Name: Nikhil sharma

Email address: nikhil.niksharma15@gmail.com

Contact number: 8057483977

Anydesk address: 204 391 522

Years of Work Experience: Fresher

Date: 20th Aug 2021

Self Case Study -2: Car Damage Insurance Prediction

"After you have completed the document, please submit it in the classroom in the pdf format."

Please check this video before you get started: https://www.youtube.com/watch?time_continue=1&v=LBGU1_JO3kg

Overview

Vehicle insurance is an insurance that is used for all the automobile that uses road as a medium. Its main purpose is to provide financial protection against

- Physical damage or bodily injury caused to us or third party or both by traffic collisions
- Liability that could arise from incidents in a vehicle or accidents

As per law it is mandatory to have basic insurance for every automobile driver or vehicle, various polices come in different shape and sizes keeping in mind of individual basic to advance needs, covers individual policies requirement due to advancement in technology doing an insurance has become hassle free not only to company but also to customers.

Vehicle insurance may have added on terms to offer financial protection against theft of vehicle, natural disaster or damage to vehicles sustained because of events other than traffic collision such as keying, damage sustained by colliding with stationary objects. Maximum or minimum coverage depends on various factors such as age of car, brand and model number, risk or theft associated to that car.

Why need it, it acts as shield and protect from financial losses by paying for damage, it reduces liability, it is much affordable when purchased online, it compensates family after accident demise etc.

Business Problem Statement

Trying to automate the process of damage car claims which usually a process of person visiting at site following visual inspection and validation of damaged car using this automation we can claim insurance faster and hassle free manner which also helps the company in terms of their services ratings and attract large pool of customers with the help of technology.

Condition: predict of vehicle provided in image is damaged or not.

Amount: Based on condition of vehicle predict insurance amount of cars.

Data Set

Data Source : data
Train image folder: contains 1399 training images with multiple dimensions.
Test image folder: contains 600 test images with multiple dimensions.
Train csv: shape: 1399x8 data points.
Test csv: shape: 600x6 data points.
Sample submission csv: 5x3 data points.

Column	Description
Image path	Represents name of image
Insurance company	Represents masked values of insurance company
Cost of vehicle	Represents cost of vehicle present in image
Minimum coverage	Represents minimum coverage provided by insurance company
Expiry date	Represents expiry date of insurance
Maximum coverage	Represents maximum coverage provided by insurance company
Condition	Represents vehicle is damaged or not
Amount	Represent insurance amount of a vehicle

Business Constraints

1. Low-medium latency requirement.
2. Misclassification is problem as if car predicted not damage and insurance is provided is huge blunder.

ML Formulation of Business Problem

- Predicting the condition of car when deals with ml problem poses classification task which can be solved by classical, deep learning, ensemble or combination of these.

- Predicting the amount of insurance based on damage condition when deals with ml problem poses regression task which can be solved by classical, deep learning, ensemble or combination of these.

Performance Metric

1. For car condition prediction evaluation metric asked to use is micro-F1, although task is binary in nature false negative and false positive plays an important role, if false positive (actually not damaged but predicted damaged) it will be huge cost for company paying for claim for incidence which actually not happens, if false negative (actually damaged but predicted not damaged), it is still manageable, as it can further send for evaluation if required.

- Micro precision
  all the points model predicted positive in each class/label, how many of than are actually positive.

$$Precision_{micro} = \frac{\sum_{class} TruePositive_{class}}{\sum_{class} TruePositive_{class} + \sum_{class} FalsePositive_{class}}$$

Credit : micro precision

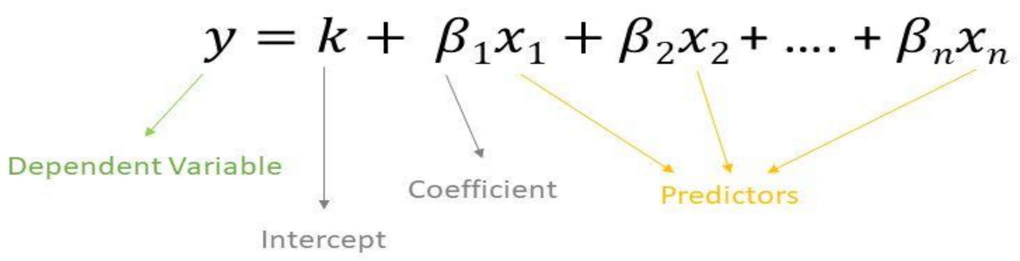
- Micro recall
  of all the label belongs to positive class; how many model detect to be positive class/label.

