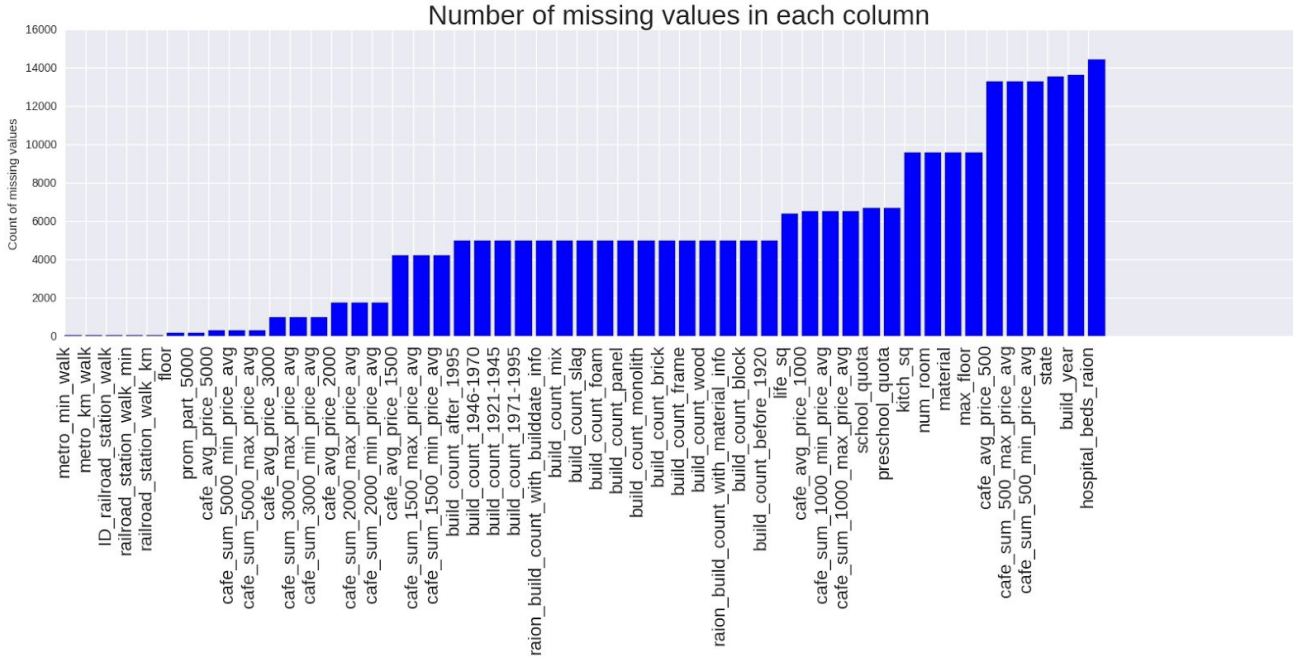
ML2017 Final Report

題目：SberBank Russian Housing Market
隊伍名稱: NTU_b03b02014_TreeTree武器走
隊伍成員: B03902043 邱綜樹、B03902046 周侑廷、B03b02014 張皓鈞
分工:

	主要工作	次要工作
邱綜樹	XGBoost Ensemble	Feature Engineering NNet, KNN, Random Forest
周侑廷	XGBoost Ensemble	Feature Engineering Random Forest (Bagging)
張皓鈞	NNet Ensemble	XGBoost Exploratory Analysis

