Feature Engineering

Agenda

- Feature engineering overview
  - Importance, examples, principles, types
- Data pre-processing
  - Data encoding, scaling/normalization, imputation
- Feature transformation
- Feature extraction
- Feature selection
- Dimension reduction
- K-means
- Exploratory Data Analysis

What is feature engineering?

- Preprocessing steps to transform existing features and create new features such that the data is best suited for the problem at hand
- Select and create the most useful predictor variables
- Decide which features should be kept, which ones to discard
- Want to maximize ratio of useful to irrelevant input
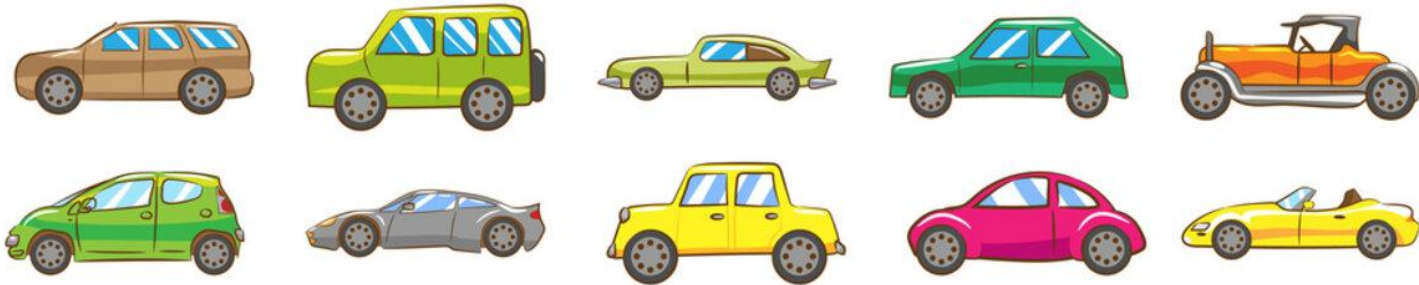 - Signal to noise ratio (SNR)

Let's look at an example!

Example: cars

Suppose we have a set of cars and their related characteristics

- Features: paint color, number doors, number wheels, fuel tank capacity, wheel diameter, car weight, age, etc.

We want to use an ML model to predict which cars have highest miles/gallon (mpg).

Discuss - Are all the features important?

Which features might be important?

Which ones are adding noise?

Discuss - Are all the features important?

Which features might be important? (mass, shape, fuel type)

Which ones are adding noise? (paint color, number of wheels/tires/doors)

Toyota Prius
57 MPG city

Ford Mustang
22 MPG city

Generally…

To keep (adds useful information):

- Features that don't depend on others (uncorrelated, low covariance)
  - E.g. weight of engine
- Features that change a lot (high variance)
  - E.g. fuel tank capacity

To discard (adds noise):

- Features that are constant or have low variance
  - E.g. # steering wheels = 1 for all cars
  - E.g. paint thickness - may vary slightly among cars but not much
- Features that are linearly dependent on other features (y=Ax+B)
  - E.g. # tires = # wheels

Why is feature engineering important?

Discuss ideas with the people around you

Why is feature engineering important?

- Enhance model generalization, which reduces overfitting
- Decrease training time
- Simplifies model to reduce parameters
- Avoid "curse of dimensionality"
  - As # features in model increases, amount data required to train model increases exponentially

Types of feature engineering

- Feature creation
  - Identify variables most useful in predictive model, subjective
  - E.g. collecting data in a survey
- Feature transformations
  - Manipulate predictor variables to improve model performance
  - E.g. scaling
- Feature extraction
  - Create new variables by extracting from raw data
  - Reduce volume into more manageable set
  - E.g. cluster analysis, edge detection, PCA
- Feature selection
  - Algorithms rank and analyze features to determine which ones are important and should be removed

Pre-processing

Understanding your raw data

Data parsing: extracting important information from strings/categorical data

- E.g., area code from phone number

Data encoding: Converting categorical/textual data into numerical format that algorithms can process

Types of categorical data:

