Video-3:

Notebook-link:

00:04 Interpreting machine learning models for better understanding of data

This is about understanding the data with Machine Learning. Random forests allow us to understand more about the data, deeper and quicker

Larger datasets - How to go about them?

Is it the same approach as normal?

2:53 Try random forests and deep learning for unstructured data

Random forests - much better with structured data
Speech, music, images - DL is better suited rather than random forests
Collaborative filtering -> Neither ML nor DL works really well

7:00 Missing values Handling - Testing set

In testing set -> missing values
Median value of the testing data may not accurately depict the distribution in the training set

Therefore, it is important that we find the distribution values according to the training dataset and fill in the missing values of the testing dataset with these values(from training dataset)

14:00 Large dataset problem statement

Corporacion Favorita Grocery Sales Forecasting ->
Location, class of the product, where is it stored | We intend to predict how many units of a particular item are sold?

Independent variable? Dependent variable?

How many units of each kind of product was sold in each store on each day during a 2-week period? -> We need to predict

The information we have is how many units of each kind of product was sold in each store in the last few years in addition to some metadata like where is it stored, what is the oil price etc..

RELATIONAL DATASETS -> Different information that we can join together to understand the data better

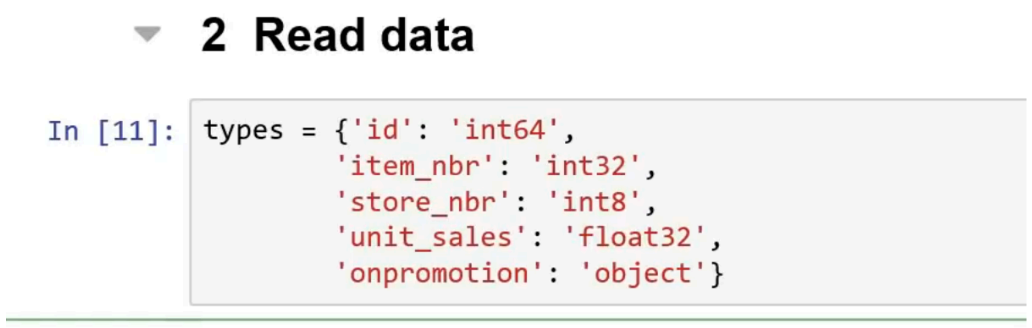
STAR SCHEMA -> Transactions table in the middle with the metadata around it

SNOWFLAKE SCHEMA -> Additional information from item table, store table etc…

19:10 Optimizing memory usage is key when working with large data sets

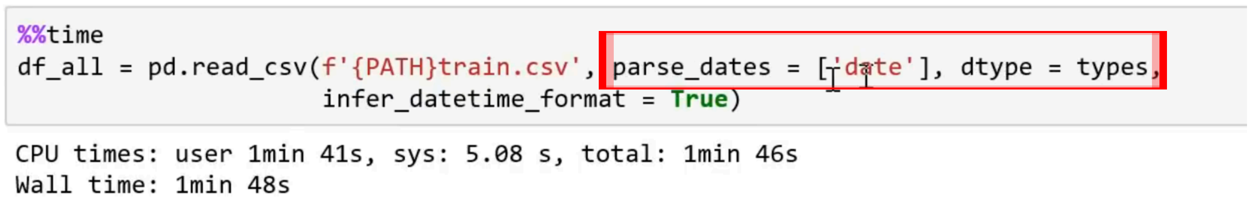
Very large datasets -> No sufficient RAM

Datatypes dictionaries



2 Read data

```
In [11]: types = {'id': 'int64',
                  'item_nbr': 'int32',
                  'store_nbr': 'int8',
                  'unit_sales': 'float32',
                  'onpromotion': 'object'}
```



```
%%time
df_all = pd.read_csv(f'{PATH}train.csv', parse_dates = ['date'], dtype = types,
                     infer_datetime_format = True)
```

```
CPU times: user 1min 41s, sys: 5.08 s, total: 1min 46s
Wall time: 1min 48s
```

Use the smallest number of bits to specify information in a specific column.