$$Recall_{micro} = \frac{\sum_{class} TruePositive_{class}}{\sum_{class} Truepositive_{class} + \sum_{class} FalseNegative_{class}}$$

Credit : micro recall

- Micro - f1
  harmonic mean of micro-pre., and micro recall

2. For claim amount prediction evaluation metric asked is r2_score also known as coefficient of determination. It basically gives the information about the goodness of fit of model i.e. the how well the predictor (independent variables) coefficient in regression equation approximates the test/real data points.
   Generally, the value to R^2 lies between 0 and 1, R^2 = 1 indicates regression coefficient perfectly fits the data but the value of R^2 can be negative which indicates the model fit on the data is absurd or worse than horizontal plane, that would be the case when wrong model was chosen.



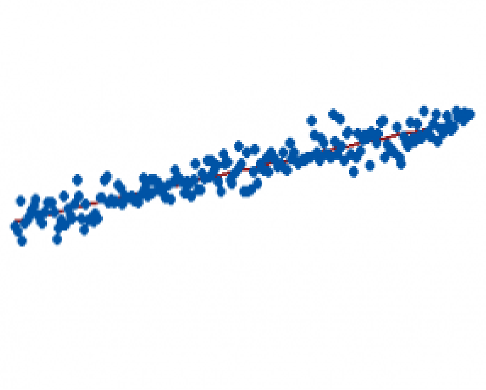
$$y = k + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n$$

Credit: multiple linear regression eq.

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

Credit: r_square

SS_{RES}: Sum Squared of residual
SS_{TOT} = Sum Squared of Total
Y_i = ith observation of actual data point
Yi_hat = ith observation of predicted data point
Y_bar = mean of actual data points



Major drawback of R2 is as the predictor increases models more complex, try to explain as much as nitty gritty of data this lead to over fitting, i.e. almost all the points lie on same line (imagine in the image left all the blue point strictly lies on red line), but this does not mean that all the predictors are responsibly contributing (explaining) to regression task (i.e. having predictive quality). To check the biasness, we have to check residual plot.

3. RMSE: It is root of mean difference of actual and predicted values, it penalizes large error.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Credit: rmse

4. Primary metric used for classification task is f1-micro, for regression task is r square as suggested in kaggle problem statement.

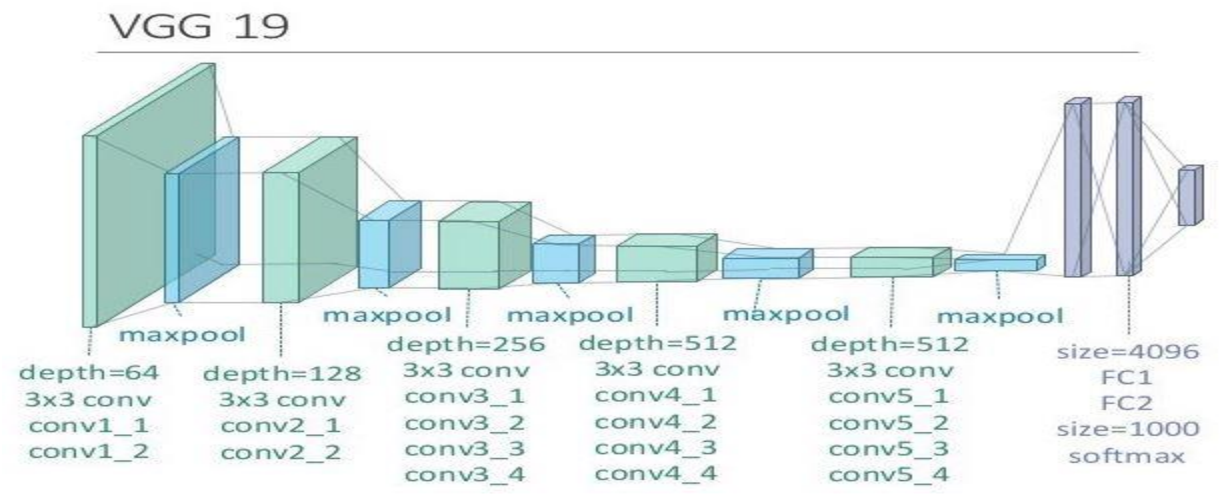
Research-Papers/Solutions/Architectures/Kernels