- Ordinal: some kind of ranking is present (e.g., education level)
- Nominal: categories have no order/ranking (e.g., cities, colors, objects)

Data encoding techniques

One-hot encoding: N unique categories in variable → N binary variables

Dummy encoding: N unique categories in variable → N-1 binary variables

Label encoding: unique categories → unique integers

- Note that ML algorithms can misinterpret the integers as having rank

One-Hot Encoding

Places
New York
Boston
Chicago
California
New Jersey

→

New York	Boston	Chicago	California	New Jersey
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Scaling and normalization

Scaling: changing **range** of data (e.g. such that range is 0-1 for all features)

- Standardizes independent features
- Important for models that make use of distance
  - E.g. K-Nearest Neighbors, support vector machines (SVM)
  - Changes of 1 are considered equal in such models
 - Think about currency conversions - change in $1 and 1€ not comparable
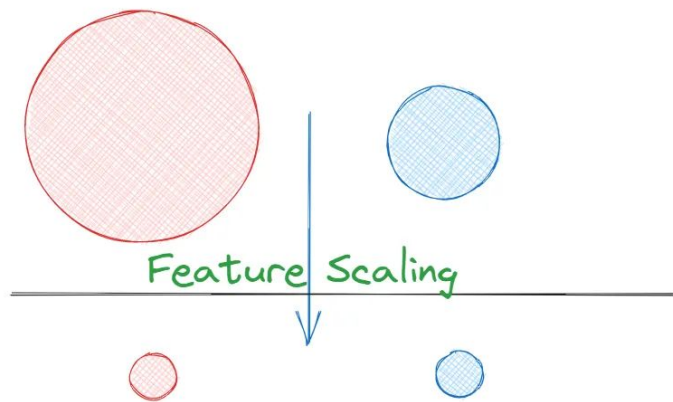
Feature Scaling

Common scaling methods

1. Min-max scaling: rescale to specific range (usually 0-1)
   a. X_scaled = (X — X_min) / (X_max — X_min)
   b. Good for bounded range and no outliers
2. Standardization: features have 0 mean and unit variance
   a. X_scaled = (X — X_mean) / X_std
   b. Good for features with unknown or non-normal distributions
3. Robust scaling: subtract median and divide by interquartile range (IQR).
   a. X_scaled = (X — X_median) / IQR
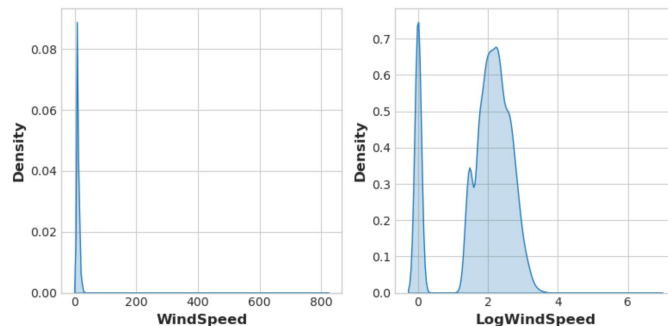   b. Good for outliers or skewed distributions

Normalization

Normalization: changing **shape** of the data

- Typically used for models that assume normal distribution of data
  - E.g. linear discriminant analysis (LDA), most models with "Gaussian" in name
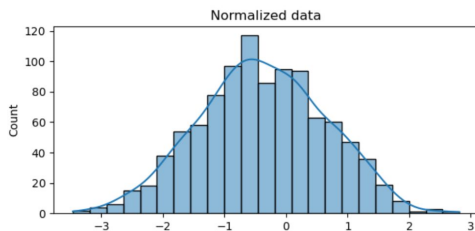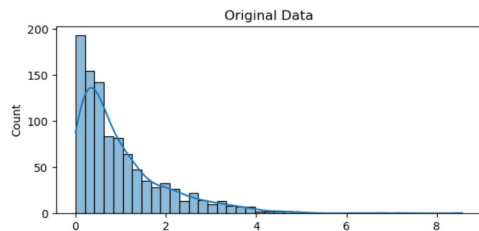
Box-Cox transformation
non-normal → normal

Log transformation
For skewed data

Important considerations

- Scale before feeding into ML model, except scale-invariant models
  - E.g. no need to scale for decision trees
- Avoid data leakage - scale only based on training data!!
  - E.g. when using cross-fold validation - split into folds, only apply normalization within the testing fold. Normalizing before cross-fold validation will mean that the test data statistics are leaked.
- Consistency - same transformations across training, validation, and testing datasets
- Use domain knowledge when choosing scaling techniques

What to do with incomplete data?

