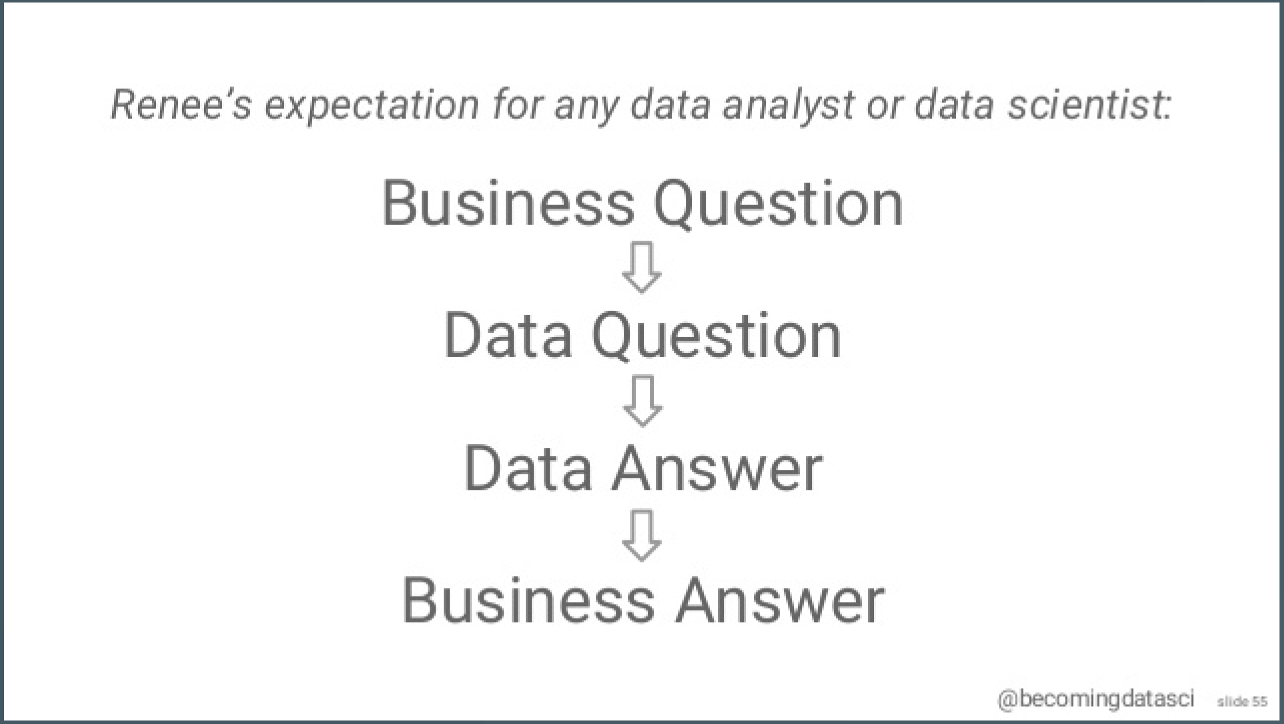
Define ML problems

Kaggle?

- Kaggle is great to practice model validation and other technical skills
- But some things are done for you: Like **defining the problem**
- I love Kaggle, but mostly agree with this critique

We've focused on the "inner arrow." Now let's zoom out!



Renee's expectation for any data analyst or data scientist:

Business Question
⇩
Data Question
⇩
Data Answer
⇩
Business Answer

@becomingdatasci slide 55

—Renee Teate, Becoming a Data Scientist, PyData DC 2016 Talk

Assignment

- Choose your target. Which column in your tabular dataset will you predict?
- Is your problem regression or classification?
- How is your target distributed?
- Choose your evaluation metric(s)
- Choose which observations you will use to train, validate, and test your model.
- Begin to clean and explore your data.
- Begin to choose which features, if any, to exclude. Would some features "leak" future information?

Choose your target

Which column in your tabular dataset will you predict?

Regression or classification?

You have options. Sometimes it's not straightforward:

- Discrete, ordinal, low cardinality target: Can be regression or multi-class classification.
- (In)equality comparison: Converts regression or multi-class classification to binary classification.
- Predicted probability: Blurs the line between classification and regression

> “ Sometimes questions that look like multi-value classification questions are actually better suited to ~~regression~~ predicted probability.
>
> For instance, “Which news story is the most interesting to this reader?” appears to ask for a category—a single item from the list of news stories. However, you can reformulate it to “How interesting is each story on this list to this reader?” and give each article a numerical score. ”

—Brandon Rohrer, Five questions machine learning can answer

> Binary classification problems can also be reformulated as ~~regression~~ predicted probability. (In fact, under the hood some algorithms reformulate every binary classification as ~~regression~~ predicted probability.)
>
> Questions of this type often begin “How likely…” or “What fraction…”
>
> How likely is this user to click on my ad? What fraction of today’s flights will depart on time?