*** Mention the urls of existing research-papers/solutions/kernels on your problem statement and in your own words write a detailed summary for each one of them. If needed you can include images or explain with your own diagrams. ***

1. TDS blog on transfer learning by Park Chansung

The author had used vgg-19 architecture on cifar-10 dataset for transfer learning dataset. Transfer learning is a subfield of machine learning and artificial intelligence which aims to apply the knowledge gained (pertained parameter) from one task (source task) to different but with similar profile (target task).

Vgg-19 architecture has used base form of vgg-16 but only difference is that there is increase in depth from 3 to 4, from block 3 onwards in vgg-19, hence able to learn more depth features, these pertained architecture serves many task, transfer learning, segmentation, feature extraction (keeping core architecture constant except last few fc layers), object identification, neural transfer, fine tuning (keeping some of the layers in core architecture constant and train via back prop. Other layers) etc.



Credit : architecture

Some subtle details are:

- It has 19 layer (16 being convolution layer, 3 fc layers)
- It takes input of rgb image of shape (224,224,3)
- Only preprocessing done was to subtract mean RGB value from each pixel over whole training set
- Use kernel 3x3 with stride equals to 1
- Spatial padding was used to preserve the spatial resolution of image
- Max pooling is done via 2x2 kernel with stride equals to 2
- Used relu to introduce non linearity to make model classify better

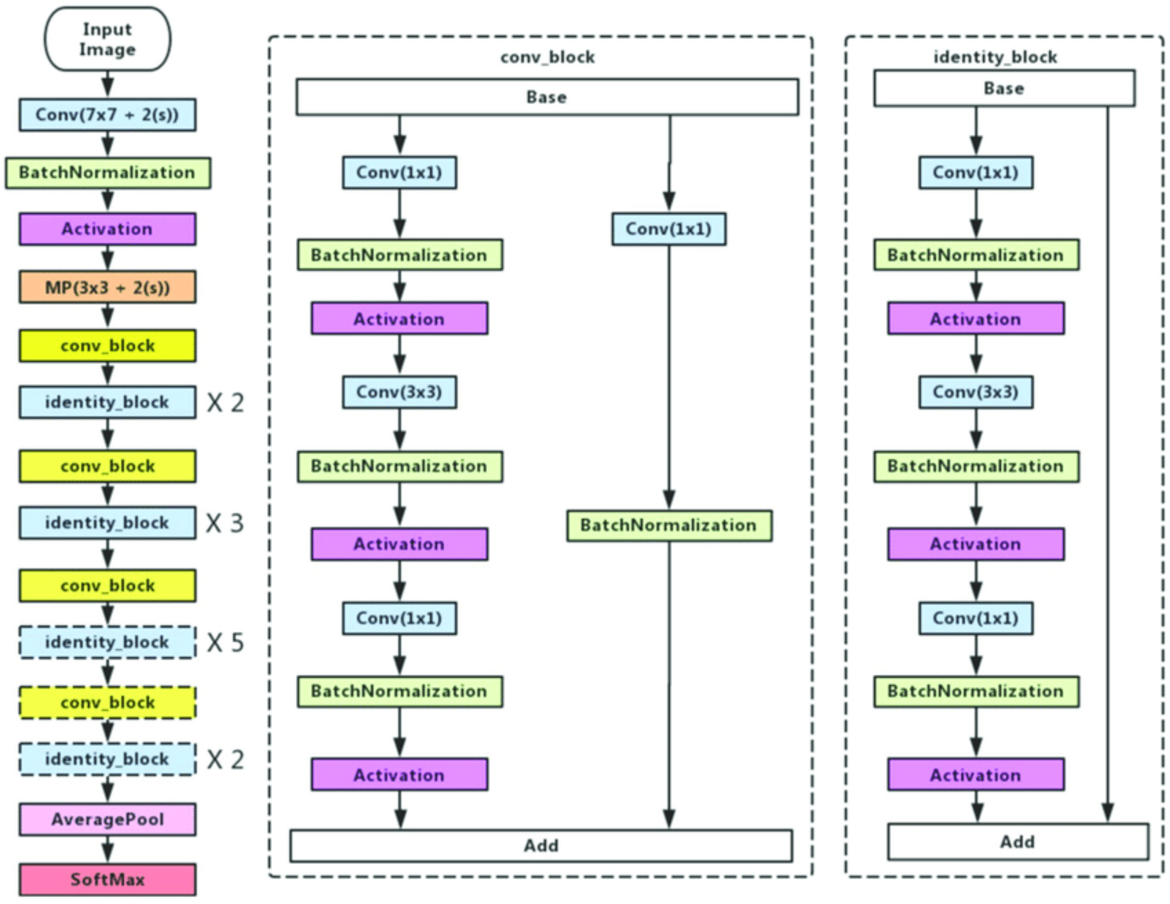
2. Kaggle kernel on data feature engineering

