1. Can you tell the difference between machine learning and traditional programming (rule-based automation)?
P=5,

در برنامه نویسی سنتی، برنامه نویس باید کد برنامه را به صورت دستی بنویسد و تمام جزئیات کارکرد برنامه را به صورت دقیق تعیین کند. در این روش، برنامه نویس باید دانش فنی و تجربه کافی برای نوشتن کد برنامه داشته باشد.

اما در یادگیری ماشین، برنامه نویس کار خود را به ماشین می‌سپرد و اجازه می‌دهد که خودش یاد بگیرد. برای این کار، برنامه نویس باید به ماشین داده‌هایی را ارائه دهد که از آن‌ها می‌تواند الگوریتم‌هایی را برای حل مسائل یاد بگیرد. در یادگیری ماشین، ماشین به صورت خودکار الگوریتم‌هایی را برای حل مسائل یاد می‌گیرد.
در برنامه نویسی سنتی کامپیوتر برای تولید خروجی از یک سری قوانین از پیش تعریف شده پیروی میکند. در یادگیری ماشین کامپیوتر از فکر انسان تقلید میکند و از طریق مدل ها به این مهم دست پیدا میکند.

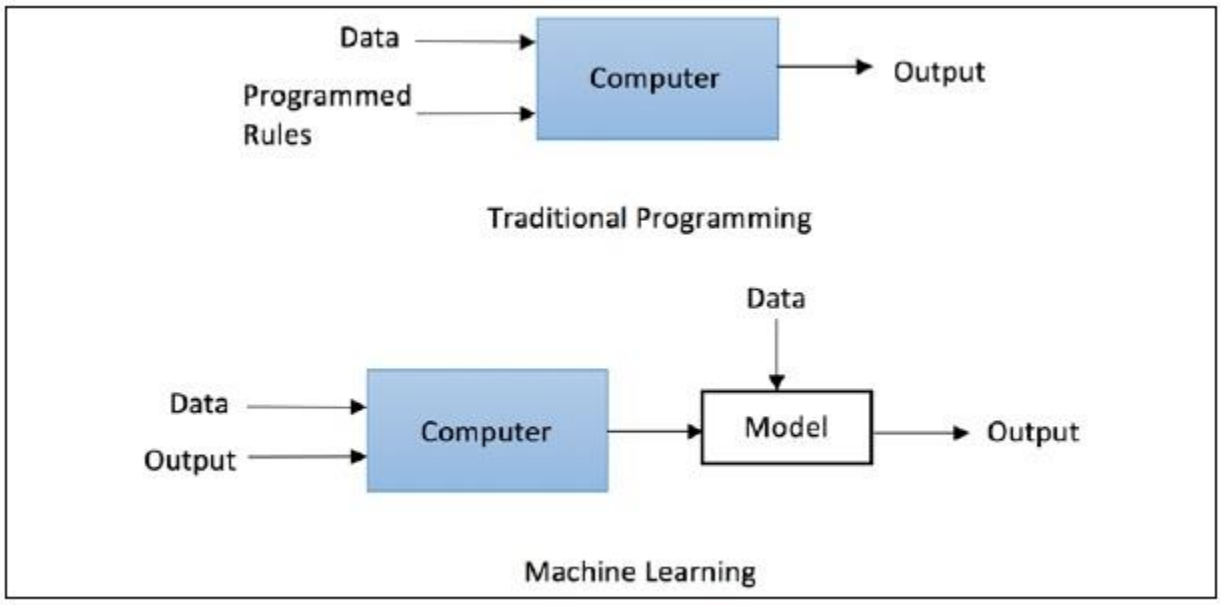


Figure 1.3: Machine learning versus traditional programming

2. What's overfitting and how do we avoid it?
P=15, [19- 25]

overfitting یکی از مشکلات عمده در یادگیری ماشین است که در آن مدل یادگیری ماشین به گونه‌ای تنظیم می‌شود که داده‌های آموزشی را به صورت دقیق ترجمه کند اما نتواند در مورد داده‌های جدید به خوبی عمل کند. به عبارت دیگر، مدل به صورت زیادی به داده‌های آموزشی بستگی دارد و قابلیت تعمیم پذیری ندارد.
به طور کلی مشکل overfitting در داده هایی جدید است یعنی با داده های آموزشی بسیار خوب عمل میکند ولی در مواجه با شرایط جدید دچار مشکل می شود (**high variance**) مانند حفظ کردن و ...