Smaller datatypes -> run faster

Instead of reading the whole dataset at once, it is good practice to find a random sample from the dataset from pandas and understand it -> **shuff**

Also -> deal with booleans with missing values-> Missing with False

23:46 Utilize most recent data for accurate predictions

Dates -> really important part of any data
Make sure dates from training dataset and testing data don't overlap

If we need to use a smaller dataset-> get the more recent part

We might have some important information about the distribution from 4 years ago, but, to understand the current relevance of the data, we have to make sure to include more recent data

25:49 Creating initial easy models and exploring data analysis

Taking log of the unit sales -> In the competition, RMSlog(E+1) as log 0 doesn't make sense is the criteria.
Clipping the sales to positive -> competition might ask us to only consider the positive sales and not include negative -> returns
Categories -> Numericals (if any)

Don't create a tree with all the 120 million samples. See what is the time it is taking for 10000 or a million. This sample size of the subset is crucial as the individual decision tree only works with random subsample of the dataset.

29:58 Optimizing code for efficient machine learning computations

Profiling -> Sometimes, when we are sure that we have made the problem as computationally efficient as possible, but, still some line of code in a library is taking a long time to run, we can try to learn this and see if we can make it easy for the library to run the code.

32:03 Optimizing random forest parameters for faster performance and improved accuracy.

Check with different values of different hyperparameters discussed in the previous lecture.

We observed that even with all these, random forests are not working too well. What might be the reason? When do the random forests work and when do they not?

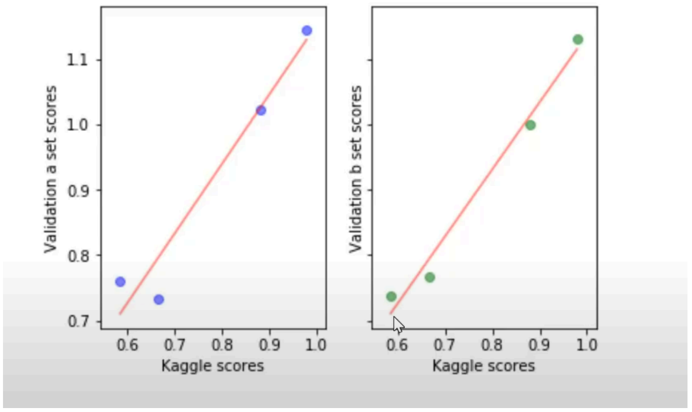
36:39 Enhancing model performance by capturing seasonality and trend effects.

38:46 Practice in finding and fixing mistakes is crucial in machine learning.

Generally, programming in machine learning is difficult because we don't get an exception when we use some different hyperparameters for instance. We only get inferior results. So, it is important to experiment and find the best model parameters and hyperparameters.

43:17 Sales forecasting based on competition insights

45:36 Validating the reliability of the model for production or test use.



In addition to the test set being used at the end of the competition, just for testing out our results, one more important use of the test set is to calibrate the validation set.

In the above graph, there are two validation sets a and b. Four models are tried on both the validation sets. The validation set should be selected if the model scores on it closely match their corresponding Kaggle scores, indicating an almost linear relationship. In this case, validation set B…

49:55 Interpreting Machine Learning Models

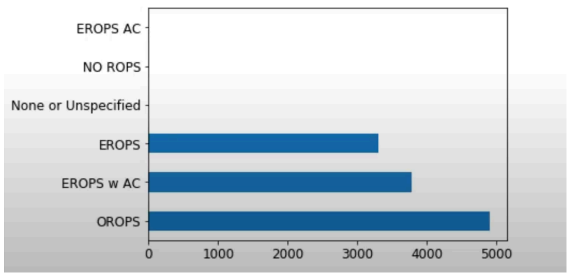
Normally, as we have seen in the OOB score calculation, our output in a problem is the average of all the outputs of all the trees that do not have that specific row included in the training set (as the training set has random rows for each tree). Instead of using the mean of the output by the trees alone, we can also use variance or standard deviation of all the values too. Higher standard deviation means that all the trees are not so confident about the decision as the output of each tree is varying too much. This standard deviation is the confidence interval in which the mean of the prediction values can vary.

