performance
SVM	Most encoding techniques will work well		Can be highly sensitive at the margin	Essential	Modifying the distribution may improve performance
Naïve Bayes	Prefer label/ordinal encoding to avoid multi-colinearity		Highly sensitive	Not essential	No specific assumptions
Gaussian Naïve Bayes	Prefer label/ordinal encoding to avoid multi-colinearity		Highly sensitive	Not essential	Normality is assumed for continuous features
Decision Trees / Random Forest	O.h.e typically produces small gains. Label/ordinal encoding may work better.	Models dynamically bin. Manual binning generally worsens performance.	Not sensitive	Not essential	No specific assumptions

ML Algorithms – Dataset Considerations Pt 2

This guide represents the general, simple implementation of each model. It should be used as a starting point only.

	Performance with limited data points (observations)	Performance with high dimensionality (many features)	Explainability	Prone to overfitting?	Assumptions
Linear Regression	Good	Prone to overfit	Great	In high dimensions	1) Homoscedasticity 2) No multicollinearity 3) No Autocorrelation 4) Zero Mean Errors 5) Endogeneity 6) Linear Relationship
Logistic Regression	Great	Prone to overfit	Great	No	Similar to linear regression: Note: Independent variables should be linearly related to the log odds
KNN	Poor	Poor	Good	No	Similar things share similar characteristics
SVM	Poor	Good	Good	Yes	Data points are separable by a boundary
Naïve Bayes	Great	Poor	Ok for technical audiences	No	Feature Independence
Gaussian Naïve Bayes	Poor	Poor	Ok for technical audiences	No	Normally distributed features may boost performance
Decision Trees / Random Forest	Good	Good	Great	Decision Trees Yes	No significant assumptions

One Hot Encoding

Many machine learning models **cannot interpret categorical features**.

One hot encoding uses **dummy variables** to transform a categorical variable into numerical ones.

Row ID	State ID
1	AB
2	NY
3	NJ
4	WS
5	OK
6	OH
7	HA
8	NX

→

Row ID	AB	NY	NJ	WS	OK	OH	HA	NX
1	1	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0
4	0	0	0	1	0	0	0	0
5	0	0	0	0	1	0	0	0
6	0	0	0	0	0	1	0	0
7	0	0	0	0	0	0	1	0
8	0	0	0	0	0	0	0	1

Note: In some scenarios we may remove one dummy variable.

SKLearn Documentation - OneHotEncoder

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

Ordinal Encoding

Instead of creating multiple columns, we could also represent an ordered categorical column with a single column of numbers.

Row ID	Size
1	Small
2	Medium
3	Large
4	Medium
5	Medium
6	Medium
7	Medium
8	Large

→

Size
1
2
3
2
2
2
2
3

- Each **category is assigned a number**, in an order that makes sense.
- One benefit is that we **create less features**, which may help some models.
- We should be careful since now a **model may interpret the sizes literally**, for example that Large is exactly 3 times bigger than small.

SKLearn Documentation - OrdinalEncoder

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OrdinalEncoder.html#sklearn.preprocessing.OrdinalEncoder

Binary Encoding

Binary encoding turns our **ordinal category values into binary numbers.**

The **digits of the binary numbers then become features of their own.**

Size	Size	Size	S1	S2
Small	1	01	0	1
Medium	2	10	1	0
Large	3	11	1	1
Medium	2	10	1	0
Medium	2	10	1	0
Medium	2	10	1	0
Medium	2	10	1	0
Large	3	11	1	1

Frequency / Count Encoding

Count encoding tells us how many times each category appears in our data.

Frequency encoding turns this information in a proportion of the total observations.

Again, **encoding turns this information into numbers**.

Size		Size	Size
Small		1	0.125
Medium		5	0.625
Large		2	0.250
Medium		5	0.625
Medium	→	5	0.625
Medium		5	0.625
Medium		5	0.625
Large		2	0.250

You might also like to explore **target mean encoding** and **hash encoding.**

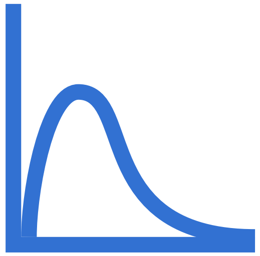
Transforming Skewed Data

Many models and statistical techniques **perform better with normally distributed data**.

- In skewed distributions with long tails, **tails can be interpreted as outliers** by some models.
- Removing skew and long tails can help us **implement linear methods more easily**.
- Anova, t-test, f-test confidence intervals all either rely on normally distributed data, or are easier to interpret with normally distributed data.

Skewed data is not a problem for all models, but as a general rule, normally distributed data will be easier to interpret and produce better results.

Transformation of variables should be done on a situational basis to solve a **model fit or interpretation problem**.



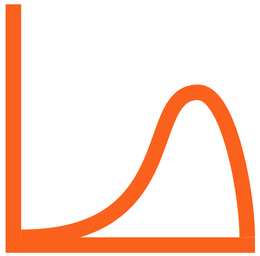
Right Skewed Distribution

Methods to
Remove Right Skew

Sqrt

Logs

Box Cox Transform



Left Skewed Distribution

Methods to
Remove Left Skew

Squares / Cubes / Powers

Log Scales

Some variables may exist on a scale with vastly different units. A good example is worldwide income, whereby some individuals may have an annual income of $25, where the richest have an income of $2,500,000,000.

Number (X)	Log (Base 10) of X
10	1
100	2
1,000	3
10,000	4
100,000	5
1,000,000	6

Number (X)	Natural Log of X
2.72	1
7.39	2
20.09	3
54.60	4
148.41	5
403.43	6

Logs help us squash significantly different values of a single variable onto a more consistent order of magnitude.

The natural log scale is particularly helpful for interpretation.

Natural Log Interpretation

Natural logs have a few useful properties for interpretation.

Which Variable is Logged	Equation	Interpretation
No logs	**y = m x + c**	**1 unit increase in X** leads to **m unit increase** in Y.
Target and input are logged	**ln(y) = m ln(x) + c**	**1 % increase in X** leads to **m % increase** in Y.
Input only is logged	**y = m ln(x) + c**	**1 % increase in X** leads to **m/100 increase** in Y
Target only is logged	**ln(y) = m x + c**	**Unit increase in X** leads to a **exp(m)-1 * 100 increase** in Y.