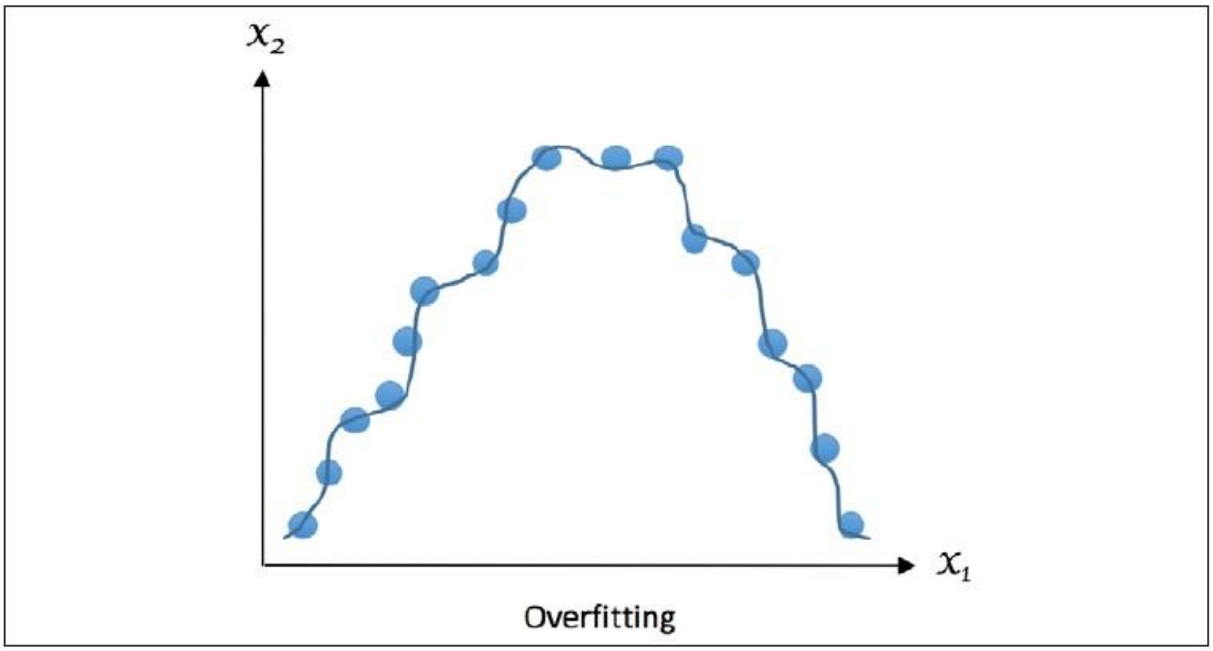
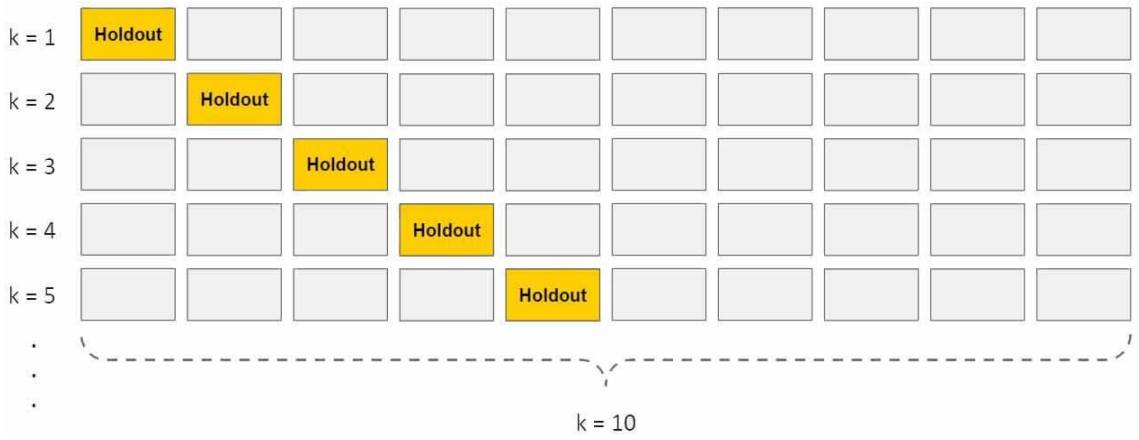