—Brandon Rohrer, Five questions machine learning can answer

How is your target distributed?

- **Classification:** How many classes? Are the classes imbalanced?
- **Regression:** Is the target right-skewed?

Classification

How many classes?

```
y.nunique()
```

Are the classes imbalanced?

```
y.value_counts(normalize=True).max()
```

Is your majority class frequency >= 50% and < 70% ? If so, you can just use accuracy if you want. Outside that range, accuracy could be misleading. What evaluation metric will you choose, in addition to or instead of accuracy?

Fraud: Precision? Recall?

	Predicted False	Predicted True
Actual False		
Actual True		

Spotify: Precision? Recall?

	Predicted False	Predicted True
Actual False		
Actual True		

ROC AUC

The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

ROC AUC is the area under the ROC curve. It ranges from 0 to 1. Higher is better. A naive majority class baseline will have an ROC AUC score of 0.5.

ROC AUC measures how well a classifier ranks predicted probabilities. So, when you get your classifier’s ROC AUC score, you need to use predicted probabilities, not discrete predictions.

ROC AUC

When you get your classifier’s ROC AUC score, you need to use predicted probabilities, not discrete predictions. Your code may look something like this:

```python
from sklearn.metrics import roc_auc_score

# Get predicted probabilities for the last class
y_pred_proba = model.predict_proba(X_val)[:, -1]

# Get Validation ROC AUC score
print(roc_auc_score(y_val, y_pred_proba))
```

Highly imbalanced classes?

Try ideas from Learning from Imbalanced Classes :

- "Adjust the class weight (misclassification costs)" — most scikit-learn classifiers have a `class_balance` parameter.
- "Adjust the decision threshold" — we did this last module. Read Visualizing Machine Learning Thresholds to Make Better Business Decisions.
- "Oversample the minority class, undersample the majority class, or synthesize new minority classes" — try the the imbalance-learn library.

Regression

Is the target right-skewed?

```python
import seaborn as sns
sns.distplot(y)
```

```python
y.describe()
```

If the target is right-skewed, you may want to **"log transform"** the target.

> “ Transforming the target variable (using the mathematical log function) into a tighter, more uniform space makes life easier for any model.
>
> The only problem is that, while easy to execute, understanding why taking the log of the target variable works and how it affects the training/testing process is intellectually challenging. You can skip this section for now, if you like, but just remember that this technique exists and check back here if needed in the future. ”

— Terence Parr & Jeremy Howard, The Mechanics of Machine Learning, Chapter 5.5

> “ Optimally, the distribution of prices would be a narrow “bell curve” distribution without a tail. This would make predictions based upon average prices more accurate. We need a mathematical operation that transforms the widely-distributed target prices into a new space. The “price in dollars space” has a long right tail because of outliers and we want to squeeze that space into a new space that is normally distributed. More specifically, we need to shrink large values a lot and smaller values a little. That magic operation is called the logarithm or log for short. ”

— Terence Parr & Jeremy Howard, The Mechanics of Machine Learning, Chapter 5.5

How to log transform

```
import numpy as np
y_train_log = np.log1p(y_train)
y_val_log = np.log1p(y_val)
model.fit(X_train, y_train_log)
y_pred_log = model.predict(X_val)
y_pred = np.expm1(y_pred_log)
mean_absolute_error(y_val, y_pred)
```

> “ To make actual predictions, we have to take the exp of model predictions to get prices in dollars instead of log dollars. ”

— Terence Parr & Jeremy Howard, The Mechanics of Machine Learning, Chapter 5.5

Regression metrics

- Mean Absolute Error is arguably most intuitive
- Root Mean Squared Error is more common, "punishes outliers" more
- R^2 is "percent of variance explained"

Regression & Classification

Choose your evaluation metric(s)

Classification & regression metrics are different!

- Don't use *regression* metrics to evaluate *classification* tasks.
- Don't use *classification* metrics to evaluate *regression* tasks.

Scikit-learn has lists of popular metrics.

Are false positives or false negatives more costly? Can you optimize for dollars?

Choose train, validate, and test sets.

Are some observations outliers? Will you exclude them?

Will you do a random split or a time-based split?

You can re-read How (and why) to create a good validation set.

Next steps

- Begin to clean and explore your data.
- Begin to choose which features, if any, to exclude. Would some features "leak" information from the future?

Leakage

> Make sure your training features do not contain data from the “future” (aka time traveling). While this might be easy and obvious in some cases, it can get tricky. …
>
> If your test metric becomes really good all of the sudden, ask yourself what you might be doing wrong. Chances are you are time travelling or overfitting in some way.

— Xavier Amatriain

What if my model isn't "good"?

Produce a detailed tribute to your wrongness. That is science!