Binning Numeric Values

Numbers recorded to a high degree of accuracy can sometimes lead to overfitting, since the model may focus too much on the noise.

Binning is used to reduce the noise in data, which may help our model focus on the general trend. Each group represents a **range of similar values**.

It is **best used to encode additional domain knowledge** into the data.

We can even **re-incode numeric values** to each group, **maintaining their ordered** characteristics.

We **should use caution when using binning**, since we are **destroying information**.

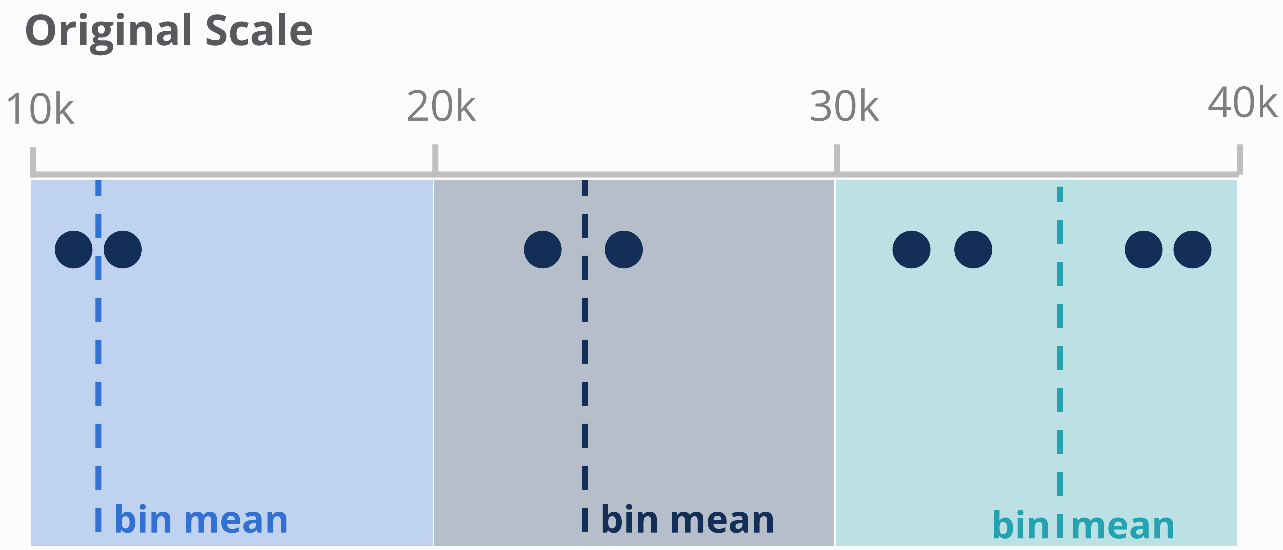
Pandas Documentation – cut and qcut (used for binning)
https://pandas.pydata.org/docs/reference/api/pandas.cut.html
https://pandas.pydata.org/docs/reference/api/pandas.qcut.html

Income	Income Tax Group	Income Class
35,650	30-40k	3
36,230	30-40k	3
84,570	80-90k	8
45,328	40-50k	4
20,303	20-30k	2
150,320	100k+	10
26,330	20-30k	2
62,320	60-70k	6
48,321	40-50k	4
72,320	70-80k	7

Bin Smoothing

Bin Smoothing allows us to keep the numerical nature of the variable, whilst losing some of the detail behind it.

First we create bins, and then we **replace original values with the mean or median of the bin**. (other variations exist)



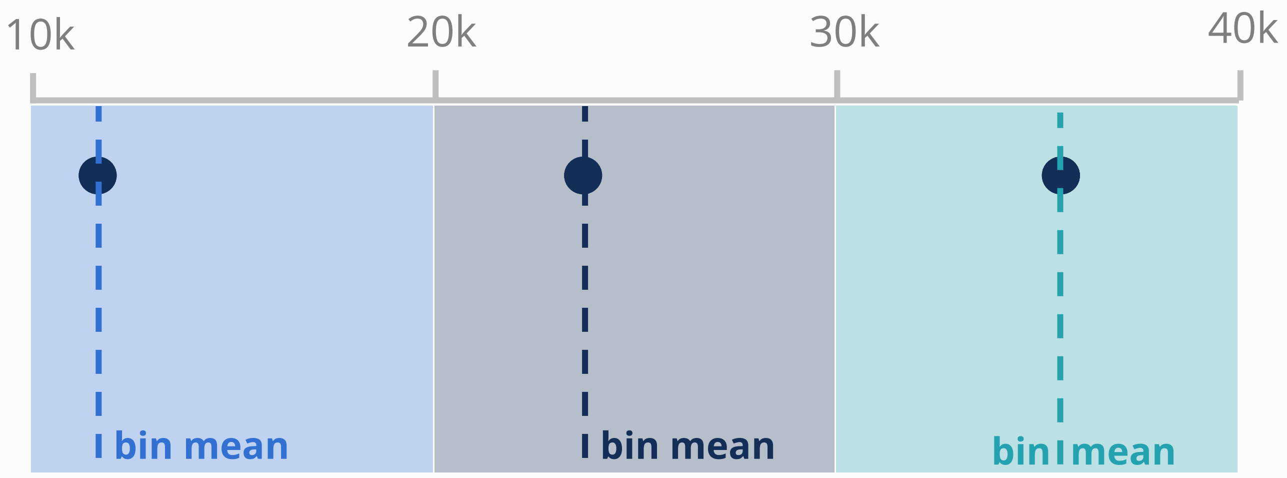
*Domain knowledge informs our bin boundaries and bins are created to represent tax brackets, which we believe impacts spending behavior.

Bin Smoothing

Bin Smoothing allows us to keep the numerical nature of the variable, whilst losing some of the detail behind it.

