Python Data Products

Course 2: Design thinking and predictive pipelines

Lecture: Missing values

Learning objectives

In this lecture we will…

- Introduce the issues around datasets with missing values
- Investigate different strategies for dealing with missing values in datasets

Motivation

- Even the simple PM2.5 dataset we introduced had missing values (indicated by "NA")
- So far we dealt with them simply by **discarding** those instances:

In [4]:
```
dataset = [d for d in dataset if d[5] != 'NA']
```

- This was an okay strategy when dealing with a single feature where missing data was rare, but how would in generalize?
- In particular, this approach wouldn't work if **many** features might be missing

Concept: Strategies for dealing with missing values

In this lecture we'll look at three strategies for dealing with missing data:

- **Filtering** (I.e., discarding missing values), as we discussed on the previous slide
- **Missing data imputation:** filling in the missing values with "reasonable" estimates
- **Modeling:** changing our regression/classification algorithms to handle missing data explicitly

Missing data imputation

Even in cases where only a small amount of data is missing, simply discarding instances may not be an option. What else can we do?

Missing data imputation seeks to replace missing values by reasonable estimates

Missing data imputation

A simple scheme would be to replace every missing value with the **average** for that feature.

What are the consequences of such a scheme?

- The average may be sensitive to outlying values (though this could be addressed by using the **median** instead)
- The imputed value may or may not be "reasonable" (e.g. consider our "gender = male" feature)

Missing data imputation

Alternately we could consider more sophisticated data imputation schemes

- Rather than imputing using the mean, does it make more sense to compute the mean **of a certain subgroup** (e.g. if "height" is missing, can we impute using the average height of users with the same gender?)
- We could also train a separate **predictor** to impute the missing values (though this is complex if there are missing values for many different features)

Modeling missing data

How can we directly model the missing values within a regression or classification algorithm?

- One simple scheme: add an **additional feature** indicating that a value is missing
- e.g.:

  feature = [1, 0, 0] for “female”
  feature = [0, 1, 0] for “male”
  feature = [0, 0, 1] for “feature missing”

Modeling missing data

What predictions does the model make under this scheme?

feature = [1, 0, 0] for “female”
feature = [0, 1, 0] for “male”
feature = [0, 0, 1] for “feature missing”

$$
\begin{aligned}
\theta \cdot \text{feature} &= \theta_0 \quad \text{for female} \\
&= \theta_1 \quad \text{for male} \\
&= \theta_2 \quad \text{for “feature missing”}
\end{aligned}
$$

Note that θ_2 **learns** what value should be predicted when this feature is missing

Summary of concepts

- Discussed some simple schemes for dealing with missing data
- Introduced the ideas of **data imputation** and **modeling missing data**

On your own...

- Extend our previous code (on pm2.5 levels vs. air temperature) to handle missing features (other than the pm2.5 measurement itself)
- Experiment with different missing data imputation schemes and note their effect on performance