The author describes various features engineering techniques related to numerical features, date-time features, text based features, nlp based features, tf-idf based features, word embedding features, topic modelling features, for this case study I use 16 features out of 5 are listed below and rest 11 are described in first cut approach.

1. Year = extracting information about year from expiry date column
2. Month = extracting information about month from expiry date column
3. Month day = extracting information about day of month from expiry date column
4. Year day = extracting information about day of year from expiry date column
5. Week day = extracting information about day of week from expiry date column

3. TDS blog on resnet by Priya Dwivedi

Residual network (resnet) is act as backbone for modern day computer vision task, it gains attention after winning ImageNet challenge 2015, key concept is that it allows to train very deep network (150++ layers) without the problem of vanishing gradients, which is handled through multiple skip connections (allows shortcut path for gradient flow), this is done through identity function learned by model whose aim is perform not worse than initial layers. She experimented on hand image dataset having six classes.



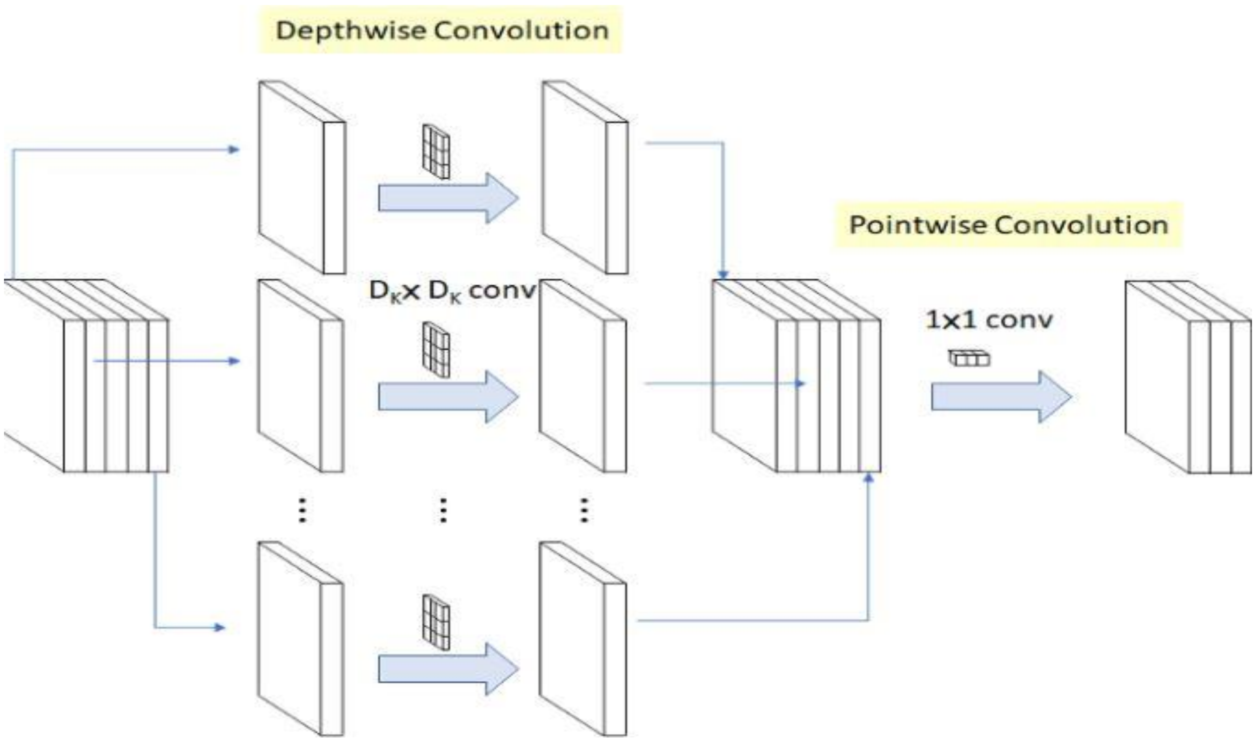
Credit : resnet50

Some subtle details are:

- Convolution with kernel size 7x7 with 64 kernel, with stride size 2x2 gives layer 1.
- Above followed by max-pool of size 3x3 with stride size 2x2.
- Next conv. (1x1(size), 64(kernel)), (3x3, 64), (1x1, 256) all three repeated 3(1(convolution block) + 2 (identity block)) times each giving total 9 layers.
- Next conv. (1x1(size), 128(kernel)), (3x3, 128), (1x1, 512) all three repeated 4(1(convolution block) + 3 (identity block)) times each giving total 12 layers.
- Next conv. (1x1(size), 256(kernel)), (3x3, 256), (1x1, 1024) all three repeated 6(1(convolution block) + 5 (identity block)) times each giving total 18 layers.
- Next conv. (1x1(size), 512(kernel)), (3x3, 512), (1x1, 2048) all three repeated 3(1(convolution block) + 2 (identity block)) times each giving total 9 layers.
- After this average pool with fc (1000 nodes) at end soft-max gives 1 layer.

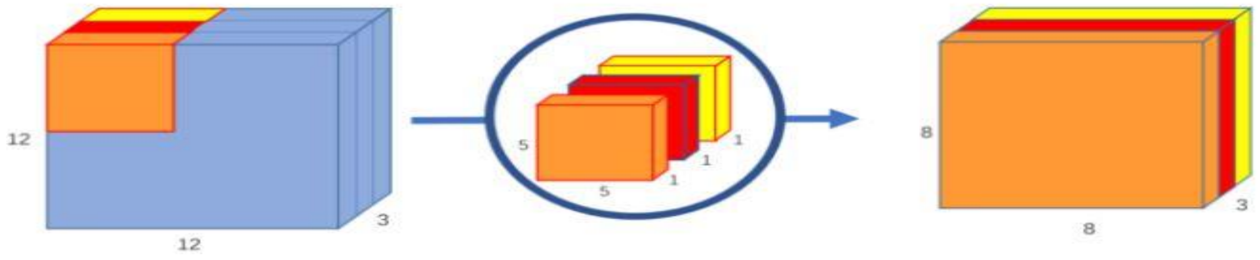
4. Medium blog on image classification mobile-net by Abhijeet Pujara

Mobile-Net is efficient and portable CNN architecture that is designed to be primarily used in low compute devices by reducing the number of parameters which can be done with the help of depth-wise convolution and point wise convolution.