First we create bins, and then we **replace original values with the mean or median of the bin**. (other variations exist)

Original Scale



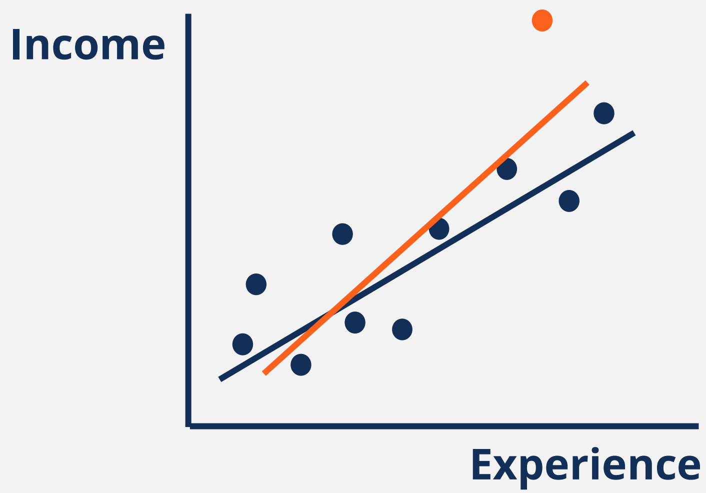
*Domain knowledge informs our bin boundaries and bins are created to represent tax brackets, which we believe impacts spending behavior.

Income	Smoothed
11,190	11,705
12,220	11,705
23,260	24,465
25,670	24,465
32,940	36,394
34,230	36,394
38,574	36,394
39,830	36,394

Feature Engineering – Dealing with Outliers

Outliers can affect the results of our analysis significantly, so we should be aware of their presence and potential influence.

An outlier or just a high value?



Potential Questions

- Is there **something we're not capturing** that might be generating these results, or is this truly a random outlier?
- **Beyond what threshold** do we consider an observation to be an outlier?
- **What impact will outliers have** on the particular model or scaling method we are using?
- Should I change my **analysis method** based on the known presence of outliers?

We should only remove or adjust outliers in our analysis if we have a very good reason.

Feature Engineering – Min Max Scaling

Normalization (min-max scaling) rescales the data to values between 0 and 1.

Income	Credit Score	Age
$56,000	755	43
$38,000	682	22
$120,000	731	38
$65,000	595	54
$52,00	784	68

Income	Credit Score	Age
0.2195	1.0000	0.4565
0.0000	0.5438	0.0000
1.0000	0.8500	0.3478
0.3293	0.0000	0.6957
0.1707	0.9563	1.0000

Features of **significantly different scale** can cause problems for our Machine Learning models. Scaling can be used to solve this problem.

SKLearn Documentation – MinMaxScaler
https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

Feature Engineering – Min Max Scaling (Normalization)

Min max scaling **does not change the shape** of the distribution of values.

The **min max boundaries are learned from the training** dataset.

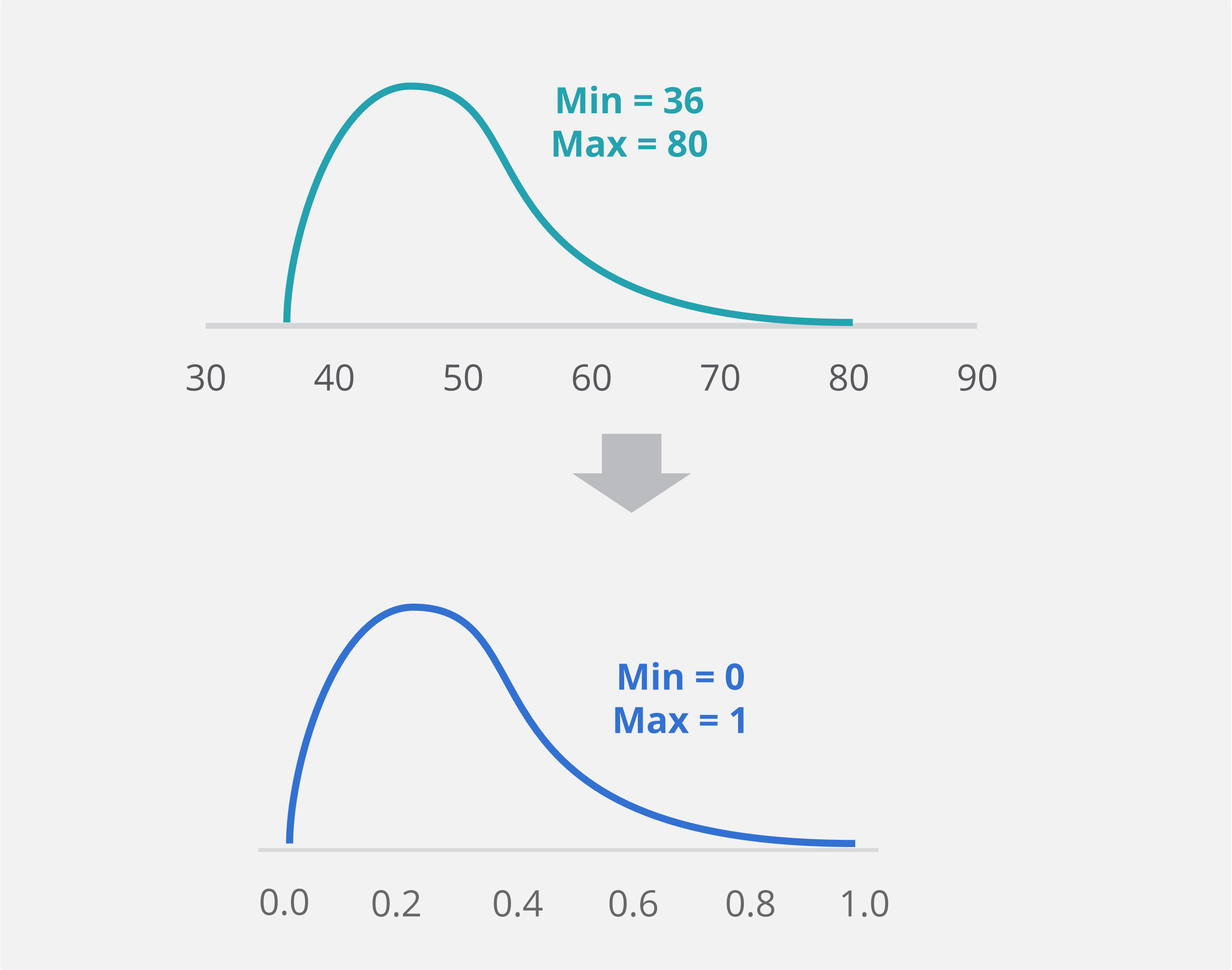
The variable range of 36 to 80 is **fit into a boundary of 0 to 1**.