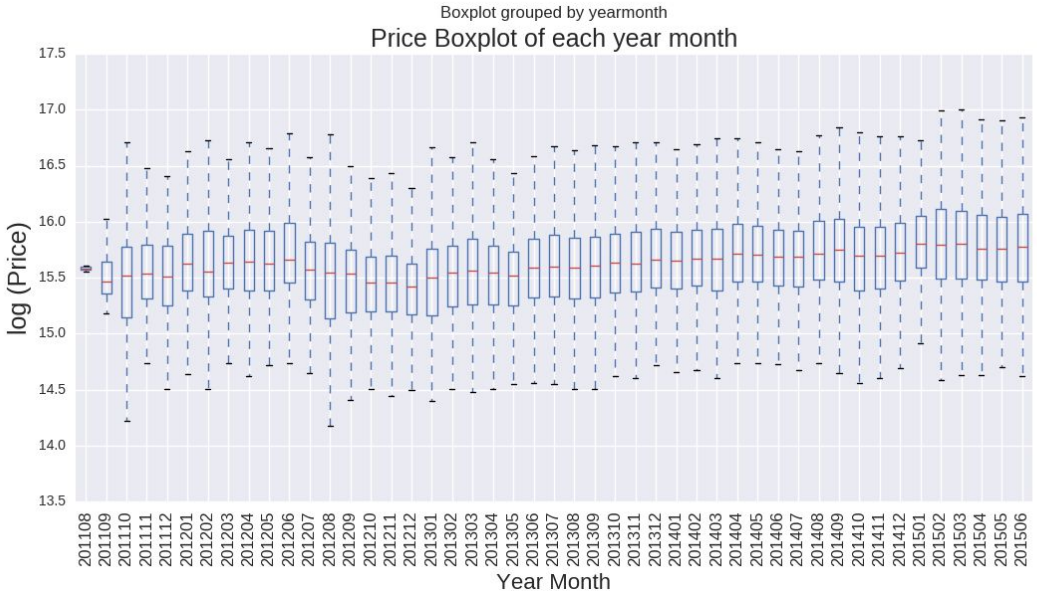
Exploratory Analysis

1. 缺少的特徵值：



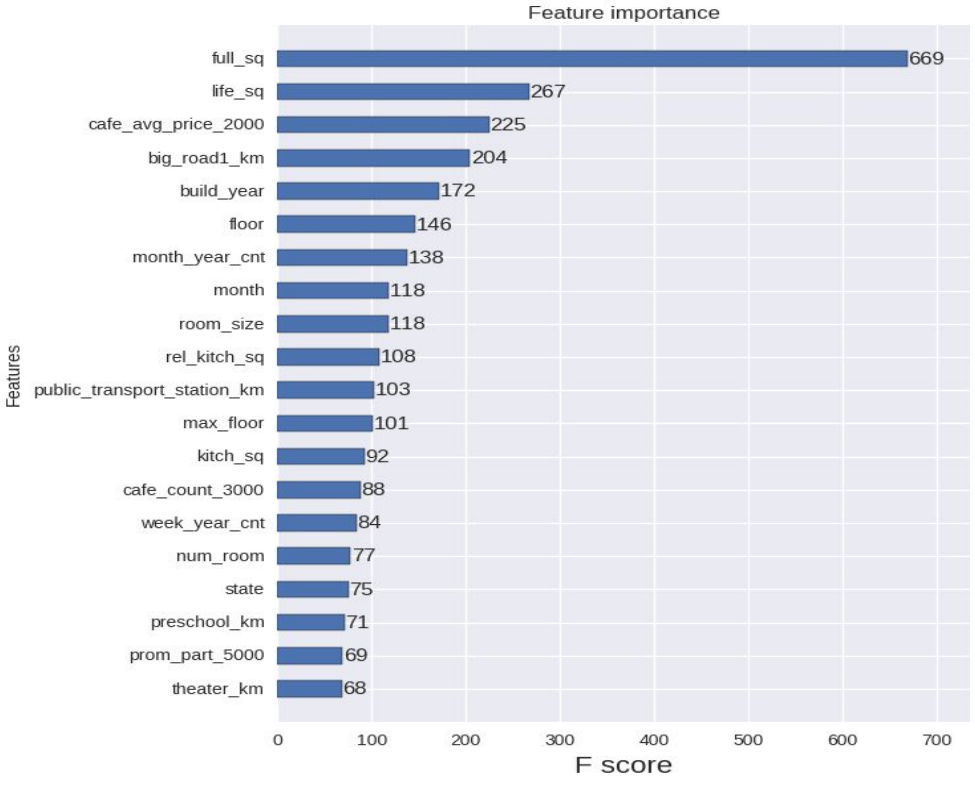
可以看到有不少特徵是缺失的，在3萬多筆的訓練資料中，有6個特徵高達有16000多筆資料是缺失的，前幾名包含該區域的醫院床數、建造年份、房屋狀態和附近500公尺咖啡店的最大最小價錢平均等等。由缺失的數量來看，我們判斷這樣的現象會嚴重影響之後的模型訓練，因此在選擇模型的時候必須考慮進去此因素。

