ALT/FNDATA ACADEMY · STUDENT STUDY SHEET

Study sheet: Module 9, machine learning on auction data

This session introduces machine learning through two concrete tasks built on auction and resale records: predicting a realized price and classifying whether a lot will sell. Both tasks look approachable at first glance and both contain a trap that catches working analysts as often as students, using information that would not have been available at the moment of prediction to predict an outcome that has already happened. The session spends real time on that trap before it spends any time on a model, because a model trained on leaked information will report excellent accuracy and will be worthless in practice. Students use the sandbox to explore the documented fields available before a sale (designer, high estimate, vendor, category, and the date itself) and export a training extract, and the class works through, conceptually and in an optional Python extension, how a regression model would predict usd_price_decimal and how a classification model would predict sold or unsold, always splitting training and test data by sale_date so that the model is evaluated the way it would actually be used, on lots it has not seen from a time period it has not seen.

What you should be able to do

1. Frame a realized-price prediction task as regression and a sold-versus-unsold task as classification, and identify which documented fields are legitimate predictive features for each.
2. Explain target leakage in plain terms and identify at least two ways it could enter a model built on this dataset, including using realized price to predict realized price.
3. Explain why a train/test split for this data must be made by sale_date rather than at random, and describe what a random split would hide about a model's real-world performance.
4. Use the sandbox to explore candidate features and export a training extract limited to fields that would be known before a sale.
5. Interpret a simple accuracy or error metric for each task and identify at least one reason the metric could look good while the model is still unfit for use.
6. (Extension) Load an exported training extract in Python, fit a baseline regression model and a baseline classification model, and evaluate each on a time-based holdout.

Key terms

- **Target leakage:** using information that would not have been available at prediction time as a feature, which inflates apparent model performance.
- **Point-in-time split:** dividing training and test data by date so that all test rows occur after all training rows, matching how the model would actually be used.
- **Regression:** a model that predicts a continuous number, here the realized price, usd_price_decimal.
- **Classification:** a model that predicts a category, here whether a lot's status is sold or unsold.
- **Baseline model:** a simple model, such as linear or logistic regression, used as a floor of comparison before trying anything more complex.
- **Feature:** an input variable available to a model, here drawn only from documented pre-sale fields such as designer, sale_estimates_high_usd_price, vendor, and sale_date-derived values.

- **Holdout set:** the portion of data set aside and not used in training, reserved to evaluate how the model performs on data it has not seen.
- **Class imbalance:** a condition where one classification outcome, such as sold, is much more common than the other, which can make a naive accuracy metric misleading.

Core sandbox query

The query the module is built around. Run it, then change the brand or category and compare.

```
SELECT designer, model, item_title, vendor, sale_date,
       sale_estimates_high_usd_price,
       usd_price_decimal,
       status
FROM all_watches_data
WHERE sale_estimates_high_usd_price > 0
ORDER BY sale_date;
```

Full lesson, more queries, and the homework are in Module 9 at academy.altfndata.com. Questions: info@altfndata.com.