Simple Linear Regression using Spark MLLib | Data Preprocessing

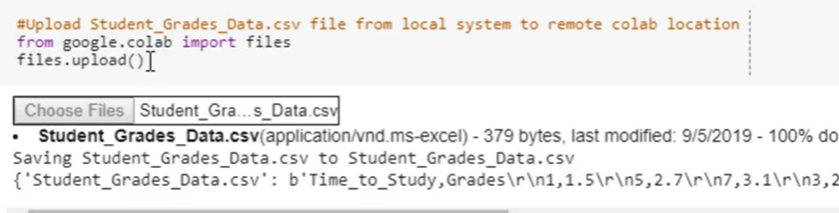


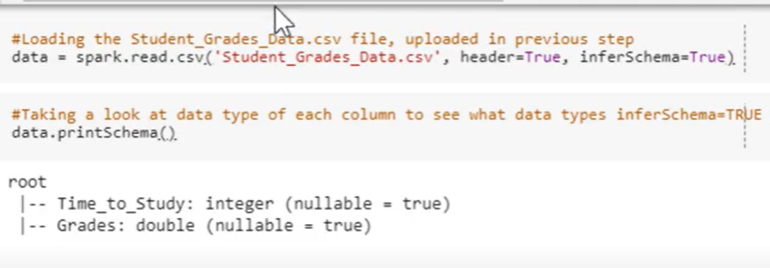
First object spark

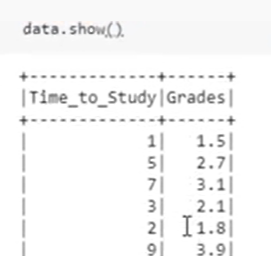
Spark object ti initiate spark session

Getor create- used to share spark contect

Can be shared with one jve

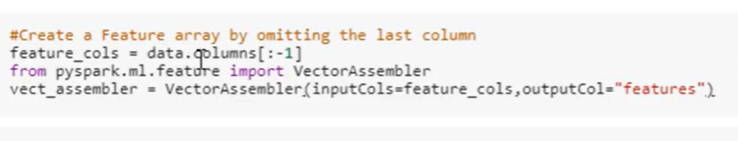




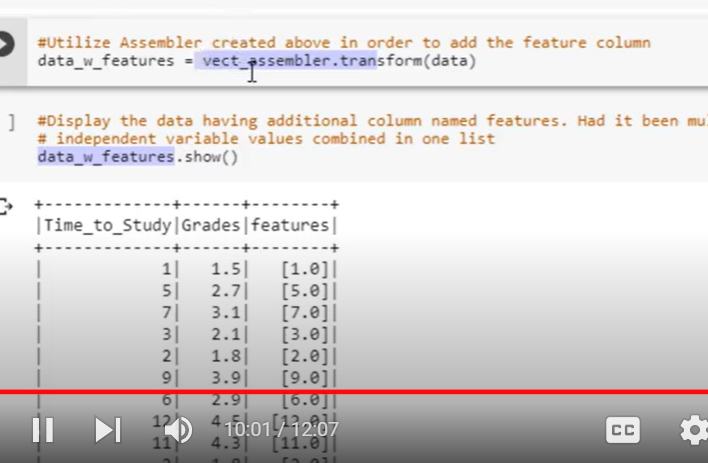


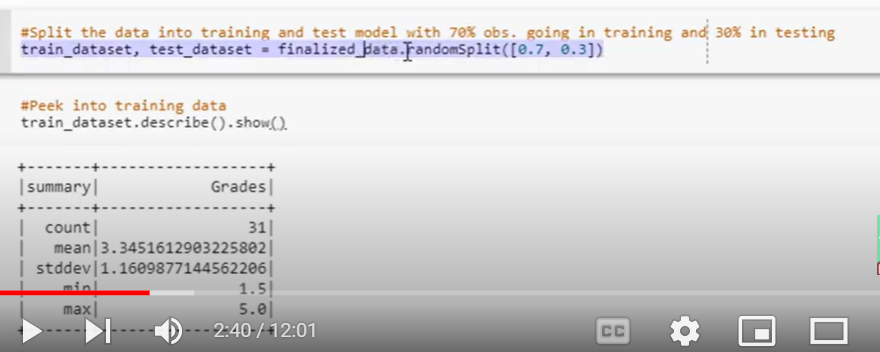
According to ml lib orr dataframe should contain only features and lablels