2. 時間與價錢的關係：

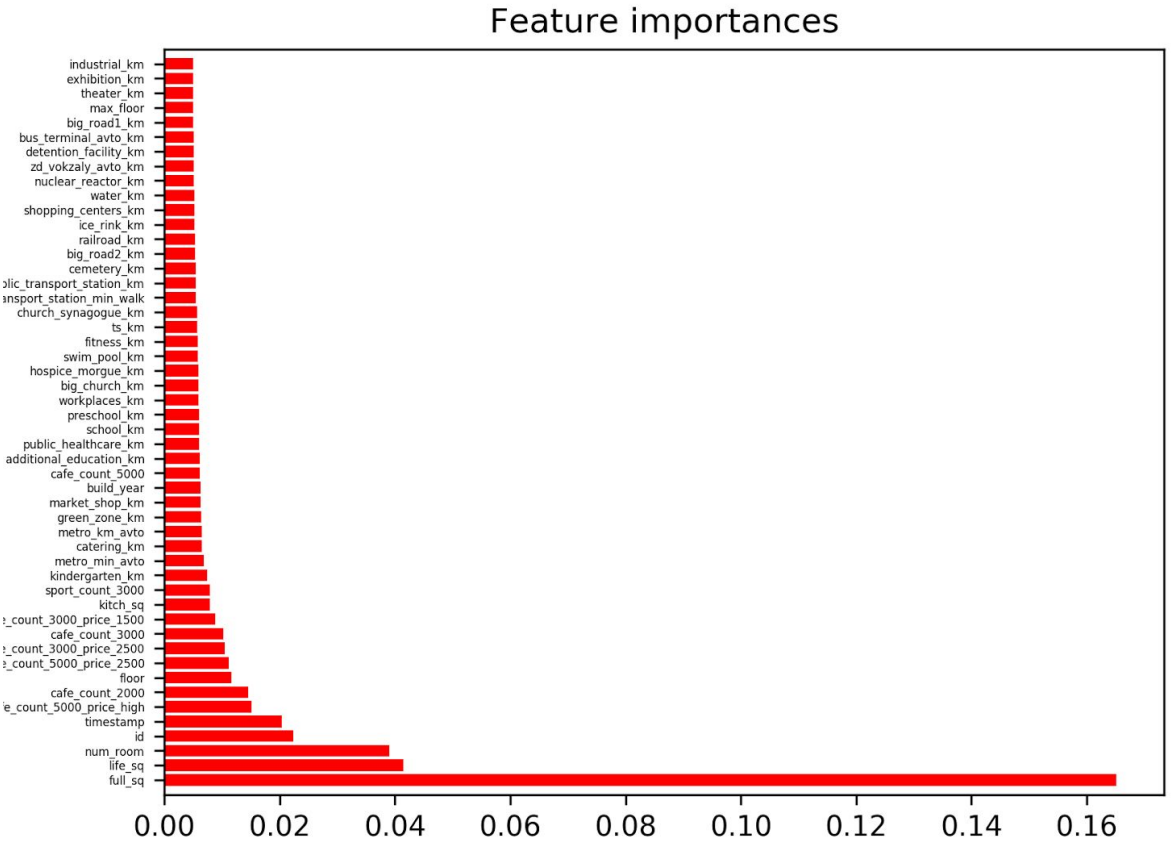


每個月的價錢取log之後的分佈如上圖，可以看到分佈的range大多是從14.0~17.0左右，轉換回來的價錢大約是1200000~24000000，範圍十分廣泛，這也可能會是造成模型在訓練的難度。因此我們考慮在訓練模型時將價錢進行log轉換，讓數值的差距變小，讓訓練模型的時候比較容易。