Datasets often have missing values. But many ML models expect complete data!

- Data entry or collection error
- Participants unwilling to answer certain survey questions
- Tests can be done for some but not all patients
- Participants leave trial

Types of missingness

Missing completely at random (MCAR): no systematic predictors of missingness

- E.g., Forgetting to input certain data entries

Missing at random (MAR): missingness is predicted by observed variables

- Probability of missing data related to observed data but not the actual missing data
- E.g., older people more likely to skip a certain question than younger people

Missing not at random (MNAR): missingness is predicted by unobserved variables

- The probability of missing data is related to the missing data itself
- E.g., participants who smoke/abuse/cheat may intentionally withhold such information

Approaches to missing data

- Remove entries with missing values
  - Only appropriate with MCAR, otherwise introduces bias
- Remove features with high percentage of missing data
  - Consider checking for feature importance before dropping
- Replace missing value with mean/median or most common category
  - Simple, but can lead to bias and doesn't add new information
- Model or regression imputation - use ML to predict missing data, using existing columns as predictor variables
- Maximum likelihood imputation - build an MLE with complete data as predictors and missing variable as target
  - Depends on assumptions of data distribution

Multiple imputation

Single imputation can lead to false precision. Multiple imputation accounts for uncertainty, which helps reduce bias and improve validity.

1. Imputation - generate missing values multiple time, resulting in multiple datasets with potential missing values
2. Same analysis on each dataset
3. Pool individual datasets into a final dataset

Incomplete Data   Imputed Data   Analysis Results   Pooled Results
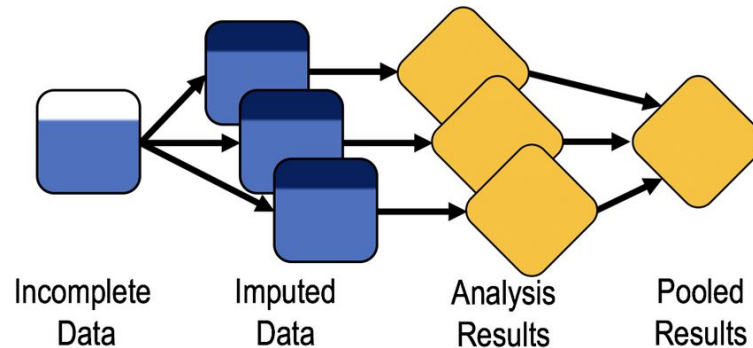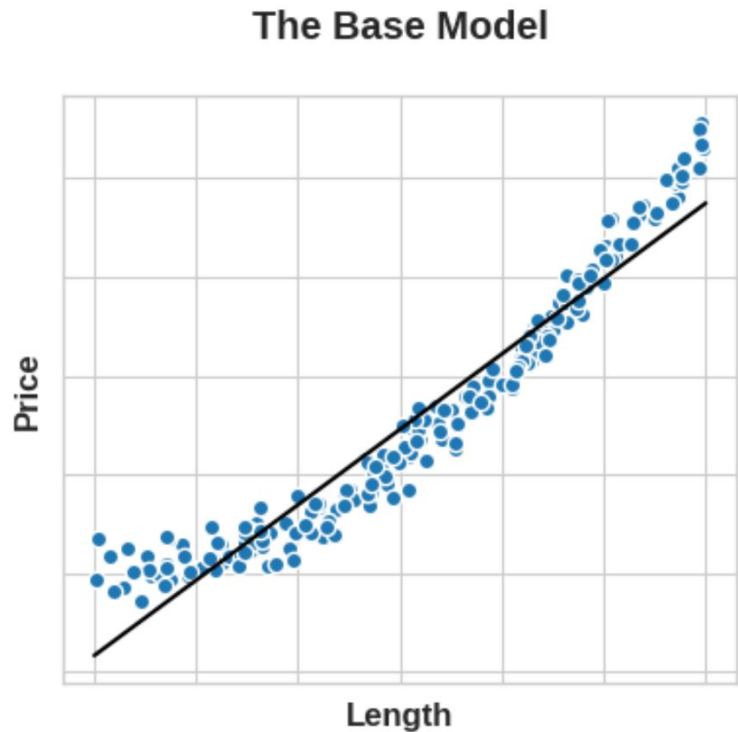
Feature transformations