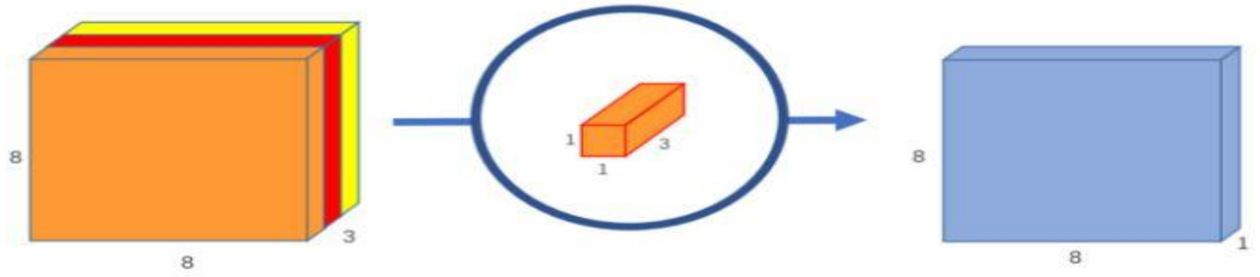


Credit : mobilenet

1. Depth-wise convolution: we give input image a convolution without changing depth i.e. if the image has n channel, so convolution filter also has n channels (filters), 1 input image channel corresponds to 1 filter each, responsible for filtering or extracting features.
2. Point-wise convolution: it is 1x1 kernel that iterates through every single point and as many as depth as we have channel or as many we want; from this we can increase the number of channels of each image, responsible for building new features through computing linear combination.



Credit: depthwise convolution



Credit: pointwise convolution

Some subtle details are:

- Total parameter in depth wise + pointwise convolution are:

  $D_K \times D_K \times M \times D_F \times D_F + M \times N \times D_F \times D_F$

  Where D_K = kernel size, M = input channel, D_F = feature map, N = output channel

  For our example: 5x5x3x12x12 + 3x1x8x8.

- Computation reduction is given by:

$$\frac{D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F}$$
$$= \frac{1}{N} + \frac{1}{D_K^2}$$

  If N is large that means most dominant factor is D_k which means roughly ($1/ D_k^2$) times in reduction in computation.

  For whole architecture see here : mobile net architecture

- Question arises can we have control over depth and width.

  Answer is yes, width multiplier (in keras it is alpha) and resolution multiplier (rho) (in keras it is depth multiplier), this reduces resolution (length and height) of image which have subsequent effect in every layer, computation cost factor reduces by rho^2. Alpha, rho E (0,1].

$$D_K \cdot D_K \cdot \alpha M \cdot D_F \cdot D_F + \alpha M \cdot \alpha N \cdot D_F \cdot D_F$$

$$D_K \cdot D_K \cdot \alpha M \cdot \rho D_F \cdot \rho D_F + \alpha M \cdot \alpha N \cdot \rho D_F \cdot \rho D_F$$

5. Analytics Vidhya blog on Gradient Boosting Regression

GBDT is an ensemble technique for regression and classification key point of this algorithm are it can adapt any loss function for optimization as long as it is differentiable, Weak learner is to make prediction; several weak learners (high bias and low variance) combines to give better prediction. We build model on features and make prediction, then we build second model on previous error (actual - prediction), how many models we build is hyper-parameter itself, therefore at every model we try to lower the error, at last we use weighing technique (gamma) to add all the model, this gamma is derived from algorithm steps itself.

$$F_{n+1}(X) = F_n(X) + \gamma_n \; H(x, e_n)$$

$$\begin{aligned}
F_0(X) &= \gamma_0 \; H_0(x, y) + e_0 \\
F_1(X) &= F_0(X) + \gamma_1 \; H_1(x, e_0) + e_1 \\
F_2(X) &= F_1(X) + \gamma_2 \; H_2(x, e_1) + e_2 \\
&\vdots \\
F_n(X) &= F_{n-1}(X) + \gamma_n \; H_n(x, e_{n-1}) + e_n
\end{aligned}$$

Credit : gbdt diagram

Some subtle details:

- Training set (x_i, y_i), differentiable loss function L (y_i, F(x)), M = no. iteration
  Initialize model constant value:

  $$F_0(x) = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(x_i, \gamma)$$

- For m=1 to M (base learners)
  - Compute pseudo residuals

$$r_{im} = -\left[\frac{\partial L(y_i, f_{m-1}(x_i))}{\partial F_{m-1}(x_i)}\right] \qquad i = 1 \ldots n$$

  - Fit base learners on pseudo residuals with new training set $D_{modified} = \{x_i , r_{im}\}$

$$F_m(x) = F_{m-1}(x) + \gamma h_m(x) \qquad \boxed{F_m(x) = F_0(x) + \gamma_1 h_1(x) + \gamma_2 h_2 \ldots}$$