3. RandomForest和XGBoost的feature importance比較
XGBoost



Random Forest



兩種model最重視的特徵皆為full_sq，也就是房屋的總坪數，其次就是life_sq。不太一樣的地方是XGB認為建造年份在前幾名重要，但是Random Forest卻認為沒有太重要。

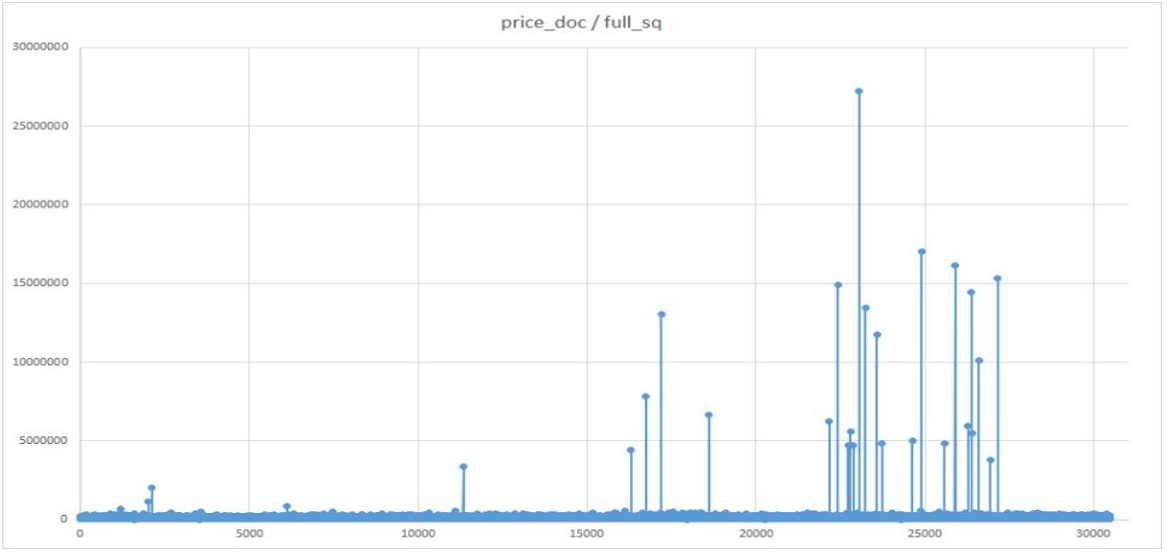
Preprocessing/Feature Engineering

由於其他參賽者發現Tverskoe地區的位址非常不合理，因此主辦單位有釋出修正版的訓練資料，所以我們首先將bad address修改成官方公佈的修正版。

另外，我們所訓練的兩個模型：eXtreme Gradient Boosting 以及Deep Neural Network各經過不太相同的資料前處理和特徵工程，因此以下分別作說明和討論：

XGBoost:

類別型特徵	使用sklearn.LabelEncoder將其轉換成0~類別數-1的數值。
NA值	由於 xgboost 套件本身支援對遺失值的處理，因此我們不需要處理資料中的NA。
價錢(預測目標)	轉成log1p，如此在訓練過程中的RMSE即為RMSLE。

資料清理	將outlier和不合理的特徵用NaN取代。 將單位面積價格過高的 training example 刪除。
	
加入衍生特徵	month_year_cnt: month + 100*year week_year_cnt: The week ordinal of the year + 100*year month: January=1, December=12 day_of_week: Monday=0, Sunday=6 rel_floor = floor / max_floor rel_kitch_sq = kitch_seq / full_sq room_size = life_sq / num_room apartment_name = sub_area + metro_km_avto
加入總體經濟資訊	由於經濟景氣對房屋價格的影響十分重要，我們將房價成長指數加入考量，新增一個 feature : average_q_price。將 2016 第二季設為 1，利用成長率的倒數填出各季的平均指數。