Example: price of land vs. length

Consider a linear model.

- Feature: length
- Target: price

Linear models can only learn linear relationships.

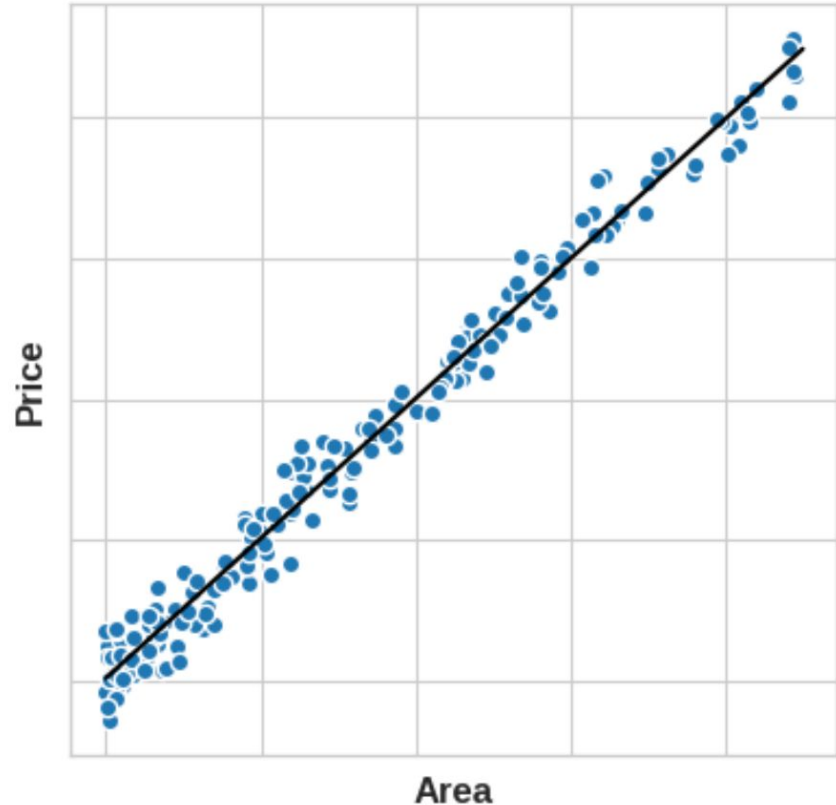**Directly fitting a linear model using length yields poor results - nonlinear.**

The Base Model

Price

Length

Example: price of land vs. length (con't)

Instead of length, let's use length^2 (AKA area)

Area: Price ratio is now linear!

Model fit is much better - now able to fit a parabola.

Using <u>transformations</u> can help model achieve best performance.

Feature extraction

Example: BMI

Imagine you have a group of people, and your goal is to predict BMI.

You have the mass of their total fat, carb, and protein intake.

	Fats (g)	Carbs (g)	Protein (g)	BMI
Person A	50	100	100	23
Person B	50	100	20	30
Person C	20	40	40	20
...	...	...	...	...

Example: BMI (con't)

Think about how different factors could be at play:
- What is total intake?
- How tall are they?
- How much are they exercising?