Figure 1.6: Example of overfitting

جلوگیری از overfitting

- **استفاده از روش ارزیابی مناسب**

یکی از موثرترین راهها برای جلوگیری از overfit شدن مدل، استفاده از روش **k-fold cross validation** است. در این روش داده به k بخش یکسان تقسیم می شود، سپس مدل به تعداد k تکرار آموزش و تست می شود. در هر تکرار یک بخش برای تست و k-1 بخش برای آموزش استفاده می شود. با اینکار از تمام ظرفیت داده برای تست استفاده می شود، ولی با یک ترفند بسیار جالب داده‌ها طوری برای تست استفاده میشوند که در بین داده های آموزش نباشند. و چون تا حدودی مدل در هر تکرار با داده زیادی آموزش می بینید، احتمال overfitting پایین می‌آید.

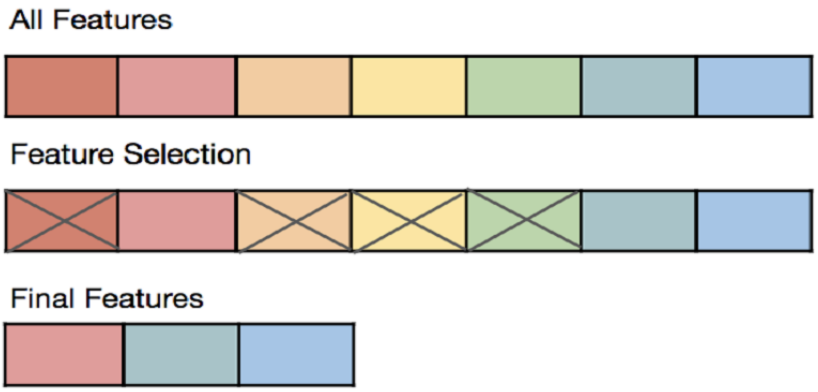


- **رگوله سازی(regularization)**

رگوله سازی با کمک تکنیکهای مختلف، مدل را مجبور می‌کند که از پیچیدگی دوری کرده و تا جایی که میتواند ساده‌تر باشد.
رگوله‌سازی به نوع مدلی که استفاده می‌کنیم بستگی دارد. برای مثال اگر مدل ما درخت تصمیم است، اینجا محدود کردن تعداد شاخه های مدل یه نوع رگوله سازی است.
در بعضی موارد برای رگوله کردن یک الگورتیم، پارامتری به تابع هزینه الگوریتم اضافه می‌کنند و بعد با کمک روشهای cross validationمقدار مناسب برای این پارمتر انتخاب میکنن

- **کاهش تعداد ویژگی‌ها**

با اینکه خیلی از مدلها به طور ذاتی انتخاب ویژگی را انجام می‌دهند، ولی ما میتوانیم به طور دستی با کمک یه سری روشهایی در ابتدا ویژگی‌های مناسب را انتخاب کرده و ابعاد داده را کاهش دهیم.



به خاطر داشته باشیم که هرچقدر ابعاد داده افزایش یابد، احتمال overfitting مدل به افزایش می یابد!

3. Name two feature engineering approaches.
P=30-31

1-Polynomial transformation
2-Binning

4. Name two ways to combine multiple models.
P=31-36

1-Voting and averaging
2-Stacking

5. Install Matplotlib (`https://matplotlib.org/`) if this is of interest to you. We will use it for data visualization throughout the book.

!pip install `matplotlib`