Deep Neural Network:
以下是修正後，對訓練資料和測試資料特徵的資料前處理：

類別型特徵	使用sklearn.LabelEncoder將其轉換成0~類別數-1的數值。
連續性特徵	分別對訓練資料和測試資料的連續性特徵進行標準化。
NA值	補中位數。
價錢(預測目標)	轉成log1p，如此在訓練過程中的RMSE即為RMSLE。
資料清理	同XGBoost
加入衍生特徵	同XGBoost

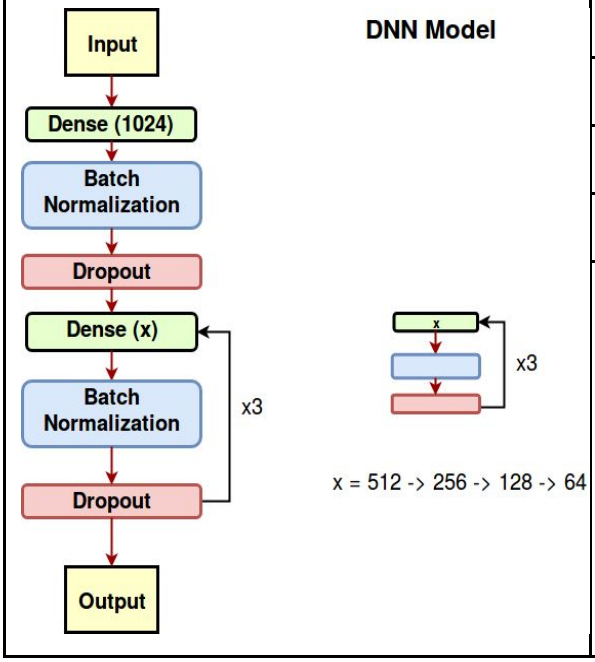
(最終訓練資料數量, 特徵數)： (30396, 299)

Model Description

1. XGBoost

我們使用的 xgboost 是一個經過優化的Gradient Boosting Library，能支援三種 base learner，並且實做 multi-thread 加快運算。另一個優點是，xgboost 能接受有 NaN 的 training data，讓我們不需要困擾該使用哪一種 cleaning method。我們使用的 objective function 是 reg:linear，並手動調整其他參數，如 eta, max_depth, subsample, colsample_bytree...。

2. NNet (Deep Neural Network)

模型架構	Activation	LeakyReLU
	Dropout Rate	0.4
	Loss function	Mean Squared Error
	Optimizer	Adam
	Validation	最後 5396 筆訓練資料
	Early stopping	Val_rmse 停止進步 連續20個epoch