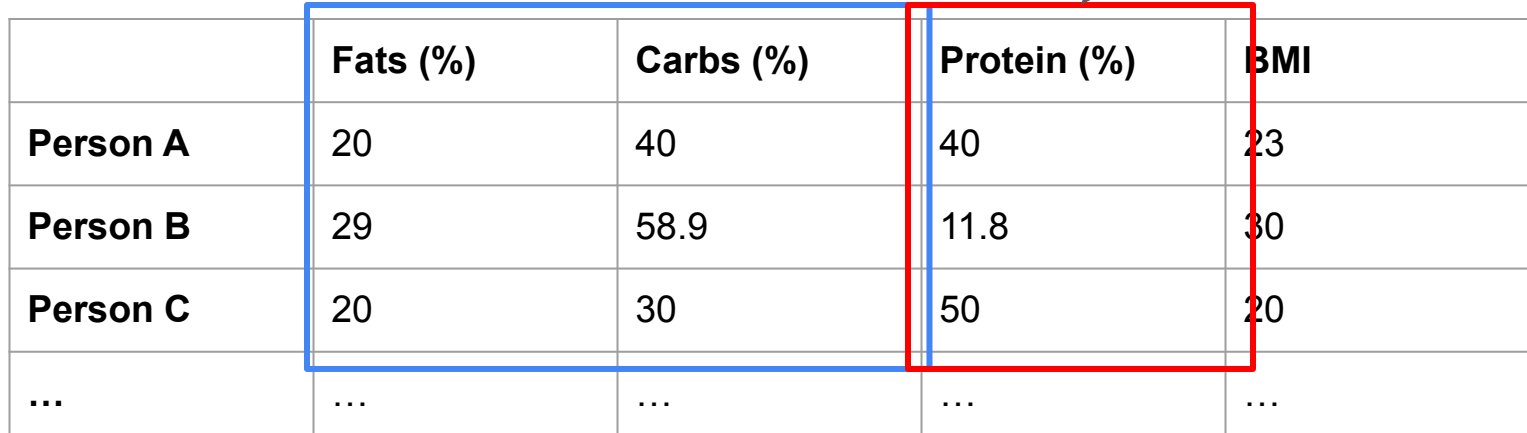Just looking at these - mass of fats and carbs don't seem to add much information!

	Fats (g)	Carbs (g)	Protein (g)	BMI
Person A	50	100	100	23
Person B	50	100	20	30
Person C	20	40	40	20
…	…	…	…	…

What if we had percentages?

Transforming the raw scores into percentages may be more informative.

These now have more of
a positive correlation!

Negative correlation

	Fats (%)	Carbs (%)	Protein (%)	BMI
Person A	20	40	40	23
Person B	29	58.9	11.8	30
Person C	20	30	50	20
…	…	…	…	…

Why would % be more informative?

With raw mass, A and B look like they consume the same amount of fat and carbs.

However, A consumes much more protein - maybe someone who is taller and exercises more (higher overall caloric needs)

The **proportions** of fat, carbs, and protein in one's diet may be more indicative of BMI.

	Fats (%)	Carbs (%)	Protein (%)	BMI
Person A	20	40	40	23
Person B	29	58.9	11.8	30
Person C	20	30	50	20
…	…	…	…	…

Clustering with K-Means

Using unsupervised learning to identify potential complex spatial relationships

Clusters help break relationships into easier chunks - cluster labels become new features

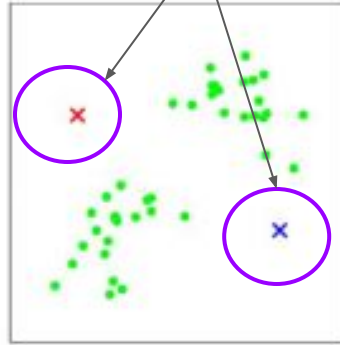K-means: assign points in dataset to one of k groups

- Centroids placed in feature space, data points assigned to nearest centroid
- Iteratively move centroid such that distance to data minimized (= compact)
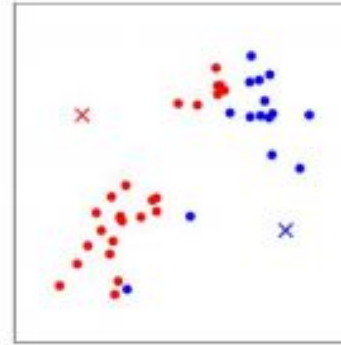
K-Means (visual)