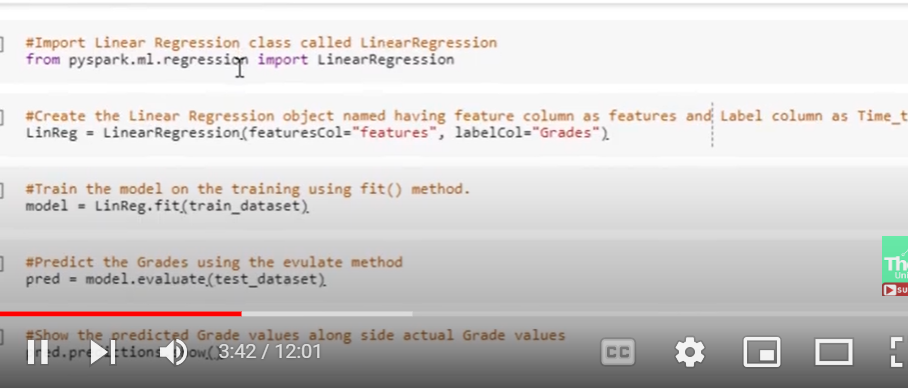
All 100 independent variables should be converted to one single array

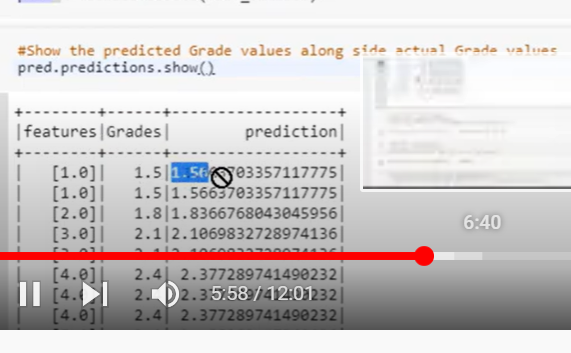


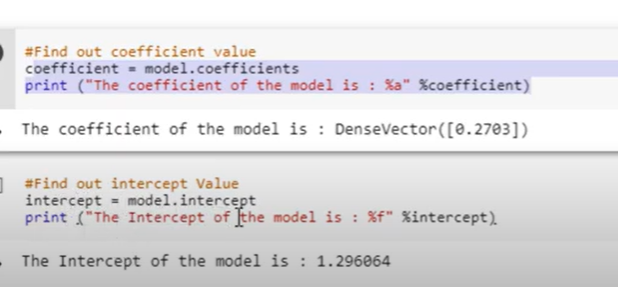
Vector assembler converts all independent variables into one column