Any value can be converted onto the new scale using the following formula:

$$\text{Scaled Value} = \frac{Value - Min}{Max - Min}$$

If a new value of 90 is observed in the testing dataset, it's scaled value will be greater than 1.

$$\frac{90 - 36}{80 - 36} = \text{Scaled Value} = 1.23$$



Feature Engineering – Standardization

Standardization performs a similar role to normalization, rescaling values to a **standardized, comparable scale**.

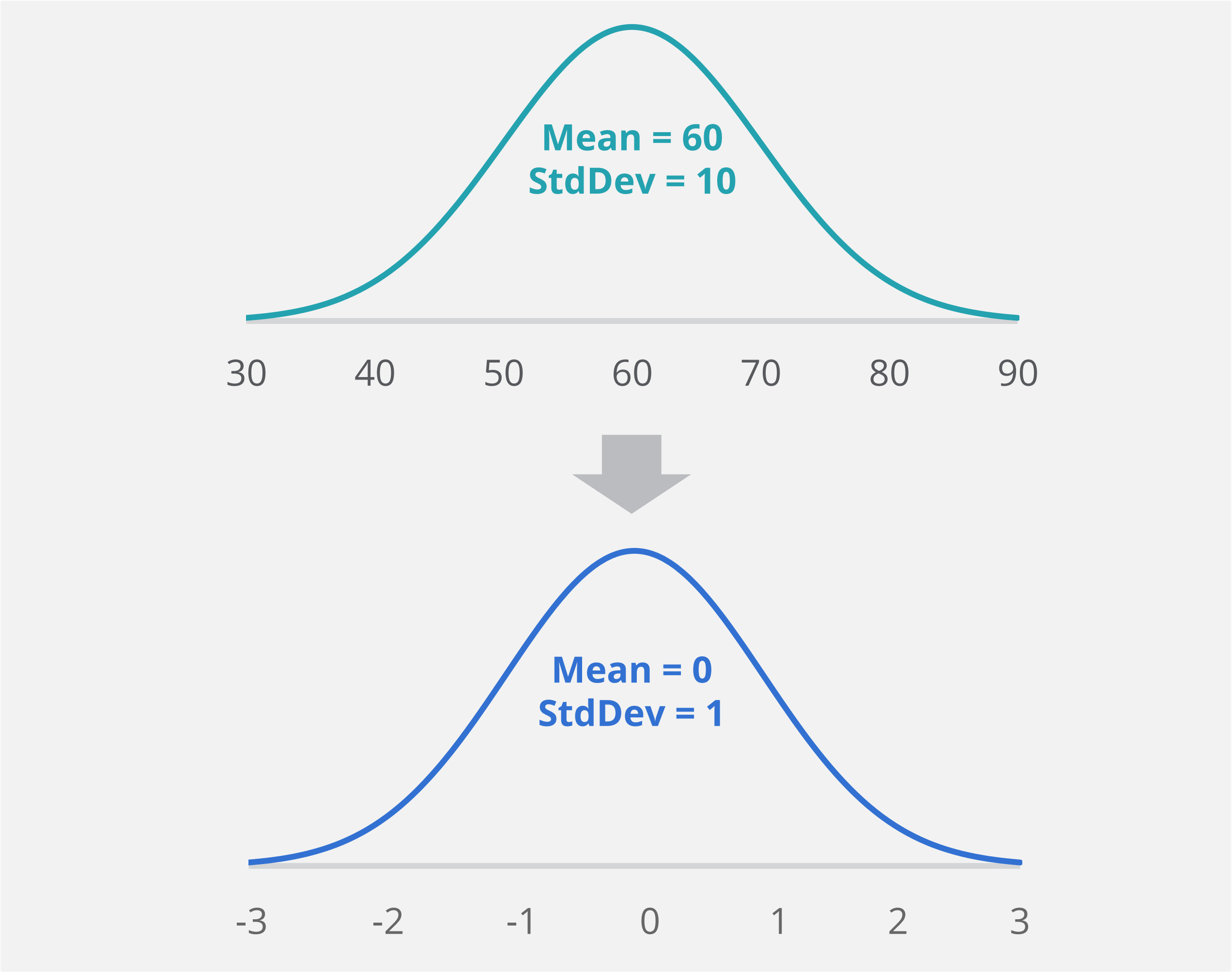
Typically used for **features with a gaussian distribution.**

Rescales a normal distribution so that it has a mean of 0 and standard deviation of 1.

No bounds on the values of the feature.

Values can be converted using this formula:

$$\text{Scaled Value} = \frac{Value - Mean}{StdDev}$$



SKLearn Documentation – StandardScaler
https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