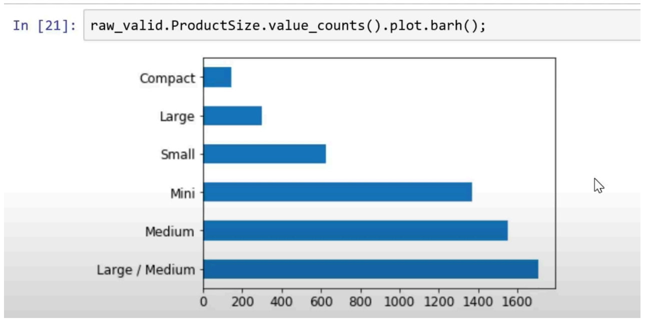
59:30 Using parallel processing in Python for faster execution

Instead of making predictions from all the trees normally, it would be great if we could use some parallel processing to speed up the predictions.

1:03:58 Grouping, averaging, and analyzing prediction accuracy

Having high prediction accuracy in our model is necessary, but not sufficient. How can we surely know if the high prediction accuracy is not a coincidence from having many trees leaving many statistical possibilities. So, it is important to group based on categories in the columns/features and see what groups can we be more confident about our predictions and what groups are we less confident about our predictions.



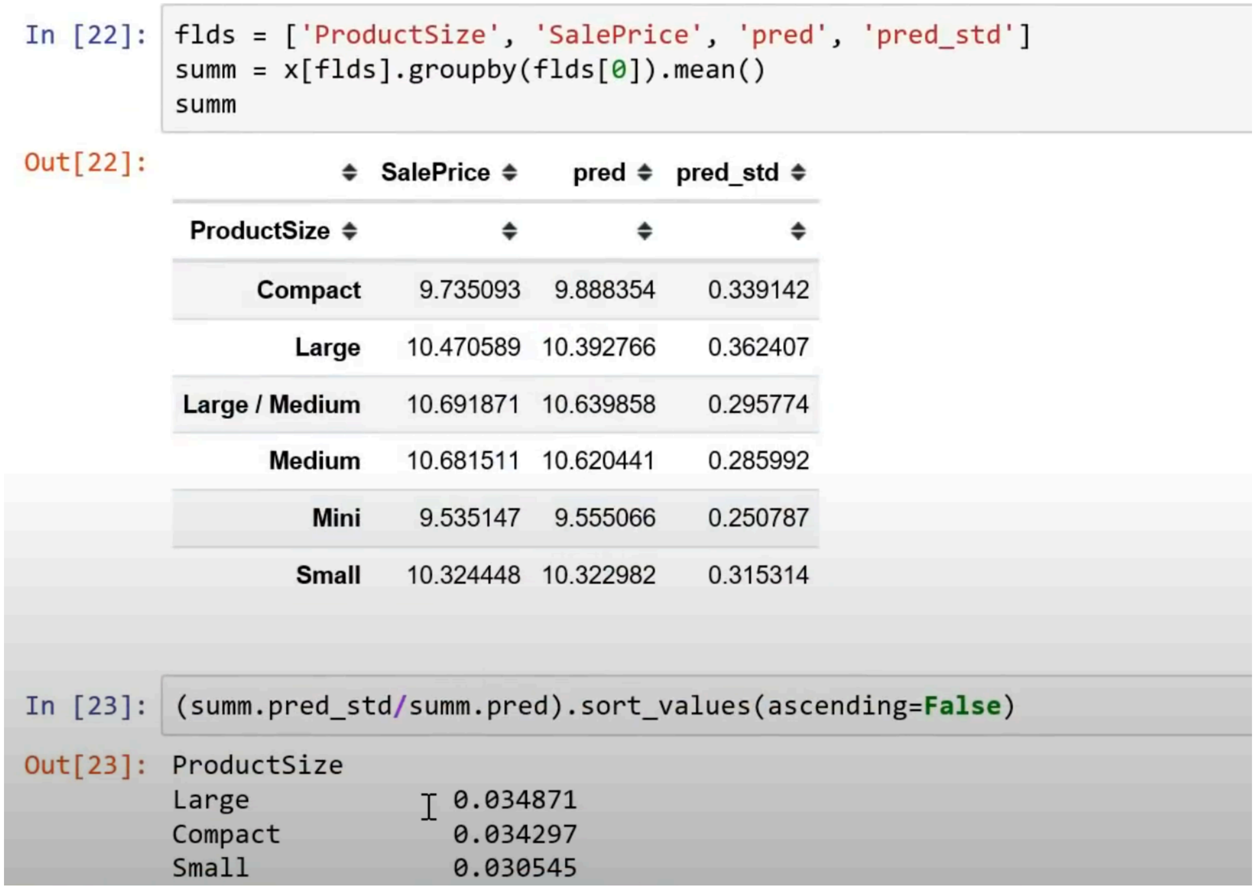


```
In [21]: raw_valid.ProductSize.value_counts().plot.barh();
```