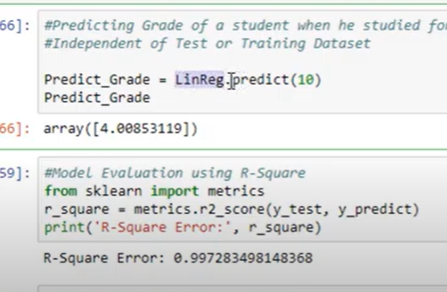


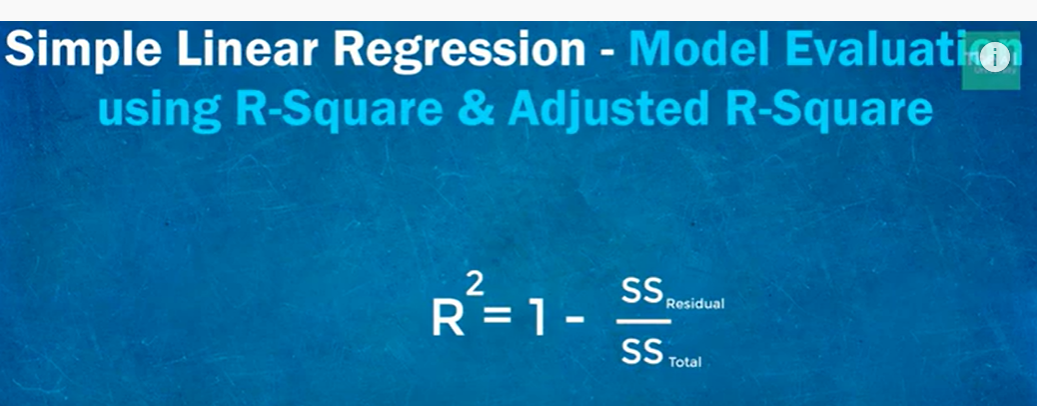












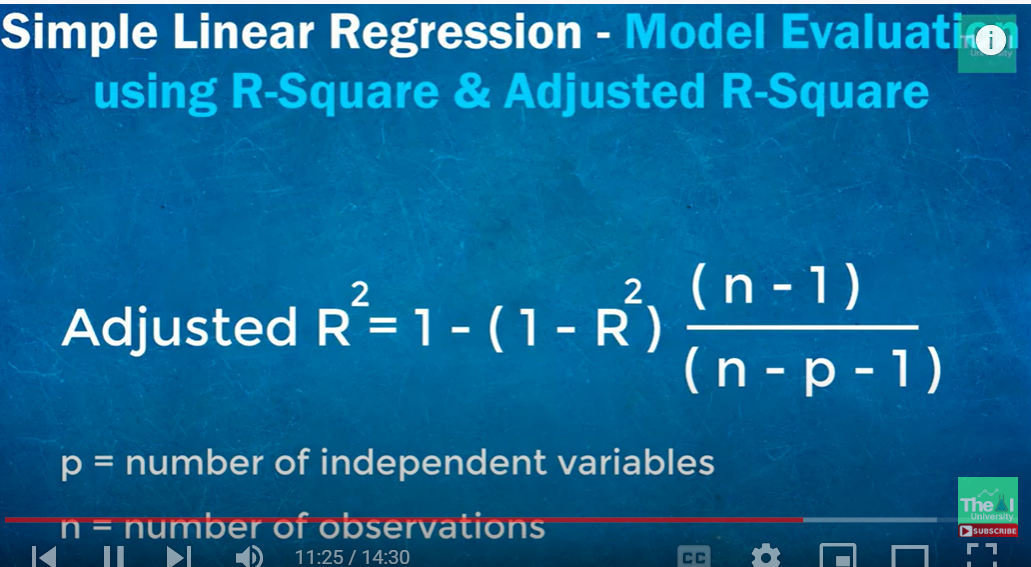
Many variables : ssresidual/sstotal will be very less reagarless of wether the feature contributing for predicting target

There will be some correlation between features and target eventough they contribute or not

Sow r2 will not decrease

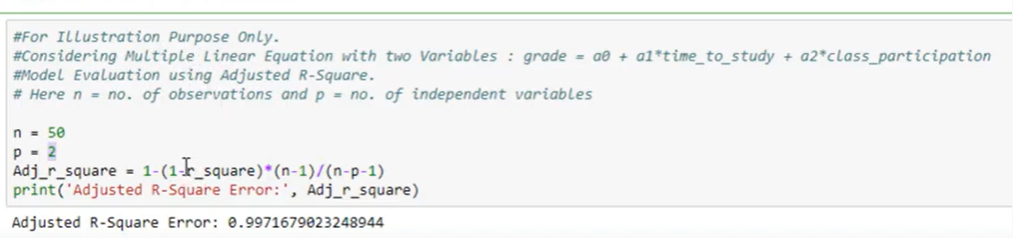
--

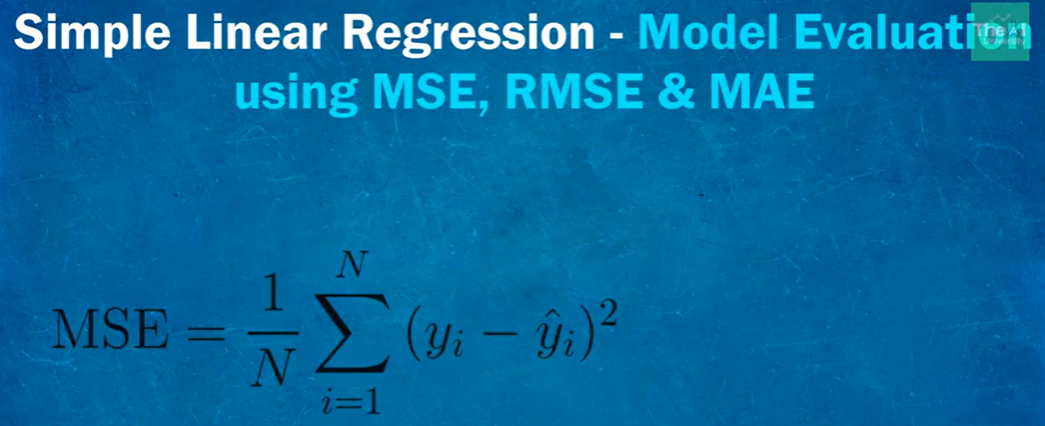
We use adjusted r2 :