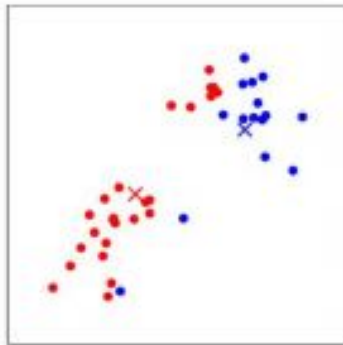Watch how cluster labels change as the centroids move
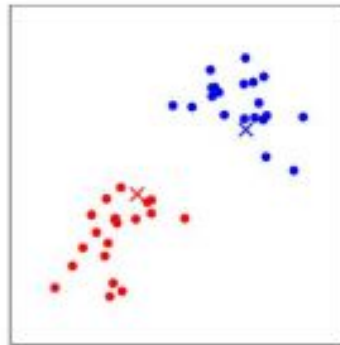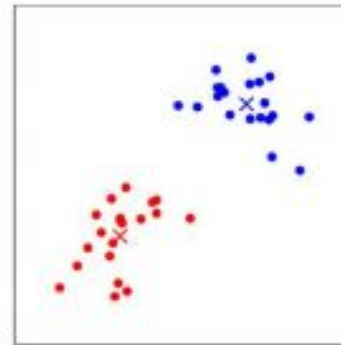

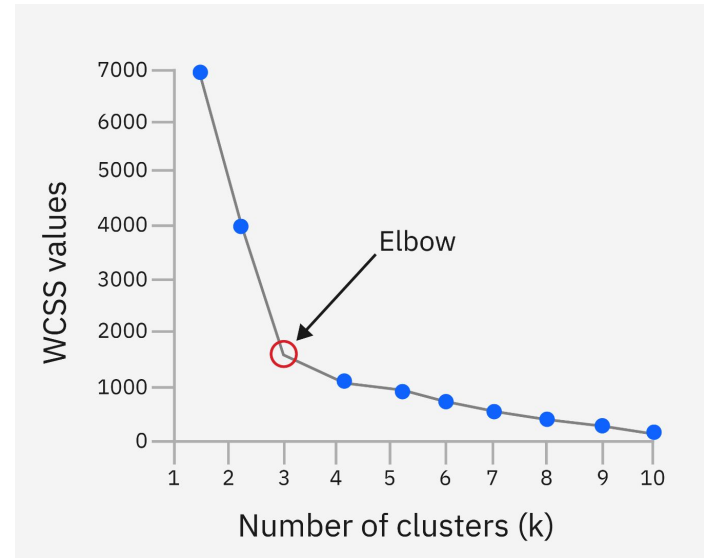(a)　　　　(b)　　　　(c)

(d)　　　　(e)　　　　(f)

Optimal k?

User selects k, so important to choose appropriate # clusters

Methods:

- Elbow method
  - Within cluster sum of squares (i.e. variance) vs. k
  - Choose point where variance
- Silhouette score
  - Measure how similar points within a cluster are compared to other clusters
 - -1 (bad) to +1 (good)

Principal Component Analysis (PCA)

Dimensionality reduction: Simplifies high-dimensional data into smaller set while preserving max variance → Makes data easier to explore, visualize, and process

PCA: dimensionality reduction method that creates "principal components" - linear combinations of original variables