These are the amount of data points in each group.

1:06:18 Confidence interval usage in machine learning

We can get an intuitive understanding of how confident are we on our predictions based on this: We can group by the respective categories, take our predictions(average values) and standard deviation of the prediction values of the respective categories. The ratio of standard deviation and the prediction values would tell us about how confident we are. Higher ratio value -> more standard deviation -> uncertain about our output or low prediction accuracy which means the same thing.


```
In [22]: flds = ['ProductSize', 'SalePrice', 'pred', 'pred_std']
         summ = x[flds].groupby(flds[0]).mean()
         summ
Out[22]:
```

	SalePrice	pred	pred_std
ProductSize			
Compact	9.735093	9.888354	0.339142
Large	10.470589	10.392766	0.362407
Large / Medium	10.691871	10.639858	0.295774
Medium	10.681511	10.620441	0.285992
Mini	9.535147	9.555066	0.250787
Small	10.324448	10.322982	0.315314

```
In [23]: (summ.pred_std/summ.pred).sort_values(ascending=False)
Out[23]: ProductSize
         Large          0.034871
         Compact        0.034297
         Small          0.030545
```

We can observe that Large,Compact and small production sizes are not doing too good. We cannot be very confident about our results in the respective categories comparatively. This also sits well with the fact that forests are not a good approach with smaller groups as it is very difficult to ensure that the representative data of the respective group is a part of the random subset of the dataset every time in every forest. (We can see from the no. of data points bar plot above that these are the groups with least no. of data points)

1:10:28 Understand and analyze the data before focusing on calculations

1:12:23 Data leakage and collinearity are key problems in machine learning models.

For understanding more about the data, we find the features that are more important (in a minute). We can see that certain features are more important and contribute to the results of the model while the rest of the features are practically non-existent. Removing the irrelevant features would not affect the model by the very definition.

There are two unique situations possible in this context. After we find that a set of features is really important, we should clarify that with our client or someone from the field if that is actually important. If they say that a very important feature in our analysis is not so important in the field, it means that there is something wrong with the data, maybe in bookkeeping, someone meddled with the data to make things easy for them. This is called Data leakage and not good for creating good models.

Collinearity -> Having two dependent columns doesn't affect random forests too much. However, in the phase where in each decision tree, all the columns(features) and the splitting points are iteratively compared with each other, it seems that both the columns are important and this may cost us some time. So, we should try to remove the columns that do not actually contribute to the results of the model.

1:16:37 Feature importance helps in understanding predictive model accuracy

How is this feature importance calculated?

We can know that a feature is important, by removing the feature and then training the random forests with the remaining columns and seeing how confident and accurate our results are from the model. However, this is computationally ineffective and we also lose a lot of time.

Instead we can take the validation set and then destroy a column by randomizing all the values across the column. Now, the random values in the column are not useful as they are not corroborating with the rest of the dataset. We can then check by what values accuracy and R-square values are taking a dip and this determines that a feature is very important.

1:23:20 Improve accuracy and share insights

After we know that a certain column is very important, we can still analyze it more, so that we can understand more about the relationship of the feature with the dependent feature.