Used when we have multiple variables

It penalize model when we use independent variable that does not help our model





Mostly used

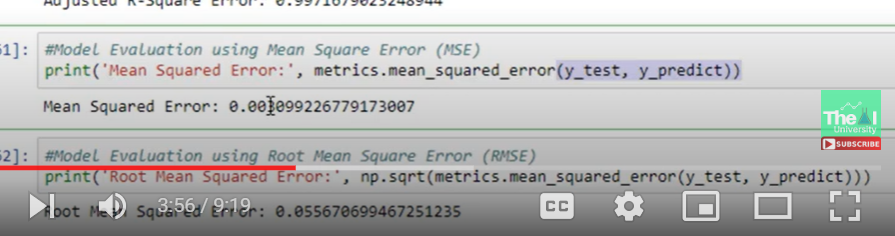
But useful

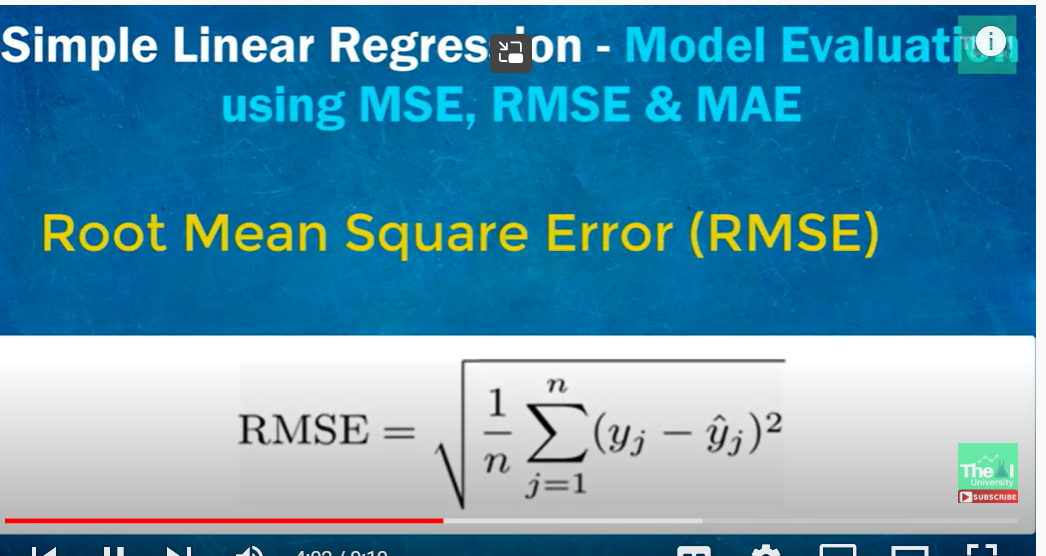
Mse close 0 : good model

Close to 1 : worst model

Prob: over estimate model if we have noise data

Outliers are penalized





Why this square root?

To bring errors on the same scale

Infact benefit of penalizing large errors

MAE:

Error as an avg of absolute diff

All differences -linear score- all individual weights are treated equally

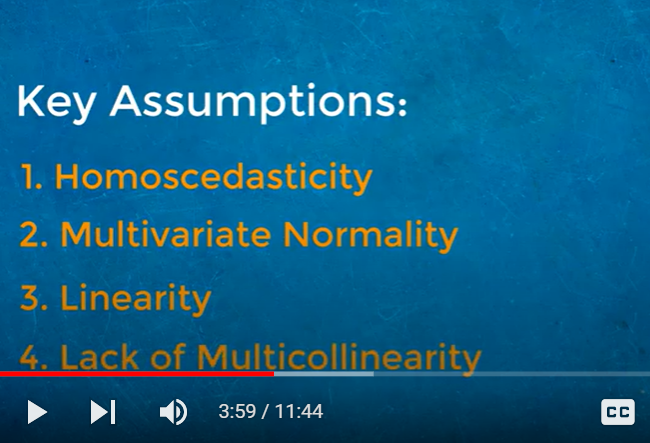
Not sensitive to outliers, doesn’t penalize huge errors

:

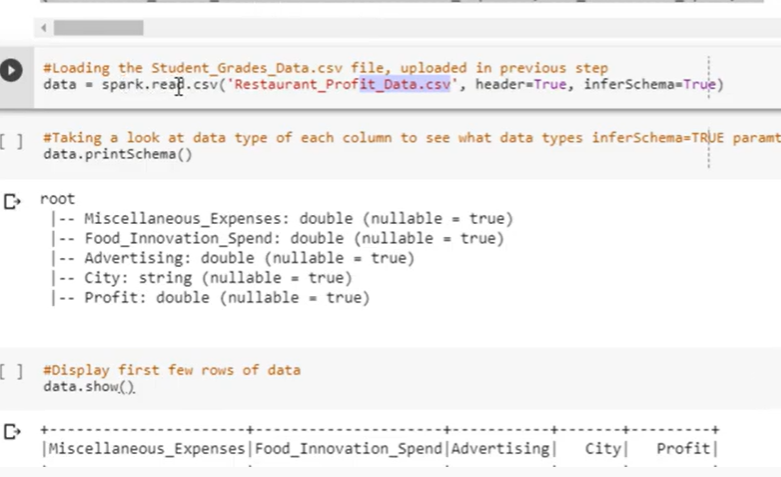
Outliers: mae

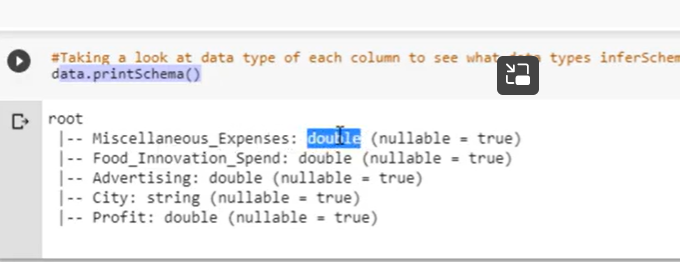
Or msc

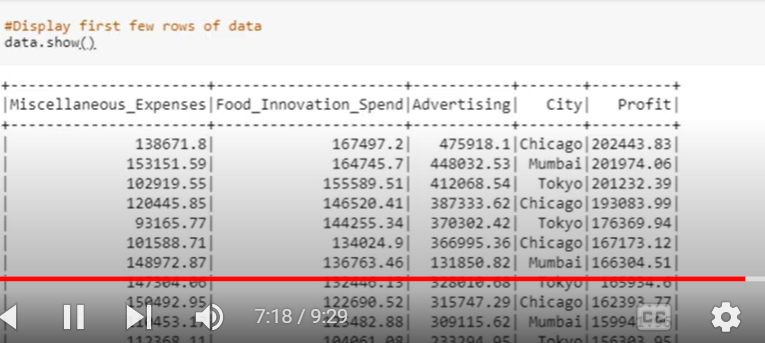
Assumptions:

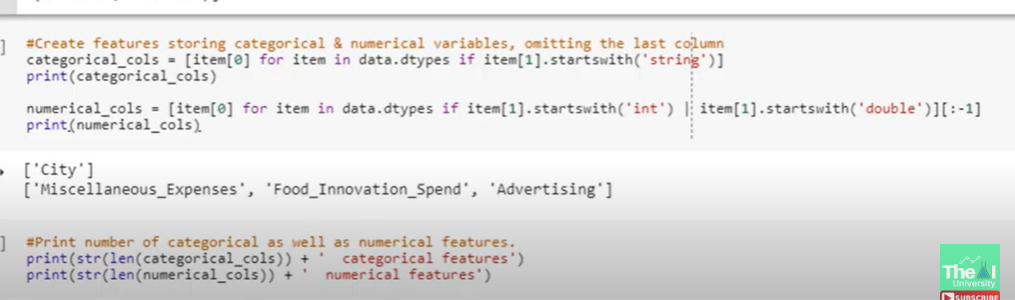


Multiple Linear Regression using Spark(PySpark) MLLib | Coding Part - 1

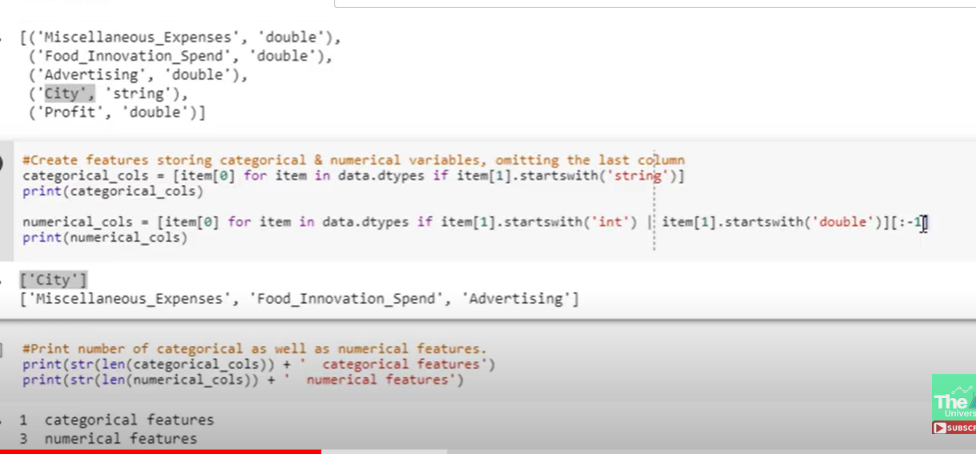




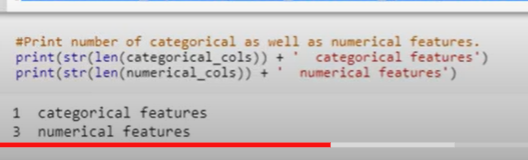


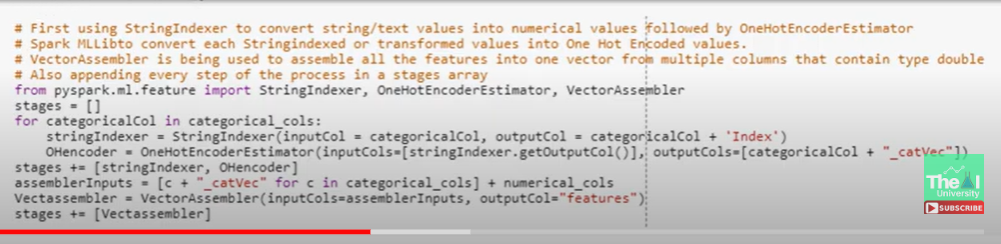






[-1] since profit is target column and we used to exclude





Empty stage vector,

We will build ml pipeline

For this we need to build seq of stages

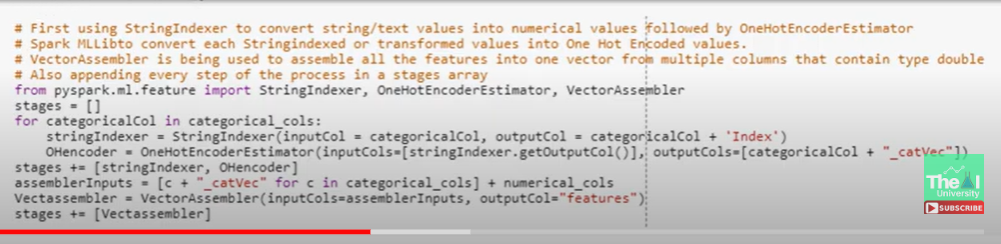
Each stage either transformer or estimator

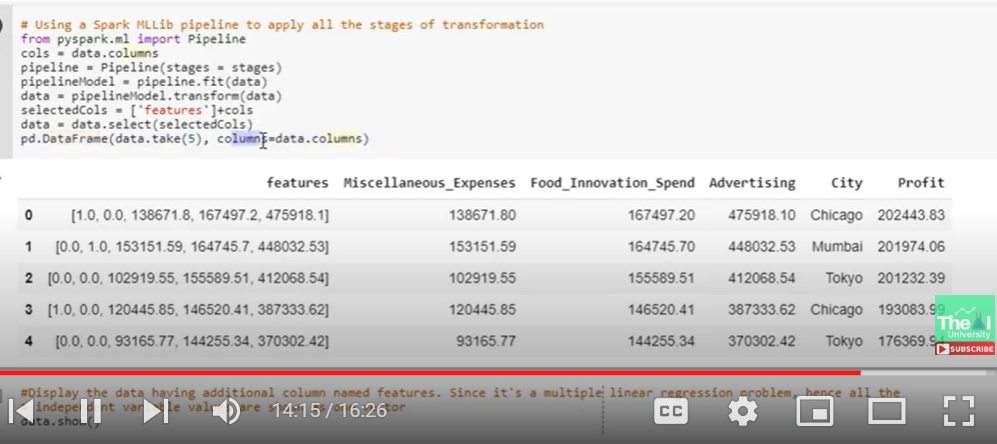
Once we built we need to define seq in pipeline

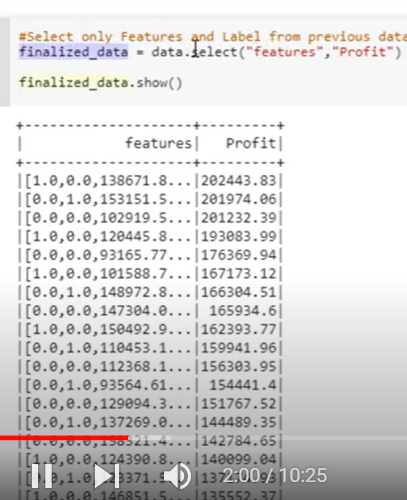
Stringindex: convert categorical into numeric based on freq