Feature Engineering – RobustScaler

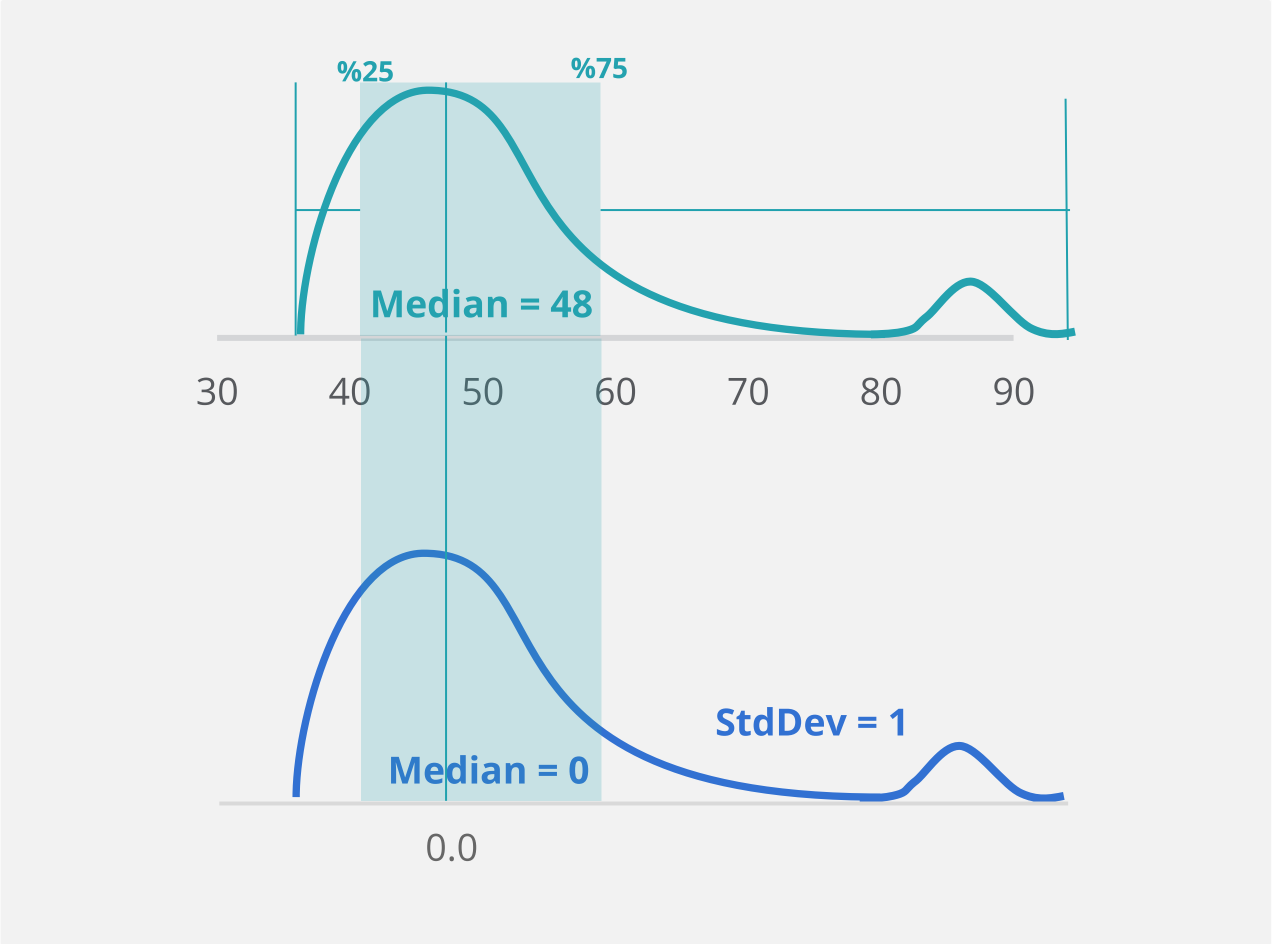
Robust Scaler tries to **overcome some pitfalls** of other methods, **in particular dealing with outliers**.

Outliers can easily impact min max scaling, and can **easily impact standard deviation** used in standardization.

Robust Scaler **focuses on scaling the interquartile range**.

Any value can be converted onto the new scale using the following formula:

$$\text{Scaled Value} = \frac{Value - Median}{\%75th - \%25th}$$



SKLearn Documentation – RobustScaler
https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html

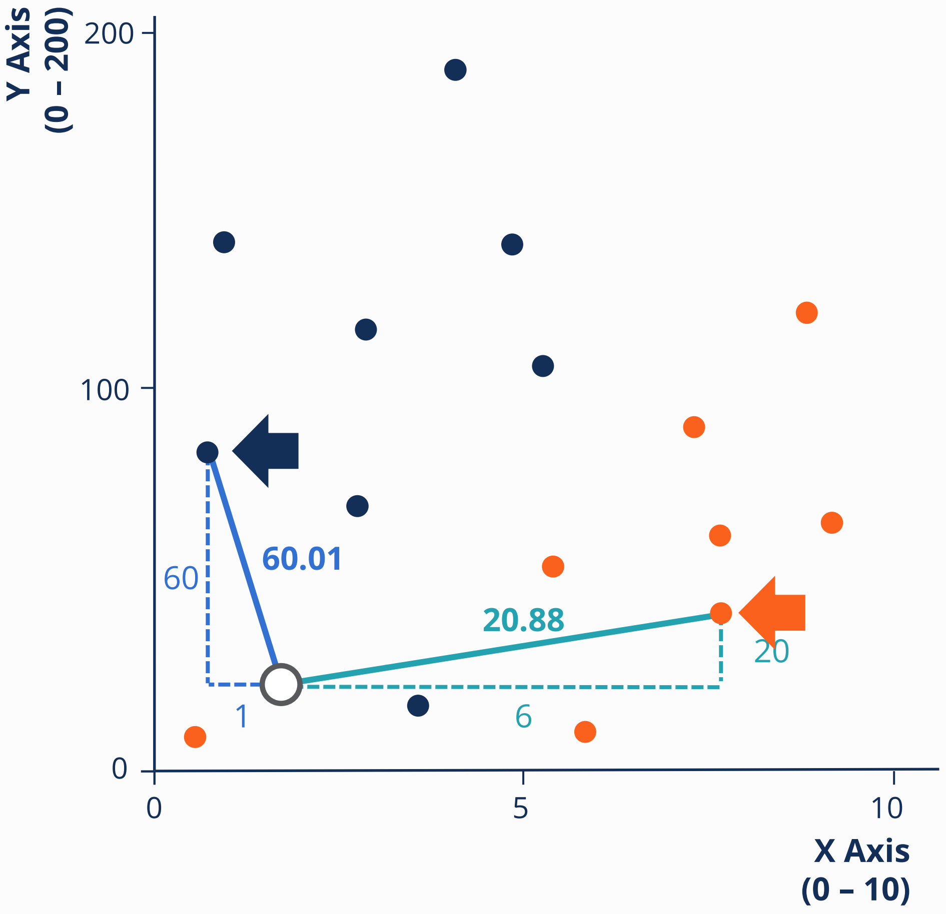
Types of Scalers

	Scaler	Definition	Use case	Syntax
SKLearn	MinMax Scaler	Scales the values so that they equate to a value between 0-1	Best used for continuous data with no outliers	MinMaxScaler()
SKLearn	Standard Scaler	Scales the values so that the mean is 0 and the standard deviation equates 1	Best used for normally distributed datasets	StandardScaler()
SKLearn	Robust Scaler	Scales the values by removing the median from the data and scaling based on the Inter Quartile Range	Best used when there are many outliers in the data	RobustScaler()
SKLearn	MaxAbs Scaler	Scales the values by taking the absolute maximum value of each column and divide each column by the absolute maximum value	Best used for continuous data with no outliers	MaxAbsScaler()

We should apply the **same scaling method to all numerical columns**.

Why Models Get Confused With Scale

Distance based models tend to unfairly weight their consideration of features with bigger values. But why?



X Our X feature takes values between 0 and 10.

Y The Y feature takes values between 0 and 200.

New data point

Which of these points is closest to the new data point?