Where F_{m-1} is previos learner, $h_m(x)$ predicted output from $D_{modified}$

  - Compute ɣ/ $ɣ_{optimum}$ from 2 for base learner

$$\gamma_{optim} = \underset{\gamma}{\arg\min} \sum_{1}^{n} L(y_i, F_m(x_i)) = \arg\min \sum_{i=1}^{n} L(y_i, f_{m-1}(x_i) + \gamma h_m(x_i)$$

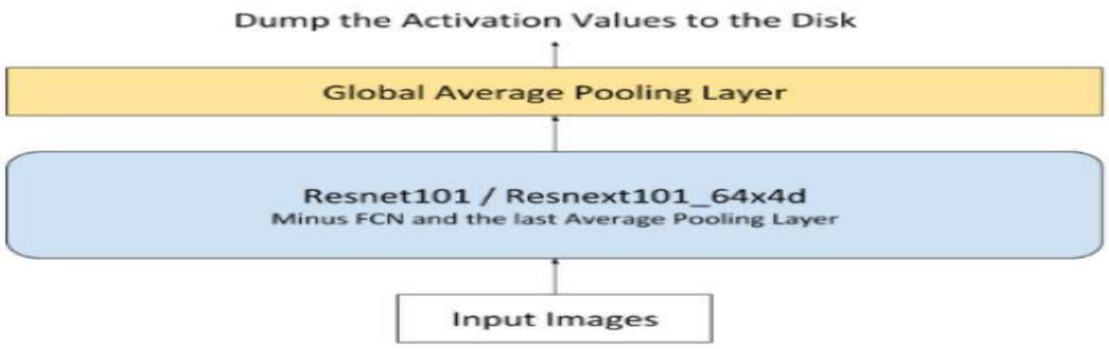
  - Final model update

$$F_m(x) = f_{m-1}(x) + \gamma_{optim.} h_m(x)$$

6. Medium Blog on avito demand prediction challenge by Ceshine Lee

He beautifully stacks the three model which is not out of box as compare to top rankers, first pre-trained language model, second image feature extraction, train regression model, main aim is to diversify the model so as to learn nitty gritty details.

- Image feature extraction

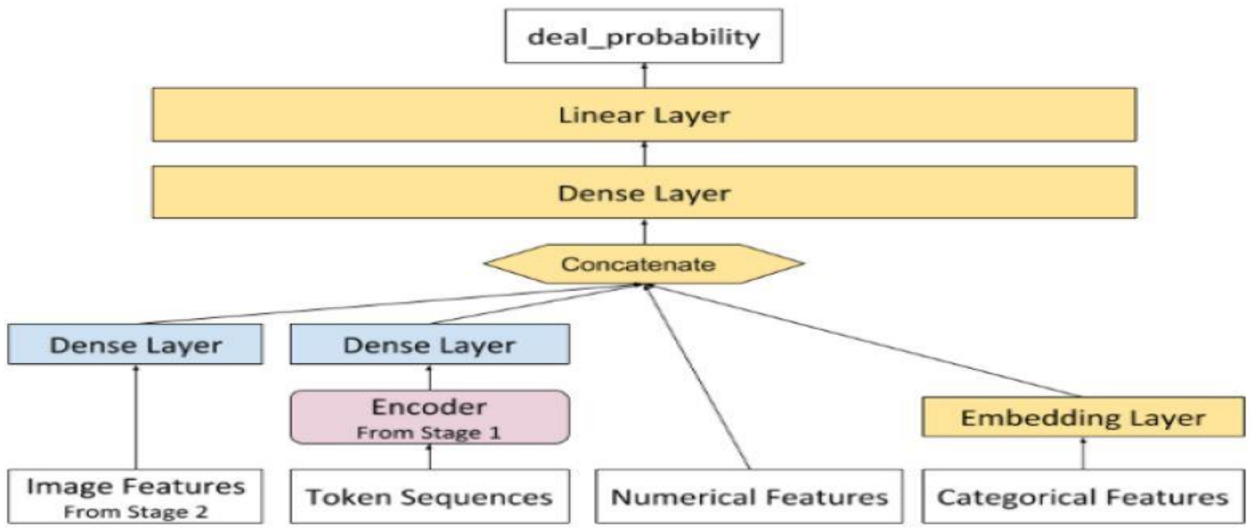


Dump the Activation Values to the Disk
Global Average Pooling Layer
Resnet101 / Resnext101_64x4d
Minus FCN and the last Average Pooling Layer
Input Images

He used Resnet101 and replace last fc and average pooling by global pooling so as to support variable sized image and also tried two image processing center cropping and square padding.