3. Ensemble

XGBoost blending

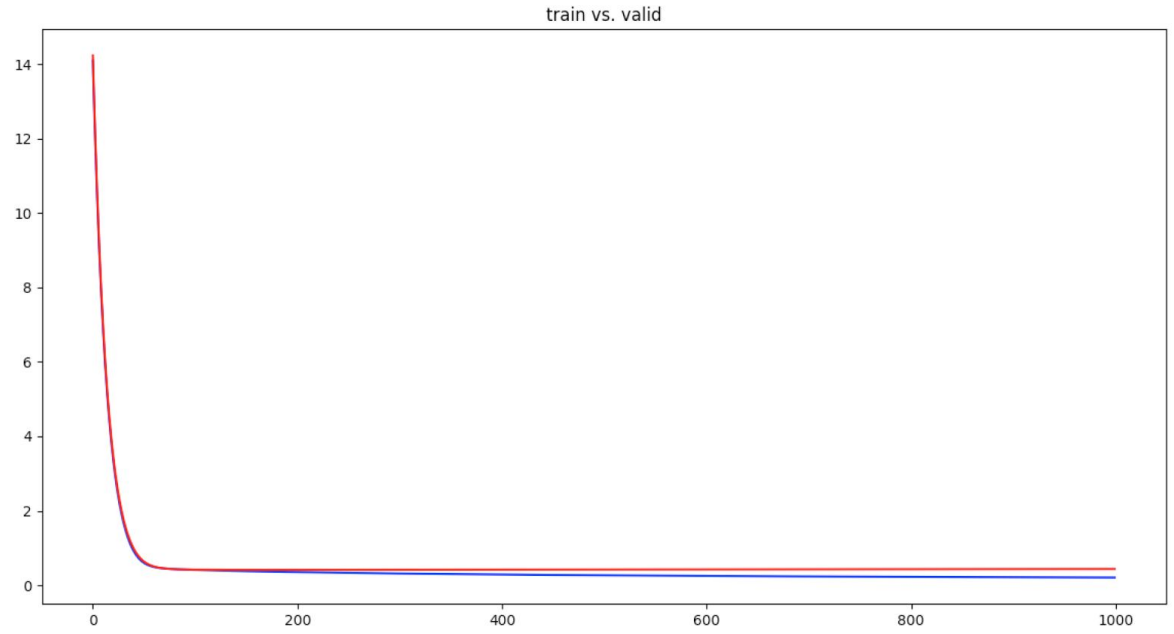
我們主要將3個XGBoost的模型預測結果依照不同的比例混合在一起，3個模型分別進行前述但不完全相同的前處理和特徵工程，使用的參數也不完全相同。

Experiments and Discussion

XGBoost (Validation last 5396)

I. iteration 數與 training / validation score的關係

- Iteration = 1~1000



II. 刪去極端資料、去除重複性高的資料

刪去極端資料	Yes	No
Training RMSLE	0.3927	0.4770
Validation RMSLE	0.4457	0.4008

III. 不同random seed的XGB作ensemble

ensemble	Yes	No
Training RMSLE	0.3576	0.4770
Validation RMSLE	0.4199	0.4008

討論：

由實驗數據我們可以發現，這份資料的training data有一些特殊的現象，就是training error 常常比 validation error高。我們認為這可能是因為資料本身品質不夠好，讓演算法很難學到pattern。因此在完成 cleaning 後，或是使用 random forest ensemble後，就能讓 data 變成品質較好的 data，讓演算法能夠學出規律，使training error小於等於validation error。

IV. 考慮時間序列 sliding window(每個window取100個transaction)
新特徵 = 第1~第100筆交易的特徵 + 第1~第99筆交易的價格
新預測目標 = 第100筆交易的價格

Sliding window	Yes	No
Training RMSLE	0.4698	0.42824
Validation RMSLE	0.4192	0.40617

討論：
　在加了time slice之後，training的loss變差了許多，可能是因為訓練資料在事實上是不完美的，一個個window滑過去後會把錯誤放大。然而validation的資料表現卻只差一點點，可以知道該部分資料是較好（較無outliner）的。