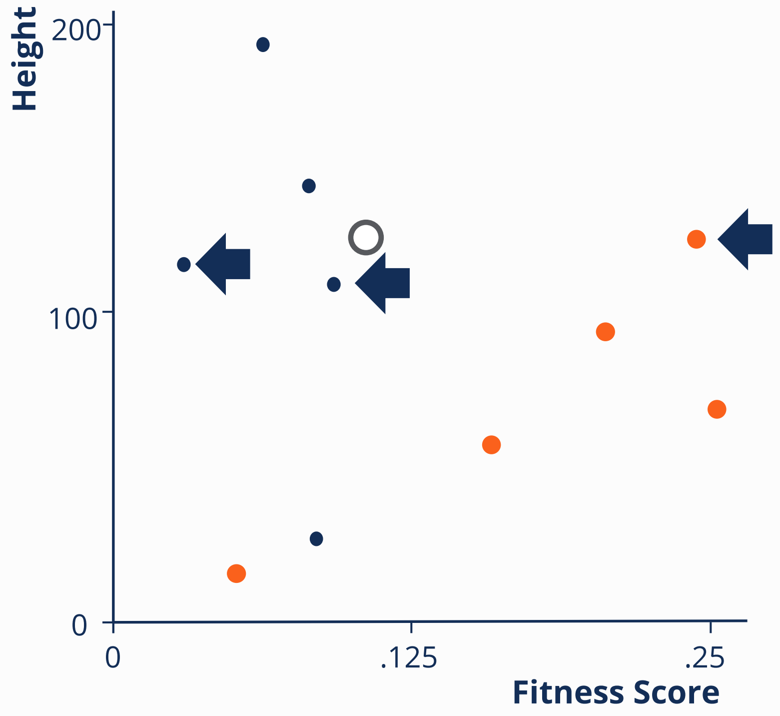
Using Pythagoras, we can see that the distances are not fairly represented.

*This is also referred to as Eucledean distance, though other methods of distance calculation do exist.

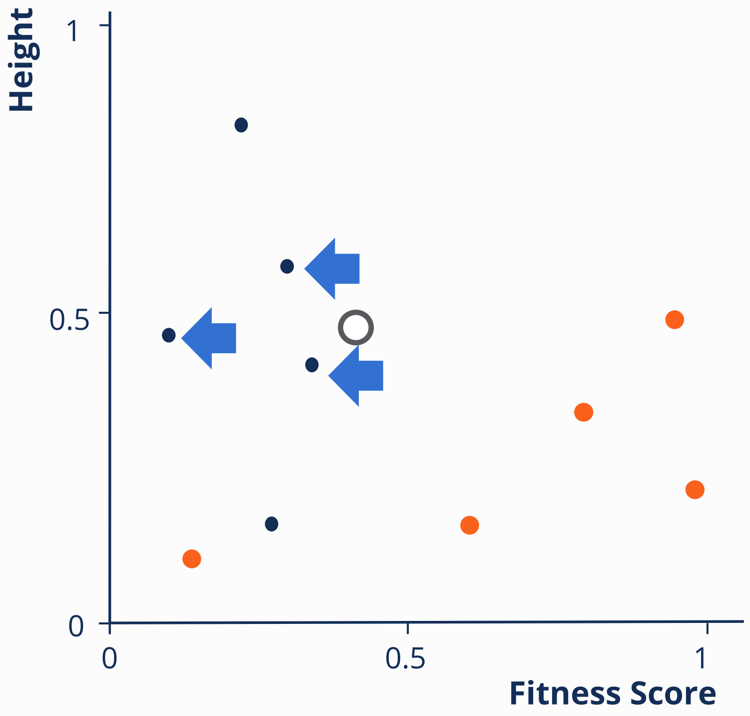
Impact of Scaling

When the scales of our input features are different, **it can cause problems for distance-based algorithms**.

This **KNN model incorrectly identifies the 3 nearest neighbors** as a result of favoring the height variable.



With **min max scaling applied**, the **model now correctly identifies the 3 nearest neighbours**.



Learning Objectives – Feature Selection

Use domain knowledge to choose the best features in your dataset

Learn how to use correlation coefficients to choose continuous features for continuous target variables

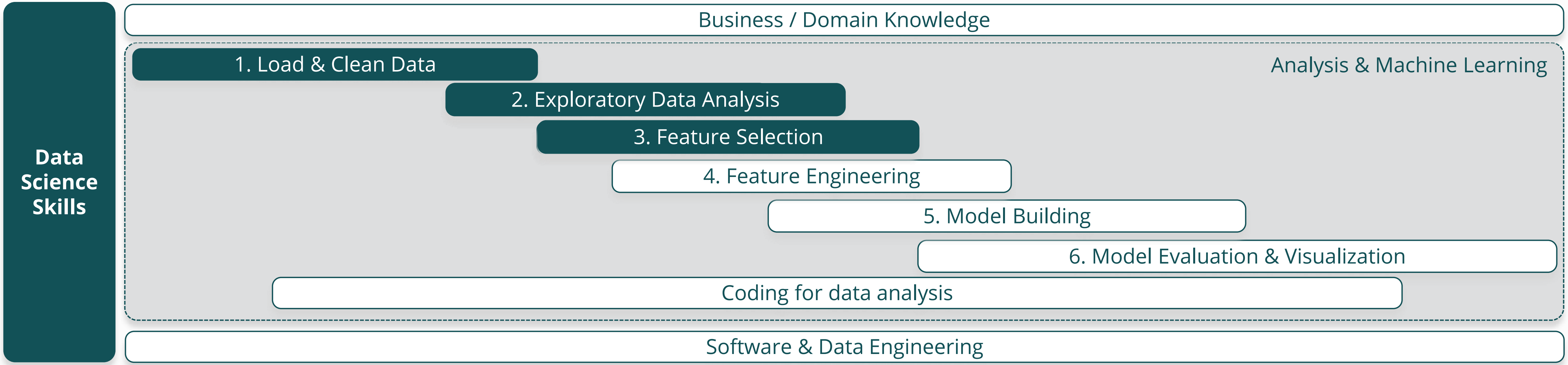
Learn how to conduct an ANOVA test to choose categorical features for continuous target variables

Learn how to conduct a Chi-Square Test to choose categorical features for categorical target variables

Learn how to use box plots to choose continuous features for categorical target variables