- Train regression model

Numerical features were normalized to zero mean and unit standard deviation. Categorical feature dimensions are mostly under 5 dimensions, used dense layer as down samplers which reduces output to 128 dimensions, feature extracted from image this numerical type is feed to GBM.



First Cut Approach

*** Explain in steps about how you want to approach this problem and the initial experiments that you want to do. ***(MINIMUM 200 words)*** ***

1. First perform EDA on a dataset to find patterns in the labels.
2. For categorical data, implement embedding layer.
3. Taking inspiration from point 1, I would like to implement resnet-50 algorithm for transfer learning.
4. Taking inspiration from point 2, other nine features are:
    1. Week number = extracting information about number of week from expiry date column.
    2. Luxury vehicles = all the vehicles which are greater than 75 percentile of cost of vehicle column.
    3. Medium segment vehicles = all the vehicles which are greater than 25 and less than 75 percentile of cost of vehicle column.
    4. Budget segment vehicles = all the vehicles which are less than 25 percentile of cost of vehicle column.
    5. Age of insurance = it is time period from today in years till time insurance expires
    6. Company count = count of number of times insurance company appears in dataset.
    7. Range of coverage = difference between minimum and maximum coverage.
    8. Insurance period = if age of insurance is greater than the median of age column than 1 else 0.
    9. Low expire = insurance expire within 2 years from now.
    10. Medium expire = insurance expire greater than two years but less than 5 years from now.
    11. High expire = insurance expire with more than 5 years from now.

4. Taking inspiration from point 3, I would like to implement vgg-19 algorithm for transfer learning.

5. Taking inspiration from point 4, I would like to implement mobile-net algorithm for transfer learning.
6. Taking inspiration from point 5, I would like to implement GBDT algorithm for regression task.
7. Taking inspiration from point 6, I would like to implement stacked model for task.
8. Will employ random search for better performance.

Models	Transfer learning	Stacking
1. vgg-19 + gbdt	no	no
2. resnet-50 + gbdt	no	no
3. mobile net + gbdt	no	no
4. custom architecture + gbdt	no	no
5. vgg-19 + gbdt	yes	yes
6. resnet-50 + gbdt	yes	yes
7. mobile net + gbdt	yes	yes
8. custom architecture + gbdt	yes	yes

Notes when you build your final notebook:

1. You should not train any model either it can be a ML model or DL model or Countvectorizer or even simple StandardScalar
2. You should not read train data files
3. The function1 takes only one argument "X" (a single data points i.e 1*d feature) and the inside the function you will preprocess data point similar to the process you did while you featurize your train data
    a. Ex: consider you are doing taxi demand prediction case study (problem definition: given a time and location predict the number of pickups that can happen)
    b. so in your final notebook, you need to pass only those two values
    c. def final(X):

preprocess data i.e data cleaning, filling missing values etc

compute features based on this X

use pre trained model

return predicted outputs

final([time, location])

   d. in the instructions, we have mentioned two functions one with original values and one without it

   e. final([time, location])   # in this function you need to return the predictions, no need to compute the metric

   f. final(set of [time, location] values, corresponding Y values) # when you pass the Y values, we can compute the error metric(Y, y_predict)

4. After you have preprocessed the data point you will featurize it, with the help of trained vectorizers or methods you have followed for your train data

5. Assume this function is like you are productionizing the best model you have built, you need to measure the time for predicting and report the time. Make sure you keep the time as low as possible

6. Check this live session: https://www.appliedaicourse.com/lecture/11/applied-machine-learning-online-course/4148/hands-on-live-session-deploy-an-ml-model-using-apis-on-aws/5/module-5-feature-engineering-productionization-and-deployment-of-ml-models