Most fre =0

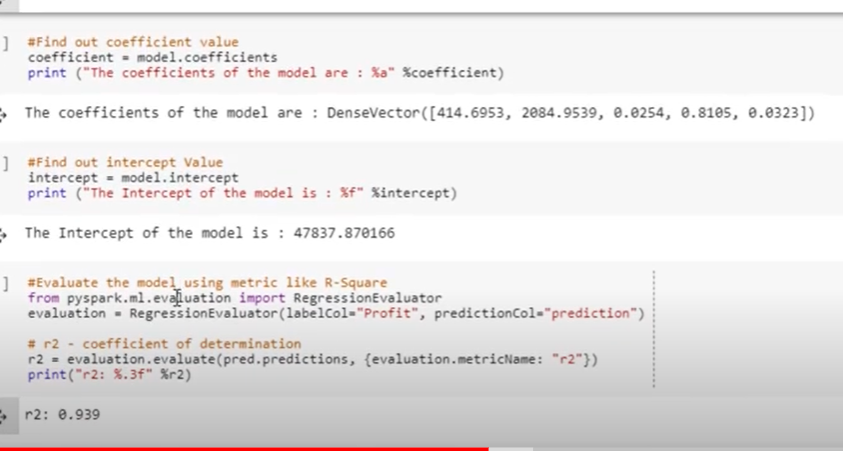
Least fre =big numb

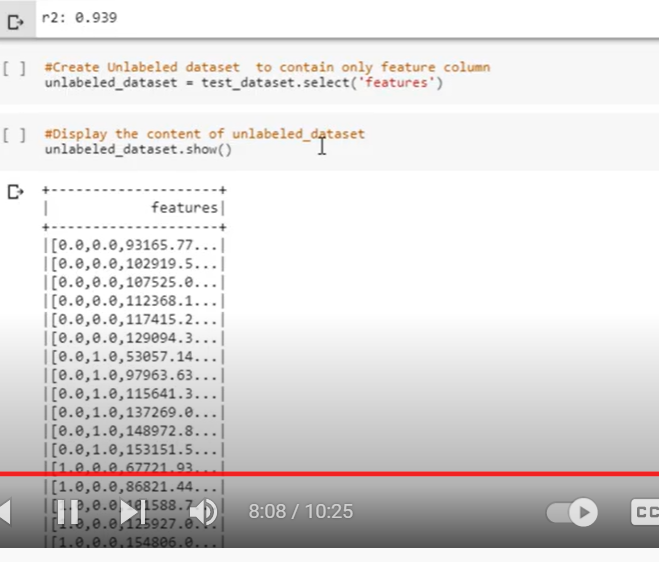


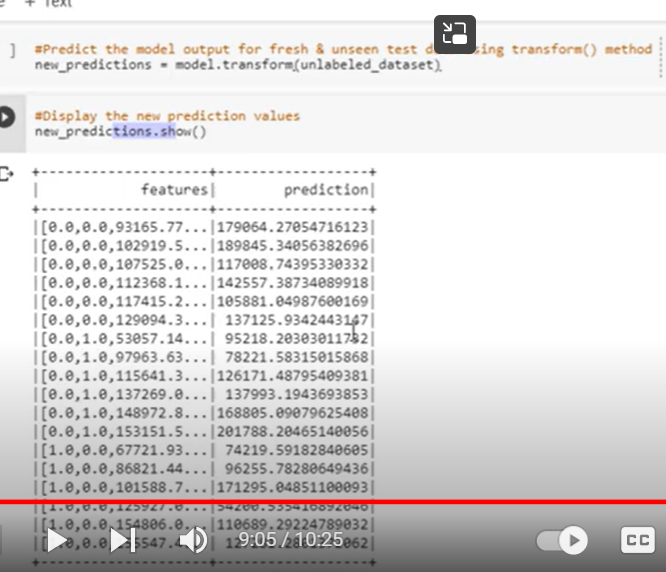












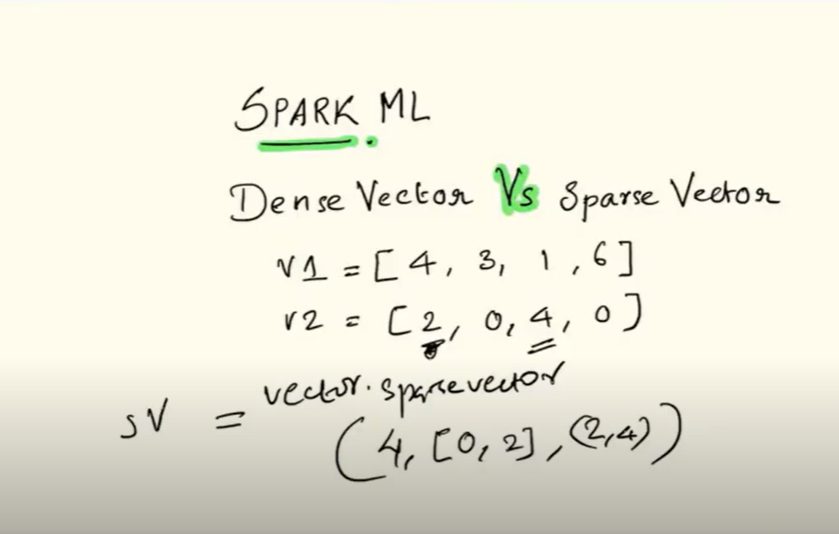
[Guide to implementing Linear Regression in Pyspark and R | by Kieran Tan Kah Wang | Towards Data Science](https://towardsdatascience.com/guide-to-implement-linear-regression-in-pyspark-and-r-26a94fe938a3)