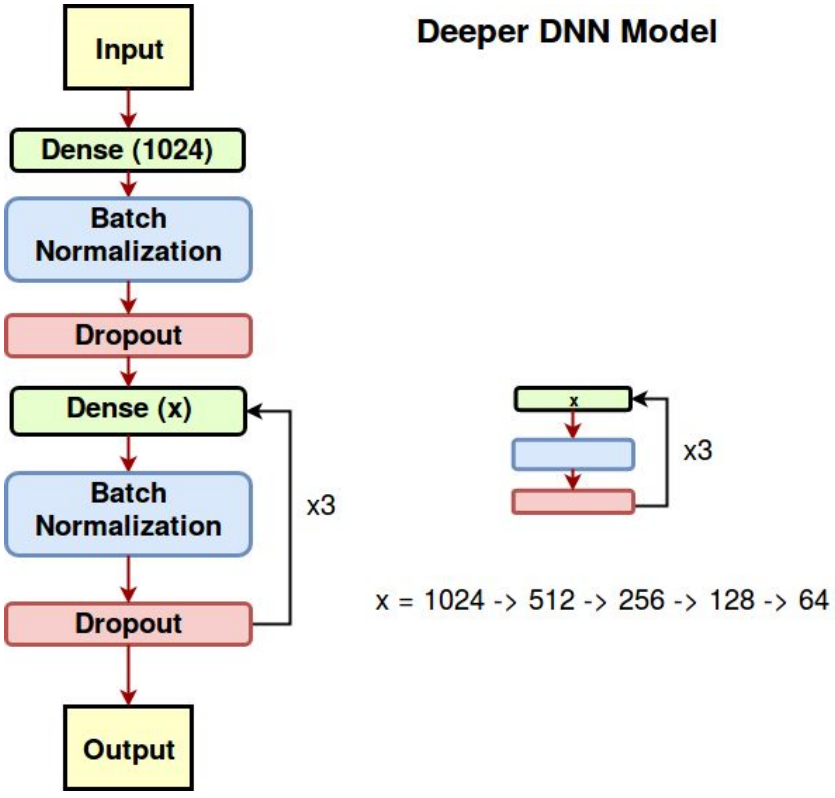
Neural Network
I. Batch Normalization的效果
　使用上述NNet的模型架構，比較有無Batch Normalization對預測表現的影響，其中RMSLE的計算是模型訓練結束後，另外計算全部訓練資料和驗證資料的RMSLE。

訓練結果：

Batch Normalization	Yes	No
Training RMSLE	0.47327	0.61672
Validation RMSLE	0.47394	0.65427
Public LB	0.41167	X

討論：
　Batch Normalization可以讓預測效果變好。原因推測是因為各特徵的數值範圍都非常廣泛，因此即使訓練前有進行標準化，每個Batch再經過一次標準化會使模型訓練表現更好。

II. Deeper Model
　由於上述模型的表現差強人意，因此我們決定加深模型，測試能否增加表現，下圖是加深後的DNN模型架構圖：



訓練結果：

	#1	#2
Deeper Model	[Dense(i) -> BN -> Dropout] x 3	[Dense(i) -> BN -> Dropout] x 4
Training RMSLE	0.46510	0.47103
Validation RMSLE	0.45269	0.47407
Public LB	0.39185	X

討論：

加深模型架構雖然會有一定的進步，但是愈深不代表一定會變好，即使利用了LeakyReLU作為activation，仍然會有gradient vanishing的風險。

III. 類別型特徵使用one-hot encoding

由於LabelEncoder會將類別型特徵任意指定一個數字作為類別的編號，因此在數值上並不能完全代表類別之間的關係距離，所以我們改使用one-hot encoding的前處理，將類別之間的關係去除，且DNN架構上使用較深的模型(II.Deeper Model #1)來進行比較。

訓練結果：

Categorical	LabelEncoder	One-hot encoding
Training RMSLE	0.46510	0.45953
Validation RMSLE	0.45269	0.45282
Public LB	0.39185	0.38736

討論：

雖然one-hot encoding會將feature space的維度從299維提高到458維，但是由於類別之間的關聯性變成互相獨立，因此在預測表現上比之前較好一些。

IV. 不同補NA值的方法

前述提到此訓練資料有許多missing value，以及許多不合理或是極端資料被前處理成NA值，因此NA值的處理方式也會影響DNN模型的訓練，在此討論的DNN模型為II.Deeper Model #1的架構，類別型特徵使用one-hot encoding。

訓練結果：

Method	Median	Mode	Mean
Training RMSLE	0.45953	0.60527	0.45745
Validation RMSLE	0.45282	0.63004	0.45992
Public LB	0.38736	X	X