Feature Selection

Feature Selection



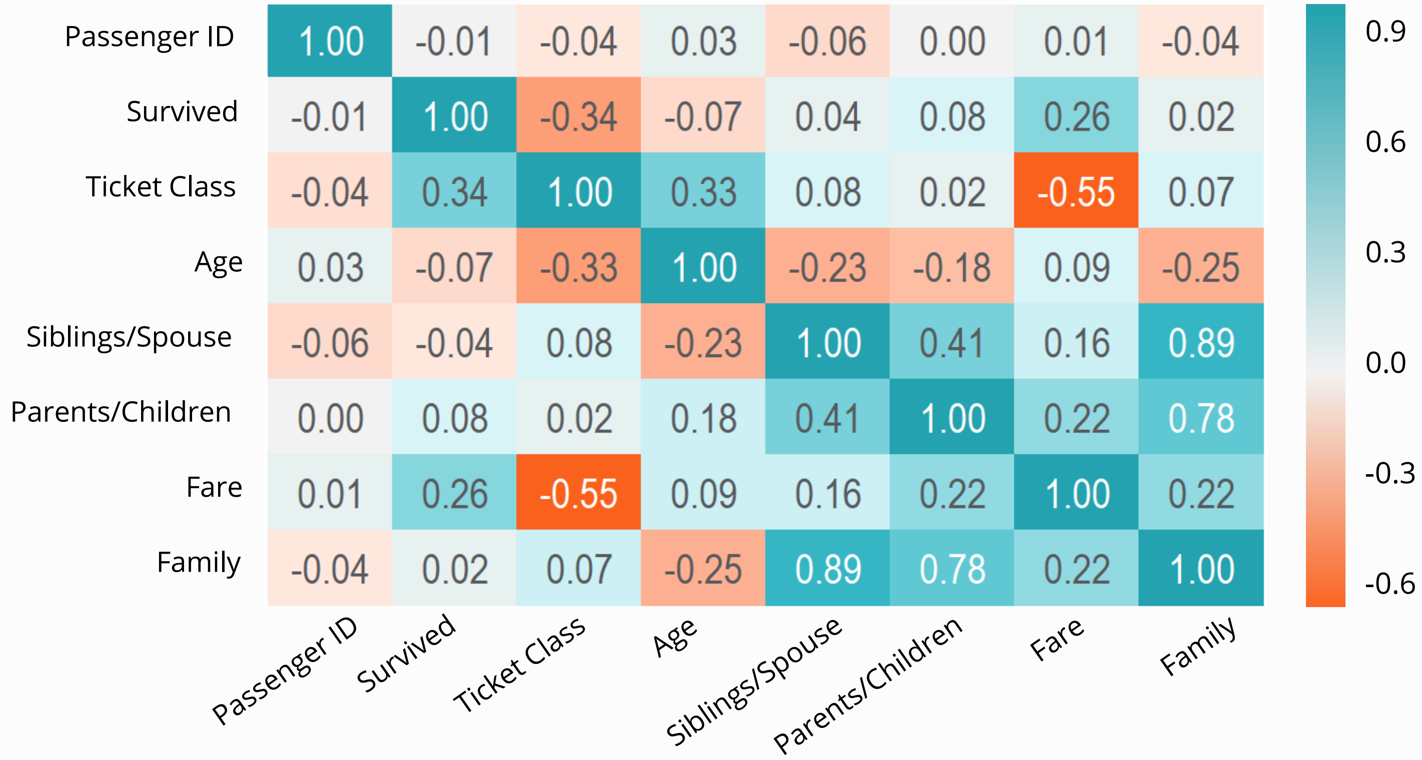
Feature Selection is about picking out the most relevant features in our dataset to use for our machine learning models.

We can **manually decide** which features to keep, or use some basic statistical methods referred to as **FILTER methods**:

- Correlation Coefficients
- ANOVA
- Chi Square
- Category Box Plots *(not a filter method)*

Correlation

With correlation coefficients, we can spot features that have a linear relationship with our target variable.

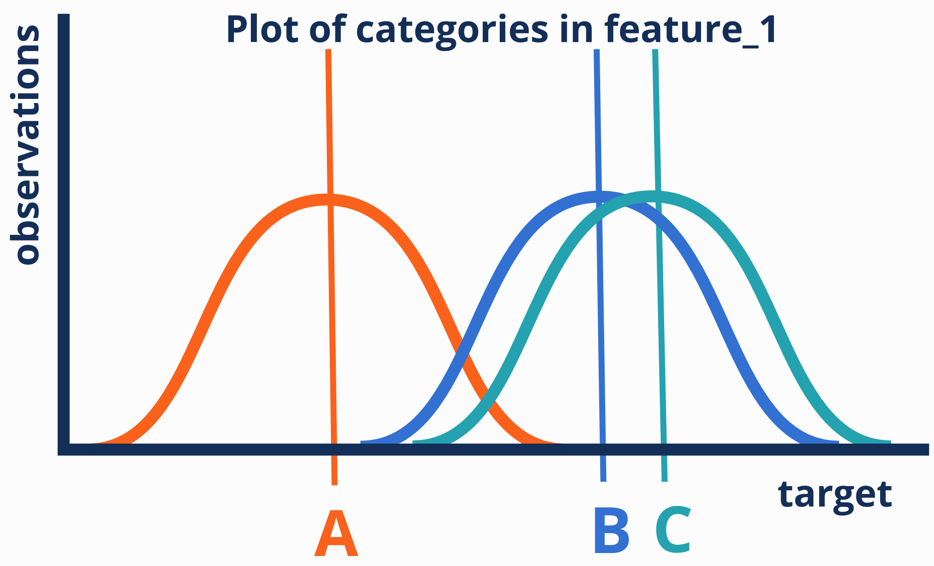


1) Finding features with high correlation to the target makes it easier to spot the features that are most relevant.
2) Additionally, we can use the correlation matrix to identify multicollinearity, that may help our decision making.

ANOVA

The one way ANOVA test helps us measure the likely predictive power of a categorical feature on a continuous target variable.

The ANOVA measures **to what extent one or more categories produces a different value for the target variable**.