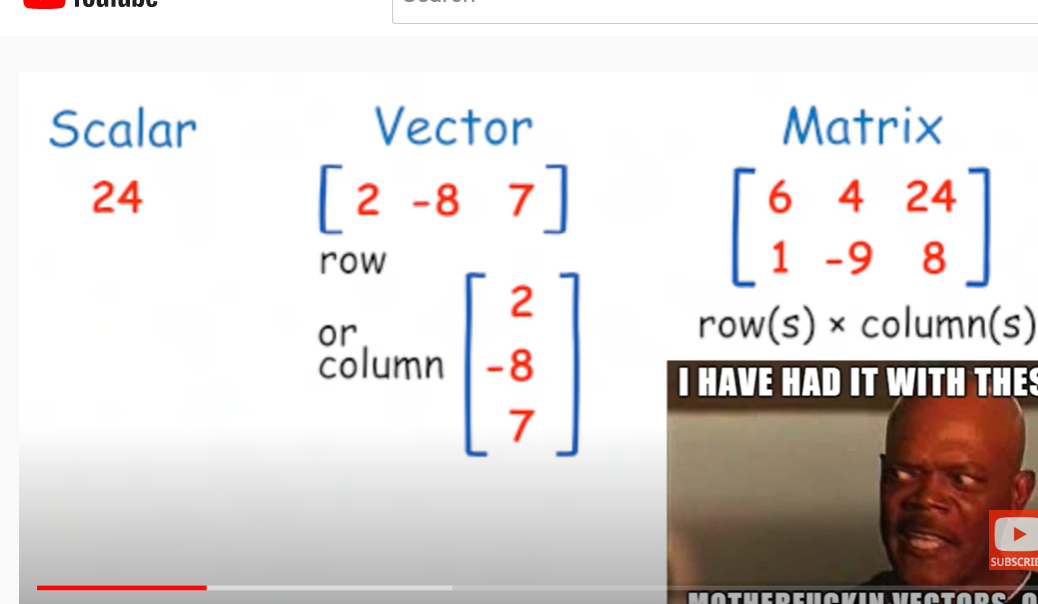
[Pyspark | Linear regression using Apache MLlib - GeeksforGeeks](https://www.geeksforgeeks.org/pyspark-linear-regression-using-apache-mllib/)

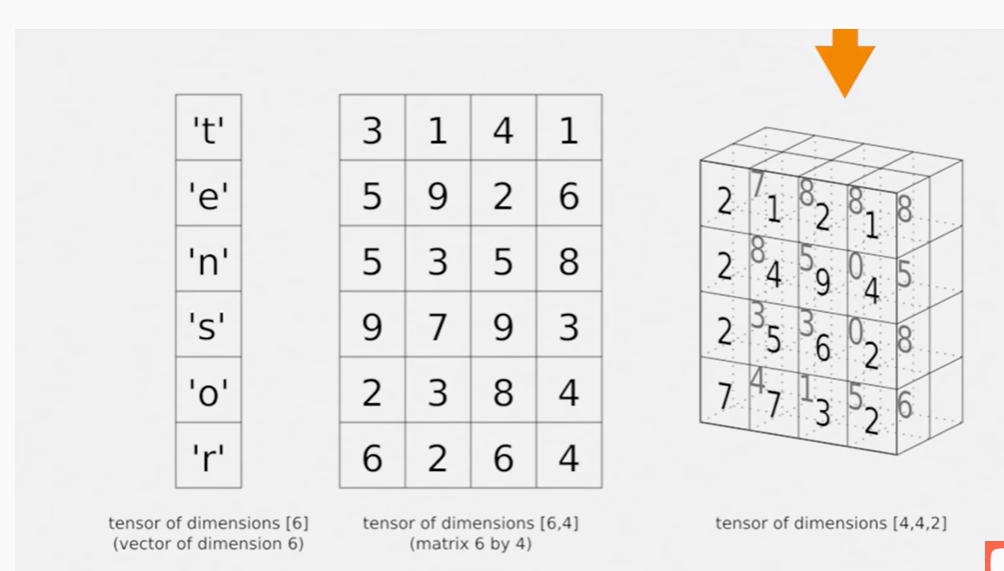
[Advanced topics - Spark 3.3.0 Documentation (apache.org)](https://spark.apache.org/docs/latest/ml-advanced.html)

Most imp reference:

[9. Regression — Learning Apache Spark with Python documentation (runawayhorse001.github.io)](https://runawayhorse001.github.io/LearningApacheSpark/regression.html)

‘’ 





Pyspark plots

[Plot data from apache spark in Python/v3 (plotly.com)](https://plotly.com/python/v3/apache-spark/)

[pyspark.pandas.DataFrame.plot.area — PySpark 3.2.1 documentation (apache.org)](https://spark.apache.org/docs/3.2.1/api/python/reference/pyspark.pandas/api/pyspark.pandas.DataFrame.plot.area.html)

[9. Regression — Learning Apache Spark with Python documentation (runawayhorse001.github.io)](https://runawayhorse001.github.io/LearningApacheSpark/regression.html)

[Visualise pipelines — Kedro 0.18.3 documentation](https://kedro.readthedocs.io/en/stable/tutorial/visualise_pipeline.html)

[7. Data Exploration — Learning Apache Spark with Python documentation (runawayhorse001.github.io)](https://runawayhorse001.github.io/LearningApacheSpark/exploration.html#univariate-analysis)