Mathematically, important features = high variance, low covariance

- Maximizing variance between axes maximizes signal-to-noise ratio
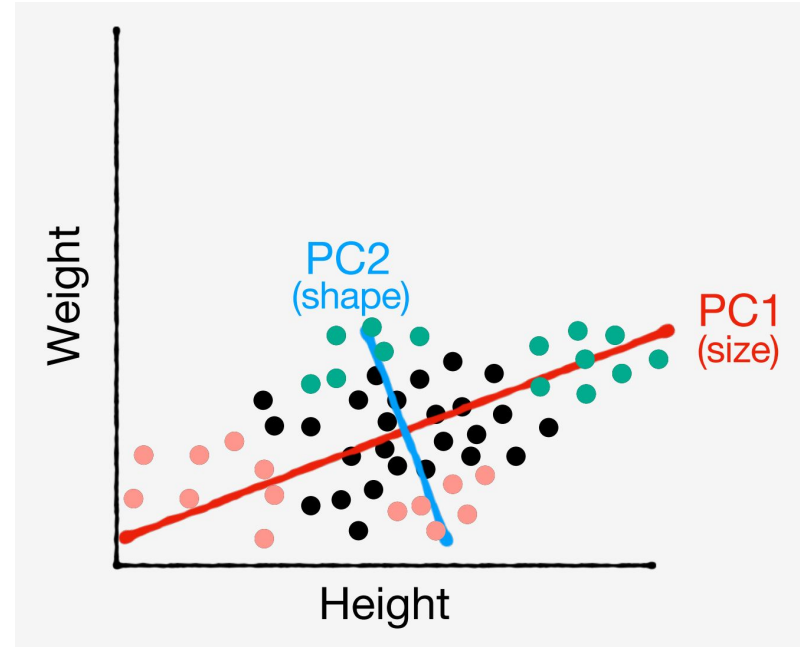
Height and weight could be represented by just PC1, which may be something like BMI
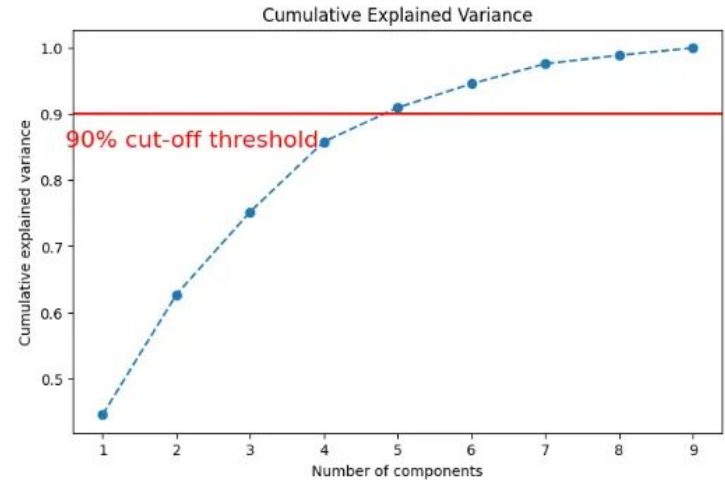
How many principal components?

Number of PCs should be manually set.

Use scree plot to look at explained variance ratio.

Plot cumulative explained variance to see how many PCs are required to get your desired level of variance.

"Loadings" can be extracted for each PC to see which original variables are contributing to the PC, and how much

Feature selection

Keep the most important features while discarding unnecessary data

Supervised:

- Wrapper - greedy algorithms that calculate relationship between feature subsets and target variables
  - Forward selection, recursive elimination
- Filter - features selected based on relationship with target (correlation)
  - ANOVA, Pearson's, Chi-Squared, then select top statistics
- Intrinsic/Embedded - some algorithms automatically select features while training
  - Decision trees, gradient boosting

References

https://maedbhk.github.io/MIT-Projects/methods/feature-engineering.html

https://medium.com/@yashbaravaliya206/feature-scaling-and-normalization-ca484a16882a

https://www.kaggle.com/code/alexisbcook/scaling-and-normalization

https://medium.com/@abhaysingh71711/mastering-feature-extraction-pca-t-sne-and-lda-in-machine-learning-part-1-0da5c3d978ad

https://www.ibm.com/think/topics/k-means-clustering

https://stanford.edu/~cpiech/cs221/handouts/kmeans.html

https://airbyte.com/data-engineering-resources/data-imputation

https://www.kaggle.com/code/residentmario/simple-techniques-for-missing-data-imputation

https://www.geeksforgeeks.org/feature-selection-techniques-in-machine-learning/