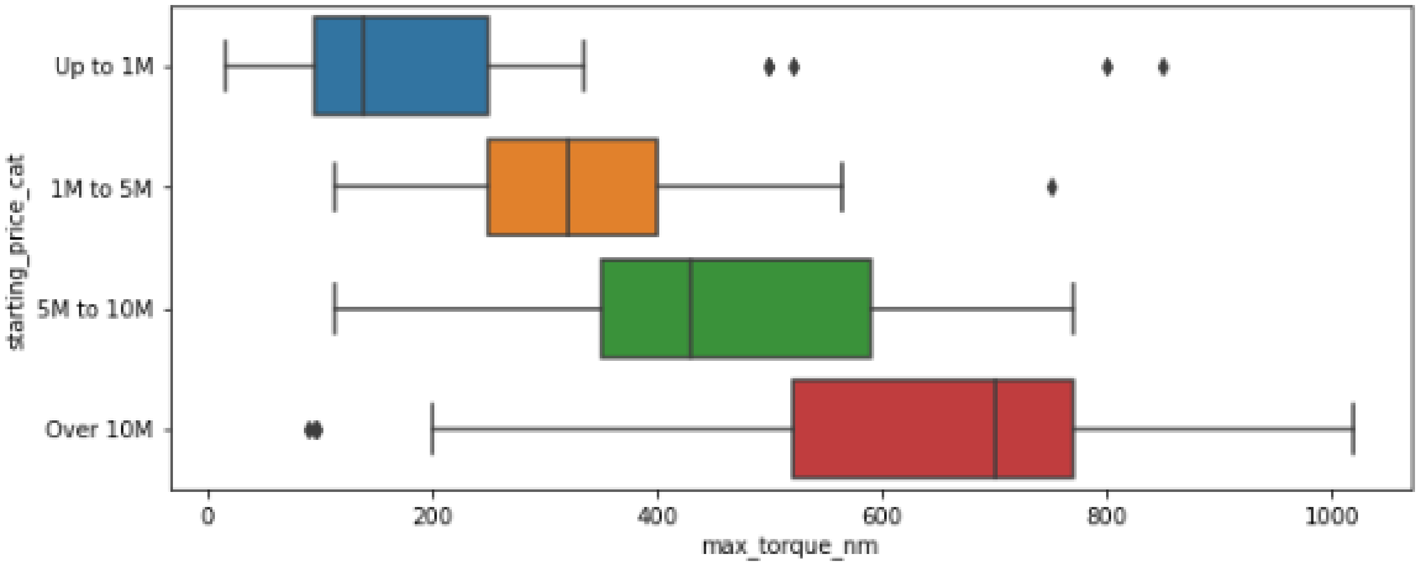
- The ANOVA test **first calculates the mean of the continuous target variable for each category**.
- It then performs an **f-test** to calculate **the extent to which atleast one of these means is different** from the others.
- Finally, **a p-value tells us if this observation is statistically significant**. If so, we can say that this feature has some predictive power.

Anova Assumptions

The ANOVA test generally assumes equal target variance in each group, and normality of residuals.

Per Category Box Plots

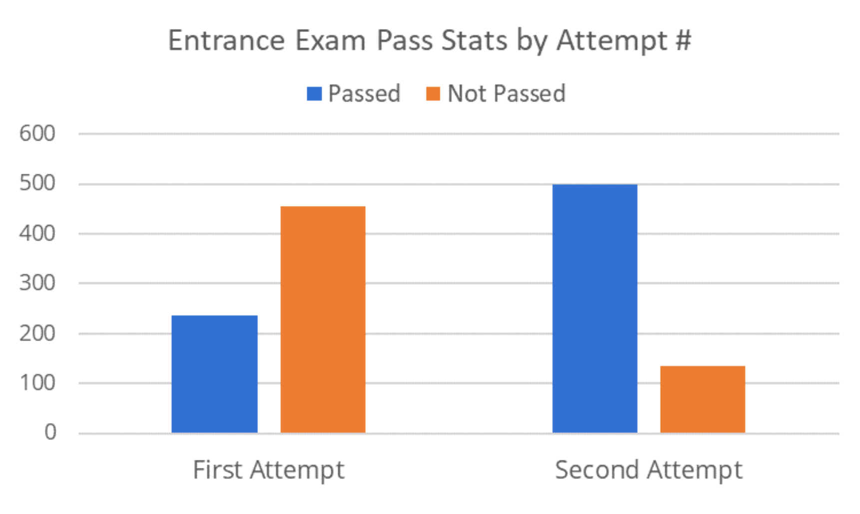
Per category box plots show us how spread out our data is, as well as statistical insights such as median and outliers.



With box plots across categories, we are able to see how our **continuous features impact out categorical target variable**.

Chi Squared Test of Independence

The Chi Squared test helps us measure the extent of a relationship between two categorical variables (input & target).



	Entrance Exam Passed		
	Passed	**Not Passed**	**Total**
First Attempt	236 **(34%)**	456 **(66%)**	692
Second Attempt	500	135	635
Total	736 **(55%)**	591 **(45%)**	1327

- **Visually, we can hypothesize that attempt number and pass rate have a relationship**, since the count plots look different across groups.
- The Chi Squared test uses a **contingency table** to measure **to what extent the per category distributions are different to each other**.
- At a **total population level, we would expect 55% of student to pass** the exam. However, at the **first attempt, it seems that only 34% of students pass**.
- The **greater the observed differences from what we expect, the higher the Chi Squared value**.
- Again, the **P-value** tells us to what extent our observed differences are **statistically significant**.

Other Feature Selection Methods

We have explored a variety of feature selection methods.

Manual feature selection methods help us reduce the number of features based on domain knowledge or data quality.

The **filter methods** we explored use basic statistics or statistical tests to help us filter out some features.

- Correlation Coefficients (continuous target – continuous inputs)
- ANOVA Testing (continuous target – categorical inputs)
- Box Plots (categorical target – continuous inputs)
- Chi Squared Testing (categorical target, categorical inputs)

In addition there exist **wrapper methods** and **embedded methods** that are more advanced. These include:

- Forward Search
- Backward Search
- Lasso Methods

Appendix

Third Party Data Sources Used

Name	Source
Indian cars dataset	https://www.kaggle.com/code/sanandachowdhury/cars-dataset/data
Airbnb dataset	http://insideairbnb.com/
Breast cancer dataset	https://archive.ics.uci.edu/ml/datasets/Breast+Cancer
KC house dataset	https://www.kaggle.com/datasets/harlfoxem/housesalesprediction