討論：

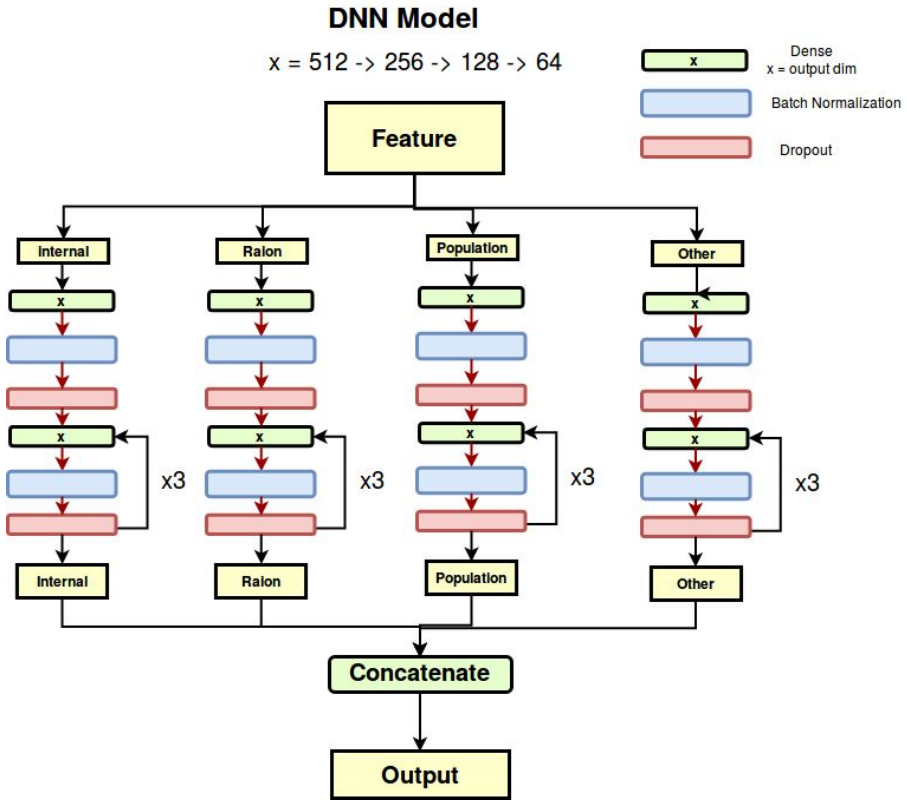
以Validation RMSLE來比較的話，NA值補中位數的表現比其他兩種較好，雖然補上平均的模型在training RMSLE上有比較低的error，但是因為validation沒有明顯的進步，因此不討論Public LB上的分數。

V. 將特徵降維：

由於加上衍生特徵之後，訓練資料的特徵維度高達299維，在特徵空間中可能算是稀疏的分佈，因此我們嘗試數種降維的方式讓特徵維度下降，並比較各種方式的效果。

1. 將特徵分成以下幾群，分別通過DNN，最後concat起來：

在此所討論的模型，我們使用最初的模型架構(see Model Description)，而NA值補median以及類別型轉換為one-hot encoding：



分群方式(1為id，2為timestamp)：

房屋內部(internal)： 3~12 + rel_floor + rel_kitch_sq + room_size

所在raion特徵(raion)：13~41

人口相關(population)： 42~68

房屋附近設施距離與其他(other)： 69~297

訓練結果：

Training Method	All features	Split and Concat
Training RMSLE	0.47327	0.49878
Validation RMSLE	0.47394	0.51963
Public LB	0.41167	0.45600

討論：

根據特徵的性質分成數群並分別訓練的表現比全部一起訓練還要來得差。

原因推測為DNN的參數需要全面性的特徵資訊才能做比較有效的訓練，人工分成數群的結果使各個DNN參數分別被訓練的較弱，因此即使集合在一起預測，表現還是不及全部特徵一起訓練的模型。

2. PCA降維

在此所討論的模型，我們使用的模型架構為II. Deeper Model#1，NA值補median以及類別型轉換為one-hot encoding：

訓練結果：

PCA components	250	200	150	100	50
Training RMSLE	0.46750	0.38164	0.44675	0.43840	0.47417
Validation RMSLE	0.47434	0.44328	0.46392	0.43866	0.47329
Public LB	X	0.39198	X	X	X

討論：

利用PCA降維確實可以些微提升預測表現，其中100~200維之間的validation RMSLE皆有些微下降，但是到了50維的表現又下降，顯示100維大概是能表現特徵資訊的最小維度。